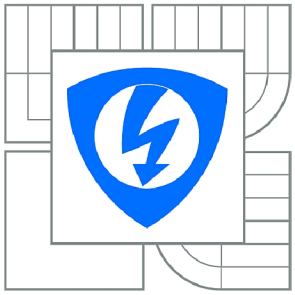
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

PRŮMYSLOVÝ PROGRAMÁTOR MIKROKONTROLÉRŮ AVR ATMEL

INDUSTRIAL PROGRAMMER ATMEL AVR MICROCONTROLLERS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

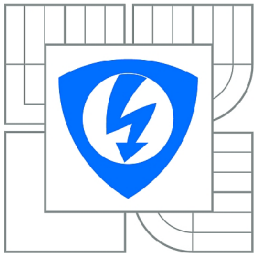
AUTOR PRÁCE — Bc. JAN GRYŽBOŇ

AUTHOR

VEDOUCÍ PRÁCE — Ing. ONDŘEJ PAVELKA

SUPERVISOR

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Jan Gryžboň	***ID:***	120773
Ročník:	2	***Akademický rok:***	2013/2014

NÁZEV TÉMATU:

Průmyslový programátor mikrokontrolérů AVR Atmel

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte a realizujte průmyslový programátor mikrokontrolérů AVR Atmel buď pomocí ISP nebo JTAG rozhraní. Proveďte porovnání s komerčními programátory, proveďte analýzu rychlosti pro ISP a JTAG rozhraní. Kritérium je zejména rychlost programování a schopnost změny částí kódu – např sériové číslo. Vytvořte k tomuto programátoru i vhodnou knihovnu pro spolupráci s C++ a C#.

DOPORUČENÁ LITERATURA:

[1] Kainka, B. :Elektronika s podporou PC, BEN, 2004. ISBN 80-86167-22-4
[2] AVR: In-System Programming - Atmel Corporation [online]. Rev. 0943E–AVR–08/08 [cit. 2013-09-20]. Dostupné z: www.atmel.com/images/doc0943.pdf

Termín zadání:	10.2.2014	***Termín odevzdání:***	28.5.2014

Vedoucí práce: Ing. Ondřej Pavelka
Konzultanti diplomové práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Cílem této diplomové práce je navrhnout a realizovat průmyslový programátor mikrokontrolérů AVR od firmy Atmel. V první části jsou uvedeny teoretické poznatky o AVR mikrokontrolérech. Dále jsou porovnány metody programování mikrokontroléru a pro vybrané metody je provedena analýza rychlostí programování. Následuje samotný návrh průmyslového programátoru, který je proveden v programovém prostředí Eagle. V předposlední části je návrh konstruován na desku plošných spojů. Závěrečná část je věnována oživení programátoru a jeho naprogramování.

Klíčová slova

AVR, Atmel, programátor, metody programování, ISP, JTAG, PWM, Eagle

Abstract

The aim of this thesis is to design and implement an industrial programmer AVR microcontrollers from Atmel. The first section provides the theoretical knowledge of the AVR microcontroller. Furthermore, comparison of methods for programming the microcontroller and the selected method is an analysis of the speed of programming. The following is the entire design of industrial programmer which is implemented in the programming environment Eagle. In the penultimate part of the proposal is designed PCB. The final section is devoted to the revival of the programmer and programming.

Key words

Atmel, AVR, programmer, programming methods, ISP, JTAG, PWM, Eagle

Bibliografická citace

GRYŽBOŇ, J. *Průmyslový programátor mikrokontrolérů AVR Atmel.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2014. 96 s. Vedoucí diplomové práce Ing. Ondřej Pavelka.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma „Průmyslový programátor mikrokontrolérů AVR Atmel“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení §11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení §152 trestního zákona č. 140/1961 Sb.

V Brně dne …… ……

(podpis autora)

Poděkování

Děkuji vedoucímu diplomové práce Ing. Ondřeji Pavelkovi. Za velmi užitečnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé práce.

V Brně dne …….. ……..

(podpis autora)

Výzkum popsaný v této diplomové práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

(vložit originál list)

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

V dnešní době řada velkých průmyslových podniků, které se zabývají mikrokontroléry firmy Atmel, řeší problematiku programátoru AVR mikroprocesorů. Firmě Honeywell, s jejichž spolupráci řeším tuto diplomovou práci, se jedná zejména o rychlost programování firmwaru do nového mikroprocesoru (přímo z výroby, bez předchozího zásahu). Další důležitou vlastností průmyslového programátoru má být modifikace kódu (Firmware + speciální identifikační údaje). Firmware bude uložen na SD kartě programátoru a další důležité údaje, jakými jsou sériové číslo a číslo kalibrace budou přeneseny unikátně pro každý mikrokontrolér pomocí USB sběrnice přes průmyslový programátor.

V úvodu diplomové práce popisuji samotnou architekturu mikroprocesorů AVR. Upřesním důležitost různých pamětí mikrokontroléru rodiny Atmel. V další kapitole provedu přehled sériových i paralelních programovacích metod. Z nichž vyberu dvě pro následnou analýzu. Třetí kapitola je věnována samotné analýze rychlosti programování pro mnou zvolené metody programování. Analýza byla prováděna v programu AVR Studio 4.19 a práce byly prováděny v laboratoři firmy Honeywell.

Praktická část se zabývá rozborem řešení návrhu průmyslového programátoru AVR. V kapitole 5 je pro lepší pochopení problémů sestaveno blokové schéma, které popisuje všechny funkce programátoru. Následuje kapitola s výběrem vhodných komponent a teoretickým doplněním informací ohledně sběrnice USB, SD karty a PWM signálu. V sedmé kapitole je navržené schéma převedeno k výrobnímu procesu, návrh schématu je vytvořen v softwarovém prostředí EAGLE. Následuje popis způsobu osazení DPS a její oživení. Následuje vytvoření softwarové části pro správnou funkci programátoru. Pro lepší pochopení základní funkce je vytvořen blokový diagram, ze kterého jsou patrnější samotné funkce. Všechny výkresy a zdrojové kódy jsou na přiloženém DVD.

1. TEORETICKÝ ÚVOD

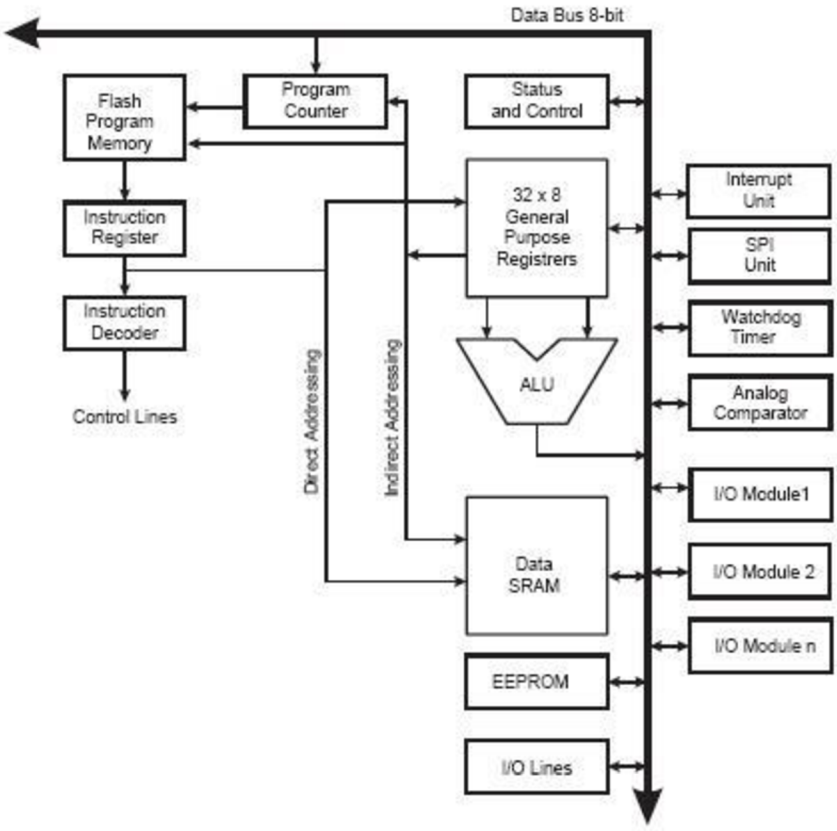
Na samém začátku diplomové práce popíšu základní RISC architekturu mikroprocesorů, paměti AVR a celkové využití mikroprocesorů AVR od firmy Atmel.

1.1 AVR Atmel

Jádro mikroprocesoru AVR tvoří architektura RISC a jedná se o výrobky společnosti Atmel. Skládá se z 32 stejných 8mi bitových registrů, které mohou obsahovat jak data, tak i adresy. Vzhledem k propojení registrů s aritmeticko-logickou jednotkou ALU provede ALU za jeden hodinový cyklus jednu operaci. Mikroprocesory AVR využívají koncepci Harwardské architektury. To znamená, že paměti programu a dat jsou odděleny.

Mikrokontroléry AVR Atmel umožňují možnost programování jak sériově přímo v systému, tak i paralelní metodou. Při programování paralelní metodou se využívá specifického návrhu obvodu, kdy po připojení programovacího napětí na určitý vývod mikrokontroléru se provede přepnutí vývodů z normálního režimu I/O portů na adresové a datové vývody vnitřní paměti. Poté je možné zaznamenat data do paralelní paměti. Po dokončení programování se obvod opět přepne zpět do svého normálního režimu. Nevýhodou při tomto paralelním programování je nutnost odpojit mikroprocesor od jakýchkoliv ostatních obvodů a umístit ho do samotného programátoru. Tato nevýhoda je při sériovém programování výhodou. Samotné programování mikroprocesoru probíhá přímo na výrobní desce a pomocí několika signálů připojených na programátor ho lze jednoduše naprogramovat. Při programování metodou ISP se využívá převážně signálů MOSI, MISO, SCK a RESET[13], [21], [27].

Firma Atmel nabízela mikroprocesory AVR ve třech řadách. První řada byla tzv. základní a dnes již nepoužívaná. Další dvě řady jsou ATtiny a ATmega, které se liší především počtem instrukcí. Dnes již většina mikrokontrolérů z řady ATmega obsahuje JTAG rozhraní, které je využito k ladění softwaru uvnitř aplikace. Dále se mikroprocesory liší použitými obvody, které jsou použity uvnitř mikroprocesoru. Například velikost pamětí SDRAM, pamětí programu FLASH a pamětí dat EEPROM. Dále počtem portů, počtem časovačů/čítačů a jejich rozlišením (8 nebo 16bit). Většina mikroprocesorů obsahuje UART, řady ATmega jich můžou mít i více. Ve většině případů dva. Obsahují také analogový komparátor, obvod Watchdog a některé i A/D převodník a spoustu dalších. Podrobnější informace ke konkrétnímu typu mikroprocesoru nalezneme ve výrobních listech od výrobce [8].



Obr. 1.1: Obecná architektura AVR (převzato z [24])

1.1.1 Paměti AVR

Všechny paměti, které obsahuje AVR Atmel jsou umístěny uvnitř samotného čipu. Flashová paměť programu, dále pak trvalá paměť dat EEEPROM a paměť dat SRAM.

Paměť programu - Flash

Instrukce programu jsou uloženy v paměti Flash (10000x přepisovatelné), uchovávající obsah i po vypnutí napájení. Velikost Flash bývá uvedena v označení součástky (např. řada ATmega64x má 64 kB Flash). Nelze použít vnější paměť pro program, veškerý kód prováděný jádrem AVR musí být uvnitř Flash [13].

Paměť dat EEPROM

Téměř všechny AVR mikrokontroléry mají interní, elektricky mazatelnou, programovatelnou paměť (EEPROM). Stejně jako Flash i EEPROM uchovává svůj obsah i po vypnutí napájení (garantovaný počet přepsání je 100000x). EEPROM se nejčastěji používá k uložení konfiguračních dat, díky kterým se může jednotka po nechtěném restartu automaticky nastavit do původního režimu [13],[27],[30].

Datová paměť SRAM

Vnitřní paměť dat SRAM je označována jako nestálá nonvolatile, což znamená v případě výpadku napájení je obsah paměti ztracen. Procesy jakými jsou zápis a čtení jsou realizovány pouze v průběhu vykonávání programu. Do paměti dat se ukládají globální proměnné a alokuje se zde prostor pro vnitřní proměnné. V případě nedostatku místa lze připojit externí paměť s maximální velikostí 64 kB [13],[27],[30].

1.1.2 Signatura

Součástí mikrokontroléru jsou 3 byty signatury. Tyto byty identifikují výrobce a typ mikrokontroléru. Tyto byty jsou uloženy na speciálním adresném prostoru mimo paměť FLASH a EEPROM. Výrobce ATMEL identifikuje první byt a hodnotě 0x1E, druhým bytem se identifikuje vlastní velikost FLASH paměti a třetí byt nám udává přesný typ obvodu.

1.1.3 Propojky

Programovací propojky, z anglického fuse bits, slouží v mikrokontroléru k nastavení jeho zdroje hodinového signál a jiných důležitých vlastností. U nastavení těchto bitů musíme dat velký pozor na správnost jeho nastavení. V případě špatného zásahu do nastavení propojek, můžeme mikrokontrolér přivést do stavu, kdy bude úplně nepoužitelný. Z takového stavu se lze dostat pouze přeprogramováním mikrokontroléru metodou vysokých napětí.

Popis vše nastavení propojek a jejich použití je vždy uvedeno v katalogovém listu daného mikrokontroléru. Hodnota propojek se nezmění příkazem Erase chip. Například propojku SPIEN, nelze programovat sériovým programováním. Hodnota propojek může nabývat buď hodnot logické 0 (naprogramována), nebo logické 1 (nenaprogramována). Hodnota propojek je uvedena jako číslo ve tvaru dekadickém, binární či hexadecimálním.

1.2 RISC architektura

Architektura počítačů RISC označuje v informační technice procesory s redukovanou instrukční sadou. Návrh těchto mikroprocesorů je zaměřen hlavně na jednoduchou, ale vysoce optimalizovanou sadu strojových instrukcí. Při porovnání s ostatními architekturami, které nabízejí celou řadu specializovaných instrukcí, je pravým opakem. Mezi hlavní rysy RISC architektury patří: komunikace procesoru s pamětí po sběrnici, délka prováděné jedné instrukce je po dobu jednoho

cyklu, využívá řetězení instrukcí. Zástupci RISC architektury jsou AMD, AVR Atmel, MIPS a další. Výrobcem těchto mikroprocesorů jsou převážně IBM a INTEL [19].

Mikroprocesory s RISC architekturou tvoří [19]:

RISC jádro - hlavní funkční jednotka, je určena pro aritmeticko-logické operace a posuvy na 32b datech, 32b adresové výpočty pro přístup k instrukcím a datům a pro řízení činnosti procesoru

VCP – vektorový koprocesor vykonávající aritmetické a logické operace na 64b pakovaných slovech

LMI a GMI – dvě identická programovatelná rozhraní pro externí paměť. Procesor podporuje 32b a 64b přístup k paměti. V druhém případě se přistupuje k dvěma sekvenčně uloženým paměťovým slovům. Paměťové adresování se provádí přes společnou 15b adresovou sběrnici pracující v módu časového sdílení. Adresování je stránkované.

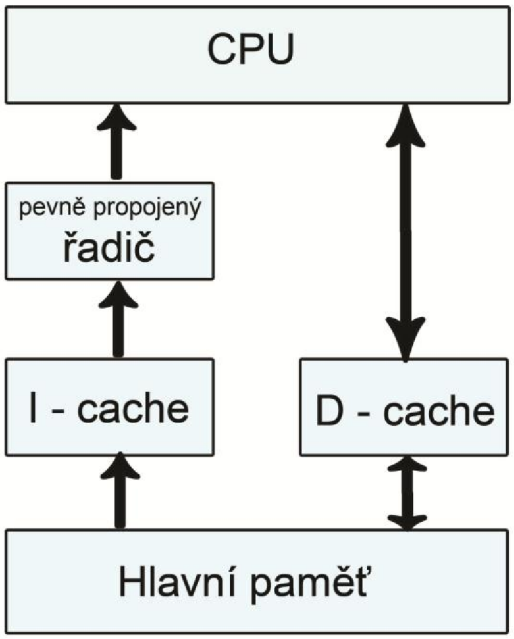
CP0 a CP1 – dva identické komunikační porty, každý zajišťuje přenos dat pomocí obousměrné 8b sběrnice mezi procesorem a externím zařízením. Komunikační porty jsou kompatibilní s porty TMS320C4x. Každý komunikační port obsahuje DMA jednotku, která umožňuje 64b přenos dat mezi portem a externí pamětí.

Výhody:

- Robustnost instrukcí
- Provádění instrukcí je stejné
- Proto možné překrývání instrukcí
- Vkládání instrukce v každém cyklu hodin
- Tím dosahování velkých rychlostí

Nevýhody:

- Delší programy
- Nutnost většího počtu registrů



Obr. 1.2: Architektura RISC s pevně propojeným řadičem a oddělenými paměťmi cache pro instrukce a data.

2. METODY PROGRAMOVÁNÍ

Existuje mnoho způsobů jak naprogramovat mikrokontroléry AVR Atmel. Lze programovat sériově či paralelně. V této kapitole popíšu některé z dostupných metod.

2.1 ISP

In-System Programming) – metoda, umožňuje programování obvodů přímo v zařízení pomocí vyhrazených vodičů určených pro komunikaci při programování. Obvykle není potřeba vyšší napájecí napětí, než je jmenovité napájecí napětí programovaného obvodu. ISP může programovat AVR při extrémně vysokých rychlostech hodin za předpokladu, že cílové AVR běží na vysoké frekvenci a podpoře programátoru. ISP vyžaduje, aby cílový mikroprocesor běžel při taktovací rychlosti nejméně čtyřikrát větší než hodiny ISP. Běžně používaná rozhraní pro ISP jsou [11]:

- **SPI** (Serial Peripheral Interface) - rozhraní pro sériovou synchronní komunikaci,
- **JTAG** (Joint Test Actoin Group) - rozhraní pro diagnostiku a testování integrovaných obvodů,
- **UART** (Universal Asynchronous Receiver Transmitter) - rozhraní pro sériovou asynchronní komunikaci.

Metodou ISP nelze nahrávat (aktualizovat) firmware zařízení bez toho, aby nedošlo k přerušení vykonávání jeho funkce (programovaný obvod je nutné přepnout do režimu programování).

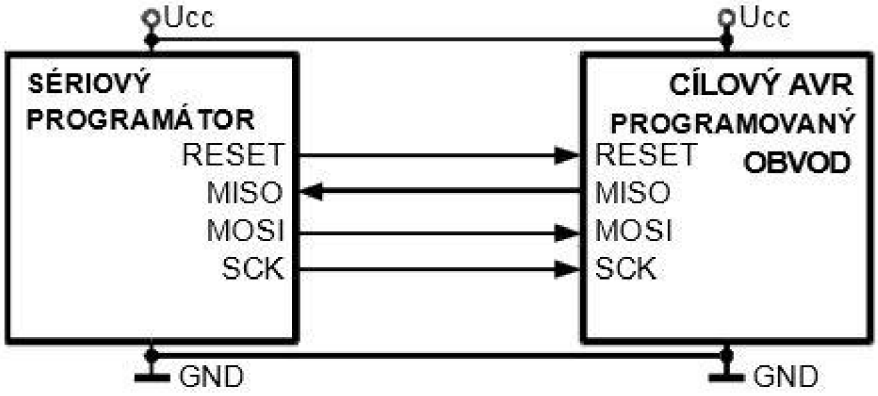
2.1.1 ISP rozhraní

Celá řada obvodů programovaných metodou ISP (tzn. programovat obvod bez nutnosti jejich vyjmutí a vložení do zvláštního zařízení) využívá, SPI rozhraní. Důsledkem toho je existence několika typů velice levných ISP programátorů pro různé obvody (CPLD, EEPROM atd.), které se připojují na sériový nebo paralelní

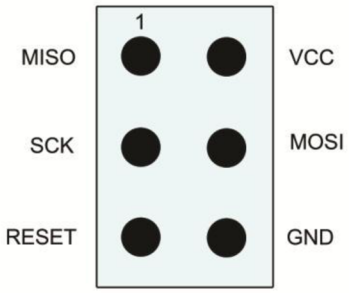
port PC. K propojení AVR mikroprocesoru s programátorem slouží kromě napájení

(v případě, že programátor nemá vlastní napájení) a země ještě další čtyři datové signály [9], [13]:

- **MOSI** - sériový vstup dat do mikroprocesoru,
- **MISO** - sériový výstup dat z mikroprocesoru,
- **RESET** - nulování mikroprocesoru,
- **SCK** - sériové hodiny (synchronizace komunikace),



Obr. 2.1: Zapojení mezi sériovým programátorem a cílovým mikroprocesorem [8]

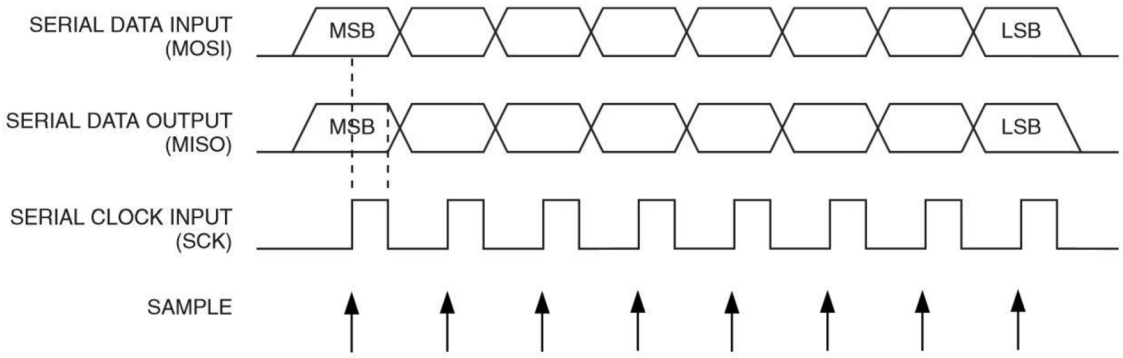


Obr. 2.2: Zapojení ISP – 6ti vývodového konektoru

Tab. 2.1: Zapojení vývodů ISP (převzato z [8])

Vývod	Zkratka	Označení	Význam
1	MISO	Master In Slave Out	Data z programátoru do počítače
2	VTG	Voltage of Target	Senzor napětí na programované součástce - neslouží k napájení
3	SCK	Serial Clock	Hodinový signál z počítače do programátoru
4	MOSI	Master Out Slave In	Data směřující z počítače do programátoru
5	RESET	Reset	Uvedení do programovacího režimu, dodává programovací napětí (kolem 13 V)
6	GND	Ground	Referenční vodič (zem) - nutno spojit se zemí programované součástky

Časový průběh komunikace zobrazuje obrázek 2.3 - jedná se o synchronní přenos dat se synchronizační složkou (SCK). Pokud zapisujeme sériová data do mikrokontroléru, pak jsou data synchronizována s náběžnou hranou hodinového signálu SCK. Pokud data z mikrokontroléru čteme, jsou synchronizována na sestupnou hranu hodinového signálu.



Obr. 2.3: Komunikace ISP – časový diagram [4]

2.2 JTAG

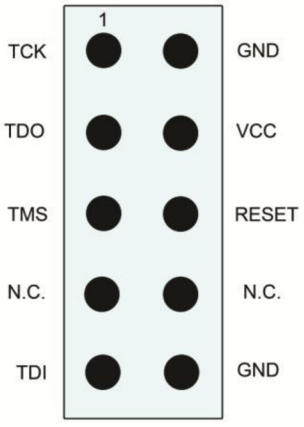
Z technického hlediska se jedná o metodu ladění AVR mikroprocesorů. Nejedná se přímo o způsob programování AVR, přesto rozhraní JTAG umožňuje programování podporovaným zařízením.

JTAG je systémový ladící nástroj, který umožňuje manipulovat a zkoumat stav podporovaného spuštěného AVR. JTAG umožňuje uživateli kdykoliv zastavit provedení kroku, manipulaci s vnitřními registry AVR a mnoho dalšího [11].

2.2.1 JTAG rozhraní

JTAG je zkratka z anglického názvu Join Test Action Group. Jedná se o standart definovaný normou IEEE 1149.1, který vychází z architektury Boundary-Scan pro testování plošných spojů a programování pamětí FLASH. Data jsou přenášena sériově. JTAG rozhraní obsahuje těchto pět signálů [11],[17]:

- **TDI** (Test Data In)
- **TDO** (Test Data Out)
- **TCK** (Test ClocK)
- **TMS** (Test Mode Select)
- **nTRST** (Test ReSeT) - volitelný



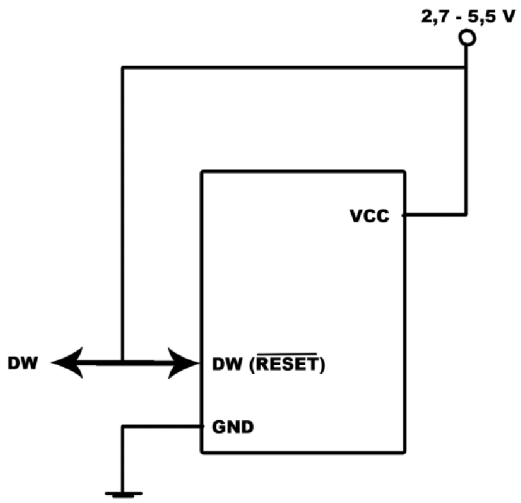
Obr. 2.4: Zapojení vývodů JTAG konektoru

Boundary-Scan

Základní myšlenka metody Boundary-Scan spočívá ve vložení jednoho článku posuvného registru mezi funkční bloky integrovaného obvodu a jeho vývody. Tento základní stavební prvek budeme nazývat Boundary-Scan "buňkou". Tento prvek je natolik univerzální, že může být použit na vstupním i výstupním vývodu integrovaného obvodu [17].

2.3 DebugWIRE

Jedná se spíše o metodu ladící, než metodu, při které lze naprogramovat rozhraní. DebugWIRE může být použit k načítání do programů podporovaných AVR. Rozhraní DebugWIRE používá jediný AVR vývod (/RESET) pro veškerou komunikaci, což je ideální pro přístroje s malým počtem vývodů na mikrokontrolérech AVR [11].



Obr. 2.5: DebugWIRE - rozhraní

2.4 Bootloader

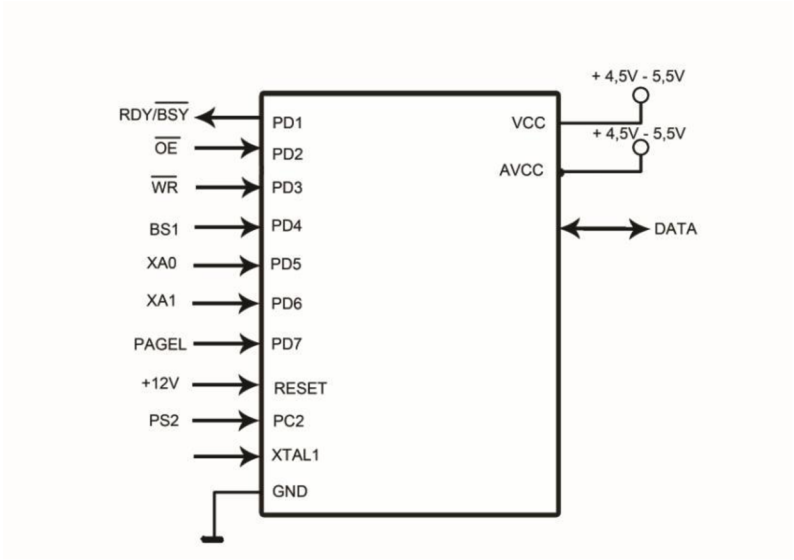
Bootloader (zavaděč) je speciální aplikace, která slouží k zavedení/nahrání aplikace nebo operačního systému do paměti a jejího následného spuštění. Ve světě počítačů je to BIOS, který spustí bootloader, ten pak načte z požadovaného média jádro operačního systému do paměti a následně jej spustí. Ve světě mikrokontrolérů AVR je to o něco jednodušší. Po zapnutí napájecího napětí se (v případě, že jsou správně nastaveny *fuses*) začne okamžitě vykonávat kód bootloaderu, který je uložen na konci paměti flash. Tento bootloader může např. přes rozhraní UART načíst aplikaci, kterou postupně uloží do paměti flash (tedy stejné paměti ve které se nachází on sám) a poté skočit na její první instrukci.

Ovšem bootloader se nemusí omezit pouze na rozhraní UART. Lze využít i jiných rozhraní – I2C, SPI, USB, Ethernet, Bluetooth, a jiné. Toto se s výhodou využívá při vývoji, protože není potřeba externí programátor, i při aktualizaci firmware na běžícím finálním zařízení, které se může nacházet třeba na druhé straně zeměkoule u zákazníka [11].

2.5 High Voltage Parallel programing (HVPP)

Paralelní programování vysokým napětím je způsob programování, který se používá jen zřídka, protože vyžaduje složitější nastavení. HVPP programování se používaná k "oživení" AVR, jejichž fusebits byly konfigurovány jinou metodou programování. Oproti sériovému programování je složitější jelikož programovaný procesor musíme vyjmout a vložit do patice programátoru, což značně zpomaluje výrobu a především počáteční vývoj kódu v AVR mikrokontroléru.

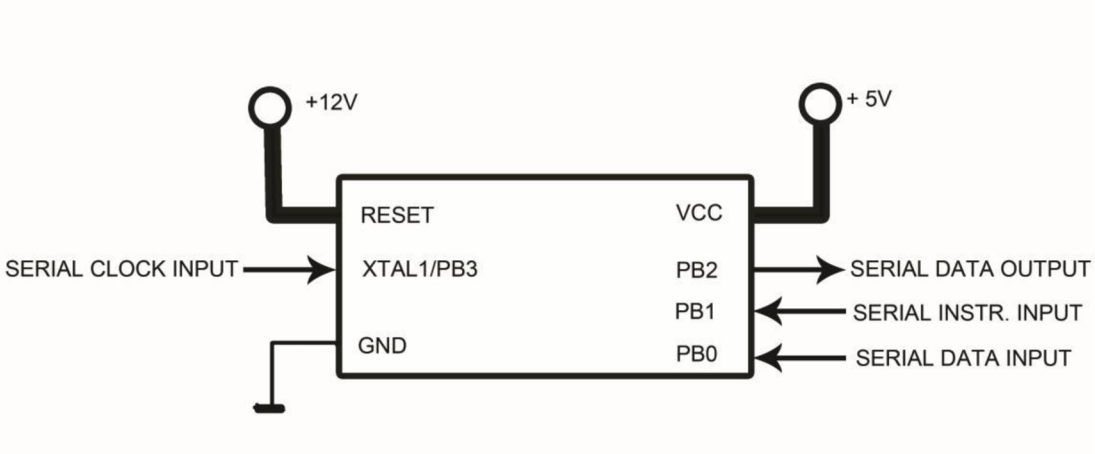
Během programování HVPP se nastavuje cílový /RESET na neobvykle vysokou hodnotu 12 V. Tuto metodu podporují například programátory STK 500 nebo AVR Dragon [3], [11].



Obr. 2.6: Ukázka spojení HVPP s STK500

2.6 High Voltage Serial Programming (HVSP)

Tato metoda programování není vhodná pro mále mikrokontroléry AVR (např. AT90S2323), jelikož nedisponují tak velkým počtem vývodů, které jsou potřeba k paralelnímu programování. Tyto malé mikrokontroléry podporují také režim programování napětím 12V, které je připojeno na vývod RESETu. Zapojení je podrobněji popsáno na obr. 7. Oproti metodě programování ISP má tato metoda řadu výhod. Je rychlejší než ISP a bez programováním vysokým napětím (HVSP) bychom nebyli schopni naprogramovat propojkou bity. HVSP nepotřebuje pro oscilátor externí krystal. Tato metoda má však i svojí nevýhodu: nepodporuje programování mikrokontroléru přímo v obvodu [3], [11].

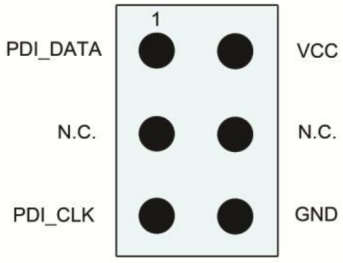


Obr. 2.7: Ideálové zapojení sériového programování vysokým napětím

Při programování HVSP se využívají čtyři vývody (PB0, PB1, PB2, PB3), po kterých jsou posílány bajty. PB3 – XTAL se využívá jako zdroj taktovacích impulzů, PB2 - výstup dat z mikrokontroléru, PB1 – vstup řídících instrukcí a PB0 – posílání dat do mikrokontroléru.

2.7 PDI

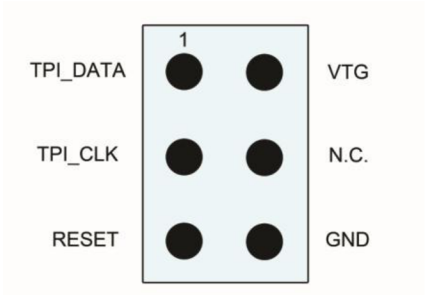
Program a Debug Interface (PDI) je proprietární rozhraní firmy Atmel pro externí programování a ladění zařízení XMEGA přímo na čipu. PDI podporuje vysokorychlostní programování všech energeticky nezávislých paměti (NVM), flash, EEPROM, pojistky, lock-bity a podpis uživatele Row. Toto je dosaženo tím, že zpřístupňuje XMEGA NVM regulátoru přes rozhraní PDI a vykonávání příkazů NVM regulátoru. PDI je dvou vývodové rozhraní pomocí RESET vývodu pro hodinový vstup (PDI_CLK) a speciální datový vývod (PDI_DATA) na vstupu a výstupu [11].



Obr. 2.8: Rozložení vývodů PDI konektor

2.8 TPI

TPI je malé programovací rozhraní pro novější mikroprocesory ATtiny řady AVR. Stejně jako DebugWire TPI používá vývod /RESET jako součást komunikačního rozhraní, ale tím podobnost s DebugWire končí. TPI kromě RESETu vyžaduje dva vývody: TPI_DATA a TPI_CLK [11].



Obr. 2.9: Rozložení vývodů TPI konektor

2.9 IAP

(*In-Application Programming*) – jedná se o modifikaci ISP metody a umožňuje aktualizaci firmwaru bez nutnosti přerušit provoz zařízení. Paměť kódu programu lze zapisovat a číst přímo z aplikace (speciální instrukce uP). K aktualizaci firmwaru lze použít libovolného rozhraní a tak nahrání aktuální verze programu po lokální sítí či z Internetu není v současné době neřešitelný problém [20].

2.10 UISP

(Micro In-System Programmer) má za cíl vytvořit nástroj k programování mikroprocesorů AVR firmy Atmel. Program je volně ke stažení na Internetu včetně zdrojových kódů, což dovoluje jeho snadné šíření i na další operační systémy. Ovládání programu se provádí z příkazové řádky a je ho možno snadno integrovat do libovolného vývojového prostředí. V dnešní době UISP podporuje většinu AVR mikroprocesorů, které lze programovat metodou ISP přes integrované SPI rozhraní [12].

Tab. 2.2: Přehled programovacích metod AVR Atmel

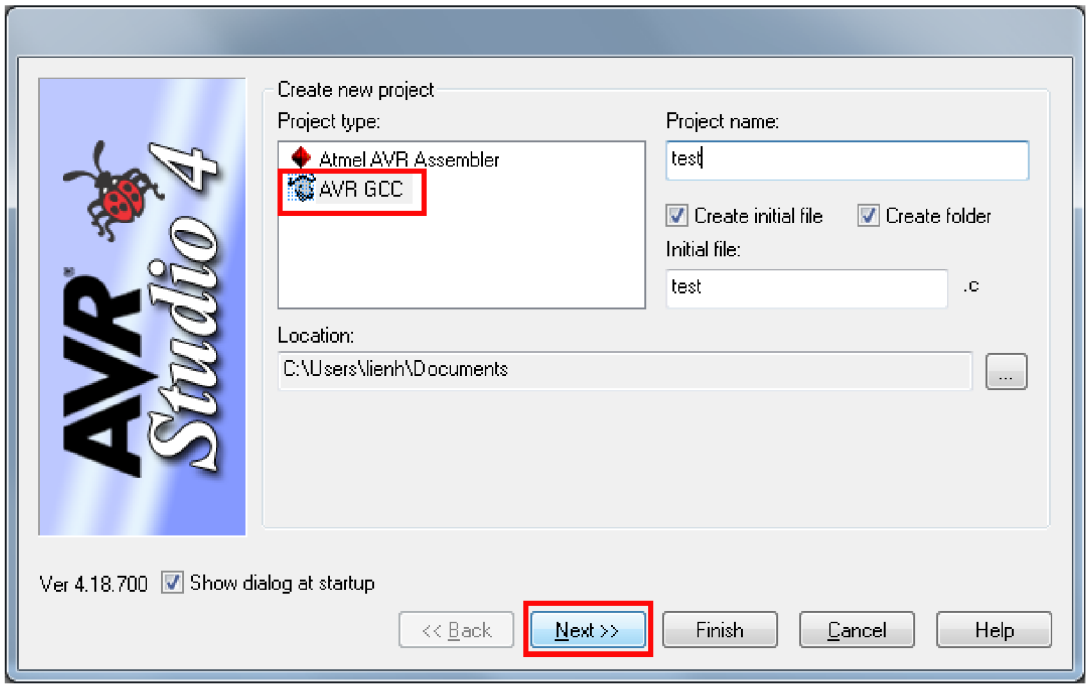
Metoda	Podporované AVR mikroprocesory	Podporované programátory AVR	Hardwarové požadavky
ISP	drtivá většina AVR	AVRISP MKI / II , JTAG MKII , STK500 , STK600 , Dragon, AVRISP klony , AVR910 Programátoři , AVRONE	Programování uvnitř aplikace. Potřeba 6-ti vývodový konektor (viz Obr. 2.2) a programovací adaptér. Signály MOSI, MISO, SCK
JTAG	drtivá většina AVR	JTAG - ICE , JTAG ICE - MKII , Dragon, JTAG ICE - klony , AVRONE , STK600 (pouze programování)	Programování uvnitř aplikace. Potřeba 10-ti vývodový konektor (viz Obr. 2.4) a programovací adaptér. Signály TMS, TDI, TDO, TCK
Debug wire	mnoho menších AVR	JTAG-ICE MKII, Dragon, AVRONE	Ladění mikrokontroléru uvnitř aplikace za pomoci jediného vývodu /RESET.
Bootloader	novější AVR		Jedná se o softwarové programování pomocí zavaděče (přepsaní programovací paměti)
High Voltage Parallel Programming	Nejvíce TINY AVR (s výjimkami)	STK500, STK600, Dragon	Při HVPP teče RESETem 12V. Náročný na HW. Zapojení viz Obr. 2.6
High Voltage Serial Programming	Mnoho TINY AVR (s výjimkami)	STK500, STK600, Dragon	Podobně jako HVPP je RESET na 12V. Zapojení např. obr. 2.7. Náročné na HW.
PDI	XMEGA AVR	STK600, AVRONE, JTAG MKII, Dragon, AVRISP MKII	Externí programování a ladění přímo na mikročipu. Je to dvou vývodové rozhraní (PDI_CLK a PDI_DATA) viz Obr. 2.8
TPI	6-Pin TINY AVR (ATtiny10, atd.)	STK600, Dragon, AVRISP MKII	Pro komunikaci potřebuje tři vývody RESET, TPI_CLK, TPI_DATA. Konektor na obrázku 2.9

3. PROGRAMOVÉ PROSTŘEDÍ - AVR STUDIO

Softwarové, vývojové programovací prostředí AVR Studio pro mikrokontroléry je volně k dispozici na oficiálních internetových stránkách Atmelu [5]. Pro náš projekt bude použita verze AVR Studio 4.18, která je nainstalována v laboratoři firmy Honeywell.

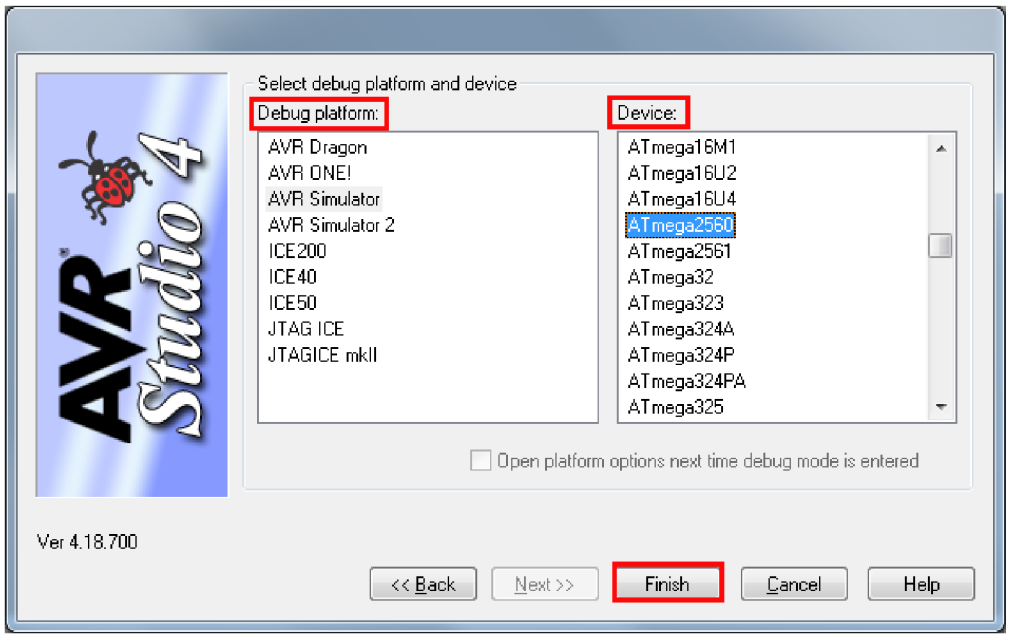
3.1 Vytvoření nového projektu

Při spuštění programu AVR Studio si vybereme nový projekt příkazem Project -> New project. V další nabídce volíme mezi programováním v jazyce C nebo v jazyce assembler. Pro náš projekt volíme jazyk C určený pro programování mikrokontroléru, proto klikneme na nabídku AVR GCC. Napíšeme název projektu a klikneme na tlačítko Next.



Obr. 3.1: Vytvoření nového projektu v AVR Studio

V dalším nabídkovém okně jak je vidět na obrázku 3.2 volíme ladící platformu a hlavně to nejdůležitější typ námi programovaného mikrokontroléru. Pro naši analýzu byly použity ladící platformy AVR Dragon, JTAGICE mkII a STK600, která se po upgradu softwaru přidala do nabídky. Důležité je i správné nainstalování všech potřebných ovladačů. U zařízení (Device) jsme testovali ATmega64, ATmega2560 a ATmega2561. Po výběru klikneme na tlačítko Finish a tím založíme nový projekt.



Obr. 3.2: Výběr ladící platformy a typ mikrokontroléru

Po založení projektu se nám otevře pracovní prostředí programu AVR Studio. Lze jej rozdělit do několika částí [16]:

1. AVR GCC - průzkumník daného projektu

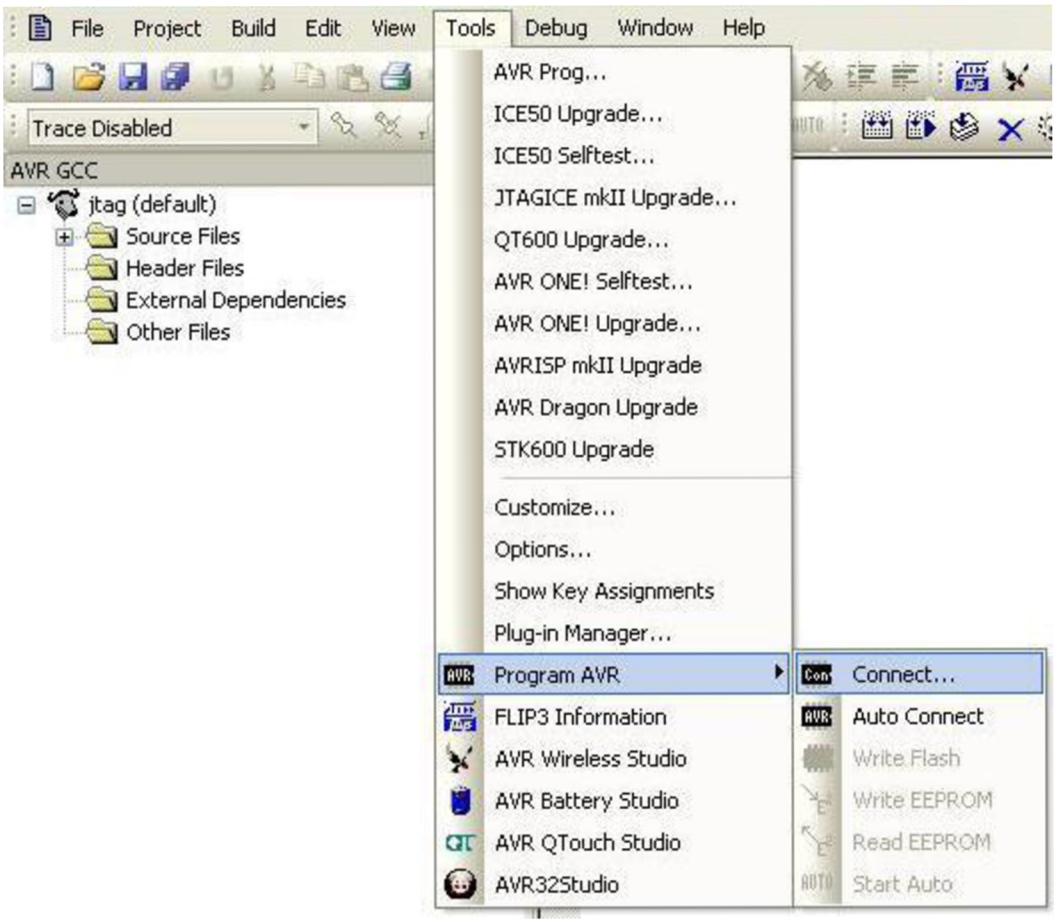
2. Oblast zdrojového kódu - v této části je psán samotný kód programu

3. I/O View - přehled všech vstupních a výstupních registrů

4. Message, Build - Message nám vypíše všechny události a v záložce Build najdeme výpis kompilátoru

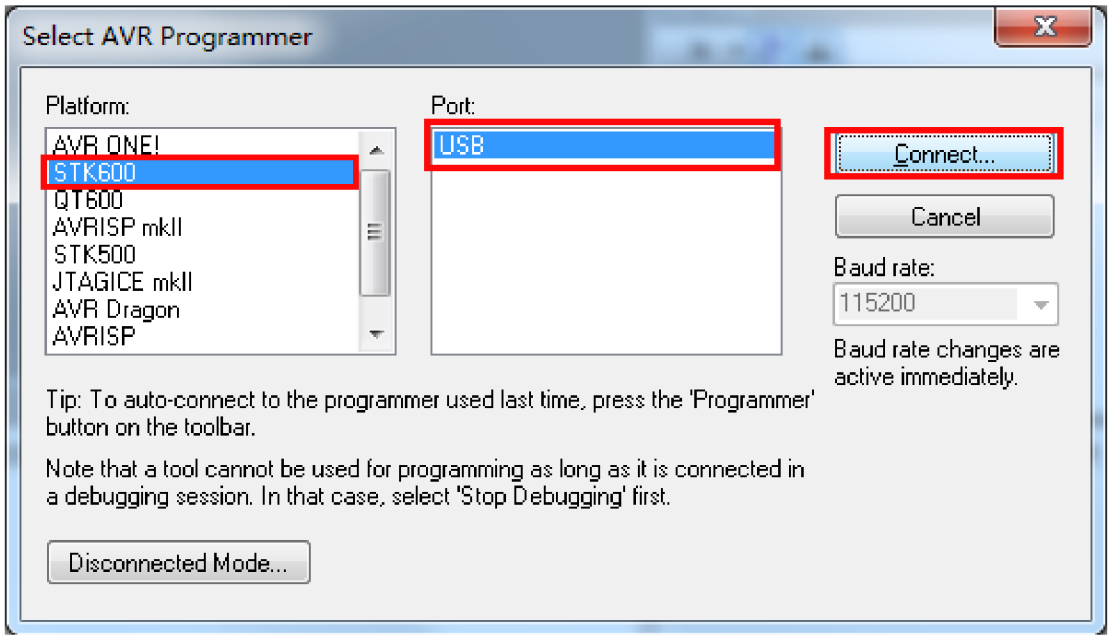
3.2 Testování spojení a programování

Dále nastavíme spojení mezi počítačem a námi zvolenou ladící platformou. Jak je vidět na obr. 3.3 klikneme na záložku Tools, ve které vybereme Program AVR a dáme Connect. Tuto akci můžeme provést i kliknutím na ikonku CON, která je na pracovní ploše.



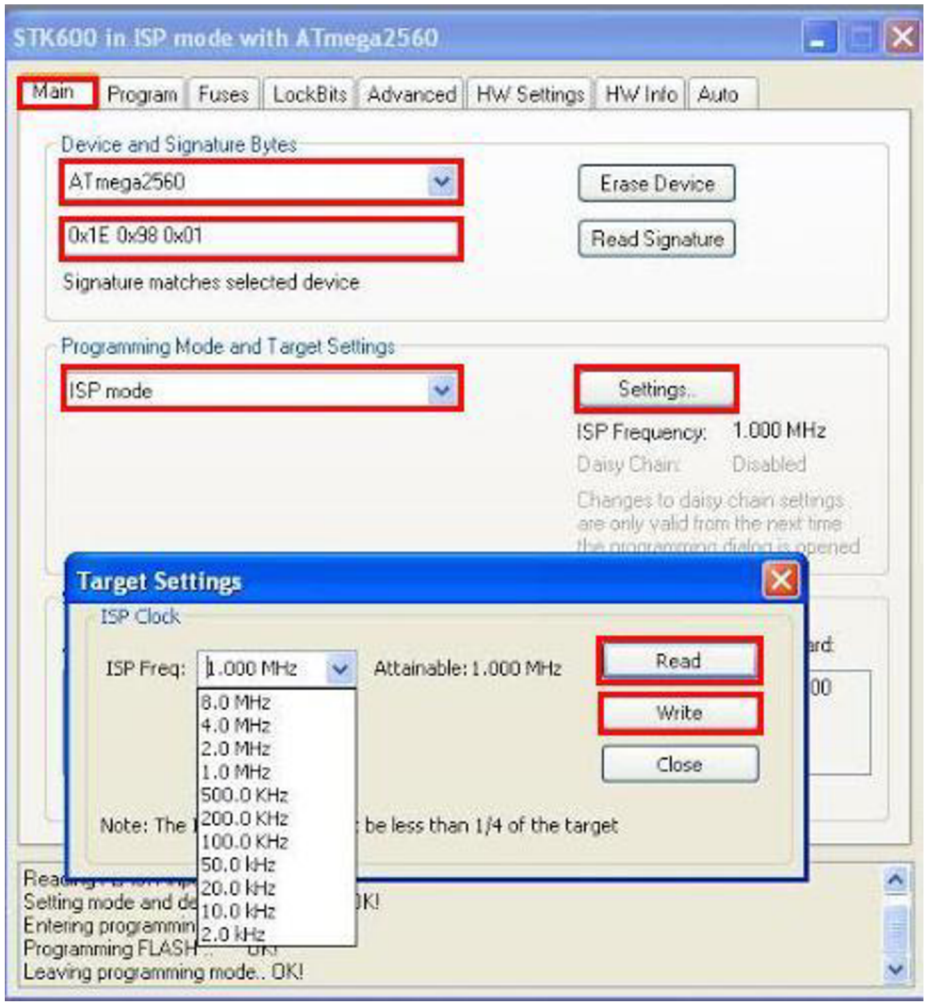
Obr. 3.3: Navázání spojení PC a programátoru

V dalším výběrovém okně volíme ladící platformu STK600 a pro připojení k počítači vybereme USB port a klikneme na tlačítko Connect. Vše je viditelné na obrázku 3.4.



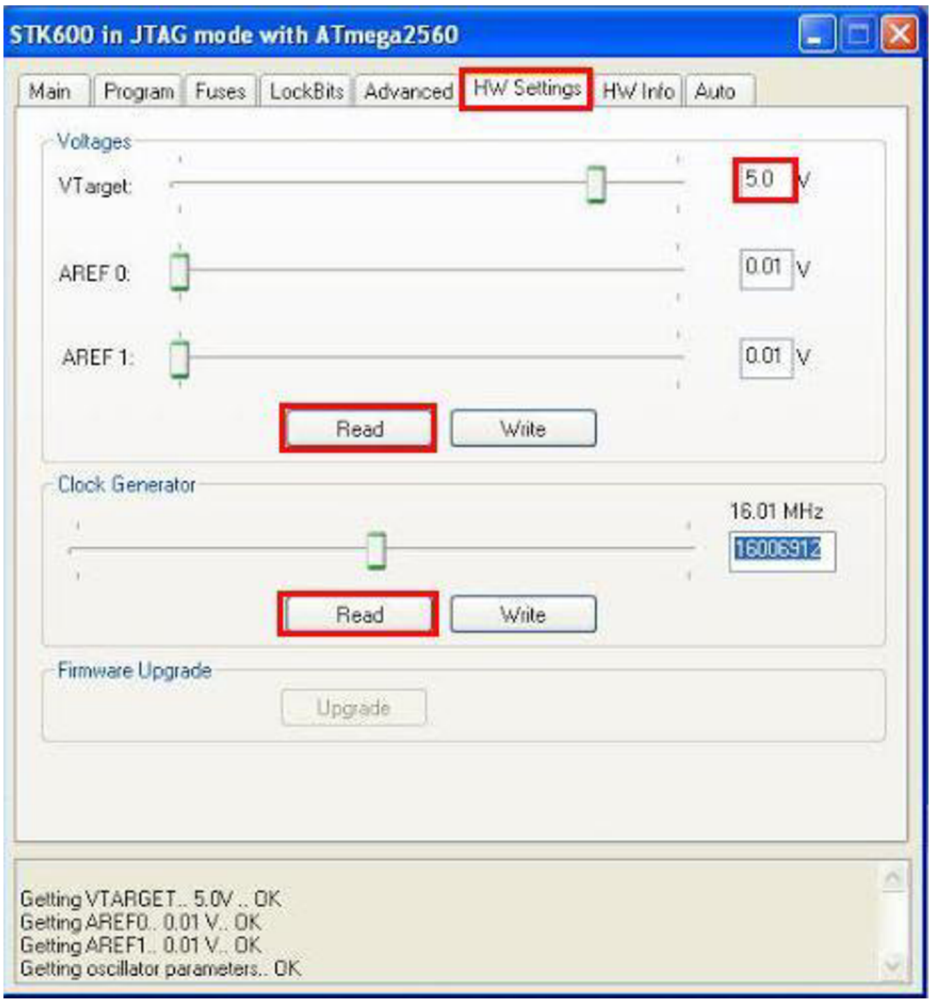
Obr. 3.4: Spojení programátoru s portem USB

Nyní se nám objeví okno, ve kterém můžeme nahrávat program. Překlikneme se však do záložky Main, kde zkontrolujeme správnost spojení. Základem je zadaný správný typ mikroprocesoru, v našem případě např. ATmega2560. Dále klikneme na tlačítko Read Signature a v okně pod výběrem mikroprocesoru se nám zobrazí adresní byty a vypíše se nám hláška: Signature matches selected device. Správnost adresných bytu lze ověřit v katalogovém listu vybraného mikrokontroléru. V případě špatně vybraného mikroprocesoru by se nám načetly nulové byty, které by byly doprovázeny chybovou hláškou o signatuře. Ve výběrovém okně Programming Mode and Target Settings zvolíme programovací mód, v našem případě buď ISP mode nebo JTAG mode. Při zvolení ISP modu klikneme na tlačítko Settings a nastavíme ISP frekvenci. Vybereme například 1 MHz a potvrdíme kliknutím na tlačítko Write. Pro kontrolu správného nastavení frekvence můžeme kliknout na Read a zkontrolovat hodnotu, která musí být ¼ krát menší než frekvence oscilátoru. Výběrové okno zavřeme tlačítkem Close.



Obr. 3.5: Záložka Main (AVR Studio)

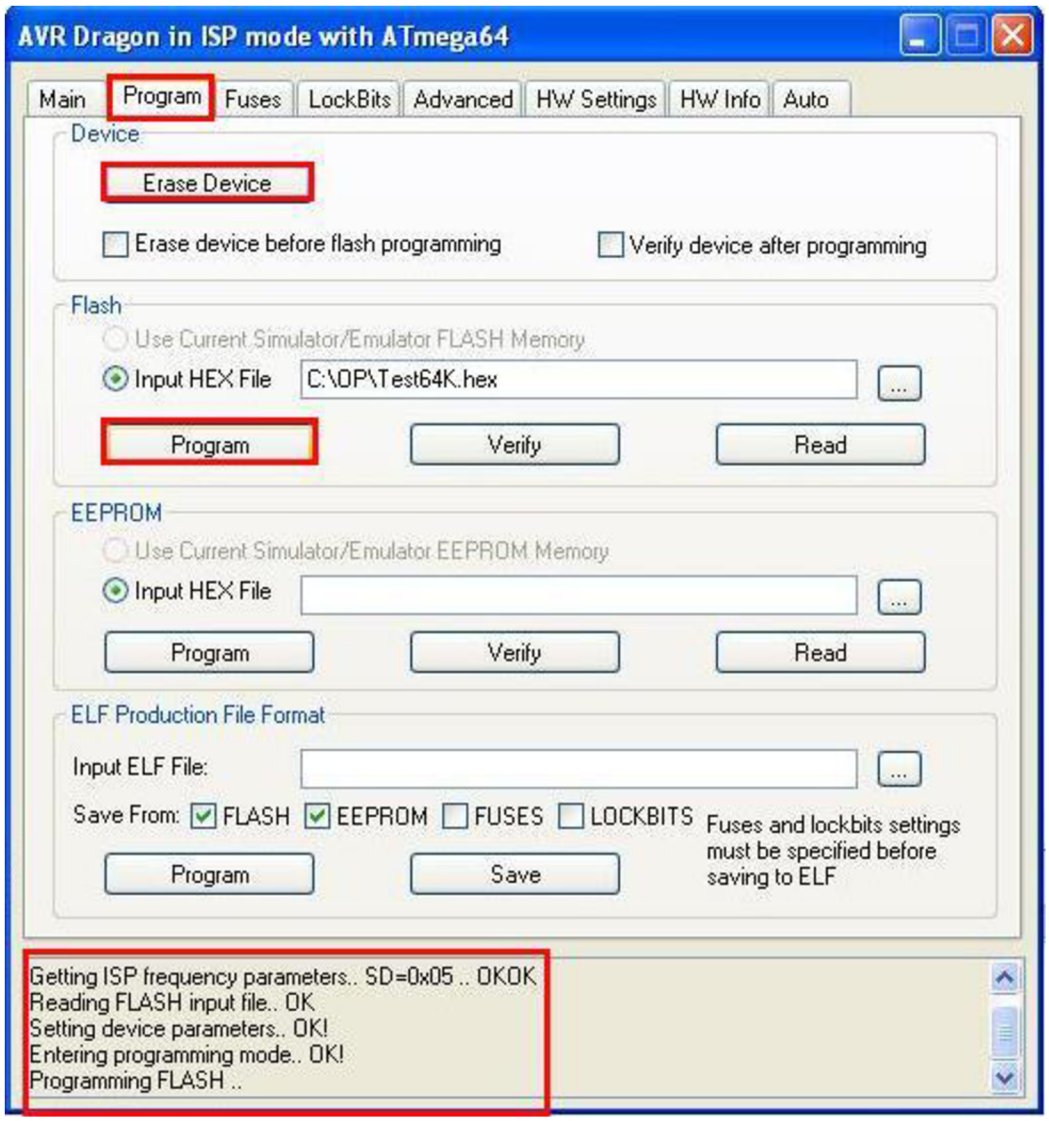
Klikneme na záložku HW settings. Zkontrolujeme, zda máme u VTarget nastaveno napájení. Pokud by byla hodnota nulová, klikneme na tlačítko Read, tím docílíme nastavení správné hodnoty (4,5 – 5 V). Ten samý krok provedeme i u Clock Generator.



Obr. 3.6: Vyčtení hodnot napájení a hodin generátoru

V záložce Fuses zkontrolujeme správné zaškrtnutí pojistek. Pojistka JTAGEN slouží pro povolení programování pomocí JTAGu, pojistka SPIEN nám povoluje programovat zařízení ISP metodou. CKDIV určuje dělení systémových hodin 8. SUT_CKSEL reprezentuje volbu kmitočtu interního RC oscilátoru. EESAVE zachovává obsah paměti EEPROM během mazání mikroprocesoru. WDTON časovač obvodu Watch dog. Hodnoty High a Low reprezentují nastavení Fuses bitů a zapisují se do mikroprocesoru. Můžeme, také zaškrtnou hodnotu Auto read, tím se nám automaticky načtou hodnoty Fuses.

V poslední řadě klikneme na záložku Program. Kde máme možnost naprogramovat Flash nebo EEPROM paměť. Můžeme si i vyčíst příslušný *.hex soubor a to kliknutím na tlačítko Read. Před samotným programováním můžeme vymazat zařízení kliknutím na tlačítko Erase Device. Pro spolehlivé naprogramování mikrokontroléru je vhodné mít zaškrtnuty položky mazání paměti před programováním a verifikace obsahu po naprogramování. Programování se provádí tlačítkem Program, kdy si nejprve najdeme cestu k *.hex souboru, který chceme zapsat do mikrokontroléru. V okně dole můžeme sledovat výpis jednotlivých akcí, zda proběhly v pořádku a nedošlo k žádné chybě. Tlačítkem Verify provádíme případnou kontrolu naprogramovaného obsahu paměti.



Obr. 3.7: Záložka Program (AVR Studia 4)

4. ANALÝZA RYCHLOSTI ROZHRANÍ

Rychlost programováni je hlavní požadavek pro návrh programátoru AVR v průmyslovém použití. Zejména zápis sériového čísla do samotného mikrokontroléru by měl být uskutečněn v co nejkratším čase. Proto jsem provedl dvě analýzy, 1. na průmyslové desce od firmy Honeywell, na které jsou dva mikroprocesory od firmy Atmel a to 64k a 256k procesor (více v kapitole 3.4 a 3.5) a 2. za pomocí STK600 (kapitola 3.3), ke kterému jsem pomocí sendvičové metody připojil ATmegu2560. Pro první analýzu jsem použil dva komerční programátory. Jedná se o AVR Dragon a JTAG ICE mkII (více viz kapitola 3.1 a 3.2). Ze všech výše uvedených metod programování jsem došel k závěru, že nejvhodnější rozhraní pro rychlé programování v průmyslu bude sériové rozhraní ISP nebo JTAG. Programování vysokým napětím je pro průmysl měně vhodné z důvodu nutnosti použití více jak 20vývodů. U metody bootloader, by nám vzrostly samotné náklady na programování z důsledku oslovení externí firmy, která by nám dodávala mikrokontroléry, ve kterých by byl nahrán firmware firmy, ale chybělo by unikátní sériové číslo. Pracovníci firmy Honeywell by tak museli mikrokontrolér programovat znovu (unikátní sériové číslo). Metoda Debug wire se spíše hodí k ladění mikrokontroléru než k jeho programování. TPI a PDI lze použít jen u určitých řad AVR mikrokontroléru, což omezuje jejich nasazení v průmyslu.

4.1 Přehled programátorů a mikrokontrolérů

Před samotným měřením si popíšeme komerční programátory, které jsem použil pro 1. a 2. analýzu. Dále je uveden i malý přehled vlastností použitých mikrokontrolérů:

4.1.1 Programátor Dragon AVR

AVR Dragon je programátorem firmy Atmel. Jedná se o velmi levný, tudíž všem dostupný vývojový a ladící prostředek pro práci s mikrokontroléry řady ATiny a ATmega. Kit AVR Dragon podporuje všechny módy programování mikrokontroléru rodiny AVR. Vývojový prostředek podporuje ve spolupráci se softwarem AVR studio celou řadu mikrokontroléru viz [6].

Základní vlastnosti AVR Dragon [6]:

- podpora programovacího a ladicího rozhraní
- programování přímo v aplikaci ISP 3vodičové
- JTAG programovací 4vodičové
- sériové programování vyšším napětím, paralelní programování
- komunikace s PC pomoci USB rozhraní
- debugWIRE - jednovodičové rozhraní AVR
- možnost externího napájení
- JTAG ladění pro součástky s FLASH pamětí 32kB

Na desce Dragon AVR je velká řada možností pro připájení konektorů. Pro naši analýzu rychlosti programování budou podstatné konektory ISP6 a JTAG10.

4.1.2 JTAG ICE mkii

AVR JTAGICE mkII je adaptér od firmy Atmel pro odlaďování programu na čipu přes rozhraní JTAG nebo debugWIRE. Adaptér umožňuje sledovat paměti flash, EEPROM i SRAM, registrový soubor, programový čítač, fuse a lock bity a všechny vstupně výstupní moduly, a další možnosti odlaďování. Adaptér se s počítačem propojuje buď přes RS232, přičemž potřebuje externí napájení v rozsahu 9-12 V. Alternativou je komunikace přes USB, čímž je zajištěno i napájení. Propojení s deskou je realizováno plochým kabelem s koncovkou JTAG, ke které jsou dodávány i redukce. Momentální stav a připojení adaptéru signalizují 3 LED. Červeně je signalizováno připojení napájení. Korektní připojení napájené desky je signalizováno zelenou LED. Poslední LED svítí v případě právě probíhající komunikace mezi deskou a PC. Instalace ovladačů programátoru by měla proběhnout společně s instalací AVR Studia. V opačném případě jsou pro ruční instalaci všechny ovladače dostupné ve složce AVR Tools [7].

Základní vlastnosti [7]:

- podpora AVR studiem
- podpora JTAG programování
- plně Real Time emulace - digitální i analogové funkce
- komunikuje s PC přes RS-232, nebo USB
- široký rozsah napájení programovaného obvodu - 1.8V až 5.5 V
- možnost napájení přes USB port

Obr. 4.1: Ukázka spojení JTAG programátoru s programovanou aplikací [7].

4.1.3 AVR STK 600

Vývojový kit STK600 je nástupcem staršího STK500. Je určen pro 8-mi a 32-bitové AVR mikroprocesory. Umožňuje rychlé vyvíjení kódu a obsahuje funkce pro podporu prototypů a testování nových vzorků. Připojení AVR mikrokontroléru k STK600 jde pomocí inovativního směrování a sendvičového systému karet, který směruje signály ze zařízení do příslušného hardwaru. Testovaná karta je k STK600 připevněna pomoci umělohmotných šroubků viz obr. 4.2. STK 600 je vybaven 6-ti vývodovým ISP konektorem a pro rozhraní JTAG slouží klasický 10-ti vývodový konektor [9].

Základní vlastnosti [8]:

- připojení přes USB port k PC - umožnění programování a ovládání
- napájení z USB sběrnice nebo z externího 10 - 15V DC zdroje
- nastavitelné VCC (0 - 5.5V)
- dvě nastavitelné referenční napětí s vysokou přesností (0 - 5.0V , 10mV res)
- hodiny oscilátor, nastavitelné za chodu z AVR Studia (0 - 50MHz , 0,1 % res)
- ISP - tinyAVR a megaAVR
- PDI Programování AVR XMEGA
- JTAG programování 8-mi bitových megaAVR , 8/16-bit AVR XMEGA

a 32 - bitových AVR zařízení

- aWire Programování 32- bitových AVR zařízení

Obr. 4.2: STK600 “Sendvičová” metoda [9]

4.1.4 Mikroprocesor ATmega64

Mikroprocesor firmy Atmel ATmega64 je 8mi bitový a vychází z RISC architektury AVR. Instrukce jsou uzavřeny v jediném hodinovém cyklu, ATmega64 dosahuje propustnosti blížící se 1 MIPS na 1 MHz, což umožňuje optimalizovat spotřebu energie v závislosti na rychlosti procesu.

Základní vlastnosti [2]:

- Paměť EEPROM 2 kB
- Paměť FLASH 64 kB
- Paměť RAM 4 kB
- Napájecí napětí 4,5 - 5,5 V
- Rozhraní SPI
- Taktovací kmitočet 0 MHz - 16 MHz
- JTAG

4.1.5 Mikrokontroléry ATmega2561 a ATmega2560

Jedná se o 8mi bitové mikroprocesory firmy Atmel. Lze programovat pomoci ISP či JTAG rozhraní. Velikost paměti FLASH je 256 kB. Mikroprocesory dosahují propustnosti 1 MIPS na 1 MHz.

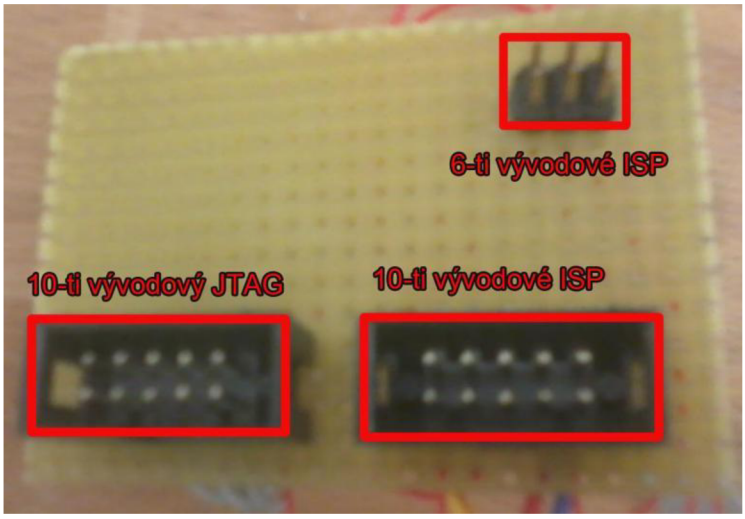
Základní vlastnosti [1]:

- Paměť EEPROM 4 kB
- Paměť FLASH 256 kB
- Paměť SRAM 8 kB
- Napájecí napětí 4,5 - 5,5 V
- Rozhraní SPI
- Taktovací kmitočet 0 MHz - 16 MHz
- JTAG

4.2 První analýza

K prvnímu měření zapůjčila firma Honeywell vývojovou desku, která byla osazena dvěma mikrokontroléry firmy Atmel. Jednalo se o typy ATmega64 a ATmega2561, základní technické parametry těchto mikrokontroléru jsou v kapitole 4.1.4 a 4.1.5. Analýza byla provedena na dvou komerčních programátorech AVR Dragon (viz kap. 4.1.1) a JTAG ICE mkii (viz kap. 4.1.2).

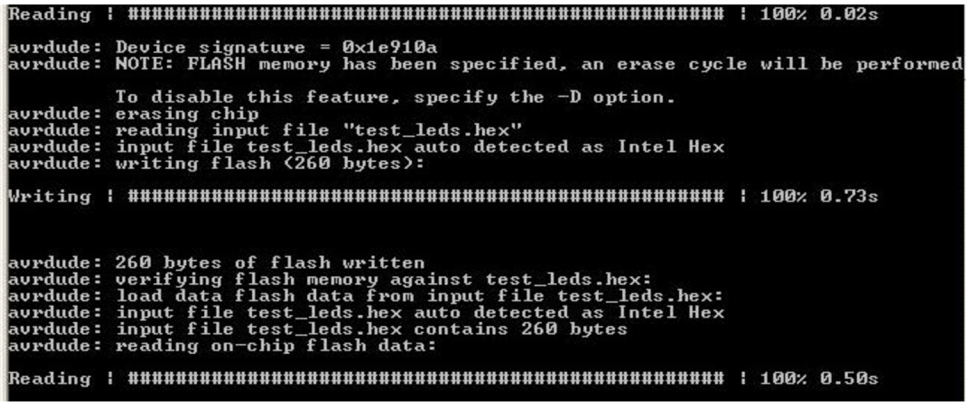
Na vývojové desce byly pro každý mikroprocesor vyvedeny dvě programovací rozhraní a to ISP a JTAG. Tyto rozhraní nebyly na desce v minulosti využívány, tudíž byly dírky pro konektor zality cínem. Za pomocí hrotové pájky a odsávačky byly dírky odsáty a připraveny pro osazení nových konektorů. Obě rozhraní, měly však nestandardní počet vývodů. Pro ISP to byly 3 pro základní datové toky SCK, MOSI a MISO. JTAG byl osazen 8mi vývodovým konektorem. Abych mohl využít komerčních programátorů, musel jsem převést signály ISP a JTAGu na standardní konektory. Konektor JTAGu z 8mi vývodového na 10-ti vývodový a ISP ze tří vývodového na 6-ti vývodový pro AVR Dragon a 10-ti vývodový pro JTAG ICE mkII. Jelikož tří vývodový ISP obsahoval jen signály SCK, MISO, MOSI musel jsem zbylé signály VCC, RESET a GND připojit z konektoru JTAG. Mnou vytvořená redukce konektorů je vidět na obr.4.3.



Obr. 4.3: Vytvořená redukční deska pro ISP a JTAG

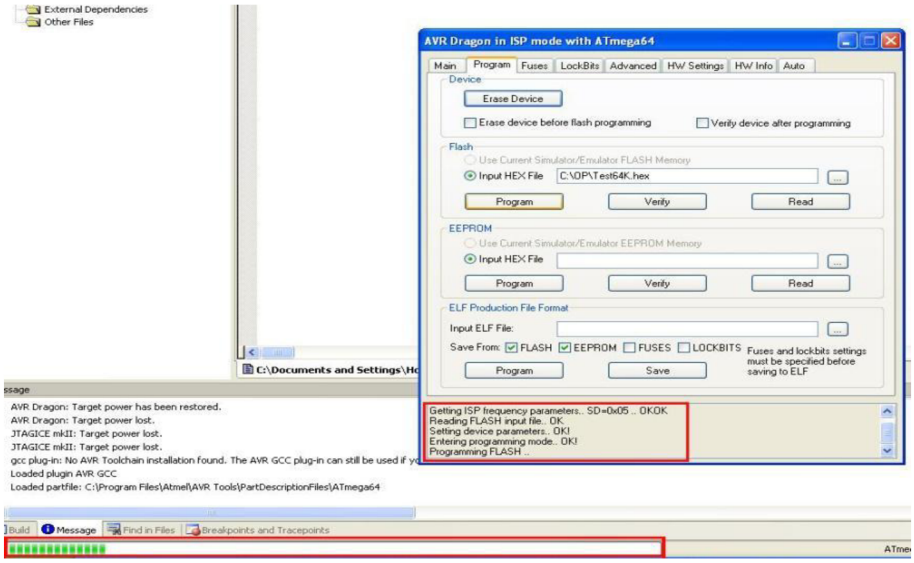
4.2.1 Měření rychlostí

Samotné měření rychlostí programování jsem prováděl dvěma metodami. První byla za pomocí odečtení času z AVR Studia, kdy při odečtu času mohlo dojít k nepatrné chybě. Druhá byla za pomocí příkazového řádku. Kdy jsem v cmd zadal příkaz *avrdude -c avrisp -p m64 -U flash:w:test_leds.hex.* Jak je vidět na obrázku 4.4 klasickým odečtením časové hodnoty po vypsání 100% zápisu mikrokontroléru.



Obr. 4.4: Měření doby rychlosti pomocí příkazového řádku a AVRdude

Průběh programování je v AVR Studio graficky znázorněn v levé dolní části okna. Na obr. 4.5 je vidět narůstající zaplnění FLASH paměti námi definovaným *.hex souborem.



Obr. 4.5: AVR Studio – grafický průběh programování

4.2.2 Měření ISP a JTAGu pomocí Dragonu a JTAG ICE mkII

Vývojovou desku oživím pomocí laboratorního zdroje a nastavím jej na hodnotu mezi 4,5 – 5 V. Zapojím konektor pro ISP, přičemž musí být zapojený i konektor JTAGu (nutnost chybějících signálů). U ISP metody měříme na frekvenci 200 kHz, 1 MHz, 2 MHz a JTAG bude nastaven na hodnotu ¼ krát menší než frekvence oscilátoru. Měření provádím na programátoru AVR Dragon a JTAG ICE mkII.

Tab. 4.1: Měření ISP na ATmega64 – AVR Dragon

Atmega 64 - AVR Dragon - ISP	1.měření [s]	2.měření [s]	3.měření [s]	4.měření [s]	5.měření [s]	Průměr [s]
200 KHz	1,45	1,46	1,44	1,47	1,46	1,456
1 MHz	0,47	0,49	0,47	0,45	0,48	0,472
2 MHz	0,3	0,34	0,35	0,33	0,32	0,328

Tab. 4.2: Měření JTAGu na ATmega64 – AVR Dragon

Atmega 64 - AVR Dragon - JTAG	1.měření [s]	2.měření [s]	3.měření [s]	4.měření [s]	5.měření [s]	Průměr [s]
1/4 int OSC	0,74	0,79	0,75	0,77	0,77	0,764

Tab. 4.3: Měření ISP na ATmega64 – JTAG ICE mkII

Atmega 64 - Jtag Ice mkii - ISP	1.měření [s]	2.měření [s]	3.měření [s]	4.měření [s]	5.měření [s]	Průměr [s]
200 KHz	2,03	2,08	2,05	2,04	2,05	2,05
1 MHz	0,67	0,67	0,67	0,65	0,67	0,666
2 MHz	0,4	0,43	0,44	0,4	0,42	0,418

Tab. 4.4: Měření JTAGu na ATmega64 – JATG ICE mkII

Atmega 64 - JTAG ICE mkII -JTAG	1.měření [s]	2.měření [s]	3.měření [s]	4.měření [s]	5.měření [s]	Průměr [s]
1/4 int OSC	0,67	0,61	0,65	0,64	0,64	0,642

Tab. 4.5: Měření ISP na ATmega2561 – AVR DRAGON

Atmega 2561 -AVR Dragon - ISP	1.měření [s]	2.měření [s]	3.měření [s]	4.měření [s]	5.měření [s]	Průměr [s]
200 KHz	32,3	32,2	32,3	32,5	32,3	32,32
1 MHz	13,5	13,4	13,5	13,55	13,43	13,476
2 MHz	8,4	8,47	8,45	8,49	8,45	8,452

Tab. 4.6: Měření ISP na ATmega2561 – JTAG ICE mkII

Atmega 2561 -Jtag Ice mkii - ISP	1.měření [s]	2.měření [s]	3.měření [s]	4.měření [s]	5.měření [s]	Průměr [s]
200 KHz	38,69	38,65	38,66	38,67	38,66	38,666
1 MHz	15,4	15,39	15,43	15,4	15,37	15,398
2 MHz	9,22	9,27	9,3	9,29	9,24	9,264

U mikrokontroléru ATmega2561 bylo změřeno jen programování pomocí ISP metodou. Při sestavování spojení s JTAG konektorem se spojení nezdařilo. Důvodem je asi špatný kontakt na vývojové desce pro konektor popř. jiná chyba, kterou se mi nepodařilo odhalit.

4.3 Druhá analýza

Při měření rychlosti programování při druhé analýze jsem použil novější model STK600. Specifikace tohoto kitu jsou v kap. 4.1.3. Deska STK600 je napájená přes USB port z PC. Za pomocí sendvičové metody, která je dobře patrná z obrázku 20, jsem připevnil čtyřmi šroubky testovanou desku s mikroprocesorem ATmega2560. Pro naprogramování firmwaru jsem jako první použil rozhraní ISP. Oboustranným, 6-ti vývodovým, kabelem jsem propojil příslušné kolíky. Spojení s AVR Studiem se však v tuhle chvíli nedařilo. Bylo zapotřebí aktualizace STK600 přímo v programu. Po této aktualizaci se nabídka spojení s STK600 objevila v nabídce. Též bylo nutné nainstalovat ovladače pro Windows.

4.3.1 Měření ISP na STK600

V programu AVR Studio jsem ověřil spojení s počítačem a v záložce Main jsem nastavil ISP metodu a vyčetl jsem signaturu. Pro měření jsem použil tento rozsah frekvencí 100kHz, 200kHz, 1MHz, 2MHz. Hodnota musela být ¼ krát menší než frekvence oscilátoru. U testovaného mikroprocesoru je to 8Mhz takže námi nastavená hodnota 2MHz je hraniční.

Tab. 4.7: Měření ISP metody na STK600

Atmega 2560 - ISP	1.měření [s]	2.měření [s]	3.měření [s]	4.měření [s]	5.měření [s]	Průměr [s]
100 KHz	55,57	55,61	55,43	55,93	55,51	55,61
200 KHz	29,47	29,78	29,33	29,49	29,62	29,538
1 MHz	9,31	9,32	9,28	9,35	9,3	9,312
2 MHz	6,89	6,83	6,42	6,42	6,63	6,638

Z naměřených údajů vyplývá, že při nastavení menší frekvence je doba programování mnohem větší než u hraniční hodnoty 2 MHz. Nutno vzít v úvahu i to, že mikrokontrolér, který programujeme přímo z výroby má nastavenou frekvenci na nižší hodnotu. Tudíž před samotným programováním musíme nastavit frekvenci na 2 MHz, aby byla rychlost efektivní.

4.3.2 Měření JTAGu pomocí STK600

Podobně jako u metody ISP, propojíme JTAG konektor 10-ti vývodovým oboustranným kabelem. Po navázání spojení v záložce Main nastavíme JTAG mode. Frekvece bude nastavena na ¼ frekvence oscilátoru.

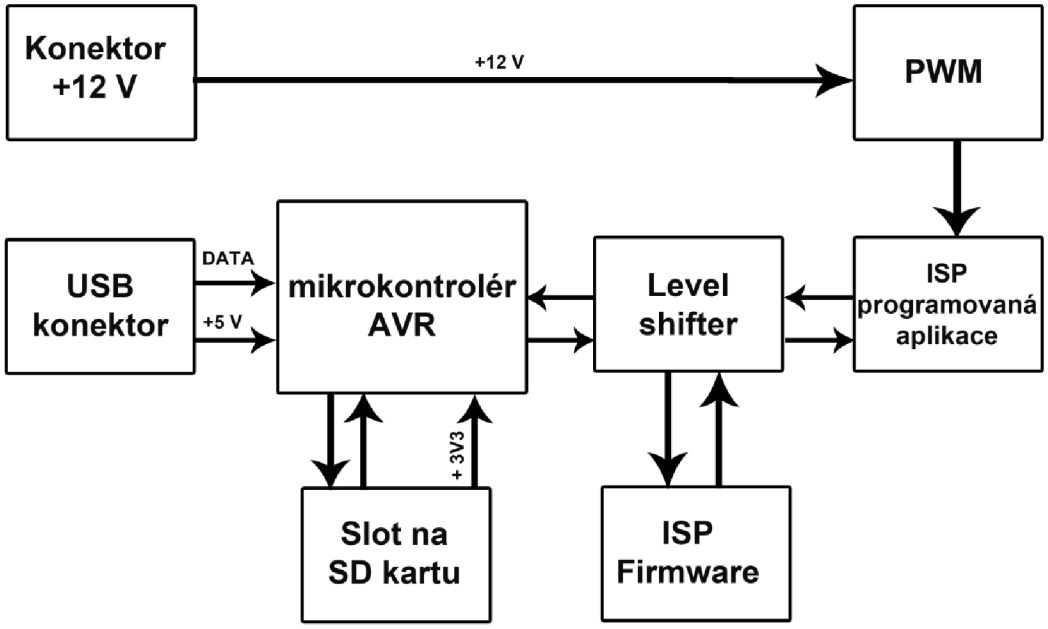
Tab. 4.8: Měření JTAGu pomocí STK600

Atmega 2560 -STK600 - JTAG	1.měření [s]	2.měření [s]	3.měření [s]	4.měření [s]	5.měření [s]	Průměr [s]
1/4 int OSC	8,21	8,8	9,09	8,79	8,56	8,69

Při porovnání průměrných hodnot programování při nastavené frekvenci 2 MHz dojdeme k závěru, že ISP přenese stejně velký firmware do 256kB Flash paměti o cca 2s rychleji než JTAG. Musíme brát v potaz i to, že mikrokontrolér z výroby musíme prvně přenastavit na vyšší frekvenci.

5. POŽADAVKY NA PROGRAMÁTOR A ZVOLENÉ ŘEŠENÍ

Aby mohl průmyslový programátor pracovat správně, rozdělíme si jej na jednotlivé funkční bloky, viz obrázek 5.1. Hlavními požadavky jsou rychlost programování a schopnost změny části kódu. U průmyslového programátoru se také cení abstence jakéhokoliv přepínače. Vše musí být řízeno pomoci signálů z mikrokontroléru [18].



Obr. 5.1: Blokové schéma

Základním a taky nejdůležitějším funkčním blokem celého programátoru bude mikrokontrolér, který řídí celý obvod a podporuje komunikaci s počítačem po USB sběrnici. Mikrokontrolér bude pracovat při napětí + 5 V, které bude dodáváno pomocí USB konektoru.

Z výsledků měření v předchozí kapitole volíme pro programování hostující aplikace konektor pro ISP metodu. Tím splníme podmínku o rychlém programování. Mezi funkční blok mikrokontroléru a ISP konektoru vložíme obousměrný „Level Shifter“, který nám oddělí 5 V logiku od logiky pracující při nižším napětí. Pro nahrání firmwaru do našeho vnitřního mikrokontroléru bude sloužit samostatný ISP konektor nebo bootloader z USB sběrnice.

Dalším blokem v programátoru bude slot na SD kartu, ve které bude uložena část kódu (firmware). Tento firmware bude pro všechny hostující mikrokontroléru, stejného typu, neměnný. SD karta pracuje v SPI módu, za pomocí souborového systému FAT16. SD karta pracuje s maximálním napětím 3,6 V, proto napětí na slot SD karty bude přivedeno o hodnotě 3,3 V ze stabilizátoru uvnitř mikrokontroléru (vývod Ucap).

U programované aplikace je požadavek na možnost změny napětí VTG pomocí signálu PWM přímo z mikrokontroléru. Tento regulační obvod bude tvořit jednoduchý regulátor napětí, na který se přivede napětí + 12 V z externího zdroje. Regulátor doplníme o operační zesilovač umožňující řízení regulátoru z mikrokontroléru. Samotné spínání napěťového signálu na vstup VTG obstará tranzistor MOSFET.

Další podmínkou pro průmyslový programátor je umožnění modifikace kódu. Tedy spolu s firmwarem, který bude uložen na SD kartě, připojit jedinečná data pro každý externí mikrokontrolér. Tato část bude řešena v počítači, kdy obsluhující zvolí daný registr EEPROM paměti, do kterého vloží příslušné údaje pro daný mikrokontrolér.

6. VÝBĚR VHODNÝCH KOMPONENT A JEJICH ZAPOJENÍ

Při výběru vhodných komponent pro průmyslový programátor, musíme brát v potaz dané požadavky – rychlost programování a absenci přepínačů. Musíme se však také dívat na použité pouzdro, aby jeho dostupnost byla i na českém trhu a také, aby se nám pouzdro shodovalo s knihovnami v návrhovém prostředí Eagle.

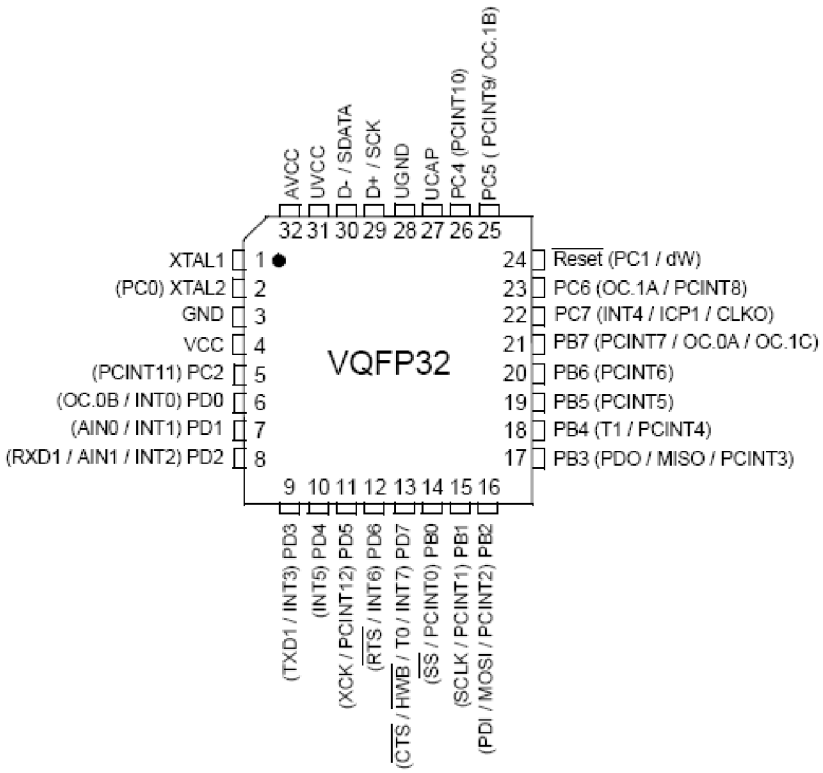
6.1. Mikrokotrolér AT90USB162

Programátor bude řídit integrovaný obvod z rodiny mikroprocesorů Atmel AVR. Jedná se o 8mi bitový procesor s 8k/16k pamětí Flash. Obvod podporuje komunikaci s USB, čímž nám například odpadne obvod RT232, který bychom museli použít např. u ATmega32 která byla v původním návrhu práce. Programování externí aplikace bude probíhat pomocí ISP. AT90USB162 obsahuje bootloader, který využijeme pro komunikaci s PC. Mikrokontrolér je vybaven dvěma jednoduchými osmibitovými čítači/časovači s PWM funkcí, která bude využitá pro řízení regulátoru napětí. Tím, že provádí výkonné instrukce v jediném hodinovém cyklu, dosahuje 1 MIPS na 1 MHz [4].

Technické vlastnosti:

Organizace paměti Flash	16k x 8bit
Kapacita paměti EEPROM	512B
Kapacita paměti SRAM	512B
Pouzdro	TQFP32
Kmitočet taktování	16MHz
Počet kanálů PWM	5
Počet 8mi bitových čítačů	1
Pracovní napětí	2,7 – 5,5V

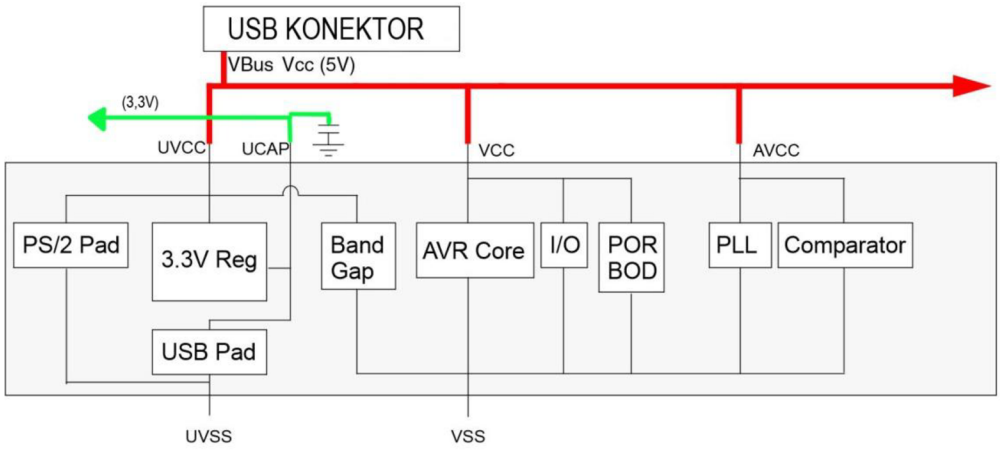
Rozložení vývodů mikrokontroléru je patrné z obr. 6.1. Blokové schéma AT90USB162 nalezneme v příloze diplomové práce.



Obr. 6.1: Rozložení vývodů mikrokontroléru (převzato z [4])

6.1.1 Stabilizace na 3,3 V

Mikrokontrolér AT90USB162 umožňuje stabilizovat napětí na 3,3 V přímo z čipu. Napětí je z regulátoru vyvedenou na vývod UCAP. Pro lepší pochopení samotného napájení je uveden obrázek 6.2.



Obr: 6.2: Schéma napájecích bloků AT90USB162, upraveno podle [4].

6.2 USB sběrnice

V dnešní době USB sběrnice proniká do měřící techniky a zejména přibývá jeho použití v průmyslových provozech. Nahrazuje tak starší sériové sběrnice RS232 a RS485. Proto jsem zvolil pro komunikaci mezi počítačem a mikrokontrolérem uvnitř programátoru USB sběrnici. Standard USB v dnešní době udává čtyři druhy rychlostí přenosu dat.

Přenosové rychlosti rozhraní USB:

- **Low Speed** 1.5 Mb/s (USB 1.1)
- **Full Speed** 12 Mb/s (USB 1.1)
- **High Speed** 480 Mb/s (USB 2.0)
- **Super Speed** 6 Gb/s

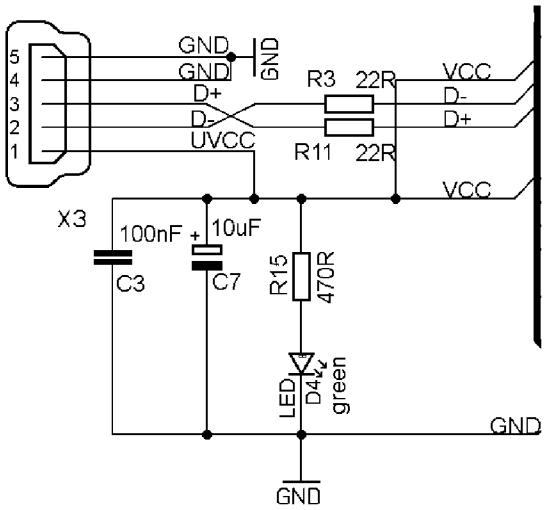
V mé diplomové práci budu používat Low Speed USB. Jedná se sice o méně používané rychlosti přenosu dat, ale v mém případě nebude datový tok nikterak veliký, proto bude rychlost okolo 1,5 Mb/s dostačující. USB sběrnice je jednomasterová a všechny aktivity vycházejí z počítače. Napájení mikrokontroléru bude vedeno z USB portu počítače, pokud je maximální odebíraný proud 100 mA (low power) nebo 500 mA (high power). Maximální délka kabelu je 5 m. V hostitelském počítači bude USB připojeno pomocí konektoru typu A a v programátoru typ MINI-B.

Specifikace USB obsahuje čtyři základní typy datových přenosů*[28]:*

- **Řídící (control) přenosy** jsou používány ke konfiguraci zařízení při jeho připojení a mohou být použity k dalším účelům
- **Hromadné (bulk) přenosy** slouží k přenosům velkého množství dat a jsou na ně kladena nejmenší omezení

- **Přerušovací (interrupt) přenosy** slouží k včasnému a spolehlivému doručení dat, nejčastěji pro asynchronní události

- **Izochronní (isochronous) přenosy** zabírají předem smluvené množství přenosového pásma a mají předem dohodnuté zpoždění. Tento druh přenosů je také nazýván proudový přenos v reálném čase (streaming real-time transfer).

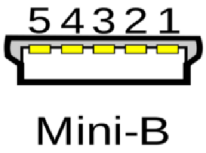


Obr. 6.3: Zapojení USB sběrnice

Zapojení USB sběrnice se skládá z konektoru X3, který obsahuje pět vývodů. Popis funkcí vývodů je uveden v tabulce 6.1 a rozložení vývodů je viditelné na obr. 6.4. Kladná i záporná linka sběrnice je chráněna rezistory R3 a R11 o hodnotách 22 Ω a jsou po ní přenášena diferenciálně vlastní data. Díky tomu má sběrnice i při vysokých rychlostech přenosu odolnost proti rušení a šumu. Pomocí vývodu UVCC je napájen mikrokontrolér přes filtrační kondenzátory. K tomuto vývodu je připojena i zelená LED dioda, která indikuje připojení USB kontoru do počítače.

6.2.1 Konektor USB

Na straně počítače se jedná o klasický konektor typů A tzv. hostitelský. V samotném programátoru, jak už jsem zmiňoval výše, bude použit typ konektoru B ve specifikaci Mini. Viz popis vývodů v tabulce 6.1 a rozložení vývodu na obrázku 6.4.



Obr. 6.4: Rozložení vývodů USB MINI B – vysvětlivky viz tabulka 6.1

Tab 6.1: Popis vývodů USB MINI B

Vývod	Popis funkce	Označení
1	+5V DC	UVCC
2	DATA-	D-
4	N/C nebo ZEM	GND
5	Zem	GND

6.3. Slot na SD kartu

SD karta z anglického Secure Digital je nástupce starší MMC karty. Jako médium je použitá cash paměť. Maximální kmitočet hodinových impulsů pro verzi 1.0 je 25 MHz. Dnes již běžně používaná specifikace v2.2 pracuje s hodinovým impulzem dvakrát větším, tedy 50 MHz. SD karty dělíme podle rychlostních tříd (class 0,2,4,6,10) a velikosti (standardní velikost, miniSD, microSD). V našem průmyslovém programátoru budeme mít slot na standardní SD kartu. SD karta podporuje komunikaci ve dvou režimech. Prvním je SPI (stejně jako u MMC karet) a druhý je pomocí protokolu SD BUS. V případě nutnosti použití microSD karty, použijeme adaptér. Na obrázku 6.4 vidíme rozložení vývodů a v tabulce 6.2 samotný popis funkce SD karty v módu SPI [26].

Obr. 6.5: Rozložení vývodů klasické SD karty

Tab. 6.2: Význam vývodů SD karty v SPI módu

Vývod	SPI mód
1	SS (Chip Select)
2	MOSI
3	GND
4	VCC (+3,3V)
5	SCK
6	GND
7	MISO
8	X
9	X

6.3.4 SPI mód

Jedním z volitelných módu SD karty je komunikační SPI mód, komunikace probíhá po SPI sběrnici. Tuto sběrnici používá většina modernějším mikrokontrolérů. Typ komunikačního módu lze vybrat jednou a to pouze během prvního příkazu reset. Jediným způsobem jak změnit komunikační mód je odpojení karty od napájení. Karta se vybírá a je aktivní přivedením log.0 na výběrový vstup SS (Chip Select). Více viz inicializace SD karty do SPI módu.

Všechna komunikace zarovnána na 8bitů (1Byte). SPI protokol se skládá ze tří hlavních částí: (Příkazy – commands, Odpovědi – responses, Bloky dat – data blocks). Celou komunikaci řídí hostitelské zařízení (mikrokontrolér). Na každý přijatý příkaz odpovídá karta response tokenem. Na každý přijatý datový blok karta odpovídá speciální data response tokenem.

U SPI módu lze vypnout ochrana kontrolním součtem (CRC). Při odeslání prvního příkazu CMD0 při inicializaci je nezbytné tento příkaz doplnit platným CRC součtem (součet je uveden v manuálu karty a je předem znám).

Inicializace SD karty do SPI módu

Inicializace SD karty začne, když v nastavení hodinového signálu zvolíme rychlost SPI sběrnice 400 kHz, což je vyžadováno pro kompatibilitu většiny SD a MCC paměťových karet. Následně musí být vysláno 74 cyklů hodinového signálu z mikrokontroléru (master). Pak kartu resetujeme příkazem CMD0 při aktivovaném SS vstupu karty a zadáme svůj klidový stav (SS při úrovni L). CRC bajt pro příkaz CMD0 a nulový argumet příkazu je 0x95. Následují příkazy CMD55 a ACMD41(Inicializace SD karty). Inicializace SD karty je dokončena, pokud je idle bit v úrovni L. Nyní se očekávají další řídící rámce. Příkazem CMD58 můžeme například zjistit, zda SD karta podporuje stejné napájecí napětí jako mikrokontrolér. Typický rozsah je jak už jsem uvedl výše 2,7V až 3,6V. Po dokončení těchto kroků můžeme hodinový signál SPI nastavit na maximální povolenou hodnotu.

Tab. 6.3: Nejdůležitější příkazy pro řízení SD karty

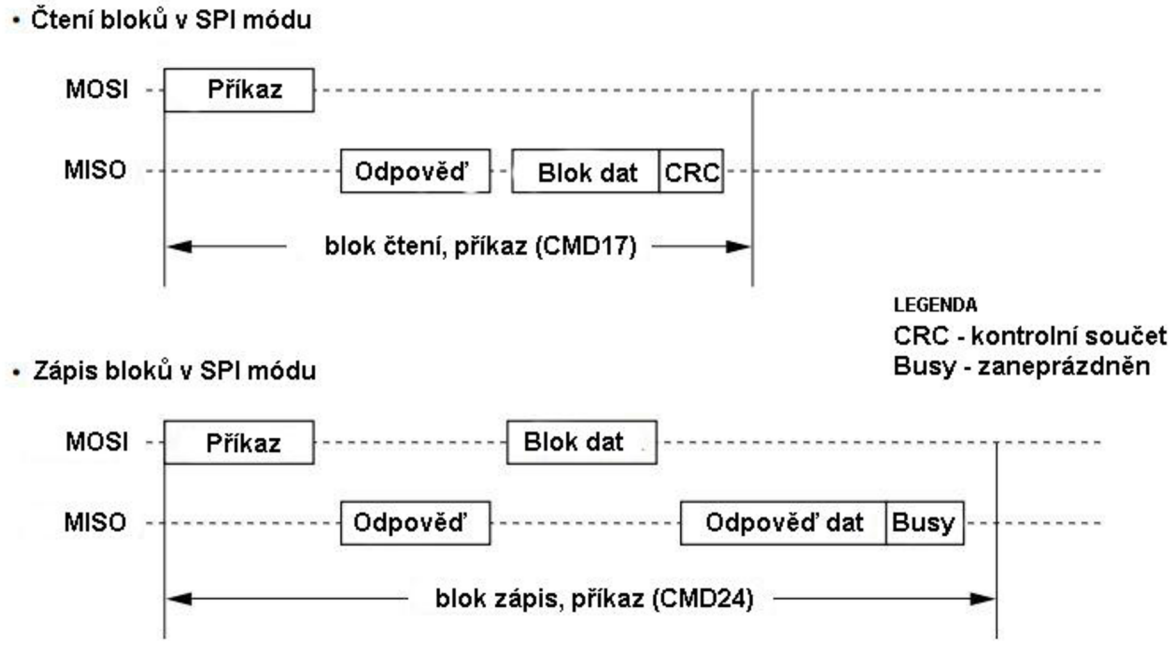
Příkaz	Argument	Typ reakce	Seznam
CMD0	Žádný	R1	Řekni kartě aby se resetovala a zadejte svůj klidový stav.
CMD16	32-bit délka bloku	R1	Vyberte délku bloku.
CMD17	32-bit adresa dat	R1	Přečtěte si jeden blok.
CMD24	32-bit adresa dat	R1	Zapište jeden blok
CMD55	Žádný	R1	Následující příkaz bude specifický pro danou aplikaci (ACMDXX).
CMD58	Žádný	R3	Přečtěte OCR (Provozní podmínky Registrace).
ACMD41	Žádný	R1	Inicializace SD karty.

Zápis a čtení dat v SPI módu

V SPI módu se zápis dat (bloků) provádí příkazem CMD24, viz tabulka 6.3. Velikost bloků můžeme definovat v CSD registru. Princip zápisu dat je patrný z obrázku 6.6, jako první se odešle příkaz pro zápis dat. Poté karta odešle odpověď, ve které jsou informace o správnosti adresy a stavu připravenosti k zápisu. Následně začne přenos datových bloků z mikrokontroléru do karty. Po dokončení přenosu celého bloku, karta odešle zpět odpověď s potvrzením o korektním

přenosu. Další vysílání dat může nastat až po skončení příznaku zaneprázdněnosti (čeká se, až je dokončen zápis dat do paměti).

Čtení dat se v SPI protokolu provádí po blocích, slouží k němu příkaz CMD17. Podobně jako u zápisu se velikost bloků definuje v registru CSD. Velikost bloků může být 512 B až 2 kB. Princip čtení dat je na obrázku 6.6. Jako první se odešle příkaz, na který karta pošle odpověď (potvrzení správnosti adresy). Pokud je adresa v pořádku, může karta zasílat datové bloky, které jsou doplněny o kontrolní součet CRC.



Obr. 6.6: Čtení a zápis dat v SPI módu (upraveno podle [26])

6.3.1 Souborový systém FAT

Souborový systém FAT je velice jednoduchý, a proto je podporován velkou škálou operačních systémů. V dnešní době evidujeme tři typy FAT souborových systémů: FAT12, FAT16, FAT32. Základním rozdílem těchto tří typů je velikost bitů z položek v aktuální struktuře FAT na disku. K dispozici je 12 bitů v záznamu pro FAT12, 16 bitů pro FAT16 a 32 bitů na vstupu FAT32 [14]. FAT dělí paměť na

sektory, bloky o paměti velké 512 bytů. Paměť se dále dělí na určité celky s různými počty sektorů [23]:

Master Boot Record

Nachází se v prvním sektoru paměti. Neobsahuje příliš mnoho informací o paměťovém zařízení. Jeho hlavním úkolem je zavedení operačního systému za pomocí zaváděcího kódu. Obsahuje čtyři tabulky rozdělení (partition table) a kontrolu signatury.

FAT Boot Record

Ve FAT Boot Recordu jsou uloženy nejdůležitější informace o souborovém systému a u pamětí s větší kapacitou uložen v sektoru určeném Master Boot Recordem.

FAT tabulka

Bezprostředně za Boot Redcordem je umístěna zvláštní datová oblast, nazývaná FAT tabulka (File allocation Table). Hlavním úkolem je popis uložení jednotlivých souborů (v tomto případě pod pojmem soubor myslíme všechny typy souborů – programy, data, apod.) na disku či disketě.

Struktura FAT tabulky je relativně složitá. Již jsme uvedli, že se jedná o oblast čistě datovou – z pohledu počítačového viru tedy není zajímavá, pokud uvažujeme o šíření viru. Virus nemůže napadnout FAT tabulku tak, aby byl v této oblasti uložen a dále se z ní šířil.

Bohužel FAT tabulka představuje ideální objekt pro destrukci, pokud se k ní virus rozhodne. Na relativně malém prostoru (několik desítek sektorů) jsou uloženy informace, jejichž zničení způsobí, že programy a data budou na počítači nepřístupná (na zařízení budou sice i nadále existovat, nicméně nebude možné určit, kde a jak jsou jednotlivé soubory uloženy).

Tabulka kořenového adresáře

Oblast paměti, která začíná v sektoru za poslední FAT tabulkou a končí v sektoru před začátkem oblasti dat. Jeho velikost je omezena – tzn., může obsahovat pouze omezený počet záznamů – souborů, na rozdíl od ostatních uživatelem definovaných adresářů, jejichž velikost je omezena pouze volným místem na pevném disku.

Data

Zbylý prostor logického disku je vyhrazen pro uživatele. Zde jsou ukládány programy, data a vytvářeny adresáře. Data souborů nemusí být uložena v clusterech hned za sebou, ale v různých částech paměti.

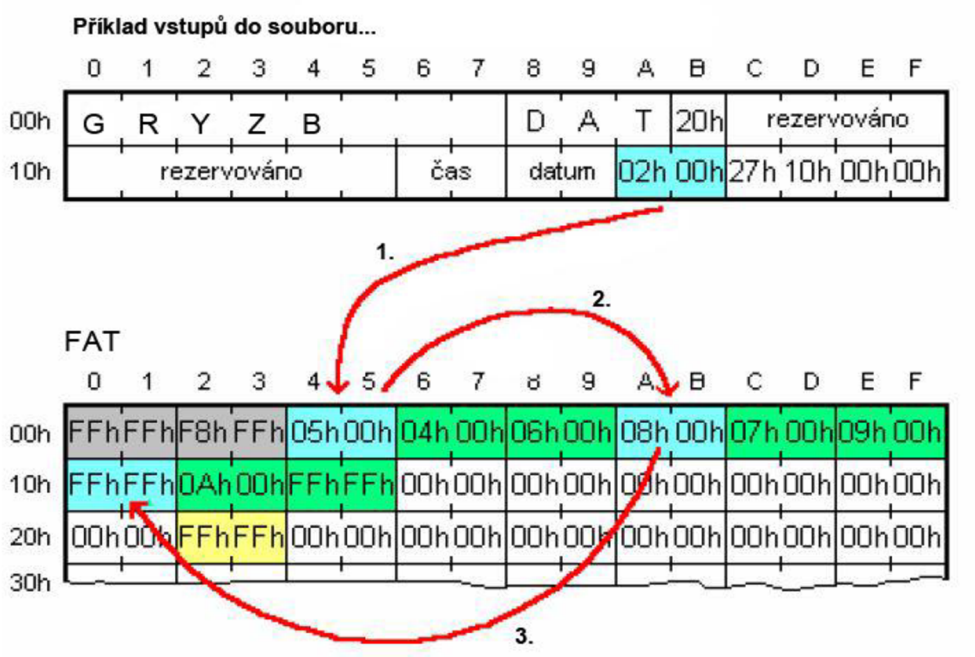
6.3.2 FAT16

Jak již s názvu File Alocation Table (FAT) vyplývá, jsou informace o umístění neboli alokací souborů u tohoto souborového systému uloženy ve zvláštní tabulce. Tabulka FAT16 pojme až 2^{16} (65556) 16ti bitových záznamů. Každý z těchto záznamů ukazuje na určitý cluster. V tabulce 6.4 je přehled jednotlivých typu clusterů. Velikost clusterů je od 0,5 kB do 32 kB. Maximální velikost diskového oddílu souborového systému FAT16 je 2 GB. V našem případě máme SD kartu o velikosti diskového pole 256 MB. Z tabulky 6.4 si můžeme povšimnout, že velikost clusteru je 4 kB.

Tab. 6.4: Přehled poměru velikosti clusterů k max. velikosti disku

Počet clusterů	Počet sektorů na cluster	Velikost clusteru	Max. velikost disku
216	1	512 B	32 MB
216	2	1 kB	64 MB
216	4	2 kB	128 MB
216	**8**	**4 kB**	**256 MB**
216	16	8 kB	512MB
216	32	16 kB	1 GB
216	64	32 kB	2 GB

FAT16 funguje tak, že vybereme z hlavní struktury soubor. Zjistíme jeho počáteční cluster a následovně zjistíme i celkovou velikost souboru. Poté přečteme první cluster, zjistíme, na kterém clusteru se nacházejí další data. Takovým způsobem pokračujeme dál, až dojdeme ke clusteru, který je relevantní jako poslední (hodnota z rozsahu FFF8h-FFFF).



Obr. 6.7: Princip čtení souborů ve FAT

Tab. 6.5: Částečný přehled clusterů a jejich význam

Kód FAT	Význam clusteru
0000h	volný cluster
0002h-FFEFh	použitý cluster, číslo udává následující cluster souboru
FFF0h-FFF6h	reservovaný cluster
FFF7h	špatný cluster
FFF8h-FFFF	použitý cluster, poslední v souboru

Výhody FAT16:

- Použití v systémech Windows a některých systémech UNIX
- Nástroje pro správu a obnovu dat
- Na objemech dat menších než 256 MB efektivní v porovnání rychlosti a uložení

Nevýhody FAT16:

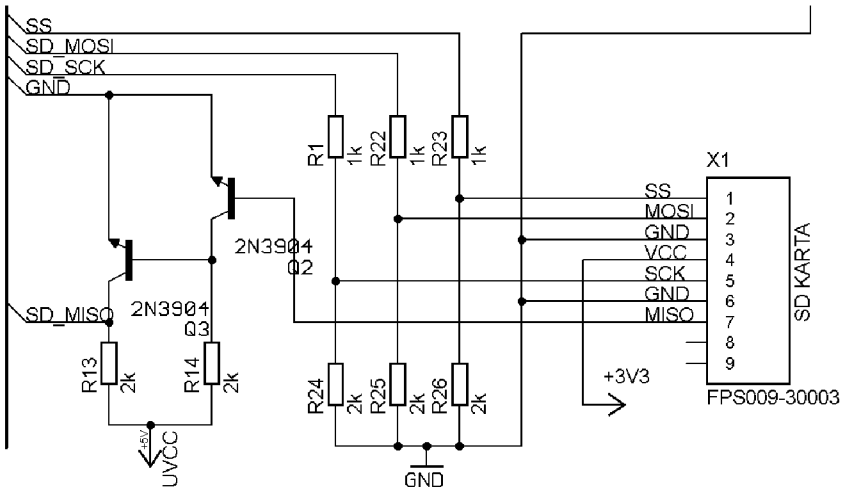
- Omezení na 65536 clustrů
- Spouštěcí sektor není zálohován
- Neexistuje žádný souborový systém pro zabezpečení systémů a kompresi souboru

6.3.3 Připojení SD karty

SD karta může pracovat s provozním napětím 2,7 V až 3,6V. V našem případě je na vývod VCC přivedeno napětí 3,3V z regulátoru uvnitř mikrokontroléru. Pro připojení SD karty s mikroprocesorem je zapotřebí čtyřech vývodů:

- hodiny (SCK)
- výběr obvodu (SS)
- data z mikroprocesoru do SD karty (MOSI)
- data z SD karty do mikroprocesoru (MISO)

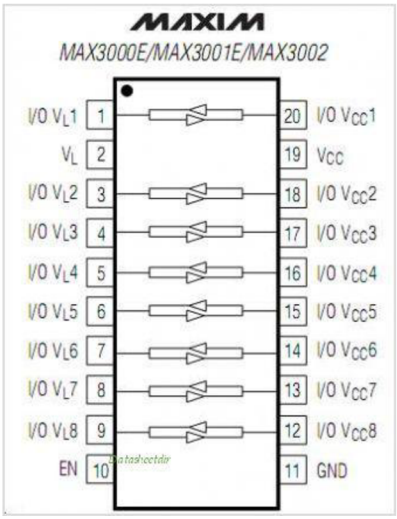
Taktéž maximální napětí na datových vstupech může být 3,6V. Proto při komunikaci s mikroprocesorem, který je napájen +5V jsou datové vstupy vedeny přes odporový dělič, viz obr. 6.8.



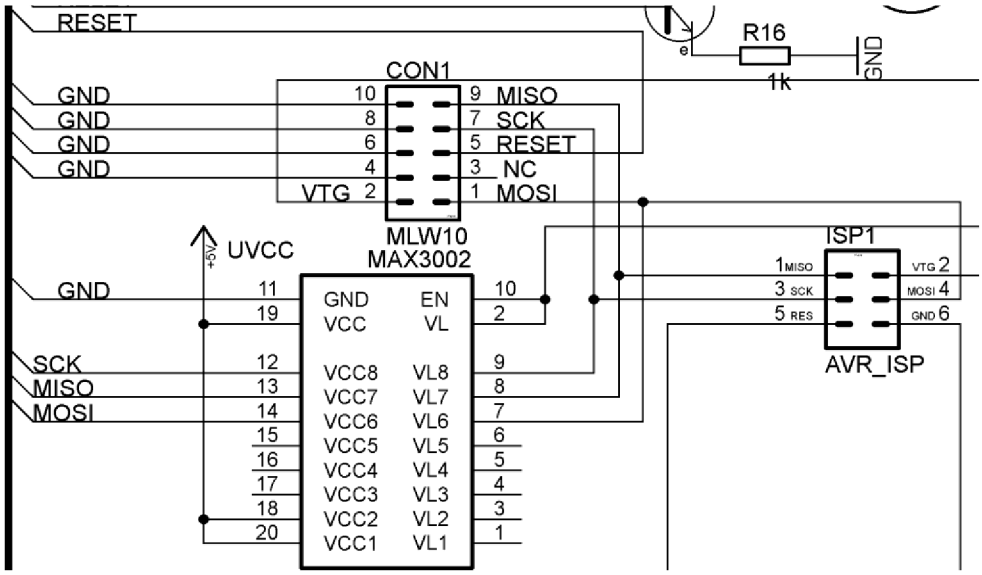
Obr. 6.8: Zapojení SD karty

6.4 MAX3002

Integrovaný obvod MAX3002 slouží pro napěťové přizpůsobení. Nesmírnou výhodou tohoto „Level Shifteru“ je jeho obousměrnost. V programátoru tedy slouží k oddělení 5V logiky z mikrokontroléru a napěťové logiky, která bude v danou chvíli na vývodu VTG. Osmi kanálový MAX3002 má na svých I/O ochrany proti ESD rušení a je schopen přenášet data až o rychlosti 20Mbit/s. Obvod také vyniká nízkou spotřebou, která je v provozním stavu menší než 10µA.



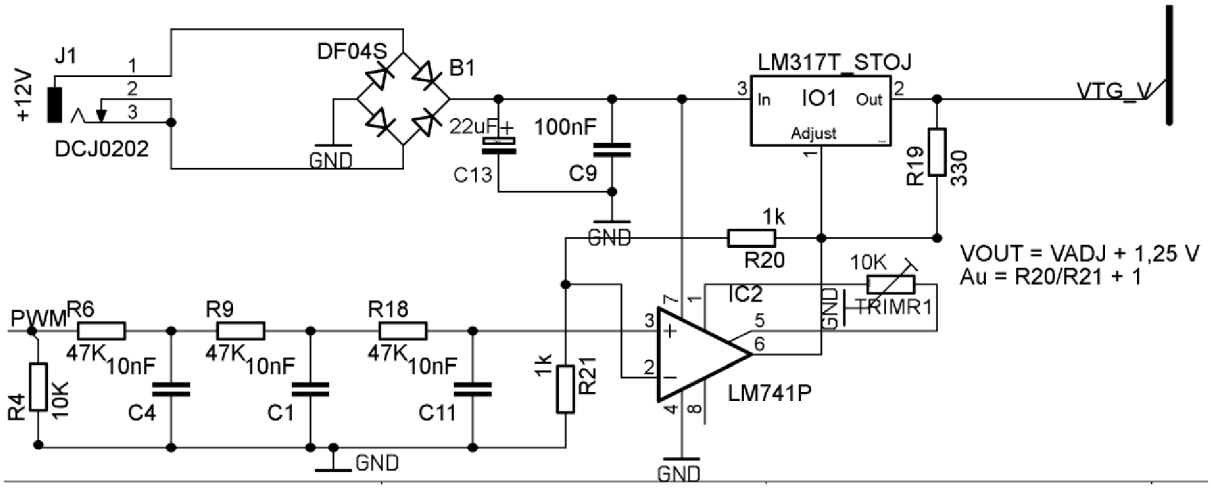
Obr. 6.9: Rozložení vývodů MAX3002 (převzato z [22])



Obr. 6.10: Zapojení MAX3002 s ISP konektorem a mikroprocesorem

6.6 Nastavování PWM signálu

Jako další požadavek průmyslového programátoru, byla možnost nastavit napěťové úrovně VTG programované aplikace, která nemá své vlastní napájení. Napěťová úroveň se bude nastavovat pomocí střídy a bude ovládána přímo z mikrokontroléru pomocí signálu PWM. Výstupní napětí VTG je možno nastavit v rozmezí 1,25 V do 11,25 V. V našem případě bude stačit napěťový rozsah od 1,25 V do 5,5 V.



Obr. 6.11: Nastavení napětí VTG pomocí PWM signálu

Pro přesnější nastavení je do obvodu přivedeno externí napájení o velikosti 12 V. Napětí 12 V jde přes diodový usměrňovač na stabilizátor LM317, který má tří vývody, přičemž žádný z těchto vývodů není připojen na zem. Integrovaný obvod se snaží na výstupu nastavit takové napětí, aby rozdíl mezi vývodem ADJUST a OUT byl právě 1,25 V. Oproti typickému zapojení LM317, kde se výstupní napětí nastavuje pomocí potenciometru, budu výstupní napájení řídit pomocí signálu PWM přímo z mikrokontroléru. Obvod je doplněn o operační zesilovač LM741P, který má možnost nastavit napěťový offset. Nastavení se provádí mezi vývody č. 1 a č. 5, kde je připojen jednoduchý odporový trimr. Zesílení operačního zesilovače určují hodnoty rezistorů R20 a R19. Zesílení se určuje podle vzorce (1).

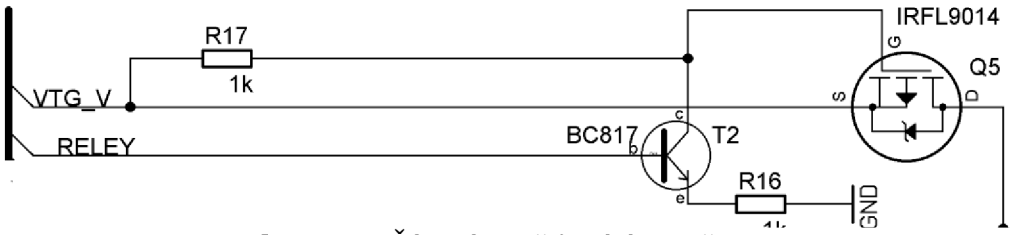
$$Au = \frac{R20}{R19} + 1 \qquad [-] \qquad (1)$$

Výstupní napětí na stabilizátoru LM741 spočítáme podle vzorce (2).

$$Vout = Uadj + 1{,}25 \qquad [V] \qquad (2)$$

Napěťový rozsah výstupního napětí bude tedy 1,25 do 11,25 V. V našem případě využije rozsah od 1,25 V do 5,5 V. Kdy převážná většina hostujících aplikací vyžaduje programovací napětí 3,3 V, 4 V nebo 5 V.

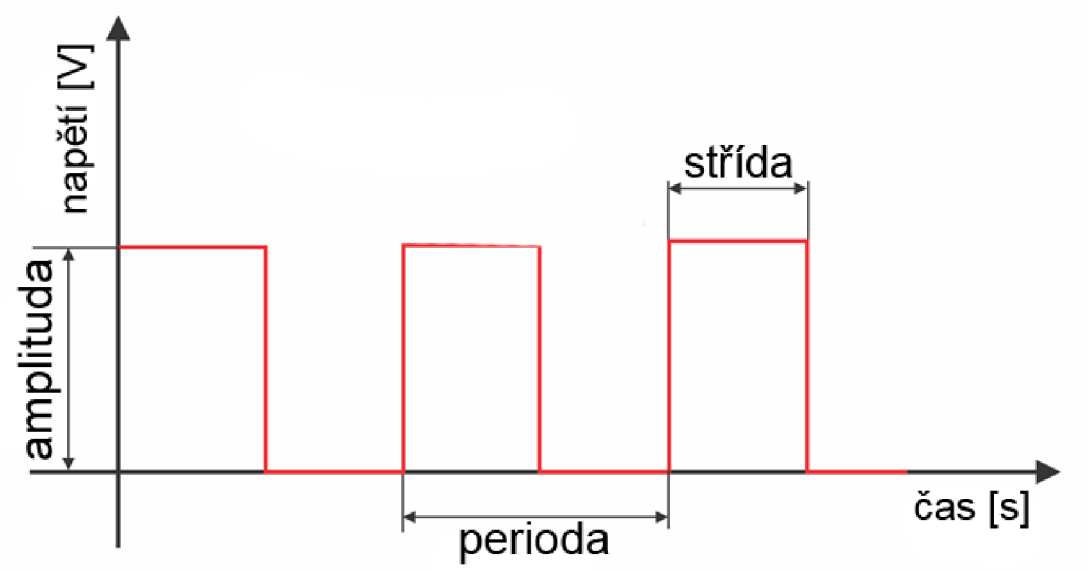
Napěťovou úroveň V_{OUT} (VTG) můžeme posílat na příslušný vývod ISP konektoru. Pokud programovaná aplikace, má své vlastní napájení bude tranzistor Q5 zavřený a tento stav bude indikován zelenou diodou D1. Tranzistor ovládáme signálem RELEY viz obr. 6.12.



Obr. 6.12: Řízení napěťové úrovně VTG

6.6.1 PWM signál

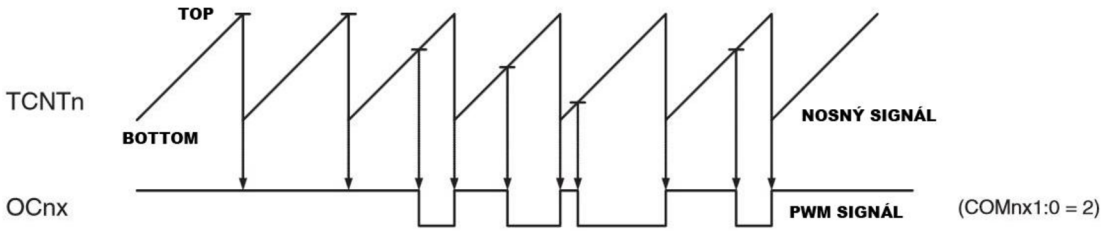
Pulsně šířková modulace, z anglického názvu Pulse Width Modulation. Jedná se o diskrétní modulace pro přenos analogového signálu pomoci dvouhodnotového signálu. Signál je přenášen pomoci střídy, kterou nastavíme pomoci mikrokontroléru. Střídu můžeme uvádět buď v %, nebo v poměru např. 1:1. Vzhledem ke svým vlastnostem je pulsně šířková modulace často využívaná pro řízení velikosti napětí nebo proudu [25].



Obr. 6.13: PWM v závislosti napětí na čase

Fast PWM

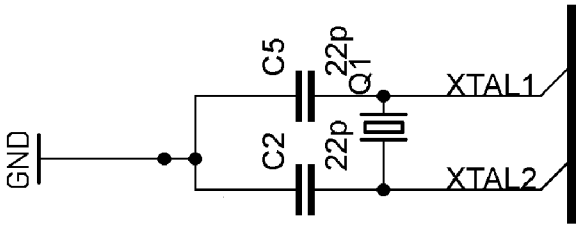
Mikrokontrolér AVR umí pracovat s několika módy pulsně šířkové modulace. Jednou z nejjednodušších variant je režim Fast PWM. Jak je vidět na obrázku 6.14 je tento režim specifický tím, že čítá ode dna až do maxima. Poté se z maxima vrátí zpět ke dnu. Jedná se o nosný signál ve tvaru trojúhelníku. Velikost maxima určuje přesnost PWM [4].



Obr. 6.14: Fast PWM modulace (AT90USB162)

6.7 Volba zdroje hodinového kmitočtu

Mikrokotrolér AT90USB162 může použít více různých způsobů generování hodinového kmitočtu. Jedná se například o interní kalibrovaný oscilátor 1/8 MHz, oscilátor s krystalem nebo externí keramický rezonátor. Musíme brát v potaz, že komunikace na USB sběrnici může probíhat, pokud mikrokontrolér pracuje minimálně na kmitočtu 12 MHz a vyšším. S vyšším kmitočtem stoupá i spotřeba mikrokontroléru. Jako vhodný kompromis je externí krystal Q1 o kmitočtu 16 MHz. Spotřeba mikrokontroléru při napájení 5 V a frekvenci krystalu 16 MHz se pohybuje okolo 10 až 25 mA. Externí krystal je blokovaný proti zemi dvěma kondenzátory o hodnotách 22 nF.



Obr. 6.15: Zapojení krystalu s blokujícími kondenzátory

6.8 Indikace Led diod

Programátor AVR mikrokontroléru obsahuje 4 indikační LED diody. Popis jejich funkcí je v tabulce 6.6.

Tab. 6.6: Indikace LED diod

Číslo LED	Barva	Popis funkce
LED D1	Zelená	Svití, pokud je připojeno na výstup ISP konektoru napětí
LED D2	Červená	Bliká, indikace chybového stavu programování
LED D3	Žlutá	Svítí, pokud jsou data vyčítána z SD karty
LED D4	Zelená	Svití, pokud je připojen USB konektor a napájí programátor

7. VÝROBA DPS

Návrh celého schématu průmyslového programátoru AVR je proveden v programovém prostředí EAGLE 5.6.0 Light. Tato starší verze je volena záměrně, jelikož u novějších verzí není plná kompatibilita se staršími verzemi. DPS je navržena podle daných pravidel a norem, které jsou k nahlédnutí v literatuře [29]. Při samotném návrhu DPS jsem si musel dát veliký pozor na správné natažení všech spojů a rozlití zemnící plochy po celé zbývající ploše. DPS je navržená jako oboustranná (TOP a BOTTOM). Místo prokovení děr desky je volena metoda protažením drátku a zapájení z obou stran DPSky.

Základní technické parametry desky:

Rozměr: 67,3mm x 62,2mm

Tloušťka materiálu: 1,5 mm

Vrtaná DPS: ANO

Nepájivá maska: ANO

Síla mědi: 17µm

Potisk: NE

Materiál DPS: FR4

Soupiska všech použitých součástek, včetně jejich popisu a pouzdra je uvedena v příloze A. Výkresy průmyslového programátoru jsou k nahlédnutí v příloze B, C, D. Fotografie osazené desky, vč. připojení USB kabelu je uvedena v příloze E. Dále je na přiloženém DVD ve složce Eagle, podsložka Library, kde jsou uvedeny všechny použité knihovny.

7.1 Osazení a oživení DPS

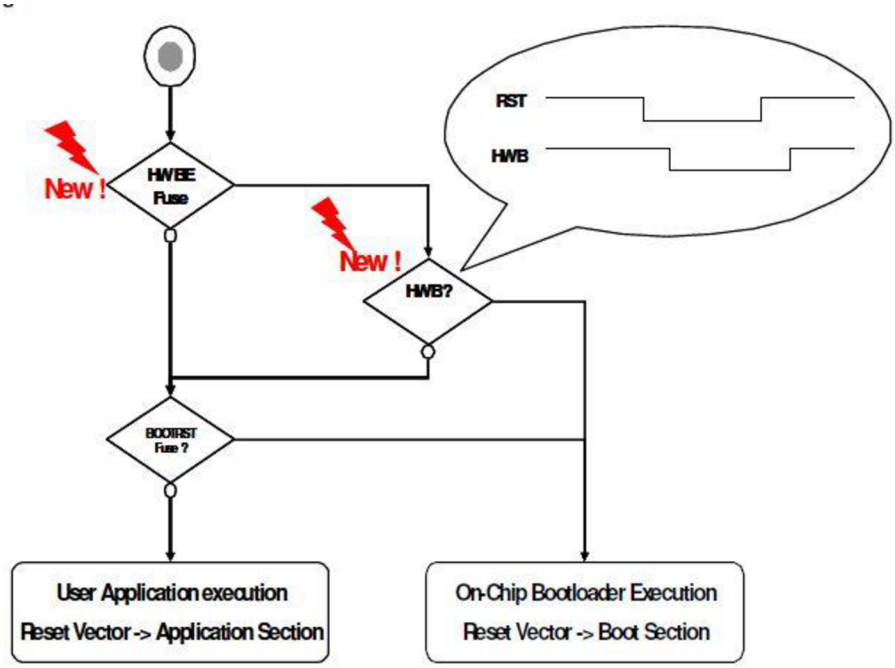
Po dokončení výrobního procesu desky průmyslového programátoru, byla deska ručně vyvrtaná podle průměrů děr. Poté byla deska nalakována přepájitelným ochranným lakem. Vyvrtané prokovy byly následně prodrátkovany a zapájeny z obou stran desky. Tím jsem docílil spojení signálů z TOP na BOTTOM vrstvu.

Poté jsem začal desku osazovat součástkami. Nejprve byla osazena napěťová část USB konektoru. Postupovalo se od SMD součástek až k těm větším. Následně byl připojen konektor a proměřeno napájecí napětí. Pomocí multimetru jsem ověřil napětí na desce, které se pohybovalo okolo 4,9 V. Následně jsem osadil mikrokontrolér a level shifter. U těchto dvou obvodů bylo pájení pečlivé, jelikož rozteč vývodů byla na minimální hranici a mohlo lehce dojít k spojení vývodů cínem. Dalšími osazenými součástkami byly veškeré SMD součástky jak na straně TOP i BOTTOM. V neposlední řadě byl osazen slot na SD kartu a konektory pro ISP programování a nahrávání firmwaru.

Kompletně osazená deska, je k nahlédnutí na fotografii v příloze G. Deska byla řádně proměřená a bylo provedeno odstranění vzniklých zkratů, které na desce vznikly při pájení a výrobě desky. Vše bylo v pořádku a mohl jsem přejít k samotnému programování mikrokontroléru AT90USB162.

8. PROGRAMOVÁNÍ MIKROKONTROLÉRU

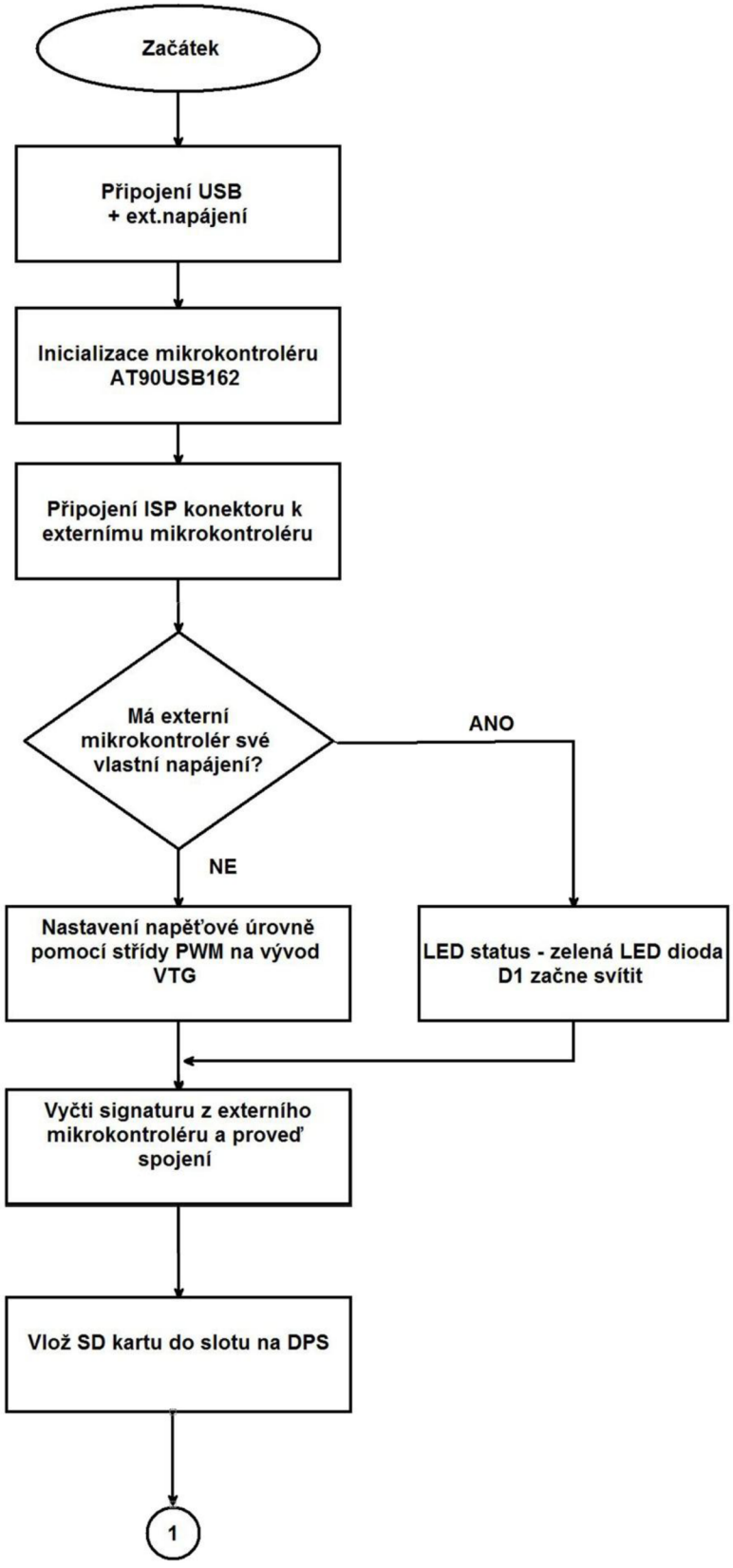
Programování mikrokontroléru AT90USB162 bylo provedeno přes programovací rozhraní ISP za pomocí deseti vývodového konektoru na DPS. Také bylo využito bootloaderu, kterým šlo zavádět program do mikrokontroléru z USB sběrnice. Popis instalace ovladačů a připojení mikrokontroléru přes USB sběrnici je v příloze F. V případě používání funkci bootloaderu, mikrokontrolér AT90USB162 musí mít dvě tlačítka: RST aktivní reset a HWB k aktivnímu vývodu HWB viz katalogový list pro AT90USB162 – bootloader [10] a obrázek 8.1.



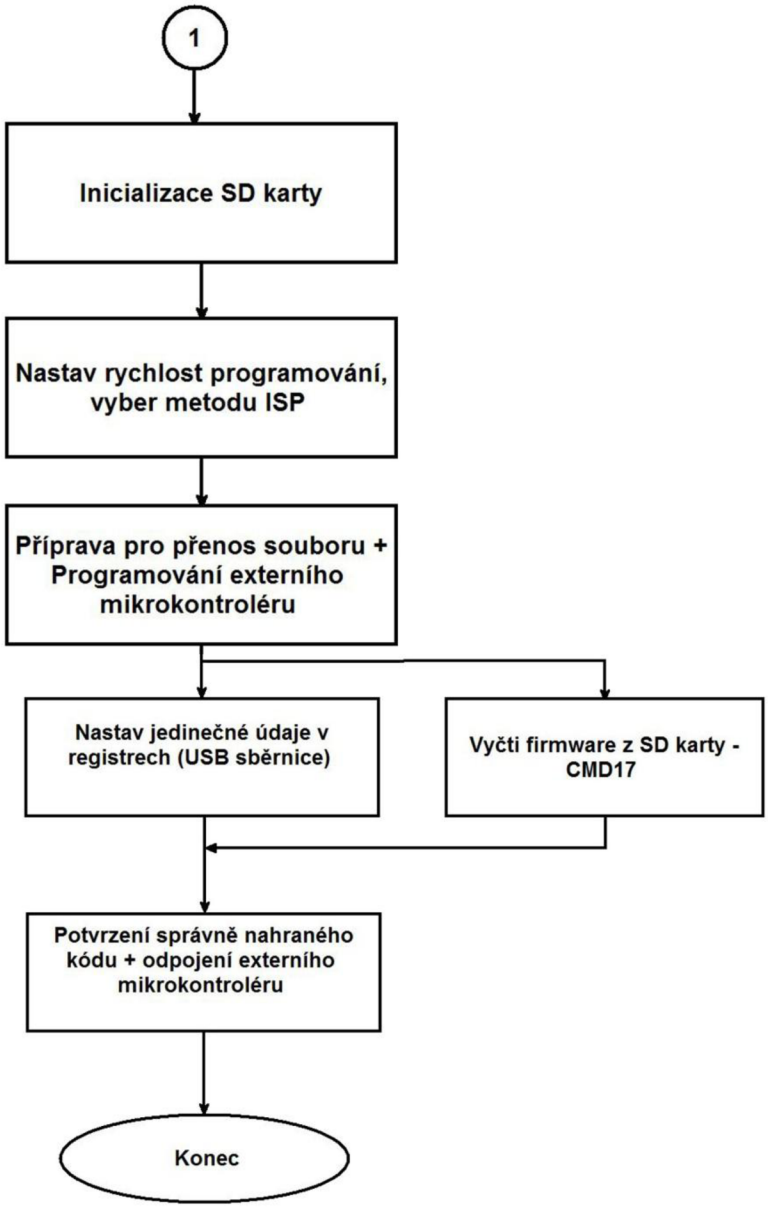
Obr. 8.1: Proces bootloaderu – převzato z [10]

Pro programování pomocí ISP konektoru jsem použil komerční programátor ASPUSB v2. Program byl vytvořen v programovém prostředí Atmel Studio 6. Byl použit programovací jazyk „AVR GCC“, což je obdoba jazyka C.

Při programování bylo využito především skript z předmětu BMPT. Pro lepší pochopení funkčnosti programátoru AVR, byl sestaven vývojový diagram, viz obrázek 8.2 a obrázek 8.3.



Obr. 8.2: Vývojový diagram 1/2



Obr. 8.3: Vývojový diagram 2/2

V následujícím textu je uveden přehled použitých funkcí a knihoven při tvorbě softwarové části průmyslového programátoru AVR. Některé volně šiřitelné knihovny, které jsou uvedeny na přiloženém DVD, jsou čerpány ze zdrojů od Rolanda Riegela [32] a od Deana Camera [31].

8.1.1 Popis funkcí programu

Hlavní program se nachází v main. c, kde je uvedena funkce main (). Zde jsou využity části kódů ze zdrojů [32], [31]. V tabulkách 8.2, 8.3 a 8.4 jsou funkce použité v knihovnách sd_raw, partiton, fat a isp.

Tab. 8.1: Funkce ve smyčce main

FUNKCE	POPIS
pwm_init()	Inicializace PWM, pomocí střídy
writeflash();	Zapíše firmware z SD karty
writeEEPROM()	Zapíše sériové číslo na registr EEPROM paměti

Tab. 8.2: Popis funkcí v knihovně partiton

FUNKCE	POPIS
struct partition_struct * partition_open (device_read_t device_read, device_read_interval_t device_read_interval, device_write_t device_write, device_write_interval_t device_write_interval, int8_t index)	Otevře index a vrací struktůru s informacemi o diskovém oddílu. Další parametry jsou fce pro čtení a zápis na diskový oddíl
struct fat_fs_struct* fat_open(struct partition_struct* partition)	Otevrírá souborový systém FAT. Parametr je partition. Úspěch log. 1, neúspěch log.0
struct fat_dir_struct* dd = fat_open_dir(fs, &directory)	Otevírá adresář.Vstupem je fs.
uint8_t partition_close (struct partition_struct * partition)	Smaže deskriptor oddílu, který vytvořila fce partition_open()

Tab. 8.3: Popis funkcí sd_raw

FUNKCE	POPIS
uint8_t sd_raw_init()	Inicializace SD karty, pokud se inicializace povede vrátí 1 v případě neúspěchu 0
uint8_t sd_raw_read (offset_t offset, uint8_t *buffer, uintptr_t length)	Z adresy offset přečte RAW (surová) data a uloží je do bufferu. Úspěch log.1, neúspch log.0
uint8_t sd_raw_write (offset_t offset, const uint8_t *buffer, uintptr_t length)	Do adresy offset zapíše RAW (surová) data, které vyšte z bufferu. Úspěch log.1, neúspch log.0

Tab. 8.4: Popis funkcí ve fat

FUNKCE	POPIS
uint8_t fat_create_file (struct fat_dir_struct * parent, const char * file, struct fat_dir_entry_struct * dir_entry)	Vytvoření nového souboru. Parametry jsou file a parent. Výstupem je dir_entry (popis nového souboru)
intptr_t fat_write_file (struct fat_file_struct * fd, const uint8_t * buffer, uintptr_t buffer_len)	Zápis dat do souboru. Vstupem je fd, délka dat je buffer_len.
uint8_t fat_delete_file (struct fat_fs_struct * fs, struct fat_dir_entry_struct * dir_entry)	Smaže adresář či přímo soubor. Vstupem je fs a fat_dir_entry_struct() - ten bude smazán
intptr_t fat_read_file (struct fat_file_struct * fd, uint8_t * buffer, uintptr_t buffer_len)	Přečte data o velikosti buffer_len ze souboru a uloží je do buffer.
void fat_close_file (struct fat_file_struct * fd)	Zavírání souboru. Vstupem je fd.
void fat_close (struct fat_fs_struct * fs)	Zavírání souborového systému FAT. Vstupem je fs.
void fat_close_dir (struct fat_dir_struct * dd)	Smaže deskriptor adresáře. Vstupem je deskriptor dd

Tab. 8.5: ISP programovací mód – popis funkcí

FUNKCE	POPIS
void init_SPI(void)	Vstup do ISP módu, inicializuje SPI pro AVR
uint8_t writeisp(uint8_t s_data)	Zápis bytu do SPI
uint8_t isp_read(void)	Ze sběrnice SPI přečte byty
void write_program_memory(uint16_t *data, uint16_t *addr, uint8_t pgmsize);	Zápíše slovo do FLASH paměti
void write_program_memory_page(uint16_t *addr);	Zápis bufferu do FLASH paměti
void chip_erase(void);	Vymaže pamět FLASH a EEPROM
void read_signature(uint8_t signatura[]);	Vyčtení signatury z mikrokontroléru
void write_data_memory(uint8_t data, uint16_t *addr);	Zápis jednoho byte do data memory
void leave_programming_mode(void);	Opuštění programovacího módu

9. ZÁVĚR

V této diplomové práci jsem se zaměřil na architekturu AVR mikrokontroléru, kde jsem zmínil pro nás velice důležité paměti dat a programu. Dále jsem popsal programovací metody, z nichž dvě byly analyzovány na rychlost programování. Tyto dvě metody (ISP a JTAG) jsem testoval na komerčních programátorech. Ověřil jsem u metody ISP, jak je závislá rychlost programování na nastavené frekvenci v programu AVR Studio. Při provádění analýz beru jako směrodatné měření na STK600 (2. analýza), protože při měření na průmyslové desce (1. analýza), musela být vytvořena další deska jako redukce pro konektory. Vinou přidané redukce docházelo ke zpomalování programování, např. délky drátků vedoucích z redukce na průmyslovou desku byly cca 20cm dlouhé. Při měření na STK600 jsem programoval FLASH paměť o velikosti 256 kB. JTAG zvládnul tuto paměť naprogramovat při frekvenci 2 MHz za 8,69 s. Metodou ISP pří nastavení frekvence z 200 kHz na 2 MHz jsem se dostal na dobu programování paměti okolo 6,6s. Obdobné výsledky jsem dostal i z první analýzy.

Z teoretických poznatků, které jsou uvedeny v kapitole 2. a 4. jsem sestavil návrh průmyslového programátoru. Návrh byl proveden v softwarovém prostředí Eagle . Základním článkem programátoru je mikrokontrolér AT90USB162, který podporuje komunikaci po USB sběrnici. V následující kapitole je popsán postup při realizaci desky plošných spojů. Kdy je deska osazena z obou stran a místo prokovů jsou obě strany prodrátkovány. Seznam použitých součástek a návrhy realizace DPS jsou uvedeny v příloze na konci práce. Oživení desky se povedlo, až poté co byly odstraněny nepatrné zkraty na DPS, které vznikly při pájení a výrobním procesu.

Softwarová část programování byla testována na externím mikrokontroléru ATmega32. Výčet dat z SD karty a následný přenos přes SPI sběrnici fungoval podle teoretických poznatků. Větší problémy nastaly při řešení nastavované úrovně PWM, kdy se signál na začátku nepatrně rozkmital, a výsledné napětí nebylo přesné. Tato část by potřebovala větší odladěnost pro její plné využití. Také by se mohlo napětí na VTG nastavovat místo PWM signálem přepínačem, ke kterému by bylo přivedeno

napětí +5 V, +3,3 V a například +4 V. Tento rozsah by měl být dostatečný pro programování externího mikrokontroléru a nastavené napětí by bylo stabilní. U části modifikace kódu, je softwarová část řešena pomocí zápisu sériového čísla na určitý registr EEPROM paměti. Toto řešení není zcela ideální, ale lepší varianta se mi nepodařila naprogramovat a odladit.

Při další modernizaci programátoru, by mohlo být zařízení umístěno do plastové krabičky. V krabičce by byl otvor pro napájení, USB konektor, ISP programovací konektor a indikační LED diody. ISP konektor pro nahrání softwaru by mohl být ukryt společně s tlačítky uvnitř zařízení. Dalším možným rozšířením by mohlo být použití kapacitně větší SD karty, na které by bylo větší množství firmwarů, se souborovým systémem FAT32.

SEZNAM POUŽITÝCH ZKRATEK A SYMBOLŮ

ALU - Arithmetic Logic Unit

AVR - Alf (Egil Bogen), Vegard (Wollan) Risc procesor

CPLD - Complex Programmable Logic Device

EEPROM - Electrically Erasable Programmable Read-Only Memory

GND - Common Ground

IAP - In-Application Programming

ISP - In-System Programming

JTAG - Joint Test Action Group

MISO - Master In Slave Out

MOSI - Master Out Slave In

PC - Personal Computer

PWM – Pulse Width Modulation

RISC - Reduced Instruction Set Computing

SCK - Serial Clock

SPI - Serial Peripheral Interface

SRAM - Static Random Access Memory

UART - Universal Asynchronous Receiver and Transmitter

USB – Universal Serial Bus

UISP - Micro In-System Programmer

VCC – IC power-supply pin

LITERATURA

Knihy, katalogové listy a webové prezentace:

[1] ATmega2560 [online]. Atmel doc2549 [cit. 12.12.2013]. Dostupný z http://www.atmel.com/images/doc2549.pdf

[2] ATmega64 [online]. 8-bit Atmel Microcontroler [cit. 27.12.2013]. Dostupný z http://www.atmel.com/images/atmel-2490-8-bit-avr-microcontroller-atmega64-l_datasheet.pdf

[3] AT90S2323[online]. Atmel doc1004 [cit. 20.12.2013]. Dostupný z http://www.atmel.com/Images/doc1004.pdf

[4] AT90USB162[online]. Atmel doc7707 [cit. 20.04.2014]. Dostupný z http://www.atmel.com/images/doc7707.pdf

[5] ATMEL CORPORATION HOMEPAGE. *Atmel* [online]. [cit. 2014-01-02]. Dostupné z: http://www.atmel.com

[6] AVR DRAGON [online]. GME czmanual-1 [cit. 21.11.2013]. Dostupný z http://www.gme.cz/img/cache/doc/752/534/prog-at-dragon-cznavod-1.pdf

[7] AVR JTAGICE mkII [online]. ATMEL doc2489 [cit. 21.11.2013]. Dostupný z http://www.atmel.com/dyn/resources/prod_documents/doc2489.pdf

[8] AVR: In-System Programming: Atmel Corporation. *Atmel.com* [online]. [cit. 2014-01-01]. Dostupné z: www.atmel.com/images/doc0943.pdf

[9] AVR STK 600 [online]. STK600 User Guide [cit. 27.12.2013]. Dostupný z www.atmel.com/Images/doc8177.pdf

[10] BOOTLOADER AT90USB[online]. Atmel doc7769 [cit. 20.04.2014]. Dostupný z http://www.atmel.com/images/doc7769.pdf

[11] CAMERA, Dean. *AVR Programming methods*. 2013.Dostupné z: http://deans-avr-tutorials.googlecode.com/svn/trunk/ProgrammingMethods/Output/ProgrammingMethods.pdf

[12] ČELEDA, Pavel. UISP - AVR In-System Programmer. [online]. [cit. 2013-11-04]. Dostupné z: http://www.hw.cz/navrh-obvodu/software/uisp-avr-in-system-programmer.html

[13] DŘÍNEK, Milan. Architektura AVR v kostce. [online]. [cit. 2014-01-01]. Dostupné z: http://avr.hw.cz/architektura/arch_avr.html

[14] FAT [online]. [cit. 2014-05-01]. Dostupné z: http://msdn.microsoft.com/en-us/windows/hardware/gg463080

[15] FLIP AVR[online]. Flip 3.4.7 [cit. 20.04.2014]. Dostupný z http://www.atmel.com/tools/flip.aspx?tab=overview

[16] FRÝZA, Tomáš. *Mikroprocesorová technika a embedded systémy: počítačová cvičení*. Brno: MJ servis s.r.o., 2011. ISBN 978-80-214-4350-1.

[17] JANČÍK, M. Programování mikrokontrolerů v systému GNU/Linux. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008.

[18] KAINKA, Burkhard. *Elektronika s podporou PC: Visual Basic v praxi*. 1. české vyd. Ostrava: HEL, 2004, 183 s. ISBN 80-861-6722-4.

[19] KLIMEŠ, C. *Realizace počítačových systémů*. Ostrava: Ostravská univerzita, 2005. Dostupné z: http://www1.osu.cz/~klimesc/public/files/RPOS1/Realizace%20pocitacovych%20sys temu.pdf

[20] MANN, Burkhard. *C pro mikrokontroléry: ANSI-C, kompilátory C, spojovací programy - linkery, práce s ATMEL AVR a MSC-51, příklady programování v jazyce C, nástroje pro programování, tipy a triky*. Vyd. 1. Praha: BEN, 2003, 279 s. ISBN 80-730-0077-6.

[21] MATOUŠEK, David. *Práce s mikrokontroléry ATMEL AVR AT90S*. 2. vyd. Praha: BEN, 2006. ISBN 80-730-0209-4.

[22] MAX3002[online]. MAX3000E-MAX3012 [cit. 20.04.2014]. Dostupný z http://datasheets.maximintegrated.com/en/ds/MAX3000E-MAX3012.pdf

[23] PEVNÝ DISK. [online]. [cit. 2014-05-01]. Dostupné z: http://www1.osu.cz/home/matejka/soft/data/harddisk.htm

[24] Programujte.com. [online]. [cit. 2013-11-01]. Dostupné z: www.programujte.com

[25] PWM [online]. [cit. 2014-04-09]. Dostupné z: http://arduino.cc/en/Tutorial/PWM

[26] SDcard [online]. HITACHI [cit. 2014-05-01]. Dostupné z: http://www.configsys.com.hk/images/THL/PDF/mmc.pdf

[27] STEVEN F. BARRETT, Steven F.Daniel J a Daniel J. PACK. *Atmel AVR microcontroller primer: programming and interfacing*. San Rafael, Calif.: Morgan. ISBN 15-982-9541-1.

[28] USB.org [online]. [cit. 2014-05-01]. Dostupné z:www.usb.com

[29] VRBA, K.; HERMAN, I.; KUBÁNEK, D. Konstrukce elektronických zařízení. Skripta FEKT VUT v Brně. Brno, 2005.

[30] ZMRZLÝ, Simeon. *Mikroprocesorová technika*. 1. vyd. Brno: Vysoké učení technické, 1996. ISBN 80-214-0799-9.

Knihovny zdrojových kódů:

[31] LUFA[online]. Dean Camera - Four Walled Cubicle [cit. 20.05.2014]. Dostupný z http://www.fourwalledcubicle.com/LUFA.php

[32] MMC/SD/SDHC [online]. Roland Riegel - card library [cit. 20.05.2014]. Dostupný z http://www.roland-riegel.de/sd-reader/

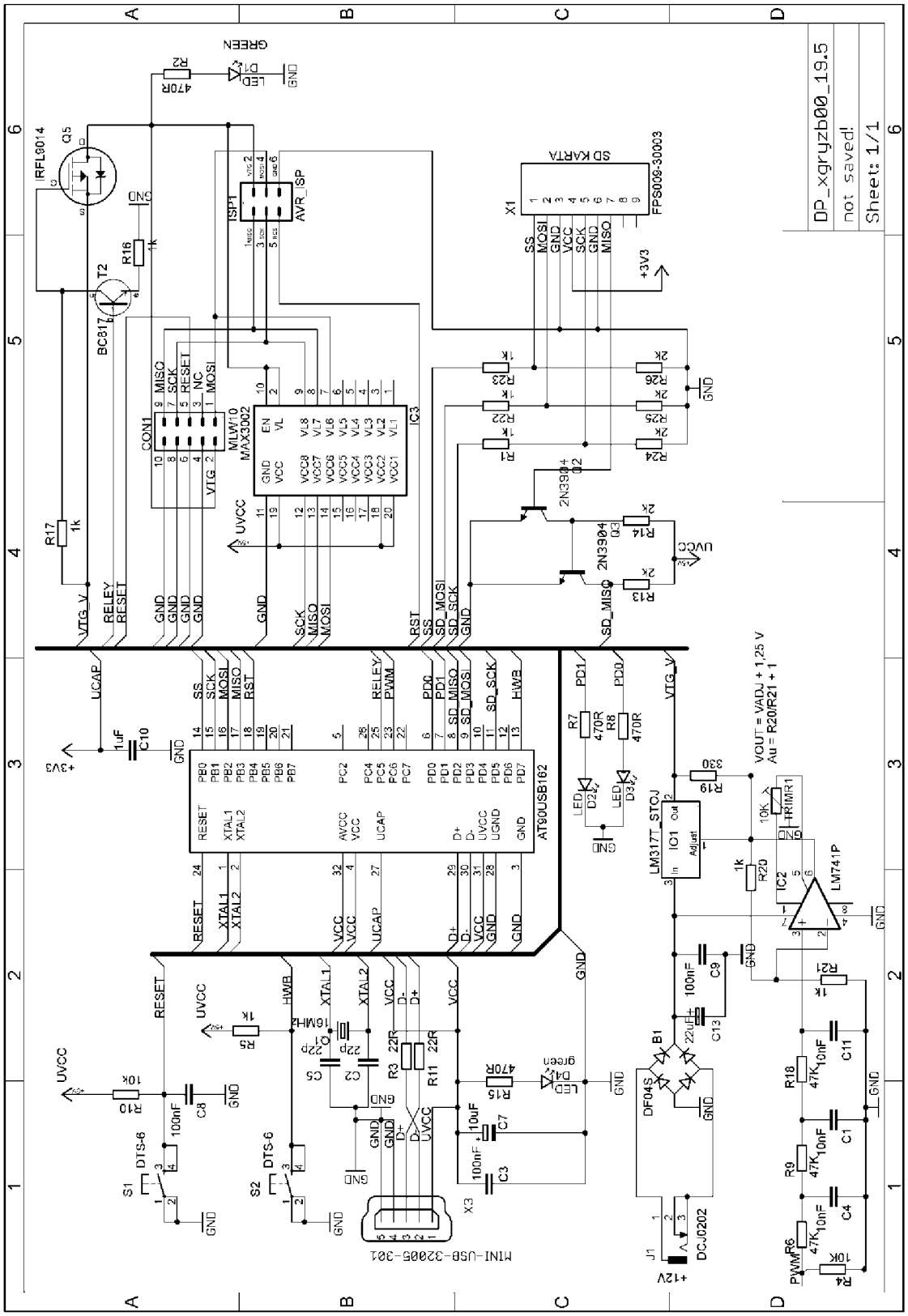
SEZNAM PŘÍLOH

A SOUPISKA SOUČÁSTEK

Tab. A.1: Soupiska součástek

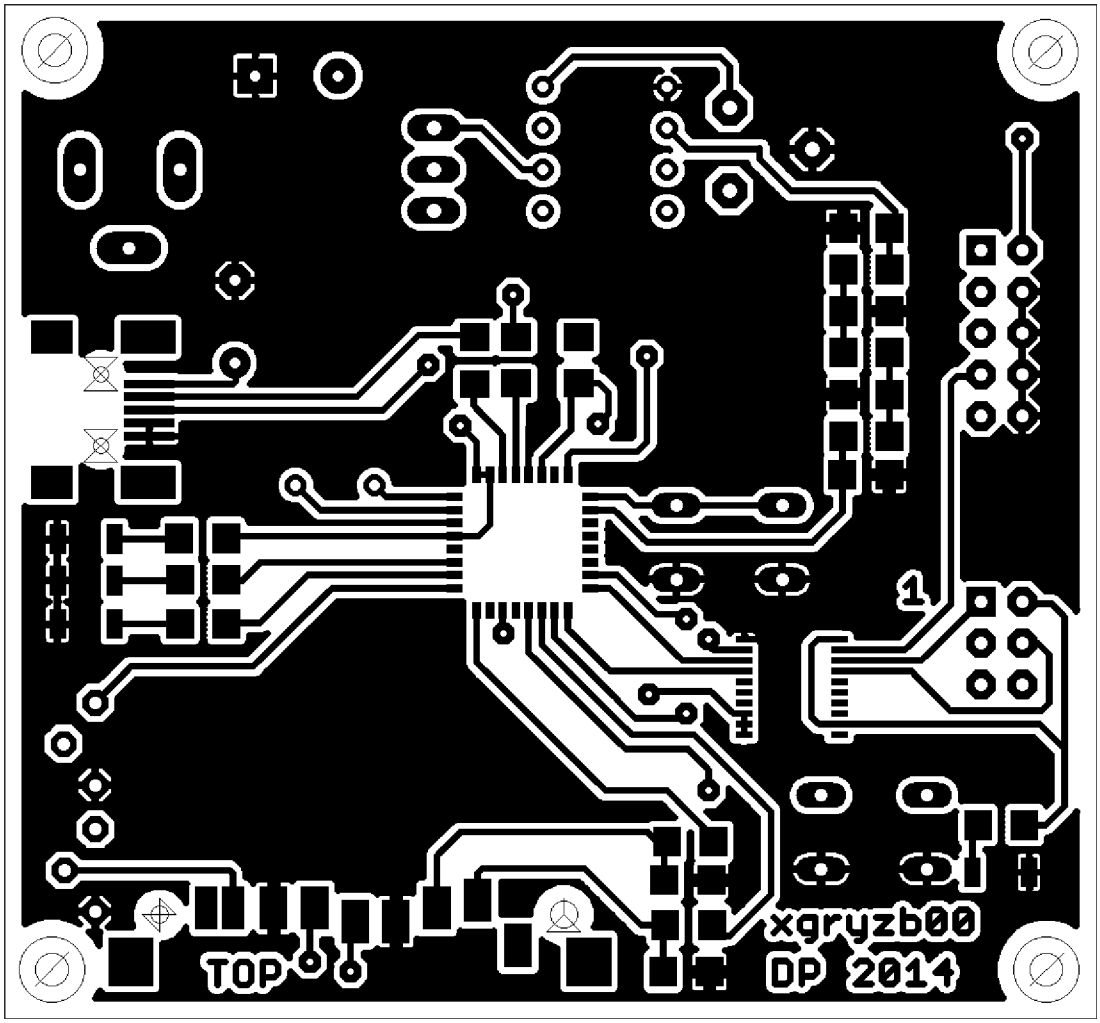
POČET	OZNAČENÍ	HODNOTA	POUZDRO	NÁZEV
2	R3, R11	22 Ω	R1206	REZISTOR
2	R4, R10	10 kΩ	R1206	REZISTOR
1	R19	330 Ω	R1206	REZISTOR
4	R2, R7, R8, R15	470 Ω	R1206	REZISTOR
3	R6, R9, R18	47 kΩ	R1206	REZISTOR
5	R13, R14, R24, R25, R26	2 kΩ	R1206	REZISTOR
8	R1, R5, R16, R17, R20, R21, R22, R23	1 kΩ	R1206	REZISTOR
2	C2, C5	22 pF	C1206	KONDENZÁTOR
3	C3, C8, C9	100 nF	C1206	KONDENZÁTOR
1	C7	2,2 uF	C1206	KONDENZÁTOR
1	C10	1 uF	C1206	KONDENZÁTOR
3	C1, C4, C11	10 nF	C1206	KONDENZÁTOR
1	C13	22 uF/25 V	5x11 RM2 BULK	KONDENZÁTOR
1	C7	10 uF	5x11 RM2 BULK	KONDENZÁTOR
2	D1, D4	GREEN	SMD 1206	LED DIODA
1	D3	RED	SMD 1206	LED DIODA
1	D2	YELLOW	SMD 1206	LED DIODA
1	TRIMR1	10 kΩ	PT6V	TRIMR
1	Q1	16 MHz	HC49U-V	KRYSTAL
1	AT90USB162		VQFP32	MIKROPROCESOR
1	X1		104B-TAA0-R	SLOT NA SD KARTU
2	Q2, Q3	2N3904	TO92	TRANZISTOR
1	IC3		TSSOP20	MAX3002
1	ISP1		MLW06G	KONEKTOR
1	CON1		MLW10G	KONEKTOR
1	T2	BC817	SOT-23	TRANZISTOR
1	Q5	IRFL9014	SOT-223	TRANZISTOR
2	S1, S2		DTS-6	TLAČITKO
1	X3	USB MINI-B	32005-301	KONEKTOR
1	J1	DCJ0202	DCJ0202	KONEKTOR
1	B1	DF04S	DFS	USMĚRŇOVAČ
1	IO1	LM317T	TO-220S	STABILIZÁTOR
1	IC2	LM741P	DIL08	OZ

B KOMPETNÍ SCHÉMA ZAPOJENÍ

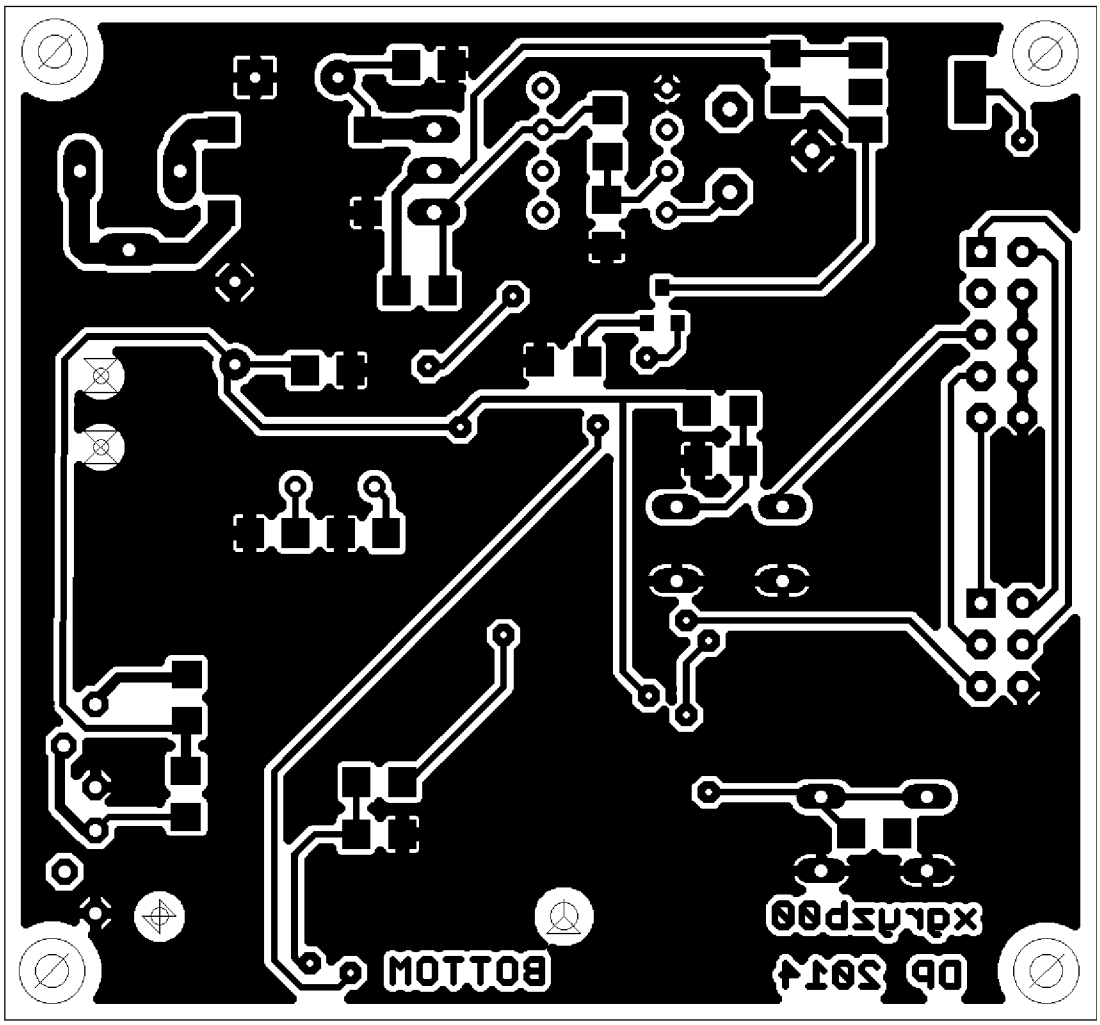


Obr. B.1: Kompletní schéma zapojení

C DPS PRŮMYSLOVÉHO PROGRAMÁTORU

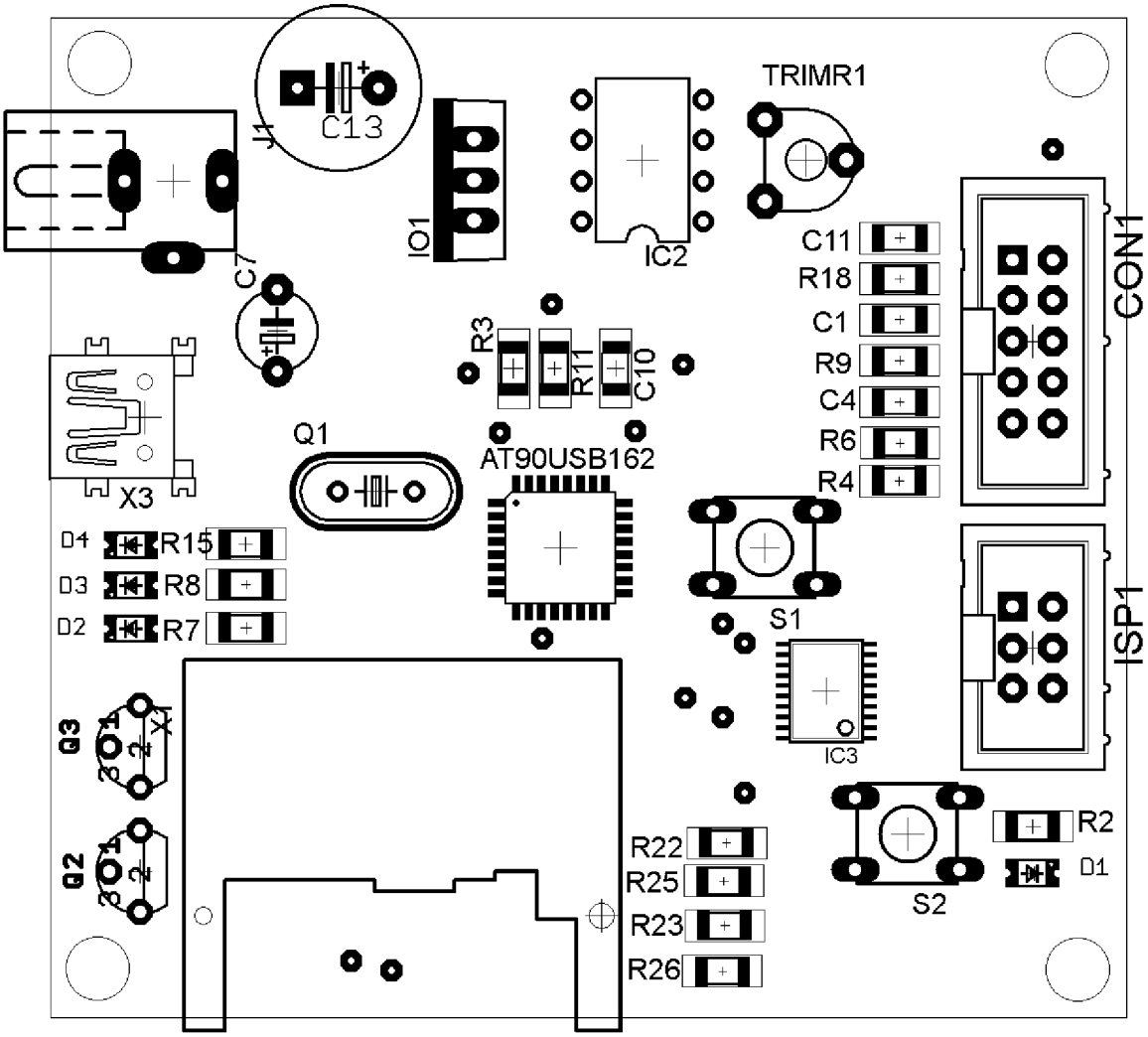


Obr. C.1: DPS průmyslového programátoru – strana TOP

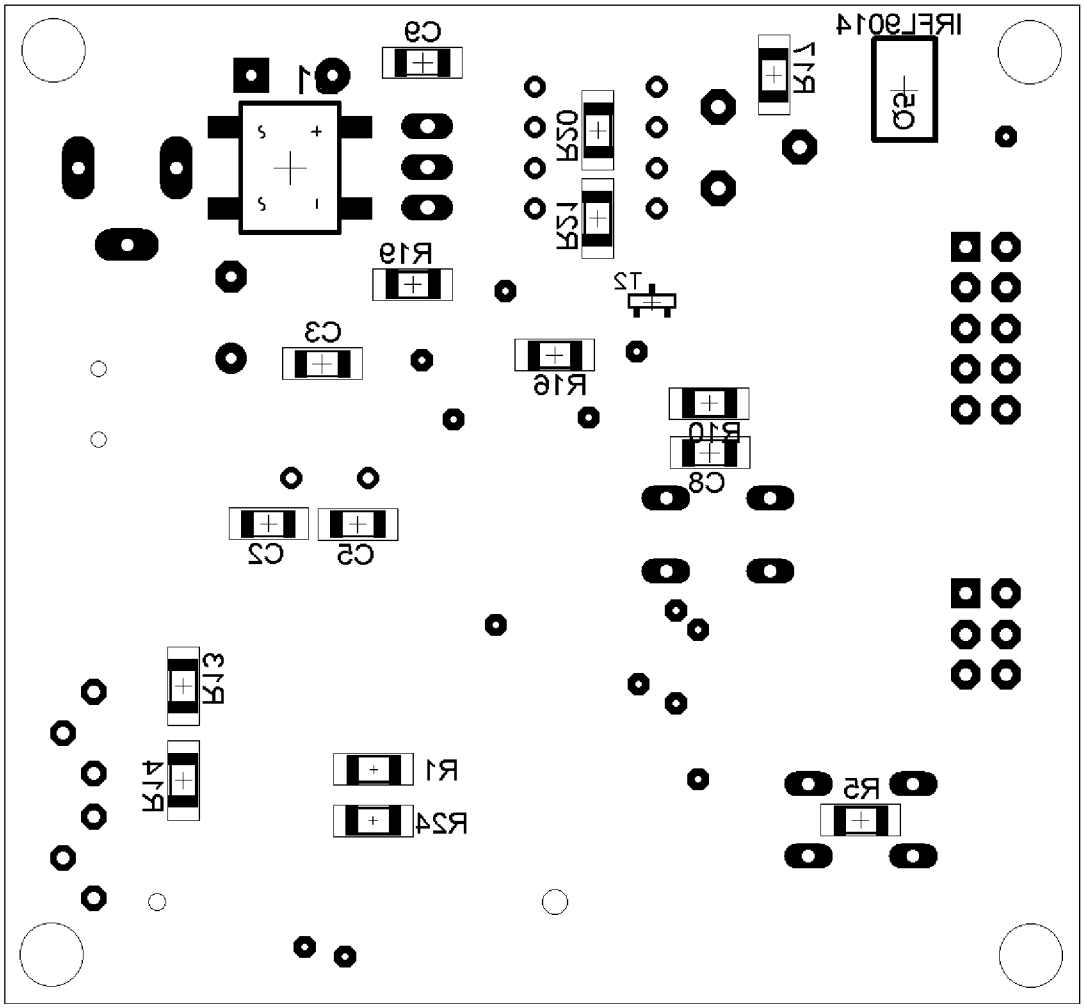


Obr. C.2: DPS průmyslového programátoru – strana BOTTOM

D ROZLOŽENÍ SOUČASTEK NA DPS

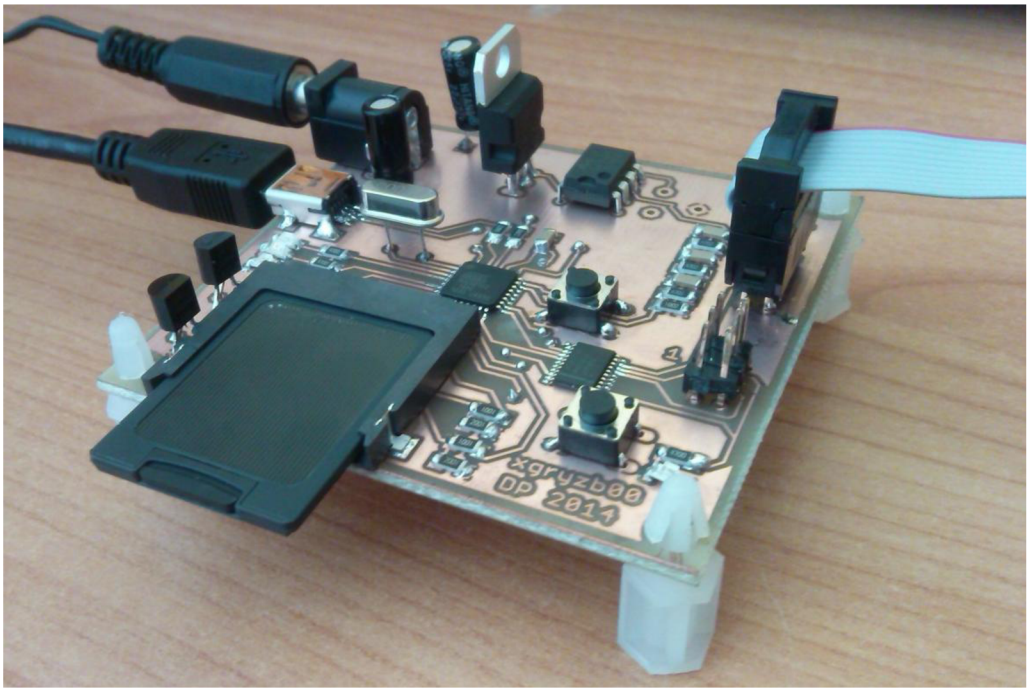


Obr. D.1: Rozložení součástek na DPS – strana TOP



Obr. D.2: Rozložení součástek na DPS – strana BOTTOM

E FOTOGRAFIE HOTOVÉHO PROGRAMÁTORU

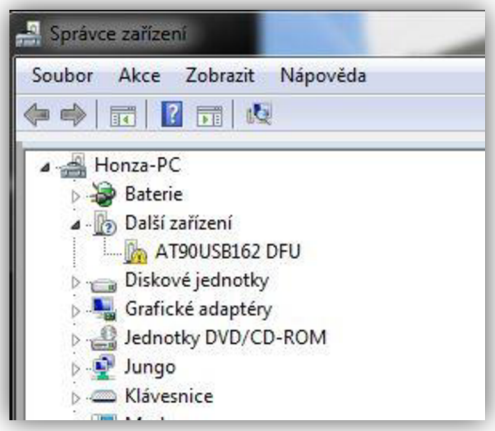


Obr. E.1: Fotografie hotového průmyslového programátoru

F INSTALACE OVLADAČE PRO PC KE KOMUNIKACI S USB

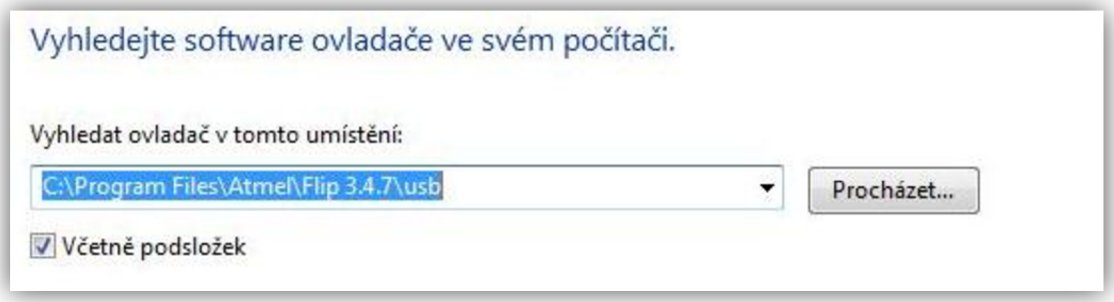
F.1 Postup pro aktualizace ovladače na USB portu PC

Ve Windows 7, na pracovní ploše počítače, klikneme na ikonku Tento počítač pravým tlačítkem myši a dáme volbu Spravovat (spustíme ji jako administrátor). Poté klikneme na Správce zařízení, kde vidíme na obrázku F.1 AT90USB162 DFU. U tohoto zařízení je výstražný vykřičník. Klikneme na něj pravým tlačítkem myši a dáme Vlastnosti → Aktualizovat ovladač.



Obr. F.1: Správce zařízení

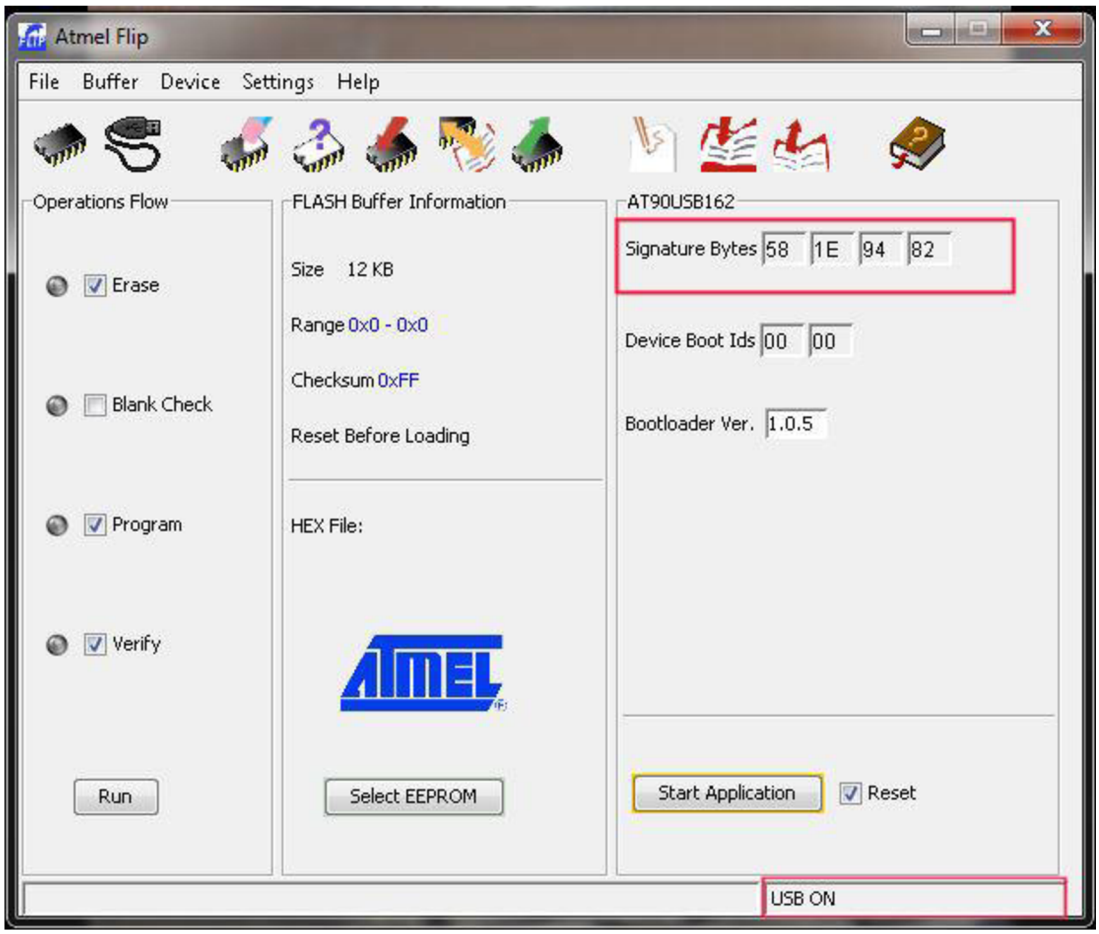
Z další nabídky zvolíme, že vybereme software pro ovladač ve svém počítači. Do složky dříve nainstalovaného programu FLIP 3.4.7 nakopírujeme nový ovladač pro naše zařízení ze zdroje [15]. Poté tuto cestu vybereme jak je vidět na obrázku F.2.



Obr. F.2: Cesta ke správnému vyhledání ovladače

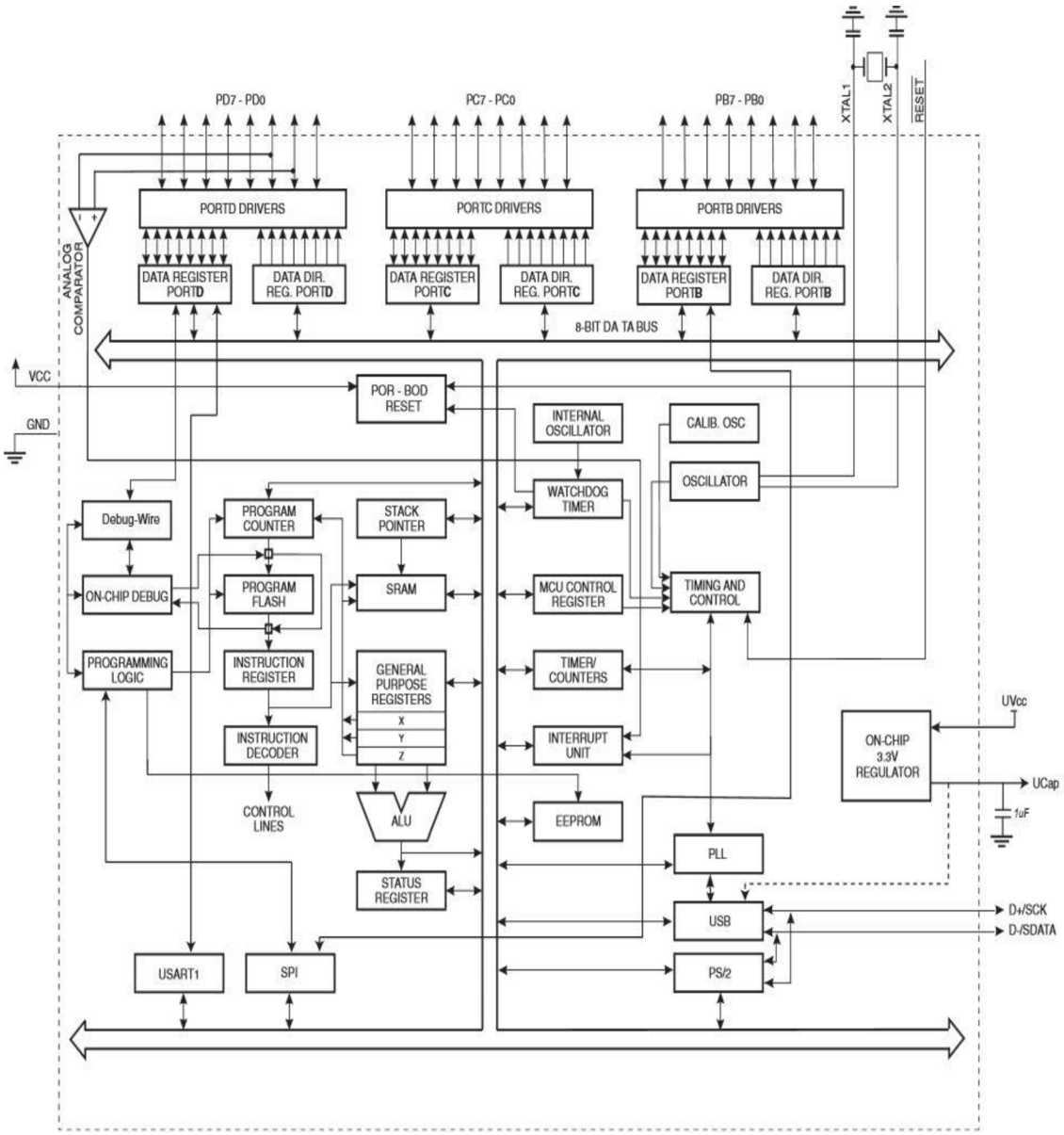
Po správné aktualizaci USB ovladače se nám vypíše hláška: Instalace ovladače zařízení byla dokončena (AT90USB162).

Spustíme nainstalovaný program FLIP 3.4.7, kde můžeme otestovat spojení USB pomocí bootloaderu. V nabídce Device Selection vybereme náš mikrokotrolér AT90USB162 a potvrdíme OK. Poté klávesovou zkratkou CTRL+U připojíme USB, správnost spojení můžeme vidět na obrázku F.3 vlevo dole vypsáním USB ON. Dále si můžeme s katalogovým listem našeho mikrokontroléru zkontrolovat signaturu.



Obr. F.3: Programové prostředí FLIP 3.4.7

G BLOKOVÉ SCHÉMA AT90USB162



Obr. G.1: Blokové schéma AT90USB162 – převzato z [4].

H OBSAH PŘILOŽENÉHO DVD

- Elektronická verze diplomové práce ve formátu PDF
- Všechny výkresy DPS navržené v programovém prostředí Eagle
- Všechny knihovny použité při návrhu DPS v Eaglu
- Zdrojové kódy navržené knihovny průmyslového programátoru
- Fotografie průmyslového programátoru a všechny vytvořené obrázky