Jihočeská univerzita v Českých Budějovicích

Pedagogická fakulta – Katedra fyziky

Diplomová práce

České Budějovice 2014 **Bc. Zdeněk Jůza**

Jihočeská univerzita v Českých Budějovicích

Pedagogická fakulta – Katedra fyziky

Měření signálů na biologických objektech

Diplomová práce

Vedoucí práce: Ing. Michal Šerý

Autor: Bc. Zdeněk Jůza

Anotace

Diplomová práce s názvem *Měření signálů na biologických objektech* zpracovává některé způsoby snímání signálů na biologických objektech: optická metoda měření EKG na hmyzu, termografická metoda měření EKG na hmyzu, EKG, EEG, EPG. V práci je zahrnuta problematika měření malých napětí elektrodami v elektrolytu. Také je uvedena konstrukce EKG, kterým bylo provedeno měření na lidech, hmyzu a jsou zde přiloženy dva záznamy z měření. Také je uvedena konstrukce dataloggeru s ATmega pro snímání EKG.

Práce se zabývá historií mikroprocesorů, základní strukturou mikroprocesorů rodiny AVR, dělením mikroprocesorů AVR atmel a popisem dvou vývojových prostředí AVR Studio a Bascom. Je zde uvedena syntaxe jazyka Bascom.

Abstract

Master thesis with the name Measuring of the signals on the biological objects concerns with some ways of monitoring the signals on the biological objects: optical method of measuring EKG on the insects, thermografical method of measuring EKG on insects, EKG, EEG, EPG. In the thesis is mentioned also the problematics of measuring the small voltages with the electrodes in the electrolyte. In the thesis is also the construction of the EKG, which was used for measurement on people, insect and there are also added two measurement reports. There is also shown the construction of the datalogger ATmega for the EKG recording.

The thesis is concerning with the history of microprocessors, basic structure of the microprocessors of the AVR family, the division of the microprocessors AVR atmel and description of the two development workspaces AVR studio and Bascom. In the thesis is also shown the syntax of the Bascom programming language.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své diplomové práce, a to v nezkrácené podobě archivované Pedagogickou fakultou, elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

Datum. 19.5.2014 Podpis studenta

Poděkování

Touto formou děkuji svému konzultantovi p. Ing. Michalu Šerému, za cenné rady a připomínky při zpracování mé práce *Měření signálů na biologických objektech*.

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH
Fakulta pedagogická
Akademický rok: **2011/2012**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Zdeněk JŮZA**
Osobní číslo:	**P110380**
Studijní program:	**N7503 Učitelství pro základní školy**
Studijní obory:	**společný základ - navazující-komb.**
	Učitelství fyziky pro 2. stupeň základních škol
	Učitelství technické výchovy pro 2. stupeň základních škol
Název tématu:	**Měření signálů na biologických objektech**
Zadávající katedra:	**Katedra aplikované fyziky a techniky**

Zásady pro vypracování:

1. historie programovatelných obvodů
2. základní struktura mikrokontrolerů AVR
3. dělení mikrokontrolérů (ATmega 8,16,32 ..., ATtiny ...)
4. SW vývojová prostředí AVR Studio, prostředí Bascom a spojení s HW vývojovým kitem
5. konstrukce měřicích přípravků pro snímání biosignálů
6. připojení přípravků k PC
7. provést testová měření a vyhodnotit je
8. zhodnocení dosažených výsledků

Rozsah grafických prací: **podle potřeby**

Rozsah pracovní zprávy: **cca 60-80 stran**

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

— **Váňa, V.: Mikrokontroléry ATMEL AVR - popis procesoru a instrukční soubor.**

— **Matoušek, D.: Práce s mikrokontroléry Atmel AVR.**

— **Šubrt, V.: Mikrokontroléry Atmel AVR - vývojové prostředí.**

— **Váňa, V.: Mikrokontroléry Atmel AVR - Bascom.**

— **Váňa, V.: Mikrokontroléry Atmel AVR - Assembler.**

— **www.atmel.com (AVRStudio, datasheet)**

— **www.hw.cz**

— **www.asix.cz**

Vedoucí diplomové práce: **Ing. Michal Šerý**
Katedra aplikované fyziky a techniky

Datum zadání diplomové práce: **27. dubna 2012**

Termín odevzdání diplomové práce: **30. dubna 2013**

Mgr. Michal Vančura, Ph.D.
děkan

L.S.

prof. RNDr. Petr Špatenka, CSc.
vedoucí katedry

V Českých Budějovicích dne 27. dubna 2012

1 Cíle

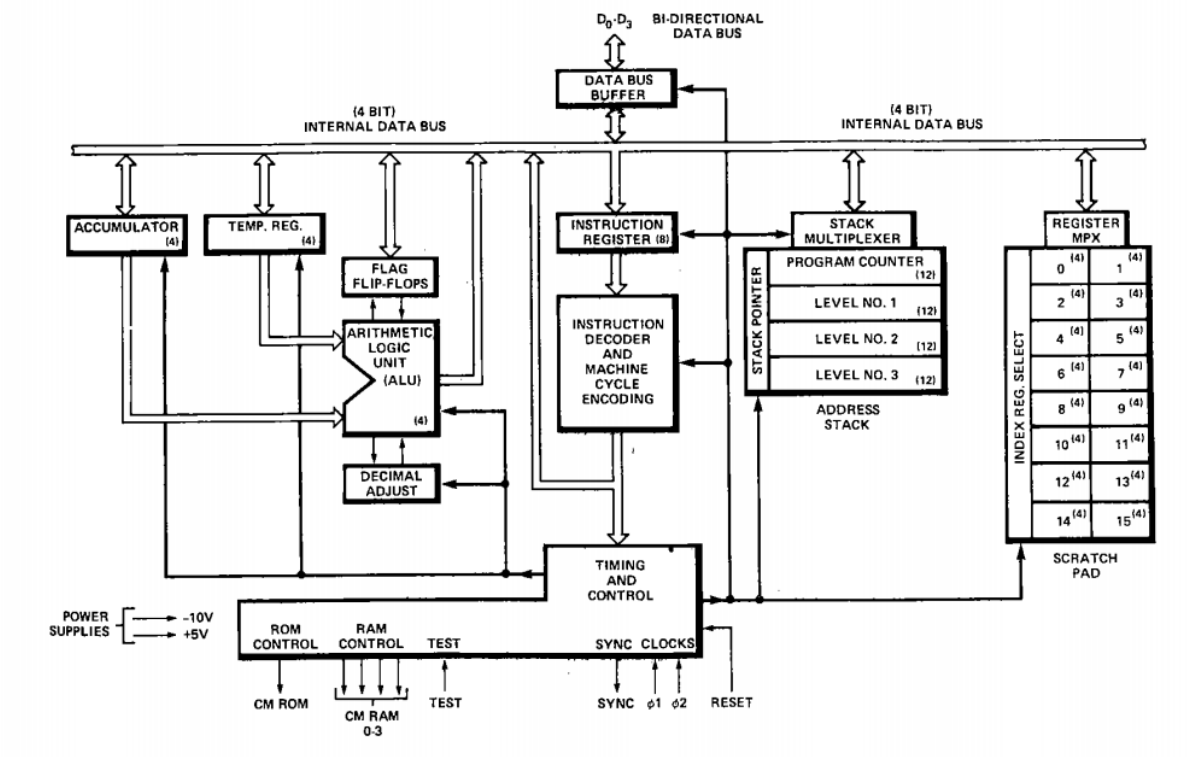
Cíle této práce jsou dva jeden teoretický, druhý praktický.

Teoretickým cílem je zmapovat historii mikroprocesorů, základní strukturu jednočipových mikroprocesorů AVR, dělení mikroprocesorů AVR, softwarová vývojová prostředí AVR Studio, prostředí Bascom a spojení hardwarovým vývojovým kitem. Konstrukce měřících přípravků pro snímání biosignálů.

Praktickým cílem je zkonstruovat měřící sondu pro měření EKG na lidech a na hmyzu. K tomu je nutný elektrický návrh sondy. Dále bude nutné popsat měřící metodu a navrhnout DPS. Nakonec naměřit data a ta vyhodnotit.

2 Teoretický rozbor

2.1 Historie programovatelných obvodů [1], [2], [3], [4], [5], [6]



Obr.č.1 Blokové vnitřní uspořádání mikroprocesoru INTEL 4004
Převzato a upraveno dle [6],

První programovatelné obvody se na trhu objevily v první polovině 60. let. Byly to obvody s malou hustotou integrace (SSI) (Small Scale Integration) a střední hustotou integrace (MSI) (Middle Scale Integration) [2].

Díky technologiím s vysokou hustotou integrace (LSI) (Large Scale Integration byl v první polovině 70. let uveden do výroby první mikroprocesor.

1971 - Intel 4004 - 4-bitový [1] vyobrazený na Obr.č.1

1972 - Intel 8008 - 8-bitový

1974 - Intel 8080 - 8-bitový - stal se základem prvních 8bitových osobních počítačů

1981 - Intel 8088 – 16-bitový s 8-bitovou sběrnicí, byl použit v prvním IBM PC

1982 - Intel 80286 – 16-bitový mikroprocesor

1985 - Intel 80386 – 32-bitový (275 000 tranzistorů).

1989 - Intel 80486 – 32-bitový (1,8 milionu tranzistorů) SX a DX verze. Verze DX obsahovala matematický koprocesor.

1993 - Intel Pentium – 32-bitový (3,3 milionu tranzistorů).

1997 - Intel Pentium II – 32-bitový (7,5 milionu tranzistorů).

1999 - Intel Pentium III - 32-bitový (9,5 milionu tranzistorů),

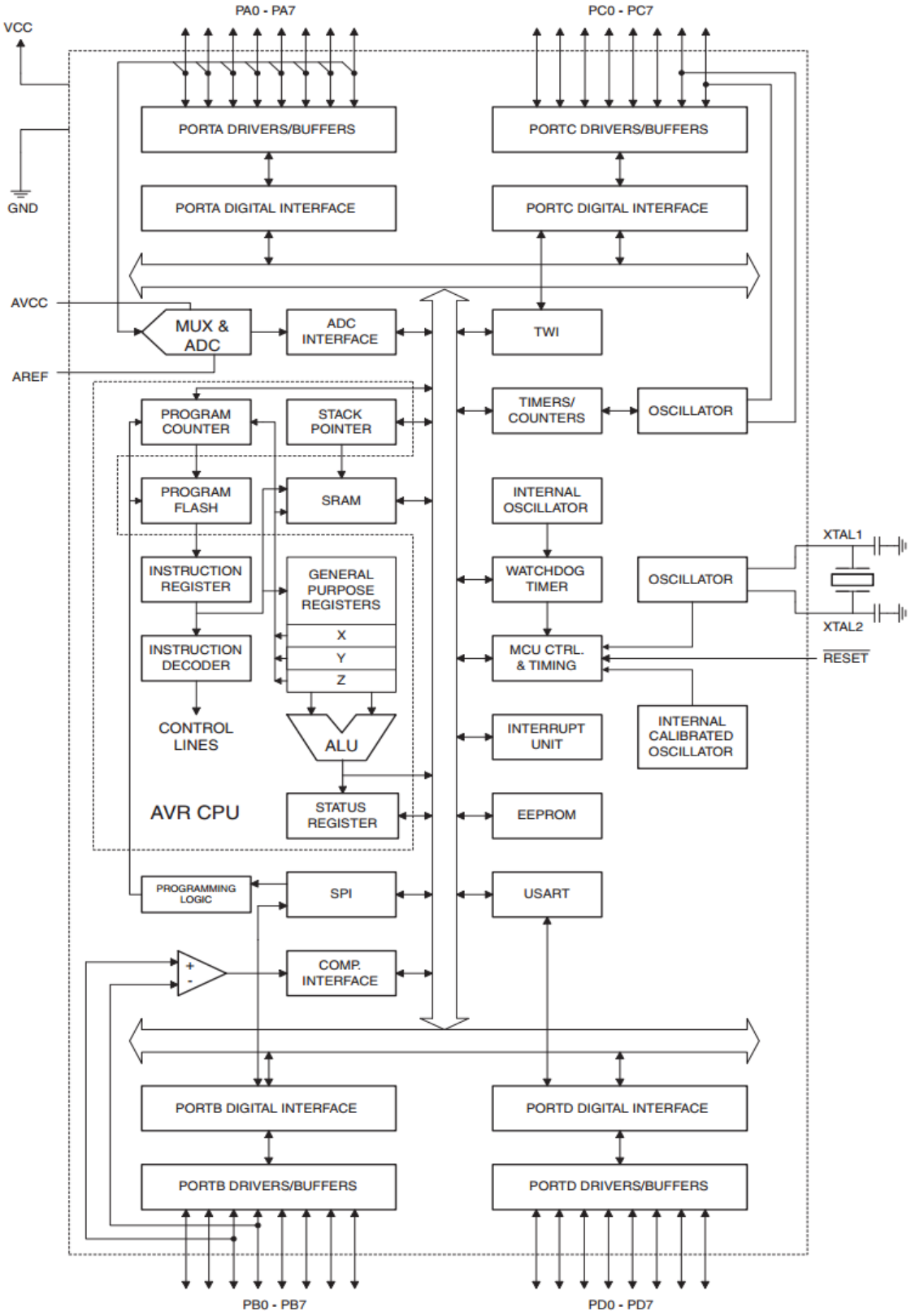
2000 - Intel Pentium 4 – 32-bitový (42 milionu tranzistorů).

2.1.1 **Historie mikroprocesorů AVR**

Za realizací technologie AVR stojí dva studenti, Alf-Egil Bogen a Vegard Wollan z Norského technologického institutu. Mikroprocesory přišly do výroby okolo roku 1997, první mikroprocesor AVR byl vyvinut v Trondheimu v institutu ASIC (Aplikačně Specifické Integrované Obvody) dnes známá jako Nordic Semiconductor, kde Alf-Egil Bogen a Vegard Wollan pracovali. Mezi úplně první sériové mikroprocesory od AVR patří AT90S8515. Má takřka stejné vývodové uspořádání jako mikroprocesor 8051. [8], [3], [7]

2.2 Základní vnitřní struktura procesorů AVR [8], [5], [7]

Číslicový systém s primitivními mikroprocesory je zpravidla realizován jako mikroprocesorový s podpůrnými obvody pomocí hradel. V devadesátých letech se objevují mikroprocesory s integrovanými podpůrnými obvody v jednom pouzdře. V současnosti programovatelné obvody obsahují v jednom pouzdře mimo programovatelné logiky i logické bloky (paměti, rozhraní, registry, čítače, komparátory) a mezi nimi také mikroprocesor [1]. Číslicový systém lze realizovat na jednom čipu s minimem vnějších součástek viz Obr.č. 2.



Obr.č. 2 Příklad jednočipového mikroprocesoru ATmega 32 obsahující množství podpůrných obvodů a několik druhů pamětí. Převzato a upraveno dle [7]

Například mikroprocesor AT-MEGA32 obsahuje čtyři osmibitové porty, konfigurovatelné jako output nebo input. Dále lze jeden port nakonfigurovat jako osm AD převodníků. Podporuje hardwarově sériovou sběrnici I^2C, Jtag interface, Boundary scan a SPI interface, programovatelný sériový UART.

Obsahuje 1x 16-bit timer/counter. 2x 8-bit timer/counter 4x PWM kanál. A to se jedná o low cost řešení mikroprocesoru.

Dají se pořídit i dražší mikroprocesory, které na jednom chipu integrují prakticky celé 32-bitové PC včetně periferií výše zmíněných a USB, LAN a další rozhraní včetně grafického.

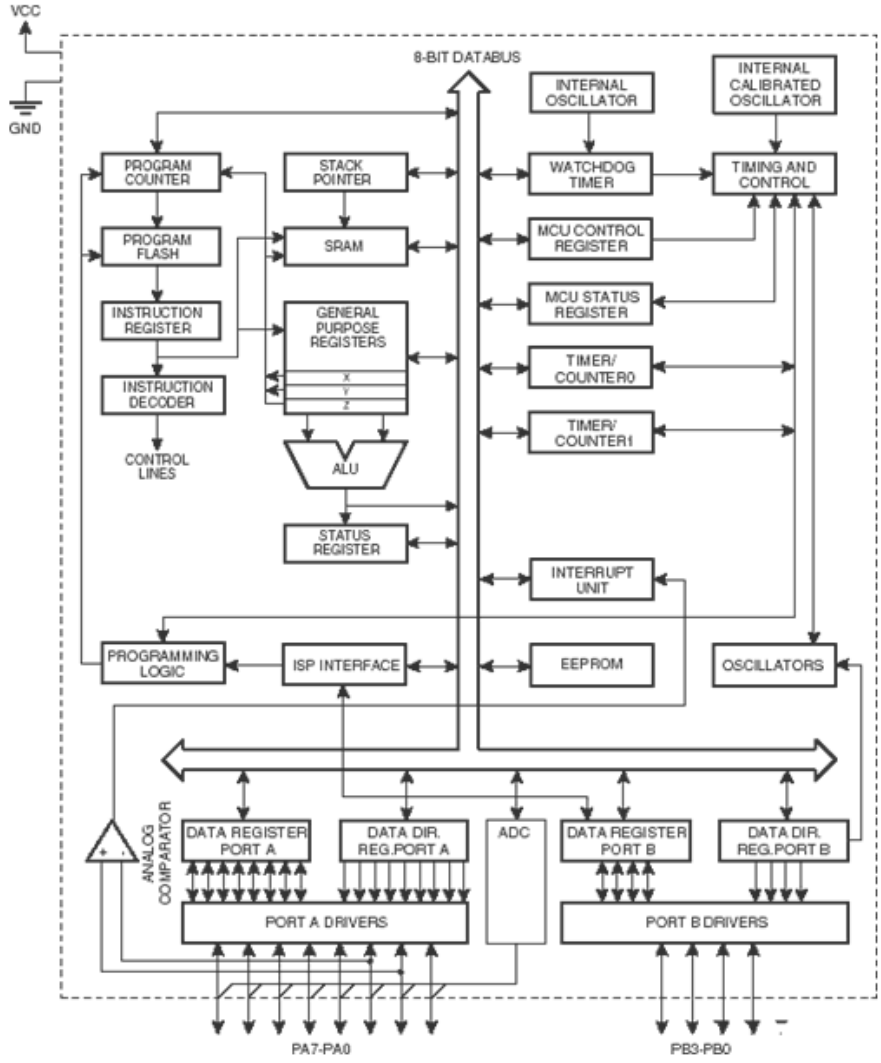
Jedná se o RISC (mikroprocesory s omezenou znakovou sadou) založené na Harvardské architektuře, což je činí jedinečnými. Alf-Egil Bogena a Vegard Wollana napadlo spojit výhody Harvardské architektury s technologií procesorů s omezenou znakovou sadou a ukládáním instrukce i data do společného registru. Skládají se z 32 osmi bitových registrů, které mohou obsahovat jak data, tak adresu a jsou připojeny k ALU (aritmeticko-logická jednotka). To umožňuje přístup do dvou registrů v rámci jednoho hodinového cyklu. ALU zpracuje v jednom cyklu jednu operaci. Tato architektura je vysoce efektivní a zajišťuje řádově vyšší výkon než běžné mikroprocesory architektury CISC (mikroprocesory s plnou znakovou sadou).

2.2.1 **Podpůrné obvody v mikroprocesoru AVR [9] :**

Podpůrné obvody jsou implementovány do struktury jednočipových mikroprocesorů z důvodu zjednodušení konstrukce. Podpůrné obvody a jejich uspořádání je patrné z blokového schématu procesoru AVR viz Obr.č. 3.

- power supervision
- qtouch support
- časovač
- čítače
- watchdog timer
- řadič displeje

- řadič klávesnice
- výstup PWM
- programovatelné hradlové pole
- paralelní porty (až desítky pinů)
- sériové porty (asynchronní, synchronní)
- porty komunikačních sběrnic (CAN, Ethernet)
- A/D převodníky
- zámkové bity
- IO porty
- řadič přerušení



Obr.č. 3 Podpůrné obvody a jejich uspořádání. Převzato a upraveno dle [8], [7]

2.2.2 **Popis některých pojmů a bloků - jejich funkce a vlastnosti**

2.2.2.1 **Zdroje hodinového signálu**

- externí krystal, přesnost se pohybuje v řádech 10^{-5}
- externí nízkofrekvenční krystal 10^{4} Hz
- interní RC oscilátor, velmi nízká přesnost
- externí reference

2.2.2.2 **IO porty**

Tyto komunikační porty jsou nejuniverzálnějším rozhraním mikroprocesoru. Počet portů a množství pinů koresponduje s použitím mikroprocesoru, např.

- ATtiny12 6 pinů,
- ATmega32 32pinů,
- ATmega103 48 pinů.

Piny lze nastavit jako vstupní i výstupní. [10]

2.2.2.3 **Zámkové bity (fuse bits)**

Slouží pro nastavení low level funkcí procesoru. [10]

- OCDEN – ladění přímo na chipu
- JTAGEN - povolení JTAG rozhraní
- SPIEN – povolení In Systém Programing
- EESAVE - při smazání paměti mikroprocesoru zůstanou data v EEPROM
- BOOTSZ1:0 velikost bootovací sekce ve Flash (128 až 1 024 slov)
- BOOTRST vektory přerušení do bootovací sekce
- BODLEVEL výběr úrovně detekce výpadku napájení
- BROWN-OUT interní oscilátor
- BODEN povolení detekce výpadku napájení
- SUT1:0 výběr start-up prodlevy (0; 4,1ms, 65ms)

- CKSEL3:0 zdroj hodinového signálu. Pro aplikace u kterých je přesnost oscilátoru nedůležitá.

2.2.2.4 **Interní napěťová reference**

Vlastní napěťová reference pro AD a DA převodníky. [10]

2.2.2.5 **Analogový komparátor**

Tato periferie slouží pro porovnávání úrovní analogových signálů. Vrací hodnotu, kterou je možné vyčítat procesorem a dokáže generovat přerušení. [10]

2.2.2.6 **Podpora dotykové vrstvy „Qtouch Support"**

Slouží jako HW driver dotykové kapacitní vrstvy. Tato vrstva se používá jako vstupní zařízení u mobilních telefonů atd. [10]

2.2.2.7 **PWM výstup**

Výstup pulzně šířkové modulace. S RC článkem lze použít jako jednoduchý DA převodník. Může sloužit rovnou pro připojení modelářského servomotoru. [10]

2.2.2.8 **A/D převodníky**

Mikroprocesor v sobě může obsahovat velké množství AD převodníků (Analog to digital convertor). AD převodník slouží k převodu hodnoty elektrického napětí nebo proudu na číslo. Například ATmega32 jich obsahuje osm, 10-bitových. [10]

2.2.2.9 **Programovatelné hradlové pole**

Pomocí hradlového pole lze složit prakticky jakákoli logická funkce. Pokud tento blok mikroprocesor obsahuje, ušetří se tím veliké množství součástek potažmo energie a místa na desce plošných spojů. [10]

2.2.2.10 **Řadič klávesnice**

Řadič klávesnice usnadňuje obsluhu klávesnice. [10]

2.2.2.11 **Řadič displeje**

Řadič displeje zajišťuje HW obsluhu displeje. [10]

2.2.2.12 **Časovač/čítač**

Časovač/čítač dokáže zajistit čítání pulzů např. při „přetečení" (stav, kdy se čítač naplní daty) a dokáže vyvolat přerušení. Dokáže také sloužit jako vstup pro PWM a pak ho lze použít např. jako vstup pro inkrementální rotační enkodér. [10]

2.2.2.13 **Řadič přerušení**

Řadič přerušení je velice důležitý z hlediska robustnosti systému. Při přerušení procesor skočí na určitý podprogram, ale přerušení mohou mít různou prioritu např. přerušení, které zavře ventil přivádějící vodu do nádrže a tlačítko které nám aktivuje display. Je nutné rozlišit přerušení podle priority. Řadič přerušení má určitý počet úrovní. To dovoluje, aby mikroprocesor řešil nastalé situace v posloupnosti podle priorit a tak zabezpečil včasné odbavení urgentních stavů před méně významnými. [9]

2.2.2.14 **Watchdog**

Watchdog, slouží jako bezpečnostní pojistka pro případ, kdy dojde z jakéhokoli důvodu k přerušení chodu mikroprocesoru. Tato pojistka čeká na impulz, který musí přijít v předem definovaném časovém okně. Pokud přijde po konci tohoto vyhrazeného času, je mikroprocesor automaticky resetován. V případě nepoužití této pojistky může dojít k přerušení chodu mikropočítače. Doba resetu je také důležitá. Watchdog bývá kombinován ještě s Power management obvodem, který vyvolá reset po poklesu napětí pod předem definovanou úroveň. Tyto systémy mají za úkol zvýšit spolehlivost zařízení. [9]

2.2.2.15 **Obvod „Power management"**

Obvod Power management zajišťuje, aby nedošlo k chybné funkci mikroprocesoru v důsledku krátkodobého poklesu napětí na napájecím zdroji.
V principu funguje tak, že monitoruje napětí na napájecí větvi mikroprocesoru.
V případě poklesu napětí pod předem stanovenou úroveň vyvolá reset procesoru a tento reset drží po stanovenou dobu. [9]

2.2.2.16 **Přerušení**

Přerušení způsobí okamžité vyvolání podprogramu vnější událostí, která nastává asynchronně vzhledem k běhu programu. Účelem přerušení je přimět procesor, aby na tuto vnější událost určitým způsobem zareagoval. [9]

Přerušení je nutné z toho důvodu, že procesor neumí nic jiného, než postupně vykonávat instrukce programu. Přerušení je i jeho jedinou možnou okamžitou reakcí na vnější stavy, které se musí řešit s vyšší prioritou než vlastní momentální instrukce.

Reakcí procesoru na vnější přerušení je tedy dočasné pozastavení vykonávání instrukcí hlavního programu a po zastavení vykonávání instrukcí hlavního programu a provedení obslužného podprogramu přerušení dojde ke skoku zpět na místo, kde přestal vykonávat hlavní program. Vše probíhá formou volání podprogramu.

2.2.2.17 **Druhy pamětí**

- Flash, mající šířku 16 bit, je přepisovatelná. Garantovaný počet 10 000 cyklů. Sériové programování zajišťuje velký uživatelský komfort.
- Programovou Flash dělíme na dva sektory:
    - Bootovací část (Boot loader). Slouží zpravidla pro ukládání programů (zavaděč), umožňující přeprogramovat bootovací i programovou část.
    - Aplikační část (Application section). Zde se ukládá aplikace, která se má vykonávat.

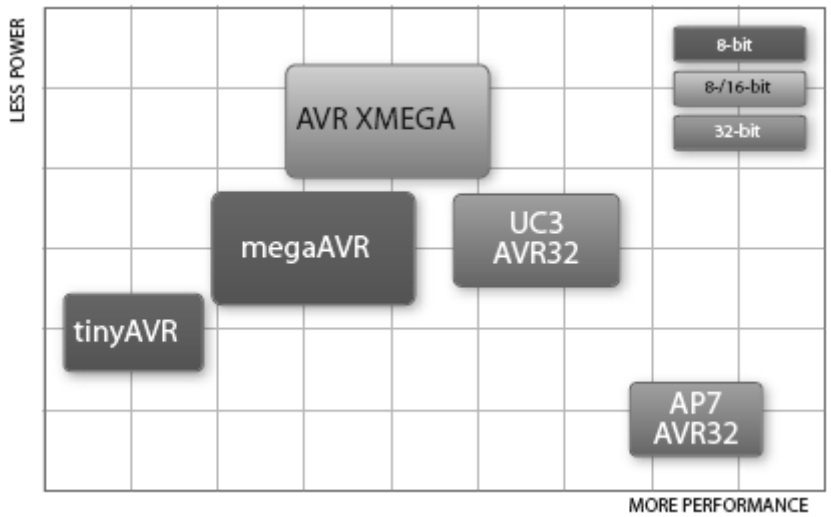
- RAM má šířku 8 bit, je rychlá, napěťově závislá a slouží pro dočasné ukládání dat.
- EEPROM slouží pro uchování dat, která mají být zachována po ukončení programu a můžou se často měnit. Výrobce garantuje až 100 000 zápisů.

2.2.3 **Výhody AVR Atmel [11]**

- Rychlost oproti konkurenci dokáže při stejné frekvenci zpracovat více dat.
- Vývojové prostředí je zdarma.
- Nízká cena programátoru a mikroprocesorů.
- Masovost, velké množství volně přístupných zapojení a kódu.
- Podpora knihoven pro připojovaný HW.
- Vývojové prostředí Bascom, které lze programovat jazykem BASIC.
- Velké množství procesorů s rozmanitými funkcemi.

2.2.4 **Základní tipy procesorů AVR Atmel [11]**

Základní rozdělení rodin AVR atmel viz Obr.č. 4



Obr.č. 4 Orientační tabulka AVR. na ose x je míra výkonu na ose y množství energie, , Převzato a upraveno dle [2]

- AT90 – Dnes již nevyráběná řada mikroprocesorů v dnešní době nahrazena ATtiny, ATmega,

- ATtiny – Miniaturní obvody s minimem vývodů určené pro jednodušší aplikace.
  Flash: 1- 8 kB
  Pouzdro: 6 – cca. 32 pin
  Malá sada rozhraní

- ATtiny85
  Flash (Kbytes): 8 Kbites
  Pouzdro: 8 pin
  Max. operační frek. (MHz): 20 MHz

CPU: 8-bit AVR
Počet portů: 1
Počet I/O: 6 pin

- ATmega – jednočipové mikroprocesory podporující celou škálu rozhraní od I^2C po USB
  Flash: 4 – 256 kB
  Pouzdro: 23 – cca. 100 pin
  Rozšířená sada instrukcí
  Velké množství modulů, integrovaných na čipu

- ATmega32
  Flash (kb): 32 kb
  Max. operační frek. (MHz): 16 MHz
  CPU: 8-bit AVR
  Pouzdro: 44 pin
  Počet I/O: 32 pin

- ATxmega – procesory, určené pro složité aplikace. Velký výkon, velká výbava modulů, integrovaných na čipu.
  Flash: 16 – 384 kB
  Pouzdro: 44 – cca. 100 pin

- ATxmega16A4U
  Flash (kb): 32 kb
  Max. operační frek. (MHz): 32 MHz
  CPU: 8-bit AVR
  Pouzdro: 44 pin
  Počet I/O: 34 pin

- ATmega pro speciální aplikace
  Jedná se o ATmega, vybavenou navíc LCD kontroléry, USB kontroléry nebo speciálními rozhraními (např. CAN)

- FPSLIC™ (AVR s FPGA)
  FPGA (programovatelné logické pole) 5k – 40k hradel
  SRAM pro programový kód

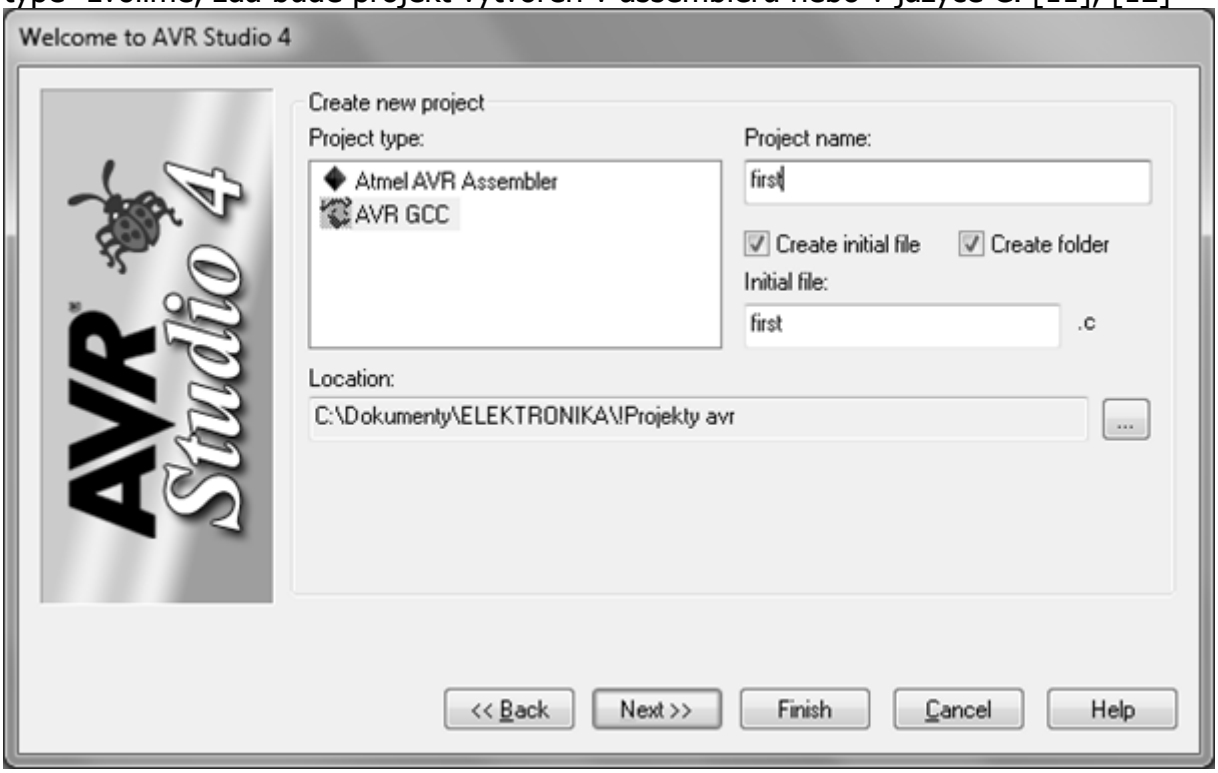
- AVR 32bit
  podpora audia a videa.
  Flash (kb): 256 kb
  Max. operační frek. (MHz): 50 MHz
  CPU: 8-bit AVR
  Pouzdro: 44 pin
  Počet I/O: 34 pin

2.3 Vývojové prostředí [11], [12]

2.3.1 **AVR Studio**

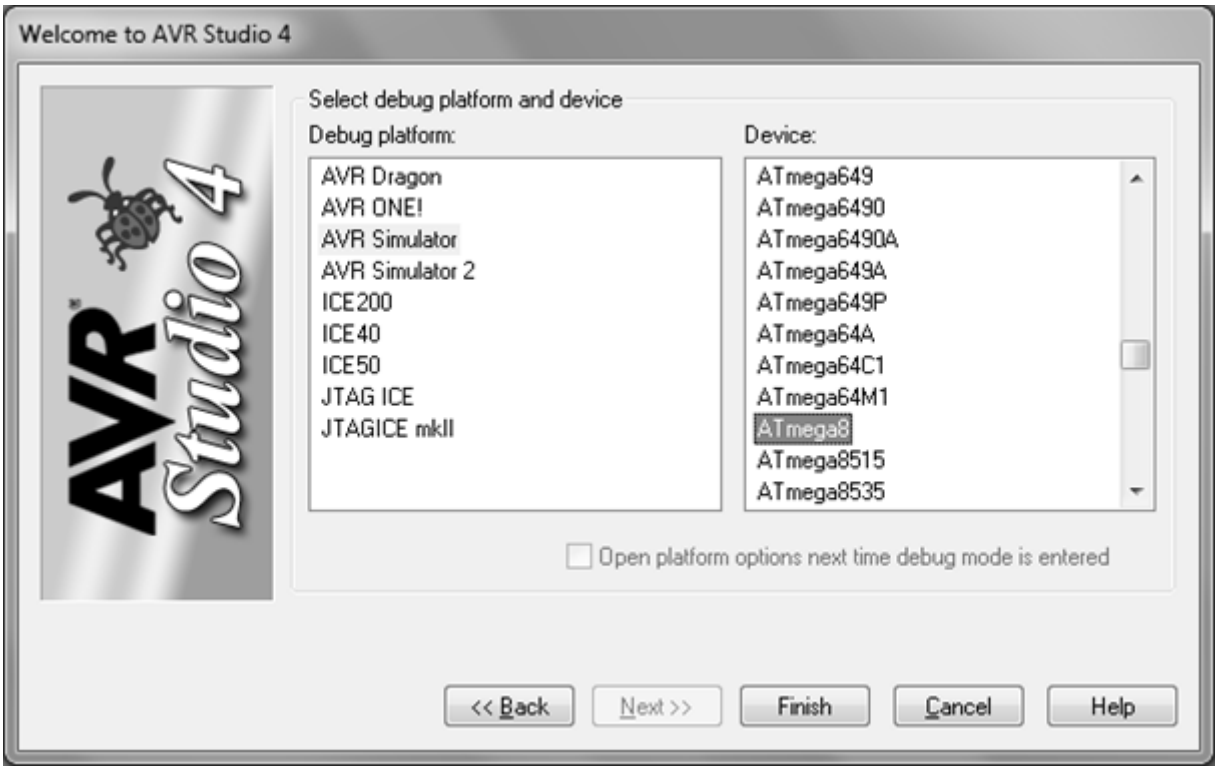
2.3.1.1 **Založení projektu**

Na úvodní obrazovce vytvoření projektu na Obr.č. 5 v podnabídce „Project type" zvolíme, zda bude projekt vytvořen v assembleru nebo v jazyce C. [11], [12]



Obr.č. 5 Úvodní obrazovka pro vytvoření projektu. Převzato a upraveno dle [11]

Kliknutím na nabídku „NEXT" bude zobrazeno okno s výběrem procesoru a podporovaného programátoru viz Obr.č. 6



Obr.č. 6 Okno výběru procesoru a programátoru. Převzato a upraveno dle [11]

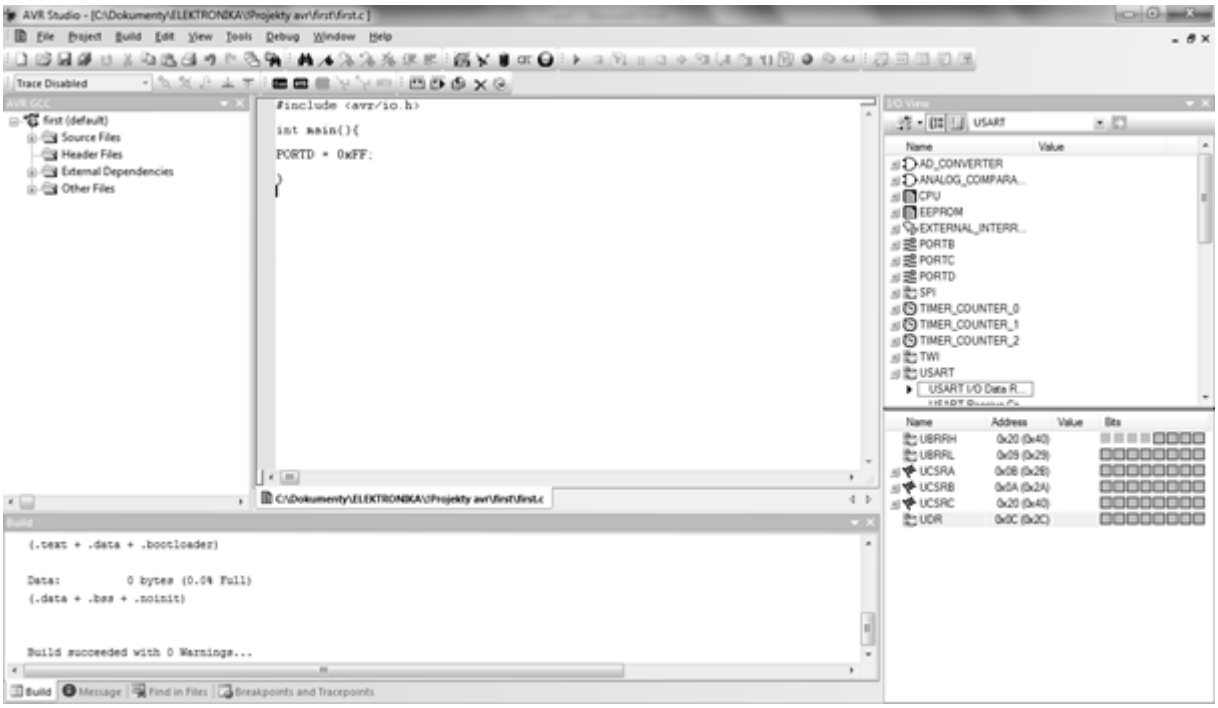
2.3.1.2 **Popis pracovního prostředí [11], [12]**

Ve vytvořeném projektu Obr.č. 7 lze psát samotný projekt v jazyce C. Vlevo nahoře lze zvolit hlavičkové soubory, které lze přes „#include" vložit do projektu.

Uprostřed obrazovky se nachází okno, kam bude psán samotný projekt.

Vpravo jsou ovládací registry procesoru, které můžeme při debuggování nastavovat nebo sledovat v čase běhu programu jejich aktuální stav.

V dolní části okna se nachází informační okno, ve kterém jsou zobrazovány informační textové řetězce, týkající se kompilátoru, využití Flash a RAM při běhu programu.



Obr.č. 7 Standartní pracovní plocha programu AVR Studio.
Nastavení projektu

Po kliknutí na tlačítko „project/configuracion/options" bude otevřeno menu. V tomto menu lze nastavovat:

- „frequency" frekvenci mikroprocesoru,
- „device" model mikroprocesoru,
- „optimalization" lze nastavit úroveň kompilace, která určí, jak moc bude program optimalizován pro rychlost. -Os vypnuty veškeré optimalizace, -O3 optimalizuje pro rychlost na úkor velikosti kódu.

2.3.1.3 **Kompilace [11], [12]**

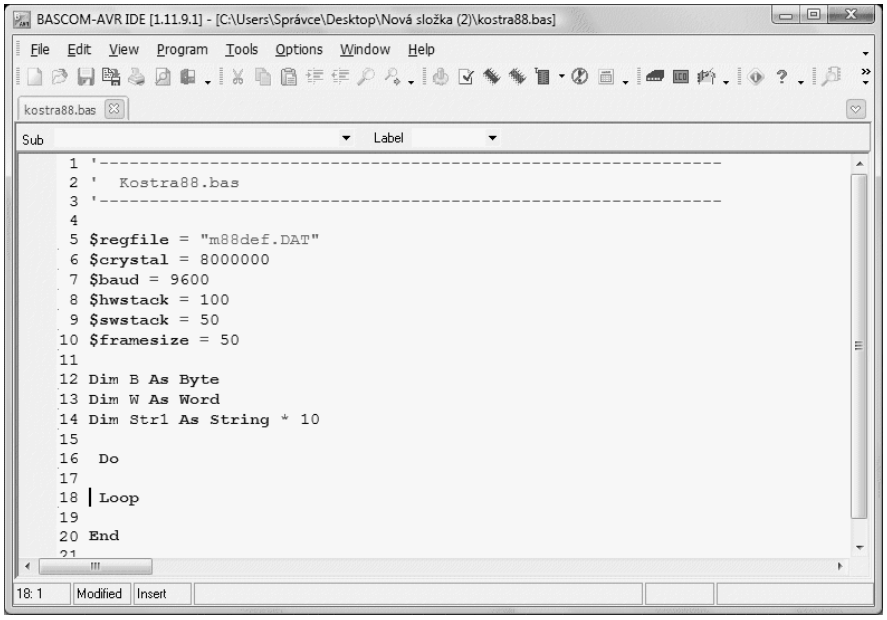
Většina projektů se skládá z několika **.c, *.h* souborů. Důvod rozdělení programu do několika částí je ten, že program je tak přehlednější, ovládací rutiny pro periferie jsou používány coby knihovny, které jsou vždy stejné. Proto jsou snáze použitelné. V průběhu kompilace dojde ke sloučení všech **.c, *.h* do jednoho souboru a následně (velice zjednodušeně) na převod do kódu, který je kompatibilní s hardwarem procesoru. Vznikne soubor **.hex*. Soubor **.hex* je následně možné nahrát do mikroprocesoru vhodným programátorem.

2.3.2 **Bascom-AVR**

Jazyk Bascom je jednoduchý jazyk pro programování mikroprocesorů. Obsahuje velké množství knihoven pro ovládání periferií. Jeho jednoduchost je vykoupena velikostí výsledného programu a nižší rychlostí při provádění kódu, proto pokud budeme požadovat opravdu rychlé reakce procesoru, bude nutné psát klíčové rutiny v assembleru.

Program Bascom-AVR obsahuje simulátor, na kterém lze odlaďovat program, aniž by byl zapsán do mikroprocesoru a navíc podporuje krokování. Dále lze sledovat stav registrů. [13], [4], [14], [12]

Je-li Bascom-AVR spuštěn, naběhne do základní nabídky viz Obr.č. 8.



Obr.č. 8 Hlavní obrazovka programu Bascom-AVR. Převzato a upraveno dle [13].

2.3.2.1 **Popis jednotlivých panelů prostředí Bascom**

2.3.2.1.1 Práce se soubory

Práci se soubory usnaďnuje panel s jejich nabídkou na Obr.č. 9. [14]

Obr.č. 9 Nabídka práce se soubory. Převzato a upraveno dle [14]

Jednotlivé ikony mají následující funkce:

- List „nový soubor"
- Složka s šipkou „otevřít"
- Disketa „ulož"
- List s disketou „ulož jako knihovnu"
- Tiskárna „vytiskne celý program"
- List s lupou „ukáže náhled"
- Dveře „konec programu"

2.3.2.1.2 Práce s bloky textu

Nabídka práce s bloky textu Obr.č. 10. [14]

Obr.č. 10 Nabídka práce s bloky textu. Převzato a upraveno dle [14]

Jednotlivé ikony mají následující funkce:

- Nůžky „vyjmout"
- Dva listy papíru „zkopíruje označené do schránky"
- Blok „vloží uložené ve schránce"
- Lupa „hledat"
- Lupa se šipkou „hledej další"

2.3.2.1.3 Nabídka práce s projektem

Nabídka práce s projektem na Obr.č. 11. [14]

Obr.č. 11 Nabídka práce s projektem. Převzato a upraveno dle [14]

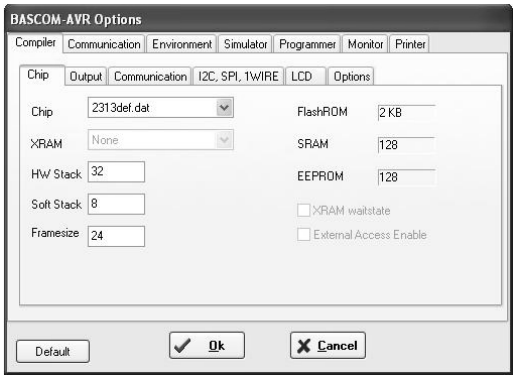
Jednotlivé ikony mají následující funkce:

- Ozubené kolo, vyvolá „options" nastavení projeku a programu Obr.č. 12
- Zatržený papír „provede kontrolu syntaxe".
- Černý integrovaný obvod „kompilátor", provede kompilaci.
- Přeškrtnutý integrovaný obvod „simulátor" - vyvolá simulátor.
- Přeškrtnutý vykřičník „provede reset procesoru".
- Kalkulačka „vyvolá souhrnné info o projektu".

2.3.2.2 **Nabídka „Compiler"**

2.3.2.2.1 Podnabídka „Chip"

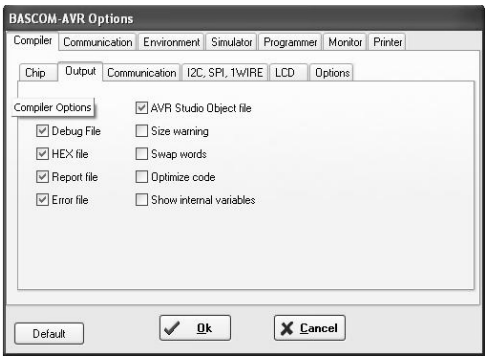
Nabídka Chip na Obr.č. 12 výběr datového souboru, který nastaví všechna potřebná nastavení pro mikroprocesor, pro který je určen. Jedná se o to, aby kompilátor věděl o portech o velikosti flash, EEPROM, SRAM paměti atd. [14]



Obr.č. 12 Nabídka options vyvolaná ikonou ozubeného kola.
Převzato a upraveno dle [14]

2.3.2.2.2 Podnabídka „Output"

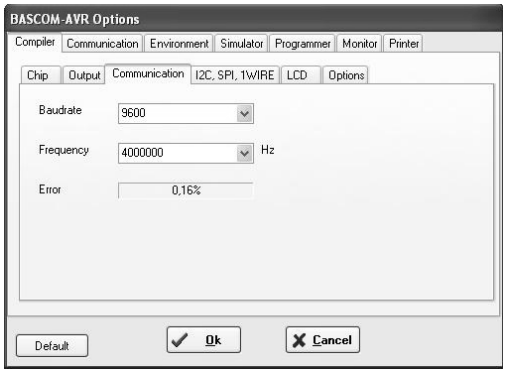
Podnabídka „Output" na Obr.č. 13. V této podnabídce lze nakonfigurovat, které soubory budou vytvořeny kompilátorem. [14]



Obr.č. 13 Podnabídka Output výstup po kompilaci. Převzato a upraveno dle [14]

2.3.2.2.3 Podnabídka „Communication"

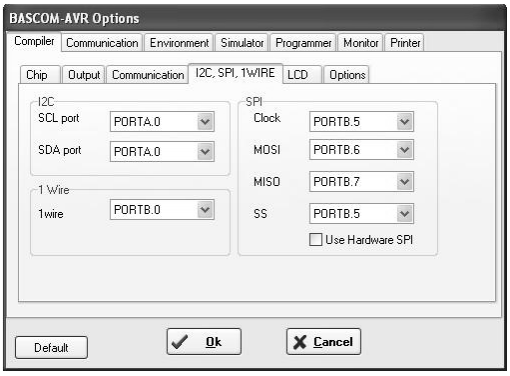
V podnabídce Compiler na Obr.č. 14. Nastavení datového toku a frekvence pro přenos dat mezi procesorem a okolím. Error určuje pravděpodobnou chybovost (dle nastavení). [14]



Obr.č. 14 Nastavení datového toku a frekvence pro přenos dat. Převzato a upraveno dle [14]

2.3.2.2.4 Výběr komunikace

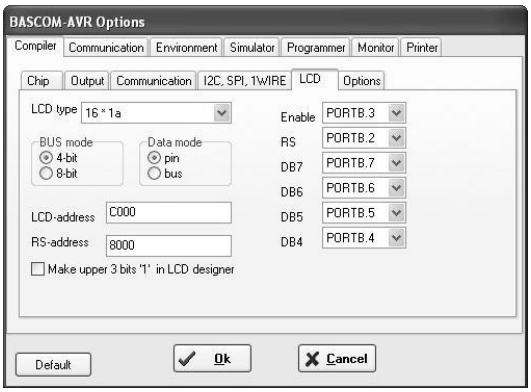
Zde se mikroprocesoru nakonfiguruje, na jakých portech budou jaká rozhraní. Na Obr.č. 15 je konfigurační okno. [14]



Obr.č. 15 Konfigurační okno pro nastavení komunikačních kanálů s procesorem. Převzato a upraveno dle [14]

2.3.2.2.5 Podnabídka LCD

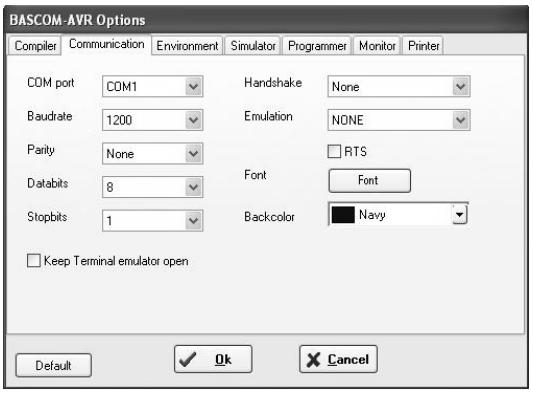
Nastavení, které specifikuje, jaký typ LCD bude použit, jak širokou bude mít sběrnici, jakou adresu a které porty a piny jej budou ovládat. Okno programu vyobrazuje Obr.č. 16. [14]



Obr.č. 16 Podnabídka „LCD". Převzato a upraveno dle [14]

2.3.2.2.6 Nabídka Comunication

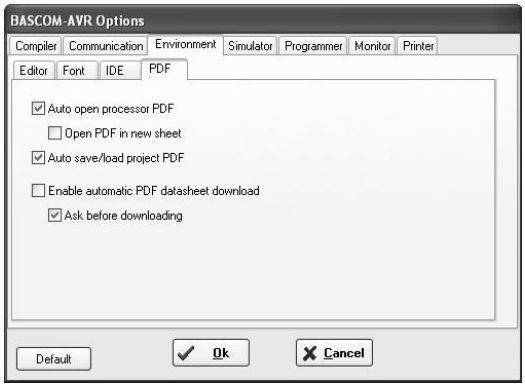
Slouží pro konfiguraci komunikace programu Bascom-AVR s MCU. Tabulku vyobrazuje Obr.č. 17. [14]



Obr.č. 17 Konfigurační okno komunikace. Převzato a upraveno dle [14]

2.3.2.2.7 Nabídka Enviroments

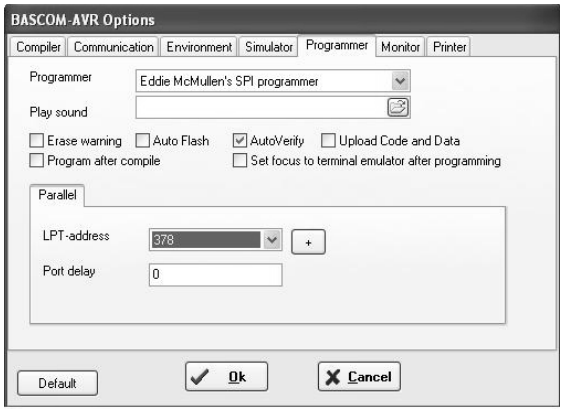
Konfigurace uživatelského komfortu (barvy atd.) a vyhrazených proměnných viz Obr.č. 18. [14]



Obr.č. 18 Nabídka enviroment. Převzato a upraveno dle [14]

2.3.2.2.8 Programmer

Tato nabídka na Obr.č. 19 umožňuje nastavení programátoru. [14]



Obr.č. 19 Nastavení programátoru. Převzato a upraveno dle [14]

2.3.2.2.9 Terminal Emulator

Umožňuje komunikovat přes zvolený port s mikroprocesorem. [14]

2.3.2.2.10 LCD Designer

Umožňuje vytvořit vlastní znak pro znakové LCD. [14]

2.3.2.2.11 Lib Manager

Manager, který zobrazí všechny vložené knihovny. [14]

2.3.2.2.12 Graphic converter

Převodník pro převod bitmap na data pro grafické LCD. [14]

2.3.2.3 **Syntaxe jazyka Bascom – AVR [5], [10], [11], [13], [14]**

$ASM
Začátek bloku kódu v asembleru.
Syntaxe:
$ASM

$BAUD
Nastavení rychlosti seriového přenosu.
Syntaxe:
$BAUD = var
příklad:
$BAUD = 4800

$CRYSTAL
Kmitočet krystalu, nebo vnitřního oscilátoru v Hz.

$EEPROM
Oznamuje překladači, že následující sekvence DATA má být uložena v souboru*.eep.

$EEPROMHEX
Soubor EEP bude vytvořen ve formátu Intel Hex.

$END ASM
Konec bloku kódu v asembleru.

$EXTERNAL
Deklarace asm rutiny z knihovny

$INC
Vloží na aktuální pozici binární soubor.
Syntaxe:
$INC návěští , size | nosize , "soubor"

$INCLUDE
Vloží na aktuální pozici ASCII soubor.
Syntaxe:
$INCLUDE "test.inc"

$INITMICRO
Zavolá po resetu proceduru _init_micro. Pokud chceme hned po resetu provést nějaké činnosti, je nutné je zapsat do _init_micro.

$LIB
Tato direktiva sděluje překladači, které knihovny v kódu budou použity.
Syntaxe:
$LIB "libname1" [, "libname2"]

$SIM
Poznámky:
Tento příkaz se vkládá do kódu kvůli správné funkci při simulování. Před kompilací ostrého programu musí být odstraněn.

$TIMEOUT
Nastaví timeout pro HW UART0.

$TINY
Tato direktiva se použije u procesorů, které nemají paměť SRAM (attiny 12,15 atd). Říká překladači, aby vynechal inicializaci zásobníků.

$WAITSTATE
Vložení čekacího stavu pro pomalejší externí paměť.

ABS
Absolutní hodnota.

ACOS
Vrací arkuskosinus úhlu.

ALIAS
Nové jméno pro proměnnou
Syntaxe:
newname ALIAS oldname

ASC
Vrací ascii hodnotu prvního znaku v řetězci.
Syntaxe:
var = ASC(string)

ASIN
Vrací arkussinus úhlu.

ATN
Vrací arkustangens úhlu.

BIN
Konvertuje číslo na binární string.

BINVAL
Konvertuje binární string na číslo.

BITWAIT
Čeká, až je bit 0 nebo 1.
Syntaxe:
BITWAIT x , SET/RESET

BYVAL
Určuje, že proměnná má být předána hodnotou.
Syntaxe:
Sub Test(BYVAL var)

CALL
Volání procedury.
Syntaxe:
CALL Test [(var1, var-n)]

CASE
Vykoná jeden z několika bloků příkazů v závislosti na hodnotě výrazu.
Syntaxe:
SELECT CASE var
CASE test1 : statements
[CASE test2 : statements]
CASE ELSE : statements
END SELECT

CHECKSUM
Vrací kontrolní součet řetězce.
Syntaxe:
PRINT Checksum(var)
b = Checksum(var)

CHR
Konvertuje ASCII hodnotu na string o délce 1 znak.
Syntaxe:
s = CHR(var)

CLS
Vymaže LCD displej.

CPEEK
Přečte bajt z paměti programu (FLASH).
Syntaxe:
var = CPEEK(adresa)

CURSOR

Nastavení LCD kurzoru.

Syntaxe:

CURSOR ON/OFF (zapnutý/vypnutý)
CURSOR BLINK/NOBLINK (bliká/nebliká)

DATA

Řada konstantních hodnot, které budou čteny příkazem READ.

Syntaxe:

DATA var1 , var2 , var3 , ...varn

DEBOUNCE

Testuje vývod portu s připojeným tlačítkem.

Syntaxe:

DEBOUNCE Px.y , stav , návěští [, SUB]

DECLARE FUNCTION

Deklarace uživatelské funkce.

Syntaxe

DECLARE SUB TEST[([BYREF/BYVAL] var as type)]

DECR

Zmenší proměnnou o 1.

DEFxxx

Jiný způsob deklarace proměnné:

Syntaxe:

DEFBIT b - všechny proměnné, začínající písmenem B budou typu BIT
DEFBYTE c - všechny proměnné, začínající C budou typu BYTE
DEFINT I - podobně INTEGER
DEFWORD x - WORD
DEFLNG l - LONG
DEFSNG s - SINGLE

DEFLCDCHAR

Definice uživatelského znaku.

Syntaxe:

DEFLCDCHAR char,r1,r2,r3,r4,r5,r6,r7,r8
char - číslo znaku (zvolíme 0-7)

DEG2RAD

Převede stupně na radiány.

Syntaxe:

var = DEG2RAD(single)

DIM

Deklarace (přiřazení typu) proměnné.

Syntaxe:

DIM var AS [XRAM/SRAM/ERAM] type [AT location/variable] [OVERLAY]

DISABLE

Zakázání přerušení.

Syntaxe:

DISABLE INTERRUPTS - zákaz všech přerušení

DISABLE INT0 - External Interrupt 0
DISABLE INT1 - External Interrupt 1
DISABLE OVF0,TIMER0, COUNTER0 - TIMER0 overflow interrupt
DISABLE OVF1,TIMER1,COUNTER1 - TIMER1 overflow interrupt
DISABLE CAPTURE1, ICP1 - INPUT CAPTURE TIMER1 interrupt
DISABLE COMPARE1A,OC1A - TIMER1 OUTPUT COMPARE A interrupt
DISABLE COMPARE1B,OC1B - TIMER1 OUTPUT COMPARE B interrupt
DISABLE SPI - SPI interrupt
DISABLE URXC - Serial RX complete interrupt
DISABLE OVF0,TIMER0, COUNTER0 - TIMER0 overflow interrupt
DISABLE OVF1,TIMER1,COUNTER1 - TIMER1 overflow interrupt
DISABLE CAPTURE1, ICP1 - INPUT CAPTURE TIMER1 interrupt
DISABLE COMPARE1A,OC1A - TIMER1 OUTPUT COMPARE A interrupt
DISABLE COMPARE1B,OC1B - TIMER1 OUTPUT COMPARE B interrupt

DISABLE SPI *- SPI interrupt*
DISABLE URXC *- Serial RX complete interrupt*

DISPLAY

Zapnutí nebo vypnutí displeje.

Syntaxe:

DISPLAY ON
DISPLAY OFF

DO-LOOP

Opakuj příkazy, dokud podmínka není pravdivá.

Syntaxe:

DO
příkazy
LOOP [UNTIL podmínka]

DTMFOUT

Pošle DTMF tón na pin OC1.

Syntaxe:

DTMFOUT číslo, délka
DTMFOUT string , délka

ECHO

Zapne nebo vypne echo v příkazu INPUT pro seriový port.

Syntaxe:

ECHO ON
ECHO OFF

ENABLE

Povolení přerušení

Syntaxe:

ENABLE INTERRUPTS - povolení všech přerušení ENABLE INT0

END

Ukončení běhu programu.

Syntaxe:

END

EXIT

Vystoupení ze smyčky, procedury nebo funkce.

Syntaxe:

EXIT FOR
EXIT DO
EXIT WHILE
EXIT SUB
EXIT FUNCTION

EXP

Vrací „e" umocněné na proměnnou typu single. „e" je základ přirozených logaritmů.

Syntaxe:

var1 = Exp(x)

FIX

Odřízne desetinnou část čísla.

Syntaxe:

var = FIX(x)

FOR-NEXT

Provede zadaný počet cyklů.

Syntaxe:

FOR var = start TO end [STEP krok]
instrukce
NEXT [var]

FORMAT

Formátuje numerický řetězec.

Syntaxe:

str1 = Format(str2, "maska")

FRAC

Vrací desetinnou část proměnné typu single.

FUSING

Vrací formátovaný string proměnné single.

Syntaxe:

cil = Fusing(zdroj, "maska")

GETADC

Přečte hodnotu z A/D konvertoru

GETKBD

Vrací hodnotu stisknutého tlačítka v maticové klávesnici 4x4.

Syntaxe:

var = GETKBD()

GOSUB

Skok na podprogram.

Syntaxe:

GOSUB návěští

GOTO

Skok na návěští.

Syntaxe:

GOTO návěští

HEX

Převede číslo na hexadecimální řetězec.

Syntaxe:

var = hex(x)

HEXVAL

Převede hexadecimální řetězec na číslo.

Syntaxe:

var = HEXVAL(x)

HIGH

Vrací nejvýznamnější bajt proměnné.

Syntaxe:

var = HIGH(x)

HIGHW

Vrací nejvýznamnější word proměnné.

Syntaxe:

var = HIGHW(x)

HOME

Nastaví kurzor LCD na začátek řádku.

Syntaxe:

HOME UPPER - kurzor na začátek prvního řádku
HOME LOWER - kurzor na začátek druhého řádku
HOME THIRD - kurzor na začátek třetího řádku
HOME FOURTH - kurzor na začátek čtvrtého řádku

IDLE

Uvede procesor do stavu IDLE.

Syntaxe:

IDLE

IF-THEN-ELSE-ENDIF

Umožňuje větvení programu podle pravdivosti podmínky.

Syntaxe:

IF podmínka THEN
příkazy
[ELSEIF podmínka THEN]
příkazy
[ELSE]
příkazy
END IF

INCR

Zvětší proměnnou o 1.

Syntaxe:

INCR var

INKEY

Vrací přijatý znak ze sériového portu. Pokud tam žádný znak není, vrátí nulu.

Vrátí nulu i v případě, že je tam binární 0, proto pro binární přenos použijeme napřed *ISCHARWAITING().*

Syntaxe:

var = INKEY()
var = INKEY(#kanál) 'u softwarového UART

INP

Vrací bajt, přečtený z určené adresy v paměti dat.

Syntaxe:

var = INP(adresa) 'adresa = 0 - $ffff

INPUT

Umožní vstup z klávesnice při spojení s PC přes RS232.

Syntaxe:

INPUT ["prompt"] , var1 [, var2 , ... varn]
INPUT #ch, var1 [, var2 , ... varn] 'ch = číslo kanálu softw. UART

INPUTBIN

Čte binární data ze seriového portu.

Syntaxe:

INPUTBIN var1 [,var2] 'var - proměnné pro uložení přijatých dat
INPUTBIN #kanál , var1 [,var2]

INPUTHEX

Podobné jako INPUT, ale zadáváme čísla v hexadecimálním tvaru.

Syntaxe:

INPUTHEX [" prompt"] , var [, varn]

INT

Vrací celou část proměnné typu Single.

Syntaxe:

var = INT(single)

ISCHARWAITING

Vrací 1 pokud v UART bufferu čeká znak.

Syntaxe:

var = ISCHARWAITING()
var = ISCHARWAITING(#channel)

LCASE

Změní všechna písmena ve stringu na malá.

Syntaxe:

Str2 = Lcase(Str1)

LCD

Zobrazí proměnnou nebo konstantu na LCD displeji.

Syntaxe:

Lcd x

LEFT

Vrací určený počet znaků od začátku řetězce.

Syntaxe:

var = Left(var1 , n)

LEN

Vrátí délku řetězce.

Syntaxe:

var = LEN(string)

LOAD

Naplní čítač hodnotou.

Syntaxe:

Load timer , hodnota

LOADADR

Vloží adresu proměnné do registrového páru.

Syntaxe:

LOADADR var , reg

LOADLABEL

Vrací adresu návěští.

Syntaxe:
Var = LOADLABEL(label)

LOCAL
Deklaruje lokální proměnnou v proceduře nebo funkci.
Syntaxe:
LOCAL var As Type

LOCATE
Přesune LCD kurzor na určenou pozici.
Syntaxe:
LOCATE y , x

LOG
Vrací logaritmus naturalis.
Syntaxe:
Target = Log(source)

LOG10
Vrací dekadický logaritmus.
Syntaxe:
Target = Log10(source)

LOOKDOWN
Vrací pozici hledaného čísla v tabulce DATA.
Syntaxe:
var =LOOKDOWN(hodnota, návěští, počet)

LOOKUP
Vrací číslo na určené pozici v tabulce DATA.
Syntaxe:
var =LOOKUP(index, návěští)

LOOKUPSTR
Vrací řetězec z tabulky DATA.
Syntaxe:
var =LOOKUPSTR(index, label)

LOW
Vrací nejnižší bajt proměnné.
Syntaxe:
var = LOW(s)

LOWERLINE
Přesune LCD kurzor na začátek druhého řádku.

MAKEINT
Spojí dva byty do Word nebo Integer.
Syntaxe:
varn = MAKEINT(LSB , MSB)
*(varn = (256 * MSB) + LSB)*

MAX
Vrací maximální hodnotu v poli Byte nebo Word.
Syntaxe:
var1 = MAX(var2)
MAX(ar(1), m ,idx)

MID
Funkce MID vrací část řetězce. Příkaz MID nahradí část řetězce jiným řetězcem.
Syntaxe:
var = MID(var1 ,st [, l])
MID(var ,st [, l]) = var1

MIN
Vrací minimální hodnotu v poli Byte nebo Word.
Syntaxe:
var1 = MIN(var2)
MIN(ar(1), m , idx)

ON VALUE
Větví program na některé z uvedených návěští podle hodnoty proměnné.

Syntaxe:
ON var [GOTO] nebo [GOSUB] label1 , label2 , label3 ... [,CHECK]

OPEN
Použijeme pro otevření softwarového sériového kanálu.

OUT
Zapíše bajt do paměti dat (i externí).
Syntaxe:
OUT adresa, hodnota

PEEK
Vrací obsah pracovního registru (R0 - R31).
Syntaxe:
var = PEEK(adresa)

POKE
Zapíše bajt do pracovního registru (R0 - R31).
Syntaxe:
POKE adresa , hodnota

POWER
Vrací mocninu čísla.
Syntaxe:
var = POWER(x , y) 'var = x na y
Poznámky:
var, x, y - typ Single

POWERDOWN
Uvede procesor do módu Powerdown.
Syntaxe:
POWERDOWN

PRINT
Zápis do sériového portu (RS232).

PRINTBIN
Pošle proměnnou binárně na sériový port.

PUSHALL - POPALL
Uložení a obnovení všech registrů používaných Bascomem.

READ
Čte hodnoty z tabulky DATA.

READEEPROM
Čte z vnitřní EEPROM.

RESET
Vynuluj bit.

RESTORE
Určuje která sekvence DATA se bude číst příkazem READ.
Nastaví ukazatel na začátek sekvence.

RETURN
Návrat z podprogramu.

RIGHT
Vrací určený počet znaků od konce řetězce.

RND
Vrací náhodné číslo
Syntaxe:
var = RND(limit)

ROTATE
Rotace všech bitů proměnné.
Syntaxe:
ROTATE var , LEFT/RIGHT [, shifts]

ROUND

Zaokrouhlí proměnnou typu Single.

Syntaxe:

var = ROUND(x)

RTRIM

Vrátí kopii stringu bez koncových prázdných znaků.

Syntaxe:

cil = RTRIM(zdroj)

SELECT CASE

Vykoná jeden z několika bloků příkazů v závislosti na hodnotě výrazu.

Syntaxe:

SELECT CASE var
CASE test1 : statements
[CASE test2 : statements]
CASE ELSE : statements
END SELECT

SET

Nastav bit (na hodnotu 1).

Syntaxe:

SET bit
SET var.x

SHIFT

Aritmetický posun.

Syntaxe:

SHIFT var , LEFT/RIGHT [, shifts]

SHIFTCURSOR

Posune LCD kurzor o 1 místo.

Syntaxe:

SHIFTCURSOR LEFT
SHIFTCURSOR RIGHT

SHIFTLCD

Posune údaj na LCD displeji o 1 místo.

Syntaxe:

SHIFTLCD LEFT
SHIFTLCD RIGHT

SIN

Vrací sinus úhlu.

Syntaxe:

var = SIN(single) [rad]

SINH

Vrací sinus hyperbolicus.

Syntaxe:

var = SINH(single)

SOUND

Pošle sérii impulzů (tón) na vývod portu.

Syntaxe:

SOUND pin, duration, pulses

SPACE

Vrací string s určeným počtem mezer.

Syntaxe:

var = SPACE(x)

SPC

LCD SPC(4)
Zobrazí 4 mezery na LCD.

SQR

Vrací druhou odmocninu.

STR
Konvertuje číslo na řetězec.
Syntaxe:
var = Str(x)

SWAP
Vymění hodnotu dvou proměnných stejného typu.
Syntaxe:
SWAP var1, var2

TAN
Vrací tangens úhlu
Syntaxe:
var = TAN(single)

TANH
Vrací tangens hyperbolicus.
Syntaxe:
var = TANH(single)

TOGGLE
Neguje hodnotu pinu nebo bitové proměnné.

TRIM
Vrací kopii stringu bez úvodních a koncových prázdných znaků.

UCASE
Změní písmena ve stringu na velká.

VAL
Konvertuje string na číslo.

VARPTR
Vrací adresu proměnné v paměti.

WAIT
Program čeká určený počet vteřin.

WAITKEY
Čeká na přijatý znak v sériovém portu.
Syntaxe:
var = WAITKEY()
var = WAITKEY(#channel) (při softwarovém UART)

WAITMS
Čekej určený počet milisekund.
Syntaxe:
WAITMS x

WAITUS
Čekej určený počet mikrosekund.
Syntaxe:
WAITUS x

WRITEEEPROM
Zapíše proměnnou do EEPROM na určenou adresu.
Syntaxe:
WRITEEEPROM var , adresa

#IF
Podmíněné větvení

2.3.2.4 **Programování AVR atmel**

Programování je proces, kdy se zkompilovaný soubor *.hex (viz str. 24), nahrává do jednotlivých pamětí (viz str. 19).

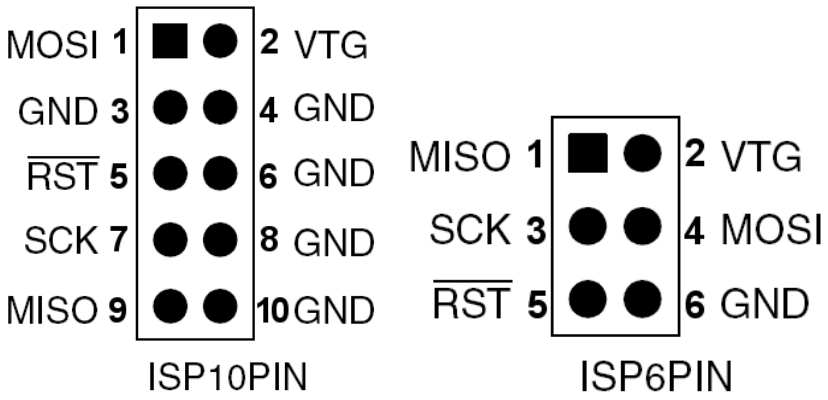
Pro naprogramování je nutné mít pohromadě tři věci, které jsou vzájemně kompatibilní. Jsou to programátor, mikroprocesor a obslužný SW. [3]

2.3.2.4.1 Programátor

Programátor je zařízení, které dokáže převést některé z rozhraní PC na některé z rozhraní podporovaných mikroprocesorem například ISP nebo JTAG. [3]

2.3.2.4.1.1Programovací rozhraní

Rozhraní ISP (In System Programming) na Obr.č. 20 je jedinečné v tom, že není nutné procesor vyndat z aplikace. Jedná se o sériové rozhraní. Které umožňuje programování, čtení a verifikaci přímo v aplikaci. [3]



Obr.č. 20 Vývodové uspořádání konektorů ISP. Převzato a upraveno dle [3]

2.4 Metody měření EKG u hmyzu

Měření EKG hmyzu je možné provést několika způsoby:

2.4.1 Optická metoda

2.4.2 Termografická metoda

2.4.3 Elektrická metoda

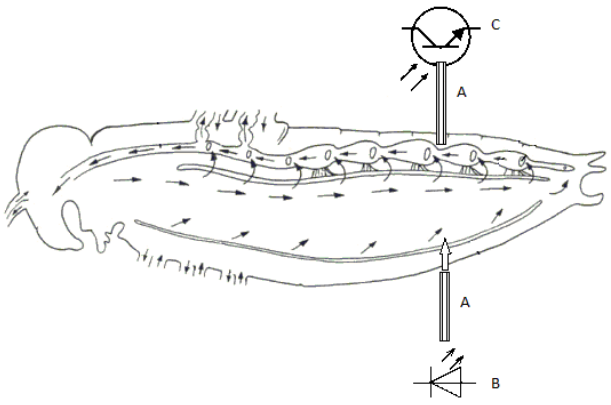
2.4.3.1 EKG

2.4.3.2 EEG

2.4.3.3 EPG

2.4.1 **Optická metoda**

Optické měření, znázorněné na obrázku Obr.č. 21, spočívá v průchodu paprsku pohybující se tkání, která mění svou schopnost absorbovat světelný paprsek. Dochází tak ke změnám intenzity procházejících paprsků, které jsou vhodně přeměněny na informaci optoelektrickým senzorem a následně přeměněny na binární informaci AD převodníkem. Vhodný převod dat do digitální podoby dovoluje snadnější zpracovávání naměřených údajů a dává také vyšší přesnost. [15]



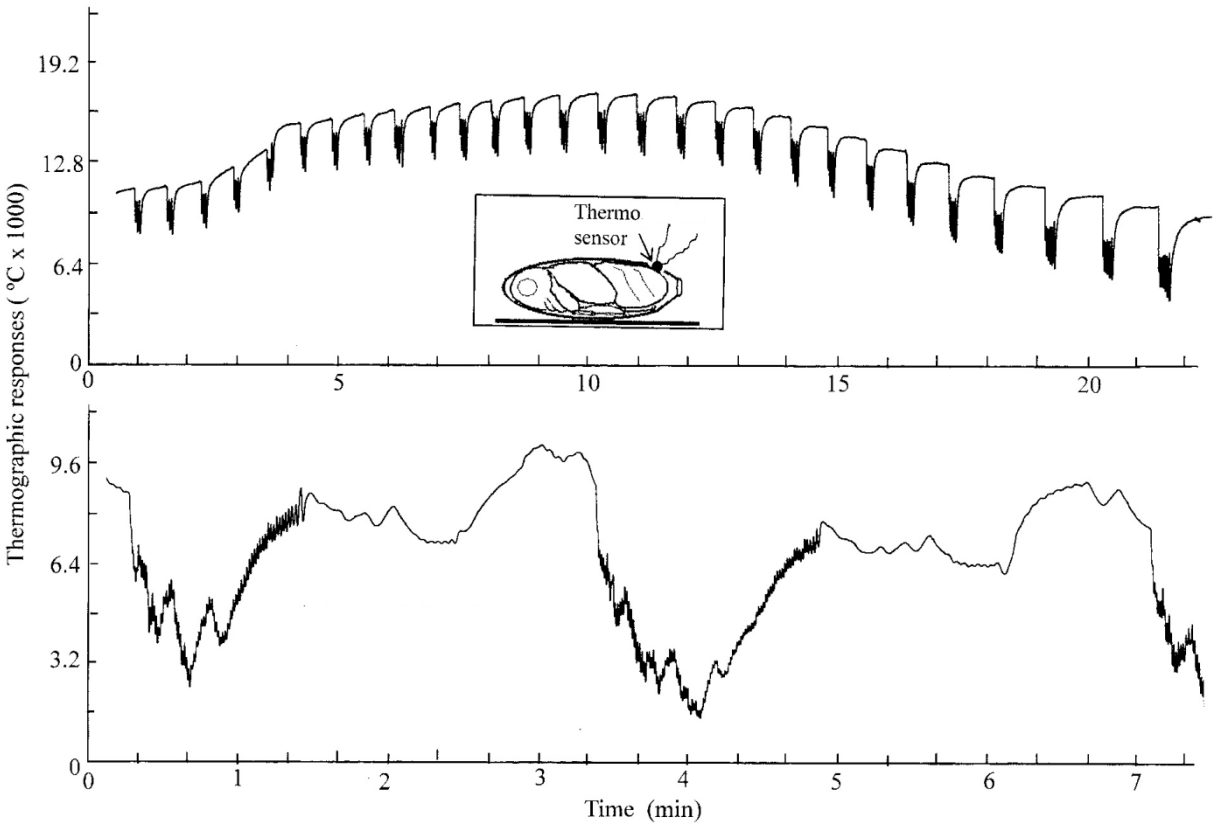
Obr.č. 21 Příklad použití optické metody: optické vlákno (A); LED dioda (B) a fototranzistor (C). Převzat a upraveno dle [16]

Tato metoda dává dobré výsledky, ale je poměrně problematická z hlediska přípravy před samotným měřením. Měřený objekt je nutné upravit. Nejprve dochází ke snížení aktivity pomocí CO^2. Následuje odebrání některých tkání pokožky, aby nebránily průchodu paprsku. Problém této metody spočívá v tom, že pokožka musí být dostatečně transparentní, aby bylo měření možné. Dále jsou zde úskalí této metody způsobené fyziologií hmyzu, měřená data mohou být zkreslena dalšími pohybujícími se tkáněmi v těle, což může vést k nesprávné interpretaci naměřených dat. [15]

2.4.2 **Termografická metoda**

Princip metody spočívá v umístění malého termistoru, zpravidla o velikosti v řádu stovek mikrometrů, do těla jedince. Termistor je vyhříván procházejícím proudem o jednotky °C nad teplotu okolí, čímž vznikne teplotní spád v okolí termistoru.

Srdeční rytmus, potažmo pohybující se tkáně, mění velikost tohoto teplotního spádu, což způsobuje změny procházejícího proudu, které jsou převáděny vhodným zesilovačem a AD převodníkem na digitální informaci a zpracovávány vhodnými algoritmy. Průběh tohoto měření a umístění tenzometru je patrné z obrázku Obr.č. 22. [15]



Obr.č. 22 Příklad umístění termistoru a data naměřená termografickou metodou. Převzato a upraveno z [15]

Výhodou této metody je, že v případě umístění termočlánku nad perikardiální oblastí dochází k přímému měření změn rychlosti proudění hemolymfy.

Tato metoda má však své nevýhody. S ohledem na nezanedbatelné tepelné kapacity termistoru a malému teplotnímu spádu zde nastává velká frekvenční závislost měřícího systému, který se zde chová jako dolní propust. [15]

2.4.3 **Elektrická metoda**

Využívá elektrod v bipolárním gelu a snímání elektrických potenciálů na přesně určených místech na biologickém objektu. Tyto elektrické signály jsou zesilovány vhodným zesilovačem a zaznamenávány.

Tímto způsobem lze řešit velké množství fyziologických signálů. U člověka je to např. EKG a EEG tedy Elektrokardiogram a Elektroencefalograf. [17]

2.4.3.1 **EKG**

Elektrokardiogram spočívá v měření signálů, které produkuje srdeční sval při práci. Z jeho časového průběhu lze odvodit, v jakém stavu je měřená osoba. Toto se provádí na základě definovaného fyziologického normálu, který byl za tímto účelem stanoven. [17]

2.4.3.2 **EEG**

Elektroencefalograf je přístroj, který se skládá z elektrod, zesilovače, filtru HP na frekvenci okolo 1Hz a záznamového zařízení. Slouží k záznamu nervové aktivity mozkové kůry.

Vyhodnocení opět probíhá na základě předem stanovených fyziologických normálů. Jejich komparace se záznamem definuje deviace, které odborník dokáže dále vyhodnotit. [18]

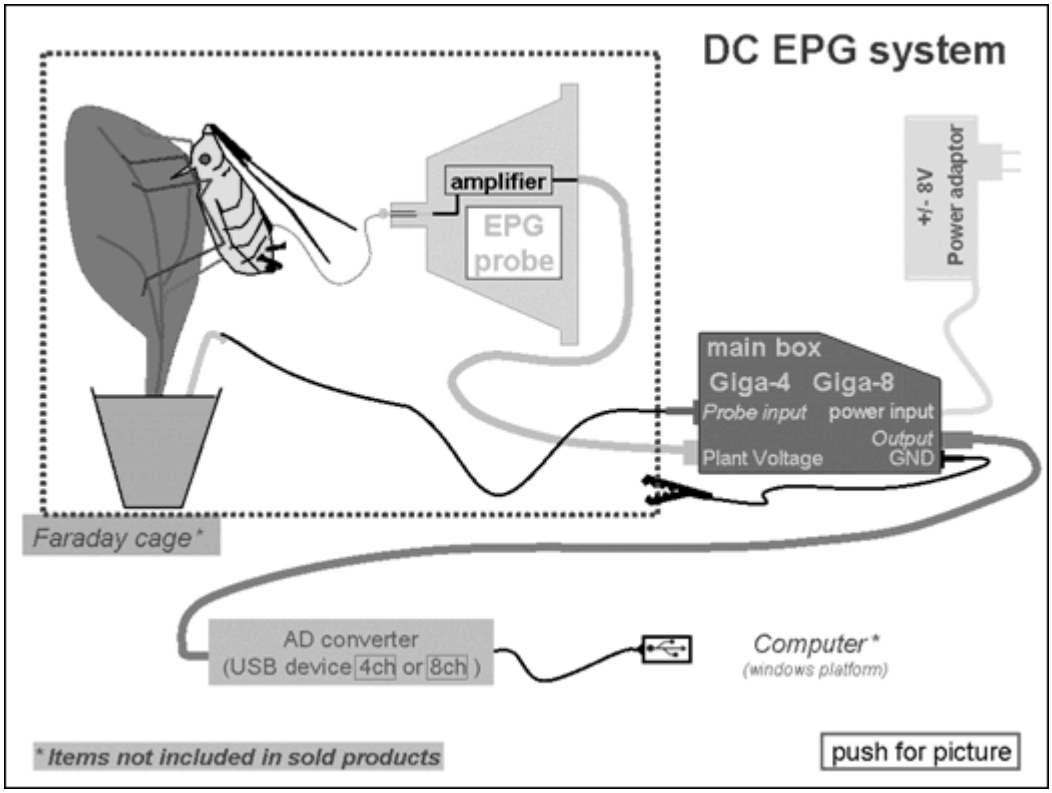
2.4.3.3 **EPG**

Elektronický penetrační graf je výzkumná technika pomocí níž se monitoruje krmení hmyzu, které probíhá propíchnutím. Hmyz, parazitující na rostlině nebo na teplokrevném živočichu, generuje elektrické signály, které lze zaznamenávat a následně poměrně dobře vyhodnocovat.

EPG se využívá pro vývoj nových plodin odolných proti škůdcům. [19]

2.4.3.3.1 Měřící metoda EPG

Měřenou soustavu tvoří rostlina a parazitující hmyz. Elektrody jsou dvě, umístěné na těle parazitujícího hmyzu a v kořenovém systému rostliny. Měřící systém a umístění elektrod na měřených objektech znázorňuje Obr.č. 23. [19]



Obr.č. 23 Systém pro měření EPG. Převzato a upraveno dle [19]

3 Praktická část

3.1.1 Připojení přípravků k PC [20]

Signály z měřených objektů jsou zesíleny a následně digitalizovány pomocí AD převodníku a tato data jsou přenášena různými rozhraními např. pro propojení osciloskopu lze použít GPIB rozhraní nebo sériovou linku popřípadě USB. Ze zde navrhovaného dataloggeru se data přenášejí pomocí rozhraní I2C a jsou převáděna na USB.

3.1.2 Konstrukce měřícího přípravku pro elektrické měření EKG u hmyzu [21], [22], [23]

Měření elektrických signálů na biologických objektech je problematické z mnoha důvodů. Budou zde popsány ty nejzásadnější, vyplývající z potřeb pro tento úkol.

Jeden z problémů je ve snímání elektrického potenciálu přímo na měřeném objektu, tedy elektrody. Elektrody, kterými je snímáno napětí, vnášejí chybu měření.

Kontakt s měřeným objektem musí mít co nejnižší odpor. Elektrody musí být vyrobeny z identického materiálu stejným postupem.

Elektrody se dotýkají objektu a přenos nosičů náboje je zajišťován elektrolytem, odtud vyvstává problém s elektrochemickým potenciálem mezi nestejnými elektrodami, nebo umístěním elektrod v nestejných elektrolytech či elektrolytech sice stejných ale s rozdílnou koncentrací kladných a záporných iontů, který vnáší chybu často řádově větší než je úroveň samotného měřeného signálu. Proto se zde užívá elektrod, které potlačují tyto vlivy (např. platinové elektrody, nebo elektrody, vyrobené ze stříbra). Pro zvýšení vodivosti se používá např. indiferentní vodivý gel. Tento gel se využívá nejen z důvodů zvýšení vodivosti, ale hlavně aby byly elektrody umístěny v roztoku se stejnou koncentrací iontů v celém jeho objemu. Také zde dochází k přemisťování iontů mezi indiferentním gelem a elektrolytem až do vyrovnání koncentrací, dále dochází k vysychání gelu, což logicky ovlivňuje koncentrace iontů.

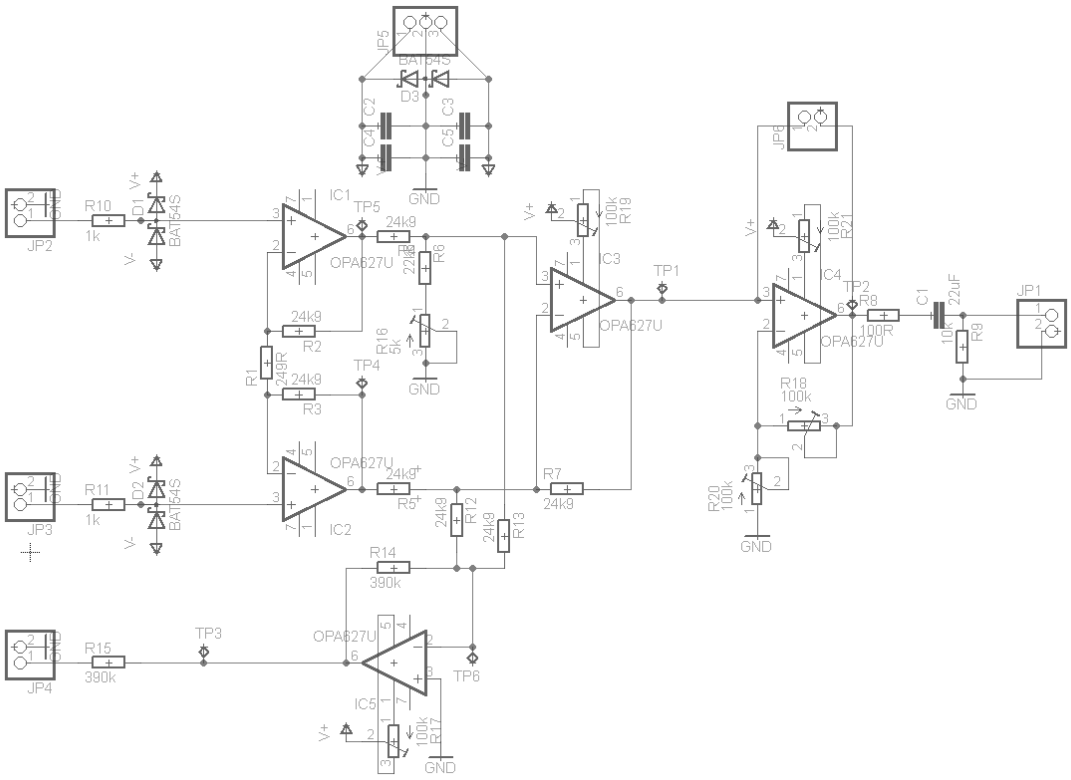
Je zde nutné zahrnout také termoelektrické vlivy, které mají opět spojitost s použitím různých materiálů, protože v elektrotechnické technologii se používá měď jako hlavní kov pro spoje a slitiny cínu z technologických důvodů v kombinaci se stříbrem, nebo platinou a pravděpodobně dojde k rozdílné teplotě elektrody a měřícího obvodu, což nutně vede ke změně potenciálů a tato chyba je dále zesilována.

Při konstrukci přípravku pro měření biosignálů se předpokládá, že tyto signály budou mít velmi malé napětí v řádech mV, a biologický objekt se bude chovat, jako zdroj s velmi velkým vnitřním odporem řádově GΩ, je tedy nutné vytvořit zesilovač s, pokud možno, řádově mnohonásobně větším vstupním odporem, proto byly zvoleny operační zesilovače Burr-Brown OPA 627 se vstupním odporem dle katalogu [24] $10^{13}\Omega$. Bohužel takto velký vnitřní odpor zdroje nese další úskalí v kapacitě vstupů samotných operačních zesilovačů dle [24] udávané 8 pF a v kapacitě vodičů. Tedy vznikne filtr dolní propust, který má dělící frekvenci v řádech Hz.

Dále se zde uplatňují rušivé napětí, naindukované na vedení s vysokou impedancí, i když ty lze velmi dobře potlačit správnou volbou zapojení zesilovače u takto velkého zesílení a takto vysokého vnitřního odporu zdroje signálu se stejně projeví dosti značně, zvláště v prostředích kde je toto rušení vyšší.

3.1.2.1 **Návrh měřícího zesilovače**

Problém byl posouzen z anatomického a neurologického hlediska, fyziognomie hmyzu dle [15], [22], [24], [25]. Dále z hledisek analýzy povahy zdroje měřených signálů, uvedené v kapitole 3.1.2 (str. 48). Po důkladném rozboru bylo rozhodnuto, že elektrické EKG bude konstruováno za pomoci operačních zesilovačů a bude zde použito zapojení přístrojového operačního zesilovače v diskrétní podobě se zavedením umělého zemního potenciálu, reprezentovaného obvodem IC5. Toto zapojení bylo shledáno nejvhodnějším z hlediska potlačení souhlasných signálů a další výhody viz [23]. Zapojení zesilovače ozřejmí schéma na Obr.č. 24. [23], [21],



Obr.č. 24 Zapojení zesilovače elektrických signálů na měřeném objektu. [23]

3.1.2.1.1 Použité součástky [23]

Rezistory byly vybrány kvalitní s nejnižším dostupným tepelným šumem a přesností 0.1%.

Trimry byly vybírány hermetizované cermentové víceotáčkové.

Kondenzátory byly vybrány keramické bezindukční.

Operační zesilovač OPA627 byl nakonec zvolen i přes svou cenu, z důvodu výborných parametrů:

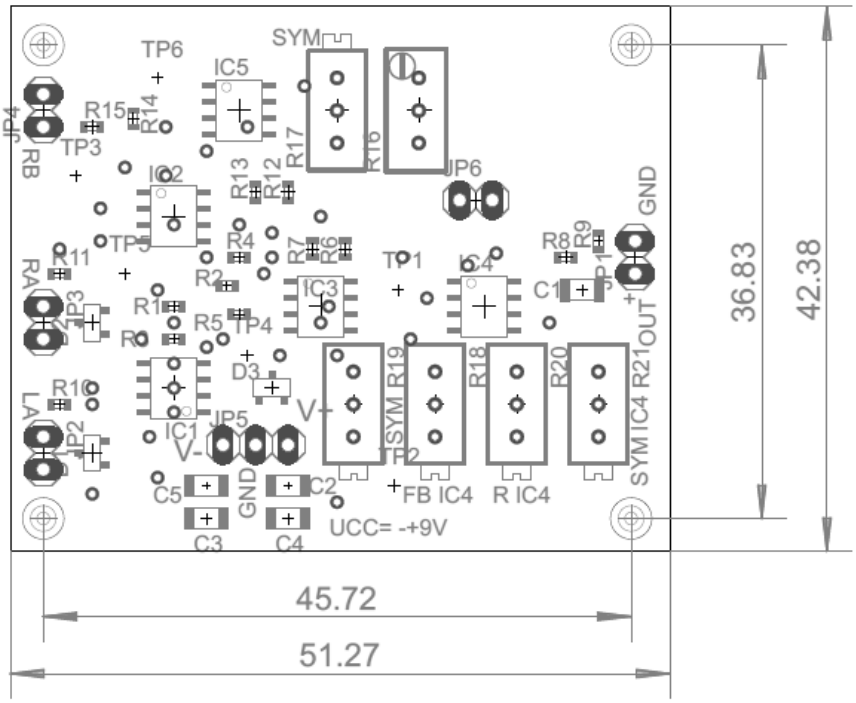
- Šum oz – 4.5 nV/Hz-2 MAX
- Krátký čas ustálení – 550 ns na 0.01%
- Nízký tepelný drift - 0.8 uV/°C MAX

Pro digitalizaci měřených dat bylo nutné použít digitální osciloskop.

Pro napájení byl použit zdroj s nízkým zvlněním

3.1.2.1.2 HW konstrukce zesilovače [24]

Zesilovač musel být navržen s ohledem na velkou rychlost ustálení OZ OPA627 aby nedošlo ke kmitání zesilovače, tomu bylo podřízeno celkové rozmístění součástek na desce plošných spojů, která je vyobrazena na Obr.č. 25.

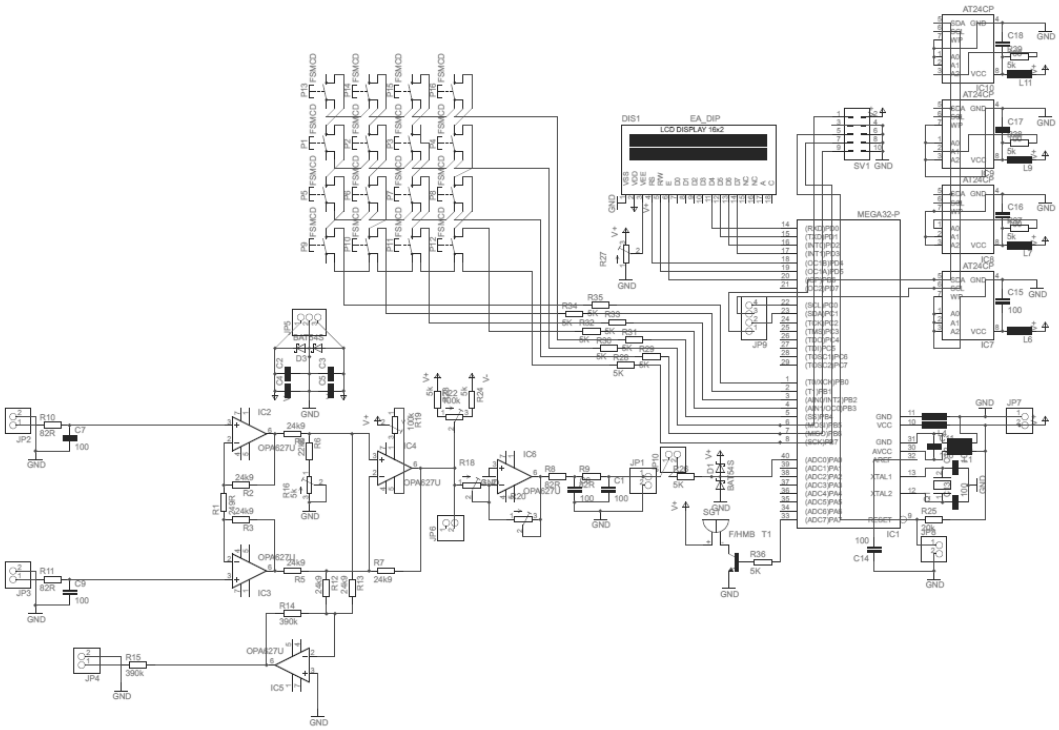


Obr.č. 25 Hard Warova konstrukce zesilovače.

3.1.2.2 **Návrh dataloggeru s měřícím zesilovačem [2]; [3]; [4]; [5]; [10]; [11]; [13]; [14]; [23]**

V návrhu dataloggeru je hlavním článkem mikroprocesor ATmega35. Tento jednočipový procesor byl nakonfigurován tak, aby byl schopen obsloužit 1x AD převodník 10 bit, display, klávesnici, a 4x 256 kbit pamětí, adresované sběrnicí I^2C.

Požadovaná funkce: datalogger zaznamenává ve zvoleném časovém rastru informaci o velikosti napětí na vstupu zesilovače. Při ukončení záznamu, nebo nenadálém odpojení napájení je konec záznamu označen ukončovacím bajtem, v němž je zapsána informace o rychlosti vzorkování a sekvence bitů, jež označuje konec záznamu. Datalogger je vybaven konektorem pro připojení programátoru, který dokáže vyčítat data přímo z pamětí, po uložení v PC lze data zpracovávat dle potřeby. Schéma na *Obr.č. 26* vyobrazuje datalogger, zesilovač a paměti.

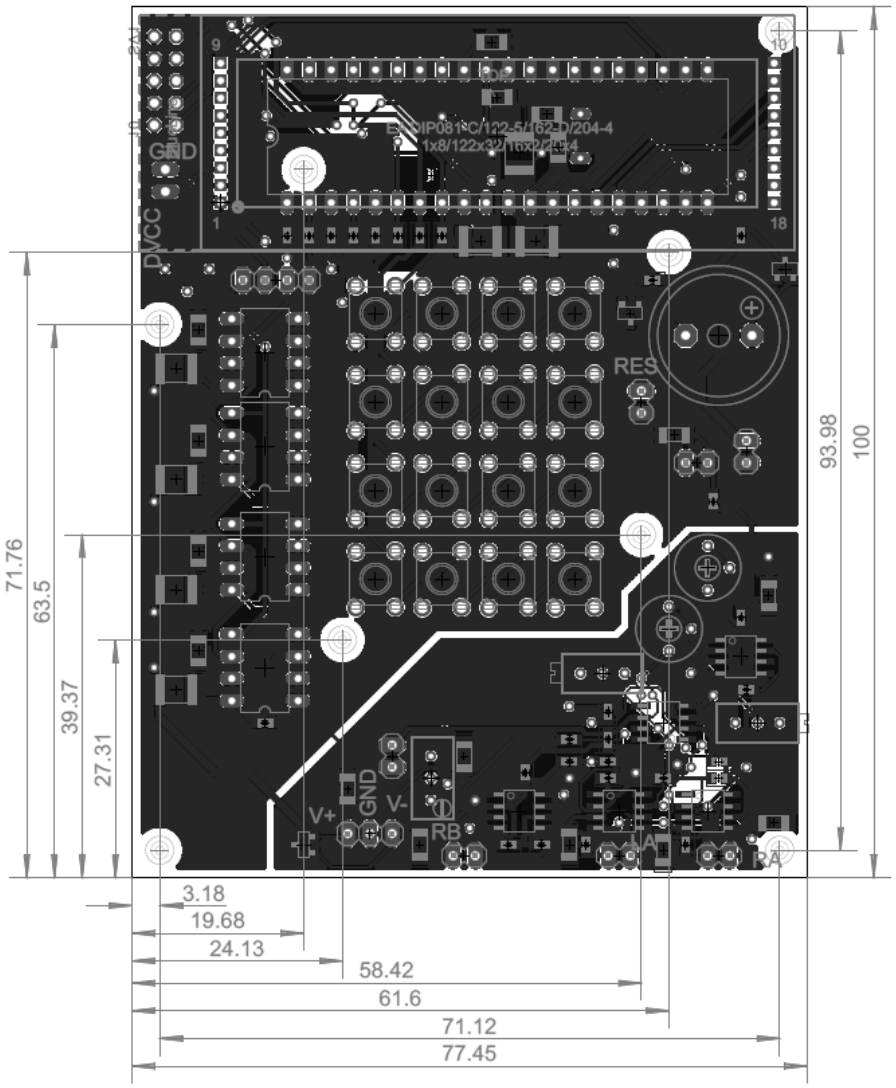


Obr.č. 26 Kompletní schéma dataloggeru se zesilovačem.

Datalogger je vybaven sirénou, která může oznámit zaplnění paměti. Dále je zařízení vybaveno konektorem pro programování mikroprocesoru v aplikaci ISP v deseti pinové variantě a maticovou klávesnicí.

3.1.2.2.1 Hardwarové řešení dataloggeru

Deska plošných spojů byla konstruována tak, aby byl eliminován vliv rušení procesoru do vstupů zesilovače. Dále bylo zařízení navrženo vhodným způsobem z hlediska ergonomie, tlačítka a display byly umístěn vhodným způsobem. Návrh desky plošných spojů se nachází na Obr.č. 27, jeho schéma je k nahlédnutí na obrazu Obr.č. 26(str.52).



Obr.č. 27 Mechanická realizace desky plošných spojů dataloggeru se zesilovačem.

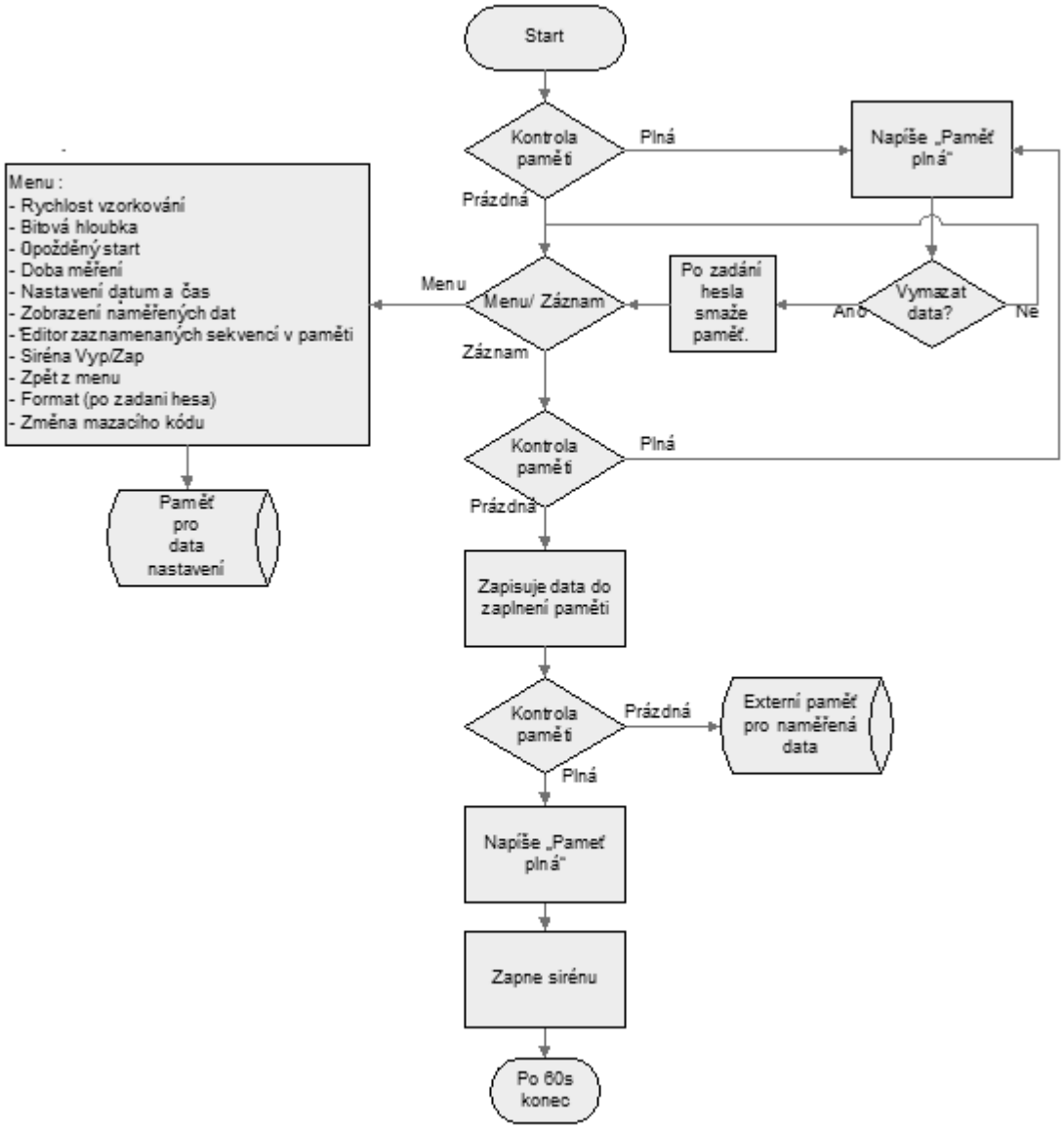
3.1.2.2.2 Návrh software,

Tato kapitola, věnující se návrhu softwaru, popisuje návrh programu a jeho funkce.

Po spouštění programu naběhne nabídka, ve které je uživatel vyzván k výběru mezi konfigurací dataloggeru, nebo přímo k zahájení záznamu. V případě skoku do Menu s nastavením bude uživateli nabídnuto několik možností konfigurace přístroje.

- Rychlost vzorkování
  Na alfanumerické klávesnici lze zadat počet vzorků za jednotku času Sp/s
- Bitová hloubka
  Uživatel si zvolí, kolik bitů se má zaznamenat min – 6bit, max – 10bit.
- Opožděný start
  Uživatel zvolí, o kolik sekund se opozdí začátek nahrávání.
- Doba měření
  Uživatel si zvolí jak dlouho má měření trvat. Pokud je zadaná „0" pak bude probíhat měření do vyčerpání paměti.
- Nastavení datum a čas
  Uživatel si před měřením nastaví datum a čas, který bude identifikovat záznam.
- Editor zaznamenaných sekvencí v paměti
  Tato funkce umožňuje mazat a pojmenovávat jednotlivé měřící sekvence.
- Siréna zap/vyp
- Zpět z Menu
  Vyskočí z menu k volbě nahrávání
- Format
  Po zadání hesla vymaže celou paměť
- Změna mazacího kódu
  Umožňuje změnit kód pro Format

Funkční diagram obslužného software na Obr.č. 28 jasně znázorňuje běh programu.



Obr.č. 28 Funkční diagram obslužného software.

3.1.2.2.1 Práce s pamětí

Před každým novým blokem dat bude zaznamenáno 8 bitů, které jednoznačně označí začátek záznamu. Po něm bude následovat 8-bitové slovo, v němž bude uchovávána informace o názvu, datové hloubce a rychlosti vzorkování. Následuje 16-bitové slovo s uloženou informací o velikosti napětí. Po tomto slově následuje vždy ukončovací 8-bitové slovo, které je vždy přepsáno dalším slovem s informací o velikosti napětí. Toto je z důvodu jednoznačného označení konce sekvence.

3.1.3 **Test měřícího přípravku**

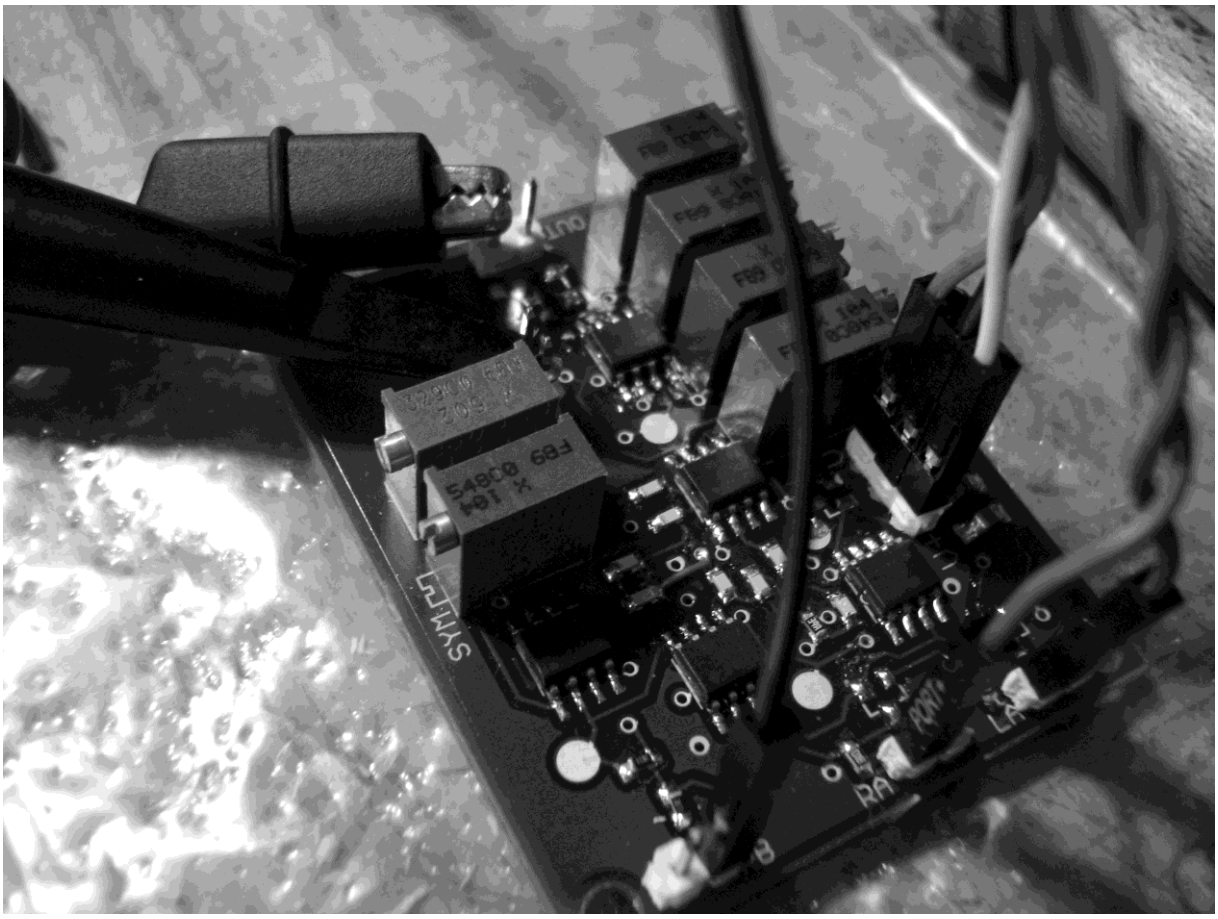
Měřící zesilovač byl připojen jak na larvě mouchy domácí, tak na člověku.

3.1.3.1 **Měření na lidech**

U člověka byly použity tři svody dle [17] a elektrody byly umístěny v indiferentním vodivém gelu. Průběh signálu je znázorněn na Obr.č. 29.

Obr.č. 29 Průběh signálu naměřený na člověku.

Měření bylo provedeno pomocí zapojení na Obr.č. 24 a zobrazení na osciloskopu Tektronix TDS210. Snímek zesilovače na Obr.č. 30.



Obr.č. 30 Zesilovač signálů EKG.

3.1.3.1.1 Hodnocení naměřených výsledků [17]

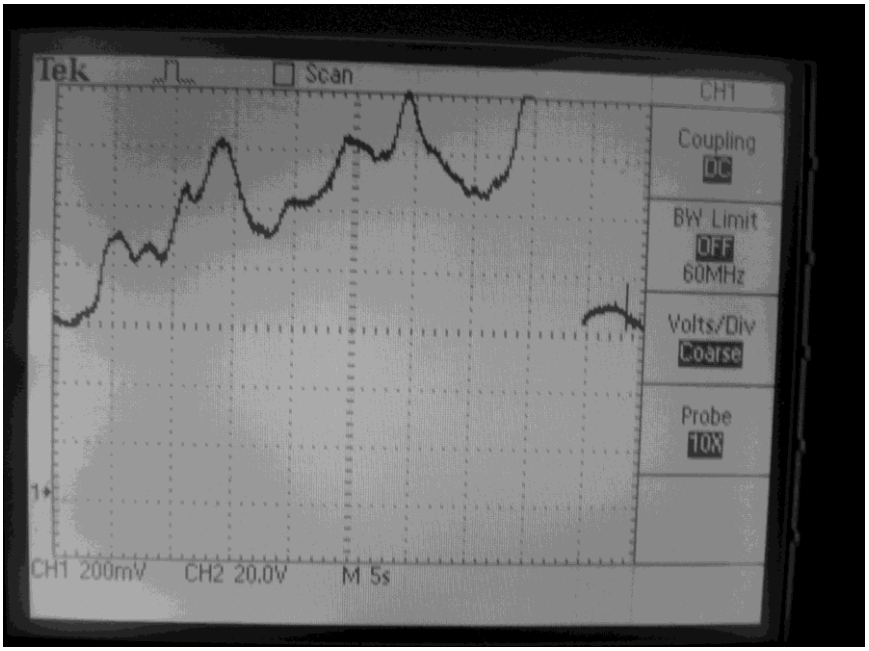
Záznam měření byl porovnán s literaturou [17] a bylo zjištěno, že naměřená křivka se neodchyluje od normy uvedené ve zdroji [17]. Tedy lze přípravek prohlásit za použitelný pro další testování na hmyzu.

3.1.3.2 **Měření na hmyzu [22], [21]**

Dále bylo měření provedeno na larvě mouchy domácí. Zkoumaný objekt byl taktéž připojen pomocí tří elektrod a indiferentního vodivého gelu. Z důvodu znehybnění byla podchlazena na teplotu cca 6°C. Měřící elektrody jsou zachyceny na Obr.č. 31. Aby bylo měření možné, musel být měřícímu zesilovači zvýšen zisk, čímž se začaly projevovat ve zvýšené míře chyby měření, popsané v kapitole 3.1.2 (str.48 48). Měření bylo opakováno a naměřený signál měl ve stejném časovém

rámci přibližně stejný tvar i amplitudu, z čehož lze teoreticky usuzovat, že jsou naměřená data relevantní, ovšem s jistotou to prohlásit nelze, jelikož zde mohlo nastat mnoho situací, které mohly měření zkreslit. Záznam z měření na Obr.č. 32 je vidět elektrický signál naměřený pomocí zesilovače.

Obr.č. 31 Připojení měřené larvy.



Obr.č. 32 Naměřený průběh na larvě.

3.1.4 **Problémy spojené s konstrukcí a měřením EKG**

Při měření jsme se potýkali s řadou problémů. V první řadě nalezení způsobu, kterým bychom hmyz dostali do stavu nepohyblivosti v průběhu měření.

Tohoto bylo po sérii pokusů dosaženo podchlazením Pelletierovým termoelektrickým článkem, jejž bylo nutné dostatečně chladit, aby si na studeném konci udržel teplotu okolo 6°C, která se ukázala jako optimální pro měření. Pokud nebyla larva uvedena do klidového stavu, nebylo možné vůbec měřit. Pohyb larvy je tak mocný s takovou vytrvalostí, že je jakýkoli způsob mechanické fixace naprosto zbytečný. Navíc je mechanická fixace nepoužitelná ještě z důvodu elektrických. Svaly zprostředkovávající pohyb vytvářejí signály s násobně vyšší amplitudou než je měřený signál, což zatíží měření takovou chybou, že jsou naměřená data nepoužitelná.

Při oživování zesilovače se ukázaly ještě další problémy a to se zpětnou vazbou, která vznikala špatně umytou deskou plošných spojů od použitého tavidla a problémy s napájecími zdroji, které vnášely do signálové cesty rušení, byl tedy vyvinout zdroj, který rušení nevnáší. Bylo nakonec rozhodnuto pro bateriový zdroj, jelikož veškeré pokusy o stabilní odstranění rušení za použití síťového zdroje se ukázaly jako nepoužitelné. Někdy se rušení objevovalo a někdy ne.

4 Závěr

Dle zadání bylo provedeno následující:

- zmapování historie programovatelných obvodů
- vysvětlena struktura mikroprocesorů AVR Atmel
- zpracována nabídka mikroprocesorů firmy Atmel podle výrobních řad a bylo jasně popsáno jaké má která řada vlastnosti
- popsána vývojová prostředí pro tvorbu programů, pro tyto mikroprocesory
- vybrány některé metody pro měření elektrických signálů u hmyzu a u člověka a tyto byli popsány
- vypsána některá použitelná rozhraní pro připojení měřících přípravků
- pro měření a vyhodnocení bylo nejprve nezbytné navrhnout měřící metodu, podle níž bylo konstruováno zařízení, které impedančně přizpůsobí záznamovou jednotku měřenému objektu
- byla provedena testová měření. Jednalo se o měření EKG na člověku a na hmyzu

Měření na člověku bylo porovnáno s literaturou a bylo zjištěno, že pro tento účel je měřící přípravek vyhovující.

Avšak naměřené hodnoty EKG pro larvu hmyzu není možné jednoznačně interpretovat. Dostupné měřící prostředky a finanční možnosti v této práci omezily podchycení všech chybových faktorů a mohly tak přispět k nižší relevantnosti výsledků. V závislosti na této skutečnosti by bylo vhodné dále se této problematice věnovat.

Citovaná literatura

1. **Marchalík, Jiří.** *Historie procesoru od pocátku až po soucasnost. Historie procesoru od pocátku až po soucasnost.* [PDF] Praha : autor neznámý, 2004.

2. Atmel AVR XMEGA vám poskytne eXtra více. *SOS ELECTRONIC S.R.O.* [Online] 11. únor 2011. [Citace: 10. červen 2014.] http://www.soselectronic.cz/?str=986.

3. **Matoušek, David.** *Práce s mikrokontroléry ATMEL AVR ATmega16.* Praha : BEN - technická literatura, 2006. ISBN 80-730-0174-8.

4. **Gadre, Dhananjay.** *Programming and customizing the AVR microcontroller.* New York : McGraw-Hill, 2001. ISBN 00-713-4666-X.

5. **Barnett, a další.** *Embedded C programming and the Atmel AVR.* Clifton Park : Thomson Delmar Learning, 2007. ISBN 14-180-3959-4.

6. **INTEL.** *INTEL.* 1970. INTEL. 4004 single chip 4-bit P-channel microprocessor.

7. **Olmr, Vít a Novák, David.** Atmel ATtiny 24/44/84 AVR. *Hw.cz.* [Online] Hw.cz, 2005. [Citace: 11. červen 2014.] http://www.hw.cz/novinky/atmel-attiny-244484-avr.html.

8. **Atmel Corporation.** *ATmega32: 8-bit Microcontroller with 32KBytes In-System Programmable Flash.* [PDF] místo neznámé : ATMEL, Atmel Corporation, 2011.

9. **Steffan.** Vysoké Učení Technické Brno. *Mikrokontroléry.* [Online] 24. duben 2009. [Citace: 11. červen 2014.] http://www.umel.feec.vutbr.cz/bdom/predn/Mikrokontrolery_1.pdf.

10. Atmel Corporation.AVR Assembler User Guide. [Online] [Citace: 3. červen 2014.] http://www.atmel.com/dyn/resources/prod_documents/DOC1022.PDF.

11. **Fargo, Luboš.** *SVETELEKTRO.COM.* [Online] 2013. [Citace: 13. červen 2014.] http://svetelektro.com.

12. **Bonfani, F., Monari, P. a Sampieri, U.** *Programming Metodology.* Fontaine : CJ International/Groupe AlterSys, 2001. IEC1131-3.

13. **Kutěj, Jaroslav.** STRÁNKY O MIKROPROCESORECH ATMEL. *mp222.* [Online] 2012. [Citace: 15. červen 2014.] http://mp222.wz.cz/index.php.

14. **Král, Břetislav.** Stránky o elektronice pro elektrotechniky. *Stránky o elektronice pro elektrotechniky.* [Online] 2012. [Citace: 12. červen 2014.] http://www.bretakral.eu.

15. **Sláma, Karel a DENLINGER, David L.** Transitions in the heartbeat pattern during pupal diapause and adult development in the flesh fly, Sarcophaga crassipalpis. *Journal of Insect Physiology.* 59, 2013, Sv. 8, stránky 767-780.

16. **Geiter, Henry B.** *E-Z ECG rhytm interpretation.* Philadelphia : F.A.Davis Company, 2006. ISBN 978-0-8036-1043-9.

17. **Šimek, Jiří.** *Jak číst elektroencefalogram: Základy praktické elektroencefalografie a stručnyý elektroencefalografický atlas.* Brno : SZdN, 1969.

18. **EPG-Systems.** Measuring Systems. *EPG-Systems.* [Online] 2013. [Citace: 6. červen 2014.] http://www.epgsystems.eu/systems.htm.

19. CIRCULATORY SYSTEM OF PERIPLANETA AMERICANA. *LEARN ZOOLOGY.* [Online] [Citace: 6. červen 2014.] http://learnzoology.files.wordpress.com/2013/12/cerculation-process.jpg.

20. **Vlach, Jaroslav a Vlachová Viktorie.** *Počítačová rozhraní přenos dat a řídicí systémy.* Praha : BEN - technická literatura, 2002. ISBN 80-730-0010-5.

21. **Kozumplík, Josef.** *Chemické zdroje proudu ve sdělovací technice.* Praha : Nakladatelství dopravy a spojů, 1981. ISBN 80-302-5326-8.

22. **Zahradník, Jiří , Chvála, Milan a Cuisin, Michel .** *La Grande encyclopédie des insectes: an outline for the use of students in entomological laboratories.* Paris : Gründ, 1989. str. 145. ISBN 978-270-0025-033..

23. **Punčochář, Josef.** *Operační zesilovače v elektronice.* Praha : BEN, 1996. ISBN 80-901-9843-0.

24. **TEXAS INSTRUMENTS, Incorporated [SBOS165,*].** Precision High-Speed Operational Amplifiers. [Online] 11. červen 2014. [Citace: 2014. červen 2014.] http://www.ti.com/lit/ds/symlink/opa627.pdf.

25. **Pechát, Jakub.** Anatomie hmyzu. *Hmyz.net.* [Online] 2007. [Citace: 1. červen 2014.] http://www.hmyz.net/anatomie.htm.

26. **Cendelín, Jiří .** Historie programovatelných automatů a jejich současné efektivní použití. *odbornecasopisy.cz.* [Online] 2013. [Citace: 6. červen 2014.] http://www.odbornecasopisy.cz/index.php?id_document=28831.