Středoškolská technika 2012

Setkání a prezentace prací středoškolských studentů na ČVUT

METEOROLOGICKÁ STANICE

Tadeáš Müller, Filip Marinkovič

SPŠST Panská

Panská 3, Praha1

Cílem tohoto projektu je sestrojení meteorologické stanice, která snímá teplotu, světlo a výsledná data posílá do počítače..

ANOTACE:

Jako náš absolventský projekt jsme si vybrali sestrojení meteorologické stanice, která snímá teplotu, světlo a výsledná data posílá do počítače. Našim úkolem bylo navrhnout a sestrojit vhodný hardware, napsat program pro mikroprocesor, který bude ovládat měřiče a zařídit komunikaci mezi počítačem a našim čipem. K propojení těchto dvou jednotek jsme využili sériový port. Hodnoty naměřené na čidlech se posílají do počítače, kde je posléze za pomoci odborného softwaru čteme. Důležité je zmínit, že světlo snímáme pomocí analogového měřiče a teplotu pomocí digitálního. K zařízení by se dala připojit i další čidla, ale oproti základním jsou poměrně finančně nákladná, a proto v naší práci máme pouze tyto dvě.

ANNOTATION:

As our graduation project we chose to construct a meteorological station that senses temperature, light and sends the resulting data into the computer. Our task was to design and construct an appropriate hardware, write a program for the microprocessor which will control measurement units and manage communication between computer and our chip. To connect these two units, we used the serial port. The values measured at the sensors are sent to the computer and with the help of specialized software readed afterwards. It is important to mention that light is measured with an analog meter and temperature with digital. The device could be connected to more sensors, but they are quite expensive compared to the basic ones, so in our work, we have only these two.

Obsah:

1.Úvod

V této dokumentaci Vám popíšeme jak realizovat tento projekt. Nejdříve se podíváme na jednotlivé součástky a řekneme si, co vlastně dělají. Dále rozebereme samotnou desku zařízení a její výrobu. Důležitá část je software, takže si nejdříve ukážeme, jak dostat programy do mikroprocesoru a jak ovládat jednotlivá čidla. Poté se dostaneme k přenosu dat do počítače a nakonec k vytvoření hlavního programu.

2.Hardware

V této pasáži se podíváme na vybavení naší stanice a její konstrukci. Deska je poměrně jednoduchá a neměl by být problém ji podle návodu sestrojit.

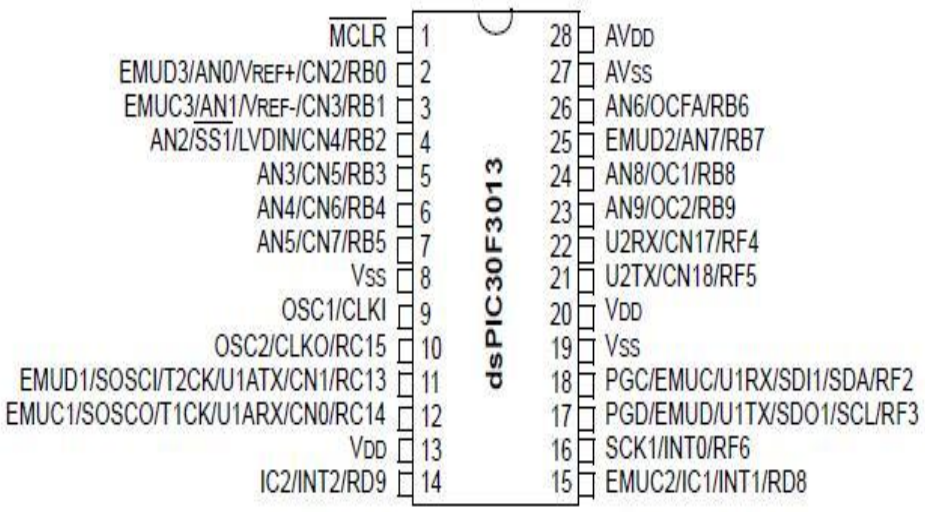
2.1 Popis, funkce základních součástek

Před popisem samotné desky je důležité pochopit, jak jednotlivé součástky pracují a jaký je vlastně jejich úkol. Rozebereme si pouze ty nejdůležitější, abychom dále pochopili princip celé desky.

2.1.1 Mikroprocesor DSPIC30F3013

Obr. 1: Jednočipový procesor

Je poměrně výkonný 16 bitový Digitální Signální Procesor (DSP) firmy Microchip. Může dosahovat výkonu až 30MIPS (Mega Instructions per Second). My využíváme vnější oscilátor 7,3728 MHz, který je vynásoben násobičkou 8krát. Dostáváme se tedy na výkon 14,7456 MIPS. DSPIC30F3013 má 28 pinů, je napájen 5V stejnosměrného napětí. Mezi jeho význačné vlastnosti patří programovatelnost v jazyce C, 84 základních instrukcí s flexibilním adresováním, vnitřní a vnější zdroje přerušení, dva 16 bitové časovače, adresovatelné UART moduly s FIFO buffry, 12 bitový A/D převodník, 1 sample/hold, vlastní přeprogramování pod kontrolou softwaru, možnost programování paměti pomocí ICSP (In circuit serial programing) a RTSP (Real Time Self programing), nízká spotřeba energie, vysokorychlostní Flash technologie. V naši práci hraje DSPIC30F3013 hlavní roli, slouží k ovládání čidel, komunikaci s počítačem a upozorňuje nás na vzniklé chyby za pomoci Led diod.



Obr. 2: Pin diagram DSPIC30F3013

Potřebné piny:

- (1) MCLR - je připojen na tlačítko a slouží jako celkový reset PICA.
- (5, 6, 7, 26) RB3-6 - připojeno na LED diody
- (8, 19, 27) Vss - zem čipu

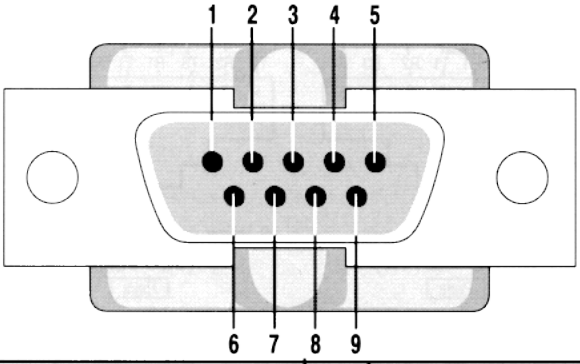
- (13, 20, 28) Vdd - napájení čipu +5V
- (24) RB8 - připojeno na data teplotního čidla Dallas, tedy pin 2
- (23) RB9 - připojeno na fototranzistor sloužící k měření intenzity světla
- (9, 10) OSC1-2 - připojen na vnější oscilátor 7,37 MHz
- (17) U1TX - vysílací pin mikroprocesoru, připojen na 11 pin MAX232
- (18) U1RX - přijímací pin mikroprocesoru, připojen na 12 pin MAX232

2.1.2 Sériový port RS232 (UART)

Obr. 3: RS232

Sériová linka se používá jako komunikační rozhraní osobních počítačů a další elektroniky. RS-232 umožňuje propojení a vzájemnou sériovou komunikaci dvou zařízení, tzn., že jednotlivé bity přenášených dat jsou vysílány postupně za sebou (v sérii) po jednom páru vodičů v každém směru. Jak jsme již mnohokrát říkali, RS232 u nás propojuje mikroprocesor s počítačem a přenáší naměřená data. Kabel je na jedné straně zakončen konektorem DE-9 F (samice) a na druhé straně jsou pouze vyvedené dráty. Sériový port je asynchronní, to znamená, že každé sekvenci datových bitů předchází jeden start bit, kterým se logická hodnota na lince přepne (původně v klidovém stavu) do opačného stavu. Po datových bitech následuje paritní bit a za ním jeden nebo více stop bitů, během kterých je linka opět v klidovém stavu. Logický stav „0“/„1“ přenášených dat je reprezentován pomocí dvou možných úrovní napětí, které jsou bipolární. Rychlost přenosu je u nás odvozena na 38400Bd. Je to dostatečně velká rychlost a zároveň nepůsobí problémy.

Vstupně/výstupní piny procesoru dsPIC samozřejmě používají běžné logické úrovně TTL logiky. Proto jsme desku osadili převodníkem úrovní RS232, známým obvodem MAX232.



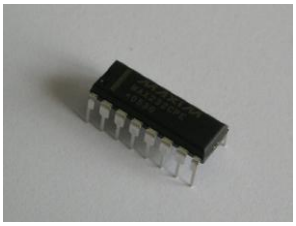
Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

Obr. 4:Pin diagram RS232

Potřebné piny:

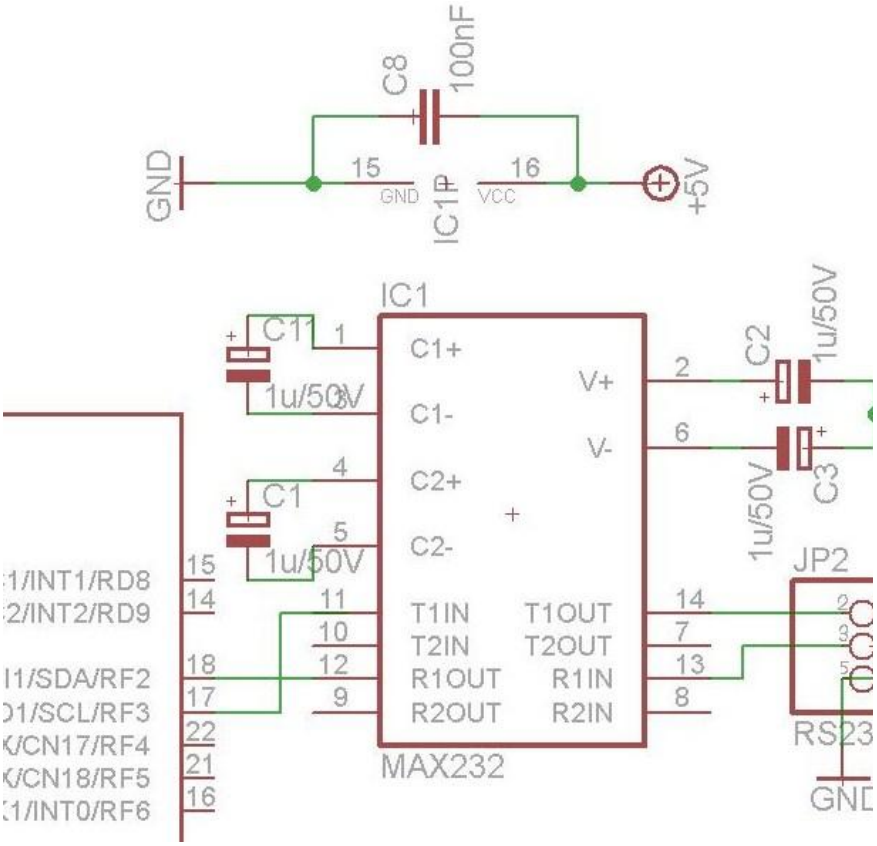
- (1,4,6) - tyto piny se musí mezi sebou navzájem propojit
- (2) RX - přijatá data, tento pin je připojen na MAX232, na jeho výstup (14)
- (3) TX - odeslaná data, tento pin je připojen na MAX232, na jeho vstup (13)
- (5) GND - zem

2.1.3 MAX232

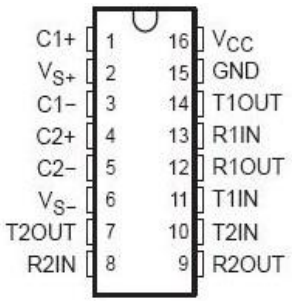


Obr. 5: Integrovaný obvod MAX232

Integrovaný obvod MAX232 obsahuje dva převodníky úrovní RS232 na TTL úrovně, tedy na 5V a 0V. A dva převodníky z TTL na RS232. Obsahuje také nábojovou pumpu a stačí mu napájení +5V. Obvod zvládá operace minimálně o rychlosti 120kbitů/s.



Obr. 6: Zapojení MAX232



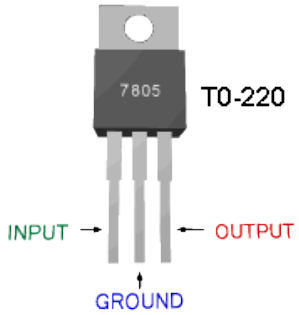
Obr. 7: Pin diagram

Potřebné piny:

- (16) Vcc - napájení +5V
- (15) GND - zem
- (14) T1OUT - připojeno na RS232 na pin (2) RX
- (13) R1IN - připojeno na RS232 na pin (3) TX
- (11) T1IN - připojeno na DSPIC30F3013 na pin (17) U1TX

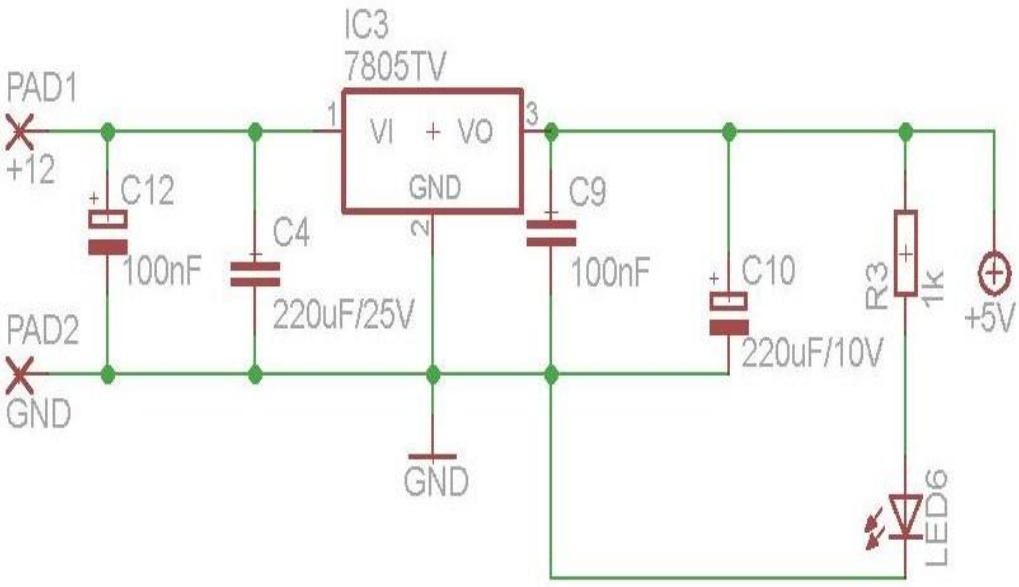
- (12) R1OUT - připojeno na DSPIC30F3013 na pin (18) U1RX
- (3, 1) C1± - připojeno na elektrolytický kondenzátor 1u/50V
- (4,5) C2± - připojeno na elektrolytický kondenzátor 1u/50V
- (2,6) - připojeno na 2 elektrolytické kondenzátory 1u/50V

2.1.4 Stabilizátor napětí 7805



Obr. 8: Integrovaný obvod 7805

V našem obvodu slouží k stabilizování napětí na +5V. Můžeme ho případně opatřit chladičem. Vstupní napětí nesmí překročit 35V.



Obr. 9: Schéma zapojení 7805

2.1.5 Led diody

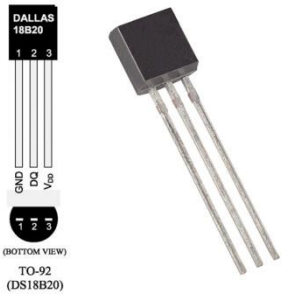
Obr. 10: Led dioda

Dioda je elektronická polovodičová součástka obsahující přechod P-N. Na rozdíl od klasických diod, LED vyzařuje viditelné světlo. Prochází-li přechodem elektrický proud v propustném směru, přechod vyzařuje (emituje) nekoherentní světlo s úzkým spektrem. Při zapojování musíme před Led diodu dát odpor. Jeho velikost se spočítá dle vztahu:

$$R = \frac{Uo - Ud}{Imax} \qquad (1)$$

V naší práci používáme Led diody pouze pro větší přehlednost, dokážou usnadnit až několik hodin práce.

2.1.6 Dallas 18b20



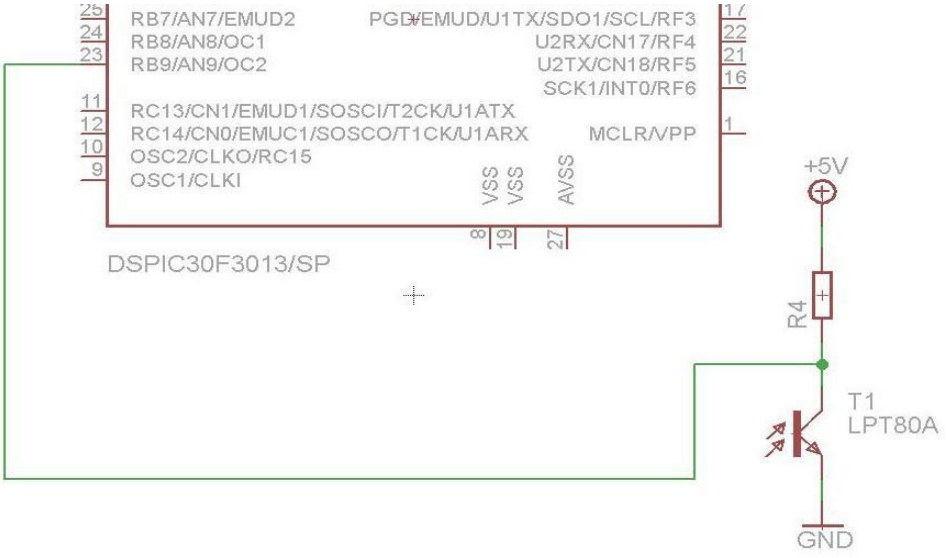
Obr. 11: Čidlo dallas

Teplotní měřič má tři piny zem, data, napájeni (+5V). Datový drát je zavěšen pull-up odporem na +5V. Tento odpor má velikost R= 4,7kΩ. Je to číslicový 9-12 bitový teploměr, který dokáže měřit teplotu od -55 do +125°C. Jeho přesnost může být až na 0,0625°C. V tomto projektu se spokojíme s přesností na stupně.

2.1.7 Fototranzistor

Obr. 12: Fototranzistor

Tato součástka je v naší práci nepostradatelnou, slouží k měření velikosti slunečního svitu. Fototranzistor je aktivní polovodičová součástka pracující s řízenými dvojicemi přechodů. Jedná se o obdobu klasického tranzistoru s tím rozdílem, že přechod je ovládán světelným zářením. Změna osvětlení nám způsobí změnu napěťové složky, kterou pomocí dsPIC snímáme a dále zpracováváme. Jako měřič slunečního svitu by se dala použít například i fotodioda.



.

Obr. 13: Schéma zapojení

Fototranzistor je zapojen jako dělič. Velikost odporu R4, záleží na typu fototranzistoru, a jak velkou změnu napětí chceme. Pro naše součástky, jsme dle vztahu:

$$U_2 = U * \frac{Rft}{R4+Rft} \qquad (2)$$

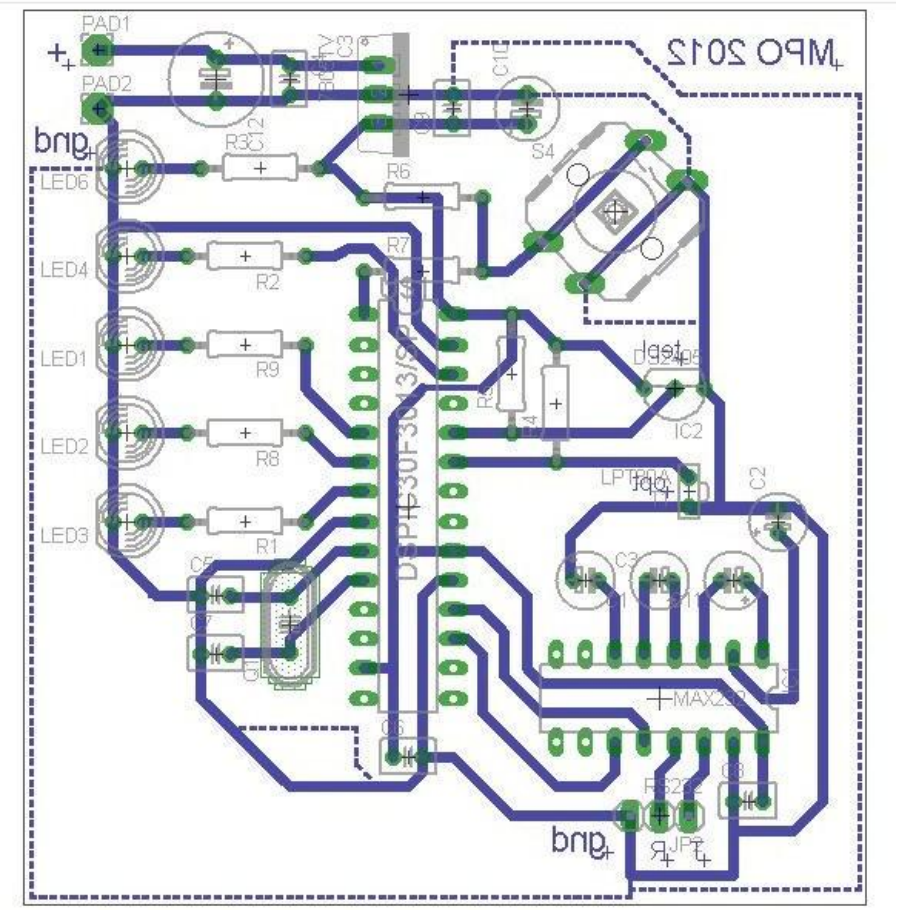
vypočítali velikost odporu R4 na 10kΩ, přičemž změna napětí bude dostatečná, aby s ní mikroprocesor zvládl pracovat.

2.2 Popis, funkce hardwarové části stanice

Vysvětlili jsme si funkce a činnosti nejdůležitějších součástek stanice, teď se podíváme na desku jako celek. Napájení vede ze sítě (230V) přes adapter (12V) do stabilizátoru, který dodává do desky 5V. Velmi důležité jsou kondenzátory C4, C6, C8, C9 o hodnotě 100nF, C12 o hodnotě 220uF/25V a C10 o hodnotě 220uF/10V, které slouží jako blokovací kondenzátory. Dále nesmíme zapomenout zapojit veškeré napájecí a zemnící piny mikroprocesoru a převaděče úrovní. Připojením Led diod do obvodu si zpřehledníme práci, jedna je připojena hned za stabilizátor a signalizuje nám, zda je napájení v pořádku. Další diody jsou připojeny na procesor a můžeme je využít dle libosti. Na čip je dále připojen krystal o frekvenci 7,37MHz s 2 kondenzátory o velikosti 22pF. Tento vnější oscilátor využíváme z důvodu lepších výpočtu a jednodušších odvození. Jak jsme si již řekli na dsPIC jsou připojena čidla a tlačítko, které využijeme pouze při realizaci bootloadru, o kterém si povíme v další části. Pokud je tlačítko rozepnuté, pin MCLR (1) je přes pull-up odpor připojen na +5V, pokud tlačítko sepneme, pin MCLR (1) se spojí se zemí a dsPIC provede reset. Mikroprocesor má ještě spojené piny RX a TX s MAX232. Na tento integrovaný obvod je připojeno RX a TX z RS232.

Příloha č. 1

Pokud si obvod budete chtít vyrobit, budete potřebovat průsvitnou folii, na kterou si natisknete návrh desky. Fotodesku, na kterou necháte pomocí svítivé komory ozářit plošné spoje a nakonec páječku a cín k připájení součástek.



Obr. 14: Návrh plošného spoje

3.Software

V této pasáži se podíváme na programovou část stanice. Tedy jak nahrát program do mikroprocesoru, jak ovládat čidla, řekneme si nejdůležitější věci o sériové lince a nakonec si povíme jak zobrazit naměřené hodnoty v počítači. V naší práci používáme programovací jazyk C.

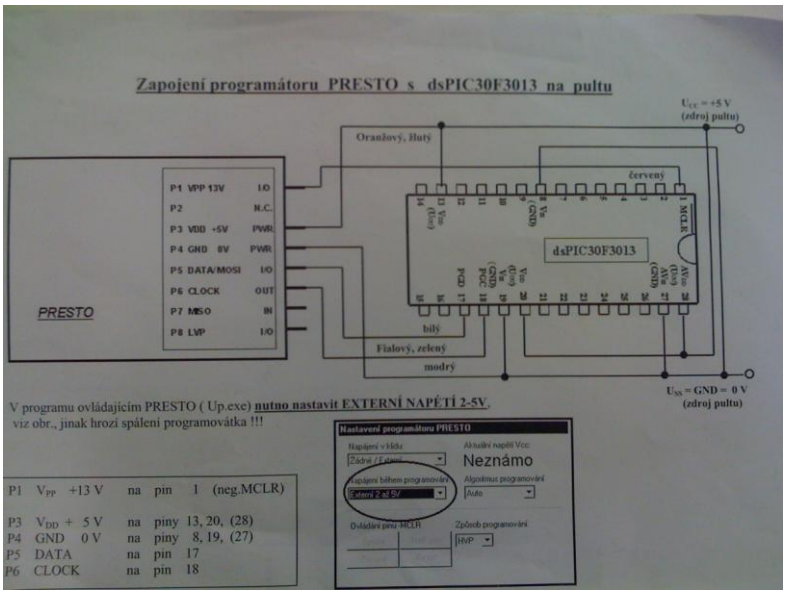
3.1 Programování DSPIC30F3013

Na začátek bychom si měli říct, jak vůbec do jednočipového procesoru nahrát napsaný program. Je více možnosti, ale my si povíme pouze o dvou, které jsme v naší práci vyzkoušeli.

3.1.1 Programátor PRESTO

Obr. 15: Asix: PRESTO

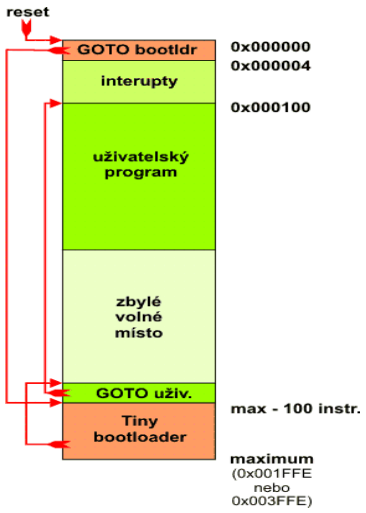
Programátor, se kterým jsme pracovali, nesl název PRESTO. Z našeho pohledu byl velice spolehlivý a jednoduchý. Pracuje na principu ICSP (In Circuit Serial Programming), která fungovala i u starších rodin procesorů PIC. Programování se provádí za pomoci pěti vodičů (Vcc, Vss, data, hodiny a Vpp) a pokud je zapojení aplikace navrženo vhodně, dá se provádět až přímo na desce osazené procesorem. Zapojíme tedy všechny napájecí a zemnící piny procesoru a připojíme PRESTO. Poté stáčí spustit vhodný program (Asix up) a naprogramovat procesor.



Obr. 16: Schéma zapojení programátoru

3.1.2 Bootloader

Na rozdíl od programátoru PRESTO bootloader využívá metodu RTSP (Real-Time Self-Programming). Metodou ICSP se do procesoru nahraje malý zavaděč, který je schopen zbytek programového vybavení sám zapsat metodou RTSP bez nutnosti mít programátor. Bootloader (zavaděč programu) nedělá nic jiného než upgrade firmwaru. U procesoru dsPIC totiž lze update firmwaru řešit napsáním malého programu, bootloaderu, kterému je schopna nový firmware poslat po sériovém portu odpovídající aplikace z počítače. Na internetu existuje spousta bootloaderů pro procesor dsPIC, ale my jsme používali dvojici program (PIC – PC) s názvem Tiny Bootloader. Je velmi jednoduchý, dobře promyšlený a v paměti procesoru PIC zabírá pouhých 100 instrukcí. Program sídlí v části paměti, kterou sám přepisuje metodou RTSP. Aby se nepřepsal, umístí se bootloader na úplný konec paměti programu. V případě Tiny bootloaderu tam tedy zabírá posledních 100 instrukcí, a proto maximální velikost našeho uživatelského programu, který budeme do procesoru chtít nahrát, nesmí teoreticky přesáhnout více jak 7900 instrukcí, ale je lepši počítat spíše se 7500 instrukcí. Aby se bootloader spustil, musí byt na adrese 0x000000 instrukce GOTO <adresa jeho začátku>. Bootloader skočí na adresu 0x000000, vždy po resetu, v našem případě po zmáčknutí tlačítka. Pro "uživatelskou" instrukci GOTO má vyhrazenou poslední paměťovou pozici před svým začátkem. Instrukce na adrese 0x000000 tedy vždy skáče na bootloader. A teprve bootloader po uplynutí nějakého časového intervalu, kdy se nic neděje, skočí na adresu těsně před svým začátkem, kde je instrukce GOTO <uživatelský program>. O to, aby se adresa startu správně přesunula (z pozice 0x000000 na pozici těsně před bootloader) se navíc stará počítačová část bootloaderu.



Obr. 17: Pozice usídlení Tiny bootloaderu

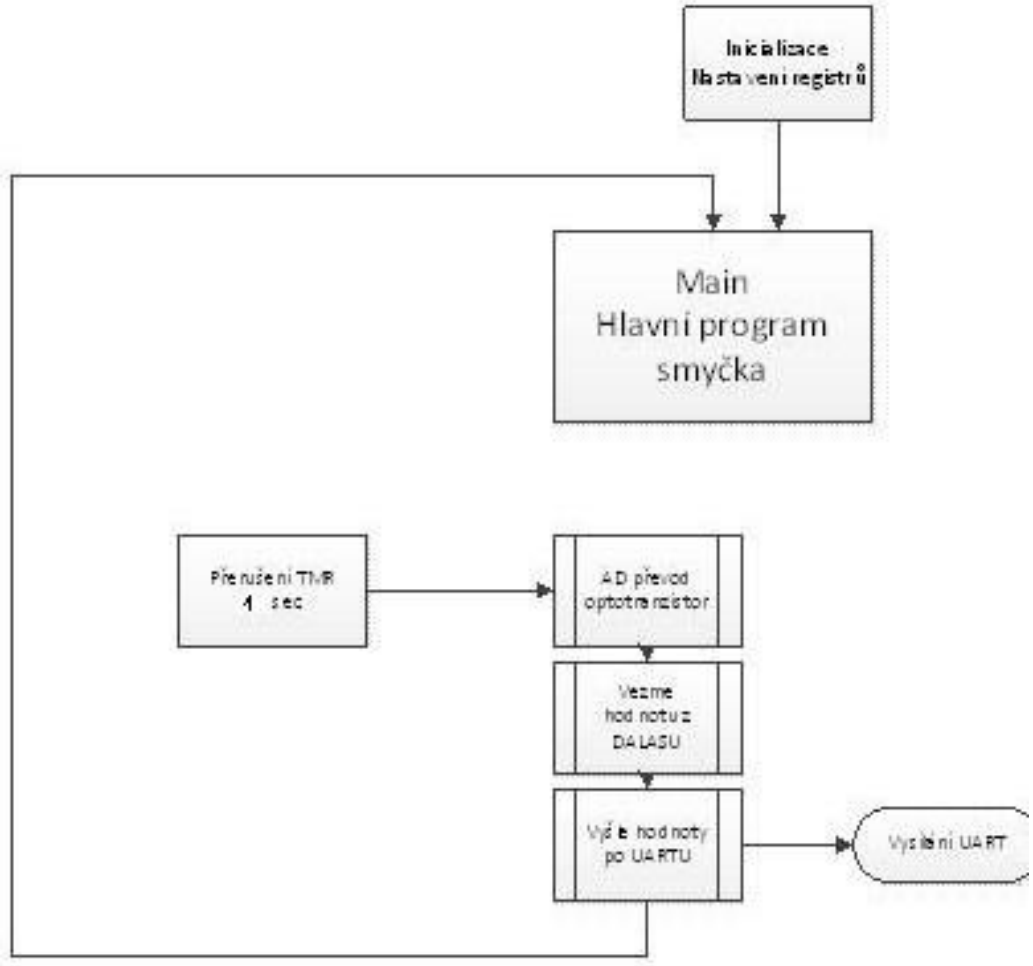
Pro úplnou funkčnost musíme bootloader nastavit pro náš mikroprocesor. Tedy nastavit správné piny sériového přenosu a jeho rychlost (38400Bd), nastavit hodiny (58,9824 MHz), nastavit timeout bootloaderu, po kterém se spustí uživatelský program (3s) a nastavit další konfigurační registry procesoru. Konkrétně se jedna o registr, který vysílá ID mikroprocesoru, dále se musí nastavit registr, kam se bootloader ukládá. V našem případě se nám bootloader nepovedlo dotáhnout do 100% funkčnosti a to z důvodu nedostatku času, kvůli odevzdání projektu. Nedokázali jsme odhalit chybu, ale pokud si s bootloadrem budete nějaký čas hrát, jistě se to vyplatí.

Tiny bootloader můžete bezplatně stáhnout na stránkách http://jakub.serych.cz.

```
.include "p30f3013.inc"

;config              __FOSC, CSW_FSCM_OFF & HS;FRC & FRC_PLL8
;config         __FBORPOR, PBOR_OFF & MCLR_EN
;Int FRC, PLLx8
config                        __FOSC, CSW_FSCM_OFF & XT_PLL8
config          __FWDT, WDT_OFF
config          __FBORPOR, PBOR_OFF & MCLR_EN
config          __FGS, CODE_PROT_OFF
;check also ATIO (below)

.equ XTFreq, 7372800                                    ;oscillator frequency
.equ PLLMode, 8                                                 ;PLL multiplier
.equ baud, 38400                                        ; desired baud rate
.equ BootLoaderDelay, 3000                      ;delay (ms) before bootloader times out

.equ Fcy, XTFreq*PLLMode/4                              ;calculate Fcy
.equ BRGH, ((Fcy/baud) / 16) - 1        ;calculate BRGH value for UART

.equ IdTypePIC, 0x92
.equ max_flash, 0x003FFE                ;dsPIC max address

.global __reset

.section .nbss, "b"
buff:       .space 4
buffer:     .space 128+6
W99:        .space 2
```

Obr. 18: Konfigurace význačných registrů

3.2 Hlavní program



Obr. 19: Vývojový diagram hlavního programu

Musíme nastavit čítač s periodou 1s:

```
T1CON = 0;              //Vynulujeme konfigurační registr časovače
    TMR1 = 0;           //Vynulujeme hodnotu časovače
    PR1 = FCY/256;      //Nastavíme periodu na 1 sekundy kvůli dallasu, aby stihnul
    T1CON = 0x8030;     //Zapneme časovač Timer1 s předděličem 1:256
```

Musíme povolit přerušení:

```
IFS0bits.T1IF = 0; //Nulování flagového bitu pro TMR1
IEC0bits.T1IE = 1; //Povolení přerušení při přetečení čítače 1
```

Je velmi důležité, aby přerušení a veškeré proměnné v něm používané byly mimo main.

3.3 Teplotní čidlo Dallas

Náš měřič je digitální, proto musíme nastavit pin procesoru (RB8) jako digitální vstup/vystup podle potřeby. Veškerá komunikace s procesorem funguje na principu tahání pull-up odporu. Nejdříve si vytvoříme základní časovač na 10us. Ve stavu klidu je na datovém drátě +5V. Při zahájení komunikace procesor musí maximálně na dobu 10us datový drát uzemnit.

Při vysílání (zápisu do dallase):

- log 0 - držím datový drát uzemněný minimálně dalších 45us
- log 1 - pustím drát, takže odpor ho zvedne na 5V a nechám ho tak minimálně 45us

Dallas si po cca 30us po zahajovacím uzemnění drátu přečte hodnotu, takže vidí, jestli jsem drát držel nebo pustil.

Při příjmu (čtení z dallase), po zahajovacím uzemnění dám pin do stavu vysoké impedance a sleduji, co se na něm děje (v PICu z něj TRISem udělám vstupní pin):

- log. 0 - drží po dobu cca 15us drát na zemi a čteme log 0.
- log. 1 - pustí drát téměř hned po zahajovacím pulzu, takže ho odpor vytáhne na 5V a čteme log 1.

První inicializační pulzy trvají kolem 500us. Vždy jedna strana (procesor nebo Dallas) tahá komunikační drát zavěšený odporem na 5V dolů k zemi a ta druhá strana sleduje, co se děje.

```
#define Convert_T 0x44
#define Read_scratchpad 0xBE
#define Port_18B20 PORTBbits.RB8  // upravit
#define Tx_18B20 TRISBbits.TRISB8 = 0 // upravit
#define Rx_18B20 TRISBbits.TRISB8 = 1 // upravit
unsigned temp;
unsigned short tempL, tempH ;

// dallas
char Reset_18B20() {
                    Tx_18B20; // Tris = 0 (output)
                    Port_18B20 = 0; // set pin# to low (0)
                    asmcekani480us(); // 1 wire require time delay
                    Rx_18B20; // Tris = 1 (input)
                    asmcekani60us(); // 1 wire require time delay
                    if (Port_18B20 == 0) { // if there is a presence pluse
                    asmcekani480us();
                    return 0; // return 0 ( 1-wire is presence)
                                                        }
                    else {
                    asmcekani480us();
                    return 1; // return 1 ( 1-wire is NOT presence)
                    }
```

Obr. 20: Reset (inicializace) dallasu

```
void Write_18B20 (char Cmd){
        char i;
        Rx_18B20; // set pin# to input (1)
        for(i = 0; i < 8; i++){
        if((Cmd & (1<<i))!= 0) {
        // write 1
                Tx_18B20; // set pin# to output (0)
                Port_18B20 = 0; // set pin# to low (0)
                        asmcekani1us(); // 1 wire require time delay
                Rx_18B20; // set pin# to input (release the bus)
                asmcekani60us(); // 1 wire require time delay
                                                        }
                else {
                //write 0
                Tx_18B20; // set pin# to output (0)
                Port_18B20 = 0; // set pin# to low (0)
                        asmcekani60us(); // 1 wire require time delay
                Rx_18B20; // set pin# to input (release the bus)
                }
                                                }
}
```

Obr. 21: Zapisování do dallasu

```
char Read_18B20 (){
        char i,result = 0;
        Rx_18B20; // TRIS is input(1)
        for(i = 0; i < 8; i++){
                Tx_18B20; // TRIS is output(0)
                Port_18B20 = 0; // genarate low pluse for 2us
                        asmcekani1us();
                        asmcekani1us();
                Rx_18B20; // TRIS is input(1) release the bus
                if(Port_18B20 != 0) result |= 1<<i;
                        asmcekani60us(); // wait for recovery time
                                                }
        return result;
                                        }
```

Obr. 22: Čtení z dallasu

```
void __attribute__((__interrupt__)) _T1Interrupt(void)
        {
        //vzít z A/D převodníku data pro světlo-vzorek1
        //použití funkcí pro dallas
                Reset_18B20();
                Write_18B20(Skip_ROM);
                Write_18B20(Read_scratchpad);

                tempL = Read_18B20();
                tempH = Read_18B20();

                tempL >>= 4;
                tempH <<= 4;
                tempH += tempL;
        //posílaní dat
        //zapnutí měření A/D převodníku
        // zapnutí měření dallasu
if(!Reset_18B20()){
                Write_18B20(Skip_ROM);
                Write_18B20(Convert_T);
}
        }
```

Obr. 23: Použití funkcí pro dallas v přerušení

3.4 Čidlo na snímání světla – fototranzistor

Náš měřič slunečního svitu je analogový. Mikroprocesor tedy snímá změnu napětí, kterou převedeme pomocí A/D převodníku. Podle velikosti změny napětí a datasheetu v počítači vyhodnotíme výsledky. Při běžném denním světle v místnosti je hodnota napětí na vstupním pinu procesoru (RB9) +5V, při posvícení např. telefonem se hodnota sníží

přibližně o 0,8V, díky těmto změnám dokážeme určit velikost slunečního svitu. Nejdříve nastavíme vstup do mikroprocesoru (RB9) jako analogový a nadále se budeme zabývat A/D převodníkem. Nejdříve se musíme postarat o jeho inicializaci.

```
//ADCON1 Registr
    ADCON1 = 0;
    ADCON1bits.ADON = 0;        //Vypnutí převodníku
ADCON1bits.FORM = 0;            //formát výstupu (0 unsigned int, 1 int, 2 unsignet fractional, 3 fractional)
ADCON1bits.SSRC = 0;//start převodu (0 ručně, 1 INT0, 2 TMR3, 3 PWM, 7 auto)
ADCON1bits.ASAM = 0;            //start samplování (0 ručně, 1 auto ihned po převodu)

//ADCON2 Registr
ADCON2bits.VCFG = 0;//Referenční napětí (0 obě z napájení, 1 +ext, 2 -ext, 3 obě z ext)
ADCON2bits.CSCNA = 0;           //Scanování vstupů
ADCON2bits.SMPI = 0; //Počet převodů na interupt - 1
ADCON2bits.BUFM = 0;            //Režim bufferů (0 16, 1 2x8)
ADCON2bits.ALTS = 0;//Režim MUXů (0 jen MUXA, 1 MUXA a MUXB)

//ADCON3 Registr
            //Krystal 7,3728 MHz * 8/4 = 14,7456 MIPS => Tcy=67,8 ns
            //Při Tad = 10 * Tcy bude Tad = 678 ns a minimální doba převodu bude 15 * Tad = přibl. 9,4 us
            //a tedy minimální vzorkovací frekvence přibližně 100 kHz
ADCON3bits.ADCS = 19; //ADCS = 2*Tad/Tcy - 1
            ADCON3bits.SAMC = 1;  //Počet Tad pro samplování při automatickém převodu
            ADCON3bits.ADRC = 0;  //Volba hodin (O systémové, 1 RC osc.)// možná PROBLEM

//ADCHS Registr
ADCHSbits.CH0NA = 0;  //Invert vstup CH0 (0 -Vref, 1 vstup AN1) // muze byt problem
ADCHSbits.CH0SA = 2;  //Neinvert vstup CH0 - vstup AN2

//ADCSSL Registr
ADCSSL = 0;                 //Scanování se nepoužívá

//ADPCFG Registr
ADPCFG = 0xFFFF;      //Všechny vstupy digitální
ADPCFGbits.PCFG2 = 0; //Jen AN2 jako analogový

//Zapnutí A/D převodníku je dobré provádět až po dokončení konfigurace
ADCON1bits.ADON = 1;
            ADCON1bits.SAMP = 1;        // a začínáme samplovat
```

Obr. 24: Inicializace A/D převodníku

Po nastavení převodníku už stačí samotný program. Celý program běží v nekonečné smyčce, ve které se čeká na interval přerušení od časovače TMR1. Používáme automatické spouštění převodu. Po příchodu intervalu od TMR1 se spustí samplování. Po době 1 Tad (což je dáno bity SAMC v registru ADCON3) se automaticky spustí převod. Výsledky se ukládají do buffru ADCBUF0. Dále je musíme posunout a vymaskovat a můžeme je odeslat po UARTU.

```
        IFS0bits.T1IF = 0;                  //Vymažeme příznak časovače (flag bit
        ADCON1bits.SAMP = 0;                //Konec samplování/začátek převodu
        while(!ADCON1bits.DONE);  //Čekáme, až bude převod hotov
        Vzorek1 = ADCBUF0;                          //Uložení vzorku z bufferu

        Vzorek1=Vzorek1 & 0b111111110000;    // maskování

Vzorek1=Vzorek1/16;                         // posouvání
```

Obr. 25 A/D převod program

3.5 Sériový port UART1

Jak jsme již mnohokrát řekli, ke komunikaci s počítačem používáme UART1. Nejdříve se podíváme na jeho inicializaci a poté na přenos dat. Nastavení pro náš mikroprocesor je následující:

- U1MODE=0x8000;
- U1STA=0x0400;
- U1BRG=23;

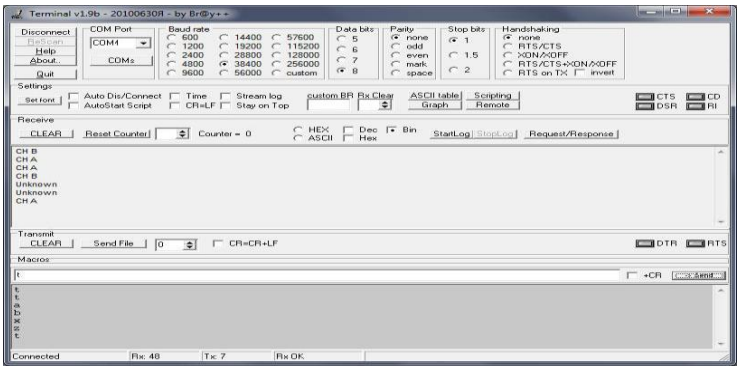
Všechna čísla jsou odvozena z datasheetu, proto pokud budete chtít nastavit sériový port jinak, stačí se podívat právě tam do sekce UART. Naše sériová linka je nastavena na rychlost 38 400Bd, využívá hlavní piny procesoru a zásobník je nastaven na okamžité odesílání. Samotný program pro odeslání dat vypadá takto:

```
//posílání
while(!IFS0bits.U1TXIF);
        U1TXREG= Vzorek1;
while(!IFS0bits.U1TXIF);
        U1TXREG= tempL;
```

3.6 PC část (příjímání, zobrazování)

Pro příjímání dat do počítače musíme zařídit snímaní sériového portu. Přesně tuto činnost nám zařídí volně stažitelný program ComTest Terminal. Je velmi jednoduchý a spolehlivý, nastavíme pouze rychlost 38400Bd, sériový port (většinou COM1) a zmáčkneme

„Připojit“. Na obrazovce se nám bude postupně zobrazovat binární kód, který pomocí ComTestu budeme ukládat do souboru.



Obr. 26: Nastavení programu ComTest Terminal

Jelikož víme, že se posílá 8bitů naměřených hodnot od fototranzistoru a 8 bitů naměřených hodnot od dallasu, tak není problém napsat program, který tato data bude zobrazovat.

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
   char c=0;
   int mocnina;      // pomocná proměná pro převod
   int i;            //pomocná pro funkci
   int svetlo=0;
   int teplota=0;
   FILE *f;
    f=fopen("název_souboru.txt","rw"); // otevření souboru
    fseek(f, 0, SEEK_SET);            // začátek
    while(c!=EOF){                    //světlo, načíta 1,0 do eof
           mocnina=128;
           for(i=0; i<=7; i++){
           c=getc(F);
           if(c==49) svetlo+=mocnina;
           mocnina/=2;                // převod jednotek
           }
           printf("Svetlo = %d\n", svetlo); // vypsání
           svetlo=0;
           if(c==EOF) break;
           mocnina=128;               //teplota
           for(i=0; i<=7; i++){
           c=getc(F);
           if(c==49) teplota+=mocnina;
           mocnina/=2;
           }
           printf("Teplota = %d\n", teplota);
           teplota=0;
           if(c==EOF) break;
           }
 fclose(F);
  system("PAUSE");
  return 0;
}
```

Obr. 27: Program na čtení hodnot

4.Závěr

Zadaný projekt se nám podařilo realizovat bez sebevětších problémů. Jediné velké stěžení byl bootloader, který jsme z důvodu časové tísně nedotáhli do konce. Tato práce nám byla velkým ponaučením. Zjistili jsme, že všechno nefunguje podle prvních představ, a že výrobek je především o dlouhodobé soustavné činnosti. K meteorologické stanici by se dala přidat další velmi zajímavá čidla a projekt tak dále rozvíjet. Na závěr bychom chtěli tuto práci doporučit všem nadšencům tohoto oboru.

Seznam použité literatury a zdrojů informací:

1. http://jakub.serych.cz/zaciname-dspicem
2. http://jakub.serych.cz/programujeme-procesory-dspic
3. Datasheet DSPIC30F3013 MICROCHIP (http://www.microchip.com)
4. Datasheet DS18B20 http://datasheets.maxim-ic.com/en/ds/DS18B20.pdf
5. Datasheet MAX232 www.ti.com/lit/ds/symlink/max232.pdf
6. Datasheet stabilizátor 7805 http:// vyhledavace.prohledejto.cz/7805-pdf/str-3/?q=7805.pdf
7. Vojta 30F3013 http://ozeas.sdb.cz/panska/30F3013/Prirucka/
8. http://cs.wikipedia.org/wiki/RS232
9. http://www.ok2jnj.wz.cz/ds18b20.htm

Seznam použitého SW:

- Eagle 4.16r2 (Easily Applicable Graphical Layout Edition) – návrh plošného spoje
- Microsoft Word – tvorba dokumentace
- Microsoft PowerPoint – tvorba prezentace
- MPLAB verze 7.21 – tvorba firmware mikroprocesoru
- ComTest Terminal – snímání sériového portu
- devC – zobrazování naměřených hodnot
- Asix up – program na nahrávání programu do DSPIC30F3013
- Tiny Bootloader - program na nahrávání programu do DSPIC30F3013

Seznam příloh:

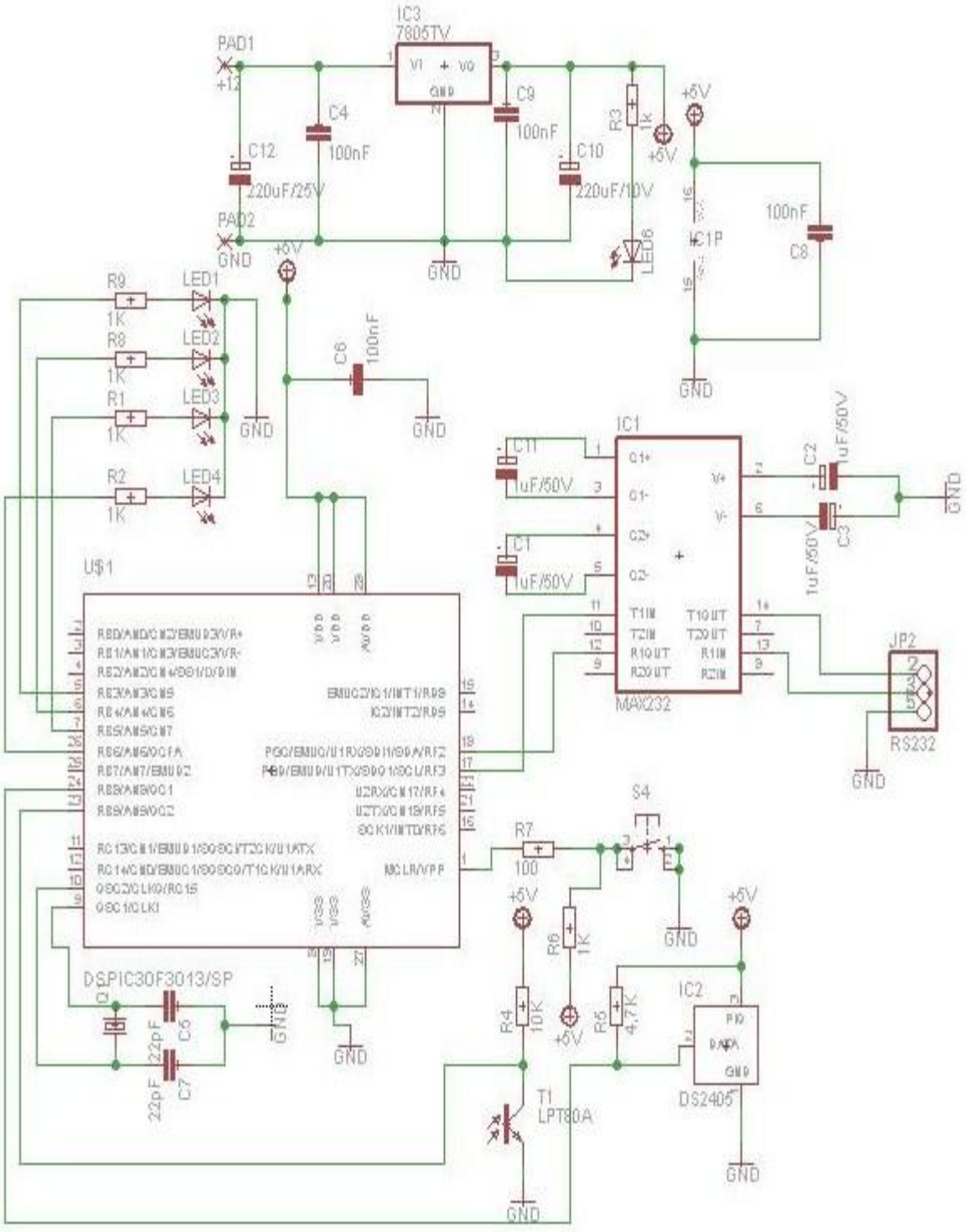
- Příloha 1: Schéma zapojení meteorologické stanice

Seznam obrázků

Poděkování:

Chtěli bychom především poděkovat panu Ing. Šerých za velkou trpělivost a užitečnou pomoc. Dále patří naše poděkováni panu Ing. Kubalíkovi za jeho konstruktivní rady a nakonec SPŠST Panská za poskytnutí vybavení.

Příloha 1:


IC3
7805TV
PAD1
+12
C4
100nF
C12
220uF/25V
PAD2
GND
C9
100nF
C10
220uF/10V
R3
1K
+5V
LED6
100nF
C8
R9
1K
LED1
R8
1K
LED2
R1
1K
LED3
R2
1K
LED4
C6
100nF
IC1
C11
1uF/50V
C1
1uF/50V
C2
1uF/50V
C3
1uF/50V
MAX232
JP2
RS232
U$1
DSPIC30F3013/SP
Q1
22pF
C5
22pF
C7
R7
100
S4
R6
1K
R4
10K
R5
4,7K
IC2
DS2405
T1
LPT80A