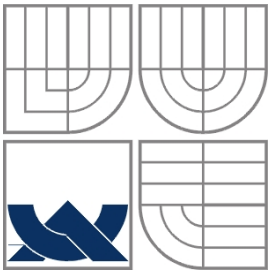
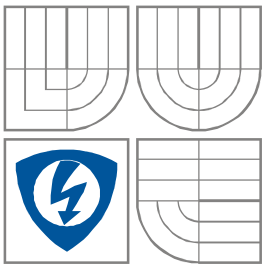


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

ZAŘÍZENÍ PRO DÁLKOVÉ OVLÁDÁNÍ TERČŮ
REMOTE CONTROL OF TARGETS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — STANISLAV TKÁČ
AUTHOR

VEDOUCÍ PRÁCE — Ing. IVANA JAKUBOVÁ
SUPERVISOR

BRNO 2009

Anotace

Úkolem této práce je vymyslet a zhotovit funkční zařízení, které má za úkol sepnout vždy jen jeden z několika (38 až 55) elektromagnetických ventilů. Ventil, který má být sepnut se volí buď z tlačítkové klávesnice, nebo z PC přes rozhraní RS 232. Zařízení musí být zhotoveno pomocí jednoho, nebo více procesorů PIC.

Annotation

The task of this work is to make a functional device, that is intended to close only one of several (38 to 55) of electromagnetic valves. Which valve is going to be switched is elected either of the push-button keypad, or from a PC via RS 232. Equipment must be made using one or more PIC processors.

Klíčová slova

Procesor, Assembler, Asynchronní komunikace, Klávesnice, Modul, Výstup, Rutina, Proměnná

Keywords

Processor, Assembler, Asynchronous communication, Keyboard, Module, Output, Routine, Variable

TKÁČ, S. *Zařízení pro dálkové ovládání terčů.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. XY s. Vedoucí bakalářské práce Ing. Ivana Jakubová.

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma Zařízení pro dálkové ovládání terčů jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

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

V Brně dne 5. června 2009 .....
podpis autora

Poděkování

Děkuji vedoucí bakalářské práce Ing. Ivaně Jakubové a konzultantovi Ing. Jiřímu Jakubovi za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne 5. června 2009 .....
podpis autora

Obsah

Seznam obrázků

Seznam vzorců

1 Úvod

V této části projektu vysvětlím konkrétně, jak má zařízení pracovat a pokusím se rozebrat podrobněji možná řešení, které mě napadly.

1.1 Upřesnění zadání bakalářské práce

Zařízení má ovládat elektromagnetické ventily, z nichž bude vždy pouze jeden v sepnutém stavu. Počet ovládaných ventilů je 38 a zařízení má umožnit rozšíření až na 55 ventilů. Ventily mají být ovládány tlačítky, kdy každý ventil má své tlačítko, nebo komunikací s PC přes rozhraní RS 232. Komunikace s PC stačí jednosměrná, ale po dohodě s konzultantem jsem zařízení doplnil a zařízení data do PC i vysílá. Klávesnice bude obsahovat ještě jedno tlačítko navíc, které bude sloužit k resetu. Po stisku resetu mají být všechny ventily vypnuty. Program pro procesor obsluhující klávesnici musí obsahovat ošetření proti zákmitům, proti držení tlačítka a musí být definován stav, který nastane po stisku dvou tlačítek současně. Napájecí zdroj musí být dimenzován minimálně na odběr dvou současně zapnutých ventilů plus odběr samotného zařízení (Ventil má indukční charakter a odebírá 9 W při napájení 24 V). Zdroj nemusí být součástí konstrukce, ale má být navržen dle požadavků. Data z PC budou odesílána ve formátu: dvě čísla podle ASCII tabulky a dva ukončovací znaky „CR“ a „LF“. Zařízení bude umístěno v rozvaděči, ale zatím není jasné, jaké bude v rozvaděči uspořádání, proto má být zařízení co nejvíce modulární. Samotná klávesnice nemusí být součástí zařízení. Hardwarová část musí obsahovat řídící část, případně napěťové regulátory a výstupní spínací prvky s ochranou proti napěťovým špičkám, které vznikají v důsledku induktivního charakteru ventilů.

Požadované parametry pro výstupní část:

U = 24 V
I = 0,395 A (Ventil 9 W => I = 0,375 A, Indikační žárovka I = 20 mA)

Parametry komunikace:

Symbolová rychlost: 9600 B/s
Datových bitů: 8
Stop bitů: 1
Bez parity

1.2 Návrhy hardwarového řešení

Projekt jsem začal vypracovávat s ohledem na upřesněné zadání, tudíž jsem se soustředil na univerzálnost a rozšiřitelnost celého zařízení. První moje myšlenka byla, aby zařízení obsahovalo pouze jeden procesor. Bylo by tedy potřeba vstupy a výstupy, obsluhovat multiplexně, jelikož procesor s potřebným počtem vstupů a výstupů by byl nad možnosti dostupného programátoru (PICSTART Plus). V tomto případě bych volil procesor PIC z řady 16 s označením 16F875, který poskytuje

dostatečnou paměť, rychlost a počet I/O pinů (23) a má v sobě implementováno USART rozhraní. Klávesnice by byla maticová 8 x 8. Vstupní piny klávesnice by byly „multiplexovány“ pomocí integrovaného obvodu CMOS 4017. Jde o tzv. „Johnsonův“ čítač. Výstupní piny klávesnice by byly připojeny přímo k mikropočítači. Klávesnice by tedy zabrala 10 pinů mikropočítače (8x vstupní pin, 1x výstup pro čítač, 1x reset čítače). Dva piny je potřeba rezervovat na komunikaci pomocí UART. Výstupy bych řešil pomocí převodníku kódu BCD na kód 1 ze 16ti (MH74154). Obvod MH74154 obsahuje 4 vstupy pro zadání BCD kódu a potom ještě vstupy G_1 a G_2, kterými lze obvod zablokovat a i přes nastavený BCD kód jsou všechny výstupy ve stavu log. 1. Takže pro tento případ by postačily 3 případně 4 tyto převodníky a zabraly by 7, nebo 8 výstupů mikropočítače. Celkem bych tímto řešením využil maximálně 20 I/O pinů mikropočítače. Vzhledem k tomu, že výše uvedený PIC poskytuje 23 I/O pinů, máme ještě 3 piny rezervu. Ze zadání požadavků na výstupy se jeví jako nejvhodnější řešení použít tranzistorová pole. Vybral jsem pole s označením ULN 2803A. Toto pole má vyhovující parametry: Maximální napětí 50 V a proud 0,5 A. Dále toto pole obsahuje ve svém pouzdru rekuperační diody, které jsou potřeba kvůli špičkám vznikajícím při vypnutí ventilu. Jedním tímto polem v pouzdře DIL je možné řídit 8 ventilů. Toto pole může být připojeno přímo na TTL logiku, avšak převodník MH74154 má na výstupech jako aktivní úroveň log. 0, takže by bylo potřeba výstupy negovat a to by bylo velice neefektivní řešení a bylo by potřeba mnoho obvodů navíc. Řešením by bylo použití převodníku BCD na kód 1 ze 16ti, který má na výstupech aktivní úroveň H, ale takovýto se mi nepodařilo v katalozích firem vyhledat, takže je možné, že se ani nevyrábí. Objevil jsem na internetu převodník kódu BCD na kód 1 z 10ti, jeho označení je CMOS 4028. Tento převodník má na výstupech aktivní úroveň log. 1, jenže má pouze 4 vstupy pro BCD kód a tudíž ho nelze blokovat a proto je jeho použití nevhodné, jelikož původní myšlenka je taková, že by vstupy BCD všech převodníků byly propojeny paralelně. Toto je jeden z důvodů, proč jsem k tomuto řešení nepřistoupil. Dalším důvodem je celkem problematické další případné rozšíření tohoto zařízení. Tímto vznikla myšlenka použít více procesorů PIC, které by spolu komunikovaly po sériové lince. Pro toto řešení jsem se rozhodl a začal ho vypracovávat. V době, kdy jsem měl projekt už hodně rozpracován, jsem narazil ještě na jedno vhodnější řešení, ale z důvodu času a vzhledem ke značně rozpracovanému projektu už nebylo možné celý projekt přepracovat. Toto řešení zde ale popíši. Jádrem tohoto řešení by byl procesor PIC, který má implementovány obvody pro komunikaci pomocí UART a I^2C sběrnici. Dále dostatek I/O pinů pro načtení tlačítka z klávesnice. Po načtení znaku z klávesnice, nebo z PC by procesor PIC odeslal informaci o tom, který výstup má být sepnut, nebo vypnut po I^2C sběrnici k obvodům PCF8574. Obvod PCF8574 je osmibitový expandér pro I^2C sběrnici. Tento obvod má osm I/O pinů a mohl by tedy řídit osm ventilů. Obvod může být adresován třemi bity a může jich tedy pracovat na jedné sběrnici osm. Celkem by bylo tedy možno řídit až 64 ventilů. Výhoda oproti mému řešení, které popisuji podrobně v následující části práce je, že by byl potřeba pouze jeden procesor a tudíž i jeden program. Rozšiřitelnost by byla snadnější, protože odpadá programování procesoru. Dále by procesor mohl mít informaci o tom, zda byl příslušný výstup opravdu sepnut a nedošlo k chybě. Cenově vychází moje řešení a řešení pomocí obvodů PCF8574 téměř stejně. Obě řešení mají oproti prvnímu nápadu výhodu v tom, že se skládají z části obsluhující klávesnici a z části, která se stará o výstupy. Pokud bude zařízení v rozvaděči rozmístěno tak, že klávesnice bude na dveřích a

zbytek zařízení uvnitř rozvaděče, tak stačí propojení napájecích a datových vodičů a není potřeba mít nataženo zbytečně velké množství vodičů, které navíc musí dovolit otevírání dveří, aniž by překážely, nebo se poškodily.

2 Řešení pomocí více procesorů

2.1 Koncept

Jak jsem naznačil v minulém odstavci, tak budu zadání projektu řešit pomocí více procesorů PIC, které budou komunikovat mezi sebou po asynchronní sériové lince. Zařízení se skládá z několika samostatných modulů a v případě potřeby se mohou vyrobit další, které zařízení rozšíří. Moduly budou dvojího druhu. Jeden bude sloužit pro obsloužení klávesnice a druhý pro řízení výstupů. Na modulu klávesnice bude umístěn konektor a příslušné obvody pro komunikaci s PC přes rozhraní RS 232. Napájení bude řešeno samostatně a navrženo dle zadání projektu. Modulů výstupů bude několik. Všechny budou stejné jak hardwarově tak softwarově a jejich počet bude záviset na tom, kolik ventilů bude potřeba ovládat. Tyto moduly budou od sebe odlišeny nastavenou adresou, která se bude moci nastavit propojkami. Program mikropočítače si zjistí, jaká je nastavená adresa a podle toho bude reagovat.

2.2 Výběr procesorů

Na začátku bylo potřeba se rozhodnout pro procesor, který by byl jednoho druhu pro oba moduly. Požadavky na procesor jsou tyto: Pro modul, který obsluhuje klávesnici je potřeba minimálně 18 I/O pinů (16 pro obsluhu klávesnice a 2 na UART komunikaci) a pro modul řídící výstupní tranzistorová pole, je zapotřebí 28 I/O pinů (24 pro ovládání ventilů, 2 pro komunikaci přes UART a 2 jako adresní bity modulu). Procesor musí mít programovou paměť typu „Flash“ a musí mít implementovánu řídící elektroniku pro UART komunikaci, přijatelnou cenu a musí být programovatelný programátorem PICSTART Plus. Vybíral jsem z několika obchodů a řešením by mohl být procesor PIC16F873A I/SP pro modul klávesnice a PIC16F874A I/SP pro modul výstupů, který nabízí GM Elektronik (dále jen GME). Tyto procesory se liší pouze počtem I/O pinů a tedy i počtem portů, ale jinak mají stejnou architekturu a používají stejné instrukce a paměti. S procesory PIC pracuji poprvé prostřednictvím této práce, ale rozhodnul jsem se jimi zabývat i nadále. V průběhu této práce jsem zkonstruoval programátor PIC podle návodu na internetu. Tento programátor používá jediný integrovaný obvod a tím je PIC18F2550 I/SP. Tento procesor má v sobě implementováno rozhraní USB 2.0 a programátor je tedy spojen s PC přes USB port. Po tom, co se mi podařilo oživit tento programátor, jsem se začal zajímat o řadu 18 PIC procesorů. Rozhodl jsem se, že když s těmito procesory začínám, tak bych mohl rovnou začít na vyšší řadě, protože mi umožní výhledově větší možnosti při konstrukci zařízení. Další věc je, že cena není o mnoho vyšší, než obdoba u nižší

řady procesorů. Na druhé straně budu mít více práce s psaním programu, protože na internetu jsou návody na programy psány nejčastěji právě pro nižší řady procesorů, než je 18. Pro konstrukci zařízení jsem tedy vybral procesor PIC18F252 I/SP (23 I/O pinů) pro modul klávesnice a PIC18F452 I/P (34 I/O pinů) pro modul řídící výstupy. Procesory se liší opět v počtech I/O pinů a počtech portů. Společné parametry jsou:

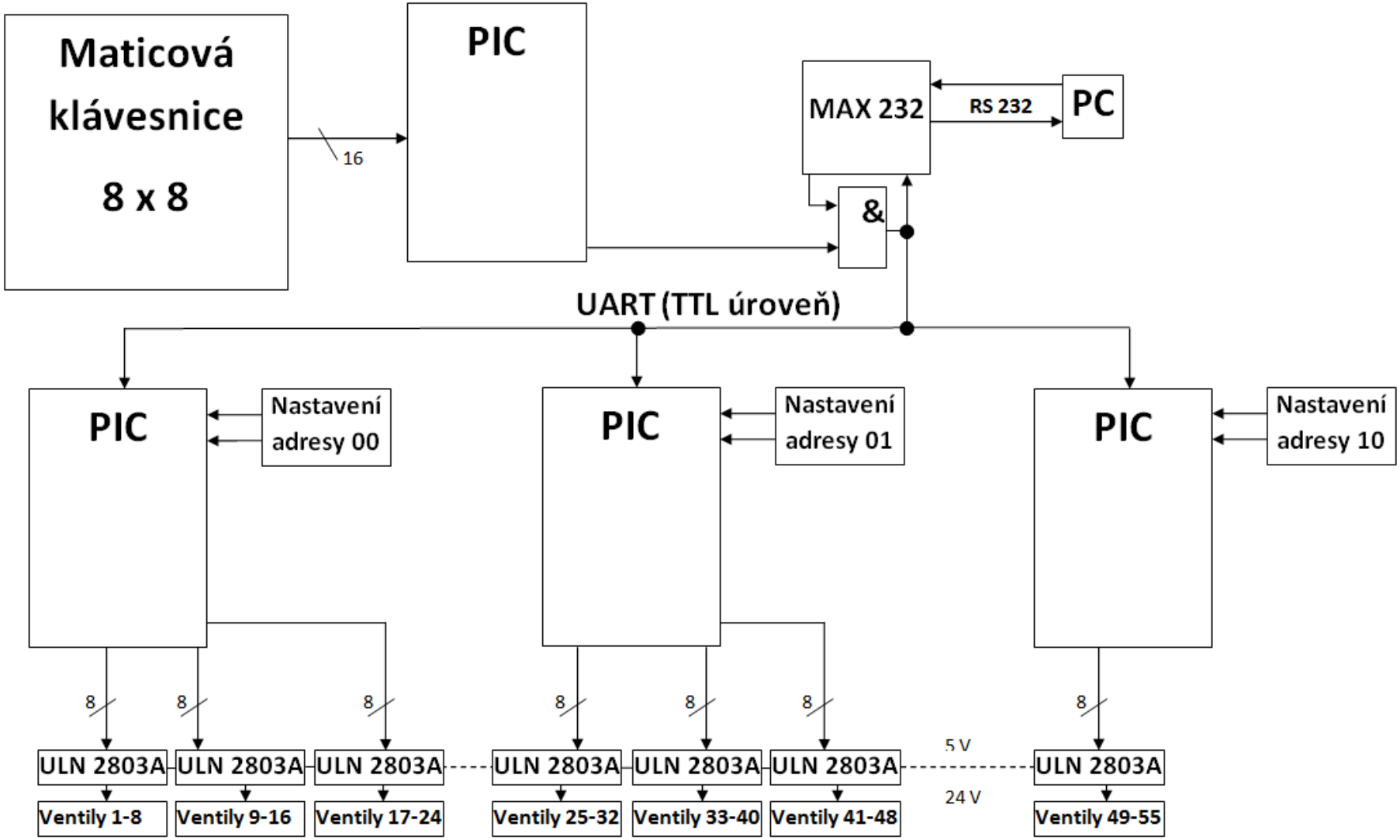
- Maximální frekvence krystalu 40 MHz
- 10 bitový AD převodník
- USART, I^2C
- PWM
- 4 x časovač + WDT
- 77 instrukcí
- Flash paměť 16384 Bajtů (8192 x 16 Slov)
- RAM 512 Bajtů

2.3 Blokový návrh konstrukce

Na obrázku č. 1 je znázorněno blokové schéma kompletního zařízení mimo napájecí zdroj. Jsou zde naznačeny napěťové úrovně. Pro řídící elektroniku je ze stabilizátoru napětí odebíráno 5 V. 24 V pro elektromagnetické ventily je odebíráno ze zdroje hned za usměrňovačem. Blok obsluhující maticovou klávesnici 8 x 8 a rozhraní RS 232 je znázorněn v horní části schématu. Hlavní částí je procesor PIC, součinový člen (74HC08), dále převodník úrovní RS 232 na TTL a opačně (MAX232). Jak je vidět ze schématu, tak cokoliv se odešle z PC, nebo z PIC je vysíláno do PC. Toto propojení není v zadání práce, ale po dohodě s konzultantem bylo vhodné toto udělat, kvůli případné archivaci odeslaných dat. Zařízení se bude ovládat buď z PC, nebo z klávesnice a proto bylo možné použít součinový člen AND. Tento člen jsem použil také z toho důvodu, že klid na lince pro UART je logická 1. Nabízelo se také použití rezistoru připojeného na +5 V a dvou diod aby se realizovala funkce AND, ale spolehlivější je použití hradla.

Spodní část schématu je složená z výstupních modulů. Každý modul obsluhuje jeden procesor. Všechny procesory mají stejný program a moduly se rozlišují nastavením dvou propojek. K modulům jsou přivedena dvě napětí. 5 V pro napájení procesoru a 24 V pro napájení tranzistorových polí a ventilů. Vstupy tranzistorových polí jsou připojeny přímo k I/O pinům procesoru, které se programem definují jako výstupní. Jeden modul obsahuje tři tranzistorová pole a může tedy řídit až 24 ventilů. Pokud není potřeba využívat všech 24 výstupů, tak není nutné všechna tranzistorová pole osazovat.

Obrázek č. 1 (Blokové schéma zařízení)



3 Hardwarová realizace

3.1 Napájecí zdroj

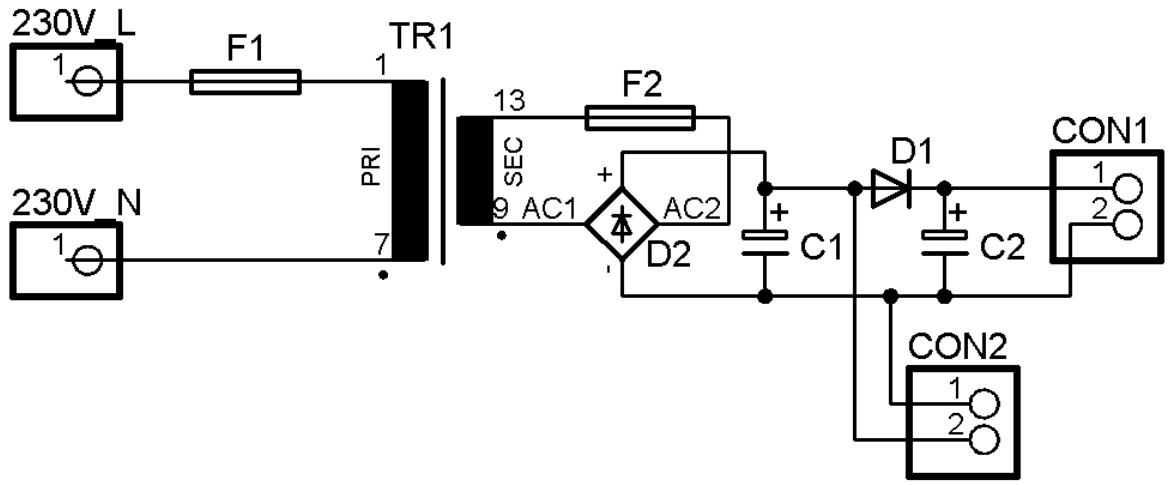
Napájecí zdroj jsem zvolil s transformátorem do desky plošných spojů. Na obrázku č. 2 je schéma tohoto zdroje a na obrázku č.3 návrh na desku plošného spoje s rozmístěním součástek. Zdroj obsahuje pojistku na primární i sekundární straně vinutí a je dimenzován dostatečně pro požadavky zadání. Při měření odběru jsem zjistil, že při třech výstupních modulech celkový odběr nepřesáhne 100 mA. Dva současně sepnuté ventily i se žárovkami odeberou celkem 0,79 A. Vybral bych tedy transformátor od firmy GES Electronics, má totiž výhodnější ceny pro transformátory než GME. Transformátor má následující parametry sekundárního vinutí: 25 VA, maximální proud sekundárního vinutí je 1,39 A a výstupní napětí 18 V. Primární napětí je 230 V. V katalogu je označen jako EI60/21 118. Pojistku na sekundární straně bych zvolil buď klasickou skleněnou 5 x 20 mm, nebo modernější vratnou pojistku typu PFRX.110. Tato vratná pojistka reaguje na proud nad 1,1 A a vydrží napětí 60 V. Na primární stranu bych volil skleněnou pojistku s hodnotou proudu 200 mA. Pro usměrnění střídavého napětí jsem zvolil usměrňovací Graetzův můstek s označením B380C1500, který dodává GME. Jeho parametry jsou: maximální proud 1,5 A a maximální napětí 380 V. Pro filtraci jsem zvolil elektrolytický kondenzátor na 35 V s kapacitou 1000 µF. Ventily údajně pracují i při nefiltrovaném napětí, ale po domluvě s konzultantem jsem ve zdroji kapacitu nevynechal. Po usměrnění a první filtraci jsem zařadil do série diodu a další filtrační elektrolytický kondenzátor. Tato část zdroje slouží k napájení modulu klávesnice, kde je stabilizátor napětí 7805CV, který dále napájí 5 V větve modulů obsluhujících výstupy. Kapacitu kondenzátoru jsem spočítal podle přibližného vzorce (napětí jsou volena dle limitů stabilizátoru):

Vzorec č. 1 (Výpočet kapacity filtračního kondenzátoru zdroje)

$$C_2 = \frac{I_{max}}{2 * f * (U_{max} - U_{min})} = \frac{I_{max}}{2 * f * \left(\left(U_{sek} * \sqrt{2} - 2 * U_D\right) - U_{min}\right)} =$$

$$= \frac{0{,}1}{2 * 50 * \left(\left(18 * \sqrt{2} - 2 * 0{,}6\right) - 8\right)} = 61{,}5\ \mu F \ => 100\ \mu F, 35\ V$$

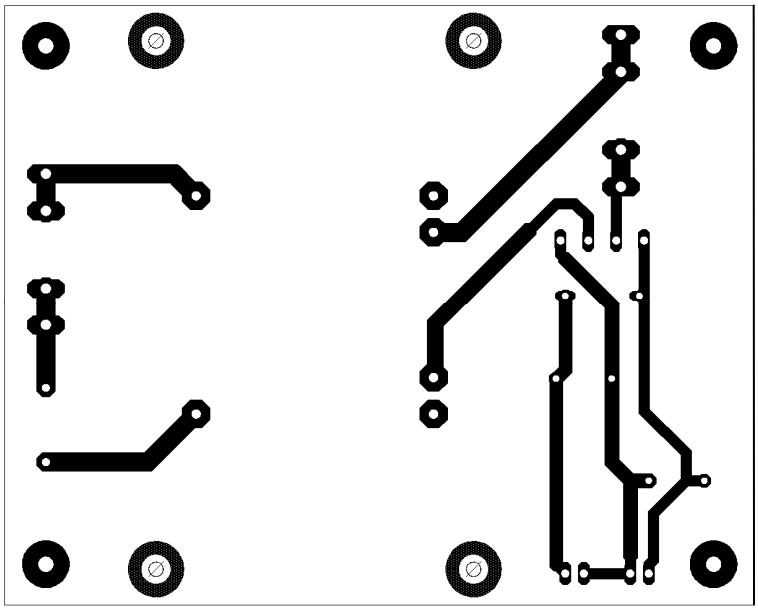
Obrázek č. 2 (Schéma napájecího zdroje)



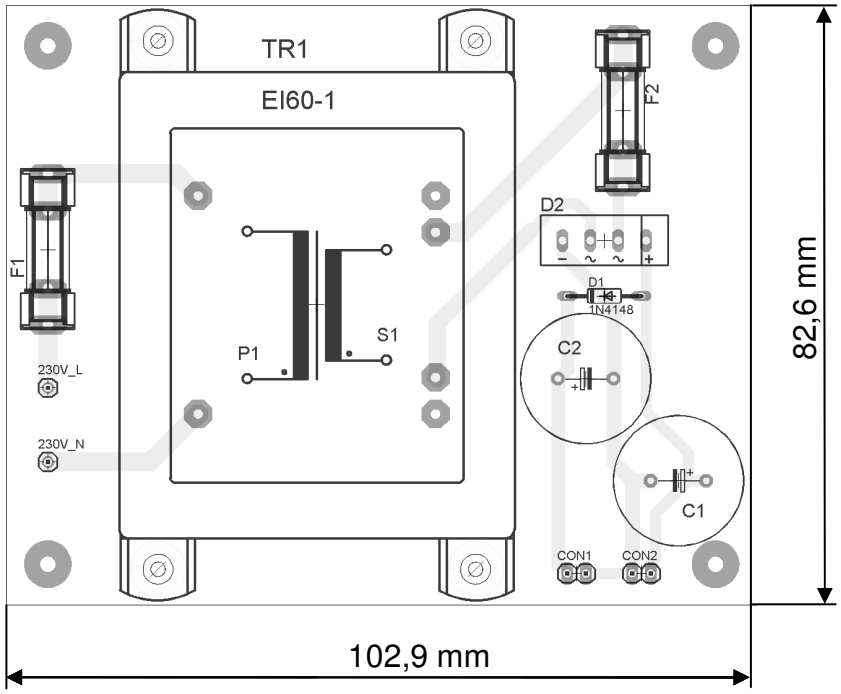
Seznam součástek:

F1	Tavná skleněná pojistka 20 x 5 mm, I = 200 mA
TR1	Transformátor, EI60/21 118, 25 VA; 230 V / 18 V / 1,39 A
F2	Tavná, nebo vratná pojistka. Vratná - PFRX.110; 1,1 A; 60 V
D1	Usměrňovací dioda, 1N4148
D2	Usměrňovací můstek B380C1500; 380 V; 1,5 A
C1	Elektrolyt, 1000 µF, 35 V
C2	Elektrolyt, 100 µF, 35 V
CON1	Pájecí špičky; Napájení řídící elektroniky
CON2	Pájecí špičky; Napájení elektromagnetických ventilů
DPS	Deska plošného spoje 102,9 x 82,6 mm

Obrázek č. 3 (Deska plošného spoje zdroje, měřítko 1:1)



Obrázek č. 4 (Osazovací schéma zdroje, měřítko 1:1)



3.2 Modul klávesnice

Při zadaném počtu tlačítek je nejvhodnější použít maticovou klávesnici. Rozhodl jsem se pro matici 8x8, neboť splňuje požadavky zadání i s rezervou (celkem 64 tlačítek). Samotná klávesnice není součástí tohoto modulu a bude připojena až v provozu, nicméně při vývoji tohoto modulu jsem na kontaktním poli měl připojeno 16 mikrospínačů do matice, abych mohl program otestovat. Na obrázku č. 5 je naznačeno, kde bude jaké tlačítko a na který pin a port procesoru budou připojeny vstupní a výstupní kontakty klávesnice.

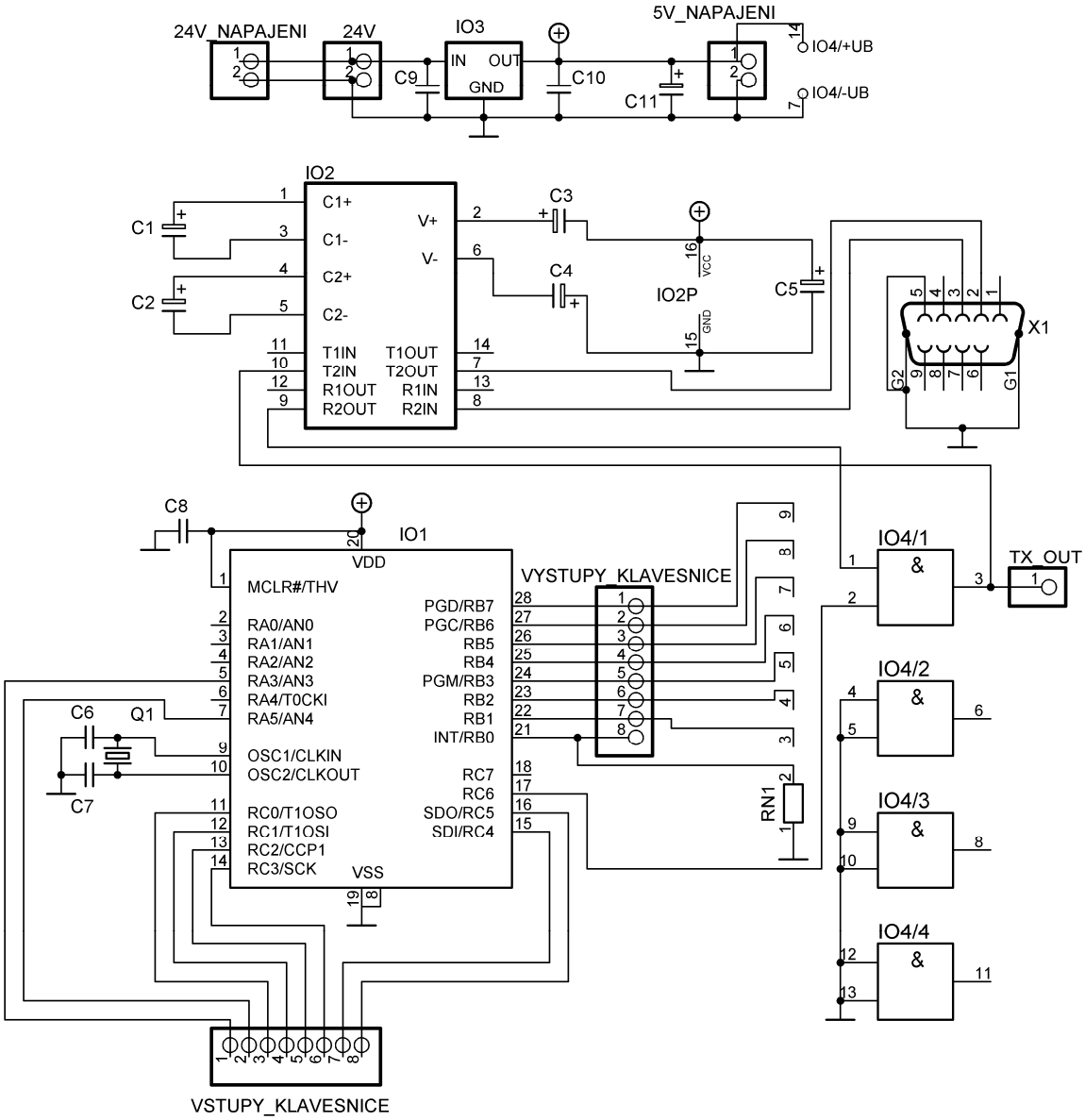
Obrázek č. 5 (Organizace maticové klávesnice 8 x 8)

Vstupní port	Port B							
Výstupní porty	0 bit	1 bit	2 bit	3 bit	4 bit	5 bit	6 bit	7 bit
Port A (3 bit)	RESET	1	2	3	4	5	6	7
Port A (5 bit)	8	9	10	11	12	13	14	15
Port C (0 bit)	16	17	18	19	20	21	22	23
Port C (1 bit)	24	25	26	27	28	29	30	31
Port C (2 bit)	32	33	34	35	36	37	38	39
Port C (3 bit)	40	41	42	43	44	45	46	47
Port C (4 bit)	48	49	50	51	52	53	54	55
Port C (5 bit)	56	57	58	59	60	61	62	63

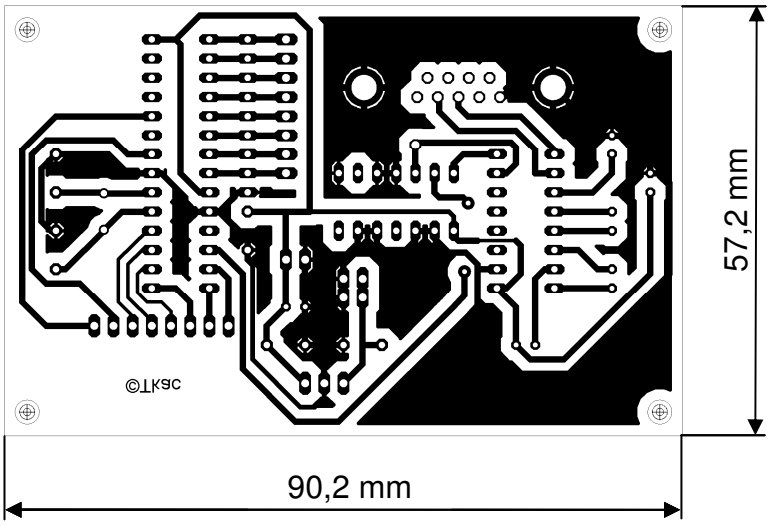
Zvolil jsem, že na modulu klávesnice bude stabilizátor napětí 7805CV a odtud se bude rozvádět napětí 5 V i pro další moduly, protože odběr modulů není velký a není tedy potřeba na každý dávat zvlášť stabilizátor. Napájecí napětí 24 V je přivedeno ze zdroje z konektoru CON1. Do tohoto modulu tedy přivádím filtrované napětí 24 V a mohu z něj odebírat stabilizované napětí 5 V, které potřebuji pro napájení výstupních modulů. Na vstupu a výstupu stabilizátoru je umístěn blokovací keramický kondenzátor 100 nF, aby stabilizátor nezačal kmitat. Na výstup jsem dal ještě filtrovací kapacitu 10 µF, kvůli filtraci napěťových špiček, které může generovat převodník MAX232. Ze stejného důvodu je připojen keramický 100 nF kondenzátor co nejblíže k procesoru PIC. Všechny elektrolytické kondenzátory použité v tomto modulu stačí na napětí 25 V. Úbytek napětí na stabilizátoru je 16 V. Maximální odebíraný proud jsem počítal s velkou rezervou 100 mA. Ztrátový výkon je tedy až 1,6 W. Proto jsem stabilizátor opatřil hliníkovým chladičem.

Konektor pro připojení k PC a tedy i obvod MAX232 jsem umístil také na tento modul. U obvodu MAX232 je připojen elektrolytický kondenzátor 10 µF, který je v doporučeném zapojení a slouží k filtraci napěťových špiček, které vznikají při práci nábojových pump uvnitř obvodu. Modul je spojen s PC přes konektor CANON9 (Dutinky), který je pro sériovou linku RS 232 standardní. Součástí modulu je samozřejmě jeden PIC18F252 I/SP, který je zasunut v patici DIP 28. Na konektor označený jako výstupy klávesnice je připojeno rezistorové pole s osmi rezistory, které definuje logickou nulu při nestisknutém tlačítku. Dále je k procesoru připojen krystal, který kmitá na 4 MHz a k němu dva kondenzátory 15 pF, které jsou doporučené pro frekvenci tohoto krystalu výrobcem. Použil jsem krystal kvůli jeho přesnému kmitočtu. RC člen by mohl na této frekvenci dělat problémy svým nestabilním kmitočtem u UART komunikace. Frekvenci krystalu jsem volil s ohledem na komunikaci UART podle tabulky z „Datasheetu“ výrobce. Jelikož funkce výrobku jako taková není náročná na vysokou rychlost prováděných instrukcí, tak jsem zvolil desetkrát menší kmitočet, než je maximální. Důsledkem toho bude mít procesor menší spotřebu proudu. Posledním integrovaným obvodem na této desce je 74HC08. Jde o čtyři rychlá hradla realizující funkci AND. Využívám pouze jedno hradlo a do něho přivádím signál z převodníku MAX232 a z USART jednotky procesoru PIC. Při asynchronní sériové komunikaci klid na lince je roven logické jedničce a jelikož zařízení bude obsluhováno buď pouze z PC, nebo pouze z klávesnice, mohl jsem si dovolit tyto zdroje spojit pomocí hradla. Nevyužité vstupy nezapojených hradel jsem připojil k zemi, abych jim definoval logickou úroveň, a nehrozilo tedy jejich kmitání. Schémata a deska plošného spoje jsou zobrazena na následujících třech obrázcích. Vše jsem nakreslil pomocí programu [1] Eagle. Pro lepší čitelnost jsem osazovací schéma zobrazil ve větším měřítku. Deska plošného spoje je jednostranná se základní roztečí vývodů součástek 2,54 mm. Než jsem desku plošných spojů navrhnul, tak jsem měl toto zapojení odzkoušeno na kontaktním poli.

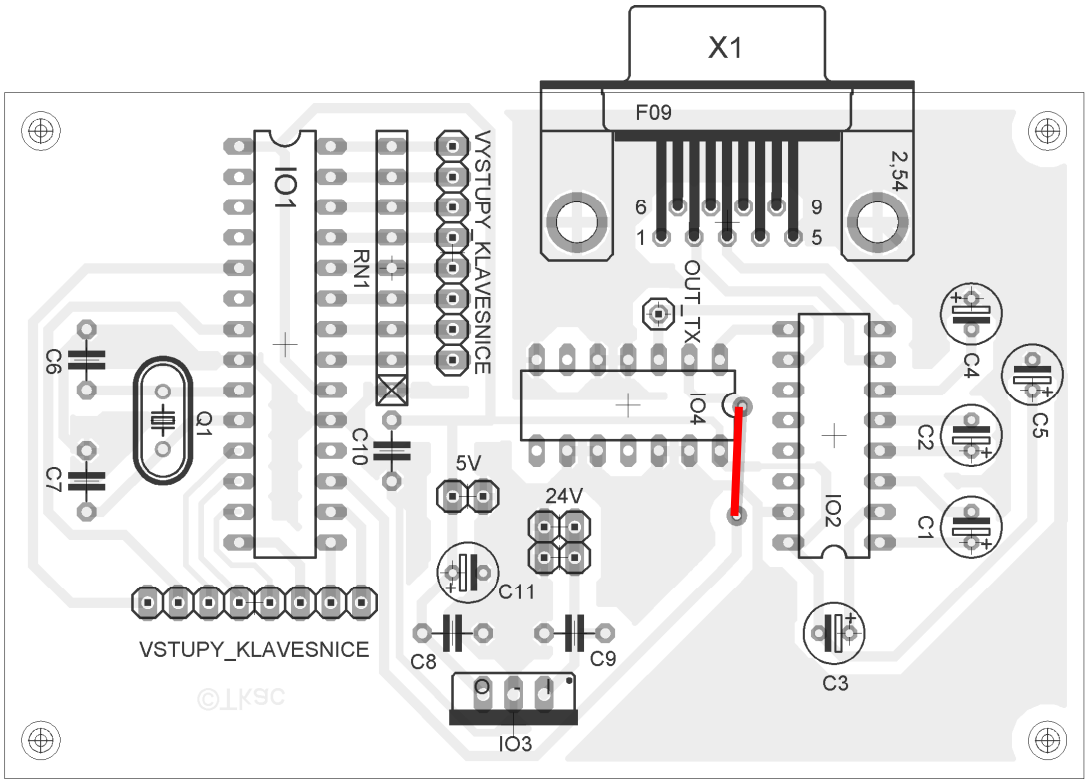
Obrázek č. 6 (Schéma zapojení modulu obsluhující klávesnici)



Obrázek č. 7 (Deska plošných spojů modulu klávesnice, měřítko 1:1)



Obrázek č. 8 (Osazovací schéma modulu klávesnice, měřítko 1,5:1)



Seznam součástek:

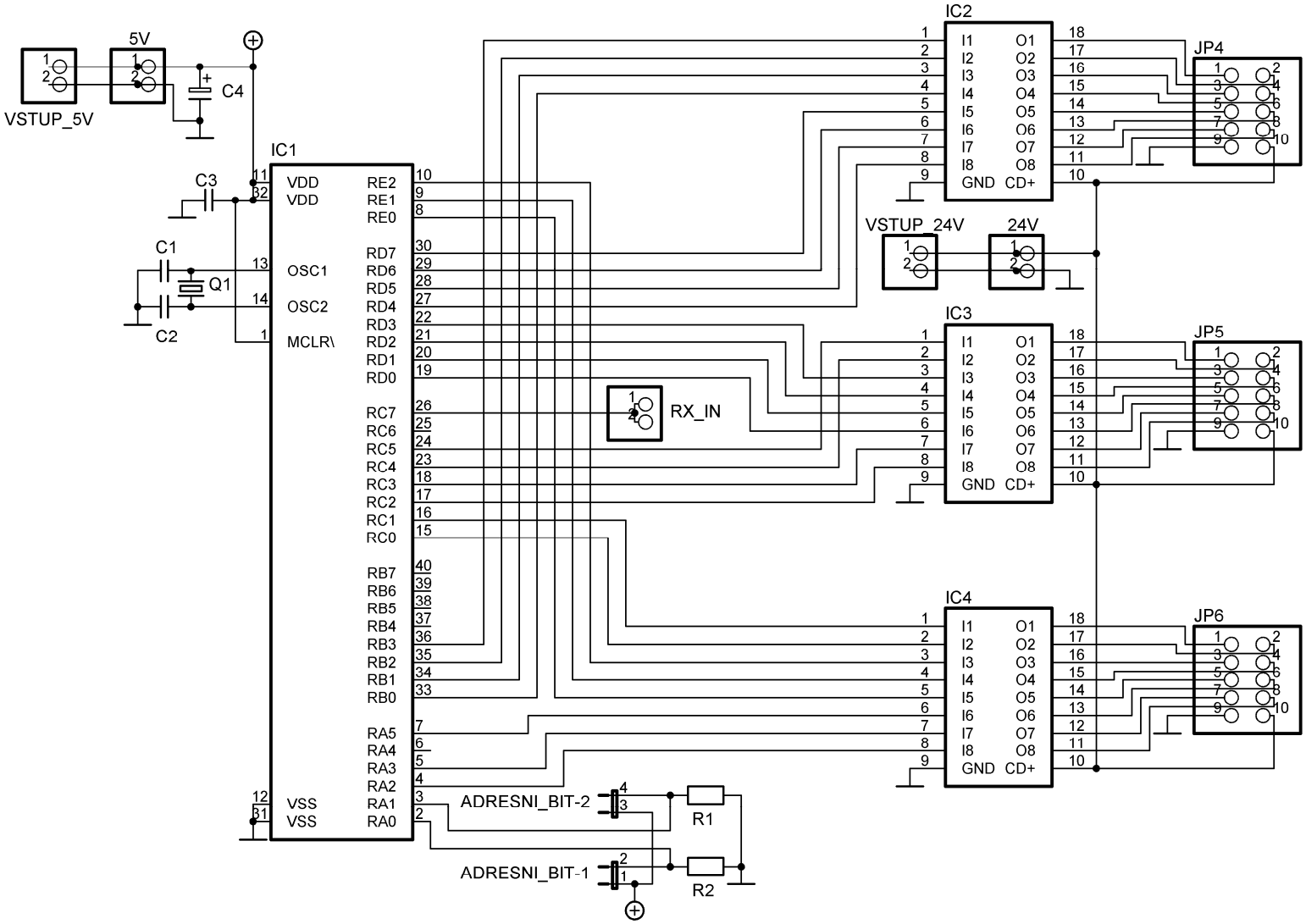
RN1	Rezistorové pole 8 x 10 kΩ
C1, C2, C3, C4, C5	Elektrolytické kondenzátory 2,2 µF, 25 V
C6, C7	Keramické kondenzátory 15 pF
C8, C9, C10	Keramické kondenzátory 100 nF
C11	Elektrolytický kondenzátor 10 µF, 25 V
Q1	Krystal 4 MHz
X1	Konektor do DPS, CANON9 (Dutinky), 90°
IO1	PIC 18F252 I/SP
IO2	MAX232, Převodník RS 232 ←→ TTL/CMOS
IO3	Stabilizátor 7805CV, I_{max} = 1,5 A
IO4	74HC08, 4 x AND, CMOS-High-Speed
DPS	Deska plošného spoje 90,2 x 57,2 mm

1 x Drátová propojka
1 x Patice DIP 28
1 x Kontaktová lišta jednořadá přímá
1 x Hliníkový chladič

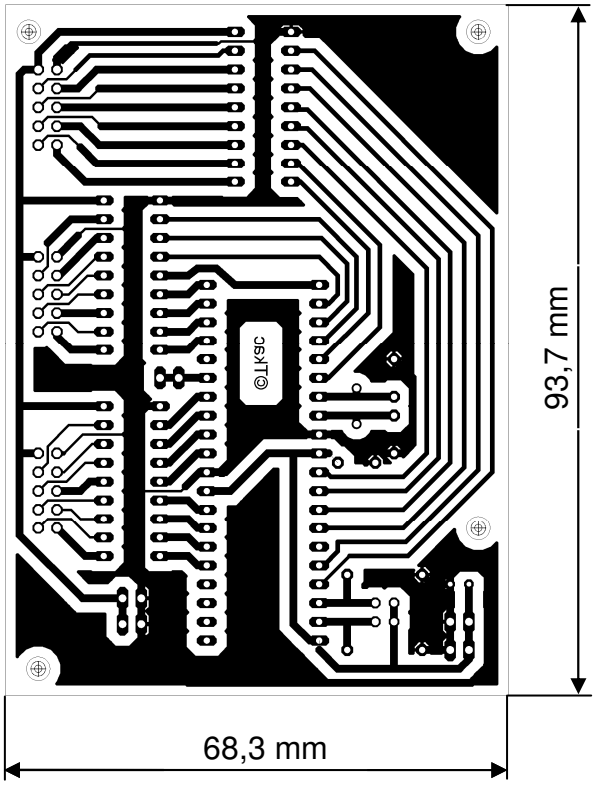
3.3 Modul pro řízení výstupů

Tento modul je na samostatné desce plošných spojů a může řídit maximálně 24 výstupů. Procesor řídí výstupy přes tranzistorová pole (ULN2803A). Přímo na výstupy tranzistorových polí jsou připojeny ventily. Tranzistorová pole jsou chráněna na svém výstupu rekuperačními diodami, takže nehrozí zničení tranzistorů při napěťových špičkách vznikajících při vypnutí elektromagnetického ventilu. Modul je napájen buď z předchozího stejného modulu, nebo z modulu klávesnice. Modul má konektory, které umožňují potřebná napětí a signál z asynchronní sériové linky propojit s dalším modulem paralelně. K modulu se přivádí jak napětí 5 V pro napájení procesoru, tak napětí 24 V pro napájení tranzistorových polí a ventilů. I když při sepnutí výstupu tranzistorového pole se daný ventil připojí k zemi, tak pro správnou funkci tranzistorového pole je potřeba na něj přivést 24 V. Napětí 24 i 5 V mají společnou zem, ale k modulům ji přivádím zvlášť kvůli spolehlivějšímu provozu zařízení. Modul je k ventilům připojen prostřednictvím třech dvouřadých konektorů. Každý konektor má deset pinů a je na něj napojeno 8 výstupů tranzistorových polí a dva piny jsou pro zem a 24 V. Tento modul obsahuje procesor PIC18F452 I/P. Frekvence je stejná jako u modulu klávesnice 4 MHz řízená krystalem. Pro filtraci napájecího napětí procesoru 5 V slouží kondenzátor C3 (Keramický) a C4 (Elektrolytický). Modul obsahuje dále dvě nastavitelné propojky, kterými lze nastavit adresu každého modulu. Použil jsem dva adresní bity, takže je možné použít 4 moduly s různou adresou a řídit tak až 92 ventilů. Na následujících třech obrázcích jsou navržená schémata a deska plošných spojů. Vše jsem navrhl v programu [1] Eagle. Deska plošných spojů je jednostranná se základní roztečí vývodů součástek 2,54 mm. Než jsem desku plošných spojů navrhnul, tak jsem měl toto zapojení odzkoušeno na kontaktním poli. Plošné spoje spojující konektory JP4 až JP6 s tranzistorovými poli jsou velmi tenké a bude jimi procházet proud okolo 0,4 A. Proto jsem tyto spoje pocínoval.

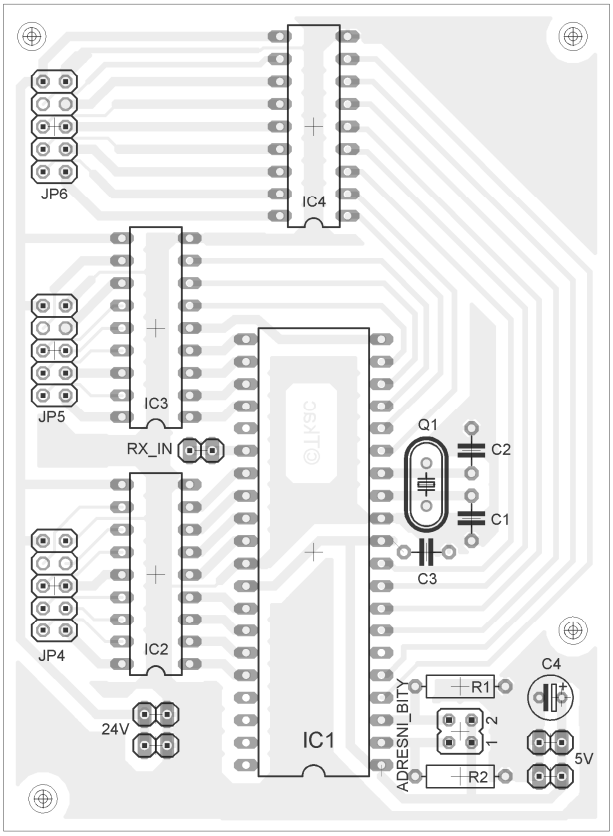
Obrázek č. 9 (Schéma zapojení modulu řízení výstupů)



Obrázek č. 10 (Deska plošných spojů modulu řízení výstupů, měřítko 1:1)



Obrázek č. 11 (Osazovací schéma modulu řízení výstupů, měřítko 1,2:1)



Seznam součástek:

R1, R2	Rezistory 10 kΩ
C1, C2	Keramické kondenzátory 15 pF
C3	Keramický kondenzátor 100 nF
C4	Elektrolytický kondenzátor 10 µF, 25 V
Q1	Krystal 4 MHz
IC1	PIC 18F452 I/P
IC2, IC3, IC4	ULN2803 (Tranzistorové pole)
JP4, JP5, JP6	Kontaktová lišta dvouřadá přímá
DPS	Deska plošného spoje 93,7 x 68,3 mm

1 x Patice DIP 40
1 x Kontaktová lišta jednořadá přímá

4 Programy

Program pro procesor modulu klávesnice i pro procesor modulu řídící výstupy je napsán v jazyce symbolických adres (JSA). Pro překlad do strojového kódu tedy používám Assembler. Jako vývojové prostředí jsem používal aplikaci [4] MPLAB IDE v8.10. Přípona souboru programu v JSA je „asm“. Po překladu jsou programy v hexadecimálním tvaru. Soubory mají příponu „h“. Pokud v popisu programu využívám čísla, tak je uvádím v dekadickém tvaru, není-li před číslem uvedena jiná číselná soustava. Funkčnost programů jsem mimo vizuální pozorování testoval i pomocí PC s využitím programu [2] Terminal. Pro kreslení vývojových diagramů jsem použil program [3] Diagram Designer. V programech využívám podprogramy a nebylo potřeba použít obsluhu přerušení. Programy obsluhují zapnutý „Watchdog“ časovač a je zapnuta ochrana resetem proti přetečení čítače instrukcí. Při použití krystalu kmitajícího na frekvenci 4 MHz, je doba vykonání jedné instrukce 1 µs, protože vykonání jedné instrukce trvá čtyři hodinové takty. Data vysílaná modulem klávesnice, nebo odesílaná z PC mají mít následující tvar, aby byly vyhodnoceny přijímacími moduly jako platné. Celkem jde o čtyři bajty. První bajt reprezentuje desítky, druhý bajt jednotky. Jedná se o číselné hodnoty podle ASCII tabulky. Čísla, která jsou nižší, než 10 budou odeslána ve tvaru např. 00, 01, 02 atd. Další dva znaky jsou ukončovací. Jedná se o znaky CR a LF. CR je v hexadecimálním tvaru 0D a LF náleží hodnota 0A. Znak CR je zároveň znakem kontrolním, pokud není odeslán, tak moduly nereagují. Ve všech programech v JSA jsem vytvořil hodně komentářů, které by měli přispět k lepšímu pochopení činnosti programů.

4.1 Program pro procesor modulu obsluhující klávesnici

4.1.1 Hlavní program

Pro názornější pochopení činnosti programu jsem vytvořil vývojový diagram hlavního programu (Obrázek č. 13) a rutiny, která převádí hodnotu načteného tlačítka z klávesnice na dva ASCII znaky (Obrázek č. 14). Následně budu tedy popisovat jednotlivé bloky vývojového diagramu pomocí samotného programu (JSA), který je v příloze 9.2.1. Na začátku programu je definován typ procesoru a inicializační soubor. Konfigurační bity jsou nastaveny až při programování. Nastavení konfiguračních bitů zobrazuje následující obrázek.

Obrázek č. 12 (Nastavení konfiguračních bitů)

Address	Value	Category	Setting
300001	21	Oscillator	XT
		Osc. Switch Enable	Disabled
300002	00	Power Up Timer	Enabled
		Brown Out Detect	Disabled
		Brown Out Voltage	4.5V
300003	0F	Watchdog Timer	Enabled
		Watchdog Postscaler	1:128
300005	00	CCP2 Mux	RB3
300006	81	Stack Overflow Reset	Enabled
		Low Voltage Program	Disabled
300008	0F	Code Protect 00200-01FFF	Disabled
		Code Protect 02000-03FFF	Disabled
		Code Protect 04000-05FFF	Disabled
		Code Protect 06000-07FFF	Disabled
300009	C0	Code Protect Boot	Disabled
		Data EE Read Protect	Disabled
30000A	0F	Table Write Protect 00200-01FFF	Disabled
		Table Write Protect 02000-03FFF	Disabled
		Table Write Protect 04000-05FFF	Disabled
		Table Write Protect 06000-07FFF	Disabled
30000B	E0	Config. Write Protect	Disabled
		Table Write Protect Boot	Disabled
		Data EE Write Protect	Disabled
30000C	0F	Table Read Protect 00200-01FFF	Disabled
		Table Read Protect 02000-03FFF	Disabled
		Table Read Protect 04000-05FFF	Disabled
		Table Read Protect 06000-07FFF	Disabled
30000D	40	Table Read Protect Boot	Disabled

Další část programu nastavuje parametry asynchronní sériové komunikace. Je potřeba povolit sériový přenos a ještě povolit vysílání. Dále je nutné nastavit „symbolovou“ rychlost. Tato rychlost se nastavuje v registru SPBRG a hodnota v registru je odečtená z tabulky v „Datasheetu“ procesoru a odvíjí se od kmitočtu krystalu. První čtyři bity portu A jsou primárně nastaveny jako analogové vstupy a je tedy potřeba změnit je na digitální vstupy či výstupy. Dále jsem nakonfiguroval výchozí stav I/O pinů (vstupy a výstupy) pro port A a C. Následuje deklarace proměnných, kterým je vyhrazen symbolický název a místo v paměti.

Předchozí část se zabývala hlavním nastavením a deklarací a nyní od návěští „Start“ už probíhá jádro programu. V první části proběhne nulování čtyř proměnných, ve kterých by mohla být uložena jiná než nulová hodnota od předchozího běhu

programu. Od návěští „keyboard“ probíhá část programu, která zjišťuje, zda nebylo stisknuto tlačítko. V průběhu programu jsou do kódu vkládány instrukce CLRWDT, které nulují „Watchdog“ časovač. Program pracuje tak, že nejprve nastaví RA3 jako výstup. V inicializaci je uložená do tohoto bitu hodnota logická jedna, takže se tato hodnota objeví na tomto pinu. Poté program testuje, zda není na některém bitu portu B jednička. Když ano, tak provede instrukci, která uloží definovanou hodnotu do W registru. Dále program nastaví RA3 jako vstup (stav vysoké impedance) a RA5 nastaví jako výstup a vše se opakuje. Takhle se postupně otestuje všech osm sloupců tlačítek. Původně jsem měl program takový, že všechny bity vstupu klávesnice byly definovány jako výstupy a postupně se na ně přiváděla jednička, ale toto řešení narazí na problém při stisku dvou tlačítek zároveň různých sloupců. V tom případě by se spojili nakrátko výstupy s nulami a jedničkou a došlo by ke zkratu výstupů procesoru.

Po testu celé klávesnice je hodnota W registru uložena do proměnné „temp“. Pokud je jeho hodnota nulová, tak se nuluje první bit proměnné „c5“ a proměnná „c4“. První bit proměnné „c5“ se používá pro ochranu proti držení tlačítka a proměnná „c4“ pro ošetření náhodného stisku tlačítka (bude popsáno v další části). Dále se program vrací na návěští „keyboard“ a testuje klávesnici znovu. Když ale testovaná hodnota „temp“ je jiná než nulová, tak program jde na návěští „ASCII“.

Návěští „ASCII“ je část programu, která ošetřuje náhodný stisk tlačítka, držení tlačítka a případně volá podprogram „UART“. Začíná se provedením logické funkce XOR proměnné „temp“ a „c4“ (c4 je ve výchozím nastavení nulová) a výsledek se uloží do „c4“. Po prvním provedení této funkce se do „c4“ uloží stejná hodnota jako je v „temp“. Poté se volá podprogram „cekej“, který funguje jako zpožďovací smyčka a trvá 0,123 s. Po návratu z podprogramu program testuje „c4“ a pokud není nulová, tak se vrací na návěští „keyboard“. Nyní mohou nastat tři stavy. První je, že bude načteno jiné tlačítko. V tom případě je klávesnice testována znovu. Druhý případ je, že není načteno žádné tlačítko (ve W registru je nulová hodnota). V tomto případě je vynulována proměnná „c4“ a opět se načítá tlačítko. Ve třetím případě je načteno stejné tlačítko a program pokračuje dále.

Další kroky programu testují první bit „c5“. Když je v něm jednička, tak to znamená, že tlačítko je stále drženo. V případě nulové hodnoty program tento bit nastaví na jedničku a pokračuje dál. Další krok nezačíná novým návěštím, ale je označen jako rutina pro převod hodnoty načteného tlačítka na ASCII znaky.

4.1.2 Rutina pro převod hodnoty načteného tlačítka na ASCII znaky

Funkci této rutiny naznačuje vývojový diagram na obrázku č. 14. Proměnná „Z“ ve vývojovém diagramu odpovídá proměnné „Desitky“ v programu. Jinak je značení proměnných totožné. Cílem této rutiny je zjistit počet desítek a počet jednotek. Tato dvě čísla jsou pak odeslány dále ke zpracování. Vykonání této rutiny generuje vždy jeden ze tří výsledků. Jeden výsledek je ten, že „X“ („X“ viz. vývojový diagram) je rovno nule a odešlou se podprogramu „UART“ tyto znaky: „0“, „0“. Pro procesory na příjmu to znamená vykonat reset. Reset je stav, kdy jsou všechny výstupy procesorů v nule (Tranzistory v tranzistorových polích jsou uzavřeny). Druhý

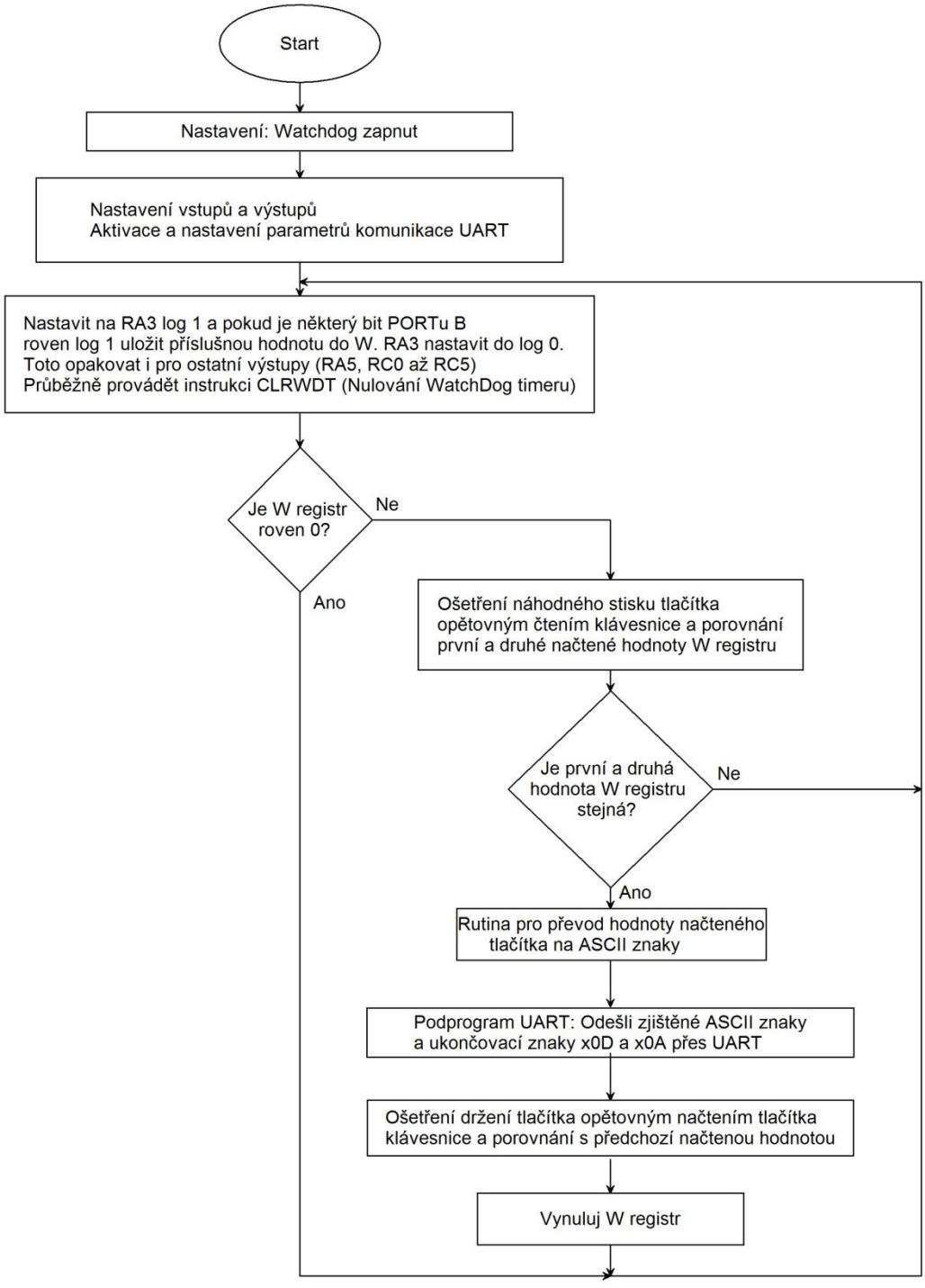
výsledek je ten, že se načtená hodnota tlačítka skládá jen z desítek a jednotková část je nulová. Podprogramu „UART“ se odesílají tyto hodnoty: „Z“, „0“. Třetí výsledek po vykonání rutiny je, že načtená hodnota tlačítka se skládá jak z desítek, tak z jednotek. Pokud jsou desítky nulové, tak se na jejich místě odešle nula a obecně jsou odeslány tyto hodnoty „Z“, „X“. Odesláním je myšleno uložení hodnot do proměnných, které pak podprogram „UART“ používá. Aby byl převod do ASCII znaků hotový, tak se v podprogramu „UART“ přičte se k oběma odeslaným znakům binární hodnota 00110000.

Nyní popíšu předešlé funkce na vývojovém diagramu. Na začátku se do proměnné „Z“ uloží jednička. Dále se od „temp“ odečte jednička a výsledek se uloží do „X“. Nyní program testuje, zdali je všech osm bitů „X“ rovno nule. Pokud ano, tak se uloží do „Z“ i do „X“ nula a volá se podprogram „UART“ (podprogram UART bude popsán v další části). Další stav je ten, kdy se „X“ nerovná nule. V tom případě program skáče na návěští „ASCII00“. Zde se nejprve odečte od „X“ 10 a poté se testuje, jestli je, nebo není „X“ záporné. Pokud je „X“ záporné, tak to značí, že byl nalezen počet všech desítek i jednotek a po provedení aritmetických operací jsou odeslány podprogramu „UART“ hodnoty „X“ a „Z“. V případě, že výsledek není záporný, tak se testuje, jestli je „X“ rovno nule. Pokud ano, tak to znamená, že načtená hodnota tlačítka se skládá pouze z desítek a odesílá se podprogramu „UART“ hodnota uložená v „Z“ a nulové „X“. Když však „X“ nulové není, přičte se k „Z“ jednička a program skáče zpět. To je v programu na návěští „ASCII00“. A cyklus se opakuje.

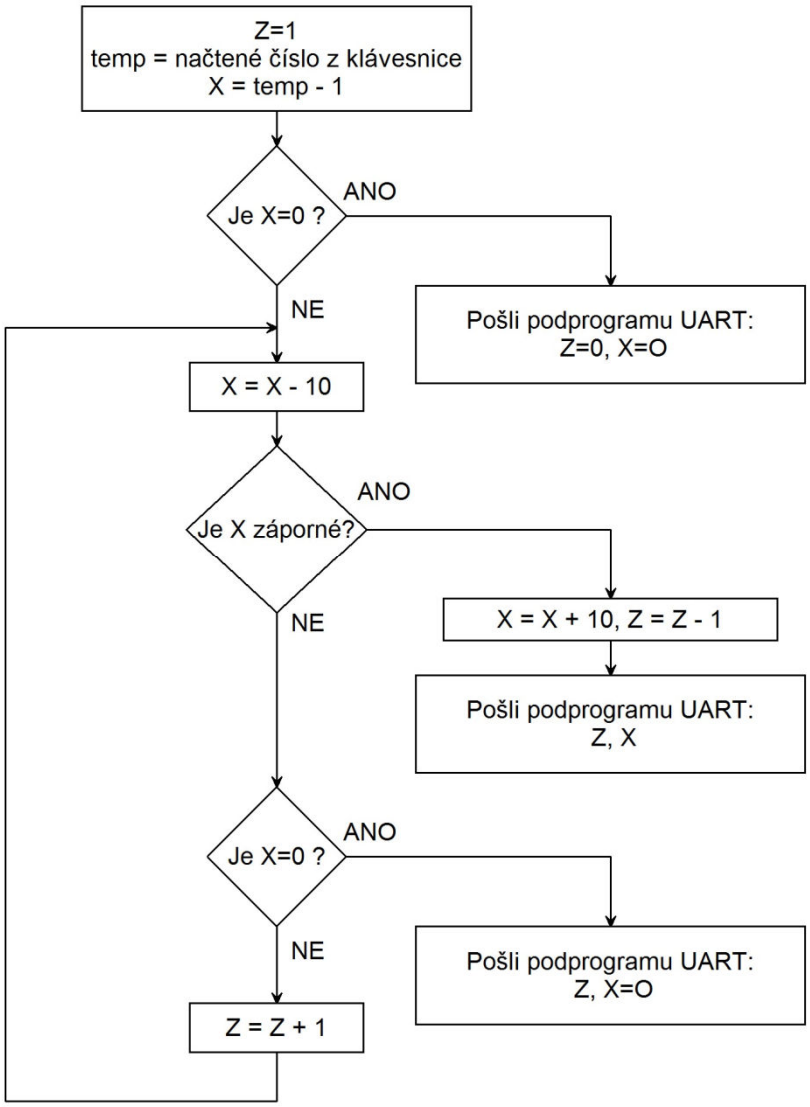
4.1.3 Popis podprogramu UART

Jak už jsem psal v předchozích odstavcích, tak tento podprogram provede součet „Z“ s binární hodnotou 00110000 a výsledek uloží zpět do této proměnné. To samé se provede s proměnnou „X“. Nyní jsou obě proměnné ve tvaru čísla podle ASCII tabulky. V dalších krocích se postupně odešle proměnná „Z“, proměnná „X“ a ukončovací znaky CR a LF. Odeslání znaků se provádí jednoduše zapsáním hodnoty do registru TXREG. Po této instrukci se čeká na nulovou hodnotu v registru TXSTA,TRMT. Nulová hodnota znamená, že vysílač je připraven na vysílání. Jednotce USART trvá odeslání symbolu při rychlosti 9600 B/s asi 0,1 ms. Celkem se tedy odešlou čtyři bajty. Po odeslání znaků je vynulována proměnná „c4“. Při návratu z tohoto podprogramu skáče program na návěští „Start“. V této chvíli následuje nulování proměnných „c4“, „X“, „temp“ a „Desitky“. Jediný příznak nese první bit proměnné „c5“ a ten je vynulován pouze, pokud nyní není stisknuto žádné tlačítko. Není-li tento bit vynulován, není možné, aby se odeslala další čtveřice bajtů. Při držení tlačítka se tedy data odešlou pouze jednou a pro další odeslání je třeba tlačítko pustit a opět stisknout (Ochrana proti opakovanému odeslání při delším držení tlačítka).

Obrázek č. 13 (Vývojový diagram hlavního programu)



Obrázek č. 14 (Rutina, která převádí hodnotu načteného tlačítka na ASCII znaky)



4.2 Program pro procesor modulu obsluhující výstupy

4.2.1 Hlavní program

I pro tento program jsem vytvořil vývojový diagram (Obrázek č. 16). Program je umístěn v příloze 9.2.2 na konci dokumentu a bude dostupný i na přiloženém CD. Na začátku programu je samozřejmě informace pro překladač o typu procesoru a inicializačním souboru. Nastavení konfiguračních bitů neprovádím v programu, ale opět při programování a jejich hodnoty zobrazuje následující obrázek.

Obrázek č. 15 (Nastavení konfiguračních bitů procesoru)

Address	Value	Category	Setting
300001	21	Oscillator	XT
		Osc. Switch Enable	Disabled
300002	0F	Power Up Timer	Disabled
		Brown Out Detect	Enabled
		Brown Out Voltage	2.0V
300003	0F	Watchdog Timer	Enabled
		Watchdog Postscaler	1:128
300005	01	CCP2 Mux	RC1
300006	81	Stack Overflow Reset	Enabled
		Low Voltage Program	Disabled
300008	0F	Code Protect 00200-01FFF	Disabled
		Code Protect 02000-03FFF	Disabled
		Code Protect 04000-05FFF	Disabled
		Code Protect 06000-07FFF	Disabled
300009	C0	Code Protect Boot	Disabled
		Data EE Read Protect	Disabled
30000A	0F	Table Write Protect 00200-01FFF	Disabled
		Table Write Protect 02000-03FFF	Disabled
		Table Write Protect 04000-05FFF	Disabled
		Table Write Protect 06000-07FFF	Disabled
30000B	E0	Config. Write Protect	Disabled
		Table Write Protect Boot	Disabled
		Data EE Write Protect	Disabled
30000C	0F	Table Read Protect 00200-01FFF	Disabled
		Table Read Protect 02000-03FFF	Disabled
		Table Read Protect 04000-05FFF	Disabled
		Table Read Protect 06000-07FFF	Disabled
30000D	40	Table Read Protect Boot	Disabled

Další část už se zabývá definicemi proměnných a jádrem programu. Na začátku je povolen „Watchdog“ časovač. Je nastavena a povolena asynchronní sériová komunikace. Dále bylo potřeba nastavit bity portu A jako digitální (Primárně jsou tyto piny nastaveny jako analogové vstupy). Další část programu nejprve uloží do všech portů nuly a potom nastaví, které piny budou vstupní a které výstupní. Toto pořadí ošetřuje stav, kdy by byl v některém portu bit nastavený do jedničky z předchozího běhu programu. Při nastavení vstupů a výstupů a až následném nulování portů by po dobu několika instrukcí byla na výstupu jednička, která by mohla způsobit problémy. Nyní následuje deklarace proměnných a jejich přiřazení k paměťovému místu. V průběhu programu jsou samozřejmě vkládány instrukce pro nulování „Watchdog“ časovače.

Samotné jádro programu začíná návěštím „Start“. Následuje návěští „Nulovani“, od kterého dál jsou nulovány všechny porty. Nyní se toto nulování neuplatní, ale později se využije pro generování stavu reset. Reset je takový stav modulu, kdy jsou všechny výstupy v nule. Program pokračuje na návěští „Main“. Od tohoto návěští dále jsou nulovány některé proměnné až po návěští „Znak1“. Od tohoto místa v programu bych popsal jeho činnost zjednodušeně takto: Program pracuje jako kruhový registr s pamětí na tři znaky. Po nově přijatém znaku kontroluje, zda je tento znak CR. Pokud není, tak přijatý znak zůstává v paměti a program čeká na další znak. Když je přijat znak CR, tak program vezme dva předchozí znaky a s nimi dále pracuje. Moje první verze programu pracovala tak, že program musel dostat všechny čtyři znaky v určitém časovém intervalu, jinak se vracel zpět na začátek, ale na konzultacích mi bylo řečeno, že při ovládání z PC budou znaky odesílány postupně, tak jak se zrovna napíší na klávesnici a ne všechny „najednou“. Proto vznikla myšlenka s postupným kruhovým ukládáním načtených bajtů. Toto

řešení má výhodu v tom, že se mohou odesílat v podstatě libovolný počet znaků, ale program pracuje vždy s dvěma posledními znaky před CR. V dohodnutém bloku dat je samozřejmě ještě poslední znak LF. S tím pracuje procesor, až zpracuje znaky před CR a jelikož se tento znak nerovná CR, tak je pouze uložen do paměti a procesor se jím dál nezabývá. Při další trojici přijatých bajtů je tento znak nahrazen. Program má ochranu proti tomu, když se do zpracovávaných znaků (před CR) dostane něco jiného než číslo nula až devět v ASCII formátu. Reakcí na tento případ je reset.

Teď tedy popíši tu samou funkci s odkazy na diagram a na samotný program. Za návěštím „Znak1“ je nejprve vynulován první bit „c6“. „c6“ slouží jako proměnná, kam se uloží jednička v případě, že byl načten znak CR. Dále program čeká na příznak, že byl přijat bajt. Když je bajt přijat, tak se uloží jeho hodnota do proměnné „Ajedna“ a následně se uloží hodnota „Ajedna“ do „cislo“. Program dále uloží jedničku do proměnné „c7“. Pozice jedničky v „c7“ určuje, ve kterém místě „kruhového registru“ se program právě nachází. V dalším kroku program volá podprogram „Detekce“. Tento podprogram popíši podrobně v následujícím odstavci. „Detekce“ vrací hodnotu v prvním bitu „c6“. Pokud je to nula, tak program čeká na další znak. Když je však v tomto bitu uložena jednička, tak program skáče na návěští „ASCII“ a zpracovává předchozí dva přijaté znaky. Jinak program pokračuje k návěští „Znak2“ a funkce je v podstatě stejná, jen se do proměnné „c7“ uloží jednička na pozici druhého bitu. Z prvního bitu je jednička nulována. Podobně tomu je i za návěštím „Znak3“.

Podprogram „Detekce“ jak už jsem psal, slouží k rozhodnutí, zda je přijatým znakem CR či nikoliv. Podprogram začíná instrukcí, která provede logickou funkci XOR proměnné „cislo“ a binární hodnoty 00001101 (hexadecimálně jde o „0D“) a výsledek uloží zpět do „cislo“. Poté testuje, zda je v „cislo“ nulová hodnota. Pokud není, tak se program skáče na návěští „ddd“ a následně se vrací z podprogramu s nulovým příznakem v „c6“. Je-li však hodnota v „cislo“ nulová, tak program nastaví první bit „c6“ na jedničku a poté zjišťuje, na kterém místě v „c7“ je jednička a podle toho pozná, které dvě proměnné (celkem jsou tři) má uložit k dalšímu zpracování. Následuje opět skok na návěští „ddd" a návrat z podprogramu.

4.2.2 Popis části programu pod návěštím „ASCII“

Po zjištění, že byl přijat znak CR, skáče program na návěští „ASCII“. Vývojový diagram je na obrázku č. 17. V proměnných „jedna“ a „dva“ jsou uloženy v ideálním případě čísla odpovídající počtu jednotek a desítek ve formě ASCII znaků. Od každého tohoto čísla se odečte konstanta binárně vyjádřená jako 00110000 a výsledkem jsou dvě čísla v BCD tvaru v rozmezí nula až devět (toto rozmezí budu dále nazývat jako správné). Výsledek se ukládá zpět do proměnných „jedna“ a „dva“. V jiném než ideálním případě odpovídají tyto hodnoty znakům podle ASCII tabulky a tyto čísla jsou vyšší, nebo nižší, než je správné rozmezí. Další instrukce programu právě zjišťují, jestli jsou výsledné hodnoty po odečtu konstanty ve správném rozmezí. Jestliže v kterémkoliv kroku program narazí na nesprávné rozmezí, skáče na návěští „Nulovani“, což znamená reset. Správné rozmezí se zjišťuje tak, že se po odečtení

konstanty zkontroluje, nulová hodnota horních čtyř bitů v bajtu (provádí se pro obě proměnné). Když jeden z nich nulový není, tak evidentně načtené číslo není ve správném rozmezí. Následuje odečet konstanty 10 od „jedna“ a „dva“ a kontroluje se, jestli je výsledek záporný. Když výsledek záporný není, tak to značí načtenou hodnotu mimo správné rozmezí (přijatý znak je mezi 15 a 10 pokud se program dostal až po toto místo). Ve správném rozmezí je po těchto krocích výsledek vždy záporný.

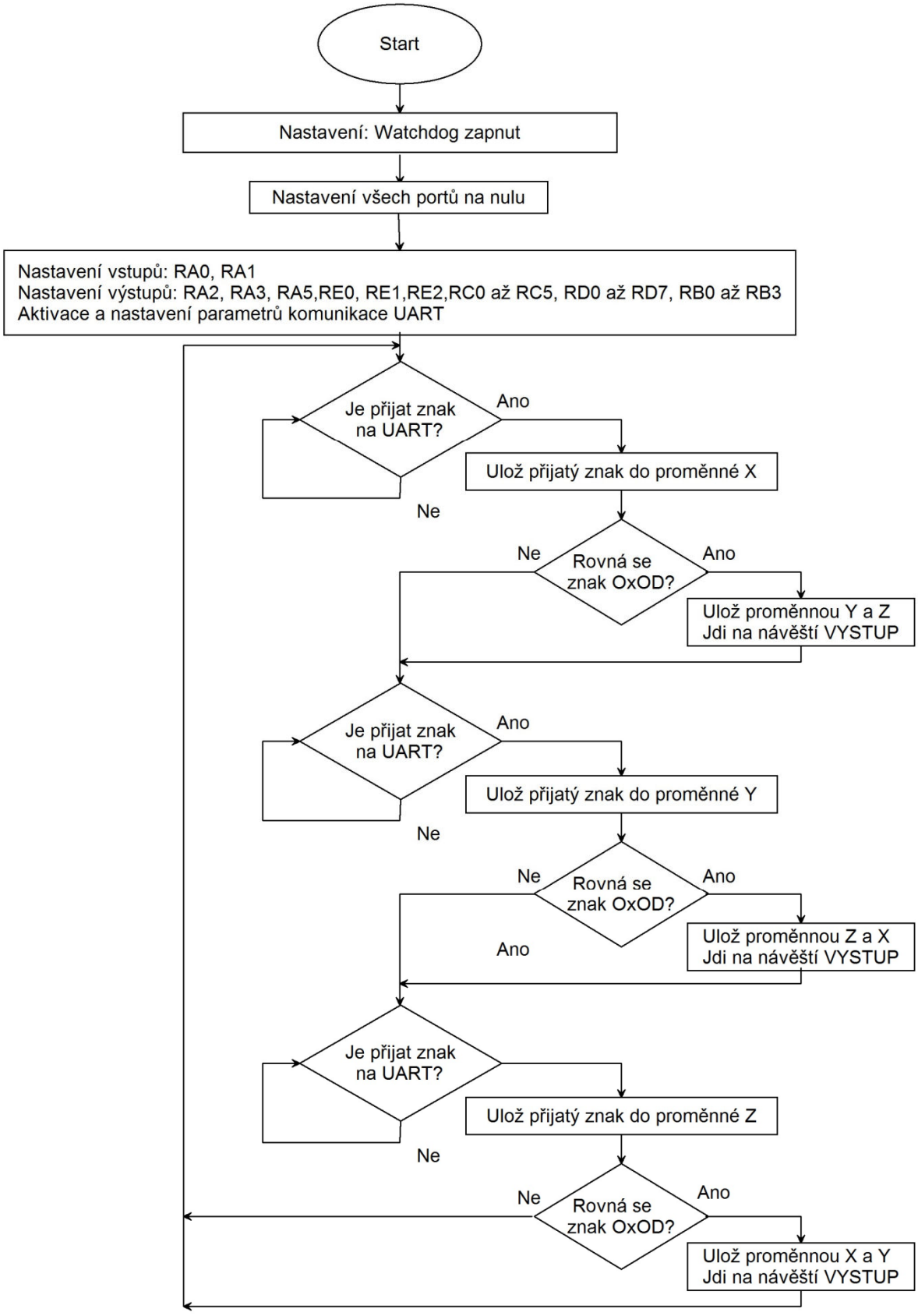
Další instrukce převádí přijaté znaky na jedno číslo. Proměnná „jedna“ reprezentuje počet desítek a proto je vynásobena konstantou 10 a sečtena s proměnnou „dva“ reprezentující jednotky. Výsledek je uložen do „dva“ a dále se pracuje jen s touto proměnnou. Jelikož tato práce je mým prvním seznámením se těmito procesory, tak se mi nedařilo provést aritmetickou operaci násobení tak, abych dostal správný výsledek i přesto, že procesor tuto aritmetickou operaci má v instrukční sadě. Proto jsou v tomto programu operace násobení realizovány cykly, ve kterých je prováděn součet stejné hodnoty právě tolikrát kolikrát potřebuji dané číslo násobit.

Následující instrukce ukládají do proměnné „Adresa“ hodnotu adresy, která je nastavena na modulu propojkami. K „Adresa“ je přičtena jednička a výsledek je uložen do „Adresa“. Další krok přičte k „dva“ konstantu 24 a uloží výsledek do „dva“. Následuje aritmetická operace, která od „dva“ odečte „Adresa“ vynásobenou konstantou 24 a výsledek se uloží do „dva“. Následně se vynulují všechny výstupy. Toto nulování je potřeba proto, aby se zrušilo případné předchozí nastavení aktivního výstupu. Tyto operace zajistí, že pokud jsou odeslané znaky určeny pro modul s touto adresou, tak v proměnné „dva“ je uložena hodnota v rozmezí 1 až 24. Nejsou-li odeslané znaky určeny pro modul s touto adresou je hodnota ve „dva“ jiná než toto rozmezí a má za následek, že výstupy zůstanou vynulovány. Jestliže se „dva“ v tomto rozmezí pohybuje, tak je nastaven příslušný výstup. Testování „dva“ probíhá tak, že se do W registru načte 1 a volá se podprogram „Hledani“. Podprogram vrací příznak prostřednictvím proměnné „c5“. Pokud je prvním bitu „c5“ jednička, tak program nastaví příslušný výstup a když je vrácena nula, tak se pokračuje bez nastavení výstupu. První bit „c5“ je nyní vynulován a pokračuje se načtením 2 do W registru a proces se opakuje pro další čísla až do 24. Na konci této rutiny program skáče na návěští „Main“ a opět čeká na příjem znaku.

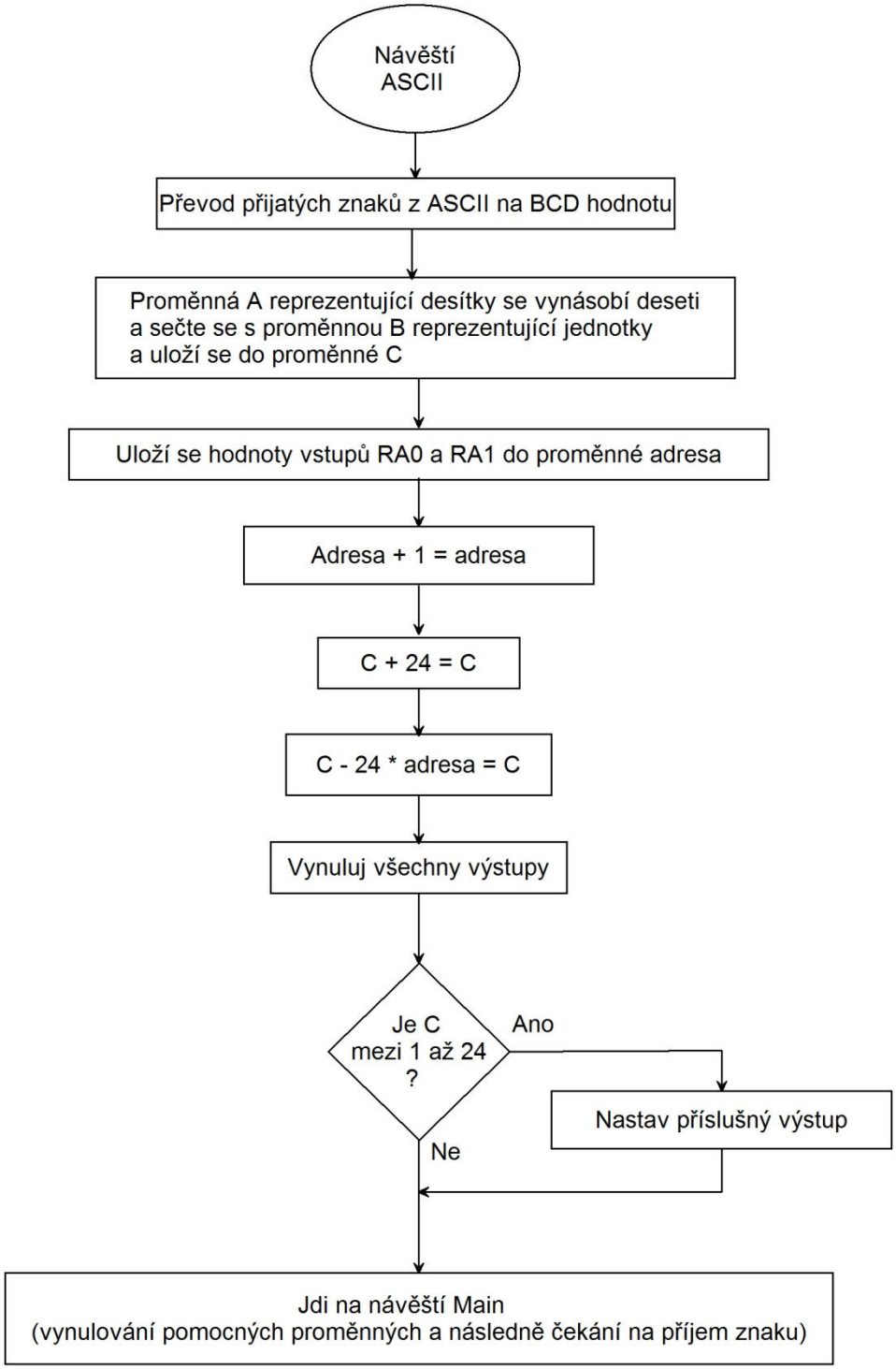
4.2.3 Popis podprogramu „Hledani“

Pro tento podprogram jsem nevytvářel vývojový diagram, protože jeho funkce je jednoduchá a podobná podprogramu „Detekce“. Tento podprogram provede aritmetickou operaci XOR hodnoty uložené ve W registru s proměnnou „dva“ a výsledek uloží do pomocné proměnné „c4“. Následně testuje, jestli je „c4“ nulové. Jestliže ano, tak se nastaví první bit „c5“ na jedničku a podprogram je u konce. Není-li však „c4“ nulové, první bit v „c5“ zůstává nulový a podprogram je u konce.

Obrázek č. 16 (Vývojový diagram hlavního programu)



Obrázek č. 17 (Vývojový diagram návěští „ASCII“)



5 Závěr

Dle technických požadavků jsem navrhnul několik možností realizace projektu a jednu si vybral. Vybral jsem řídící mikropočítače PIC také dle zadaných kritérií. Vzhledem k vybraným mikropočítačům jsem navrhnul elektrická schémata pro zdroj a řídící elektroniku. Tyto schémata jsem použil pro návrh a realizaci desek plošných spojů. Deska plošného spoje pro zdroj byla pouze navržena, protože konstruována být nemusela. Funkčnost navržených zapojení jsem nejdříve otestoval na kontaktních polích a poté jsem zařízení zrealizoval. Zrealizované přípravky jsem otestoval pomocí PC (Modul obsluhující klávesnici) a pomocí měření napětí na výstupech tranzistorových polí (Modul řídící výstupy). Oba přípravky jsou funkční. Zařízení mohlo být navrženo a zrealizováno jednodušším způsobem. Rozboru tohoto způsobu se věnuje kapitola 1.2. Realizované řešení však plně splňuje požadavky zadání. Pro hlavní části programu a některé podprogramy či návěští jsem vytvořil vývojové diagramy. Činnost programů vysvětluji pomocí těchto diagramů a pomocí samotných programů v JSA. Programy jsem psal bez předchozích znalostí programování mikropočítačů PIC s pomocí „Datasheetu“ k procesorům. Do příloh jsem zařadil oba programy jak ve formátu JSA, tak v hexadecimálním tvaru po překladu Assemblerem. Dále jsem dal jako přílohu fotografie hotových jednotlivých modulů a testovacích kontaktních polí. Na přiloženém CD je uložena samotná práce a samostatně programy v hexadecimálním formátu a ve formě JSA.

6 Seznam použitého software

[1] Eagle 4.16 (Editor schémat a Desek plošných spojů), free Edition, CadSoft

[2] Terminal v1.9b (Komunikace přes RS 232), Br@y++

[3] Diagram Designer (Editor vývojových diagramů), Michael Vinther

[4] MPLAB Integrated Development Environment, version 8.10, Microchip Technology Inc.

7 Seznam použité literatury

[5] HRBÁČEK, Jiří. *Komunikace mikrokontroléru s okolím 1.* Praha : BEN Technická literatura, 2002

[6] HRBÁČEK, Jiří. *Komunikace mikrokontroléru s okolím 2.* Praha : BEN Technická literatura, 2002

[7] HRBÁČEK, Jiří. *Mikrořadiče PIC16CXX a vývojový kit PICSTART*, Praha : BEN Technická literatura, 1996

[8] HRBÁČEK, J. *Programování mikrokontrolérů PIC16CXX*, Praha : BEN Technická literatura, 2001

[9] Technická dokumentace Microchip Technology Inc. Dostupná na WWW : <http://ww1.microchip.com/downloads/en/DeviceDoc/39564c.pdf>

[10] Technická dokumentace Microchip Technology Inc. Dostupná na WWW : <http://ww1.microchip.com/downloads/en/DeviceDoc/39582b.pdf>

[11] Technická dokumentace Maxim Integrated Products. Dostupná na WWW : <http://www.gme.cz/_dokumentace/dokumenty/433/433-100/dsh.433-100.1.pdf>

8 Seznam zkratek

USART	Synchronní a asynchronní sériové rozhraní (Universal Synchronous Asynchronous Receiver Transmitter)
UART	Asynchronní sériové rozhraní
PC	Osobní počítač (Personal Computer)
PIC	Jednočipový mikropočítač vyráběný firmou Microchip Technology
ICSP	Programování mikropočítače přímo v zařízení (In Circuit Serial Programming)
I^2C	Sériová sběrnice vyvinutá firmou Philips (Inter-Integrateg Circuit)
WDT	Časovač „hlídací pes“ je periferie, která vyvolá restart systému při jeho zaseknutí (Watchdog Timer)
PWM	Pulzně šířková modulace (Pulse Width Modulation)
JSA	Jazyk symbolických adres
ASCII	Tabulka definující znaky anglické abecedy a jiné znaky používané v informatice. (American Standard Code for Information Interchange)
CR	Řídící znak, který posune kurzor na začátek řádku (Carriage Return)

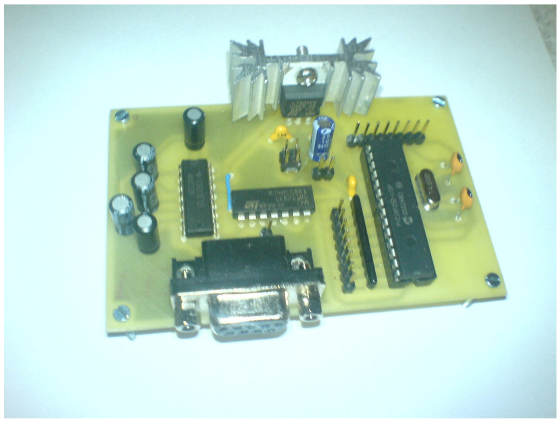
LF	Posun na další řádek (Line Feed)
CD	Kompaktní disk, Datové médium (Compact Disc)
BCD	Způsob kódování dekadických čísel pomocí binárních čtyřbitových čísel (Binary Coded Decimal)

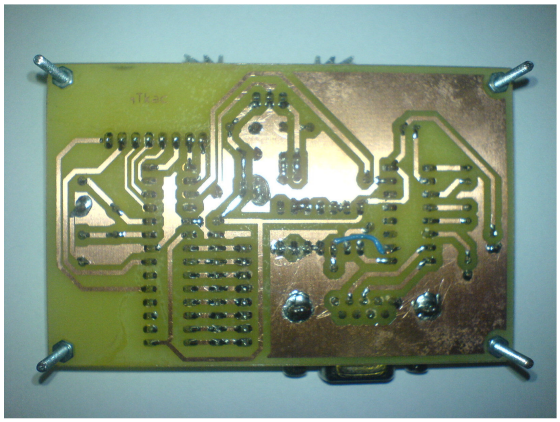
9 Přílohy

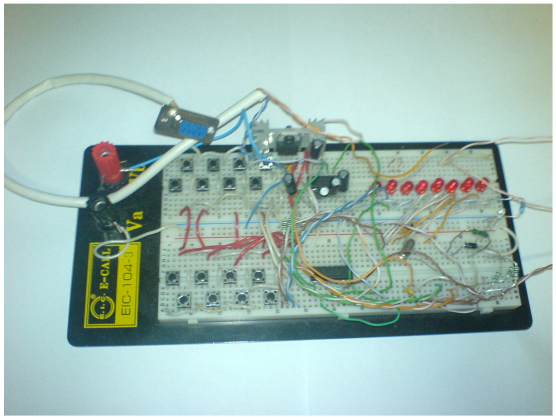
9.1 Fotografie

9.1.1 Fotografie modulu obsluhujícího klávesnici

Pohled ze strany součástek, plošných spojů a vývojové testovací kontaktní pole

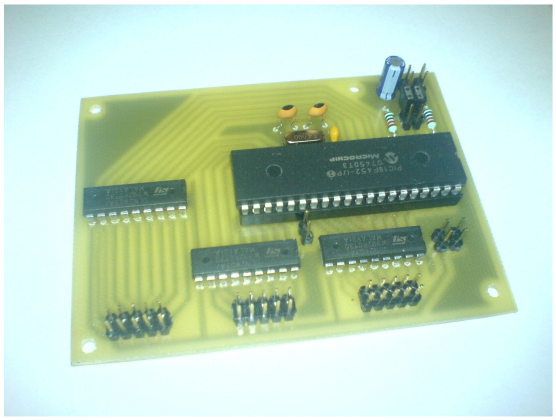


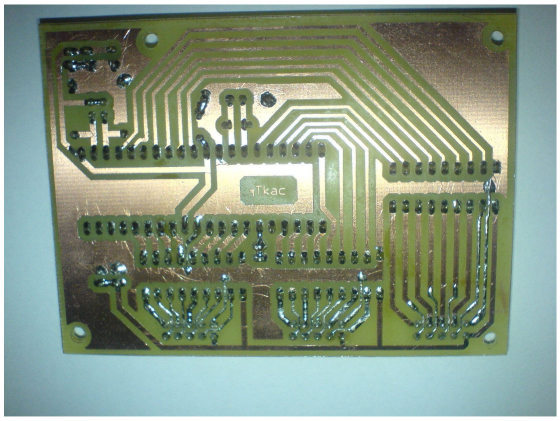


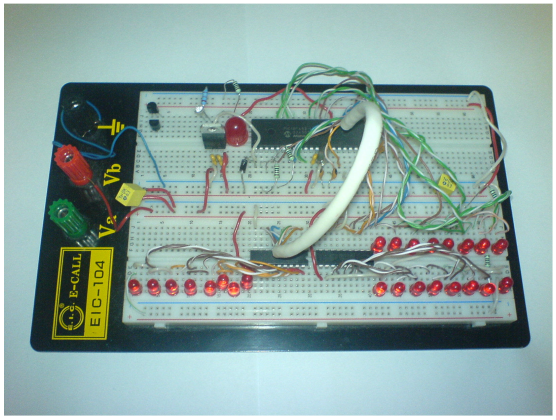


9.1.2 Fotografie modulu řídícího výstupy

Pohled ze strany součástek, plošných spojů a vývojové testovací kontaktní pole







9.2 Programy ve formátu JSA

9.2.1 Program procesoru obsluhující modul klávesnice

```
        LIST P=18F252

        INCLUDE <P18F252.INC>

;*********************************
;Inicializace
        CONFIG2H EQU h'300003'    ; Nastavení Watchdog timeru

        MOVLW  b'00001111'              ;Následná dělička WDT  nastavená na 1:128
        MOVWF  CONFIG2H,0
        BSF    SWDTEN,0                 ;Povolení WDT

        MOVLW  b'00101100'              ;Povoleni vysilani TX a nastaveni BRGH (high speed)
        MOVWF  TXSTA,0

        MOVLW  b'10000000'              ;Povoleni serioveho prenosu
        MOVWF  RCSTA,0

        MOVLW  b'00011001'              ;Nastaveni rychlosti 9600 B/s
        MOVWF  SPBRG,0

        CLRF LATA
        MOVLW 0x07 ; Configure A/D
        MOVWF ADCON1 ; for digital inputs

;*********************************
;Nastaveni vstupu a vystupu
        movlw b'01111111'
        movwf TRISA,0

        movlw b'10111111'
        movwf TRISC,0

        movlw b'00101000'
        movwf PORTA,0

        movlw b'00111111'
        movwf PORTC,0

;*********************************
;Deklarace promenych

c1                      EQU h'20' ;proměnné pro cykly
c2                      EQU h'21'
c3                      EQU h'22'

temp            EQU h'23'
X                       EQU h'24'
Desitky         EQU h'25'
c4                      EQU h'26'
c5                      EQU h'27'

                CLRF    c5
;*********************************
;Program

Start

                CLRF    c4
                CLRF    X
                CLRF    Desitky
                CLRF    temp

keyboard
```

```
                CLRWDT

MOVLW       0
                MOVWF temp

 BCF     TRISA,3          ; Testování sloupce 1
 BTFSC   PORTB,0             ; tlačítko po tlačítku
 MOVLW   1               ; pokud je tlačítko
 BTFSC   PORTB,1             ; stisknuté zapíše se
 MOVLW   2               ; do pracovního registru W
 BTFSC   PORTB,2             ; hodnota, která bude
 MOVLW   3               ; později určovat číslo tlačítka
 BTFSC   PORTB,3
 MOVLW   4
 BTFSC   PORTB,4
 MOVLW   5
 BTFSC   PORTB,5
 MOVLW   6
 BTFSC   PORTB,6
 MOVLW   7
 BTFSC   PORTB,7
 MOVLW   8
 BSF     TRISA,3

 BCF     TRISA,5          ; Testování sloupce 2
 BTFSC   PORTB,0
 MOVLW   9
 BTFSC   PORTB,1
 MOVLW   h'a'
 BTFSC   PORTB,2
 MOVLW   h'b'
                BTFSC    PORTB,3
 MOVLW   h'c'
 BTFSC   PORTB,4
 MOVLW   h'd'
 BTFSC   PORTB,5
 MOVLW   h'e'
 BTFSC   PORTB,6
 MOVLW   h'f'
 BTFSC   PORTB,7
 MOVLW   h'10'
 BSF     TRISA,5

                CLRWDT

 BCF     TRISC,0          ; Testování sloupce 3
 BTFSC   PORTB,0
 MOVLW   h'11'
 BTFSC   PORTB,1
 MOVLW   h'12'
 BTFSC   PORTB,2
 MOVLW   h'13'
 BTFSC   PORTB,3
 MOVLW   h'14'
 BTFSC   PORTB,4
 MOVLW   h'15'
 BTFSC   PORTB,5
 MOVLW   h'16'
 BTFSC   PORTB,6
 MOVLW   h'17'
 BTFSC   PORTB,7
 MOVLW   h'18'
 BSF     TRISC,0

                BCF     TRISC,1          ; Testování sloupce 4
 BTFSC   PORTB,0
 MOVLW   h'19'
 BTFSC   PORTB,1
 MOVLW   h'1a'
 BTFSC   PORTB,2
 MOVLW   h'1b'
 BTFSC   PORTB,3
 MOVLW   h'1c'
 BTFSC   PORTB,4
```

```
MOVLW   h'1d'
BTFSC   PORTB,5
MOVLW   h'1e'
BTFSC   PORTB,6
MOVLW   h'1f'
BTFSC   PORTB,7
MOVLW   h'20'
BSF     TRISC,1

        CLRWDT

        BCF     TRISC,2         ; Testování sloupce 5
BTFSC   PORTB,0
MOVLW   h'21'
BTFSC   PORTB,1
MOVLW   h'22'
BTFSC   PORTB,2
MOVLW   h'23'
BTFSC   PORTB,3
MOVLW   h'24'
BTFSC   PORTB,4
MOVLW   h'25'
BTFSC   PORTB,5
MOVLW   h'26'
BTFSC   PORTB,6
MOVLW   h'27'
BTFSC   PORTB,7
MOVLW   h'28'
BSF     TRISC,2

        BCF     TRISC,3         ; Testování sloupce 6
BTFSC   PORTB,0
MOVLW   h'29'
BTFSC   PORTB,1
MOVLW   h'2a'
BTFSC   PORTB,2
MOVLW   h'2b'
BTFSC   PORTB,3
MOVLW   h'2c'
BTFSC   PORTB,4
MOVLW   h'2d'
BTFSC   PORTB,5
MOVLW   h'2e'
BTFSC   PORTB,6
MOVLW   h'2f'
BTFSC   PORTB,7
MOVLW   h'30'
BSF     TRISC,3

        CLRWDT

        BCF     TRISC,4         ; Testování sloupce 7
BTFSC   PORTB,0
MOVLW   h'31'
BTFSC   PORTB,1
MOVLW   h'32'
BTFSC   PORTB,2
MOVLW   h'33'
BTFSC   PORTB,3
MOVLW   h'34'
BTFSC   PORTB,4
MOVLW   h'35'
BTFSC   PORTB,5
MOVLW   h'36'
BTFSC   PORTB,6
MOVLW   h'37'
BTFSC   PORTB,7
MOVLW   h'38'
BSF     TRISC,4

        BCF     TRISC,5         ; Testování sloupce 8
BTFSC   PORTB,0
MOVLW   h'39'
BTFSC   PORTB,1
MOVLW   h'3a'
```

```
        BTFSC   PORTB,2
        MOVLW   h'3b'
        BTFSC   PORTB,3
        MOVLW   h'3c'
        BTFSC   PORTB,4
        MOVLW   h'3d'
        BTFSC   PORTB,5
        MOVLW   h'3e'
        BTFSC   PORTB,6
        MOVLW   h'3f'
        BTFSC   PORTB,7
        MOVLW   h'40'
        BSF     TRISC,5

        MOVWF   temp

                        CLRWDT
; Testování nulového vstupu

        BTFSC   temp,0          ; test nenulového vstupu
        GOTO    ASCII           ; zde probíhá testování
        BTFSC   temp,1          ; bitů
        GOTO    ASCII           ; registru temp jeden
        BTFSC   temp,2          ; po druhém
        GOTO    ASCII
        BTFSC   temp,3
        GOTO    ASCII
                        BTFSC   temp,4
        GOTO    ASCII
                        BTFSC   temp,5
        GOTO    ASCII
        BTFSC   temp,6
        GOTO    ASCII
        BTFSC   temp,7
        GOTO    ASCII

                        BCF             c5,0
                        CLRF    c4

      GOTO      keyboard


;*******************************
                        ;Podprogramy:

;Podprogram zpozdeni

cekej

                        MOVLW   h'ff'
                        MOVWF   c3

wait3       MOVLW   h'0f'
                        MOVWF   c2

wait2       MOVLW   d'10'
                        MOVWF   c1

wait1
                        CLRWDT
                        DECFSZ  c1,1
                        GOTO    wait1
                        DECFSZ  c2,1
                        GOTO    wait2
                        DECFSZ  c3,1
                        GOTO    wait3

            RETURN

;Podprogram převodu tlačítka klávesnice na ASCII znaky

ASCII

            ;Ošetření náhodného stisku tlačítka
            MOVF    temp                ; Pokud je "c4" jiné než "temp", tak se načítá opět tlačítko klávesnice
```

```
        XORWF  c4                       ; Když je i podruhé načteno stejné tlačítko program pokračuje dál

        CALL cekej

        BTFSC   c4,0
        GOTO    keyboard
        BTFSC   c4,1
        GOTO    keyboard
        BTFSC   c4,2
        GOTO    keyboard
        BTFSC   c4,3
        GOTO    keyboard
        BTFSC   c4,4
        GOTO    keyboard
        BTFSC   c4,5
        GOTO    keyboard
        BTFSC   c4,6
        GOTO    keyboard
        BTFSC   c4,7
        GOTO    keyboard


        ;Ošetření držení tlačítka

        BTFSC   c5,0            ; Pokud se první bit "c5" rovná log 1 je stále drženo tlačítko,
        GOTO    keyboard ; program se vrací a znovu testuje klávesnici

        BSF             c5,0            ; Nastaví první bit "c5" na log 1. Pokud není tento bit vynulován je
                                        ; tím značeno, že tlačítko je stále drženo

        ;Rutina pro převod hodnoty v temp na ASCII znaky
        MOVLW  b'00000001'
        MOVWF  Desitky

        MOVFF   temp,X          ;Uloží hodnotu temp do X
        DECF    X,1                     ;Dekrementuje X a uloží zpět do X

                CLRWDT
    BTFSC   X,0             ; test nulového X
    GOTO    ASCIIOO
    BTFSC   X,1     ;V případě, že X=0 odešle se podprogramu UART "Desitky = O, X = 0"
    GOTO    ASCIIOO
    BTFSC   X,2
    GOTO    ASCIIOO
    BTFSC   X,3
    GOTO    ASCIIOO
                BTFSC   X,4
    GOTO    ASCIIOO
                BTFSC   X,5
    GOTO    ASCIIOO
    BTFSC   X,6
    GOTO    ASCIIOO
    BTFSC   X,7
    GOTO    ASCIIOO
                MOVLW 0
                MOVWF Desitky
                MOVWF X
                CALL UART
                GOTO Start


ASCIIOO

                CLRWDT
                MOVLW  h'a'                     ;Odecte 10 od X
                SUBWF   X,1

                BTFSC   X,7             ;Pokud je X záporné jdi na ASCIIZX
    GOTO    ASCIIZX

                BTFSC   X,0             ; test nulového X
    GOTO    ASCIIZA
    BTFSC   X,1     ;V případě, že X=0 odešle se podprogramu UART "Desitky, X = 0"
    GOTO    ASCIIZA
    BTFSC   X,2
```

```
        GOTO    ASCIIZA
        BTFSC   X,3
        GOTO    ASCIIZA
                BTFSC   X,4
        GOTO    ASCIIZA
                BTFSC   X,5
        GOTO    ASCIIZA
        BTFSC   X,6
        GOTO    ASCIIZA


                MOVLW 0
                MOVWF X

                CALL UART
                GOTO Start

ASCIIZX

        CLRWDT
        MOVLW  h'a'
        ADDWF  X,1                      ;Přičte k X 10

        MOVLW  h'1'
        SUBWF  Desitky,1 ;Odečte od Desitky 1

        CALL    UART
        GOTO    Start

ASCIIZA

        CLRWDT
        MOVLW  h'1'
        ADDWF  Desitky,1        ;Přičte k Desitky 1

        GOTO    ASCIIOO


;Podprogram odesilani pres UART

UART

        CLRWDT
        MOVLW  b'00110000'
        ADDWF  X,1
        MOVLW  b'00110000'
        ADDWF  Desitky,1

        CLRWDT
        BTFSS   TXSTA,TRMT
        GOTO    $-1
        MOVFF   Desitky,TXREG

        CLRWDT
        BTFSS   TXSTA,TRMT
        GOTO    $-1
        MOVFF   X,TXREG

        CLRWDT
        BTFSS   TXSTA,TRMT                      ;Odesle znak 0x0D
        GOTO    $-1
        MOVLW  b'00001101'
        MOVWF  TXREG

        CLRWDT
        BTFSS   TXSTA,TRMT                      ;Odesle znak 0x0A
        GOTO    $-1
        MOVLW  b'00001010'
        MOVWF  TXREG

        CLRF    c4

RETURN

end
```

9.2.2 Program procesoru obsluhující modul výstupů

```
        LIST P=18F452

        INCLUDE <P18F452.INC>

;*********************************
;Inicializace

CONFIG2H        EQU     h'300003' ; Nastavení Watchdog timeru

        MOVLW  b'00001111'              ;Následná dělička nastavená na 1:128
        MOVWF  CONFIG2H,0
        BSF             SWDTEN,0                ;Povolení WDT

        MOVLW  b'00101100'              ;Povoleni vysilani TX a nastaveni BRGH (high speed)
        MOVWF  TXSTA,0

        MOVLW  b'10010000'              ;Povoleni serioveho prenosu
        MOVWF  RCSTA,0

        MOVLW  b'00011001'              ;Nastaveni rychlosti 9600 B/s
        MOVWF  SPBRG,0

        CLRF LATA
        MOVLW 0x07              ; Configure A/D
        MOVWF ADCON1  ; for digital inputs


;*********************************
;Nastaveni vstupů, výstupů a počátečních hodnot výstupů


        movlw b'00000000' ; Všechny výstupy nastaví do log. 0
        movwf PORTA,0
        movwf PORTB,0
        movwf PORTC,0
        movwf PORTD,0
        movwf PORTE,0

        movlw b'00010011' ; Zde jsou definovány I/O piny jako
        movwf TRISA,0           ; vstupy a výstupy

        movlw b'11100000'
        movwf TRISB,0

        movlw b'10000000'
        movwf TRISC,0

        movlw b'00000000'
        movwf TRISD,0

        movlw b'00001000'
        movwf TRISE,0

;*********************************
;Deklarace promenych

c1              EQU h'20' ;proměnné pro cykly
c2              EQU h'21'
c3              EQU h'22'

adresa  EQU h'23'       ;Proměnná pro uložení adresy modulu


jedna   EQU h'25'       ; Pomocné proměnné kterých se
dva             EQU h'26'       ; využívá při operacích s načtenými znaky

cislo   EQU h'28'       ; Používá se při zjišťování, zda jde, nebo nejde o znak 0x0D

mezivysledek    EQU h'29'
c4              EQU     h'30'   ; Pomocná proměnná pro hledání znaku
```

```
c5              EQU     h'31'       ; Pomocná proměnná pro hledání znaku
c6              EQU h'32'           ; Pomocná proměnná pro detekci znaku 0xOD
c7              EQU h'36'           ; Pomocná proměnná pro detekci znaku 0xOD

Ajedna  EQU h'33'           ; Proměnné do kterých se ukládají přijaté znaky
Adva    EQU h'34'
Atri    EQU h'35'

;*********************************

Start


Nulovani

        CLRWDT

        movlw b'00000000' ;Všechny výstupy nastaví do log. 0
        movwf PORTA
        movwf PORTB
        movwf PORTC
        movwf PORTD
        movwf PORTE

Main

        CLRWDT

        CLRF    jedna
        CLRF    dva

        CLRF    Ajedna
        CLRF    Adva
        CLRF    Atri

        CLRF    mezivysledek
        CLRF    c5

Znak1

        CLRWDT
        BCF             c6,0
        BTFSS   PIR1,RCIF
        GOTO    Znak1
        MOVFF   RCREG,Ajedna
        MOVFF   Ajedna,cislo        ; Podprogram detekce zjistuje, jestli promenna
        MOVLW   b'00000001'                 ; cislo je rovna 0x0D
        MOVWF   c7                                  ;Pozice log 1 v c7 urci, ktere znaky se budou zpracovavat
        CALL    Detekce
        BTFSC   c6,0
        GOTO    ASCII

Znak2

        CLRWDT
        BCF             c6,0
        BTFSS   PIR1,RCIF
        GOTO    Znak2
        MOVFF   RCREG,Adva
        MOVFF   Adva,cislo
        MOVLW   b'00000010'
        MOVWF   c7
        CALL    Detekce
        BTFSC   c6,0
        GOTO    ASCII

Znak3

        CLRWDT
        BCF             c6,0
        BTFSS   PIR1,RCIF
        GOTO    Znak3
        MOVFF   RCREG,Atri
        MOVFF   Atri,cislo
        MOVLW   b'00000100'
```

```
        MOVWF  c7
        CALL   Detekce
        BTFSC  c6,0
        GOTO   ASCII

        GOTO   Znak1

;*******************************

ASCII

;Převod ASCII na čísla

        MOVLW  b'00110000'              ;desítky
        SUBWF  jedna,1                  ;Převádí ASCII znak na BCD hodnotu
        MOVLW  b'00110000'              ;Jednotky
        SUBWF  dva,1                    ;Převádí ASCII znak na BCD hodnotu


;Kontrola, že přijaté znaky jsou 0 - 9

        BTFSC  jedna,7                  ; Zjistuje, jestli horni 4 bity jsou nulové
        GOTO   Nulovani         ;Pokud nejsou nulové všechny výstupy se vynulují
        BTFSC  jedna,6
        GOTO   Nulovani
        BTFSC  jedna,5
        GOTO   Nulovani
        BTFSC  jedna,4
        GOTO   Nulovani

        BTFSC  dva,7
        GOTO   Nulovani
        BTFSC  dva,6
        GOTO   Nulovani
        BTFSC  dva,5
        GOTO   Nulovani
        BTFSC  dva,4
        GOTO   Nulovani


        MOVLW  h'a'                     ; Zjišťuje se, jestli přijaté číslo není větší než 9
        SUBWF  jedna,1                  ; Pokud je větší všechny výstupy se vynulují
        BTFSS  jedna,7
        GOTO   Nulovani
        MOVLW  h'a'
        ADDWF  jedna,1

        MOVLW  h'a'
        SUBWF  dva,1
        BTFSS  dva,7
        GOTO   Nulovani
        MOVLW  h'a'
        ADDWF  dva,1



        MOVLW  h'a'
        MOVWF  c1
        CLRWDT

aaa
        MOVF   jedna,0                  ;Desítky se vynásobí deseti
        ADDWF  mezivysledek,1
        DECFSZ c1
        GOTO aaa

        MOVF   mezivysledek,0   ; a sečtou se s jednotkami
        ADDWF  dva,1                    ; Výsledek se uloží do proměnné „dva“


;Načtení adresy přijímače
        CLRF   adresa

   BTFSC  PORTA,0                       ; Do bitů proměnné "adresa" se uloží hodnoty vstupů
        BSF            adresa,0         ;RA0 a RA1
```

```
        BTFSC   PORTA,1
        BSF             adresa,1

        MOVLW  d'24'                    ; K proměnné „dva“ se přičte dekadických 24
        ADDWF dva,1

        INCF    adresa                  ; Inkrementace o 1 proměnné "adresa"
ccc
        MOVLW  d'24'                    ; Cyklus odečte od proměnné "adresa" dekadických 24
        SUBWF  dva,1                    ; právě tolikrát kolik je hodnota "adresa"
        DECFSZ adresa                   ; Výsledek se uloží do  „dva“
        GOTO ccc


        movlw b'00000000' ;Všechny výstupy nastaví do log. 0
        movwf PORTA
        movwf PORTB
        movwf PORTC
        movwf PORTD
        movwf PORTE



        MOVLW  d'1'             ; Podprogram "Hledani" zjišťuje jestli hodnota uložená ve
        CALL Hledani            ; W registru je stejná jako hodnota v proměnné „dva“
        BTFSC   c5,0
        BSF     PORTA,2                 ; Pokud je stejná nastaví příslušný výstup na log 1

        BCF     c5,0                    ; c5 je proměnná, kterou vrací podprogram a určuje
        MOVLW  d'2'             ; jestli je, nebo není hodnota ve "W" a „dva“ stejná
        CALL Hledani
        BTFSC   c5,0
        BSF     PORTA,3

        BCF     c5,0
        MOVLW  d'3'
        CALL Hledani
        BTFSC   c5,0
        BSF     PORTA,5

        BCF     c5,0
        MOVLW  d'4'
        CALL Hledani
        BTFSC   c5,0
        BSF     PORTE,0

        BCF     c5,0
        MOVLW  d'5'
        CALL Hledani
        BTFSC   c5,0
        BSF     PORTE,1

        BCF     c5,0
        MOVLW  d'6'
        CALL Hledani
        BTFSC   c5,0
        BSF     PORTE,2

        BCF     c5,0
        MOVLW  d'7'
        CALL Hledani
        BTFSC   c5,0
        BSF     PORTC,0

        BCF     c5,0
        MOVLW  d'8'
        CALL Hledani
        BTFSC   c5,0
        BSF     PORTC,1

        BCF     c5,0
        MOVLW  d'9'
        CALL Hledani
        BTFSC   c5,0
        BSF     PORTC,2
```

```
BCF     c5,0
MOVLW   d'10'
CALL Hledani
BTFSC   c5,0
BSF     PORTC,3

BCF     c5,0
MOVLW   d'11'
CALL Hledani
BTFSC   c5,0
BSF     PORTD,0

BCF     c5,0
MOVLW   d'12'
CALL Hledani
BTFSC   c5,0
BSF     PORTD,1

BCF     c5,0
MOVLW   d'13'
CALL Hledani
BTFSC   c5,0
BSF     PORTD,2

BCF     c5,0
MOVLW   d'14'
CALL Hledani
BTFSC   c5,0
BSF     PORTD,3

BCF     c5,0
MOVLW   d'15'
CALL Hledani
BTFSC   c5,0
BSF     PORTC,4

BCF     c5,0
MOVLW   d'16'
CALL Hledani
BTFSC   c5,0
BSF     PORTC,5

BCF     c5,0
MOVLW   d'17'
CALL Hledani
BTFSC   c5,0
BSF     PORTD,4

BCF     c5,0
MOVLW   d'18'
CALL Hledani
BTFSC   c5,0
BSF     PORTD,5

BCF     c5,0
MOVLW   d'19'
CALL Hledani
BTFSC   c5,0
BSF     PORTD,6

BCF     c5,0
MOVLW   d'20'
CALL Hledani
BTFSC   c5,0
BSF     PORTD,7

BCF     c5,0
MOVLW   d'21'
CALL Hledani
BTFSC   c5,0
BSF     PORTB,0

BCF     c5,0
MOVLW   d'22'
```

```
        CALL Hledani
        BTFSC   c5,0
        BSF     PORTB,1

        BCF     c5,0
        MOVLW   d'23'
        CALL Hledani
        BTFSC   c5,0
        BSF     PORTB,2

        BCF     c5,0
        MOVLW   d'24'
        CALL Hledani
        BTFSC   c5,0
        BSF     PORTB,3

        GOTO Main


        ;Podprogramy:

;Podprogram hledani jestli se nactene cislo rovna prijatemu

Hledani

        XORWF   dva,0           ; Provádí se XOR "W" a „dva“
        MOVWF   c4
        CLRWDT

        BTFSC   c4,0            ; Pokud je "c4" rovno nule "W" i „dva“ mají stejnou hodnotu
        GOTO    bbb                     ; Pokud stejné nejsou podprogram vrací "c5,0" s nulovou hodnotou
        BTFSC   c4,1
        GOTO    bbb
        BTFSC   c4,2
        GOTO    bbb
        BTFSC   c4,3
        GOTO    bbb
        BTFSC   c4,4
        GOTO    bbb
        BTFSC   c4,5
        GOTO    bbb
        BTFSC   c4,6
        GOTO    bbb
        BTFSC   c4,7
        GOTO    bbb

        bsf     c5,0                    ; Nastaví první bit "c5" do logické 1

bbb

RETURN


;podprogram pro detekci 0xOD v prijatem znaku

Detekce

        CLRWDT
        MOVLW   b'00001101'
        XORWF   cislo

        BTFSC   cislo,0
        GOTO    ddd
        BTFSC   cislo,1
        GOTO    ddd
        BTFSC   cislo,2
        GOTO    ddd
        BTFSC   cislo,3
        GOTO    ddd
        BTFSC   cislo,4
        GOTO    ddd
        BTFSC   cislo,5
```

```
        GOTO    ddd
        BTFSC   cislo,6
        GOTO    ddd
        BTFSC   cislo,7
        GOTO    ddd

        BSF             c6,0

        BTFSC   c7,0
        GOTO    xxx
        BTFSC   c7,1
        GOTO    yyy
        BTFSC   c7,2
        GOTO    zzz

xxx

        MOVFF   Adva,jedna
        MOVFF   Atri,dva
        GOTO    ddd

yyy

        MOVFF   Atri,jedna
        MOVFF   Ajedna,dva
        GOTO    ddd

zzz

        MOVFF   Ajedna,jedna
        MOVFF   Adva,dva

ddd

RETURN

end
```

9.3 Programy v hexadecimálním tvaru

9.3.1 Program procesoru obsluhující modul klávesnice

```
:020000040000FA
:100000000F0E036E00802C0EAC6E800EAB6E190EC0
:10001000AF6E896A070EC16E7F0E926EBF0E946E30
:10002000280E806E3F0E826E276A266A246A256A31
:10003000236A0400000E236E929681B0010E81B2F5
:10004000020E81B4030E81B6040E81B8050E81BA8A
:10005000060E81BC070E81BE080E9286929A81B070
:10006000090E81B20A0E81B40B0E81B60C0E81B856
:100070000D0E81BA0E0E81BC0F0E81BE100E928A3B
:1000800004009490 81B0110E81B2120E81B4130E4F
:1000900081B6140E81B8150E81BA160E81BC170EEA
:1000A00081BE180E9480949281B0190E81B21A0EFE
:1000B00081B41B0E81B61C0E81B81D0E81BA1E0EB6
:1000C00081BC1F0E81BE200E94820400949481B0E6
:1000D000210E81B2220E81B4230E81B6240E81B886
:1000E000250E81BA260E81BC270E81BE280E94846F
:1000F000949681B0290E81B22A0E81B42B0E81B65E
:100100002C0E81B82D0E81BA2E0E81BC2F0E81BE11
:10011000300E94860400949881B0310E81B2320E74
:1001200081B4330E81B6340E81B8350E81BA360EE5
:1001300081BC370E81BE380E9488949A81B0390EF6
:1001400081B23A0E81B43B0E81B63C0E81B83D0EB1
:1001500081BA3E0E81BC3F0E81BE400E948A236E52
:10016000040023B0DEEF00F023B2DEEF00F023B492
:10017000DEEF00F023B6DEEF00F023B8DEEF00F094
:1001800023BADEEF00F023BCDEEF00F023BEDEEF8B
:1001900000F02790266A19EF00F0FF0E226E0F0E76
:1001A000216E0A0E206E0400202ED3EF00F0212EC7
:1001B000D1EF00F0222ECFEF00F012002352261ACA
:1001C000CDEC00F026B019EF00F026B219EF00F0E8
:1001D00026B419EF00F026B619EF00F026B819EF93
:1001E00000F026BA19EF00F026BC19EF00F026BE89
:1001F00019EF00F027B019EF00F02780010E256EEF
:1002000023C024F02406040024B023EF01F024B21C
:1002100023EF01F024B423EF01F024B623EF01F023
:1002200024B823EF01F024BA23EF01F024BC23EF1C
:1002300001F024BE23EF01F0000E256E246E52EC77
:1002400001F015EF00F004000A0E245E24BE44EF16
:1002500001F024B04DEF01F024B24DEF01F024B4D1
:100260004DEF01F024B64DEF01F024B84DEF01F051
:1002700024BA4DEF01F024BC4DEF01F0000E246EC6
:1002800052EC01F015EF00F004000A0E2426010ED6
:10029000255E52EC01F015EF00F00400010E25265A
:1002A00023EF01F00400300E2426300E2526040032
:1002B000ACA258EF01F025C0ADFF0400ACA25EEF88
:1002C00001F024C0ADFF0400ACA264EF01F00D0EFC
:1002D000AD6E0400ACA26AEF01F00A0EAD6E266AA4
:0202E00012000A
:00000001FF
```

9.3.2 Program procesoru obsluhující modul výstupů

```
:020000040000FA
:100000000F0E036E00802C0EAC6E900EAB6E190EB0
:10001000AF6E896A070EC16E000E806E816E826EB1
:10002000836E846E130E926EE00E936E800E946E4D
:10003000000E956E080E966E0400000E806E816EA6
:10004000826E836E846E0400256A266A336A346A7F
:10005000356A296A316A040032909EAA2BEF00F0BB
:10006000AECF33F033C028F0010E366E52EC01F003
:1000700032B05DEF00F0040032909EAA3BEF00F03A
:10008000AECF34F034C028F0020E366E52EC01F0E0
:1000900032B05DEF00F0040032909EAA4BEF00F00A
:1000A000AECF35F035C028F0040E366E52EC01F0BC
:1000B00032B05DEF00F02BEF00F0300E255E300E19
:1000C000265E25BE1CEF00F025BC1CEF00F025BA13
:1000D0001CEF00F025B81CEF00F026BE1CEF00F06E
:1000E00026BC1CEF00F026BA1CEF00F026B81CEF6F
:1000F00000F00A0E255E25AE1CEF00F00A0E252644
:100100000A0E265E26AE1CEF00F00A0E26260A0E08
:10011000206E040025502926202E8AEF00F0295059
:1001200026262236A80B0238080B22382180E2626DA
:10013000232A180E265E232E99EF00F0000E806E03
:10014000816E826E836E846E010E35EC01F031B0EB
:1001500080843190020E35EC01F031B08086319010
:10016000030E35EC01F031B0808A3190040E35EC8D
:1001700001F031B084803190050E35EC01F031B0E2
:10018000848231900 60E35EC01F031B084843190D8
:10019000070E35EC01F031B082803190080E35EC5D
:1001A00001F031B082823190090E35EC01F031B0AE
:1001B000828431900A0E35EC01F031B082863190A4
:1001C0000B0E35EC01F031B0838031900C0E35EC24
:1001D00001F031B0838231900D0E35EC01F031B079
:1001E000838431900E0E35EC01F031B0838631906E
:1001F0000F0E35EC01F031B082883190100E35ECE5
:1002000001F031B0828A3190110E35EC01F031B03D
:10021000838831901 20E35EC01F031B0838A319031
:10022000130E35EC01F031B0838C3190140E35ECA7
:1002300001F031B0838E3190150E35EC01F031B004
:10024000818031901 60E35EC01F031B08182319011
:10025000170E35EC01F031B081843190180E35EC79
:1002600001F031B0818623EF00F02618306E0400D3
:1002700030B051EF01F030B251EF01F030B451EF36
:1002800001F030B651EF01F030B851EF01F030BA63
:1002900051EF01F030BC51EF01F030BE51EF01F0F1
:1002A0003180120004000D0E281A28B087EF01F0EB
:1002B00028B287EF01F028B487EF01F028B687EF66
:1002C00001F028B887EF01F028BA87EF01F028BCC9
:1002D00087EF01F028BE87EF01F0328036B077EF6C
:1002E00001F036B27DEF01F036B483EF01F034C097
:1002F00025F035C026F087EF01F035C025F033C07A
:1003000026F087EF01F033C025F034C026F012004C
:00000001FF
```