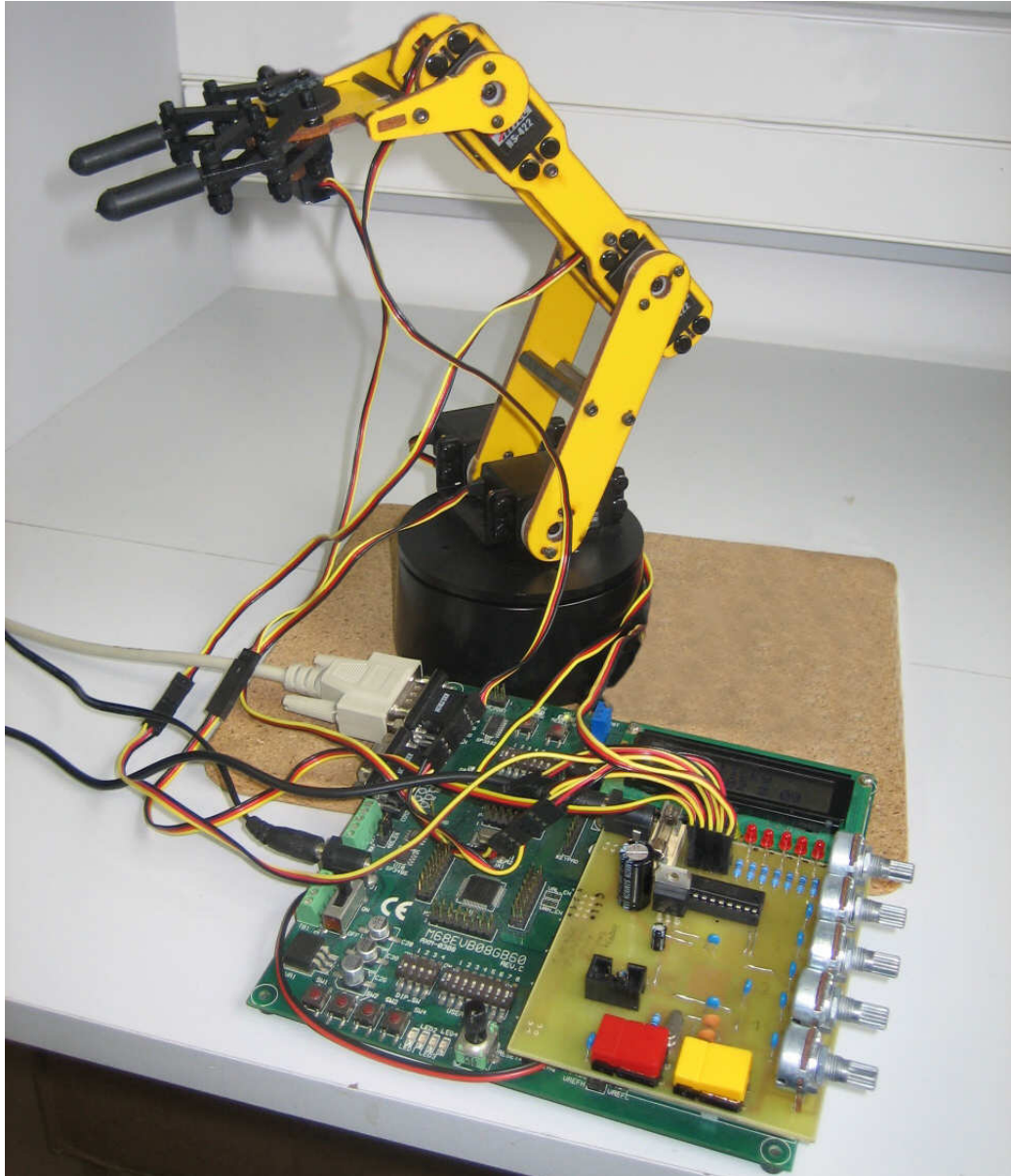


ZÁVĚREČNÁ PRÁCE Z PŘEDMĚTU MIROPOČÍTAČE

ROBOT LYNX 5



DOKUMENTACE & NÁVOD

Modul rozhraní pro připojení robota LYNX 5

Seznamte se s modulem rozhraní pro řízení robota, jeho ovládáním a signalizací poruch. Vyzkoušejte si ovládání robota pomocí rozhraní v manuálním režimu abyste získali představu o jeho pohybových možnostech. Seznamte se důkladně s funkcemi z knihovny „pwm.h“. Před tím, než začnete programovat, zamyslete se nad tím, jak bude váš program fungovat. V nejlepší případě si nakreslete vývojový diagram. Ušetříte si tím mnoho času při realizaci úlohy.

Úvod

Hlavní funkce modulu rozhraní je kontrola řídicích PWM signálů z vývojového kitu a jejich převod z napěťové úrovně 3,3V se kterou pracuje vývojový kit na úroveň 5V, se kterou pracuje elektronika servopohonů robota. Pokud jsou PWM signály na výstupu vývojového kitu správně generovány, jsou stejné signály o napěťové úrovni 5V na výstupu modulu rozhraní.

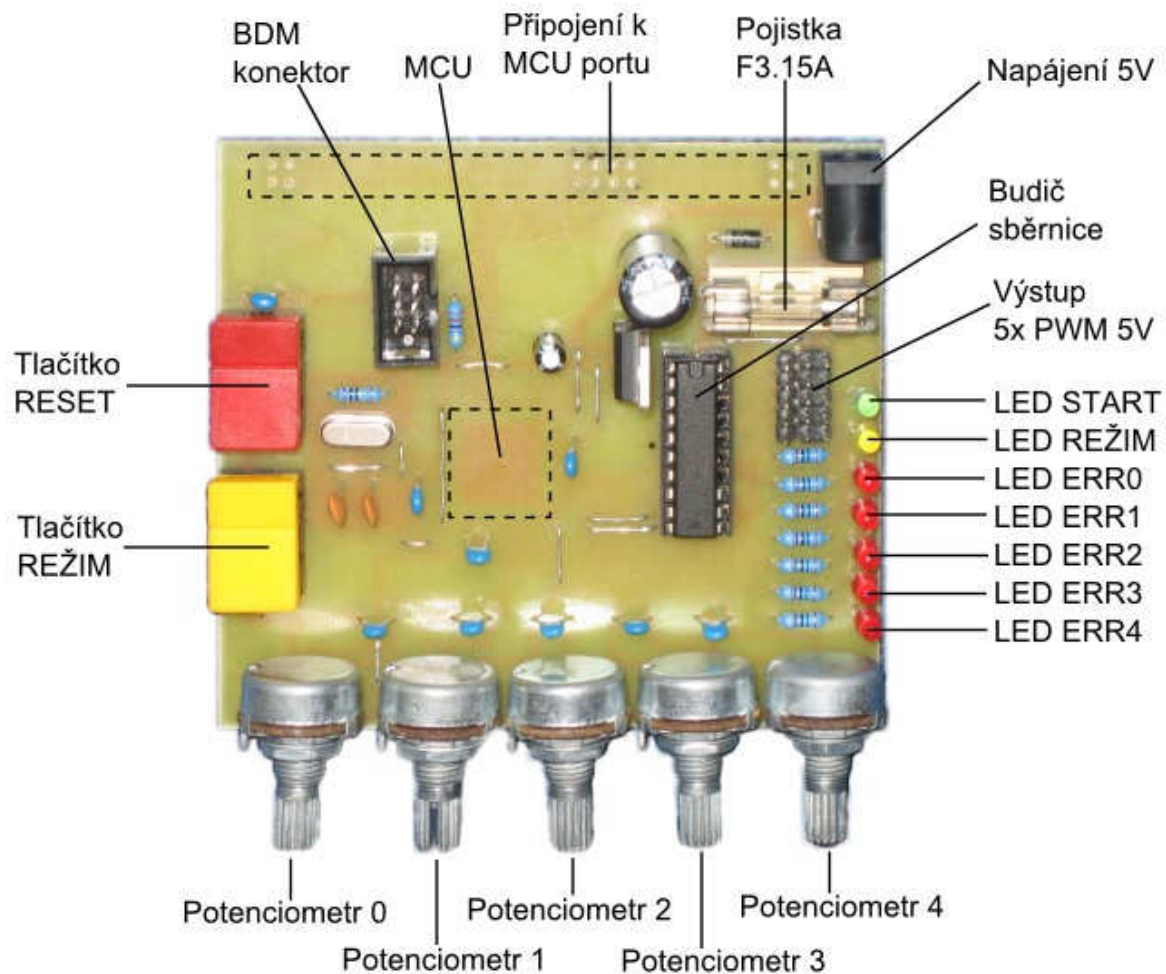
Pro snadnou programovou obsluhu je napsána knihovna podprogramů v jazyce C „pwm.h“, pomocí které lze generovat PWM s libovolnou periodou (v rámci možnosti časovače MPC) a střídou. Tato knihovna obsahuje také funkce přímo vytvořené pro ovládání jednotlivých servopohonů i celého robota, které můžete použít ve vašem programu aniž byste museli znát princip fungování servopohonů a vlastnosti PWM signálu, pro jejich řízení.

Připojení modulu rozhraní a jeho zprovoznění

Ujistěte se, zda je vývojový kit vypnutý. Zasuňte modul rozhraní do MCU_PORTu tak, aby modul rozhraní zakryl nepájivé kontaktní pole. Připojte jednotlivé pohony k modulu rozhraní přesně tak, jak je to znázorněno na obrázku níže. Dbejte na správnou polaritu při zapojování konektorů jednotlivých servopohonů. Černý kabel by měl být na straně u signalizačních LED a žlutý na straně IO. Pokud zapojíte pohony v obráceném pořadí (například prohodíte pohon zápěstí a ramena) bude sice fungovat kontrola PWM signálů, ale nebude fungovat kontrola povolených poloh robota! Po zapojení všech pohonů nejdříve připojte napájecí adaptér k modulu rozhraní a poté zapněte vývojový kit. Ihned po zapojení by se měla rozsvítit zelená LED. Pokud tato LED nesvítí zkontrolujte, zda není spálená pojistka, nebo napájecí zdroj.



Popis modulu rozhraní



Tlačítko RESET – použijete v případě chybné funkce modulu rozhraní (což by se stát nemělo).

Tlačítko REŽIM – slouží pro výběr ovládání servopohonů pomocí signálů z vývojového kitu, nebo manuálního ovládání pomocí pěti potenciometrů.

Potenciometr 0 – v manuálním režimu slouží pro ovládání pohonu základny robota.

Potenciometr 1 – v manuálním režimu slouží pro ovládání pohonu ramena robota.

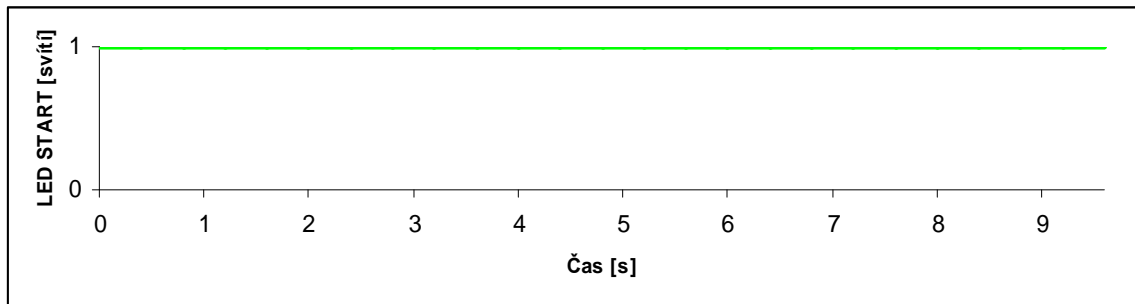
Potenciometr 2 – v manuálním režimu slouží pro ovládání pohonu loktu robota.

Potenciometr 3 – v manuálním režimu slouží pro ovládání pohonu zápěstí robota.

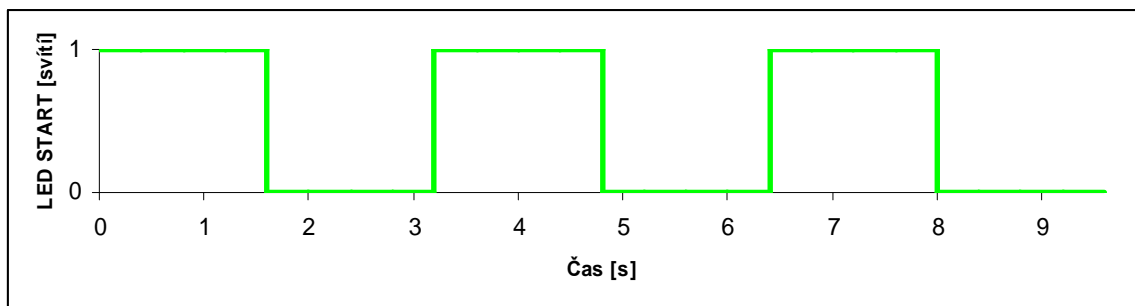
Potenciometr 4 – v manuálním režimu slouží pro ovládání pohonu chapadla robota.

LED START – signalizuje správnou činnost modulu a polohu servopohonů robota. Po inicializaci modulu LED svítí.

- LED svítí, pokud je robot v povolené poloze:

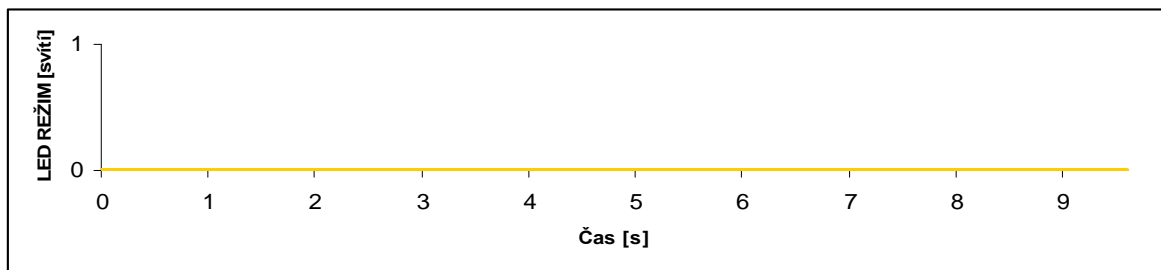


- LED bliká, pokud by řídicí signály mely robota natočit do havarijní polohy:

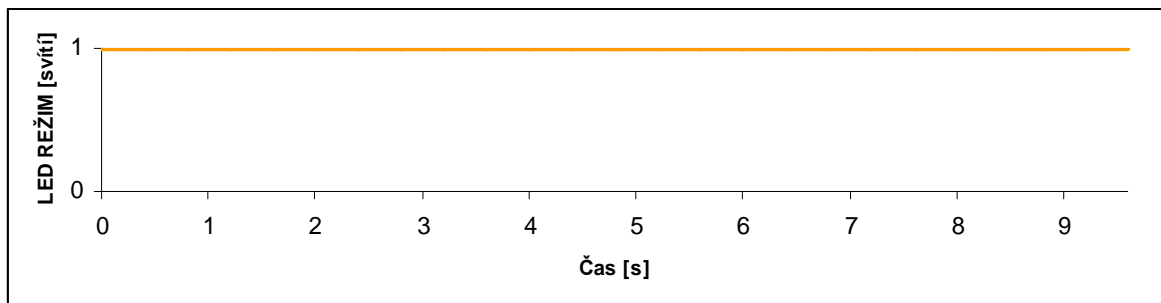


LED REŽIM – signalizuje režim ovládání.

- LED nesvítí, pokud je robot řízen signály z vývojového kitu:

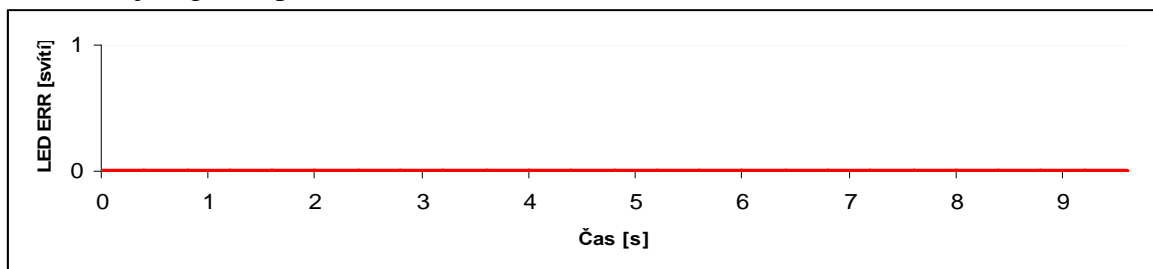


- LED svítí, pokud je robot řízen manuálně pomocí potenciometrů:

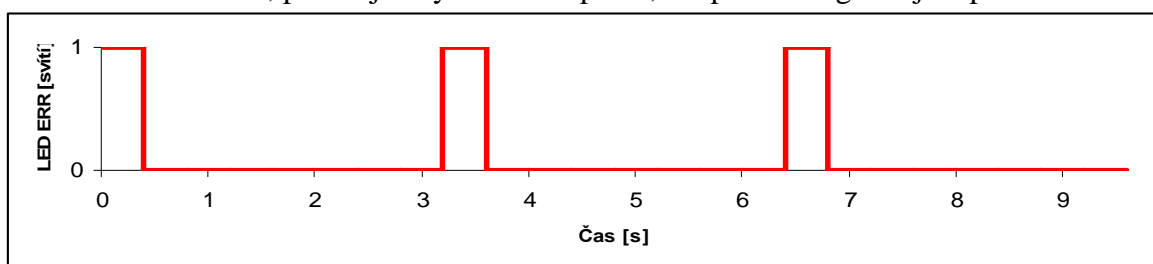


LED ERR0-4 – v automatickém režimu slouží pro signalizaci chyby PWM signálu. Číslování LED odpovídá pořadí kanálů TPM2 na vývojovém kitu (ERR0 signalizuje chybu TPM2CH0, ERR1 signalizuje chybu TPM2CH1, atd.).

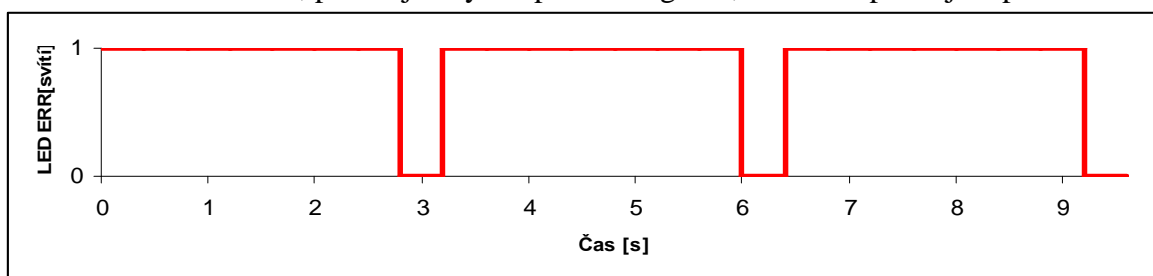
- Pokud je signál v pořádku LED nesvítí:



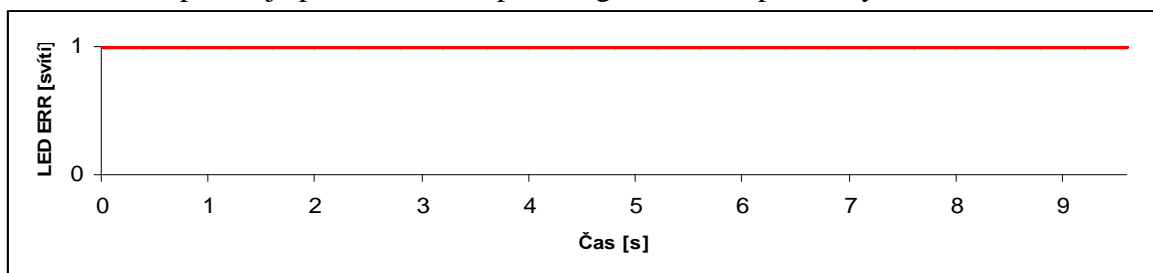
- LED krátce bliká, pokud je chybná délka pulsu, ale perioda signálu je v pořádku:



- LED krátce zhasíná, pokud je chybná perioda signálu, ale délka pulsu je v pořádku:



- LED svítí pokud je perioda i délka pulsu signálu mimo povolený rozsah:



Stručný popis funkcí z knihovny pwm.h

Knihovna „pwm.h“ pracuje s časovačem TPM2 a používá všechny jeho kanály. Proto se ve svém programu vyhněte použití tohoto časovače a pinů PTD3 – PTD7, které sdílejí kanály tohoto časovače. Při používání funkcí pro řízení pohonů robota odpovídají kanály časovače TPM2 jednotlivým pohonům:

- kanál 0 – základna
- kanál 1 – rameno
- kanál 2 – loket
- kanál 3 – zápěstí
- kanál 4 – chapadlo

int pwm_init(int perioda)

Nastaví dobu přetečení časovače 2 (tím se nastaví perioda PWM) a všechny jeho kanály do režimu PWM.

Vstupy:

- perioda <1 – 419> – nastavení periody PWM v milisekundách

Výstupy:

- 1 – vše proběhlo v pořádku

int pwm_duty_cycle(int kanal, int sirka)

Nastaví střihu PWM signálu na požadovaném kanálu TPM2.

Vstupy:

- kanal <0 – 4> – výběr kanálu
- sirka <0 – 10000> - střih 0 ≈ 0 %, 10000 ≈ 100 %

Výstupy:

- 0 – chyba vstupu
- 1 – vše proběhlo v pořádku

int servo_init(void)

Nastaví dobu přetečení časovače 2 (tím se nastaví perioda PWM) na 20ms a všechny jeho kanály do režimu PWM. Nastaví šířku pulsu na všech kanálech na 1,5 ms.

Vstupy:

- Tato funkce nemá žádné vstupy.

Výstupy:

- 0 – nezdařilo se
- 1 – vše proběhlo v pořádku

int servo_set(int kanal,int poloha)

Okamžité nastavení polohy vybraného servopohonu do zvolené polohy.

Vstupy:

- kanal <0 – 4> – výběr serva
- poloha <0 – 500> - poloha 0 ≈ první krajní poloha servopohonu (délka pulsu 1 ms), 500 ≈ druhá krajní poloha servopohonu (délka pulsu 2 ms).

Výstupy:

- 0 – chyba vstupu
- 1 – vše proběhlo v pořádku

int servo_get(int kanal)

Funkce vrací aktuální hodnotu polohy servopohonu, nebo informaci o chybě.

Vstupy:

- kanal <0 – 4> – výběr serva

Výstupy:

- <0 – 500> – poloha 0 ≈ první krajní poloha servopohonu (délka pulsu 1 ms), 500 ≈ druhá krajní poloha servopohonu (délka pulsu 2 ms).
- -1 – chyba vstupu
- -2 – nezdařilo se

int servo_speed(char speed)

Nastavuje rychlost pohybu servopohonu, při použití funkcí „int servo_move(int kanal, int pozPoloha“ „int robot_move(int kan0, int kan1, int kan2, int kan3, int kan4)“.

Vstupy:

- speed <0 – 4> – rychlost 0 ≈ nulová rychlost (servopohony stojí), rychlost 1 ≈ nejnižší rychlost (pohyb pohonu z jedné krajní polohy do druhé krajní polohy za 25 sekund), rychlost 4 ≈ nejvyšší rychlost (pohyb pohonu z jedné krajní polohy do druhé krajní polohy za 6,25 sekundy).

Výstupy:

- 0 – chyba vstupu
- 1 – vše proběhlo v pořádku

int servo_move(int kanal, int pozPoloha)

Plynulý pohyb vybraným pohonem do zvolené cílové polohy rychlostí nastavené pomocí funkce „int servo_speed(char speed)“.

Vstupy:

- kanal <0 – 4> – výběr serva
- poloha <0 – 500> – poloha 0 ≈ první krajní poloha servopohonu (délka pulsu 1 ms), 500 ≈ druhá krajní poloha servopohonu (délka pulsu 2 ms).

Výstupy:

- 0 – chyba vstupu
- 1 – vše proběhlo v pořádku

int robot_move(int kan0, int kan1, int kan2, int kan3, int kan4)

Plynulý pohyb celým robotem do zvolené cílové polohy rychlostí nastavené pomocí funkce „int servo_speed(char speed)“.

Vstupy:

- kanal0 <0 – 500> – požadovaná poloha servopohonu, který ovládá základnu robota. 0 ≈ první krajní poloha servopohonu (délka pulsu 1 ms), 500 ≈ druhá krajní poloha servopohonu (délka pulsu 2 ms).
- kanal1 <0 – 500> – požadovaná poloha servopohonů, které ovládají rameno robota. 0 ≈ první krajní poloha servopohonů (délka pulsu 1 ms), 500 ≈ druhá krajní poloha servopohonu (délka pulsu 2 ms).
- kanal2 <0 – 500> – požadovaná poloha servopohonu, který ovládá loket robota. 0 ≈ první krajní poloha servopohonu (délka pulsu 1 ms), 500 ≈ druhá krajní poloha servopohonu (délka pulsu 2 ms).
- kanal3 <0 – 500> – požadovaná poloha servopohonu, který ovládá zápěstí robota. 0 ≈ první krajní poloha servopohonu (délka pulsu 1 ms), 500 ≈ druhá krajní poloha servopohonu (délka pulsu 2 ms).
- kanal4 <0 – 500> – požadovaná poloha servopohonu, který ovládá chapadlo robota. 0 ≈ první krajní poloha servopohonu (délka pulsu 1 ms), 500 ≈ druhá krajní poloha servopohonu (délka pulsu 2 ms).

Výstupy:

- <0 – 4> – číslo prvního kanálu, na kterém byla zaznamenána chyba na vstupu
- 5 – vše proběhlo v pořádku

int is_final_position(void)

Tato funkce testuje zda se už robot dostal do polohy zadané v parametrech funkce „int robot_move(int kan0, int kan1, int kan2, int kan3, int kan4)“.

Vstupy:

- Tato funkce nemá žádné vstupy.

Výstupy:

- 0 – robot se pohybuje do žádané polohy
- 1 – robot se nachází v žádané poloze

void cekej_100ms(void)

Funkce spotřebuje přibližně 100 ms času mikropočítače.

Vstupy:

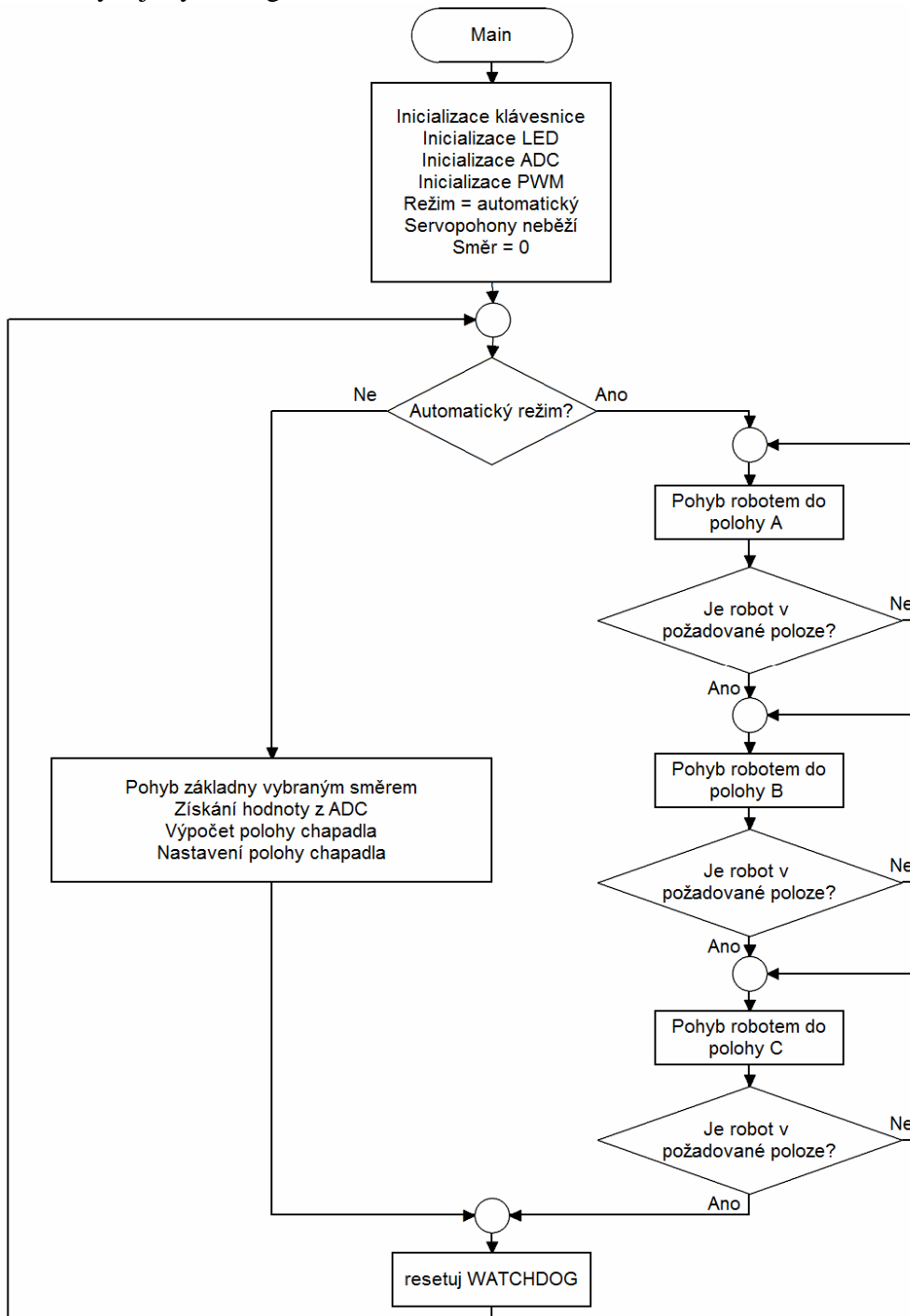
- Tato funkce nemá žádné vstupy.

Výstupy:

- Tato funkce nemá žádné výstupy.

Ukázkový program

Tlačítko SW1 slouží k výběru režimu. V automatickém i manuálním režimu je možnost pozastavit činnost robota tlačítkem SW2. Tlačítko SW3 je aktivní pouze v manuálním režimu a slouží k přepínání směru otáčení základny. Chapadlo robota se v manuálním režimu ovládá pomocí potenciometru. V automatickém režimu se pořád dokola postupně vykoná sekvence třech pohybů. Pokud dojde k požadavku na změnu režimu z automatického na manuální, dokončí se sekvence pohybů a až poté dojde k přepnutí. LED1 svítí v automatickém režimu. LED2 svítí pokud jsou pohony robota pozastaveny. LED3 a LED4 signalizují směr otáčení základny. Pro lepší pochopení je celý program znázorněn pomocí dvou vývojových diagramů:



```

void (*const obsluha)(void) @0xFFD2 = kbi_int;

void main(void)
{
    ulong temp;

    PTADD = 0x00;          // PTA4 vstupni rezim
    PTAPE = 0xF0;         // pull-up pro PTA4 zapnut

    PTFDD=0x0F;          // bit 0 portu F jako vystup
    PTFPE = 0x00;         // pull-up vypnuty
    PTFD=0xFF;           // zhasneme LED diody

    KBI1SC = 0x06;        // nast. ridiciho reg. KBI - preruseni povoleno
    KBI1PE = 0xF0;        // PTA4-7 vstup pro KBI

    ATD1C = 0xE4;         // zapnuti prevodniku, 8 bit vysledek
    ATD1PE = 1;           // pin PTB0 prepneme do rezimu vstupu A/D prevodniku

    servo_init();         // inicializace casovace 2

    EnableInterrupts;     /* enable interrupts */

    for(;;)
    {
        if(rezim == 1){    //Manualni rezim

            servo_move(0,smer);

            ATD1SC = 0;     // start prevodu, jednorazovy, kanal 0
            while (ATD1SC_CCF == 0) ;

            temp = ((ulong)ATD1RH*500)/255;
            servo_set(4,temp); // nastaveni chapadla
        }else{             // Automaticky rezim

            // poloha A
            do{robot_move(0,450,400,300,330);
            }while(!is_final_position());

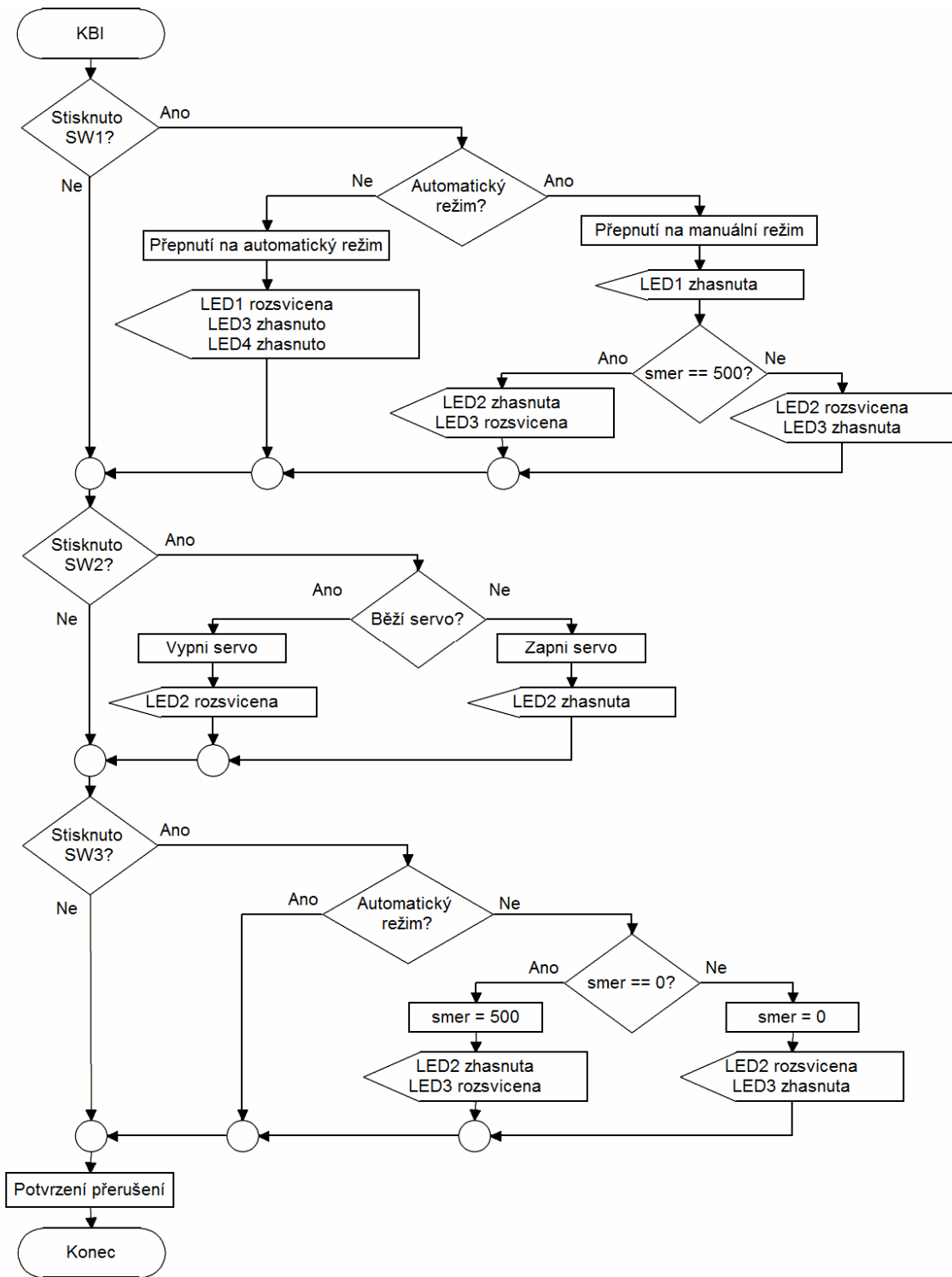
            // poloha B
            do{robot_move(200,250,100,500,330);
            }while(!is_final_position());

            // poloha C
            do{robot_move(0,300,450,500,220);
            }while(!is_final_position());

        }

        __RESET_WATCHDOG(); /* feeds the dog */
    } /* loop forever */
    /* please make sure that you never leave main */
}

```



```

#include <hidef.h>           /* for EnableInterrupts macro */
#include "derivative.h"     /* include peripheral declarations */
#include "pwm.h"

char rezim = 0;             //0 = automat
char bezi = 0;
int smer = 0;

interrupt void kbi_int(void)
{
    cekej_100ms();          // cekani na dozneni zakmitu na tlacitku
    // SW1 - volba rezimu
    if( PTAD_PTAD4 == 0 )
    {
        if ( rezim == 0 ){
            rezim = 1;
            PTFD_PTFD0 = 1;    //zhasnuti LED1
            // signalizace smeru leva/prava
            if(smer == 500){
                PTFD_PTFD2 = 1;    // zhasnuti LED3
                PTFD_PTFD3 = 0;    // rozsviceni LED4
            }else{
                PTFD_PTFD2 = 0;    // rozsviceni LED3
                PTFD_PTFD3 = 1;    // zhasnuti LED4
            }
        }
        else{
            rezim = 0;

            // signalizace leva/prava vypnuto
            PTFD_PTFD0 = 0;    // rozsviceni LED1
            PTFD_PTFD2 = 1;    // zhasnuti LED3
            PTFD_PTFD3 = 1;    // zhasnuti LED4
        }
    }
    // SW2 - START / STOP v manualnim rezimu
    if( PTAD_PTAD5 == 0 )
    {
        if(bezi == 1)
        {
            PTFD_PTFD1 = 0;    // rozsviceni LED2
            servo_speed(0);
            bezi = 0;
        }else{
            PTFD_PTFD1 = 1;    // zhasnuti LED2
            servo_speed(2);
            bezi = 1;
        }
    }
    // SW3 - vlevo / vpravo
    if( PTAD_PTAD6 == 0 )
    {
        if( rezim == 1 )      // manualni rezim
        {
            if(smer == 0)
            {
                PTFD_PTFD2 = 1;    // zhasnuti LED3
                PTFD_PTFD3 = 0;    // rozsviceni LED4
                smer = 500;
            }else{
                PTFD_PTFD2 = 0;    // rozsviceni LED3
                PTFD_PTFD3 = 1;    // zhasnuti LED4
                smer = 0;
            }
        }
    }
    KBI1SC_KBACK = 1;        // potvrd KBI modulu prijem preruseni
}

```

Zadání úlohy

Vytvořte program pro vývojový kit 68EVB08GB60, pomocí kterého bude možné ovládat činnost robota LYNX 5. Tlačítko SW1 bude sloužit pro výběr režimu. Režim může být buď manuální, nebo automatický. Režim bude možno změnit kdykoli během provádění programu. Při přepnutí režimu z manuálního na automatický pokračuje pohyb robota do polohy, do které se pohyboval před přepnutím na manuální režim. Robot si tedy pamatuje, kde skončil v automatickém režimu.

Manuální režim:

- Na prvním řádku displeje bude zobrazen text „Manuální“
- Potenciometr bude sloužit k ovládání pohonu základny robota. Na konci prvního řádku bude vypsána délka pulsu PWM signálu, který řídí otáčení základnou. Ostatní pohony robota se budou ovládat pomocí tlačítek.
- Tlačítko SW3 bude sloužit pro volbu pohonu, který se bude ovládat. Na druhém řádku displeje bude zobrazen název ovládaného pohonu („chapadlo“, „zápěstí“, „loket“, „rameno“) a délka řídicího pulsu PWM signálu pro vybraný pohon.
- Tlačítkem SW4 se bude přepínat mezi pohybem doleva (první krajní poloha) a pohybem doprava (2. krajní poloha).
- Pokud bude aktivní pohyb vybraného pohonu doleva bude tento stav signalizován pomocí rozsvícení LED2. V opačném případě, pokud bude aktivní pohyb pohonu doprava bude svítit LED3.
- Tlačítkem SW2 se zastaví / spustí vybraný pohon.
- Pokud vybraný pohon běží bude tento stav signalizován pomocí rozsvícení LED1, pokud bude pohon zastaven bude LED1 zhasnuta.
- Příklad výpisu na displej:

Manuální 1.00 ms
Chapadlo 1.84 ms

Automatický režim

- Využijte svou představivost a naprogramujte robota na libovolnou automatickou opakující se činnost (přemísťování předmětu z bodu A do bodu B). Fantazii se meze nekladou.
- Na prvním řádku displeje bude zobrazen text „Automatický“
- Na druhém řádku displeje bude zobrazena informace o postupu prováděné činnosti (např. „Postup: 42%“, nebo „Operace 12 z 17“);
- Tlačítkem SW2 bude možnost robota pozastavit. Při opětovném spuštění tohoto tlačítka bude robot pokračovat ve své činnosti přesně tam kde skončil.
- Signalizace zastavení robota bude signalizována stejně jako v manuálním režimu pomocí LED1.
- LED2 a LED3 nebudou mít v automatickém režimu žádnou signalizační funkci a proto budou zhasnuté.
- Tlačítka SW3 a SW4 nebudou mít na chování robota v automatickém režimu žádný vliv.
- Příklad výpisu na displej:

Automaticky
Operace 02 z 15

Použitá literatura

Doležel Josef: Rozhraní pro připojení výukového robota k vývojovému kitu M68EVB908GB60, bakalářská práce, 2013. Fakulta Aplikované informatiky, UTB ve Zlíně.