

STANISLOVAS PLESKAS

VALDIKLIAI

Praktiniai darbai

VILNIUS
2012

UDK 681.5 (076.1)

S.Pleskas.
Valdikliai
Praktiniai darbai
ISBN 978-609-408-493-5

Knyga skirta valdiklių programavimo pradinių įgūdžių suteikimui. Knygą sudaro keturi skyriai: mikrovaldiklio PIC16F84 programavimas, Arduino Uno modulio programavimas, Festo firmos programuojamo loginio valdiklio FC20 programavimas, valdiklio CROUZET Milenium3 programavimas.

Programos rašomos assemblerio, C, STL ir FBD kalbomis. Knygoje pateikiama informacija apie valdiklių parametrus, praktinių darbų atlikimo metodika ir praktinių darbų užduotys. Paaiškinama kaip programa įkeliama į valdiklį ir stebimas jos darbas.

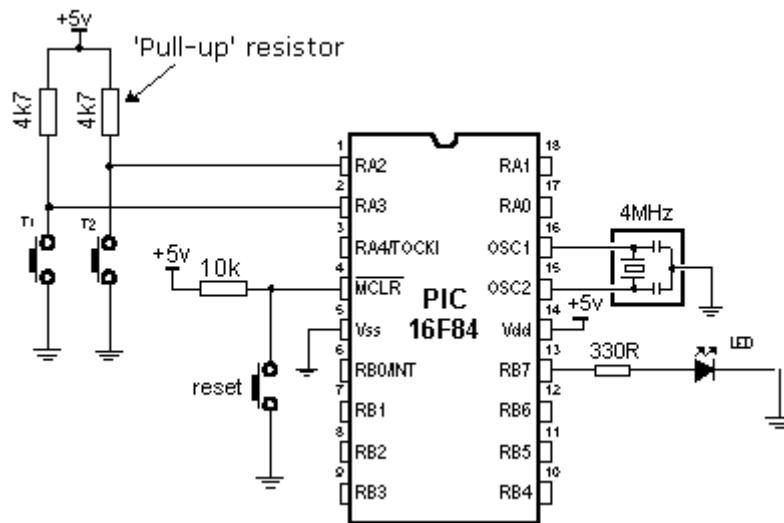
Leidinys skirtas neuniversitetinių aukštųjų mokyklų studentams, kurie projektuoja programuojamus elektronikos ir automatikos įtaisus.

© Stanislovas Pleskas

TURINYS

1. PIC16F84 MIKROVALDIKLIS	4
1.1. Praktinis darbas Nr1. Duomenų įrašymas į mikrovaldiklio PIC16F84A prievadus	5
1.2 Praktinis darbas Nr.2. Programa su ciklu.....	9
1.3 Praktinis darbas Nr.3. Programa su mygtuku.....	13
1.4 Praktinis darbas Nr 4. Paprogramės	17
1.5 Praktinis darbas Nr.5. Mikrovaldiklio užprogramavimas ir programos išbandymas	20
2. ARDUINO MODULIS	23
2.1. Arduino UNO modulio sandara	23
2.1.1 Arduino programos įdiegimas.....	24
2.1.2. Programavimo aplinka	25
2.1.3. Programos įkrovimas	26
2.2. Praktinis darbas Nr. 1 Mirksintis spinduolis.....	26
2.3. Praktinis darbas Nr 2 „Bėganti šviesa“	28
2.4. Praktinis darbas Nr.3. Pneumatikos cilindų valdymas.....	31
2.5. Praktinis darbas Nr.4 Nuolatinės srovės variklio programinis valdymas	34
3 „FESTO“ PROGRAMUOJAMAS LOGINIS VALDIKLIS	36
3.1. Valdiklio išvadai	36
3.2. Praktinis darbas Nr1. Nuosekli programa	37
3.2.1. Projekto sukūrimas.....	37
3.2.2. Programos rašymas	39
3.2.3. PLV prijungimas ir programavimas.....	40
3.2.4. Programos išsaugojimas ir pernešimas.	41
3.2.5. Darbas Online režimu	42
3.3 Praktinis darbas Nr. 2. Programa su laikmačiu ir skaitikliu.....	43
3.4. Praktinis darbas Nr.3. Paprogramės	46
4. „CROUZET“ PROGRAMUOJAMAS LOGINIS VALDIKLIS	48
4.1. Valdiklio praktinių darbų stendas	48
4.2. Praktinis darbas Nr.1 Programos rašymas ir darbo simuliacija.....	50
4.3. Praktinis darbas Nr2. Programa su laikmačiu	53
4.4. Praktinis darbas Nr.3. Programa su skaitikliu	56
4.5. Praktinis darbas Nr.4. Pneumatikos cilindų programinis valdymas.....	59
INFORMACIJOS ŠALTINIAI	61

1. PIC16F84 MIKROVALDIKLIS



1.1 pav. Mikrovaldiklio įjungimo schema su kvarciniu rezonatoriumi

1.1 pav. parodytoje schemoje išvadai RA2 ir RA3 yra įėjimai ir prie jų prijungti valdymo mygtukai. Išvadas RB7 yra išėjimas. Prie jo prijungtas indikacinis spinduolis. Plačiau apie schemą [1] 79 p.

PIC16F84 mikrovaldikliai (MV) dažnai programuojami supaprastinta assemblerio kalba. Naudojamos 35 komandos. Komandų sąrašas yra [1] 59 p. Komandų paaiškinimas yra [1] 60-71 p.

Programoje naudojami skaičiai gali būti užrašomi įvairiai. Pvz. skaičius 12 užrašomas:

- ◆ dvejetainiame pavidale B'00001100',
- ◆ dešimtainiame – D'12',
- ◆ šešiolyktainiame – H'0C' arba 0x0C, 0CH. Skaičius nuo 0 iki 9 galima rašyti įprastai.

Programos sandara aprašyta [1] (52, 57 ÷ 58) p. Programa prasideda pavadinimu. Čia gali būti nurodytas autorius bei aprašyta programos paskirtis, veikimas. Rekomenduojama trumpai aprašyti kas jungiama prie mikrovaldiklio išvadų. Visa tai komentarai. Prieš juos turi būti kabliataškis.

Pvz.

; Mirksintis spinduolis

; Spinduolis prijungtas prie RA0, mygtukas – RB0

Pirmoji programos eilutė pradedama rašyti atitrukta nuo krašto (be kabliataškio). Čia nurodomas mikrovaldiklio tipas

LIST P=16F84

Kitos dvi eilutės – konfigūracija. Antroje eilutėje nurodyta, kad bus naudojamas standartinis priskyrimų failas *p16f84.inc*. [1] 89 p. Jis turi būti toje pat direktorijoje, kur ir programa. Trečioje eilutėje nurodyta, kad išjungtas programos nuskaitymo draudimas *CP_OFF*, išjungtas sargybinis laikmatis (Watchdog) *WDT_OFF*, įjungtas maitinimo įjungimo laikmatis *PWRTE_ON*, pasirinktas kvarcinis rezonatorius *HS_OSC*.

Pvz.

LIST P=16F84

#include "p16f84.inc"

__CONFIG_CP_OFF&_WDT_OFF&_PWRTE_ON&_HS_OSC

Konfigūraciją galima atlikti ir mikrovaldiklio užprogramavimo metu. Tada šios dvi eilutės programoje nereikalingos.

Valdikliai. Praktiniai darbai

1.1. Praktinis darbas Nr1. Duomenų įrašymas į mikrovaldiklio PIC16F84A prievadus

DARBO TIKSLAS. Išstudijuoti MV PIC16F84A prievadus, duomenų atminties specialiuosius registrus STATUS, TRISA, TRISB, PORTA, PORTB, assemblerio direktyvas LIST, END ir komandas CLRF, BSF, BCF, MOVLW, MOVWF.

DARBO PRIEMONĖS. Kompiuteris, programa MPLAB v.5.4 arba MPLAB v.6.0

TEORINĖ DALIS

Programa gali būti rašoma bet koku teksto redaktoriumi. Rekomenduojamas „notepad“. Parašytas failas įrašomas į kompiuterį su plėtiniu asm. Pvz., *mirksi.asm*. Pavadinime gali būti iki 8 ženklų. Lietuviškos raidės nenaudojamos.

Pirmoji programa parodyta žemiau. Ji įrašo į išėjimo registrą PORTA skaičių 5.

```

;Skaiciaus irasymas.
LIST          P=16f84
;priskyrimai, registrai
PORTA        equ    5H
PORTB        equ    06
TRISA        equ    0x85
TRISB        equ    86H
STATUS       equ    03
;priskyrimai, registrų bitai
RP0          equ    05
; inicializacija
INIT  BSF          STATUS, 5
      CLRF         TRISA
      CLRF         TRISB
      BCF          STATUS, 5
;programos branduolys
Main  CLRF         PORTA
      CLRF         PORTB
      MOVLW       0x5
      MOVWF       PORTA
      GOTO        $
      END

```

Programoje pirmame (iš kairės) stulpelyje rašomos žymės. Jos naudojamos peršokant iš vienos programos vietos į kitą. Antrame stulpelyje rašoma komanda. Trečiame stulpelyje rašomas operandas – registro pavadinimas arba skaičius. Po kablelio dažniausiai nurodomas bito numeris. Viena programos eilutė paaiškinta 1.1 lentelėje.

1.1 lentelė. Programos eilutės pavyzdys

Žymė	Komanda	Operandas	Bito Nr.
INIT	BSF	STATUS	5

Kai kuriose komandose po kablelio rašomas 1 arba 0 nurodo, kur siunčiamas operacijos rezultatas.

Programos priskyrimų dalyje kiekvienam programoje naudojamam registru suteikiamas adresas [1] 2.2.4 skyrius. Adresai nurodomi 16-ainėje sistemoje. Tipiniai registrų adresai ir bitų numeriai yra [1] 89 – 90 p. Pvz., registru PORTB šioje programoje suteiktas adresas 06, registru STATUS suteikiamas adresas 03, registro TRISA adresas yra 0x85.

Toliau programos priskyrimų dalyje nurodomi registrų bitų numeriai (gali būti nuo 0 iki 7). Pvz., bitas RP0 yra 5- as.

Kita programos dalis yra inicializacija [1] 2.2.6 skyrius. Joje yra užprogramuojama, kurie mikrovaldiklio išvadai bus išėjimai, o kurie įėjimai. Inicializacija pradedama eilute BSF STATUS, 5. Ši eilutė pakeičia registrų banką iš nulinio (bank0) į pirmą (bank1) [1] 12 p. Inicializacija baigiama eilute BCF STATUS, 5. Ši eilutė atkeičia registrų bankus iš pirmo į nulinį.

Jei registro PORTA išvadai turi būti išėjimai, tai inicializacijoje registre TRISA įrašomi 0. Jei išvadai turi būti įėjimai – įrašomas 1. Analogiškai su registru PORTB.

Pvz., į registrą TRISA įrašius dvejetainį skaičių B'00101100' registro PORTA išvadai RA2, RA3 ir RA5 bus įėjimai, o visi kiti – išėjimai. Prie įėjimų galima jungti valdymo mygtukus, o prie išėjimų – pvz., indukacinius spinduolius.

Kita programos dalis yra programos branduolys (Main). Pirmoji programos eilutė išvalo registrą PORTA (registro visi bitai tampa 0), antroji – išvalo registrą PORTB. Po to skaičius 5 įrašomas į darbinį registrą W. Kitoje eilutėje skaičius 5 įrašomas į registrą PORTA. Po to programa stabdoma ir registre PORTA matome rezultatą. Programos pabaigą rodo direktyva END.

Programoje naudojamos komandos paaiškinamos žemiau.

BSF – suteikia nurodyto registro nurodytam bitui 1 reikšmę. Čia STATUS registro 5 bitas tampa 1.

CLRF –išvalo nurodytą registrą. Čia TRISA. Visiems aštuoniems registro bitams suteikiama 0 reikšmė.

BCF- suteikia nurodyto registro nurodytam bitui 0 reikšmę. Čia STATUS registro 5 bitas tampa 0

MOVLW – įrašo nurodytą skaičių į darbinį registrą. Nusiųsti (MOV) skaičių (L) į darbinį registrą W. Po komandos įvykdymo skaičius 5 (B'00000101) bus registre W .

MOVWF – nusiunčia skaičių, esantį darbiniam registre W į nurodytą registrą, čia PORTA. Po komandos įvykdymo skaičius 5 bus registre PORTA

GOTO \$ – stabdo programą šioje eilutėje. Tai nestandartinė komanda. Vietoje jos programa stabdoma taip:

```
stop    NOP
        GOTO stop
```

NOP – palaukti vieną taktą (pvz., 1 μ s). GOTO – peršokti prie nurodytos žymės, čia stop.

Kai įvykdoma programa rezultato reikia ieškoti registruose PORTA, PORTB, darbiniam registre W. Be to gali keistis STATUS registro C ir Z bitai.

DARBAS SU MPLAB PROGRAMA [1] 96 p

Parašius programą ir į kompiuterį įrašius failą su plėtiniu asm, paleidžiama programa MPLAB. Pagrindinio lango meniu juosta parodyta 1.2 pav. Mygtukų paskirtis: 1 – programos paleidimas normaliu režimu, 2 – programos stabdymas, 3 – programos žingsnis, 4 – reset, 5 – stebėjimo langas, 6 – kompiliavimo mygtukas.



1.2 pav. Programos pagrindinis langas

Vadovaujantis nurodymais knygoje [1] 96p į programą įkeliamas asm failas. Ištaisomos klaidos ir, spustelėjus 6-ą mygtuką, sukuriama programos kodas – failas tuo pačiu pavadinimu su plėtiniu hex. Šis kodas įkeliamas į mikrovaldiklį specialiu įtaisu – programatoriumi.

EKSPERIMENTINĖ DALIS

DARBO ETAPAI:

- programos assembleriu rašymas,
 - įkėlimas į MPLAB aplinką,
 - programos darbo simuliacija kompiuteryje,
 - programos klaidų požymių išsiaiškinimas.
1. Nukopijuojama 5 psl. parodyta programa ir sukuriamas failas su plėtiniu asm. Pavadinimą sudaro 8-ios studento pavardės raidės. Failas įrašomas kompiuteryje į direktoriją „*Darbai*“.
 2. Paleidžiama programa MPLAB.
 3. Įkeliamas asm failas į MPLAB aplinką. Jei yra klaidų, atsidaro klaidų langas kuriame rodomos klaidos. Jos ištaisomos ir uždaromas klaidų langas.
 4. Kompiuterio ekrane atidaromi langai:
 - a) stebėjimo langas (5 mygtukas). Atsidarius langui pasirenkama PORTA, Propertes, Binary, Add ir PORTB, Add. Po to langas uždaromas;
 - b) specialių registrų SFR langas;
 - c) sukurtas asm failas FILE – OPEN – pažymimas asm failas , OK. Po to reikia spūstelėti RESET (4 mygtukas) ir turi atsirasti asm faile juoda linija. Tai reiškia, kad programos darbą galima simuliuoti.
 5. Simuliuojamas programos darbas po vieną žingsnį. Stebima komanda juodoje eilutėje. Išsiaiškinama, ką registruose turi pakeisti ši komanda. Po to spaudžiamas trečias mygtukas ir stebimi SFR lange pasikeitimai registruose. Pasikeitimai rodomi raudona spalva. Užrašomos išvados.

 6. Simuliuojamas programos darbas automatiniame sulėtintame režime. Pasirenkama Debug-Run - Animate arba spaudžiama Ctrl + F9. Matoma judanti juoda linija. Norint programą pakartoti nuo pradžių spaudžiamas Reset mygtukas 4. Užrašomos išvados.

 7. Išsiaiškinami programos klaidų požymiai:
 - a) Ar rašant programą turį įtakos didžiosios ir mažosios raidės. Viename programos stulpelyje vienoje eilutėje pakeičiama raidė, trumpai paspaudžiamas 6 mygtukas ir užrašomas klaidos pranešimas. Jei klaidų langas neatsidaro, reiškia toks pakeitimas leistinas. Atstatoma programa ir pakeitimas daromas kitame stulpelyje. Užrašomos pastabos.

 - b) Vietoje PORTA programoje registrų priskyrimo dalyje įrašoma PRTA, trumpai paspaudžiamas 6-as mygtukas ir užrašomas klaidos pranešimas. Atstatoma programa.

 - c) Vietoje PORTA programos branduolio dalyje trečiame stulpelyje įrašoma PRTA, trumpai paspaudžiamas 6 mygtukas ir užrašomas klaidos pranešimas. Atstatoma programa.

d) Programoje ištrinama eilutė *PORTB equ 06* . Paleidžiama programa ir užrašomos išvados. Atstatoma programos eilutė.

e) Programoje ištrinama eilutė *GOTO \$* . Paleidžiama programa ir užrašomos išvados.

f) Vietoje šios ištrintos eilutės įrašomas programos tipinis sustabdymas

h) Išsiaiškinama kurią komandą vykdant išėjimo registre atsiranda rezultatas.

8. Modernizuojama programa. Į registrą *PORTA* įrašomas skaičius 11111111. Užrašoma koks rezultatas gausis įvykdžius programą.

ATASKAITA

- Aprašomas eksperimentinis darbas.
- Užrašomos išvados apie gautus rezultatus.
- Ties kiekviena tiriamos programos eilute įrašomas komentaras (ką ta eilutė padaro).
- Paaiškinama kur ir koks rezultatas bus kai įvykdomas žemiau parašytas programos fragmentas. Kiekvienos eilutės rezultatas nerašomas.

```
start    MOVLW B'10111011
          MOVWF    PORTA
          BCF     PORTA,7
          BSF     PORTA,6
```

Rezultatas gali būti registruose *PORTA*.....ir *W*....., gali keistis bitas *C*.... ar *Z*.....

ATAKYMAI Į KLAUSIMUS

- Kiek bitų turi mikrovaldiklio registrai *PORTA* ir *PORTB*?
- Kokios dalys sudaro programą?
- Kurioje programos dalyje užprogramuojami mikrovaldiklio išvadai (įėjimas/išėjimas).
- Ką reiškia trečiame programos stulpelyje po kablelio parašytas 1?
- Ką reiškia, jei po komandos įvykdymo bitas *Z* yra 1, arba bitas *C* yra 1?

1.2 Praktinis darbas Nr.2. Programa su ciklu

DARBO TIKSLAS. Tobulinti darbo su programa MPLAB įgūdžius. Išstudijuoti komandas GOTO, NOP, DECFSZ, INCF, DESF, MOVF, assemblerio direktyvą EQU ir programos ciklą sudarymo principus.

TEORINĖ DALIS

Ciklas – tai grupė komandų, kurios kartojasi. Ciklas pradedamas žyme, pvz., *ciklas*. Ciklas baigiasi besąlyginio perėjimo komanda GOTO, Komanda GOTO skirta programos vykdymą perkelti į pažymėtą vietą. Pavyzdyje ji grąžina programą į eilutę, pažymėtą žyme ciklas. Šia eilute programoje yra sukuriamas uždaras ciklas, kurį mikrovaldiklis nuolatos kartoja. Ciklas apima programos fragmentą tarp ciklo pradžios žymės (šiuo atveju žymė yra žodis ciklas) ir komandos GOTO ciklas, liepiančios grįžti į pažymėtą vietą. Kaip žymę galima naudoti trumpą informatyvų žodį, kuris nesutampa su mikrovaldiklio komanda arba assemblerio direktyva. Pvz., žyme negali būti žodis Data. Pavyzdys.

```

ciklas BSF          PORTB,2
        BCF          PORTB,2
        GOTO ciklas

```

Vykdamas programą prievado PORTB antras bitas įgaus vieneto reikšmę, po to nulio reikšmę. Po to bus grįžtama prie žymės ciklas ir viskas kartosis. Jei prie mikrovaldiklio linijos RB2 bus prijungtas spinduolis, tai jis mirksės. Dažnį apsprendžia kvarcinis rezonatorius.

Ciklas dažnai naudojamas kai reikia suformuoti laiko intervalą. Žemiau pavyzdyje parodytas ciklas formuoja 10 ms laiko intervalą. Jame panaudotas papildomas registras time1.

```

;10 ms vėlinimo ciklas.
time   MOVLW        d'13'   ;įrašyti į W registrą dešimtainį ;skaičių, pateiktą tarp kabučių
        MOVWF       time1   ;perkelti W registro turinį į time1 registrą
        CLRW        ;ištrinamas darbinis registras W
Ciklas DECFSZ       W,1     ;atimti vienetą iš registro W turinio ir, kai jis bus lygus nuliui,
        GOTO        Ciklas ;peršokti komandą goto Ciklas
        DECFSZ     time1,1  ;pereiti į paprogramės eilutę, pažymėtą žyme Ciklas
        peršokti          ;atimti vienetą iš registro time1 turinio ir, kai jis bus lygus nuliui,
                               ; komandą goto Ciklas
        GOTO        Ciklas ;pereiti į paprogramės eilutę, pažymėtą žyme Ciklas
Stop   NOP          ; stabdymo ciklas
        GOTO Stop
        END

```

Čia yra viršutinis ciklas, kuris kartojamas 256 kartus (atėmus iš 0 vienetą gaunamas skaičius 255). Vieno ciklo trukmė lygi 3 mašiniams ciklams, nes komanda DECFSZ yra įvykdoma per 1 mašininį ciklą (išskyrus atvejį, kai yra vykdomas peršokimas), o komanda GOTO – per 2 mašininis ciklus.

Apatinio ciklo kartojimų skaičius yra nustatomas įkeliant atitinkamą skaičių į registrą time1. Pateiktame pavyzdyje apatinio ciklo kartojimų skaičius yra 13, todėl bendras įvykdymo laikas sudaro $(3 \mu s \times 256 \times 13) + (3 \mu s \times 13) = 10\,023 \mu s$. Prie šio laiko dar reikia pridėti komandų MOVLW ir MOVWF įvykdymo laiką ($1 \mu s + 1 \mu s$), komandos DECFSZ įvykdymo laiko pailgėjimą, kai vykdomas peršokimas ($1 \mu s \times 14$ peršokimų). Todėl bendras programos vykdymo laikas sudaro $10\,039 \mu s \approx 10$ ms. Šią programėlę yra patogiu naudoti sudarant ilgus, nustatytos trukmės vėlinimo ciklus. Pavyzdžiui, 1 s trukmės ciklas gali būti sudarytas įtraukiant šią programėlę į ciklą, kuriame ji yra kartojama 100 kartų.

KOMANDŲ PAAIŠKINIMAS. Sakykim pradinėje būsenoje registre PORTB ir time1 yra skaičius d'10'.

DECFSZ PORTB,1 – vykdamt komandą registre PORTB įrašytas skaičius mažinamas vienetu ir įrašomas į tą patį registrą. Gaunamas skaičius 9. Jei atėmus 1 registre lieka nulinis rezultatas yra praleidžiama viena komanda.

INCFSZ time1,1 – vykdamt komandą registre time1 įrašytas skaičius didinamas vienetu ir įrašomas į tą patį registrą. Gaunamas skaičius 11. Jei pridėjus 1 registre yra nulinis rezultatas yra praleidžiama viena komanda. Tai atsitinka kai prie registre esančio skaičiaus 255 pridedamas 1.

INCF PORTB,1 – registro PORTB turinys padidinamas 1 ir įrašomas į tą patį registrą. PORTB gaunamas skaičius 11.

INCF time1,0 – registro time1 turinys padidinamas 1 ir įrašomas į darbinį registrą W. Jame gaunamas skaičius 11. Registre time1 lieka tas pat skaičius 10

DECFSZ time1,1 – registro time1 turinys sumažinamas 1 ir įrašomas į registrą time1. Jame gaunamas skaičius 9.

DECFSZ PORTB,0 – registro PORTB turinys mažinamas 1 ir įrašomas į darbinį W registrą. Jame gaunamas skaičius 9, o registre PORTB lieka skaičius 10.

MOVF PORTB, 0 – komanda persiunčia registro PORTB turinį į darbinį registrą W. Jei po kablelio yra 1, tai turinys persiunčiamas į tą patį registrą. Tada informacija registre PORTB išlieka ta pati.

MOVWF PORTA – persiunčiamas W registro turinys į registrą PORTA. Registre W lieka tas pat skaičius.

EKSPERIMENTINĖ DALIS

1. Nukopijuojama į kompiuterį žemiau parodyta programa Mikro1 ir įrašomas asm failas į direktoriją darbai savo vardu. Ši programa pakaitom uždegs ir gesins po du spinduolius. Vienu metu šviečia du kairieji spinduoliai. Po to jie gęsta ir šviečia du dešinieji spinduoliai

;Mikro1- programa. Prie RB3,RB4,RB5,RB6 prijungti šviesos diodai. Per 470 omu rezistorius jie sujungti su GND. Du ;sviecia,du -ne. Po to keiciasi.

LIST P=16F84

```

PORTA=5
PORTB=6
TRISA=85H
TRISB=86H
STATUS=3
RP0=5

        CLRF  PORTA
        CLRF  PORTB
init    BSF   STATUS,RP0
        CLRF  TRISA
        CLRF  TRISB
        BCF   STATUS,RP0
main    MOVLW B'00011000'
        MOVWF PORTB
        MOVLW B'01100000'
        MOVWF PORTB
stop    NOP
        GOTO  stop
        END

```

Įkeliama programa Mikro1 į MPLAB aplinką ir simuliuojamas po žingsnį jos darbas. Užrašomos išvados apie rezultatą.

1. Išsiaiškinama ir parašomos išvados kaip pasikeis programos darbas jei:
 - a) pašalinama (priešais parašomas kabliataškis) inicializacijoje eilutė CLRF TRISA arba CLRF TRISB;

b) priskiriant registrams adresus vietoje TRISB=86H parašoma TRISB=66H (arba kitas bet koks skaičius).

2. Modifikuojama programa taip, kad joje būtų ciklas, kuris priverstų visą laiką mirksėti lemputes. Simuliuojamas programos darbas automatinio režimu ir parašomos išvados apie tai kas matoma ekrane, koku dažniu mirksi lemputės. Užrašomos modifikuotos programos eilutės.

3. Parašoma programa, kuri į registrą DATA įrašytų skaičių D'150'. Programa įkeliama į MPLAB aplinką. Parašoma išvadas apie kilusias problemas ir jų sprendimą. Simuliuojamas programos darbas.

4. Modifikuojama ši programa taip, kad tas pat skaičius būtų įrašytas ir į registrą PORTB. Programa išbandoma. Parašomos išvados apie pasikeitimus SFR registru lange. Užrašomos modifikuotos programos eilutės.

5. Modifikuojama programa sudarant ciklą. Registre PORTB skaičius didinamas ir kai pasiekia 256 programa sustoja. Stebimas C ir Z bito pasikeitimas. Užrašomos išvados ir modifikuotos programos eilutės.

ATASKAITA

- Eksperimentinio darbo aprašymas ir išvados.
- Tirtos programos (tik branduolys).
- Parašoma programos inicializacijos dalis, kai prie mikrovaldiklio linijų RB0, RB2, RB4, RA2 RA5 prijungti mygtukai.
- Nubraižoma elektros principinė schema, kurioje prie mikrovaldiklio prijungta 24V elektros lemputė.
- Paaiškinama kur ir koks bus rezultatas, kai bus įvykdytas programos fragmentas

```
Main    MOVLW    B'00110010
        MOVWF   PORTB
        INCF   PORTB,0
        MOVWF  DATA
```

Rezultatas gali būti: registruose PORTB....., DATAir W.....,
gali keistis bitai C.....ar Z.....

ATSAKYTI Į KLAUSIMUS

- Koks laiko intervalas tarp lempučių persijungimo ir ką matytumėm prijungus lemputes?
- Kodėl programa neleidžia naudoti žodžio DATA?
- Ką rodo C ir Z bitų reikšmė 1?
- Ką matysime klaidų lange jei praleisime programos eilutę STATUS = 3?
- Ką reiškia programos eilutėje po kablelio įrašytas skaičius 5 arba 0?

1.3 Praktinis darbas Nr.3. Programa su mygtuku

DARBO TIKSLAS: Tobulinti darbo su programa MPLAB įgūdžius. Išsiaiškinti kaip į mikrovaldiklį įvedama informacija iš jutiklių ar mygtukų. Išstudijuoti komandas BTFSC, BTFSS .

TEORINĖ DALIS

Mygtukais operatorius įveda informaciją į mikrovaldiklį. Į programą mygtukas įtraukiamas inicializacijoje įrašant į registro TRISA arba TRISB atitinkamą bitą vienetai.

Pvz., įrašius į registrą TRISB skaičių B'00001110', trys PORTB linijos – RB1, RB2, RB3 bus įėjimai, o kitos – išėjimai. Galima prijungti tris mygtukus arba jutiklius.

Programos branduolyje mygtukas apklausiamas. Tam naudojamos komandos BTFSC PORTB,d arba BTFSS PORTB,d. Čia d yra skaitmuo nuo 0 iki 7. Jis nurodo registro bito, kuris užprogramuojamas įėjimu, numerį.

Pvz., jei registre TRISB 4-am bitui suteikiama 1 reikšmė, tai registro PORTB linija RB4 tampa įėjimu. Kai mygtukas nenuspaustas linijoje yra + 5 V, t.y. loginis 1. Kai mygtukas nuspaudžiamas linijoje yra 0 V, t.y. loginis 0. Apklausa vykdoma taip:

```
start   BTFSC PORTB,4      ;ar 4 bitas yra 0. Jei ne- vykdoma kita komanda - GOTO.
                                     ; Čia start - žymė
        GOTO start         ; pereinama prie žymės start
        xxxxxxxxxxx        ;kita komanda
```

Jei 4-as bitas yra 0, viena komanda praleidžiama ir vykdoma komanda kitoje eilutėje (xxxxxxxx).

Kitas apklausos variantas (naudojamas rečiau).

```
start   BTFSS, 4           ;ar 4 bitas yra 1. Jei ne- vykdoma kita komanda - GOTO.
        GOTO start         ; pereinama prie žymės start
        xxxxxxxx          ;kita komanda
```

Jei 4-as bitas yra 1, viena komanda praleidžiama ir vykdoma komanda kitoje eilutėje (xxxxxxx).

Komandą BTFSC patogiu naudoti kai reikia paeiliui apklausti kelis mygtukus. Programos pavyzdys Pvz, 1. Yra 3 mygtukai, prijungti prie RB0, RB1, RB2 (M1, M2, M3) 3 lemputės prijungtos prie RA0, RA1, RA2 (L1, L2, L3). Nuspaudus pirmą mygtuką užsidega pirmą lemputė, nuspaudus antrą mygtuką – antra ir t.t.. Būtina spausti mygtukus paeiliui.

```
Pvz. 1
Myg1   BTFSC PORTB,0
        GOTO Myg1
        BSF      PORTA,0
Myg2   BTFSC PORTB,1
        GOTO    Myg2
        BSF      PORTA,1
Myg3   BTFSC PORTB,2
        GOTO    Myg3
        BSF      PORTA,2
stop xxx
```

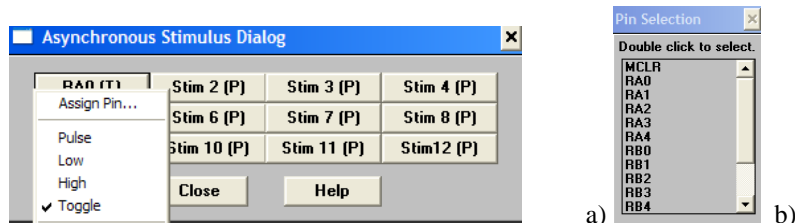
Jei mygtukai spaudžiami bet kokia tvarka patogiau naudoti komandą BTFSS. Programos fragmentas parodytas Pvz. 2. Yra 3 mygtukai, prijungti prie RB0, RB1, RB2 (M1, M2, M3) 3 lemputės prijungtos prie RA0, RA1, RA2 (L1, L2, L3). Nuspaudus atitinkamą mygtuką užsidega atitinkama lemputė.

```
Pvz. 2
main   BTFSS PORTB,0
        BSF      PORTA,0
        BTFSS   PORTB,1
        BSF      PORTA,1
        BTFSS   PORTB,2
```

BSF PORTA,2
GOTO main

PROGRAMOS SU MYGTUKU DARBO SIMULIAVIMAS KOMPIUTERYJE

Programos lange pasirenkama Debug – Simulator Stimulus – Asynchronous Stimulus. Atsidaro langas su 12 mygtukų. Spaudžiamas dešinysis pelės klavišas ir atsidaro langas Assign Pin .



1.4 pav. Mygtukų langas

Jame varnele pažymima Toggle. Tai reiškia, kad vieną kartą paspaudus mygtuką jis susijungia, o antrą kartą spustelėjus – išsijungia, t.y. mygtukas nuspaudžiamas dvigubu pelės kairiojo mygtuko spūstelėjimu. Langelis užsidaro.

Vėl atidaromas langas Assign Pin. Spaudžiama Assign Pin ir atsidaro langas 1.4 pav. b. Jame išvardyti mikrovaldiklio išvada. Reikia dvigubai spūstelėti pele išvadą, prie kurio yra prijungtas mygtukas. Langelyje 1.4 pav. a pasirodo išvado pavadinimas. Tai pakartojama su kiekvienu mygtuku.

Prieš simuliuojant programą reikia stebėjimo lange patikrinti ar mygtukai nesujungti. Jei atitinkamo registro (PORTA arba PORTB) bitas, prie kurio prijungtas mygtukas turi 1 reikšmę – viskas tvarkoje. Jei jo reikšmė 0, reiškia mygtukas nuspaudžiamas ir jį reikia atleisti. Tai atliekama vieną kartą spūstelėjant pele mygtuką ir menių juostoje „pėdas“ (3 mygtukas 1.2 pav.). Atitinkamas bitas įgauna reikšmę 1.

Paleidus programą simuliaciniu režimu juoda linija programoje šokinėja ties mygtuko apklausos vieta. Dvigubas spūstelėjimas pele ant atitinkamo mygtuko lange 1.4 pav., a ir programa vykdoma toliau.

PASIRUOŠIMAS NAMUOSE

- Parašoma mirksinčios lemputės programa. Lemputė, prijungta prie RA2 linijos užsidega ir gęsta.
- Parašoma trijų lempučių valdymo programa pagal pvz.,1.

EKSPERIMENTINĖ DALIS

1. Į MPLAB aplinką įkeliama mirksinčios lemputės programa ir išbandoma simuliaciniame režime. Užrašomos pastabos.

2. Programa modifikuojama. Vienu metu mirksi 3 lemputės, prijungtos prie RA0, RA1, RA2. Užrašomos modifikuotos programos eilutės. Programa išbandoma ir užrašomos išvados.

3. Programa modifikuojama. Lemputės pradeda mirksėti po to, kai trumpam (1 sek) papaudžiamas mygtukas, prijungtas prie mikrovaldiklio linijos RB3. Programa išbandoma ir parašomos išvados. Užrašomos modifikuotos programos eilutės.

4. Programa modifikuojama. Į programą įvedamas antras STOP mygtukas, kuris sustabdytų lempučių mirksėjimą. Mygtukas prijungtas prie RB4. Programa išbandoma ir parašomos išvados apie trūkumus (mygtuko paspaudimo ypatumus). Užrašomos modifikuotos programos eilutės.

5. Į MPLAB aplinką įkeliama namuose parašyta programa pagal pvz.,1. Programa išbandoma ir parašomos išvados apie trūkumus

6. Modernizuojama programa taip, kad spustelėjus kitą mygtuką šviečianti lemputė užgęstų, o užsidegtų kita. Programa išbandoma. Užrašomos modifikuotos programos eilutės.

7. Modernizuojama programa taip, kad ji veiktų be sustojimo. Užrašomos modifikuotos programos eilutės.

ATASKAITA

- Eksperimentinio darbo aprašymas ir išvados.
- Išbandytos programos (tik branduolys). Viena iš jų su komentarais.
- Programų veikimo aprašymas (algoritmas).
- Nubraižoma elektros principinė schema kaip mygtukas prijungiamas prie mikrovaldiklio RA1 linijos. Paaiškinama schemas elementų paskirtis.

ATSAKYTI Į KLAUSIMUS

- Kurie išvadai reikalingi, kad matytumėm mikrovaldiklio veikimą?

- Kuo skiriasi linijos RB1 ir RB5?
- Kiek laiko reikia laikyti nuspaustą STOP mygtuką mirksinčios lemputės programoje?

1.4 Praktinis darbas Nr 4. Paprogramės

DARBO TIKSLAS: Išstudijuoti komandas CALL, RETURN, DECFSZ ir paprogramių sudarymo principus. Sudaryti laiko intervalo formavimo paprogramę.

TEORINĖ DALIS

Paprogramė – tai grupė komandų, kuri naudojama keliose programos vietose. Paprogramei yra suteikiamas pavadinimas (mūsų atveju laikas). Pagrindinėje programoje paprogramė yra iškviečiama naudojant komandą CALL ir paprogramės pavadinimą, pvz., CALL laikas. Programa, pasiekusi komandą CALL, pradeda vykdyti atitinkamos paprogramės komandas. Komanda RETURN, kuri yra įrašoma paprogramės pabaigoje, grąžina į pagrindinę programą. Grįžus į pagrindinę programą yra vykdoma komanda, einanti po komandos CALL.

Paprogrames patogu naudoti, kai programoje yra pasikartojantis fragmentas. Tuomet tokį programos fragmentą galima apiforminti kaip paprogramę ir iškviešti atitinkamose programos vietose. Tai leidžia sutrumpinti programą ir padaryti ją lengviau suprantamą bei sutaupyti vietos mikrovaldiklio programų atmintyje. Tačiau paprogramių naudojimas pailgina programos įvykdymo laiką, nes reikia vykdyti papildomas komandas CALL ir RETURN. Paprogramė paprastai rašoma programos teksto pabaigoje prieš assemblerio direktyvą END.

KOMANDŲ PAAIŠKINIMAS

CALL laikas – paprogramės, kurios pavadinimas laikas iškvietimas. Programa peršoka prie žymės laikas.

RETURN – paprogramės pabaiga. Programa grįžta į tą vietą, iš kurios buvo iškviesta.

DECFSZ COUNT,1 – vykdamas komandą registre COUNT įrašytas skaičius mažinamas vienetu ir įrašomas į tą patį registrą. Jei atėmus 1 registre lieka nulinis rezultatas yra praleidžiama viena komanda.

BTFSS PORTB,2 – tikrinamas registro PORTB antras bitas. Jei jo reikšmė 0 – vykdoma kita komanda, jei reikšmė 1 – praleidžiama viena komanda.

ADDLW k – registro W turinys sudedamas su k (8-ių bitų konstanta). Rezultatas išsaugomas W registre.

Pvz. 3

Suma	CLRWF		; išvalomas registras W
	ADDLW	D'12'	
	MOVWF	PORTB	

Įvykdžius šias komandas registruose W ir PORTB bus skaičius 12.

LAIKO INTERVALO FORMAVIMAS

Laiko intervalai naudojami pvz., rašant impulsų generatoriaus programas. Jie nustato impulsų trukmę bei dažnį. Čia naudojamų laiko intervalų trukmė skaičiuojama ms. Įvairiems mirksiukams naudojami kelių sekundžių laiko intervalai. Taip pat laiko intervalai reikalingi laiko relėse. Čia jie matuojami minutėmis. 20 ms laiko intervalo paprogramė DELAY parodyta [1] 73psl. Ji naudojama, kai reikia programiškai panaikinti kontakto kibirkščiavimo efektą. Laiko intervalo paprogramė, kuri formuoja kelių sekundžių laiko intervalą parodyta žemiau. Ji paaiškinta [1] 56 psl. Kaip skaičiuojamas laikas aprašyta praktiniame darbe Nr.2.

Pvz.4.

laikas	MOVLW	D'100" ;	
	MOVWF	kint2	; čia registras, kurio pavadinimas kint2.
			;Nuo jame įrašyto skaičiaus priklauso laikas.
laikas2	MOVLW	D'200" ;	
	MOVWF	kint1	; čia registras, kurio pavadinimas kint1

```

CLRW
laikas1 ADDLW      1
        BTFSS     STATUS,z
        GOTO     laikas1
        DECFSZ   kint1, 1
        GOTO     laikas1
        DECFSZ   kint2, 1
        GOTO     laikas2
        RETURN

```

Programoje panaudoti du bendrieji registrai kint1 ir kint2, į kuriuos įrašomi kintamieji – skaičiai, kurie apsprendžia laiko intervalo trukmę. Programos pradžioje registrams reikia priskirti adresus.

PROGRAMOS SU LAIKO INTERVALU DARBO SIMULIAVIMAS

Simuliavimo metu programa dirba sulėtintai (100 ÷ 200 kartų). Todėl net ir nedidelės trukmės laiko intervalo paprogramės simuliacija gali užimti keliasdešimt minučių. Todėl prieš simuliuojant programos darbą reikia laikinai ją pakeisti. Galimi du būdai. Vienas iš jų yra prieš kiekvieną CALL komandą padėti kabliataškį. Jei yra daug kreipimusi į paprogramę bus sugaišta daug laiko.

Kitas būdas – laikinai pakeičiama paprogramė. Iš karto po paprogramės pavadinimo įterpiama komanda RETURN (pvz.,5).

```

Pvz.5.
laikas RETURN
        MOVLW     D'100" ;
        MOVWF     kint2
        ;Ir taip toliau

```

Programa šoka į paprogramę ir tuoj pat iš jos sugrįžta. Įrašant programą į mikrovaldiklį komanda RETURN arba kabliataškiai pašalinami.

Antrą būdą rekomenduojama naudoti tuo atveju, kai programa stringa. Dažnai, rašant laiko paprogramę, padaroma klaida. Todėl ieškant programos klaidos pirmiausia aplenkama paprogramė. Rekomenduojama pasirašyti veikiančią laiko paprogramę ir kopijavimo būdu įkelti ją į naują programą.

PASIRUOŠIMAS NAMUOSE

Parašoma trijų lempučių valdymo programa pagal pvz., 2 iš praktinio darbo Nr.3.

EKSPERIMENTINĖ DALIS

1. Įkeliama namuose parašyta programa į MPLAB aplinką ir išbandoma. Užrašomos pastabos apie programos trūkumus.

2. Modernizuojama programa taip, kad kai užsidega viena lemputė, kitos užgesų. Tam reikia prieš uždegant vieną lemputę kitas užgesinti. Tai atliekama sukuriant tris paprogrames, pvz., pirma, antra, trečia. Programa išbandoma ir parašomos išvados.

3. Įkeliama į MPLAB aplinką mirksinčios lemputės programa iš praktinio darbo Nr.3.

4. Modernizuojama programa įvedant laiko paprogramę. Kai uždegama lemputė iššaukiama paprogramė. Kai užgesinama lemputė vėl iššaukiama paprogramė. Programa išbandoma simuliaciniame režime. Užrašomos išvados apie kilusias problemas ir jų sprendimą.

6. Programa laikinai pakeičiama ir išbandoma. Užrašomos išvados.

ATASKAITA

- Eksperimentinio darbo aprašymas ir išvados.
- Išbandytos programos (tik branduolys).
- Nubraižoma elektros principinė schema kaip jutiklis, kurio išėjimo signalas +24 V prijungiamas prie mikrovaldiklio.

ATSAKYTI Į KLAUSIMUS

- Ką reiškia skaičiai po kablelio komandose: `BSF PORTB,3`, `INCF REZULT, 1`, `MOVF PORTB,0`, `BTFSC PORTA,3`.
- Ką reikia pakeisti programoje norint naudoti registrus `kint1` ir `kint2`?
- Ką rodo konfigūracijos eilutėje kodas `H'3FFC'`?

1.5 Praktinis darbas Nr.5. Mikrovaldiklio užprogramavimas ir programos išbandymas

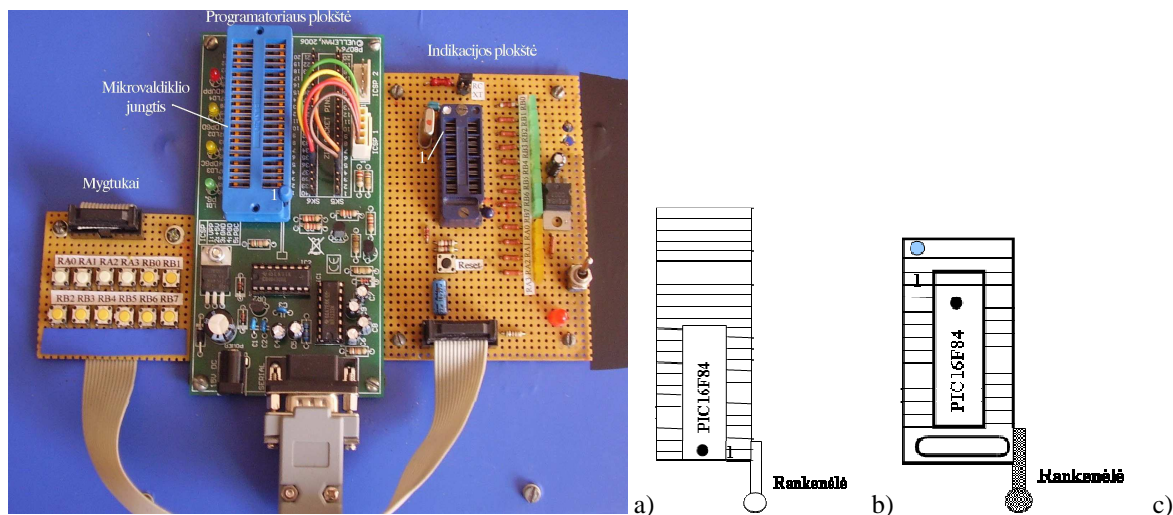
DARBO TIKSLAS: užprogramuoti mikrovaldiklį ir išbandyti programą realiaime įtaise.

TEORINĖ DALIS

Prieš programuojant mikrovaldiklį reikia turėti programos hex failą, pvz., mirksiukas.hex. Šį failą sukuria MPLAB programa toje pat vietoje ir tuo pačiu pavadinimu kaip asm failas.

Mikrovaldiklį užprogramuoja specialūs įtaisai – programatoriai. Į juos įdedamas mikrovaldiklis ir paleidžiama speciali programa į kurios aplinką įkeliami hex failai. Prieš programuojant parenkamas mikrovaldiklio tipas ir nustatoma konfigūracija jei rašant programą nebuvo konfigūracijos eilutės [1] 54 p. Kaip dirbti su programatoriumi aprašyta [1] 91p, o nesudėtingų programatorių elektros principinės schemas parodytos [1] 87 p.

PROGRAMATORIUS



1.5 pav. Programatoriaus stendas.

Programatorių 1.5 pav., a sudaro keturios dalys: programatoriaus spausdintinė plokštė, indikacijos plokštė, valdymo mygtukai ir maitinimo šaltinis. Prie kompiuterio programatorius prijungiamas per RS232 prievadą. Programatoriuje naudojama programa PicProg2006. 1.5 pav., b parodytas mikrovaldiklio įdėjimas į lizdą programatoriuje, o 1.5 pav., c – indikacijos plokštėje.

MIKROVALDIKLIO UŽPROGRAMAVIMAS

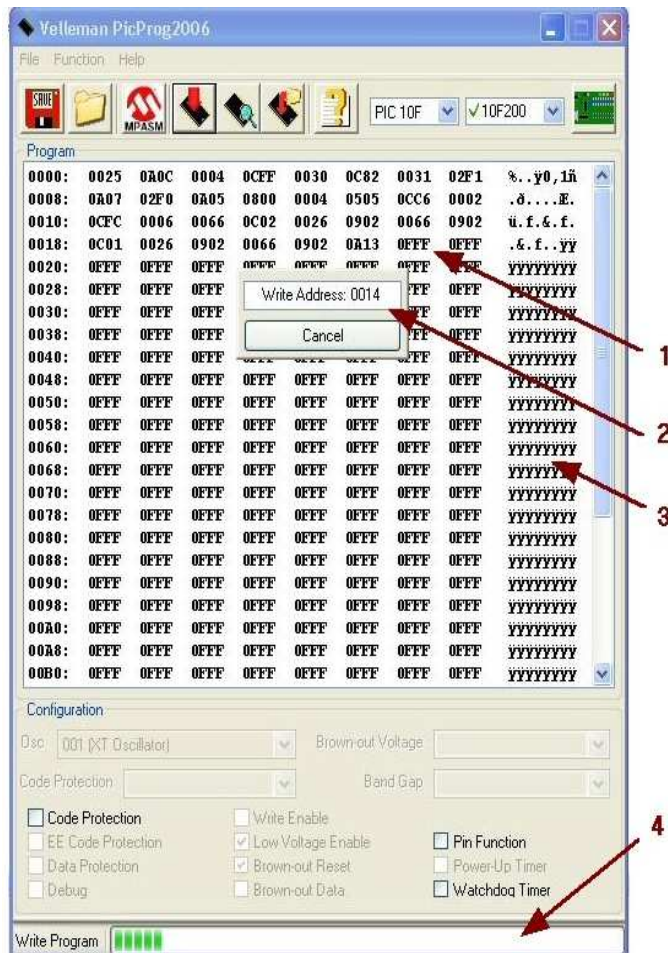
Mikrovaldiklis užprogramuojamas tada, kai yra kompiuteryje išbandyta programa ir sukurtas jos kodas - hex failas, pvz. mirksiukas.hex. Į programatoriuje esantį mikrovaldiklio lizdą įdedamas mikrovaldiklis.



1.7 pav. Mygtukų juosta

1. Išsaugoti HEX failą. 2. Įkrauti HEX failą. 3. Įkrauti MPASM įrankį. 4. Nuspaudus šį mygtuką vykdoma komanda Write all (žiūrėti aprašymą Write PIC) 5. Nuspaudus šį mygtuką vykdoma komanda Read all (žiūrėti aprašymą Read PIC). 6. Įrašyti duomenis iš HEX failo į piką tiesiogiai. 7. Iškviešti Help failą. 8. Pic pasirinkimas. 9. Įrangos konfigūracija

Iš mygtukų juostos išsirenkamas reikiamas mikrovaldiklis – PIC16F84. Tada spaudžiamas mygtukas 2 (įkrauti HEX failą) ir nurodoma failo vieta diske.



1.8 pav. Langas su programa

1. HEX kodas, kuris bus išsaugotas. 2. Adresų skaitiklis. Parodo kur atmintyje įrenginys skaito ir rašo. 3. ASCII kodo versija. 4. Progreso juosta. Vizualiai parodo programavimo arba skaitymo procesą

Tada spaudžiamas mygtukas 4 (Write all) pasirodys pranešimas. Spaudžiama Yes.

PASIRUOŠIMAS NAMUOSE

Parašoma programą pagal pav.,2 iš 4-o praktinio darbo. Trys mygtukai ir trys lemputės. Kuris mygtukas trumpai nuspaudžiamas ta lemputė mirksi kas 1 sek.

EKSPERIMENTINĖ DALIS

1. Įdedamas į programatoriaus lizdą mikrovaldiklis .
2. Įkeliamas iš praktinio darbo Nr.4 mirksinčios lemputės programos su laiko paprograme hex failas į PicProg2006 programos aplinką.
3. Parenkamas RC generatorius ir išjungiamas Watchdog Timer. Užprogramuojamas mikrovaldiklis.

4. Perkeliamas mikrovaldiklis iš programatoriaus lizdo į demonstracinės plokštės lizdą.
5. Išbandomas programos veikimas.
6. Pakartojamas 3 ir 4 punktai neišjungiant Watchdog Timer. Išbandomas programos veikimas ir užrašomos pastabos apie pasikeitimus.
7. Įkeliami namuose parašyta programa į mikrovaldiklį ir išbandoma stende. Užrašomos pastabos.

ATASKAITA

- Eksperimentinio darbo aprašymas ir išvados.
- Namuose parašyta programa.
- Paaiškinama, kokie mikrovaldiklio išvadai naudojami programavimo metu ir kokie juose signalai (įtampos).

2. ARDUINO MODULIS

2.1. Arduino UNO modulio sandara

Arduino modulis buvo sukurtas 2005 metais, kai Massimo Banzi ir David Cuartielles sukūrė paprastą ir nebrangią platformą studentams, kurios pagrindu buvo galima kurti įvairius automatikos prietaisus.

Arduino – tai elektronikos konstruktorius. Prie modulio plokštės galima jungti begalę periferinių įrenginių – jutiklius (temperatūros, apšvietimo, slėgio, pagreičio ir t.t.), šviesos diodus, skystų kristalų (LCD) ekranėlius, servo variklius. Papildomi elementai pateikiami kaip priedėliai (Shields) ir kaip patobulinimai (Tinkerkit), kurių pagalba išplečiamas standartinis Arduino funkcionalumas.

Arduino Uno valdiklio 2.1 pav. pagrindas yra Atmega328 mikrovaldiklis. Arduino plokštė turi 14 skaitmeninių įėjimų/išėjimų iš kurių 6 gali būti naudojami kaip impulsų trukmės modulatoriai (PWM). Taip pat yra 6 analoginiai įėjimai, kvarcinis 16 Mhz generatorius, USB jungtis, maitinimo jungtis, speciali jungtis programavimui ir Reset mygtukas.

Darbo metu valdiklis prijungiamas prie kompiuterio USB prievado arba prie 12 V maitinimo šaltinio.



2.1 pav. Valdiklio Arduino Uno modulis

Arduino Uno plokštėje yra įvairūs mazgai, kurie užtikrina ryšį su kompiuteriu ir išoriniais prietaisais. Atmega328 turi nuoseklaus duomenų perdavimo sąsają UART TTL (5 V). Naudojami išvadai 0 (RX) ir 1 (TX). Plokštėje esantis lustas ATmega8U2 užtikrina šios sąsajos ryšį su USB. Išorinės kompiuterio programos susiriša su plokšte per virtualų COM portą. Atmega8U2 programa naudoja standartinę tvarkyklę USB COM. Naudojant Windows operacinę sistemą papildomai reikia failo ArduinoUNO.inf. Šviesos diodai RX ir TX plokštėje mirksės perduodant duomenis per lustą FTDI arba USB sąsają.

Pasinaudojant biblioteka SoftwareSerial <http://arduino.ru/Reference/Serial> galima nuosekliai perduoti duomenis per bet kurį valdiklio skaitmeninį išvadą.

Kiekvienas iš 14 skaitmeninių išvadų gali būti užprogramuotas kaip įėjimas ar išėjimas pasinaudojant funkcija `pinMode()`, `digitalWrite()`, ir `digitalRead()`. Išvaduose gaunama +5 V įtampa. Kiekvienas išvadas turi vidinį apkrovimo rezistorių (20-50 kΩ). Jis pagal nutylėjimą atjungtas ir prijungiamas programiškai. Išvado maksimali srovė iki 40 mA.

Nuoseklus duomenų perdavimo išvadai 0 (RX) ir 1 (TX). Jie naudojami norint gauti TTL lygio priėmimo (RX) ir perdavimo (TX) signalus. Šie išvadai prijungti prie papildomos nuoseklus duomenų perdavimo sąsajos Atmega 8U2 (USB į TTL).

Išvadai 2(interrupt0) ir 3(interrupt1) skirti išorinei pertraukčiai. Jų konfigūravimą aprašo funkcija `attach Interrupt()`.

PWM išvadai 3, 5, 6, 9, 10, ir 11. Juose galima impulsų trukmės moduliacija su 8 bitų rezoliucija panaudojant funkciją `analogWrite ()`.










Išvadai SPI (Serial Peripheral Interface): 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK) naudojami nuosekliam duomenų perdavimui į išorinius įrenginius pasinaudojant SPI biblioteka .

Prie 13 skaitmeninio išvado prijungtas LED diodas, kuris šviečia esant 13 išvade +5 V.

Arduino Uno plokštėje analoginiai išėjimai pažymėti A0 iki A5. Kiekvieno rezoliucija 10 bitų (gali turėti 1024 reikšmes). Tipinė jų įtampa +5 V. Tačiau įtampą galima padidinti programiškai naudojant išvadą AREF ir funkciją `analogReference()`. Analoginiai išvadai A4 ir A5 turi papildomas funkcijas. Per juos vykdomas ryšys I2C (TWI). Ryšiui naudojama biblioteka `Wire`. Mygtukas „Reset“, kai jame žemas lygis, perkrauna mikrovaldiklį.

2.1.1 Arduino programos įdiegimas

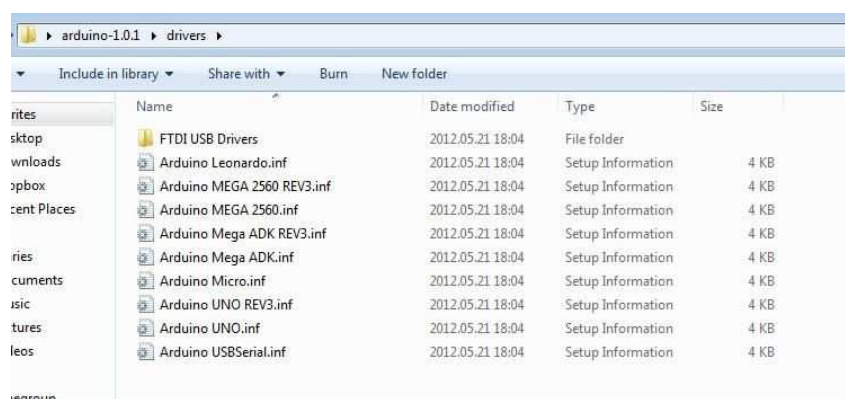
1. Atsisiunčiama naujausia arduino mikrovaldiklio programos versija (*arduino-1.0.5-windows*), šiame tinklalapyje: <http://arduino.cc/en/Main/Software> .
2. Atsisiųstas aplankas išpakuojamas su archyvavimo programa.
3. Du kartus pele paspaudžiama ant aplanko ir jis atsidaro. Aplankale yra programos failai.
4. Atidaromas arduino programos langas, paspaudus du kartus kairiu pelės klavišu ant `arduino.exe` (2.2 pav.).

 drivers	2012.05.21 18:04	File folder
 examples	2012.05.21 18:05	File folder
 hardware	2012.05.21 18:05	File folder
 java	2012.05.21 18:06	File folder
 lib	2012.05.21 18:05	File folder
 libraries	2012.05.21 18:05	File folder
 reference	2012.05.21 18:05	File folder
 tools	2012.05.21 18:05	File folder
 arduino.exe	2012.05.21 18:05	Application

2.2 pav. Programos paleidimas

Instaliuojama Arduino Uno modulio tvarkyklė

1. Prijungiama Arduino plokštė prie kompiuterio USB prievado. Palaukiama kol automatinis Windows tvarkyklių instaliavimas prasidės. Deja, nepavyks įdiegti plokštės tvarkyklių šiuo būdu.
2. Spaudžiama “Start Menu”, einama į “Control Panel” nustatymus. Tuomet atsidaromas “System and Security” langas. Atidaromas “System”, atsidariusiame lange pasirenkama “Device Manager”
3. Atsidariusiame “Device Manager” programos lange susirandama “Ports (COM & LPT)”. Paspaudus turi pasirodyti “Arduino UNO (COMxx)”. Pasirinkus šią nuorodą spaudžiama dešiniu pelės klavišu ir spaudžiama “Update Driver Software”.
4. Spaudžiama “Next” ir pasirenkama "Browse my computer for Driver software".
5. Pasirenkamas tvarkyklės failas "ArduinoUNO.inf" 2.3 pav. ir instaliuojamas.



2.3 pav. Arduino modulių pasirinkimas

2.1.2. Programavimo aplinka

Arduino programavimo aplinką 2.4 pav. sudaro kodo teksto redaktorius, pranešimų sritis (klaidų laukas), teksto išvedimo langas, dažnai naudojamų komandų įrankių juosta su mygtukais.



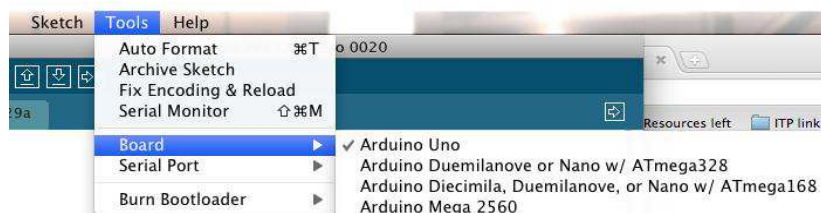
2.4 pav. Arduino pagrindinio lango meniu

Programos mygtukų paskirtis aprašyta žemiau.

1. Programos kompiliavimas/tikrinimas. Klaidų lauke atsiranda pranešimai jei yra klaidų.
2. Programos įkėlimas į Arduino modulį.
3. Naujo projekto pradžia.
4. Pasirinkto projekto atidarymas.
5. Parašytos programos išsaugojimas

Pirmą kartą atsidarius programos langą reikia pasirinkti Arduino modulio tipą. Pasirenkama: Tools, Board, Arduino Uno (2.5 pav.).


Programa, parašyta Arduino aplinkoje, vadinama Sketch. Ji rašoma teksto redaktoriais. Išsaugant ar eksportuojant programą pranešimų srityje atsiranda pranešimai. Ten taip pat rodomos klaidos. Įrankių juostos mygtukai leidžia patikrinti parašytą programą, sukurtą, atidaryti ir išsaugoti programą, atidaryti nuosekliosios magistralės stebėjimą (monitoring).



2.5 pav. Arduino modulio tipo pasirinkimas

2.1.3. Programos įkrovimas

Jis vykdomas po to kai Arduino programinė aplinka sujungiama su aparatine dalimi. Tam reikia USB kabelio su jungtimis USB-A ir USB-B. Prijungus Arduino modulį prie kompiuterio pradeda plokštėje šviesti žalias spinduolis PWR.

Įkeliant į modulį parašytą programą (sketch) pvz., „Bega“ 2.6 pav., naudojama Arduino modulyje įrašyta įkrovimo programa (Bootloader). Ji leidžia įkrauti programą nenaudojant papildomos įrangos – programatorių. Ši programėlė dirba keletą sekundžių kol perkraunama nauja programa. Įkrovimo programos darbą indikuoja modulyje esantis spinduolis (13 pin). Spaudžiamas antras mygtukas  ir automatiškai kodas įkraunamas į Arduino modulį.



2.6 pav. Programos fragmentas

2.2. Praktinis darbas Nr. 1 Mirksintis spinduolis.

DARBO TIKSLAS: rašyti programą C kalba, gebėti įkelti programą į Arduino modulį.

DARBO PRIEMONĖS. Kompiuteris su programine įranga arduino-1.0.2-windows, USB kabelis, Arduino stendas, 12 V maitinimo šaltinis, knyga (pdf.) Arduino visiems.

PASIRUOŠIMAS NAMUOSE

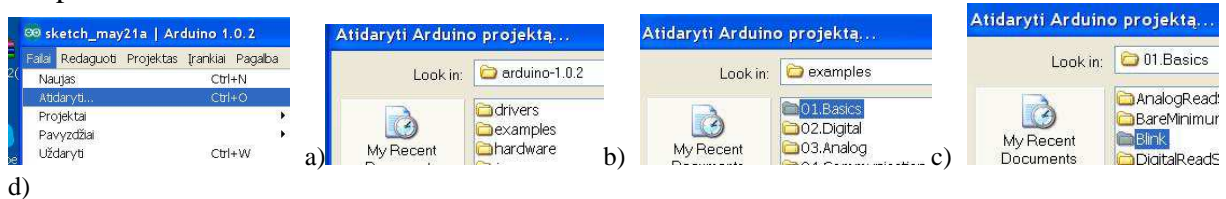
Išanalizuojamos ir aprašomos mirksinčio spinduolio programos komandos pasinaudojant internetine knyga [3].

```
// Led prijungiamas prie 13 išvado
int led = 13;
// ciklas void setup atliekamas vieną kartą
void setup() {
  //nustatome išvadą led (13) kaip išėjimą
  pinMode(led, OUTPUT);
}
// ciklas void loop atliekamas pastoviai
void loop() {
```


```
digitalWrite(led, HIGH); // išvade nustatome loginį vieneta (mūsų atveju 5V)
ir LED pradeda šviesti
delay(1000); // palaukiama 1 sekundė
digitalWrite(led, LOW); // išvade nustatome loginį nulį (visais atvejais
0V) ir LED nustoja šviesti
delay(1000); // palaukiama 1 sekundė
}
// šioje vietoje ciklas baigsis ir mikrovaldiklis pradės kartoti ciklą void
loop iš naujo.
```

EKSPERIMENTINĖ DALIS

1. Išnagrinėjama Arduino modulio plokštė ir surašomi pagrindiniai jos komponentai.
2. Prijungiamas Arduino modulis prie kompiuterio
3. Paleidžiama programa Arduino.exe
4. Atidaromas pavyzdyje esantis lemputės mirksėjimo programos kodas taip, kaip parodyta 2.7 pav.



2.7 pav. Šviesos diodo mirksėjimo programos atidarymas

5. Pasirenkama  . Kompiluojamas kodas ir automatiškai įkraunamas į Arduino modulį. Paspaudus "Upload" pradės mirksėti RX ir TX šviesos diodai, kurie praneša apie perkėlimą. Jei yra užrašomi klaidų pranešimai.

Pasibaigus programos įrašymui pradeda mirksėti žalias šviesos diodas Arduino modulio plokštėje, prijungtas prie 13 išvado.

6. Modifikuojama programa taip, kad mirksėjimas sulėtėtų 3 kartus. Užrašomi programos pakeitimai. Programa išbandoma ir užrašomos išvados. Jei yra užrašomi klaidų pranešimai.

7. Modifikuojama programa taip, kad vienu metu mirksėtų trys šviesos diodai, prijungti prie 2,3,4 išvadu. Užrašomi programos pakeitimai. Programa išbandoma ir užrašomos išvados.

ATASKAITA

- Aprašomas eksperimentinis darbas.
- Užrašomos išvados.
- Pateikiami programų modifikuoti fragmentai su komentarais.
- Paaiškinamos pinMode(), digitalWrite(), delay(), setup() ir loop() funkcijos.
- Paaiškinami terminai: komentaras per kelias eilutes; komentaras per vieną eilutę.
- Kaip patikrinamos programos klaidos? Užrašomi ir paaiškinami darbo metu gauti klaidų pranešimai.

2.3. Praktinis darbas Nr 2 „Bėganti šviesa“

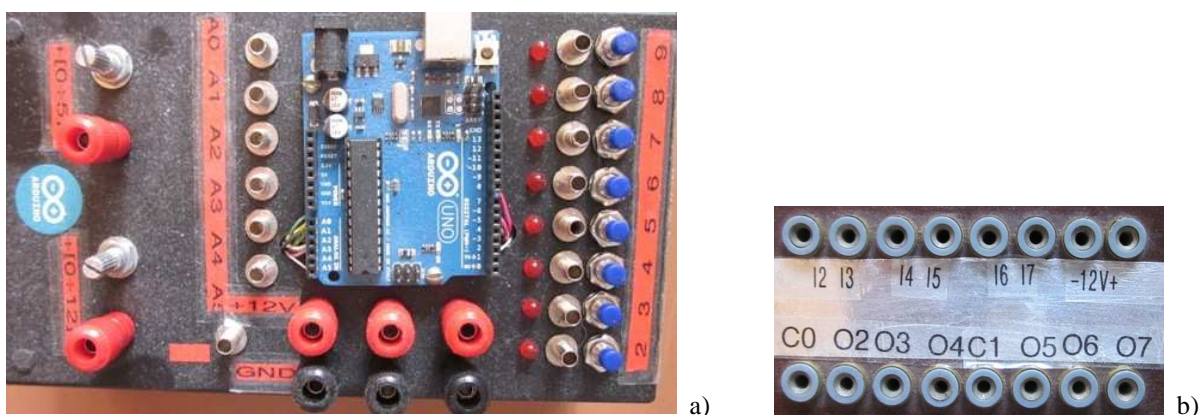
DARBO TIKSLAS: rašyti programą su mygtuku C kalba;

DARBO PRIEMONĖS. Kompiuteris su programine įranga *arduino-1.0.2-windows*, USB kabelis, Arduino stendas, 12 V maitinimo šaltinis, knyga (pdf.) „Arduino visiems“.

TEORINĖ DALIS

Praktiniam darbe naudojamas „Arduino“ praktinių darbų stendas parodytas 2.8 pav., a. Dešinėje yra diskretiniai Arduino modulio 8 išvadai (kištukiniai lizdai) nuo 2 iki 9. Jie gali būti užprogramuoti kaip įėjimai ar išėjimai. Arduino modulio įėjimo signalų įtampa +5 V. Šalia yra valdymo mygtukai. Nuspaudus mygtuką į atitinkamą įėjimą patenka +5 V signalas.

Jei išvadas užprogramuotas kaip išėjimas, tai jame gali būti 0 V arba 12 V signalai jei prijungta +12 V maitinimo įtampa. Kai išėjime atsiranda signalas pradeda šviesti atitinkamas indikacinis spinduolis. *Pastaba. Indikacija veikia jei yra prijungta +12 V maitinimo įtampa.*



2.8 pav. a) Arduino stendas, b) tarpinių relių stendas

Apačioje yra 3 poros maitinimo šaltinio gnybtų. Prie jų jungiama +12 V maitinimo įtampa. Kairėje yra 6 modulio analoginių įėjimų išvadai nuo A0 iki A5. Signalų įtampa juose gali keistis nuo 0V iki +5V. Atskiru atveju signalo įtampa gali keistis nuo 0V iki +10 V, jei prie atitinkamo modulio išvado prijungiama atraminė +5 V įtampa. Analoginio signalo įtampa gali būti gaunama iš jutiklių arba iš kairėje stendo esančių gnybtų. Įtampos dydis reguliuojamas potenciometrais.

Tarpinių relių blokas parodytas 2.8 pav., b. Jame yra įėjimai nuo I2 iki I7. Jie jungiami prie atitinkamų Arduino modulio išėjimų kištukiniais laidais. Modulio išėjimai yra reliniai (atviras kontaktas) ir suskirstyti į dvi grupes. Viena grupė O2, O3, O4. Jos išėjimų vienas išvadas sujungtas tarpusavyje ir su išvadu C0. Kita grupė yra O5, O6, O7. Jos bendras išvadas yra C1.

Kai Arduino modulio išėjime atsiranda signalas, relių modulyje suveikia atitinkama relė ir sujungia kontaktą. Atitinkamas išėjimas sujungiamas su bendru išvadu.

PASIRUOŠIMAS NAMUOSE

1. Parašoma programa su 3 šviesos diodais (prijungti prie 2, 3, 4 išvadų), kurie paeiliui persijungia kas sekundę.

2. Paaiškinama žemiau parodyta programa.

```
// constants won't change. They're used here to
```

```
// set pin numbers:
```

```
const int buttonPin = 2; // the number of the pushbutton pin
```

```
const int ledPin = 13; // the number of the LED pin
```

```

// variables will change:
int buttonState = 0;    // variable for reading the pushbutton status
void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}
void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);
  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}

```

EKSPERIMENTINĖ DALIS

1. Prijungiamas Arduino modulis prie kompiuterio
2. Paleidžiama programa Arduino.exe
3. Įkeliamas namuose parašyta programa ir išbandoma. Bandant programas reikia prijungti išorinį 12 V maitinimo šaltinį. Užrašomos pastabos ir klaidų pranešimai.

4. Modifikuojama programa .Pakeičiama programoje **delay (1000)** į *int time = 1000*; Programa išbandoma ir užrašomos pastabos.

5. Modifikuojama namuose parašyta programa taip, kad persijungimas prasidėtų spustelėjus mygtuką M1, prijungtą prie 8 išvado. Užrašomi pakeitimai. Programa išbandoma ir užrašomos pastabos.

5. Modifikuojama programa taip, kad persijungimas prasidėtų spustelėjus mygtuką M1, prijungtą prie 8 išvado. Spustelėjus mygtuką M2, prijungtą prie 9 išvado, persijungimas sustotų. Užrašomi pakeitimai. Programa išbandoma ir užrašomos pastabos.

ATASKAITA

- Aprašomas eksperimentinis darbas.
- Užrašomos išvados.
- Pateikiami programų modifikuoti fragmentai su komentarais
- Paaiškinamos *int time = 1000* ir *int multiple = 2* eilutės,

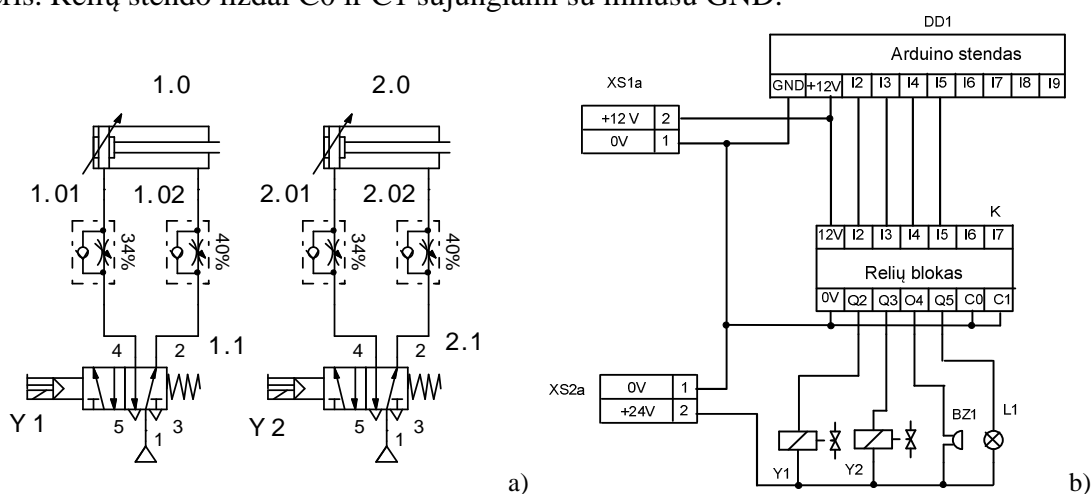
2.4. Praktinis darbas Nr.3. Pneumatikos cilindrų valdymas

DARBO TIKSLAS: taikyti elektropneumatikos schemose valdiklius, gebėti jungti elektropneumatikos schemas, gebėti rašyti programas valdikliui C kalba.

DARBO PRIEMONĖS: Arduino stendas, tarpinių relių blokas, pneumatikos dvikrypčiai cilindrai, elektropneumatikos skirstytuvai 5/2 monostabilūs, 24 V lemputė ir zumeris, kompiuteris su įdiegta programa *arduino-1.0.2-windows*, USB adapteris, 12 V ir 24 V maitinimo šaltiniai, oro paruošimo mazgas, programos rašymo metodiniai nurodymai „Arduino visiems“.

TEORINĖ DALIS

Šiuolaikinėse automatikos sistemose signalus priima, apdoroja ir nukreipia į vykdymo įtaisus įvairūs valdikliai. Jie pakeičia relines schemas. Vykdyto įtaisai valdomi pagal parašytą programą. Prie Arduino stendo kištukiniais laidais per tarpinių relių bloką prijungiami du 24 V elektropneumatikos skirstytuvai 5/2 be atminties, 24V lemputė ir 24V zumeris. Relių stendo lizdai C0 ir C1 sujungiami su minusu GND.



2.9 pav. Elektropneumatikos komponentų stendas: a) pneumatinė dalis, b) elektrinė dalis

Elektropneumatikos schemą sudaro dvi dalys: pneumatinė dalis 2.9 pav., a ir elektrinė dalis 2.9 pav., b. Pneumatinė dalis braižoma panaudojant specialią programą FluidSim-P. Pneumatinėje dalyje 1.0 ir 2.0 dvikrypčiai pneumatikos cilindrai, 1.01, 1.02, 2.01, 2.02 – droseliai, kurie reguliuoja cilindrų darbo greitį. 1.1 ir 2.1 – elektropneumatiniai skirstytuvai 5/2 (5 angos ir dvi padėtys) su gražinimo spyruokle. Trikampis – oro tiekimo lizdas.

Veikiant programai iš Arduino stendo ateina 12 V signalai. Nuo jų suveikia tarpinės relės.

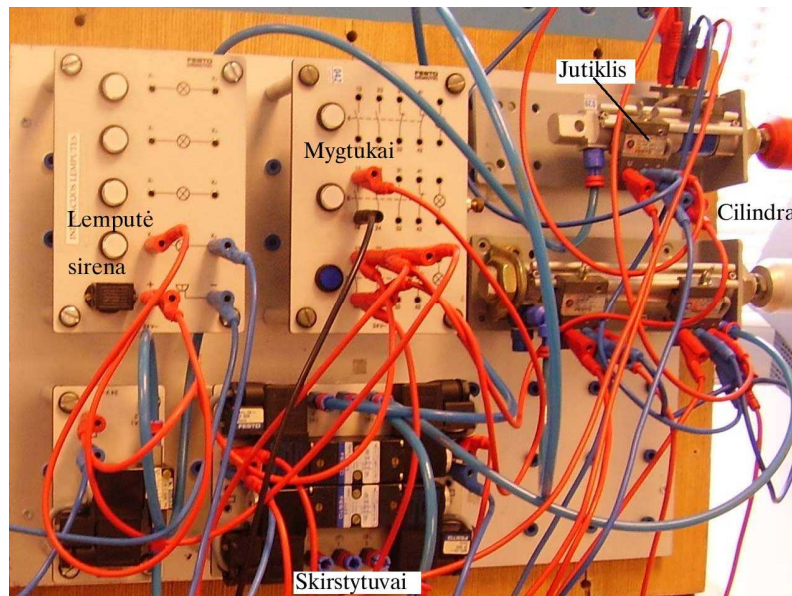
Kai skirstytuvai nesuveikę, suspaustas oras iš oro tiekimo lizdo 1 per skirstytuvo išėjimą 2 patenka į dešiniąją cilindro angą ir laiko cilindro kotą įtrauktą. Kai į skirstytuvo solenoidą (pvz., Y1) paduodamas iš valdiklio Q1 minusinis signalas, skirstytuvai persijungia ir jo išėjimas 2 sujungiamas su aplinka, o suspaustas oras patenka į išėjimą 4 ir per jį į cilindro 1.0 kairiąją angą. Cilindro kotas išstumiamas. Iš cilindro išstumiamas oras lėtai per droselį 1.02 ir per skirstytuvo angas 2 ir 3 patenka į atmosferą. Jei droselis bus daugiau atidarytas (pvz., 50%) tai kotas judės greičiau.

Kai dingsta įtampa skirstytuvo solenoide Y1 spyruoklė perjungia skirstytuvą į pradinę padėtį. Vėl suspaustas oras paduodamas per angą 2 ir cilindro kotas įtraukiamas. Jo greitį reguliuoja droselis 1,01.

Elektrinėje dalyje 2.9 pav., b Y1 ir Y2 skirstytuvų solenoidai, DD1 – valdiklis, SB1 – start mygtukas. BZ1 – zumeris, L1 – lemputė. Vienas visų komponentų išvadas sujungiamas su +24 V. Kai suveikia relių bloko atitinkamas išėjimas, pvz. Q2, per jį į solenoidą Y1

pasiduoda minusas, per solenoidą prateka srovė ir perjungia skirstytuvą. Jei suveikia Q5 išėjimas minusas paduodamas į lemputę ir ji šviečia.

Pastaba. Valdymo mygtukai yra Arduino stende prie išėjimų/įėjimų I8 ir I9.



2.8 pav. Elektropneumatikos stendo pavyzdys

PASIRUOŠIMAS DARBUI (namuose)

1. Parašoma programa, kurioje cilindras 1.0 išstumia kotą spustelėjus vieną mygtuką (I8), o spustelėjus kitą – (I9) kotą įtraukia.
2. Parašoma programa, kurioje spustelėjus mygtuką I8 cilindro 2.0 kotas išstumiamas ir būna 3 sek, po to įtraukiamas pastovi 3 sek ir kartoja be galo.

EKSPERIMENTINĖ DALIS

1. Sujungiama elektropneumatikos schema 2.9 pav. su Arduino stendu.
2. Stendas prijungiamas prie kompiuterio.
3. Namuose parašyta pirmoji programa įkeliama į valdiklį ir išbandoma. Užrašomos pastabos.
4. Įkeliama antroji namuose parašyta programa ir išbandoma. Užrašomos pastabos.
5. Modernizuojama 2 programa. Išstūmus 2.0 cilindro kotą pradeda šviesti lemputė. Užrašomi programos pakeitimai, programa išbandoma ir užrašomos pastabos.

6. Parašoma programa panaudojant visus elektropneumatikos komponentus. Paleidus programą lemputė pradeda mirksėti. Spustelėjus mygtuką I8 lemputė gęsta, 1.0 cilindro kotas išstumiamas. Po to išstumiamas 2.0 cilindro kotas. Įjungiamas garsinis signalas pypteli 1 kartą. Po 4 sek antro cilindro kotas įtraukiamas. Po jo įtraukiamas ir pirmo cilindro kotas ir lemputė pradeda mirksėti.

ATASKAITA

- Aprašomas eksperimentinis darbas.
- Užrašomos išvados.
- Pateikiami modernizuojama programų fragmentai.
- 6 punkto programos tekstas su paaiškinimais. Nubraižoma darbo diagrama.

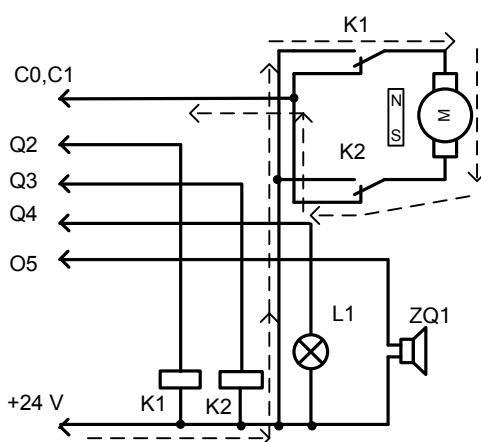
2.5. Praktinis darbas Nr.4 Nuolatinės srovės variklio programinis valdymas

DARBO TIKSLAS: taikyti elektromechanikos schemose Arduino modulį, gebėti jungti elektros schemas, gebėti rašyti programas mikrovaldikliui C kalba.

DARBO PRIEMONĖS: Arduino stendas, tarpinių relių blokas, nuolatinės srovės variklio stendas, 24 V lemputė ir zumeris, kompiuteris su įdiegta programa *arduino-1.0.2-windows*, USB adapteris, 12 V ir 24 V maitinimo šaltiniai, oro paruošimo mazgas, programos rašymo metodiniai nurodymai „Arduino visiems“.

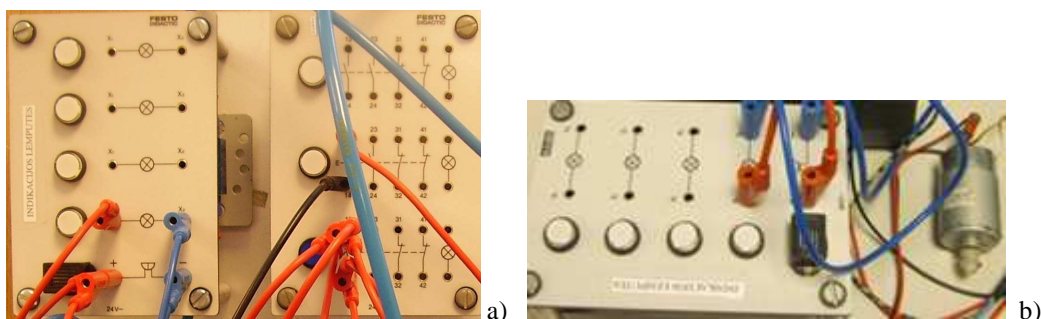
TEORINĖ DALIS

Šiuolaikinėse automatikos sistemose signalus priima, apdoroja ir nukreipia į vykdymo įtaisus valdikliai. Jie pakeičia relines schemas. Vykdyto įtaisai valdomi pagal parašytą programą. Prie Arduino modulio stendo kištukiniais laidais per tarpinių relių bloką yra prijungti 2.10 pav. parodyti komponentai. Tai nuolatinės srovės su pastoviais magnetais 24V elektros variklis, dvi 24 V elektromagnetinės relės, 24 V lemputė ir 24 V zumeris. Relių bloko lizdai C0 ir C1 sujungiami su minusu GND.



2.10 pav. Elektromechanikos komponentų stendas

Veikiant programai iš Arduino modulio į relių bloką patenka 12 V signalai. Atitinkama relė suveikia ir relių bloko išėjime gaunamas signalas minusas. Pvz., jei suveikia išėjimas Q2, per jį pasiduoda minusas į relę K1. Prateka srovė per relės K1 apviją ir relė suveikia. Ji perjungia savo kontaktą K1. Per variklio apviją teka srovė (iš +24 V, kontaktas K1, variklis M, kontaktas K2, minusas). Ji parodyta punktyru. Variklis sukasi į vieną pusę. Jei suveiks kita relė, variklis sukis į kitą pusę. Jei suveiks abi relės variklis nesisuks, nes persijungs abu kontaktai K1 ir K2.



2.11 pav. praktinio darbo komponentai: a) lempučių, zumeris ir mygtukai, b) variklio stendas

PASIRUOŠIMAS DARBUI (namuose)

1. Parašoma programa, kurioje spustelėjus mygtuką M1 variklis pasisuka 3 sek, pastovi 3 sek ir kartoja be galo. M1 prijungtas prie 7 Arduino modulio išvado.

2. Parašoma programa, kurioje yra 2 mygtukai. Jei pradžioje paspaudžiamas mygtukas M1,

Valdikliai. Praktiniai darbai

variklis pasisuka į dešinę 3 sek, pastovi 3 sek ir kartoja be galo. Jei pradžioje paspaudžiamas mygtukas M2, variklis analogiškai sukasi kairėn. M2 prijungtas prie 8 Arduino modulio išvado. Jei variklis sukasi mygtukai neveikia.

EKSPERIMENTINĖ DALIS

1. Prie arduino stendo prijungiamas relių blokas. Prie jo pagal 2.10 pav. prijungiami elektromechanikos komponentai ir prijungiamas 12V maitinimas.
2. Stendas prijungiamas prie kompiuterio.
3. Namuose parašyta pirmoji programa įkeliama į valdiklį ir išbandoma. Užrašomos pastabos.

4. Modernizuojama programa. Įvedamas mygtukas „Stop“. Jis prijungtas prie 9 Arduino modulio išvado. Mygtuką nuspaudus variklis sustoja, trumpam įjungiamas zumeris ir programa grįžta į pradžią. Programa išbandoma ir užrašomos pastabos.

5. Įkeliama antroji namuose parašyta programa ir išbandoma. Užrašomos pastabos.

6. Modernizuojama 2 programa. Įjungus maitinimą pradeda mirksėti lemputė. Paspaudus M1 ar M2 variklis pradeda sukis, o lemputė užgęsta. Nuspaudus mygtuką „Stop“ Variklis sustoja, lemputė pradeda mirksėti ir vėl galima spausti mygtukus. Kai variklis sukasi veikia tik mygtukas „Stop“. Užrašomi programos pakeitimai, programa išbandoma ir užrašomos pastabos.

ATASKAITA

- Aprašomas eksperimentinis darbas.
- Užrašomos išvados.
- Pateikiami modernizuojama programų fragmentai.

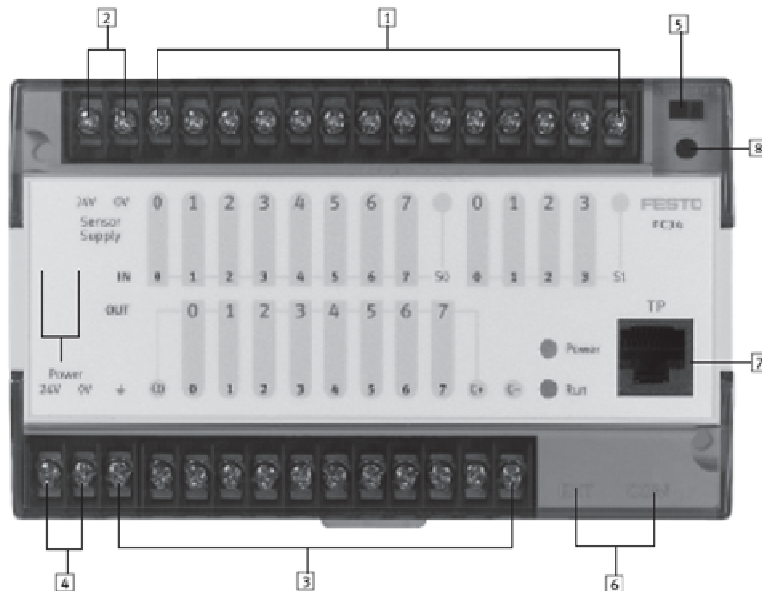
3 „FESTO“ PROGRAMUOJAMAS LOGINIS VALDIKLIS

3.1. Valdiklio išvadai

Praktiniuose darbuose naudojamas Festo firmos *Compact* tipo programuojamas loginis valdiklis FC34. Valdiklis turi 12 įėjimų ir 8 išėjimus. Jo priekis parodytas 3.1 pav. Priekyje yra kompiuterinio tinklo jungtis 7, jutiklių maitinimo šaltinio išvadai 2, maitinimo (24 V) išvadai 4, programos paleidimo/stabdymo jungiklis 5, programos darbo režimo indikatorius 8. Jei jis šviečia žaliai – programa veikia, jei raudonai – neveikia. Adapterio prijungimo ir praplėtimo lizdai 6. Prie vieno (kairiojo) jungiamas laidas iš RS232 prievado adapterio. Prie kito – antras PLV.

Įėjimai (viršuje 1) yra suskirstyti į dvi grupes. Vienoje grupėje yra įėjimai nuo I0.0 iki I0.7. Kitoje – nuo I1.0 iki I1.3. Kiekviena iš grupių turi bendrą išvadą S0 ir S1. Išėjimai (apačioje 3) taip pat suskirstyti į dvi grupes. Pirmoje grupėje yra du reliniai išėjimai O0.0 ir O0.1, kurie turi bendrą išvadą C0. Kitoje grupėje yra 6 tranzistoriniai išėjimai nuo O0.2 iki O0.7. Jie turi bendrą išvadą C+.

PLV įmontuoti į praktinių darbų standus, kuriuose yra tarpinės relės. Visi standų išėjimai yra reliniai.



3.1pav. Programuojamas loginis valdiklis FC 34

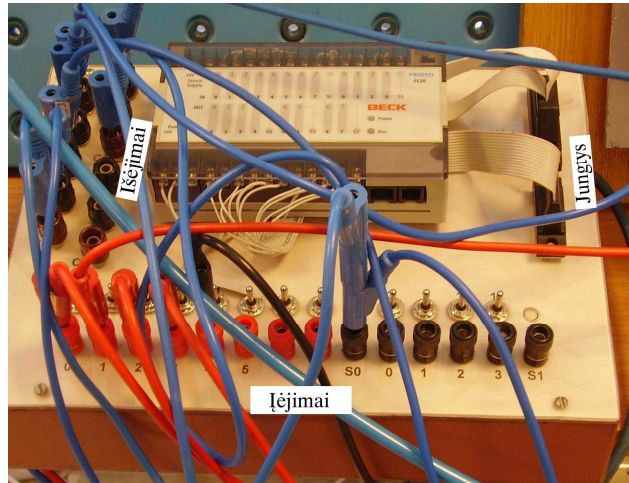
Valdiklis prie kompiuterio jungiamas per adapterį prie RS232 prievado. Taip pat galima valdiklį jungti į kompiuterių tinklą.

3.2. Praktinis darbas Nr1. Nuosekli programa

DARBO TIKSLAS: rašyti programą STL kalba

DARBO PRIEMONĖS: PLV laboratorinis stendas, kompiuteris, valdiklio ir kompiuterio sujungimo adapteris.

TEORINĖ DALIS



3.2 pav. PLV stendas

3.2.1. Projekto sukūrimas

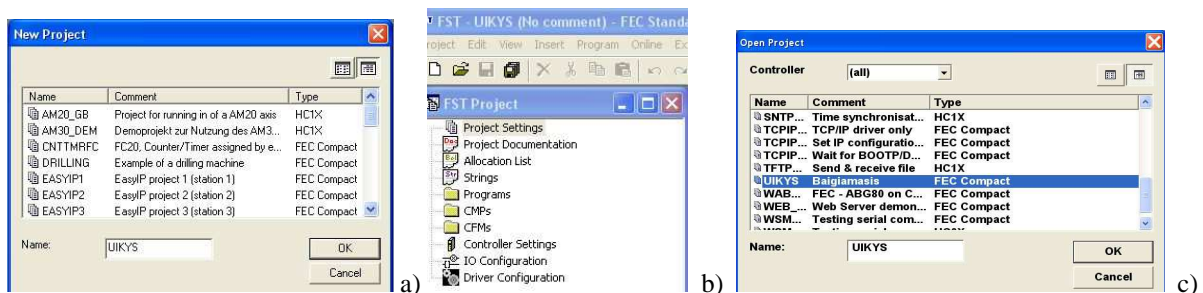
Programos rašymui ir valdiklio programavimui naudojamas programų paketas FST4.10. Programa rašoma STL kalba.

Paleidus programą per meniu „**START**→**Programs**→**Festo Software**→**FST 4.10**→**FST 4.10**“ pasirodo meniu langas 3.3 pav.



3.3 pav. Meniu langas

Meniu punkte [*Project*] pasirenkamas [*New*]. Atsidaro langas (3.4 pav., a), kur langelyje *Name* įrašomas naujo projekto pavadinimas (*UIKYS*) ir spaudžiamas „*OK*“ mygtukas. Rekomenduojama projekto pavadinime naudoti 7 pavidės raides. Jei projektas ne naujas, bet tęsiamas senas, tada pasirenkama *Open* ir atsidariusiame lange pasirenkamas projekto pavadinimas (*UIKYS* – 3.4 pav., c).



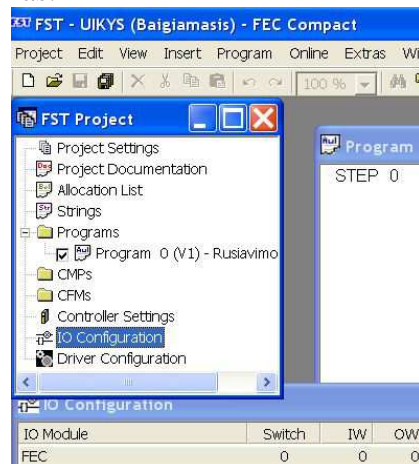
3.4 pav. Naujas projektas arba projekto pasirinkimas

Lange *FST Project* (3.4 pav., b) pasirenkama *Project Setings*. Langelyje „*Controller*“ pasirenkamas naudojamojo valdiklio tipas, tai yra *FEC Compact* 3.5 pav. Langelyje „*Comment*“ rašomas komentaras (nebūtina).



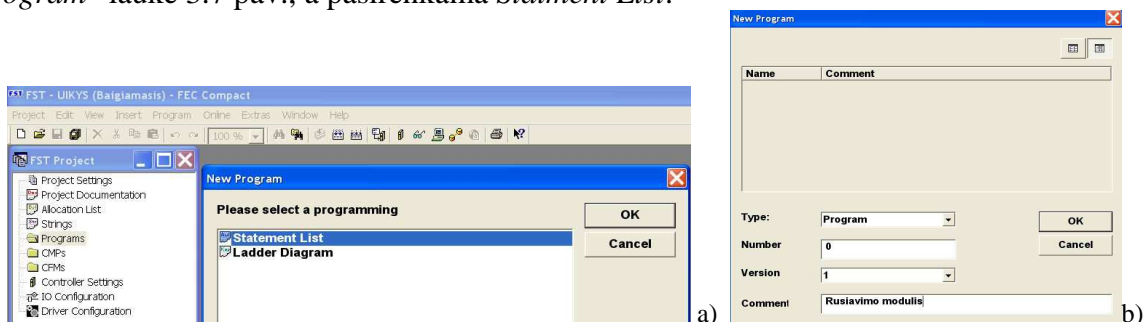
3.5 pav. PLV parinkimas

Po to reikia sukonfigūruoti įėjimo/išėjimo modulius Tam kairėje pusėje esančiame langelyje (3.6 pav.) dukart paspaudžiamas punktas „*IO Configuration*“ ir naujame atsidariusiame baltame lange paspaudžiamas dešinys pelės klavišas. Iš atsidariusio meniu pasirenkamas „*Insert IO module*“. Atsidariusiame naujame langelyje pasirenkamas valdiklio modelis *FEC* ir spaudžiamas „**OK**“ mygtukas. Atsiranda skaičiai 0,0,0 (3.6 pav. apačia). Nuliai atsiranda tada, kai naudojamas tik vienas valdiklis. Pasirinkus valdiklį „*IO Configuration*“ langas uždaromas.



3.6 pav. Įėjimo/išėjimo modulių konfigūracija

Lange „*FST Project*“ (3.6 pav.) pasirenkama „*Programs*“ dešiniu pelės klavišo paspaudimu. Iš meniu pasirenkama „*Insert Program*“ lauką. Atsiradusiame lange „*New Program*“ lauke 3.7 pav., a pasirenkama *Statement List*.



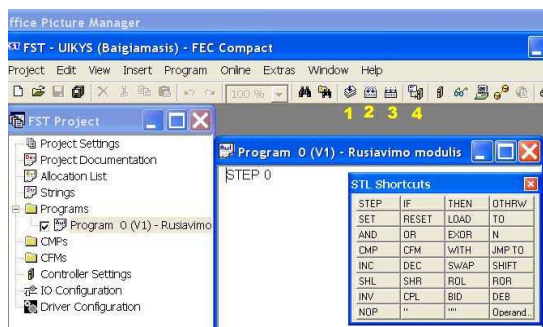
3.7 pav. a) Programos kalbos ir b) programos numerio pasirinkimas

Nuspaudus *OK* atsidaro langas. 3.7 pav.,b. Jame *Number* – programos numeris. „*Version*“ – programos versija. „*Comment*“ – komentaras apie programą. Šiame lange pakanka užrašomos

komentarą, pvz. programos paskirtį. Spaudžiama OK ir galima pradėti rašyti programą.

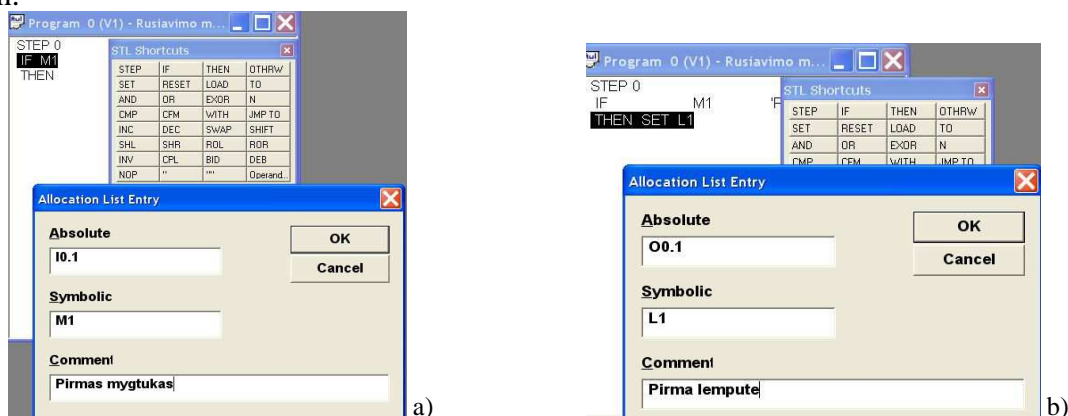
3.2.2. Programos rašymas

Programa rašoma žingsniais – *STEP*. Žingsnio pradžioje surašomos sąlygos, o antroje dalyje surašomi veiksmai.



3.8 pav. Programos rašymo langas

Programos lange yra komandų meniu – *STL Shortcuts*. Spragtelėjus kairiu pelės klavišu ant komandos, ji atsiranda programoje. Programa pradeda žingsniu *STEP 0*. Žingsnių numeriai eina didėjančia tvarka, bet nebūtinai iš eilės. Rekomenduojama *Step 2*, *Step 4* ir pan.



3.9 pav. Adresų priskyrimas

Žingsnis pradamas *IF...* – jei yra mygtuko (jutiklio) M1 signalas (3.9 pav.) Po M1 spaudžiama *ENTER*. Atsidaro langelis 3.9 pav., a, kuriame įrašoma prie kurio valdiklio įėjimo prijungtas mygtukas – *Absolute 10.1*. Čia galima įrašyti komentarą (pirmas mygtukas). Spaudžiama *OK* ir langelis dingsta. Programoje atsiranda eilutė, užsibaigianti komentaru. Šioje žingsnio dalyje galima naudoti komandas *AND*, *OR*, *N* ir pan. Pvz., *M1 AND M2* – jei yra signalai iš M1 ir M2. *M1 OR M2* – jei yra signalai iš M1 arba M2.

Antra dalis pradeda *THEN ...* Čia rašomi veiksmai, kuriuos reikia atlikti: įjungti (*SET*), išjungti (*RESET*), peršokti (*JUMP TO*) ir t. t. (3.9 pav., b). *SET L1* – įjungti lemputę L1. Po L1 spaudžiama *ENTER*. Atsidaro langelis, kuriame užrašoma, prie kurio valdiklio išėjimo prijungta lemputė – *Absolute 00.1* (nulinio išėjimo). Kai surašoma ką reikia įjungti ar išjungti spaudžiama *OK*. Po to rašomas kitas žingsnis – *STEP 2* ir t. t. Šioje žingsnio dalyje negalima naudoti komandų *AND* ir *OR*.

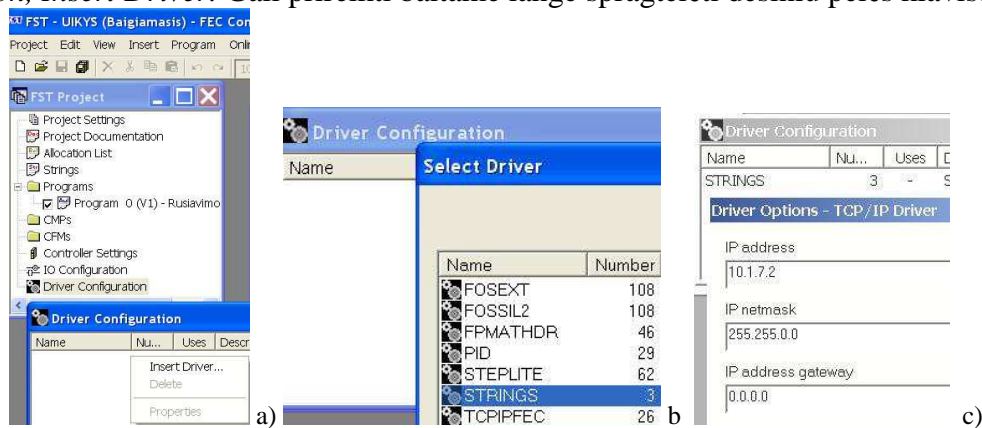
Visada galima pažiūrėti kur ir kas prijungta. Lange FST Project 3.11 pav., a pasirenkama Allocation List (3.10 pav.). Čia matyti, pvz., kad VZ (relė) prijungta prie išėjimo 00.0, jutiklis B1 prijungtas prie įėjimo I0.0 ir t.t. prijungimus galima pakeisti. Reikia norimoje vietoje paspausti pelės klavišą ir atsidariusiame lange atlikti pakeitimą.

Operand	Symbol	Comment
O0.0	VZ	Variklis zėmyn
O0.1	VA	Variklis aukstyn
I0.0	B1	apatinis jutiklis
I0.1	M3	3 auksto mygtukas
I0.2	M2	2auksto mygtukas

3.10 pav. Prijungimų langas

Parašius programą tikrinama sintaksė. Tam meniu spaudžiamas 1 mygtukas (3.8 pav.). Atsidariusiame lange (apačioje, kairėje) parodoma kurioje eilutėje yra klaidos. Vėl grįžtama į programą ir klaidos taisomos.

Kai klaidų nėra prie programos prijungiamos tvarkyklės. Jungiant per kompiuterių tinklą reikia šių tvarkyklių: *STRINGS*, *TCPIPFEC*. Pasirenkama (3.11 pav., a): *Driver Configuration*, *Insert Driver*. Gali prireikti baltame lange spragtelėti dešiniu pelės klavišu.

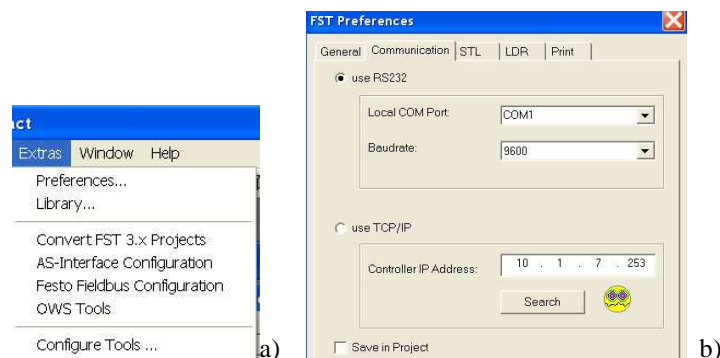


3.11 pav. Tvarkyklių prijungimas prie programos.

Pasirenkama tvarkyklė 3.11 pav., b lange (pvz., *STRINGS*) ir spaudžiama OK. Po to pasirenkama tvarkyklė *TCPIPFEC*. Joje nurodomas valdiklio IP adresas, pvz., 3.11 pav.,c.

3.2.3. PLV prijungimas ir programavimas

Programuojamas loginis valdiklis prijungiamas prie kompiuterio RS232 prievado per specialų adapterį. Nustatomi programos parametrai. Meniu pasirenkama *Extras, Preferences* (3.12 pav.,a) ir atsidaro langas (3.12 pav.,b).



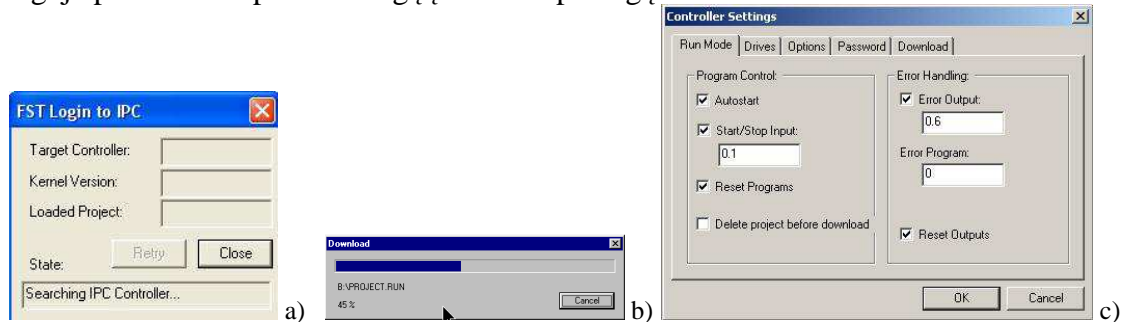
3.12 pav. PLV prijungimas prie kompiuterio

Jame taškeliu pažymima *use RS232* ir pasirenkamas prievadas – *COM1* arba *COM2* bei greitis *9600*. Varnele pažymima *Save in Project*.

Jei valdiklis jungiamas per kompiuterių tinklą, tada reikia taškeliu pažymėti *use TCP/IP* ir įrašyti valdiklio IP adresą (pvz., 10.1.7.253). Prieš tai turi būti įkrauta tvarkyklė

TCPIPFEC (3.11 pav., b) ir joje įrašytas valdiklio adresas 3.11 pav., c).

Po to, kai nustatytas prisijungimo būdas (per COM1 ar tinklą), spaudžiami paėiliui meniu mygtukai 2, 3 (3.8 pav.). Tada visos projekto dalys (programos, paprogramės, protokolai ir pan.) sujungiamos į visumą. Tai parodoma atsidariusiame lange apačioje. Jame užrašomos esamos klaidos. Po to įkraunama programa į valdiklį. Spaudžiamas 4 meniu mygtukas. Atsidariusiame lange 3.13 pav., a parodoma ar surastas valdiklis, gali būti paklausta ar keisti programą (*YES*) ir rodoma kaip įkraunama programa 3.12 pav., b. Pabaigoje pranešama apie sėkmingą įkrovimo pabaigą.



3.13 pav. a) PLV paieška, b) programos įkrovimas, c) PLV nustatymai

Programos paleidimas. Meniu pasirenkamas 5 mygtukas 3.14 pav. *Control Panel*. Atsidariusiame lange paleidžiama programa. Trikampis mygtukas programą valdiklyje paleidžia, o stačiakampis – stabdo. Jei nėra ryšio su valdikliu, pasirodo langas 3.13 pav., a ir ieškomas valdiklis. Pačiame valdiklyje yra indikatorius *Run*. Jei jis žalias – programa veikia, jei raudonas – programa baigė darbą.



3.14 pav. Programos paleidimas

Galima po įkrovimo programą paleisti automatiškai. Tam prieš įrašant programą atidaromas langas *Controller Settings* ir varnele pažymima *Autostart* 3.13 pav., c. *Autostart* nuostata kiekvieną kartą nusiuntus programą perkrauna valdiklį ir paleidžia iš naujo nusiųstąją programą. *Start/Stop Input* nuostatoje galima pasirinkti kurio įėjimo pagalba bus galima paleisti ir sustabdyti programos vykdymą. *Error Output* nuostatoje pasirenkamas išėjimas, kuriame bus vienetinis lygis tada, kai valdiklyje įvyks klaida.

Programą taip pat paleisti galima rankiniu būdu. Valdiklyje yra 5-as jungiklis *Run* 3.1 pav., 5.

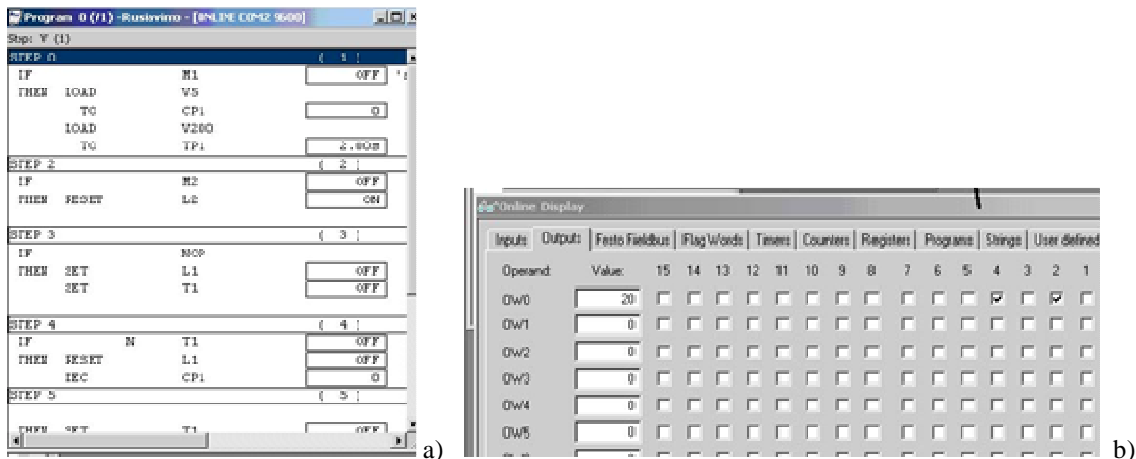
3.2.4. Programos išsaugojimas ir pernešimas.

Programa automatiškai išsaugoma direktorijoje *FST4, Project*. Norint pernešti programą sukuriama zip failas, pvz., *jonas1.zip*. Vienetas reiškia, kad pirma programa. Pasirenkama *Project, Backup* ir pasirenkama įrašymo vieta (direktorija *Darbai* arba *atmintukas*) ir failo pavadinimas *jonas1.zip*. Spaudžiama *OK*.

Failas atgaminamas pasirenkant *Project, Restore* ir surandama įrašymo vieta (direktorija *Darbai* arba *atmintukas*) ir failo pavadinimas *jonas1.zip*. Spaudžiama *OK*.

3.2.5. Darbas Online režimu

Programos lange spragtelėjus dešiniu klavišu atidaromas langas, kuriame pasirenkamas darbo režimas *Online*. Atsidaro langas 3.15 pav., a. Mėlyna linija rodo kuris programos žingsnis vykdomas. Dešinėje lango pusėje matyti, kurie įėjimai ir išėjimai turi signalus. Šiame lange spragtelėjus dešiniu klavišu atsidariusiame lange pasirenkama režimas *Editor* ir grįžtama į programos rašymo režimą.

3.15 pav. Darbas *Online* režimu

Nuspaudus meniu mygtuką 6 (3.14 pav.), atsidaro langas *Online Display* (3.15 pav., b), kur galima matyti įėjimų, išėjimų, skaitiklių (*Counters*) laikmačių (*Timers*) ir kitas būsenas. Čia pasirinkta valdiklio išėjimai *Outputs*. Varnelė reiškia kad išėjimas (parodytu atveju 4 ir 2) įjungtas, t.y. relės kontaktas sujungtas. Sustabdžius programą (paspaudus stačiakampį) galima norimą išėjimą įjungti (varnelė) ir pažiūrėti ar valdomame mechatronikos modulyje įvyko pakitimas. Išsikvietus *Inputs* galima matyti kurie jutikliai suveikę arba patikrinti ar jie geri. Jutikliui suveikus atsiranda atitinkamame langelyje varnelė.

EKSPERIMENTINĖ DALIS

1. Paleidžiama programa FST4.
2. Sukuriamas naujas projektas savo vardu.
3. Parašoma programa: Nuspaudus mygtuką M1 pradeda šviesti lemputė L1. Nuspaudus mygtukus M2 ir M3 kartu, lemputė užgęsta ir programa grįžta į pradžią.
4. PLV prijungiamas prie kompiuterio (jei nebuvo prijungtas) ir įjungiamas programuojamo loginio valdiklio maitinimas.
5. Įkeliama programa į PLV ir išbandomas jos veikimas.
6. Pereinama į *Online* režimą ir stebimas programos darbas.
7. Įrašomas į direktoriją *Darbai arba atmintinę* programos zip failas.

ATASKAITA

- Aprašomas eksperimentinis darbas. Problemos ir jų sprendimo būdas.
- Programos tekstas.
- Išvados

3.3 Praktinis darbas Nr. 2. Programa su laikmačiu ir skaitikliu.

DARBO TIKSLAS: rašyti programas su laikmačiu ir skaitikliu STL kalba, susipažinti su komandų *OTHRW* ir *JMP TO* taikymu, perkelti projektą iš vieno kompiuterio į kitą.

DARBO PRIEMONĖS: PLV laboratorinis stendas, kompiuteris, valdiklio ir kompiuterio sujungimo adapteris.

TEORINĖ DALIS

3.16 pav., a parodytas programos fragmentas, kuriame naudojama komanda *OTHRW* (kitu atveju). Ji nurodo ką daryti jei nevykdoma pirmoje žingsnio dalyje nurodyta sąlyga. Paprasčiausiu atveju įrašoma komanda *NOP* ir programa pereina prie kito žingsnio.

Programos fragmente apklausiami 3 mygtukai. Nuliniame žingsnyje programa bus tol, kol nebus nuspaustas mygtukas M1. tik po to bus įjungta L1 ir pereita prie kito žingsnio. 3 žingsnyje programa nesustos. Patikrinama ar nuspaustas mygtukas M2. Jei nuspaustas – įjungžiama L2. Jei ne – programa pereina į 5 žingsnį. Čia ji sustoja ir laukia kol bus paspaustas M3. Tada įjungžiama L3 ir programa baigia darbą.

<pre> STEP 0 IF M1 THEN SET L1 STEP 3 IF M2 THEN SET L2 OTHRW NOP STEP 5 IF M3 THEN SET L3 </pre>	<pre> STEP 0 IF M1 THEN SET L1 OTHRW NOP STEP 3 IF M2 THEN SET L2 OTHRW NOP STEP 5 IF M3 THEN SET L3 OTHRW JMP TO 0 </pre>
a)	b)

3.16 pav. a) komandų *OTHRW* ir b) *JMP TO* taikymas

Komanda *JMP TO* nurodo į kurią vietą programa turi persokti. Pvz., 3.16 pav., b programa persoks į nulinį žingsnį. Mygtukai bus apklausiami paeiliui nuolat jei nei vienas iš jų nebus nuspaustas. Toks programos fragmentas naudojamas kai galima spausti mygtukus ne iš eilės. Jei bus paspausti M1 ar M2 tai bus įjungžiama L1 ar L2 ir programa toliau apklaus mygtukus. Tačiau nuspaudus M3 bus įjungžiama L3 ir programa sustos. Pvz., jei pradžioje paspaudžiamas M3, tai įjungžiama L3 ir programa sustoja. Kad to nebūtų reikia po L3 įjungimo įterpti papildomą eilutę *JMP TO 0*.

Laikmatis naudojamas kai reikia suformuoti laiko intervalą. PLV turi 32 laikmačius. Jie programoje žymimi T1, T5 ir t. t. Prieš panaudojimą laikmatis įkraunamas. Į jo atmintį pvz., TP1 ar TP5 įrašomas reikiamas skaičius, pvz., V50 ar V100. V raidė rodo, kad tai skaičius, 100 reiškia 1 sekundę.

Laikmačio įkrovimas dažniausiai atliekamas nuliniame žingsnyje 3.17 pav., a. Čia paruošiamas laikmatis T0 vienai sekunde. Laikmatis įkraunamas vieną kartą.

<pre> Program 0 (V1) - e_pneum* STEP 0 IF NOP THEN LOAD V100 TO TP0 STEP 3 IF SB1 'start mygtukas THEN SET CIL1 </pre>	<pre> Program 3 (V1) - P3* STEP 10 THEN SET L2 STEP 13 IF J5 THEN SET T1 STEP 14 IF N T1 THEN SET L1 STEP 18 </pre>
a)	b)

3.17 pav. a) laikmačio įkrovimas, b) laikmačio panaudojimas

Laikmačio panaudojimas programoje parodytas 3.17 pav., b. Tai atliekama 13-ame žingsnyje. Jei suveikė jutiklis J5, tada įjungžiamas laikmatis (SET T1). Nuspaudus po T1 *Enter*, atsidaro langas. Jame galima nieko nežymėti, tik spausti OK.

Sekančiame žingsnyje *STEP 14* būtina laukti kol pasibaigs laikmačio laikas. Kai nėra laikmačio T1 (*IF N T1*), Tada įjungiami L1 (*SET L1*) ir pereinama prie kito žingsnio. Tą patį laikmatį galima naudoti įvairiose programos vietose.

Laikmatis naudojamas mirksinčios lemputės programoje. Programos algoritmas paprastas. Lemputė įjungiami ir įjungiamas laikmatis. Kai laikmatis baigiasi lemputė išjungiami ir vėl įjungiamas laikmatis. Kai laikmatis baigiasi peršokama į programos pradžią.

Skaitiklis programoje naudojamas, kai reikia suskaičiuoti, kiek kartų suveikė vykdymo įtaisas ar kiek valdymo impulsų pateko į valdiklio įėjimą. Skaitiklis gali būti realizuojamas panaudojus valdiklio registrą. JRegistras žymimas R0, R1... Iš viso yra 32 registrai. Kaip į programą įterpiamas skaitiklis parodyta 3.18 pav.

```

STEP 12
IF          NOP
THEN LOAD  V0
   TO      R0
STEP 16
IF          ????
THEN      ???
-----|
          INC          R0
STEP 18
IF      (  R0
         <  V5  )
THEN  JMP TO 16
OTHRW  NOP

```

3.18 pav. Skaitiklio realizavimas

12 žingsnyje išvalomas registras R0, t.y. jame įrašomas 0. 16 žingsnyje registro turinys padidinamas 1 (*INC R0*). Kitame 18 žingsnyje registro turinys palyginamas su užduotu skaičiumi (5). Jei skaičius registre mažesnis už užduotą programa šoka į 16 žingsnį ir kartoja eilę komandų. Jei skaičius registre yra 5 vykdoma komanda *OTHRW* ir programa pereina į kitą žingsnį (stabdoma).

PASIRUOŠIMAS DARBUI (namuose)

1. Parašoma programa. Yra 3 mygtukai: M1, M2 ir M3. Jie spaudžiami paeiliui. Nuspaudus mygtuką įjungiami atitinkama lemputė: M1 – L1, M2 – L2, M3 – L3. Po trečio mygtuko paspaudimo programa grįžta į pradžią. Sukuriamas programos zip failas.
2. Parašoma analogiška programa, kurioje mygtukai gali būti spaudžiami bet kokia eile. Sukuriamas programos zip failas.
3. Parašoma mirksinčios kas 1 sek. lemputės programa ir sukuriamas programos zip failas.

EKSPERIMENTINĖ DALIS

1. Įkeliama į valdiklį pirma namuose parašyta programa ir išbandoma. Užrašomos pastabos.
2. Modernizuojama programa taip, kad vienu metu degtų tik viena lemputė. Užrašomos pakeistos programos eilutės ir pastabos. Įsirašomas zip failas.
3. Įkeliama į valdiklį antra namuose parašyta programa ir išbandoma. Užrašomos pastabos.
4. Modernizuojama programa taip, kad vienu metu degtų tik viena lemputė. Užrašomos pakeistos programos eilutės ir pastabos. Įsirašomas zip failas.

5. Įkeliama į valdiklį trečia namuose parašyta programa ir išbandoma. Užrašomos pastabos.

6. Modernizuojama programa taip, kad lemputė pradėtų mirksėti tik paspaudus mygtuką M1 ir nustotų mirksėti nuspaudus M2. Užrašomos pakeistos programos eilutės ir pastabos. Įsirašomas zip failas.

7. Modernizuojama 5 punkto programa taip, kad lemputė pradėtų mirksėti nuspaudus M1. Sumirksėjusi 6 kartus užgęstų ir vėl pradėtų mirksėti nuspaudus M1. Užrašomos pakeistos programos eilutės ir pastabos. Įsirašomas zip failas.

ATASKAITA

- Aprašomas eksperimentinis darbas.
- Pateikiami pakeistų programų tekstai.
- Išvados.

3.4. Praktinis darbas Nr.3. Paprogramės

DARBO TIKSLAS: rašyti programas su paprogramėm STL kalba.

DARBO PRIEMONĖS: PLV laboratorinis stendas, kompiuteris, valdiklio ir kompiuterio sujungimo adapteris.

TEORINĖ DALIS

Dažniausiai programose be pagrindinės – nulinės programos P0 būna kelios paprogramės. Jos žymimos P1, P2 ir t.t. Vienu metu gali veikti visos paprogramės ir nulinė programa. Projektas be nulinės programos P0 neveiks.

Nulinėje programoje dažniausiai įkraunami laikmačiai ir aprašomi valdymo elementai. Paprogramės valdo atskirus įtaisus, prijungtus valdiklio išėjime. Paprasčiausi atveju tai gali būti dvi lemputės, kurios mirksi skirtingu dažniu. Vieną lemputę valdo viena paprogramė, pvz., P1, o kitą – P2.

```

STEP 0
IF      NOP
THEN LOAD V100
      TO TP0
      LOAD V200
      TO TP1
STEP 4
IF      M1
THEN SET P1
OTHRW  NOP
STEP 7
IF      M2
THEN SET P2
OTHRW  NOP
STEP 10
IF      M3
THEN RESET P1
      RESET P2
      JMP TO 4
OTHRW  JMP TO 4
a)
STEP 0
IF      NOP
THEN SET L1
      SET T0
STEP 3
IF      N TO
THEN RESET L1
      SET T0
STEP 5
IF      N TO
THEN JMP TO 0
b)

```

1.19 pav. a) Pagrindinė programa P0, b) paprogramė P1

Pagrindinėje programoje 3.19 pav., a pirmos lemputės mirksėjimą įjungia mygtukas m1, o antros – M2. Trečias mygtukas išjungiantis. Jį nuspaudus lemputės nustoja mirksėti. Antros lemputės paprogramė P2 analogiška P1, tik panaudojamas kitas laikmatis T1.

PASIRUOŠIMAS DARBUI (namuose)

1. Parašoma dviejų mirksinčių lempučių programa su paprogramėm pagal 3.19 pav. Sukuriamas programos zip failas.
2. Parašoma mirksinčios lemputės programa. Lemputė sumirksi 6 kartus ir užgęsta. Sukuriamas programos zip failas.

EKSPERIMENTINĖ DALIS

1. Įkeliama namuose parašyta programa į valdiklį ir išbandoma kelis kartus. Stebimas programos darbas *ONLINE* režime. Užrašomos pastabos apie problemas. Kodėl nuspaudus M3 kartais lieka šviesti lemputė?

2. Modernizuojama programa taip, kad išvengti problemų. Programa išbandoma ir užrašomos pastabos.

3. Parašoma ir pademonstruojama savo sugalvota programa su paprogramėm panaudojant namuose parašytą mirksinčios lemputės su skaitikliu paprogramę. Sukuriamas programos zip failas.

ATASKAITA

- Aprašomas eksperimentinis darbas.
- Pateikiami pakeistų programų tekstai.
- Užrašomos išvados.

ATSAKYTI Į KLAUSIMUS

- Kaip laikinai pašalinama paprogramė?
- Kaip automatiškai paleidžiama programa?
- Kas rodo, kad programa veikia?

4. „CROUZET“ PROGRAMUOJAMAS LOGINIS VALDIKLIS

4.1. Valdiklio praktinių darbų stendas



4.1 pav. „Crouzet“ valdiklio praktinių darbų stendas

Valdiklio stendas parodytas 4.1 pav. Jame yra *CD 12 24VDC Milenium 3 SMART* valdiklis su keturiais diskretiniais I1 – I4 ir keturiais analoginiais įėjimais IB – IE (*4 DISCR +4(0-10V)*) ir keturiais reliniais išėjimais Q1 – Q4 (*4 RELAY*). Valdiklio įėjimai prijungti prie kištukinių lizdų. Prie jų prijungti ir valdymo mygtukai. Nuspaudus mygtuką, įėjime atsiranda +24 V įtampa.

Į analoginius įėjimus galima paduoti reguliuojamą įtampą nuo 0 V iki +10 V iš kištukinių lizdų, pažymėtų +(0 ÷ 10)V. Įtampa reguliuojama dviem potenciometrais. Reguluojama įtampa įjungžiama šone esančiu jungikliu. Pradinėje padėtyje jungiklis išjungtas ir indikacinis spinduolis nešviečia.

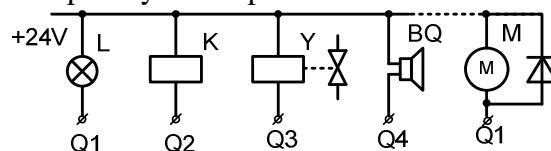
Valdiklio išėjimai yra relės atviri kontaktai. Dviejų kontaktų galai sujungti su gnybtu C0, o kitų kontaktų galai – su C1. Prie kitų kontaktų galų prijungti gnybtai Q1 ÷ Q4.

Stendas maitinimas +24 V įtampa. Ją rodo indikacinis spinduolis.

Programa valdikliui rašoma naudojant programų paketą *CLSM3-V2-3-AC5*. Jos pagrindinis langas parodytas 4.2 pav.

Elektromechanikos komponentų prijungimas prie stendo.

Prie stendo gali būti jungiami: elektropneumatikos skirstytuvai, elektromagnetinės relės, lemputės, zumeris ir pan. Vienas reikalavimas – komponentų maitinimo įtampa turi būti +24 V. Vienas komponento išvadas jungiamas prie +24 V, o kitas – prie stendo lizdų Q1, Q2, Q3, Q4. Komponentų prijungimas parodytas 4.2 pav.



4.2 pav. Komponentų prijungimas: L-lemputė, K- relė, Y-elektropneumatinio skirstytuvo solenoidas, BQ- zumeris, M-nuolatinės srovės variklis su apsauginiu diodu.

Stendo lizdai C0 ir C1 sujungiami su minusu.

4.2. Praktinis darbas Nr.1 Programos rašymas ir darbo simuliacija

DARBO TIKSLAS:

- ◆ sukurti naują projektą;
- ◆ rašyti programas valdikliui FBD kalba;
- ◆ simuliuoti programos darbą kompiuteryje.

DARBO PRIEMONĖS: CROUZET valdiklio stendas su USB adapteriu, kompiuteris su programa CLSM3-V2-3-AC5.

TEORINĖ DALIS



4.2 pav. Pagrindinis programos langas.

Pagrindiniame programos lange 4.2 pav. pasirenkama *File*, *New* ir pasirenkamas valdiklio tipas. Praktiniuose darbuose bus naudojamas *CD 12 24VDC SMART* valdiklis su keturiais diskretiniais ir keturiais analoginiais įėjimais (4 *DISCR* + 4(0-10V) ir 4 *RELAY* – keturiais reliniais išėjimais. Spaudžiama *Next*, *Next* ir pasirenkama programos kalba – *FBD*. Spaudžiama *Next* ir gaunamas programos rašymo langas.

Įėjimų/išėjimų paruošimas

Programos rašymo lange matomi 8 valdiklio įėjimai ir 4 išėjimai. Pasirenkamas programos rašymo režimas *E* (Edit). Meniu pasirenkama *IN/OUT* 4.3 pav.



4.3 pav. Įėjimų/išėjimų langas

Pele pertempiamas *DI* langelis į *I1* vietą. Tai reiškia, kad prie valdiklio įėjimo *I1* bus prijungtas diskretinis signalas. Du kart paspaudus pele ant langelio *I1* galima pasirinkti signalo šaltinį 4.4 pav.



4.4 pav. Signalų šaltinio pasirinkimas

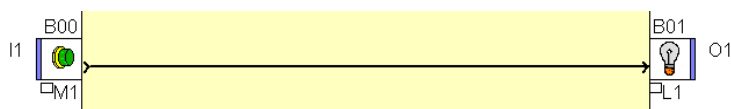
Du kart paspaudus pele pasirenkamas, pvz., mygtukas. Galima įrašyti komentarą (pvz. *M1*). Uždėjus varnelę langelyje *Display*, komentaras bus matomas programoje.

Analogiškai pasirenkamas valdomas objektas. Langelis *DO* 4.3 pav. pertempiamas ant išėjimo *O1*. Du kart paspaudus pele ant langelio *O1* galima pasirinkti valdomą objektą 4.5 pav. Pasirenkama, pvz., lemputė. Galima užrašyti komentarą *L1*.



4.5 pav. Valdomi objektai

Laikant nuspauštą kairį pelės klavišą sujungiamas mygtuko išėjimas su lemputės įėjimu. Programa parodyta 4.6 pav.



4.6 pav. Programa „lemputė“

Programos darbas simuliuojamas pasirinkus darbo režimą *S* (Simulation). Atsidariusiame lange spaudžiama *OK*. Kairiu pelės klavišu spaudžiamas mygtukas, raudona linija rodo, kad signalas patenka į išėjimą ir matyti šviečianti lemputė. Kitas pelės paspaudimas gesina lemputę.

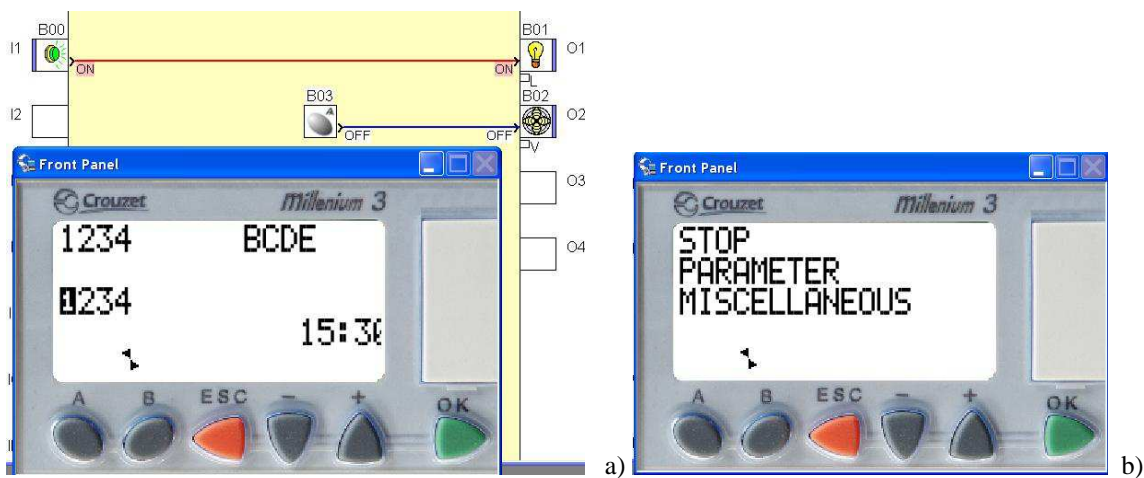
Pulto mygtukų panaudojimas

Mygtukus *A*, *B*, *ESC*, *+* ir minus, esančius valdiklyje galima naudoti valdymui dirbant valdikliui realiame laike. Tai lyg papildomi įėjimai. Pasirenkama *HMI/COM* 4.7 pav. ir pele pertempiamas mygtukas į programą.



4.7 pav. HMI meniu

4.8 pav., a parodyta programa su 2 mygtukais. Mygtukas B00 prijungtas prie valdiklio įėjimo I1. Mygtukas A (B03) yra valdiklio priekyje. Simuliacijos režime pele nuspaudus mygtuką B00 šviečia lemputė, o nuspaudus mygtuką A – įjungiamas ventiliatorius



4.8 pav. Programa su valdiklio pulto mygtuku ir valdiklio priekio vaizdas. a) indikacinis režimas, b) valdymo režimas

Programos išbandymas

Programos veikimą galima išbandyti be prijungto valdiklio simuliaciniame režime. Meniu pasirenkama *S*. Atsidariusiame lange spaudžiama *OK*. Linijos, kuriose yra signalas (+24 V) tampa raudonos. Mygtukas pele spaudžiamas du kartus. Vienas paspaudimas mygtuką sujungia, kitas – atjungia. Automatikos įtaisuose mygtukas nuspaudžiamas ne ilgiau kaip 1 sek.

Simuliaciniame režime kompiuterio ekrane galima gauti valdiklio priekio vaizdą. Meniu pasirenkama *WINDOW*, *FRONT PANEL*. Gaunamas valdiklio priekio vaizdas 4.8 pav. Galimi du variantai: indikacinis režimas 4.8 pav., a ir valdymo režimas 4.8 pav., b. Režimai perjungiami spaudžiant mygtuką *ESC*.

Indikaciniame režime viršutinėje eilutėje 1, 2, 3, 4, B, C, D, E rodomi įėjimai, o apatinėje – išėjimai. Jei skaičius ar raidė juodame fone, reiškia kad yra signalas (+24 V). Žemiau esantis ženklas rodo ar veikia programa. Tada ženklas sukasi.

Valdymo režime veikiant programai mirksi užrašas *STOP*. Du kart paspaudus mygtuką *OK* programa sustoja ir pasirodo užrašas *RUN*. Programą galima paleisti du kart paspaudus mygtuką *OK*.

Jei prijungtas valdiklis tada galima pasirinkti monitoringo režimą M. Jo metu sukuriama ryšys tarp programos funkcinių blokų ir realių automatikos komponentų.


EKSPERIMENTINĖ DALIS

1. Sujungiamas valdiklis su kompiuteriu per USB prievadą.
2. Prijungiamas +24 V maitinimas valdiklio stendui.
3. Paleidžiama programa ir sukuriamas naujas projektas. Pavadiname 6 pavardės raidės.
4. Nukopijuojama programa pagal 4.8 pav.
5. Programa įkeliama į valdiklį. Spaudžiamas meniu mygtukas 2 (4.2 pav.). Užrašomos pastabos.

6. Įjungiamas simuliacinis režimas. Išbandoma programa. Užrašomos pastabos.

7. Iškviečiamas į ekraną valdiklio priekio vaizdas (Front Panel). Išbandomi mygtukai ir užrašomos pastabos.

8. Spaudžiamas mygtukas M ir išbandomas režimas „Monitoringas“. Išbandomi mygtukai ekrane. Užrašomos pastabos.

9. Parašoma ir išbandoma mirksinčios kas sekundę lemputės programa, kurioje panaudotas šis elementas . Užrašomos pastabos.

ATASKAITA

- Darbo aprašymas
- Programos
- Išvados

4.3. Praktinis darbas Nr2. Programa su laikmačiu

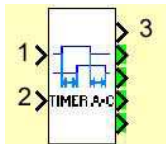
DARBO TIKSLAS:

- ◆ rašyti programas valdikliui FBD kalba, kuriose panaudojami laikmačiai;
- ◆ simuliuoti programos darbą kompiuteryje.

DARBO PRIEMONĖS: CROUZET valdiklio stendas su USB adapteriu, kompiuteris su programa CLSM3-V2-3-AC5.

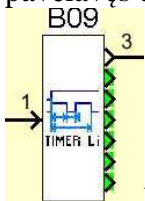
TEORINĖ DALIS

Laikmatis naudojamas formuojant įvairius laiko intervalus. Galimi 5 laikmačių variantai: A-C, trumpų impulsų generatorius (BW), Li/L, B/H, totalizer (universalus).



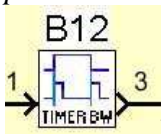
A-C laikmatis. 1 – valdymo įėjimas, 2 – reset įėjimas, 3 – išėjimas. Galimi trys laikmačio variantai.

1. Išėjimo signalas atsiranda pavėlavęs užduotą laiką nuo signalo atsiradimo įėjime. Jei dingsta įėjimo signalas išėjimo signalas neatsiranda.
2. Išėjimo signalas atsiranda iš karto, kai yra įėjime signalas. Išėjime signalas dingsta pavėlavęs užduotą laiką nuo įėjimo signalo dingimo. Kol yra įėjimo signalas išėjimo signalas nedingsta.
3. Išėjime signalas atsiranda pavėlavęs užduotą laiką nuo įėjimo signalo atsiradimo. Jis dingsta pavėlavęs užduotą laiką nuo įėjimo signalo dingimo.

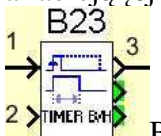


Li laikmatis - tai impulsų generatorius. Laikmatis išėjime 3 generuoja nustatytos trukmės impulsus kol įėjime 1 yra signalas. Dar galima užprogramuoti laiko intervalą, kurio metu bus generuojami impulsai. Taip pat galima užduoti impulsų skaičių. Impulsas atsiranda iš karto po įėjimo signalo.

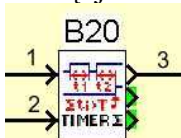
L – analogiškas laikmatis gaunamas pasirinkus *Function L*. Jame išėjimo signalas atsiranda po kurio laiko nuo įėjimo signalo atsiradimo.



BW laikmatis. Jo išėjime generuojamas trumpas impulsas nuo priekinio, galinio ar abiejų įėjimo impulso frontų. Tai pasirenkama programuojant.



B/H laikmatis. Tai nustatytos trukmės impulsų generatorius. Kai įėjime 1 atsiranda signalas išėjime generuojamas užprogramuotos trukmės impulsas nepriklausomai nuo įėjimo impulso trukmės. Kitas išėjimo signalas generuojamas įėjime atsiradus kitam impulsui. 2 – reset įėjimas.



Universalus laikmatis. Galima užprogramuoti įvairius darbo režimus:

Tt – laikmatį įjungia teigiamas impulsas 1 įėjime. Signalas išėjime 3 atsiranda iš karto ir išsijungia po nustatyto laiko. Iš naujo įjungti laikmatį galima padavus naują įėjimo signalą.

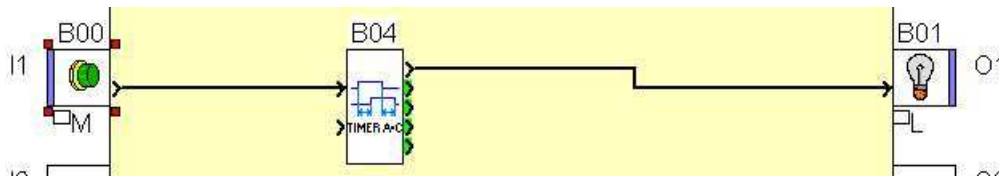
Laikmatį sustabdyti galima reset signalu.

At – laikmačio išėjimas įjungiamas praėjus užduotam laikui po to kai dingsta įėjimo signalas. Iš naujo įjungti laikmatį galima padavus signalą į reset įėjimą.

Ht - laikmačio išėjimas išjungiamas praėjus užduotam laikui po to kai dingsta įėjimo signalas. Iš naujo įjungti laikmatį galima padavus signalą į reset įėjimą.

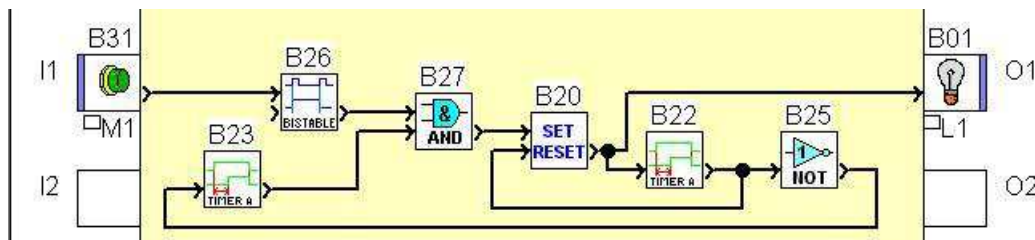
T - laikmačio išėjimas įjungiamas praėjus užduotam laikui po to kai atsiranda įėjimo signalas. Iš naujo įjungti laikmatį galima padavus signalą į reset įėjimą.

PASIRUOŠIMAS DARBUI (namuose)



4.9 pav. Laikmačio bandymo programa

1. Nukopijuojama 4.9 pav. parodyta programa. Pasirenkami funkcinio bloko B04 įvairūs variantai (laikmačiai), užprogramuojami jų parametrai ir išbandomas laikmačių veikimas simuliaciniame režime. Aprašomas laikmačių veikimas esant įvairiems nustatymams.



4.10 pav. Mirksinčios lempučių su reguliuojamu laiko intervalu programa

2. Aprašomi programoje 4.10 pav. panaudoti funkciniai blokai. Pvz., B26 – impulsinė relė. Turi dvi padėtis: įjungta ir išjungta. Vienu impulsu relė įjungiamą, kitu impulsu išjungiamą. Relė išjungti galima ir padavus impulsą į kitą relės įėjimą Reset.

Nukopijuojama programa ir išbandoma simuliaciniame režime.

EKSPERIMENTINĖ DALIS

SUJUNGIMAI. Stendo lizdai C1 ir C2 sujungiami su minusu. Vienas lempučių išvadas prijungiamas prie +24V. Antri išvada jungiami į valdiklio išėjimus Q1, Q2, Q3, Q4.

1. Sujungiamas valdiklis su kompiuteriu per USB prievadą specialia (balta, stačiakampe) jungtimi.
2. Prijungiamas +24 V maitinimas valdiklio stendui.
3. Pasirenkama Controller, Connection, Configure, Test ir patikrinama ar valdiklis susijungė su kompiuteriu
4. Pasirenkama Write to the controller arba mygtuką 2 (4,2 pav.), įkeliami 4.10 pav. programa į valdiklį ir išbandoma. Užrašomos pastabos.

5. Valdiklio ekranas perjungiamas į indikacinį režimą. Tam paspaudžiamas valdiklio priekyje mygtukas ESC. Ekране turi pasirodyti skaičiai 1234 ir raidės BCDE. Kai

paspaudžiamas mygtukas įvyks pakitimas valdiklio ekrane (apie skaičių atsiras stačiakampis). Užrašomos pastabos.

6. Modernizuojama įkelta programa. Įvedami valdymo mygtukai, esantys valdiklio priekyje. A mygtuku programa paleidžiama, o B – stabdoma. Užrašomos pastabos.

7. Įjungiamas monitoringo režimas. Programa stabdoma ir paleidžiama mygtukais kompiuterio ekrane. Užrašomos pastabos.

8. Parašoma mirksinčios lemputės programa panaudojant Li laikmatį. Vieną kartą spustelėjus mygtuką lemputė pradeda mirksėti, antrą kartą spustelėjus – nustoja. Programa išbandoma ir užrašomos pastabos.

9. Modernizuojama programa. Lemputė nustoja mirksėti, kai sumirksi „n“ kartų. Mirksėjimas įjungiamas valdiklio A mygtuku. Programa išbandoma ir užrašomos pastabos.

ATASKAITA

- Darbo aprašymas.
- Modernizuotos programos
- Išvados.

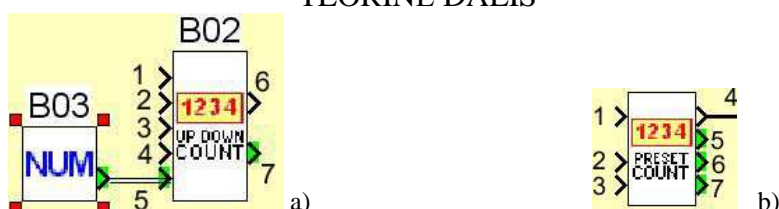
4.4. Praktinis darbas Nr.3. Programa su skaitikliu

DARBO TIKSLAS:

- ◆ rašyti programas valdikliui FBD kalba, kuriose panaudotas skaitiklis;
- ◆ naudoti programose ekraną;
- ◆ simuliuoti programos darbą kompiuteryje.

DARBO PRIEMONĖS: CROUZET valdiklio stendas su USB adapteriu, kompiuteris su programa CLSM3-V2-3-AC5.

TEORINĖ DALIS



4.9 pav. a) Sumuojantis/minusuojuantis skaitiklis, b) išankstinio nustatymo skaitiklis

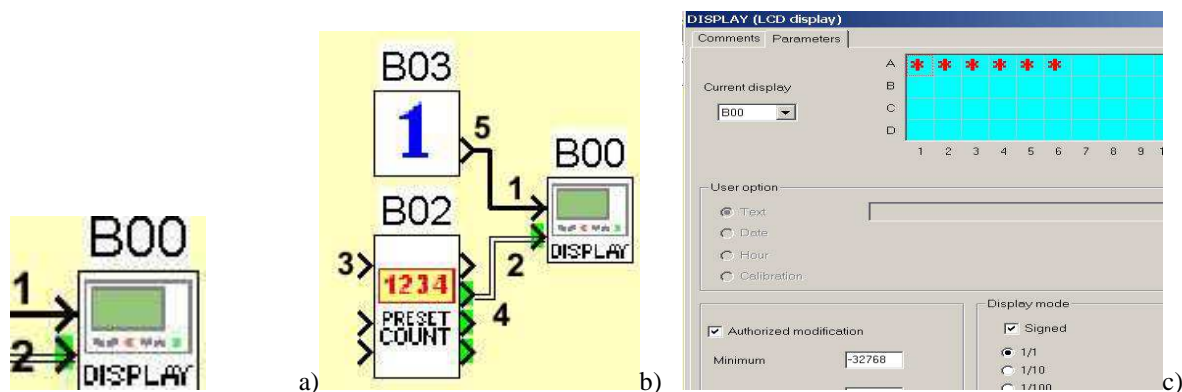
Sumuojantis/minusuojuantis skaitiklis parodytas 4.9 pav., a.. 1– sumuojantis įėjimas, 2– minusuojuantis įėjimas, 3 – reset įėjimas, 4 – užbaigiantis skaičiavimą įėjimas, 5 – užduotas skaičius iki kurio skaičiuojama (nuo -32768 iki +32767 vertės), 6 – skaitiklio išėjimas, 7 – suskaičiuotų impulsų skaičius. Prie šio išėjimo galima jungti ekrano įėjimą. Skaičius užduodamas panaudojant funkcinį bloką B03. Kai skaitiklis suskaičiuoja iki užduoto skaičiaus išėjime 6 atsiranda signalas.

Išankstinio nustatymo skaitiklis parodytas 4.9 pav., b. 1 – sumuojantis įėjimas, 2 – minusuojuantis įėjimas, 3 – inicializacijos įėjimas. Skaitiklis skaičiuoja kai šiame įėjime yra loginis 1. 4 – išėjimas. Kai skaitiklis suskaičiuoja užduotą impulsų skaičių jame atsiranda signalas. 5,6,7 – informaciniai išėjimai. Prie jų galima prijungti ekraną.

Ekranų panaudojimas

Ekranas gali rodyti tekstą arba skaičius. Galima naudoti iki 8 ekranų, tačiau vienu metu du displejai veikti negali, reikšmės rodys tik vienas. Ekranas aktyvuojamas paduodant signalą į 1 įėjimą 4.10 pav., a. Per 2-ą (signalinį) įėjimą paduodama skaičiaus reikšmė. Jei 1 įėjimas neprijungtas – ekranas aktyvuojamas. Ekranas išjungiamas kai šiame įėjime yra loginis 0.

Tipinis ekrano prijungimas parodytas 10 pav.,b. Komponentas B02 –skaitiklis. Kai į jo įėjimą 3 paduodami impulsai, tai išėjime 4 gaunamas skaičius. Komponentas B03 – loginio signalo formuotuvai. Simuliacijos režime ant jo paspaudus pelės klavišų išėjime pasikeičia signalas, atsiranda loginis 1 ir ekranas aktyvuojamas.

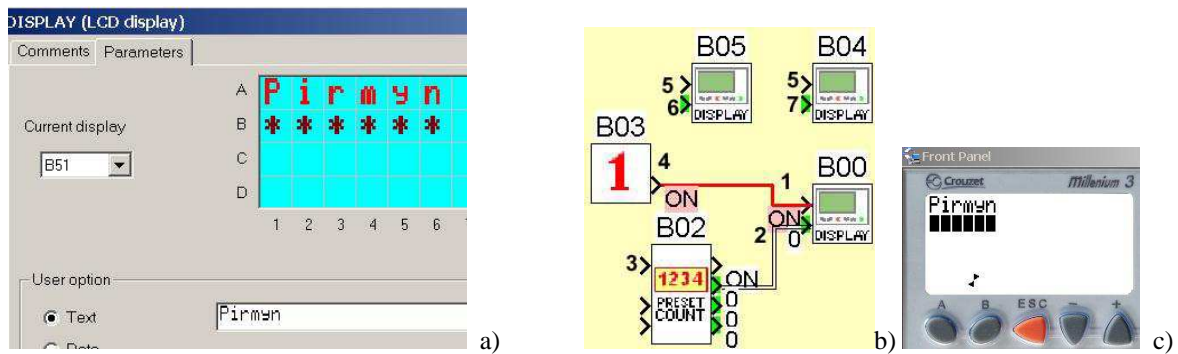


4.10 pav. Ekrano prijungimas ir nustatymas. 1 – aktyvavimo įėjimas, 2 – signalinis įėjimas

Pele paspaudus ant komponento B00 atsidaro nustatymų langas 10 pav., c. Jame tašku pažymėta, kad reikšmės bus sveiki skaičiai. Ekranas turi 4 eilutes. Pele galima pasirinkti kurioje eilutėje bus rodomas skaičius.

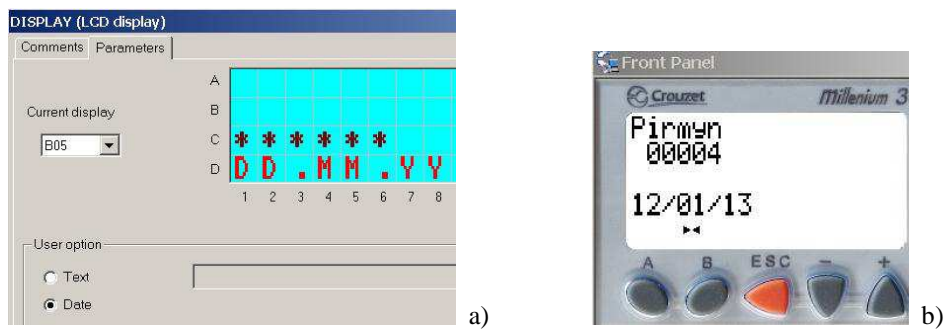
Teksto įterpimas

Teksto įterpimui reikia naudoti antrą ekraną (4.11 pav., b) B04. Jo signalinis įėjimas 7 turi būti neprijungtas. Paspaudus pele ant komponento B04 atsidaro nustatymų langas 4.11 pav., a. Jame pažymima *Text* ir eilutė A, į kurią bus rašomas tekstas. Po to rašomas tekstas *Pirmyn*.



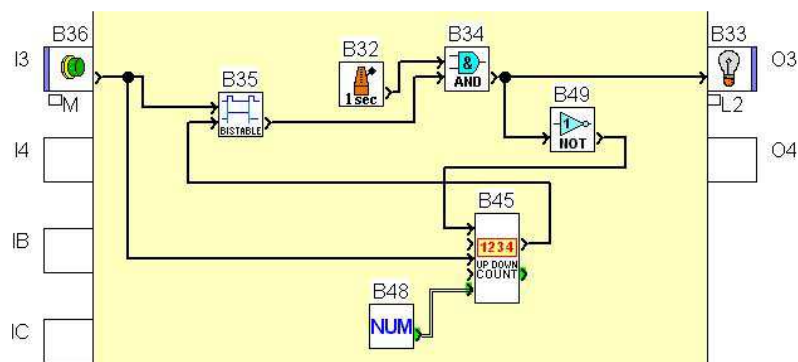
4.11 pav. Teksto rašymas

Simuliacijos metu valdiklio ekrane bus matyti tekstas ir antroje eilutėje – skaičius. Analogiškai galima tekstą įrašyti ir į kitą eilutę. Į ją pvz., įrašoma data.



4.12 pav. Datos indikacija

Tam reikalingas trečias ekranas B05 (4.11 pav., b). Atidaromas nustatymų langas 4.12 pav., a. Jame pažymima 4 eilutė ir pasirenkama *Date*. Simuliacijos metu aktyvuoti displėjai B04 ir B05 (nes neprijungti). Įjungus B03 loginis signalas aktyvuoja ir B00 ekraną, kuris rodo skaičių. Valdiklio ekrane gaunamas vaizdas parodytas 4.12 pav., b.



4.13 pav. Mirksinčios lempučių programa

PASIRUOŠIMAS DARBUI (namuose)

Nukopijuojama 4.13 pav. parodyta programa. Paaiškinamas jos veikimas ir aprašomi jos funkciniai blokai. Programa išbandoma simuliaciniame režime.

Parašoma 4-ių lempučių mirksėjimo programa ir išbandoma simuliaciniame režime.

EKSPERIMENTINĖ DALIS

1. Į valdiklį įkeliama namuose nukopijuota programa. Nustatomas mirksėjimų skaičius 8. Spustelėjus mygtuką M1, skaitiklis skaičiuoja iki 0. Įjungiamas monitoringo režimas ir išbandoma programa. Užrašomos pastabos.

2. Modernizuojama programa. Spustelėjus mygtuką M1, skaitiklis skaičiuoja nuo 0. Po 8-o lemputės mirksėjimo programa sustoja. Įjungiamas monitoringo režimas ir išbandoma programa. Užrašomos pastabos.

3. Modernizuojama programa. Prie skaitiklio prijungiamas ekranas, kuriame matyti užrašas (nurodo dėstytojas), impulsų skaičius ir data. Programa išbandoma ir užrašomos pastabos.

4. Į programą įvedamas avarinis mygtukas M2. Jį spustelėjus lemputės mirksėjimas nutrūksta. Spustelėjus mygtuką M1 mirksėjimas pradedamas nuo pradžių.

6. Į valdiklį įkeliama 4-ių mirksinčių lempučių programa, išbandoma ir užrašomos pastabos.

7. Modernizuojama programa. Vienu mygtuku mirksėjimas įjungiamas, kitu – išjungiamas.

ATASKAITA

- Darbo aprašymas
- Programos
- Išvados

4.5. Praktinis darbas Nr.4. Pneumatikos cilindrų pograminis valdymas

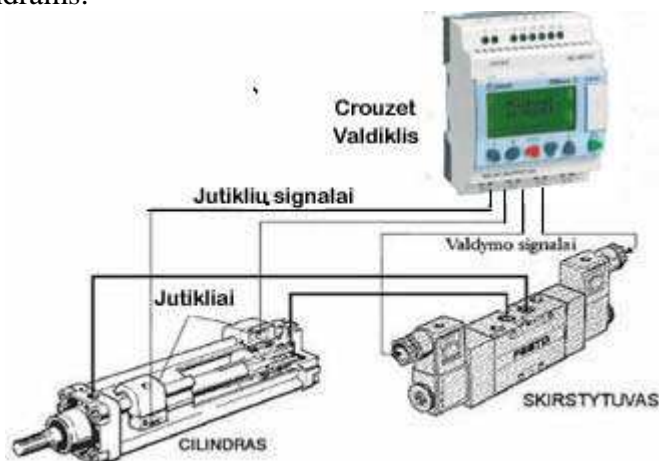
DARBO TIKSLAS:

- ♦ taikyti elektropneumatikos schemose valdiklius;
- ♦ gebėti jungti elektropneumatikos schemas;
- ♦ gebėti rašyti programas valdikliui FBD kalba.

DARBO PRIEMONĖS: CROUZET valdiklio stendas su USB adapteriu, elektropneumatikos stendas su dviem dvikrypčiais cilindrais ir elektropneumatiniiais monostabiliu ir bistabiliu skirstytuvais 5/2, 24 V lemputė, zumeris, mygtukas, 4 priartėjimo jutikliai (įvairūs), programa CLSM3-V2-3-AC5.

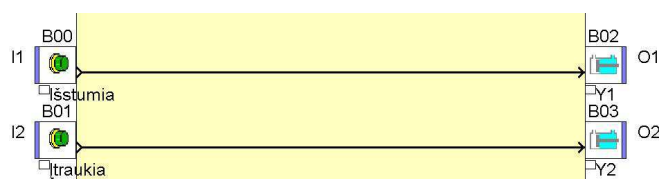
TEORINĖ DALIS

Šiuolaikinėse automatikos sistemose signalus priima, apdoroja ir nukreipia į vykdymo įtaisus programuojami loginiai valdikliai (PLV). Jie pakeičia relines schemas. Elektropneumatikos sistemos su valdikliu sandara parodyta 4.14 pav. Signalai iš jutiklių patenka į valdiklį. Ten jie apdorojami pagal programą. Valdiklis suformuoja valdymo signalus ir nukreipia juos į elektropneumatinius skirstytuvus. Skirstytuvai valdo oro tiekimą pneumatiniams cilindrams.



4.14 pav. Elektropneumatikos sistema su Crouzet valdikliu

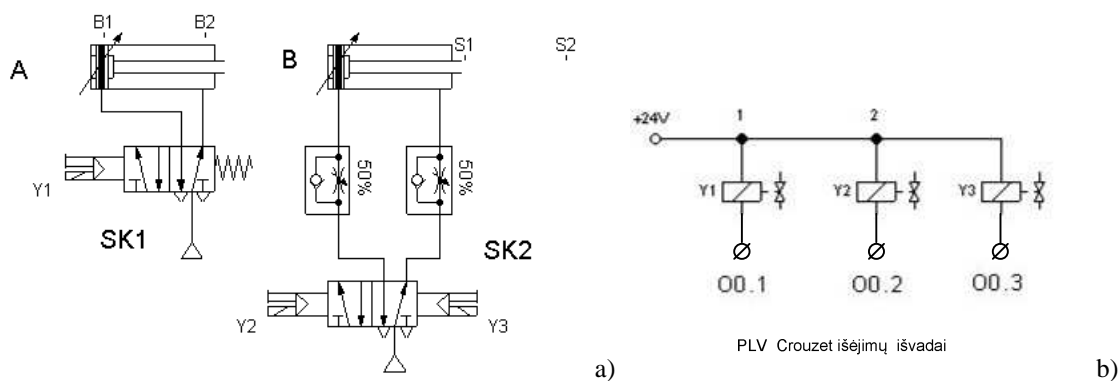
Prie PLV įėjimų jungiami valdymo mygtukai ir jutikliai, o prie išėjimų – indikaciniai elementai (lemputės), elektromechaniniai vykdymo įtaisai (varikliai, elektropneumatiniai skirstytuvai ir pan.).



4.15 pav. Programos FBD kalba pavyzdys

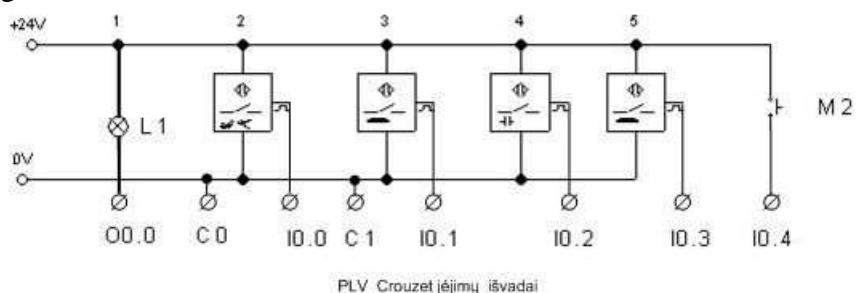
Programa rašoma FBD (funkcinių blok diagramų) kalba. Programos fragmentas parodytas 4.15 pav. Kairėje yra įėjimai, dešinėje – išėjimai. Tokia programa gali valdyti pneumatikos cilindrą B (žr. žemiau) su bistabiliu elektropneumatikos skirstytuvu. Spustelėjus vieną mygtuką cilindro kotas išstumiamas. Spustelėjus kitą mygtuką – įtraukiamas.

4.16 pav., a parodyta įtaiso su valdikliu elektropneumatikos schemos dalis. A ir B – cilindrai, S1, S2 ir B1, B2 jutikliai, SK1 – monostabilus skirstytuvas 5/2, SK2 – bistabilus skirstytuvas 5/2.



4.16 pav. a) elektropneumatinė schemos dalis, b) skirstytuvų prijungimas prie valdiklio

Sirstytuvų prijungimas prie valdiklio parodytas 4.16 pav., b. Vienas išvadas jungiamas prie +24 V. Kiti išvadais jungiami prie valdiklio išėjimų. Įjungus valdiklio išėjimą skirstytuvo išvadas sujungiamas su maitinimo šaltinio minusu.

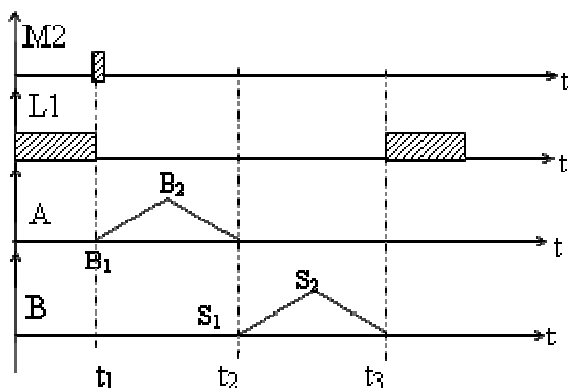


4.17 pav. Elektrinė schemos dalis

Elektrinėje schemos dalyje 4.17 pav. parodytas jutiklių, mygtuko ir lempučių prijungimas ir nurodyti PLV įėjimai ir išėjimai, prie kurių prijungti komponentai.

PASIRUOŠIMAS DARBUI (namuose)

1. Su FluidSim programa sudaroma elektropneumatikos schema pagal darbo diagramą 4.18 pav. M2 –mygtukas, S1, S2 ir B1, B2 – įvairūs priartėjimo jutikliai, L1 – lempučių. A, B – pneumatikos cilindrai



4.18 pav. Darbo diagrama

2. Parašoma programa valdikliui FBD kalba pagal darbo diagramą ir įrašoma į nešiojamą laikmeną. Paleidus programą šviečia lempučių L1. Spustelėjus mygtuką M2 lempučių gęsta ir pirmo cilindro kotas išstumiamas ir tuoj pat įtraukiamas. Po to antro cilindro kotas išstumiamas ir tuoj pat įtraukiamas. Lempučių vėl užsidega.

EKSPERIMENTINĖ DALIS

Sujungimai. Valdiklio stende gnybtai C0 ir C1 sujungiami su minusu ir išjunginama +10V įtampa. Elektropneumatikos skirstytuvų solenoidų ir lemputės vienas išvadas jungiamas prie +24V, o kitas prie valdiklio išėjimo. Jutiklių signalinis laidas jungiamas į valdiklio įėjimą. Jutiklių, maitinimo ir valdiklio stendo minusai sujungiami tarpusavyje ir su maitinimo šaltiniu.

1. Sujunginama elektropneumatikos schema pagal 4.16 pav. ir 4.17 pav.
2. Patikrinama ar teisingai prijungti jutikliai prie įėjimų. Tam paspaudžiamas valdiklio priekyje mygtukas ESC. Ekrane turi pasirodyti skaičiai 1234 ir raidės BCDE. Įjungiamas atitinkamas jutiklis. Kai suveiks jutiklis įvyks pakitimas valdiklio ekrane (apie skaičių atsiras stačiakampis).
3. Kompiuteris sujungiamas USB laidu su valdikliu. Ir paleidžiama programa CLSM3-V2-3-AC5.
4. Įkeliami namie parašyta programa į valdiklį ir išbandoma.
5. Modifikuojama programa taip, kad lemputė po mygtuko spūstelėjimo mirksėtų 5 sek., o tik po to pradėtų dirbti cilindrai. Išbandomas programos veikimas ir užrašomos pastabos.

6. Modifikuojama programa taip, kad cilindrai dirbtų nepertraukiamai po mygtuko spūstelėjimo. Išbandomas programos veikimas ir užrašomos pastabos.

ATASKAITA

- Darbo aprašymas.
- Panaudotų komponentų apibūdinimas.
- Modifikuotų programų fragmentai ir jų veikimo aprašymas
- Išvados

INFORMACIJOS ŠALTINIAI

1. Pleskas, Stanislovas. *PIC16F84 mikrovaldiklis*. Mokomoji knyga. Vilnius:Vilniaus kolegija, 2006.102p. ISBN 9955-519-65-7
2. Baškys, A. *Mikrokompiuteriai*. Laboratorinių darbų užduotys ir metodikos nurodymai. Vilnius: „Technika“, 2006. 164p. Prieiga per internetą: http://leidykla.vgtu.lt/docs/el/A.Baskys_%20Mikrokompiuteriai.pdf.
3. Matic, N. PIC microcontrollers. 2003. 178. Prieiga per internetą: <http://www.mikroelektronika.co.yu/english/product/books/PICbook/picbook>
4. Arduino visiems. Prieiga per internetą <http://www.darysiupats.lt/knygos/knyga-arduino-visiems.html>
5. Bian W. Evans. Arduino programuotojo bloknotas. Prieiga per internetą: <http://www.darysiupats.lt/knyga-arduino-programuotojo-bloknotas-4.html> .

Stanislovas Pleskas

Valdikliai
Praktiniai darbai

2012-02-12. 8,37 sp. l. Tiražas 100 egz.
Spausdino UAB "Artika", Vaižganto g. 9, LT-28214 Utena