KAUNO TECHNOLOGIJOS UNIVERSITETAS

VYTAUTAS DEKSNYS, VACLOVAS JASTRAMSKAS

ĮTERPTINĖS SISTEMOS

1 dalis

8 skilčių sistemos

Kaunas Technologija 2000

Pratarmė

Viena iš būdingiausių šiandieninės mokslinės-techninės pažangos ypatybių yra platus mikroelektronikos gaminių naudojimas įvairiose ūkio srityse. Jau daugelį metų JAV ekspertų sudaromame "kritinių technologijų", apimančių visas gamybos, tyrimo ir kūrybos kryptis ir turinčių įtaką šalies ekonominiam ir gynybiniam potencialui sąraše mikroelektronikos ir informacinės technologijos tradiciškai užima pirmą vietą.

Mikroprocesoriuose - ypač sudėtinguose mikroelektronikos įtaisuose - įdiegti pažangiausi mokslo ir inžinerinės minties laimėjiimai. Jie naudojami šiandien daugelyje žmogaus veiklos sričių - nuo kosminių tyrimų, gamybinės veiklos iki medicinos ir buities. Tai personaliniai kompiuteriai, automatizuotos informacijos rinkimo ir apdorojimo sistemos, valdymo ir kontrolės sistemos, telekomunikacijos ir kt. Mikroprocesorių adaptacija, sprendžiant konkretų uždavinį, atliekama kuriant programines priemones. Aparatinė adaptacija daugeliu atvejų vyksta projektuojant reikalingas sąsajos schemas, jungiamas prie gana reguliarios mikroprocesorinės sistemos struktūros.

Mikroprocesorinėje technikoje išsiskyrė savarankiška didelių integrinių grandynų klasė – vieno kristalo mikrokompiuteriai (mikrovaldikliai), skirti įvairios paskirties įrenginių "intelektualizacijai". Jų architektūra – tai mikroprocesorių ir mikroprocesorinių sistemų architektūros evoliucijos rezultatas, užtikrinantis žymiai mažesnę sistemos aparatinės dalies apimtį bei kainą.

Knygoje aptariami aštuonių skilčių mikroprocesoriai ir mikrovaldikliai, jų pagrindu sukurtos įterptinės sistemos, projektavimo metodika, sprendžiami sąsajos ir atsparumo trikdžiams klausimai. Dėstomoji medžiaga remiasi populiariais pramonės išleidžiamais Neumano ir Harvardo struktūrų mikroprocesorių pavyzdžiais.

Vadovėlis skirtas telekomunikacijų ir elektronikos fakulteto studentams. Juo gali naudotis ir gretimų specialybių studentai, inžinieriai ir visi besidomintys įterptinėmis mikroprocesorinėmis sistemomis, jų projektavimu ir taikymu.

Antroji knygos dalis skiriama 16-os bitų, trečioji - 32 bitų mikroprocesoriams ir mikrovaldikliams.

Autoriai dėkoja recenzentams docentams V. Dzenkauskui ir P. Kanapeckui bei prof. R . Žilinskui, suteikusiems vertingų pastabų ir pasiūlymų, taip pat studentams A.Mačiui ir P.Kaškonui, padėjusiems sumaketuoti tekstą.

1. Mikroprocesorinių įtaisų projektavimo ir naudojimo ypatumai

1.1. Pagrindinės sąvokos ir apibrėžimai

Nagrinėjant mikroprocesorinių sistemų kūrimo ir funkcionavimo klausimus, vartojama gana daug įvairių sąvokų ir apibrėžimų . Paminėsime pagrindinius.

Mikroprocesoriumi (MP) vadiname programiniu būdu valdomą skaitmeninį informacijos apdorojimo įtaisą, sudarytą iš vienos ar kelių didelių integrinių grandynų (DIG). MP, kaip ir kiekvieną procesorių, sudaro du įtaisai: operacinis įtaisas, atliekantis operacijas su operandais, ir valdantysis įtaisas, kuris suformuoja valdančiuosius signalus operaciniam įtaisui pagal iš atminties nuskaitytą komandos kodą [1].

Mikroprocesorinę sistemą (MPS) sudaro MP, atmintis bei įvesties ir išvesties įrenginiai. Ji skirta skaitmeninio informacijos apdorojimo algoritmų programinei realizacijai. Pagrindinis informacijos apdorojimo elementas čia yra MP. DIG skaičius, sudarantis MPS, svyruoja nuo vieno iki keleto dešimčių .

Galimi tokie MPS funkcinių elementų fizinio realizavimo atvejai:

- kiekvieną funkcinį elementą atitinka atskiras integrinis grandynas (IG), realizuojantis jo funkcijas;
- vieno elemento funkcijas realizuoja keletas IG;
- keli (ribiniu atveju visi) funkciniai elementai realizuoti viename DIG.

Paprastai laikomės tokios terminologijos. Jeigu visi MPS funkciniai elementai yra viename DIG, jis vadinamas vienkristaliu mikrokompiuteriu, arba mikrovaldikliu (MV) (microcontroller). Jeigu mikrovaldiklio (MV) savųjų resursų nepakanka, jie išplečiami išorinių įrenginių (atminties ir išorinių sąsajų) sąskaita. Tada sakoma, kad mikrovaldiklis veikia mikroprocesoriaus režimu.

Procesorinis elementas gali būti sudarytas iš kelių DIG, kurie atlieka procesoriaus valdančiojo bei operacinio įtaiso funkcijas. Tokio procesoriaus skilčių skaičius priklauso nuo sujungtų į bendrą modulį operacinio įtaiso funkcijas atliekančių DIG kiekio,o toks MP vadinamas daugelio kristalų MP.

Mikrokompiuteris - MPS rūšis - išbaigtas savarankiškas įtaisas, turintis priemones ryšiui su išoriniais įtaisais, t.y. sujungtus į bendrą konstrukciją valdymo pultą, maitinimo šaltinį, indikatorių ir t.t. Geriausias tokios MPS pavyzdys - personalinis kompiuteris (PC) .

Įterpiamas (embedded) mikrokompiuteris neturi individualaus valdymo pulto, maitinimo šaltinio, dekoratyvinio apiforminimo ir skirtas konstruktyviai

įterpti (įmontuoti) į skaičiavimo, komunikacijų ar valdymo sistemą. Jis dažniausiai būna sumontuotas vienoje (vienplokštis įterpiamas mikrokompiuteris) arba keliose plokštėse.

MV turi išvystytas ryšio priemones su valdymo objektais (informacijai apie objekto būseną priimti bei valdantiesiems signalams formuoti), tačiau dažniausiai neturi ryšio su operatoriumi priemonių (kartais tam tikslui numatomos nuoseklių mainų sąsajos).

Valdikliais dažnai vadinami ne tik valdantieji mikrokompiuteriai, bet ir MPS sąsajos, ryšio su periferiniais įtaisais priemonės, pvz., klaviatūros valdikliai, tiesioginių mainų valdikliai, decentralizuoto valdymo valdikliai ir pan.

MPS architektūra yra abstrakti kompiuterio vaizdavimo forma, aprašanti struktūrinę, schemotechninę ir loginę jo organizaciją. Architektūros sąvoka yra kompleksinė, apjungianti:

- struktūrą;
- ryšio su struktūrinės schemos mazgais metodus ir priemones;
- sąsajos skilčių skaičių ir organizavimo principus;
- registrų kiekį ir darbo su jais būdus;
- duomenų formatą;
- komandų sąrašą ir adresavimo metodus;
- pertraukčių aptarnavimo būdus.

MPS turi atskiroms atchitektūroms (Harvardo ir Neumano) bendrų ir individualių bruožų. Individualumas reiškiasi programiniame modelyje, t.y. toje MPS dalyje, kuri gali būti programuojama.

1.1. Elektroninių įtaisų aparatinė ir programinė realizacija

Kuriant šiuolaikinę elektroninę aparatūrą (EA), galimi trys skaitmeninių įtaisų darbo algoritmų realizavimo būdai : aparatinis, programinis ir mišrusis.

Pirmuoju atveju naudojami specialūs funkciniai blokai. Šių blokų ir ryšių tarp jų visuma nustato realizuojamą algoritmą. Tokiu būdu sukurtas įtaisas yra specializuotas - keičiant jo darbo algoritmą, būtina keisti įtaiso struktūrą. Tokie įtaisai vadinami standžios logikos įtaisais.

Realizuojant įtaiso darbo algoritmą programiniu būdu, pasirenkami universalūs operaciniai blokai - procesoriai, kurie nuosekliai, pagal vartotojo sudarytą programą, atlieka operacijas.

MPS kaip tik ir naudojamas programinis algoritmų ir įtaisų realizavimo būdas. Natūralu, kad plačiai taikomas ir mišrusis – aparatinis-programinis būdas. Praktiškai susiduriama su kai kurių aparatinių funkcijų programiniu

atlikimu, arba programinių funkcijų aparatiniu atlikimu. Pirmame variante standžiosios logikos įtaisas naudojamas kai kurioms aparatinėms funkcijoms atlikti MPS. Antrojo varianto įtaise, veikiančiame pagal nustatytą programą, naudojami standžios logikos elementai kai kurioms operacijoms atlikti. Šio varianto plačiai taikomas pavyzdys yra aparatinis daugiklis, naudojamas daugybos operacijai atlikti.

Pagrindinis programinio algoritmų realizavimo būdo trūkumas yra mažas, palyginti su standžiosios logikos įtaisais, veikimo greitis. Ši savybė sietina su nuoseklia atskirų operacijų atlikimo forma lanksčiosios (programuojamos) logikos įtaisuose. Viena iš priemonių tokių įtaisų veikimo greičiui padidinti - operacijas vykdyti ne nuosekliai, o lygiagrečiai.

Įtaisų programinio atlikimo būdas turi nemaža privalumų. Visų pirma - universali programos sudarymo galimybė. Sprendžiant įvairius praktinius uždavinius, tereikia keisti tik programos algoritmą ir jos tekstą. Keičiasi tradiciniai sistemos projektavimo metodai - schemos projektavimas pakeičiamas jos darbo modeliavimu universaliame kompiuteryje, dažniausiai PC.

Pasirinkus MP ir mikroprocesorinius DIG, galima sukurti nedidelių gabaritų ir masės, nebrangiai kainuojančius bet didelio patikimumo kontroliuojančiuosius, valdančiuosius ir apdorojančiuosius įtaisus bei sistemas, tiesiogiai įterpiamus į prietaisus, mašinas ir įrenginius .

Šiuo metu MP ir jiems gimiringi DIG pasiekė tokį išsivystymo lygį, kad MP naudojimo sferos yra tiesiog neribotos. Kiekvienoje žmogaus veiklos srityje praverčia potencialios valdančiųjų ir signalus apdorojančiųjų MPS galimybės. Plačiausiai šiuo metu MPS paplitusios valdymo sistemose, matavimo prietaisuose ir sistemose, telekomunikacijų sistemose, kompiuterių išorinių įtaisų valdikliuose, buitinėje, automobilinėje technikoje ir pan. Ypač neribotos MPS taikymo galimybės telekomunikacijų technikoje perdavimo ir derinimo valdymui, belaidžio ryšio įtaisų (wireless), elektroninių automatinių telefono stočių (ATS) parametrų kontrolei, skaitmeninio laidinio ir mobiliojo ryšio realizacijai, kompiuterinių tinklų valdymui ir t.t.

1.1. MPS aparatinės ir programinės priemonės

MPS, kaip ir bet kurią kitą skaičiavimo sistemą, sudaro dviejų tipų priemonės – aparatinės ir programinės. Aparatinės priemonės - tai visi fizikiniai įtaisai - DIG, moduliai, jungiamosios linijos. MPS programinės priemonės - tai visos programos, kurios naudojamos pagrindinių funkcijų vykdymui, darbo kontrolei, testavimui ir derinimui. Nekeičiant MPS aparatinių

priemonių, o keičiant tik programines, galima objektą perderinti, kad jis spręstų naujus uždavinius ir vykdytų naujas funkcijas.

Aparatinės MPS priemonės saugo programos turinį, formuoja apdorojimui pateikiamus duomenis, vykdo programas, atlieka duomenų apdorojimą bei palaiko ryšį su objektu. Programinės priemonės, valdydamos aparatūrą ir objektą, realizuoja MPS ir objekto valdymo programines funkcijas.

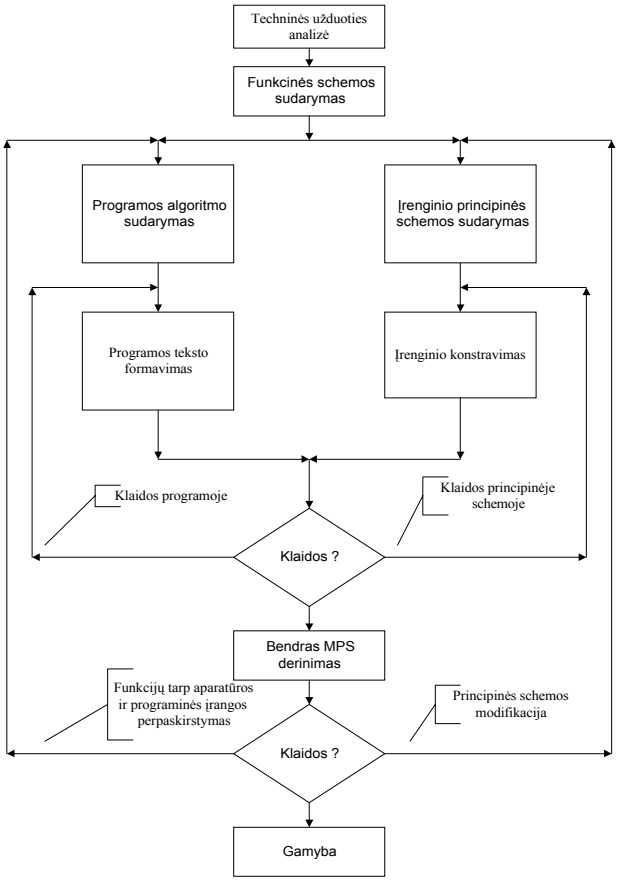
Taigi MPS sudaryta iš dviejų dalių - aparatūros (hardware) ir programos (software). Ir aparatūra, ir programa gali turėti modulinę struktūrą. Programose tokia strukūra labai patogi, nes atskirus modulius (blokus) galima projektuoti kartu keletui programuotojų, projektavimo pabaigoje susiejant juos į vientisą programinį modulį.

MPS aparatinės priemonės realizuojamos šiuolaikinės mikroschemotechnikos gaminiais - MP, MV, operatyviosios atminties (OA) ir pastoviosios atminties (PA) DIG, programuojamomis loginėmis matricomis (PLM), loginėmis schemomis, įvesties ir išvesties (I/O) formuotuvais ir t.t. Programos saugotojai gali būti - PA, OA, išsaugančios informaciją atjungus maitinimą (NVSRAM - nonvolatile static random access memory), magnetinės juostos, lankstūs ir kieti magnetiniai diskai bei kiti informacijos kaupikliai. Dažniausiai programų kaupikliais būna įvairių tipų PA IG (ROM - read only memory), programuojamos gamybos metu (ROM), vieną kartą vartotojų programuojamos (PROM - programmable read only memory), daug kartų vartotojo programuojamos, seną informaciją ištrinant ultravioletiniais spinduliais (EPROM - electric programmable read only memory) arba elektriniais signalais (EEPROM - erasable electric programmable read only memory). Pastaruoju metu labai populiarios FLASH tipo PA, kuriose panaudotos teigiamiausios EPROM ir EEPROM atminčių savybės, t.y. gaunamas didžiausias intergracijos laipsnis (atminties elementų skaičius ploto vienete) ir galimybė programuoti atminties turinį neišėmus DIG iš schemos (in-circuit reprogramming) [15,16]. PC programos vykdomos iš RAM (random access memory) ir DRAM (dynamic random access memory) tipų OA.

1.1. MPS projektavimo ciklas

Pagrindiniai MPS projektavimo etapai pateikti 1 pav. Pirmame projektavimo etape formuojami reikalavimai ir jų pagrindu sudaroma funkcinė MPS specifikacija. Šie reikalavimai derinami su vartotoju (užsakovu ar pirkėju), t.y. išsiaiškinama, ko jis nori iš būsimosios MPS. Kitas būdas reikalavimams formuoti - būsimų gaminių asortimento planavimas, kai vartotojų reikalavimai gali būti nustatyti pasitelkus rinkos analizės metodus. Čia iš bendros problemos, dažnai pateiktos abstrakčiai ir nepriklausomai nuo

jos realizavimo technikos, suformuojami konkretūs techniniai reikalavimai ir apribojimai projektuojamai MPS. Tai visų pirma gaminio paskirtis, atliekamų funkcijų sąrašas, veikimo sparta arba MPS funkcijų vykdymo laikas, tikslumas, patikimumas, gabaritai, masė, energetiniai parametrai ir pan. Jų visuma turi būti orientuota į pateiktos problemos sprendimą.



1 pav. MPS projektavimo etapai

Funkcinė specifikacija nustato funkcijas, kurias turi atlikti MPS, kad patenkintų vartotojo keliamus reikalavimus. Ji taip pat užtikrina sąsają tarp sistemos ir aplinkos. Specifikaciją sudaro du komponentai:

- funkcijų, atliekamų MPS, sąrašas;
- sąsajos tarp sistemos ir aplinkos aprašymas.

Ją tikslinga suskirstyti į tris kategorijas: įėjimus, išėjimus ir funkcijas. Tai patogu pateikti lentelių arba sąrašų pavidalu. Į funkcinę specifikaciją gali būti įtrauktos ir atskirų sistemos dalių schemos. Sudėtingesnes sistemas kartais tikslinga suskaidyti į mažesnės apimties posistemius ir nustatyti funkcines specifikacijas kiekvienam posistemiui atskirai. Vietoj MPS atliekamų funkcijų aprašymo galima pateikti jos realizavimo aprašymą. Toks aprašymas leistinas, jeigu reikia parodyti, kaip sistema veikia. Kai tik funkcinė specifikacija nustatyta, ji kartu su vartotojo reikalavimais tampa MPS projektavimo pagrindu. Kitame projektavimo etape sudaroma bendra MPS struktūra ir funkcionavimo schema, nustatomos ją sudarančios dalys ir jų tarpusavio sąveika. Prieš pradedant projektuoti MPS programines ir aparatines priemones, būtina nustatyti funkcijas, kurias geriau atlikti su aparatūra, ir funkcijas, atliekamas programiniu būdu (nuodugniai įvertinant vykdomų funkcijų aparatinio ir programinio realizavimo privalumus ir trūkumus). Jeigu nepablogėja projektuojamos sistemos charakteristikos, būtina stengtis kuo daugiau funkcijų realizuoti programiniu būdu. Būtų idealu, jeigu nuo šio momento projekto struktūra nepasikeistų, tačiau praktiškai tai pasiekti pavyksta retai.

Atliekant detalų aparatūros ir programos projektavimą, dažnai tenka konstatuoti, kad kai kurias aparatūros funkcijas geriau atlikti su programa, ir atvirkščiai. Tokiu atveju vėlesniuose projektavimo etapuose gali būti modifikuotas išankstinis projektinis sprendimas.

Priėmus kompromisinį MPS vykdomų funkcijų realizavimo sprendimą, lygiagrečiai projektuojama aparatūra ir programa. Po to vyksta autonominis jų derinimas ir testavimas. Baigus autonominį derinimą, atliekamas sujungimas ir bendras derinimas. Išryškėjusios klaidos taisomos grįžtant į pirminius projektavimo etapus.

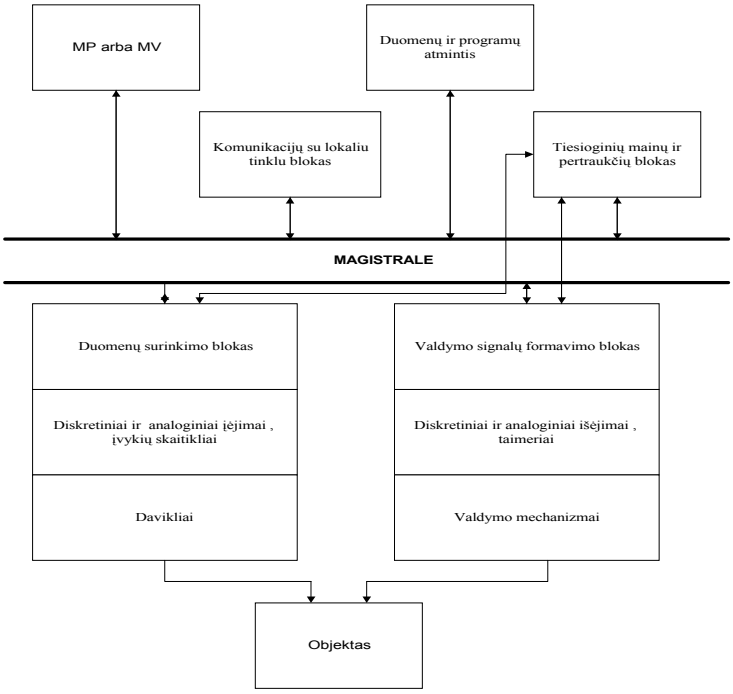
1.1. Kontroliniai klausimai

1. Pateikite MP, MV ir MPS apibrėžimus.
2. Kas yra standžiosios ir lanksčiosios logikos įtaisai? Kokie jų realizavimo trūkumai ir privalumai?
3. Apibūdinkite aparatines ir programines MPS priemones.
3. Išvardykite ir apibūdinkite pagrindinius MPS projektavimo etapus.
3. Kas yra programinis modelis?
3. Apibrėžkite kompiuterio architektūros sąvoką?
3. Kokie yra MPS projektavimo etapai?

1. MPS aparatinių priemonių projektavimas

1.1. MPS struktūrų ypatumai

Apibendrinta MPS struktūrinė schema pateikta 2 pav. Ją sudaro prie bendros magistralės



2 pav. Apibendrinta MPS struktūrinė schema

prijungti MP arba MV, duomenų ir programų atminties, tiesioginių mainų ir pertraukčių, komunikacijų su lokaliniu skaitmeniniu tinklu blokai, taip pat ryšio su objektu sąsajos elementai. Informacija apie objekto būseną gaunama iš daviklių (sensorių), kurie prie MPS prijungiami per matavimo keitiklius, keičiančius matuojamą fizikinį dydį dažniausiai į įtampą, srovę, impulsų trukmę, lygiagretų arba nuoseklų skaitmeninį kodą arba impulsų pasikartojimo dažnį. Į objektą siunčiami valdymo signalai dažniausiai būna įtampos, srovės, impulso pločio, impulsų skaičiaus, koduotos sekos arba jų pasikartojimo dažnio pavidalo. Jei MPS panaudotas MV, sudedamųjų dalių skaičius sumažėja, nes

dažnai į MV DIG sudėtį būna įkomponuota programų ir duomenų atmintis, pertraukčių blokas, komunikacijų su lokaliniu tinklu blokas, įvykių skaitikliai, taimeriai, analoginiai-skaitmeniniai (ASK) ir skaitmeniniai–analoginiai keitikliai (SAK).

2 pav. pateikta MPS struktūra yra gana reguliari. Dažnausiai atskiro objekto MPS funkcinę individualybę nustato MPS darbo programa. Konkrečiam objektui specialiai parenkamas MP arba MV tipas, OA ir PA DIG tipai ir kiekis, suprojektuojamos sąsajos su objektu schemos.

Reikia atkreipti dėmesį į magistralinį-modulinį MPS organizavimo principą. Atskiri sistemos blokai ir įtaisai yra funkcionaliai užbaigti moduliai su savomis vidinėmis valdymo schemomis. Visi moduliai sujungti į bendrą magistralę, susidedančią iš įvairios paskirties linijų. Šiomis linijomis atskirų sistemų moduliai atlieka tarpusavio mainus. Esant tokiai organizacijai sistema yra "atvira". Ją galima praplėsti prijungus naujus modulius, o modernizuoti - pakeitus atskirus modulius kitais. Nors sistemos moduliai informacinius mainus atlieka per bendrą magistralę, tačiau kiekvienu konkrečiu laiko momentu galimi mainai tik tarp dviejų modulių. Tokiu būdu mainai vyksta suteikiant magistralę atskiriems sistemos moduliams. Mainams vadovauja sistemos procesorius arba tiesioginių mainų valdiklis (direct memory access (DMA) controller). MPS našumas gali būti padidintas naudojant ne vieną, o kelis procesorius (daugiaprocesorinė sistema). Tokioje sistemoje mainams vadovauja ir konfliktines situacijas sprendžia arba svarbiausiasis procesorius (host (master) processor), arba specialus modulis - sistemos arbitras. Didelio našumo MPS (ypač daugiaprocesorinėse) būna ne viena bendra, o keletas magistralių, dažniausiai duomenų mainams su periferiniais įrenginiais, kitais procesoriais (slave processors) ar atmintimi. Tokios struktūros pavyzdys galėtų būti firmos Analog Devices signalų procesorių SHARC (Super Harvard Architecture) šeima [20].

1.1. Procesoriaus modulis

MP arba MV yra pagrindinė sudedamoji MPS dalis. Būtent jo savybės lemia visos MPS darbo našumą. Pastarąjį įvertinti nėra labai paprasta. Ne visada procesoriaus sinchronizacijos dažnio dydis lemia našumą (ypač, kai naudojamas atliekamų operacijų lygiagretinimas procesoriuose su vidiniais konvejeriais). Čia tikslinga paminėti konvejerizavimo principą. Jame komandos vykdymas yra suskirstomas į keletą etapų. Jų kiekis gali būti skirtingas. Pavyzdžiui MP i486 naudojamas penkių pakopų konvejeris, o jame yra tokie komandos vykdymo etapai:

- komandos kodo nuskaitymas iš atminties,
- komandos dešifravimas,
- operandų adreso formavimas,

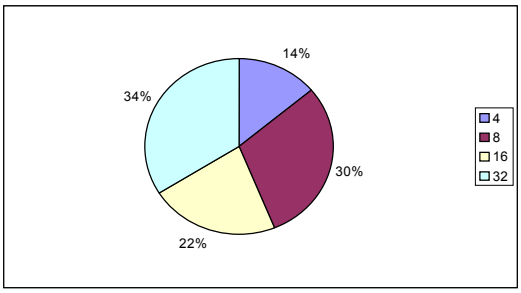
- komandos vykdymas,
- rezultato įrašymas.

Šiuo atveju lygiagrečiai vyksta keli etapai. Pavyzdžiui, nuskaičius komandos kodą iš atminties, jis perduodamas dešifravimui ir nuskaitymo įrenginys tampa laisvas. Taigi jis gali pradėti nuskaityti sekančios komandos kodą. Tokiu būdu lygiagrečiai vykdomos kelios komandos, tik jos yra skirtinguose įvykdymo etapuose. Toks procesoriaus darbo būdas labai padidina skaičiavimo našumą.

Jeigu MP turi vieną konvejerį, tai jo architektūra vadinama skaliarine jeigu kelis - superskaliarine (tokią architektūrą turi PENTIUM tipo MP).

Dėl minėtų priežasčių, našumui nustatyti pasirenkamos specialios testinės programos. Jos leidžia nustatyti atliekamų operacijų skaičių per sekundę. Kaip našumo matavimo vienetai dažniausiai naudojami MIPS (milion integer per second – milijonai aritmetinių operacijų su sveikais skaičiais per sekundę) arba MFLOPS (milion floating point operations per second - milijonai aritmetinių operacijų su slankiojo kablelio skaičiais per sekundę). Aišku, kad MP našumas labai priklauso nuo duomenų magistralės skilčių skaičiaus. Esant tam pačiam skaičiavimo tikslumui, didesnio duomenų skilčių skaičiaus MP dirbs sparčiau.

Projektuojant įvairios paskirties elektroninę aparatūrą, dažniausiai kuriamos įterptinės MPS (embedded microprocessor system). Tam dažnai naudojami 4, 8, 16 ir 32 skilčių universalūs MP ir MV. Jų pasiskirstymas rinkoje pateikiamas 3 pav.



3 pav. MP ir MV pasiskirstymas rinkoje pagal duomenų skilčių skaičių

Plačiausiai paplitę 8 ir 32 skilčių MP ir M. Įterptinėse sistemose plačiai naudojami 32 skilčių skaitmeniniai signalų procesoriai SSP (DSP – Digital Signal Processor), toliau plečiantys savo įtaką rinkoje. Gana gyvybinga ir 8 skilčių vartotojų rinka. Sparčiai vystosi pramoniniu standartu tapusi MCS51 šeima ir naujos, ypač mažų periferinių sąsajų procesorių PSP (PIC – Periferal Interface Computer) [22] ir sensorių signalų procesorių SSP (Sensor Signal Procecessor), šeimos. 16-os bitų MP sritis pildosi besivystančia matavimo

signalų procesorių MSP (Metering Signal Processor) šeima [14]. Įterptinėse sistemose plačiai taikomi 16-os skilčių universalių MP (labai dažnai x86 [16,21] šeimos) deriniai su išplėstu programiniu modeliu (80186 ir 80188). Dažnai gamintojai pateikia MP, suderinamus su ankstesniais modeliais, bet padidinto našumo. Iš tokių galima paminėti Dallas Semiconductor firmos išleidžiamus MV DS80C320 – C550, kurie komandų sąrašu visiškai suderinami su MCS51, bet apie 3-4 kartus našesni [17]. Firma Philips yra išleidusi 16-os skilčių 8052 variantą P51XA30. Tai labai patogu vartotojui, kuris minimaliomis sąnaudomis gali padidinti MPS našumą [23].

Šioje mokymo priemonėje nagrinėsime 8-nių skilčių MP, MV ir PSP aparatines ir programines priemones. Kaip pavyzdžiu imsime populiariausias universalių MP , MV ir PSP šeimas.

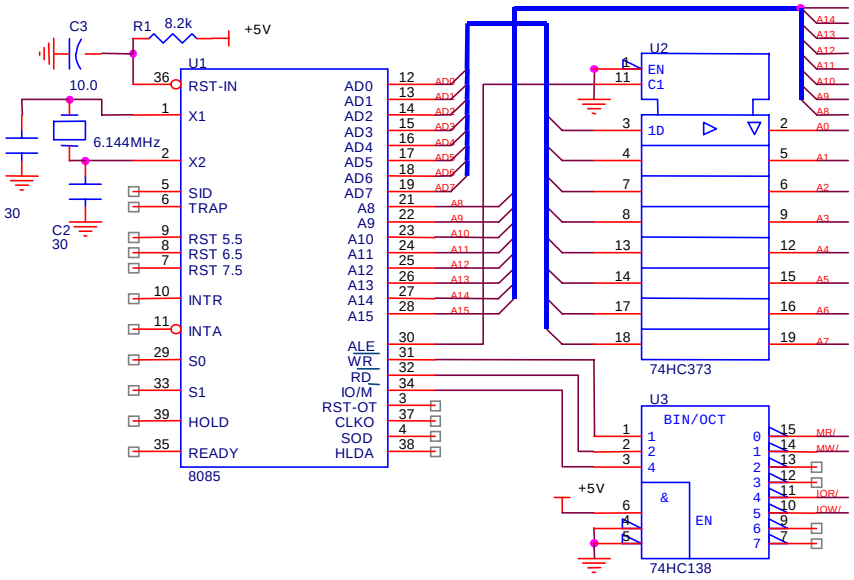
16-os ir 32-jų skilčių sistemos nagrinėjamos antroje ir trečioje knygos dalyse.

1.1. Universalūs 8-ių skilčių mikroprocesoriai

Vienas paprasčiausių, bet buvęs gana populiarus universalus 8 skilčių MP yra I8085A [16]. Tai pagerintas 8 skilčių MP I8080A [8] variantas, gaminamas KMDP technologija (maitinimo galia neviršija 0,1 W) . Komandos ciklo trukmė sumažinta iki 1,3 mks.

MP I8080A ir I8085A vidinė struktūra gana panaši: analogiška adresų erdvė, apdorojamų žodžių ilgis, komandų formatai ir adresavimo metodai. Dėl šių priežasčių I8085A gali pakeisti I8080A, išlaikant tą pačią programinę įrangą ir suteikiant MPS naujas savybes (maitinimo galios sumažėjimą, greitaveikos padidėjimą, MPS reikalingų DIG kiekio sumažėjimą, programinio modelio išplėtimą). MPS mikroprocesoriaus modulio su I8085A principinė schema pateikta 4 pav. Šiuo atveju į MP modulį įeina MP I8085A (U1), jaunesniosios adreso dalies fiksavimo registras U2 ir valdymo signalų dešifratorius U3, formuojantis analogiškus valdymo signalus kaip ir MPS su MP 8080A. Prie X1 ir X2 išvadų prijungiamas kvarcinis rezonatorius, nustatantis vidinio generatoriaus dažnį, lygų 6,144 MHz. Tuomet MP taktinis dažnis – 2,048 MHz. Vietoj kvarcinio rezonatoriaus galima imti LC arba RC grandis, taip pat išorinius sinchronizuojančius impulsus.

R1, C3 grandis skirta MPS nustatyti į pradinę padėtį, įjungus maitinimo įtampą. Išvaduose S0 ir S1 esantys signalai charakterizuoja MP būseną: kodas 00 rodo, kad MP yra HLT būsenoje (stovi), 10 reiškia rašymą, 01 - skaitymą, o 11 - komandos išrinkimą. Prie MP išvadų CLKO ir RST-OT gali būti prijungti įvairūs MPS mazgų sinchronizacijos ir nustatymo įėjimai, o išvadai SID ir SOD yra atitinkamai nuosekliai priimamų ir perduodamų duomenų įėjimas ir išėjimas.

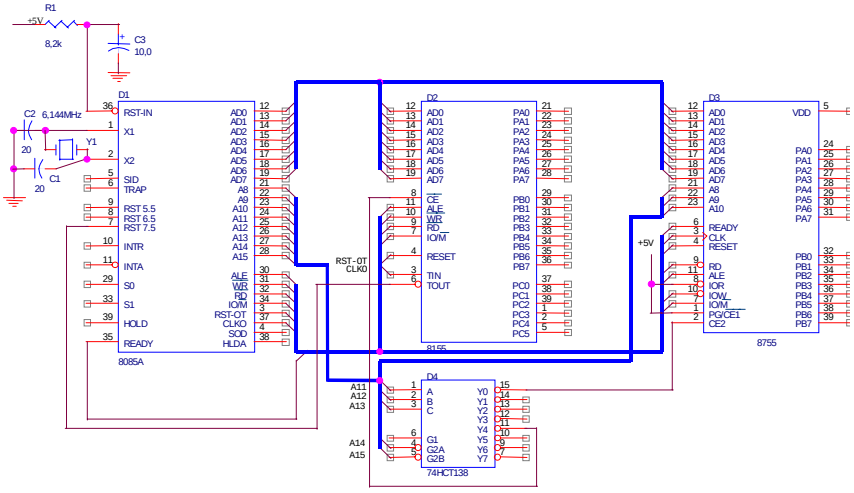


4 pav. MP modulis su I8085A

Atminties bei išvesties - įvesties valdymas šioje MPS vykdomas trimis signalais: IO/M/ (įvestis-išvestis arba atmintis), WR/ (write - rašymas) ir RD/ (read - skaitymas). Primename, kad sistemose su MP I8080A tam tikslui buvo naudojami keturi signalai: IOW/, IOR/, MW/ ir MR/. Bet kuri iš šių signalų sistemų gali būti pakeista į kitą (šiuo atveju tai padaryta pasitelkus dešifratorių 74HCT138).

Šis MP turi penkis pertraukčių įėjimus, vienas iš jų (INTR - interrupt request) pagal funkcinę paskirtį visiškai analogiškas MP I8080A įėjimui INT. Priminsime, kad MP, gavęs pertraukties signalą per šį įėjimą, iš duomenų magistralės priima vektorių-komandą RSTn, formuojamą išorinio įtaiso, pateikusio pertraukties signalą. Įvykdęs šią komandą, MP vykdo komandą, kurios kodas saugomas programų atmintyje su adresu 8n, vykdymo. Grįžimo į pertrauktą programą adresas įsimenamas dėkle (stack), kuris formuojamas DA. Kitų keturių pertraukties įėjimų RSTn signalai iš karto generuoja perėjimus prie komandų, kurių kodai saugomi atmintyje su adresais 8n (išoriniam įtaisui nereikia į duomenų magistralę pateikti savo pertraukties vektoriaus - komandos RSTn). Perėjimo adresai pertrauktims per įėjimus RST 5.5, RST 6.5, RST 7.5 ir TRAP (RST 4.5) yra tokie: 002Ch, 0034h, 003Ch ir 0024h. Pirmųjų trijų pertraukčių signalai gali būti “maskuojami” (draudžiami, arba kartais sakoma, kad jiems uždedama “kaukė”) programoje su komanda SIM (su

komanda RIM šią kaukę galima nuskaityti). Pertraukties signalas įėjime TRAP negali būti maskuojamas. Pertrauktys per šį įėjimą paprastai vykdomos įvairiomis avarinėmis situacijomis (sumažėjus maitinimo įtampai ar pan.).



5 pav. MPS su MP 8085A ir programuojamomis sąsajos DIG.

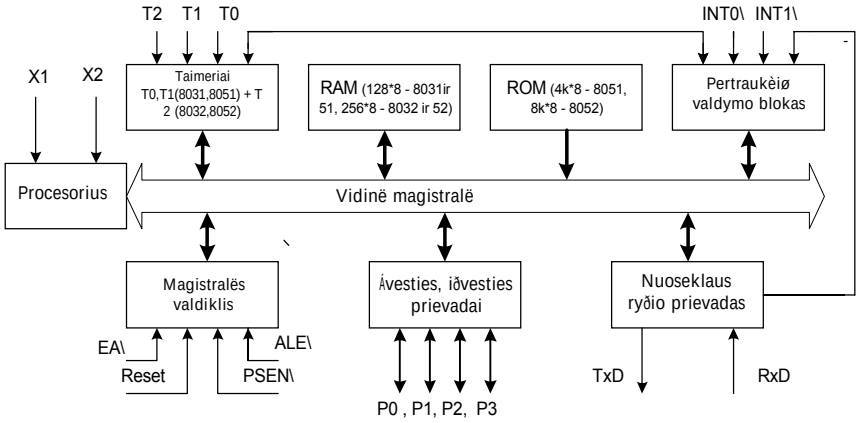
Visi pertraukčių signalai turi savajį prioritetą. Mažėjimo tvarka jie išdėstyti taip: TRAP, RST 7.5, RST 6.5, RST 5.5, INTR. Atėjus iš karto keliems pertraukčių signalams, bus aptarnautas iš pradžių tas, kurio prioritetas yra aukščiausias (aptarnaujama pertrauktis gali būti pertraukta bet kurio neuždrausto aukštesnio prioriteto pertraukties signalo). Aktyvūs signalų, veikiančių pertraukčių įėjimuose, parametrai yra tokie: RST 5.5, RST 6.5 ir INTR įėjimai aktyvūs vienetiniu lygiu, RST 7.5 - kylančiu frontu, TRAP- vienetiniu lygiu ir kylančiu frontu.

Prie MP 8085A labai patogu prijungti programuojamus periferinius DIG 8155 (D2) ir 8755 (D3) (5 pav.). Su pirmąja iš jų vartotojas gali sudaryti 24 programuojamų diskretinių įvesties/išvesties linijų sąsają. Be to, šiame DIG esantis 14 skilčių skaitiklis / taimeris leidžia skaičiuoti išorinius įvykius arba formuoti programuojamus laiko intervalus. Su DIG 8755A galima sudaryti dviejų 8 skilčių prievadų sąsają su individualiai programuojamomis dviejų krypčių linijomis, be to, vartotojas gali panaudoti 2k*8 baitų talpos vidinę EPROM programai saugoti.

MPS su MP 8085A struktūrą galima išplėsti pasitelkus programuojamą įvykių skaitiklį/taimerį 8253 (8254), pertraukimų valdiklį 8259, tiesioginių mainų valdiklį 8257, dinaminės klaviatūros ir indikatoriaus valdiklį 8279, rastrinio indikatoriaus valdiklį 8275, prietaisinės sąsajos IEEE-488 valdiklį

8293 , duomenų kodavimo/dekodavimo mazgą 8294 ir kitus programuojamus DIG [12,16].

1.1. MCS-51 šeimos mikrovaldikliai



6 pav. Apibendrinta MCS-51 vidinė struktūra

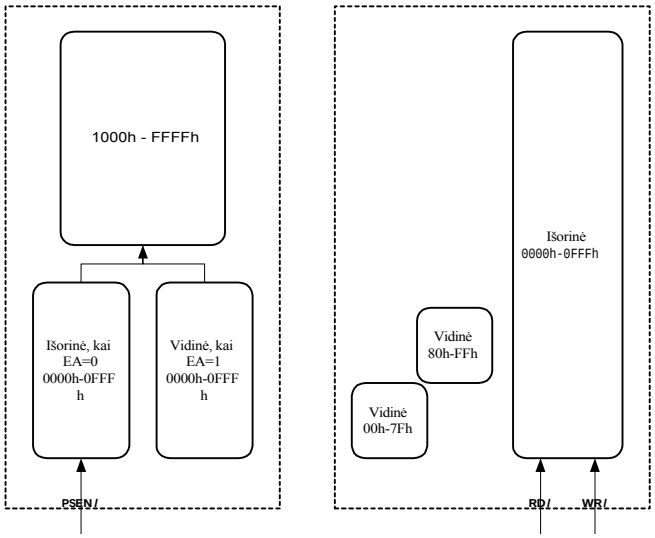
Tai vienkristaliai mikrokompiuteriai, turintys keturis programuojamus įvesties ir išvesties prievadus, sąveikaujančius su išore TTL schemų loginiais lygiais su trijų būsenų išėjimais [2,7,8,12,17,19]. Apibendrinta vidinė mikrovaldiklio (MV) struktūra pateikta 6 pav.; žymėjimas principinėje schemoje ir išoriniai valdymo signalai – 9 pav.

Visi MCS-51 (microcomputer system) šeimos mikrovaldikliai yra Harvardo architektūros ir turi atskirai adresuojamas programų ir duomenų atmintis PA ir DA (7 pav.).

Kai kurie šeimos nariai turi vidinę programų atmintį. Jos kiekis gali būti įvairus. MV 8051 turi 4K*8 vidinės programų atminties. Jei į MV išvadą EA/ tieksime loginį vienetą, tai procesorius adresų srityje 0000-0FFFh kreipsis į vidinę PA. Jei vartotojo programa yra didesnės apimties, tai adresų sritis nuo 1000h turi būti užpildyta išorinės PA ląstelėmis (skaitymas šiuo atveju atliekamas signalu PSEN/ (program store enable). 8031 vidinės PA neturi, todėl visa PA turi būti išorinė. Šiuo atveju į išvadą EA/ reikia tiekti loginį vienetą.

DA atminties struktūra kitokia. MV 8051 ir 8031 turi 128*8 bitus vidinės DA, o 8032 ir 8052 – 256*8 bitus. Vidinė DA yra paskirstyta į dvi sritis, kurių adresavimo laukai yra 00h – 7Fh ir 80h – FFh. Į pirmąją sritį galima kreiptis naudojant tiesioginę ir netiesioginę adresaciją, į antrąją – tik tiesioginę (žr. 8 pav.). Į išorinę DA galima kreiptis per vidinį procesoriaus 16-os skilčių

duomenų nuorodos registrą - rodyklę DPTR (data pointer). Rašymas ir skaitymas sinchronizuojami atitinkamai signalais WR/ ir RD/. Dalis DA atminties ląstelių yra rezervuota vidiniams registrams (žr. 1 lentelę). Duomenų ir programų atmintis gali būti išplėsta iki 64K*8 bitų kiekvienam tipui, prijungiant papildomus išorinės atminties blokus.



7 pav. MCS-51 programų ir duomenų atminties paskirstymo struktūra

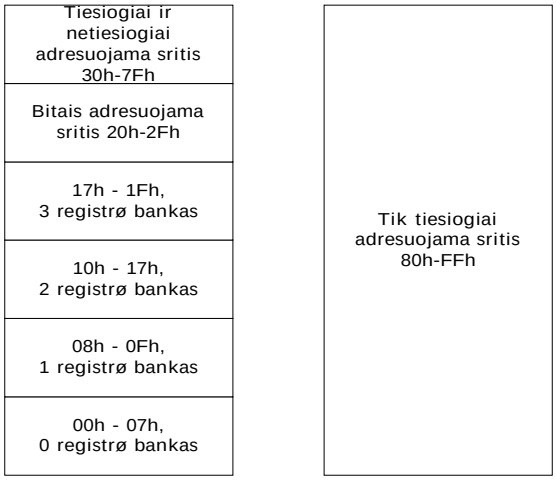
Specialiosios paskirties registrai 1 lentelė

Įrenginys	Adresas
A – akumuliatorius	E0h
B – akumuliatoriaus išplėtimo registras	F0h
PSW – procesoriaus būvio žodis	D0h
SP – dėklo (stack) rodiklis	81h
DPTR – duomenų nuorodos registras	
Vyriausiasis baitas (DPH)	83h
jauniausiasis baitas (DPL)	82h
P0 - 0 prievadas	80h
P1 - 1 prievadas	90h
P2 - 2 prievadas	A0h
P3 - 3 prievadas	B0h
IP – prioritetų registras	B8h
IE - pertraukčių "kaukės" registras	A8h
TMOD – taimerio režimo registras	

TCON - taimerio valdymo registras	89h
TH0 - taimeris 0 (vyriausiasis baitas)	88h
TL0 - taimeris 0 (jauniausiasis baitas)	8Ch
TH1 - taimeris 1 (vyriausiasis baitas)	8Ah
TL1 - taimeris 1 (jauniausiasis baitas)	8Dh
SCON – siųstuvo (imtuvo) valdymo registras	8Bh
SBUF – siųstuvo (imtuvo) duomenų registras	98h
PCON - galios valdymo registras	99h
	87h

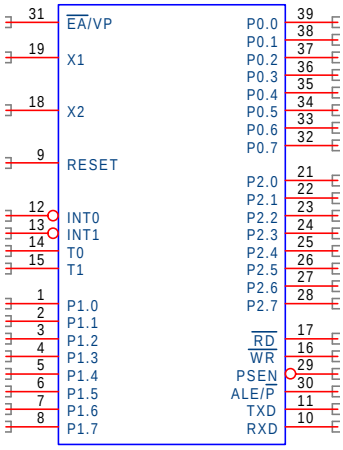
Visi keturi MV prievadai skirti informacijai baitais arba bitais įvesti arba išvesti. Duomenų perdavimo kryptis prievaduose gali būti nustatoma individualiai. Dirbant mikroprocesoriaus režimu, prievadai P0 ir P2 naudojami atminties kiekiui (duomenų ir programų) padidinti. Tai atliekama, prijungus išorinės atminties DIG. Šiuo atveju per prievadą P0 laikinio multipleksavimo būdu iš pradžių išvedamas adreso jaunesnysis baitas, o po to perduodamas arba priimamas duomenų baitas. Per prievadą P2 išvedamas vyresnysis adreso baitas. Visi prievado P3 išvadai gali būti panaudoti alternatyvioms funkcijoms formuoti. Jų paskirtis tokia: RD/ (P3.7)- skaitymas, WR/ (P3.6) - rašymas, T1 (P3.5), T0(P3.4) – vidinių taimerių T1 ir T0 įėjimai, INT1/ (P3.3), INT0/ (P3.2) - pertraukčių signalų įėjimai (aktyvūs loginiu vienetu arba krintančiu frontu), TXD (P3.1), RXD (P3.0) - nuoseklaus prievado siųstuvo išėjimas ir imtuvo įėjimas, dirbant jiems universalaus asinchroninio siųstuvo (imtuvo) (USART – universal synchronous asynchronous receiver transmiter) režimu, o dirbant postūmio registro (SPI – serial peripheral interface) režimu - atitinkamai sinchronizacijos išėjimas ir duomenų įvestis ir išvestis.

Prievadas P0 yra dvikryptis, o prievadai 1,2 ir 3 - kvazidvikrypčiai. Po signalo RESET į visų prievadų fiksatorius automatiškai įrašomi loginiai vienetai, kurie prievadus nustato įvesties režimui. Akivaizdu, kad vartotojas gali nustatyti bet kurį prievadą įvesties arba išvesties režimui (primename, kad esant išorinei atminčiai, prievadai 0 ir 2 tam tikslui negali būti panaudoti, nes per juos organizuojamas MV ryšys su išorine atmintimi).



8 pav. Vidinės duomenų atminties paskirstymas

Prievadų P1, P2 ir P3 išėjimo linijas galima apkrauti vienu TTL schemos įėjimu (apkrovos srovė ne didesnė kaip -1,6 mA), prievado P0 - dviem TTL schemų įėjimais.Visų prievadų linijos gali veikti ir su mažos galios KMDP schemomis, tačiau tada būtina kiekvieną jų prijungti per 10 - 15 kiloomų rezistorių (pull-up rezistor) prie maitinimo šaltinio, norint pakelti loginio vieneto lygį , kuris MDP schemose yra aukštesnis nei TTL schemose.



9 pav. MV I8051 išorinio valdymo signalai

Signalas ALE/ išoriniame registre fiksuoja jaunesnįjį išorinės atminties adreso baitą, formuojamą prievado P0 išėjimo buferyje.

Signalu PSEN/ skaitoma iš išorinės programų atminties. Kreipiantis į vidinę programų atmintį, jis neformuojamas.

RESET išvado paskirtis yra dvejopa. Pirmu atveju per jį perduodamas pradinio nustatymo, įjungus maitinimą, signalas, kitu atveju prie jo gali būti prijungtas nedidelės galios išorinis maitinimo šaltinis, maitinantis MV, perjungtą į sumažinto energijos sunaudojimo darbo režimą.

INT0/ ir INT1/ yra iš išorės paduodami pertraukties signalai, T0 ir T1 yra MV vidinių taimerių (skaitiklių) įėjimai, X1 ir X2 naudojami išorinio kvarcinio rezonatoriaus prijungimui, TxD, RxD yra duomenų perdavimo ir priėmimo nuoseklia sąsaja įvadai, o RD/, WR/ atlieka skaitymo ir rašymo į išorinius įrenginius signalų funkcijas.

Pasibaigus signalui RESET (jo trukmė turi būti ne mažesnė kaip kelios dešimtys sinchronizacijos periodų), procesorius pradeda vykdyti programą, esančią PA nuo 0000h adreso. Toliau PA yra išdėstyti pertraukties vektorių adresai (žr. 8 ir 18 pav.). Jie prasideda adresu 03h ir tęsiasi iki 2Bh. Skirtumas tarp pertraukties vektorių pradinių adresų yra 8 baitai. (Pertraukties vektoriaus pradiniame adrese dažniausiai yra besąlyginio perėjimo komandos kodas ir perėjimo adresas.)

Pagrindinis MV blokas yra aštuonių skilčių aritmetinis-loginis įrenginys (ALĮ). Jis atlieka sudėties, atimties, dalybos ir daugybos aritmetines operacijas, logines operacijas IR, ARBA, suma moduliu du, postūmio ir kitas operacijas.

Sudėtingas komandas ALĮ vykdo kaskadiniu paprasčiausių operacijų atlikimo mechanizmu. Pavyzdžiu gali būti sąlyginio perėjimo komandos vykdymas. Čia net tris kartus vyksta komandų skaitiklio inkrementavimas, du kartus skaitymas iš atminties, dviejų kintamųjų palyginimas, formuojamas perėjimo adresas ir sprendžiama, ar vykdyti, ar nevykdyti perėjimą. Visos šios operacijos atliekamos per 2 mks (kai sinchronizacijos dažnis 12MHz).

Svarbiausia ALĮ savybė ta, kad numatyta galimybė atlikti operacijas ne tik su baitais, bet ir su bitais. Atskiri bitai gali būti nustatyti, ištrinti, invertuoti, perduoti, patikrinti, panaudoti loginėse operacijose. Akumuliatorius yra operando šaltinis ir rezultato fiksavimo vieta, atliekant aritmetines, logines ir daugelį perdavimo komandų. Registras B dirba kaip laikino saugojimo registras.

Atlikdamas operacijas, ALĮ formuoja požymius, kurie fiksuojami procesoriaus būvio žodyje (PSW). 2 lentelėje pateikti visi požymiai ir jų formavimo sąlygos.

PSW registro turinys 2 lentelė

Požymis	Vieta PSW	Paskirtis ir formavimo sąlygos
C	PSW 7	Pernešimo požymis. Formuojamas atliekant aritmetines ir logines operacijas
AC	PSW 6	Papildomo pernešimo požymis. Formuojamas atliekant sumos ir skirtumo skaičiavimo komandas ir nustatomas kai yra pernešimas iš 3 skilties arba pasiskolinimas iš jos
F0	PSW 5	Vartotojo požymis. Jį galima nustatyti ir ištrinti, taip pat programiniu būdu patikrinti
RS1 RS0	PSW 4 PSW 3	Darbo registrų (R0-R7) banko numerio nustatymo skiltys
OV	PSW 2	Perpildymo požymis. Nustatomas vykdant aritmetines ir logines komandas
-	PSW 1	Nenaudojamas
P	PSW 0	Lyginumo požymis. Nustatomas vykdant aritmetines bei logines komandas. Rodo lyginį vienetų skaičių rezultate

Pagal registrų banko numerį darbo registrų adresai nustatomi taip, kaip parodyta 3 lentelėje.

Registrų bankų adresai vidinėje duomenų atmintyje 3 lentelė

RS1	RS0	Bankas	Adresai
0	0	0	00-07h
0	1	1	08-0Fh
1	0	2	10h-17h
1	1	3	18h-1Fh

8 bitų dėklo rodiklis SP gali adresuoti bet kurią vidinės duomenų atminties sritį. Po pradinio nustatymo (RESET) jo turinys tampa lygus 07h. Dėklo rodiklis inkrementuojamas rašant į dėklą, dekrementuojamas skaitant. Parenkant dėklo vietą atmintyje, reikia įvertinti programos duomenų vietą ir maksimalų dėklo gylį. Neteisingai nustačius dėklo parametrus, programos duomenys ir dėklo turiniys gali sutapti.

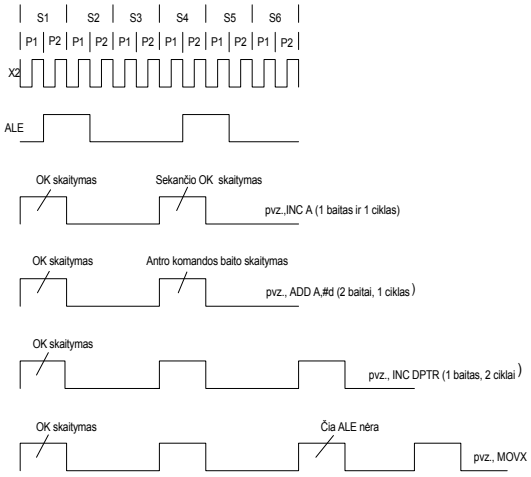
Dvibaitis duomenų nuorodos registras (rodyklė) DPTR naudojamas 16 skilčių adreso fiksacijai dirbant su išorine atmintimi. Jis tinka ir kaip du atskiri 8 skilčių duomenų registrai DPH ir DPL.

Į MV 8051 įeina registrų poros TH0, TL0 ir TH1, TL1, kurios naudojamos kaip du 16-os skilčių taimeriai arba įvykių skaitikliai.

SBUF sudaro du registrai - imtuvo duomenų registras ir siųstuvo duomenų registras. Įrašius duomenis į SBUF, tuoj pat prasideda baito perdavimas nuoseklia forma. Priimtas baitas taip pat nuskaitomas iš SBUF.

IP,IE,TMOD,TCON,SCON ir PCON naudojami atitinkamai pertraukčių iš taimerio, nuoseklaus ryšio ir galios nustatymo schemų valdymui bei būsenos bitų fiksacijai.

Vidinio generatoriaus darbo dažnį nustato išorinis kvarcinis rezonatorius, prijungtas prie MV išvadų X1 ir X2. Jis generuoja sinchronizacijos signalus, iš kurių formuojamas 12 arba 6 periodų valdymo automato būsenų mašininis ciklas. Kiekviena valdymo automato būsena sudaryta iš P1 ir P2 fazių. P1 fazėje atliekama operacija ALĮ, o P2 fazėje vyksta tarpregistriniai mainai. Visas mašininis ciklas sudarytas iš 12 fazių (10 pav.). Ši laikinė diagrama iliustruoja procesoriaus valdymo įrenginio darbą, atliekant įvairaus sudėtingumo komandas.



10 pav. Komandų vykdymo laiko diagramos

Dauguma MK 8051 komandų atliekamos per vieną mašininį ciklą. Kai kurios komandos, operuojančios žodžiais, atliekamos per du mašininius ciklus. Tik dalybos ir daugybos komandos atliekamos per keturis mašininius ciklus.

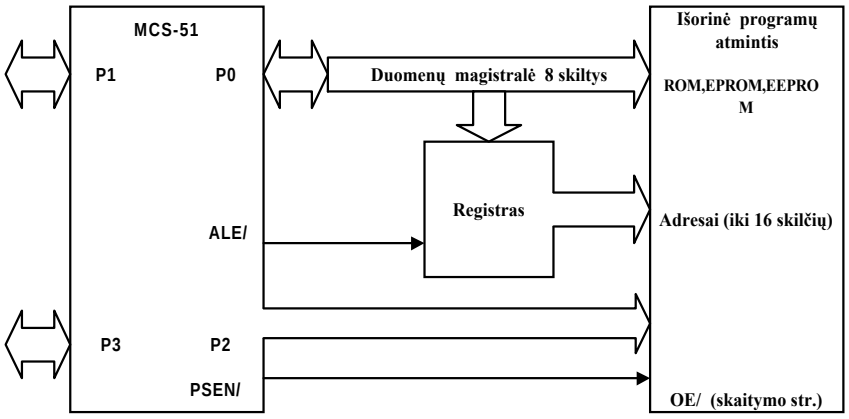
Dirbant su įvesties ir išvesties prievadais, galima naudoti bitines ir baitines operacijas. Kai prievadas tuo pačiu metu yra operando šaltinis ir imtuvas, mikroprocesoriaus valdymo įrenginys realizuoja specialų ciklą "skaitymas-modifikacija-įrašymas". Šiuo atveju įvestis vyksta ne iš išorinių prievadų išvadų, bet iš vidinio fiksuojančio registro. Tokiu būdu išvengiama neteisingo anksčiau išvestos informacijos skaitymo. Toks režimas realizuojamas vykdant komandas:

ANL- loginis "IR";
ORL - loginis "ARBA";
XRL - suma moduliu du;

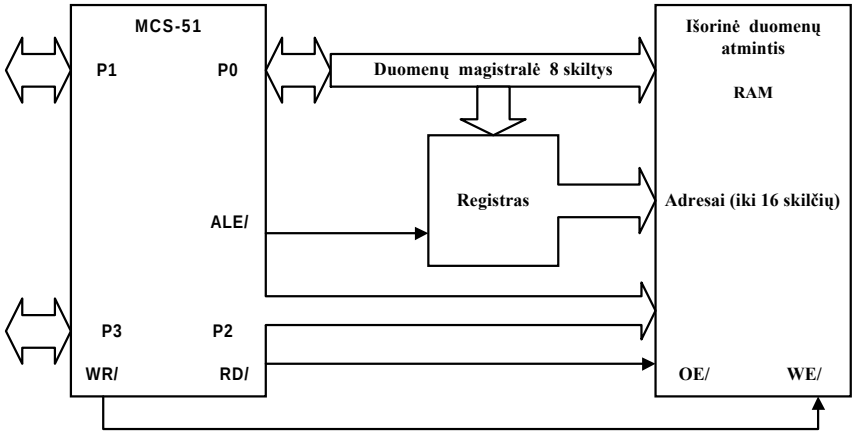
JBC – pereiti, jei adresuojamame bite yra “1”,o po to bitą ištrinti;
CPL - bito inversija;
INC - prievado turinio inkrementavimas;
DEC - prievado turinio dekrementavimas;
DJNZ - prievado turinio dekrementavimas ir perėjimas, jei jo turinys nelygus nuliui;
MOV PX.Y, C - pernešimo bito persiuntimas į X prievado Y bitą;
SET PX.Y - X prievado Y bito nustatymas;
CLR PX.Y - X prievado Y bito trynimas.

1.1. Išorinės DA ir PA prijungimas

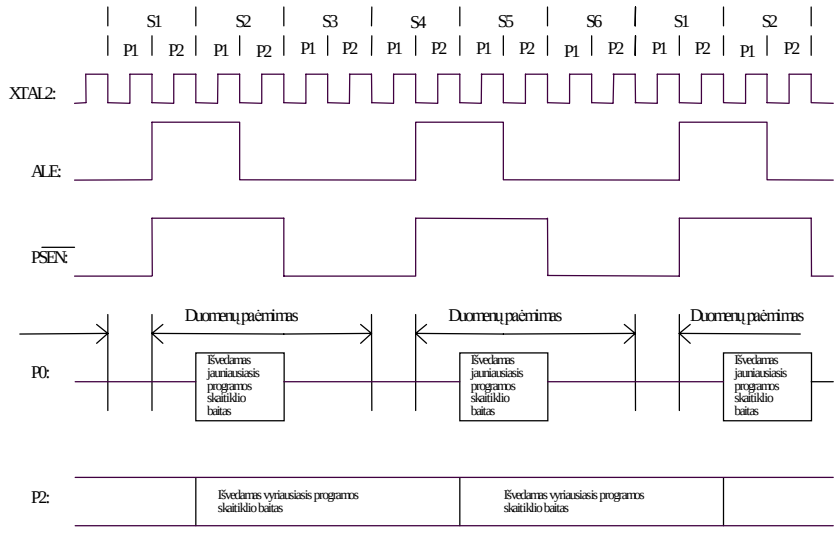
Kai MV veikia mikroprocesoriaus režimu, prijungiama išorinė programų (IPA) ir duomenų (IDA) atmintys (11 ir 12 pav). Kreipiantis į IPA, generuojamas signalas PSEN/, o skaitymas ir rašymas į IDA valdomas signalais RD/ ir WR/, kurie formuojami išvaduose P3.6 ir P3.7. Dirbant su DA, naudojamas 16-os (MOV A,@DPTR) arba 8-ių (MOV X A, @Ri) bitų adresai. Vyresnysis adreso baitas visada formuojamas prievado P2 išvaduose. Jeigu tolesnis išorinis atminties ciklas vyksta ne tuoj pat, tai adreso turinys prievade P2 išsaugomas iki kito kreipinio į išorinę atmintį. Skaitymo iš IPA laiko diagramos pateiktos 13 pav. Čia signalas ALE naudojamas, siekiant suderinti laike prievado P0 išėjime vykstančius procesus , t.y. jaunesniojo IPA adreso baito perdavimą ir duomenų priėmimą iš IPA. ALE kiekviename mašininiame (S1-S6) cikle įgauna vieneto lygį du kartus, netgi tada, kai komandos cikle nėra kreipimosi į IPA (S4-S5). Vykdant komandą MOVX, pirmojo ALE signalo nebūna . Taigi, jei MPS su MV 8051 dirba be išorinės atminties, ALE kartojasi periodiškai, dažniu, lygiu 1/24 kvarcinio rezonatoriaus dažnio. Tada šį signalą galima panaudoti išorinių įrenginių sinchronizacijai. Per prievado P0 išvadus, multipleksuojant laike, perduodamas jaunesnysis adreso baitas ir duomenys. Signalas ALE/ naudojamas jaunesniojo adreso baito fiksavimui išoriniame registre. Išvesties režimu duomenys P0 išvaduose pasirodo tik esant aktyviam signalui WR/. Įvesties režimu duomenys per P0 įvedami signalo RD/ frontu. Darbas su IPA įmanomas tik tuo atveju, jeigu yra aktyvus signalas EA/ arba jei komandų skaitiklio turinys viršija 0FFFh reikšmę. MV 8051 vidinės programų atminties turinys įrašomas tik vieną kartą ir negali būti vartotojo pakeistas. MV 80751 vidinė programų atmintis yra EPROM tipo ir turi galimybę informaciją ištrinti ultravioletiniais spinduliais. MK 8031 vidinės PA iš viso neturi. Pirmu ir antru atveju, vykdant vartotojo programos derinimą, patogu sutapdinti duomenų ir programų atminties sritis. Tai galima atlikti su schema, pateikta 14 pav.



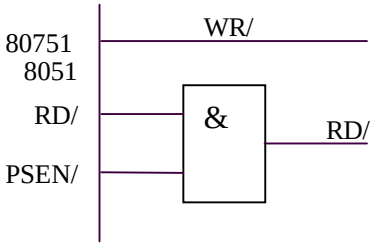
11 pav. MCS-51 išorinės programų atminties prijungimas



12 pav. MCS-51 išorinės duomenų atminties prijungimas



13 pav. Skaitymas iš IPA



14 pav. Programų ir duomenų atminties adresų laukų sutapdinimas MV 8051 ir 80751.

1.1. Vidiniai skaitikliai – taimeriai

Vartotojas savo tikslams gali panaudoti du 16-os skilčių programuojamus įvykių skaitiklius - taimerius. Dirbant taimerio režimu, skaitiklio turinys inkrementuojamas kas 12-a MV sinchronizacijos generatoriaus periodų. Dirbant skaitiklio režimu, skaitiklio turinys inkrementuojamas krinančiu signalų, veikiančių įėjimuose T0 arba T1, frontu. Maksimalus įėjimo signalo

dažnis lygus 1/24 MV sinchronizacijos dažnio. Taimerių darbo režimų valdymui naudojami TMOD (taimerio režimo) ir TCON (taimerio valdymo) registrai. Jų formatai pateikti 4 lentelėje.

Skaitiklių-taimerių režimo valdymo registro formatas 4 lentelė

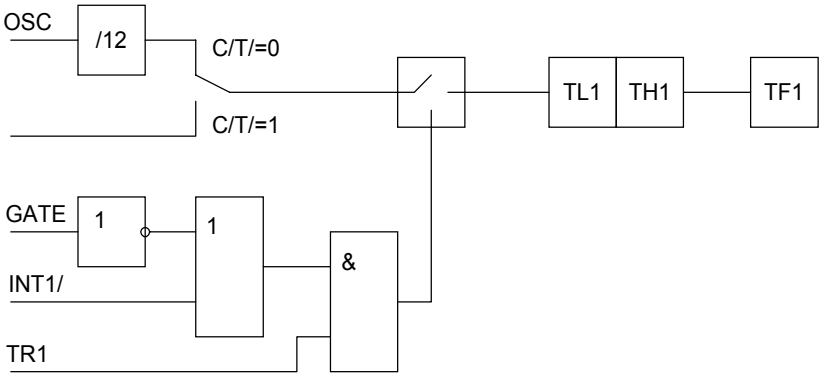
Pavadinimas	Pozicija	Paskirtis
Gate	TMOD.7 C1 TMOD.3 C0	Skaitiklio darbo blokavimas. Jei šis signalas nustatytas, tai skaitiklis veikia kol įėjime INT/ aukštas signalo lygis ir valdymo bitas TR nustatytas
C/T/	TMOD.6 C1 TMOD.2 C0	Taimerio arba skaitiklio darbo režimo nustatymas. Jei bitas nustatytas, tai dirba išorinių įvykių skaitiklis
M1 M2	TMOD.5 C1 TMOD.1 C0	Darbo režimo numerio kodas

M1	M0	Darbo režimas
0	0	- 0 darbo režimas. Skaitiklis veikia kaip 5 skilčių dažnio daliklis
0	1	- 1 darbo režimas.16-os skilčių skaitiklis
1	0	- 2 darbo režimas. 8-ių skilčių automatiškai perkraunamas skaitiklis/taimeris. TH saugo reikšmę, kuri turi būti perkraunama į TL tuoj pat po persipildymo
1	1	- 3 darbo režimas.1-as skaitiklis/taimeris stabdomas, TL0 dirba kaip 8-ių skilčių skaitiklis/taimeris, o jo režimas priklauso nuo TH0 valdančių bitų

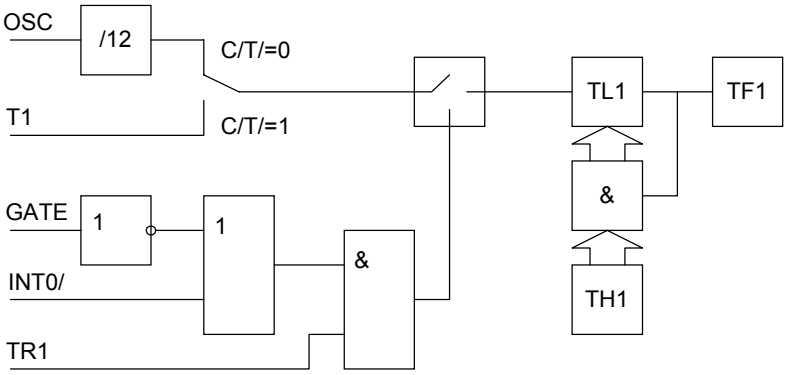
5 lentelėje pateiktas įvykių skaitiklių/taimerių valdymo registro formatas, o 15-17 pav. - jų darbo režimų iliustracija.

Taimerio valdymo registro TCON formatas 5 lentelė

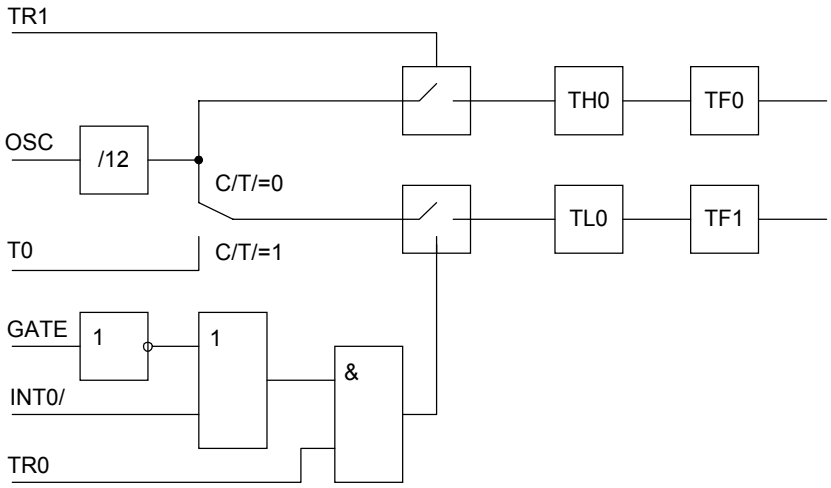
Bitas	Bito numeris	Paskirtis
TF1	TCON.7	T1 taimerio perpildymo požymis. Jis trinamas
TR1	TCON.6	aptarnaujant pertrauktį
TF0	TCON.5	T1 taimerio valdymo bitas
TR0	TCON.4	T0 taimerio perpildymo požymis
IE1	TCON.3	T0 taimerio valdymo bitas
IT1	TCON.2	INT1/ pertraukties fronto požymis
IE0	TCON.1	INT1/ pertraukties tipo valdymo bitas
IT0	TCON.0	INT0/ pertraukties fronto požymis INT0/ pertraukties tipo valdymo bitas



15 pav. Taimerio (įvykių skaitiklio) 0 darbo režimo iliustracija



16 pav. Taimerio (įvykių skaitiklio) 2 darbo režimo iliustracija



17 pav. Taimerio (įvykių skaitiklio) 3 darbo režimo iliustracija

1.1. Nuoseklių komunikacijų grandys

Universalus asinchroninis siųstuvas-imtuvas (UART - universal asynchronaus receiver transmitter) bitas po bito priima ir perduoda informaciją jaunesniasiais bitais į priekį pusiaudupleksiniu režimu. Į UART sudėtį įeina priimantis ir siunčiantis postūmio registrai ir duomenų registras SBUF. Baito įrašymas į jį automatiškai iškviečia perrašymą į postūmio registrą ir perdavimo pradžią. Imtuvo buferinis registras leidžia sutapatinti anksčiau priimto baito skaitymą ir kito priėmimą.

UART gali dirbti keturiais režimais:

- 0 režimas. Šiuo režimu informacija perduodama ir skaitoma per įėjimą RxD. Perduodami ir priimami 8 bitai. Per įėjimą TxD perduodami postūmio sinchronizacijos impulsai. Bitų perdavimo dažnis lygus 1/12 kvarcinio rezonatoriaus dažnio.
- 1 režimas. Čia per TxD perduodami, o per RxD priimami 10 informacijos bitų: startinis, 8 duomenų ir stopinis. Perdavimo greitis nustatomas taimeriu.
- 2 režimas. Čia per TxD perduodami, o per RxD priimami 11 informacijos bitų: startinis, 8 duomenų, programuojamas 9 ir stop bitai. Į 9 bito poziciją galima dėti PSW lyginumo skiltį. Perdavimo greitis lygus 1/32 arba 1/64 kvarcinio rezonatoriaus dažnio, priklausomai nuo registro PCON skilties SMOD turinio (6 lentelė).

- 3 režimas. Sutampa su antruoju, išskyrus tai, kad perdavimo dažnį nustato 1 taimeris.

UART darbo režimas nustatomas su registro SCON dviem vyriausiomis skiltimis. Jo formatas pateiktas 6 lentelėje.

SCON registro formatas 6 lentelė

SM0	SCON 7	UART režimo valdymo bitas.
SM1	SCON 6	UART režimo valdymo bitas.
SM2	SCON 5	Nustatomas programiniu būdu draudžiant priimti žodį, kuriame devintas bitas lygus 0
REN	SCON 4	Leidimas priimti. Nustatomas ir gesinamas programiniu būdu, leidžiant arba draudžiant nuoseklų duomenų priėmimą
TB8	SCON 3	Devintojo bito perdavimas; nustatomas ir gesinamas programiniu būdu įvedant šį bitą
RB8	SCON 2	Devintojo bito priėmimas; nustatomas ir gesinamas aparatiniu būdu šio bito fiksacijai
TI	SCON 1	Siųstuvo pertraukimo požymis. Nustatomas aparatiniu būdu, baigus siųsti baitą. Gesinamas užbaigus pertraukties
RI	SCON 0	aptarnavimą Imtuvo pertraukimo aptarnavimo požymis.Nustatomas aparatiniu būdu, priėmus baitą. Gesinamas aptarnavus pertrauktį

SM0	SM1	
0	0	- įvedimo (išvedimo) postūmio registras
1	1	- UART - 8. Kintamas darbo greitis
0	0	- UART - 9. Fiksuotas darbo greitis
1	1	- UART - 9. Kintamas darbo greitis

0 režimu perdavimo greitis (transfer rate) priklauso tik nuo kvarcinio rezonatoriaus darbo dažnio $f_0 = f_{kv}/12$. Per vieną mašininį ciklą UART perduoda vieną informacijos bitą. 1, 2 ir 3 režimais priėmimo (perdavimo) greitis priklauso nuo registro PCON skilties SMOD (0 arba 1) turinio. 2 režimu perdavimo greitis randamas iš šios sąlygos:

$$f_2 = (2^{SMOD}/64) f_{kv},$$

kai SMOD=0, perdavimo greitis lygus

$$(1/64) f_{kv},$$

kai SMOD=1 -

$$(1/32) f_{kv}.$$

1 ir 3 režimais perdavimo greitis formuojams taimeriu. Šiuo atveju perdavimo greitis priklauso nuo T1 taimerio persipildymo dažnio

$$f_{1,3} = (2^{SMOD} / 32) f_{per1},$$

Pertrauktis nuo T1 taimerio šiuo atveju turi būti blokuojama. T1 gali dirbti bet kuriuo režimu, bet patogiusias taimerio darbo režimas su autoperkrovimu. Tada

$$f_{1,3} = (2^{SMOD} / 32)(f_{kv} / 12)(256 - TH1).$$

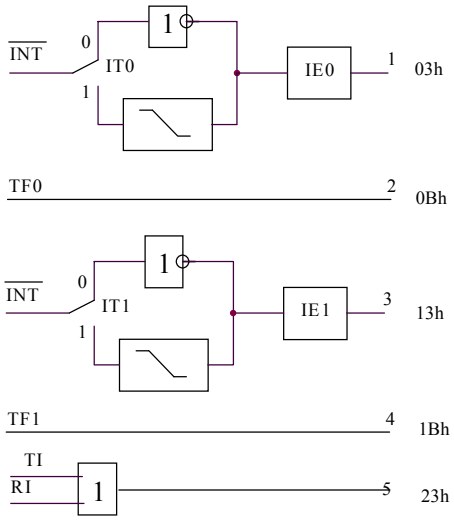
1.1. Pertraukčių grandys

Pertrauktys gali būti inicijuotos išoriniais signalais INT0/ ir INT1/. Jie gali būti potencialiniai arba impulsiniai, priklausomai nuo registro TCON skilčių IT0 ir IT1 turinio. Jeigu IT0 (IT1) yra loginis 0, tai pertrauktis bus inicijuota loginio vieneto lygiu išvaduose INT0/ (INT1/). Jeigu IT0 (IT1) yra loginis 1, tai pertrauktis bus inicijuota pagal krintantį signalo išvaduose INT0/ (INT1/) frontą. Pertrauktys gali būti draudžiamos arba leidžiamos. Tai atlieka pertraukimų "kaukės" registras IE. Jo formatas pateiktas 7 lentelėje

IE registro formatas 7 lentelė

EA	IE 7	Visų pertraukčių leidimas (draudimas) (1-leisti, 0-drausti)
ES	IE 4	UART pertraukties leidimas (draudimas) (1-leisti, 0-drausti)
ET1	IE 3	T1 skaitiklio/taimerio leidimas (draudimas) (1-leisti, 0-drausti)
EX1	IE 2	INT1/ pertraukties leidimas (draudimas) (1-leisti, 0-drausti)
ET0	IE 1	T0 skaitiklio/taimerio leidimas (draudimas) (1-leisti, 0-drausti)
EX0	IE 0	INT0/ pertraukties leidimas (draudimas) (1-leisti, 0-drausti)

Programos vykdymo metu galima keisti ir pertraukčių aptarnavimo prioritetą. Tai atliekama su aptarnavimo prioriteto registru IP. Jo formatas pateiktas 8 lentelėje.



18 pav. MCS-51 pertraukčių schema

IP registro formatas 8 lentelė

PS	IP 4	UART prioriteto bitas. Nustatomas arba gesinamas programiniu būdu, nustatant aukščiausią (žemiausią) prioritetą
PT1	IP 3	1 taimerio prioriteto bitas. Nustatomas arba gesinamas programiniu būdu, nustatant aukščiausią (žemiausią) prioritetą
PX1	IP 2	INT1/ prioriteto bitas. Nustatomas arba gesinamas programiniu būdu, nustatant aukščiausią (žemiausią) prioritetą
PT0	IP 1	0 taimerio prioriteto bitas. Nustatomas arba gesinamas programiniu būdu, nustatant aukščiausią (žemiausią) prioritetą INT0/ prioriteto bitas. Nustatomas arba gesinamas programiniu būdu, nustatant aukščiausią (žemiausią) prioritetą
PX0	IP 0	

18 pav. pateikta MV I8051 pertraukčių schema. Čia TF0 yra 0 skaitiklio (taimerio) formuojamas pertraukties požymis, TF1 - 1 skaitiklio (taimerio)

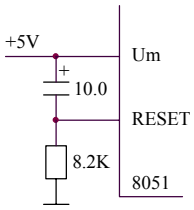
formuojamas pertraukties požymis. Gavęs pertraukties signalą, MV I8051 suformuoja komandos LCALL kodą ir į dėklą padeda programos skaitiklio PC turinį. Pagal atitinkamą vektoriaus adresą perėjimo komanda JMP turi būti dedama į pradinį pertraukties aptarnavimo paprogramės adresą. Čia turi būti numatytos pagrindinės programos kintamųjų įrašymo į dėklą komandos. Paprogramės turi baigtis komanda RETI (grįžti iš pertraukties aptarnavimo paprogramės).

1.1. MCS-51 pradinis nustatymas ir maitinimo galios valdymas

MV I8051 pradiniam nustatymui į išvadą RESET siunčiamas loginis "1". Jo trukmė turi būti ne mažesnė kaip 24 sinchronizacijos periodai. Veikiant signalui RESET, išvadai ALE/ ir PSEN/ veikia įvedimo režimu, o PC, ACC, B, PSW, DPTR, TMOD, TCON, T/C0, T/C1, IE, IP ir SCON registrai bei PCON registro vyriausioji skiltis yra gesinami. Dėklo rodiklyje SP nustatomas adresas 07h, prievaduose P0÷P3 - 0FFh, SBUF turinys lieka atsitiktinis, RAM turinys nepakinta.

Signalą RESET galima suformuoti su 19 pav. pateikta schema. Įjungus maitinimo šaltinį, loginis vienetas įėjime RESET bus palaikomas tol, kol įsikraus kondensatorius per rezistorių. Projektuojant reikia turėti galvoje, kad RESET signalo trukmė privalo būti didesnė už maitinimo šaltinio įtampos nusistovėjimo laiką, t.y. užtrukti laiko intervalą nuo įjungimo iki tol, kol šaltinio įtampa pasieks nominalią reikšmę (su +- 10 % tolerancija).

MCS-51 šeimos elementai, žymimi su raide C (80C31,80C51), gaminami KMDP technologija. Šie MV gali veikti sumažintos maitinimo galios režimais. Jeigu galios valdymo registro PCON (jo formatas pateiktas 9 lentelėje) skiltyje IDL nustatomas loginio vieneto lygis, MV pereina į tuščiosios eigos režimą (Idle mode). Čia sinchronizacijos generatorius veikia, visi registrai ir DA išsaugo savo turinį. Taimeriai, nuoseklių komunikacijų ir pertraukimų blokai gauna sinchrosignalus, bet CPE sinchronizacija nutraukiama. Signalai ALE/ ir PSEN/ įgauna "1" būseną.



19 pav. Signalo RESET formavimo schema

PCON registro formatas 9 lentelė

SMOD	PCON 7	Dvigubas perdavimo greitis. Jei SMOD=1, greitis dvigubas Kai SMOD=0 viengubas
GF1	PCON 3	Vartotojo požymis
GF0	PCON 2	Vartotojo požymis
PD	PCON 1	Mažos maitinimo galios požymis
IDL	PCON 0	Tuščiosios eigos požymis

Išeiti iš tuščiosios eigos režimo galima tik signalu RESET, arba padavus pertraukties signalą. Bet koks leistas pertraukties signalas gesins IDL ir nutrauks tuščiosios eigos režimą. Požymiais GF1 ir GF0 vartotojas gali naudotis savo nuožiūra.

Nustačius bitą PD, sinchronizacijos generatorius sustabdomas, RAM ir registrų turiniai išlieka nepakitę, prievadų turiniai įgauna jų fiksuojančių registrų turinių reikšmes. ALE/ ir PSEN/ gesinami. MV maitinimo įtampą turi gauti per įėjimą RESET (ji negali būti mažesnė kaip 2 V); U_m gali būti išjungta. Šis darbo režimas patogus tuo atveju, kai gaunamas avarinis pertraukimo signalas iš maitinimo bloko apie sumažėjusią maitinimo įtampą. Tada atitinkama šio pertraukimo aptarnavimo paprogramė turi išsaugoti MV registrų turinius vidinėje duomenų atmintyje ir pervesti MV į tuščiosios eigos režimą.

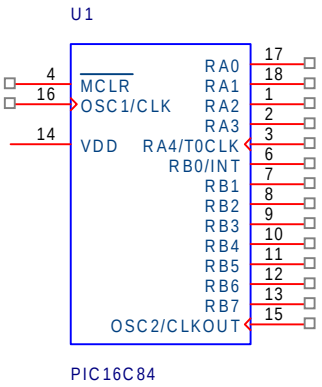
Išeiti iš tuščiosios eigos režimo galima tik su signalu RESET. Čia visi registrų turiniai pakinta, o vidinės duomenų atminties turinys išsaugomas.

2.10. Programuojami sąsajų valdikliai

Nesudėtingiems programuojamiems įrenginiams projektuoti galima naudoti PIC (programmable interface controller - programuojamas sąsajos valdiklis). Tai pigūs, greiti ir lanksčiai naudojami MP. Juos gamina daugelis firmų, tokios kaip Motorola, Atmel, Arizona Microchip ir kt. Ypač populiari rinkoje pastarosios produkcija. Toliau aptarsime vieną iš jos gaminių - PIC16C84. Tai pakankamai našus 8 bitų mikrovaldiklis, sukurtas naudojant RISC (reduced instruction set computer) architektūrą. Šiame MV panaudota Harvardo struktūra – programos ir duomenų atminties valdymas yra atskiras. Komandų adreso magistralė yra 14 bitų, o duomenų – 8 bitų. Tokia architektūra leidžia visas komandas (išskyrus programinių perėjimų) įvykdyti per vieną mašininį ciklą (tradicinės Neumanno architektūros atveju per bendrą magistralę nuskaitoma komanda, po to – duomenys, dėl to pailgėja komandos vykdymo laikas). Tokia koncepcija užtikrina paprastą, tačiau našią komandų sistemą (tik 35 komandos), sukurtą taip, kad visos bitinės, baitinės ir registrinės operacijos

atliekamos pakankamai greitai, sutapatinus laike komandų nuskaitymą ir vykdymą (dviejų pakopų konvejeris).

Programų atminčiai panaudota EEPROM yra labai patogi derinimo stadijoje, nes atsiranda perprogramavimo galimybė. PIC16C84 puikiai tinka auto-mototechnikoje - vidaus degimo variklių valdymui, parametrų kontrolei, elektroninėse spynose, apsaugos įrenginiuose, elektroninėse kortelėse ir pan. MV dedamas į 18-os išvadų DIP (dual in line package) tipo korpusą (yra paviršinio montažo korpusai (SMD) ir kristalai be išvadų). Atskirų išvadų paskirtis ir žymėjimas principinėje schemoje pateikti 20 pav.

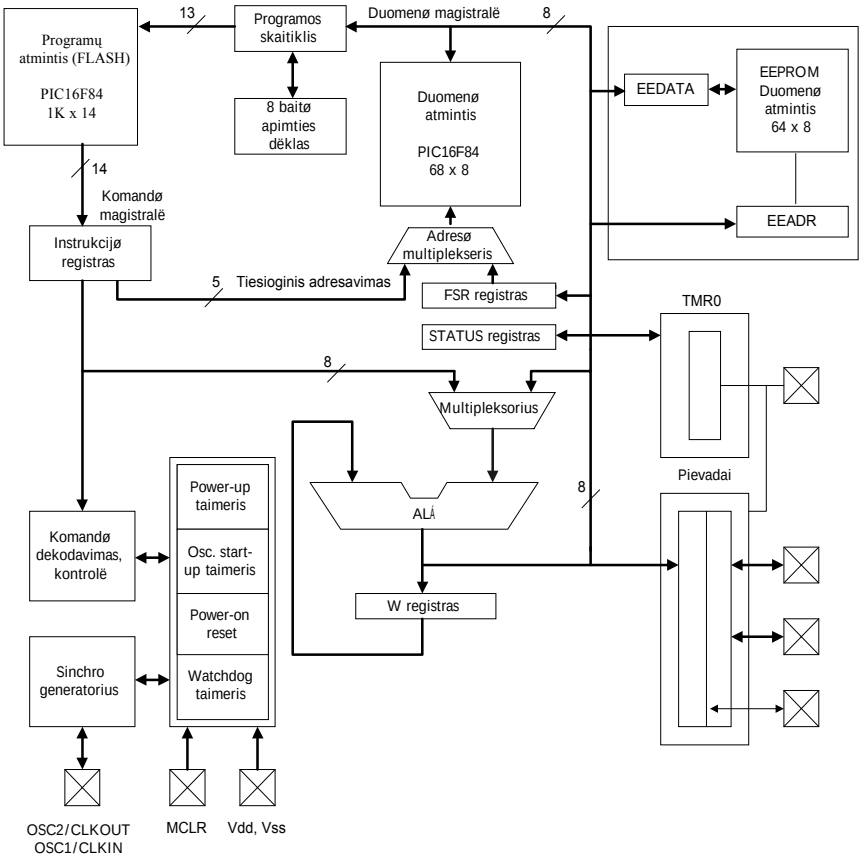


20 pav. MV PIC16C84 išvadų paskirtis

Apibendrinta vidinė programuojamo sąsajos valdiklio (PIC) struktūra vaizduojama 21 pav. Jis turi operatyviąją atmintį RAM, 64 baitus EEPROM tipo duomenų atminties, 13 įvesties (išvesties) išvadų, taimerį (skaitiklį), 1Kx14 bitų apimties EEPROM rūšies programų atmintį, 8 bitų aritmetinį-loginį įrenginį, kaupiklio tipo darbinį registrą W.

Operacijos atliekamos su darbinio registro W ir bet kurio kito registro turiniais. Darbinis registras yra 8 bitų ir neadresuojamas. ALU būseną atspindi registro STATUS turinys. Operatyvioji atmintis skirstoma į dvi dalis – specialiosios paskirties registrus SPR (pirmieji 12 RAM adresų) ir bendrosios paskirties registrus BPR (0Ch-2Fh, 10 lentelė).

Netiesioginio adresavimo registras INDO fiziškai neegzistuoja Jis naudoja išrinkimo registrą FSR netiesioginiam vieno iš 64 galimų registrų išrinkimui. Bet kuri komanda, naudojanti INDO, iš tikrųjų kreipiasi į duomenų registrą, kurį nurodo FSR.



21 pav. MV PIC 16C84 struktūra

Taimerio (skaitiklio) registro RTCC turinys gali padidėti veikiant išoriniam (įėjimas RA4\RTCC) arba vidiniam signalui, pasikartojančiam vykdomų komandų dažniu, todėl taimeris gali būti panaudotas išorinių įvykių kiekiui skaičiuoti ir laiko intervalams matuoti. Išorinių įvykių arba vidinio šaltinio signalų dažnis gali būti padalytas į PIC įterptu 8 bitų programuojamu dalikliu.

Būsenos žodžio registras STATUS panašus į PSW registrą, esantį MV 8051.

IRP	RP1	RP0	TO	PD	Z	DC	C

Jame yra keliamojo vieneto požymis C (nustatomas atliekant sudėties ir atimties komandas, kai yra keliamasis (skolinimosi) vienetas iš vyriausiosios skilties), dešimtainio keliamojo vieneto požymis DC (atliekant anksčiau minėtas komandas yra keliamasis vienetas iš jaunesniosios tetrados į vyresniąją), nulinio rezultato požymis Z. PD (Power Down) ir TO (Time Out) yra aparatiniai būsenos bitai (nustatomi į "1" įjungus maitinimą arba su komanda CLRWDT.Komanda SLEEP grąžina šių bitų reikšmę į "0").

RAM organizacija 10 lentelė

Adresas	0 bankas	1 bankas	Adresas
00h	INDO	INDO	80h
01h	TMRO	TMRO	81h
02h	PCL	PCL	82h
03h	STATUS	STATUS	83h
04h	FSR	FSR	84h
05h	PORTA	TRISA	85h
06h	PORTB	TRISB	86h
07h	_____	_____	87h
08h	EEDATA	EECON1	88h
09h	EEADR	EECON2	89h
0Ah	PCLATH	PCLATH	8Ah
0Bh	INTCON	INTCON	8Bh
0Ch 2Fh	36 BPR (SRAM)	Taip pat	8Ch AFh
30h 7Fh	Neegzistuojanti (skaitomi nuliai)	Taip pat	B0h FFh

Pagal šių bitų būseną galima nustatyti MV "Nustatymo į pradinę būseną" (NPB) priežastį (11 lentelė).

PD ir TO bitų reikšmės 11 lentelė

TO	PD	NPB priežastis
0	0	Išėjimas iš "SLEEP" režimo pasibaigus WDT taimerio laikui
0	1	WDT taimerio laiko pabaiga (ne "SLEEP" režimas)
1	0	SLEEP komanda
*	0	Išėjimas iš "SLEEP" atėjus išoriniam signalui \MCLR
1	1	Maitinimo įjungimas arba komanda CLRWDT
*	*	"0" impusas įėjime \MCLR

RP1,RP0 bitai išrenka duomenų atminties puslapį, esant tiesioginiam adresavimui (00- nulinis puslapis (00h- 7Fh), 01 – pirmasis puslapis (80h- FFh), 10 – antrasis puslapis (100h – 17Fh), 11 – trečiasis puslapis (180h –

1FFh)). Kiekvienas puslapis apima 128 baitus. PIC16C84 panaudotas tik nulinis puslapis, kiti puslapiai įeis į vėlesnes PIC modifikacijas.

IRP bitas išrenka duomenų atminties puslapius, esant netiesioginei adresacijai (0 – 0,1 puslapiai (00-FFh), 1 – 2,3 puslapiai (100h – 1FFh)). Jis turėtų būti naudojamas vėlesnėse PIC modifikacijose. Šiuo metu jis tinkamas kaip bendros paskirties rašymo (skaitymo) bitas.

Išrinkimo registras FSR kartu su registru INDO (neegzistuojančiu) naudojamas netiesioginiam bet kurio iš 64 registrų adresavimui. Fiziškai realizuoti 36 vartotojo BPR (0Ch – 2Fh adresai) ir 15 SPR, išdėstytų skirtingais adresais.

Įvesties ir išvesties registrai PORTA, PORTB atitinka du IN\OUT prievadus. A prievadas turi 5 skiltis PA4 – PA0, kurios gali būti individualiai užprogramuotos duomenims įvesti arba išvesti su registru TRISA. B prievadas turi 8 skiltis PB7 – PB0, programuojamas su registru TRISB. Loginio vieneto įtampos nustatymas atitinkamoje registrų TRIS skiltyje (programuojamos prievadų skilties numeris atitinka TRIS registrų atitinkamų numerių skiltis) nustato prievado skiltį įvedimui. Įvedant prievado turinį, tiesiogiai nuskaitoma išvadų būsena, išvedant į prievadą, rašoma į buferinį prievado registrą.

PIC16C84 turi ir EEPROM tipo duomenų atmintį. Ši atmintis (64x8 bitų dydžio) nepriklauso RAM adresų sričiai. Kreiptis į ją galima tik per EEDATA ir EEADR registrus. EECON1 registras - tai 5 bitų valdymo registras (trys vyriausieji bitai visada skaitomi kaip “0”). Jo formatas yra toks:

x	x	x	EEIF	WRERR	WREN	WR	RD

Bitai RD ir WR nustato atitinkamai skaitymą ir rašymą (su loginiu “1”). Jie gali būti nustatyti tik programiniu būdu, gesinami – aparatiniu būdu, užbaigus skaitymo (rašymo) operacijas.

WREN bitas leidžia rašyti į duomenų EEPROM (nustatomas programiniu būdu – gesinamas aparatiniu). Programinis rašymo bito WR gesinimo draudimas draudžia įrašymą užbaigti nelaiku. Įjungus maitinimą, WREN bitas gesinamas.

WRERR bitas nustatomas į “1”, jeigu įrašymo operacija nutraukiama NPB signalu \MCLR arba WDT. Rekomenduojama tikrinti šį bitą rašymo metu ir prireikus pakartotinai perrašyti duomenis.

EEIF=”1”, kai įrašymas į duomenų EEPROM yra sėkmingai baigtas (jį privalo gesinti programa). Atitinkantis jį pertraukimo leidimo bitas EEIE yra registre INTCON.

EECON2 registras fiziškai neegzistuoja. Jį skaitant, visuomet nuskaitomi nuliai. Jis naudojamas rašymo į duomenų EEPROM metu.

Pertraukties paraiškų ir “kaukės” registre INTCON reikalingi visi bitai: 5 vyriausieji yra “maskavimo” (leisti-drausti), o trys jauniausieji - pertraukčių paraiškų bitai. Registro INTCON formatas yra toks:

GIE	EEIE	RTIE	INTE	RBIE	RTIF	INTF	RBIF

PIC16C84 darbą gali pertraukti keturi šaltiniai: išorinis pertraukties šaltinis per įvadą RBO\INT (pertraukimo bitas INTF), taimerio RTCC persipildymo signalas (bitas RTIF), rašymo į duomenų EEPROM pabaigos signalas (pertraukimo bitas EEIF yra registre EECON1) ir prievado B skiltyse RB7 – RB4 veikiančių signalų pokyčiai (bitas RBIF). Visos pertrauktys turi tą patį adresą\vektorių – 04h., tačiau kiekvieną pertraukimo šaltinį nurodo atitinkama INTCON registro paraiškos skiltis (pertraukimo požymis arba “vėliavėlė”). Ją gesinti gali tik programa. Pertrauktys gali būti draudžiamos individualiai arba visos kartu su bitu GIE (“0” draudžia visus pertraukimus). GIE bitas automatiškai nustatomas į “0”, įjungus maitinimą, išoriniu signalu \MCLR ir pasibaigus WDT taimerio skaičiavimo laikui. Individualių pertraukčių draudimo bitų pozicijose esantys loginiai nuliai draudžia pertrauktis nuo atitinkamo šaltinio, loginiai vienetai – leidžia .

Pažymėtina, kad procesoriui pradėjus aptarnauti pertrauktį, GIE bitas turi būti gesinamas; tokiu būdu draudžiami kiti pertraukimai. Grįžimo iš pertraukties aptarnavimo paprogramės į pagrindinę programą adresas perduodamas į dėklą, o į programos skaitiklį įrašomas adresas 0004h. Pertraukties šaltinis nustatomas pertraukties apdorojimo paprogramėje, analizuojant INTCON (EECON1) registro turinį (paprogramė taip pat turi pasirūpinti ir pertraukties bito gesinimu). Grįžimo iš pertraukties komanda RETFIE baigia pertraukties paprogramę ir nustato bitą GIE, tuo būdu leisdama kitus pertraukimus, bei grąžina procesorių į pagrindinę programą.

Registras OPTION skirtas rašyti ir skaityti. Jo turinį sudaro valdantieji bitai, nusakantys dažnio daliklio konfigūraciją. Šis aštuonių bitų daliklis gali būti įjungtas prieš taimerį RTCC arba po WDT taimerio. Registro OPTION formatas yra toks:

RBPU	INTEDG	RTS	RTE	PSA	PS2	PS1	PS0

PSA ir PS2 – PS0 bitai nustato vieną iš anksčiau paminėtų įtaisų, su kuriuo dirba daliklis (PSA=0 – RTCC, PSA=1 – WDT), ir dalijimo koeficientą (12 lentelė).

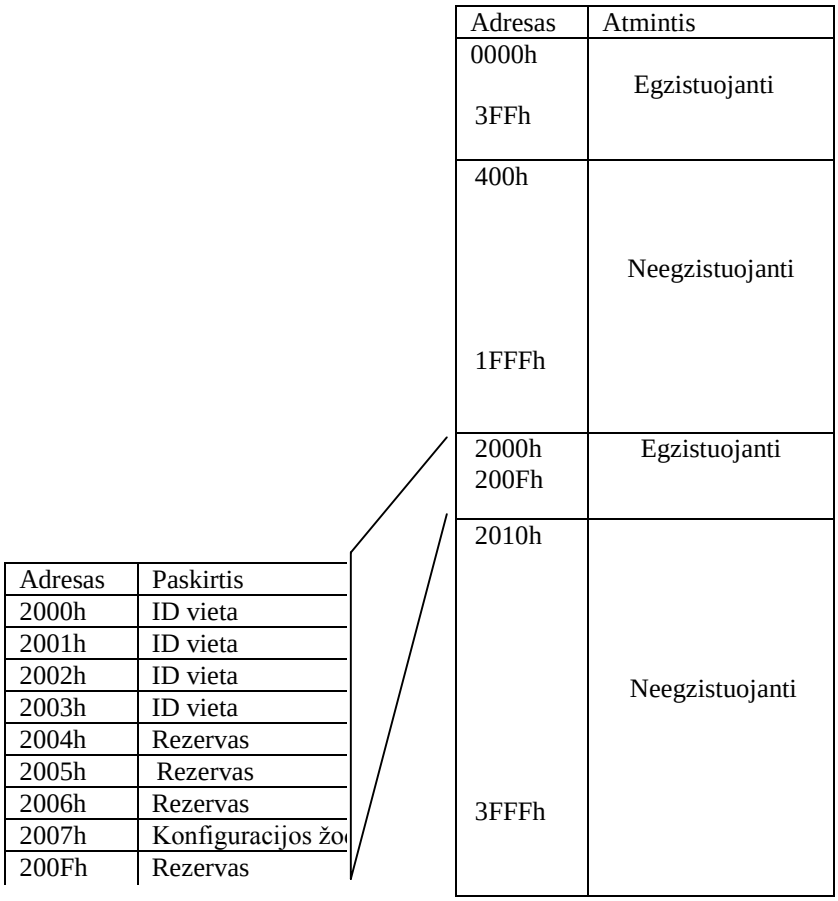
Dalijimo koeficientai 12 lentelė

PS2---PS0	RTCC	WDT
000	2	1
001	4	2
010	8	4
011	16	8
100	32	16
101	64	32
110	128	64
111	256	128

Bitas RTE nustato RTCC reakcijos nuo išorinio įėjimo signalo fronto pobūdį (kai RTE=0, RTCC turinys padidėja vienetu, - nuo teigiamo išorinio signalo fronto, kai RTE=1 – nuo neigiamo). RTS nurodo RTCC signalo šaltinį (0- signalas nuo vidinio generatoriaus, 1- išorinis signalas), INTEDG bitas apibūdina aktyvų išorinio pertraukimo signalo frontą (0 - programos pertraukimą inicializuoja neigiamas INT signalo frontas, 1- teigiamas). Kiekvienas prievado B išvadas turi vidinę apkrovą (apie 100 mka), įjungtą į maitinimo liniją. Ši apkrova automatiškai atjungiama, jei išvadas B užprogramuojamas išvedimui (įjungiant MV maitinimą, visos apkrovos taip pat atjungiamos). RBPU inversinis bitas nusako šios apkrovos prijungimą prie prievado B (RBPU=0 - aktyvi apkrova bus prijungiama pagal prievado B darbo algoritmą, RBPU=1- aktyvi prievado B apkrova yra atjungta).

Programos skaitiklis (PC) naudojamas programos atminties EEPROM ląstelių adresams generuoti. Jis yra 13-os bitų. Tai leidžia adresuoti 8k x 14 bitų atminties talpį. Aštuonios jaunesniosios PC skiltys gali būti įrašytos ir nuskaitytos per registrą PCL, 5 vyresniosios skiltys perduodamos iš registro PCLATCH, kurio adresas 0Ah .

EEPROM tipo programinė atmintis užima 3FFFh adresus. Mikrovaldiklyje fiziškai prieinamos tik 1K atminties ląstelių. Kreipimasis į aukštesnius kaip 3FFh adresus faktiškai yra kreipimasis į tą patį baitą. Taigi, programavimo metu programinės atminties erdvė yra 0000h- 3FFFh, kur pirmoji pusė (0000h-1FFFh) yra programinė atmintis, o antroji pusė (2000h - 3FFFh) - konfigūracijos atmintis, iš kurios tik 2000h- 2007h sritis fiziškai egzistuojanti (22 pav.).



22 pav. PA paskirstymas

Konfigūracijos žodis, įrašomas į EEPROM jos programavimo metu, turi penkis bitus. Šie bitai gali būti užprogramuoti (skaitomi kaip loginis “0”) arba palikti neužprogramuoti (skaitomi kaip “1”), parenkant norimą įtaiso konfigūracijos variantą. Konfigūracijos žodžio adresas yra žemiau programos kodų ir programiniu būdu nepasiekiamas. Jo formatas yra toks:

13	5	4	3	2	1	0	
1		1	CP	PWRTE	WDTE	FOSC1	FOSC0

Atskirų bitų paskirtis aptarta anksčiau, toliau pateiksime kodines jų reikšmes.
Generatoriaus tipo išrinkimo bitai FOSC1, FOSCO: 00 – LP, 01 – XT, 10 – HS, 11 – RC.
WDTE: 0 – WDT draudžiamas, 1 – WDT darbas leistas.
PWRTE: 0 – KGST užlaikymo nebus, 1 – KGST užlaikymas bus.
CP: 0 – apsaugos kodas įjungtas, 1 – apsaugos kodas išjungtas.

Programa, įrašyta į EEPROM, gali būti apsaugota nuo jos nuskaitymo arba pakeitimo, nustačius konfigūracinį bitą CP į "0" (tas pats tinka ir duomenų EEPROM). Nustatytą CP bitą galima ištrinti tik kartu su visos atminties turiniu (pirmiausia bus ištrinta programinė EEPROM, po to duomenų EEPROM ir paskiausiai - CP bitas). Tai patogu autorizuojant sukurtą programinę įrangą.

Be konfigūracijos bito, MV turi dar keturis žodžius, rašomus ir skaitomus tik su programatoriumi. Jie skirti vartotojo identifikavimo kodui ID, kontrolinei sumai ar kitai informacijai saugoti (programa jų rašyti ar skaityti negali). Jeigu yra nustatytas CP bitas, rekomenduojama naudoti identifikavimui tik 7 jauniausius ID bitus, į vyresniuosius įrašant nulius (tada ID žodį galima skaityti).

EEPROM programuajamas nuosekliai bitas po bito, tam tikslui pasitelkus vidinį EEPROM programavimo automatą. Pats vartotojas gali įrašyti suderintą programą į EEPROM, naudodamas tik penkis PIC išvadus (13 lentelė).

Išvadų paskirtis programuojant 13 lentelė

Išvadas	Paskirtis	Tipas	Aprašymas
RB6	CLOCK	I	Taktinis dažnis
RB7	DATA	I\O	Duomenų įvedimas (išvedimas)
MCLR	Vpp	P	Programavimo įtampa (12v-14v)
Vdd	Vdd	P	Maitinimo įtampa (4,5v- 5,5v)
Vss	Vss	P	Bendras

PIC turi 13-os bitų aštuonių lygių aparatinį dėklą. Dėklo sritis nepriklauso nei duomenų, nei programų atminčiai, o jo rodiklis vartotojui nepasiekiamas. Į dėklą įrašomas PC turinys, kai procesorius vykdo komandą CALL arba pereina į pertraukimo aptarnavimo paprogramę. Skaitymą iš dėklo ir jo turinio grąžinimą į PC vykdo grįžimo iš paprogramių komandos RETLW, RETFIE, RETURN. Pažymėtina, kad PCLATH registro turinys nekinta atliekant operacijas su dėklu.

PIC16C84 nustatymo į pradinę būseną procese gali dalyvauti keletas jo blokų, todėl galimi tokie nustatymo variantai:

- nustatymas įjungus maitinimą;
- nustatymas išoriniu signalu \MCLR, esant normaliam darbo režimui;

- tas pats, tik esant SLEEP darbo režimui;
- nustatymas pasibaigus WDT užlaikymui, kai yra normalus darbo režimas;
- tas pats, tik esant SLEEP darbo režimui.

Pažymėtina, kad ne visų registrų turinys pakinta nustačius į pradinę būseną (NPB). Kai kurių turinys po NPB tampa atsitiktinis. Tuo tarpu kitus registrus NPB inicializuoja (išskyrus WDT SLEEP nustatymo variantą – 14 lentelė).

Registrų būsena po NPB 14 lentelė

Registras	Adresas	Įjungus maitinimą	\MCLR (abu atvejai), WDT (norm. režimas)	WDT (SLEEP režimas)
W	-	xxxx xxxx	Uuuu uuuu	uuuu uuuu
INDF	00h	---- ----	---- ----	---- ----
RTCC	01h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	02h	0000h	0000h	PC + 1
STATUS	03h	0001 1xxx	000q quuu	uuuq quuu
FSR	04h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	05h	---x xxxx	---u uuuu	---u uuuu
PORTB	06h	xxxx xxxx	uuuu uuuu	uuuu uuuu
EEDATA	08h	xxxx xxxx	uuuu uuuu	uuuu uuuu
EEADR	09h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCLATH	0Ah	---0 0000	---0 0000	---u uuuu
INTCON	0Bh	0000 000x	0000 000u	uuuu uuuu
TRISA	85h	---1 1111	---1 1111	---u uuuu
TRISB	86h	1111 1111	1111 1111	uuuu uuuu
EECON1	88h	---0 x000	---0 q000	---0 uuuu
EECON2	89h	---- ----	---- ----	---- ----
OPTION	81h	1111 1111	1111 1111	uuuu uuuu

Žymių reikšmės: u – nepakitęs; x – neapibrėžtas; - - fiziškai nerealizuotas bitas, skaitomas kaip “0”; q – reikšmė priklauso nuo sąlygų.

PIC kristalas turi įterptą maitinimo įjungimo detektorių. Paleidimo taimeris (T) pradeda skaičiuoti laiką po to, kai maitinimo įtampa pasiekia 1,2---1,8V dydį. Pasibaigus užlaikymo laikui (apie 72 mks), manoma, kad įtampa pasiekė nominalią reikšmę, ir įjungiamas kitas, kvarcinio generatoriaus stabilizavimo taimeris KGST. Programuojamas konfigūracijos bitas PWRTE leidžia arba draudžia taimerio KGST užlaikymą (naudojant RC generatorių, KGST neįjungiamas). Šis laikas priklauso nuo maitinimo įtampos dydžio, aplinkos temperatūros, be to, atskiri PIC kristalai gali turėti skirtingas jo reikšmes. KGST suskaičiuoja 1024 pradėjusio veikti kvarcinio generatoriaus impulsus (manoma, kad per šį laiką generatoriaus darbo režimas stabilizavosi). Po to įjungiamas išorinio NPB signalo \MCLR laukimo taimeris INLT. Jeigu tokio

signalo nesulaukiama, generuojamas vidinis NPB signalas, ir MV pradeda vykdyti programą (toks MV nustatymo į pradinę būseną organizavimas būtinas tais atvejais, kai sistemoje veikia keletas mikrovaldiklių ir būtina, kad jie visi sinchroniškai pradėtų veikti).

Darbo monitoriaus (Watch Dog) taimeris WDT veikia nepriklausomai nuo kitų MV blokų. Jo RC generatorius, nereikalaujantis jokių išorinių grandinių, veikia netgi tuo atveju, jeigu pagrindinis MV generatorius atjungtas. WDT generuoja signalą NPB, jeigu generacija nėra uždrausta specialiu konfigūracijos bitu WDTE. Nominalus WDT užlaikymo laikas 18 msek. (be daliklio). Jis, kaip ir anksčiau aprašytų taimerių laikas, priklauso nuo maitinimo įtampos ir aplinkos temperatūros. Panaudojus vidinį dažnio daliklį, WDT užlaikymo laiką galima padidinti iki 2,5 sek. (dalijimo koeficientas programuojamas registro OPTION skiltimis – 11 lentelė).

Pagrindinė WDT paskirtis – išvengti avarinių darbo režimų, dėl atsitiktinių išorinių ar vidinių trikdžių "pakibus" MV . WDT naudojimo idėja yra ta, kad reguliariai, pvz., su komanda CLRWDT, yra organizuojamas WDT nustatymas į pradinę būseną iki to momento, kol dar nesibaigė jo užlaikymo laikas ir MV nebuvo nustatytas. Jeigu procesorius atsitiktinai išėjo už programos ribų ir komanda CLRWDT nebus įvykdyta per tam tikrą laiką, procesorius bus nustatomas į pradinę būseną, visi jo registrai bus iš naujo inicializuoti, o visa sistema grįš į pradinę padėtį.

Kita WDT paskirtis – išvesti procesorių iš sumažinto energijos naudojimo režimo, į kurį jis perjungiamas su komanda SLEEP (dirbant šiuo režimu procesorius naudoja tik apie 1mkA srovę). Akivaizdu, kad iš SLEEP režimo procesorių gali išvesti ir išorinis NPB signalas, kokio nors daviklio suveikimo signalas ir pan.

PIC tipo mikrovaldikliams gali būti panaudoti keturių tipų taktinių impulsų generatoriai (TIG, įterpti į kristalą). Vartotojas gali užprogramuoti du konfigūracijos žodžio bitus (FOSC1, FOSC0), pasirinkdamas vieną iš keturių variantų:

- XT – standartinį kvarcinį rezonatorių,
- HS – aukšto dažnio kvarcinį rezonatorių,
- LP – žemo dažnio kvarcinį rezonatorių,
- RC – RC tipo grandinę.

Nustačius pirmuosius tris variantus, prie MV mikroschemos išvadų OSC1 ir OSC2 prijungiamas kvarcinis arba keraminis rezonatorius, o ketvirtojo varianto atveju – RC grandis (generuojamo dažnio stabilumo požiūriu tai nestabiliausias atvejis). Pažymėtina, kad vidinio generatoriaus darbas labai priklauso nuo maitinimo įtampos dydžio, ypač RC grandinės atveju.

2. MPS sąsajos projektavimas. Bendros sąvokos

Į MPS sudėtį, be MP, įeina daug skirtingo sudėtingumo ir veikimo principo įtaisų: programų ir duomenų atmintys, įvairūs įvesties ir išvesties prievadai, keitikliai kodas - analogas ir analogas - kodas, indikatoriai, klaviatūros ir kt. Sąsaja turi užtikrinti ryšį ir informacijos mainus tarp visų įtaisų, įeinančių į MPS sudėtį. Mainai gali vykti skirtingu greičiu. Perduodamos informacijos kiekis taip pat gali būti skirtingas (adreso, komandos kodo skaitymas iš atminties, mygtuko kodo įvedimas iš klaviatūros ir pan.) .

Sąsaja (interface) sprendžia funkcinį, elektrinį ir mechaninį suderinamumą tarp MPS ir objekto .

Funkcinis modulių suderinamumas reiškia tai, kad moduliai , dalyvaujantys informacijos mainuose, generuoja tam tikrus valdančiuosius signalus . Čia turi būti nustatomos mainų taisyklės arba protokolas.

Elektrinis modulių suderinimas užtikrinamas signalų įtampos ir srovės amplitudžių, leidžiamų įėjimo ir išėjimo talpių ir apkrovos varžų suderinamumu.

Mechaninis suderinamumas numato tam tikrų unifikuotų konstrukcijų, plokščių, jungčių ir pan. naudojimą.

2.1. Sąsajų tipai

Projektuojant sąsajas sprendžiami tokie uždaviniai:

- numatoma galimybė sudaryti MPS iš skirtingo sudėtingumo įtaisų, modernizuoti sistemą įjungus naujus arba pakeitus senus įtaisus ir programas;
- numatoma galimybė realizuoti efektyvius informacijos mainus MPS, sudarytoje iš įtaisų su skirtingais duomenų perdavimo greičiais ir veikiančiais sinchroniškai bei asinchroniškai vienas kito atžvilgiu;
- numatoma įvesties ir išvesties operacijų unifikacija, t.y. šioms operacijoms neturėtų daryti įtakos konkrečių išorinių įtaisų specifika;
- numatoma galimybė sutapatinti (ypač didelio našumo sistemose) laike įvesties ir išvesties operacijas bei programos vykdymą;
- numatoma arbitro funkcijų galimybė susikloščius konfliktinėms situacijoms, kai keli įtaisai naudojasi bendra magistrale;
- numatoma informacijos mainų sinchronizacija, elektrinis signalų lygių suderinimas, įtaisų adresų dešifravimas ir t.t.

Šiuos uždavinius spręsti padeda MPS naudojamas magistralinis-modulinis sistemos organizavimo principas, unifikuoti komandų ir duomenų formatai bei pasirinkti architektūriniai sprendimai.

Vieno procesoriaus MPS-e atskiru laiko momentu mainai vyksta tik tarp dviejų modulių. Priminsime, kad MPS naudojami trys skirtingi tokių mainų organizavimo būdai:

- programos valdomi mainai; jie vyksta MP iniciatyva, t.y. procesorius, pasitikrinęs tam tikras sąlygas, vykdo įvesties ir išvesties komandas;

- mainai pagal pertraukties signalus iš periferinių įtaisų (atėjus paraiškai pertraukti, laikinai nutraukiamas programos vykdymas ir įvyksta mainai);

- tiesioginiai mainai su atmintimi DMA (direct memory access); šiuo atveju mainai vyksta išorinio įrenginio iniciatyva, nedalyvaujant MP. Mainus valdo specialus DMA valdiklis.

Visų šių mainų būdų realizavimą, iškylančių konfliktinių situacijų sprendimą turi užtikrinti sisteminė sąsaja. Taigi sąsaja turi turėti adresų ir duomenų perdavimo linijas, pertraukties ir DMA paraiškų priėmimo bei jų patvirtinimo linijas, kitų valdančiųjų signalų perdavimo ir priėmimo linijas, kurias galima suskirstyti į duomenų (DM), adresų (AM) bei valdymo (VM) magistrales.

8 skilčių MPS-e paprastai naudojamos standartinės sisteminės sąsajos. Jos yra unifikuotos savo linijų ir magistralių sudėtimi, prijungimo schemų rinkiniais, informacijos mainų valdymo algoritmų rinkiniais ir pan. Labiausiai paplitusios yra "Microbus" ir "Multibus I" sąsajos. Pirmoji dažniausiai naudojama su MP I8080A, I8085A, Z80A, M6805 ir kt. Pagrindiniai jos parametrai yra šie: procesorių skaičius - 1, leidžiamas magistralių ilgis – 0,3 – 0,6 m, duomenų magistralės skilčių skaičius - 8. Bendras sąsajos linijų skaičius - 34. "Multibus" orientuota naudoti 8 ir 16 skilčių MP (I8080A, I8085A, I8086, I8088 ir kt.). Pagrindiniai parametrai: DM - 16,8, AM - 20, signalinių linijų skaičius - iki 100, procesorių skaičius - bet koks, leidžiamas magistralės ilgis – 0,8 -3 m.

Galima išskirti ir žemesnio lygio sąsajas. Tai MP magistralės sąsajos su įvairiais atminties moduliais, klaviatūra, indikatoriumi ir pan. Esant didelei jungiamų įtaisų prie MPS įvairovei ir sudėtingumui, sąsajos sprendžiamų uždavinių yra gana daug. Todėl stengiamasi paskirstyti sąsajos aparatines ir programines priemones tarp atskirų įtaisų:

- valdymo įtaiso, esančio MP viduje;

- sąsajos įtaiso, esančio tarp MP, iš vienos pusės, ir periferinio įtaiso PĮ - iš kitos;

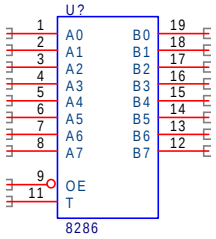
- sudėtingų PĮ, specializuotų kaip valdymo įtaisai (valdikliai). Tokių įtaisų pavyzdžiais (sąsajos DIG) gali būti magistralių formuotuvas, buferiniai registrai, lygiagretūs ir nuoseklūs adapteriai.

2.2. MPS magistralių organizavimas

Prie bendros MPS magistralės gali būti prijungti įvairūs įtaisai. Vienos krypties magistralėje vienas prie jos prijungtų įtaisų yra signalų šaltinis, visi kiti - imtuvai. Jei nenaudojamas tiesioginių mainų režimas, MP AM yra signalų šaltinis, o PĮ AM - imtuvas. Organizuojant vienos krypties magistralę,

jos normaliam darbui užtikrinti pakanka pasiekti jos elementų elektrinį suderinamumą.

Kita situacija susidaro, organizuojant dvikryptę DM. Prie kiekvienos magistralės linijos prijungiami daugelio elementų išėjimai; vieni jų gali turėti aukštą signalo lygį, kiti - žemą. Viso to rezultatas - faktiška magistralės linijos įtampos lygio būsena bus neapibrėžta. Prijungus kelių TTL elementų išėjimus lygiagrečiai, juos galima sugadinti. Todėl, norint suderinti tarpusavio darbą į bendrą DM, naudojami elementai, galintys atsijungti nuo apkrovos. Tokie elementai privalo turėti atviro kolektoriaus arba aukšto impedanso (trijų būsenų - OFF status) išėjimus. Dauguma DIG, naudojamų duomenims perduoti, turi trijų būsenų išėjimus. Jos turi vieną arba keletą



23 pav. Dvikryptis magistralės formuotuvas

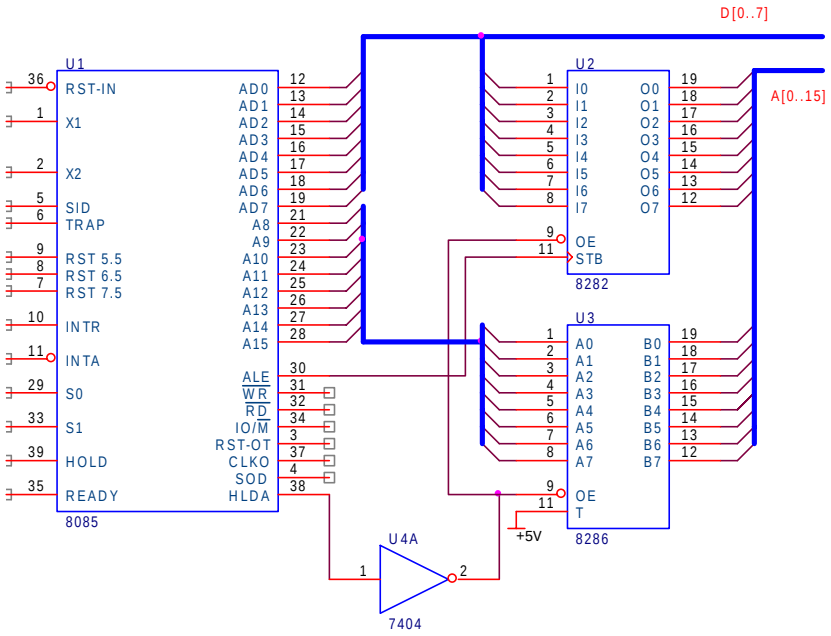
valdančiųjų įėjimų CS/ (chip select - grandyno išrinkimas), CE/ (chip enable - grandyno leidimas) arba OE/ (output enable - išėjimo leidimas). Paprastas tokių IG pavyzdys gali būti magistralės formuotuvas (23 pav.).

Magistralės formuotuvas 8286, davus signalą OE/=1, perveda visus savo išvadus B į aukšto išėjimo impedanso būseną. Signalu T yra valdoma duomenų perdavimo kryptis. Jei T=1, perdavimas vyksta iš A į B, kai T=0 – iš B į A. Toks formuotuvas patogus buferizuojant MP adreso ir duomenų magistrales. 24 pav. parodyta schema, atliekanti adreso ir duomenų magistralių buferizavimą MPS-e su MP 8085A. Formuotuvas 8286 panaudotas adreso magistralei buferizuoti. Maksimali jo apkrovos srovė gali siekti 48 mA. Tokią magistralę galima apkrauti 30-ia TTL ventilių. Iš MP išeinantis DMA patvirtinimo signalas HLDA po inversijos panaudotas formuotuvo 8286 ir adreso jaunesniosios dalies fiksavimo registro 8282 išėjimams pervesti į aukšto išėjimo impedanso būseną. Adreso jaunesnyjį baitą (A0..A7) registre 8282 fiksuoja signalas ALE. Kadangi 8282 išvadus į aukšto išėjimo impedanso būseną gali pervesti nulinio lygio signalas, veikiantis išvade OE, į pastarąjį siunčiamas invertuotas signalas HLDA.

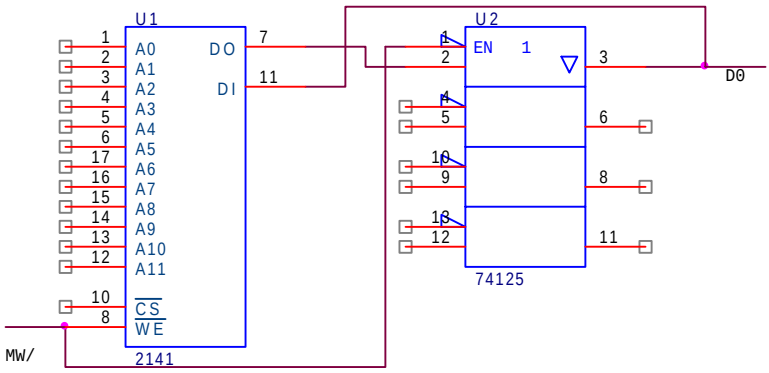
Mikroprocesorinėse sistemose dažnai tenka jungti dvikryptes ir vienos krypties magistrales. Pavyzdys - atminties DIG su atskirais įėjimais (išėjimais), jungiamos prie DM. Vienos krypties ir dviejų krypčių magistralių sujungimą galima atlikti su magistralės formuotuvu, kuriame perdavimo kryptis keičiasi, veikiant valdančiam signalui. 25 pav. parodyta OA su atskirais duomenų įėjimais DI ir išėjimais DO prijungimas prie dvikryptės DM.

2141 yra 1K*1 statinio tipo duomenų atmintis. Ji turi atskirus duomenų įvedimo DI ir išvedimo DO išvadus. Ją prijungti prie DM galima per 74125 tipo buferį su aukšto išėjimo impedanso būsena. Šiuo atveju duomenys iš atminties bus nuskaitomi į DM, kai bus aktyvus signalas CS/ ir pasyvus WR/. Įrašymas vyks tuo metu, kai bus aktyvūs signalai CS/ ir WR/.

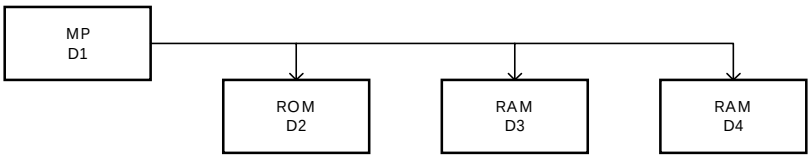
Kad signalai magistralėse nebūtų iškraipyti, jų lygiai ir laikinės jų charakteristikos turi būti elektriškai suderinti. Magistralių elektrinio suderinamumo sąlyga yra leistino apkrovimo užtikrinimas kiekvienam signalo šaltinio išėjimui. Adresų magistralei šie išėjimai yra MP adresų išėjimai (išskyrus DMA režimą, kai adresus formuoja DMA valdiklis, o ne MP). Panagrinėsime pavyzdį sistemos, kurios fragmentas pateiktas 26 pav.



24 pav. Adreso ir duomenų magistralių buferizavimas MPS su MP 8085



25 pav. Atminties DIG su atskirais duomenų įėjimais ir išėjimais prijungimas prie DM



26 pav. Adreso linijų apkrovimas

MP (D1) adresų išėjimai turi sužadinti trijų atminties mikroschemų (D2,D3 ir D4) adresų įėjimus. Sužadinimas bus garantuotas, jei nebus viršyta leistina MP adreso linijos apkrovos srovė ir talpis. Apkrovos srovę reikia nustatyti loginio nulio ir loginio vieneto atvejams.

Loginiam "1" apkrovos srovė bus:

$\sum I_H = 2I_H$ RAM + I_H EPROM = (40+40+40)μA < I_H MP =150μA.

Loginiam "0":

$\sum I_L = 2I_L$ RAM + I_L EPROM = (0,4× 2+0,25)mA < I_L MP = 1,6 mA.

Vadinasi, 26 paveikslo adreso magistralėje, suderinamumo sąlygos yra įvykdytos. Jeigu RAM DIG kiekis padidėtų iki 8, tai šiuo atveju:

$\sum I_H = (40 \times 9)$μA =360 μA > I_H MP =150μA.

$\sum I_L = 8I_L$ OA + I_L PA = (0,4× 8+0,25)mA=3,45mA> I_L MP =1,6 mA.

Srovių suderinamumo sąlyga yra neįvykdyta ir AM būtina buferizuoti su magistralės formuotuvu. Įjungę po MP magistralės formuotuvą 8286, gausime: $\sum I_H$ =360 μA < I_H 8286 =5mA,

$\sum I_L$ =3,45mA< I_L MP =32 mA.

Tikrinant srovių suderinamumą, paprastai reikia įvertinti visų įtaisų, prijungtų prie AM, įėjimo sroves, taip pat ir I/O įtaisų. Jeigu, organizuojant didelės talpos atminties modulius, jų reikalaujamos srovės yra tokios didelės, kad neužtenka ir formuotuvo leistinos apkrovos, būtina imti atminties DIG su mažomis įėjimo srovėmis (pvz., KMDP struktūros).

Be statinės srovinės apkrovos visada reikia patikrinti ir talpinę apkrovą. Anksčiau išnagrinėtame pavyzdyje: C=(5×8+10)pF=50pF< C_0=300pF. Čia C_0=300pF - 8286 leidžiama talpinė apkrova.

Skaičiuojant DM elektrinį suderinamumą, visada reikia nagrinėti du atvejus:

- kai vyksta perdavimas iš MP į magistralę su kitais įtaisais;
- kai vyksta perdavimas nuo kiekvieno atskiro įtaiso į magistralę, prie kurios prijungti kiti įtaisai.

2.3. MPS adresų erdvė ir jos paskirstymas

Mikroprocesorinėje sistemoje kreipimasis į atminties ląsteles, taip pat į išvesties ir įvesties įtaisus vyksta pagal adresinį principą. Kreipiantis į atitinkamą sistemos elementą, MP į AM perduoda šio elemento adresą (numerį). Atminties ląstelių ir potencialiai MP adresuojamų išvesties ir įvesties registrų visuma vadinama adresų erdve.

Priminsime, kad daugelis MP (tarpe jų I8085A, Z80, I8086, I8088 ir kt.) turi savo komandų sistemose atskiras kreipimosi į išvesties ir įvesties įrenginius komandas (IN ir OUT), taip pat atskiras kreipimosi į atmintį komandas (STA, LDA,STAX, LDAX, MOV). Todėl, kreipiantis išvardytomis komandomis į atmintį ir periferinius įrenginius, jų adresų erdvės yra atskirtos (izoliuotos) viena nuo kitos, nors jų adresai ir gali sutapti. Šiam atskyrimui MP I8085 naudojamas specialus valdantysis signalas IO/ $\overline{M}$. Kreipiantis į atmintį, jo reikšmė lygi “0”, o į PĮ - “1”.

Galima organizuoti ir vadinamąjį sutapdintąjį AĮ ir PĮ adresavimo būdą: pavyzdžiui, nesant IN ir OUT komandų arba naudojant tas pačias kreipimosi į atmintį ir periferinius įrenginius komandas. Paprasčiausiu atveju skiriamuoju požymiu, kreipiantis į atmintį arba periferinį įrenginį, gali būti AM vyriausioji skiltis A15. Kai A15=0, tai nurodoma, kad kreipiamasi į atmintį, kai A15=1 - kreipinys į periferinį įrenginį. Likusios AM skiltys adresuoja konkretų išrenkamą įtaisą (ląstelę, prievadą ir pan.). Tuo būdu visa 64K adresų aibė padalijama po 32K į dvi dalis. Šiuo atveju kreiptis į periferinius įrenginius su IN ir OUT komandomis jau negalima. Aišku, kad 64K*8 MP adresuojamą sritį galima dalyti tarp atminties ir periferijos ne po lygiai, naudojant kaip skiriamąjį požymį ne vieną A15, o keletą AM skilčių. Šiame skyrelyje ilgiau apsistosime prie atminties adresų erdvės paskirstymo dalykų.

Projektuojant MPS, jos adresų erdvė paskirstoma tarp DA, PA ir periferijos. Bendros adresų erdvės atveju paskirstymą atlieka sistemos projektuotojas priklausomai nuo darbinės programos apimties, reikalingos atminties duomenims, konstantėms, kintamųjų saugojimui ir t.t. Adresų erdvė paprastai dalijama į puslapius, kurių apimtis proporcinga 2^n (n=1....m, m - AM skilčių skaičius). Tada vyresniąsias AM (m-n) skiltis galima panaudoti puslapių numeriams formuoti, o jaunesniąsias - ląstelėms puslapyje adresuoti.

Puslapio talpį paprastai nulemia naudojamų atminties DIG organizacija. Jeigu pasirenkami DIG su 2K*8 organizacija, adresų erdvę derėtų dalyti į puslapius po 2K adresų kiekvieną. Tada palengvėja adreso dekodavimas. Jeigu naudojami DIG su skirtinga organizacija (pvz., ROM 1k žodžių, RAM - 2k žodžių), tai tikslinga parinkti puslapio talpį pagal mažesnį žodžių kiekį. Nebūtina panaudoti visą atminties erdvę. Visada reikia turėti adresų rezervą, numatant plėtimosi ir reorganizavimo galimybę tolesnio modernizavimo procese. Konkrečių adresų parinkimas priklauso nuo jų dekodavimo technikos. Tarp RAM ir ROM sričių galima palikti ir tuščią "tarpą". Konkrečios talpos RAM ar ROM parinkimą nustato ne programos ilgis, o artimiausios didesnės atitinkamos atminties DIG talpa. Parenkant konkretų DIG, svarbiausia pasirinkimo sąlyga turėtų būti reikalavimai veikimo greičiui, atminties bloko talpai ir maitinimo galios suvartojimui.

Modifikuotos Harvardo struktūros mikroprocesoriai turi atskirą programų ir duomenų atminties erdvę. Pavyzdžiui MCS-51 šeimos MP gali tiesiogiai adresuoti 64K*8 programų ir tiek pat duomenų atminties. Adresavimas čia atliekamas tomis pačiomis 16-a adreso skilčių, bet programų atmintis įjungiama, esant aktyviam signalui PSEN/ .

2.4. Adresų dekodavimas

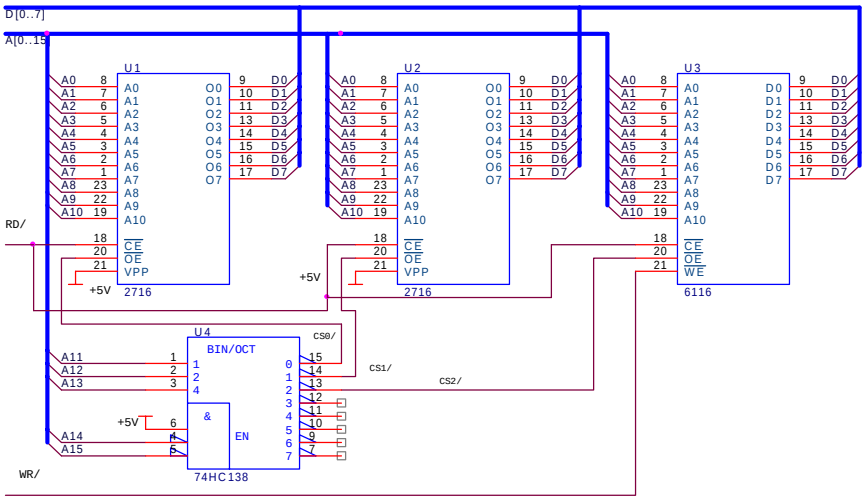
MPS atminties įtaisą paprastai sudaro keletas modulių (keletas IG - PA ir OA). Kreipiantis į atmintį, veikia tik vienas modulis. Modulio išrinkimo signalų (CS/ – chip select) MP dažniausiai neformuoja. Šie signalai formuojami atminties bloko sąsajos mazge iš signalų, perduodamų per AM ir VM.

Puslapių išrinkimo signalai formuojami iš vyresniųjų AM skilčių. Jeigu puslapio talpis 2^n ląstelių, o AM skilčių skaičius – m, tai puslapio numeris dekoduojamas iš (m-n) vyresnių skilčių. Jaunesnės AM skiltys naudojamos ląstelių adresavimui puslapio viduje. I8085 atveju procesorius formuoja AM signalus (A0-A15) ir valdymo signalus $\overline{RD}, \overline{WR}, IO/M$ /. Paprastai atminties DIG darbui reikia adreso (An), išrinkimo (CS/) ir darbo režimo valdymo (RD/,WR/) signalų.

MPS atminties sąsajos uždavinys - suformuoti ir suderinti MP magistralės signalus su atminties IG signalais. Visa tai pademonstruosime pavyzdžiu. Tarkime, kad reikia suprojektuoti 6K baitų talpos atminties bloką. Jame RAM –

2K*8, EPROM – 4K*8. Tada panaudosime dvi EPROM DIG su ultravioletiniu informacijos trynimu - 2716 ir vieną statinės RAM atminties DIG - 6116. Kiekviename iš šių DIG gali tilpti 2K*8 informacijos. Atminties modulio prijungimas prie MPS pateiktas 27 pav.

Reikalinga ląstelė renkama su vienuolika AM signalų A0 - A10. Visi šie signalai lygiagrečiai jungiami prie atitinkamų atminties DIG išvadų. Konkretus atminties IG išrenkamas signalais OE/ (output enable). Jie formuojami dešifratoriumi su 74HC138 iš vyresniųjų AM skilčių A11 - A15. U1 bus aktyvi, kai A11 - A15 skiltys bus 00000 būsenoje, U2 - 00001, U3 – 00010. Tuo būdu EPROM užims adresų sritį 0000-0FFFh (U1 – 0000-07FFh, U2 – 0800-0FFFh), o RAM - 1000-17FFh. Viso bloko adresų sritis - 0000-17FFh.



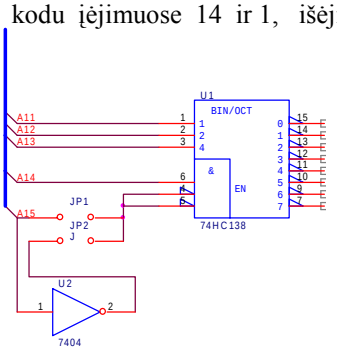
27 pav.Atminties bloko prijungimas

Jeigu MPS reikia didesnės duomenų ir programų atminties apimties, galima imti didesnį vieno ir kito tipo atminties DIG skaičių. Šiuo atveju galima panaudoti ir likusius dešifratoriaus išvadus. Taip prijungtu dešifratoriumi galima adresuoti 16K*8 atminties, įvairiai ją konfigūruojant.

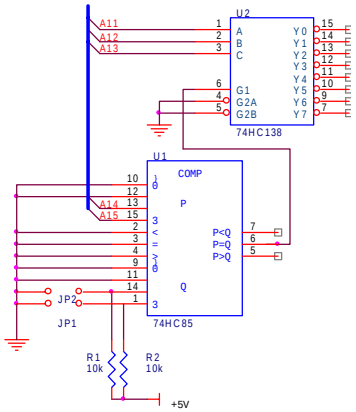
Dažnai reikia perpaskirstyti duomenų ir programų atminties sritis. Tada patogu naudoti trumpiklius ir papildomą loginę schemą (28 pav.) arba skaitmeninį komparatorių (29 pav.).

Kaip parodyta 28 pav., trumpikliais JP1 ir JP2 galima keisti atminties bloko adresų sritį. Jeigu trumpai sujungiame JP2 (JP1 tuo tarpu atviras), tai bloko adresų sritis tampa 8000-97FFh. Adresų sritį galima perskirstyti ir su

skaitmeniniu komparatoriumi (29 pav.). Bloko adresų zonai nustatyti čia imami trumpikliai JP1 ir JP2. Sutapus kodui, veikiančiam IS U1 įėjimuose 13 ir 15, su trumpikliais nustatomu kodu įėjimuose 14 ir 1, išėjime 6 formuojamas



28 pav. Atminties bloko adresų perpaskirstymas komutuojant A15



29 pav. Atminties bloko adresų perskirstymas su skaitmeniniu komparatoriumi

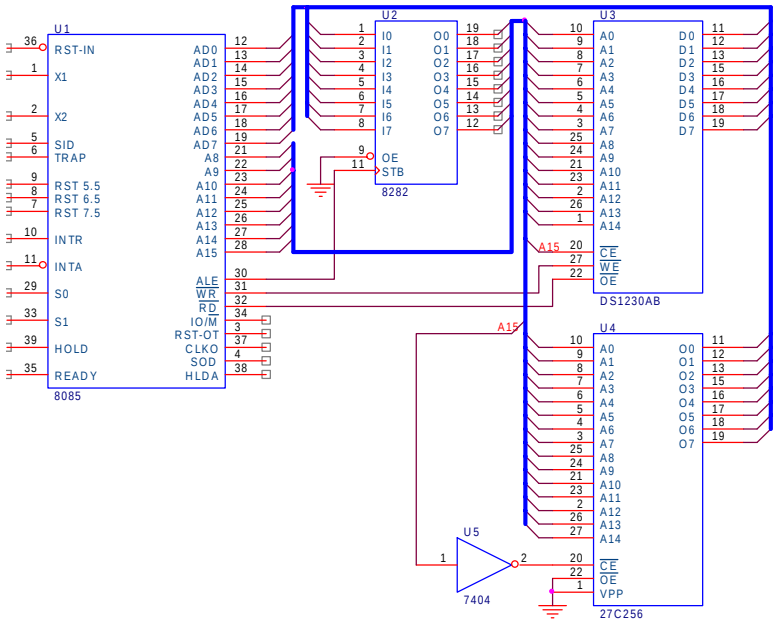
loginis vienetas, kuris sąlygoja dešifratoriaus U2 darbą. Tokiu būdu trumpikliais JP1 ir JP2 galima pasirinkti vieną iš keturių bloko adresų srities variantų (15 lentelė).

Adresų srities parinkimo variantai 15 lentelė

Trumpikliai	Bloko adresų sritis

Su JP1 ir JP2	0000 – 17FFh
Su JP2 ir be JP1	4000 – 57FFh
Su JP1 ir be JP2	8000 – 97FFh
Be JP1 ir JP2	C000 – D7FFh

Naudojant didesnės talpos atminties DIG, adresų dekodavimas supaprastėja. 30 pav. parodyta MPS su 8085A, kurioje yra 32K*8 RAM ir 32K*EPROM, principinė schema. Čia RAM ir EPROM dešifravimui panaudota vyriausioji MP adresų magistralės skiltis A15. Taigi EPROM adresų sritis yra 0000-7FFFh, o RAM 8000-FFFFh.

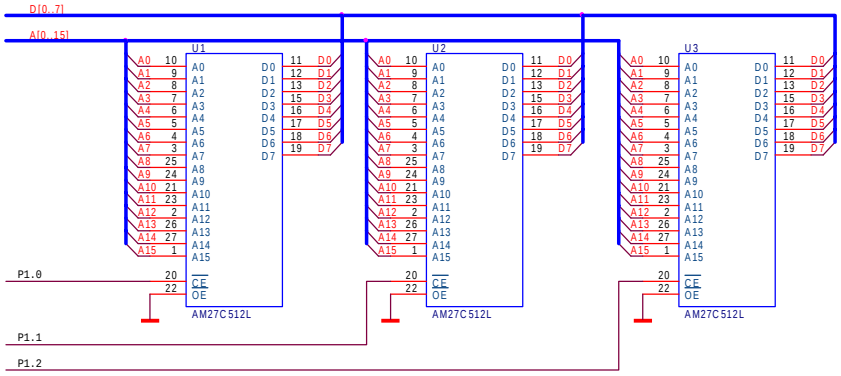


30 pav. Atminties prijungimas prie MP 8085

3.5. Adresų erdvės praplėtimas

MPS adresų erdvė, nustatoma AM skilčių skaičiumi, kai kuriais atvejais gali būti per maža. AM skilčių skaičių nusako MP architektūra. Tačiau yra įvairių MPS adresų erdvės padidinimo metodų. Vienas jų - atminties bankų formavimo metodas. Čia banko terminu nusakomas atskirtas nuo kitų atminties blokas, paprastai turintis ir PA, ir DA. Metodo esmę iliustruoja 31 pav.

Sistema turi M nepriklausomų atminties blokų (šiuo atveju du blokus NVSRAM po 64K*8, kiekvieną jų sudaro du DIG DS1230 po 32K*8 talpos kiekvienas), prijungtų prie sistemos AM ir DM. Kiekvienas blokas turi darbo leidimo įėjimą CE/. Priklausomai nuo jo lygio, blokas prijungiamas arba atjungiamas nuo DM, pervedant jį į aukšto impedanso būseną. Informacija apie tai, kokį bloką reikia prijungti prie DM, perduodama per MV I8051 prievado P1 išėjimą P1.0; kurį iš DIG tame banke reikia prijungti prie DM, nustato vyriausioji AM skiltis A15. Taip galima išplėsti MPS atminties resursus, kai jos adreso linijų skaičius yra ribotas. Šiuo atveju tiesiogiai adresuojama 64K*8, o netiesiogiai 128K*8.



31 pav. Adersų erdvės išplėtimas

3.6. Programų atminties projektavimas

PA naudojama MP programos tekstui, konstančių bei įvairių lentelių turiniui saugoti. Šiose atmintyse informacija turi būti išsaugoma ir išjungus MPS maitinimą. Šiam tikslui galima panaudoti PA su gamybos metu įrašytu turiniu – ROM, vieną kartą pas vartotoją įrašomu turiniu – PROM-OTP (one time programmable), pastovias atmintis su informacijos trynimu ultravioletiniais spinduliais – EPROM ir pastovias atmintis su informacijos trynimu elektriniais impulsais – EEPROM ir FLASH. Įterptinėse sistemose labiausiai paplitę yra EPROM ir EEPROM. Toliau pavyzdžiais išsiaiškinsime atminties blokų projektavimo su šiomis atmintimis metodiką. (Atminties turinys į šių tipų atmintis įrašomas specialiais įrenginiais – programatoriais.)

EPROM sutinkamos įvairios talpos ir gaminamos įvairiomis elektroninėmis technologijomis. Įterptinėse sistemose dažniausiai norima pasiekti mažiausią energijos suvartojimą, todėl rekomenduotina naudoti KMDP (CMOS-Complementary Metal Oxide Semiconductor, žymimos su raide "C") technologija gaminamas atmintis, pasižyminčias maža vartojama galia. 16 lentelėje pateikti kai kurie EPROM tipai ir jų pagrindinės charakteristikos. 32 pav. pateiktos tipinės EPROM darbo laiko diagramos.

EPROM tipai ir jų pagrindinės charakteristikos 16 lentelė

Eil Nr	Tipas	Talpa	Išrinkimo laikas (Tacc)	Maitinimo srovė statiniu režimu	Maitinimo srovė dinaminiu režimu	Apkrovos srovė (mA)/ įėjimų talpa (pF)/ išėjimų talpa (pF)
1	2732A	4K*8	200-450 ns	35 mA	100mA	2,1 / 4 / 8
2	2764A	8K*8	180-200 ns	40 mA	75 mA	2,1 / 4 / 8
3	27C64	8K*8	150 ns	0.1 mA	20 mA	2,1 / 6 / 12
4	27128A	16K*8	200 ns	40 mA	100 mA	2,1 / 4 / 8
5	27C128	16K*8	150 ns	0.1 mA	30 mA	2,1 / 6 / 12
6	27256	32K*8	170 ns	50 mA	125 mA	2,1 / 4 / 8
7	27C256	32K*8	120 ns	0.05 mA	20 mA	2,1 / 6 / 12
8	27512	64K*8	170 ns	40 mA	125 mA	2,1 / 4 / 8
9	27513	4*16K*8	170 ns	40 mA	125 mA	2,1 / 4 / 8
10	27010	128K*8	200 ns	50 mA	150 mA	2,1 / 4 / 8
11	27210	64K*16	150 ns	40 mA	175 mA	2,1 / 4 / 8
12	27011	8*16K*8	200 ns	50 mA	150 mA	2,1 / 4 / 8
13	AT27C020	256K*8	85-150 ns		25 mA	2,1 / 4 / 8
14	AT27C2048	128K*8	70 ns	0,1 mA	35 mA	2,1 / 4 / 8
15	AT27C040	512K*8	80 ns	0,1 mA	30 mA	2,1 / 4 / 8
16	AT27C4096	256K*16	85 ns	0,1 mA	40 mA	2,1 / 4 / 8
17	AT27C080	1M*8	100 ns	0,1 mA	40 mA	2,1 / 4 / 8
16	AT27C4096	256K*16	85 ns	0,1 mA	40 mA	2,1 / 4 / 8

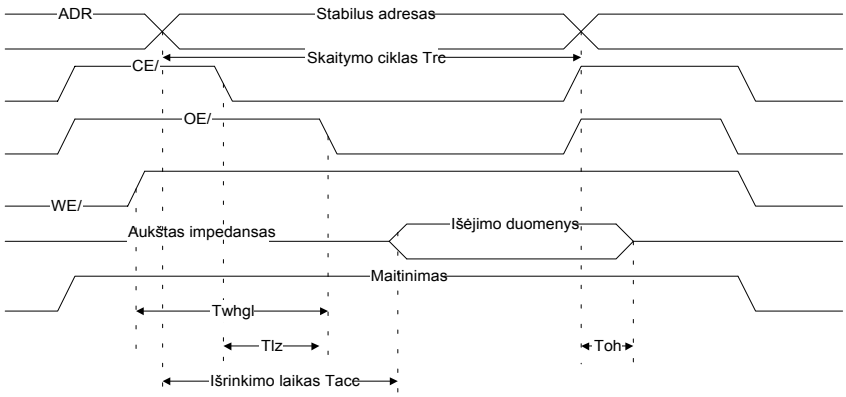
Matome, kad pirmiausia turi būti suformuotas EPROM ląstelės adresas ADR. Kol jis stabilus, turi būti suformuotas aktyvus signalas CE/ (chip

enable) ir išėjimus atidarantis OE/ (output enable). Tada, jeigu signalas WE/ yra pasyvus (skaitymo režimu), po tam tikro laiko, vadinamo išrinkimo laiku (Tacc – access time), DIG išėjimuose pasirodo duomenys. Čia papildomai turi būti laikomasi dar dviejų sąlygų. Turi praeiti tam tikras išsaugojimo laikas tarp pasyvaus rašymo signalo pradžios ir aktyvaus signalo OE/ pradžios – Twhgl ir laikas tarp aktyvaus signalo CE/ pradžios ir aktyvaus signalo OE/ pradžios - Tlz. Nuskaityti duomenys pakeitus adresą dar išlaikomi laiko intervalą Toh.

Minėti laiko intervalai pateikiami EPROM žinynuose. Svarbiausias jų yra Tacc (access time). Būtent šis laiko intervalas nulemia atminties greitaveiką, būtent pagal šį parametrą reikia parinkti DIG konkrečiai MPS su pasirinktu MP.

Dabar pagal MP 8085A skaitymo darbo laiko diagramas nustatysime reikalavimus EPROM charakteristikoms.

33 pav. pateiktos 8085A laiko diagramos skaitymo iš atminties (duomenų arba programų) režimu. Brėžinyje parodyti 4 MP taktai - T1,T2, laukimo taktas Tw ir T3. Pirmame takte MP per išvadus A8-A15 pateikia vyresnįjį adresų baitą, o per AD0-AD7 – jaunesnįjį. Pastarasis išoriniame registre signalu ALE užfiksuojamas (27 pav.). Tada atminties ląstelės adresas į EPROM išlaikomas laiko intervalą Tacc eeprom. Takte T2 formuojamas skaitymo signalas RD/, kuris išlaikomas laiko intervalą Trd. Šio intervalo trukmė priklauso nuo to, ar MP gauna aktyvų signalą READY iš atminties įrenginio, ar ne. Jeigu šio signalo negauna, tai jo trukmė yra 300 ns, kai MP sinchronizacijos dažnis

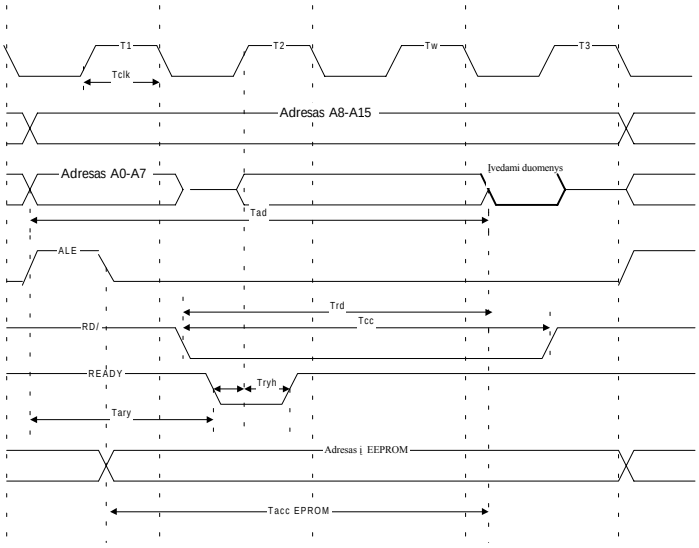


32 pav. Tipinės EPROM darbo laiko diagramos

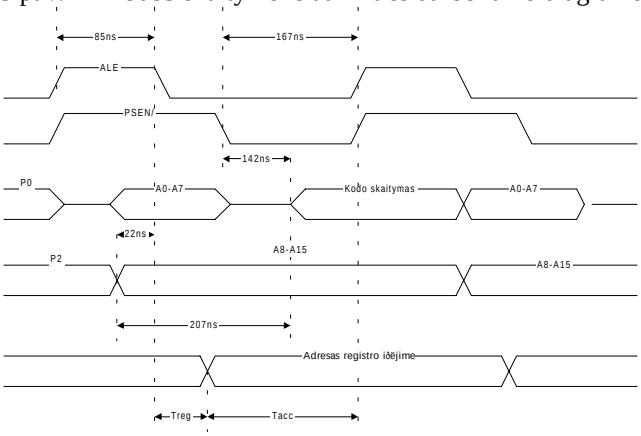
6,144 MHz. Jeigu gauna, tai jo trukmė priklauso nuo signalo READY trukmės (pastarojo trukmė gali būti labai didelė, bet kartotinė MP periodų skaičiui).

Taigi iš šių darbo laiko diagramų matome, kad esant nepageidautiniems MP laukimo taktams, reikalingas EEPROM išrinkimo laikas neturi viršyti 300 ns. Jeigu jis didesnis, tai reikia atminties bloke formuoti READY signalo generatorių, kuris "užlaikytų" MP tol, kol EEPROM pateiks į magistralę duomenis arba reikia sumažinti MP sinchronizacijos dažnį. Tokį generatorių galima sukonstruoti, pasitelkus monostabilų multivibratorių, bet, kaip rodo 16 ir 17 lentelių duomenys, daugelis EEPROM turi mažesnį išrinkimo laiką negu 300 ns, ir galima apsieiti be laukimo taktų formavimo schemos. Tokiu atveju pasiekiamas maksimalus MPS našumas, nes išnyksta laukimo taktas Tw. Paprastai EEPROM yra lėtesnės už RAM, todėl greitaveiksmėse MPS, įjungus maitinimą, EEPROM turinys, naudojant laukimo taktus, kraunamas į RAM,o programa vykdoma iš RAM. Akivaizdus tokių MPS pavyzdys yra personalinis kompiuteris, kurio BIOS (basic input output system) saugomos EPROM-e arba EEPROM-e turinys įjungus maitinimą kraunamas į DRAM (dynamic random access memory.)

34 pav. pateiktos MP 8051 skaitymo iš išorinės programų atminties darbo laiko diagramos, kai MP sinchronizacijos dažnis 16 MHz. Duomenų adresas fiksuojamas išoriniame registre pagal krintantį signalo ALE/ frontą. Išoriniame fiksuojančiame registre gaunamas tam tikras signalo plitimo vėlinimas Treg. Šis laiko intervalas priklauso nuo pasirinkto registro tipo. Jeigu naudosite 74HCT373 tipo registrą, tai Treg bus 15-30ns, 74AC373 tipo registrui šis laiko intervalas sutrumpėja iki 10-13 ns. Kai signalas PSEN/ yra aktyvus ir praeina 142 ns po jo neigiamo fronto, prasideda duomenų įvedimo į MP procesas. Išorinė programų atmintis jau turi kaupti duomenis į duomenų magistralę. Atminties išrinkimo laikas Tacc=216ns-Treg. Taigi būtent pagal šį parametrą reikia parinkti išorinės programų atminties išrinkimo laiką. Naudojant lėtesnį išorinį registrą, jis turi būti mažesnis, ir atvirkščiai - jei registras greitesnis, galima panaudoti lėtesnį ir pigesnį atminties DIG.



33 pav. MP 8085 skaitymo iš atminties darbo laiko diagramos



34 pav. MP 8051 išorinės programų atminties skaitymo laiko diagramos

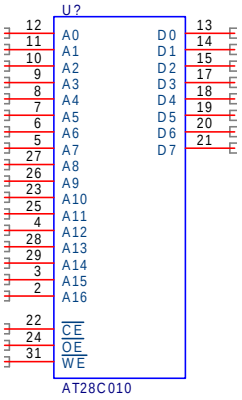
Labai patogi programų saugojimo priemonė yra EEPROM, arba FLASH. Jos pagrindinė ypatybė ta , kad informacija trinama ne ultravioletine šviesa, kaip EPROM, bet elektriniais signalais. Programavimą galima atlikti net neišėmus šių DIG iš MPS plokštės (in-circuit programming). Tai labai patogu rekonfigūruojant MPS darbo algoritmą, keičiant konstantes kokio nors prietaiso kalibravimo metu, saugant besikeičiančią tarnybinę informaciją ir pan.

Dirbant skaitymo režimu šių DIG darbo laiko diagramos nedaug skiriasi nuo EPROM, todėl jas galima vieną kita pakeisti (dažnai netgi išvadų numeracija sutampa). 17 lentelėje pateiktos kai kurių EEPROM charakteristikos.

EEPROM tipai ir jų pagrindinės charakteristikos 17 lentelė.

Nr	Tipas	Talpa	Išrinkimo laikas/baito įrašymo laikas	Maitinimo srovė statiniu režimu	Maitinimo srovė dinaminiu režimu	Apkrovos srovė (mA)/ įėjimų talpa (pF)/ išėjimų talpa (pF)
1.	AT28C16	2K*8	150 ns/0.2ns	0,1 mA	30 mA	2,1 / 4 / 8
2.	AT28C64	8K*8	120 ns/0.2ns			2,1 / 4 / 8
3.	AT28C256	32K*8	150 ns / įrašoma po 64 baitus per 3 ms	0,2 mA	50 mA	2,1 / 4 / 8
4.	AT28C010	128K*8	120 ns / įrašoma po 128 baitus per 10 ms	0,2 mA	40 mA	2,1 / 4 / 8
5.	AT28C040	512K*8	150 ns / įrašoma po 256 baitus per 10 ms		80 mA	2,1 / 4 / 8
6.	AT28H64B	8K*8	55 ns / įrašoma po 64 baitus per 3-10 ms	0,1 mA	30 mA	2,1 / 4 / 8
7.	AT28HC256	32K*8	70 ns / įrašoma po 64 baitus per 3-10 ms	3 mA	80 mA	2,1 / 4 / 8

Be kitų charakteristikų, pateikta informacija apie bloko dydį ir informacijos rašymo trukmes. Čia informacija rašoma ir trinama blokais. Gaminamos EEPROM ir su nuosekliu informacijos įrašymu ir skaitymu. Dažniausiai naudojamos dvilaidės (I2C) ir trilaidės (SPI) prijungimo schemos.



35 pav. EEPROM 28F010

Aptarkime 256K*8 talpos Intel firmos EEPROM 28F010 išvadus, jų paskirtį ir programavimą (žr. 35 pav.)

17 adreso linijų A0-A16 skirta atminties ląstelės adresui nustatyti skaitymo ir rašymo režimu. Linijos D0-D7 skirtos duomenų baitui įrašyti ir skaityti lygiagrečiai. Signalu CE/ aktyvinama DIG vidinė logika, įėjimo buferiai, adreso dešifratorius ir skaitymo stiprintuvai. Signalu OE/ aktyvinamas DIG išėjimo buferis skaitymo režimu. WE/ yra rašymo signalas. Jo kylančiu frontu fiksuojamas adresas, o krintančiu - užfiksuojami duomenys. DIG maitinamas iš Vcc=+5V įtampos šaltinio. Šis DIG turi dar vieną trynimo (programavimo) maitinimo įtampą Vpp. DIG darbo režimai išvardyti 18 lentelėje.

28F010 darbo režimai 18 lentelė

Operacija	Vpp	A0	A9	CE/	OE/	WE/	D0-D9
Skaitymas	5V	A0	A9	Iki 0,8V	Iki 0.8V	5V	Skaitomi duomenys
Išėjimas uždraustas	5V	X	X	Iki 0,8V	5V	5V	Aukšto impedanso būsena
Statinis darbo režimas	5V	X	X	5V	X	X	Aukšto impedanso būsena
Identifikacija	5V	iki 0.8V	11.5-13V	Iki 0,8V	iki 0.8V	5V	89h
Rašymas	11,4-12,6V	A0	A9	Iki 0,8V	5V	Iki 0,8V	Įrašomi duomenys

Lentelės duomenys rodo, kad skaitymas vyksta esant maitinimo įtampai, aktyviems CE/ ir OE/ signalams ir įėjime WE/ veikiant aukštam loginiam

lygiui. DIG duomenų išėjimai bus aukšto impedanso būsenoje, jeigu įėjime OE/ bus pasyvus loginis lygis. Kai įėjime CE/ bus pasyvus loginis lygis, DIG bus statiniame darbo režime. Šiuo atveju bus vartojama minimali maitinimo galia (maitinimo srovė tik 50 mkA). Jeigu įėjimuose CE/ ir OE/ bus aktyvūs loginiai lygiai, o į adresų liniją A0 bus paduotas loginis nulis bei į A9- +12V įtampa (kai WE/ pasyvus), tai duomenų išėjimuose gausime gamintojo identifikacijos kodą (pvz.,firmai Intel-89h). Informacijos įrašymas nustatytu adresu vyksta tuo metu, kai maitinimo įtampa pakeliama nuo 5 iki 12 voltų, CE/ ir WE/ yra aktyvūs , o OE/ - pasyvus.

EEPROM 28F010 komandų sąrašas 19 lentelė

Komanda	Ciklų skaičius	Operacija 1	Adresas 1	Duomenys 1	Operacija 2	Adresas 2	Duomenys 2
Skaitymas	1	Rašymas	X	00h			
Identifikacijos kodo skaitymas	3	Rašymas	X		Skaitymas		Nuosekliai pateikiami gamintojo ir įtaiso įdentifikacijos kodai
Trynimas	2	Rašymas	X	20h	Rašymas	X	20h
Ištrinto turinio tikrinimas	2	Rašymas	X	A0h	Skaitymas	X	FFh
Programavimas	2	Rašymas	X	40h	Rašymas	Nustatyti adresai	Reikalingi duomenys
Įrašyto turinio tikrinimas	2	Rašymas	X	C0h	Skaitymas	Nustatyti adresai	Būtini duomenys
Stop	2	Rašymas	X	FFh	Rašymas	X	FFh

DIG turinio trynimas ir programavimas atliekamas per vidinį komandų registrą, kai įėjime Vpp veikia 12V įtampa. Įrašymas į šį registrą vyksta signalo WE/ krintančiu frontu. Atskirų firmų EEPROM programavimas skiriasi. Firmos Intel EEPROM DIG 28F010 komandų sąrašas pateiktas 19 lentelėje.

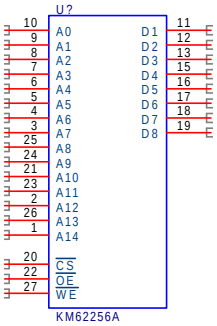
2.7. Duomenų atminties blokų projektavimas

Duomenų (operatyvioji) atmintis naudojama laikinam kintamųjų saugojimui. Duomenų atmintys skirstomos į statines (su nuosekliu arba atsitiktiniu adreso formavimu) ir dinamines (DRAM). Dažniausiai jose informacija prarandama išjungus maitinimo įtampą, bet naudojamos ir energonepriklausomos statinės duomenų atmintys (NVSRAM - nonvolatile random access memory), kurios gaminamos KMDP technologija, naudojančia itin mažą maitinimo galią. NVSRAM korpuse sumontuota ličio baterija, užtikrinanti maitinimą ne trumpesniam kaip dešimties metų laikotarpiui. Statinės atmintys (jas dar vadina registrinio tipo atmintimis) su nuosekliu

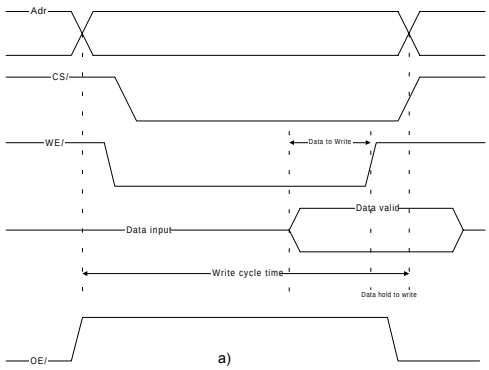
adreso formavimu yra dviejų tipų – pirmas įrašytas skaitomas pirmas (FIFO – first input first output) ir pirmas įrašytas skaitomas paskutinis (FILO – first input last output). Statinėse ir dinaminėse atmintyse su atsitiktiniu adreso formavimu reikalingos ląstelės adresas gali būti tiekiamas bet kokia (random) atsitiktine tvarka.

Toliau aptarsime duomenų atminties DIG valdymo signalus, jų darbo laiko diagramas ir pateiksime jų prijungimo pavyzdžius prie MP 8085A ir MV 8051. 36 pav. vaizduojamas KMDP tipo statinės RAM KM62256A DIG žymėjimas principinėje schemoje.

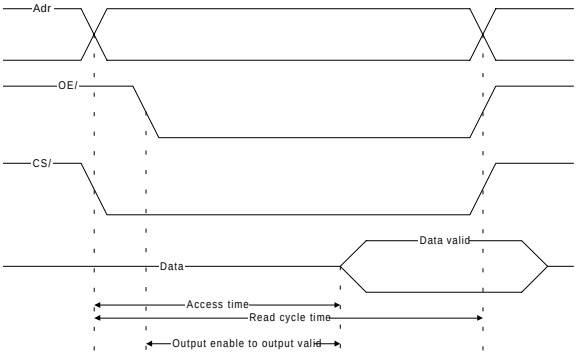
Šis DIG yra 32K*8 talpos, todėl jo atminties ląstelėms adresuoti naudojama penkiolika adreso linijų (A0-A14). Duomenų linijos D1-D8 naudojamos dvikrypčiams mainams (skaitymui ir rašymui). Signalas CS/ skirtas DIG išrinkimui, OE/ - skaitymui, WE/ - rašymui. Kai išrinkimo signalas pasyvus, DIG veikia pasyviu režimu ir vartoja tik apie 500mkA srovę. Dinaminiu darbo režimu DIG vartojimo srovė išauga iki 40 mA. 37a pav. pateiktos skaitymo iš atminties darbo laiko diagramos. Pradžioje turi būti suformuotas adresas. Jis turi būti išlaikomas ne mažesnį laiko intervalą kaip skaitymo ciklo trukmė (read cycle time). Šis laiko intervalas pateikiamas žinynuose. To paties tipo atminties DIG jis gali svyruoti. Po adreso turi eiti skaitymo signalas OE/, o praėjus išrinkimo laikui (access time), išėjime pasirodys nustatytu adresu nuskaityti duomenys (data valid). Žinynuose pateikiamas ir dar vienas parametras – laiko intervalas nuo skaitymo signalo pradžios iki duomenų pasirodymo (output enable to data valid). Taigi, duomenys išėjime pasirodys, jeigu bus laikomasi šių sąlygų: duodamas ir išlaikomas nustatytą laiką adresas ir skaitymo signalas, o DIG išrinkimo signalas visą skaitymo ciklą aktyvus. Kai CS/ pasyvus, duomenų linijos bus aukšto išėjimo impedanso būsenoje.



36 pav. Statinio tipo RAM KM62256A



a)

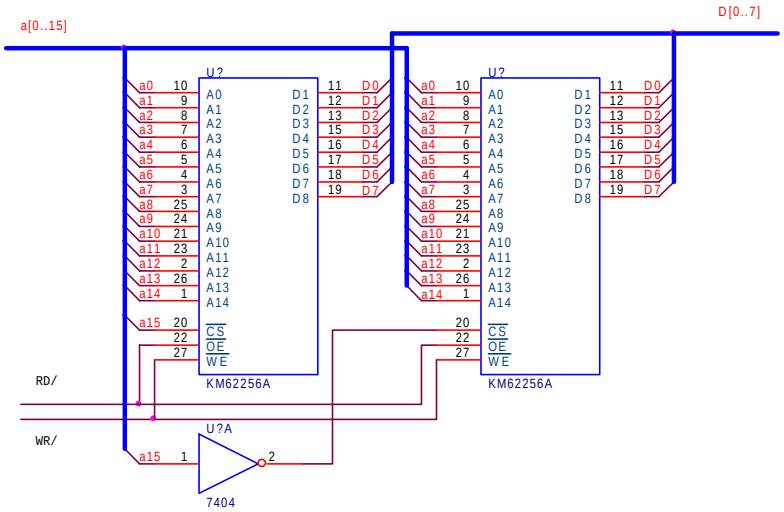


b)

37a ir b pav. Rašymo ir skaitymo į duomenų atmintį KM62256 laiko diagramos

Rašymo laiko diagramos pateiktos 37a pav. Rašymo ciklas prasideda davus ląstelės, į kurią norima įrašyti baitą, adresą. Skaitymo signalas OE/ turi būti pasyvus (loginis vienetas). Suformavus adresą, duodami išrinkimo CS/, rašymo WR/ signalai ir duomenys. Duomenų fiksavimas atminties ląstelėje vyksta pagal teigiamą WR/ signalo frontą, tačiau įrašomi duomenys turi būti stabilūs tam tikrą laiko intervalą (data to write). Jie taip pat turi būti išlaikomi stabilūs dar tam tikrą laiką po teigiamo rašymo signalo fronto (data hold to write).

38pav. pateikiama 64K*8 talpos duomenų atminties bloko MPS su MP 8085A principinė schema.



38 pav. 64K*8 duomenų atminties principinė schema

3. Įvesties ir išvesties mazgų projektavimas

Pasitelkus sąsajos schemą (interface), MP susiejamas su periferiniais įrenginiais, davikliais ir valdymo mechanizmais. Per juos priimamos ir perduodamos komandos bei duomenys. Sąsajos įtaisai nėra vien tik aparatinės priemonės. Tai techninių, mechaninių ir organizacinių priemonių visuma, skirta MP sujungti su aplinka.

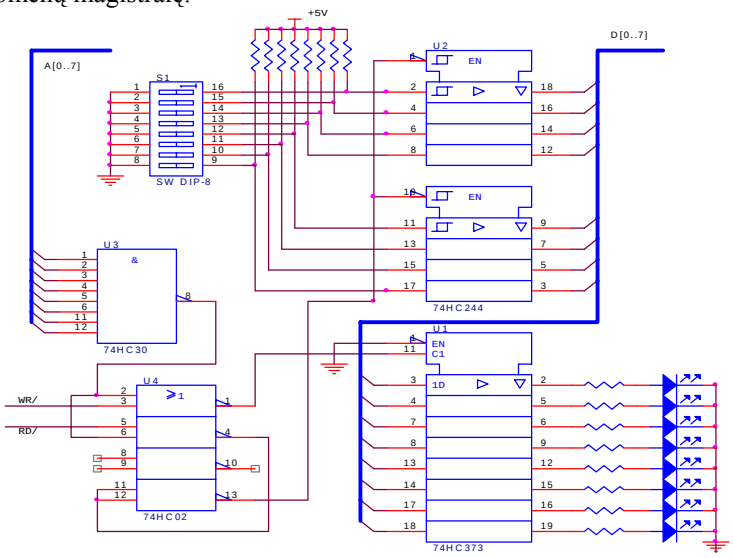
Mainai tarp MP ir periferijos vyksta per duomenų magistralę, valdymui panaudojant valdymo magistralės signalus ir dalį adresų magistralės signalų. Dažniausi yra nuoseklūs ir lygiagretūs mainų būdai. Jie gali būti realizuojami būsenos apklausos (pooling), programos pertraukties (interrupt) ir tiesioginių mainų su atmintimi (direct memory access) režimais.

3.1. Sąsajos adreso dešifravimo technika

Nesudėtingiems lygiagretiems mainams organizuoti tinka paprasčiausi registrai ir buferiai. 39 pav. pavaizduotas sąsajos mazgas su 8 diskretiniais įėjimais ir 8 diskretiniais išėjimais. IG U3 atlieka adreso (FFh) dešifratoriaus

funkciją, U2 – 8 jungiklių bloko S1 būsenos įvesties buferio funkciją, U1 – 8 bitų fiksuojančio registro funkciją.

Jungiklių S1 būseną galima nuskaityti su įvedimo komandomis. MP 8085 atveju tai būtų IN 0FFh. Įvykdę šią komandą, akumuliatoriuje turėsime jungiklių bloko kodą. Įvykdę komandą OUT 0FFh, į registrą U1 įrašysime kodą, kurio turinį indikuos šviesos diodai. IG U4 panaudotas registro fiksavimo signalui C1 formuoti, kai valdymo magistralėje yra rašymo signalas WR/ , o adresų magistralėje kodas FFh. Be to, čia formuojamas ir buferio U2 valdymo signalas, kai valdymo magistralėje yra signalas RD/, o adresų magistralėje kodas FFh. Šiuo atveju buferio išėjimai iš aukšto išėjimo impedanso būsenos pervedami į aktyvią būseną, ir jungiklių būsena patenka į duomenų magistralę.



39 pav. 8 bitų lygiagrečios sąsajos mazgas

Minėtu atveju duomenų mainus inicijuoja MP. Jo programa nustato buferio apklausos ir išvesties į registrą periodiškumą.

Panagrinėkime kiek sudėtingesnį atvejį, kai mainus inicijuoja išorinis įrenginys (40 pav.). Sakysim reikia suskaičiuoti TTL lygių išorinius įvykius. Paimsime programuojamą laiko intervalų taimerį 8253 (U1), būsenos ventilį U2, adresų dešifratorius U3,U4 ir U5 , bei skaitymo signalo formuotuvą U6.

Įvykiams skaičiuoti tinka ir programuojamas taimeris arba paprasčiausias ventilis su aukšto impedanso išėjimu (U2), kurio išvadas prijungtas prie duomenų magistralės linijos D1. Ventilis tam tikru momentu turi būti

atidaromas signalu, formuojamu adreso dešifratoriumi (U4 ir U5), kurio išėjime signalas yra aktyvus, kai jaunesniajame adresų magistralės baite yra kodas 7Fh, ir loginiu ventiliu U6, kuris formuoja ventilio atidarymo signalą, esant aktyviems dešifratoriaus išėjimo signalui ir valdymo magistralės signalui RD/ (jei MP vykdo komandą IN 7Fh).

Programinės apklausos būdu išorinius įvykius galima apskaičiuoti pagal algoritmą, atvaizduotą 41 pav. Šį algoritmą atitinkanti programa, užrašyta MP 8085A asemblerio kalba, atrodo taip: .

```
Gr:   IN 07Fh   ; nuskaityti ventilio būseną
      CPI 2     ; tikrinti D1
      JZ Gr     ; jei nulis , kartoti iš pradžių
      INX D     ; priešingu atveju padidinti vienetu įvykių skaitiklį
      JMP Gr    ; ir vėl kartoti iš pradžių
```

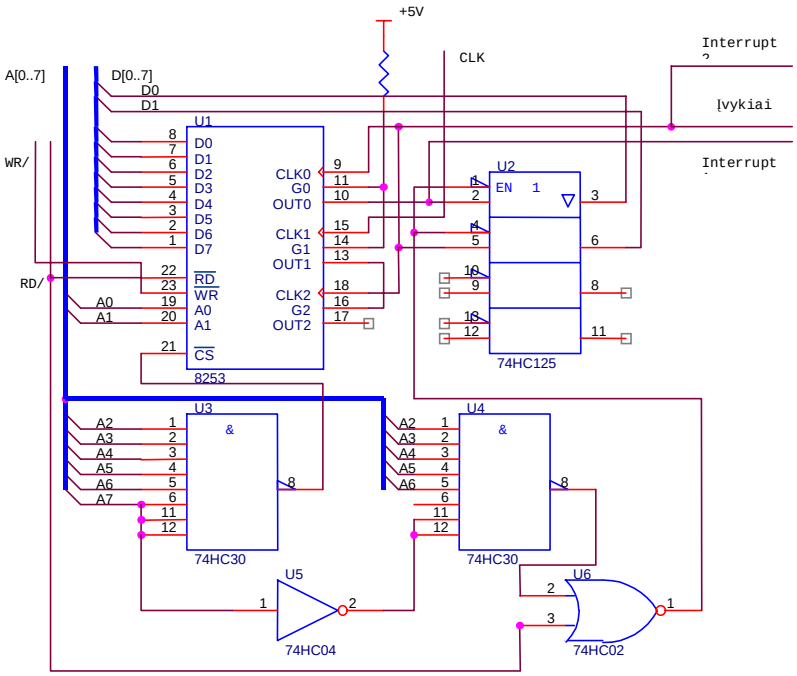
Matome, kad toks įvykių skaitiklis paprastai programuojamas, bet yra labai neracionalus. Čia nuolatos tenka "sukti tuščią ciklą".

Šiuo atveju daug tinkamesnis programos pertraukties mechanizmas. Jeigu įvykių signalą paduotume į MP pertraukties prašymo įėjimą INTERRUPT1, tai, atitinkamoms aparatinėms priemonėms palaikant, galėtume pertraukti kokią nors programą ir suskaičiuoti įvykius. Pertraukties aptarnavimo paprogramės algoritmas šiuo atveju tampa dar paprastesnis (žr. 42 pav.).

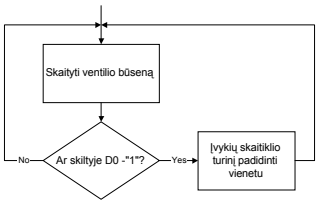
Jeigu įvykių apskaičiavimą realizuotume atminties ląstelių poroje (adresai 1000h ir 1001h), tai paprogramės tekstas atrodytų taip:

```
Skaitiklis:  PUSH H          ; išsaugoti dėkle registrų H,L turinius
             LHLD  1000h     ;  1000h ir 1001h  ląstelių turinius nuskaityti
                             ; į H,L;
             INX H           ; padidinti vienetu H,L turinį
             SHLD 1000h      ; išsaugoti atmintyje
             POP H           ; grąžinti H,L turinius iš dėklo
             RET             ; grįžti iš paprogramės
```

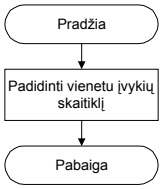
Ši paprogramė modifikuoja registrų poros H,L turinius, todėl būtina juos prieš pakeitimą išsaugoti dėkle (dėklo tipo atmintis yra toks atminties organizavimo būdas, kai pirmas įrašytas skaitomas paskutiniu), o pabaigoje grąžinti tokį, kokį turėjo pertraukta programa. Padedama į dėklą su komanda PUSH H, o išimama su POP H. Niekada nevalia pamiršti, kad PUSH ir POP komandų skaičius paprogramėje būtų vienodas, nes priešingu atveju, vykdant grįžimo į pertrauktą programą komandą RET, kuri atkuria (beje, taip pat iš dėklo) pertrauktos programos skaitiklį, pastarasis bus nuskaitytas klaidingai (ne pagal tą adresą).



40 pav. Programuojamas išorinių įvykių skaitiklis

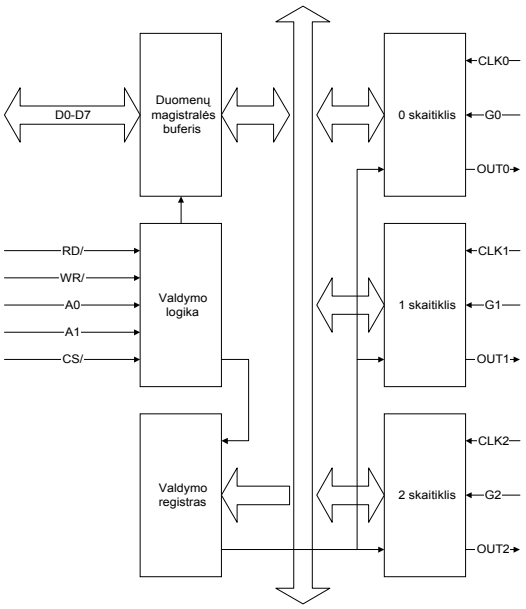


41 pav. Būsenos apklausos algoritmas



42 pav. Pertraukties aptarnavimo paprogramės algoritmas

Abu paminėti įvykių apskaičiavimo pavyzdžiai tinka palyginti lėtiems įvykiams suskaičiuoti. Kai įvykių pasikartojimo dažnis viršija keliolika kilohercų, MP jau nespėja atlikti parodytų programų. Tada patogu naudoti programuojamo taimerio DIG 8253. Pastaroji leidžia skaičiuoti įvykius, kurių pasikartojimo dažnis siekia 2MHz (yra modifikacijų, pvz., 8254, leidžiančių skaičiuoti 10 MHz dažniu pasikartojančius įvykius).



43 pav. 8253 struktūrinė schema

Pirmiausia išsiaiškinkime 8253 struktūrą (43 pav.) ir jo programavimą. Ją sudaro dvikryptis duomenų magistralės buferis, kurio viena pusė tiesiogiai jungiama prie MP duomenų magistralės, o kita - prie vidinės DIG duomenų magistralės. Prie valdymo loginės schemos jungiamos dvi jauniausiosios adreso magistralės skiltys A0 ir A1, MP valdymo magistralės signalai RD/ ir WR/, bei

DIG aktyvinimo signalas CS/. Prie vidinės duomenų magistralės jungiami trys 16-os skilčių skaitikliai. Kiekvienas iš jų turi skaičiavimo įėjimo (CLK0,CLK1 ir CLK2), leidimo skaičiuoti (G0,G1 ir G2) bei išvesties signalus (OUT0,OUT1 ir OUT2).

MPS taimerio sąsaja labai paprasta. Pakanka sudaryti tik adreso dešifratorių, kuris aktyvintų 8253 pasirinktoje atminties srityje. DIG turi keturis adresuojamus registrus, todėl turi būti parinktas dešifratorius, aktyvinantis DIG keturiems laisviems adresams.

8253 programavimas atliekamas su valdymo registru (jis adresuojamas kai A0=1 ir A1=1). Valdymo registro formatas yra toks:

D7	D6	D5	D4	D3	D2	D1	D0
SC1	SC0	RL1	RL0	M2	M1	M0	BCD

Skiltimis RL0 ir RL1 nustatoma dalijimo koeficiento įvedimo forma

RL1	RL0	Įvedimo forma
0	0	Įvedamas tik vyriausiasis dalijimo koeficiento baitas
0	1	Įvedamas tik jauniausiasis dalijimo koeficiento baitas
1	0	Nuosekliai įvedami jauniausiasis ir vyriausiasis dalijimo koeficiento baitai
1	1	Skaitiklio fiksavimas

Skiltimis SC0 ir SC1 parenkamas programuojamo skaitiklio numeris

SC1	SC0	Skaitiklis
0	0	nustatomas 0 skaitiklis
0	1	nustatomas 1 skaitiklis
1	0	nustatomas 2 skaitiklis

Kiekvienas iš trijų 8253 skaitiklių gali veikti šešiais darbo režimais. Jie parenkami režimo registro skiltimis M0,M1 ir M2.

M2	M1	M0	Darbo režimas
0	0	0	0 režimas
0	0	1	1 režimas
X	1	0	2 režimas
X	1	1	3 režimas
1	0	0	4 režimas
1	0	1	5 režimas

X – bet kokia būsena.

Dažnį dalyti ir įvykius skaičiuoti galima dvejetainiu arba dvejetainiu-dešimtainiu (BCD) kodu. Tai pasirenkama režimo registro skiltimi BCD. Jei čia loginis 0 – skaitiklis dvejetainis, jei 1- dvejetainis –dešimtainis.

0 darbo režimas.

Nustačius šį darbo režimą, išėjimo išvade yra loginio nulio lygis. Įvedus dalijimo koeficientą, išėjimas lieka loginio nulio būsenos, bet skaitiklis pradeda skaičiuoti. Jo išėjime pasirodo loginis vienetas, kai 16-os bitų skaitiklis, skaičiuodamas atgal (…3,2,1,0), pasiekia nulį. Reikia atkreipti dėmesį į tai, kad skaitiklis nepradeda skaičiuoti, kol neįvedamas dalijimo koeficientas, kuris nustatomas vienu arba dviem baitais.

1 darbo režimas.

Nustačius dalijimo koeficientą ir pasibaigus signalo kylančiam frontui įėjimo išvade G, išėjimo išvade OUT atsiranda loginis nulis. Jis lieka tol, kol skaitiklis, skaičiuodamas atgal (…3,2,1,0), pasiekia nulį. Šis režimas dar vadinamas programuojamo monovibratoriaus darbo režimu, nes čia, parinkus dalijimo koeficientą, galima pagal G išvade veikiančio signalo frontą, formuoti norimos trukmės išėjimo signalą. Išėjimo signalo trukmė gali kisti nuo 2 iki 65535 įėjimo signalo periodų.

2 darbo režimas.

Tai programuojamojo dažnio generatoriaus darbo režimas. Išėjimo išvade gauname neigiamus įėjimo signalo periodo trukmės impulsus, kurių pasikartojimo dažnis priklauso nuo įvesto dalijimo koeficiento. Kai jis lygus 3, išėjimo dažnis bus tris kartus mažesnis už įėjimo. Dažnio dalijimo koeficientas gali kisti nuo 2 iki 65535.

3 darbo režimas.

Tai programuojamojo dažnio meandro formos signalų generatoriaus darbo režimas.

4 darbo režimas.

Nustačius šį darbo režimą, išėjimo išvade yra aukštas loginis lygis. Įvedus dalijimo koeficientą, skaitiklis pradeda skaičiuoti. Pasiekus nulinį kodą, išėjimo išvade gaunamas neigiamas įėjimo signalo periodo trukmės impulsas.

5 darbo režimas.

Nustačius šį darbo režimą, išvadee O yra aukštas loginis lygis. Įvedus dalijimo koeficientą ir po teigiamo signalo įėjime G fronto, skaitiklis pradeda skaičiuoti. Pasiekus nulį, išėjime gaunamas neigiamas, įėjimo signalo periodo trukmės impulsas. Visų darbo režimų darbo laiko diagramos pateiktos 44 pav.

Aptarkime, kaip reikia užprogramuoti 43 pav. pateiktą įvykių skaitiklį. Tarkime, kad mums reikia įvykdyti pertrauktį, kai pasiekiamas įvykių skaičius 548D arba 224h, be to, įvykiai yra dažni. Jų pasikartojimo dažnis yra apie 100 kHz.

Šiuo atveju naudosime 0 skaitiklį, o jo išėjimo signalu inicijuosime pertrauktį INTERRUPT1. Pasirinkime 2 darbo režimą ir suformuokime valdymo registro kodą. Tegu skaitiklis veikia dvejetainiu režimu. Dalijimo koeficientą įvesime abiem baitais, nes jis didesnis už 255 (jei jis būtų mažesnis arba lygus 255, tai užtektų tik jaunesniojo baito).Tada valdymo registro turinio dvejetainis pavidalas bus 00111100B, arba 3Ch.

Sudarykime programą

```
MVI A,3Ch
OUT 0FFh  ; įrašome valdymo registro kodą
MVI A,24h
OUT 0FCh  ; įrašome jaunesnįjį dalijimo koeficiento baitą
MVI A,2
OUT 0FCh  ; įrašome vyresnįjį dalijimo koeficiento baitą
```

Taip programuojamo taimerio (įvykių skaitiklio) DIG 8253 paruošiamas įvykius skaičiuoti. Jis kaskart po 548 įvykių išėjimo išvade formuos neigiamą impulsą, kuris iškvies vykdymui pertraukties aptarnavimo paprogramę. Kad schema veiktų, reikia į įėjimą G0 duoti vieneto loginį lygį, prijungiant G0 per 10 kOm rezistorių prie +5V įtampos maitinimo šaltinio.

Prireikus visus tris vidinius skaitiklius galima sujungti nuosekliai. Taip dalijimo koeficientą galima padidinti iki 65535**3. Skaitikliai ir valdymo registras išrenkami adresais, nurodytais 20 lentelėje:

DIG 8253 adresavimas 20 lentelė

A1	A0	Kanalo numeris
0	0	0 skaitiklis (skaitymui ir rašymui)
0	1	1 skaitiklis (skaitymui ir rašymui)
1	0	2 skaitiklis (skaitymui ir rašymui)
1	1	Valdymo registras (tik rašymui)

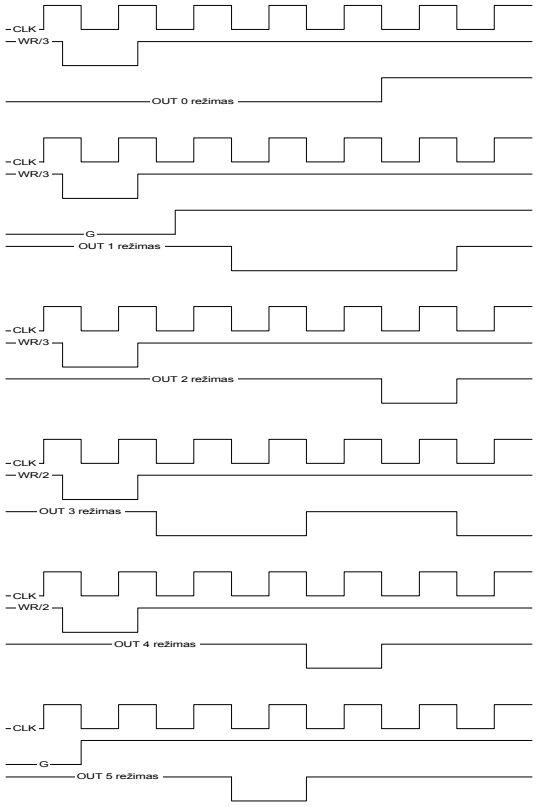
Matome, kad valdymo registro turinį galima tik įrašyti iš MP. Turinį nuskaityti galimybės nėra.

DIG 8253 galima panaudoti kaip programuojamą dažnio daliklį, laiko intervalų generatorių, taip pat signalų dažnio ir laiko intervalo trukmės matuoklį. Visada yra galimybė bet kuriuo metu nuskaityti bet kokio kanalo

skaitiklių turinius. Jeigu reikia skaičiavimą stabdyti, kol įvyks nuskaitymas į MP, į valdymo registrą reikia įrašyti kodą:

D7	D6	D5	D4	D3	D2	D1	D0
SC1	SC0	0	0	X	X	X	X

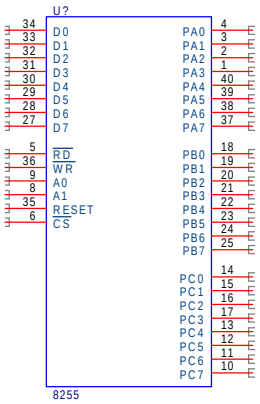
Panagrinėkime vieną iš DIG 8253 panaudojimo pavyzdžių. Tarkime, kad reikia išmatuoti įvykių pasikartojimo periodą. Tam tikslui į pirmo kanalo įėjimą CLK1 duokime MPS sinchronizacijos signalą CLK (MPS su MP 8085A šis dažnis bus lygus pusei kvarcinio rezonatoriaus dažnio, t.y. 6,144 / 2 = 3,072 MHz) ir jį panaudokime laiko "langui" formuoti. Laiko "lango" trukmę galime pasirinkti programiniu būdu, leidžiant dirbti pirmąjam skaitikliui/taimeriui trečiuoju režimu. Jei dažnio dalijimo koeficientą pasirinksime 30720, tai išėjime O1 bus 100Hz dažnio meandro formos signalas, kurio periodas 10ms. Tiriamus įvykius paduodame į antrojo taimerio (skaitiklio) įėjimą CLK2. Suskaičiavę įvykių skaičių N per 10ms (jis tiesiog nuskaitomas iš antro taimerio (skaitiklio) registrų), galime rasti išorinių įvykių pasikartojimo periodą (t=10ms/N), arba dažnį (1/t).



44 pav. 8253 darbo laiko diagramos

3.2. Lygiagreti sąsaja

Lygiagrečiam informacijos įvedimui ir išvedimui galima panaudoti ne tik registrus, bet ir programuojamus įvesties – išvesties DIG. Vienas jų yra 8255A. Jo žymėjimas principinėje schemoje pateiktas 45 pav., o išvadų paskirtis - 21 lentelėje.



45 pav. Programuojama lygiagrečių mainų DIG 8255

8255 išvadų paskirtis 21 lentelė

Išvadai	Paskirtis
D0 - D7	Dvikryptė 8 skilčių duomenų Magistralė
RESET	Pradinio nustatymo įvadas
CS/	DIG aktyvinimo įvadas
RD/	Skaitymo įvadas
WR/	Rašymo įvadas
A0,A1	Vidinių prievadų adresavimo įvadai
PA0 – PA7	A prievado linijos
PB0 – PB7	B prievado linijos
PC0 – PC7	C prievado linijos
Vcc	Maitinimas +5V
Vss	Maitinimas (bendras)

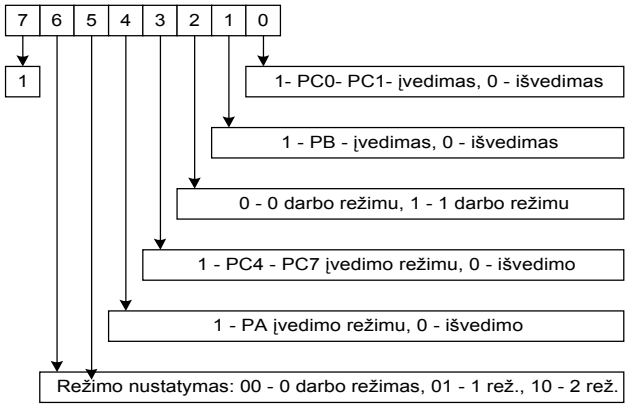
Šis DIG turi 24 individualiai programuojamas linijas, kuriomis galima duomenis įvesti arba išvesti dirbant trimis darbo režimais. Pagrindinės DIG operacijos pateikiamos 22 lentelėje. Kaip matome, A0 ir A1 išvadai naudojami PA,PB ir PC prievadų registrams, taip pat valdymo registrui adresuoti. Į juos kreipiamasi, kai signalas CS/ yra aktyvus. Į prievadus ir valdymo registrą galima informaciją įrašyti , o nuskaityti galima tik iš prievadų registrų. DIG darbo režimas nustatomas valdymo registre (control register) įrašytu kodu. Šis kodas turi būti įrašytas po to, kai įjungiamas maitinimas ir baigiasi signalas RESET (teigiamas impulsas RESET įėjime ištrina visų prievadų ir valdymo registro turinius). DIG darbo režimai yra tokie: 0 darbo režimas (mode 0) -tai paprasto įvedimo ir išvedimo režimas; 1 darbo režimas

(mode1) yra strobuojamo įvedimo ir išvedimo režimas; 2 darbo režimas (mode 2) yra dvikrypčių mainų režimas.

Pagrindinės DIG 8255 operacijos 22 lentelė

A1	A0	RD/	WR/	CS/	Skaitymas
0	0	0	1	0	Iš PA į duomenų magistralę
0	1	0	1	0	Iš PB į duomenų magistralę
1	0	0	1	0	Iš PC į duomenų magistralę
					Rašymas
0	0	1	0	0	Iš duomenų magistralės į PA
0	1	1	0	0	Iš duomenų magistralės į PB
1	0	1	0	0	Iš duomenų magistralės į PC
1	1	1	0	0	Iš duomenų magistralės į valdymo registrą
X	X	X	X	1	Duomenų magistralė aukšto išėjimo impedanso būsenos
X	X	1	1	0	Duomenų magistralė aukšto išėjimo impedanso būsenos

Programuojant 0 režimu, prievado PC linijos gali būti padalytos į dvi dalis (jaunesnysis PC0 - PC3 ir vyresnysis PC4 – PC7 pusbaičiai) (46 pav.).



46 pav. Valdymo žodžio formatas dirbant 0 režime

1 darbo režimas gali būti panaudotas lygiagretiems asinchroniniams duomenų mainams su “lėtais” MPS periferiniais įrenginiais. Šiuo atveju DIG prievadai skirstomi į dvi dalis. Kiekvienoje yra 8 skilčių duomenų magistralė ir 4 linijos valdymo signalams bei būsenai.

Strobuojamo įvedimo atveju valdymo signalai yra šie:

STB/ - strobavimo signalas, ateinantis iš išorinio įrenginio. Žemas šio signalo lygis įrašo duomenis į DIG PA arba PB;

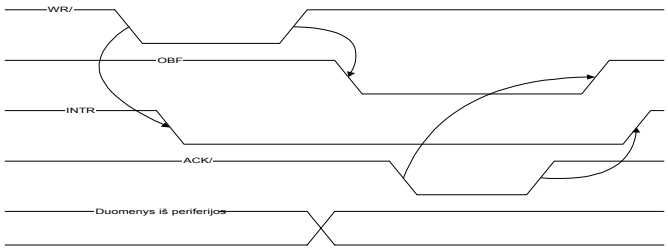
IBF (input buffer full) – aukštas šio signalo lygis rodo, kad duomenys į DIG buferį yra įrašyti. Šis signalas pašalinamas, kai vyksta skaitymas su signalu RD/;

INTR (interrupt request) – aukštas signalo lygis šiame išėjime rodo, kad duomenys į DIG įrašyti ir MP gali juos pasiimti.

Gali būti naudojami du strobuojamo įvedimo kanalai. Tada DIG išvadų pasiskirstymas būtų toks: PA0-PA7 ir PB0-PB7 - priimami duomenys, STBa/ (PC4) - strobas A kanalui, STBb/ (PC2) - strobas B kanalui, IBFa (PC5) - A kanalui , IBFb (PC1) - B kanalui , INTRa (PC3) – A kanalui , INTRb (PC0) – B kanalui. Valdymo žodžių kodas A kanalui būtų B8h , o B kanalui – 86h. 47 pav. pateiktos strobuojamos įvesties darbo laiko diagramos, o 48 pav. – strobuojamos išvesties. Strobuojamos išvesties darbo režimu signalų pasiskirstymas yra toks:

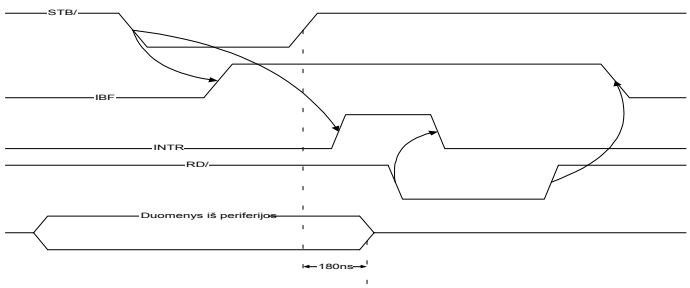
- aktyvus OBF/ (output buffer full) rodo, kad MP išvedė duomenis į 8255;
- aktyvus ACK/ (acknowledge) rodo, kad tai atsakas iš periferinio įrenginio , priėmusio duomenis;
- aukštas lygis išėjime INTR rodo, kad periferinis įrenginys jau pasiėmė duomenis ir pasiruošęs gauti tolesnius.

DIG išvadų paskirtis šiuo atveju būtų tokia: OBFa/ - PC7, OBFb/ - PC1, ACKa/ - PC6, ACKb/ - PC2, INTRa – PC3 , INTRb – PC0.



47 pav. Strobuojamos įvesties laiko diagramos

DIG 8255 galima panaudoti įvairiems lygiagrečios sąsajos uždaviniams spręsti. 49 pav. pateiktas keitiklių analogas - kodas prijungimo prie 8255 schemos fragmentas.

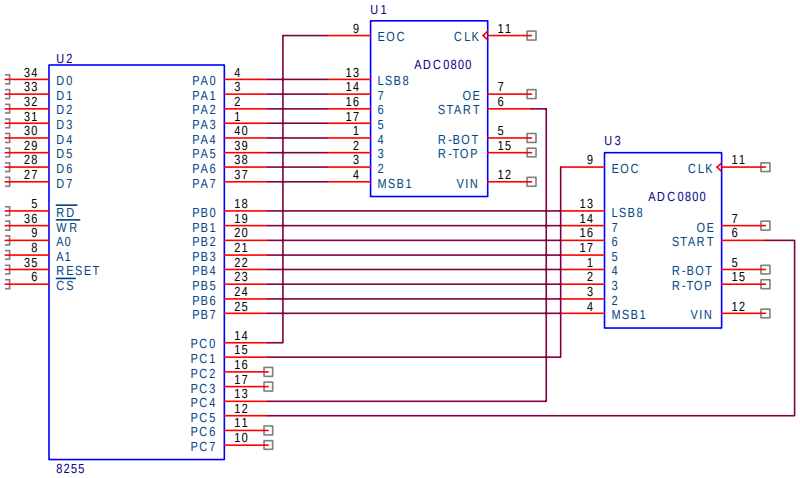


48 pav. Strobuojamos išvesties laiko diagramos

Čia DIG veikia 0 darbo režimu, prievadai PA ir PB panaudoti 8 skilčių duomenims iš keitiklių priimti, o PC prievado skiltys PC4 ir PC5 panaudotos keitikliams paleisti, PC0 ir PC1 - kontroliuoti keitimo pabaigą. Šiuo atveju PA, PB prievadai ir PC0-PC1 turi dirbti įvesties režimu, o PC4-PC5 – išvesties. Tokiu būdu valdymo registro kodas bus 93h (žr. 46 pav.).

Sutarkime, kad MPS DIG 8255A užima 40h-43h adresų sritį. Tada programos tekstas nuosekliam duomenų nuskaitymui iš abiejų keitiklių ir rezultatams patalpinti į atminties ląstelės 1000h ir 1001h atrodys taip (MP 8085A).

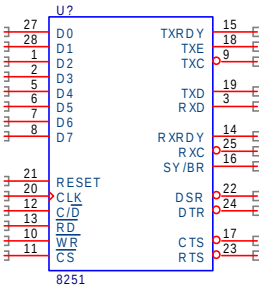
```
        MVI  A,93h      ; įrašome valdymo kodą
        OUT 43h
        MVI A,30h       ; paleidžiame abu keitiklius kartu trumpu teigiamu
                        ; impulsu
        OUT 42h
        XRA A
        OUT 42h
Ne:     IN 42h          ; tikriname keitimo pabaigą
        ANI 3
        CPI 3           ;  ar baigė keisti abu keitikliai
        JZ  Ne
        IN 40h          ; jei baigė , padėti rezultatus
                        ; į atmintį
        STA 1000h
        IN 41h
        STA 1001h
```



49 pav. Keitiklių analogas – kodas prijungimas prie 8255

3.3. Nuosekli sąsaja

Labai dažnai mainams tarp MPS ir periferinių įrenginių pasitelkiamos nuoseklaus sinchroninio ar asinchroninio ryšio priemonės. Tam tikslui gali būti panaudotas programuojamasis universalus sinchroninio-asinchroninio ryšio imtuvas-siųstuvas (USART – universal synchronous asynchronous receiver transmitter) 8251A.



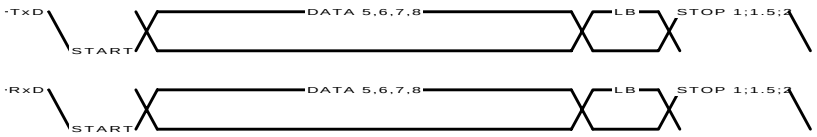
50 pav. USART 8251 žymėjimas principinėje schemoje

Jo žymėjimas principinėje schemoje pateiktas 50 pav., o išvadų paskirtis – 23 lentelėje.

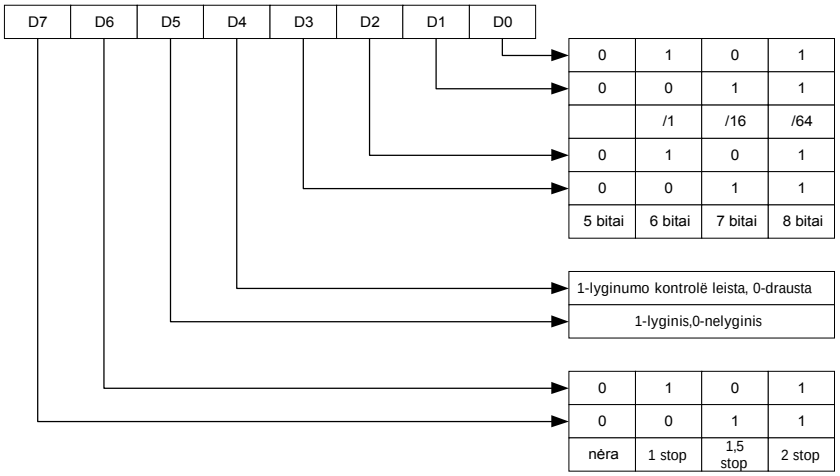
8251 išvadų paskirtis 23 lentelė

Išvadas	Paskirtis
D0 – D7	Dvikryptė duomenų magistralė. Per ją vykdomi duomenų ir valdymo baitų mainai su MP
C/D/	Valdymo komanda arba duomenys rašant arba skaitant. Jei šioje skiltyje 1 – perduodama arba priimama komanda , jei 0 – duomenys.
RD/	Skaitymas
WR/	Rašymas
CS/	DIG aktyvinimas
CLK	Sinchronizacija. Tai USART taktinis TTL lygių signalas. Į šį įėjimą patogu duoti MP 8085 sinchronizacijos išėjimo signalą
RESET	Pradinis nustatymas
TxC	Siųstuvo sinchronizacija. Į šį įvadą paduodamas 1/1,1/16 arba 1/64 perdavimo dažnio signalas
TxD	Siunčiami duomenys
RxC	Imtuvo sinchronizacija. . Į šį įvadą duodamas 1/1,1/16 arba 1/64 priėmimo dažnio signalas
RxD	Priimami duomenys
RxRDY	Imtuvas pasiruošęs. Šis signalas indikuoja , kad imtuvas priėmė nuosekliu būdu perduotus duomenis. Tuo pranešama MP , kad jis nuskaitytų imtuvo buferį
TxRDY	Siųstuvas pasiruošęs. Šis signalas praneša MP , kad USART yra pasiruošęs perduoti tolesnį simbolį. Signalas gesinamas, kai MP įvykdo rašymą į išvedimo buferį
DSR/	Duomenų nustatymui pasiruošta.Tai įėjimo signalas.MP gali jį nuskaityti , siekiant nustatyti išorinio modemo būseną
DTR/	Duomenys terminalui paruošti
SYNDET/BD	Sinchronizacija detektuota (trūkis detektuotas)
RTS/	Reikalavimas siuntimui
CTS/	Tuščias siuntimas
TxEMPTY	Siųstuvas tuščias

Asinchroniniu būdu perduodamų ir priimamų duomenų žodžio formatas pateiktas 51 pav. Pirmiausia pasirodo START bitas, pranešantis, kad po jo eis informacinis žodis. Pastarojo ilgis parenkamas programiniu būdu ir gali būti 5,6,7 arba 8 bitų ilgio. Jeigu leista lyginumo kontrolė, tai po duomenų eina lyginumo bitas LB. Po jo eina 1, 1.5 arba 2 STOP bitai. Valdančiojo žodžio, nustatančio asinchroninį darbo režimą, formatas pateiktas 52 pav. .



51 pav. Asinchroniniu būdu perduodamos informacijos formatas



52 pav. Asinchroninio darbo režimo komandos formatas

D0 ir D1 skiltys nustato tris asinchroninio režimo atmainas sinchronizavimo signalų atžvilgiu (su sinchronizavimo signalų dažniu, 1/16 ir 1/64 sinchronizavimo dažnio dalimis), D3 ir D2 skiltys -perduodamų bitų skaičių, D4,D5 - kontrolės režimą, D7 ir D6 - perduodamų stopbitų skaičių.

Nustačius reikalingą darbo režimą, DIG 8251 darbui dar reikia suformuoti komandą. Štai jos formatas:

EH	IR	RTS	ER	SBR	RxE	DTR	TxEN

Čia TxEN (transmit enable) – siuntimas leidžiamas, jei į šią skiltį įrašomas 1, priešingu atveju - uždraustas. DTR (data terminal ready) - klausimas terminalo siųstuvui, ar jis pasiruošęs duomenis perduoti. Įrašius 1 į šią skiltį, DIG išėjime DTR/ gausime loginį 0. RxE (receive enable) – priėmimas leidžiamas: 1 – duomenų priėmimas leidžiamas, 0 – draudžiamas. SBR (send

break character) – perdavimo pabaiga: 1 - išėjime TxD bus 0, 0 – normalus darbo režimas. ER (error reset) – klaidų nustatymas: 1 gesina klaidų požymius PE,OE ir FE. RTS (request to send) – klausimas terminalo imtuvui, ar jis pasiruošęs duomenis priimti: įrašius 1, DIG išėjime RTS/ gausime loginį 0. IR (internal reset) – įrašius loginį vienetą, DIG grįžta į darbo režimo nustatymo pradžią. EH (enable search for SYNC) – įrašius 1, leidžiama sinchronizacijos simbolio paieška.

DIG darbą kontroliuoja būsenos registras. Jo formatas yra toks:

DSR	SYNDET	FE	OE	PE	TxEMPTY	RxRDY	TxRDY

Čia PE (parity error) – lyginumo kontrolės klaida, OE (overrun error) požymis yra nustatomas, kai MP nenuskaitė anksčiau atsiųsto baito (buferio persipildymas), FE (framing error) požymis nustatomas, kai nedetektuojamas STOP bitas . Likusieji bitai atitinka DIG išvadus.

Išsiaiškinę programavimo instrukcijas, galime sudaryti programą, kuri periodiškai siųstų baitą 42h, programiniu būdu apklausinėdama siųstuvo būseną. Tarkime, kad DIG užima MPS adresų sritį 30h,31h.

```
        MVI A,40h
        OUT 31h         ;  paruošiame įvesti 8251 režimo komandą
        MVI A,FDh       ; dirbsime, nedalydami išorinio siuntimo,
                        ;  sinchronizacijos dažnio
                        ; siųsime po 8 bitus , su lyginumo kontrole
                        ;  perduodami 1, kai vienetų
                        ; skaičius duomenų baite yra lyginis
        OUT 31h
        MVI A,1         ; leisime dirbti tik siųstuvui
        OUT 31h
GR1:    MVI A,42h
        OUT 30h         ; perduodame duomenis
GR:     IN 31h
        ANI 1           ; tikriname, ar jau išsiųsta
        JNZ GR
        JMP GR1         ; jei taip - kartoti
```

Paleidę šią programą, oscilografu galėtume išvade TxD kontroliuoti siunčiamų duomenų laiko diagramą.

Daug patogiau būtų organizuoti mainus pertraukties režimu. Tam tikslui DIG išvadą TxRDY reikia sujungti su MP pertraukties įėjimu.

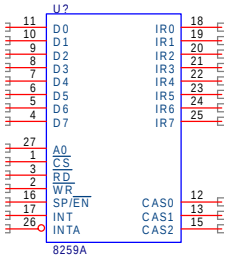
8251 prijungti prie MP 8085 ir MV 8051 magistralių labai paprasta. Išvadas C/D/ jungiamas prie jauniausiosios adresų magistralės skilties A0.

Duomenų magistralės linijos jungiamos lygiagrečiai. Skaitymo ir rašymo išvadai prijungiami prie vienvardžių MP arba MV išvadų. DIG aktyvinimo signalas formuojamas išorinio dešifratoriaus schemoje (reikia numatyti dviejų adresų sritį). SYNC signalui formuoti galima panaudoti išorinį generatorių (gerai tinka programuojamas taimeris 8253) arba MP 8085 atveju – CLKO išvade veikiantį TTL lygio meandro formos signalą. Informacijos nuoseklaus perdavimo greičiai standartizuoti: 50, 75, 150, 300, 600,1200, 2400, 4800, 9600, 19200, 14400, 28800, 38400 bitų per sekundę. Šiuo atveju sinchronizacijai tenka naudoti kvarcuotus generatorius, kurių dažnis yra kartotinis šio tinklelio dažniams. Kai kuriais atvejais labai patogus USART, į kurio sudėtį įeina ir generatorius. Tada MP sinchronizacijos dažnis gali būti bet koks. Pavyzdžiui, į DIG 8250 įeina programuojamas taimeris. DIG NS16C550, be programuojamo taimerio, turi ir vidinį 16-os bitų talpos FIFO (first input first output) tipo buferį. Juo naudojantis galima minimizuoti pertraukčių į MP skaičių ir padidinti MPS našumą.

USART kartu su lygių keitimo schemomis, veikiantys asinchroniniu režimu, labai plačiai naudojami personalinių kompiuterių (PC) ryšiui su periferija. Jungtys COM 0-3 (RS232) veikia valdomos analogiškų USART. Per jas jungiami modemai, spausdinimo įrenginiai, įvairūs matavimo ir valdymo prietaisai.

3.4. Pertraukčių valdiklis

Anksčiau buvo aptartos MP I8085 ir MV I8051 pertraukčių sistemos. Jeigu pertraukčių įėjimų skaičius nepakankamas, galima naudoti specialų pertraukčių valdiklį 8259A. Jo žymėjimas principinėje schemoje pateiktas 53 pav., o išvadų paskirtis - 24 lentelėje.



53 pav. Pertraukčių valdiklis 8259A

Pertraukčių valdiklio I8059A išvadų paskirtis 24 lentelė

Išvadai	Paskirtis
D0-D7	Dvikryptė duomenų magistralė
RD/	Skaitymas
WR/	Rašymas
CS/	DIG aktyvinimas
A0	Komandos išrinkimo įvadas
CAS0-CAS2	Kaskadinio jungimo linijos
SP/EN/	Valdiklio master/slave valdymo signalas
INT	Pertraukties išėjimo išvadas
INTA/	Pertraukties patvirtinimo įėjimo išvadas
IR0-IR7	Pertraukčių prašymo išvadai

8259A veikia taip:

- išoriniai įrenginiai į išvadus IR0-IR7 perduoda loginio vieneto lygio pertraukčių prašymo signalus;
- 8259A juos priima, paskirsto prioritetus ir į MP perduoda signalą INT;
- MP, baigęs vykdyti eilinę komandą, į INTA/ perduoda patvirtinimo signalą;
- 8259A į duomenų magistralę perduoda CALL (kviesti paprogramę) komandos kodą;
- 8259A dar dviem INTA/ impulsais perduoda jaunesnįjį ir vyresnįjį aptarnavimo paprogramės adreso baitus.

8259A programavimas atliekamas valdymo komandomis (54 pav.):

A0	D7	D6	D5	D4	D3	D2	D1	D0
0	A7	A6	A5	1	0	F	S	0
1	A15	A14	A13	A12	A11	A10	A9	A8

54 pav. 8259A programavimo komandos

Kai A0 = 0, nustatomos pertraukties aptarnavimo paprogramės pradinio adreso skiltys A5-A7, intervalas tarp aptarnavimo paprogramių pradinių adresų F (kai F = 1, intervalas yra keturi baitai, kai F=0 – aštuoni) ir pertraukčių valdiklių skaičius MPS yra S (kai S=1 – valdiklis yra vienas, kai S=0 – daugiau). Sujungus aštuonis 8259A valdiklius į kaskadinę schemą, galima sukurti 64 įėjimų prioritetinio pertraukčių sistemą.

Kai A0 = 1, perduodamas vyriausiasis pertraukties aptarnavimo paprogramės adreso baitas.

Pateiksime pavyzdį. Sakysim, mums reikia suprojektuoti pertraukčių sistemą MPS su 8085A, kurioje būtų panaudotas 8259A, o pertraukties

aptarnavimo paprogramių pradinių adresų lentelė programų atmintyje yra pradiniu adresu C000h. Sistemoje yra vienas 8259A. Jis adresuojamas 40h ir 41h adresais. Atstumai tarp pertraukčių aptarnavimo paprogramių pradinių adresų 4 baitai.

Suformuokime pirmą ir antrą komandų baitus. Pirmasis bus 16h. Jis turi būti perduodamas 40h adresu. Antrasis bus C0h. Jis turi būti perduodamas 41h adresu. Taigi 8259A pradinio nustatymo programa atrodys taip:

```
MVI A,16h    ; pradinis nustatymas
OUT 40h
MVI A,C0h
OUT 41h
```

Pradinių adresų lentelė bus tokia:

```
ORG C000h
JMP pertr0     ; pereiti į 0 pertraukties aptarnavimą
NOP            ; tuščia komanda , nes atstumas tarp adresų - 4
JMP pertr1     ; pereiti į 1 pertraukties aptarnavimą
NOP
JMP pertr2     ; pereiti į 2 pertraukties aptarnavimą
NOP
JMP pertr3     ; pereiti į 3 pertraukties aptarnavimą
NOP
JMP pertr4     ; pereiti į 4 pertraukties aptarnavimą
NOP
JMP pertr5     ; pereiti į 5 pertraukties aptarnavimą
NOP
JMP pertr6     ; pereiti į 6 pertraukties aptarnavimą
NOP
JMP pertr7     ; pereiti į 7 pertraukties aptarnavimą
NOP
```

Komandomis “JMP pertrn” nurodoma pertraukties aptarnavimo paprogramės vieta MPS programų atmintyje. Paprogramė būtinai turi dėkle išsaugoti modifikuojamų registrų turinius, “gesinti” 8259A pertraukties požymį, jeigu reikia – leisti aptarnauti kitas pertrauktis, baigtis grįžimo iš paprogramės komanda RET. Pavyzdys gali būti toks:

```
pertr0:  EI          ; leisti pertrauktį su aukštesniu prioritetu
         PUSH PSW    ; išsaugoti A
         IN Nr       ; įvesti
         STA 100h    ; rezultatą dėti į 100h ląstelę
```

```
MVI A,20h
OUT 40h     ; “gesinti” pertraukties požymį
POP PSW     ; grąžinti iš dėklo A
RET         ; grįžti iš paprogramės
```

3.5. Kontroliniai klausimai

1. Kokie blokai sudaro MPS? Paaiškinkite jų paskirtį.
2. Kodėl MPS sudaroma naudojant magistralinį-modulinį principą? Kokie šios MPS organizacijos privalumai?
3. Išvardykite pagrindinius MP struktūros elementus. Kokia jų paskirtis?
4. Kokius uždavinius sprendžia MP programų pertraukties blokas?
5. Kokiems uždaviniams spręsti naudojamas MP dėklas? Koks turi būti jo gylis ir kokios aparatinės MP priemonės padeda organizuoti dėklą operatyvinėje atmintyje?
6. Išvardykite pagrindinius MP I8085 ir MV I8051 struktūrinius skirtumus. Kokias atmintis gali adresuoti MV 8051? Kokie yra šių atminčių adresų laukai?
7. Kaip vyksta duomenų (komandų) skaitymas iš išorinės programų atminties IPA? Paaiškinkite šį procesą laiko diagramomis.
8. Kokias MV I8051 atmainas žinote? Kuo jos skiriasi viena nuo kitos?
9. Kokiems tikslams naudojami, kokiais darbo režimais gali dirbti ir kaip valdomi MV I8051 taimeriai?
10. Kokiais darbo režimais gali dirbti MV 8051 universalus asinchroninis siųstuvas-imtuvas UART? Kurio registro turinys nustato UART darbo režimus? Paaiškinkite jo atskirų bitų paskirtį?
11. Kokie signalai inicijuoja MV 8051 programos pertrauktis? Kokias funkcijas atlieka pertraukčių “kaukės” ir aptarnavimų prioriteto registrai IE ir IP?
12. Kaip atliekamas MV I8051 pradinis nustatymas ir maitinimo galios valdymas?
13. Pademonsruokite, kaip prijungti prie MV I8051 išorinės programų ir duomenų atmintis.
14. Ką reiškia funkcinis, elektrinis ir mechaninis atskirų MPS modulių suderinamumas?
15. Kokie uždaviniai sprendžiami projektuojant MPS sąsajas? Kokie naudojami trys skirtingi mainų tarp atskirų MPS modulių organizavimo būdai ?
16. Paaiškinkite MPS sisteminių ir žemesniojo lygio sąsajų esmę. Kokias standartines sistemines sąsajas ir tarpinių sąsajų įtaisus žinote?
17. Paaiškinkite, kaip organizuojama vienos ir dviejų krypčių MPS magistralės. Kokiu tikslu būtina atlikti magistralių elektrinį suderinamumą?

18. Kas yra MPS adresų erdvė? Paaiškinkite izoliuotąjį ir sutapatintąjį adresavimo būdą. Kas nulemia atminties erdvės puslapio talpą?

19. Paaiškinkite atminties blokų sudarymo ir prijungimo prie MPS principus. Kokie valdymo signalai reikalingi konkrečiai atminties ląstelei išrinkti ir kaip jie formuojami?

20. Paaiškinkite adresų erdvės praplėtimo bankų metodo esmę.

21. Kokiais principais remiantis ir kaip projektuojami įterptinių MPS programų atminties blokai?

22. Paaiškinkite duomenų atminties blokų ir jų sąsajų su MPS projektavimo principus.

23. Kaip organizuojami nesudėtingi lygiagretūs mainai tarp MP ir periferijos?

24. Kokie pagrindiniai blokai sudaro DIG 8253? Kaip ji programuojama? Pateikite keletą šio DIG panaudojimo pavyzdžių.

25. Kokia DIG 8255 paskirtis? Paaiškinkite jos išvadų paskirtį. Kokiais darbo režimais ji gali dirbti ir kaip nustatomas konkretus darbo režimas?

26. Kokiu formatu vyksta nuoseklūs asinchroniniai informacijos mainai tarp MPS ir periferijos? Pakomentuokite komandos ir būsenos žodžių paskirtį. Kaip prijungti USART 8251 prie MP 8085 ir MV 8051 magistralių?

27. Kokia yra DIG 8259 paskirtis? Kaip atliekamas jos programavimas?

5. MPS programinių priemonių projektavimas

5.1. Programų sudarymo metodai

Vystantis mikroprocesorinei technikai, visą laiką didėjo ir jos programinio aprūpinimo apimtis. Programos darėsi vis didesnės ir sudėtingesnės. Vienas pagrindinių programinio aprūpinimo sudarymo būdų yra sistemotechninio projektavimo metodologijos naudojimas. Ši metodologija jungia modulinio ir "žemėjančio" projektavimo metodus.

Modulinis projektavimas. Bendras sudėtingų uždavinių sprendimo principas yra jų skaidymas į smulkesnius. Taip sprendimą galima lengviau padalyti projektuotojų grupei. Tai plačiai naudojama kuriant sudėtingas programas. Kiekvienas modulis yra užbaigta programa, sprendžianti tam tikrą bendro uždavinio dalį.

Norint užtikrinti modulių suderinamumą, kiekviename jų nustatoma, kurie duomenys patenka ir kuriuos reikia grąžinti programos pabaigoje. Suskaidžius programą į dalis, supaprastėja jos realizacija, palengvėja skaitymas,

užtikrinamos tokios programos charakteristikos, kaip suprantamumas, modifikuotumas, efektyvumas, patikimumas, lankstumas ir pan.

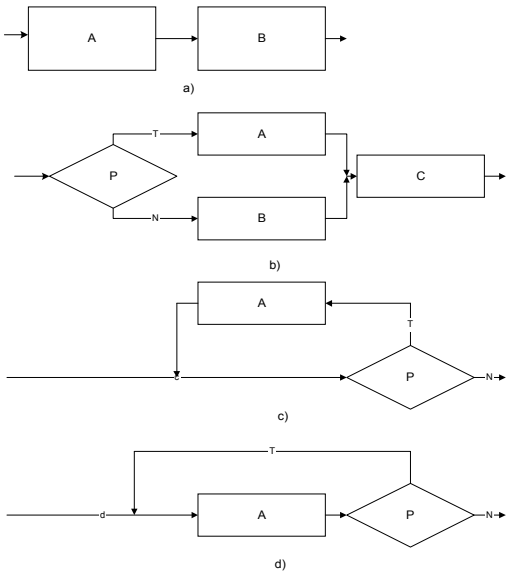
Programa pateikiama kaip modulių hierarchinė struktūra. Viršutinio lygio moduliai atlieka bendresnes funkcijas ir naudoja žemesnių lygių modulius, kuriuose detalizuojamas uždavinio sprendimas.

"Žemėjantis" projektavimas. Šis metodas panašus į daikto didinimo procesą, norint gauti detalesnį jo vaizdą. Pirmiausia pagal šį metodą koduojami, testuojami ir suderinami viršutinio lygio moduliai. Čia vietoj kol kas dar galutinai nesukurtų žemesniojo lygio modulių naudojami trumpikliai-aklės. Aklė – tai labai paprastos struktūros modulis, kurio įvedimo ir išvedimo duomenys atitinka pakeisto modulio duomenis. Dažnai aklėje, be įėjimo ir išėjimo duomenų aprašymo, būna tik vienas spausdinimo operatorius, pranešantis, kad šioje programos vietoje iškviečiama aklė. Taigi, pradedant viršutiniu, pačiu bendriausiu, lygiu , kiekviename tolesniame – žemesniajame lygyje patikslinamos programos atliekamos funkcijos, kol visiškai realizuojamos. Pagrindinis metodo privalumas tas, kad svarbiausias dėmesys kreipiamas į tinkamą programos projektavimą, o ne tik į detalų viso uždavinio supratimą. Jeigu pirmas projektavimo etapas korektiškas, tai kiekvienas tolesnis etapas yra tik ankstesnio šiek tiek pakeisto etapo patikslinimas. Tuo būdu gali būti lengvai patikrintas visų projektavimo etapų programos kūrimo korektiškumas.

Projektuojant naudojamas ir priešingas "žemėjančiam" – "aukštėjantis" projektavimo metodas. Tada visų pirma projektuojami pačio žemiausio lygio moduliai. Šis metodas dažnai tinka kuriant naujus mikroprocesorinius komplektus, specializuotos elektroninės aparatūros DIG ir pan. Kai kuriant EA žinoma elementinė bazė ir parinktas MP, patartina naudoti "žemėjantį" projektavimą.

Struktūrinis programavimas. Metodo pagrindas - Bemo ir Džekopini įrodyta teorema apie programų struktūrizavimą. Pasak šios teoremos, kad ir koks sudėtingas būtų uždavinys, jo sprendimo programos blokinę schemą visada galima sudaryti tik iš keleto elementarių struktūrų: nuoseklios sekos, sąlyginės sekos, ciklinės sekos (55 pav.).

Nuoseklią sekos struktūrą sudaro du ar daugiau nuosekliai atliekamų A ir B blokų (55a pav.). Sąlyginė struktūra užtikrina pasirinkimą iš dviejų alternatyvų. Čia nurodoma loginė sąlyga, turinti reikšmes T (taip) ir N (ne) ir nustatanti, kuris iš dviejų blokų - A arba B - bus įvykdytas (55 b pav.). Kartojimo struktūra leidžia atlikti ciklinius apskaičiavimus pagal dvi schemas (55 c, d pav.). Šios elementarios struktūros tarpusavyje sujungtos, gali sudaryti sudėtingesnes struktūras pagal tas pačias elementarias schemas su vienu įėjimu ir vienu išėjimu.



55 pav. Blokinių struktūrų tipai

Reikėtų pabrėžti, kad struktūriniame programavime galima nenaudoti besąlyginio perėjimo operatorių, tokių kaip GO TO, RETURN, JMP ir pan.

Programinės įrangos (PĮ) projektavimo procesas gana sudėtingas. Programa, kaip ir aparatinės priemonės, turi būti tiksli, patikima, stabiliai veikianti, patogiai besisiejanti su vartotoju, ji turi būti gerai skaitoma bei suprantama. PĮ projektavimo proceso valdymui yra skirta daug monografijų. Kai kurios jų pateikiamos literatūros sąraše [27,28]. PĮ projektavimo institutas (SEI - Software Engineering Institute) yra pasiūlęs PĮ galimybių išbaigtumo modelį (CMM-Capability Maturity Model). Čia išskiriami penki išbaigtumo lygmenys:

- 1 lygmuo. Pradinis. Programavimo procesas charakterizuojamas kaip chaotiškas ir didvyriškas. Nustatoma ir vykdoma keletas procesų, kurių baigtis priklauso nuo individualių programuotojo pastangų. Taip dirba daugelis nepatyrusių programuotojų. Jų kredo - programos darbo patikimumą galima pasiekti ne teisingu projektavimu, bet begaliniu testavimu. Šiame lygmenyje dar nėra formalaus programos kūrimo proceso valdymo;

- 2 lygmuo.Atkartojamas. Tai formaliai dokumentuoto proceso įžanga. Čia išskiriamos atskiros proceso dalys, siekiant valdyti finansus, darbo grafiką ir funkcinę produkto paskirtį. Perėjus į antrą lygmenį, reiškia, kad organizacija

pasirinko projekto valdymą, sudarė programavimo grupę, formaliai įteisino programavimo metodus ir techniką. Šiame lygmenyje projektas planuojamas, vykdomas reikalavimų ir konfigūracijos valdymas, programos kokybės užtikrinimas, nuolatos stebimas projektavimo procesas;

- 3 lygmuo. Nustatytas. Šis lygmuo pagrindžia nepertraukiamą projektavimo valdymo proceso parametrų tobulinimą. Projektavimo procesas toje organizacijoje integruojasi į standartinį. Visi organizacijoje vykdomi projektai naudoja tinkamiausią toms sąlygoms programavimo procesą ir atitinkamą programinę įrangą. Šiame lygmenyje vykdomas organizacinis proceso tikslinimas išskiriant svarbias dalis, suformuojama apmokymo programa, vyksta grupės darbo koordinacija ir atskirų etapų sujungimo valdymas;

- 4 lygmuo. Valdomas. Įvertinamas programavimo procesas ir gauto produkto kokybė. Abi sudedamosios dalys kiekybiškai įvertinamos. Šiame lygmenyje vykdomas proceso kiekybinis ir kokybinis valdymas;

- 5 lygmuo. Optimizuojantis. Nepertraukiamas proceso valdymo gerinimas su kokybiniu grįžtamuoju ryšiu, besiremiančiu pažangiomis idėjomis ir technologijomis. Šiame lygmenyje atliekamas proceso ir technologijų modernizavimas, klaidų ir defektų prevencija.

Šie penki lygmenys leidžia spręsti apie programavimo proceso išbaigtumą ir produkto galimybes.

Valdant programinės įrangos projektavimo procesą, patartina naudotis programų metrika (software metrics) [27]. Ji padeda suprasti ir įsivaizduoti projektavimo proceso etapus bei įvertinti programos kokybę. Programinės įrangos kokybės sistema (SQS – software quality system) sprendžia du uždavinius [28]:

- užtikrina kokybę kūrimo procese;
- išlaiko kokybę per eksploataciją.

SQS sudaro devyni elementai:

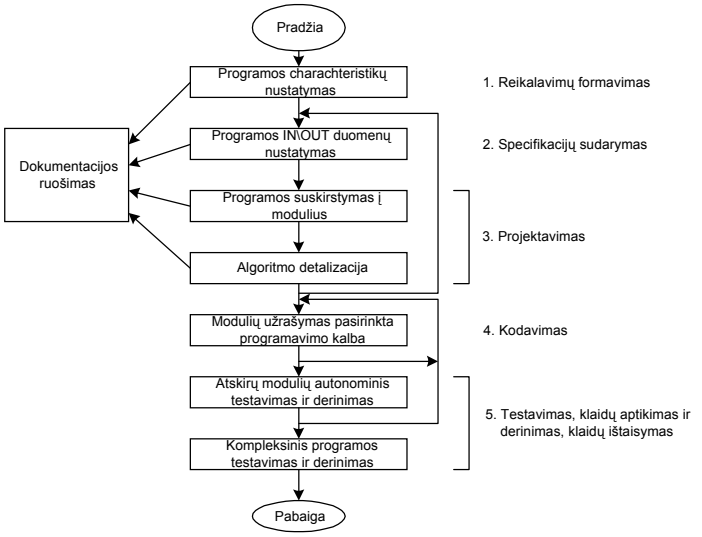
- standartai;
- recenzavimas;
- testavimas;
- defektų analizė;
- konfigūracijos valdymas;
- apsaugos priemonės;
- dokumentavimas;
- apmokymas;

- pardavimo valdymas.

5.2. Technologinis programų kūrimo procesas

MPS, įterptos į EA sudėtį, programinio aprūpinimo kūrimas turi nemaža specifinių ypatumų, palyginti su PC programinio aprūpinimo sukūrimu [11]. Visų pirma nustatoma MPS tarpusavio ryšys su įvairiomis sistemos sudedamosiomis dalimis: elektronine, elektrine, elektromechanine ir kt., išskiriamas skaičiavimų procedūrų rinkinys, realizuojamas MPS, taip pat įėjimo ir išėjimo signalų sąrašai.

Programų kūrimo technologinį procesą galima suskirstyti į keletą etapų (56 pav.). Reikalavimų programai formulavimo etape nustatomos kuriamos programos charakteristikos, tiksliai apibrėžiamos funkcijos, kurias programa turės atlikti, ir reikalavimai, kuriuos ji privalės tenkinti.



56 pav. Technologinis programų kūrimo procesas

Specifikacijų sudarymo etape nustatomi įėjimo ir išėjimo duomenys bei jų ryšiai. Specifikacija vadinama konkrečios MPS programai skirtas išsamus reikalavimų aprašymas. Specifikacijos yra išorinės ir vidinės. Į išorines specifikacijas įeina:

- programos vardas ir visų įėjimo taškų vardai;
- programos funkcijos;

- parametrų, perduodamų programai, skaičius ir tvarka;
- visų įėjimo ir išėjimo parametrų formatų, charakteristikų nustatymas;
- galimų klaidų aprašymas ir nuorodos, kaip sistema turi į jas reaguoti.

Kad būtų lengviau projektuotojui dirbti, šiame projektavimo etape galima pateikti pagrindinių klausimų suvestinę [18]:

1. Įėjimų specifikacija.

Apibūdinami įėjimai, nurodomi ir išsamiai aprašomi prijungiamų įtaisų tipai.

1.1. Įėjimo signalų tipai ir formos: skilčių skaičius, skaičių vaizdavimo forma, loginiai lygiai, signalų trukmės, mainų greičiai.

1.2. Įtaisų darbo laiko diagramos ir valdymo signalai, procesoriaus informavimas apie duomenų paruošimą.

1.3. Klaidos įvedamojoje informacijoje. Klaidų tikrinimo procedūros.

1.4. Nagrinėjamojo įėjimo programinis ryšys su kitais įėjimais.

2. Išėjimų specifikacijoje kartojasi visi tie patys klausimai, kaip ir pirmame punkte, tik išėjimo signalų ir išėjimo informacijos atžvilgiu.

3. Apdorojimo algoritmas.

3.1. Jei vienu metu sprendžiama keletas uždavinių, nurodomi jų prioritetai.

3.2. Apribojimai atminties talpai ir programos vykdymo laikas.

3.3. Specialūs atvejai (jei jų būna) ir jų apdorojimas.

3.4. Rezultatų tikslumas.

4. Klaidų apdorojimas.

4.1. Galimų klaidų apibūdinimas.

4.2. Sistemos atsistatymas po klaidos, patiriant minimalius laiko ir duomenų nuostolius.

4.3. Klaidos ar sutrikimai, galintys sukelti vienodą sistemos reakciją, jų atskyrimas.

4.4. Klaidos, dėl kurių iškviečiamos specialios sistemos procedūros.

5. Ryšys su operatoriumi.

5.1. Operatoriaus ir MPS bendravimo priemonės (jungikliai, indikatoriai) ir procedūros.

5.2. Dažniausios operatoriaus klaidos. Operatoriaus informavimas apie jo klaidas bei aparatūros sutrikimus.

5.3. Pagalbinės priemonės nepatyrusiam operatoriui.

Programos kūrimo procese dažnai tenka atlikti keletą specifikacijos sudarymo ir projektavimo ciklų. Tai būna tokiais atvejais, kai projektuojant išryškėja prieštaravimų, neapibrėžtumų ar kitų specifikacijos trūkumų. Tik tada, kai programa, vadovaujantis išorine specifikacija, bus suprojektuota kaip susietų tarpusavyje modulių visuma, kiekvienas modulis ir visi jo ryšiai su

kitais moduliais turi būti aprašyti vidinėmis specifikacijomis, o tuomet projektuojamas kiekvienas modulis atskirai.

Projektavimo etape programos struktūra nustatoma pagal tokią formulę: "programa=algoritmas+duomenų struktūra". Šiame etape bendras uždavinys, kurį reikia išspręsti, suskaidomas į atskirus fragmentus. Kiekvieną fragmento sprendimą vykdo atskiras programinis modulis. Programos suskaidymą į modulius turėtų iliustruoti ir dokumentuoti blokinė schema, rodanti modulių tarpusavio ryšius. Nustačius duomenų struktūras, pereinama prie detalaus algoritmų kūrimo.

Labai svarbi programos kūrimo proceso dalis yra dokumentacijos įforminimas. Vykdant mažus projektus, rekomenduojama:

- sudaryti reikalavimų specifikaciją;
- sudaryti projekto aprašymą;
- sudaryti testavimo raportą.

Vidutinės apimties projektams reikia sudaryti mažo projekto dokumentaciją ir:

- sukaupti preliminaraus projekto medžiagą;
- sukaupti detalaus projekto medžiagą;
- sudaryti testavimo planą.

Didelės apimties projektams reikia sudaryti vidutinės apimties projekto dokumentaciją ir:

- aprašyti testavimo sąlygas;
- aprašyti reikalavimus sąsajai ir pateikti jos projektą.

Labai didelės apimties projektams reikia sudaryti ankščiau minėtą dokumentaciją , taip pat:

- duomenų bazės projektą;
- paruošti vartotojo instrukciją;
- vartotojų apmokymo planą;
- palaikymo planą.

Kodavimo etapas dažnai vadinamas tiesiog programavimu. Šiame etape anksčiau suprojektuota programa užrašoma viena iš pasirinktų programavimo kalbų. Kodavimas yra vienas geriausiai ištirtų programinio aprūpinimo kūrimo etapų. Reikėtų įsisamoninti, kad kokybišką ir patikimą programą sukursime tuomet, jeigu ji kokybiškai ir korektiškai bus suprojektuota.

Galima pasiūlyti keletą rekomendacijų, kurių reikėtų laikytis sudarant programas :

- stenkitės kurti paprastą, o ne sudėtingą programą;
- tiksliai ir aiškiai reikškite mintis;
- nesustokite prie pirmo programos varianto;

- parinkite tokį duomenų pateikimo būdą, kuris supaprastina programą;
- nevenkite perrašyti nepavykusios programos vietos;
- įsitikinkite, kad įėjimo duomenys atitinka priimtinus apribojimus;
- užtikrinkite, kad neteisingai nustatyti įėjimo duomenys bus atpažinti;
- kuo rečiau naudokite besąlyginį operatorių GO TO (JMP), nes tada algoritmo logika tampa sunkiau suprantama, programos tekstas sunkiau skaitomas. Vietoje jo naudotinos struktūrinio programavimo konstrukcijos;
- aprašykite visus kintamuosius, naudodami komentarus;
- visų pirma stenkitės gauti teisingą programą ir tik po to ją optimizuoti.

Derinimo ir testavimo etapai - daugiausia darbo reikalaujantys programų kūrimo etapai. Statistika rodo, kad šiems etapams tenka apie 40% viso programos kūrimo darbo laiko. Šių etapų tikslas - išryškinti ir likviduoti programos klaidas, įsitikinti, kad programa teisingai atlieka savo funkcijas. Vėliau bus plačiai kalbama apie šiuos svarbius programos kūrimo etapus bei specialias derinimo priemones.

5.3. Bendra MPS programavimo technologija

Galutinis MPS programavimo rezultatas yra programos dokumentacija, programos tekstas, objektinė byla, užprogramuotas PA DIG. Programos dokumentacija (pradinis tekstas, transliacijos, krovimo, derinimo, DIG programavimo protokolai) yra pagrindinė sukurtos programos pateikimo forma, skirta programos ir MPS gamybai, modernizacijai ir eksploatacijai, o užprogramuotos pastovios atminties DIG - galutinė MPS programinių modulių pateikimo forma MPS gamybai ir eksploatacijai.

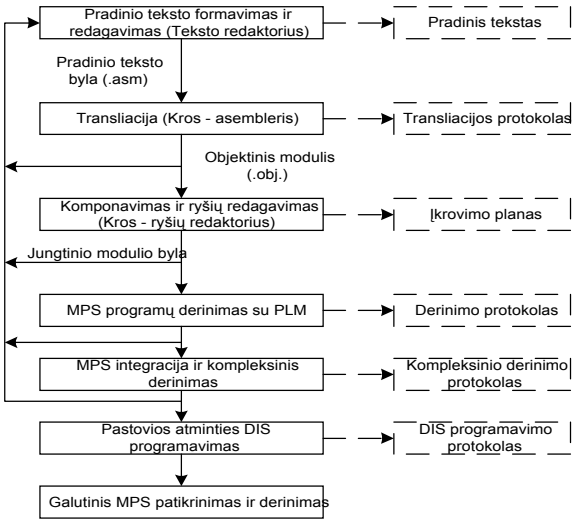
Paskutinieji du MPS programų kūrimo technologinio proceso etapai (57 pav.) galimi panaudojus rezidentines arba krosines priemones . Tai specialios paskirties mikro-ESM arba PC su atitinkamais aparatiniais priedėliais ir programų paketais. Juose MP gali būti toks pat arba savo komandų sąrašu suderinamas su projektuojamoje MPS naudojamu, arba ne. Pirmuoju atveju bus rezidentinė sistema, antruoju - krosinė.

Programų kūrimo procesą su krosinėmis priemonėmis iliustruoja 58 pav. Čia kaip programų aprašymo priemonė paimta projekte naudojamo MP asemblerio kalba. Su programa “Teksto redaktorius” rankiniu būdu rašome programos pradinį tekstą, jį redaguojame ir koreguojame. Tokiu būdu gautos programos pradinis tekstas gali būti įrašytas į kompiuterio kaupiklį kaip byla su plėtiniu **.asm** (asemblerio byla).

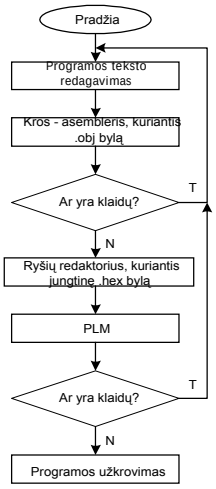
Krosasembleris pradinį programos tekstą, užrašytą asembleriu, pakeičia į objektinį kodą. Objektinė byla - tai programa, užrašyta mašininiais kodais ir įdėta nuo paleidimo (dažniausiai 0000h) adreso. Kros-asembleris keičia **xxx.asm** bylą į objektinę bylą su plėtiniu **xxx.obj.**

Kad būtų galima naudoti modulinį programavimo principą, į programavimo sistemą įtrauktas ryšių redaktorius (linker). Jo funkcijos - atskirų tos pačios programos objektinių modulių komponavimas ir sukomponuotos programos pasiuntimas programuotojo nurodytu adresu. Ryšių redaktoriaus darbo rezultatas - jungtinis failas su plėtiniu **xxx.hex.**

Programinis-loginis modelis (PLM simulator) modeliuoja MP komandų vykdymą ir yra labai patogi programų derinimo priemonė. PLM imituoja MP architektūrą ir darbo logiką, taip pat MPS PA ir DA PC operatyvioje atmintyje. Programos derinimo procesas su PLM vyksta dialogo režimu ir galima ištaisyti visas sintaksės ir logikos klaidas, padarytas programos algoritme. PLM darbo rezultatas – į MPS programų atmintį kraunama darbinė programa.



57 pav. MPS programų kodavimas, testavimas ir derinimas



58 pav. Programos derinimo algoritmas

Reikia atkreipti dėmesį į tai, kad klaidas programoje galima aptikti įvairiuose programos kūrimo etapuose. Norint jas ištaisyti, tenka grįžti į pradinį teksto redagavimo etapą.

5.4. MPS programavimo kalbos ir transliatoriai

Kiekviena MPS turi specifinę mašininę kalbą. Ši kalba - skaitmeninė, rašoma dvejetainiais, aštuntainiais arba šešioliktainiais kodais, ir todėl nepatogi programuoti. Visos kitos programavimo kalbos savo skirtingumu nuo mašininės skirstomos į dvi grupes: žemo ir aukšto lygio.

Žemo lygio kalboms priklauso pseudokodai ir makrokalbos, asembleriai ir makroasembleriai. Tai kalbos, priklausančios nuo konkretaus MP tipo.

Pseudokode mašininių komandų skaitmeniniai kodai pakeičiami raidiniais arba raidiniais-skaitmeniniais ekvivalentais. Duomenis leidžiama žymėti simboliniais vardais. Makrokalbose, be mašininių komandų ir pseudokodų, naudojamos makrokomandos, neturinčios analogų mašininėje kalboje. Verčiant į mašininę kalbą, pseudokodai keičiami į šios komandos skaitmeninius kodus, o makrokomanda - mašininių komandų grupe.

Naudojant aukšto lygio kalbas (HLL - high level language), daugumą algoritmų galima aprašyti vartotojui patogia forma, artima įprastam matematinių ir loginių veiksmų užrašymui. Tokių kalbų, naudojamų MPS programavimui, pavyzdžiai yra BASIC, PASCAL, FORTRAN, PL/M, C, C++

ir kt. Aukšto lygios kalbos vartojimo galimybės nepriklauso nuo MPS naudojamo MP ar MV tipo. Čia reikalingas atitinkamas kompiliatorius , kuris HLL parašytą programą verčia į asemblerį. Šiuo metu dažniausiai naudojami kompiliatoriai iš C ir C++.

HLL kalba parašytos programos tekstas daug suprantamesnis, be to užima 2-10 kartų mažiau eilučių negu programos, užrašytos asembleriu. Priimta laikyti, kad vidutinio programuotojo darbo našumas yra 5-20 suderintų programos eilučių per dieną (be vizualinių priemonių), nepriklausomai nuo to, kokio lygio kalba programa rašoma. Aukšto lygio kalbos pagreitina programų rašymo ir derinimo procesą. Tačiau programų, parašytų aukšto lygio kalba, krovimo moduliai užima 15-200% daugiau vietos programų atmintyje, o jų vykdymui reikia 15-300% daugiau laiko, negu programų, užrašytų asembleriu. Todėl šiuo metu didelė dalis MPS, dirbančių realiu laiku (tai būdinga daugumai įterptinių MPS), programų rašoma asembleriu. Mažesnės kvalifikacijos programuotojams patartina naudoti HLL, kurių kompiliatoriai pakankamai optimalūs. Juose dažnai numatoma galimybė pasirinkti optimizavimo pagal vykdymo laiką arba programos apimtį kriterijus. Galima programuoti ir mišriai, t.y. tą programos dalį, kuri nėra kritiška įvykdymo laikui, galima programuoti su HLL, o tą dalį, kurioje reikia užtikrinti maksimalią greitaveiką, su asembleriu.

Programą, užrašytą ne mašinine programavimo kalba, būtina pakeisti į mašininę. Toks pakeitimas vadinamas transliacija, o programa, atliekanti šį keitimo procesą - transliatoriumi. Aukšto lygio kalbų transliatoriai vadinami kompiliatoriais ir interpretatoriais, žemo lygio - asembleriais. Todėl ir pseudokodai, ir makrokalbos dažniau vadinamos asemblerio kalbomis. Toliau pateiksime kai kurias bendras MP ir MV asemblerių charakteristikas ir parametrus, nors šių kalbų realizacija gana skirtinga. Tai pasakytina apie šių kalbų pseudokodus, formatą, adresavimo būdus, duomenų nustatymo būdus ir pan.

Kiekviena MP programos eilutė, parašyta asembleriu, turi keturis laukus:

ŽYMĖ OK OPERANDAI KOMENTARAI

Žymės laukas (LABEL) skirtas duoto operatoriaus simboliniam vardui užrašyti. Žymė - tai raidžių ir skaičių seka, prasidedanti raide ir paprastai neviršijanti aštuonių simbolių. Transliuojant MP programą, žymė pakeičiama absoliutiniu arba santykiniu adresu. Žymės reikšmė nustato atitinkamos komandos arba duomenų elemento adresą atmintyje, į kurį reikia programai pereiti. Būtina pažymėti, kad žymė programoje gali būti parašyta tik prie vieno operatoriaus, tačiau nukreipimai į ją gali būti organizuojami tiek kartų, kiek tai būtina. Žymė dažnai baigiasi dvitaškiu.

Operacijos kodo (OK) lauke (OPCODE) rašomos MP arba MV komandos kodas. Šiame lauke leidžiama rašyti pseudokomandų vardus bei transliacijos valdymo direktyvas. Daugumos komandų pavadinimai yra jų pagrindines funkcijas aprašančių sakinių sutrumpinimai, pvz., MOV (MOVe) - perduoti, persiųsti, JNZ (Jump if Non Zero) - pereiti, jeigu ne nulis ir pan., arba tiesiogiai nustato komandos funkcijas, pvz., PCHL komanda reiškia registrų poros HL turinio perrašymą į programų skaitiklį PC, o komanda PUSH įkrauna dėklą (MP I8085 komandų sistema).

Paprastai OK laukas neviršija keturių pozicijų, o tarp jo ir operando lauko būtinas nors vienas tarpas (space).

Operando laukas (OPERAND) priklauso nuo OK lauko ir gali nurodyti vieną ar keletą operandų, atskirtų kableliu, arba iš viso gali būti būti tuščias. Kaip operandai asemblerio kalbose gali būti nurodyti MP registrai, konstančių simboliniai kodai, žymės, nustatytos MP programoje. Kai kuriuose asembleriuose galima operando lauke rašyti aritmetines išraiškas. Išraiškų rezultatai suskaičiuojami transliavimo metu.

Skaitmeniniai duomenys gali būti pateikti šešioliktainiame kode, po skaičiaus rašant raidę H (Hex), dešimtainio kodo skaičius gali baigtis raide D (Decimal), dvejetainio kodo skaičius turi baigtis raide B (Binary).

Komentarų lauką (COMMENTS), prasidedantį tam tikru skiriamuoju ženklu (kabliataškiu arba įstrižu brūkšniu su žvaigždute /* … */), asemblerio transliatorius visiškai ignoruoja, todėl jame gali būti bet koks tekstas. Jo turinys turėtų paaiškinti toje eilutėje atliekamus veiksmus programos vykdymo kontekste.

Programa, parašyta asemblerio kalba, turi mašinines komandas ir pseudokomandas. Pseudokomandos, arba transliatoriaus direktyvos, naudojamos transliacijos procesui valdyti. Jos užrašomos tuo pačiu formatu kaip ir asemblerio komandos, tačiau nekeičiamos į objektinį kodą.

START – programos pradžia.
END - programos teksto pabaiga.
DS - atminties rezervavimas. Operande nurodomas baitų skaičius, reikalingas DA rezervuoti.
DB - baito apibrėžimas. Operandu gali būti nurodytas išraiškų arba simbolių eilučių sąrašas. Sąrašo elementai atskiriami eilute (DB 0F2h, DB ‘Baitas’).

DW - žodžio apibrėžimas. Pseudokomandos operandu gali būti nurodyta išraiškų arba simbolinių eilučių sąrašas, kurio elementai atskiriami vienas nuo kito kableliu (DW 0F200h).

EQU - reikšmės suteikimas. Žymės vietoje nurodomas vardas be dvitaškio. Šis vardas neturi kartotis kitose pseudokomandose (arg EQU 14h) .

ORG – transliacijos pradžios adreso nustatymas. Pseudokomanda nustato programos komandų skaitiklį. Transliuojant tolesnė pagal eilę komanda arba duomenų elementas turės adresą, nurodytą šios pseudokomandos operando lauke. Jeigu pseudokomandos ORG nebus parašytos prieš pirmą programos komandą, tai komandų skaitiklyje nustatomas nulinis adresas.

5.5. Paprogramės

Programų su bet kuria programavimo kalba sudarymas pradedamas nuo sprendžiamo uždavinio sąlygų studijavimo ir algoritmo sukūrimo. Algoritmas - tai uždavinio sprendimo planas. Jo kūrimui rekomenduojama naudoti standartines valdymo perdavimo konstrukcijas, pateiktas 55 pav. Algoritmas projektuojant yra tikrinamas ir, jeigu to reikia, smulkinamas (tikslinamas). Patikrinus algoritmą galima rašyti programos tekstą. Pasikartojančias komandų grupes reikėtų įforminti kaip paprogrames (subroutine). Paprogramės padaro programą lengviau skaitomą. Patartina nepiknaudžiauti formuojant paprogrames. Jeigu jos yra pakankamai trumpos, o algoritmas kritiškas vykdymo laikui, geriau naudoti nuoseklią struktūrą. Šiuo atveju sutaupysite laiko, nes išvengsite operacijų su dėklu.

Paprogramės yra dviejų rūšių: atviros ir uždaros. Atvira paprogramė įterpiama taškuose, kur reikia ja pasinaudoti. Jos pavyzdys gali būti makrokomanda. Uždara paprogramė yra atskiras pagrindinės programos modulis. Pagrindinės programos taškuose, kuriuose ją reikia panaudoti, užrašomas kreipinio operatorius. Pabrėžtina, kad paprogramės terminas dažniau vartojamas uždarai paprogrames pažymėti, vien jau dėl to, kad ji yra nepriklausomas modulis, kurį gali panaudoti kiti moduliai. Taigi paprogramės yra programų modulinio projektavimo pagrindas. Kaip jau anksčiau pabrėžta, naudojant šį metodą, galima didelę ir sudėtingą programą suskaidyti į mažesnes ir paprastesnes dalis, kiekvieną jų atskirai derinant. Tuo būdu pagreitėja ir supaprastėja visos programos derinimas.

Bet kurio į asemblerio programą įeinančio modulio tekstą rekomenduojama suskirstyti į keletą dalių: įvadinę, aprašančią ir pagrindinę.

Įvadinėje dalyje nurodoma modulio paskirtis, programuotojo pavardė, programos sukūrimo data, versija, modifikacijos ir jų datos.

Aprašančioje dalyje yra direktyvos, nustatančios objektinio modulio vardą, makrokomandų (jeigu jų yra) aprašymai, vardų ir atminties sričių, naudojamų pradiniame modulyje, aprašymai. Paprogramėms šioje dalyje papildomai nurodoma, kaip pateikti ir grąžinti duomenis, kokius vidinius MP registrus naudoti .

Pagrindinę dalį sudaro algoritme nurodytus veiksmus atliekantys asemblerio operatoriai. Ši dalis baigiasi direktyva END.

Paprogramių organizavimui MP komandų sistemoje numatomos sąlyginio ir besąlyginio iškvietimo ir grįžimo komandos (CALL ir RET). Vykdant komandą CALL, dėkle įsimenamas grįžimo adresas, o į programos skaitiklį PC įrašomas paprogramės pradinis adresas. RET komanda užbaigia paprogramę ir į PC grąžina pagrindinės programos adresą.

Naudojant paprogrames, iškyla kai kurių MP vidinių registrų naudojimo problemų. Būtina užtikrinti, kad paprogramės nepakeistų pagrindinėje programoje naudotų registrų turinių. Tokių registrų turiniai turi būti laikinai išsaugoti MPS atmintyje, o pasibaigus - atkurti. Kaip laikina registrų turinio saugykla paprastai naudojamas dėklas. Registrų turinių laikino išsaugojimo ir atkūrimo komandas tikslinga įterpti į paprogrames (paprogramės pradžioje ir pabaigoje).

Duomenų mainams (parametrų perdavimui) tarp pagrindinės programos ir paprogramių naudojami įvairūs būdai. Paprasčiausiai duomenis perduoti per MP bendros paskirties registrus, į kuriuos gali būti įrašyti duomenys arba jų adresas. Šio būdo trūkumas - nedidelis perduodamų parametrų skaičius dėl riboto bendros paskirties registrų (BPR) kiekio. Sudėtingesnis būdas - panaudoti duomenų atminties ląsteles, į kurias gali kreiptis ir iškviečianti programa, ir paprogramė. Į šią atminties sritį iškviečianti programa įrašo parametrų (duomenų) reikšmes, o paprogramė jas nuskaito, apdoroja, o rezultatus įtraukia į kitas fiksuotas atminties ląsteles, kuriose jos tampa prieinamos kviečiančiai programai.

5.6. Programų derinimas ir derinimo priemonės.

Patirtis rodo, kad nors ir kruopščiai, laikantis visų taisyklių ir rekomendacijų, pereinami visi projektavimo etapai, programose klaidų išvengti dažniausiai nepavyksta. Pagrindiniai aktyvūs klaidų aptikimo ir likvidavimo programose metodai yra testavimas ir derinimas.

Testavimo tikslas - aptikti programoje esančias klaidas, o derinimo - aptiktas klaidas likviduoti. Paprastai testavimo ir derinimo etapai savo trukme yra vieni ilgiausių palyginti su kitais programos kūrimo ciklo etapais.

Projektuojant programą, testavimo objektas gali būti reikalavimai ir specifikacijos, algoritmai, moduliai, programų grupės ir pan. Smulkiau panagrinėkime programinių modulių testavimą. Duomenis apie kitų objektų testavimą bei testavimo metodus galima rasti literatūroje [3,27,28].

Atskirų programinių modulių tikrinimui naudojami įvairūs testavimo metodai. Tikrinant programą visų pirma reikėtų naudoti rankinį testavimo metodą. Tokiu būdu galima nustatyti 30-70% visų algoritmo klaidų. Pagrindinis rankinio testavimo dokumentas yra programos tekstas (listing),

gautas sutransliavus programą ir ištaisius sintaksės klaidas. Rankinį testavimą atlieka programos kūrėjas.

Rankiniu būdu testuojant, atliekami tokie tikrinimai:

- analizuojant programos tekstą, aptinkamos klaidos operatoriuose, duomenų aprašymuose bei transformavimuose;
- patikrinami apskaičiavimai bei jų tikslumas;
- patikrinama programos veikimo logika, esant konkrečioms įėjimo duomenų reikšmėms, jų apdorojimo maršrutams.

Aprašyto derinimo proceso galimybės nesunkiai realizuojamos su krosinėse sistemose esančiu PLM. PLM - tai programinė priemonė, skirta MPS programos loginiam suderinimui. Jo sustambinto algoritmo schema pateikta 59 pav.

Panagrinėsime pagrindinių PLM paprogramių paskirtį ir funkcijas. Dialogo paprogramė užtikrina vartotojo ryšį su PLM, leidžia nustatyti reikiamą darbo režimą ir pradinius duomenis, o reikalingus tolesnio režimo realizavimo programinius požymius apibrėžia režimo nustatymo paprogramė. Pradinių sąlygų nustatymo paprogramė vykdo transliaciją vartotojo įvedamų sąlygų, tokių kaip adresų, derinamos programos ribų, dėklo ribų, sustojimo sąlygų, pertraukčių leidimo ar draudimo ir pan. Įkrovimo paprogramė krauna vartotojo programą į DA sritį, skirtą MPS pastovios atminties imitacijai. Interpretatoriaus paprogramė imituoja derinamo MP architektūrą, o disasemblerio paprogramė perveda interpretuojamą komandą į ją atitinkantį asemblerio operatorių.

Tipiniai PLM darbo režimai yra tokie:

1. Programos vykdymas sustojant kontroliniuose taškuose (break points). Juos galima nustatyti pagal įvairias sąlygas: sustojimas prie nurodyto adreso, sustojimas pagal informacijos iš tam tikros atminties ląstelės arba registro skaitymo arba rašymo momentą arba tos informacijos tam tikrą reikšmę. Sustojus programai kontroliniame taške, visų MP architektūros elementų turiniai indikuojami PC ekrane.
2. Programos vykdymas nuo nustatyto adreso, sustojant prie nustatyto komandos kodo.
3. Programos vykdymas žingsnio režimu su visų MP elementų turinių indikacija.
4. Programos vykdymas iki galo be sustojimų.
5. Suderintos programos išsaugojimas byloje.

Rekomenduojama tokia programos derinimo su PLM darbų seka:

1. Įvykdyti programą su įėjimo duomenų testo rinkiniais ir nustatyti. ar yra klaidų.

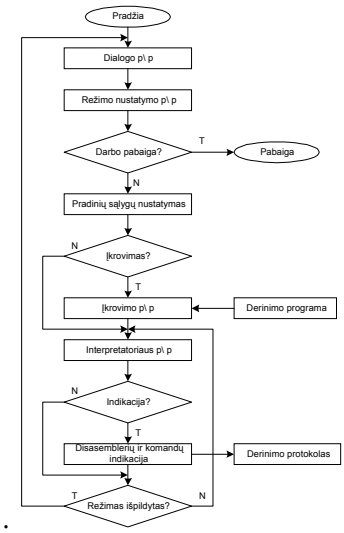
2. Išskirti programos sritį, kurioje gali būti klaidų. Peržiūrėti programos tekstą - kartais vizualiai galima aptikti klaidą. Priešingu atveju - nustatyti kontrolinį sustojimo tašką apytikriai išskirtos srities viduryje.

3. Įvykdyti programą dar kartą. Jeigu programa buvo pertraukta iki kontrolinio sustojimo taško, vadinasi klaida įvyko anksčiau. Kontrolinį tašką reikia nustatyti arčiau ir grįžti prie antrojo punkto.

4. Jeigu programa įvykdyta iki kontrolinio taško, atidžiai išnagrinėti registrų, programos kintamųjų ir parametrų reikšmes, siekiant įsitikinti jų teisingumu. Jeigu konstatuota klaida, tenka grįžti prie antro punkto.

5. Jeigu klaidos tokiu būdu nepavyko aptikti, įvykdyti programą žingsnio režimu, ypač kontroliuojant perėjimus programoje, taip pat registrų ir atminties turinius kontroliniuose taškuose. Jeigu klaida lokalizuota, ją reikia ištaisyti ir grįžti į algoritmo pradžią (1 punktas).

Reikia atkreipti dėmesį į tai, kad nepaisant visų PLM privalumų, iki galo (kompleksiškai) MPS programos suderinti nepavyksta. Kompleksinis programos derinimas gali būti atliktas tik dirbant kartu su realiu objektu arba MPS fizikiniu modeliu realiu laiku. Šiuo atveju naudojamos aparatinės-programinės derinimo priemonės - scheminiai emuliatoriai (incircuit emulators)



59 pav. PLM algoritmo schema

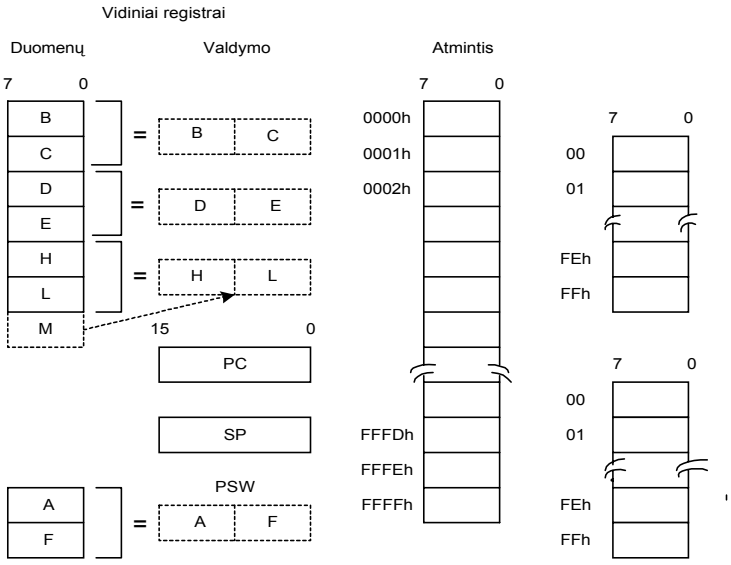
5.7. Mikroprocesoriaus 8085A programavimas

5.7.1. I8085 programinis modelis

Šiame skyriuje susipažinsime su MP 8085A programavimu ir programų derinimo sistema.

Projektuojant MPS programinį aprūpinimą asemblerio kalba, reikia žinoti sistemoje naudojamo MP programinį modelį ir komandų sistemą. MP 8085A programinis modelis pateiktas 60 pav.

MP 8085A yra aštuoni bendros paskirties 8 skilčių registrai (BPR), kurie gali būti sujungti į 16 skilčių registrų poras (RP). Šios poros gali saugoti 16 bitų operandus arba būti kaip atminties ląstelių rodikliai. Pagrindinis atminties rodiklis yra HL-pora. Atminties ląstelė, kurios adresą nusako HL poros turinys, žymima raide M. MP I8085A gali adresuoti po 256 įvesties ir išvesties prievadus bei 64K baitus atminties (programų + duomenų).



60 pav. 8085A programinis modelis

5.7.2. I8085 komandų sistema ir adresavimo būdai

MP 8085A adresams nurodyti naudoja keletą operandų adresavimo būdų. Pats paprasčiausias, bet ir neekonomiškiausias (naudojamos tribaitės

komandos) yra tiesioginis . Čia komandos operando lauke yra 16 bitų atminties adresas (pvz., LDA 3000h).

Betarpiškas adresavimo būdas (adresavimas tiesioginiu operandu) tinka tada, kai reikia į programą įterpti fiksuotas duomenų reikšmes. Šiuo atveju MP komandą sudaro vienas operacijos kodo (OP) baitas ir vienas arba du betarpiškų duomenų baitai (pvz. CPI 0F2h) .

Naudojant registrinę adresaciją, operandas yra registre, kurio adresas nurodomas komandoje.Tai vienbaitės ir greičiausiai vykdomos komandos (pvz., MOV A,B).

Vienbaitėmis komandomis su netiesiogine adresacija galima gana greitai kreiptis į visas MP atminties ląsteles. Iš pradžių į atminties rodiklį (BC, DE, HL poras ir SP) įrašomas pradinis duomenų sekos atmintyje adresas, o kreipiniui į gretimą ląstelę užtenka tik inkrementuoti (dekrementuoti) atminties rodiklio (registrų poros) turinį. Taigi netiesioginė adresacija plačiai naudojama masyvo tipo duomenų struktūrų apdorojimo, kreipinių į dėklą komandose ir pan. (pvz., MOV A,M).

Šis MP naudoja taip pat komandas su vadinamąja neryškia adresacija. Operandas yra visuomet tam tikrame konkrečiame MP registre - šiuo atveju A (pvz., visos postūmio komandos).

MP 8085A komandų sistema pateikta priede (31 lentelė). Čia panaudoti tokie pažymėjimai. Raide R pažymimas vienas iš MP registrų (B, C, D, E, H, L, M arba A). Šiuos registrus atitinka kodai R (0, 1, 2, 3, 4, 5, 6, 7). Dviejų baitų komandose operando kodas žymimas kaip N_0. Trijų baitų komandose antro ir trečio baitų kodai N_0N_1 nurodo atminties ląstelės adresą. Rodyklės stulpelyje "komandos turinys" nurodo duomenų kryptį, pvz., užrašas $r \leftarrow N_0$ reiškia, kad skaičius yra registre r, $r_1 \leftarrow r_2$ - registro r_2 turinys persiunčiamas į registrą r_1 , o užrašas $[rr'] \leftarrow A$ - registro A turinys persiunčiamas į atminties ląstelę, kurios adresas nurodytas registrų poroje *rr* (rr=BC, DE, HL). Dviejų baitų skaičius laužtiniuose skliausteliuose [N1N0] reiškia, kad duomenis reikia paimti arba įrašyti į atminties ląstelę adresu N,N_0. Vieno baito skaičius lenktuose skliaustuose – įvesties ir išvesties prievado adresas (numeris). Mažoji c raidė komandos aprašo stulpelyje nurodo keliamojo vieneto požymį, o Ai (I=0,1,.., m, m+1,.., 7) - A registro i-ą skiltį. Operacijų kodai pateikiami šešioliktainiais ir dvejetainiais skaičiais. Dvejetainiais skaičiais pateiktas operacijos kodas tų komandų, kurios turi keletą (keliasdešimt) modifikacijų. Šių komandų operacijos koduose skiltys III (II) nurodo operando imtuvo (destination), o skiltys SSS (SS) - operando siųstuvo (source) dvejetainį kodą (adresą), pvz., komandos MOV A, C kodas yra 01111010, čia, pradedant

vyriausiomis skiltimis, 01 – komandos kodas, 111 – registro imtuvo adresas, 010 – registro siųstuvo adresas.

Nukreipimo komandų grupėje ekonomijos sumetimais komandoms nenurodyti antrojo ir trečiojo baito kodai. Šios komandos yra trijų baitų (išskyrus vieno baito grįžimo iš paprogramės komandas). Antrame (jaunesnioji dalis) ir trečiame (vyresnioji dalis) baituose yra adresas, prie kurio reikia pereiti. Šis adresas vykdant komandas įrašomas į komandų skaitiklį PC.

5.7.3. Programų paketas MP I8085A derinimui

Šis paketas skirtas MP 8085A (I8080,Z80A) programinio aprūpinimo projektavimui ir derinimui. Jam eksploatuoti gali būti panaudoti IBM tipo arba su jais suderinami PC, valdant DOS operacijų sistemai.

Krosinių programų kompleksą sudaro trys pagrindinės programos. Čia neįtrauktas teksto redaktorius, skirtas MP programų pradinių tekstų formavimui, redagavimui ir koregavimui bei jų užrašymui į magnetinį kaupiklį. Gali būti panaudotas bet kuris teksto redaktorius, įeinantis į PC sisteminį programinį aprūpinimą. Priminsime, kad programos pradinio teksto byla būtinai rašoma su plėtiniu **.asm .**

Sistemos makroasemblerio transliatorius verčia programą, užrašytą asemblerio mnemonika, į programą, užrašytą šešioliktainiais kodais. Gaunama vadinamoji objektinė byla. Transliatorius paleidžiamas, įvedus PC monitoriaus ekrane tokią komandinę eilutę avmac85.exe AAA .asm [ENTER], kur avmac85.exe - MP I8085 makroasemblerio vykdomoji byla, AAA.asm - transliuojamos programos teksto bylos pavadinimas.

Transliacijos rezultatas - sukurta objektinė (su plėtiniu **.obj**) ir teksto (su plėtiniu **.prn**) byla. Transliacijos tekste pateikiama tokia informacija (eilės tvarka): komandos adresas, komandos objektinis kodas (šešioliktainiais skaičiais), eilutės numeris, pradinis programos tekstas (asemblerio kalba). Jeigu transliacijos metu pradiniame programos tekste nebuvo aptikta asemblerio kalbos sintaksės klaidų, suformuojama MP programos objektinė byla, skirta įtraukti į MPS atmintį su pradiniu adresu, nurodomu programos tekste su transliatoriaus direktyva ORG (kai šios komandos nėra - programa pradedama nuo nulio). Šiuo atveju sakoma, kad transliatorius formuoja absoliutinius duomenis. Jų priekyje turėtų būti parašyta transliatoriaus direktyva ASEG. Makroasembleris gali formuoti ir kilnojamuosius (adresų erdvėje) programinius duomenis. Tada kilnojamosios programos duomenų priekyje turi būti parašyta krosasemblerio direktyvos CSEG ir DSEG. Tokių programinių duomenų kilnojimą atminties adresų erdvėje atlieka ryšių redaktoriaus

programa (linker). Jeigu MP programa suprojektuota (ir sutransliuota) iš kelių modulių, tai modulių sujungimą į vientisą programą ir atlieka ryšių redaktorius.

Kreipinys į ryšių redaktorių šiuo atveju yra toks: avlink.exe AAA.hex=AAA.obj. xxx.hex bylos formatas gaunamas Intel formos (žr. 62 pav.). Įrašas prasideda žyme "**:**", po to eina eilutėje esančių simbolių skaičius (10-BCD formate) ir pradinis adresas. Tada eina įrašo tipo požymis (00 – duomenys, 01 – įrašo pabaiga). Toliau yra 16 duomenų baitų ir kontrolinė suma. PLM iškviečiamas su vykdomąja PLM byla avsim 85.exe. Ekrane pateikiamos galimos MPS modelio konfigūracijos (vieno MP arba su adapterių I8155, 8355 kombinacijomis). Vartotojas pasirenka norimą variantą, paspaudęs tą variantą atitinkantį simbolio (A, B, C ar D) klavišą. Po to ekrane pasirodo 61 pav., kuriame galima matyti visus MP programinio modelio elementus.

```
    LABEL   OPERATION      8085    AVSIM 8085 Simulator/Debugger      V1.31
    0000H   no   memory        CPU REGISTERS          FLAGS    SCL SPD DSP SKP
CURSOR
    0001H   no   memory     C  Accumulator    Z P S AC  OFF HI  ON  OFF  MENU
    0002H   no   memory     0  00000000:00:_   0 0 0 0       Cycles:
    0003H   no   memory       addr          data              PINS
    0004H   no   memory     PC:0000 n FF FF FF FF FF FF FF FF Intr Bus:00 Intr:0
    0005H   no   memory     SP:0000 n FF FF FF FF FF FF FF FF Rim:0000111 Trap:0
    0006H   no   memory           FF FF FF FF FF FF FF FF          R7.5:0
    0007H   no   memory     BC:0000 n FF FF FF FF FF FF FF FF _________   R6.5:0
    0008H   no   memory     DE:0000 n FF FF FF FF FF FF FF FF _________   R5.5:0
    0009H   no   memory     HL:0000 n FF FF FF FF FF FF FF FF _________    Sid:0
    000AH   no   memory                                      Sod:0
    000BH   no   memory
    000CH   no   memory     Memory Space
    000DH   no   memory     0000 FF FF FF FF FF FF FF FF _________  I/O Address
    000EH   no   memory     0008 FF FF FF FF FF FF FF FF _________  I:0
    000FH   no   memory     0010 FF FF FF FF FF FF FF FF _________  00:_:00000000
    0010H   no   memory     0018 FF FF FF FF FF FF FF FF _________  I:1
    0011H   no   memory     Memory Space                  00:_:00000000
    0012H   no   memory     0020 FF FF FF FF FF FF FF FF _________  I:2
```

0013H no memory 0028 FF FF FF FF FF FF FF FF _____ 00:_:00000000

0014H no memory 0030 FF FF FF FF FF FF FF FF _____ I:3

0015H no memory 0038 FF FF FF FF FF FF FF FF _____ 00:_:00000000

>Select Command - or use arrow keys

Dump Expression commandFile Help IO Load --space-- ESC to screen

61 pav. AVSIM85 programos pagrindinio meniu langas

MP 8085A programinis-loginis modelis visiškai imituoja I8085A architektūrą ir turi 64 kilobaitų virtualią atmintį. PLM branduolys yra dialogo su vartotoju blokas, sudarytas meniu principu.

Į meniu (komandų) režimą ir atgal į vykdymo režimą pereinama pakartotinai spaudžiant klavišą "ESC". Vartotojas iš meniu pasirenka pageidaujamą komandą ir nuspaudžia klavišą "Enter". Pasirinktoji komanda realizuojama meniu komandų apdorojimo paprogramėmis, kurios savo ruožtu naudoja reikalingas programas iš paprogramių bibliotekos. Vartotojas krauna savo programą, iškviesdamas iš komandų grupės komandas "LOAD", "Programm" ir įveda savo programos bylos vardą su plėtiniu .hex. Toliau nuosekliai kreipiantis į komandų grupės "SET" komandas Memory-Map /Random Access/ RAM memory ir nurodant apatinį (Lower address) bei viršutinį (Upper adress) adresą, nustatoma atitinkama PLM atminties sritis, kaip MP DA . Procesorius ir jo registrai nustatomi į pradinę padėtį su komandomis RESET/CPU.

Primename, kad adresai užrašomi šešioliktainėje skaičiavimo sistemoje, skaičiaus išraišką baigiant simboliu "h", o kiekvieną pasirinktą meniu komandą PLM įvykdo tik nuspaudus klavišą "Enter".

PLM suteikia galimybę vartotojui stebėti ir keisti ekrane dviejų atminties langų (2 po 32 ląsteles) turinius. Viršutinį langą vartotojas gali perkelti į bet kurią PLM virtualios atminties sritį, pasirinkdamas tokias komandas: DUMP/1/ABSOLUTE ir nurodydamas lango (DUMP) pradžios adresą. Perkeliant apatinį langą, komandų seka turi būti tokia: DUMP/2/ABSOLUTE bei lango pradžios adresas.

Vartotojas savo programą gali vykdyti ir žingsnio režimu, spausdamas funkcinį klavišą F10 (vienas žingsnis) arba F9 (grąžina atgal per vieną instrukciją). Nuspaudus klavišą F1, paleidžiama vartotojo programa nuo adreso, nurodyto programiniame skaitiklyje PC. Programa bus vykdoma tol, kol bus pertraukta arba baigta . Komandų vykdymo (simuliacijos) greitį galima pakeisti funkciniu klavišu F5. Klavišai F2, F3 ir F4 skirti programos stabdymo žymekliui valdyti. Klavišų Ctrl-C kombinacija užtikrina išėjimą iš bet kurios

komandos į aukštesnio lygio meniu. Iš PLM į DOS grįžtama, įvykdžius komandą “Quit”.

62 pav. Intel formos xxx.hex bylos formatas

Pateiktojo vartotojo darbo su PLM pradžiamokslio pakanka pirmam bendravimo su PLM etapui. Išsamesnes žinias apie PLM ir darbo įgūdžius su juo vartotojas įgaus, dirbdamas su modeliu laboratorinių darbų metu, studijuodamas PLM meniu komandų (ypač “Help”) aprašus.

5.8. MV 80C51 (31,32,52) programavimas

MV programavimo sistemos ypatumą nustato jo naudojimo sritys bei MV struktūrinė realizacija viename DIG. Programų ir duomenų atmintys yra viename kristale su MP. Valdančiosios programos apimtis gali būti nuo kelių iki kelių dešimčių kilobaitų. Programą galima sudaryti naudoant asemblerio ir C (C++) kalbas. Dažniausiai MV programavimo sistemos sudaromos krossistemų principu panaudojant PC.

5.8.1. MV programinis modelis

MV programinis modelis jau buvo aptartas antrame skyriuje (6 - 17 pav.). Programų atmintis turi 16 bitų adresų magistralę, o jos elementų adresus nustato komandų skaitiklio turinys arba komandos, kuriose suformuojami 16-os skilčių adresai.

MV 8051 PA maksimalus talpis gali siekti iki 64 kB, adresuojama ląstelė - vienas baitas. MV 8051 kristale yra įterpta vidinė 4 K*8 programų atmintis, likusius 60K*8 galima organizuoti išorėje, prijungus papildomus IG. (8031 iš

viso jos neturi. Čia tenka naudoti tik išorinę PA). Kadangi programų atmintis organizuota santykiu 4K / 60 K, programuotojas nepastebi, kada MV aritmetinis-loginis įtaisas išrenka baitą iš vidinės PA, o kada – iš išorinės. Išrinkimo strobavimo signalas yra PSEN/, kurį MV suformuoja, kai adreso reikšmė PC viršija 0FFFH (4K) . Be to, MV turi išvadą EA/, į kurį pasiuntus "0", vidinė programų atmintis atjungiama, o dirbti vartotojui galima tik su išorine programų atmintimi (pradedant nuliniu adresu). Kai EA/=1, dirba ir vidinė, ir išorinė PA. Kreipiantis į išorinę PA, visada naudojamas 16-os bitų adresas, kurio jaunesnysis baitas perduodamas per prievadą P0, o vyresnysis baitas - per prievadą P2. Baitas, nuskaitytas iš IPA, įvedamas per prievadą P0 (šiuo atveju P0 prievadas dirba multipleksavimo režimu). Aptarnaujant pertrauktis bei nustatant MV į pradinę būseną, kai kurie konkretūs jaunesnieji PA adresai naudojami trumpoms pertraukčių programoms arba besąlyginio perėjimo komandoms saugoti (0000H - "nustatymas į pradinę būseną", 0003H - INT0, 000BH - T/C 0, 0013H - INT1, 001BH - T/C 1, 0023H - T1+R1).

Vidinė duomenų atmintis sudaryta iš dviejų sričių: 128 baitų DA (0 - 7FH adresai) ir specialių funkcijų registrų srities (80H ÷ FFH adresai).

DA ląstelės gali būti adresuojamos, naudojant tiesioginį ar netiesioginį adresavimo būdą. 32 jauniausieji DA baitai sugrupuoti į keturis bankus po 8 registrus kiekviename. Programos komandos gali kreiptis į registrus, naudodamos jų vardus (R0 - R7). Du programos būsenos registro PSW bitai (RS0 ir RS1) nurodo banko, į kurio registrus kreipiamasi, adresą (numerį). Tokio kreipinio komandos vykdomos žymiai greičiau negu komandos, naudojančios tiesioginį adresavimo būdą. Kiti 16 baitų (20H ÷ 2FH adresai) sudaro DA sritį, kurioje ląstelės kiekvieną skiltį vartotojas gali adresuoti atskirai.

Specialios paskirties registrų (SFR – special function registers) sritį sudaro prievadų registrai-fiksatoriai, taimerių registrai, valdymo registrai ir kt. Į šiuos registrus galima kreiptis tik tiesioginio adresavimo komandomis, be to, vienuolikai specialių funkcijų registrų galima naudoti ir baitinį, ir bitinį adresavimą.

Išorinė duomenų atmintis (iki 64 kB) gali būti sudaryta iš papildomų atminties IG, prijungiamų prie MV. Vidinės ir išorinės duomenų atminties adresai nesikerta, nes kreipiamasi į jas skirtingomis komandomis (pvz., darbui su išorine duomenų atmintimi yra skirtos specialios komandos MOVX, neturinčios jokios įtakos vidinei duomenų atminčiai). Kreipiantis į išorinę DA, visuomet naudojamos komandos su netiesioginiu adresavimu - ląstelės adresas būna arba aktyvaus DA registrų banko registruose R0 ir R1, arba duomenų nuorodos registre DPTR. Pirmuoju atveju bus formuojamas 8 skilčių adresas, antruoju - 16-os skilčių. Adresas bus perduodamas per prievadą P0 (jaunesnysis baitas) ir prievadą P2 (vyresnysis baitas), o mainai (skaitymas ar rašymas) vyks

per P0 (multipleksavimo režimas). Duomenų mainus su išorine duomenų atmintimi lydi MV signalai RD/ ir WR/.

5.8.2. Operandų adresavimo būdai

MCS51 šeimos MV naudojami penki adresavimo metodai:

betarpiškas (immediate)	MOV A,#55h,
tiesioginis (direct)	MOV A,15h,
netiesioginis (indirect)	MOV A,@R0,
išorinis (external)	MOVX A,@DPTR,
netiesioginis perstumtas (code indirect)	MOVCA,@A+DPTR.

Betarpišku adresavimas vadinamas todėl , kad operandas atmintyje iš karto eina po operacijos kodo. Pavyzdyje parodyta , kad akumuliatoriaus turinys įgauna betarpišką šešioliktainę reikšmę 55h. Šis adresavimo būdas labai greitas, nes operando reikšmė yra pačioje komandoje. Bet šis būdas nėra pakankamai lankstus, nes įkraunama reikšmė fiksuojama programos transliacijos metu.

Tiesioginio adresavimo atveju duomenys randami pagal adresą atmintyje. Pavyzdyje parodyta, kad duomenys į akumuliatorių nuskaitomi iš vidinės MV atminties ląstelės, kurios adresas yra 15h. Šis adresacijos būdas irgi pakankamai greitas, nors duomenys nėra komandos dalis. Čia greitis pasiekiamas dirbant su vidine duomenų atmintimi. Šis būdas daug lankstesnis už betarpiškąjį, nes iš atminties paimama reikšmė gali būti kintanti. Pasirinkus šį adresavimo būdą, reikia atsiminti, kad galima naudoti tik vidinės duomenų atminties sritį 00h iki 7Fh. Viršutinė sritis 80h iki FFh yra MV specialios paskirties registrų adresai.

MV 8052 turi 256 baitus vidinės atminties. Į viršutinę jos sritį galima kreiptis tik naudojant netiesioginį adresavimo metodą! Su komanda MOV A,@R0 nuskaitomi duomenys iš vidinės duomenų atminties, kurios adresas saugomas registre R0. Taigi, norint į akumuliatorių įkrauti duomenis iš atminties ląstelės, kurios adresas 88h, visų pirma reikia nustatyti registro R0 turinį su komanda MOV R0,#88h, ir tik paskui vykdyti komandą MOV A,@R0.

Šis adresavimo būdas netinka, kreipiantis į bendrosios paskirties registrus.

Į išorinę duomenų atmintį galima kreiptis, naudojant išorinį tiesioginį adresavimo būdą. Yra tik dvi šiuo būdu veikiančios komandos MOVX A,@DPTR ir MOVX @DPTR,A. Abi jos kaip rodyklę naudoja registrą DPTR. Taigi, pasirinkus šį adresavimo būdą, reikia nustatyti DPTR.

Minėtų adresavimo būdų kombinacijos leidžia realizuoti 21 adresavimo būdą.

5.8.3. Komandų sistema

Komandų sistemą sudaro 111 bazinių komandų. Komandą sudaro vienas, du arba trys baitai. Pirmas baitas visuomet nurodo operacijos kodą. Antras ir trečias baitas turi arba operando adresą, arba tiesiogiai patį operandą. Operandai gali būti keturių tipų: bitai, 4 bitų skaičiai, baitai ir 16-os bitų žodžiai. Komandos įvykdomos per vieną, du arba keturis (daugyba ir dalyba) mašininius ciklus. Jeigu MV sinchronizacijos dažnis yra 12 MHz, vieno ciklo komanda įvykddoma per 1mks., dviejų - per 2 mks.

Komandų sistemoje nėra įvesties (IN) ir išvesties (OUT) komandų. Šių komandų funkcijas gali atlikti kreipimosi į registrus komandos (pvz., MOV direct, A).

Programos būsenos žodį (PSW) sudaro 4 požymiai: C - keliamojo vieneto, AC - tarpinio keliamojo vieneto, OV - perpildymo ir P - lygiškumo (pariteto).

Požymis P tiesiogiai priklauso nuo A bitų, kurių reikšmė lygi “1”, skaičiaus (P=1, kai vienetinių A bitų skaičius nelyginis, o jeigu lyginis, tai P=O). AC požymis, vartojamas atliekant aritmetinius veiksmus su dvejetainiais-dešimtainiais (hexadecimal) skaičiais; C=1, jeigu iš vyriausiojo rezultato bito vyksta perkėlimas arba skolinimas, o OV=1 - jeigu atliekant sudėties (atimties) operacijas, rezultatas netelpa į 7 bitus, ir aštuntasis bitas negali būti interpretuojamas kaip ženklo bitas (dalijant iš nulio OV=1, dauginant OV=1, jeigu daugybos rezultatas >255). Priede pateikta komandų, kurios veikia požymius, sąrašas. Lyginumo požymį veikia visos komandos, keičiančios A turinį. Reikėtų atkreipti dėmesį į tai, kad MV asembleris leidžia naudoti programose simbolinius specialių funkcijų registrų ir jų atskirų bitų vardus, pvz., šeštojo A bito simbolinis vardas komandoje atrodys taip: ACC.5.

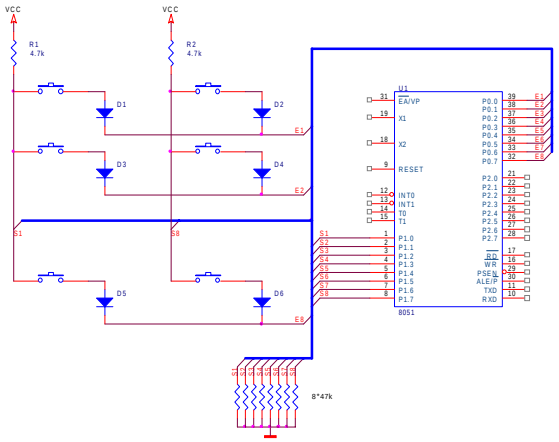
MV komandų sistemą sąlyginai galima suskirstyti į penkias grupes:

- aritmetinės komandos;
- loginės komandos;
- duomenų perdavimo komandos;
- operacijų su bitais komandos;
- nukreipimo komandos.

Priedo 31-35 lentelėse pateiktas komandų sąrašas.

5.8.4. Projektų pavyzdžiai

Programuojant įterptines sistemas, labai dažnai tenka projektuoti mygtukų pultą. Tam geriausiai tinka matricinis klaviatūros sudarymo principas. 63 pav. pateikta 8*8 matricinės klaviatūros principinė schema.



63 pav. Matricinės klaviatūros prijungimo schema

Matricinę klaviatūrą sudaro 64 mygtukai, kurie yra padalyti į 8 stulpelius po 8 eilutes kiekviename iš jų (63 pav. suprastintai pavaizduoti tik du stulpeliai ir trys eilutės). Matome, kad principinė schema yra labai paprasta. Jai valdyti yra panaudoti du MV MCS51 prievadai (P0 ir P1). Prie prievado P0 yra prijungtos aštuonios eilutės, prie P1 - aštuoni stulpeliai. Prie kiekvieno stulpelio yra prijungti aštuoni rezistoriai po 47 kiloomus prie maitinimo šaltinio nulinio potencialo. Tai padaryta tam, kad atvirame prievado P1 įėjime būtų užtikrintas loginio nulio potencialas. Kiekvienas stulpelis per 4,7 kiloomo rezistorių prijungtas prie +5V maitinimo įtampos šaltinio. Tokiu būdu užtikrinamas loginio vieneto potencialas prievado P1 įėjime, kai yra nuspaustas mygtukas. Diodai yra panaudoti apsaugoti prievadus tuo atveju, kai vienu metu yra nuspaudžiami keli mygtukai.

Valdymo programa nuosekliai nuskaito aštuonių stulpelių kodus ir patalpina juos atmintyje. 64 pav. pateiktas nuspaustų mygtukų kodų nustatymo paprogramės tekstas.

```
; Matricinės klaviatūros 8*8 skenavimo paprogramė
; Ji nuskaito 64 mygtukų būseną ir ją patalpina
; vidinės RAM ląstelėse 20h - 27h
Scan:   Mov R0,#20h          ; nustatomos rodyklės į RAM
        Mov R1,#28h
        Mov A,#80h           ; nustatomas 1 į vyriausio bito poziciją
Pradžia:
        Mov P1,A             ; nustatyti stulpelį
        RR A                 ; postūmis sekančiam
        Mov R2,A             ; atsiminti einamąjį kodą
        Mov A,P0             ; skaityti eilutes
        XCH A,@R0
        Mov @R1,A            ; išsaugoti
        Inc R0               ; inkrementuoti rodykles
        Inc R1
        Mov A,R2
        JNB ACC.7,Pradžia    ;ciklas visiems 8 stulpeliams
        Ret
```

64 pav. Matricinės klaviatūros 8*8 valdymo paprogramė

65 pav. pateikta universalaus valdiklio (UV) principinė schema. Jame panaudotas MCS51 šeimos MV 80C552 (U1), kurio sudėtyje yra aštuonių kanalų po dešimt bitų analoginių duomenų surinkimo sistema. UV sudėtyje yra:

- 64K*8 27C512 tipo EPROM (U4) [www.atmel.com];
- 32K*8 DS1230AB tipo NVSRAM [www.dalsemi.com] (U5);
- MG1206 tipo skystųjų kristalų indikatorius (U6);
- FX909A GMSK 19,2 kbps modemo IG [www.cmlmicro.co.uk] (U7);
- VL16C450 tipo ACE (asynchronous communication element) [www.ti.com] (U12);
- 24C02 tipo nuosekli EEPROM [www.national.com], (U23);
- DS1228 tipo dvipusis TTL-RS232 keitiklis [www.dalsemi.com] (U22);
- DS1287 tipo realaus laiko laikrodžio IG [www.dalsemi.com] (U3);
- šešių kanalų impulsų skaitiklio blokas (U25 ir U26);

- infraraudonųjų spindulių nuoseklių komunikacijų blokas [www.hp.com] (U29 ir U30);
- aštuonių diskretinių įėjimų su galvanine izoliacija blokas (ISO1-ISO8);
- aštuonių diskretinių išėjimų su galvanine izoliacija blokas (U19,U8,U9,U10,U16,U18,U21,U24,U28);
- aštuonių diskretinių išėjimų be galvaninės izoliacijos blokas (U27);
- 74HC138 tipo adresų dešifratorius (U11);
- sąsajos RS485 formavimo schemos (U13 ir U17).

Toks UV, kaip įterptinis mazgas, gali būti panaudotas įvairiems komunikacijų, valdymo, matavimų ir kitiems uždaviniams spręsti. Jame panaudoti KMDP elektroninės struktūros IG vartoja labai nedidelę maitinimo srovę (apie 80 mA).

UV vartotojas turi galimybę pasirinkti keturis komunikacijų būdus. Tai vielinės arba belaidės komunikacijos per GMSK modemą, kuris užtikrina 19,2 kbps mainų greitį. Prisijungus CB radijo stotį, galima realizuoti pusiau dupleksinius mainus. IrDA standartu funkcionuojantis komunikacijų infraraudonaisiais spinduliais mazgas užtikrina belaides komunikacijas su kitu valdikliu arba PC 115,2 kbps greičiu, kai atstumas neviršija 6 m. Standartinės sąsajos RS232 pagalba galima komunikuoti standartiniais greičiaias su kitais UV arba PC. Sąsaja RS485 gali būti panaudota komunikacijoms didesniais atstumais (naudojant vyniotų porų penktos kategorijos FTP kabelį, ryšio atstumas gali siekti 2-3 km, o mainų greitis iki 1,5 mbps).

Pateiksime keletą atskirų UV mazgų programavimo pavyzdžių C kalboje.

1. Reikia išmatuoti įėjimo įtampą, kuri prijungta jungties JP4.1 įvade. Įtampa yra nuolatinė, kintanti diapazone (0 iki +5V).

Žemiau yra pateiktas nuolatinės įtampos keitimo į 10-ties skilčių kodą programos tekstas

```
/* --- Itampos matavimas --- */
unsigned int Itampa (void)
{
  unsigned char ASK;
  ADCON=0;                          //  įjungti 0 ASK kanalą
  ADCON |= 0x08;                    //  pradėti  keitimą
  while(!(ADCON & 0x10)) ;          //  laukti keitimo pabaigos
  ASK=(ADCON & 0xC0)>>6;
  return(ADCH+256*ASK);             // grąžinti  10-ties bitų įtampos kodą
 }
```

Keitimą valdo MV vidinis registras ADCON. Jo trys jauniausios skiltys nustato analoginio multipleksoriaus kanalą. Kanalai numeruojami nuo 0 iki 7. Kadangi matuojama įtampa yra prijungta prie 0-nio kanalo, tai į ADCON iš pradžiių įrašome nulį. Toliau į trečiąją skiltį įrašome vienetą ir prasideda keitimo procesas. Jo trukmė priklauso nuo MV sinchronizacijos dažnio ir yra apie keliolika mikrosekundžių. Keitimo pabaiga nustatoma, kai ADCON ketvirtoje skiltyje pasirodo loginis vienetas. Keitimo rezultato aštuonios jaunesniosios skiltys yra registre ADCH, o dvi vyriausiosios - šeštoje ir septintoje ADCON skiltyse. Perstūmę pastarąsias per šešias skiltis į dešinę, padauginę iš 256 ir sudėję su ADCH turiniu, gausime rezultatą. Jis bus patalpintas dvibaičiame unsigned int formato skaičiuje, kurį funkcija grąžins į pagrindinę programą. Gauto skaičiaus kitimo diapazonas yra nuo 0 iki 1023. Jeigu mums reikalinga rezultatą gauti įtampos matavimo vienetais, turime atlikti normavimą. Matavimo skiriamoji geba arba ASK jauniausios skilties vertė bus 5/1024=4.88mV. Keitimo rezultatą padauginę iš šio skaičiaus gausime įėjimo signalo vertę milivoltais. Šiuo atveju programos tekstas kiek pasikeis.

```
/* --- Itampos matavimas --- */
float Itampa_float (void)
{
  unsigned char ASK;
  ADCON=0;                          // įjungti 0 ASK kanalą
  ADCON |= 0x08;                    // pradėti keitimą
  while(!(ADCON & 0x10)) ;          // laukti keitimo pabaigos
  ASK=(ADCON & 0xC0)>>6;
  return(4.88*(float)(ADCH+256*ASK));          // grąžinti 10-ties bitų
įtampos kodą
  }
```

Čia tenka naudoti slankaus kablelio skaičių vaizdavimo formą.

Abiem aukščiau pateiktais atvejais keitimas organizuojamas ASK būsenos apklausos būdu keitimo pradžią inicijuojant programiniu būdu, nes į MV įėjimą STADC yra pastoviai paduotas loginio vieneto potencialas.

Esant reikalui ASK keitimo procedūrą galima realizuoti ir programos pertraukties būdu, nes keitimo pabaigos signalas gali generuoti vidinį pertraukties signalą, kurio vektoriaus adresas yra 53h. Keitimas gali būti pradėtas ir nuo išorinio įvykio, kuris turi patekti į įėjimą STADC.

2. Išmatuotą įtampos reikšmę reikia išvesti į skystųjų kristalų indikatorių.

Žemiau pateikiamos C kompiliatoriaus preprocesoriaus direktyvos ir indikatoriaus valdymo funkcijos.

```
// Skystųjų kristalų indikatoriaus registrų adresų nustatymas

/*++++++ LCD valdymo registro adresas ++++++++*/
at 0x8100 xdata char IND_BUS;
/*++++++ LCD isvedimo duomenu registras ++++++*/
at 0x8102 xdata char IND_ISV;
/*++++++ LCD busenos registras +++++++*/
at 0x8101 xdata char IND_IV;

//     LCD režimo valdymo baito išvedimo) funkcija
//      Išveda LCD darbo režimo nustatymo baitą
void print(unsigned char c)
{
   while((IND_IV & 0x80)!=0)  ;
   IND_BUS=c;
}

//  Indikuojamo simbolio išvedimas i LCD
// Išveda indikuojamą simbolį , perduodamą ASCII pavidale
void printd(unsigned char c)
{
   while((IND_IV & 0x80)!=0)  ;
   IND_ISV=c;
}

// Simbolių eilutes išvedimas i LCD
// Isveda simbolių eilutę , tikrindamas jos pabaigos simbolį NULL
void priint(str)
char *str;
{    char ii;
       for(ii=0; str[ii]; ii++)
                printd(str[ii]);
}
```

Pagrindinė programa, naudodama aukščiau pateiktas funkcijas atlieka matavimą ir rezultato indikaciją.

```
main ()
{
unsigned int Voltai;
```

```
    unsigned char buffer[50],i;
    Voltai=Itampa();
    sprintf(buffer,"Itampa=%02dV",Voltai);

    print(0x01);       // Ištrinti indikatoriaus parodymus
    print(0x38);       // indikatoriaus darbo režimas
    print(0x80);        // pirmo išvedamo simbolio pozicija 1 eilutėje kairiajame
krašte
    priint(buffer);
     }
```

3. Per sąsają RS232 priimti komandą iš PC, ją identifikuoti. Identifikavus išmatuoti įtampą ir jos reikšmę grąžinti į PC.

Šiam uždaviniui realizuoti panaudosime aukščiau pateiktas ir papildomas funkcijas.

```
    /*--- Simbolio perdavimo per UART funkcija --- */
     void ras(unsigned char simb)
     {
       S0BUF=simb;
       while(!TI) ;
       TI=0;
    }
      /* --- Simbolio priemimo per UART funkcija ---
    unsigned char sk()
       {
          while(!RI);
          RI=0;
          psimb=S0BUF;
          return (psimb);
       }

    void UART (void) interrupt 4 using 2
    {
     if( sk()  = = 'A' )         // jeigu atpažinta komanda
       {
       Voltai=Itampa();    // matuoti įtampą
        ES=0;                 // drausti perrauktis iš UART
        ras(Voltai/256);     // perduoti dviem baitais. Pirma jaunesnįjį,
        ras(Voltai%256);   // po to vyresnįjį
        ES=1;                 // vėl leisti perrauktis iš UART
       }
```

```
}
```

Uždavinyje panaudotas pagrindinės programos pertraukties būdas. Kai priimamas simbolis iš PC ir jo kodas atitinka A raidės kodą, kviečiama įtampos matavimo paprogramė. Išmatuota įtampos reikšmė perduodama per UART dviem nuosekliais baitais.

4. Matuoti įtampą 500ms laiko tarpais ir jos reikšmę 19,2kbps greičiu perduoti į PC. Kas 500ms regeneruoti vidinį MV WD.

Šiam uždaviniui spręsti panaudosime realaus laiko laikrodžio DS1287 IG. Nustatysime, kad jis kas 500ms formuotų pertraukties signalą į įėjimą INT0 (tam reikia įdėti trumpiklį JP2). Jo aptarnavimo funkcija turi matuoti įtampą ir jos reikšmę perduoti į PC. Šiam uždaviniui realizuoti panaudosime aukščiau pateiktas ir papildomas funkcijas.

Preprocesoriaus direktyvas reikia papildyti DS1287 registrų adresų nustatymu

```
at 0x800A xdata char REG_A;
at 0x800B xdata char REG_B;
at 0x800C xdata char REG_C;
at 0x800D xdata char REG_D;
```

```
// MV pradinis nustatymas
void pradzia(void)
{
 TMOD=0x2D;      // nustatome T1 autoreload darbo režimui
 ES0=0;          // draudžiame pertrauktis iš UART
 TR1=1;
 TH1=0xFD;       //    11.059 MHz dažnio dalijimo koef-tas greičiui 9.6 kbps
 SM0=0;          //  UART darbo režimo nustatymas
 SM1=1;
 SM2=0;
 REN=1;
 REG_B=0x4f;
 REG_A=0x2F;     /* DS1287 pertraukčių periodas  500 ms */
 EA=1;
 EX0=1;
 PCON=0x90;      // komunikacijų greitis 19,2 kbps
 }
    // Pertraukties įėjime INT0 aptarnavimo funkcija (naudoja 1 registrų banką)
    void INT0 (void) interrupt 0 using 1
```

```
{
char tr;
Voltai=Itampa();     // matuoti įtampą
 ras(Voltai/256);    // perduoti dviem baitais. Pirma jaunesnįjį,
 ras(Voltai%256);    // po to vyresnįjį
PCON=0x90;           // regeneruoti WD DS1232
T3=0;
tr=REG_C;            // gesinti aptarnavimo požymį
}
```

5. Suprogramuoti duomenų mainus per modemą FX909

Modemas FX909 (arba panašus) dirba pusiau dupleksiniu režimu, todėl siuntimas ir priėmimas yra visiškai nesusiję ir programuojami atskirai.

Siunčiant reikia:

1) inicializuoti modemą;
2) suformuoti ir išsiųsti paketo "pradžią" (header);
3) perduoti duomenis.

Modemo inicializacijos metu yra užpildomi jo vidiniai registrai, kurie nustato darbo režimą.Toliau yra formuojama paketo "pradžia", susidedanti iš 6 baitų (jei reikia, galima formuoti ir naudojant didesnį baitų skaičių); pirmieji du baitai yra naudojami bitų sincronizacijai. Toliau esantys du baitai reikalingi paketo sinchronizacijai. Pagal juos modemas nustatomas paketo pradžiai. Paskutiniai du baitai yra siunčiami vartotojo nuožiūra ir gali būti naudojami kaip komandos.

Po paketo "pradžios" yra siunčiami duomenys, t.y. jie tiesiog surašomi į modemo duomenų buferį ir perduodama komanda "Siųsk".

Toliau pateikiamas siuntimo programos fragmentas:

```
<…>
//*******INICIALIZACIJA**********
    do
        {
    IE_0=1;                  //Pertrauktis iš modemo leisti
    IE_7=1;                  //Visas pertrauktis leisti
    TCON=0x05;
    XBYTE[COMM]=RES;         //"Reset" komanda modemui
    XBYTE[CTRL]=0x70;        // Nustatyta 01110000 – kvarcinis
                               generatorius 9,8304 MHz, daliklis 512,
                               duomenų perdavimo sparta 19,2kbps
```

```
        XBYTE[MODE]=0xB0;       // Nustatyta 10110000 – pertrauktis
                                valdikliui leisti, duomenų neinvertuoti,
                                siuntimo režimas, leidžiama “scramble”
                                operacija.
        STBUS=XBYTE[STAT];     // Nuskaitomas būvio registras
        }
        while (!(STBUS&0x40));  // Jei buferis nelaisvas, inicializaciją
                                pakartoti
//*****PAKETO “Pradžios ” FORMAVIMAS IR SIUNTIMAS*****
        XBYTE[DBUF]=0xCC;       // 2 bitų sinchronizacijos baitai
        XBYTE[DBUF]=0xCC;
        XBYTE[DBUF]=0xA2;       // 2 paketo sinchronizacijos baitai
        XBYTE[DBUF]=0x65;
        XBYTE[DBUF]=0xD8;       // 2 komandos baitai
        XBYTE[DBUF]=0x21;
        DELAY (6);              // 2 bitų periodų vėlinimas  filtro
                                    būsenai stabilizuotis
        XBYTE[COMM]=T7H;        // Siųsti  paketo “pradžią”
        EXTINT ();              // Laukti pertraukties

//*********DUOMENŲ SIUNTIMAS***************
        IE_0=0;                 //Kol duomenys bus rašomi į modemo
                                //buferį, drausti pertrauktį
        WRITE ();               // Surašyti duomenis į buferį
        XBYTE[COMM]=TDB;        // Siųsti duomenis
        IE_0=1;                 // Leisti pertrauktį
        EXTINT ();              // Laukti pertraukties
//********”HANG” BAITAS*************
        XBYTE[DBUF]=HBYTE;      // “hang” baitą įrašyti į buferį . . .
        XBYTE[COMM]=TSB;        // ir išsiųsti
```

```
EXTINT ();                  // Laukti, kol ištuštėjęs siuntimo buferis
                            //suformuos pertraukties signalą
<...>
```

Siuntimo pabaigoje dar išsiunčiamas vadinamasis “hang” baitas, kurį imtuvas ignoruoja. Jis reikalingas tam, kad siųstuvui baigiant dirbti nebūtų prarasta duomenų pabaiga.

Priėmimas realizuojamas analogiškai. Jį programuojant reikalinga:

1) modemo inicializacija;
2) paketo sinchronizacija;
3) duomenų priėmimas.

Modemo inicializacija turi tokią pat prasmę kaip ir siuntimo režime, t.y. modemas paruošiamas darbui.

Paketo sinchronizacija naudojama jo pradžios aptikimui; jei dirba keli modemai, tai ji atlieka ir adresacijos vaidmenį. Tik aptikęs savo kombinaciją, susidedančią iš dviejų baitų, modemas priims duomenis.

Toliau yra priimami duomenys. Jei bet kuriame iš šių etapų aptinkama klaida, kurios modemas negali ištaisyti, jis nutraukia priėmimą ir pradeda viską iš pradžių (tai reikia numatyti ryšio protokole).

Priėmimo programos fragmentas:

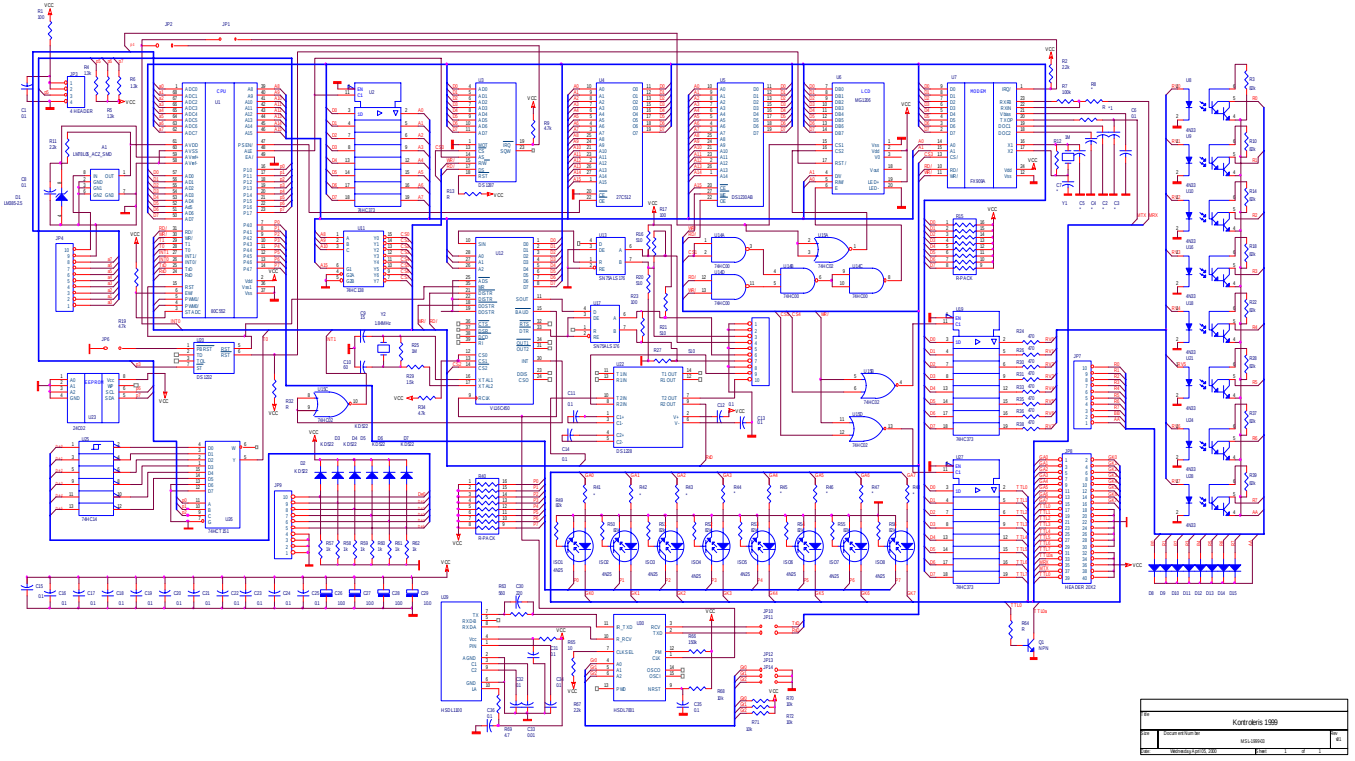
```
//********INICIALIZACIJA***************
BACK:
    IE_0=1;             //Pertrauktis iš modemo leista
    IE_7=0;             // Visus kartu drausti
    TCON=0x05;
    XBYTE[COMM]=RES;    //”Reset” komanda
    XBYTE[CTRL]=0x7E;       // Nustatyta 01111110 – kvarcinis
                            generatorius 9,8304MHz, dalikis 512,
                            duomenų perdavimo sparta 19,2kbps, PLL
                            juosta vidutinė, detektuoti nykstantį signalą
                            (lossy peak detect)
    XBYTE[MODE]=0x90;   // Nustatyta 10100000 – pertrauktis leisti,
                            duomenų neinvertuoti, priėmimo režimas,
                            leisti “scramble” operaciją
```

```
        DELAY (6);              // Dviejų bitų periodų vėlinimas reikalingas
                                tam, kad leisti nusistovėti žemo dažnio filtro
                                pereinamiesiems procesams
//*************BITŲ SINCRINIZACIJA****************
        XBYTE[DBUF]=0xA2;   // Du baitai – paketo sinchronizacijos baitai,
                                reikalingi aptikti paketo pradžiai
        XBYTE[DBUF]=0x65;
        XBYTE[COMM]=LFSB1; //Komanda, duodama modemui aptikus
                                 //nešantyjį signalą,  reikalinga bitų
                                 // sinchronizacijai
        IE_7=1;                  //Pertrauktis leisti
        EXTINT();                // Laukti pertraukties
        if (bb)                    // Jei bus klaida,  nustatyti "bad" bitą  bb,
                                priėmimą nutraukti ir  programą vykdyti nuo
                                pradžių
                {
                bb=0;
                goto BACK;
                }
  //*************PAKETO SINCRONIZACIJA**************

        DELAY (28);             // 12 bitų periodų vėlinimas reikalingas
                                //automatiniam signalo lygio nustatymui bei
                                //automatiniam bitų sinchronizacijos
                                //nustatymui
        XBYTE[COMM]=SFH; // "Ieškoti paketo pradžios"
        EXTINT();               // Laukti pertraukties
        if (bb)                 //Tikrinti "bad" bitą
                {
                bb=0;
                goto BACK;
```

```
                }
//*************DUOMENŲ PRIĖMIMAS**************

        XBYTE[COMM]=RDB;    // Priimti duomenų bloką
        EXTINT();                   //Laukti  pertraukties
        if (bb)                      // Tikrinti “bad” bitą
                {
                bb=0;
                goto BACK;}
```

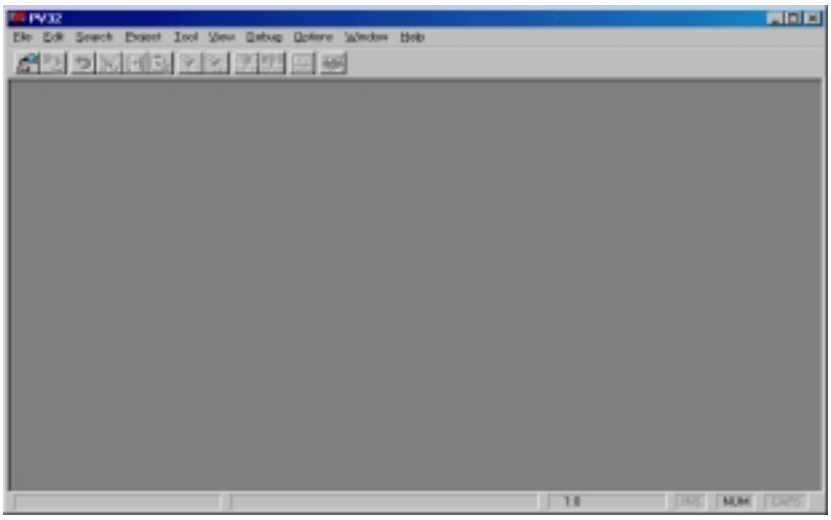


65 pav. Universalaus valdiklio principinė schema

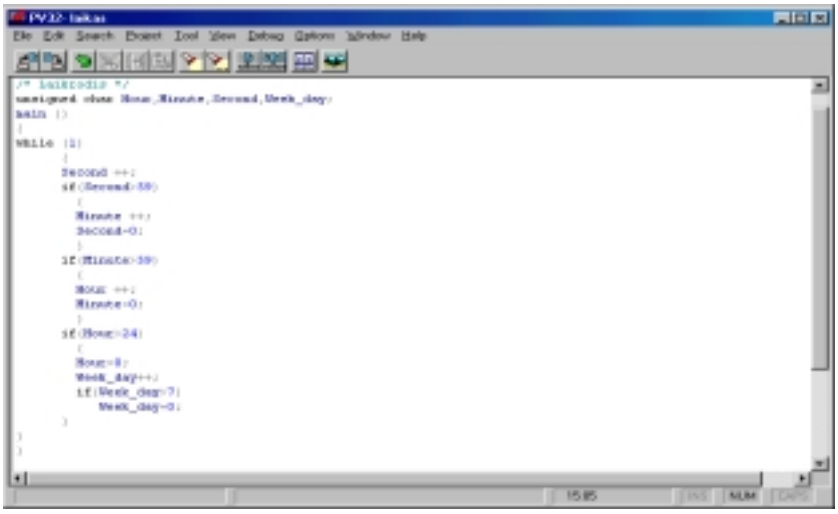
5.8.5. MCS51 derinimo programų paketas

Programoms derinti galima naudoti programų paketą AVSIM51, dirbantį DOS aplikoje. Jis skirtas MPS su MV 80C51 (ir giminingų) programinių priemonių projektavimo ir derinimo procesui asemblerio kalboje automatizuoti. Šias programines priemones sudaro teksto redaktorius, asemblerio transliatorius, ryšių redaktorius bei programinis-loginis modelis. Komplekso techninis aprūpinimas, operacinė aplinka bei darbo su juo pradžiamokslis visai sutampa su MP I8085 komplekso analogiškomis priemonėmis. Daug patogesnis yra Windows 95/98/NT aplinkai skirtas derinimo paketas PV32. Jį pasitelkus galima ruošti programinę įrangą ne tik asemblerio, bet ir C kalba. Toliau pateikiamas trumpas darbo su šiuo paketu aprašymas.

Pirmiausia paleidžiama MP darbo modeliavimo aplinka, du kartus paspaudus “pelės” mygtuką ant ikonos PV32 (žr. 66 pav.).

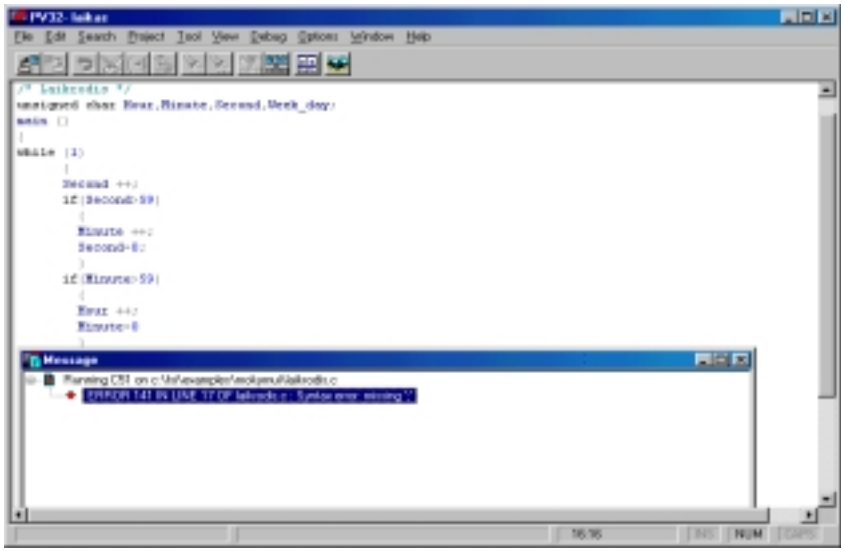


66 pav. Pagrindinis programos PV32 langas



67 pav. Programos teksto langas

Po to “pelės” rodyklė statoma pozicijoje “Project”, ir čia, paspaudus kairįjį mygtuką, atsidaro projekto formavimo langas, kuriame pasirenkamas naujas “New” arba atidaromas esantis projektas . Jo plėtinys **.prj** .



68 pav. Pranešimo langas

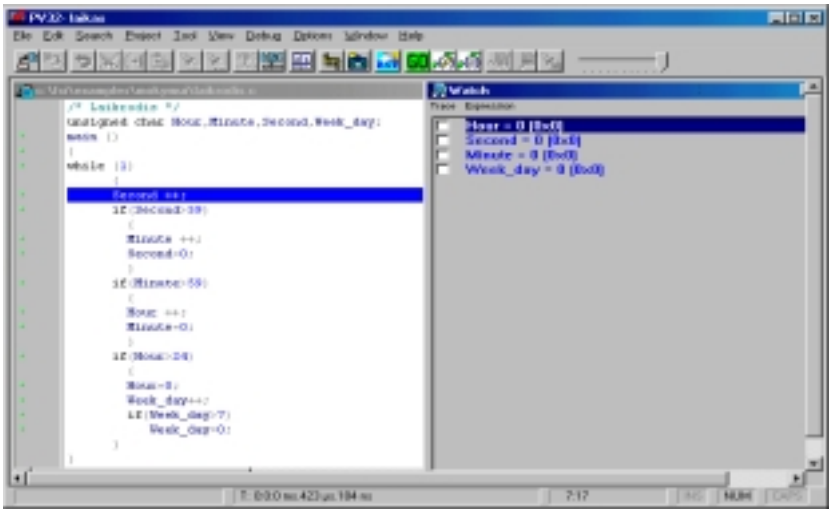
Laboratorijos PC disko kataloge **“Mokymui”** yra sudarytas projektas **mokymui.prj** . Ant šio pavadinimo statome “pelės” rodyklę ir spaudžiame mygtuką “Open”. Po to pasirodo C kalba parašytos programos, realizuojančios paros laiko skaičiavimą, tekstas (žr. 67 pav). Programą sudaro pagrindinė programa “Main”, į kurios sudėtį įeina begalinis ciklas While(1). Pastarajame yra didinamas sekundžių skaitiklis “Second”, tikrinama sąlyga, ar jis neviršija skaičiaus 59, jei viršija, vienetu didinamas minučių skaitiklis “Minute”. Toliau analogiškai tikrinamos minučių skaitiklio kitimo ribos ir, jeigu skaitiklis viršija skaičių 59, didinamas valandų skaitiklis “Hour”. Valandų skaitiklio kitimo ribos 00-23. Pasibaigus parai, didinamas savaitės dienos skaitiklio turinys, kurio kitimo ribos 1 – 7.

Šiame lange galima redaguoti programos tekstą. Suredaguota programa kompiliuojama: tam tikslui “pelės” rodyklė statoma prie užrašo “Project” ir spaudžiamas mygtukas “Build all”. Čia programa kompiliuojama ir formuojamas pranešmų langas “Message”. Jeigu programoje yra sintaksės klaidų, tai bus pateiktas konkretus pranešimas su nuoroda į klaidingą programos eilutę. 68 pav. yra parodytas pranešimo atvejis, kai 17 eilutėje pamirštas kabliataškis. Jeigu programa sukompiliuota be klaidų, galima pereiti prie jos darbo tikrinimo: “pelės” rodyklė statoma prie užrašo “Debug” ir spaudžiamas “pelės” mygtukas. Ekrane pasirodo vykdymo modeliavimo langas (69 pav.). Programą derinti labai patogu, stebint kintamųjų reikšmes. Tada reikia pereiti į pagrindinio lango “Meniu” ir, pastačius “pelės” rodyklę prie užrašo “Window”, pasirinkti punktą “Tile debug”. Po to dešinėje pusėje pasirodys langas “Watch”, kuriame stebėsime kintamųjų reikšmes. Dabar pozicijoje “New Value” įrašoma reikalinga nauja reikšmė. Tada spaudžiamas mygtukas “Modify”.

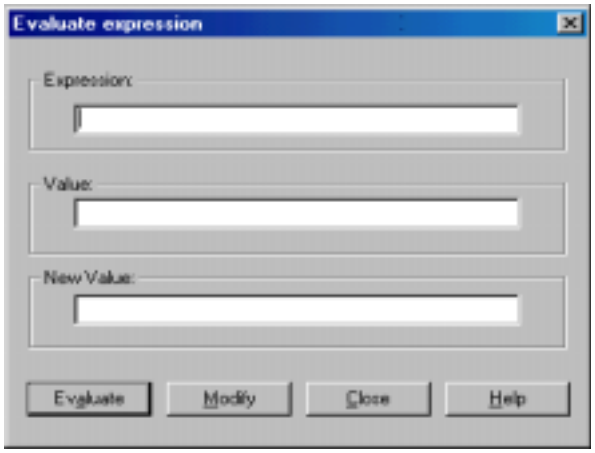
Galima pažingsniui tikrinti programos darbą. Tai atliekama, spaudant klaviatūros mygtuką F7. Prireikus kintamųjų reikšmes galima pakeisti, paspaudus kombinaciją Ctrl+M. Tada ekrane pasirodo 70 pav. langas. Čia pozicijoje “Expression” nurodomas kintamojo vardas (pvz., Minutė) .

Programą galima paleisti vykdyti ir automatiniu režimu. Tada spaudžiamas mygtukas “GO”, o dešinėje pusėje esančia rodykle galima valdyti modeliavimo greitį. Taip derinant programą, labai patogu įstatyti stabdymo taškus. Juos reikia įdėti tose vietose, kur svarbu patikrinti programos vykdymo eigą.

Stabdymo taškai įstatomi (gesinami) mygtuku F5 (žr. 71 pav.). Pažymėta eilutė paryškinama raudona spalva. Modeliavimą sustabdyti galima mygtuku “STOP”. Mikroprocesorius į pradinę būseną nustatomas mygtuku RST. Modeliuojant galima įvertinti programos etapų įvykdymo laiką (71 pav. apačioje).

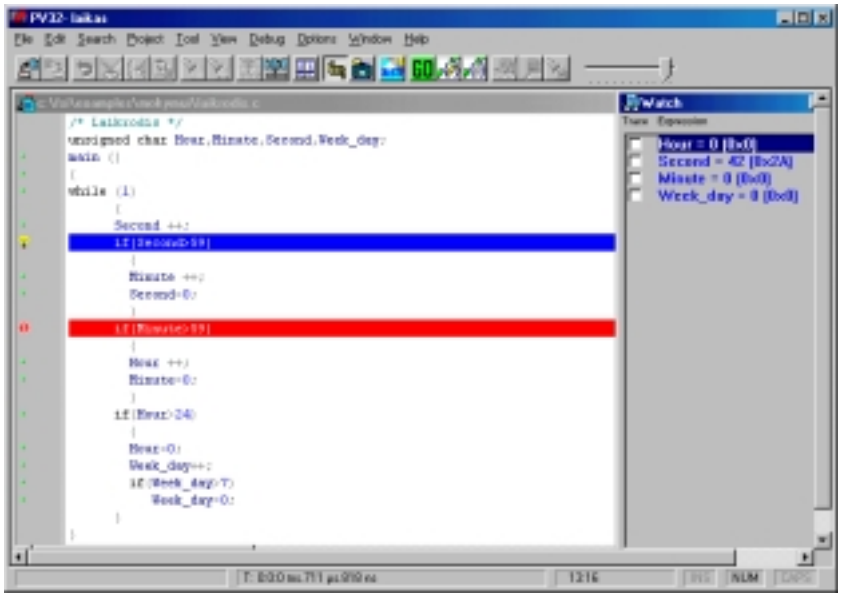


69 pav. Darbo modeliavimo langas



70 pav. Kintamojo turinio modifikacijos langas

Šiuo paketu galima derinti ir asemblerio kalba parašytas programas. Atsakymus į iškylančius klausimus galima rasti “Help” lange.



71 pav. Stabdymo taško įterpimas

5.9. Kontroliniai klausimai

1. Apibūdinkite modulinį ir "žemėjantį" programų projektavimo metodą.
2. Kas yra struktūrinis programavimas?
3. Apibūdinkite pagrindinius programų kūrimo technologinio proceso etapus.
4. Kuo skiriasi ir kuo panašios rezidentinių ir krosinių programų kūrimo sistemos?
5. Koks yra darbinių programų sukūrimo proceso algoritmas?
6. Kaip klasifikuojamos MPS programavimo kalbos?
7. Išvardykite keturis privalomus asemblerio eilutės laukus ir paaiškinkite jų paskirtį.
8. Pagrįskite teiginį: paprogramės yra modulinio programavimo pagrindas.
9. Į kokias dalis rekomenduojama suskirstyti kiekvieno pradinio programinio modulio tekstą? Kokia informacija įtraukiama į kiekvieną dalį?
10. Kuriomis komandomis MP kreipiasi į paprogrames? Kaip tas komandas vykdo MP? Kokiais būdais perduodami duomenys iš pagrindinės programos paprogramei, ir atvirkščiai?
11. Paaiškinkite programų rankinio testavimo metodo esmę.

12. Kokie tipiniai darbo režimai naudojami PLM? Pateikite programos derinimo su PLM darbų seką?

13. Išvardykite MP I8085 mazgus, į kuriuos vartotojas gali kreiptis komandomis (MP programinis modelis).

14. Kokie operandų adresavimo būdai naudojami MP I8085 komandose? Paaiškinkite jų esmę.

15. Kokie pagrindiniai programiniai paketai sudaro MP I8085 krosinių programų kompleksą? Kokia jų paskirtis?

16. Kas sudaro MV I8051 programinį modelį?

17. Kokius operandų adresavimo būdus naudoja MP I8085? Paaiškinkite jų esmę?

6. MPS atsparumo trikdžiams užtikrinimas

Netgi be schemotechninių klaidų sukurta įtaiso principinė schema kartu su programine įranga, įrašyta į PA, negarantuoja, kad toks įtaisas jį pagaminus ir prijungus prie objekto veiks patikimai, jeigu nėra pasirūpinta atsparumo trikdžiams priemonėmis.

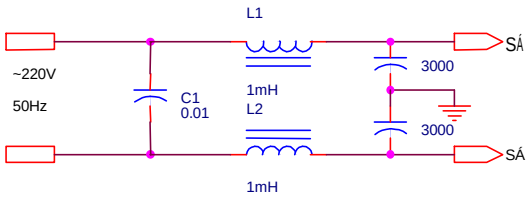
Įtaiso su MP ar MV darbas gali būti nestabilus, jis gali " pakibti" dėl to, kad pirminio maitinimo tinklo laidai yra greta signalinių laidų, kad maitinimo šaltinis praleidžia tinkle atsirandančius impulsinius trikdžius, kad nestabili įtaiso maitinimo įtampa ir pan.

Toliau aptarsime nestabilumo šaltinius ir jų šalinimo priemones.

6.1. Trikdžių, sukeliamų pirminio maitinimo tinkle, slopinimas

Pramoninio maitinimo tinklo (~ 220 v, 50 Hz) forma kai kada gali labai skirtis nuo sinusinės: galimi iškilimai, įdubimai, lėtas amplitudės kritimas ir t.t.. To priežastis – pereinamieji procesai dėl komutacijos, staigūs apkrovimo galios pasikeitimai, pvz., įjungus galingą elektros variklį, elektrinę krosnį, žaibas ir pan. Todėl pagal galimybes būtina tokius trikdžius slopinti.

Trumpalaikius trikdžius slopinti galima pasyviniais filtrais pirminio maitinimo šaltinio įvade. Pramonės išleidžiamių filtrų slopinamų dažnių juostos plotis yra nuo 0,15 iki 300 MHz. Vienas iš filtro principinės schemos variantų pateiktas 72 pav.



72 pav. Tinklo filtras

Filtre rekomenduojama naudoti aukštadažnius kondensatorius, induktyvines rites be šerdies arba su aukštadažne šerdimi.

Labai svarbu ir tinkamai prisijungti prie elektros tinklo. Atskirais atvejais patartina tinklo kabeliui naudoti ekraną (įprastą plieninį vamzdį, sujungtą su įžeminimu arba ekranuotą kabelį). Prie tam tikros laido ir galingų elektros bei radijo trikdžių šaltinių tarpusavio orientacijos laido atkarpoje gali būti indukuoti gana aukštos įtampos amplitudės signalai. Laidas plieniniame vamzdyje yra ekranuotas, todėl indukuota įtampa bus nežymi.

73 pav. vaizduoja tris galimus skaitmeninio įtaiso (SĮ) prijungimo prie elektros tinklo atvejus: a) blogą, b) geresnį ir c) gerą.

Jeigu jėgos transformatoriaus pirminė ir antrinė apvija suvyniotos ant tos pačios ritės, tai dėl talpinio ryšio tarp abiejų apvijų impulsiniai trikdžiai gali pereiti iš pirminio tinklo į antrinį. Yra keturi tokių trikdžių slopinimo būdai:

a) jėgos transformatoriaus pirminė ir antrinė apvija vyniojama ant skirtingų ričių, - tarpapvijinis talpis sumažėja, tačiau taip pat sumažėja transformatoriaus naudingo veiksmo koeficientas (dalis magnetinio srauto patiria trumpąjį jungimą per išorę);

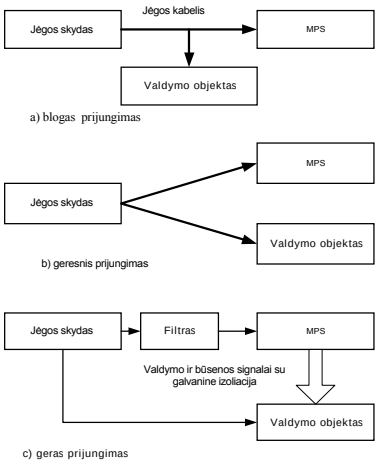
b) pirminė ir antrinė apvija vyniojama ant tos pačios ritės, tačiau atskiriamos ekranu (neuždaru!) iš varinės folijos (storis ≥ 0,2 mm). Ekranas jungiamas su MPS korpusine žeme;

c) pirminė apvija visai ekranuojama, o ekranas įžeminamas;

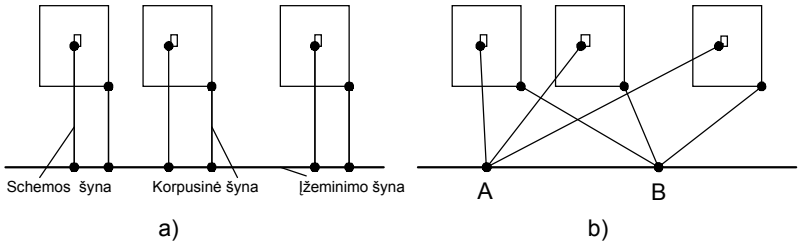
d) pirminė ir antrinė apvija įdedama į individualius ekranus, tarp jų dedamas skiriamasis ekranas, o visas transformatorius dedamas į metalinį korpusą.

Ekranai ir korpusas įžeminami. Be to, tinklo laidus įtaiso viduje reikia pravesti ekranuotu laidu, ekraną įžeminti. Negalima arti vienas kito kloti tinklo ir signalinių laidų (netgi ekranuotų).

Konstruktyviai išbaigtuose blokuose yra dviejų tipų “žemės” - korpusinė ir signalinė. Pirmoji - saugumo technikos reikalavimu turi būti prijungta prie įžeminimo šynos. Antroji (jos atžvilgiu atskaitoma įtampos signalų lygiai) neturi būti sujungta su korpusine žeme MPS viduje - jai turi būti išvestas atskiras gnybtas, izoliuotas nuo korpuso. Schemos žemės sujungiamos individualiais laidais A taške, korpusinės – B taške, pagal galimybes priartintame prie A taško (74 pav.)



73 pav. Skaitmeninio įtaiso jungimo prie elektros tinklo būdai



74 pav. Atskirų MPS prijungimo prie "žemės" schemos

Geriausio įžeminimo varianto parinkimas priklauso nuo konkrečių "vietinių" sąlygų, ir atskirais atvejais tai įvyksta po serijos bandymų. 74a pav. schemoje pateiktas blogas MPS žemių sujungimo atvejis, 74b pav. schemoje - geras.

IG persijungiant iš vienos būsenos į kitą vyksta staigūs, trumpalaikiai, iš antrinio maitinimo šaltinio pareikalaujami srovės šuoliai. Užsikrauna parazitiniai kondensatoriai, kurie išsikraudami sukelia trumpalaikes impulsines sroves žemės šyna. Maitinimo ir žemės šynos turi tam tikrus induktyvumus. Šios srovės dėl tarpusavio sąveikos indukuoja tarp maitinimo ir žemės šynų

įvairaus poliarumo įtampas. Jeigu žemės ir maitinimo šynos padarytos labai plonos ir nėra prijungtų aukštadažnių kondensatorių, tai, persijungiant vienu metu keletui skaitmeninių schemų, tolimajame plokštės krašte gali indukuotis įtampos, siekiančios keletą voltų. Todėl, projektuojant plokštes, reikia laikytis šių taisyklių:

a) maitinimo ir "žemės" šynos turi būti mažiausio induktyvumo, būtinai gardelinio tipo, pagal galimybę uždengiant visą laisvą plokštės plotą (ground plane);

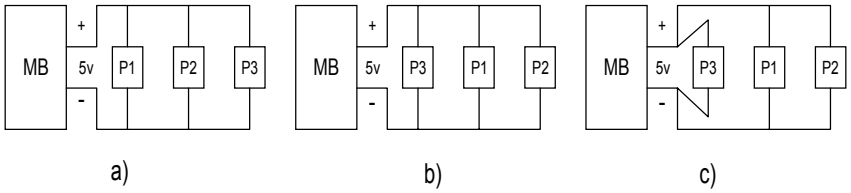
b) maitinimas ir "žemė" prie plokštės privedami per kelis jungties kontaktus, tolygiai išdėstytus per visą jungtį;

c) kiekvienam IG arti maitinimo išvadų reikia jungti aukšto dažnio kondensatorius (0,02 - 1 mkF); spausdintinės plokštės kampuose prie pat jungties jungiami elektrolitiniai kondensatoriai (50 - 100 mkF) žemadažnei filtracijai;

d) mišriose analoginėse-skaitmeninėse schemose IG maitinimo įtampas patartina prijungti per droselius (bead);

e) greitaeigių schemų išėjimus prie MDP schemų įėjimų (ypač DRAM, kurių įėjimo talpis siekia 10 pF) reikia jungti per nuoseklius rezistorius (33-47 omų). Tai sumažina pereinamojo proceso trukmę ir maitinimo srovės šuolį įkraunant įėjimo talpį;

f) MPS plokštes, naudojančias skirtingas sroves, tikslinga jungti prie maitinimo bloko (MB) atskirais laidais. 75 pav. pavaizduotos plokštės P1, P2, P3 naudoja iš MB keletą kartų besiskiriančias sroves didėjimo kryptimi. 75a pav. schema rodo blogą, 75b - geresnį, 75 c -gerą jungimo atvejį.



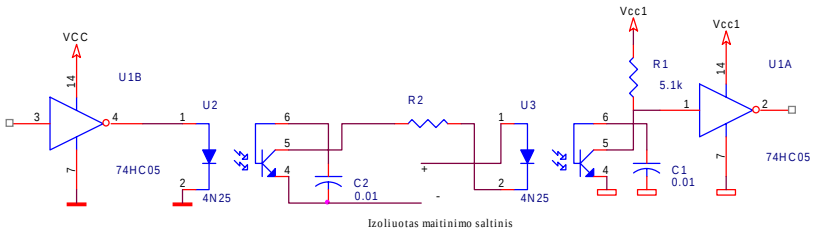
75 pav. MPS plokščių prijungimo prie MB atvejai

Projektuojant greitaeigiškas sistemas su skaitmeniniais ir analoginiais komponentais, reikia atskirti analogines ir skaitmenines schemos dalis spausdintinio montažo plokštėse, kad sumažinti jų tarpusavio sąveiką. Reikėtų vengti lygiagrečiai talpinti analoginių ir skaitmeninių signalų laidininkus. Pagal galimybę jie turi kirstis statmenai .

Analoginiams įėjimams ir išėjimams patartina naudoti koaksialinius kabelius, šarvą įžeminant tik viename gale. Jeigu projektuojama analoginių duomenų surinkimo sistema, analoginės fiksacijos schemą ir analoginį skaitmeninį keitiklį (ASK) reikėtų talpinti vienoje plokštėje, galimai arčiau vienas kito. Visiems įžeminimams reikėtų naudoti mažo impedanso plokštumas (ground plane) dvipusėje spausdintinio montažo plokštėje.

IG įėjimų ir išėjimų apsaugai nuo viršįtampių, kurie gali atsirasti dėl tarpusavio indukcijos, reikia naudoti specialius nuo jų saugančius greitaeigius ir didžiasrovius diodus, apsaugai nuo statinio krūvio – specialius apsaugos IG (ESD protection circuitry) arba varistorius [26].

Jeigu sistemos įtaisai išdėstyti palyginti dideliais atstumais vienas nuo kito (per dešimtis ar šimtus metrų), tai sunku tikėtis, kad jų "žemės" visada turės tą patį potencialą. Taigi išlyginančios srovės žemės laidininkuose sukels impulsinius trukdžius (dėl jų baigtinės varžos). Pastarieji gali sukelti klaidingą schemos reakciją. Išeitis – galvaninis atskyrimas, panaudojus optronines poras (žr. 76 pav.). Galimos dvi atskyrimo schemos – su aktyviu siųstuvu ir su aktyviu imtuvu.



76 pav. Galvaninis atskyrimas su aktyviu siųstuvu

Schema su aktyviu siųstuvu turi siunčiantį U2 ir priimantį optroną U3. Pasiuntus impulsinius signalus, šviesos diodas U2 periodiškai spinduliuoja šviesą. Fototranzistorius atsidaro ir užsidaro. Srovė teka per tranzistoriaus sandūrą emiteris-kolektorius, per šviesos diodą U3 į izoliuotą maitinimo šaltinį. U3 fototranzistorius taip pat užsidaro ir atsidaro.

Optinis atskyrimas žymiai pagerina ryšio kanalų atsparumą trikdžiams. Rezistorius R2 nustato srovę linijoje ir apriboja srovę imtuvo šviesos diodui.

Toks galvaninio atskyrimo būdas dar vadinamas srovės kilpa. Jai rekomenduojama parinkti 20 arba 40 mA srovės reikšmes loginiam vienetui perduoti. Parenkant R2 nominalą, reikia įvertinti ir linijos laidininkų varžą.

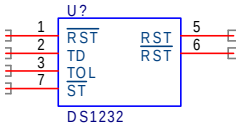
Naudojant srovės kilpos sąsają, svarbu tinkamai pasirinkti optroninių porų IG. Reikalingas žadinimo srovės dydis nėra pakankama sąlyga. Ne mažiau svarbios optoporų dinaminės charakteristikos, užtikrinančios schemos greitaveiką. Optoporas galima pasirinkti iš [24].

6.2. MPS darbo apsaugos ir kontrolės priemonės

Labai patogi MP ir MV darbo kontrolės priemonė yra jų darbo monitoriai (watchdog). Jie atlieka tokius veiksmus:

- stabdo MP arba MV darbą, kai maitinimo įtampa krinta žemiau leistinos ribos;
- automatiškai paleidžia jį, maitinimo įtampai pasiekus nominalią reikšmę iki nustatytos tolerancijos ribų;
- seka MP arba MV darbo eigą;
- formuoja MP arba MV paleidimo signalą.

Tokio monitoriaus pavyzdys gali būti IG DS1232 [25] (77 pav.).



77 pav. DS1232 žymėjimas principinėje schemoje

IG DS1232 atlieka maitinimo, programos darbo kontrolės ir išorinio paleidimo funkcijas. Vidiniu komparatoriumi kontroliuojama MPS maitinimo įtampa (5%, kai išvadas TOL jungiamas su GND arba 10%, kai TOL jungiamas su +5V išvadu). Mažėjant maitinimo įtampai, suformuojamas MPS stabdymo signalas RST arba jo inversija RST/ . Kai maitinimo įtampa pasiekia nominalią reikšmę minus nustatyta tolerancija, RST signalas užlaikomas aktyvaus lygio dar mažiausiai 250 ms, kad baigtųsi pereinamieji procesai maitinimo grandyse, ir MP paleidžiamas iš naujo. Kita jo funkcija - išorinio paleidimo valdymas. Jis atkartoja mygtuko, prijungto prie pirmo IG įvado, nuspaudimo būseną ir garantuoja RST aktyvų laiko intervalą, ne mažesnį už 250 ms. Trečia paskirtis - apsaugos taimeris. DS1232 turi vidinį taimerį, kuris verčia RST ir RST/ signalus į aktyvią būseną, jei strobo įėjimas ST/ tam tikrą laiką nevaldomas ir yra žemo lygio. Apsaugos taimerio relaksacijos laikas gali būti nustatomas iš išorės. Užlaikymo reikšmės apytiksliai gali būti 250ms, 600ms, 1200 ms, priklausomai nuo to, kur prijungtas įvadas TD (kai prijungiama prie GND – 250ms, kai paliekamas atviras – 600ms, kai prijungiamas prie +5V – 1200 ms).

Naudojant šį IG, įvadas ST/ prijungiamas prie tam tikro MP arba MV prievado išvado, kuriame periodiškai vartotojo programa generuoja impulsinį signalą. Jo pasikartojimo periodas pasirenkamas pagal reakcijos į MPS "pakibimą" trukmę. Jeigu į MPS įeina realaus laiko laikrodis, generuojantis pertrauktį kas 1 sek, tai patogu pertraukties aptarnavimo paprogramėje įrašyti dvi komandas, kurios formuos MV 8051 atveju išėjime P1.0 impulsinį signalą:

```
SET P1.0   ; nustatyti vienetą išėjime P1.0
CLR P1.0   ; nustatyti nulį išėjime P1.0
```

Šio impulsinio signalo trukmė priklausys nuo MV sinchronizacijos dažnio, o periodas sutaps su pertraukties periodu. Tada IG DS1232 nuolatos kontroliuos pertraukties aptarnavimo procesą, ir, jeigu dėl kokių nors priežasčių (trumpalaikis trikdis per maitinimo grandines, gedimas RAM bloke ir pan.) jis nebus aptarnautas ilgiau nei per 1200 ms, ji į MV generuos pradinio nustatymo signalą RST, ir MPS bus paleista iš naujo.

Labai svarbus ir programinės įrangos autorinių teisių apsaugos aspektas. Patartina naudoti mišrias aparatines-programines apsaugos nuo kopijavimo priemones. Čia gerai tinka specialūs IG su unikaliu serijos numeriu (TOUCH memory [25]), MP ir MV su programų atminties apsaugos nuo nesankcionuoto skaitymo priemonėmis. MPS naudojamas logines schemas patartina talpinti į programuojamas logines matricas, pvz., CPLD (complex programmable logic device [25]), kurios turi apsaugos nuo nesankcionuoto skaitymo saugiklį, įstatomą programuojant.

7. Duomenų perdavimas ryšio linijomis

Norint duomenis patikimai perduoti laidinio ryšio linijomis, reikia gerai žinoti jų charakteristikas ir parinkti tinkamą sąsajos rūšį bei mainų protokolą.

Pirmiausia susipažinsime su ryšio linijų tipais bei jų parametrais, išsiaiškinsime suderinimo būdus, aptarsime pagrindinius sąsajų standartus.

Duomenų perdavimas - tai maksimalaus greičio tiesioginiai mainai tarp dviejų geometriškai atskirtų loginių ventilių. Standartinės TTL ir KMDP schemos veikia dažnių diapazone iki 40 MHz. Turi būti imtasi specialių priemonių atspindžiams ir tarpusavio įtakai minimizuoti, taip pat trikdžių įtakai mažinti. Dėl šių priežasčių ryšio atstumai nėra labai nedideli. Lemiamas veiksnys yra trikdžiai, atsirandantys dėl ryšio linijų tarpusavio sąveikos ir "žemės" potencialų skirtumo ryšio linijos galuose.

Specializuoti komunikacijų IG veikia su padidintais signalų lygiais, todėl pavyksta pagerinti santykį signalas - trikdis. Naudojamos nesimetrinės ir diferencialinės (simetrinės) jungimo schemos su įtampos ir srovės šaltinio

darbo režimais. Nesimetrinese schemose jungimui dažniausiai naudojami koaksialiniai kabeliai , diferencialinėse - vyniotos poros.

Nesimetriniams (single ended) duomenų mainams būdingi tokie privalumai:

- paprasta schema, nes čia yra mažiausiai sujungimų;
- maža schemos savikaina.

Trūkumai:

- menkas atsparumas trikdžiams;
- intensyvus aukštadažnis spinduliavimas;
- koaksialinis kabelis jį sumažina , bet yra brangus;
- ribotas sujungimo atstumas (transmission length) ir perdavimo greitis (data rate) dėl signalų tarpusavio interferencijos.

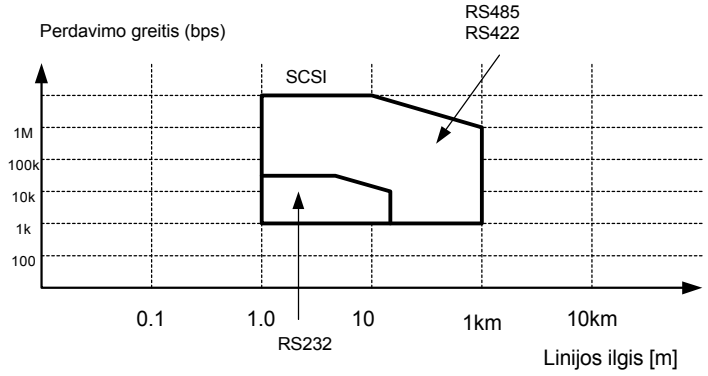
Diferencialiniai (differential) duomenų mainai pasižymi didesniu atsparumu trikdžiams. Linijose atsirandantys sinfaziniai trikdžiai slopinami diferencialinio imtuvo grandyse. Galima pamineti tokius privalumus:

- didelis atsparumas trikdžiams;
- mažas aukštadažnis spinduliavimas;
- didesnis perdavimo atstumas ir mainų greitis.

Trūkumai būtų tokie:

- padidėjusi linijos valdymo elementų (drivers) kaina;
- brangesnis ryšio kabelis.

78 pav. ir 25 lentelėje pateikiamos simetrinių ir nesimetrinių sąsajos standartų charakteristikos.



78 pav. Sąsajų charakteristikos

Duomenų mainų standartai užtikrina patikimą ryšį tarp įvairių firmų gaminamos produkcijos. Jie taikomi nesimetrinėms ir diferencialinėms sąsajos

formoms. Kai kurios standartų techninės charakteristikos pateikiamos 25 lentelėje.

Sąsajos standartai 25 lentelė

Parametras	EIA-232	RS-423-A	RS-422-A	RS-485	TIA/EIA-644 LVDS – low-voltage differential signaling
Darbo režimas	nesimetrinis	nesimetrinis	diferencialinis	diferencialinis	žemos įtampos diferencialinis
Siųstuvų ir imtuvų skaičius	1 siųstuvas ir 1 imtuvas	1 siųstuvas ir 10 imtuvų	1 siųstuvas ir 10 imtuvų	32 siųstuvai ir imtuvai	1 siųstuvas ir 10 imtuvų
Linijos ilgis	20m	1200m	1200m	1200m	
Mainų greitis (bps)	20K	100K	10M	10M	655M
Įtampų lygiai (V)	+-3	+-7	+-7	-7 iki 12	
Siųstuvo apkrova (Om)	3kOm iki 7kOm	min 450	min 100	min 60	100 Om
Siųstuvo trumpo jungimo srovės riba (mA)	500 į Um	150 į GND	150 į GND	250 į -7V arba 12V	
Imtuvo įėjimo varža (kOm)	Nuo 3 iki 7	4	4	12	
Imtuvo jautrumas	+-3V	+-200mV	+-200mV	+-200mV	

Standartas EIA-232 arba RS232 buvo rekomenduotas ANSI (American National Standard Institution) nesimetriniams duomenų mainams tarp kompiuterių ir jų periferinių įrenginių. Paskutiniai jo pakeitimai atlikti 1991m. liepos mėn. Standartas dar vadinamas EIA/TIA-232-E ir pripažįstamas EIA (Electronic Industries Association) ir TIA (Telecommunications Industry Association). Ankstesnės standarto versijos rekomendavo 20 kilobitų per sekundę (kbs) mainų greitį. Kadangi jis vėliau pasiekė 200 kbs, atsirado

modifikacijos “C,D ir E”. Nusakytas maksimalus apkrovos talpis - 2500 pF, kai linijos ilgis siekia 20m.

Sąsajos RS422 standartas sukurtas 1975 metais, jo modifikacija RS-422-A - 1978 m. Ši sąsaja taikytina simpleksiniam ryšiui. Čia vienas siųstuvas dirba su 10-ia imtuvų. Duomenų perdavimo greitis siekia 10Mbps, kai linijos ilgis iki 1200m.

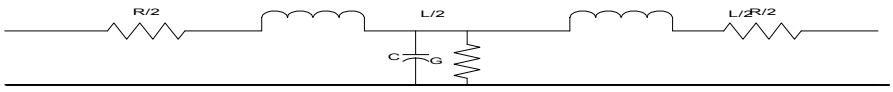
RS485 yra tolesnis RS422 sąsajos patobulinimas, skirtas pusiau dulpeksiniams duomenų mainams ir daugiataškėms komunikacijoms. Standartas apibrėžia tik sąsajos fizinį lygmenį. Standarte apibrėžiamas linijos ilgis, naudojant kabelį 24AGW, kurio slopinimas yra 6 dB. Perdavimo greitis nelimituojamas.

SCSI (Small Computer Systems Interface) yra ANSI rekomenduota pramoninio standarto sąsaja, naudotina mainams tarp kompiuterio ir periferijos. Tai lygiagreti baitininė sąsaja, skirta greitiems duomenų mainams palygint nedideliais atstumais. SCSI magistralė yra dvikryptė ir suderinta abiejuose galuose. Sąsaja veikia 6 metrų atstumu. Maksimalus mainų greitis neaptariamas, bet gali siekti 10 Mbps (million transfers per second) arba 80 Mbps. Didesniems atstumams (iki 25m) SCSI kaip fizinį lygį siūlo naudoti RS485 standartą. 16-os bitų SCSI gali pasiekti iki 160Mbps greitį.

Toliau aptarsime duomenų perdavimo linijų charakteristikas bei jų suderinimo sąlygas ir metodus.

Ekvivalentinė linijos schema (79 pav.) susideda iš paskirstyto induktyvumo L, paskirstytos varžos R, rodančios linijos ominę varžą, talpumo C ir laidumo G, rodančio linijos nuostolius. Šių elementų reikšmės priklauso nuo ilgio ir todėl žymimos atitinkamo elemento nominalo verte ilgio vienetui, t.y.: nH/cm, pF/cm, Ω/cm ir S/cm. Perdavimo linijos impedansas apskaičiuojamas pagal šią formulę:

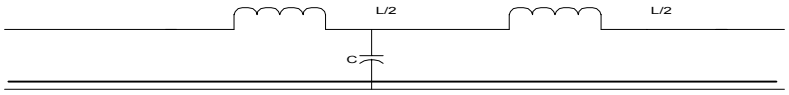
$$z_0 = \sqrt{\frac{j \cdot \omega \cdot L' + R'}{j \cdot \omega \cdot C' + G'}}.$$



79 pav. Ekvivalentinė ryšio linijos schema

Matome, kad linijos kompleksinė varža priklauso nuo dažnio. Tai labai nepatogu skaitmeninėse schemose, kur vienu metu gali veikti kelių dažnių skaitmeniniai srautai.

Skaitmeninėse schemose žemas dažnis nekelia didelių sunkumų. Aukštesniuose dažniuose (didesniuose kaip 10 kHz) ritės impedansas jϖL palyginti su laidų varža R sparčiai didėja. Didėja ir linijos admitansas jϖC. Jis taip pat sparčiau didėja palyginti su laidumu G. Todėl aukštesniuose dažniuose R ir G galima nepaisyti. Tuomet ekvivalentinė schema supaprastėja (80 pav.):



80 pav. Ekvivalentinė schema be nuostolių

Tada perdavimo linijos impedansas apskaičiuojamas pagal formulę:

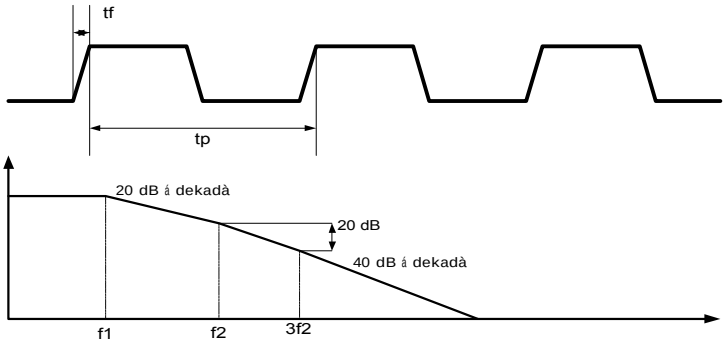
$$Z_0 = \sqrt{\frac{L}{C}}.$$

Impedansas čia yra realus skaičius ir nepriklauso nuo dažnio.

Dar vienas svarbus duomenų perdavimo parametras yra signalo sklidimo laikas

$$t_p = \sqrt{L \cdot C}.$$

Tipinė šio laiko reikšmė dažniausiai naudojamuose kabeliuose yra apie 5ns/m. Taigi signalo sklidimo greitis v=200 000 km/s sudaro apie 60% šviesos greičio. 81 pav. pateiktas skaitmeninio signalo, kurio pasikartojimo periodas t_p ir frontų trukmė t_f, spektras (dažnis $f_1 = 1/t_p$). Virš jo aukštesnių harmonikų amplitudės mažėja 20 dB į dekadą greičiu. Dažnio f_2 reikšmė priklauso nuo fronto trukmės t_f.

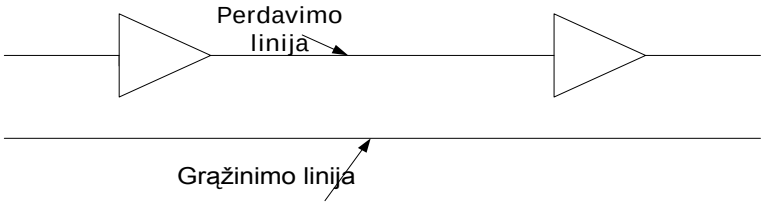


81 pav. Skaitmeninio signalo spektras

$$f_2 = \frac{1}{\pi \cdot t_f}.$$

Jeigu duomenų perdavimo greitis 1 Mbit/s, o fronto trukmė t_r = 5 ns, tada f_1 = 500 kHz, f_2 = 64 MHz. Jei perdavimo sistemoje aktualūs interferencijos reiškiniai, projektuojant mažinama signalo galia, mažinamas duomenų perdavimo laikas arba naudojamos schemos, formuojančios ilgus frontus.

Perdavimo sistema visada turi dvi linijas (žr. 82 pav.): signalo padavimo liniją ir grįžimo liniją. Abi šios linijos veikia greta esančius įrenginius. Kuo didesnis plotas tarp šių linijų tuo didesnė tikimybė, kad elektromagnetinė energija plis į didesnį tūrį ir veiks greta esančius įrenginius. Be to, signalas gali turėti keleta grįžimo kelių (pvz., per apsauginį žemės laidą).



82 pav. Signalo perdavimo ir grąžinimo linijos

Perdavimo linijomis laikomos tos linijos, kuriose dvigubas signalo sklidimo laikas pasidaro ilgesnis už signalo frontą, arba kai atspindžiai linijoje nedaro įtakos signalo fronto trukmės intervale. Kai signalo fronto trukmė t_f = 5 ns , o signalo sklidimo laikas t_p = 5 ns/m, tada kritinis linijos ilgis yra 1 m. Praktiškai, kai signalo sklidimo laikas yra daug ilgesnis, kritinis linijos ilgis yra daug trumpesnis. Nuo perdavimo linijos ilgio priklauso signalo vėlinimas: kuo linija ilgesnė, tuo didesnis jos talpis, todėl ir signalo frontas linijos gale pailgėja.

Linijoje atsirandantys atspindžiai (pradžioje, gale ir nuo vidinių netolygumų) gali pakeisti duomenų formą ir šie gali būti neteisingai imtuvo priimti. 83 pav. schemoje matome nesudėtingą būdą signalo formai perdavimo linijose analizuoti. Čia įtampa iš generatoriaus, kurio išėjimo varža Rg, signalo amplitudė V0, jungikliu J paduodama į liniją (jos ilgis L), kurios banginė varža Z0, o signalo plitimo laikas tp. Linijos gale yra apkrova, kurios varža Ra. Signalo amplitudė linijos pradžioje bus

$$V = V_0 * Z_0 / R_g + Z_0 .$$

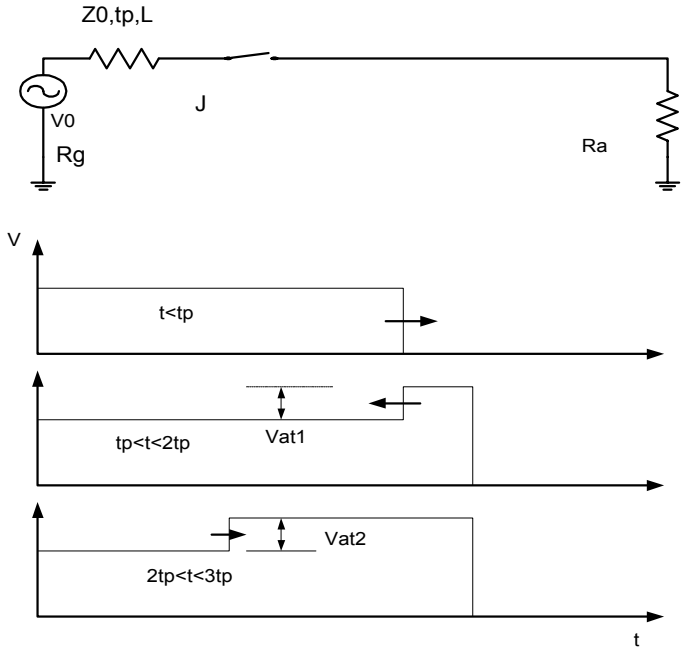
Kai ši signalo amplitudė pasiekia linijos galą, kuriame nėra suderinimo, dalis signalo grįžta atgal. Atspindėto signalo amplitudė apskaičiuojama pagal formulę

$$V_{at1} = V * \frac{R_a - Z_0}{R_a + Z_0} .$$

Kai atsispindėjusi banga grįžta, ji vėl atspindima, nes generatoriaus varža nesuderinta su linijos. Šis procesas kartojasi, kol signalas sugeriamas dėl linijoje esančių nuostolių. Praktiškai įvairių atspindėtų bangų apskaičiavimą galima nutraukti po trečio atspindžio, nes toliau atspindėtos bangos amplitudės labai mažėja ir jų galima nepaisyti.

Praėjus ilgesniam laiko tarpui įtampa linijoje nusistovi ir gali būti apskaičiuojama pagal šią formulę:

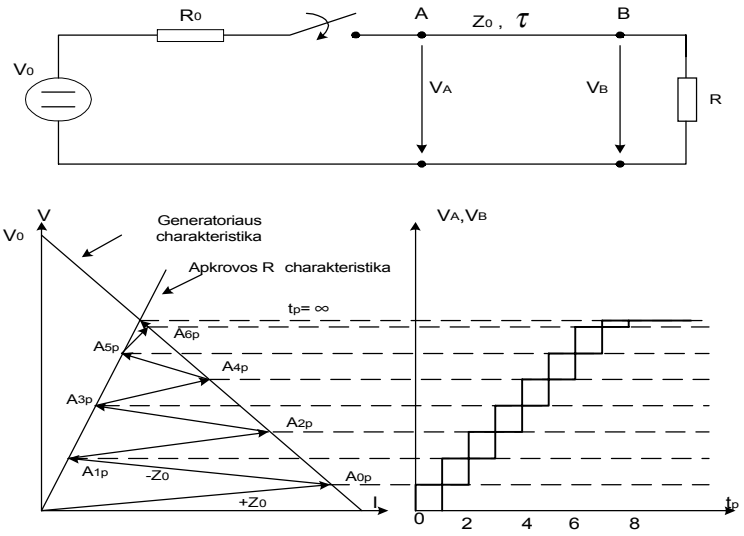
$$V_{t=\infty} = V_0 \cdot \frac{R_a}{R_0 + R_a} .$$



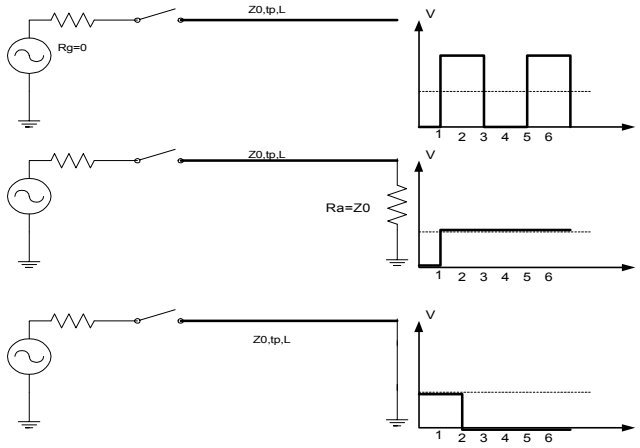
83 pav. Signalo plitimas linijos atkarpoje

Atspindžių susidarymo duomenų perdavimo linijose procesams nagrinėti tinka Bergerono metodas (žr. 84 pav.)[22]. Analizei braižomos generatoriaus ir apkrovos voltamperinės charakteristikos. Jų susikirtimo taškas parodo įtampą nusistovėjusiu režimu. Signalo formos konstravimas prasideda nuo to momento, kai įjungiamas jungiklis V_t =0 ir I_t = 0. Iš šio taško brėžiame nuožulnią tiesę. Kai ši tiesė pasiekia generatoriaus charakteristiką, gauname signalo įtampos reikšmę linijos pradžioje. Toliau brėžiame neigiamą tiesę apkrovos rezistoriaus charakteristikos link ir gauname įtampos reikšmę linijos gale konkrečiu laiko momentu. Toliau brėžiame analogiškas tieses, kol pasiekiame charakteristikų susikirtimo tašką. Pažymėję visus gautus taškus, gauname įėjimo ir išėjimo signalų reikšmes.

Aptarkime kelis ypatingus linijos apkrovimo atvejus (85 pav.).

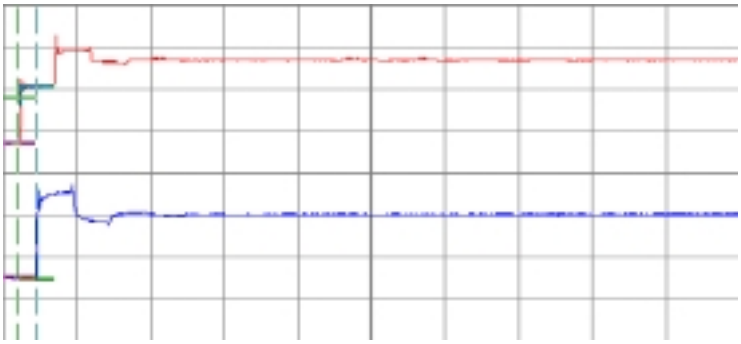


84 pav. Bergerono diagrama atspindžių analizei



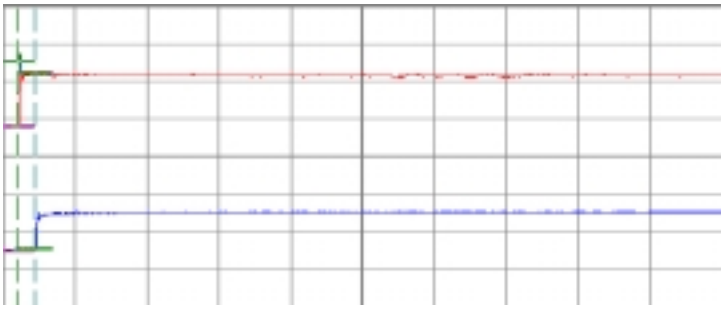
85 pav. Atviro, suderinto ir trumpo jungimo linijos gale atvejai

Pirmoje scemoje pavaizduota linija su atviru galu. Jos pradžioje prijungtas generatorius, kurio išėjimo varža R_0= 0 Ω. Sakykim, linija yra be nuostolių. Linijos atspindžio koeficientas yra -1 generatoriaus įėjime ir +1 linijos gale. Linijos gale stebime bėgančią bangą. Kitoje schemoje linija apkrauta suderinta apkrova. Čia yra stovinčios bangos režimas.



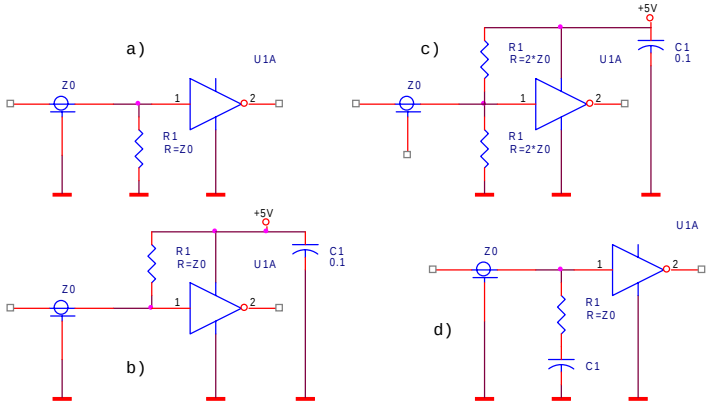
86 pav. Signalo forma linijos pradžioje ir gale, kai jos galas atviras

Paskutiniame pavyzdyje linija yra trumpai sujungta. Generatoriaus varža lygi linijos varžai. Kai įjungiamas jungiklis, signalas, kurio amplitudė $0,5V_0$, keliauja į linijos galą ir atsispindi su neigiama amplitude. Kai atspindėtas signalas pasiekia generatoriaus įėjimą, nusistovi trumpo jungimo įtampa. Ši schema gali būti naudojama kaip impulsų formuotuvas, kur impulso trukmė lygi dvigubam signalo plitimo laikui. Praktiškai sunku rasti šaltinį su nuline išėjimo varža. 87 pav. pavaizduotas atvejis, kai impulsų generatorius su 50 omų išėjimo varža yra apkrautas tokios pat banginės varžos kabelio atkarpa su atviru galu. Šiame brėžinyje galime stebėti plitimo laiką (pažymėta laiko žymekliais – 112 ns). Po dvigubo plitimo laiko linijos pradžioje pasirodo atspindys nuo jos galo ir sumuojasi su generatoriaus išėjimo signalu. Toliau formuojasi kitas laiptelis, nes vyksta antrinis astspindys nuo generatoriaus išėjimo. Jo amplitudė dėl linijoje esančių nuostolių sumažėja. Linijos gale signalo frontas pasirodo vėliau, praėjus plitimo laikui. Tačiau taip pat signalo amplitudė nusistovi po trijų laiptelių, bet signalo formos linijos pradžioje ir gale skiriasi.



87 pav. Signalo forma linijos pradžioje ir gale, kai jos galas suderintas

Kai linijos galas yra suderintas, signalų forma linijos pradžioje ir gale sutampa, atsiranda tik plitimo vėlinimas (žr. 87 pav.). Suderinimas paprastai atliekamas linijos gale jungiant rezistorių, kurio varža lygi linijos banginei varžai (88a pav.). Jeigu linijos pradžioje yra schema su atviru kolektoriumi, tai galima naudoti 88b pav. schemą, jeigu schema su aukšto impedanso būsena - 88c pav. Jeigu dirbama su KMDP schemomis, tikslinga sumažinti nuolatinę srovę, nuosekliai suderinimo varžai jungiant kondensatorių (88d pav.).



88 pav. Linijos galo suderinimo schemos

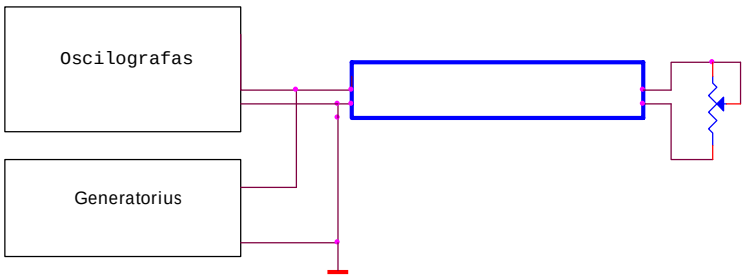
Šios grandinės laiko pastovioji turi būti apie keturis kartus didesnė už signalo plitimo laiką, tada signalo iškraipymai yra minimalūs.

Gali pasitaikyti atvejų, kai signalo šaltinio varža mažesnė už linijos banginę varžą. Tada nuosekliai jungiamas rezistorius, kurio nominalas lygus skirtumui tarp linijos banginės varžos ir signalo šaltinio išėjimo varžos.

Dažnai praktiškai tenka matuoti perdavimo linijos ar koksialinio kabelio banginę varžą. Tai atliekama pasitelkus impulsų generatorių, oscilografą ir potenciometrą (žr. 89 pav.).

Čia potenciometro rankenėlė sukama tol, kol oscilografo ekrane bus neiškraipytas signalo fronto vaizdas.

Projektuojant ryšio linijas (taip pat ir spausdinto montažo plokštes), reikia atsižvelgti ir į gretimų linijų signalų tarpusavio įtaką. Linijos veikia viena kitą dėl tarp jų egzistuojančio parazitinio impedanso Zc. Kai dvigubas signalo plitimo laikas didesnis už fronto trukmę, tarpusavio įtaka (crosstalk) procentais gali būti įvertinta pagal šią formulę:



89 pav. Linijos banginės varžos matavimo schema

$$C = 100/1 + 2 * Z_c / Z_0 .$$

Toliau pateiksime keletą spausdinto montažo plokštės takelių išdėstymo variantų (91 pav.). Pirmajame pavyzdyje linijos išdėstytos lygiagrečiai viena kitai. Kai atstumas tarp jų 0,5 mm, ryšio impedansas Zc= 100 Ω. Kai žemės takelis pravestas šalia linijų ir atstumas iki jų apie 2 cm, tuomet Z_0=200 Ω, o tarpusavio ryšys 50 %. Kitame pavyzdyje žemė yra spausdintos plokštės apačioje. Tuomet linijos banginė varža sumažėja Z_0 =80 Ω, o ryšio varža padidėja Zc = 125 Ω. Tarpusavio ryšys sumažėja (25%). Šis būdas daugiasluoksnėse plokštėse ir sumažina triukšmų lygį. Trečiame pavyzdyje ekranas yra tarp linijų. Ekranas su žeme sujungiamas linijos gale. Tada linijos varža Z_0 = 100 Ω, ryšio varža Zc= 400 Ω. Ryšys tokioje plokštėje yra mažesnis 11 %.

Dėl linijos ominės varžos R ir nuotėkio laidumo G, ilgėjant linijai, signalas silpsta. Įtampa linijos gale V_2 gali būti apskaičiuota pagal šią formulę:

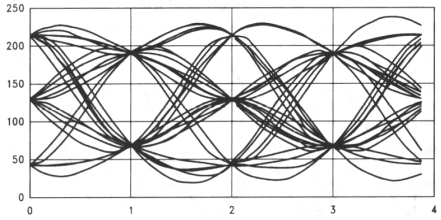
$$V_2 = V_1 \cdot e^{\frac{-R' \cdot l}{2 \cdot Z_0} + \frac{-G' \cdot l \cdot Z_0}{2}};$$

čia V1 – signalo amplitudė linijos įėjime.

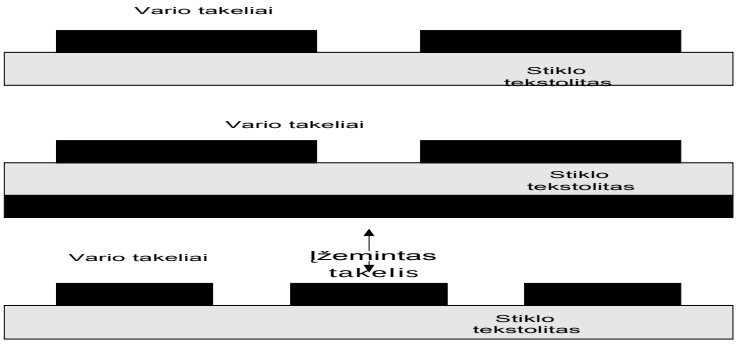
Jei linijos banginė varža Z_0 = 100 Ω, ilgis l = 1 km, ilgio vieneto varža R=0.1 Ω/m, tuomet įtampa linijos gale sumažėja apie 40 %. Ši formulė nepaiso " skin " efekto.

Duomenų perdavimo sistemose susiduriame ne su vienu diskretiniu dažniu, o su dažnių juosta, kuri priklauso nuo siunčiamų bitų sekos savybių. Pereinamosios linijos charakteristikos filtruoja signalą, todėl priėmimui naudojamas slenkstinis įrenginys (kartais diferencialinis ir su histereze).

Duomenų srautų analizei patogus atsitiktinių stačiakampių impulsų sekų generatorius. Jis dažniausiai būna sudarytas iš nuoseklaus postūmio registro, kurio tam tikri išvadai prijungti prie kelių "arba" funkciją atliekančių loginių elementų. Davus tokio generatoriaus signalą į liniją, pastarosios gale galima stebėti "akies" diagramą (eye pattern) (žr. 90 pav.), t.y. stačiakampiai impulsai įgauna akies formos signalo pavidalą. "Akies" aukštis parodo efektinę signalo amplitudę, o jos plotis - siunčiamų duomenų laiko intervalą.



90 pav. "Akies" diagrama



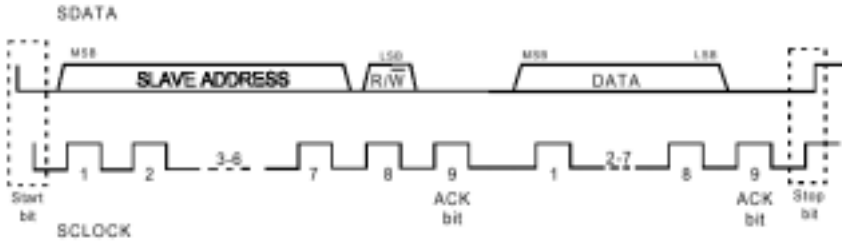
91 pav. Spausdinto montažo plokštės takelių išdėstymo variantai

8. Populiarių sąsajų rūšys

EA tarpusavio mainuose naudojama daugelis sąsajų standartų. Populiarūs RS232, RS422, RS423, RS485 jau buvo trumpai aptarti. Toliau detaliau aptarsime plačiai naudojamas sąsajas I2C ir USB.

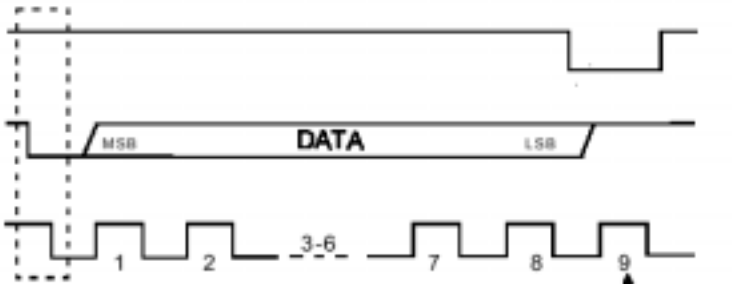
8.1. Sąsaja I2C

Atskirai aptarsime buitinėje EA plačiai paplitusį I2C (Inter Integrated Circuit) sąsajos standartą. Tai dvilaidės (trečia žemė) sąsajos pavyzdys, sukurtas Philips frmoje. Įrenginiai čia sujungiami dviem signalais SCLOCK ir SDATA. Signalas SCLOCK naudojamas perdavimo ir priėmimo sinchronizacijai, SDATA – dvikrypčiams duomenų mainams. Abu signalai yra dvikrypčiai. Šioje sąsajoje skiriami du įrenginiai - vedantysis (master) ir vedamasis (slave). Prie sąsajos linijų gali būti prijungta keletas įrenginių. Mainų darbo laikinė diagrama pateikta 92 pav.



92 pav. Mainų I2C magistralėje darbo laikinė diagrama

Mainai prasideda nuo “Start” bito, kurį generuoja vedantysis įrenginys. “Start” būsena prasideda pagal krintantį signalo SDATA frontą, kai signalas SCLOCK yra loginio vieneto būsenos. Po “Start” būsenos vedantysis įrenginys siunčia per liniją SDATA vedamajam įrenginiui baitą (vyriausiuoju bitu į priekį). Čia yra vedamojo įrenginio adresas ir skaitymo ar rašymo (R/W) būsenos bitas. Pirmi septyni bitai nusako vedamojo įrenginio adresą. Aštuntasis bitas parodo pranešimo perdavimo kryptį: “0” rodo, kad vedamasis įrenginys perduoda vedančiajam, “1” – kad priims iš vedančiojo. Priėmę adresą, visi vedamieji įrenginiai palygina savo adresą su priimtuoju. Jei šie sutampa, vedantysis įrenginys perduoda patvirtinimą (93 pav.).



93 pav. Patvirtinimo laiko diagrama

Patvirtinimas matomas kaip loginis “0” lygis linijos SDATA devintame sinchronizacijos takto periode. Jei patvirtinimas nėra gautas, vedantysis įrenginys generuoja būseną “STOP”, kurioje pagal teigiamą signalo SDATA frontą linijoje SCLOCK yra “1” būsena. 26 lentelėje pateiktas mainų tarp vedančiojo ir vedamojo algoritmas, kai vedantysis siunčia, o vedamasis priima, 27 lentelėje - kai vedamasis siunčia, o vedantysis priima.

Algoritmas, kai vedantysis siunčia, o vedamasis priima 26 lentelė

Vedantysis	Vedamasis
Siunčia “START” bitą	
Siunčia vedamojo adresą	Priima adresą
Laukia patvirtinimo bito	Siunčia patvirtinimą
Siunčia duomenis	Priima duomenis
Laukia patvirtinimo bito	Siunčia patvirtinimą
Siunčia duomenis	Priima duomenis
Laukia patvirtinimo bito	Siunčia patvirtinimą
Siunčia “STOP” bitą	

Algoritmas, kai vedamasis siunčia , o vedantysis priima 27 lentelė

Vedamasis	Vedantysis
	Siunčia “START” bitą
Priima adresą iš vedančiojo	Siunčia vedamojo adresą
Siunčia patvirtinimą	Laukia patvirtinimo
Siunčia duomenis	Priima duomenis
Laukia patvirtinimo	Siunčia patvirtinimą
Siunčia duomenis	Priima duomenis
Laukia patvirtinimo	Siunčia patvirtinimą
	Siunčia “STOP” bitą

Kai kurie MCS 8051 šeimos MV turi aparatines priemones ryšiui per sąsają I2C. Čia galima paminėti firmos Philips DIG 80C552, firmos Analog Devices DIG ADuC812 ir kt. Priede pateiktas asemblerio tekstas MV ADuC812, kai MV veikia vedamojo ir vedančiojo darbo režimais. Tokia sąsaja buitinėje technikoje (televizoriuose, videomagnetofonuose, videokamerose, radijo imtuvuose, PC monitoriuose ir kt.) pasitaiko labai dažnai. Per ją atliekamas ryšys su nuoseklaus tipo EEPROM, realaus laiko laikrodžiais, videokomutatoriais, skaitmeniniais termometrais ir kitais IG.

8.2. Sąsaja USB

USB (universal serial bus) yra šiuo metu sparčiai populiarėjantis sąsajos tipas, kuris ateityje pakeis RS232: personalinių kompiuterių ryšys su klaviatūromis, skaitmeniniais telefonais, manipuliatoriais ir kitais išoriniais įrenginiais. USB topologija yra žvaigždinė (95 pav.), o prie jos galima prijungti 127 loginius įrenginius. Mainų greitis siekia 12Mbps, arba 1.5Mbps izochroniniams ir asinchroniniams (su pertrauktimi) ryšiams. Sujungimui tinka 4 kontaktų jungtys diferencialiniam ryšiui, naudojant NRZ (be grįžimo į nulį) kodavimą [22]. Ryšio atstumas - iki 5 m segmentui. Rekomenduojama imti 28 AWG susuktos poros (galima ir nesusuktos, tik tada ryšio atstumas sutrumpėja iki 3 m segmentui) kabelį, kurio banginė varža 90Ω. 94pav. pateikta USB įrenginių sujungimo schema susuktos bei ekranuotos ir nesusuktos laidų poros atvejais. Pirmu atveju pasiekiamas 12 Mbps mainų greitis, antru – 1,5 Mbps. Ryšio linijos buferinės schemos (drivers) turi turėti atitinkamai 22Ω ir 44Ω išėjimo varžas. Signalų frontų trukmės - 4ns ir 20ns.

USB struktūroje pagrindinė dalis yra valdiklis, kuris dažniausiai yra sudedamoji PC dalis. Kiti įrenginiai jungiami per kartotuvus (HUB). Plačiausiai paplitę keturių kanalų HUB-ai, kurie išleidžiami kaip IG.

USB naudojami tokie komunikacijų lygmenys:

- Kadras. Tai laiko intervalas tarp dviejų siunčiamų paketų pradžios. Jis yra 1 ms trukmės;
- Paketas. Paketą sudaro trys elementai: valdymo informacija (informacijos šaltinis, imtuvas, duomenų ilgis), duomenys, taip pat klaidų detektavimo ir korekcijos informacija. (žr. 96 pav.);
- Protokolas. Tai sugrupuotas duomenų "ryšulys". Jį sudaro požymio, duomenų ir patvirtinimo paketai. USB valdiklis inicijuoja mainus, kurių tipą nusako PID (packet ID) – paketo identifikacijos informacija.

Izochroniniai mainai vyksta periodiškai stabiliu greičiu (realiu laiku). Perduodamos žymės ir duomenys. Tipinis izochroninių mainų pavyzdys yra skaitmeninio telefono ryšys.

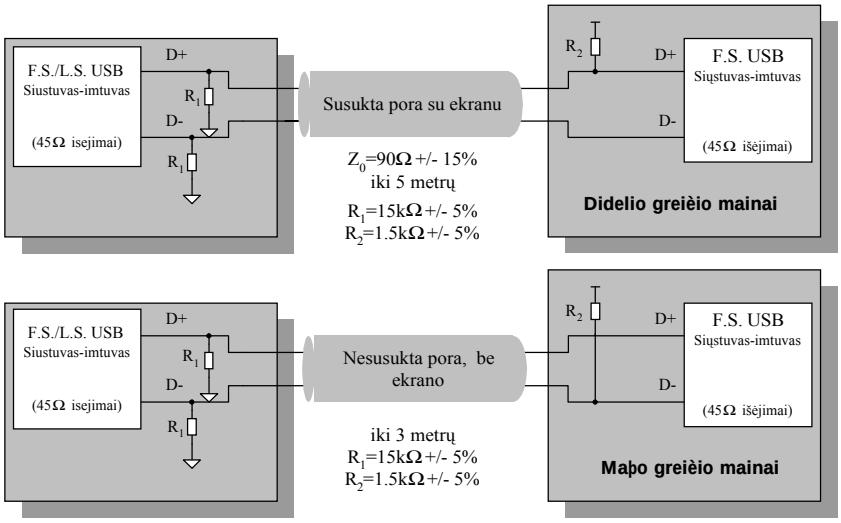
Pagal pertrauktį duomenys gali būti perduodami iš USB įrenginio bet kuriuo laiko momentu. Perdavimo greitis priklauso nuo įrenginio techninių charakteristikų.

Valdymo duomenys USB programinėje įrangoje naudojami norint sukonfigūruoti prietaisus, kai jie pirmą kartą užklausiami. Kontrolinis perdavimas leidžia pasijungti prie skirtingų prietaiso sudedamųjų dalių. Numatyta, kad valdymo duomenys palaiko programinės įrangos, vartotojo ir jo funkcijų konfigūracijos ryšių tipą.

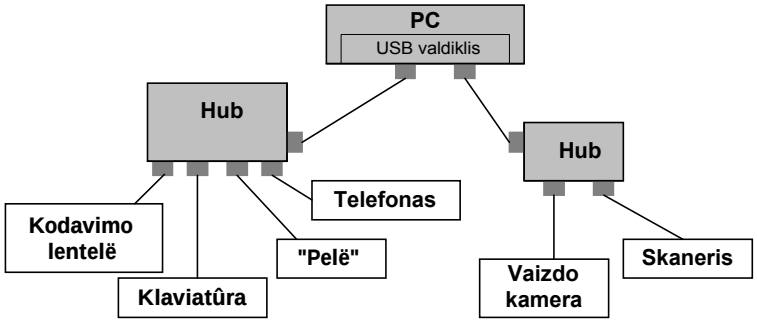
Žymę sudaro PID, kuris nustato, koks bus paketas (įvedimo, išvedimo ar pradinio nustatymo), taip pat ADDR - adreso ir ENDP – paketo pabaigos laukai. Išvedimo ir pradinio nustatymo atvejais adreso ir pabaigos laukai identifikuoja įrenginius, kurie gaus duomenų paketus. Įvedimo atveju jie nustato, kuris iš įrenginių turėtų perduoti duomenų paketą. Tik valdiklis gali suformuoti šiuos paketus. Žymės paketas turi penkis CRC (perteklinė kontrolinė suma) bitus.

Duomenų laukas gali užimti nuo 0 iki 1023 baitų. Duomenys perduodami pradedant jauniausiuoju baitu.

Patvirtinimo lauką sudaro tik PID. Jis naudojamas pranešti apie duomenų perdavimo būseną ir gali sugrąžinti pranešimą apie sėkmingą duomenų, valdymo ir būsenos priėmimą.

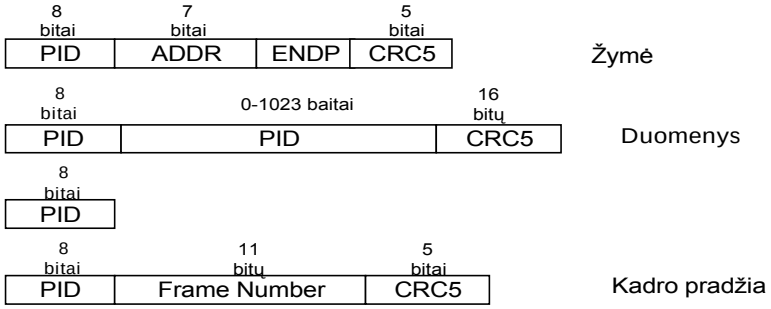


94 pav. USB prijungimas



95 pav. USB žvaigždinė struktūra

Kadrų paketo pradžią formuoja valdiklis. Ji kartojasi kas 1ms ±0,05. Jį sudaro PID (nurodo paketo tipą), 11 bitų kadro numeris ir 5 bitų CRC.



96 pav. Paketų formatai

8.3. Kontroliniai klausimai

1. Kokios pagrindinės priežastys sąlygoja trukdžių, sukeliamų pirminio maitinimo tinkle, atsiradimą? Kokiomis priemonėmis galima šiuos trukdžius slopinti?
2. Kokios pagrindinės priežastys sąlygoja trukdžių, sukeliamų antrinio maitinimo tinkle, atsiradimą? Kokiomis priemonėmis galima šiuos trukdžius slopinti?

3. Kokios yra pagrindinės skaitmeninių įtaisų įžeminimo taisyklės?

4. Kuriais atvejais tikslinga atskirti grandines skaitmeninėje sistemoje naudojant optinį išrišimą? Paaiškinkite "srovės kilpos" sąvoką. Kokio tipo skaitmeninėse sistemose ji naudojama?

5. Kas tai yra ir kokius veiksmus atlieka MP arba MV darbo monitorius (watchdog)?

6. Kaip ir nuo ko priklauso signalų, perduodamų kabeliais, iškraipymai? Kokiais būdais naudojantis galima šių iškraipymų išvengti?

7. Kokie signalai naudojami sąsajoje I2C? Kaip ir kokia informacija jais perduodama?

8. Paaiškinkite mainų per sąsają I2C laiko diagramą.

9. Kokia sąsajos USB struktūra ir kokie mainų tipai?

9. MPS projektavimo pavyzdys

Reikia suprojektuoti apsaugos signalizacijos sistemą [9]. Tarkime, kad būsimasis šios sistemos vartotojas (projekto užsakovas) pateikė tokius savo reikalavimus:

- aptikti, kada atsidaro durys arba langas;
- aptikti, jeigu kas nors juda saugomos teritorijos viduje;
- turėti galimybę perspėti pažeidėją ir iškviesti pagalbą;
- numatyti galimybę sistemai atsistatyti, jeigu operatorius pamiršo ją aktyvinti;
- sistemos valdymas turi būti paprastas;
- klaidingų aliarmų skaičius turi būti minimizuotas.

Sudarysime projektuojamos sistemos funkcinę specifikaciją.

Priminsime, kad funkcinė specifikacija turi nustatyti, kurios funkcijos turi būti įvykdytos, norint patenkinti vartotojo reikalavimus, bei kokiomis priemonėmis užtikrinti sąsają tarp sistemos ir aplinkos.

Dėl patogumo funkcinės specifikacijos informaciją skirstome į tris kategorijas: ĮĖJIMAI, IŠĖJIMAI ir FUNKCIJOS. (Skliaustuose pateikta ĮĖJIMŲ, IŠĖJIMŲ ir funkcijų paskirtis, laikantis vartotojo reikalavimų.)

A. ĮĖJIMAI

1. Kontaktiniai detektoriai (durų arba langų nesankcionuotam atidarymui aptikti).
2. Judesio detektorius (judesiui aptikti).
3. Jungiklis (sistemai valdyti).

B. IŠĖJIMAI

1. Šviesos signalas (operatoriaus perspėjimas, kad reikia nustatyti sistemą).
2. Garso signalas (pažeidėjui perspėti ir pagalbai iškviesti).

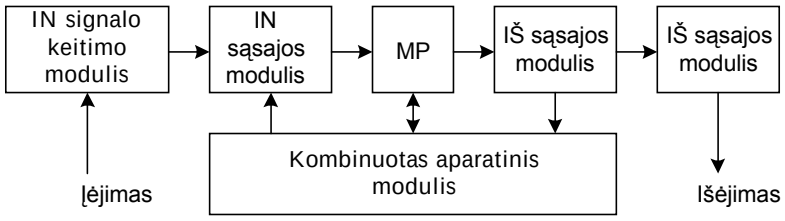
C. FUNKCIJOS

1. Sistema įjungiama ir išjungiama jungikliu.
2. Vizualinis signalas įsijungia:
   2.1. suveikus kontaktiniams detektoriams arba
   2.2. esant sužadintam judesio detektoriui ne mažiau kaip 5 sek (kad sumažėtų klaidingų aliarmo signalų tikimybė).
3. Garso signalas įsijungia po 60 sek, kai buvo įjungtas vizualinis signalas, jeigu per šį laiką sistema nebuvo nustatyta jungikliu.

Pateikti užsakovo reikalavimai ir pagal juos sudaryta sistemos funkcinė specifikacija sudaro pirmą dokumentacijos lygį. Tai yra pagrindas sistemai projektuoti bei modernizuoti.

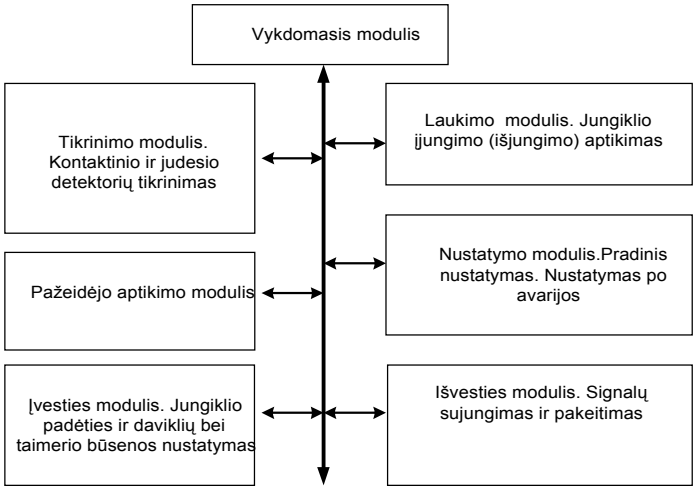
Prieš pradedant detaliai projektuoti sistemos aparatines ir programines priemones, iš pradžių reikia paskirstyti visas sistemos atliekamas funkcijas tarp programinių ir aparatinių priemonių. Tam tikslui sudarome modulinę sistemos aparatinių priemonių struktūrą (97 pav.).

Įvesties (IN) ir išvesties (IŠ) signalų keitimo (transformacijos) moduliai suteikia sistemai galimybę keistis signalais su išorine aplinka. Paprastai yra įvairūs keitikliai analogas - kodas, kodas - analogas, signalų lygių keitikliai ir pan.. Mūsų atveju į IN modulį įeis kontaktiniai davikliai, judesio daviklis bei jungiklis. IN, IŠ sąsajų bei MP modulių paskirtis aiški ir nereikalauja jokių komentarų. Kombinuotas aparatinis modulis realizuoja sistemos funkcijas, kurios gali būti atliekamos ir aparatiniu, ir programiniu būdu. Iš tikrųjų mūsų projektuojamoje sistemoje reikalingą vėlinimo funkciją (5 ir 60 sek laiko intervalams formuoti) galima realizuoti vien tik programiniu būdu arba panaudoti tam tikslui programuojamą taimerį. Nuo to, koks sprendimas bus priimtas, priklausys kaip projektuoti aparatines ir programines laiko intervalų formavimo priemones.



97 pav. Sistemos struktūrinė schema

Tarkime, kad nusprendėme laiko intervalų formavimui naudoti programuojamą taimerį. Tai sumažina MP apkrovimą, o naudojant MV, taimeris įeina į jo sudėtį. Šiuo atveju sistemos algoritmo struktūra pateikta 98 pav. Struktūrą sudaro keturi funkciniai lygiai. Pats aukščiausias funkcinis lygis yra bendriausias, o žemiausias - daugiausia detalizuotas. Aukščiausiame programinių priemonių modulinės struktūros lygyje yra valdanti funkcija, organizuojanti kitų funkcijų vykdymą. Šią valdančią funkciją pavadinsime vykdymo procedūra, turėdami omenyje, kad vykdomasis modulis turi programines priemones, reikalingas vykdymo procedūrai atlikti. Žemesnių lygių moduliai (išskyrus "pažeidėjo aptikimo " modulį) turi daugiau kaip vieną procedūrą. Jos išvardytos kaip viena. Tikslų kiekvieno modulio lygį bus galima nustatyti tik tada, kai bus nustatyti procedūrų tarpusavio ryšiai. Pačių modulių pavadinimai parenkami tokie, kad jų esmė atitiktų funkcijas, realizuojamas su procedūromis, esančiomis tame modulyje. Pavyzdžiui, ĮVESTIES modulį galime pavadinti "signaliniu" moduliu, kadangi sistemos išėjimai yra tik signalai.

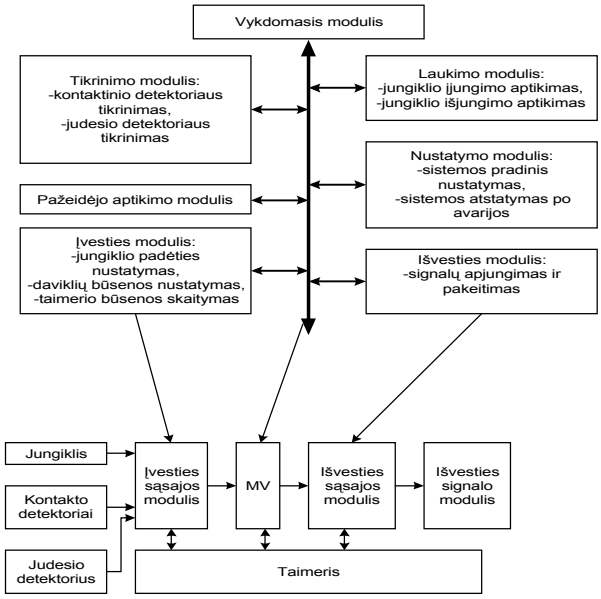


98 pav. Sistemos algoritmo struktūra

Pažymėtina, kad atskirų funkcijų ir procedūrų priskyrimo vienam ar kitam moduliui būdas priklauso nuo projektuotojo. Jeigu modulis turi gana daug procedūrų, jis gali būti suskaidytas į keletą mažesnių modulių arba keletą paprastesnių modulių, turinčių nedaug procedūrų. Projektuojamoje sistemoje galima lengvai sujungti į vieną modulį " tikrinimo ", " laukimo" ir " pažeidėjo aptikimo " modulius. Kadangi projektuojamoji sistema yra nesudėtinga, to

nedarysime. Daug sudėtingesnė sistema turėtų būti suskaidyta į posistemius. Kiekvienam jų turėtų būti sudaryta atskira funkcinė specifikacija, o kiekvienas posistemis turėtų būti projektuojamas kaip atskira nedidelė sistema.

Apsaugos sistemos visa modulinė struktūra, jungianti aparatinių ir programinių priemonių modulius, pateikta 99 pav. Nuo šio momento, laikantis pirmame skyriuje pateiktos metodikos, aparatines ir programines priemones galima projektuoti lygiagrečiai, nepriklausomai viena nuo kitos. Principinė projektuojamosios MPS schema parodyta 100 pav. Jos pagrindas - MV I80C51. Be to, reikalingas 12MHz dažnio kvarcinis rezonatorius ir RC grandis, nustatanti MV į pradinę būseną, įjungus maitinimo įtampą. Kadangi vartotojo programa bus saugoma vidinėje programų atmintyje, į išvadą EA duotas loginio vieneto lygis.



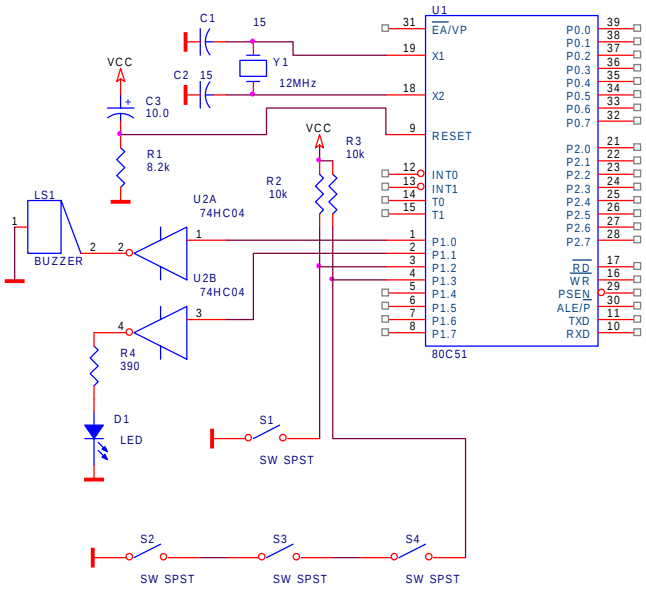
99 pav. Kompleksinė modulio struktūra

MV ryšys su davikliais ir vykdančiais įtaisais užtikrinamas per vieną jo prievadą (P1); kiti jo prievadai gali būti panaudoti vėliau modifikuojant MPS (įvedant daugiau daviklių, prijungiant klaviatūrą, slaptą raktą, perduodant pavojaus signalą telefono linija ir pan.).

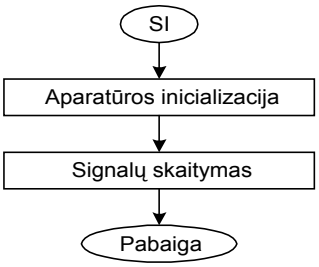
Dėl palyginti nedidelės MV išėjimų apkrovos galimybės visiems valdymo įtaisams valdyti naudojamas srovės stiprinimas. MV sąsaja sudaryta taip, kad bet kurio vykdančiojo įtaiso (sirenos, lemputės) įjungimui būtina prievado P1 atitinkamoje skiltyje suformuoti loginio nulio lygio signalą. Tada, įjungus maitinimo įtampą (MV pradinė būsena), abu valdymo įtaisai bus pasyvūs.

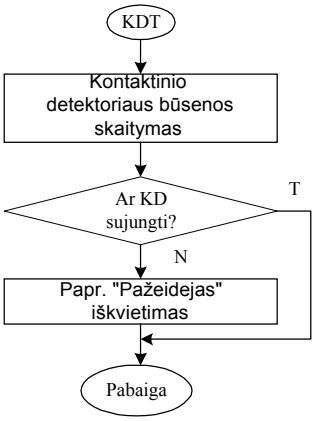
Programinio aprūpinimo projektavimą pradėsime nuo aukščiausio lygio procedūrų projektavimo, t.y. nuo vykdomosios procedūros. Laikantis "žemėjančio" projektavimo principų, vykdomoji procedūra neturėtų atlikti tikrinimų arba priimti sprendimus vietoj žemesnio lygio procedūrų. Ji turėtų paskirstyti " pareigas " žemesnio lygio procedūroms.

Įjungus maitinimo šaltinį, sistema yra inicializuojama. Po to ji ciklišk ai atlieka tolesnes procedūras tol, kol būna įjungtas maitinimo šaltinis. Jų darbo algoritmai pateikti 101 -104 pav.

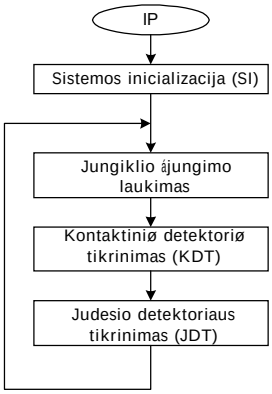


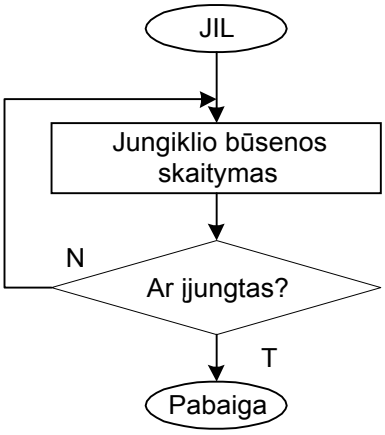
100 pav. Sistemos principinė schema.





101 pav. Sistemos inicializacija 102 pav. Kontaktinių detektorių tikrinimas



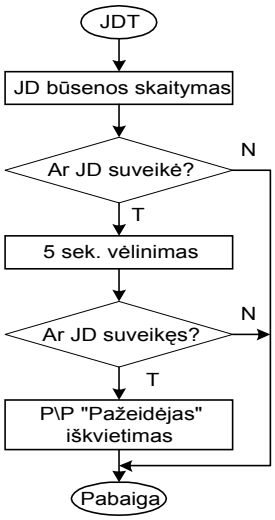


103 pav. Vykdomosios procedūros algoritmas 104 pav. Jungiklio įjungimo laukimas

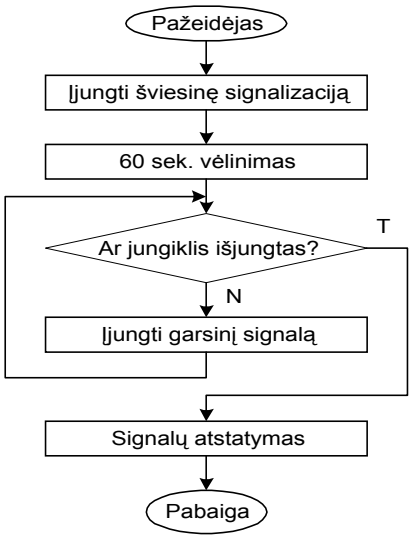
102 ir 105 pav. pateiktose schemose yra kreipiniai į paprogramę “Pažeidėjas“. Jos darbo algoritmo schema pateikta 106 pav.

Baigę projektuoti visas procedūras, patiksliname programinių modulių žemėjančius ryšius. Kaip matyti iš 106 pav., procedūra “Pažeidėjas“ iškviečia procedūras iš modulių “Laukimas“ ir “Atstatymas“, todėl būtina pastaruosius du modulius perkelti į žemesnį lygį, palyginti su procedūra “Pažeidėjas“.

Kaip jau minėta, kokybiškai ir korektiškai suprojektuotą programą užrašyti programavimo kalba (koduoti) nėra sudėtinga.



105 pav. Judesio detektoriaus tikrinimas



106 pav. Paprogramės “Pažeidėjas” darbo algoritmas

10. PRIEDAI

10.1. MP 8085 komandų sąrašas

I8085 komandų sistema 28 lentelė

1. Persiuntimo ir įkrovimo komandos

MVI r, N0	00DDD110	Taktų 7(10)	Požymių neveikia

LXI r, N0 N1	00RR0001	10	-- / --
LXI SP, N0 N1	00110001	10	--/ --
LDA N0 N1	00111010	13	--/ --
STA N0 N1	00110010	13	--/ --
LDAX r	00RR1010	7	--/ --
STAX r	00RR0010	7	--/ --
MOV r1, r2	01DDDSSS	5 (7)	--/ --
LHLD N0 N1	00101010	16	--/ --
SHLD N0 N1	00100010	16	--/ --
XTHL	11100011	18	--/ --
XCHG	11101011	4	--/ --
PCHL	11101001	5	--/ --
SPHL	11111001	5	--/ --
PUSH r	11RR0101	11	--/ --
PUSH PSW	11110101	11	--/ --
POP r	11RR0001	10	--/ --
POP PSW	11110001	10	Į visus

2. Loginės komandos

ANA r	10100SSS	A ← A ∩ r	4 (7)	Z, P, S
ORA r	10110SSS	A ← A ∪ r	4 (7)	---/ ---
XRA r	10110SSS	A ← A ∀ r	4 (7)	---/ ---
ANI N0	11100110	A ← A ∩ N0	7	---/ ---
ORI N0	11110110	A ← A ∪ N0	7	---/ ---
XRI N0	11101110	A ← A ∀ N0	7	---/ ---
CMP r	10111SSS	A ≡ r , A turinį lygina su r turiniu	4 (7)	Į visus
CPI N0	11111110	A ≡ N0	4 (7)	---/ ---
RLC	00000111	A(m+1)←Am, A0←A7, c← A7	4	c
RRC	00001111	Am ← A(m+1), A7 ← A0, c ← A0	4	--/ --
RAL	00010111	A(m+1) ← Am, A0 ← c, c ← A7	4	--/ --
RAR	00011111	Am ← A(m+1), A7 ← c, c ← A0	4	--/ --

3. Aritmetinių veiksmų komandos

INR r	11SSS111	5 (10)	C,Z,P,S
DCR r	00SSS101	5 (10)	----/ ----
INX r	00RR0011	5	Neveikia
INX SP	00110011	5	---/ ---
DCX r	00RR1011	5	---/ ---
DCX SP	00111011	5	---/ ---
ADD r	10000SSS	4(7)	Į visus
SUB r	10010SSS	4(7)	---/ ---
ADC r	10001SSS	4(7)	---/ ---
SBB r	11011SSS	4(7)	---/ ---

ADI N0	11000110	7	---/ ---
SUI N0	11010110	7	---/ ---
ACI N0	11001110	7	---/ ---
SBI N0	11011110	7	---/ ---
DAD r	00RR1001	10	---/ ---
DAD SP	00111001	10	---/ ---

4. Kitos komandos

NOP	00000000	Tuščia operacija	4	Neveikia
HLT	01110110	Sustojimas	7	---/ ---
CMA	00101111	A ← A , A turinio inversija	4	---/ ---
CMC	00111111	c ← c, keliamo vieneto inversija	4	c ← c
STC	00110111	c ← 1	4	c=1
DAA	00100111	A turinio dešimtainė korekcija	4	Neveikia
IN N0	11011011	A ← (N0), įvesti iš prievadžio N0	10	---/ ---
OUT N0	11010011	(N0) ← A, išvesti į prievadį N0	10	---/ ---
DI	11110011	Drausti pertrauktis	4	---/ ---
EI	11111011	Leisti pertrauktis	4	---/ ---
RSTn	11n111	S ← PC, PC ← 8n, n= 0,1,2,3,4,5,6,7	5/11	---/ ---

5. Nukreipimo komandos

JMP	11000011	PC←(B3)(B2),nukreipti be sąlygų	10	Neveikia
CALL	11001101	[SP-1][SP-2]←(PC),(SP)←(SP)-2,--/-	17	---/ ---
JC	11011010	Nukreipti, jei c=1	10	---/ ---
JNC	11010010	---/ --- c=0	10	---/ ---
JZ	11001010	---/ --- z=1	10	---/ ---
JNZ	11000010	---/ --- z=0	10	---/ ---
JP	11110010	---/ --- s=1	10	---/ ---
JM	11111010	---/ --- s=0	10	---/ ---
JPE	11101010	---/ --- p=1	10	---/ ---
JPO	11100010	---/ --- p=0	10	---/ ---
CC	11011100	Nukreipti į p/p, jei c=1		---/ ---
CNC	11010100	---/ --- c=0		---/ ---
CZ	11001100	---/ --- z=1		---/ ---
CNZ	11000100	---/ --- z=0		---/ ---
CP	11110100	---/ ---		---/ ---

		s=1		
CM	11111100	---/ --- s=0		---/ ---
CPE	11101100	---/ --- p=1		---/ ---
CPO	11100100	---/ --- p=0		---/ ---
RET	11001001	Besąlyginis grįžimas iš p/p		---/ ---
RC	11011000	Grįžti iš p/p, jei c=1		---/ ---
RNC	11010000	---/ --- c=0		---/ ---
RZ	11001000	---/ --- z=1		---/ ---
RNZ	11000000	---/ --- z=0		---/ ---
RP	11110000	---/ --- s=1		---/ ---
RM	11111000	---/ --- s=0		---/ ---
RPE	11101000	---/ --- p=1		---/ ---
RPO	11100000	---/ --- p=0		---/ ---

31 lentelėje yra panaudoti tokie pažymėjimai. Raide r žymimas vienas iš MP registrų (B, C, D, E, H, L, M arba A). Šiems registrams atitinka kodai R (0, 1, 2, 3, 4, 5, 6, 7). Dviejų baitų komandose operando kodas žymimas kaip N_0. Trijų baitų komandose antro ir trečio baitų kodai N_0N_1 nurodo atminties ląstelės adresą. Rodyklės, naudojamos "komandos turinys" stulpelyje, nurodo duomenų kryptį, pvz., užrašas $r \leftarrow N_0$ reiškia, kad skaičius patalpinamas registre 7, $r_1 \leftarrow r_2$ - registro r2 turinys persiunčiamas į registrą *r1*-, o užrašas $[rr'] \leftarrow A$ - registro A turinys persiunčiamas į atminties ląstelę, kurios adresas nurodytas registrų poroje *rr'* (rr'=BC, DE, HL). Dviejų baitų skaičius kvadratiniuose skliausteliuose [N1N0] reiškia, kad duomenis reikia paimti arba patalpinti į atminties ląstelę N1N0 adresu, vieno baito skaičius lenktuose skliaustuose - I/O porto adresas (numeris). Mažoji c raidė komandas aprašo stulpelyje nurodo keliamojo vieneto požymį, o Ai (I=0,1,.., m, m+1,.., 7) - A registro i-ją sliktį.

Tų komandų , kurios turi keletą modifikacijų, operacijos kodo skiltys DDD nurodo operando imtuvo, o skiltys SSS - operando siųstuvo dvejetainį adresą, pvz. komandos MOV A, C kodas yra 01111001 (A kodas yra lygus 111, o C - 001). RR skiltys kode nurodo registrų porą (BC - 00, DE - 01, HL - 10, AF - 11).

Nukreipimo komandų grupėje ekonomijos sumetimais komandos pateiktos nenurodant antrojo ir trečiojo baitų kodų. Šios komandos yra trijų baitų (išskyrus vieno baito grįžimo iš paprogramės komandas). Antrame ir trečiame baituose yra adresas, prie kurio reikia pereiti. Adresas, vykdant šias komandas, įrašomas į PC.

10.2. MV 8051 komandų sąrašas

I8051 komandos, modifikuojančios požymius 29 lentelė

Komanda	Požymiai	Komanda	Požymiai
ADD	C, OV, A	CLR C	C=O
ADDC	C, OV, AC	CPL C	C=C/
SUBB	C, OV, AC	ANL C, b	
MUL	C=O, OV		
DIV	C=O, OV	ORL c, b	C
DA	C	ORL c/b	C
RRC	C	MOV c, b	C
RLC	C	CJNE	C
SETB C	C=1		

I8051 komandų sistemoje panaudotų simbolių reikšmės 30 lentelė

A	Kaupiklis
AC	Pagelginis keliamasis vienetas
ACC	Kaupiklio simbolinis vardas
ad	RDA (0-127), prievadžio ar spec. funkcijų registro adresas
add	Tiesioginis 8-bitų imtuvo adresas
ads	Tiesioginis 8-bitų siųstuvo adresas
ad11	Tiesioginis 11-bitų nukreipimo adresas
ad16	Tiesioginis 16-bitų nukreipimo adresas
ad16h	Vyresnysis 16-bitų adreso baitas
ad16l	Jaunesnysis 16-bitų adreso baitas
B	Kaupiklio pagelbinis registras
#d	Tiesioginis 8-bitų operandas (konstantė)
#d16	Tiesioginis 16-bitų operandas
#d16h	Vyresnysis tiesioginio 16-bitų operando adreso baitas
#d16l	Jaunesnysis tiesioginio 16-bitų operando adreso baitas
i	OK bitas, nustatantis netiesioginio adreso registrą I=0,1 (R0,R1)
PSW	Programos būsenos žodis (A+F)
PX.Y	X prievadžio Y bito simbolinis vardas
rel	8-bitų santykinis nukreipimo adresas (-127--128)
rrr	OK bitų grupė, nustatanti BPR (R0--R7)
Ri	Apibendrintas netiesioginio adreso registro vardas (R0 arba R1)

RS	PSW bitas, išrenkantis registrų banką
TCON	Taimerio valdymo/būsenos registras
IF	Taimerio perpildymo požymis
TH	Taimerio vyresnysis baitas
TL	Taimerio jaunesnysis baitas
TMOD	Taimerio režimų registras
←	Priskyrimo operatorius
↔	Tarpusavio sukeitimo operatorius
∩, ∪, ∀	Loginių operacijų IR, ARBA ir suma mod. 2 operatoriai
@	Netiesioginio adresavimo ženklas
#	Tiesioginio operando ženklas
(Y)	Registro arba atminties ląstelės Y turinys
((Y))	Atminties ląstelės, adresuojamos Y turiniu, turinys
	VKESM komandų skaitiklio momentinė reikšmė
AND,OR,NOT	Loginės operacijos IR, ARBA ir NE,vykdomos transliuojant programą
B,H	Dvejetainio ir šešioliktainio kodų ženklai
HIGH/ LOW	Vyresniojo/jaunesniojo baito išskyrimo iš d16 transliuojant loginė opracija

Duomenų perdavimo komandos 31 lentelė

Pseudokodas	OK	T	B	C	Operacija	Pastaba
MOV A,Rn	11101rrr	1	1	1	(A)←(Rn)	n=0—7
MOV A,ad	11100101	3	2	1	(A)←(ad)	
MOV A,@Ri	1110011i	1	1	1	(A)←((Ri))	i=0,1
MOV A,#d	01110100	2	2	1	(A)←#d	A įkrovimas konst.
MOV Rn,A	11111rrr	1	1	1	(Rn)←(A)	Siųsti iš A į Rn
MOV Rn,ad	10101rrr	3	2	2	(Rn)←(ad)	
MOV Rn,#d	01111rrr	2	2	1	(Rn)←#d	Įkrauti į Rn konst.
MOV ad,A	11110101	3	2	1	(ad)←A	A siųsti tiesiog. adr.
MOV ad,Rn	10001rrr	3	2	2	(ad)←(Rn)	
MOV add,ads	10000101	9	3	2	(add)←(ads)	
MOV ad,@Ri	1000011i	3	2	2	(ad)←((Ri))	
MOV ad,#d	01110101	7	3	2	(ad)←#d	Konst. tiesiog. adr.
MOV @Ri,A	1111011i	1	1	1	((Ri))←(A)	Siųsti iš A į RDA
MOV @Ri,ad	0110011i	3	2	2	((Ri))←(ad)	
MOV @Ri,#d	0111011i	2	2	1	((Ri))←#d	
MOV DPTR,#16	10010000	13	3	2	(DPTR)←#16	Užkrauti DPTR
MOVC	1001001	1	1	2	(A)←((A)+(DPTR))	Iš PA siųsti į A

A,@A+DPTR	1					
MOVC A,@A+PC	10000011	1	1	2	(A)←((A)+(PC))	(PC)←(PC)+1
MOVX A,@Ri	1110001i	1	1	2	(A)←((Ri))	
MOVX A,@DPTR	11100000	1	1	2	(A)←((DPTR))	
MOVX @Ri,A	1111001i	1	1	2	((Ri))←(A)	
MOVX @DPTR,A	11110000	1	1	2	((DPTR))←(A)	
PUSH ad	11000000	3	2	2	((SP))←(ad)	(SP)←(SP)+1
POP ad	11010000	3	2	2	(ad)←(SP)	(SP)←(SP)-1
XCH A,Rn	11001rrr	1	1	1	(A)↔(Rn)	Sukeisti A su Rn
XCH A,ad	11000101	3	2	1	(A)↔(ad)	
XCH A,@Ri	1100011i	1	1	1	(A)↔((Ri))	
XCHD A,@Ri	1101011i	1	1	1	(A0-A3)↔((Ri(0-3)	Keistis jaun. tetr.

Aritmetinės komandos 32 lentelė

Pseudokodas	OK	T	B	C	Operacija	Pastabos
ADD A,Rn	00101rrr	1	1	1	(A)←(A)+(Rn)	A sudėti su Rn n=0-7
ADD A,ad	00100101	3	2	1	(A)←(A)+(ad)	A + su t. adr. baitu
ADD A,@Ri	0010011i	1	1	1	(A)←(A)+((Ri))	A + su baitu iš RDA
ADD A,#d	00100100	2	2	1	(A)←(A)+#d	Sudėti A su konst.
ADDC A,Rn	00111rrr	1	1	1	(A)←(A)+(Rn)+(c)	c—keliamasis vienetas
ADDC A,ad	00110101	3	2	1	(A)←(A)+(ad)+(c)	
ADDC A,@Ri	0011011i	1	1	1	(A)←(A)+(Ri)+(c)	
ADDC A,#d	00110100	2	2	1	(A)←(A)+#d+(c)	
DA A	11010100	1	1	1		A dešimt. korekcija
SUBB A,Rn	10011rrr	1	1	1	(A)←(A)-(c)-(Rn)	
SUBB A,ad	10010101	3	2	1	(A)←(A)-(c)-((ad))	
SUBB A,@Ri	1001011i	1	1	1	(A)←(A)-(c)-((Ri))	
SUBB A,#d	10010100	2	2	1	(A)←(A)-(c)-#d	
INC A	00000100	1	1	1	(A)←(A)+1	Prie A pridėti 1
INC Rn	00001rrr	1	1	1	(Rn)←(Rn)+1	n=0---7
INC ad	00000101	3	2	1	(ad)←(ad)+1	
INC @Ri	0000011i	1	1	1	((Ri))←((Ri))+1	
INC DPTR	10100011	1	1	2	(DPTR)←(DPTR)+1	
DEC A	00010100	1	1	1	(A)←(A)-1	Iš A atimti 1
DEC Rn	00011rrr	1	1	1	(Rn)←(Rn)-1	

DEC ad	00010101	3	2	1	(ad)←(ad)-1	
DEC @Ri	0001011i	1	1	1	((Ri))←((Ri))-1	i=0,1
DIV AB	1000010	1	1	4	(A),(B)<-(A):(B)	A dalyba iš B
MUL AB	10100100	1	1	4	(B)(A)←(A)×(B)	A daugyba iš B

Loginės komandos 33 lentelė

Pseudokodas	OK	T	B	C	Operacija	Pastaba
ANL A,Rn	01011rrr	1	1	1	(A)←(A)∩ (Rn)	n=0--7
ANL A,ad	01010101	3	2	1	(A)←(A)∩ (ad)	
ANL A,@Ri	0101011i	1	1	1	(A)←(A)∩ ((Ri))	i=0,1
ANL A,#d	01010100	2	2	1	(A)←(A)∩ #d	
ANL ad,A	01010010	3	2	1	(ad)←(ad) ∩(A)	
ANL ad,#d	01010011	7	3	2	(ad)←(ad) ∩ #d	
ORL A,Rn	01001rrr	1	1	1	(A)←(A)∪ (Rn)	
ORL A,ad	01000101	3	2	1	(A)←(A)∪ (ad)	
ORL A,@Ri	0100011i	1	1	1	(A)←(A)∪ ((Ri))	
ORL A,#d	01000100	2	2	1	(A)←(A)∪ #d	
ORL ad, A	01000010	3	2	1	(ad)←(ad) ∪(A)	
ORL ad, #d	01000011	7	3	2	(ad)←(ad) ∪ #d	
XRL A,Rn	01101rrr	1	1	1	(A)←(A) ∀ (Rn)	
XRL A,ad	01100101	3	2	1	(A)←(A) ∀ (ad)	
XRL A, @Ri	0110011i	1	1	1	(A)←(A) ∀ ((Ri))	
XRL A, #d	01100100	2	2	1	(A)←(A) ∀ #d	
XRL ad, A	01100010	3	2	1	(ad)←(ad) ∀ (A)	
XRL ad, #d	01100011	7	3	2	(ad)←(ad) ∀ #d	
CLR A	11100100	1	1	1	(A)← 0	
CPL A	11110100	1	1	1	(A)←(A)	
RL A	0010001	1	1	1	(A(n+1))←	n=0--6

	1				(An), (A0)← (A7)	
RLC A	00100011	1	1	1	(A(n+1))← (An),(A0) ←©,©←(A7	
RR A	00000011	1	1	1	(An)←(A(n+1)), (A7)← (A0)	
RRC A	00010011	1	1	1	(An)←(A(n+1)),(A7)← ©,©←(A0	
SWAP A	11000100	1	1	1	(A0--A3) ↔ (A4--A7)	

Operacijų su bitais komandos 34 lentelė

Pseudokodas	OK	T	B	C	Operacija	Pastaba
CLR c	11000011	1	1	1	©← 0	
CLR bit	11000010	4	2	1	(b) ← 0	Bito nust. į 0
SETB c	11010011	1	1	1	© ← 1	c-kel. vienetas
SETB bit	11010010	4	2	1	(b) ← 1	
CPL c	10110011	1	1	1	© ← ©	c inversija
CPL bit	10110010	4	2	1	(b) ← (b/)	bito inversija
ANL c,bit	10000010	4	2	2	© ← © ∩ (b)	
ANL c,/b	10110000	4	2	2	© ← © ∩ (b/)	b/-bito inversija
ORL c,bit	01110010	4	2	2	© ←© ∪ (b)	
ORL c,bit	10100000	4	2	2	© ←© ∪ (b/)	b/- bito inversija
MOV c,bit	10100010	4	2	1	© ← (b)	
MOV bit,c	10010010	4	2	2	(b) ← (c)	

Nukreipimo komandos 35 lentelė

Pseudokodas	OK	T	B	C	Operacija

LJMP ad16	00000010	12	3	2	(PC) ← ad16
AJMP ad11	00001	6	2	2	(PC)←(PC)+2,(PC(0-10)←ad11
SJMP rel	10000000	5	2	2	(PC)←(PC)+2,(PC)←(PC)+rel
JMP	01110011	1	1	2	(PC)←(A)+(DPTR)
JZ rel	01100000	5	2	2	(PC)←(PC)+2;jei (A)=0,tai (PC)←(PC)+rel
JNZ rel	01110000	5	2	2	(PC)←(PC)+2;jei (A)≠0,tai (PC)←(PC)+rel
JC rel	01000000	5	2	2	(PC)←(PC)+2; jei ©=1,tai (PC)←(PC)+rel
JNC rel	01010000	5	2	2	(PC)←(PC)+2; jei ©=0, tai (PC)←(PC)+rel
JB bit,rel	00100000	11	3	2	(PC)←(PC)+3; jei (b)=1, tai (PC)←(PC)+rel
JNB bit,rel	00110000	11	3	2	(PC)←(PC)+3; jei (b)=0, tai (PC)←(PC)+rel
JBC bit,rel	00010000	11	3	2	(PC)←(PC)+3;jei(b)=1,tai(PC←(PC+rel,b=0
DJNZ Rn,rel	11011rrr	5	2	2	(PC)←(PC)+2, (Rn)←(Rn)-1;jei (Rn)≠0,tai (PC)←(PC)+rel
DJNZ ad,rel	11010101	8	3	2	(PC)←(PC)+2, (ad)←(ad)-1; jei (ad)≠0, tai (PC)←(PC)+rel
CJNE A,ad,rel	10110101	8	3	2	(PC)←(PC)+3;jei(A)≠(ad),tai(PC)←(PC)+rel,jei (A)<(ad),tai (c)←1,kitaip (c)←0
CJNE A,#d,rel	10110100	10	3	2	(PC)←(PC)+3,jei(A)≠#d,tai(PC)←(PC)+rel, jei(A) < #d, tai (c)←1, kitaip (c)←0
CJNE Rn,#d,rel	10111rrr	10	3	2	(PC)←(PC)+3,jei(Rn)≠#d,tai(PC)←(PC)+rel jei (Rn)<#d, tai ©←1, kitaip ©←0
CJNE@Ri,#d,rel	1011011i	10	3	2	(PC)←(PC)+3,jei((Ri))≠#d, tai (PC)←(PC)+ rel,jei((Ri))<#d,tai©←1,kitaip ©←0
LCALL ad16	00010010	12	3	2	(PC)←(PC)+3, (SP)←(SP)+1,((SP))←(PC(0-7)), (SP)←(SP)+1, ((SP))←(PC(8-15)), (PC)← ad16
ACALL ad11	10001	6	2	2	(PC)←(PC)+2, (SP)←(SP)+1,((SP))←(PC(0--7)), (SP)←(SP)+1, ((SP))←(PC(8-15)), (PC)← ad16
RET	00100010	1	1	2	(PC(8-15))← ((SP)), (SP)←(SP)-1, (PC(0-7)) ←((SP)), (SP) ←(SP)-1
RETI	00100010	1	1	2	(PC(8-15)) ← ((SP)), (SP) ← (SP)-1 (PC(0-7)) ← ((SP)), (SP) ← (SP)-1
NOP	00000000	1	1	1	(PC)← (PC) +1; tuščia operacija

10.3. I2C sąsajos valdymo programų tekstai

```
;  Sąsajos I2C valdymas kai MV dirba vedamojo režime


        BITCNT          DATA    8h          ; bitų skaitiklis
        BYTECNT         DATA    030h        ;  bitų skaitiklis
SLAVEADD        DATA    032h        ; vedamojo adresas

FLAGS           DATA    28h
NOACK           BIT     FLAGS.0 ; I2C neteisingo atsakymo požymis
BUSY            BIT     FLAGS.1 ; I2C užimtumo požymis
ERROR           BIT     FLAGS.2 ; I2C klaidos požymis
MISTAKE         BIT     P3.4

CSEG            ; programos kodo segmento pradžia
ORG 0000H
      JMP PRADZIA

ORG 007BH                   ; Paprogramės

;---------------------------------------------------------------------
; Vėlinimas signalams ( SCLOCK , SDATA )
;---------------------------------------------------------------------

DELAY:

      NOP
      RET

;---------------------------------------------------------------------
; STOP bito perdavimas
;---------------------------------------------------------------------

SENDSTOP:

      SETB    MDE         ; MV SDATA išvadas išvedimui
      CLR     MDO         ;  SDATA paruoštas stopui
      SETB    MCO         ; stop sinchronizacija
      ACALL   DELAY
      SETB    MDO         ; formuojamas stop
```

```
        CLR     BUSY
        RET

    ;-------------------------------------------------------------------
    ; Baito siuntimas vedamajam
    ;-------------------------------------------------------------------

    SENDBYTE:

        MOV     BITCNT,#8
        SETB    MDE
        CLR     MDO
        CLR     MCO
    LOOP:
        RLC     A           ;  siųsti bitą
        MOV     MDO,C
        SETB    MCO         ; sinchronizacija on
        CLR     MCO         ; sinchronizacija off
        DJNZ    BITCNT,LOOP
        CLR     MDE         ; duomenų liniją perjungti patvirtinimo
priėmimui
        SETB    MCO         ; sinchronizacija patvirtinimui
        JNB     MDI,NEXT    ; tikrinimas
        SETB    NOACK       ; patvirtinimo nėra
    NEXT:
        CLR     MCO             RET
    ;-------------------------------------------------------------------
    ; START bito ir vedamojo adreso perdavimas
    ;-------------------------------------------------------------------

    BITSTART:
        SETB    BUSY
        SETB    MDE
        CLR     NOACK
        CLR     ERROR
        JNB     MCO,FAULT
        JNB     MDO,FAULT
        CLR     MDO             ; formuojamas
        ACALL   DELAY           ; START
        CLR     MCO             ; bitas
    FAULT:
        CLR     MISTAKE         ; nustatomas klaidos požymis
```

```
      MOV     A,SLAVEADD      ; priimti vedamojo adresą
      ACALL   SENDBYTE        ; kviesti adreso perdavimo paprogramę
      RET
;---------------------------------------------------------------------
; Adreso ir duomenų perdavimas į vedantįjį
;---------------------------------------------------------------------
SENDATA:
      ACALL   BITSTART
      JB      MDI,NEXT1
      MOV     A,#00
SLOOP:
      MOVX    A,@DPTR
      ACALL   SENDBYTE
      INC     DPTR
      JB      NOACK,NEXT1
      DJNZ    BYTECNT,SLOOP
NEXT1:
      ACALL   SENDSTOP
      MOV     A,FLAGS
      ANL     A,#07h
      JZ      RETURN
      SETB    P3.4
      CLR     I2CRS
RETURN:
      RET

;---------------------------------------------------------------------
; Baito priėmimas iš vedamojo
;---------------------------------------------------------------------
RCVBYTE:
      MOV     BITCNT,#8       ; Bitų skaitiklis.
      CLR     MDE             ;Duomenų išvadas - priėmimui
      CLR     MCO
LOOP2:
      SETB    MCO
      CLR     MCO
      MOV     C,MDI           ; priimamas duomenų bitas
      RLC     A               ; bito talpinimas į baitą
      DJNZ    BITCNT,LOOP2
      PUSH    ACC
      SETB    MDE
      MOV     A,BYTECNT
```

```
      CJNE    A,#1,SACK
      SETB    MDO
      SJMP    NACK
SACK:
     CLR     MDO            ; perduodamas patvirtinimas
NACK:
      SETB    MCO            ; patvirtinimo sinchronizacija
      POP     ACC
      ACALL   DELAY
      CLR     MCO
      SETB    MDO
      ACALL   DELAY
      CLR     MDE
      RET

;---------------------------------------------------------------------
;  Duomenų priėmimas iš vedamojo
;---------------------------------------------------------------------
RCVDATA:
      INC     SLAVEADD       ; nustatoma skaitymo būsena
      ACALL   BITSTART       ; perduodamas vedamojo adresas
      JB      NoAck,RDEX     ; tikrinama ar atsiliepia
RDLoop:
      ACALL   RCVBYTE        ; priimti sekantį baitą
      MOV     @R1,A          ; duomenis išsaugoti
       INC     R1
      DJNZ    BYTECNT,RDLoop
RDEX:
      ACALL   SENDSTOP       ; STOP perdavimas
       RET
;*************************************************************
;  Pagrindinė programa
;*************************************************************
PRADZIA:
      MOV     SP,#040h
      CLR     NOACK
      MOV     SLAVEADD,#088H
      MOV     BYTECNT,#3
      MOV     I2CCON,#0A8h
      MOV     R1,#035h
      ACALL   RCVDATA
```

```
END
```

Kai MV dirba vedamojo režime.

```
BYTECNT       DATA    030h    ; skaitiklis

FLAGS         DATA    28h
GO            BIT     FLAGS.0       ; pertraukčių požymis
RC            BIT     FLAGS.1       ; rašymo pertraukties požymis
TR            BIT     FLAGS.2       ; skaitymo pertraukties požymis

CSEG                                ; programos kodo segmentas
ORG 0000H

      JMP pradzia1


ORG 003Bh                 ; I2C vedamojo pertraukties aptarnavimas

      JB     RC,RECEIVE
      JB     TR,TRANSMIT

ORG 007BH                 ; Paprogramės

;---------------------------------------------------------------------
; Pertraukties aptarnavimas priėmimui
;---------------------------------------------------------------------
RECEIVE:
      SETB    GO
      MOV     @R1,I2CDAT      ; duomenys vidiniame RAM
      CLR     I2CI            ; gesinti pertraukties požymį
      RETI
;---------------------------------------------------------------------
; Pertraukimo aptarnavimas perdavimui
;---------------------------------------------------------------------

TRANSMIT:
      SETB    GO
      MOV     I2CDAT,R0
      CLR     I2CI
      RETI
```

```
RCVBYTE2:

        NOP
        RET

;-------------------------------------------------------------------------
; Baito priėmimas
;-------------------------------------------------------------------------

RCVBYTE:

        JNB     GO,$            ; laukti pertraukties
        INC     R1
        CLR     GO              ; gesinti požymį sekančiai pertraukčiai
        RET

;-------------------------------------------------------------------------
; Baitų priėmimas
;-------------------------------------------------------------------------

RCVDATA:

        MOV     BYTECNT,#4      ; 4 bytai : addresas + 3 baitai duomenų
LOOP2:
        ACALL   RCVBYTE
        DJNZ    BYTECNT,LOOP2
        RET

;-------------------------------------------------------------------------
; Baito siuntimas
;-------------------------------------------------------------------------

SENDBYTE:
        JNB     GO,$            ;laukti pertraukties
        INC     R0
        CLR     GO
        RET

;-------------------------------------------------------------------------
;Baitų siuntimas
;-------------------------------------------------------------------------
```

```
SENDATA:

    MOV    BYTECNT,#3     ;vedamajam siunčiami 3 duomenų baitai
LOOP:
    ACALL  SENDBYTE
    DJNZ   BYTECNT,LOOP
    RET

;*************************************************************
; Pagrindinė programa
;*************************************************************

pradzia1:

    CLR    GO              ; gesinti požymius
    MOV    I2CADD,#044h    ; vedamojo adresas
    MOV    SP,#020h
    MOV    IE,#80h         ; leisti pertrauktis
    MOV    IE2,#01h        ; leisti I2C pertrauktį
    MOV    I2CCON,#000h    ; vedamojo darbo režimas
    SETB   RC              ; spec. požymis pertraukčių aptarnavimui
    MOV    R0,#033h        ; duomenys
    ACALL  RCVBYTE2        ; adreso priėmimas iš vedančiojo
    CLR    RC
    SETB   TR
    ACALL  SENDATA         ; vedamasis siunčia 3 baitus
    END
```

10.4. PIC komandų sistema ir programavimo elementai

PIC šeimos mikrovaldikliai turi efektyvią komandų sistemą, susidedančią tik iš 35 komandų. Visos komandos, išskyrus perėjimų komandas (joms atlikti reikia dviejų MC), išpildomos per vieną mašininį ciklą (keturi TIG įtampos periodai). Tada esant maksimaliam 10 MHz TIG dažniui, komandos išpildymas užtrunka tik 400 (800) nsek. Kiekvieną komandą sudaro 14 bitų žodis, dalijamas į operacijos kodą ir operandą (tiesioginiai duomenys, atminties ląstelės, registrai ir pan.) Komandų sistema pateikta 36 lentelėje.

Visas komandas galima suskirstyti į tris grupes: operacijų su baitais, bitais ir perėjimų komandas. TRIS ir OPTION komandos patalpintos į komandų sistemą todėl, kad suderinti ankstesnės šeimos valdiklių PIC16Cx veiksmus su PIC16C84 valdikliais (vėlesnėse PIC16C84 modifikacijose jų gali ir nebūti).

Programavimas PIC asemblerio kalba neturėtų sudaryti ypatingų sunkumų jau vien dėl to, kad komandų sistemą sudaro tik 35 originalūs mnemokodai. Kas liečia programos rašymo taisykles, asemblerio operatoriaus sudėtį, tai jie lygiai tokie patys, kaip ir anksčiau aprašytų MP (operatorių sudaro keturi privalomi laukeliai, komentarų laukas atskiriamas nuo operandų lauko kabliataškiu, komandose gali būti naudojami ne tik skaimeninės operandų reikšmės, bet ir simboliniai jų vardai, įvedami asemblerio direktyva EQU, ir t.t.) Dauguma komandų (aritmetinės, loginės, persiuntimo) savo atliekamais veiksmais yra labai panašios į analogiškus veiksmus atliekančias MP I8085 ar MV I8051 komandas. Žemiau aptarsime tik charakteringas PIC komandas, spręsdami keletą mokomųjų uždavinių.

Uždavinys Nr.1. Tarkime kad prie PIC prievado išvadų prijungti aštuoni šviesos diodai, o prie prievado A nulinės skilties išvado – elektromechaninio daviklio kontaktai, kuriuos sujungus į A išvadą patenka loginis "1". Programa, aptikus šį faktą, turi pašalinti kontaktų virpėjimo įtaką ir uždegti šviesos diodus.

```
;   "Šviesos diodų uždegimas".
;    Įvedimo (išvedimo) registrai
CONTRPORT  EQU       05h
DATAPORT    EQU       06h
; RAM ląstelė
BUFER1          EQU       0Ch
BUFER2          EQU       0Dh
; Valdantieji registrai
TRISA           EQU       85h
TRISB           EQU       86h
;  Prievadų A ir B inicializavimo žodžiai
INITA           EQU       B'11111111'
INITB           EQU       B'00000000'

ORG             0h
GOTO            BEGIN
;
ORG             100h
BEGIN:
      MOVLW      INITA        ;Užkrauti  INIT A konstantę į W
      MOVWF      TRISA        ;Nustatyti A prievadą įvedimui
      MOVLW      INITB        ;Užkrauti INITB konstantę į W
      MOVWF      TRISB        ;Nustatyti B prievadą išvedimui
      MOVLW      00h          ;Apnulinti W registrą
      MOVWF    DATAPORT       ;Užgesinti šviesos diodus
```

```
        MOVLF     03h          ;Į W įrašyti 3h
        MOVWF     DARB1        ;W turinį persiųsti į DARB1
LOOP1:
        BTFSS  CONTRPORT,0  ;Patikrinti , ar A0=1
        GOTO      LOOP1        ;Laukti,kol A0=0
        MOVLW     0FFh         ;Į W įrašyti 0FFh
.       MOVWF     DARB2        ;W turinį persiųsti į DARB2
LOOP2:
        DECFSZ    DARB2,1      ;Sumažinti DARB2 turinį vienetu
        GOTO      LOOP2        ;ir grįžti atgal, jei (DARB2)≠0
        DECFSZ    DARB1        ;Sumažinti DARB1 turinį vienetu
        GOTO      LOOP1        ;ir grįžti prie LOOP1, jei  (DARB1)≠0
        MOVWF   DATAPORT    ;Uždegti šviesos diodus
        GOTO      $            ;Amžinas ciklas

 END
```

Iš pradžių su asemblerio direktyva EQU suteikėme registrų simboliniams vardams konkrečias jų reikšmes (adresus). Pirmosios keturios programos eilutės nustato prievadų A ir B konfiguraciją (A – įvedimui, B – išvedimui), sekančios dvi gesina šviesos diodus.

Komanda BTFSS CONTROLPORT,0 tikrina nulinę prievado A skiltį ir, jeigu šis bitas lygus “1”, praleidžiama tolesnė komanda (GOTO LOOP1). Panašiai veikia ir komanda DECFSZ DARB2,1, kuri sumažina DARB2 registro turinį vienetu ir, jeigu registro turinys tapo lygus 0, nevykdoma komanda GOTO LOOP2.

Komandos DECFSZ šioje programoje veikia uždelsimo cikle, sudarytame iš jų ir komandų GOTO. Vidinis uždelsimo ciklas vykdomas 255 kartus, išorinis – tris kartus. Jeigu MV taktinis dažnis lygus 4 MHz., mašininis ciklas bus įvykdytas per 1 mksek, vidinio ciklo trukmė bus lygi 255(1+20=765 mksek, o uždelsimo ciklo - Tužd.=3*(765+3)=2.3 msek. Taigi praėjus šiam laikui, bus uždegti šviesos diodai, o programa pereis į amžiną ciklą (asemblerio direktyva $ reiškia PC momentinę reikšmę, todėl komanda GOTO $ nurodo perėjimą ten, kur šiuo momentu randamės).

PIC 16C84 komandų sistema 36 lentelė

Mnemonika, operandai	Aprašymas	Ciklai	Veikia į požymius
Operacijų su baitais komandos			
ADDWF f, d	Sudėti W su f	1	C, DC, Z
ANDWF f, d	Loginis IR tarp W ir f	1	Z

CLRF	f	Išvalyti f	1	Z
CLRW		Išvalyti W	1	Z
COMF	f, d	Registro f turinio inversija	1	Z
DECF	f, d	Dekrementuoti f	1	Z
DECFSZ	f, d	Dekrementuoti f, praleisti komandą, jei 0	1(2)	
INCF	f, d	Inkrementuoti f	1	Z
INCFSZ	f, d	Inkrementuoti f, praleisti komandą, jei 0	1(2)	
IORWF	f, d	Loginis ARBA tarp W ir f	1	Z
MOVF	f, d	Registro f turinio persiuntimas	1	Z
MOVWF	f	Persiųsti W turinį į f	1	
NOP		Tuščia operacija	1	
RLF	f, d	Perstumti f turinį į kairę per C	1	C
RRF	f, d	Perstumti f turinį į dešinę per C	1	C
SUBWF	f, d	Atimti W iš f	1	C, DC, Z
SWAPF	f, d	Sukeisti vietomis f tetradas	1	
XORWF	f, d	Suma mod.2 W su f	1	Z
ADDLW	k	Konstantės k sudėtis su W turiniu	1	C, DC, Z
ANDLW	k	Loginis IR tarp W irk	1	Z
IORLW	k	Loginis ARBA tarp W irk	1	Z
SUBLW	k	Atimti W iš k	1	C, DC, Z
MOVLW	k	Konstantę k įkrauti į W	1	
XORLW	k	Suma mod.2 W su k	1	Z
OPTION		W turinio įkrovimas į OPTION registrą	1	
TRIS	f	TRIS registro įkrovimas	1	
Operacijų su bitais komandos				
BCF	f, b	Išvalyti bitą f registre	1	
BSF	f, b	Nustatyti bitą f registre	1	
BTFSC	f, b	Praleisti komandą, jei bitas lygus 0	1(2)	
BTFSS	f, b	Praleisti komandą, jei bitas lygus 1	1(2)	
Perėjimų ir valdymo operacijų komandos				
CALL	k	Iškviesti paprogramę	2	
CLRWDT		Išvalyti WDT taimerį	1	
GOTO	k	Pereiti nurodytu adresu	2	
RETLW	k	Grįžti iš paprogramės įkraunant k į W	2	
RETFIE		Grįžti po programos pertraukties	2	
RETURN		Grįžti iš paprogramės	2	
SLEEP		Pereiti į tuščios eigos režimą	1	

Paaiškinimai

Operacijų su baitais komandose “f” žymi registrą, su kurio turiniu vykdomi veiksmai, “d” bitas nurodo rezultato imtuvą. Jeigu “d”= 0, tai rezultatas bus

patalpintas į W registrą, priešingu atveju pasiliks komandoje, nurodytoje registre f.

Operacijų su bitais komandose “b” nurodo bito, su kuriuo komanda atlieka veiksmus, numerį, o “f” – registrą, kuriame yra šis bitas.

Komandose, kurios atlieka perėjimus programoje ir veiksmus su konstantėmis, “k” nurodo aštuonių arba vienuolikos bitų konstantę.

Žemiau pateikiamas tekstas programos, nuskaitančios serijos numerį iš Multidrop (tai firmos Dallas Semiconductor [17] sąsajos tipas) įtaiso (tai elektroninis raktas DS1990 – TOUCH MEMORY) ir sulyginančios jį su atmintyje saugomu “pažįstamu” numeriu. Jei numeriai sutampa, įjungiamas šviesos diodas. Pakartotinai prijungus raktą, šviesos diodas išjungiamas. Tokį įtaisą galima panaudoti apsaugos, identifikacijos, sankcionuoto praėjimo sistemose. Brėžiniuose pateiktos duomenų mainų Multidrop magistralėje laiko diagramos – įtaiso inicializavimas, duomenų rašymas ir skaitymas. Detalų Multidrop protokolo aprašymą, taip pat IG aprašymus galima rasti firmos Dallas Semiconductor literatūroje [25]. Duomenų mainus realizuoja paprogramės RESET (pradinis nustatymas), WRITE (baito rašymas) ir READ (baito nuskaitymas). Baitas skaitomas ir rašomas pradedant nuo vyriausio bito. Maksimalus mainų greitis yra 16,3 kbps. Laiko intervalai programoje formuojami panaudojant programinį vėlinimą. Vykdant mainus **BŪTINA** uždrausti pertrauktis.

```
; Darbo su Dallas Multidrop įtaisu pavyzdys
; Programa nuskaito įtaiso serijinį numerį iš TOUCH MEMORY, sulygina
; su atmintyje saugomu numeriu ir, jei jie sutampa,
; atitinkamai įjungia arba išjungia šviesos diodą.
; TOUCH MEMORY prijungta prie PORTB.0 (reikalingas 4,7k pull up rezistorius),
; šviesos diodas prijungtas prie PORTB.1

        list P=16c84      ; PIC tipas

;       Specialios paskirties registru aprašymas
;.....
PORTA   =     0x05   ;A PRIEVADO REGISTRAS
PORTB   =     0x06   ;B PRIEVADO REGISTRAS
TRISA   =     0x85   ;A PRIEVADO KRYPTIES REGISTRAS
TRISB   =     0x86   ;PRIEVADO KRYPTIES REGISTRAS
STATUS  =     0x03   ;STATUS REGISTRAS
FSR     =     0x04   ;IŠRINKIMO REGISTRAS FSR
INDF    =     0x00   ;NETIESIOGINIO ADRESAVIMO REGISTRAS
TMRO    =     0x01   ;8 BITŲ TAIMERIS
```

```
    PCL     =       0x02      ;JAUNESNIEJI PC  BITAI
    PCLACH  =       0x0A    ;VYRESNIEJI PC  BITAI
    ;.............................................
    ;   STATUS REGISTRO BITAI
    ;.............................................
    C       =       0x00     ;CARRY FLAG
    DC      =       0x01     ;DIGIT CARY FLAG
    Z       =       0x02      ;ZERO FLAG
    PD      =       0x03     ;POWER DOWN FLAG
    TO      =       0x04     ;TIME OUT FROM WDT
    RP0     =       0x05     ;RAM PAGE BIT 0
    RP1     =       0x06     ;RAM PAGE BIT 1
    IRP     =       0x07     ;RAM BANK
    ;
    ;.....................................................................
    ;EEPROM REGISTRAI
    ;.....................................................................
    EEDATA  =       0x08     ;EEPROM DUOMENŲ REGSITRAS
    EEADR   =       0x09      ;EEPROM ADRESO REGISTRAS
    EECON1  =       0x08     ;EEPROM SKAITYMO RAŠYMO KONTROLĖS
REGISTRAS
    EECON2  =       0x09    ;       -"-
    ;.....................................................................
    ;EECON1 REGISTRO BITAI
    ;.....................................................................
    EERD    =       0x00       ;SKAITYMO BITAS
    EEWR    =       0x01      ;RAŠYMO BITAS
    EEWREN  =       0x02     ;LEIDIMO RAŠYTI I EEPROM ATMINTĮ
LEIDIMO BITAS
    EEWRERR =       0x03     ;RAŠYMO KLAIDOS BITAS
    EEIF    =       0x04          ;RAŠYMO PABAIGIMO BITAS
    ;.....................................................................
    ;FUNKCIJŲ REGISTRAS
    ;.....................................................................
    OPT_REG =       0x81
    PS0     =       0x00
    PS1     =       0x01
    PS2     =       0x02
    PSA     =       0x03        ;DALIKLĮ PRIJUNGIA PRIE WDT ARBA PRIE
TMR0
    T0SE    =       0x04       ;ISORINIO TAIMERIO SIGNALO FRONTAS
```

```
    T0CS    =       0x05      ;SIGNALO ŠALTINIS TAIMERIUI 0-VIDINIS, 1-
IŠORINIS
INTEDG  =       0x06        ;PERTRAUKČIŲ FRONTO BITAS
RBPU    =       0x07     ;AKTYVIOS APKROVOS PAJUNGIMAS PRIE
PORTO B
OPT_INI =       0x01    ;OPCIJŲ REGISTRO INICIALIZAVIMAS
    ;.....
    ;PERTRAUKČIŲ  APTARNAVIMO REGISTRAS
    ;.....
    INTCON  =       0x0B    ;PERTRAUKČIŲ REŽIMO REGISTRAS
    RBIF    =       0x00       ;PERTRAUKTIES PER PORTĄ B (RB<7:4>)
BITAS
    INTF    =       0x01       ;PERTRAUKTIES PER RB0/INT BITAS
    RTIF    =       0x02       ;PERTRAUKTIES PERSIPILDŽIUS TAIMERIUI
BITAS
    RBIE    =       0x03      ;PERTRAUKTIES PER PORTĄ B LEIDIMAS
    INTE    =       0x04      ;PERTRAUKTIES PER RB0/INT LEIDIMAS
    RTIE    =       0X05      ;PERTRAUKTIES TAIMERIU LEIDIMAS
    EEIE    =       0x06       ;PERTRAUKTIES ĮRAŠIUS DUOMENIS Į
EEPROM LEIDIMAS
    GIE     =       0x07       ;VISŲ RŪŠIŲ PERTRAUKČIŲ LEIDIMAS
    LED     =       0x01       ;LED
    DSBIT   =       0x00
    TMP     =       0x0C
    TMP1    =       0x0D
    TMP2    =       0x0E
    TIME1   =       0x0F
    TIME2   =       0x10
    TMP0    =       0x19
    COUNT   =       0x1A
    ;*****************************************************************
***********
        Org     0x00
        Goto    program       ;RESET vektorius

        Org     0x04          ;pertraukčių vektorius
        Goto    int
    ;*****************************************************************
***********

    ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++++
```

```
   ;+                  Makrokomandos
   ;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++

   INTSON: Macro
        Bsf    INTCON,GIE
        Endm


   INTSOFF Macro
        Bcf   INTCON,GIE
        Endm


   DQLOW:  Macro
        Bcf    PORTB,DSBIT    ; DQ BIT READY LO
        Bsf    STATUS,RP0
        Bcf    TRISB,DSBIT     ; DQ BIT NOW O/P
        Bcf    STATUS,RP0
        Endm


   DQHIZ:  Macro
        Bsf    STATUS,RP0
        Bsf    TRISB,DSBIT     ; DQ BIT NOW I/P
        Bcf    STATUS,RP0
        Endm


   PAUSE:  Macro   DLYF
        Movlw   (DLYF / 5) - 1
        Movwf   TMP0
        Call    DLY5N
        Endm

   jcode   macro   address       ;  perėjimas nurodytu adresu
        Call    code
        Andlw   0x01            ;  kodas geras?
        Btfsc   STATUS,Z
        Goto    address         ;  pereiti nurodytu adresu, jei geras kodas
        endm
   ;***************************************************************
```

```
int     Clrf    INTCON          ; pertrauktys nenaudojamos
        Retfie                  ; grižti į pagrindinę programą
;*****************************************************************
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
; Paprogramės
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
serial                          ; numeris
        Addwf    PCL
        Retlw    0x01               ;įtaiso identifikacijos numeris
        Retlw    0x54
        Retlw    0xA0
        Retlw    0x84               ;serijinis numeris
        Retlw    0x01
        Retlw    0x00
        Retlw    0x00
        Retlw    0x80               ;kontroline suma

;-----------------------------------------------------------------
; Numerio nuskaitymo paprogramė nuskaito numerį
;iš MULTIDROP įtaiso ir surašo į 8 baitų masyvą nuo adreso 11h
;-----------------------------------------------------------------
code
RESET   DQLOW
        PAUSE               .600
        INTSOFF
        DQHIZ
        PAUSE   65          ; 67 mks laukti atsakymo
        Nop
        Nop
        Movf    PORTB,W
        INTSON
        Andlw       1 << DSBIT
        Movwf       TMP1
        PAUSE       300
        Movf        TMP1,W          ; atsakymas i W

        Btfss       STATUS,Z        ; grįžti, jeigu raktas neatsiliepia
        Retlw       0x01            ; požymis, kad kodas nenuskaitytas

;       RESET pabaiga
;-----------------------------------------------------------------
```

```
        Movlw     0x08          ; nuskaitomų baitų skaičius
        Movwf     COUNT         ; į skaitiklį
        Movlw     0x11          ; pirmo  baito adresas
        Movwf     FSR           ; į duomenų nuorodos registrą
        Movlw     0x33          ; READ ROM komanda
WRITE   Movwf     TMP2          ; išsaugomi perduodami duomenys
        Movlw     8
        Movwf     TMP1            ; bitų skaitiklis
DSTXLP: INTSOFF
        DQLOW
        PAUSE     10
        Rrf       TMP2
        Btfsc     STATUS,C
        Bsf       PORTB,DSBIT     ; jei bitas=1, prievade aukštas lygis
        INTSON
        PAUSE     70
        DQHIZ
        Nop
        Decfsz    TMP1            ; kartoti ciklą, jei ne paskutinis bitas
        Goto      DSTXLP
;       WRITE pabaiga
gcloop
        READ  Movlw     8
        Movwf     TMP1
DSRXLP: INTSOFF                   ; baito skaitymas
        DQLOW
        PAUSE     10
        DQHIZ
        Nop
        Nop
        Movf      PORTB,W
        INTSON
        Andlw     1 << DSBIT      ; 0, jei prievade žemas lygis
        Addlw     .255
        Rrf       TMP2
        PAUSE     60
        Decfsz    TMP1
        Goto      DSRXLP
        Movf      TMP2,W
;       READ pabaiga
        Movwf     INDF            ; įrašom baitą i atmintį
        Incf      FSR             ; padidinam adresą
```

```
        Decfsz     COUNT          ; sumažinam baitų skaitiklį
        Goto       gcloop         ; nuskaityti sekanti baitą

;-----------------------------------------------------------------
;grąžina 0, jei kodas geras, 1 - jei blogas
;-----------------------------------------------------------------
;nuskaityto numerio sulyginimas su saugomu atmintyje
        Movlw      0x08           ; baitų skaičius
        Movwf      TMP1
        Clrf       TMP2           ; pavyzdinio kodo indeksas
        Movlw      0x11           ; nuskaityto kodo pirmo baito adresas
        Movwf      FSR            ; į duomenų nuorodos registrą
vrloop  Movf       TMP2,W             ; lenteles indeksas į W
        Call       serial         ; paimti pavyzdinio kodo baitą
        Subwf      INDF           ; palyginti su nuskaitytu kodu
        Btfss      STATUS,Z       ; pereiti į  sulyginimo paprogramę
        Retlw      0x01           ; nustatyti, ar kodas netinka
        Clrf       INDF
        Incf       TMP2,f
        Incf       FSR,f
        Decfsz     TMP1           ; kartoti ciklą, jei ne paskutinis baitas
        Goto       vrloop
        Retlw      0x00           ; 0 - jei kodas geras

DLY5N:  Nop
        Nop
        Decfsz  TMP0
        Goto    DLY5N
        Return

short_delay
TT3     Movlw   .255              ; vėlinimo laikas - 200 ms
        Movwf   TIME2
TT2     Movlw   .255
        Movwf   TIME1
TT1     Decfsz  TIME1
        Goto    TT1
        Decfsz  TIME2
        Goto    TT2
        Retlw   0x00

;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
;+                    pagrindinė programa                    +
;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

program                      ; pagrindinė programa
        Movlw   0x00
        Tris    PORTA
        Tris    PORTB        ; visi prievadai nustatomi kaip išėjimai
        Clrf    FSR
        Bcf     PORTB,LED    ;išjungiamas LED

ciklas  jcode   invert_led
        Goto    ciklas
invert_led
        Movf    PORTB,W      ; B prievadas  į W
        Xorlw   b'00000010'  ; invertuojamas LED bitas
        Movwf   PORTB
wait    jcode   wait         ; palaukti, kol bus atjungta TOUCH
MEMORY
        Call    short_delay
        jcode   wait
        Goto    ciklas

        end
```

Toliau pateikiamas ryšio per sąsają I^2C programos pavyzdys. Pateikiamos paprogramės duomenų rašymui ir skaitymui.

Paprogramių paskirtis:

I2C_START	-	mainų pradžia
I2C_STOP	-	mainų pabaiga
I2C_RB	-	baito nuskaitymas
I2C_WB	-	baito rašymas
I2C_ACK	-	SLAVE patvirtina, kad priėmė baitą
I2C_ACM	-	MASTER patvirtina, kad priėmė baitą

```
;paprogramės reikalingos darbui su I2C sąsaja
      list P=16c84      ;nurodomas procesoriaus tipas

;      Specialios paskirties registrų aprašymas
;.....................................
PORTA   =      0x05    ; A PRIEVADO REGISTRAS
PORTB   =      0x06    ; B PRIEVADO REGISTRAS
```

```
    TRISA   =       0x85      ; A PRIEVADO KRYPTIES REGISTRAS
    TRISB   =       0x86      ; B PRIEVADO KRYPTIES REGISTRAS
    STATUS  =       0x03    ; STATUS REGISTRAS
    FSR     =       0x04        ; IŠRINKIMO REGISTRAS FSR
    INDF    =       0x00        ; NETIESIOGINIO ADRESAVIMO REGISTRAS
    TMRO    =       0x01        ; 8 BITŲ TAIMERIS
    PCL     =       0x02        ; JAUNESNIEJI PC SKAITIKLIO BITAI
    PCLACH  =       0x0A    ; VYRESNIEJI PC SKAITIKLIO BITAI
    ;
    ;.....................................................
    ;STATUS REGISTRO BITAI
    ;.....................................................
    C       =       0x00    ;       CARRY FLAG
    DC      =       0x01    ;       DIGIT CARY FLAG
    Z       =       0x02    ;       ZERO FLAG
    PD      =       0x03    ;       POWER DOWN FLAG
    TO      =       0x04    ;       TIME OUT FROM WDT
    RP0     =       0x05    ;       RAM PAGE BIT 0
    RP1     =       0x06    ;       RAM PAGE BIT 1
    IRP     =       0x07    ;       RAM BANK
    ;
    ;.....................................................................
    ;EEPROM REGISTRAI
    ;.....................................................................
    EEDATA  =       0x08    ;EEPROM DUOMENŲ REGSITRAS
    EEADR   =       0x09    ;EEPROM ADRESO REGISTRAS
    EECON1  =       0x08    ;EEPROM SKAITYMO RAŠYMO KONTROLĖS
REGISTRAS, BANK1, RP0=1
    EECON2  =       0x09    ;       -"-
    ;.....................................................................
    ;EECON1 REGISTRO BITAI
    ;.....................................................................
    EERD    =       0x00        ; SKAITYMO BITAS
    EEWR    =       0x01        ; RAŠYMO BITAS
    EEWREN  =       0x02    ; LEIDIMO RAŠYTI Į EEPROM ATMINTĮ
LEIDIMO BITAS
    EEWRERR =       0x03    ; RAŠYMO KLAIDOS BITAS
    EEIF    =       0x04        ;RAŠYMO PABAIGIMO BITAS
    ;
    ;.....................................................................
    ;OPCIJŲ REGISTRAS
    ;.....................................................................
```

```
    OPT_REG =       0x81    ; FUNKCIJŲ REGISTRAS
    PS0     =       0x00
    PS1     =       0x01
    PS2     =       0x02
    PSA     =       0x03        ; DALIKLĮ PRIJUNGIA PRIE WDT (KAI 1)
ARBA PRIE TMRO (KAI 0)
    T0SE    =       0x04        ; IŠORINIO TAIMERIO SIGNALO FRONTAS
0-TEIGIAMAS, 1-NEIGIAMAS
    T0CS    =       0x05        ; SIGNALO ŠALTINIS TAIMERIUI 0-VIDINIS,
1-ISORINIS
    INTEDG  =       0x06        ; PERTRAUKTIES FRONTO BITAS
    RBPU    =       0x07        ; AKTYVIOS APKROVOS PAJUNGIMAS
PRIE PORTO B 0-PAJUNGTA, 1-ATJUNGTA
    OPT_INI =       0x01        ; FUNKCIJŲ REGISTRO INICIALIZAVIMAS
    ;.....................................................................
    ;PERTRAUKČIŲ APTARNAVIMO REGISTRAS
    ;.....................................................................
    INTCON  =       0x0B    ; PERTRAUKČIŲ REŽIMO REGISTRAS
    RBIF    =       0x00        ; PERTRAUKTIES PER PORTĄ B (RB<7:4>)
BITAS
    INTF    =       0x01        ; PERTRAUKTIES PER RB0/INT BITAS
    RTIF    =       0x02        ; PERTRAUKTIES PERSIPILDŽIUS
TAIMERIUI BITAS
    RBIE    =       0x03        ; PERTRAUKTIES PER PORTA B LEIDIMAS
    INTE    =       0x04        ; PERTRAUKTIES PER RB0/INT LEIDIMAS
    RTIE    =       0X05        ; PERTRAUKTIES TAIMERIU LEIDIMAS
    EEIE    =       0x06        ; PERTRAUKTIES IRAŠIUS DUOMENIS I
EEPROM LEIDIMAS
    GIE     =       0x07        ; VISŲ RŪŠIŲ PERTRAUKČIŲ LEIDIMAS


    SDA =           0x00    ; duomenų bitas prievade
    SCL =           0x01    ; taktų bitas prievade

    TMP =           0x0C
    TMP1            =       0x0D
    HH  =           0x0E
    MM  =           0x0F
    SS  =           0x10
    COUNT           =       0x11
```

```
      Org    0x00          ;reset vektorius
      Goto   program

      Org    0x04
      Goto   int           ;pertraukčių vektorius

int   Clrf   INTCON        ;taip daroma, kai pertrauktys nenaudojamos
      Retfie

;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
;              I2C  C O N T R O L
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++

APCFR  =     B'xxxxxxxx'    ; I2C slave įrenginio adresas skaitymui
APCFW  =     B'xxxxxxxx'    ; I2C slave įrenginio adresas rašymui
;
W4T                         ;1 mks vėlinimas (kai Fclk=4MHz)
      Nop
      Nop
      Nop
      Return
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
;+             Makrokomandų aprašymas                  +
;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
SDA_R  MACRO               ; SDA bito nustatymas skaitymui
      Bsf    STATUS,RP0    ; SDA --> nuskaitymui
      Bsf    PORTB,SDA
      Bcf    STATUS,RP0
      ENDM

SDA_W  MACRO               ; SDA bito nustatymas rašymui
      Bsf    STATUS,RP0    ; SDA --> rašymui
      Bcf    PORTB,SDA
      Bcf    STATUS,RP0
      ENDM

I2C_START
      Bcf    PORTB,SDA
      Call   W4T
      Bcf    PORTB,SCL
      Call   W4T
```

```
      Retrurn
I2C_STOP
      Bcf    PORTB,SDA
      Call   W4T
      Bsf    PORTB,SCL
      Call   W4T
      Bsf    PORTB,SDA
      Call   W4T
      Return
I2C_ACK                 ;ACK iš SLAVE (baito nuskaitymo patvirtinimas)
      SDA_R
      Bsf    PORTB,SCL
      Call   W4T
      Bcf    PORTB,SCL
      Call   W4T
      SDA_W
      Return
I2C_ACM                 ;ACK iš master
      SDA_W
      Bsf    PORTB,SCL
      Bcf    PORTB,SDA
      Call   W4T
      Bcf    PORTB,SCL
      Bsf    PORTB,SDA
      Call   W4T
      SDA_R
      Return


;     Baito rašymas į IIC iš W registro
I2C_WB  Movwf   TMP
      Movlw   8         ;nuskaitomų bitų skaičius
      Movwf   TMP1
_WB1    Rlf     TMP
      Btfsc   STATUS,C      ;
      Goto    _WB3          ;
      Bcf     PORTB,SDA     ;
      Goto    _WB2          ;
_WB3    Bsf     PORTB,SDA
_WB2    Bsf     PORTB,SCL     ;taktinis impulsas
      Call    W4T
      Bcf     PORTB,SCL
```

```
        Decfsz  TMP1            ;sumažinti bitų skaitiklį
        Goto    _WB1            ;kartoti ciklą, jei ne paskutinis bitas
        Return                  ;grįžti į pagrindinę programą

;       Baito nuskaitymas iš IIC
I2C_RB  SDA_R                   ;SDA - skaitymui
        Clrf    TMP             ;
        Movlw   8               ;bitų skaičius į atmintį
        Movwf   TMP1            ;
_RB1    Bsf     PORTB,SCL       ;taktinis impulsas
        Rlf     TMP             ;
        Movlw   0xFE            ;
        Andwf   TMP             ;
        Btfsc   PORTB,SDA       ;nuskaitomas bitas
        Bsf     TMP,0           ;
        Bcf     PORTB,SCL       ;
        Call    W4T             ;palaukti
        Decfsz  TMP1            ;sumažinamas bitų skaitiklis
        Goto    _RB1            ;kartoti ciklą, jei ne paskutinis bitas
        SDA_W
        Movf    TMP, W          ;baitas  į W
        Return                  ;grįžti į pagrindinę programą

;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;Tipinė komandų seka, dirbant su IIC sąsaja:
;Baito rašymas:
          Call    I2C_START             ;mainų pradžia
        I2C_WRITE       APCFW           ;SLAVE adresas rašymui
        Call        I2C_ACK             ;patvirtinimas, kad baitas priimtas
          I2C_WRITE         ADR  ;SLAVE ląstelės adresas
          I2C_ACK                       ;
        I2C_WRITE       BYTE            ;baito rašymas
        Call        I2C_ACK             ;
        Call        I2C_STOP            ;mainų pabaiga
;Baito skaitymas:
        Call        I2C_START           ;
        I2C_WRITE       APCFR           ;
        Call        I2C_ACK             ;
        I2C_WRITE       ADR             ;adresas
        I2C_ACK                         ;
        Call        I2C_RB              ;nuskaityti baitą
```

```
        Call    I2C_ACM             ;master patvirtina, kad baitas
priimtas
        Call    I2C_STOP            ;mainų pabaiga
    ;Masyvo rašymas i I2C atmintį:
        Movlw   SIZE                ;baitų skaičius
        Movwf   COUNT               ;į skaitiklį
        Movlw   STATRT              ;masyvo pradžios adresas
mikrovaldiklio atmintyje
        Movwf   FSR                 ;į duomenų nuorodos registrą
        Call    I2C_START           ;mainų pradžia
        I2C_WRITE       APCFW       ;adresas rašymui
        I2C_ACK                     ;
        I2C_WRITE       ADR         ;
        I2C_ACK                     ;
    w_loop      Movf    INDF,W      ;baitas - į W
        Call    I2C_WB              ;įrašyti
        Call    I2C_ACK             ;
        Incf    FSR                 ;sekančio baito adresas
        Decfsz  COUNT               ;ar ne paskutinis baitas?
        Goto    w_loop              ;jei ne, rašyti sekantį
        Call    I2C_STOP            ;rašymo pabaiga
```

Daugiau informacijos apie PIC mikrovaldiklius, taip pat jų praktinio panaudojimo pavyzdžius galima rasti firmos Arizona Microchip interneto puslapyje [37].

11. LITERATŪRA

Pagrindinė:

1. William Kleitz. Microprocessor and Microcomputer Fundamentals. Prentice Hall ,1998,320 p.
2. James W. Stewart. The 8051 Microcontroller . Prentice Hall ,1999,368 p.
3. Kennet J. Short. Embedded Microprocessor Systems Design. Prentice Hall, 1998,352 p.
4. John B.Peatman. Design with PIC Microcontrollers . Prentice Hall, 1998,352 p.
5. Thomas L.Floyd. Digital Fundamentals. Prentice Hall ,1996,871p.
6. Raymond J.A. Buhr, Donald L.Bailey. Introduction to Real Time Systems. Prentice Hall , 1999,430p.
7. Jan Axelson. The microcontroller Idea Book, Prentice Hall ,1999.
8. S. Yerala. Programming and Interfacing the 8051. Prentice Hall ,1999.

Papildoma:

9. М. Фридмен, Л. Ивенс. Проектирование систем с микрокомпъютерами – М.:, Мир, 1986.
10. В. П. Корячко. Микропроцессоры и микроЭВМ в радиоэлектронных средствах. – М. : Мир, 1990.
11. Программирование микропроцессорных систем. – Под ред. В. Ф. Шанъгина. – М.: Высшая школа, 1990.
12. В. В. Сташин и др. Проектирование цифровых устройств на однокристалъных микроконтроллерах. – М.: Мир, 1990.
13. Под. ред. Э.В. Евреинова. Цифровая и вычислительная техника – М.: Радио и связь, 1991.
14. М. Рафикузаман. Микропроцессоры и машинное проектирование микропроцессорных систем (в двух книгах). – М.: Мир, 1988.
15. Однокристалъные микроЭВМ. Справочник. – М.: Бином, 1994.
16. В. В. Баранов и др. Полупроводниковые БИС запоминающих устройств. Справочник. – М.: Радио и связъ, 1986.
17. Г. Я. Мирский. Микопроцессоры в измерителъных приборах. М.: Радио и связъ, 1984.
18. V. Bulovas. Mikroprocesoriai. V.: Mokslas, 1989.
19. V.Deksnys, V.Jastramskas Mikroprocesoriai ir jų taikymas radioelektronikoje. 1 dalis.- K.: Technologija, 1994.
20. MCS-85 Users manual. Intel Corporation .1978.
21. Texas Instruments. Digital design seminar ,1999.
22. http:\\www.ti.com
23. http:\\www.atmel.com
24. http:\\www.intel.com
25. http:\\www.dalsemi.com
26. 80C51-based 8-bit microcontrollers. Philips Semiconductors,1994.
27. http:\\www.8052.com
28. http:\\www.analog.com/dsp.
29. http:\\www.amd.com
30. http:\\www.microchip.com
31. http:\\www.philips.com
32. http:\\www.liteon.com
33. http:\\www.vantis.com
34. http:\\www.st.com
35. Joseph Raynus. Software Process Improvement with CMM.Artech House. 1999. 212 p.
36. John W. Horch. Practical Guide to Software Quality Management. Artech House .1999. 263 p.
37. http:\\www.microchip.com

Turinys