ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

С.Т. Тынымбаев

КОМПЬЮТЕРЛІК ЖҮЙЕЛЕР АРХИТЕКТУРАСЫ

Ақпараттық-коммуникациялық технологиялар бойынша Республикалық оқу-әдістемелік кеңесі оқулық ретінде ұсынған

Алматы 2019

КІРІСПЕ

Компьютерлік техниканың әртүрлі аппараттық қырларына арналған пәндерінің ішінде «Компьютерлік жүйелер архитектурасы» берілетін ғылыми мағұлматтардың түйіндемесі бола отырып, келесі компьютерлік желілер бағытына арналған әртүрлі пәндерге негіз болады.

«Компьютерлік жүйелер архитектурасы» пәнінің мақсаты заманауи есептеу машиналары мен есептеу жүйелерінің негіздерін және олардың жұмыс жасау принциптерін оқыту. Ақпараттық технологиялар бағытында дайындалатын мамандықтар үшін қаралып отырған пәннің, мазмұны төменгідей болып келеді:

- Есептеу техникасының даму кезеңдері;
- Есептеу машиналарын құрудың негізгі принциптері, олардың құрылымдық сұлбалары мен көрсеткішері;
- Операндтарды көрсету тәсілдері, команда типтері мен олардың пішімдері;
- ЭЕМ жады құрылғыларын ұйымдастыру;
- Операциялық және басқару құрылғыларын ұйымдастыру;
- Шиналарды ұйымдастыру;
- Енгізу-шығару құрылғыларын ұйымдастыру;
- Әртүрлі класты ЭЕМ ұйымдастыру сәулеті;
- Параллель есептеулер мен параллель жүйелер;
- Көпмашиналы және көппроцессорлы есептеу жүйелері.

Оқулықта осы аталған тақырыптар толығымен қамтылған, оқулық 16 тараудан тұрады.

Бірінші тарауда есептеу техникасының даму кезеңдері туралы деректер берілген.

Екінші тарауда Фон-Нейман тұжырымдамасының негізгі принциптері талқыланып, Фон-Нейман машинасының жұмыс жасау принциптері қаралады және есептеу машиналарының жұмысын бейнелеуге мүмкіндік беретін шағын операциялар мен шағын бағдарламалар тілі қаралған.

Есептеу машиналары мен жүйелерінің құрылымдық сұлбалары мен олардың негізгі көрсеткіштері үшінші тарауда келтірілген.

Төртінші тарауда машиналарда әртүрлі операндтарды (екілік, ондық, символдық) көрсету тәсілдері қаралған.

Есептеу машиналарының арифметикалық және логикалық командалары, енгізу-шығару командалары, командалар ағымын басқару және көппроцессорлы есептеу жүйелерінің командалары қаралады. Командалар пішімі мен мекендеу тәсілдері де бесінші тарауда қамтылған.

Алтыншы тарауда қатаң және магистралды операциялық құрылғылардың ерекшеліктері қамтылған. Қосу, көбейту, бөлу операциясын орындайтын, бекітілген және жылжымалы үтірлі операциялық құрылғылардың құрылымдық

сұлбалары және көбейту, бөлу операцияларын жеделдететін аппарттық (матрицалық, бұтақ тәрізді, конвейерлік) тәсілдері қаралады.

Жетінші тарауда жады құрылғылары қаралады. Олардың сипаттамалары мен жіктелінуі, негізгі құрылымдары, жады шағын сұлбасының құрылымы, негізгі жадыны ұйымдастыру тәсілдері, стек, КЭШ, ассоциативтік жадылардың жұмыс жасау принциптері, ауани жадыны ұйымдастыру, жедел жадыны қорғау тәсілдері қаралады. Магниттік, оптикалық дисктер негізінде құрылған жады құрылғыларының жұмыс жасау принциптері және магниттік таспа негізінде құрылған сыйымдылығы жоғары есте сақтау құрылғылары қаралады.

Сегізінші тарау есептеу машиналарының басқару құрылғысына арналған. Тарауда басқару құрылғысының атқаратын қызметі осыған сәйкес құрылымдық сұлбасы, микробағдарламалық автоматтардың түрлері, бағдарламаны үзу жүйесі және векторлық үзу контроллерін ұйымдастыру тәсілі қаралады.

Тоғызыншы тарауда шиналардың төрелік механизмдері, хаттамалары және түрлері келтіріледі.

Енгізу-шығару жүйелері туралы деректер оныншы тарауда келтірілген. Олардың орталық процессорларға қосылу тәсілдері, енгізу-шығару модулінің құрылымы, енгізу-шығару операцияларын ұйымдастыру тәсілдері және енгізу-шығару арналары мен процессорлары туралы деректер беріледі.

Он бірінші тарауда қазіргі есептеу машиналарында процессорларды ұйымдастыру тенденциялары қаралады. Командалар конвейерлері, суперскалярлық процессорлар, CISC және RISC процессорлары, VLIW процессорлары және көпядролы процесорларды ұйымдастыру түрлері келтіріледі.

Он екінші тарауда компьютерлерде параллель өңдеу деңгейлері, параллель өңдеу өлшемдері мен заңдылықтары, дерек және командалар ағындарының тұрғысынан есептеу жүйелерінің Флинни жіктемесі қаралады.

Он үшінші тарауда есептеу жүйелерінің ішкі байланыстарын ұйымдастыру тәсілдері, параметрлері, жіктелінуі, деректерді маршрутизациялау функциялары, статикалық және динамикалық топологиялар қаралады.

Он төртінші тарау есептеу жүйелерінде жады құрылғыларын ұйымдастыру: ортақ жадылы, үлестірілген ортақ жадылы және үлестірілген жадылы архитектура түрлері қаралады.

Он бесінші тарауда жеке команда ағыны және көптік деректер ағынымен жұмыс жасайтын есептеу жүйелері қаралады. Оларда векторлық, матрицалық, ассоциативтік және систоликалық құрылымды есептеу жүйелерінің жұмыс жасау принциптері талданады.

Он алтыншы тарауда көптік команда және көптік мәліметтер ағынымен параллель жұмыс жасайтын симметриялы көппроцессорлар, параллель векторлы, жаппай параллель өңдеу, үлестірілген жадылы кластерлік есептеу жүйелері және транспьютерлер негізінде құрылған есептеу жүйелері қаралады.

Кітаптың әр тарауында бақылау сұрақтары келтірілген. Оқулық «Есептеу техникасы және бағдарламалық қамтамасыз ету», «Ақпараттық қауіпсіздік жүйелері» білім беру бағдарламалары бойынша оқитын бакалаврлар мен магистранттарға арналған.

1. ЕСЕПТЕУ ТЕХНИКАСЫНЫҢ ДАМУ КЕЗЕҢДЕРІ

Есептеу техникасының даму кезеңдерін буындарға бөліп қарау кеңінен тараған.

Есептеу техникасының буындары деп белгілі бір уақыт аралығындағы есептеу машиналарының даму кезеңімен белгіленетін құрылғыларды өндіру технологиялары, функционалдық мүмкіндіктер деңгейі және бағдарламалық қамтамалары шамалас машиналар тобын айтуға болады. Есептеу машиналарының даму кезеңдерін әдетте Джон Фон Нейман машинасымен байланыстырады, өйткені Фон Нейман ұсынған бағдарламаларын машина жадында сақтау тұжырымдамасы қазіргі кезде кез-келген есептеу машиналары мен жүйелерінде қолданыста.

Осыған байланысты ЕТ (есептеу техникасы) даму тарихын үш кезеңге бөліп қарауға болады:

- Фон Нейманға дейінгі кезең
- Фон Нейман есептеу машиналары мен жүйелері
- Фон Нейман есептеу машинасынан кейінгі кезең

Есептеу машиналарын буынға бөлуді технологияның өзгеруімен де байланыстырады. Нөлдік буынға механикалық немесе электромеханикалық машиналарды жатқызса, одан кейінгі буындарға есептеу техникасының элемент негіздері – электронды шамдар, жартылай өткізгіш аспаптар, интегралдық сұлбалар (кіші, үлкен, өте үлкен және ультра үлкен) интегралдық микросұлбалар жатады.

1.1. Нөлдік буын (1642-1945)

Нөлдік буынға механикалық және электромеханикалық компьютерлер жатады. Оған мысал Блез Паскальдің (1623-1662) механикалық машинасын айтуға болады. 1642 жылы жасалынған машина тек екі (қосу және алу) операцияны орындауға мүмкіндік береді. 1672 жылы немістің ұлы математигі Готфрид Вильгельм Лейбниц (1646-1726) жасаған механикалық машинасы қосу және алу операциясынан басқа көбейту және бөлу операцияларын орындауға мүмкіндік берді.

1836 жылы Кэмбридж университетінің профессоры математик Чарльз Бэббидж (1792-1872) «аналитикалық машинасының» жобасын ұсынды. Аналитикалық машина төрт бөліктен: жады құрылғысынан, есептеуіш құрылғысынан, енгізу құрылғысынан (перфокартадан оқу), деректерді шығару

құрылғысынан (перфоратор) тұрады. Жады құрылғысы 50 разрядты 1000 ондық сандарды сақтауға арналған. Ұсынылған аналитикалық машинаны жасау үшін дәлдігі өте жоғары, мыңдаған тісті доңғалақтар керек болды. Ондай доңғалақтарды XIX ғасырда жасау мүмкін емес еді. Бірақ Бэббидж идеясын бүгінгі машиналарда кездестіруге болады. 1938 жылы неміс инженері Конрад Цузе (1910-1995) бағдарламаланатын механикалық Z есептеуішті жасап шықты. Жады құрылғысы 1000 бит құрады. Цузенің құрған ЕМ кейінгі кезде дүние жүзіндегі бірінші жасалынған компьютер деп жүр. 1940 жылы Цузе Z2 машинасын релейлік логикада жасады. Ал 1941 жылы Z3 электромеханикалық бағдарламалық есептеуішті жасады. Есептеуіш 2100 электромеханикалық релелерден тұрды. Бағдарлама перфолентада сақталды. Жады сыйымдылығы 64, 22 биттік сөзден тұрды. Көбейту операцясы 3-5 с. 1943 жылы Говард Айкеннің басшылығымен бағдарламамен басқарылатын бірінші есептеуіш Mark машинасын жасады. Машина бірнеше есептеуіштерден тұрды. Әрбір есептеуіш ортақ есептің бір бөлігін өңдеді. Басқару бір құрылғы арқылы жүргізіледі. Командалар қағаз перфоленталардан оқылды. Есептеуіш 23-разрядты сандармен жұмыс жасады.

1945 жылы Цузе Z4 есептеуішін аяқтады. Z4 есетеуішінің архитектурасы қазіргі машиналар архитектурасына өте ұқсас. Жады құрылғысы мен процессор жеке құрылғы ретінде құрылған. Процессор төрт операцияны орындады, оның үстіне ол жылжымалы үтірлі сандармен де амалдар орындай алды және түбір табу операциясын орындауға мүмкіндік берді. Бағдарлама перфолентада сақталып олар тізбекпен оқылып отырды.

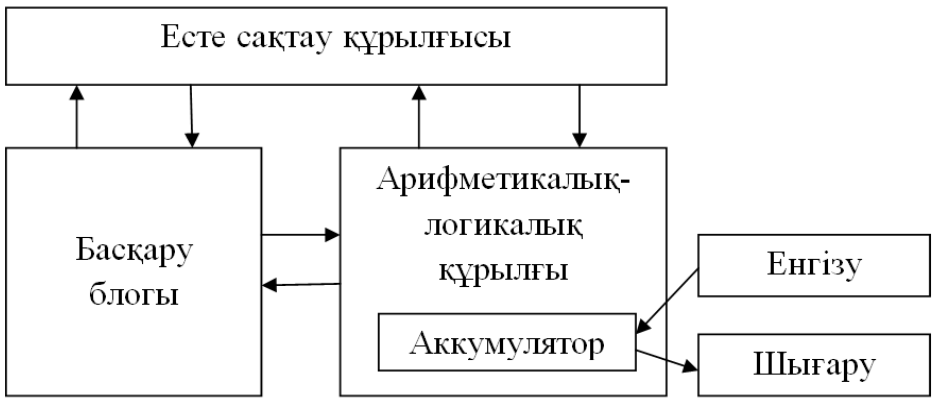
1.2. Бірінші буын (1937-1953)

Бірінші буынды машиналардың элемент негізі электронды-вакуумды шамдар болды. Өйткені мұндай шамдардың жылдамдығы электромеханикалық релелерге қарағанда өте жоғары (мыңдаған есе). Бірінші электронды есептеу машинасына бірнеше есептеуіштер таласа алады. Олардың ішінде бірінші ABC (Atanasof-Berry Computer) арнайы калькуляторын атауға болады. ABC 1939 жылдан 1942 жылдар аралығында профессор Джон Атанасовтың аспиранты К.Берримен жасаған калькулятор көмегімен сызықтық теңдеулер жүйесін шешу үшін қолданылады. ABC жады 50 битті 50 сөздерді сақтауға мүмкіндік берді. Жады элементтері ретінде регенерациялау тізбегі бар конденсаторлар қолданылды. Сырт жады ретінде перфокарталар іске асырылды.

Бірінші ЭВМ-ге таласушы ретінде 1943 жылы Англияда жасалынған Colossus есептеуіші жатады. Оны Томми Флауэрс профессор Макс Ньюменнің

жетекшілігімен жасаған. Colossus-тың көмегімен немістің шифрлаушы машинасы «Лоренц Шлюссель-цузат-40» кодаларын шешеуге қолданды.

Біріншілікке кандидат үшінші есептеуішке жалпы мақсатты бағдарламаланатын электронды калькулятор ENIAC (Electronic Numerical Integrater and Computer – электронды цифрлық интегратор және компьютер) жатады. Калькулятродың идеясының авторы Пенсильван университетінің қызметкері Джон Мочли (1907-1970). Калькулятор 1946 жылы Преспер Эккертпен бірге іске қосылған. ENIAC калькуляторы сутегі бомбасын жасау бағдарламасында кеңінен қолданылды. ENIAC салмағы 30 тонна, құрамы 18000 радиошамдардан тұрды. Секундына 5000 қосу және 360 көбейту амалдарын ондық санақ жүйесінде орындайды. ENIAC жобасына кеңесші ретінде Джон Фон-Нейман қатысқан болатын. Джон Фон-Нейман сонымен қатар, Принстон орталығында өзінің IAS (Immediate Address Storadge) деген машинасын жобалауға кіріскен болатын. EDVAC және IAS машиналарының жобаларын жасау үстінде Фон-Нейманның ойға түйген идеяларына мыналар жатады: есепті шығару алдында бағдарлама және деректер цифрлық түрде компьютердің ішкі жадында сақталу керек және ENIAC машинасында қолданылатын ондық арифметика (әрбір ондық разрядты көрсету үшін он электронды шам қолданылады) бинарлы арифметикамен алмастырылуы керек. Жоғарыдағы аталған идеяларды іске асыратын жоба қазіргі кезде *Фон-Нейман есептеу машиналары* деп аталады. Бағдарламалары ішкі жадыда сақталатын бірінші машина қатарына EDSAC машинасы жатады. Фон-Нейман машинасының архитектурасы 1.1-суретте келтірілген.



1.1-сурет. Фон Нейман машинасының сұлбасы

Фон Нейман машинасы 5 бөліктен: машина жадынан, арифметикалық және логикалық құрылғысынан (АЛҚ), басқару құрылғысынан, енгізу-шығару құрылғысынан тұрады. Жады 4096 сөзден, ал сөз әрқайсысы 20 биттен тұратын екі команда немесе 40 разрядты бүтін сан. Әр команданың 8 биті команда типін көрсетсе, қалған 12 биттері жадының 4096 сөздерінің бірін көрсете алды.

Арифметикалық-логикалық құрылғының құрамындағы 40 битті регистр-аккумулятор арқылы деректер компьютер жадымен және енгізу-шығару құрылғыларымен алмастыру жүргізіледі. Джон Фон Нейманның IAS және EDSAC машиналары компьютерлік техниканың ары қарай дамуына зор ықпал етті.

1953 жылы IBM фирмасы IBM-701 машинасын сериялы түде нарыққа шығарды.

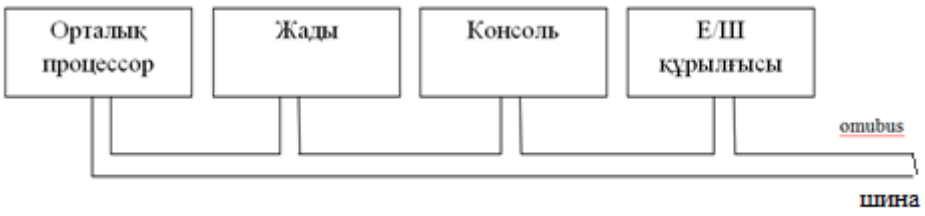
1951 жылы С.А.Лебедевтің басқаруымен МЭСМ (кіші электрондық есептеу машинасы) жасалынды. Бұл СССР-да және Европада жасалған бірінші электронды машина болды.

1952 жылы (И.С.Брук, Н.Я.Матюхин, А.Б.Залкинд) М-1 машинасы жасалынды.

1953 жылы Европадағы ең жылдам жұмыс жасайтын есептеу машинасы БЭСМ (С.А.Лебедев) пайдалануға берілді. Оның жылдамдығы 8000-10000 операция/секунд.

1.3. Екінші буын (1954-1962)

Екінші буынды машиналардың пайда болуы жартылай өткізгішті элементтердің (транзисторлар мен диодтардың) пайда болуымен байланысты. Транзисторларды Bell Laboratoriec зертханасының мамандары Джон Бардин, Уолтер Браттейн, Уильям Шокли ойлап тапқан. Сол үшін олар 1956 жылы Нобель сыйлығына ие болған. Транзисторлар негізінде құрылған бірінші компьютерлерге МТИ зертханасында жасалынған ТХ-0 (транзисторлық экспериментарлық компьютерлер) ТХ-0, ТХ-2 және Bell Labs зертханаларында жасалынған TRADIC (Transistor Digital Computer- транзисторлық цифрлық компьютер) компьютерлері жатады. 1961 жылы DEC (Digital Equipment Corporation - цифрлық ақпаратты өндіретін корпорация) фирмасы PDP-1 машинасын шығарды. 64 Кбайт жадылы компьютер құрамында 512*512 пиксельді дисплей қосылған болатын. DEC фирмасы PDP-1 компьютерінен кейін 12 биттік PDP-8 мини компьютерін шығара бастады. Оның құны басқа компьютерлерге қарағанда төмен болды. PDP-8 компьютеріне енгізілген негізгі жаңалық - компьютердің барлық құрылғылары бір шина арқылы алмасуға мүмкіндік алды (1.2-сурет).



1.2-сурет. PDP-8 компьютерінің шинасы

Екінші буынды машиналардың ерекшелігі ретінде жартылай өткізгіштік элементтерімен қатар жады құрылғыларын құруға магниттік өзекшелердің кеңінен қолдануын атауға болады. Магниттік өзекшелер негізінде құрылған жадының негізгі артықшылығы оларға сақталған деректерге еркін қатынас құра аламыз. Яғни кез-келген уақытта деректің кез-келген элементіне олардан деректерді оқу үшін немесе оларға деректерді жазу үшін қатынас құра аламыз. Екінші буынды машиналардың құрамына жылжымалы үтірлі сандарды өңдейтін аппараттық блоктар енгізілді. Есептеу машинасының құрамына бөлек енгізу-шығару блоктары енгізілді. Мұндай блоктар процессор жұмысын енгізу-шығару операцияларынан босатып, оның өнімділігін өсіруге мүмкіндік берді. Екінші буынды кеңестік машиналарға Урал-4, Урал-11, БЭСМ-2, М-40, Минск-2, Минск-32, «Днепр» машиналары жатады.

1.4. Үшінші буын (1963-1972)

Үшінші буынды машиналар интегралды микросұлбалар негізінде құрылды. Жады құрылғыларында магниттік өзекшелердің орнына жартылай өткізгіштер қолдана бастады.

Процессорлар негізінен микробағдарламалық басқару құрылғылары арқылы құрылды. Командалар мен операциялар конвейерлік тәсілмен орындала бастады. 1964 жылы Сеймур Крей CDC-6600 жаңа есептеу жүйесін құрды. Жүйе он тәуелсіз функционалды блоктардан және 32 тәуелсіз жады модульдерінен тұрды. Жүйе жылдамдығы -1 Mflops (жылжымалы үтірлі сандармен секундына миллион операция жасалған), 1969 жылы Крей жасаған CDC-7600 жүйесінің жылдамдығы 10 Mflops-қа жетті. CDC-7600 жүйесі - бірінші конвейерлік есептеу жүйесі деп аталды. Үшінші буынды машинаға IBM-360, IBM-370 машиналары, SOLOMON, ILIAC-IV жүйелері, конвейерлік-векторлық жүйелер TI-ASC және STAR-100 жүйелері жатады. IBM-360 жүйелерінің негізінде Кеңес одағы мен социалистік елдерде ЕС-1010/1060 іске қосылды.

Кеңес одағында жасалған үшінші буынды машиналарына БЭСМ-6 (С.А.Лебедев) машинасы жатады. Оның жылдамдығы секундына 1 миллион операция. Одан басқа М-220, М-222,МИР-1 машиналарын жатқызуға болады.

1.5. Төртінші буын (1972-1984)

Төртінші буынды машиналар үлкен интегралды сұлбалар (Large-scale integration,LSI) және өте үлкен интегралды сұлбалар (Very large-scale integration,VLSI) негізінде құрылған. Сұлбалардың бірінші түрінің кристалында 1000 транзистор орналасса, екінші түрінде 100000 транзистор орналасады. Транзисторлардың мұндай тығыздығы бір микросұлбада компьютердің барлық

негізгі құрылғыларын орналастыруға мүмкіндік берді. Әртүрлі микропроцессорлар мен микро-ЭВМ-дер пайда болды. Олардың қатарына командалары ықшамдалған есептеу машиналары (RISC, Reduced Instruction Set Computer) жатады. Мұндай машиналар процессор командаларының орындалу жылдамдығын арттыруға алып келетін қарапайым сұлбалар арқылы құрылған. Күрделі командалар ішкі бағдарламалар арқылы жылдам орындалатын қарапайым командалар арқылы орындалады.

Өнімділігі жоғары есептеу жүйелері көбінесе векторлық процессор негізінде дамуын жалғастырды.

С бағдарламалау тілі пайда болды. UNIX операциялық жүйесін С тілінде жазу арқылы UNIX операциялық жүйесін басқа есептеу машинасында қолдануға мүмкіндік берді.

1.6. Бесінші буын (1984-1990)

Бесінші буында есептеу жүйелері көптеген машиналардан немесе процессорлардан тұрады. Есеп шығару үстінде әрбір пайдаланушыға жеке процессор бөлінеді және керек болған жағдайда әрбір пайдаланушыларға қосымша процессорлар ресурстарын пайдалануға мүмкіндік берді.

Бесінші буын жүйелері екі түрлі архитектуралық ерекшеліктерімен белгіленеді. Біріншісінде барлық машиналар бір жады арқылы жұмыс жасайды. Екіншісінде әрбір машина өзінің жеке жады арқылы жұмыс жасайды.

Есептеу жүйесінің бірінші түріне Sequent Balance 800 жүйесі жатады. Мұнда сыйымдылығы үлкен негізгі жады 20 процессорлармен ортақ бөлініп пайдаланылады.

Бесінші буынды есептеу жүйесінің екінші бағытында әрбір процессор өзінің жады модульдерімен қамтамасыз етілген. Процессорлар бір-бірімен алмасу желілері арқылы байланысқан. Мысал ретінде Intel фирмасының iPSC жүйесін келтіруге болады. Жүйе 128 процессорларды біріктіре алады.

Бесінші буынды есептеу жүйесінің үшінші бағытына бірнеше қарапайым процессорлардан тұратын орталық құрылғы арқылы басқарылатын, әрқайсысы өз деректермен бірдей операция орындайтын жүйелер жатады. Мұндай есептеу класына Thinking Machines Inc. фирмасының Connection Machine жүйесі және Mas Par Inc. фирмасының MP-1 жүйесі жатады.

1.7. Алтыншы буын (1990-....)

Есептеу машиналарының бірінші төрт буындары бір-бірінен негізінен элемент негіздерімен өзгешеленеді. Бесінші және алтыншы буын есептеу жүйелерінің элемент негіздері бірдей болғандықтан олар бір-бірінен тек архитектуралық өзгешеліктерімен ерекшеленеді. Алтыншы буынды есептеу жүйелері параллель процессорлар массиві арқылы құрылады. Мұндай жүйелер

көптеген (бірнеше мың) өзара байланысқан автономды есептеу машиналарынан тұрады. Мұндай жүйелердің есептеу қуаты суперкомпьютерлермен бәсекелесе алады.

Алтыншы буынды есептеу жүйесінің екінші өзгешелелігі – жұмыс стансаларының деңгейі. Жұмыс стансаларының процессорларында RISC архитектурасы, конвейерлеу және параллель өңдеу тәсілі кеңінен қолданылады. Кейбір жұмыс стансаларының өнімділігі төртінші буынды суперкомпьютерлерімен тең түседі.

Бақылау сұрақтары.

1. Есептеу машиналарының даму кезеңдерін неге Фон-Нейман машинасымен байланыстырады?
2. Есептеу машиналарының буынға бөлінуі қандай белгілермен анықталады?
3. Бағдарламамен басқарылатын бірінші электромеханикалық машина қай жылы шықты және кім жасады?
4. Бірінші электрондық машиналарға қандай есептеу машиналарын жатқызуға болады?
5. Бағдарламалары ішкі жадыда сақталатын бірінші есептеу машинасы қатарына қандай машиналарды жатқызуға болады?
6. Бірінші ортақ шиналы машинаға қандай машина жатады?
7. Қенес одағы кезінде ең көп таралған әмбебап үшінші буынды машиналарға қандай есептеу машиналарының модельдері жатады?
8. Төртінші буынды машиналардың элемент негізі ретінде қандай интегралдық сұлбалар қолданылды?
9. Бесінші және алтыншы буынды машиналар қандай архитектуралық ерекшеліктерімен анықталады?

2. БАҒДАРЛАМАЛАРДЫ МАШИНА ЖАДЫНДА САҚТАУ ТҰЖЫРЫМДАМАСЫ

Есептеу машинасы деп берілген алгоритм арқылы дискретті деректерді автоматты түрде өңдеуге арналған техникалық жабдықтар жиынын айтамыз.

Алгоритм ұғымы математика мен есептеу техникасының іргелі ұғымдарының бірі. Халықаралық стандарттар ұйымының (ISO) анықтамасына сәйкес *алгоритм* деп операциялардың ақырлы саны арқылы есептер шешімін анықтайтын ережелердің ақырғы жиынын айтуға болады.

Алгоритмның негізгі қасиеттеріне дискреттілік, нақтылық, жалпыламалық, нәтижелік жатады.

- *Дискреттілік* - алгоритм дискретті ақпараттармен (мысалы, цифрлық не символдық) өңдеу жүргізгендігін көрсетеді.
- *Анықтылық* – алгоритмде жасалатын барлық әрекеттер көрсетіледі және әрбір әрекетті басқа түрде түсіндіруге жол берілмейді.
- *Жалпыламалық* – алгоритмнің тек қандай да бір бірегей мәндеріне ғана емес, бастапқы деректердің мәндер жиындарына да қолданушылығын білдіреді.
- *Алгоритмнің нәтижелігі* – керекті нәтижені ақырғы қадам саны арқылы алу мүмкіндігін көрсетеді.

Қаралған алгоритм қасиеттері арқылы оларды есептеу машинасында іске асыру мүмкіндігін көрсетеді. Алгоритммен туындайтын процестерді есептеу процестері деп атаймыз.

Есептеу машинасында есептеу алгоритмдері командалар жиынтығынан тұратын бағдарлама түрінде көрсетіледі.

2.1. Фон-Нейман тұжырымдамасының негізгі принциптері

Есептерді шығару алдында бағдарлама командалары жадыда сақталса ондай есептеу машиналары *бағдарламалары жадыда сақталатын есептеу машиналары* деп аталады. Бағдарламаны машинаның жадында сақтау идеясын бірінші болып Фон Нейман [38] жариялаған. Фон Нейман тұжырымдамасын төменгі 4 принципке алып келуге болады:

- Екілік кодтау;
- Бағдарламамен басқару;
- Жадының біркелкілігі;
- Жадының мекенделуі;

Екілік кодтау принципі

Бұл принципке сәйкес кез-келген ақпарат - команда және дерек 0 және 1 екілік цифрлары өрнектеледі.

Әрбір дерек түрі (типі) екілік кодымен көрсетіледі және олардың өз пішімдері болады. Пішімдері екілік сандар (биттер)болған кезде оны өріс деп атайды. Сандар пішімі 2 биттер тобынан тұрады, біріншісі – таңба өрісі, екішісі – разряд мәндер өрісі.

Команда пішімі екі өрістен (2.1-сурет) тұрады: біріншісі – операция коды (ОПК), екіншісі мекен-жай өрісі (мекен-жай бөлігі - МЖБ).

Операция коды (ОПК)		Мекен-жай бөлігі (МЖБ)	
r-1	0	p-1	0

2.1-сурет. Команда құрылымы

Операция коды қандай операция орындау керектігін көрсететін нұсқама: ол r – разрядты екілік сандармен көрсетіледі.

Мекен-жай бөлігінің түрі және оны құратын мекен-жайлар саны команда типіне байланысты болады. Деректі өңдеу командаларында МЖБ өңделетін объекттердің (операндтардың) немесе нәтиженің мекен-жайлары көрсетіледі. Одан басқа есептеу ретін өзгертетін командалар бағдарламаның келесі командасының мекен-жайын немесе енгізу/шығару операциялары орындалған кезде – енгізі/шығару құрылғысының мекен-жайы көрсетеді. Мекен-жай бөлігіде екілік кодамен көрсетіледі. Оның ұзындығын p – арқылы белгілейміз. Сонымен есептеу машинасының командасы (r+p) – разрядты екілік сан болады.

Бағдарламамен басқару принципі

Есепті шығару алгоритміне қатысатын керекті барлық есептеулер бағдарлама түрінде көрсетіледі. Бағдарлама басқару сөздерінің тізбектерінен – командалардан тұрады. Әр команда есептеу машинасында орындалатын операциялар жиынтығының бірін көрсетеді. Бағдарлама командалары машина жадының іргелі ұяшықтарында қатар орналасады. Орындау тәртібі олардың орналасу тәртібіне байланысты – табиғи тәртіппен орналасады. Керек кезінде арнайы командалар арқылы командалардың орындалуының табиғи тәртібі өзгертілуі мүмкін. Орындау тәртібін өзгерту шешімі алдыңғы команда

нәтижесін талдау арқылы немесе шартсыз команданың нұсқауымен анықталады.

Жадының біркелкілік принципі

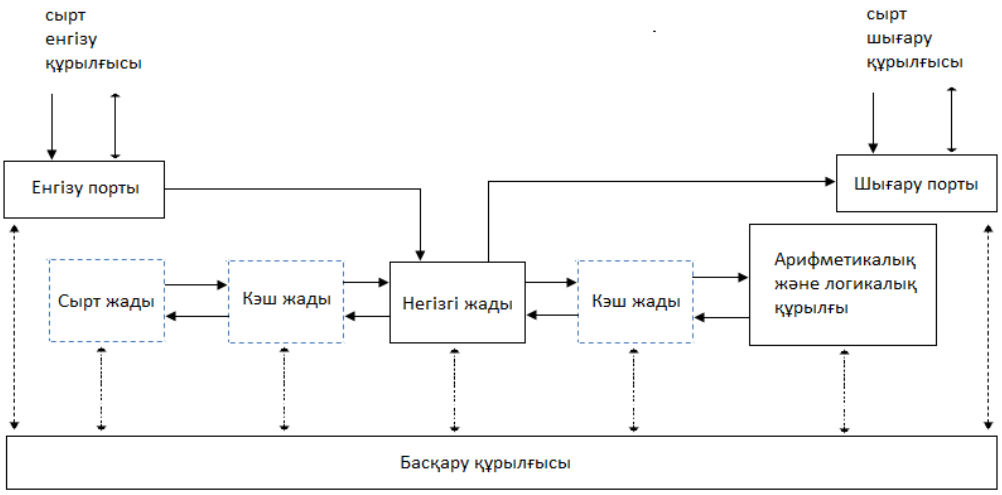
Командалар мен деректер бір жадының ұяшықтарында сақталады. Сырт қарағанда оларды бір-бірінен айыру мүмкін емес. Оларды тек пайдалану тәсілі арқылы айырамыз. Бұл – командалар үстінен сандар сияқты әртүрлі операциялар орындауға мүмкіндік береді. Фон Нейман тұжырымдамасында командалар мен деректердің бір жадыда орналасу тәсілі ұсынылған. Мұндай машиналар архитектурасын *принстон архитектурасы* деп атайды, өйткені ондай архитектуралы машина Принстон университетінде жасалынған. Осы уақытта Гарвард университетінде де компьютердің басқа моделі жасалынған, мұндай компьютерлерде дерек пен командалар әртүрлі жадыда сақталған. Архитектураның мұндай түрін *Гарвард архитектурасы* деп атайды.

Жадыға мекендеу принципі

Кез-келген компьютер жадысы нөмірленген ұяшықтардан тұрады. Ұяшықтарға нөмірлері арқылы процессор еркін қатынас құра алады. Командалар мен деректердің екілік кодтары сөз деп аталатын ақпарат бөліктеріне бөлініп жады ұяшықтарында сақталады. Оларға қатынас құру үшін ұяшықтар нөмірімен анықталатын мекен-жайлар қолданылады.

2.2. Фон Нейман машинасының құрылымдық сұлбасы.

Жоғарыда айтылған Фон Нейман принциптерін іске асыратын машина – Фон Нейман машинасы жадыдан, басқару құрылғысынан, арифметикалық және логикалық құрылғыдан (АЛҚ) және енгізу/шығару құрылғыларынан тұрады (2.2-сурет). (Суретте көрсетілген КЭШ жадылары және сырт жады типтік Фон Нейман машинасының құрамына арнайы енбеген).



2.2-сурет. Фон-Нейман машинасының құрылымдық сұлбасы

Кез-келген есептеу машиналарында деректер мен бағдарламаларда енгізетін және шығаратын сырт құрылғылары болады. Дерек сырт құрылғылармен енгізілетін порт арқылы машина жадына енгізіледі. Есептеу нәтижелері шығару порты арқылы сыртқы шығару құрылғыларына беріледі. 2.2-суреттен көрініп тұрғандай, сырт құрылғылары есептеу машинасымен порттар арқылы байланысады және алмасады. Мұнда порттар есептеу машинасымен сырт құрылғылар жұмыстарын ұштастыру міндетін атқарады. Енгізу және шығару порттар жиынтығын енгізу/шығару құрылғысы (ЕШҚ) немесе есептеу машинасының енгізу шығару модулі (ЕШМ) деп атайды. Әрбір сырт құрылғылары есептеу машинасымен алмасу үшін енгізу-шығару командасының мекен-жай бөлігінде әрбір порттың бірегей нөмірі көрсетіледі.

Есте сақтау (жады) құрылғылары

Компьютер жады бір-бірімен байланысқан күрделі көп деңгейлі есте сақтау құрылғыларынан (ЕСҚ) тұрады.

Деректерді ұзақ уақытқа сақтау үшін сырт жады пайдаланады. Белсенді бағдарламаға керекті командалар мен бағдарламалар негізгі жадыда (НЖ) сақталады. Жадыда әрбір сөз жеке нөмірленген ұяшықтарда орналасады.

Негізгі жадының кез-келген ұяшықтарына кез-келген ретпен қатынас құруға болады. Мұндай жады түрін еркін қатынасты жады құрылғысы деп атаймыз. Қазіргі ЕМ негізгі жады жартылай өткізгіш жедел есте сақтау құрылғыларынан тұрады. Мұндай есте сақтау құрылғысында сақталған деректер электр қорегін үзіп тастағанда жоғалып кетеді, яғни жады энергияға тәуелді болады. Негізгі жадының бір бөлігі энергияға тәуелсіз болу үшін оның

құрамына тұрақты есте сақтау құрылғысын (ТЕСҚ) қосады. Мұндай ТЕСҚ еркін қатынасты жадыға жатады, бірақ одан дерек көбінесе оқылады.

Негізгі жады машинаның басқа құрылғыларымен басқару құрылғысында орналасқан дерек регистрі (МРг) арқылы дерек алмасады. Жадыдан деректі оқығанда оқылған дерек МРг-ге жазылады. Деректі жазарда дерек алдын ала МРг-ге қабылданады. Жадыға оқу/жазу басқару құрылғыларынан келетін арнайы сигналдар арқылы орындалады.

Негізгі жадыда деректерді белгілі бір ұяшықтарға жазу немесе оқу үшін оның ұяшық нөмірін - мекен-жайын көрсету керек. Ол үшін басқару құрылғысының құрамында жады мекен-жайының регистрі (ЖМЖРг) орналасады.

Негізгі жады ұяшықтары байтпен өлшенеді. Деректерді 2, 4 немесе 8 байт арқылы көрсетуге болады. Мұндай жағдайда сан мекен-жайы ретінде *кіші байт мекен-жайы* алынады. Мысалы 32 разрядты сан 200, 201, 202 және 203 мекен-жайларында жазылса, онда сан мекен-жайы 200 болады. Мекендеудің мұндай тәсілін *кіші байт* арқыла мекендеу деп атайды. Мекендеудің басқа тәсілінде кіші мекен-жайда көп байтты санның ең жоғарғы байты орналасады. Мұндай тәсілді *үлкен байт* арқылы мекендеу тәсілі деп атайды. Кіші байт арқылы мекендеу Intel фирмасының микропроцессорларында кеңінен қолданылады. Үлкен байт арқылы мекендеу Motorola фирмасының микропроцессорлары мен IBM фирмасының үлкен машиналарында кеңінен қолданылады. Көптеген есептеу машиналарында бір мекендеу тәсілінен екіншісінен өтетін арнайы командалар бар.

Көлемі үлкен бағдарламалар мен деректер сыртқы есте сақтау құрылғыларында сақталады. Мұндай жады құрылғылары энергиядан тәуелді емес, олар магниттік, оптикалық дисктер немесе таспалар арқылы жасалынады. Оларда ақпарат файлдар түрінде сақталады (ISO стандартына сәйкес, файл – аты анықталған бір блок ретінде өңделтін жазбалар жиынтығы).

Қазіргі есептеу машиналарының құрамына аралық КЭШ жады кіреді. КЭШ жадының сыйымдылығы үлкен емес, бірақ жылдамдығы өте жағары. КЭШ жадыда негізгі жадының жиі қолданылатын командалары мен деректері сақталады.

Жоғарыда сөз болған жады құрылғыларынан басқа арифметикалық-логикалық құрылғылардың (АЛҚ) құрамына кіретін регистрлік жады жатады. Мұндағы әрбір регистрді жады құрылғысының бір ұяшығы деп қарауға болады. Регистрлік жады жылдамдығы КЭШ жады жылдамдығынан жоғары.

Регистрлер саны әдетте көп емес. Регистрлер жиынтығын регистрлер жадысы немесе жылдамдығы өте жоғары жедел жады деп, не болмаса регистрлік файлдар деп атайды. Регистрлерде жиі қолданылатын константа, аралық нәтижелер немесе қызметтік ақпараттар сақталады.

Басқару құрылысы (БҚ)

БҚ есептеу машинасының маңызды бөлігі. БҚ бағдарламаның автоматты түрде орындалуын басқару сигналдары (БС) қалыптастыру арқылы ұйымдастырады. Басқару сигналдар тізбегі 2.2-суретте пунктир сызықтарымен көрсетілген. БҚ құрамына команда санауышы (КСАН), команда регистрі (КР), операция кодының дешифраторы (ОКД), микробағдарламалық автомат (МБА) және жоғарыда аталған жады мекен-жай регистрі (МЖРг) және жады дерек регистрі жатады (МРг).

БҚ кез-келген Фон Нейман ЕМ ажырамас элементі. Мұндай машинада, команда реті бағдарламада көрсетілгендей өзгермесе, онда ол табиғи ретпен орындалады. Мұндай жағдайда кезекті команда мекен-жайы орындалып жатқан мекен-жайды бірге өсіру арқылы табамыз (егер кез-келген команда бір ұяшықта орналасса). Мұндай есептеуді команда санауышы орындай алады. Есептеу алдында команда санауышына бірінші команда жазылған ұяшық КСАН мекен-жайы жазылады. Кезекті команда орындау үстінде, егер команда реті өзгермейтін болса онда команда санауышының мәні +1 КСАН сигналы арқылы бірге өсіріледі. Команданың табиғи ретін өзгерту үшін КСАН бірді қосудың орнына өту нүктесінің мекен-жайын жазамыз. Кейбір есептеу машиналарында КСАН орнына бағдарламалық санауыш БСАН немесе команда көрсеткіші (КК) деген атаулар қолданылады.

Команда санауышы арқылы анықталған мекен-жайдан команда оқылып, ол команда регистріне (КРг) жазылады. КРг команданы оны орындалып болғанша сақтайды. КРг екі бөліктен операция кодасының (ОПК) бөлігінен және мекен-жай бөлігінен (МЖБ) тұрады.

Операция коды дешифратор арқылы микробағдарламалық автомат (МБА) кодына түрлендіріледі. Компьютерде дешифратор арқылы операцияның екілік коды унитарлық кодқа түрлендіріледі. Унитарлық кодтың әрқайсысы тиісті операцияны басқаратын микробағдарламалық автоматты іске қосып, басқару сигналдарын қалыптастырады. Осы басқару сигналдары арқылы командада көрсетілген операндтар мен әртүрлі операция орындалады.

Арифметикалық және логикалық құрылғы (АЛҚ) немесе Операциялық құрылғы (ОПҚ)

Аты айтып тұрғандай, құрылғыда арифметикалық және логикалық операциялар орындалады. АЛҚ блоктарына операциялық блоктар (ОПБ), операндтар регистрлері, белгілер регистрі (БРг) және аккумулятор (АКК) жатады.

Операциялық блоктар (ОПБ) АЛҚ барлық операциялар орындалатын негізгі блоктар. Қазіргі ОПБ комбинациялық сұлбалар арқылы құрылады.

Операндтар регистлері операциялық блоктың кірісінде орналасқан регистрлер. Оларда операндтар операция аяқталғанша сақталады.

Белгілер регистрінде (БРг) арифметикалық не логикалық операциялар орындалып болған соң нәтиже белгілері жазылады.

Нәтиже белгілеріне нольге тең белгісі, аса толу, жоғары разрядталған шағын тасымал және т.с.с жатады.

Аккумуляторды (АКК) басқару құрылғысына да, АЛУ құрамына кіретін регистр деп қарауға болады. Аккумулятор әр түрлі қызмет атқарады. Аккумуляторға операцияға қатысатын операндтардың бірі бұрын орындалған операцияның, орындалып отырған операция нәтижесі жазылады. Енгізу/шығару операциялары да аккумулятор арқылы жүргізіледі.

2.3. Микрооперациялар мен микрокомандалар тілдері.

Есептеу машиналарының әртүрлі құрылғыларын бейнелеу үшін әртүрлі бейнелеу тілдері қолданылады. Ондай тілдер құрылғыларды тәптіштеу дәрежесіне байланысты әртүрлі болады. Мысалы электрон сұлбалары үшін бейнелеу тілдері ретінде сұлбалардың электр тізбегінің дифференциялдық теңдеулері жатады. Ал операциялық құрылғылар, жады түйіндері және басқару автоматтары үшін формалды бейнелеу тіліне микрооперация және микрокомандалар тілдері жатады.

Сөздерді, регистрлер мен шиналарды бейнелеу.

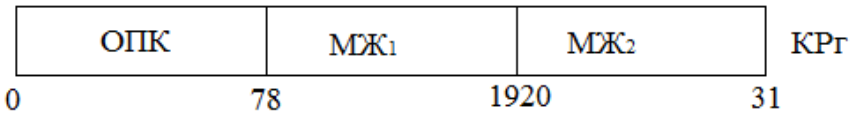
Сөздерді (сандарды, логикалық коданы т.б.) бейнелеу үшін оның аты-идентификатор мен разряд көрсеткіші керек. Идентификатор ретінде әріптерден басталатын кез келген әріптер мен цифрлар тізбегін алуға болады. Разряд көрсеткіші ÷ белгісімен бөлінген жоғарғы және төменгі разрядтар нөмірлерінен тұрады. Көрсеткіш квадрат жақшаға алынады. Мысалы сөз – сан

$X_{20}=x_1, x_2,.....,x_n;$ $X_{20}=\alpha_1, \alpha_2,.....,\alpha_n$;

Мұндағы x_i – екілік разрядтарды мынадай түрде көрсетуге болады.

X_{20}=[0÷n]

Регистрді бейнелеу үшін оның аты, идентификаторы және разряд көрсеткіші керек. Мысалы, команда регистрін (КРг) бейнелеу түрі



2.3-сурет. Регистрді бейнелеу

Регистрді бейнелеу былай болады:

КРг [0÷31], ал оның бөліктері ішкі регистрлері (команда өрістері төмендегідей көрсетіледі):

КРг [0÷7] немесе КРг [ОПК],

КРг [20÷31] немесе КРг [$МЖ_2$].

Ішкі регистрдің разряд көрсеткішінде оның идентификаторын көрсетуге болады. Жеке разрядтарды, мысалы 9 – разрядты КРг [9] түрінде көрсетуге болады. Сөздерді (командаларды) және сигналдарды беруге арналған сандар (тізбектер) жиынтығын шина деп атайды. Цифрлық құрылғыларға сырттан келетін немесе сырт құрылғыларынан берілетін сөздер регистр ретінде бейнеленеді. Мысалы 32 разрядты команда берілетін шинады бейнелеу төмендегідей жазылады: КҚ[0÷31]

Массив пен жадыны бейнелеу

Бейнелеуде олардың идентификаторы (мысалы, негізгі жадының 4-модулі – НЖ 4) және квадрат жақша ішінде сөздердің жоғарғы және төменгі нөмірі, массивтін төменгі және жоғарғы шекарасы, сөз разрядтарының нөмірлеу реті көрсетіледі. Мысалы n разрядты r сөзі бар негізгі жадының 4-модулі массиві былай бейнеленеді: НЖ 4[0÷r-1, 0÷n-1]. Мұндағы n разрядты жадының j-n ұяшығы және К-жадының К-разрядты (массивтің бағанасы) сәйкес былай жазылады НЖ 4[j, 0÷ n-1] және [0÷ n-1] және НЖ 4[0÷ r -1,К].

Микрооперацияларды бейнелеу

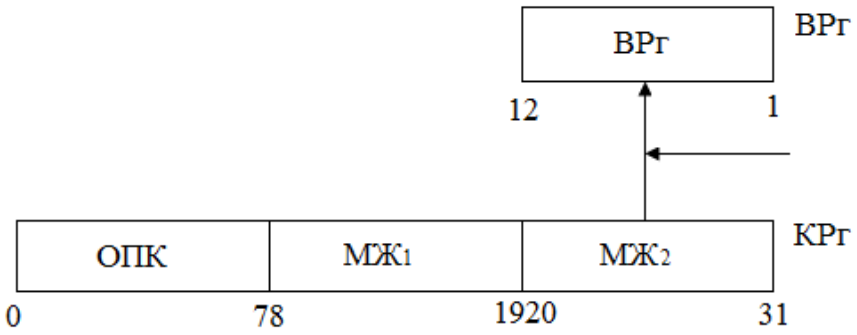
Микрооперациялар арқылы деректермен түрлендірулер жүргізіледі. Мұндай түрлендірулерге бір немесе екі операндтар разрядтарымен жүргізілетін арифметикалық, логикалық, функционалдық түрлендірулер жатады.

Мысалы регистрге сөз жазу, кодты жіберу, кері код алу, екі операндпен ЖӘНЕ, НЕМЕСЕ операцияларын орындау, жылжыту, санау, қосу, алу, көбейту,

бөлу, сандарды салыстыру т.б. операциялар. Микрооперациялар операцияға қатысатын операндтар санына байланысты бір орынды және екі орынды болып келеді. Микрооперация микрооператормен және операцияны орындайтын басқару сигналының идентификаторы – енімен бейнеленеді. Екі орынды операция микрооперацияға мысал:

F: АРг[k÷k+l]:=ВРг[m÷m+l]* СРг[n÷n+l].

Басқару сигналының идентификаторы (екі) микрооператорымен қос нүктемен бөлінеді. Меншіктеу микрооператоры(:=) белгісімен көрсетеді. Формулада түрлендіру (*) белгісі операцияға қатысатын операндтар орны көрсетілген. Меншіктеу белгісінің сол жағында түрлендіру нәтижесін сақтайтын регистр (немесе оның бөлігі) көрсетілген. Егер микрооперация арқылы тек сөз берілсе онда микрооператор формуласында тек сөз бейленеді. Мысалы екінші A_2 операндын команда регистрінен КРг мекен-жай регистріне ВРг беру төмендегідей көрсетіледі (2.4-сурет):



2.4-сурет. $МЖ_2$ мекен-жайын КРг-ден ВРг-ге беру

Бер В: ВРг[12÷1]:= КРг[20÷31];

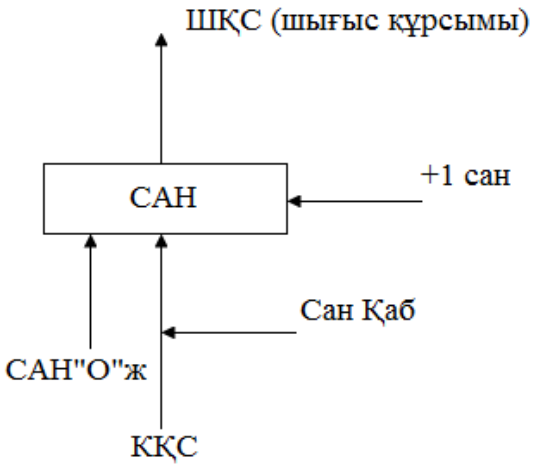
Немесе Бер В: ВРг[12÷1]:= КРг[20÷31].

Мұнда Бер В – басқару сигналының ені $МЖ_2$ мекен-жайын КРг регистрінен ВРг регистріне беру.

Микрооперация басқару сигналы 0 немесе 1 мәндері алады: 1 – микрооперация белсенді болады, 0 – микрооперация орындалмайды. 1 басқару сигналы (ені) арқылы бір тактта бірнеше микрооперациялар орындалуы мүмкін. Мұндай жағдайда бір-бірімен үтірмен бөлінеді. Микрооперацияларға мысалдар. Санауыш кодын өсіру (2.5-сурет). Санауышта мынадай микрооперациялар орындалады:

- санауышқа (О) жазу (САН «0» ж);
- кіріс шинасынан кодты қабылдау (Сан Қаб);

- санауыш мәнін (кодты) бірге өсіру (+1 сан).



2.5-сурет.Санауыш кодын өсіру

Жылжыту микрооперациялары

Жылжыту микрооперацияларына арифметикалық (АЖЫЛ), логикалық (ЛЖЫЛ), циклдік (ЦЖЫЛ) жылжыту микрооперациялары жатады. Микрооперациялардың оң жақ қатарынан (СЛ-солға, және ОҢ-оңға) жылжыту бағыты көрсетіледі және жақша ішінде жылжыту саны көрсетіледі. Мысалы, С регистр мазмұнын төрт разрядқа оңға жылжыту микрооперациясы төмендегідей жазылады:

Жыл: СРг:= АЖыл ОҢ (4) СРг.

Жылжыту микрооперациясы бір регистрден екіншісіне «қисық беру» арқылы көп орындалады. А регистрінен В регистріне дерек К разрядқа оңға немесе солға жылжытып беру микрооперациясы төмендегідей көрсетуге болады:

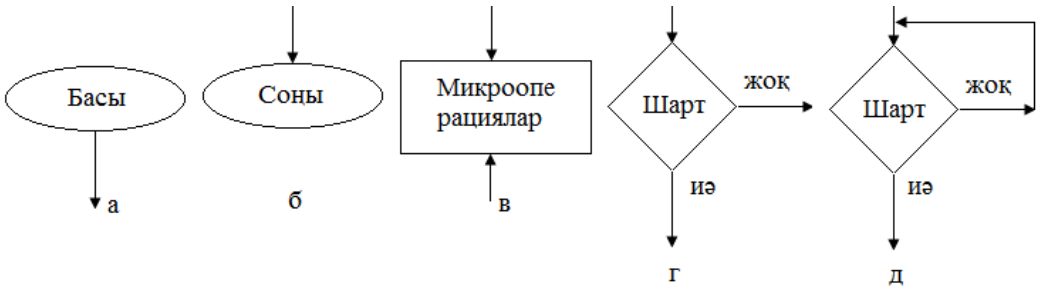
Жыл К: ВРг:= ОҢ (К) АРг;

Жыл К: ВРг:=СЛ (К) АРг.

Микрокоманданы бейнелеу

Микрокомандалар микрооперациялар сияқты бейнеленеді. Олар – микрокомандалар ені мен бір-бірінен үтірмен бөлінген осы микрокомандада орындалатын микрооперациялар немесе басқару сигналдарының жиыны.

Микробағдарлама графика түрінде көрсетілуі мүмкін. Олардың әр төбесі микрокомандаларға немесе тізбекпен орындалатын микрокомандалар тобына сәйкес келеді. Микропрограмама құрғанда бес түрлі төбелер пайдаланылады (2.6-сурет).



2.6-сурет. Граф төбелері: а-басы; б-соңы; в-операторлық; г-шартты; ә-күту

Бастапқы төбе (2.6, а-сурет) микробағдарламаның басталуын көрсетеді, оның шығысы жоқ. Соңғы төбе (2.6, б-сурет) микробағдарламаның соңын көрсетеді, оның тек кірісі болады. Операторлық төбеге (2.6, в-сурет) бір машиналық тактта орындалатын микрооперациялар жазылады. Төбеде бір шығыс және бір кіріс болады. Шартты төбе (2.6, г-сурет) есептеу процесін тармақтауға пайдаланады. Төбеде бір кіріс, екі шығыстар бар. Шарт орындалса есептеу үрдісі «иә» тармағымен, ал орындалмаса – «жоқ» тармағымен өрбиді. Күту төбесі (2.6, ә-сурет) арқылы құрылғы жұмысының күтілуін белгілейді.

Бақылау сұрақтары.

1. Есептеу машиналарында есеп алгоритмдері немен көрсетіледі?
2. Команда пішімі қандай өрістерден тұрады?
3. Жадының біркелкілігі деген не?
4. Негізгі жады деректерімен қандай типтік түйіндер арқылы алмасады?
5. Басқару құрылғысында команда санауышы не үшін керек?
6. Команда регистрі деректерді қанша уақыт аралығында сақтайды?
7. Белгілер регистрінде қандай деректер сақталады?
8. Микрооперация және микрокоманда дегеніміз не?
9. Микрооперациялар мен микрокомандалар тілдері не үшін керек?

3. ЕСЕПТЕУ МАШИНАЛАРЫ МЕН ЖҮЙЕЛЕРІНІҢ ҚҰРЫЛЫМДЫҚ СҰЛБАЛАРЫ ЖӘНЕ ОЛАРДЫҢ НЕГІЗГІ КӨРСЕТКІШТЕРІ

Есептеу машиналары мен жүйелерінің архитектуралық артықшылықтары мен кемшіліктері оларды құратын құрылғылардың бір-бірімен байланыстыру тәсілдерінде жатыр.

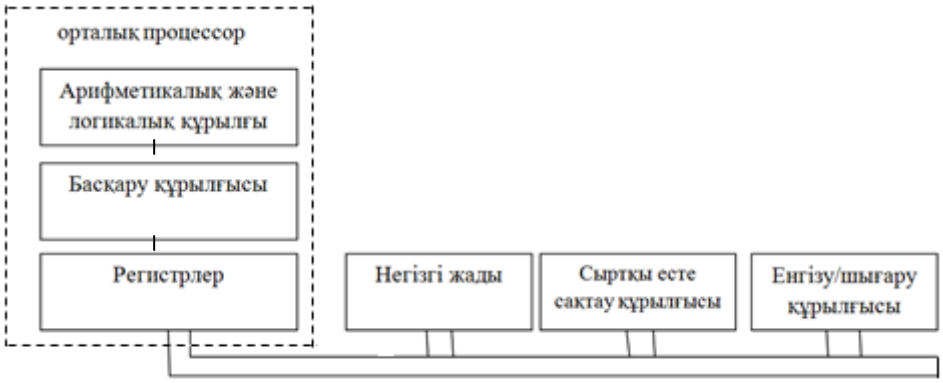
3.1. Есептеу машиналары мен жүйелерінің құрылымдық сұлбалары

Есептеу машиналарының құрылымы

Қазіргі кезде есептеу машиналары екі түрде құрылады: біріншісінде - машина құрылғалары бір-бірімен тікелей байланыста болады, екінші түрінде олар шина арқылы байланысады.

Бірінші түріне жоғарыда қаралған (1.1-сурет) Фон Нейман машинасы жатады. Бұл машинада құрылғылар (арифметикалық-логикалық құрылғы, басқару құрылғысы машина жады, енгізу/шығару құрылғысы) бір-бірімен тікелей байланысқан. Мұндағы әр байланыстың өзіндік ерекшеліктері бар. Олар осы байланыс арқылы берілетін ақпараттың түрлерімен (дерек, мекен-жай, басқару сигналдары), олардың берілу қарқындылығымен т.б анықталады. Олардағы шина линияларының саны, өткізгіштік қабілетін қамтамасыз ететін тетіктер құрамы әртүрлі. Фон Нейман машиналарында оның жылдамдығына әсер ететін ең маңызды байланыс ол –«Процессор - негізгі жады» байланысы. Өйткені ол арқылы дерек және командалар ағымы үлкен екпінділікпен берілу керек.

Ортақ шиналы машиналарда барлық құрылғылар бір шина магистраліне қосылған. Осы магистраль арқылы командалар, деректер және басқару сигналдары ағындары беріледі (3.1-сурет).

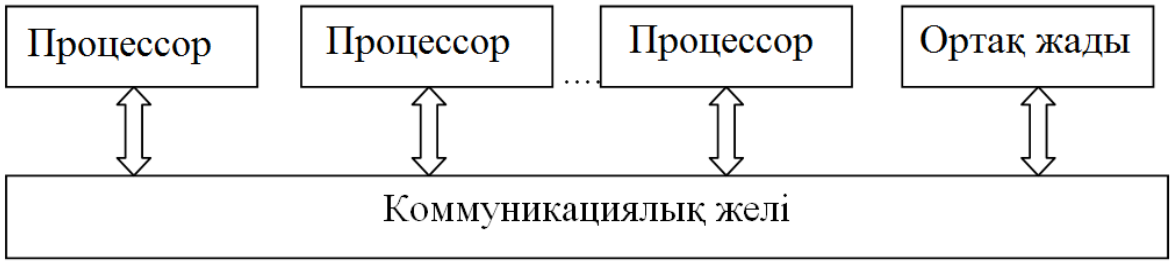


3.1-сурет. Ортақ шина арқылы құрылған есептеу машинасы

Ортақ шиналы машиналардың артықшылығы - оның құрамын өзгерту мүмкіндігінде. Сол арқылы машина өнімділігін, жады сыйымдылығын арттыруға және енгізу/шығару блоктарын қосу арқылы машиналарға қосылатын сырт құрылғыларын көбейтуге мүмкіндік аламыз. Осы мүмкіндіктерге байланысты қазіргі компьютерлерде ортақ шиналы архитектура кеңінен қолданылады. Ортақ шиналы машиналардың кемшілігі ондағы шина санында. Өйткені машинаның әртүрлі құрылғылары тек бір шина арқылы кезекпен алмасады. Екі құрылғы бір-бірімен алмасып жатқанда қалғандары күтіп тұру керек. Ортақ шинада жүктеме поцессор мен негізгі жады арасында алмасу барысында орын алады. Мұндай алмасу үстінде командалар жадыдан оқылады, орындалып болған соң нәтижелер қайтадан жадыға беріледі, сондықтан шина ресурсы енгізу/шығару операциясына аз бөлінеді. Ортақ шина жүктемесін азайту үшін қазіргі машиналар құрамына қосымша шиналар енгізілген.

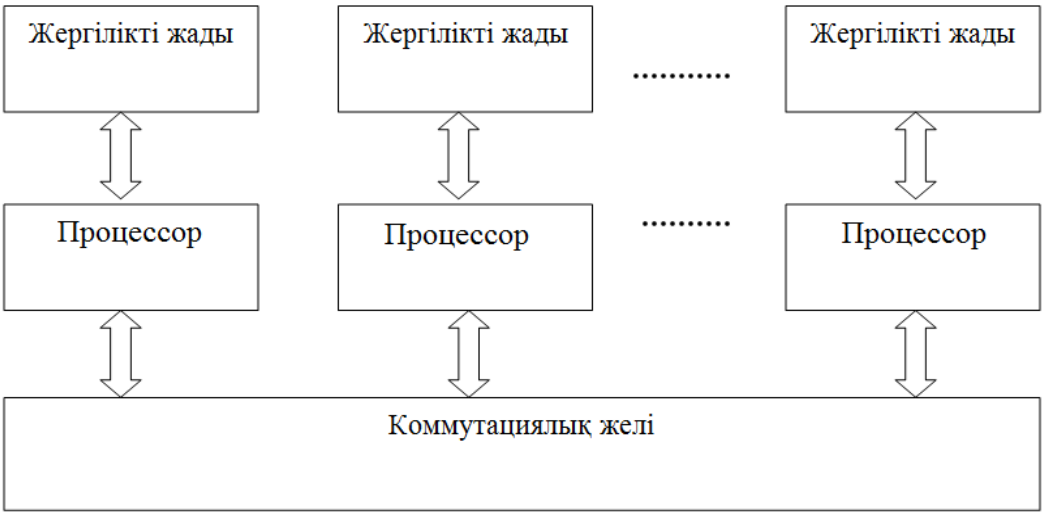
Есептеу жүйелерінің құрылымдары

Есептеу жүйесі деп құрамында бірнеше процессорлары немесе толыққанды есептеу машиналары бар жүйені айтамыз. Процессорлар және есептеу машиналары бір-бірімен жады арқылы екі жолмен біріктіреді.



3.2-сурет. Ортақ жадылы есептеу жүйесінің құрылымдық сұлбасы

Біріншісін ортақ жадылы есептеу жүйесі деп атаймыз (3.2-сурет). Мұндай жүйеде процессорлар коммуникациялық желі арқылы ортақ бір жадымен байланысып жұмыс атқарады. Есептеу жүйесінің екінші түрінде әр процессордың өз жадысы болады (3.3-сурет). Мұндай есептеу жүйелері үлестірілген жадылы есептеу жүйелері деп аталады. Мұндай жүйелерде жеке есептеу машиналары бір жүйеге біріктіріледі. Жүйе құрылғылары деректермен коммуникациялық желі немесе хабарлар арқылы алмасады.



3.3-сурет. Үлестірілген жадылы есептеу жүйесінің құрылымдық сұлбасы

3.2. Есептеу машиналарының негізгі көреткіштері

Есептеу машиналарының негізгі көрсеткіштеріне олардың жылдамдығы, өнімділігі, сенімділігі, құны және жады сыйымдылығы жатады.

Жылдамдық

Жылдамдықтың екі түрін: нақтылы (номинальді) және орташа жылдамдықтарды қараймыз.

Нақтылы жылдамдық есептеу машиналарының стандартты операцияларды орындау мүмкіндіктерін көрсетеді. Стандарт операция ретінде қысқа қосу операциясын алады. Егер $\tau_{қос}$ арқылы қосу уақытын белгілесек, онда нақтылы жылдамдық төменгі формуламен анықталады:

$$\vartheta_{нақ} = \frac{1}{\tau_{қос}} \left[\frac{оп}{сек}\right]. \qquad [3.1]$$

Орта жылдамдық эталонды алгоритмді немесе алгоритмдер класын есептеуге жұмсалатын уақыт мөлшерін көрсетеді. Орта жылдамдық мөлшері ЕМ және алгоритм параметрлерімен анықталады. Онда

$$\vartheta_{орт} = \frac{1}{\sum_{i=1}^{e} \tau_i q_i} \left[\frac{оп}{сек}\right]. \qquad [3.2]$$

Мұнда вектор $\{\tau_1, \tau_2, \ldots, \tau_e\}$ ЕМ командалар жүйесін сипаттайды, ал вектор $\{q_1, q_2, \ldots, q_e\}$ операциялардың жиілік векторы деп аталатын есептеу алгоритмін сипаттайды.

[3.2] формуласы арқылы бір алгоритммен жұмыс жасайтын машинаның орташа жылдамдығын анықтауға болады. Егер шығарылатын есеп бірнеше жеке алгоритмдерден тұрса, онда орташа жылдамдық мына формуламен анықталады:

$$\vartheta_{орт}^{m} = \frac{1}{\sum_{j=1}^{m}\sum_{i=1}^{e}\beta_j q_{ji}\tau_i}\ [\frac{оп}{сек}], \qquad [3.3]$$

Формуладағы m – жеке алгоритмдер саны; β_j – j – жеке алгоритмдердің орындалу жиілігі; q_{ji} - j – жеке алгоритміндегі i – типті операцияның өнімділік жиілігі.

ЕМ өнімділігі – бір уақыт аралығында орындалатын эталонды алгоритмдер (ЭА) санымен анықталды:

$$П = \frac{1}{T_{ЭА}}[\frac{ЭА}{сек}] \qquad [3.4]$$

Сенімділік

Сенімділік – есептеу машинасының белгілі уақыт аралығында берілген бағдарламамен (алгоритммен) дұрыс есеп шығару қасиеті. Есеп шығару үстінде әртүрлі кездейсоқ факторлардың әсерінен машина істен шығуы немесе жаңылыс орын алуы мүмкін. Машинаның істен шығуын (бас тартудың) оның жұмыс жасау қабілетінің жоғалуымен түсіндіруге болады. Әдеттте сенімділік машинаның белгілі бір уақыт аралығында қалтқысыз жұмыс жасау ықтималдығымен p (t) анықталады. Қалтқысыз жұмыс жасау ықтималдығы бас тарту қарқындылығы λ (t) арқылы анықталады. Бас тарту қарқындылығы – бір уақыт аралығында орын алатын орташа бас тару саны. Әдетте, ЕМ элементтерінің бас тарту қарқындылығы тұрақты. Мұндай жағдайда бас тартпай жұмыс жасау ықтималдылығы экспоненциялық заңмен анықталады:

$$P(t) = e^{-\lambda t} \approx 1 - \lambda t. \qquad [3.5]$$

Есептеу машинасы әртүрлі элементтер типінен тұрады, сондықтан оның бас тарту қарқындылығы мына формуламен анықталады:

$$\lambda = \sum_{i=1}^{m} \lambda_i k_j,$$

Мұндағы λ_i - i – типті элементтердің бас тарту қарқыны; k_j – j типті элементтер саны; m – элементтер типінің саны.

Машинаның бас тарту қарқындылығы белгілі болса, онда бас тарту аралықтарының орташа уақытын (T_0) төмендегі формуламен есептеуге болады:

$$T_0 = \frac{1}{\lambda}. \qquad [3.6]$$

Бастарту пайда болған соң оны орынына қайта келтіру үшін $T_{ок}$ уақыт жұмсалады. $T_{ок}$ орташа мәні кездейсоқ сан. Сенімділік тұрғысынана қарағанда машинаны пайдалану қасиетін жұмысқа дайын болу коэффициентімен анықтауға болады:

$$K_д = \frac{T_0}{T_0 + T_{ок}}. \qquad [3.7]$$

Есте сақтау (жады) құрылғыларының сыйымдылығы

Есте сақтау құрылғыларының сыйымдылығы - оларда бір уақытта сақталатын бит не байт санымен анықталады. Әдетте бит және байт сөздеріне қосымша кило, мега, тера, пета және т.б. өлшем бірліктері тіркеледі.

- Килобайт – К (1Кбайт = 1024 байт);
- Мегабайт – М (1Мбайт = 1024 Кбайт);
- Гигабайт – Г (1Гбайт = 1024 Мбайт);
- Терабайт – Т (1Тбайт = 1024 Гбайт) т.с.с.

Құны

Есептеу машинасының құны (S) оның құрамына кіретін құрылғылардың құнымен анықталады. Машина құнын өсіру арқылы оның өнімділігін, сенімділігін, есеп шығару дәлдігін өсіруге болады.

Бақылау сұрақтары.

1. Тікелей байланысқан есептеу машиналарының кемістігі неде?
2. Ортақ шиналы машиналардың кемшілігі мен артықшылығы неде?
3. Нақтылы және орта жылдамдықты қалай түсінуге болады?
4. Есептеу машиналарының өнімділігі қалай анықталады?
5. Дайын болу коэффиценті қалай анықталады?
6. 1 терабайтта (Тбайт) канша мегабайт (Мбайт) бар?

4. ЕСЕПТЕУ МАШИНАЛАРЫНДА ОПЕРАНДТАРДЫ КӨРСЕТУ

Машина командаларымен өңделетін деректерді *операндтар* деп атайды. Базалық операндтарға сандар, символдар және логикалық деректер жатады. Олардан басқа ЕМ-де күрделі ақпараттық бірліктер: графикалық, аудио-видео, анимациялық ақпараттар да өңделеді. Мұндай ақпараттар жоғарыда айтылған базалық операндтардың туындысы. Олар файлдар түрінде сыртқы есте сақтау құрылғыларында сақталады. Сандық ақпараттар үш түрге бөлінеді. Олар: бекітілген үтірлі сандар (бүтін, бөлшек сандар) жылжымалы үтірлі сандар және логикалық деректер.

4.1. Бекітілген үтірлі сандар

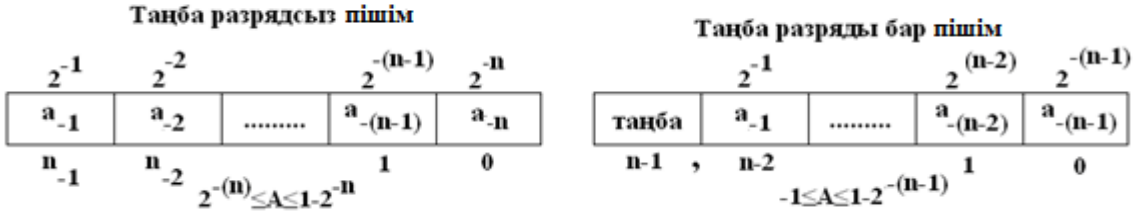
Бекітілген үтірлі сандар (БҮС) сан таңбасы мен q-лік кодта көрсетілген модульден тұрады. Мұнда q – есептеу жүйесінің негізі немесе базасы. Қазіргі есептеу машиналарында екілік (q=2), сегіздік (q=8) немесе оналтылық (q=16) есептеу жүйелері қолданылады. Оң санның таңбасы 0 цифрымен, ал теріс санның таңбасы 1 санымен кодталады.

Таңба орналасқан сан кодының разрядын кодтың таңба разряды деп атаймыз, ал санның мән разрядтары орналасқан разрядтарын кодтың цифрлық разрядтары деп атаймыз.

Операндтар түрлері және пішімдері

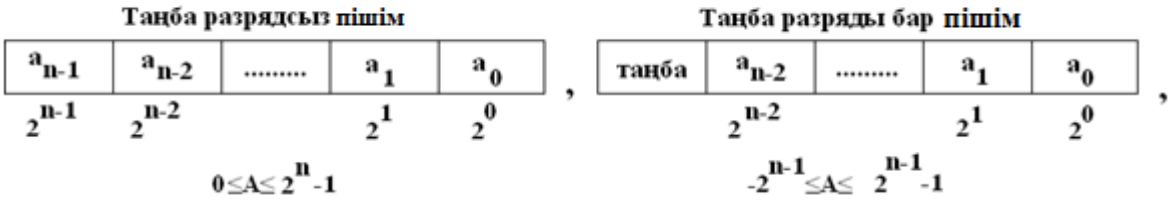
Таңба разряды цифрлық разрядтарының сол жағында орналасады.

Бекітілген үтірлі сандарды екі түрлі пішімде көрсетуге болады: үтір санның таңба разрядынан кейін және үтір санның ең соңғы (кіші) разрядынан кейін. Бірінші түрінде тек дұрыс бөлшекті сандарды көрсетуге болады (4.1-суретте).



4.1-сурет. Бөлшек сандарды бекітілген үтірмен көрсету

Үтірі санның соңғы кіші разрядынан кейін қойылған жағдайда тек бүтін сандар көрсетіледі. Бұл ең көп таралған тәсіл. Мұнда да бүтін сандар таңбасымен не таңбасыз көрсетілуі мүмкін (4.2-сурет).



4.2-сурет. Бүтін санды бекітілген үтірмен көрсету

Қатталған бүтін сандар

Қазіргі процессорларда қатталған бүтін сандарды өңдейтін командалар бар. Мұндай сандар мультимедиялық ақпараттарды өңдегенде орын алады. *Қаттама* деп бірнеше бүтін сандарды ұзындығы үлкен сөзге біріктіруді айтады. Тисті команда арқылы сөздегі сандар параллель орындалады. Мысалы: AVX (Advanced Vector Extensions) технологиясында бүтін сандар 256 разрядты сөзге біріктіріледі.

4.2. Ондық сандар

Кейбір есептерді шығару үстінде ондық сандар массивін өңдеуге тура келеді. Мұндай жағдайда әрбір ондық санды төрт разрядты екілік санмен кодтау тәсілі қолданылады, басқаша айтқанда әр ондық сан екілік-ондық кодпен (BCD – Binary Coded Decimal) алмастырылады.

Әдетте 8421 кодын қолданады. 0,1,2,3,4,5,6,7,8,9 ондық сандарды 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001 екілік комбинацияларымен сәйкес кодталады. Қалған алты (1010, 1011, 1100, 1101, 1110, 1111) комбинациясы санның таңбасын “ - ”, “ + ” және басқа арнайы таңбаларды көрсету үшін қолданылады. Әдетте “ + ” 1100 (C_{16}), ал “ - ” 1101 (C_{16}) комбинациясымен кодталады.

Байт		Байт			Байт	
Цифр	Цифр	Цифр	Цифр	.....	Цифр	Цифр

а

Байт		Байт			Байт	
Зона	Цифр	Зона	Цифр	.....	Зона	Цифр

4.3-сурет. Ондық сандардың пішімдері: а – қатталған пішім; б - зоналық пішім

Есептеу машиналарында ондық сандарды екі түрлі пішімде көрсетеді: қатталған және зоналық пішімдер (4.3-сурет). Екі пішімде де әрбір ондық сан екілік-ондық кодымен (тетрадамен) ауыстырылады. Ең көп таралған – қатталған пішім. Мұндай пішім арқылы сандарды тек сақтап қоймай, олармен әртүрлі операциялар орындауға болады.

Мұндай пішімде сандар байттар тізбегін құрады, әрбір байт екі ондық санның цифрларын құрайды. Кіші байттың оң тетрадасына санның таңбасы жазылады. Ондық сан бүтін байттар санын құрады. Егер мұндай шарт орындалмаса, онда сол жақтағы байттың жоғарғы төрт разряды нольдермен толтырылады. 4.4-суретте - 9865 саны қатталған пішімде көрсетілген .

0	9	8	6	5	минус
0000	1001	1000	1100	1010	1101

4.4-сурет. 9865 санын қатталған пішімде көрсету

Зоналық пішім (4.5-сурет) үлкен IBM 360/370/390 машиналарында қолданылады. Мұнда әрбір цифрларды жазу үшін бір байт бөлінеді. Төменгі төрт разрядтарында цифрлар коды жазылса, ал жоғарғы тетрададағы (зонаға) арнайы “Зона” коды жазылады. “Зона” коды цифрлар және таңба кодасымен сәйкес келмейді. Аталған машиналарда мұндай код – 1111=F_{16}. Ал ондық санның ең кіші разряды жазылған байттың зона өрісінде сан таңбасы жазылады. 4.5-суретте – 9865 санының зоналық пішімде жазылуы көрсетілген.

Байт		Байт		Байт		Байт	
Зона	9	Зона	8	Зона	6	Зона	5
1111	1001	1111	1000	1111	0110	1101	0101

4.5-сурет. 9865 санын зоналық пішімде көрсету

Зоналық және қаттама пішімдерінде таңбаны төменге орналастыру арқылы ондық санды кез-келген ұзындықта көрсетуге мүмкіндік аламыз. Біздер ондық сандарды 8421 кодымен көрсету вариантын қарадық. Төменде кодтаудың басқа вариантары көрсетілген (4.1-кесте).

4.1-кесте. Ондақ цифраларды кодтау варианттары.

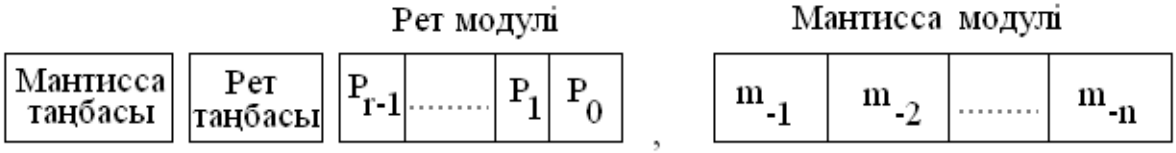
Ондық цифрлар	BCD 8421	Excess-3 (Стибиц коды)	BCD2421 (Айкен коды)	BCD84-2-1
0	0000	0011	0000	0000

1	0001	0100	0001	0111
2	0010	0101	0010	0110
3	0011	0110	0011	0101
4	0100	0111	0100	0100
5	0101	1000	1011	1011
6	0110	1001	1100	1010
7	0111	1010	1101	1001
8	1000	1011	1110	1000
9	1001	1100	1111	1111

4.3. Жылжымалы үтірлі сандар

Жылжымалы үтірлі сандар түрінде көрсеткенде, әрбір сан екі цифрлар тобына бөлінеді. Цифрлардың бірінші тобын мантисса, екінші тобын реті деп атайды. Сан $X = \pm mq^{\pm p}$ түрінде көрсетіледі, мұнда *X*- сан, *m*- санның мантиссасы, р- санның реті, q - санау жүйесінің негізі.

Сандарды жылжымалы үтірлі пішінде көрсету үшін мантисса мен рет таңбаларын және q - санау жүйесіндегі олардың модулін, санау жүйесінің негізін көрсету керек (4.6-сурет). Егер m және р мәндерін өзара өзгертсек онда үтір орны өзгеріп "жылжып" отырады, сондықтан сандардың мұндай көрсетуін *жылжымалы үтірлі сандармен көрсету* деп атаймыз.

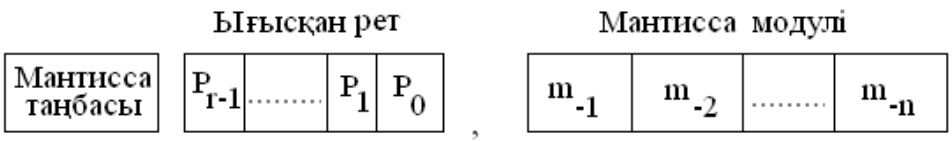


4.6-сурет. Жылжымалы үтірлі сандарды көрсету пішімі

Жылжымалы үтірлі сандардың көрсету ауқымы және дәлдігі санақ жүйесінің негізі және рет пен мантиссаның разрядтылығына байланысты әр түрлі болады. Есептеу машиналарында негіз мәні 2-ден басқа да негіздер қолданылуы мүмкін. Мысалы IBM фирмасының машиналарында негіз мәні 16. Мұндай рет мәні машинада сандардың көрсету ауқымын кеңейтуге мүмкіндік береді.

Көптеген есептеу машиналарында рет сандарымен операция жүргізуді жеңілдету үшін реттер тек оң сандар арқылы өрнектеледі. Ол үшін бастапқы рет мәніне белгілі оң бүтін санын қосамыз. Санның мұндай ретін *ығысқан* немесе *ығыстырылған рет* деп атайды. Әдетте ығысу мәні рет ауқымының

жартысына тең болады. Ығыстырылған рет оның барлық өрісін қамтиды, оның ішіне рет таңбасы да кіреді (4.7-сурет).



4.7 -сурет. Ығыстырылған ретті жылжымалы үтірлі сандар пішімі

Жылжымалы үтірлі сандар әдетте қалыпты пішімде көрсетіледі. Мұндай пішімде мантисса модулі бірден кіші ($|q| < 1$), және үтірден кейінгі цифр нөл болмауы керек. Осы жолмен алынған мантисса *қалыптасқан* деп аталады. Есептеу машиналарында жиі қолданылатын есептеу жүйелері үшін:

екілік: $X = m * 2^p, (1 > |m| \geq 1/2)$;

сегіздік: $X = m * 8^p, (1 > |m| \geq 1/8)$;

он алтылық: $X = m * 16^p, (1 > |m| \geq 1/16)$.

Егер мантиссаның жоғарғы і разрядтары нөлге тең болса, оларды қалыптастыру үшін мантиссаны солға і разрядтарға жылжытамыз және осы разряд санынан рет мәнін азайтамыз. Мұндай операцияны *қалыптастыру операциясы* деп атаймыз. Қалыптастыру операция кезінде сан мәні өзгермейді.

Егер жылжымалы үтірлі санды жазу үшін екілік негіз қолданылса (q=2), онда мантисса дәлдігін өсірудің және бір тәсілі - бірді жасыру тәсілін қолдануға болады. Оны мына мысалдан көруге болады. Қалыптастырылған мантисса 0,101000 (1) болсын дейік. “Бірді жасыру” тәсілін қолдансақ мантисса мәнін былай жазуға болады 0,010001. Соңғы жазуға мантиссаның жоғарғы разрядында жазылған “1”-ді жасырып қалдық та мантиссаның төменгі разрядына бірінші жазудағы сыймай қалған “1”-ді жаздық, яғни мантисса дәлдігін жоғарылаттық. Жасырылған бірлік арифметикалық операция жүргізілер алдында қайта орнына келтіріледі, ал оны жадыға жазарда алып тастаймыз.

4.8-сурет. IEEE 754 стандартының негізгі пішімдері. а – бір дәлдікті пішім; б – қос дәлдікті пішім

Бір процессорда жазылған бағдарламаны басқа процессорларда қолдануды жеңілдету үшін жылжымалы үтірлі сандар пішімі IEEE 754 стандартымен анықталған. Қазір бұл стандарт барлық процессорларда және арифметикалық процессор серіктестерінде кеңінен қолданылады. Стандартта 32-биттік (бір дәлдік) және 64-биттік (қос дәлдікті) пішімдер анықталған (4.8-сурет). Бұларда рет разрядтарының саны - 8 және 11. Есептеу жүйесі – екілік.

Қатталған жылжымалы үтірлі сандар

Жоғарыда айтылған AVX технологиясында жылжымалы үтірлі сандармен көрсетілетін мультимедиялық ақпараттарды өңдеуге қолданылатын арнайы командалар бар. Мұндай командалар арқылы мультимедиялық ақпараттарды өңдеу өнімділігін арттыруға болады. Әрбір команда жылжымалы үтірлі сандармен көрсетілген 32-разрядтың сегіз операндтармен және 64-разрядты 4 операндтармен жұмыс жасайды. Операндтар 256 разрядтық топқа қатталады (4.9-сурет).

255 0

32	32	32	32	32	32	32	32

Бір дәлдікпен қатталған (8*32бит) жылжымалы үтірлі сандар.

255 0

64	64	64	64	64	64	64	64

Қос дәлдікпен қатталған (4*64бит) жылжымалы үтірлі сандар.

4.9-сурет.AVX технологиясындағы жылжымалы үтірлі сандардың қатталу пішімі

Логикалық деректер

Логикалық деректердің элементі - логикалық (буль) айнымалылар. Логикалық айнымалылыр «1» немесе «0» мәндерін қабылдайды. Есептеу машиналары машиналық сөз ұзындығымен анықталатын логикалық айнымалылар жиынтығымен операция орындайды. Мұндай сөздер ЖӘНЕ, НЕМЕСЕ, ЕМЕС, т.с.с. командалар арқылы өңделеді. Әрбір сөз разрядтары жеке-жеке бір уақытта өңделеді.

Қатарлар

Қатарлар деп – бит және байттардың, сөздер және қос сөздердің үздіксіз тізбегін айтамыз. Биттік қатар байттың кез-келген позициясынан басталуы

мүмкін. Биттік қатар 2^{32} биттен тұруы мүмкін. Байттық қатар байттардан, сөзден немесе қос сөзден тұруы мүмкін. Қатар байтының ұзындығы 2^{32}-1 байттардан тұруы мүмкін. Келтірілген цифрлар 32 разрядты есептеу машинасына тән.

Егер байттық қатарлардың байттары символдар коды болса, онда оларды мәтіндік қатар деп атаймыз. Мәтіндік қатар үзінді үлкен ауқымда өзгеретін болғандықтан, оның соңғы байтына тежеуіш кодын жазу керек.

4.4. Символдық ақпараттар

Символдық ақпараттарға әріптер, цифрлар, тыныс белгілері, математикалық т.б. символдар жатады. Әрбір символға белгілі бір екілік комбинация сәйкестендіріледі. Әртүрлі символдар жиыны және оларға сәйкестендірілген екілік кодтарды *кодтау кестесі* деп атаймыз. Қазіргі кезде көптеген кодтау кестелері қолданылады. Олардың қатарына кеңітілген екілік – кодталған код EBCDIC (Extended Binary Coded Decimal Interchange Code) және Американың деректер алмасу стандартты коды ASCII (American Standard Code For Information Interchange) жатады.

EBCDIC коды IBM фирмасының есептеу машиналарының ішкі коды ретінде қолданылады. Ол DKON (деректерді өңдейтін екілік код) атымен де белгілі, жеті разрядты ASCII стандартты коды 8-разряд тақ бүтін жазуға арналған. 7 разрядпен 128 символды (латын әріптерін, цифрлар, негізгі математикалық операциялар символдары және пунктуация белгілерін) көрсетуге болады. ASCII коданың Еуропалық модификациясы latin 1 (стандарт ISO 8859-1) деп аталады. Стандартта барлық 8 разрядтар пайдаланылады. Қосымша комбинациялар (128-255 кодалары) Батыс Еуропа тілдерінде кездесетін арнайы әріптерді, псевдографика символдарын, грек алфавитінің кейбір әріптерін белгілеу үшін және кейбір математикалық т.б. символдар үшін пайдаланылады. Осы кодтық кесте модификациялары барлық елдерде кеңінен қолданылады. Сондықтан ISO 8859 стандарты халықаралық стандартқа жатады.

128-255 қосымша комбинацияларын пайдалануға байланысты ISO 8859 стандартының бірнеше варианттары бар (4.2-кесте).

ASCII коды ыңғайлы болғанмен, ол арқылы көптеген символдарды көрсетуге жеткіліксіз екен. Сондықтан 1993 ж. Apple Computer, Microsoft, HP, Dell және IBM компанияларының концорциумы 16-битті ISO 10646 стандартын қабылдады. Стандарт Unicode атымен белгілі. Жаңа код арқылы “тірі” және “өлі” тілдердің барлық символдарын 16 разрядты екілік сандар арқылы кодтауға мүмкіндік бар.

4.2-кесте. ISO 8859 стандартының варианттары.

Стандарт	Қолдану аясы
ISO 8859-1	Батыс Еуропа тілдері
ISO 8859-2	Орталық және Шығыс Еуропа елдерінің тілдері
ISO 8859-3	Оңтүстік Еуропа, мальт, эсперанто тілдері
ISO 8859-4	Солтүстік Еуропа елдерінің тілдері
ISO 8859-5	Кирилица символды славян елдерінің тілдері
ISO 8859-6	Араб тілі
ISO 8859-7	Қазіргі грек тілі
ISO 8859-8	Еврит және идиш тілдері
ISO 8859-9	Түрік тілі
ISO 8859-10	Солтүстік еуроппа елдерінің (Лапландия, Исландия) тілдері
ISO 8859-11	Тай тілі
ISO 8859-13	Балтық елдерінің тілдері
ISO 8859-14	Келт тілі
ISO 8859-15	Европа тілдерінің комбинациялық кестесі
ISO 8859-16	Әр түрлі Албан, Хорват. Ағылшын, Фин, Француз, Неміс, Венгер, Ирланд, Итальян, Поляк, Румын, Славьян тілдерінің өзгеше символдары үшін

Бақылау сұрақтары.

1. Бекітілген үтірлі сандарда бүтін екілік сандарды көрсету үшін үтір санның қай разрядынан кейін қойылуы керек?
2. Бекітілген үтірлі сандарды көрсету ауқымы немен анықталады?
3. Қатталған бүтін сандар не үшін керек?
4. Ондық сандар қандай пішімдерде көрсетіледі?
5. Стибиц кодын 8421 кодынан қалай алуға болады?
6. Жылжымалы үтірлі сандарда рет саны нені көрсетеді?
7. Жылжымалы үтірлі сандар пішімінде ығыстырылған рет не үшін енгізілген?

8. Жылжымалы үтірлі сандар пішімінде қалыптастыру операциясы не үшін керек?
9. EBCDIC коды мен ASCII кодаларының айырмашылығы неде?

5. ЕСЕПТЕУ МАШИНАЛАРЫ МЕН ЖҮЙЕЛЕРІНІҢ КОМАНДАЛАРЫ ЖӘНЕ МЕКЕНДЕУ ТӘСІЛДЕРІ

5.1. Есептеу машиналарының командалары

Есептеу машиналарының командалары мынадай түрлерге бөлінеді:

- деректер алмасу командалары;
- арифметикалық және логикалық операциялар командалары;
- ЖККД командалары;
- қатарларды өңдеу командалары;
- түрлендіру командалары;
- енгізу/шығару командалары;
- командалар ағымын басқару командасы.

Деректер алмасу командалары

Деректер алмасу үшін командада мынадай ақпараттар болуы керек:

- операндтарды беруші (ақпарат көзі) және алушы жақтардың мекен-жайлары – жады ұяшығының мекен-жайы, процессор регистрінің нөмірі немесе операндтың стекте орналасқандығы туралы деректер;
- алмасатын дерек ұзындығы (байтпен не сөз санымен көрсетіледі);
- әр операндтарды мекендеу тәсілдері (ол арқылы команданың мекен-жай бөлігінен операндтың физикалық мекен-жайы анықталады).

Алмасу командалары арқылы процессор мен негізгі жады, процессор ішінде және жады ұяшықтары арасында деректер алмасады. Процессор ішіндегі алмасуда командалар «регистр-регистр» пішімінде жүргізіледі. Ал процессордың жадымен алмасуында команда «регистр-жады» пішімінде, жады ішінде – «жады-жады» пішімінде жүргізіледі.

Арифметикалық және логикалық өңдеулер командалары

Бұл топ командалары арқылы арифметикалық және логикалық өңдеулерді әртүрлі пішімдерде орындау қамтамасыз етіледі.

Арифметикалық және логикалық операцияларды орындап, нәтиже алумен қатар арифметикалық және логикалық құрылғыларда нәтиже белгілері (жалаушалар) қалыптасады. Оған мынадай белгілер жатады: Z (Zero) – нөлдік нәтиже; N (Negative)–теріс нәтиже; V (Verflow) – аса толу; C (Carry) – тасымал.

Бүтін сандармен операция орындау

Бүтін сандар бекітілген үтірмен көрсетіледі. Олармен орындалатын стандартты операциялар жиынтығына мыналарды жатқызуға болады:

- екі орынды арифметикалық операциялар (екі операндтармен орындалатын операциялар: қосу, азайту, көбейту және бөлу);
- бір орынды арифметикалық операцияға (бір операндты операция) операндтың абсолют мәнін (модулін) табу, операнд таңбасын өзгерту жатады;
- салыстыру операциясы (бүтін екі санды салыстыру). Салыстыру нәтижесінде олардың арақатынас мәнін табу (=, <>, >, <, <=, >=).

Және де бұл операцияларға бүтін саннан қалдық табу, тасымалды ескеру, операнд мәнін бірге өсіру (инкремент) сияқты операциялар да жатады.

Жылжымалы үтірлі сандармен операция орындау.

Жылжымалы үтірлі сандармен жұмыс жасау үшін мынадай операциялар қолданылады:

- негізгі арифмтикалық операциялар: қосу, азайту, көбейту және бөлу;
- «=,<>, >, <, <=, >=« белгілерін анықтау үшін салыстыру операциялары.

Логикалық операциялар

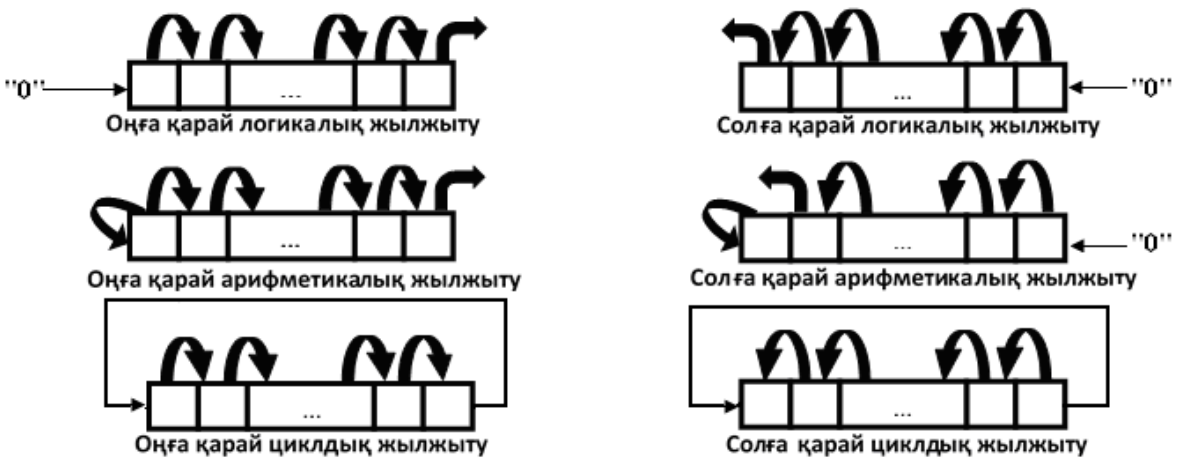
Логикалық операцияны орындайтын командаларға «ЕМЕС», «ЖӘНЕ», «НЕМЕСЕ» және «ЕКІЛІК МОДУЛЬ АРҚЫЛЫ ҚОСУ» командалары жатады. Оларға сөздің жеке биттерімен немесе басқа мекен-жай бірліктерімен логикалық операциялар жүргізіледі.

Жылжыту операциялары

Жылжыту операциясы – жылжыту командалары арқылы деректердің барлық разрядтарын оңға немесе солға қарай логикалық, арифметикалық және циклдік жылжытуларды білдіреді (5.1-сурет).

Логикалық жылжытуда дерек сөзінің барлық разрядтары оңға не солға жылжиды. Разряд сеткасынан шығып кеткен биттер жоғалады, босаған позиция нөлмен толтырылады.

Арифметикалық жылжытуда дерек таңбасы бар бүтін сан ретінде қаралады. Жылжыту кезінде таңба мәні өзгермейді. Оңға қарай жылжыту кезінде босаған позиция таңбаның разряд мәнімен толтырылады, ал солға қарай жылжытылғанда нөлдермен толтырылады.



5.1-сурет. Жылжыту операцияларының түрлері

Циклдік жылжытуларда сөздің барлық разрядтары жылжытылады. Сөз разрядтарынан шығып кеткен разрядтар қарама-қарсы жақтағы босаған позицияға жазылады.

Ондық сандар операциялары

Ондық сандар ЕМ екілік сандармен кодталып көрсетіледі. IBM 360/370/390 машиналарында мұндай сандарды өңдеу үшін арнайы командалар арқылы қосу, алу, көбейту, бөлу операциялары орындалады. Қазіргі басқа машиналарда ондай командалар жоқ. Ондай амалдар бүтін санды арифметикалық командалар арқылы имитациялап, алынған нәтижені түзету арқылы жүзеге асырылады.

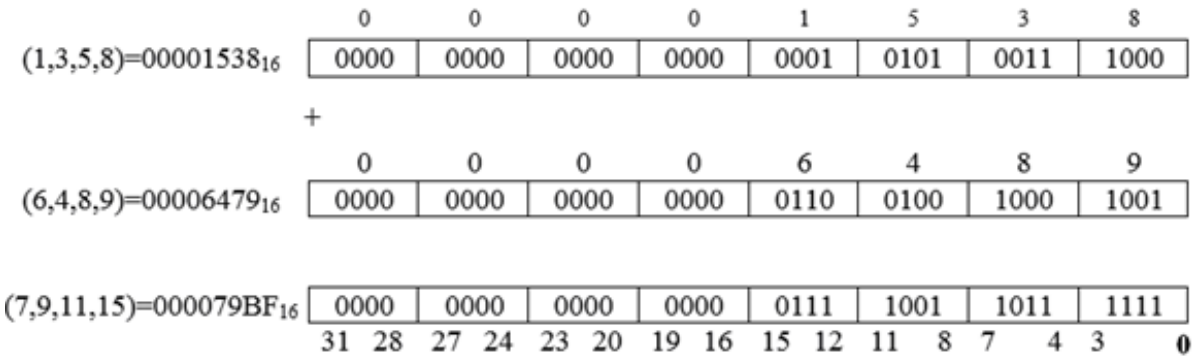
ЖККД командалары

Мұндай команда типтері *жеке команда ағымы – көп деректер ағымы* дегенді көрсетеді. Әдетте команда екі санмен операция орындаса, ЖККД командалары сандардың екі тобымен операция орындайды. Мұндай командалардың операндтары операцияға қаттама пішімдерінің бірінде көрсетіледі.

ЖККД командалары алғашқы рет Pentium MMX 57 микропроцессорының командалар жүйесіне кіреді. Мұндағы MMX (MultiMedia Extention) – мультимедиалық кеңейту. MMX командалары қатталған бүтін сандарды параллель өңдеуге мүмкіндік береді. Арифметикалық операцияларды орындау кезінде топқа кіретін әрбір сан көрші сандармен байланыссыз жеке дара сан ретінде қаралады. MMX командалары өңделетін деректің ерекшеліктіктеріне байланысты қаныққан арифметика деп аталатын арифметика қолданылады. Қаныққан арифметикада, егер қосу кезінде нәтиже оған тиісті позициядан

шығып кетсе, онда нәтиже сол позицияға сыятын үлкен екілік санымен алмастырылады.

5.2-суретте 32 разрядты пішімде қатталған екі төрт разрядты сандар тобын қосу операциясының орындалуы көрсетілген.



5.2-сурет. Қатталған бүтін сандарды қанықкан арифметика арқылы қосу

Қазіргі есептеу машиналары командалары бар ЖККД командалар жиынтығы SSE – Streaming SIMD Extension («бір команда-көптеген дерек» принципімен өңделетін ағын) деген аббревиатурамен белгілі. 2008 жылы жаңа стандарт – Advanced Vector Extension (AVX) дүниеге келді. AVX стандарты SSE командалар жиынтығының 256 разрядтық кеңейтілген түрі. ЖККД командалар жиынтығы IBM фирмасының Power PC сериясында Alti Vec деген атпен іске қосылған.

Қатарлармен жұмыс жасайтын командалар.

Мұндай командалармен қатарларды қозғау, салыстыру және оларды іздеу операциялары орындалады.

Командаларды түрлендіру

Мұндай командаларды деректерді көрсету пішімдерін өзгерту үшін қолданады. Оған мысалы, ондық сандарды екілік жүйеге түрлендіру немесе 8 разрядты символдарды ASCII кодынан EBCDIC кодына, не кері түрлендіру операциялары жатады.

Енгізу/шығару командалары

Енгізу/шығару командаларының қатарына енгізу/шығару кезінде сырт құрылғыларды басқару командалары, сырт құрылғылардың күйін тексеретін командалар жатады.

Сырт күйін тексеру командалары арқылы оның енгізу/шығару операциясына дайындығын тексереді. Егер байланысқан сырт құрылғысы дайын болмаса, қандай себептермен дайын еместігін (белгілерін) анықтайды.

Оған: «қорек көзі қосылмаған», «алдыңғы енгізу/шығару операциясы анықталған жоқ», т.б. белгілер жатады. «Құрылғы дайын» деген белгіні алған соң процессор жаңа операцияны бастайды. Сырт құрылғылармен дерек алмасу енгізу/шығару командаларымен орындалады. Енгізу командасы енгізу/шығару модуліне дерек элементін (байт немесе сөз) сырт құрылғыдан алып оны дерек шинасына беру командасын шығарады. Ал шығару командасы енгізу/шығару модуліне дерек шинасына алып, оны сырт құрылғысына жіберуге бұйрық береді.

Команда ағындарын басқару

Бағдарламаларда келесі команда мекен-жайын команда санауыш кодын ағынды команда кодына өсіру арқылы немесе команда санауышына басқа мекен-жай кодын жүктеу арқылы анықталады. Бірінші тәсілде командалар табиғи тәртіппен (жады құрылғысына жазылу тәртібімен) орындалады. Екінші тәсілде табиғи тәртіп бұзылады: басқару арнайы команда арқылы бағдарламаның басқа нүктесіне беріледі. Мұндай командаларда оның мекен-жай бөлігінде өту нүктесінің мекен-жайы (келесі орындалатын команда мекен-жайы) жазылады.

ЕЖ командалар жүйесінде есептеу ретін өзгертуге алып келетін командалардың үш түрін көрсетуге болады. Олар:

- шартсыз өту;
- шартты өту;
- процедураны шақыру және процедурадан қайта оралу.

Шартсыз өту командасында көрсетілген мекен-жайға ешбір шартқа тексерілмей өтеді.

Шартты өту командасында, командада көрсетілген мекен-жайға тек белгілі шарт орындалғанда өтеді. Шарт орындалмаса, онда келесі реттегі команда орындала береді. Шарт ретінде алдыңғы орындалған арифметикалық немесе логикалық операциялардың нәтижелерінен қалыптасатын белгілер жатады. Әрбір белгілер процессордағы белгілер регистрінің разрядтарында тіркеледі. Барлық бағдарламалау тілдерінде процедура механизмдері кеңінен қолданылады.

Процедураны шақыру командасы бағдарламаның кез-келген нүктесінен процедураның бастапқы командасын орындауға өтуді қамтамасыз етеді. Процедурадан қайта оралу командасы шақыру командасынан кейін орналасқан қайта оралу нүктесіне орналасуын басқарады.

5.2. Команда пішімдері

Кез-келген команда екі бөліктен: операциялық кодтар жазылатын операциялық бөлігінен (ОПК) және мекен-жай бөлігінен (МЖБ) тұрады. Операция бөлігінде орындалатын операция түрі, ал мекен-жай бөлігінде операцияға қатысатын операнд (немесе операндтар) мекен-жайы (мекен-жайлары) және нәтиже жазылатын мекен-жайлары көрсетіледі.

Қазіргі есептеу машиналарында операцияларға қатысатын операнд мекен-жайы командасында көрсетілген мекен-жайды түрлендіру арқылы анықтайды. Ол үшін әртүрлі мекендеу тәсілдерін (МТ) команданың арнайы өрісінде көрсетеді.

Команданың жалпылама пішімі 5.3-суретте көрстілген.

ОПК	МТ	Мекен-жай бөлігі (МЖБ)

5.3-сурет. Команданың жалпылама пішімі

Команданың жалпы ұзындығы Rk төмендегі формуламен анықталады:

$$R_k = \sum_{i=1}^{l} R_{M_i} + R_{\text{ОПК}} + R_{\text{МТ}}$$

Мұндағы l – команда мекен-жайының саны; R_{M_i} – i-ші мекен-жай разряд саны; $R_{\text{ОПК}}$ – операциялық код өрісінің разряд саны; $R_{\text{МТ}}$– мекендеу тәсілі өрісінің разряд саны.

$$R_{\text{ОПК}} = [log_2 N_{\text{ОПК}}]$$

Мұндағы $N_{\text{ОПК}}$ – командалар жүйесіндегі операция саны; R_{M_i} және $R_{\text{МТ}}$ төмендегі формулалармен анықталады.

$$R_{M_i} = [log_2 N_i] \text{және} R_{\text{МТ}} = [log_2 N_{\text{МТ}}]$$

Мұндағы

N_i – i-мекен-жай арқылы қатынас құруға болатын жады ұяшықтарының саны;

$N_{\text{МТ}}$ – мекендеу тәсілдерінің саны.

Командаларда мекен-жай санына байланысты төрт мекен-жайлы, үш, екі, бір, бір жарым мекен-жайлы командалар пішімідері болады.

Әртүрлі мекен-жайлы команда пішімдері 5.4-суретте көрсетілген:

Операция	Мекен-жай			
ОПК	1-операнд	2-операнд	Нәтиже	Келесі команда

а

Операция	Мекен-жай		
ОПК	1-операнд	2-операнд	Нәтиже

б

Операция	Мекен-жай	
ОПК	1-операнд	2-операнд/нәтиже

в

Операция	Мекен-жай
ОПК	1 немесе 2-операнд

г

Операция	Мекен-жай	
ОПК	регистр	1-операнд

д

5.4-сурет. Команда пішімдері. а - төрт мекен-жай; б - үш мекен-жай; в - екі мекен-жай; г - бір мекен-жай; д - бір жарым мекен-жай пішімдері

Суреттен 4,3,2 мекен-жайлы команда пішімдерінан әрбір мекен-жай өрісінің атқаратын қызметі көрініп тұр.

Бір мекен-жайы бар команда пішімін қолданғанда бірінші операнд орталық процессордың құрамында болатын аккумуляторда сақталады. Операция аяқталған соң нәтиже аккумуляторда сақталады.

Бір мекен-жайы бар команда пішімінде 1-операнд және нәтиже сақтау үшін аккумулятордың орнына процессор регистрлерінің бірін қолдануға болады. Процессор регистрлерінің саны аз болғандықтан регистр мекен-жайының ұзындығы қысқа болады. Сондықтан мұндай мекен-жай пішімін бір жарым мекен-жайы бар команда пішімі деп атайды.

5.3. Мекендеу тәсілдері

Есептеу машиналарының мекен-жай кеңістігінде орналасқан операнд орнын анықтау үшін команда пішімінде арнайы өріс көрсетіледі. Осы өріс арқылы операндтардың әртүрлі мекендеу тәсілдері МТ туралы дерек беріледі. Мекендеу тәсілдерін қараудан бұрын «орындаушы мекен-жай» ($МЖ_О$) және «командалық мекен-жай» ($МЖ_к$) коды ұғымдарына тоқталамыз. *Операндтардың орындаушы мекен-жайы* деп жады құрылғысының операндтарды қабылдайтын не операндтарды сыртқа беретін ұяшық нөмірін айтады. Бұл код есте сақтау құрылғысының (ЕСҚ) мекен-жай кірісіне беріледі және осы код арқылы нақты көрсетілген ұяшыққа қатынас құрады. Егер операнд негізгі жады емес, процессор регистрінде сақталса, онда оның орындаушы мекен-жайы сол регистрдің нөмірі болып табылады.

Команданың мекен-жай коды деп команданың мекен-жай өрісіндегі екілік кодты айтады. Мекен-жай коды арқылы операндтың орындаушы коды қалыптасады. Ол үшін әртүрлі мекендеу тәсілдері қолданылады. Мекендеу тәсілдері деректерді өңдеу үдерістері параметріне ықпалын тигізеді. Бір тәсіл команда ұзындығын өзгертпей жады сыйымдылығын өсіруге алып келсе, басқасы - дерек массивімен өңдеуді жеделдетуге алып келеді, үшінші - ішкі бағдарламамен жұмыс жасауды жеңілдетеді т.с.с.

Қазіргі есептеу машиналарында әртүрлі мекендеу тәсілдері қолданылады. Төменде жиі қолданылатын тәсілдерді қараймыз.

Тікелей мекендеу (ТМ)

Тікелей мекендеу команданың мекен-жай өрісінде мекен-жай орнына операндтың өзі жазылады (5.5-сурет).

команда

ОПК	МТ	операнд

5.5-сурет.Тікелей мекендеу

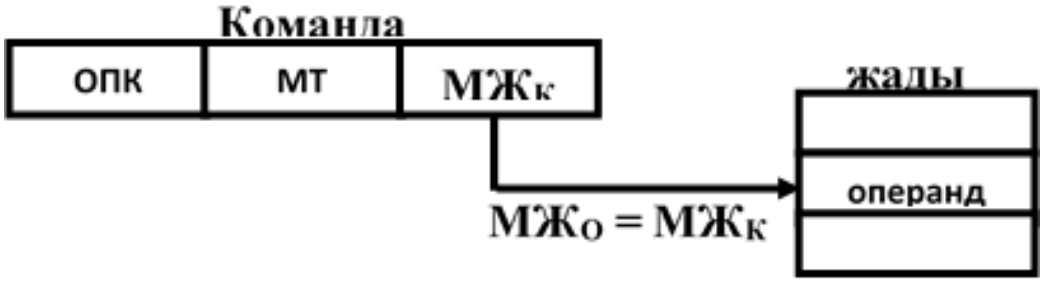
ТМ арифметикалық операциялар орындау, салыстыру операциясын орындау және тұрақты сандарды регистрге жүктеу үшін қолданылуы мүмкін.

ТМ команданы орындау уақытын азайтады, өйткені жадыға қатынас құрудың керегі болмайды. Операндты сақтау үшін жады ұяшығы керек емес.

Тура мекендеу (ТКМ)

Тура мекендеу тәсілінде мекен-жай коды қатынас құратын жады құрылғысының ұяшық нөмірін көрсетеді. (5.6-сурет)

Мұндай мекендеу тәсілінде $МЖ_О = МЖ_К$



5.6-сурет.Тура мекендеу

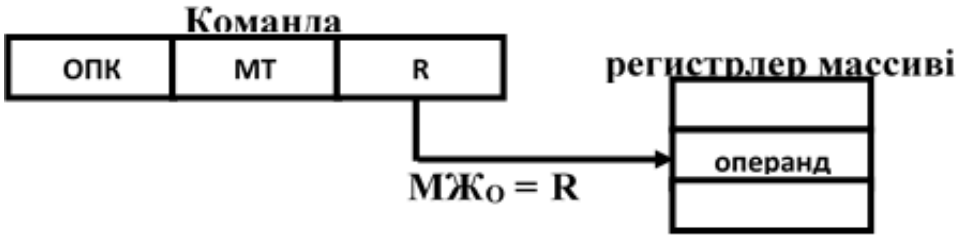
Жанама мекендеу (ЖМ)

Тікелей мекендеу тәсілінің кемістігін жоюдың бір түрі - жанама мекендеуде шектеулі мекен-жай өрісі арқылы операндтың толық разрядты мекен-жайын көрсететін ұяшық нөмірі көрсетіледі. (5.7-сурет).

5.7-сурет. Жанама мекендеу

Регистрлік мекендеу (РМ)

Регистрлік мекендеу тура мекендеуге ұқсас. Мұнда команданың мекен-жай өрісі жады құрылғысының ұяшық нөмірінің орнына процессор регистрінің нөмірін көрсетеді. (5.8-сурет). Регистр мекен-жайын R әріпімен белгілейміз.



5.8-сурет. Регистрлік мекендеу

Артықшылығы - мекен-жай өрісінің қысқалығы және жады құрылғысына қатынас құрылмауы.

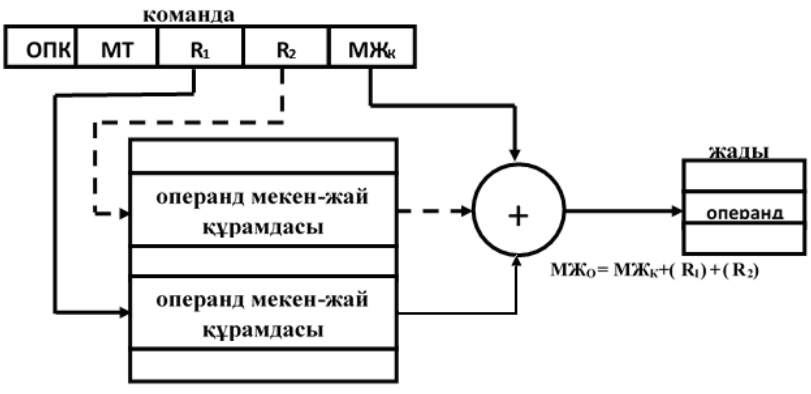
Жанама регистрлік мекендеу (ЖРМ)

Жанама регистрлік мекендеу тәсілінде негізгі жадыда сақталған операнд мекен-жайы командалық мекен-жай (R) арқылы берілген процессор регистрінің бірінен оқылады (5.9-сурет).

5.9-сурет. Жанама регистрлік мекендеу

Ығыстырмалы мекендеу (ЫМ).

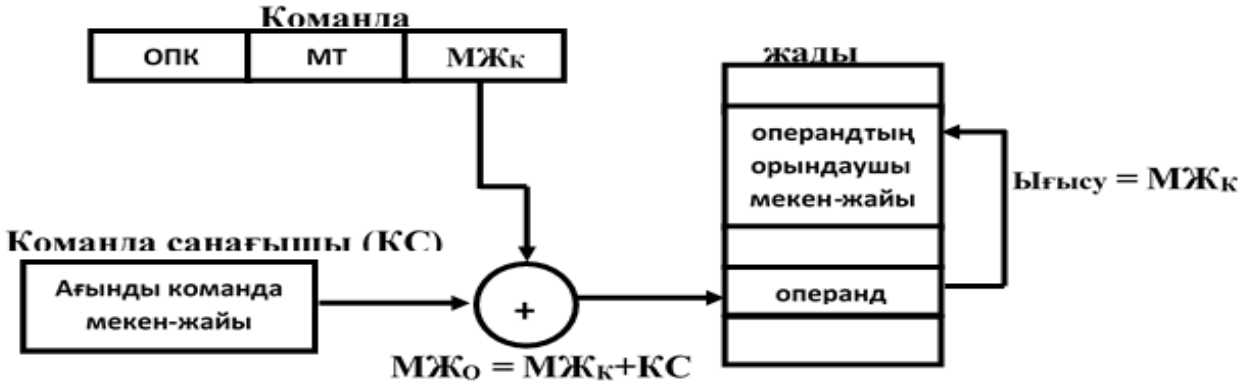
Ығыстырмалы мекендеу тәсілінде орындаушы мекен-жай ($МЖ_О$) команданың мекен-жай өрісіндегі мекен-жай ($МЖ_К$) процессор регистрінің біріндегі немесе бірнешеуіндегі кодаларды қосу арқылы қалыптасады (5.10-сурет). Ығыстырмалы мекендеуде команданың мекен-жай бөлігі бір өрістен тұрып, ол базалық мекен-жай құруы мүмкін.



5.10-сурет. Ығыстырмалы мекендеу

Салыстырмалы мекендеу (СМ)

Салыстырмалы мекендеуде операндтың орындаушы мекен-жайы команданың мекен-жай кодын команда санауышы кодымен қосу арқылы қалыптасады. (5.11-сурет). Сонымен командадағы мекен-жай коды ағындағы команда мекен-жайына қарағанда ығысу болып табылатынын байқау қиын емес.



5.11-сурет. Салыстырмалы мекендеу

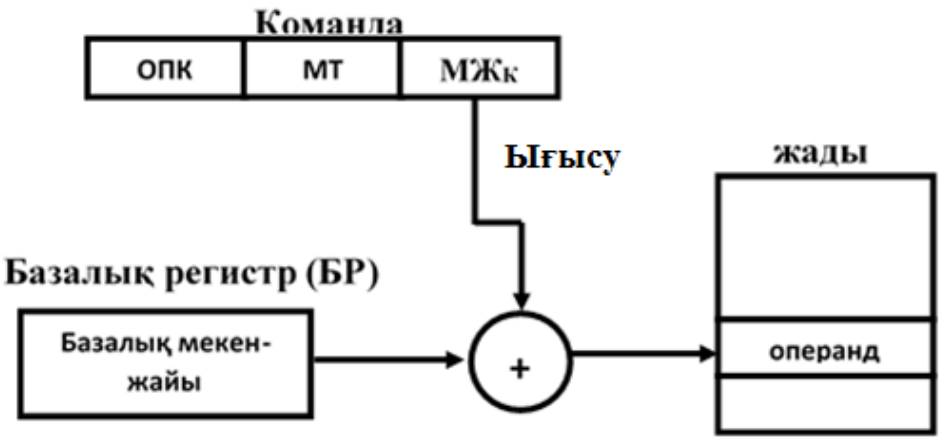
Салыстырмалы мекендеудің арқасында бағдарламаны жады құрылғысының кез-келген жеріне «қозғауға» болады.

Базалық регистрлерімен мекендеу (БРМК)

Базалық регистрлерінде (БР) толық разрядты мекен-жай сақталады. Команданың $МЖ_К$ өрісінде БР мекен-жайынынан ығысу мәні сақталады. Орындаушы мекен-жай $МЖ_О=МЖ_К+БР$ (5.12-сурет).

Базалық регистр ретінде жалпы мақсатты регистрлердің бірін пайдалануға болады. Мұндай жағдайда команда өрісінде регистрдің мекен-жайын (R) көрсету керек.

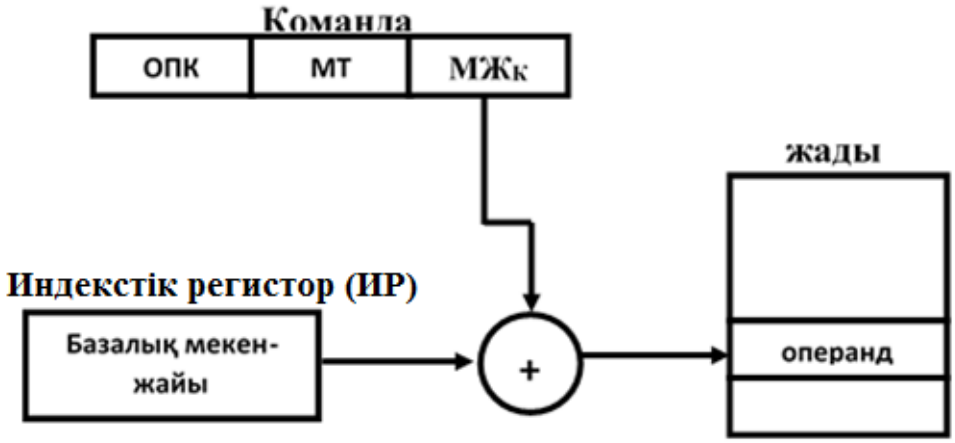
Базалық регистрмен мекендеу тәсілі деректер массивінің элементтеріне қатынас жасау үшін қолданылады. Базалық регистрге массивтің бастапқы мекен-жайы жазылады, ал массивтің элемент мекен-жайы $А_К$ өрісі арқылы беріледі. Ол массивтің бастапқы мекен-жайына қарағанда ығысу болып табылады.



5.12-сурет. Базалық регистрмен мекендеу

Индекстік мекендеу (ИМ)

Индекстік мекендеу $МЖ_К$ өрісі жады ұяшығының мекен-жайын, ал регистр - осы мекен-жайдан ығысу мәнін сақтайды (5.13-сурет).



5.13-сурет. Индекстік мекендеу.

Индекстік мекендеу интерактивтік есептеулерді орындауда кеңінен қолданады.

Беттік мекендеу (БМ)

Беттік мекендеуде жады мекен-жай кеңістігі беттеріне бөлінеді. Беттер бастапқы мекен-жайлармен анықталады. Олар база болып табылады. Беттің жоғарғы разрядтары арнайы регистрде - бет мекен-жайының регистрінде (БМЖРг) сақталады. $МЖ_К$ кодында бет ішіндегі ығысу коды көрсетіледі. Ол орындалатын мекен-жайдың кіші бөлігі болып табылады. $МЖ_О$ бет мекен-жайының регистрінен оқылатын кодамен конкатенация (біріктіру) жолымен қалыптастырамыз (5.14-сурет).

5.14-сурет. Беттік мекендеу

Блоктық мекендеу.

Блоктық мекендеу жадының көршілес ұяшықтарда орналасатын, деректерді өңдеу өлшемдері блогымен анықталатын командаларда қолданылады. Мұндай тәсіл сырт есте сақтау құрылғыларымен және векторларымен операция жасауға өте ыңғайлы.

5.4. Командалар циклі

ЕМ-де бағдарлама оның құрамындағы командалар арқылы орындалады. Командаларды негізгі жадыдан оқу (іріктеу) және оларды орындау әрекеттері бірнеше кезеңнен тұрады. Олар:

- команданы іріктеу (КІ);
- команданы декодтау (КД);
- орындаушы мекен-жайын анықтау (есептеу) (ОА);
- операндтарды іріктеу (ОІ);
- операцияны орындау (ОО);
- нәтижені жазу (НЖ);
- келесі команда мекен-жайын қалыптастыру (КМЖҚ).

Аталған командалар кезеңдерін команданың стандартты циклі деп атайды. Командалар түріне (типіне) байланысты кез-келген командаларда аталған кезеңдер болуы шарт емес. Дегенмен іріктеу, декодтау, орындау және келесі команданың мекен-жайын қалыптастыру міндетті түрде орындалады.

Енді әр кезеңге қысқаша тоқталамыз.

Команданы іріктеу. Кез-келген команда циклінде орталық процессор жадыдан команданы команда санауышындағы мекен-жайға байланысты іріктейді (оқиды). Команда коды (екілік код) команда регистріне орналасады.

Көптеген машиналарда бір команда жадының бірнеше ұяшықтарында орналасуы мүмкін. Мұндай жағдайда команда ұзындығы команда кодының бірінші сөзінде көрсетіледі.

Команданы декодтау. Команданы іріктеп команда регистріне орналастырып болған соң, ол дешифратор арқылы кері кодталады (декодаланады).

Декодтау нәтижесінде:

- команда регистрінде орналасқан команда коды толық па? Егер толық болмаса қалған команда сөзін іріктейміз;
- команданы орындау үшін кезекті қандай әрекеттер жасау керек;
- команданы орындауға қатысатын операндтар қай жадыдан алынуы керек (регистр нөмірі немесе негізгі жады ұяшығының мекен-жайы);
- алынған нәтиже қайда берілуі керек.

Орындалатын мекен-жайын анықтау. Әдетте команданың мекен-жай бөлігінде орындалатын мекен-жай кодына сәйкес келе бермейтін командалық мекен-жай көрсетіледі. Сондықтан команданы орындау үшін мекен-жайлық код алдын ала орындаушы мекен-жайға түрлендірілуі керек. Түрлендіру тәсілі команда өрісінде көрсетілген мекендеу тәсілдерімен анықталады. Мысалы, индекстік мекен-жай бөлігінде индекс мекен-жайындағы кодты қосу керек.

Операндтарды іріктеу. Жоғарғы кезеңде анықталған орындалатын мекен-жай арқылы операнд негізгі жадыдан оқылып процессор регистрінің біріне орналастырылады.

Операцияларды орындау. Қаралып отырған кезеңде командада көрсетілген операция орындалады.

Нәтижені жазу. Нәтиже процессор регистрінің біріне, не негізгі жады ұяшықтарының біріне жазылады.

Келесі команда мекен-жайын қалыптастыру. Егер келесі орындалатын команда орындалып жатқан команданың шектес ұяшығында орналасса, онда команда мекен-жайы команда мекен-жайының кодын бірге өсіру арқылы анықталады. Олай болған жағдайда келесі команданың мекен-жайын команда санағына команданың мекен-жай бөлігін орналастыру арқылы қостастырамыз.

Бақылау сұрақтары

1. Процессор ішіндегі алмасу қандай пішімде жүргізіледі?
2. Нәтиже белгілеріне қандай деректер жатады?
3. Екі орынды арифметикалық операцияларға қандай операциялар жатады?
4. Логикалық операцияларды орындайтын командаларды атаңыз.
5. Логикалық жылжыту операцияларынан арифметикалық жылжыту операцияларының өзгешелігі неде?
6. ЖККД командаларының ерекшеліктері неде?
7. Қанықкан операцияның орындалу ерекшелігі қандай?
8. Қоманда ағымдарын басқаратын командаларға қандай командалар жатады?
9. Команданың жалпылама пішімі неше өрістен тұрады?
10. Тікелей мекендеу тәсілдерін қолдану арқылы не ұтамыз?
11. Деректер массивтерінің элементтеріне қатынас құру үшін қандай мекендеу тәсілін қолдану тиімді?
12. Бағдарламаны жады құрылғыларының кез-келген жеріне «қозғау» үшін қандай мекендеу тәсілін қолданамыз?

6. ОПЕРАЦИЯЛЫҚ ҚҰРЫЛҒЫЛАР

Есептеу машиналарында операциялар әртүрлі деректермен әртүрлі операциялық құрылғыларда (ОПҚ) орындалады. ОПҚ өңделетін операндтар типтеріне байланысты

- Бекітілген үтірлі сандар ОПҚ;
- Жылжымалы үтірлі сандар ОПҚ;
- Деректерді логикалық өңдеу ОПҚ;
- Ондық сандармен амалдар орындалатын ОПҚ деп бөлінеді.

Әдетте ондық деректер,бекітілген үтірлі сандар ОПҚ-да орындалады.

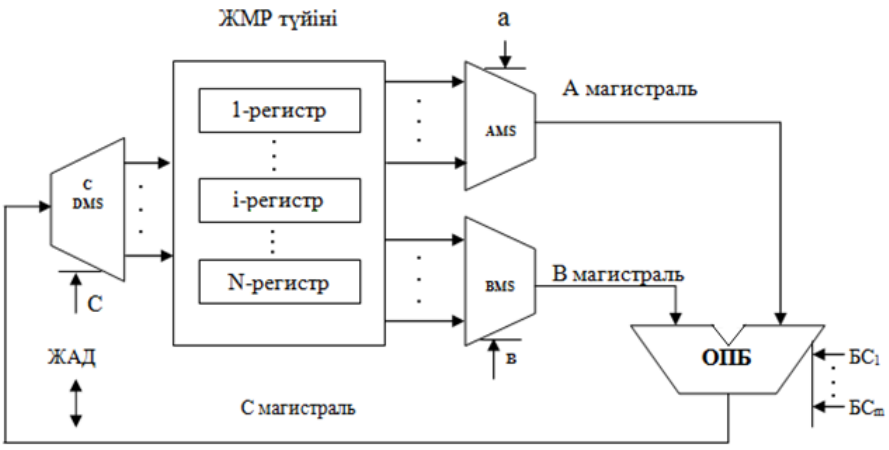
ОПҚ әртүрлі типтік түйіндерден тұрады. Оларға операндтарды уақытша сақтайтын регистрлер, деректерді беруге арналған шиналар, комбинациялық сұлбалар: қосындылауыштар, салыстыру сұлбалары, логикалық операция орындайтын логикалық сұлбалар т.с.с. жатады. Бұлардың барлығында басқару құрылғысының микрокомандаларымен әртүрлі микрооперациялар орындалады.

Операциялық құрылғыларды құрылымдық ұйымдастыруына байланысты екі түрге – «қатаң құрылымды ОПҚ және магистральді ОПҚ деп бөлуге болады.

Қатаң құрылымды ОПҚ-да әрбір типтік түйінге жеке комбинациялық сұлбалар бекітіледі, солар арқылы типті микрооперациялар орындалады. Мұндай құрылымдарда сұлба біркелкілігі өте төмен, сондықтан оларды үлкен интегралдық сұлбада іске асыру қиындық туғызады.

Магистральді ОПҚ-ның барлық ішкі регистрі бір түйінге – жалпымақсатты регистрге (ЖМР) біріктіріледі, ал комбинациялық сұлбалар – операциялық блокқа (ОПБ) біріктірілген. ОПБ пен регистрлар бір-бірімен магистраль арқылы байланысқан, сондықтан мұндай ОПҚ магистральді деп аталады.

Магистральді ОПҚ-ға мысал 6.1-суретте келтірілген:



6.1-сурет. Магистральді ОПҚ

ЖМР түйіні N регистрлерден тұрады. Олар А және В магистральдарымен мультиплексорлар (AMS және BMS) арқылы қосылған. ЖМР кез келгені AMS және BMS арқылы А және В магистраліне қосыла алады. Қосылатын регистрлер нөмірі а және в мекен-жайларымен анықталады. Олар AMS немесе BMS мультиплексорларының мекен-жайлық кірістеріне беріледі.

А және В магистральдері арқылы операцияға қатысатын операндтар ОПБ кірістеріне беріліп, олармен әртүрлі микрооперациялар басқару сигналдары ($БС_i$) арқылы орындалады. Сонымен ОПБ кез келген микрооперация ЖМР-де сақталған кез келген операндтармен орындалады. Микрооперациялар нәтижесі С магистралі және С демультиплексоры (CDMS) арқылы ЖМР біріне қабылданады.

Магистральді ОПҚ негізгі артықшылығы – әмбебаптығы және оның құрамындағы сұлбалардың біркелкілігі өте жоғары. Бұл оларды интегралды сұлбалар кристаллдарында орналастыруды жеңілдетеді. Магистральді ОПҚ қазіргі ЕМ-де өте жиі қолданылады. Магистральді ОПҚ оның құрамындағы магистраль санымен, жалпымақсатты регистрлерді ұйымдастыру тәсілімен және операциялық блоктың түріне байланысты бірнеше түрге бөлінеді.

Магистральдер деректерді беру бағытына байланысты бір бағытты және екі бағытты болады. Бір бағытты магистральдарға деректер тек бір бағытта бере алады, екіншісінде деректер екі жаққа кезекпен беріледі.

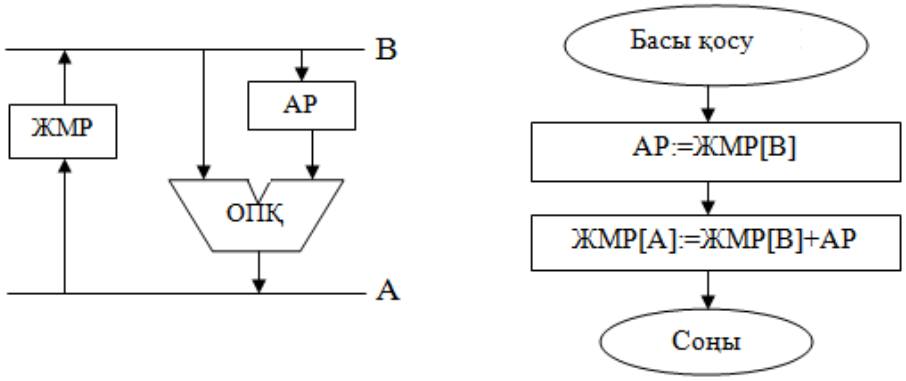
Үш магистральді ОПҚ 6.2-суретте көрсетілген. Мұнда «қосу» тәрізді операциялық микробағдарламасы да келтірілген.

а б

6.2-сурет. Үшмагистральді ОПБ: а-құрылымы; б-қосу микрооперациясы

Үшмагистральді ОПҚ қосу сияқты операция бір тактта орындалады, яғни бір тактта операндтар ЖМР-лерден оқылып А және Б магистральдары арқылы ОПБ беріледі, операция орындалады. Нәтиже С магистралі арқылы ЖМР біріне жазылады. Негізгі кемшілігі: магистральдар интегралды микросұлбада көп орын алады.

Екі магистральді ОПБ микросұлба кристаллында аз орын керек еткенімен, ОПБ құрамына аралық регистр (АР) керек етеді. АР операция орындалар алдында операндтардың бірін уақытша сақтайды, операция мұндай жағдайда екі тактта орындалады: бірінші тактта операндтарының бірі ЖМР оқылып, АР жазылады, екінші тактта операция ЖМР-ден оқылған екінші операндымен АР сақталған операндпен операция орындалып, нәтиже ЖМР біріне жазылады (6.3-сурет).

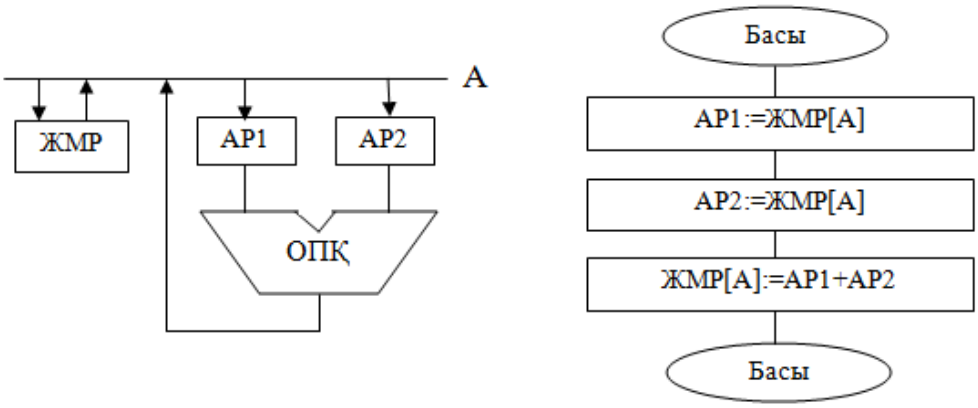


а б

6.3-сурет. Екімагистральді ОПҚ: а-құрылымы; б-қосу микробағдарламасы

Бірмагистральді ОПҚ-ны іске қосқанда микросұлба кристаллында магистраль үшін көп орын керек емес. Бірақ операцияларды орындау үшін ОПҚ құрамына екі аралық регистрлер АР1 және АР2 енгізілу керек (6.4-сурет). Операцияны орындау үшін үш такт керек.

Бірінші тактта АР1-ге бірінші операнд қабылданады, екінші тактта АР2 екінші операндты қабылдайды, үшінші тактта АР1 мен АР2 сақталған операндтармен операция орындалып, нәтиже ЖМР біріне жазылады.



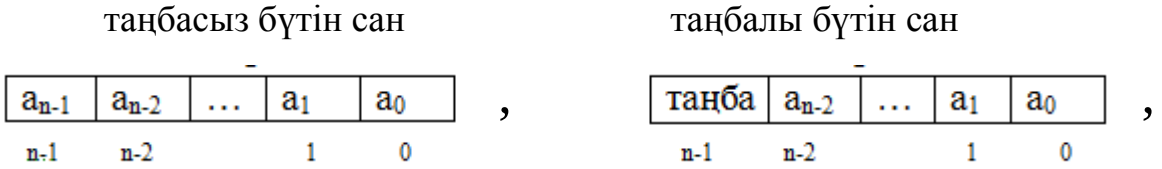
а) б)

6.4-сурет. Бірмагистральді ОПҚ: а-құрылымы; б-қосу микробағдарламасы

6.1. Бекітілген үтірлі сандар ОПҚ

Бекітілген үтірлі сандарды таңбасыз және таңбамен көрсетуге болады. Таңбасыз түрінде көрсеткенде санның барлық n позициясы санды көрсететін цифрлардан тұрады, ал санды таңбасымен көрсеткенде (n-1)-ші разряд арқылы санның таңбасы (0-оң, 1-теріс), ал қалған разрядтары санды көрсететін цифрлардан тұрады.

Қазіргі көптеген ЕМ бекітілген үтірлі сандар үтір сан кодасының 0-разрядының оң жағында бекітіледі. Мұндай тәсілмен бүтін сандарды көрсетуге болады (6.5-сурет). Осыған сәйкес келетін ОПҚ бүтін санды ОПҚ деп атайды.



6.5-сурет. Бүтін сандарды бекітілген үтір түрінде көрсету

6.6-сурет. Дұрыс бөлшектерді бекітілген үтір түрінде көрсету

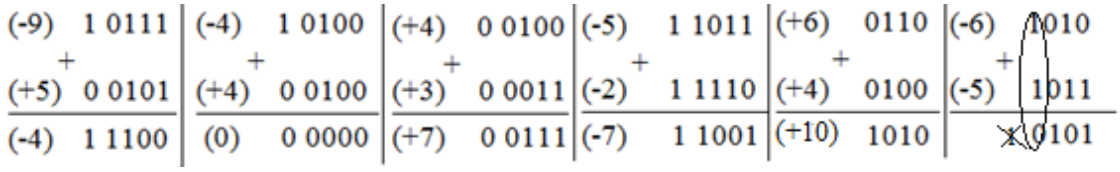
Жылжымалы үтірлі сандармен арифметикалық операциялар орындағанда операндтар мантиссаларын бекітілген үтірлі сандар деп қарауға болады. Мұнда мантиссалар бүтін сан емес дұрыс бөлшектер болады. Мұндай бөлшектер таңбасыз болса, үтір (n-1) разряд алдында орналасады, ал таңбасы бар бөлшек сандарда үтір (n-1)-разрядпен (n-2)-разрядтардың арасында орналасады (6.6-сурет).

Бекітілген үтірлі сандар, әдетте қосымша кодта көрсетіледі және нәтиже де қосымша кодта көрсетіледі.

Бекітілген үтірлі сандарды өңдейтін ОПҚ таңбасы бар, не таңбасыз сандармен қосу, алу, көбейту және бөлу амалдарын орындауға арналған.

6.1.1. Қосу және алу

Қосу және алу операцияларын орындағанда операнд таңба разрядтары мәнді сандары сияқты өңделеді. Теріс сандар операцияға қосымша кодта қатынасады. 6.7–суретте бүтін сандармен орындалған қосу операцияларына мысалдар келтірілген.

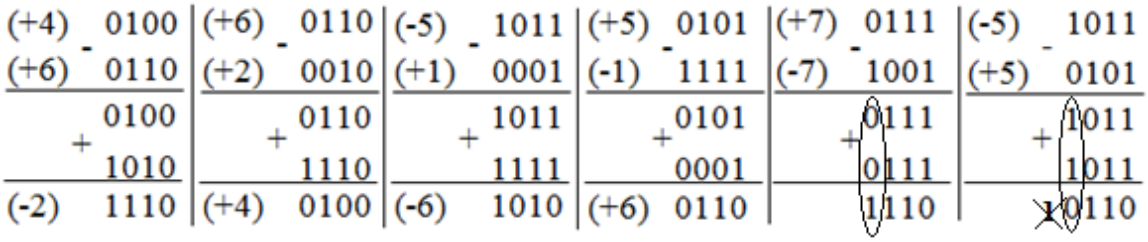


а б в г д е

6.7-сурет. Қосымша кодта сандарды қосуға мысалдар: а,б,в,г,д- аса толуы жоқ сандарды қосу; е – аса толуы қалыптасатын сандарды қосу

Алу операциясын қосу операциясына алып келеді. Ол үшін азайтқыш таңбасын кері таңбаға өзгертеміз. Егер өзгерткенге дейінгі таңба теріс болса, ол операцияға қосымша кодта, ал оң болса, тура кодта қатысады. Азайғыш өз таңбасымен операцияға тура не қосымша кодта қатысады.

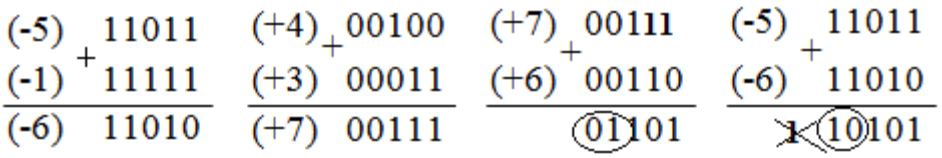
6.8-суретте қосымша кодта орындалатын азайту (алу) операциясына мысалдар келтірілген.



а б в г д е

6.8-сурет. Қосымша кодта сандарды бір-бірінен алуға мысалдар: а,б,в,г- аса толусыз азайту амалдары; д,е- аса толулары қалыпталатын мысалдар.

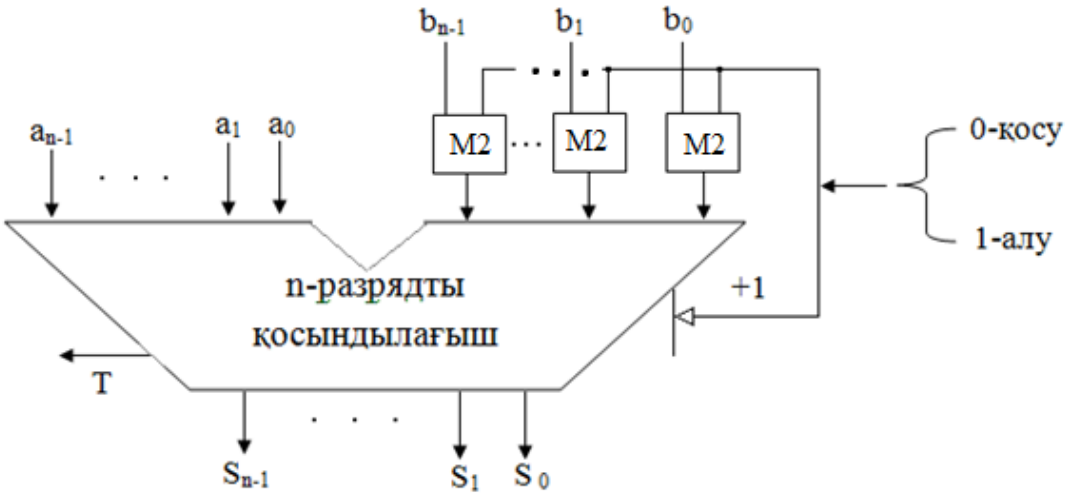
Аса толуды анықтау үшін, модификацияланған қосымша кода кеңінен қолданылады. Ол үшін таңба мәнін кодтағанда оған екі разряд белгіленеді және бұл екі разрядтар да операцияға қатысады. Операция орындалып болған соң, егер нәтиженің екі таңба разрядтары бірдей болса, онда ол аса толудың қалыптаспағанын көрсетеді. Керісінше, егер олар әртүрлі болса, онда олар аса толуды көрсетеді (6.9-сурет).



а) б) в) г)

6.9-сурет. Модификацияланған қосымша кодта сандарды қосуға мысалдар: а,б-аса толу жоқ; в,г-аса толу бар.

6.10-суретте бекітілген үтірлі сандармен алу және қосу операциясы арналған ОПҚ құрылымы келтірілген.



Т-тасымал

6.10-сурет. Бекітілген үтірлі сандарды алгебралық қосу блогы

А операнды n-разрядты қосындылауыш кірістеріне өзгеріссіз беріледі. В операнды қосындылауыштың кірістеріне n екілік модульмен қосу (mod 2) сұлбасы арқылы беріледі. Орындалатын операция түріне байланысты В операнды қосындылауыш кірісіне тура кодта немесе кері кодта беріледі. Бұл орындалатын операция қосу болса, m2 сұлбаларының кірістеріне «0» беріледі, ал орындалатын операция алу болса, онда m2 сұлбаларының кірістеріне «1» беріледі. Бірінші жағдайда В операндасы қосындылауыш кірістеріне тура кодта беріліп, S=A+B операциясы орындалады. Екінші жағдайда В коды кері кодта беріледі. Сұлбадан көрініп тұрғандай «+1»қосындылауыштың төменгі разрядына да беріледі. Қосындылауышта S=A+B+1=A+ (-B)=A-B операциясы, яғни алу операциясы орындалады.

6.10-суреттегі құрылғыны аса толуды анықтайтын (V) сұлбамен толықтыру керек. Мұндай сұлба төмендегідей логикалық өрнекпен анықталады:

$$V=\overline{a_{n-1}} \wedge \overline{b_{n-1}} \wedge S_{n-1} \vee a_{n-1} \wedge b_{n-1} \wedge \overline{S_{n-1}}$$

6.1.2. Көбейту

Есептеу машиналарында көбейту операциясы әдетте көбейгішті одан бұрын қалыптастырылған жекеленген көбейгіш қосындысына (ЖКҚ) біртіндеп қосып, алынған қосындыны бір разрядқа оңға қарай жылжыту арқылы орындалады. Көбейту үстінде көбейткіш кодының разрядтары бірінен соң бірі оның мәнін (0 немесе 1) анықтайтын сұлбаға беріледі. Егер кезекті берілген разрядта 1 тұрса, онда көбейгіш қосындылауыштағы ЖКҚ-на қосамыз, кезекті разрядта 0 тұрса, онда ЖКҚ-на көбейгіш қосылмайды, тек қосындылауыш коды

бір разрядқа оңға жылжытылады. Екі n-разрядты сандарды көбейткенде көбейтінді разрядтар саны 2n болады.

Көбейтінді таңбасы көбейгіш пен көбейткіштің таңбаларын екілік модулімен қосу арқылы табамыз. Егер олардың таңбалары бір бірімен сәйкес келсе, онда көбейтінді таңбасы оң болады, сәйкес келмесе, онда көбейтінді таңбасы теріс болады. Көбейту операциясын көбейткіштің төменгі не жоғарғы разрядтарын талдап орындауға және көбейту үстіінде көбейгішті не ЖКҚ жылжытуға болады.

Осыған байланысты көбейтудің төрт тәсілін атап көрсетуге болады. Осылардың ішінен ең көп тараған тәсіл – көбейткіштің төменгі разрядынан бастап көбейгішті жылжытпай, ЖКҚ оңға жылжытып көбейту тәсілі. Аталған тәсілді іске қосқанда ОПҚ элементтерінің (көбейгіш және көбейткіш регистрі, ЖКҚ регистрі және қосындылауыш) разрядтары n болады.

Таңбасыз сандарды көбейту

Таңбасыз сандарды көбейту алгоритмі төменде келтірілген қадамдармен анықталады.

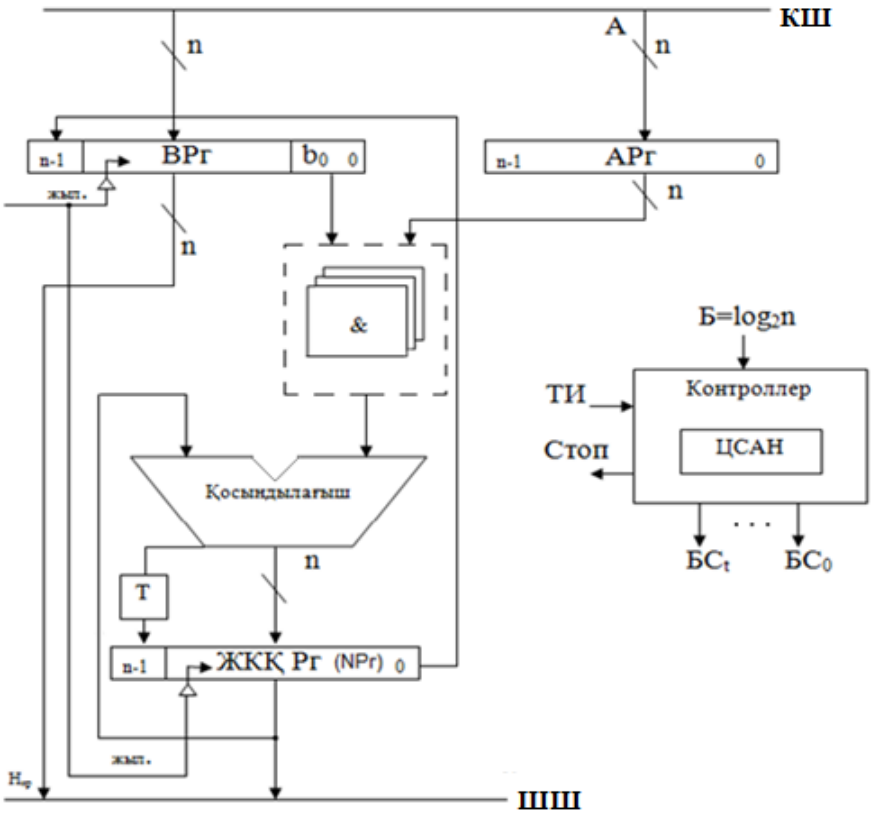
1. ЖКҚ бастапқы мәнін нөл деп қараймыз.
2. Көбейткіштің кезекті (төменгі) цифрын (b_i) талдаймыз. Егер $b_i=1$ болса, онда ЖКҚ көбейгішті қосамыз, $b_i=0$ болса, көбейгішті қоспаймыз.
3. ЖКҚ және көбейткіш разрядтарын бір разрядқа оңға жылжытамыз.
4. 2 және 3 пункттері көбейткіштің барлық разрядтары талданып біткенше қайталанады.

Көбейту амалына мысал 6.11–суретте көрсетілген.$A=1010_2=10_{10}$, $B=1011_2=11_{10}$.

	ЖКҚ регистрі (ЖКҚ Рг)	Көбейткіш регистрі (В Рг)
ЖКҚ бастапқы мәні	00000000	1011
А	+ 1010	
1-ЖКҚ	10100000	
1-ЖКҚ жылжыту	01010000	101
А	1010	
2-ЖКҚ	11110000	
2-ЖКҚ жылжыту	01111000	10
0	+ 0000	
3-ЖКҚ	01111000	
3-ЖКҚ жылжыту	+ 00111100	1
А	1010	
4-ЖКҚ	11011100	
4-ЖКҚ жылжыту	01101110	

$A*B=01101110_2=110_{10}$

6.11-сурет. ЖКҚ мен көбейткішті жылжыту арқылы таңбасыз санды көбейтуге мысал



6.12-сурет. Таңбасыз сандарды көбейту

Таңбасыз сандарды көбейтетін көбейту құрылғысы 6.12–суретте көрсетілген. Көбейту басталарда А көбейгіші n-разрядты А регистріне (МЖРг), В көбейткіші В регистріне (ВРг) кіріс шинасы (КШ) арқылы қабылданады. Операцияны орындау алдында жекеленген көбейтінді қосындысының регистріне (ЖКҚ Рг) немесе нәтиже регистріне (NPг) нөл жазамыз. Цикл санауышына (ЦСАН) көбейткіш разряд санынан екілік кодын ($Б=\log_2 N$) жазамыз. Көбейту үшін n қадам керек. Әрбір қадамда ЖӘНЕ сұлбалары арқылы n-разрядты қосындылауыштың (ҚОС) оң кірісіне көбейгіш разрядтарына 0 беріледі. ҚОС сол жақ кірісіне нәтиже регистрінен NPг кезекті ЖКҚ беріледі. ҚОС шыққан тасымал Т триггерінде сақталады. Жаңадан алынған жекеленген көбейтінді қосындысы NPг жіберіледі. NPг мен ВРг регистрлерін бірге 2n-разрядтары жылжытылатын регистр деп қарауға болады. NPг мен ВРг регистрлерін жылжыту үстінде Т триггерінде сақталған тасымал NPг регистрінің босаған жоғарғы разрядына жазылады. Өз кезегінде жылжу кезінде ВРг регистрінің төменгі разряды босайды. Оның орнына NPг шыққан төменгі разряд мәні жазылады. ВРг мен NPг регистрлерін жылжыту үстінде талданған b_i разряды жоғалады, оның орнына келесі қадамда талданатын разряд

мәні орналасады. Одан кейін ЦСАН кодын бірге азайтамыз. Қаралған тізбектер n рет қайталанады. Егер көбейткіштер кезекті разряд мәні b_i=1 болса, онда қосу және жылжыту операцияларын орындаймыз, ал кезекті көбейткіштің кезекті разряды b_i=0 болса, онда қосу операциясы орындалмайды, тек жылжыту операциясын орындалады. Көбейту процесі такт импульстарымен (ТИ) синхронизацияланып, цикл санауышы нөл мәнін алғанда контроллер СТОП сигналын шығарады. Көбейту нәтижесі (N) ВРг мен NРг регистрлерінде қалыптасады (ВРг-төменгі разрядтар, NРг-жоғарғы разрядтар). Нәтиже алынған регистрлерден шығыс шиналары (ШШ) арқылы шығарылады.

Таңбалы сандарды көбейту

Таңбалы сандардың жоғарғы разрядында таңба мәні көрсетілсе, қалған n-1 разрядтарында мән цифрлары көрсетіледі. Мұндай сандарды көбейтудің қарапайым тәсілінде көбейтіндінің таңбасы бөлек (сан таңбаларын екі модулімен қосу арқылы), ал сандардың абсолют мәндерін таңбасыз сандар ретінде бөлек көбейту арқылы анықтауға болады. Бірақ та есептеу машиналарында бекітілген үтірлі сандар қосымша кодта көрсетіледі. Ондай сандарды тура кодқа түрлендірмей, осы қосымша кодта көбейту тиімді.

Ол үшін Робертсон алгоритмі [34] кеңінен қолданылады. Робертсон алгоритмінің жоғарыда қаралған алгоритмнен өзгешеліктері бар.

Бірінші өзгешелігі мынада: көбейту үстінде жекеленген көбейтінді (ЖК) оң және теріс сандар болуы мүмкін. Мұндай сандарды жылжытқанда арифметикалық жылжыту орындалуы керек.

ЖК оңға жылжытқанда таңба разряды жылжымау керек, ал босаған позицияға ЖК оң сан болғанда-нөл, теріс сан болғанда-бір жазылуы керек.

Екінші өзгешелік теріс көбейткішке көбейту кезінде пайда болады. Көбейткіш теріс сан болған соң, ол қосымша кодта $-[B]_{\text{қос}}$ жазылады.

Бүтін сан үшін $[B]_{\text{қос}}=1\ b_{n-2}\ldots b_0$, мұнда $b_{n-1}=1$, онда:

$$|B| = 2^n - [B]_{\text{қос}} = 1(00..0) - (1\ b_{n-1} \ldots b_0)$$
$$= 1(00\ldots0) - (100\ldots0) - (\ b_{n-1} \ldots b_0) = (10..0) - (\ b_{n-1} \ldots b_0)$$
$$= 2^{n-1} - \sum_{0}^{n-2} b_i, \text{өрнектен } [B]_{\text{қос}} = |B| = -2^{n-1} + \sum_{0}^{n-2} b_i$$

Робертсон алгоритмінде бұл төмендегі жолмен іске асырылады. Көбейткіштің таңба разряды операцияға оның қалған цифрлық разрядтары сияқты көбейтуге қатысады. Көбейткіштің төменгі b_0 разрядынан бастап жоғарғы b_{n-2} разрядына дейін талданып қосу және жылжыту операциялары стандартты түрде жүргізіледі. Ол операциялар $[B]_{\text{қос}}$ өрнегінің екінші жартысымен көрсетілген. Көбейткіш теріс сан болғанда оның b_{n-1} позициясында

1 жазылады. Әдетте талданатын разряд 1 мәнін алғанда ЖКҚ-на көбейгіш 2^{n-1} салмағымен қосылатын. Робертсон алгоритмінде оның орнына таңба разрядын талдағанда 2^{n-1} салмағы бар көбейгішті ЖКҚ қоспаймыз, керісінше одан алып тастаймыз. Мұндай түзету амалы тек таңбасы теріс санға көбейткенде орын алады.

Жоғарыдағы жағдайларды еске ала отырып, таңбалы екі санды көбейтудің алгоритмін былай жазуға болады:

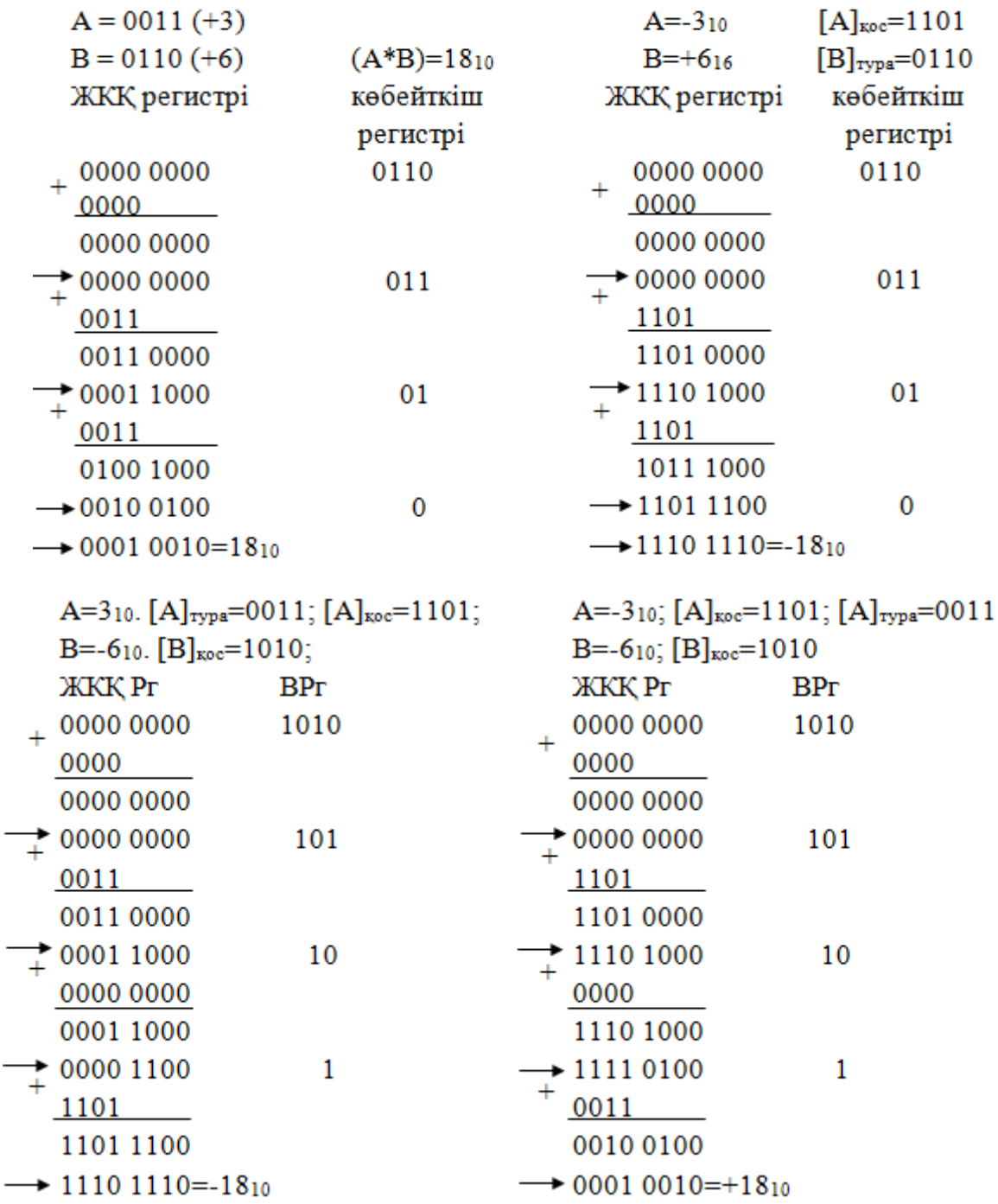
1. Көбейтуге қатысатын теріс сандарды қосымша кодқа түрлендіреміз.
2. ЖКҚ нөлге тең деп аламыз.
3. Көбейткіштің кезекті (төменгі) цифрын (b_i) талдаймыз. Егер ол $b_i=1$, онда ЖКҚ на көбейгішті қосамыз, $b_i=0$ болса – көбейгішті қоспаймыз.
4. ЖКҚ және көбейткіш разрядтарын бір разрядта оңға жылжытамыз.
5. 3 және 4 пункттер көбейткіштің барлық мән цифрлары талданып біткенше қайталанады.

Көбейткіштің таңба разряды талданады. Егер ол 1 болса (теріс көбейткіш), онда ЖКҚ-тан көбейткішті алып тастаймыз (көбейгіш кері таңбамен қосымша кодта).

Егер операндтар бүтін сан болса, онда арифметикалық жылжыту жүргізіледі (жылжыту кезінде босаған позиция ЖКҚ таңба мәнімен толтырылады). Егер көбейгіш сандар дұрыс бөлшек сандар болса, онда қосымша жылжыту жүргізілмейді.

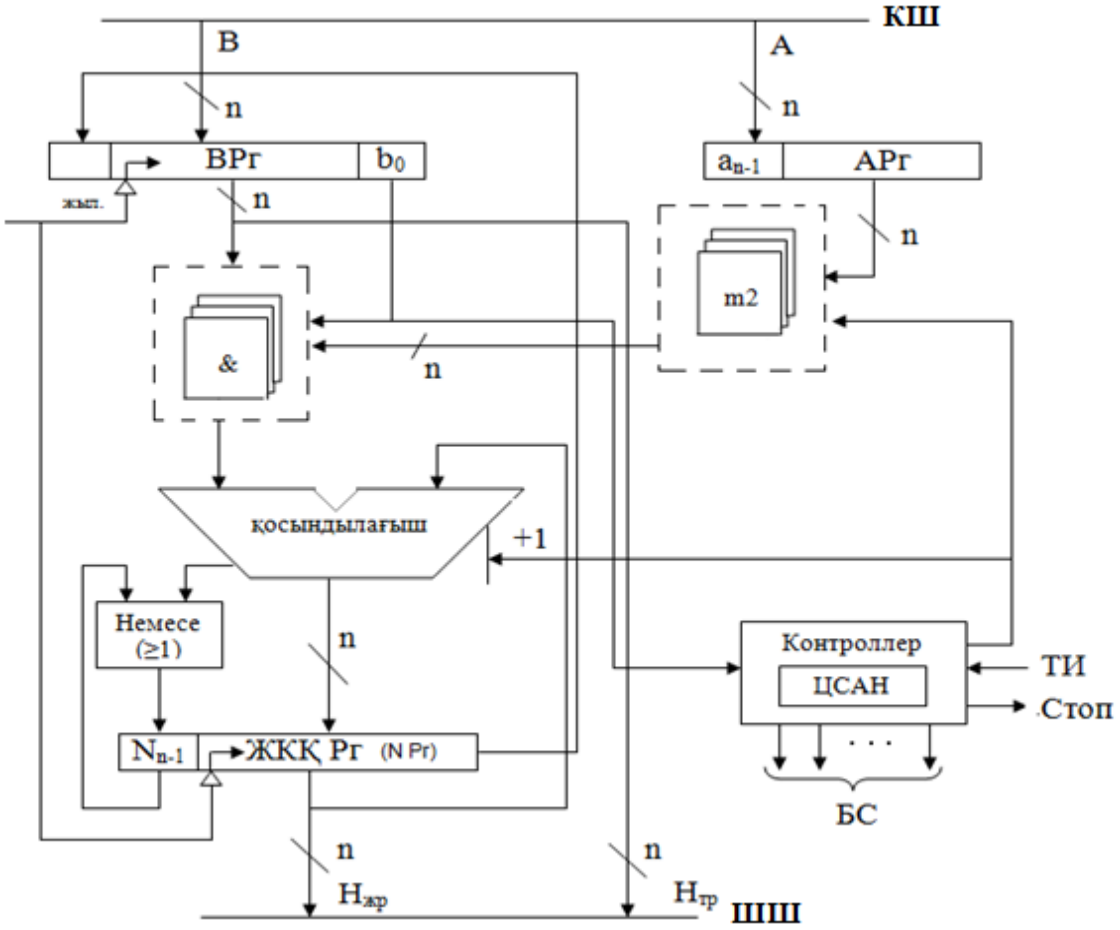
Егер көбейтілетін сандар таңбалары әртүрлі болса, онда көбейтінді теріс сан қосымша кодьта қалыптасады.

Робертсон алгоритмімен бүтін сандарды көбейтуге мысалдар 6.13–суретте қарастырылған.



6.13-сурет. Робертсон алгоритмінен бүтін сандарды көбейтуге мысалдар

Робертсон алгоритмі бойынша сандарды таңбасымен көбейтетін құрылғы 6.14–суретте көрсетілген.



6.14-сурет. Бүтін сандарды таңбасымен көбейту құрылғысы

Робертсон алгоритміне байланысты көбейту құрылғысының 6.14–суретте көрсетілген сұлбадан айырмашылығын байқауға болады. Робертсон көбейткішінің құрамына екі модулімен қосу (m2) сұлбасы енгізілген. Екі модулімен қосу сұлбасы нәтижені түзету кезеңінде орындалатын көбейгішті ЖКҚ-сынан алу амалын орындауға мүмкіндік береді. Екінші айырмашылығы ЖКҚ регистрінің таңба разрядының кірісіне «НЕМЕСЕ» сұлбасы қосылған. Ол арқылы ЖКҚ- ны арифметикалық жылжыту орындалады.

Көбейту амалдарын жеделдету

Көбейту амалдарын жеделдету тәсілдері екі түрге: логикалық және аппараттық тәсілдерге бөлінеді. Екі тәсілді де іске асыру үшін ОПҚ құрамына қосымша сұлбалар енгізіледі.

Логикалық тәсілдерде қосымша шығындар ОПҚ разряд санына байланысты болмайды, бірақ ОПҚ басқару бөлігінің шығыны артады.

Көбейтуді жеделдетудің логикалық тәсілдері екі топқа бөлінеді:

- Көбейту үстінде қосындылау санын азайтатын тәсіл;
- Көбейту үстінде көбейткіштің бірнеше разрядтарын талдап көбейту.

Бірінші топқа мысал ретінде Бут алгоритмдерін [24], Леман алгоритмін [31] атауға болады.

Екінші тәсілге көбейткіштің екі, үш, төрт разрядтарын бір тактте талдап көбейтетін тәсілдер жатады.

Аппараттық тәсілдерге біртактылы матрицалық және тармақты сұлбалық көбейткіштер жатады.

Бут алгоритмі

Бут алгоритмінің негізінде екілік цифрлар тізбегінің мынадай арақатынастар теңдігі жатады: $2^m+2^{m-1}+\ldots+2^k=2^{m+1}-2^k$, мұнда m және k-бірліктерден тұратын топтың ең шеткі разрядтар нөмірі. Мысалы $011111=2^6-2^1$. Бұл жерде егер көбейткіште бірнеше бірліктен тұратын топ кездессе (мысалы, 011111), жекеленген көбейтінді қосындысына көбейгіш 2^k -нен 2^m -не дейін өсетін салмағымен қосылып отырады (біздің мысалда тиісті салмағымен көбейтінді бес рет қосылады). Бут алгоритмі бойынша көбейгіш ЖКҚ-нан 2^k салмағымен бір рет азайтылып, 2^{m+1} салмағымен бір рет қосса болғаны.

Бут алгоритмін орындау үшін үш операция: жылжыту, қосу және алу операцияларын орындау керек. Қосу немесе алу операцияларын қысқартудан басқа Бут алгоритмі таңбасыз және таңбалы сандарды көбейтуге мүмкіндік береді.

Бут алгоритмінде көбейткіш екілік жүйеден {0,1} басы артық жүйеге {$\bar{1}$,0,1} өтеміз. Жаңа жүйеде кезекті разряд 1 болса, онда көбейгішті ЖКҚ-ға қосамыз, $\bar{1}$ – болса, онда ЖКҚ-дан көбейгішті аламыз, 0–болса ешқандай операция орындалмайды. Барлық жағдайда итерация аяқталғанда ЖКҚ оңға қарай бір разрядқа жылжытылады.

Алгоритмді іске қосу үшін әрбір итерация басында көбейткіштің екі разрядтарын ағынды (b_i) және оның алдыңғы (b_{i-1}) разрядтарын ($b_i\ b_{i-1}$) талдау керек. Көбейткіштің төменгі разряды үшін (i=0) алдыңғы разряд мәні 0-ге тең. Әрбір і қадамда (i=0,1…n-1)тағымды $b_i\,b_{i-1}$ комбинация талданады.

Комбинация 10 бірліктер тізбегінің басталуын көрсетеді. Мұнда ЖКҚ-дан көбейгішті алып тастаймыз. Комбинация 01 бірліктер тізбегінің аяқталғанын көрсетеді. Мұнда көбейгіш ЖКҚ-ға қосылады. Комбинация 00 көбейткіште бірлік тізбегінің жоқ екендігін, ал 11- мұндай бірлік тізбегінің ішінде екендігін көрсетеді. Екі комбинация кезінде ешбір арифметикалық операция орындалмайды.

Әр цикл орындалып болған соң, көбейткіштің разряд саны есептеліп ЖКҚ разрядтары оңға қарай немесе көбейгіш разрядтары солға қарай жылжытылады. Цикл келесі разряд жұптары үшін қайталанады. Енді Бут

алгоритмімен көбейтілетін мысалдар қарайық. Ескертетін жағдай: ЖКҚ-ны оңға қарай жылжытқанда босаған позицияға таңба разрядының мәні жазылады. (6.15-сурет)

1-мысал. A=0 1010; B=00111

$[-A]_{кос}$=10110; A*B=70_{10}

Жекеленген көбейтінді қосындысы регистрінің (ЖКҚ Рг) бастапқы мәні=0. В көбейткіші ВРг регистріне жазамыз.

ЖКҚ Рг	ВРг
0000 00000	0011(10) -бірлік тізбек басы
+ 1011 0	(-A)
1011 00000	
→ 1101 10000	001(11) -бірлік тізбек іші
→ 1110 11000	
→ 1111 01100	00(11) -бірлік тізбек іші
→ 1111 10110	(01) -бірлік тізбек соңы
+ 0101 0	(+A)
0100 10110	
→ 0010 00110=70_{10}	(00) -тізбек сырты

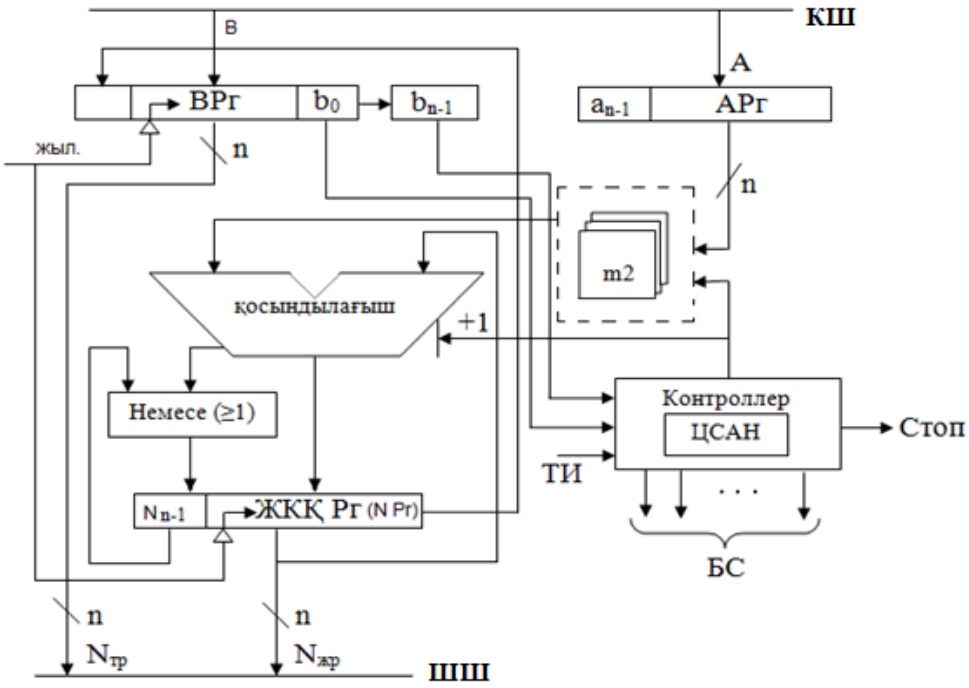
2-мысал. A=11000; B=0011;

A*B=-8*3=-24_{10}

ЖКҚ Рг	ВРг
0000 0000	01(10)
+ 1100 0	
1100 0000	
→ 1110 0000	0(11)
→ 1111 0000	(01)
+ 1100 0	
1011 0000	
→ 1001 1000=-24_{10}	

6.15-сурет. Бут алгоритмімен көбейтуге мысалдар.

Таңбалы сандарды Бут алгоритмімен көбейтетін құрылғы 6.16–суретте көрсетілген.



6.16-сурет. Сандарды таңбасымен Бут алгоритмі арқылы көбейту сұлбасы.

Көбейту құрылғысы жоғарыдағы көбейту құрылғыларына ұқсас. Тек мұнда көбейткіштің b_{n-1} разрядын сақтайтын қосымша триггер енгізілген. Контроллер қосу не азайту амалдарын басқаруға керек сигналдарын анықтау үшін оған b_0 және b_{n-1} разряд мәндерін талдайды. Операция басталар алдында b_{n-1} мәні 0.

Модификацияланған Бут алгоритмі

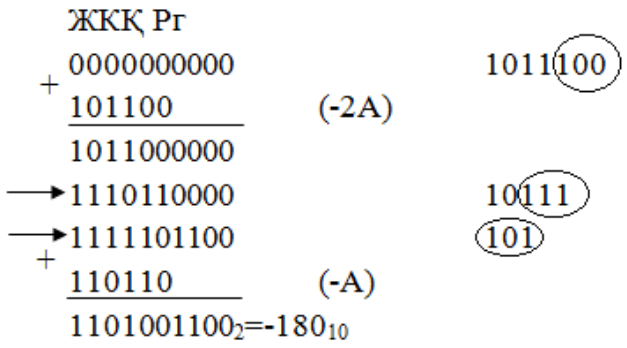
Модификацияланған Бут алгоритмінде екілік жүйеде көрсетілген көбейткіш коды {0,1} басы артық жүйеге $\{\bar{2},\bar{1},0,1,2\}$ түрленеді, мұнда әрбір сан көбейгішті ЖКҚ-ға қосу алдында көбейтетін коэффицент болып табылады. Алгоритмде көбейту үстінде көбейткіштің үш разрядтары b_{i+1} b_i b_{i-1} талданады (екі разряд ағымды, бір разряд алдыңғы үштіктің жоғарғы разряды). 0 мен 1-ден тұратын үштіктегі екілік сандардың комбинациясына байланысты әртүрлі (қосу, көбейгішті алу және екі еселенген көбейгішті ЖКҚ-ға қосу немесе одан алу) амалдары немесе әрекеттері орындалады (6.1-кесте).

6.1-кесте. Модификацияланған Бут алгоритмінің логикасы

i+1	i	i-1	Код $\{\bar{2},\bar{1},0,1,2\}$	Орындалатын әрекеттер
0	0	0	0	Ешқандай әрекеттер орындалмайды.
0	0	1	1	Көбейткішті ЖКҚ-ға қосу.
0	1	0	1	Көбейткішті ЖКҚ-ға қосу.

0	1	1	2	ЖКҚ-ға екі еселенген көбейткішті қосу.
1	0	0	$\bar{2}$	ЖКҚ-ға екі еселенген көбейткішті алу.
1	0	1	$\bar{1}$	ЖКҚ-дан көбейгішті алу.
1	1	0	$\bar{1}$	ЖКҚ-дан көбейгішті алу.
1	1	1	0	Ешқандай әрекеттер орындалмайды.

6.17-суретте $A=0.01010_2$ саны мен $B=-010010_2$ санын модификацияланған Бут алгоритмімен көбейтейік (ондық жүйеде 10* (-18)=-180).



6.17-сурет.10* (-18) сандарын модификацияланған Бут алгоритмімен көбейту.

Леман алгоритмі

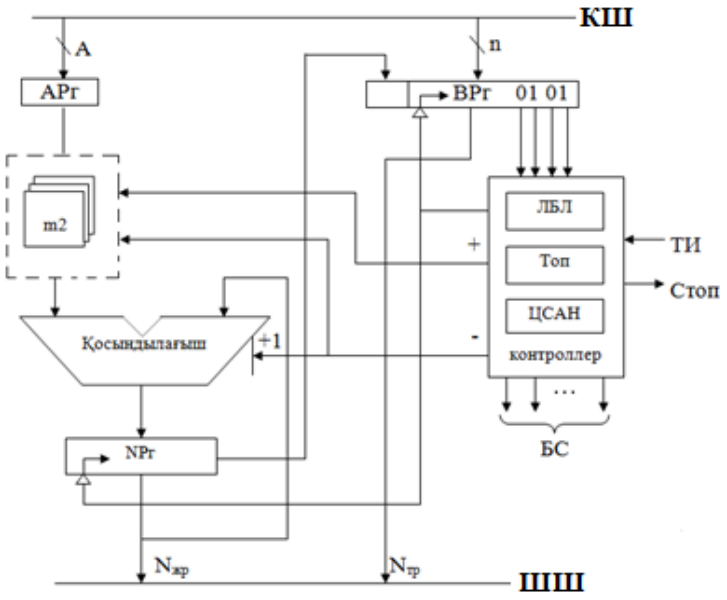
Көбейтуді жеңілдетудің логикалық тәсілдеріне Леман алгоритмі жатады. Бұл алгоритм бойынша қосу операциясының саны орта есеппен n/3-ке тең.

Леман алгоритмі мына арақатынасқа негізделген: $+2^{k+1}-2^k=+2^k$ және $-2^{k+1}+2^k=-2^k$. Егер нөлдерден тұратын екі топ k-позицияда тұрған бірмен бөлініп тұрса, онда k-позицияда орындалатын алу амалын және (k+1) позицияда орындалатын қосу амалын орындау. k-позицияда тек қосу амалымен алмастыруға болады. Егер бірліктерден тұратын екі топ k-позициясында тұрған нөлмен бөлініп тұрса, онда k-позицияда қосу амалын және (k+1) позицияда алу амалын орындаудың орнына k-позицияда тек алу амалымен алмастыруға болады. Көбейту амалының і-қадамында орындалатын әрекеттерді келесі логикалық өрнекпен көрсетуге болады:

$$d_i=(b_i \oplus b_{i-1})\bar{d}_{i-1}; \quad s_i=d_i b_{i+1} \vee \bar{d}_i s_{i-1};$$

Мұнда b_i– көбейткіштің разряд цифры; d_i– екілік сан, $d_i=1$ болса, ол көбейткіштің тиісті разряды үшін арифметикалық операция орындау керектігін көрсетеді; s_i– арифметикалық амал таңбасы. Екілік сан $d_i=1$ болса, онда $s_i= b_i+1$. Егер $s_i=0$ болса, онда көбейгіш ЖКҚ-ға қосылады, ал егер $s_i=1$ болса, онда ЖКҚ-дан алынады.

Леман алгоритмімен көбейтетін құрылғы 6.18 –суретте көрсетілген.



6.18-сурет. Леман көбейткіші

Контроллер құрамында операция түрін сақтайтын триггер Топ, көбейткіштің екі төменгі разрядтарының мәніне және Топ күйіне байланысты қосу (+) және алу (-) сигналдарын шығаратын логикалық блок (ЛБЛ) бар. Операцияның разряд сандарын санап отыратын және көбейткіштің барлық разрядтары талданып болғанда СТОП сигналын шығаратын цикл санауышы (ЦСАН) бар. Контроллер аталған сигналдардан басқа микрооперацияларды орындайтын басқару сигналдарын қалыптастырып отырады.

Құрылғының қалған операциялық блоктарының жұмысы жоғарыда қаралған көбейту құрылғысының жұмысына ұқсас.

Топ 1 күйінде тұрса, онда ол алу операциясына көрсетеді, 0 күйі қосу операциясына көрсетеді.

Көбейту операциясын бастау алдында Топ «0» күйіне түсіріледі, көбейткіштің төменгі разрядында 1 пайда болғанша ол жылжытылады. Егер келесі разрядта 0 жазылса, онда ЛБЛ қосу (+) сигналын шығарады, келесі разрядта 1 жазылған болса, онда ЛБЛ алу (-) сигналын шығарып, Топ триггеріне 1 күйін жазады. Егер Топ 1 күйінде болса, онда ЖКҚ және көбейткіш регистрінің төменгі разрядында 1 пайда болғанша жылжытылады. Көбейткіштің келесі разрядында 1 пайда болса, онда алу (-) сигналы шығарылады, ал «0» болса, ЛБЛ қосу (+) сигналын шығарады. Ол Топ «0» күйіне түсіреді.

Көбейткіштің бірнеше разрядтарын талдап көбейту тәсілдері

Көбейту амалын жеделдетуге алып келетін тәсілдердің бірі - бір циклде (бір қадамда) көбейткіштің бірнеше разрядтарын талдап көбейту тәсілі.

Алдымен көбейткіштің төменгі екі разрядтарын (жұбын) талдап көбейтуді қарайық.

Егер жұп 00 болса, ЖКҚ тек оңға қарай 2 разрядқа жылжытылады. Егер жұп мәні 01 болса, онда ЖКҚ-ға дара көбейгіш қосылып, жаңа ЖКҚ оңға 2 разрядқа жылжытылады. Жұп мәні 10 болса, екі еселенген көбейгіш (солға бір разрядқа жылжытылған) ЖКҚ-ға қосылып, ол оңға қарай 2 разрядқа жылжытылады. Талданып отырған жұп мәні 11 болса, онда ЖКҚ-дан дара көбейгіш алынып тасталып жаңадан қалыптасқан ЖКҚ оңға 2 разрядқа жылжытылады. Жұп мәні 11 болғанда, ЖКҚ-ға үш еселенген көбейгіш мәнін қосындылаудың орнына, одан дара көбейгішті алып тастаймыз. Дұрыс нәтиже алу үшін ЖКҚ-ны екі разрядқа оңға жылжытар алдында төрт еселенген көбейгішті қосу керек. Бірақ ЖКҚ-ны екі разрядқа жылжытқан соң, ол 4 есе азаяды. Түзету енгізу үшін келесі қадамда ЖКҚ-ға дара көбейгішті қосамыз. Түзету келесі разряд жұптарын талдағанда еске алынады. Егер келесі жұп 00 болса, онда ол 01 деп өңделеді. Егер келесі жұп 01 болса, онда ол - 10, ал 10 болса ол - 11 деп қаралады. Егер жұп 11 болса, онда ол 00 ретінде қаралады. Қосымша бірлік келесі жұпты талдағанда еске алынады. Ол жеке триггерде сақталады.

Қосымша бірлікті еске ала отырып, көбейткіш разряд жұптарын өңдеу тәртібі 6.2 –кестеде көрсетілген.

6.2-кесте. Көбейткіш разряд жұптарын өңдеу тәртібі

Көбейткіш жұптарының разрядтар мәні	Төменгі жұптан келетін қосымша бірлік	Келесі жұпқа берілетін қосымша бірлік	Әрекеттер	Көбейгіш есе мәні
00	0	0		0
01	0	0	+	1
10	0	0	+	2
11	0	1	-	1
00	1	0	+	1
01	1	0	+	2
10	1	1	-	1
11	1	1		0

Енді көбейткіштің үш разрядтарын талдап көбейтетін тәсілді қысқаша қарастырайық. Көбейткіштің үш разрядтарын талдап көбейту тәртібі 6.3–кестеде келтірілген.

6.3-кесте. Көбейткіштің үш разрядтарын өңдеу тәртібі

Көбейткіштің үш разрядтары	Төменгі үштіктен келетін қосымша бірлік	Келесі үштікке берілетін қосымша бірлік	Әрекеттер	Көбейгіш есе мәндері
000	0	0		0
001	0	0	+	1
010	0	0	+	2
011	0	0	+	3
100	0	0	+	4
101	0	1	-	3
110	0	1	-	2
111	0	1	-	1
000	1	0	+	1
001	1	0	+	2
010	1	0	+	3
011	1	0	+	4
100	1	1	-	3
101	1	1	-	2
110	1	1	-	1
111	1	1		0

6.3-кестеден көрініп тұрғандай 2 және 4 еселенген көбейгіш мәндерін көбейгіш мәнін 1 және 2 разрядқа оңға жылжыту арқылы қалыптастыруға болады. Үш еселенген көбейгіш алдын ала есептеліп қосымша регистрде сақталады.

6.4-кесте. Көбейткіш разрядтарын жоғарғы разрядтан бастап өңдеу тәртібі.

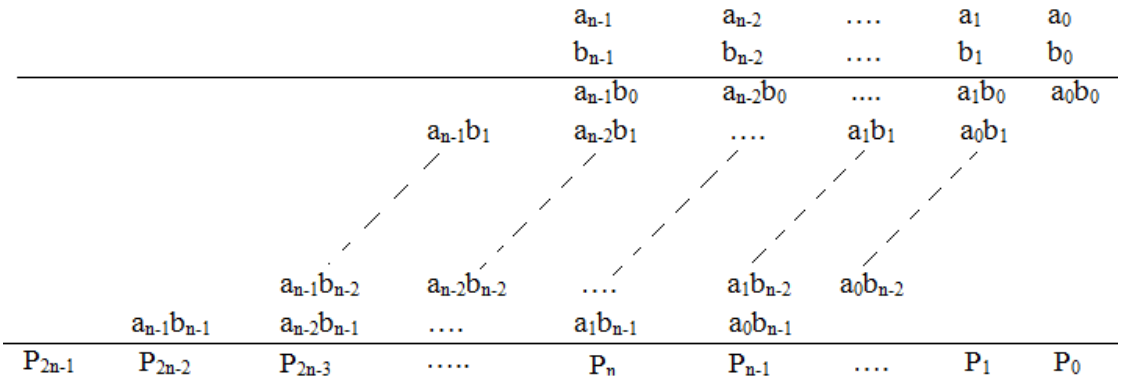
Көбейткіштің көрші жоғарғы жұбының төменгі разряды	Көбейткіш разрядының жұбы	Әрекеттер	Көбейгіш есе мәндері
0	00		0
0	01	+	2
0	10	+	2
0	11	+	4
1	00	-	4
1	01	-	2
1	10	-	2
1	11		0

Жоғарыда көбейткіш разрядтарын талдау төменгі разрядтан басталатын тәсілдерді қарадық. Басқа тәсілде көбейткіш разрядтарын талдау жоғарғы разрядтан басталады. Бұл тәсілдерде көбейтудің әр қадамында көбейткіштің екі

немесе одан да көп разрядтары талдануы мүмкін. 6.4–кестеде көбейткіштің көрші жоғарғы жұбының төменгі разряд мәнін еске ала отырып, көбейткіш жұбын өңдеу тәртібі келтірілген.

Көбейту амалдарын жеделдетудің аппараттық тәсілдері

Көбейту амалдарын жеделдетудің аппараттық тәсілдерінде аппараттық шығынды өсіру нәтижесінде көбейту уақытын азайтамыз. Көбейту үстінде жекеленген көбейтінді параллель қалыптастырылады, қосындылау санын қысқартып, жекеленген көбейтіндіні қосындылау үстінде тасымалдарды тарату уақытын азайтады. $A=a_{n-1}\ a_{n-2}\ \dots\ a_0$ санын $B=b_{n-1}\ b_{n-2}\dots b_0$ санына көбейтудің жалпы сұлбасы 6.19–суретте келтірілген.



6.19-сурет. Таңбасыз n-разрядтарды көбейту сұлбасы

Сұлбадан көрініп тұрғандай жекеленген көбейтіндінің (ЖК) дара разрядары $a_i\ b_1$ түріндегі көбейту. Олардың әрқайсысы көбейгіш пен көбейткіштің белгілі бір разрядтары. Жекеленген көбейтінді биттері мәні n^2 ЖӘНЕ сұлбалары арқылы анықталады. Сандарды қосымша кодта көбейткенде олар $a_i\bar{b}_I$, $\bar{a}_i b_i$ немесе $\bar{a}_i\bar{b}_i$ түрінде көбейтіледі. ЖӘНЕ элементінің жиынтығын конъюкторлық блоктар (КБ) деп атаймыз. Мұндай КБ барлық көбейту сұлбаларында орын алады. Көбейту сұлбалары негізінен ЖК қосындылау тәсілімен анықталады. Осы тұрғыдан алғанда қосындылау сұлбалары матрицалық және тармақты құрылымды [154] деп екіге бөлінеді. Екеуінде де қосындылау үшін бір разрядты бір-бірімен байланысқан қосындылауыштар массиві қолданылады. Матрицалық көбейткіштерде қосындылауыштар матрица түрінде ұйымдастырылған, ал бұтақты көбейткіштер қосындылауыштары әртүрлі бұтақ тәрізді ұйымдастырылған. Олар бір-бірінен қосындылауыштар санымен және қосындылау кезінде орын алатын тасымалдардың таралу уақытымен өзгешеленеді.

Таңбасыз сандардың матрицалық көбейткіші

А және В сандарды (n-разрядты) көбейту формуласын төмендегідей көрсетуге болады:

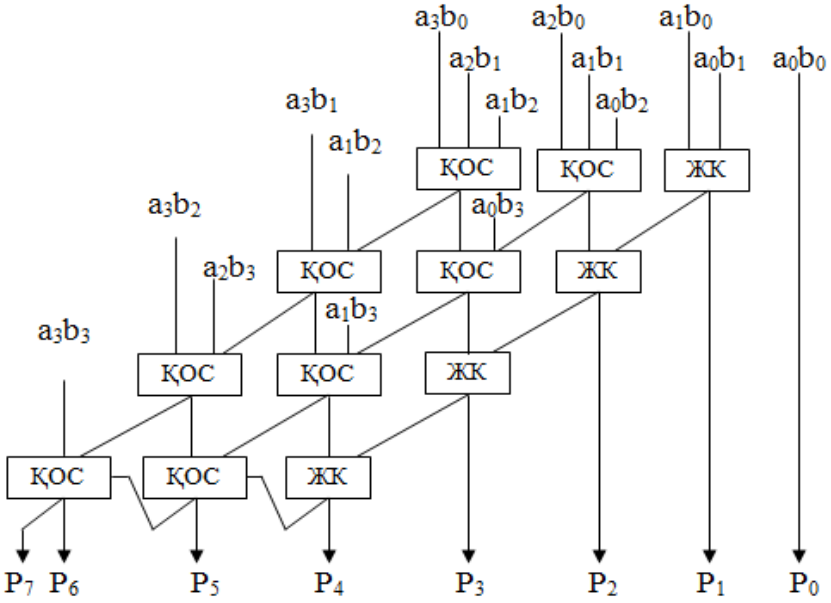
$$P = A \times B = \left(\sum_{i=0}^{n-1} a_i \times 2^i\right) \times \left(\sum_{j=0}^{n-1} b_j \times 2^j\right) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} a_i b_j \times 2^{i+j}$$

Сандарды көбейту үшін n-разрядты жекеленген көбейтінділерді параллель қалыптастырып, оларды матрица түрінде ұйымдастырылған қосындылауыштар арқылы қосындылау керек. Матрицалық көбейткішті Эдвард Браун (Edward Louis Braun) 1963 жылы ұсынған.

6.20, а-суретте таңбасыз төрт разрядты сандарды (n=4) көбейту матрицасы, ал 6.20, б–суретте осы матрицамен жұмыс жасайтын Браун көбейткіші көрсетілген. Матрица элементтері (a_ib_i) конъюкторлық блокта орналасқан 16 ЖӘНЕ элементтері (n^2=4=16) арқылы параллель қалыптастырылады.

				x a_3	a_2	a_1	a_0
				b_3	b_2	b_1	b_0
				a_3b_0	a_2b_0	a_1b_0	a_0b_0
		+	a_3b_1	a_2b_1	a_1b_1	a_0b_1	
		a_3b_2	a_2b_2	a_1b_2	a_0b_2		
	a_3b_3	a_2b_3	a_1b_3	a_0b_3			
P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0

а



б

6.20-сурет. а-таңбасыз төрт разрядты сандарды көбейту матрицасы; б-таңбасыз төрт разрядты сандарды көбейтетін Браун көбейткіші

Браун көбейткішінде жартылай қосындылауыш (ЖҚ) саны - n, ал қосындылауыштар (ҚОС) саны - n^2-2n. Көбейткіш жылдамдығы кіріс сигналы маршрутымен анықталады. Біздің суретте ол бір «ЖӘНЕ» элементі, екі ЖҚ және 2n-4 ҚОС арқылы өтеді.

Қосымша кодта көрсетілген сандарды көбейтетін матрицалық көбейткіш

Жоғарыда қаралған Браун көбейткіші таңбасыз сандарды көбейтуге арналған. Егер операндтар таңбасы теріс сандар болса, онда олар қосымша кодта көрсетіледі. Мұндай сандарды көбейтудің өзіндік өзгешеліктері бар.

Алдымен, қосымша кодта көрсетілген сандарды көбейтудің математикалық принципіне тоқталуымыз керек. A= (a_{n-1} a_{n-2} … a_n) санының a_{n-1} бит сан таңбасы болсын, онда оны қосымша кодта былай жазуға болады:

$$A = -a_{n-1} \times 2^{n-1} + \sum_{i-0}^{n-2} a_i \times 2^i$$

Оң +A және теріс –A сандарын қосымша кодта мынадай арақатынас арқылы көрсетуге болады [32]:

$$\begin{array}{c} +A - a_{n-i} \times 2^{n-1} + \sum_{i-0}^{n-2} a_i \times 2^i + \ 0 \\ \updownarrow \quad \updownarrow \quad \updownarrow \quad \updownarrow \\ -A - (1 - a_{n-1}) \times 2^{n-1} + \sum_{i=0}^{n-2} (1 - a_i) \times 2^i + 1 \end{array}$$

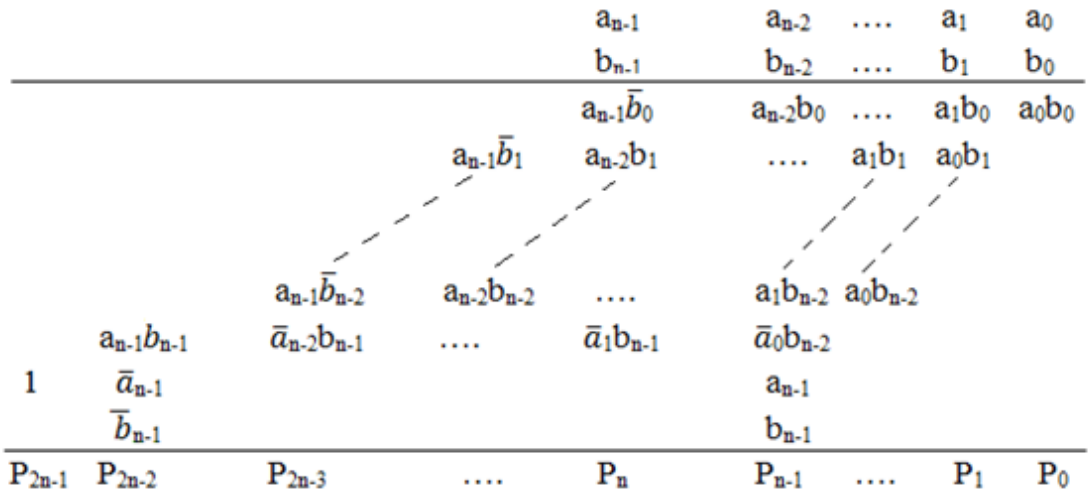
Жоғарыдағы өрнектен көрініп тұрғандай, санның таңбасын өзгерту үшін битті терістеу, яғни a_i мәнін (1-a)-ге алмастыру және 0-ді 1-ге, не кері алмастыру керек. Мұндай алмастырулар арқасында таңбасы теріс сандары қосымша кодта көбейтетін матрицалық көбейткіштерді құруға мүмкіндік береді.

Осындай матрицалық көбейткіш сұлбасына Чарльз Бо (Charles Baugh) және Брюс Вули (Bruce Wooley) матрицалық көбейткіші жатады [23].

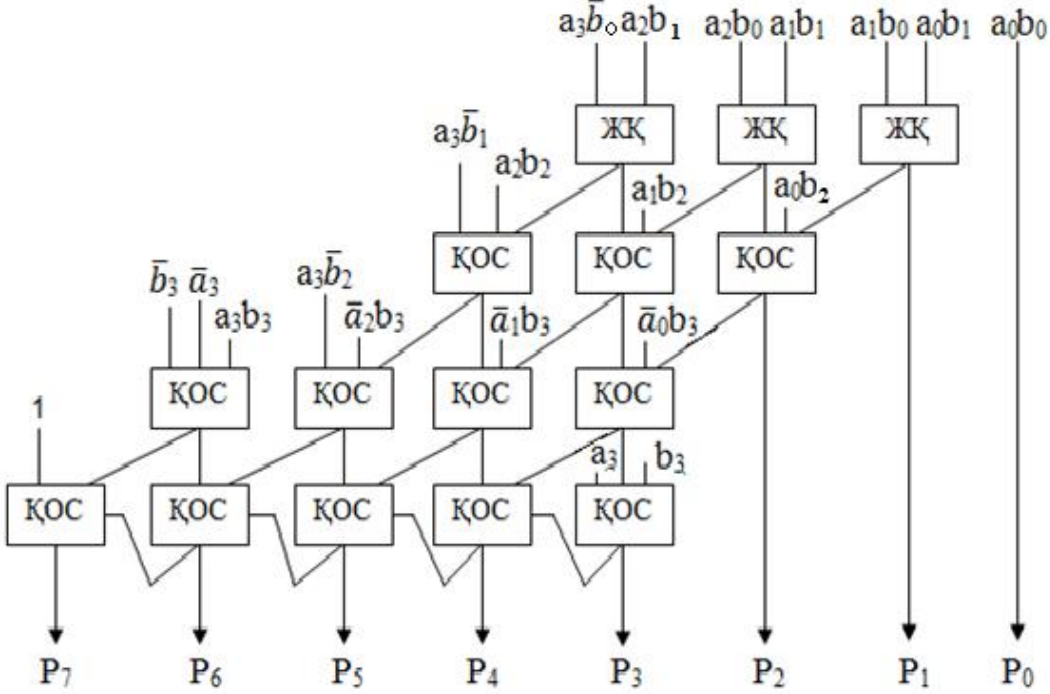
Бо-Вули алгоритмінде сандарды қосымша кодта көбейту мына формуламен анықталады:

$$P = A \times B = -2^{2n-1} + \left(\bar{a}_{n-1} + \bar{b}_{n-1} + a_{n-1}b_{n-1}\right) \times 2^{2n-2} + \sum_{i=0}^{n-2}\sum_{j=0}^{n-2} a_i b_j \times 2^{i+j} + a_{n-1}\sum_{j=0}^{n-2} \bar{b}_j \times 2^{n+j-1} + b_{n-1}\sum_{i=0}^{n-2} \bar{a}_i \times 2^{n+i-1} + (a_{n-1} + b_{n-1}) \times 2^{n-1}$$

Алгоритмді іске қосатын көбейту матрицасы 6.21–суретте көрсетілген, оған сәйкес келетін көбейту құрылғысы 6.22–суретте келтірілген.



6.21-сурет. Бо-Вули бойынша n-разрядты сандарды көбейту матрицасы



6.22-сурет. Қосымша кодта берілген төрт разрядты сандардың матрицалық көбейту сұлбасы (Бо-Вули көбейткіші)

Көбейту үстінде «теріс» таңбасы бар жекеленген көбейтінді қосындылаудың соңғы сатысына ығыстырылған. Жекеленген көбейтіндіні алу операциясы оларды терістеу арқылы қосу операциясымен алмастырылған. Сұлбаның кемістігіне матрицаның соңғы қатарында қосымша қосындылауыш керек етеді. Мұндай схемада n (n-2)+4 толық қосындылауыш және (n-1) жартылай қосындылауыш керек етеді.

Бұтақ тәрізді көбейту сұлбалары

Матрицалық көбейткіштерден кідірістерді азайтуда бұтақ тәрізді көбейту сұлбалары арқылы азайтуға болады. Матрицалық көбейткіштерде әрбір кезекті жекеленген көбейткіш ЖКҚ тек қосындылауыштар қатары арқылы қосындыланады, және n-разрядты сандарды көбейту үшін осындай n қатар керек.

Бұтақ тәрізді көбейту құрылғысында да ЖКҚ қосындылауыштар қатарлары іске қосылады, бірақ ол қосындылауыштар бұтақ тәрізді ұйымдастырылған. Соның арқасында қосындылауышта қатар саны $\log_2 n$ санына дейін қысқартылады.

Бұтақ тәрізді көбейту үш сатыдан тұрады:

- Бірінші сатыда әртүрлі $a_i b_i$ логикалық көбейтулер қалыптастырылады;
- Екінші сатыда жекеленген көбейтінділерін сығу үдерісі жүргізіледі;
- Үшінші сатыда қорытынды қосындылау жүргізіліп нәтиже қалыптастырылады.

Бірінші сатыда әртүрлі $a_i b_i$ логикалық көбейтулер конъюнктрлік блоктарда қалыптасады. Жекеленген көбейгіштерін сығу - бұл бұтақ тәрізді көбейткіштің ең негізгі бөлігі. «Сығу» кезінде көбейту матрицасының қатар санын кезеңмен азайтамыз. Сығу құралы ретінде қосындылауыштар мен жартылай қосындылауыштар пайдаланылады. Оларды әдетте санауыштар деп атайды, өйткені олардың шығыстарындағы код екілік санауыштардың шығыстарындағы код сияқты кірістеріндегі бірлік санымен анықталады. Бұтақ шығысында екі вектор: қосынды векторлары және тасымал векторлары қалыптасады.

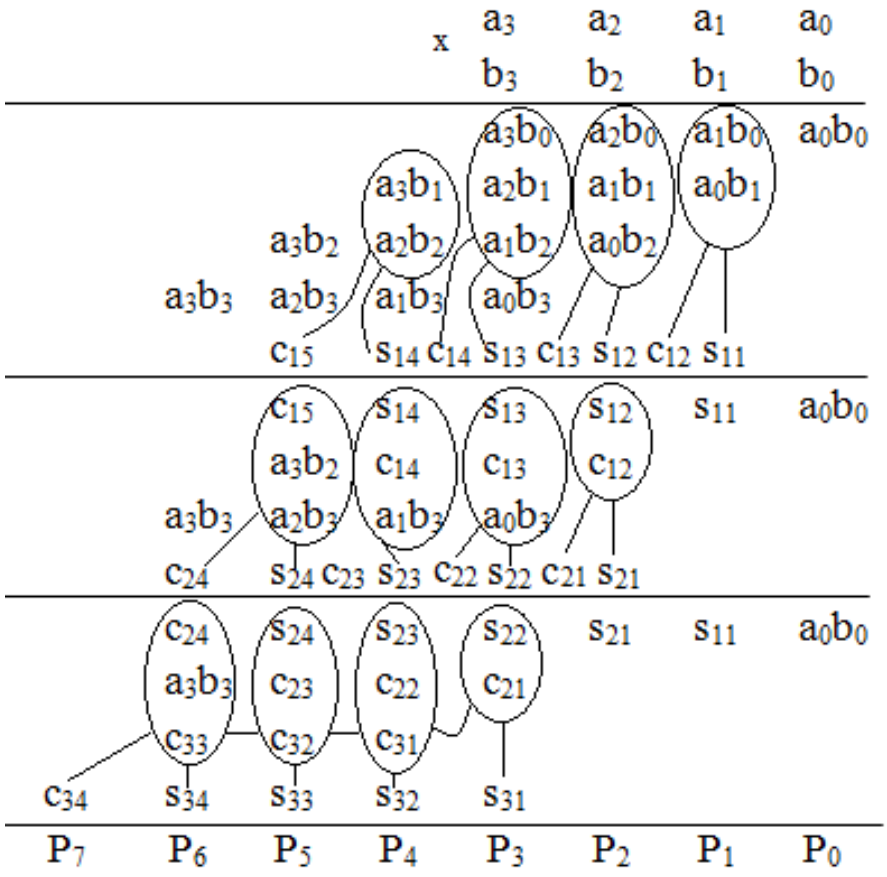
Көбейткіштің ең соңғы сатысында қосынды векторлары мен тасымал векторлары көп разрядты қосындылауыштар арқылы нәтиже қалыптасады.

Қосынды векторлары мен тасымал векторларын қалыптастыру тәсілдеріне байланысты әртүрлі бұтақ тәрізді көбейту сұлбалары бар. Олардың ішіндегі ең көп тарағаны - Уоллес көбейткіші.

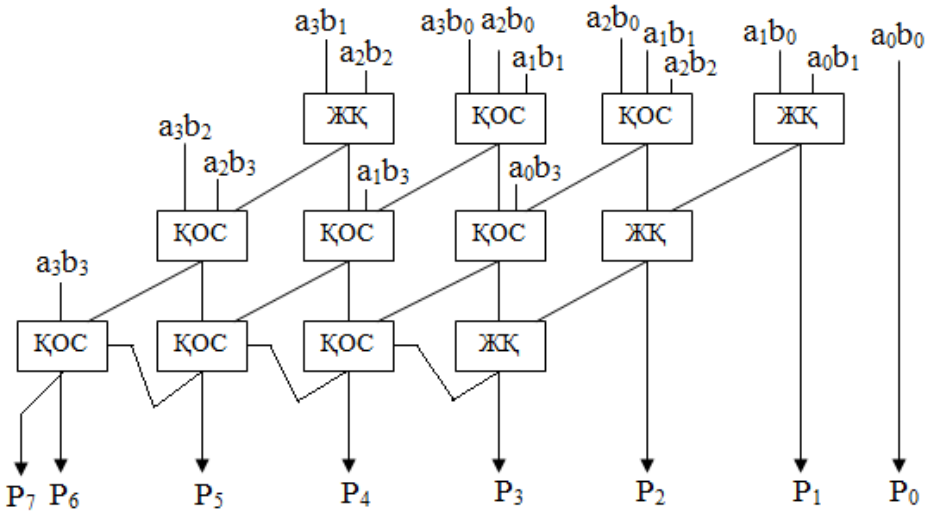
Уоллес көбейткіші

Уоллес көбейткішінде жекеленген көбейтінді матрицасының әр қатары үш-үштен топталады. Толық қосындылауыштар (ҚОС) үш битті бағаналарды сықса, ал жартылай қосындылауыштар (ЖҚОС) - екі битті бағаналарды сығады. Үш қатар жиынтығына ілікпеген қатарлар келесі сығу сатысында қаралады.

Төрт разряд санды көбейтетін Уоллес бұтағын құру логикасы 6.23–суретте көрсетілген. Уоллес көбейткіші (6.24-сурет) үш сатыдан тұрады. Конъюнкторлар блогын 16 ЖӘНЕ элементтері құрады. Көбейткіште төрт жартылай қосындылауыш (ЖҚОС), бес толық қосындылауыштар бар. Соңғы қатарда қосынды және тасымал векторлары бір ЖҚОС және үш толық қосындылауыш арқылы нәтиже қалыптасады. Әдетте ол үшін тасымалдары параллель таралатын көп разрядты қосындылауыштар қолданылады.



6.23-сурет. Уоллес бұтақтарымен жекеленген көбейгіштерді қосындылау сұлбасы



6.24-сурет. Уоллес бұтағымен көбейтілген 4х4 көбейткіші

Параллель көбейткіштерді конвейерлеу

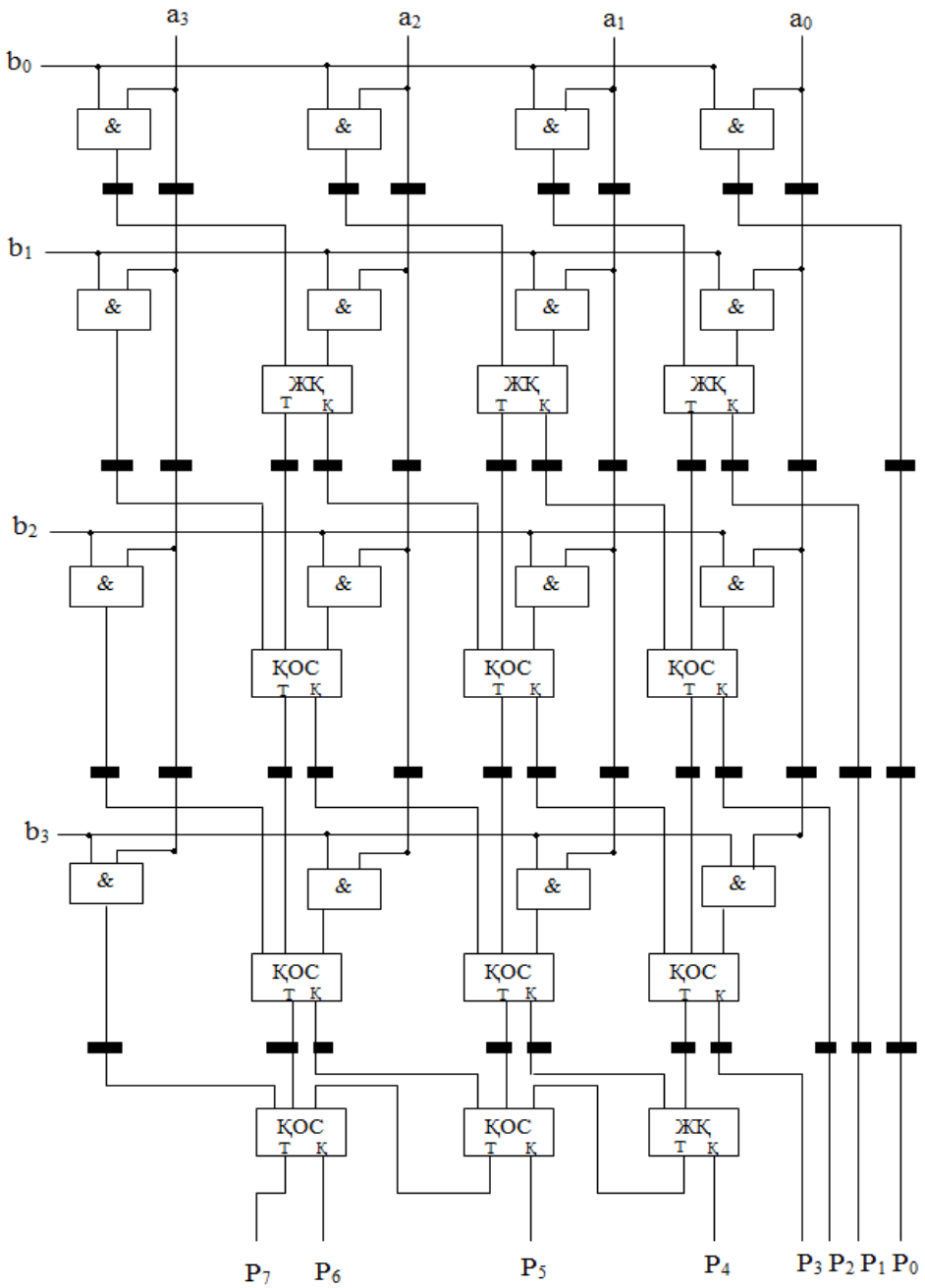
Матрицалық және Уоллес көбейткіші тәрізді қосындылауыштар бұтағымен құрылған параллель көбейткіштер жұмысын конвейерлеу арқылы құрылғының өнімділігін арттыруға болады.

Конвейерлеу кезінде операцияның орындалуы бірнеше кезеңдерге (сатыларға) бөлінеді. Әр кезеңде көбейту процедурасы өз сатысында орындалады. Конвейерлеу кезінде операцияның орындалуы бірнеше кезеңдерге (сатыларға) бөлінеді. Әр кезеңде көбейту өзінің сатысында параллель орындалады.

Конвейерлеу кезінде операция бірнеше кезеңде орындалатын ішкі операцияларға бөлінеді. Әрбір ішкі операция өз конвейерлік сатысын құратын логикалық аппараттардан тұрады. Мұндай конвейерлік сатылар көбейту үстінде қатар жұмыс жасайды.

Конвейердің і-ші сатысынан алынған нәтиже келесі і+1 сатыға беріледі. Нәтиже конвейер сатылары арасында орналасқан аралық жады арқылы беріледі. Өз ішкі операциясын орындаған конвейер сатысы нәтижені аралық жадыға орналастырып, кезектегі келесі деректі өңдеуге кіріседі. Сол кезде кезекті конвейер сатысы кірісінде орналасқан аралық жадыда сақталған деректі алып өңдеуге кіріседі. Конвейер жұмысы такт сигналдары арқылы үйлестіріледі. Такт сигналының периоды конвейердегі ең баяу жұмыс жасайтын сатысымен анықталады. Көбейткіштерді конвейерлеу кезінде, егер конвейер k сатыдан тұрса, онда оның кірісіне деректер конвейерсіз өңделетін көбейту құрылғысына қарағанда k есе аз интервалмен беріледі. Осыған байланысты көбейткіш шығысында нәтиже де осы темппен қалыптасады.

Мысал ретінде 6.25–суретте конвейерленген матрицалық Браун көбейткіші келтірілген. Суреттегі қара тікбұрыштар арқылы аралық жадыны құратын триггерлер белгіленген.



6.25-сурет. Конвейерленген матрицалық Браун көбейткіші

Разрядтылығы ұлғаймалы көбейту құрылғылары

Жоғарыда қаралған аппараттық тәсілмен құрастырылған көбейту құрылғыларының разрядтылығы шектеулі болып келеді. Разрядтылығы ұлғаймалы көбейту құрылғысын разрядтылығы кіші көбейту модульдері арқылы көбейтуді рекурсивтік декомпозициялау арқылы құруға болады. Мұндай көбейту құрылғыларын құрудың екі тәсілі бар.

Бірінші тәсілде ұлғаймалы көбейткіштер, разрядтары кіші көбейткіштер модульдер арқылы құрылады. Ұлғайтылған көбейткіш үшін олар жекеленген көбейтінді қалыптастырғыш (ЖКҚ) болады. Мұндай ұлғайтылған көбейткіштерде көбейтінді мәнін табу үшін қосымша қосындылауыш - жекеленген көбейтінділеу қосындылауышы (ЖККҚОС) керек.

Екінші тәсілінде ұлғайтылған көбейту құрылғысы көбейткіш-қосындылауы шмодульдері (КҚМ) арқылы құрылады. Мұндай модульдерде ЖКҚ және ЖККҚОС бір сұлбаға біріктірілген.

Бірінші тәсілмен құрастырылған ұлғаймалы көбейту құрылғысын қарастырайық. Ол үшін А және В операндтарын теңдей екі бөлікке бөлейік: жоғарғы ($А_ж$,$В_ж$) және төменгі ($А_т$,$В_т$) бөліктерге бөлейік. Онда,

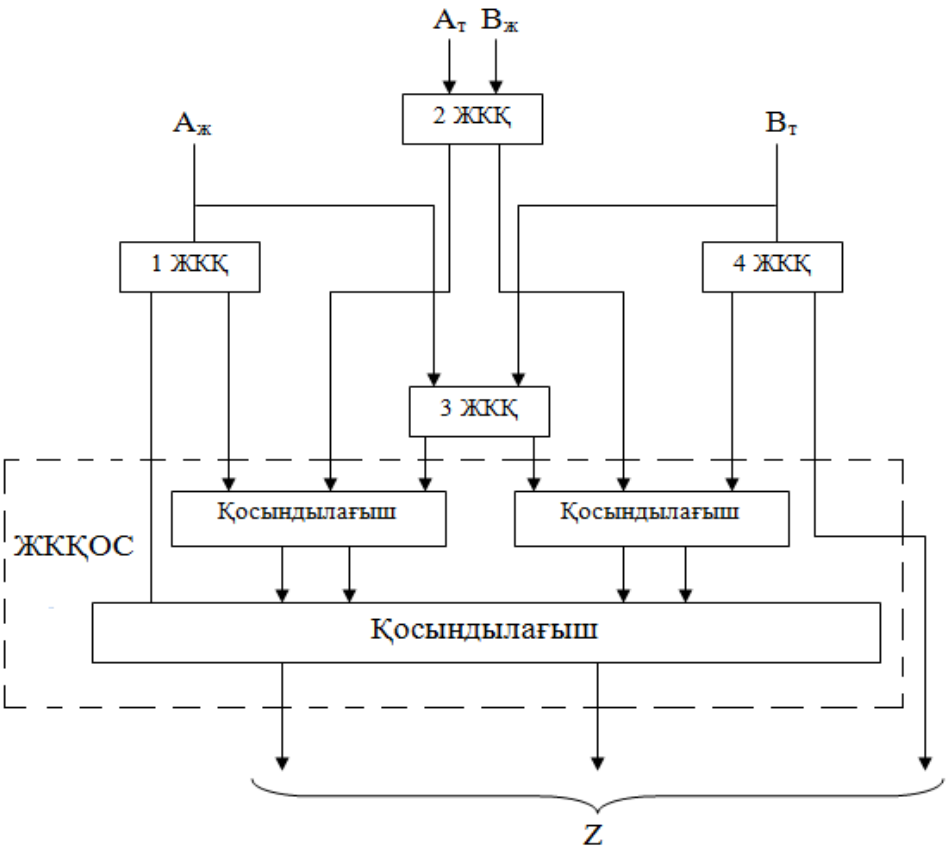
$Z=A*B= (А_ж+А_т) (В_ж+В_т)=А_жВ_ж+А_тВ_ж+А_жВ_т+А_тВ_т$;

Формуладан:

1. ЖКҚ = $А_жВ_ж$
2. ЖКҚ = $А_тВ_ж$
3. ЖКҚ = $А_жВ_т$
4. ЖКҚ = $А_тВ_т$

Нәтиже 1ЖКҚ÷4ЖКҚ жекеленген көбейтінді қалыптастырғыштарынан шыққан кодтардың разрядтарын сәйкестеп, көбейтінділер қосындылауышында (ЖККҚОС) қосу арқылы алынады.

6.26-суретте жекелеген көбейтінді кодтарын қосу тәртібі келтірілген.



6.26-сурет. Жекеленген көбейтінді кодтарын ЖККҚОС-та қосу тәртібі

Кіші көбейткіштер модуль ретінде (2х2), (3х3), (4х4) т.б. көбейту модульдерін алуға болады. Оларды базалық модульдер деп атаймыз.

Базалық модульдерді әртүрлі элементтер негізінде құруға болады. Оларға қарапайым логикалық элементтер, тұрақты жады, жартылай және толық екілік қосындылауыштар жатады.

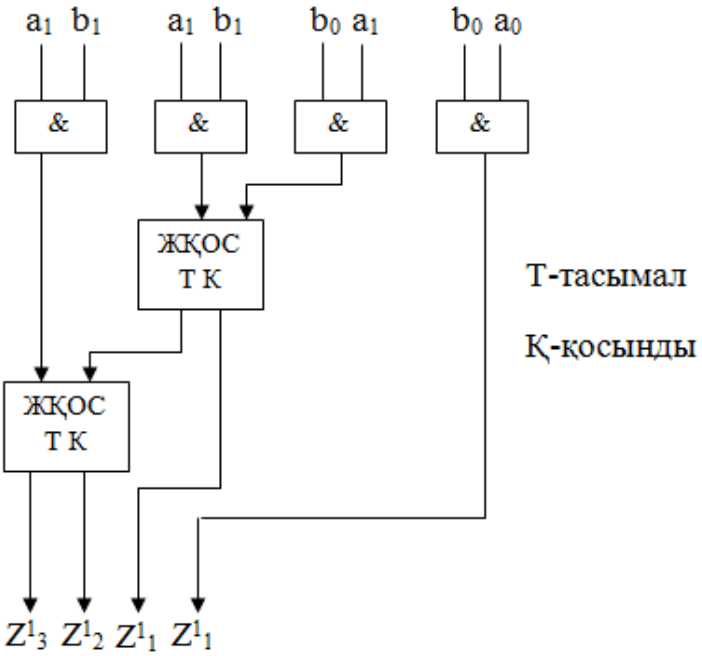
Мысал ретінде жартылай қосындылауыштар арқылы синтезделген базалық (2х2) модулі арқылы ұлғаймалы көбейткіштерді қарайық.

$A=a_1a_0$; $B=b_1b_0$болса,

Онда:

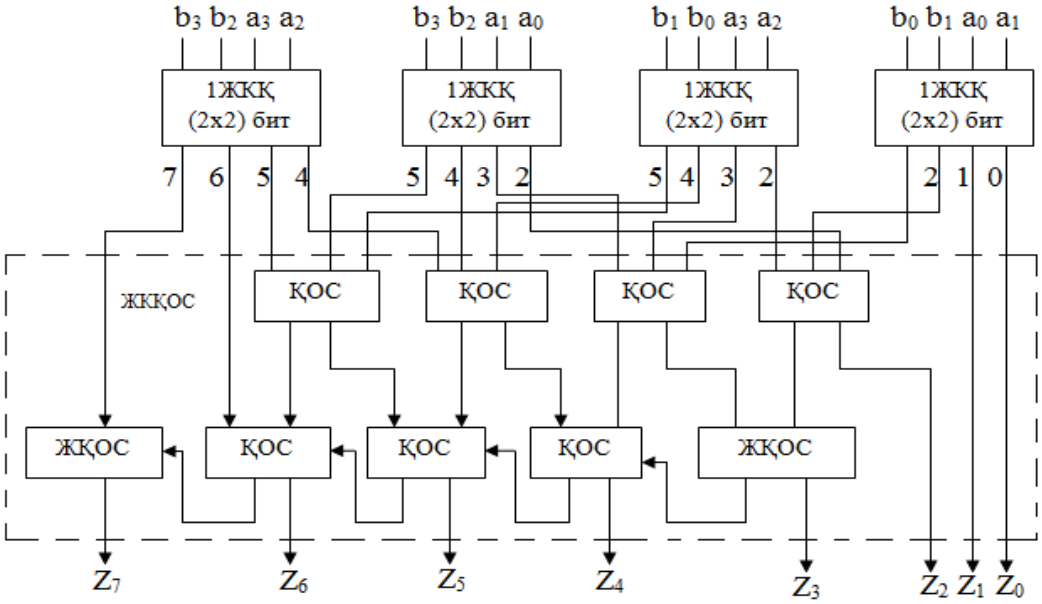
$$\begin{array}{rcccc} & & & a_1 & a_0 \\ & & \times & b_1 & b_0 \\ \hline & & & a_1b_0 & a_0b_0 \\ + & & a_1b_1 & a_0b_1 & \\ \hline & Z^1_3 & Z^1_2 & Z^1_1 & Z^1_1 \end{array}$$

Матрица бағаналарын қосындылай отырып, екі разрядты сандарды көбейтетін құрылғы сұлбасын аламыз (6.27-сурет).



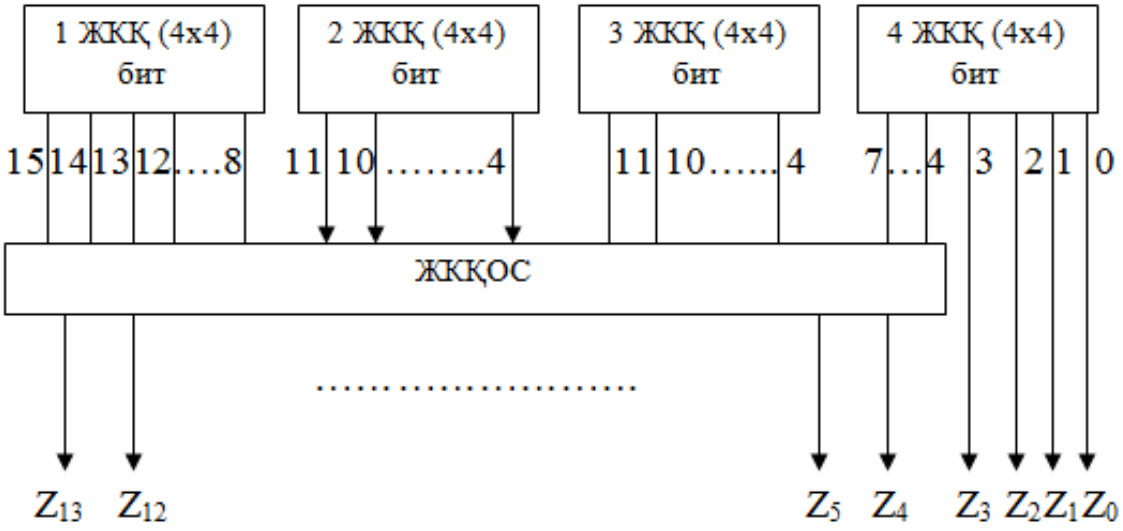
6.27-сурет. (2х2) базалық көбейткіші (модулі)

(2х2) базалық модулі арқылы құрылған (4х4) көбейткіші 6.28–суретте келтірілген. Мұнда жекеленген көбейтінді қалыптастырғыштары (1ЖКҚ÷4ЖКҚ) ретінде (2х2) базалық модулі қолданылады. Ол үшін әрбір ЖКҚ төрт разрядты сандардың тиісті разрядтарын беру керек. Көбейтінді ЖККҚОС-ында қалыптасады.



6.28-сурет. (2х2) модулімен құрастырылған (4х4) көбейткіш

6.29-суретте (2х2) модулі негізінде синтезделген (4х4) көбейткіші арқылы құрылған (8х8) көбейткіштің құрылымдық сұлбасы көрсетілген



6.29-сурет. (8x8) бит көбейткішінің құрылымдық сұлбасы

Ұлғаймалы көбейткіштерді құрудың екінші тәсілінде олар көбейткішті-қосындылауыш блоктары (ККБ) арқылы құрылады. ККБ-ында Z=AxB+C+D операциялары орындалады. Мұнда $C=C_{n-1}\ C_{n-2}\ \ldots C_0$ және $D=d_{n-1}\ d_{n-2}\ldots d_0$ операндтары үшін қосымша кірістер қарастырылады.

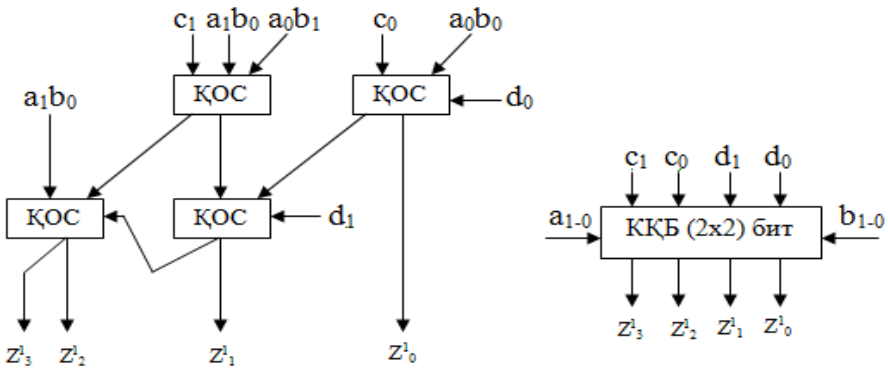
ККБ модульдер өлшемдері әртүрлі (2x2) бит, (3x3) бит, (4x4) бит, (8x8) бит т.с.с. болады. Ұлғаймалы көбейту құрылғысын осы модульдер арқылы синтездеуге болады. Мысал ретінде (2x2) битті ККБ модулін синтездеп, оның негізінде (4x4) бит және (8x8) бит көбейту құрылғысын құруды қарастырайық.

Біздің мысалымызда $A=a_1a_0$; $B=b_1b_0$; $C=c_1c_0$ және $D=d_1d_0$.

Сонда:

$$
\begin{array}{rcccc}
 & & a_1 & a_0 & \\
 & & \times & & \\
 & & & b_1 & b_0 \\
\hline
 & & a_1b_0 & a_0b_0 & \\
+ & & & & \\
 & a_1b_1 & a_0b_1 & & \\
 & & c_1 & c_0 & \\
 & & d_1 & d_0 & \\
\hline
Z^1_3 & Z^1_2 & Z^1_1 & Z^1_0 &
\end{array}
$$

Екілік қосындылауыш арқылы алынған мұндай ККБ сұлбасы 6.30–суретте келтірілген.

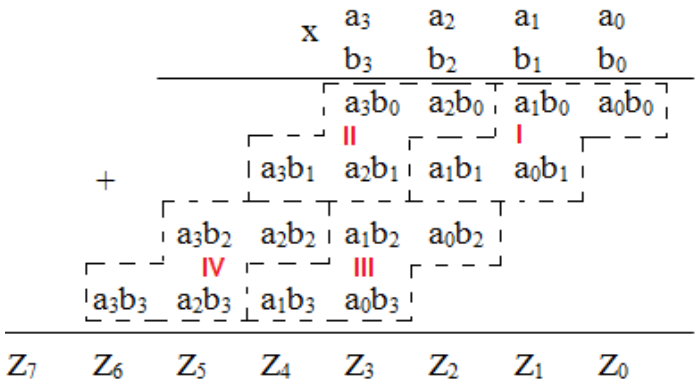


а б

6.30-сурет. (2х2) битті ККБ: а-функционалдық сұлба; б-шартты белгісі.

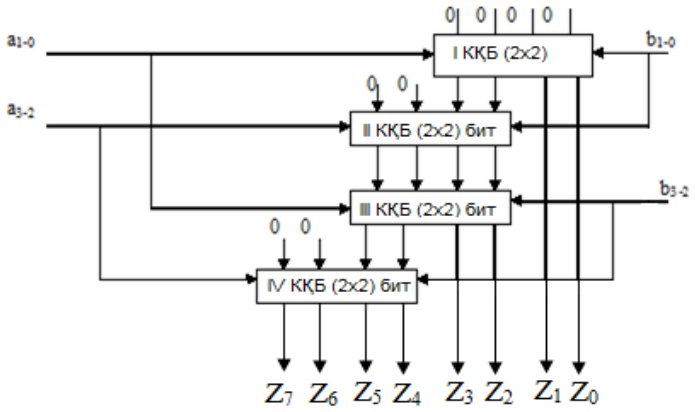
Сұлбада a_0b_0, a_1b_0, a_0b_1 және a_1b_1 төрт ЖӘНЕ сұлбалары арқылы алынады (олар суретте көрсетілмеген).

(4х4) битті көбейту құрылғысының матрицасы 6.31–суретте көрсетілген.



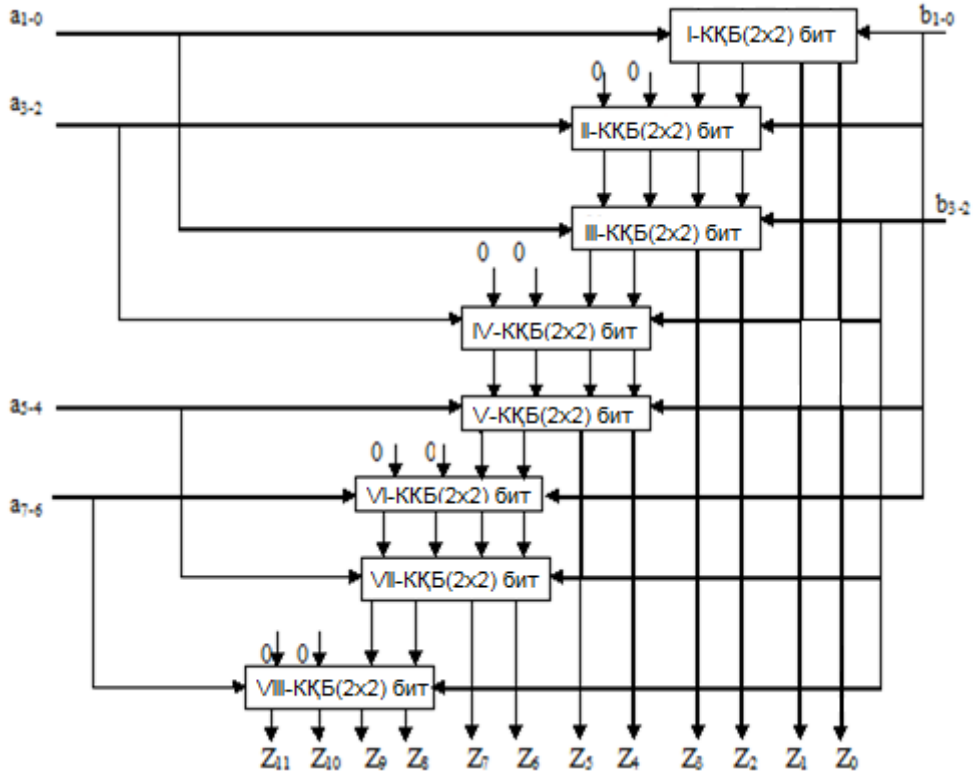
6.31-сурет. (4х4) битті көбейткіш құрылғысының матрицасы

6.31-суреттен көрініп тұрғандай (4х4) битті көбейту құрылғысын құру үшін төрт ККБ керек екен. Оларды 6.32–суретте көрсеткендей ретпен қосу керек. 6.32–суретте (2х2) битті ККБ арқылы алынған (4х4) битті көбейткішінің құрылымдық сұлбасы келтірілген.



6.32-сурет. (4х4) битті көбейткішін (2х2) бит ККБ арқылы құрастыру сұлбасы

Разрядтылығы (8х4) бит көбейткішін құрастыру үшін 8 (2х2) битті ККБ керек, немесе 2 (4х4) битті ККБ жеткілікті екендігін көз жеткізу қиын емес. Мұндай (8х4) битті көбейту құрылғысының құрылымы 6.33–суретте келтірілген.



6.33-сурет. (8х4) битті көбейту құрылғысы (2х2) бит ККБ арқылы құру сұлбасы.

6.1.3. Бөлу

Бөлу амалы алу (қосу) және жылжыту амалдары арқылы орындалады. А санын В санына бөлу кезінде Q бөліндісі мен R қалдықты анықтаймыз:

$$\frac{A}{B}=Q+\frac{R}{B} \quad (A=QB+R;\ R<B)$$

Бөлінгіш А (a_{2n-1} a_{2n-2}…a_1 a_0) екілік жүйеде 2n разрядымен көрсетіледі, бөлгіш В (b_{n-1} b_{n-2} b_1…b_0), бөлінді Q (q_{n-1} q_{n-2}…q_1 q_0) және қалдық R (r_{n-1} r_{n-2}…r_1 r_0) n разрядты сөздермен көрсетіледі. Бүтін және бөлшек сандар бөлу процедурасы бірдей болғандықтан бөлу алгоритмдеріне мысалдар бүтін сандармен көрсетіледі. 2n разрядты санды n разрядты санға бөлгенде бөлінді n разрядты саннан үлкен болуы мүмкін. Мұндай жағдайда аса толу орын алады. Сондықтан бөлу алдында тексеру жүргізу керек. Егер бөлінгіштің жоғарғы n разрядтары бөлгіштен аз болса (бүтін сан үшін), немесе бөлінгіш бөлгіштен аз болса (бөлшек сан үшін), онда аса толу орын алмайды.

Әдетте бөлу амалы екі тәсілмен орындалады:

- Бірінші тәсілде бөлінгіш қозғалмайды (жылжымайды), бөлу үстінде бөлгіш оңға жылжып отырады;
- Екінші тәсілде бөлгіш қозғалмайды (жылжымайды), бөлу үстінде бөлінгіш солға қарай жылжып отырады.

Бірінші тәсілдің кемістігі: алу (қосу) операциясын қосындылауыш және регистр разрядтылықтары -2n.

Екінші тәсілде жылжымайтын бөлгіш B n-разрядты регистрде сақталады, ал A бөлінгіш екі n разрядты регистрлерде сақталады. Бөлу үстінде қалыптасатын бөлінді Q цифрлары A бөлінгішін солға жылжытқанда босайтын разрядтарға жазылады.

Төменде мысал ретінде таңбасыз бүтін сандарды бөлудің екі негізгі алгоритмін қараймыз.

Қалдықты қалпына келтірмей бөлу

Алгоритмде бөлгіш қозғалмайды, бөлінгіш солға қарай жылжытылады. Кезекті жекеленген қалдық теріс болса, оған бөлгіш қосылып, қалдық қайта қалпына келтіріледі.

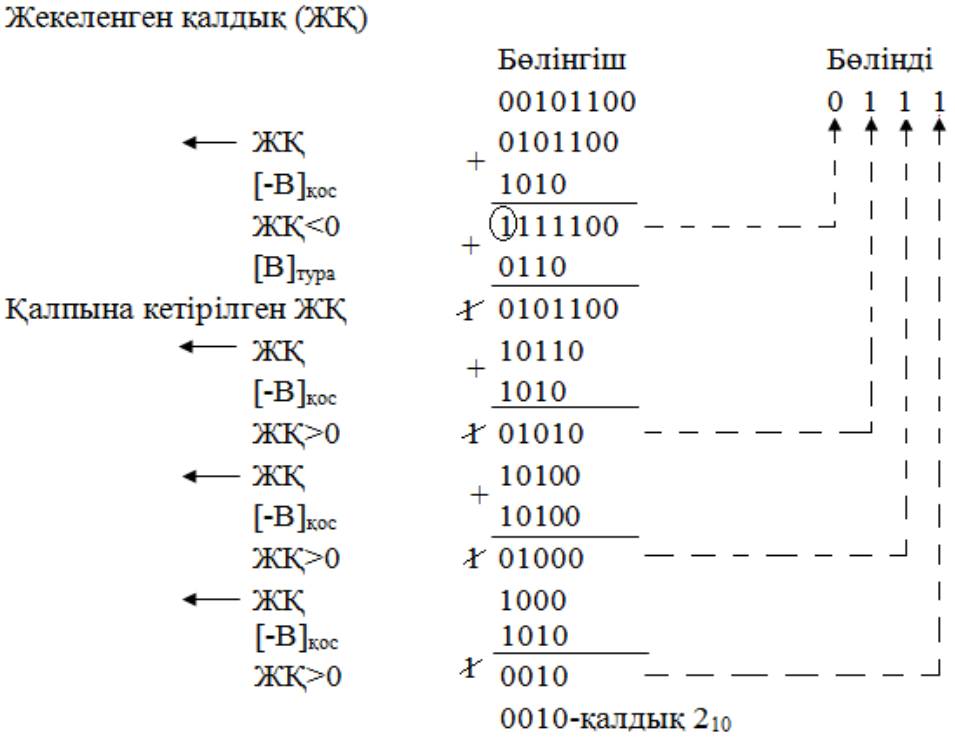
Бөлу алгоритмін төмендегідей ретпен көрсетуге болады:

1. Бастапқы жекеленген қалдықты (ЖҚ) бөлінгіштің жоғарғы разрядтарына тең деп аламыз;
2. ЖҚ бөлгішті алып (бөлгішті қосымша кодада ЖҚ-қа қосып), жаңа қалдықтың таңбасын талдаймыз;
3. Егер қалдық таңбасы оң болса, онда бөлуді тоқтатып аса толу сигналын шығарамыз. Егер таңба теріс болса ЖҚ бөліндіні солға қалдыққа қосып бөлуді жалғастырамыз;
4. ЖҚ солға бір разрядқа жылжытылады, босаған төменгі разрядқа бөліндінің келесі разряды жазылады;
5. Жылжытылған ЖҚ-дан бөлгіш алынып тасталады, нәтиже таңбасы талданады;
6. Бөлінді модулінің кезекті цифры нәтиже таңбасымен анықталады. Егер нәтиже таңбасы оң болса, бөліндінің разрядына 1 жазамыз, егер таңба теріс болса, онда 0 жазамыз;
7. 4÷6 пункттер бөліндінің барлық цифрлары анықталғанша орындалады.

Қалдықты қалпына келтіріп бөлуге мысал 6.34–суретте келтірілген.

Бөлінгіш A=00101100_2=44_{10};

Бөлгіш B=0110_2=6_{10}, $[-B]_{кос}$=1010



6.34-сурет. Қалдықты қалпына келтіріп бөлуге мысал

Қалдықты қалпына келтірмей бөлу

Қалдықты қалпына келтіріп бөлудің өз кемістігі бар. Ол тәсілдің атынан көрініп тұр – бөлу үстінде жекеленген қалдықтарды қайта қалпына келтіру керек. Бұл бөлу операциясын баяулатады. Сондықтан нақты бөлу операциясында қалдықты қалпына келтірмей бөлу алгоритмі кеңінен қолданылады. Мұндай алгоритмде 1÷4 және 7 пункттер толығымен қалдықты қалпына келтіріп бөлу алгоритміне сәйкес келеді, тек 5 және 6 пункттерін төмендегідей оқу керек:

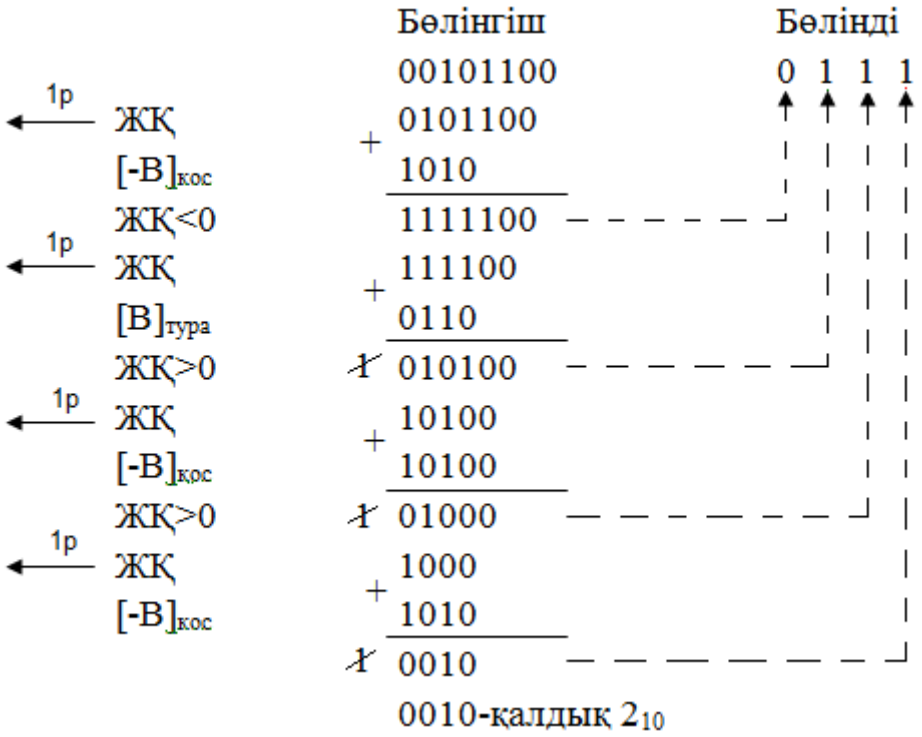
Егер қалдық таңбасы оң болса, онда жылжытылған ЖҚ-дан бөлгіш алынып тасталады. Егер қалдық таңбасы теріс болса, онда жылжытылған ЖҚ-қа бөлгіш қосылады.

Бөлінді модулінің кезекті разрядына егер операция (қосу, алу) нәтижесі оң болса, онда ноль жазамыз. Егер теріс нәтиже алынса, онда бөлінідінің кезекті разрядына бір жазамыз.

Қалдықты қалпына келтірмей бөлуге мысал 6.35–суретте келтірілген.

Бөлінгіш A=00101100_2=44_{10};

Бөлгіш B=0110_2=6_{10}



6.35-сурет. Қалдықты қалпына келтірмей бөлуге мысал.

Сандарды таңбасымен бөлу

Көбейту амалы сияқты бөлу амалында бөлінгіш пен бөлгіштің абсолют мәндерін бөліп, бөлінді таңбасына операндтар таңбалары бірдей болса, «+» меншіктеп, ал таңбалары әртүрлі болса, «-» меншіктеуге болады.

Қосымша кодта көрсетілген сандарды модульдеріне өтпей бөлуге болады. Қалдықты қалпына келтірмей бөлу алгоритмінен сандарды таңбасымен бөлудің өз ерекшеліктері бар.

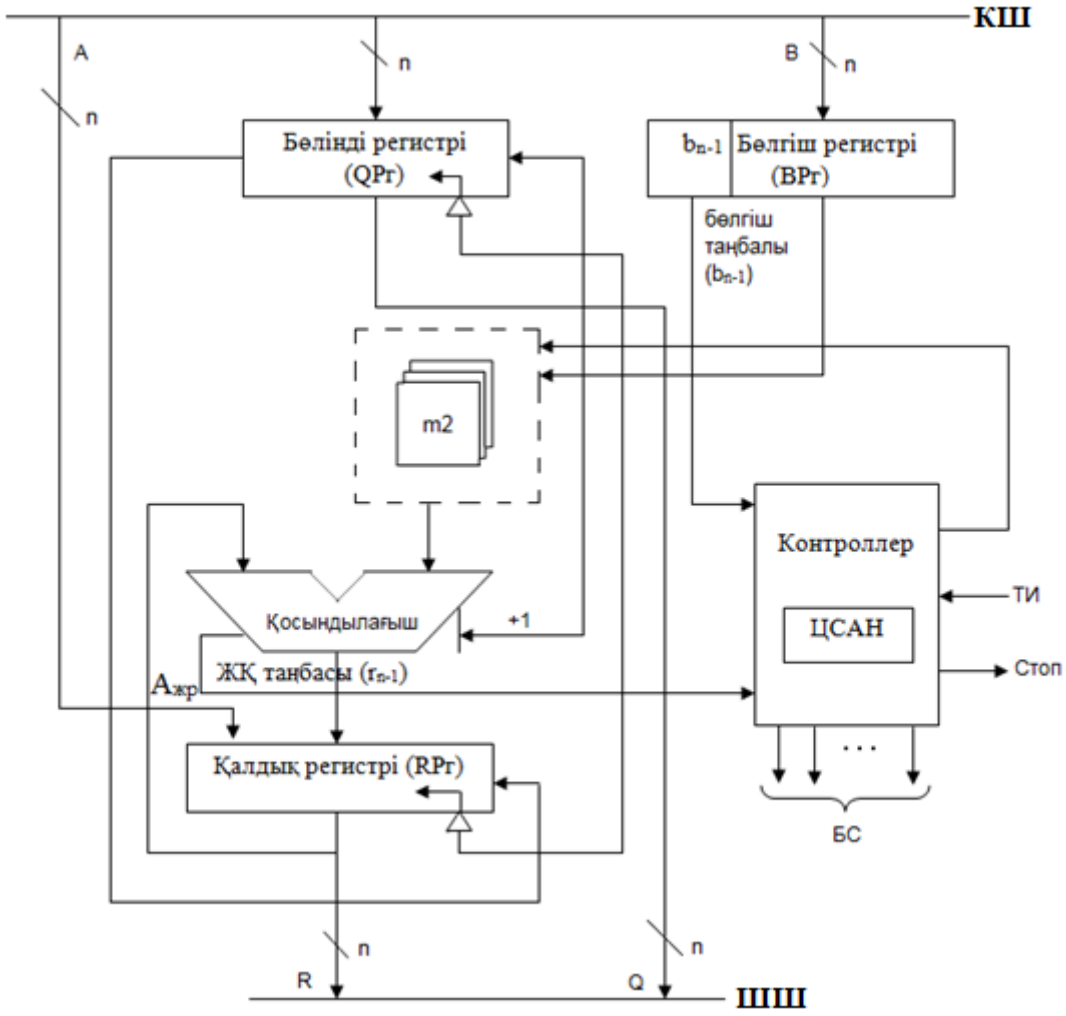
ЖКҚ-лармен жасалынатын әрекеттер ЖКҚ-мен бөлінгіш таңбаларына байланысты болады. Егер олардың таңбалары бір-бірімен сәйкес келсе, онда кезекті бөлу итерациясында бөлінгіш ЖКҚ-ынан алынады және бөліндінің кезекті разрядына 1 жазамыз. ЖКҚ және бөлінді таңбалары әртүрлі болса, онда ЖҚ-ға бөлінгіш қосылады және бөлінді разрядына 0 жазамыз. Бөлу операциясы аяқталған соң, операция нәтижесіне бөліндіні бірге өсіру арқылы түзету енгізуі керек болуы мүмкін. Мұндай түзету мынадай жағдайларда енгізіледі:

- A>0 және B<0;
- A<0, B>0 және R≠0;
- A<0, B<0 және R=0.

Есептеу машиналарында егер теріс қалдық қалыптасса, онда оған бөлгіш модулін қосу арқылы оң санға түрлендіреді.

Бөлу құрылғысы

Қалдықты қалпына келтірмей бөлу алгоритмін іске қосатын бөлу құрылғысы 6.36–суретте көрсетілген. Бөлу операциясы орындаларда 2n разрядты бөлінгіштің (А) жоғарғы разрядтарын ($А_{жр}$) кіріс шина арқылы (КШ) қалдық регистріне (RPг) және бөлінгіштің төменгі разрядтарын ($А_{тр}$) бөлінді регистріне (QPг) қабылдаймыз. Бөлгішті n-разрядты бөлгіш регистріне (BPг) қабылдаймыз. Цикл санауышына цикл санын жазамыз. Бөлудің әрбір циклында регистрге RPг және QPгбір уақытта солға қарай бір разрядқа жылжып отырады. Бөлінді таңбасымен (b_{n-1}) жекеленген қосынды таңбаларының мәндеріне байланысты бөліндінің кезекті цифрының мәні және керекті әрекеттер – бөліндіге ЖКҚ-ны алу немесе қосу әрекеттер түрі анықталады.



6.36-сурет. Қалдықты қалпына келтірмей бөлу алгоритмімен іске қосатын құрылғы.

Аталған таңбалар контроллер арқылы $r_{n-1}+b_{n-1}$ өрнегіне сәйкес өңделеді. Алу операциясы қосымша код арқылы қосу операциясымен алмастырылады. Таңбаны өзгерту және қосымша кодқа түрлендіру бөлгішті қосындылауыш

кірісіне кері кодта беру арқылы және қосындылауыштың төменгі разрядына бірді қосу арқылы іске асырылады. Кері код m2 сұлбасы арқылы қалыптасады. Бөліндінің кезекті цифры бөлінді регистрін (QРг) жылжытқанда босаған төменгі разрядқа жазылады. Әр циклда ЦСАН бірге кемітіліп отырады. Процедура бөліндінің барлық цифрлары анықталып біткенше жалғасады. Операция аяқталған соң, бөлінді регистрінде орналасады, ал қалдық регистрінде (RРг) қалдық орналасады. Қорытынды кезеңде керек болған жағдайда нәтижеге түзету енгізіледі (бөліндіні бірге өсіреді).

Бүтін сандарды бөлуді жеделдету

Бөлу операциясын жеделдету тәсілдеріне бөлу итерациясын қысқартуға алып келетін алгоритмді және аппараттық тәсілдер жатады. Төменде бөлу операциясын жеделдетуге мүмкіндік беретін қазіргі уақытта көп тараған тәсілдер қаралады.

SRT алгоритмі

SRT алгоритмінің аты оның авторларының бірінші әріптерінен (Sweeney, Robertson, Tocher) құралған [26, 34, 36]. Алгоритм қалдықты қалпына келтірмей бөлудің өзгертілген түрі. Алгоритмнің әр циклында ЖКҚ-ны жылжытудан басқа бөлгішті алу немесе қосу операциялары орындалады. SRT алгоритмінде де ЖҚ әр итерацияда жылжытылады, бірақ қосу және алу операциялары жаңадан алынған ЖҚ мәніне байланысты орындалмауы мүмкін. Бұл бөлу операциясын жеделдетуге алып келеді.

Алгоритм жылжымалы үтірлі сандарды өңдеуге арналған болатын, сондықтан мұндай сандар мантиссалары операцияға қалыптастырылған түрде қатысады.

Бөлінді 2n+1 разрядты сан, ал бөлгіш–n-разрядты сан. Бөлу операциясы бөлгіштің барлық нөлдерінсіз орындалады, басқаша айтқанда санның ең жоғарғы разрядындағы бірліктің алдындағы нөлдердің барлығын алып тастау керек. Мұндай операция жылжымалы үтірлі сандардағы қалыптастыру операциясына ұқсас. Сондықтан бұдан ары қарай мұндай операцияны *қалыптастыру операциясы* деп атаймыз. Бөлгіштің k нөлдерінен арылу үшін k разрядқа солға қарай жылжыту керек. Сондай санға (k санына) бөлінгішті де солға жылжытамыз. Одан ары n итерация орындалып, бөлінді және қалдық цифрлары анықталады. Әрбір і-итерация кезінде мынадай әрекеттер орындалады:

$$q_i = \begin{cases} 1, \text{ егер } 2R^{(i-1)} \geq B \\ 0, \text{ егер } -B \leq 2R^{(i-1)} < B \\ -1, \text{ егер } 2R^{(i-1)} < -B \end{cases}$$

$$R^{(i)} = 2R^{(i-1)} - q_i B$$

Жоғарыдағы өрнектен көрініп тұрғандай бөлінді басы артық есептеу жүйесінде көрсетіледі. Мұнда бөлінді цифрлары үш түрлі мән алады: -1,0,1.

Барлық n итерация аяқталған соң, егер соңғы қалдық теріс сан болса, онда қалдыққа және жекеленген қалдыққа түзетулер енгізіледі. Ол үшін қалдыққа бөлгішті қосамыз, ал бөліндіден төменгі разряд салмағымен бірді алып тастаймыз.

Бөлінгіш қосымша кодта QРг және RРг регистрлеріне, ал бөлгіш BРг регистріне қабылданады. Бөлу кезінде төмендегі әрекеттер орындалады:

Егер B бөлгішінің жоғарғы разрядтарында k нөлдер (B>0 саны үшін) немесе бірлік сандар (B<0 болғанда) жазылған болса, онда олар солға қарай k разрядтарға жылжытылады.

Бөлінгіш	Бөлгіш	q
000001001	0111	
000010010	1110	
000010010		0
000100100		
000100100		00
001001000		001
+ 010010000		
10010		
110110000		
110110000		0 0 1-1
+ 101100000		0 0 0 1
01110		
001000000		
00010		

6.37-сурет. SRT алгоритміне мысал

Қаралған процедураны 6.37–суретте келтірілген мысалмен көрсетуге болады.

0-ден n-1 дейінгі i үшін:

1. Егер жекеленген қалдықтың жоғарғы үш разрядтары бірдей мән алса, онда жекеленген қалдық (RРг және QРг) солға қарай бір разрядқа жылжытылады және қалдық разрядына q_i=0 жазамыз.
2. Егер жекеленген қалдықтың жоғарғы үш разрядтары әртүрлі болса және ЖҚ теріс болса, онда q_i=-1, әрі RРг мен QРг (ЖҚ) солға қарай бір разрядқа жылжытылып оған бөлгіш қосылады.

3. Егер жекеленген қалдықтың жоғарғы үш разрядтары әртүрлі болса және ЖҚ өзі оң сан болса, онда q_i=1 және ЖҚ (RPг және QPг) солға қарай бір разрядқа жылжытылып одан бөлгіш алынып тасталады.
4. Егер 2-пункт аяқталғаннан кейін теріс қалдық қалыптасса, онда түзету енгізіледі (қалдыққа бөлгіш қосылады, ал бөліндіден бір алынады).
5. Қалдық оңға қарай k разрядқа жылжытылады.

Бөлу амалын көбейту арқылы орындау

Көбейту операциясын жеделдете орындайтын аппараттық құрылғылардың (матрицалық, бұтақты көбейту сұлбалары) пайда болуына байланысты бөлу операциясын жеделдету мақсатымен бөлу операциясын көбейту операциясы арқылы орындауға мүмкіндік беретін алгоритмдер дами бастады. Олардың қатарына Голдшмит (Robert Elliott Goldshmidt) және Ньютон-Рафсон алгоритмдері жатады. Екі алгоритм де дұрыс бөлшектерді бөлуде, атап айтқанда жылжымалы үтірлі сандардың мантиссаларын бөлуде өте ыңғайлы.

Ньютон-Рафсон алгоритмі

Аталған алгоритмде А санын В санына бөлу операциясы В санының кері мәнін ($\frac{1}{B}$) А санына көбейтумен алмастырады, яғни

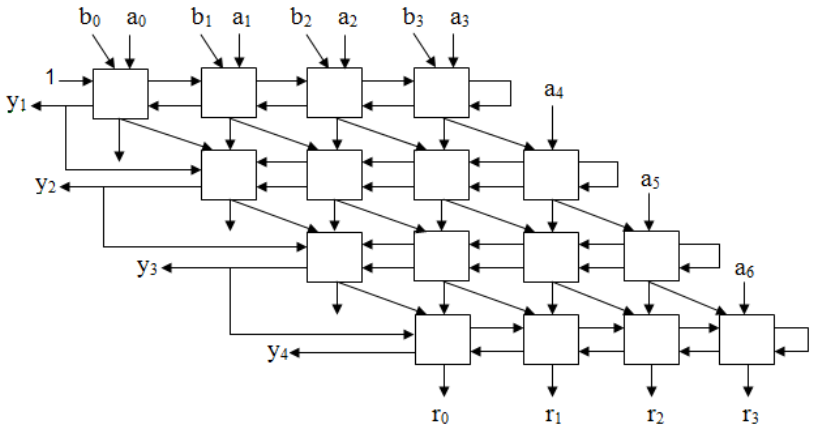
$Q=\frac{A}{B}=A*\frac{1}{B}$

Алгоритмнің ең басты міндеті $\frac{1}{B}$ санын тиімді жолмен анықтау болып табылады. Мұндай тәсілге Ньютон-Рафсон тәсілі жатады. Ол арқылы $f(x)=\frac{1}{x}-B=0$ теңдеуінің рекуренттік арақатынас арқылы түбірін табамыз. Ол арқылы $f(x)=\frac{1}{x}-B=0$ яғни ($X=\frac{1}{B}$) теңдеуінің $X_{i+1}=X_i\ (2-X_iB)$ рекуренттік арақатынас арқылы түбірін табамыз. Қалыптастырылған бөлгіштің ($0,5\leq B<1$) бастапқы мәні $X_0=2,9142-2B$ формуласы арқылы анықталады. Х мәнін есептеу $|X_{i+1}-X_i|\leq\Delta$ шарты орындалғанша жүргізіледі. Әдетте 32-разрядтық сандар үшін төрт итерация жеткілікті екен, ал 64-разрядты сан үшін - бес итерация керек. Сондықтан тәсілді n-разрядты санға қолданғанда $2[\log_2 n]-1$ көбейту операциясы қажет екен.

Қаралған тәсілді қолдануға мысал ретінде $A_0=0,217$, $B_0=0,450$ сандарын алайық.

$x_0=2,9142-2B=2,9142-0,90=2,014$

$x_1=2,014\ (2-2,014*0,450)=2,203;\quad |2,203-2,014|=0,189\geq\Delta$



x_2=2,203 (2-2,203*0,45)=2,225; |2,225-2,203|=0,022≥Δ

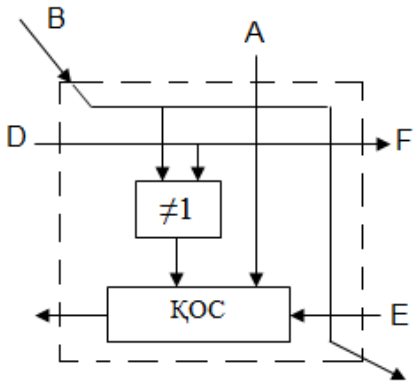
x_3=2,225 (2-2,225*0,45)=2,226; |1,226-2,226|≈0,000<Δ

үшінші итерациядан кейін $\frac{1}{B}=x_3=2,226$ және Q=0,217*2,226=0,483

Матрицалық бөлу сұлбасы

Жекеленген бөлінгішті матрицалық сұлба арқылы есептеуге болады (6.38-сурет). Сұлбада бөлінгіш $A=a_6a_5a_4a_3a_2a_1a_0$, бөлгіш $B=b_3b_2b_1b_0$, бөлінді $Q=q_3q_2q_1q_0$, қалдық $R=r_3r_2r_1r_0$.

Сұлба қалдықты қалпына келтірмей бөлу алгоритмімен бөледі. Әр қатардағы параллель қосындылауыштар алдыңғы жекеленген қалдықтың мәніне байланысты қосу немесе алу операциясын орындайды. Алу операциясы кірісіне берілген кодтармен екілік модуль бойынша қосу операцияларын орындайтын элементтер (≠1) арқылы және әрбір қосындылауыштар қатарының үшінші кірістеріне берілетін «1» арқылы қосымша код арқылы орындалады. Әрбір қалдық R_i (соңғы қалдық $R=r_3r_2r_1r_0$) қосымша кодта алынады.



а) б)

6.38-сурет. Матрицалық бөлу құрылғысы. а-құрылымдық сұлбасы, б-құрылғы ұяшықтары.

6.2. Жылжымалы үтірлі сандардың операциялық құрылғылары

Жылжымалы үтірлі сандардың ОПҚ бекітілген сандар ОПҚ қарағанда күрделі. Сондықтан соңғысы қазіргі есептеу машиналарында жеке құрылғы

ретінде құрылады. Мұндай құрылғыларда бүтін сандар ОПҚ сияқты төрт арифметикалар: қосу, алу, көбейту, бөлу операциялар орындалады.

6.2.1. Қосу және алу

X және Y сандар реттері $P_x > P_y$ болсын,

Онда :

$$\pm m_z qP_z = (\pm m_x qP_x) + (\pm m_y qP_y) = (\pm m_x qP_x + \left(\pm \frac{m_y}{qP_x - P_y}\right) qP_x$$

$$= \left(\pm \frac{m_x \pm m_y}{qP_x - P_y}\right) qP_x = \pm mzqP_z$$

Формулада m_x, m_y, m_z –қалыптастырылған операндтар және нәтиже мантиссалары; P_x, P_y, P_z – операндтар мен нәтижесі ығыстырылған реттері.

Қосу және алу алгоритмдерін төмендегідей ретпен көрсетуге болады:

Сандар реттері теңестіріледі. Модулі кіші санның ретін үлкен санның модуліне теңестіреміз. Ол үшін кіші санның мантиссасын реттер айырымына (ΔP) оңға жылжытамыз.

Мантиссаларды қосып (алып) нәтижесінде мантисса қосындысын (айырымын) табамыз.

Нәтиже ретінде үлкен санның ретін аламыз.

Алынған қосынды (айырымын) қалыптастырылады. Сан реттерін теңестіру оларды салыстырудан басталады. Реті кіші санның мантиссасын теңестіру кезінде оңға қарай реттер айырымына жылжытамыз.

Реттерді салыстыру кезінде бес түрлі жағдай болуы мүмкін:

1. P_x-P_y>m (m-мантисса разрядтар саны). Қосу нәтижесі ретінде бірдей бірінші қосылғышты алуға болады, реттерді теңеу үстінде екінші санның мантиссалары нөл мәнін алады.
2. P_x-P_y>m нәтиже ретінде екінші қосылғыш алынады.
3. P_x-P_y=0 мантиссалар жылжытылмай қосындыланады.
4. P_x-P_y=k_1 (k_1<P) екінші қосылғыш мантиссасы k_1 разрядтарына оңға жылжытылып, мантиссалары қосындыланады.
5. P_x-P_y=k_2 (k_2<m) мантиссаларын қосу алдында k_2 разрядына бірінші қосылғыш оңға жылжытылады.

Нәтиже ретінде үлкен санның ретін аламыз. Нәтиже алғаннан кейін, егер $|M_z|<\frac{1}{2}$ болса, онда қалыптастыру операциясы жүргізіледі. Ол үшін нәтиже мантиссасы солға бір разрядқа жылжытылып, нәтиже ретін 1-ге азайтамыз. Бұл операция $m_z \geq \frac{1}{2}$ шарты орындалғанша жалғасады.

Нәтиже мантиссасы нөл болса ($m_z=0$) қолыптастыру операциясы жүргізіледі. Егер нәтиже реті $+P_z$ болса және ол машинаның разряд сеткасынан шығып кетсе (оң аса толу), онда процессор рет аса толуын көрсететін үзу сигналын шығарады. Егер алынған нәтиже реті $-P_z$ болса және ол машина разряд сеткасынан шығып кетсе (теріс аса толу), онда процессор рет аса толуын көрсететін үзу сигналын шығарады. Егер алынған нәтиже реті $-P_z$ болса және ол машина разряд сеткасынан шығып кетсе (теріс аса толу), онда процессор нөлдік нәтижені көрсететін үзу сигналын шығарады. Операнд реттерін теңестіру кезінде разрядтары жылжытылатын мантиссаның төменгі разрядтары жоғалуы мүмкін, сондықтан жылжымалы үтірлі сандар жуықтатылып өңделеді. Қателік шамасын азайту үшін нәтиже дөңгелектенеді. Ол үшін ОПҚ қосындылауыш разрядты бір немесе бірнеше разрядтарға ұлғайтылады.

Дөңгелектеу операциясы орындалып болған соң, қалыптастыру нәтижесі өзгеруі мүмкін. Сондықтан дөңгелектеуден кейін қалыптастыру операциясы қайтадан жүргізілуі керек.

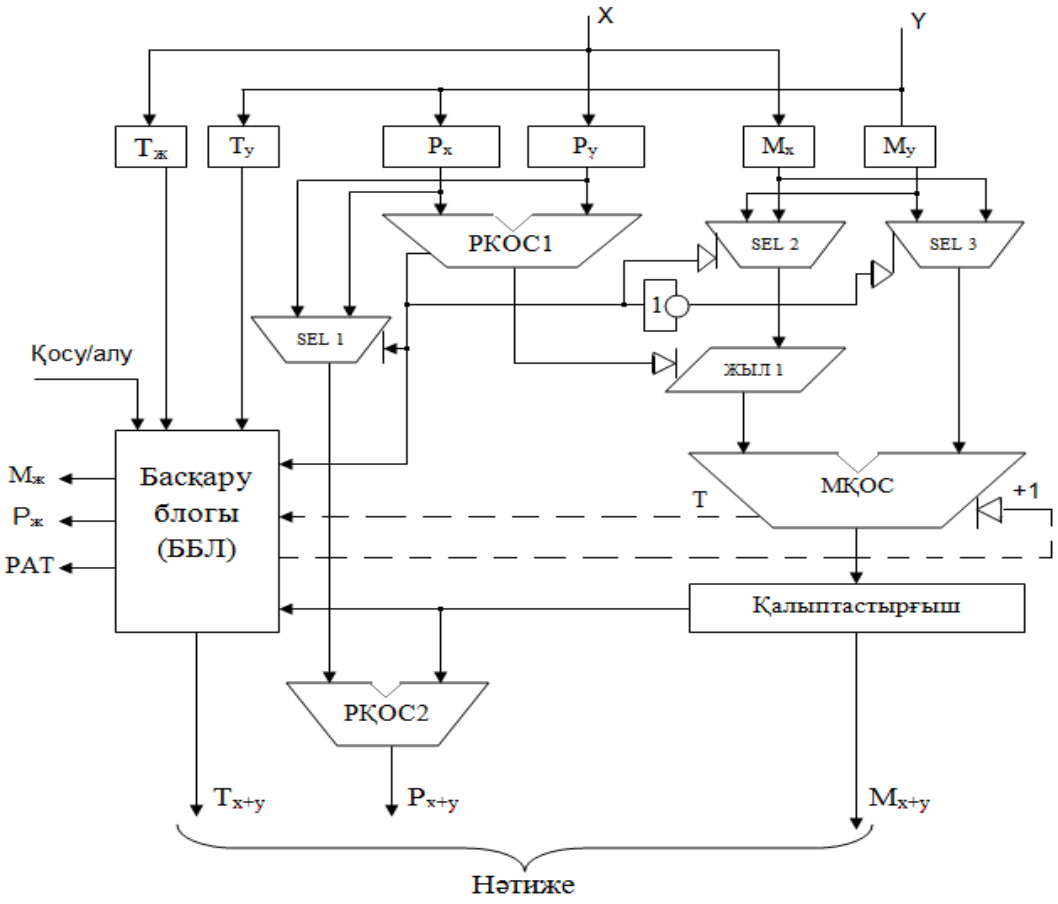
Жылжымалы үтірлі сандарды қосатын және алатын құрылғының құрылымдық сұлбасы 6.39–суретте көрсетілген.

Х және У сандарының таңбалары, реттері және мантиссалары T_x, T_y, P_x, P_y және M_x, M_y регистрлеріне қабылданады.

Реттерді теңестіру бірінші рет қосындылауыш (Р ҚОС1), селекторлар SEL1 және SEL2 және 1-жылжытқыш арқылы жүргізіледі. Ол үшін Х санының ретінен У саны Р ҚОС1 арқылы алынады. Нәтижесінде реттер айырымы (ΔР) және айырым таңбасы анықталады. Айырым таңбасы арқылы үлкен санның реті SEL1 селектордың шығысына беріледі.

Айырым таңбасының мәніне байланысты кіші санның мантиссасы ЖЫЛ1 жылжытқыш кірісіне беріледі. Жылжытқыш реттер айырымы ΔР мәніне байланысты кірісіне берілген мантиссаны оңға қарай жылжытады. Мантисса қосындылауышы (МҚОС) жылжытылған кіші санның мантиссасын үлкен санның мантиссасына қосып екі санның қосындысының (айырымын) қалыптастырады. Қосу (алу) үстінде мантиссалардың таңбалары (T_x және T_y) ескеріледі. Алынған нәтиже қалыптастырылады. Ол қалыптастырғыш құрамында қосынды разрядтарының жоғарғы разрядтарындағы нөлдер санын санайтын сұлба және қалыптаспаған мантиссаны ΔР-ға солға қарай

жылжытатын жылжытқыш арқылы іске асырылады. Нөлдер санауышында қалыптасқан санға үлкен санның реті Р ҚОС2 қосындылауышы арқылы түзету енгізіледі. Нәтиже мантиссасы қалыптастырғыш, рет мәні Р ҚОС2 қосындылауыш шығысында қалыптасады. Нәтиже таңбасын ББЛ басқару блогының шығысынан аламыз. ББЛ нәтиже таңбасынан басқа мантиссаның жоғалуы ($m_ж$), реттің жоғалуы ($Р_ж$) және реттер аса толуы ($Р_{ат}$) туралы сигналдар алынады.



6.39-сурет. Жылжымалы үтірлі сандармен қосу және алу амалын орындайтын құрылғы

6.2.2. Көбейту

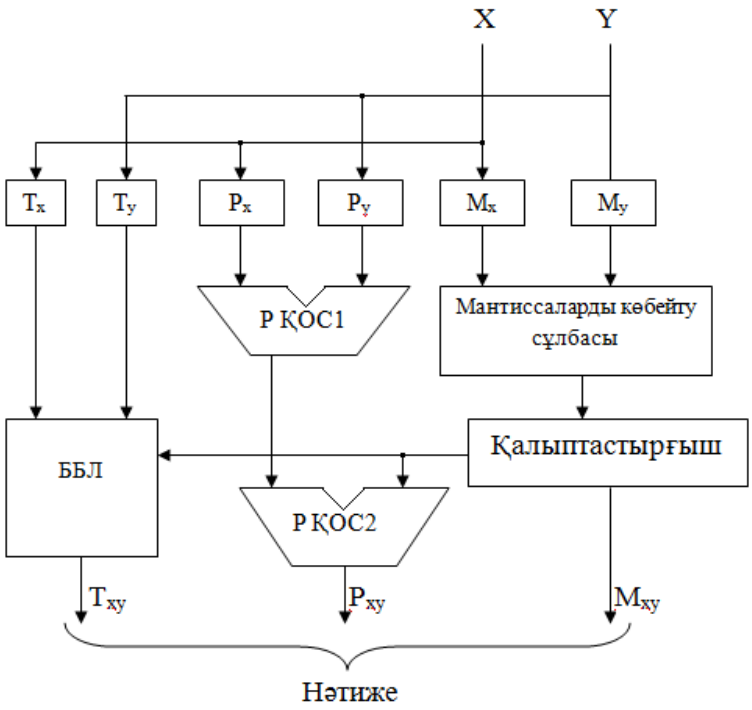
Жылжымалы үтірлі сандарды көбейту төмендегі формулаға сәйкес орындалады:

$$\pm m_z q P_z = (\pm m_x q P_x) \times (\pm m_y q P_y) = (\pm m_x \times m_y) q\,(P_x + P_y)$$

Жылжымалы үтірлі сандарды көбейткенде сан реттерді қосындыланады, ал мантиссалары көбейтіледі. Көбейтінді қалыптастырылып, егер көбейтілетін операндтар таңбасы бірдей болса, онда көбейтінді таңбасына плюс, бірдей болмаса, онда минус меншіктеледі.

Егер көбейгіш не көбейткіш мәндерінің бірі 0 болса, онда мантиссаларды көбейтпей көбейтіндіге 0 жазамыз. Егер рет мәндерін қосқанда аса толу пайда болса және рет теріс сан болса, онда мантиссаларды көбейтпей нәтиже ретінде 0 жазуымызға болады.

Егер сан реттерін қосу үстінде аса толу пайда болса және рет таңбасы оң болса, онда нәтиже мантиссасын қалыптастыру кезінде аса толу жойылып кетуі мүмкін. Жылжымалы сандарды көбейту құрылғысы 6.40–суретте келтірілген.



6.40–сурет. Жылжымалы үтірлі сандарды көбейту құрылғысы

6.2.3. Бөлу

Жылжымалы үтірлі сандарды бөлу төменде көрсетілген формула арқылы орындалады:

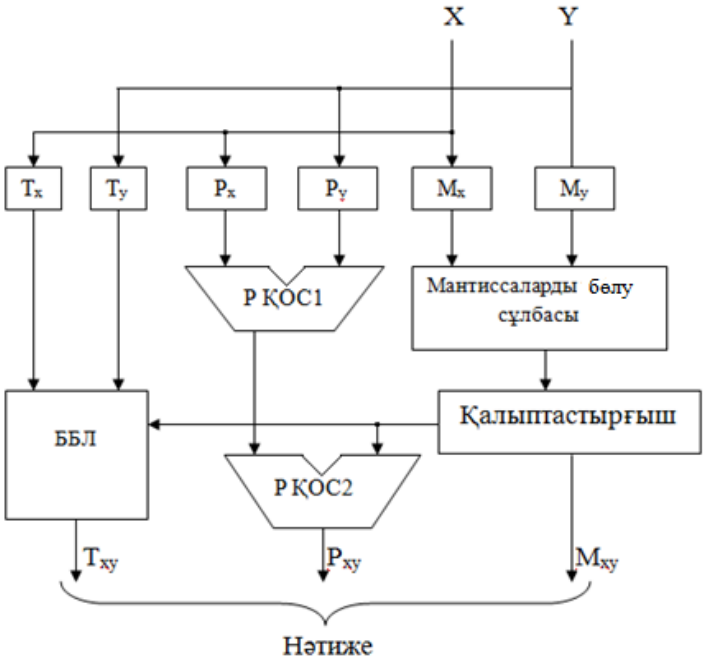
$$\pm m_z P_z = \frac{\pm m_x q P_x}{\pm m_y q P_y} = \left(\pm \frac{mx}{my}\right) P_x - P_y$$

Жылжымалы үтірлі сандарды бөлгенде бөлінді мантиссасы бөлінгіш мантиссасын бөлгіш мантиссасына бөлу арқылы, ал рет мәні бөлгіш пен бөлінгіш реттерінің айырымымен анықталады. Бөлінді қалыптастырылып, оған бөлінгіш пен бөлгіштің таңбалары бірдей болса, плюс, ал бірдей болмаса, минус мәнін меншіктейміз.

Қалыптастырылған жылжымалы үтірлі сандарды бөлгенде бөлінгіш мантиссасы бөлгіш мантиссасынан үлкен болуы мүмкін, онда бөлінді мәні аса толумен қалыптасады. Мұндай аса толуды болдырмас үшін, бөлгіш

разрядтарын операцияны орындау алдында солға жылжыту керек. Әдетте мантисса мәндерін қалдықты қалпына келтірмей бөлу алгоритмімен орындалады.

Жылжымалы үтірлі сандарды бөлу құрылғысының сұлбасы 6.41–суретте келтірілген. Көбейту құрылғысына ұқсас. Тек мантиссаларды көбейту блогының орнына бөлу құрылғысын қосу керек. Сан реттері қосылудың орнына бөлінгіш ретінен бөлгіш реті алынады.



6.41–сурет. Жылжымалы үтірлі сандарды бөлу құрылғысы

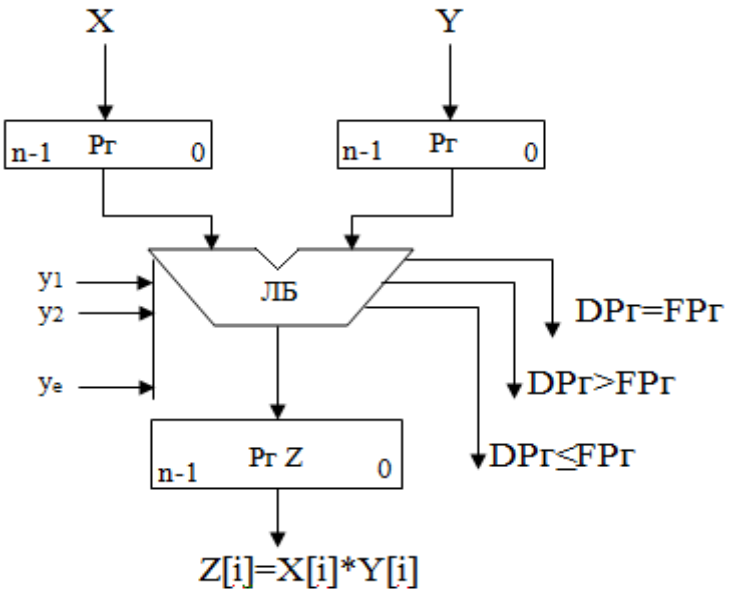
6.3. Логикалық операцияларды орындау

Есептеу машиналарында арифметикалық операциялардан басқа негізгі логикалық операциялар және жылжыту операциялары орындалуы керек. Мұндай операциялар бүтін сандармен жұмыс жасайтын, ОПҚ құрамына кіретін қосымша логикалық блоктар (ЛБ) арқылы орындалады. Логикалық блоктарда разряд аралық логикалық операциялар: екі модулімен қосу, логикалық қосу, көбейту, разряд мәндерін салыстыру т.б. операциялар орындалады.

Логикалық операцияны орындауға арналған блоктың құрылымы 6.42-суретте көрсетілген.

Есептеу машиналарында Буль айнымалысы бір битпен көрсетіледі, бірақ іс жүзінде логикалық операциялар бірден көптеген логикалық айнымалылармен (байтпен) немесе машиналық сөздердің барлық биттерімен бір логикалық операция орындалады. Мұндай өңдеулерде разряд аралық тасымалдар болмайды. Логикалық блок жоғарыда аталған барлық операцияларды

орындауға мүмкіндік береді. Орындалатын операциялар y_1, $y_2 \ldots y_e$ басқару сигналдары арқылы іске асырылады. Нәтиже ZPг регистрінде қалыптасады.



6.42-сурет. ЛБ құрылымдық сұлбасы

Бақылау сұрақтары

1. Қатаң құрылымды ОПҚ артықшылығы мен кемістігі неде?
2. Магистральді ОПҚ қатаң құрылымды ОПҚ саластырып олардың артықшылығы мен кемістіктерін көрсетіңіз
3. Қосу/алу блогында алу операциясы қосу операциясымен қалай ауыстырылады?
4. Магистральді ОПҚ түрлерін салыстырып олардың артықшылығы мен кемістіктерін көрсетіңіз
5. Логикалық және аппараттық жеделдету тәсілдерінің айырмашылығы неде?
6. Бут тәсілімен көбейтуге алгоритм құрыңыз
7. Көбейтудің Браун сұлбасы мен Уоллес сұлбасын салыстырыңыз
8. Көбейтуді жеделдетудің аппараттық тәсілдерінің ерекшеліктері
9. Браун көбейткішіне конвейер құрыңыз
10. Ұлғаймалы аппараттық көбейту құрылғысын синтездеудің қандай тәсілдерін білесіз?
11. Бүтін сандарды бөлу операциясын жеделдетудің қандай тәсілдерін білесіз?
12. Жылжымалы үтірлі сандармен орындалатын операциялардың қайсысы күрделі?
13. Жылжымалы үтірлі сандармен әртүрлі операция орындағанда нәтиже қалай қалыптастырылады?
14. Мантиссаның жоғалуы деп нені айтамыз?

7. ЖАДЫ ҚҰРЫЛҒЫЛАРЫ

7.1. Жады құрылғыларының сипаттамалары

Кез келген есептеу машиналарында деректер оның жады құрылғыларында сақталады. Жады құрылғыларының атқаратын қызметтеріне деректерді қабылдау (жазу), оларды сақтау және сыртқа беру (оқу) жатады. Сонымен жады құрылғылары үш режимде – жазу, сақтау, оқу режимінде жұмыс жасайды. Оқу және жазу үдерістерін бірге жады құрылғысына *қатынас құру* үдерісі деп атайды.

Жады құрылғылары олардың сипаттамасына, орналасу орнына және процессормен алмасу тәсілдеріне байланысты екі: ішкі және сыртқы жады түрлеріне бөлінеді. Ішкі жадының бір бөлігі процессормен бір кристаллда орналасады (регистрлік жады, КЭШ-жады), ал қалған бөлігі жүйелік тақтайшада (негізгі жады, жоғарғы деңгейлі КЭШ-жады) орналасады. Жылдамдығы төмен, бірақ сыйымдылығы жоғары жады құрылғылары (қатты денелі, магниттік және оптикалық дискалар, магниттік таспалар) есептеу машиналарының сыртқы жадына жатады.

Жады құрылғысының сипаттамаларына мыналар жатады:

- Сыйымдылық;
- Қатынас құру тәсілі;
- Жылдамдық;
- Физикалық түрі;
- Құны.

Жады сыйымдылығы жады құрылғысын құратын ұяшықтар санымен анықталады. Бір жады ұяшығының сыйымдылығы бір битке тең. Сондықтан жады сыйымдылығы да битпен өлшенеді. Әдетте деректер одан ірі өлшемдер арқылы өлшенеді. Оларға килобит (K-kilo), мегобит (M-mego), гигабит (G-giga), тера (T-tera) т.б. жатады.

1Кбит=2^{10}=1024 бит, 1Мбит=2^{20}=1048576 бит т.с.с. Мысалы, егер жады сыйымдылығы 16 Кбит болса, онда жады құрылғысында 16х2^{10} немесе 16384 ұяшықтар болады.

Жады құрылғыларында төрт түрлі *қатынас құру тәсілдері* қолданылады. Олар: тізбекті, тура, еркін және ассоциативтік қатынас құру тәсілдері.

Еркін қатынас құру тәсілінде жады құрылғыларының әр ұяшығына бірегей физикалық мекен-жай (мекен-жай) меншіктеледі. Құрылғының кез келген ұяшығына қатынас құру бірдей уақыт алады және қатынас кез келген (еркін) тәртіпте орындалуы мүмкін. Мұндай құрылғыларға негізгі жады жатады.

Ассоциативтік тәсілде жады құрылғысына жазылатын немесе одан оқылатын деректер жады мекен-жайлары емес деректің белгілері арқылы жүргізіледі. Ол үшін жады ұяшығындағы дерек белгілері сұраным белгілерімен параллель салыстырылады. Ассоциативтік тәсіл КЭШ-жадыны, ауани жадыны құрғанда қолданылады.

Жады құрылғысының жылдамдығы оның негізгі сипаттамасына жатады. Оның жылдамдығын анықтау үшін әдетте төменде аталған төрт параметр қолданылады:

- *Деректерді іріктеу уақыты.* Оған деректі оқу операциясының басталу уақыты мен оқылған деректі жады құрылғысынан сыртқа беру уақыты аралығын айтады.
- *Деректі сақтау уақыты* - жады құрылғысында жазылған деректі қайта жазу уақытына дейін сақтаудың уақыт аралығы.
- *Жады құрылғысына қатынас құру циклі* немесе қатынас құру периоды ($Т_Ц$). Қатынас құру циклі – жады құрылғысына жасалынатын екі қатынас құру сигналдары тізбегінің минималды уақыт аралығы.
- *Деректерді беру жылдамдығы* – бір уақыт аралығында жады құрылғысынан оқылатын немесе оған жазылатын деректер мөлшері.

Жады құрылғыларының *физикалық түріне* келсек, қазіргі есептеу машиналарында ішкі жады жартылай өткізгіш технологиясына негізделеді. Олар энергияға тәуелді немесе оған тәуелсіз болып келеді. Энергияға тәуелді жады құрылғыларында қорек көзін алып тастаса, онда сақталған деректер өзгеріп кетуі немесе жоғалуы мүмкін.

Энергияға тәуелсіз жады құрылғыларында қорек көзіне тәуелді болмайды.

Жады құрылғыларының құны олардың жалпы құнының битпен саналатын сыйымдылық мөлшерінің қатынасымен, яғни бір бит деректі сақтау құнымен анықталады.

Жады құрылғыларының сипаттамалары бір-бірімен қарама-қайшылықта болады. Мысалы, сыйымдылығы үлкен жадының жылдамдығы да үлкен болмайды, ал жылдамдығы жоғары жады құрылғысының құны өте жоғары болады. Жады жүйесін құрарда бағасын ескере отырып, кезекті сыйымдылық пен жылдамдығы жоғары болатын әртүрлі жады құрылғысын таңдау керек.

Осыған байланысты машиналар жады әртүрлі типті көп сатылы иерархиялық жады жүйесі түрінде құрылады:

Регистрлік жады құрылғылары ($Р_{ГЖ}$). Олар процессордың құрамына кіреді. Регистрлік жадылардың арқасында процессордың құрамына кірмейтін, алмасу уақыты жоғары басқа деңгейдегі жады құрылғыларына қатынас санын азайтуға мүмкіндік береді. Қазіргі есептеу машиналарында $Р_{ГЖ}$ сыйымдылығы 32÷64 бит. КЭШ-жады кезекті операцияға қатысатын деректердің көшірмелерін сақтайды. Қазіргі есептеу машиналарында үш деңгейлі КЭШ-жады қолданылады: L1 деңгейлі (сыйымдылығы 32 КБ-64КБ, жылдамдығы 10 нс), L2 деңгейлі (сыйымдылығы 256 КБ-2МБ, жылдамдығы 25нс), L3 деңгейлі (сыйымдылығы 4МБ-8МБ, жылдамдығы 50 нс). Жылдамдығы жоғары КЭШ-жадылары арқылы есептеу машинасының өнімділігі артады.

Негізгі жады (жедел, тұрақты және жартылай тұрақты) процессормен алмасу режимінде жұмыс жасайды. Ағымды моментте орындалып жатқан бағдарламаның фрагменті міндетті түрде негізгі жадыда сақталады. Негізгі жады сыйымдылығы 1Гб-4Гб, жылдамдығы - 50-100 нс.

Қатты денелі дисктер. Соңғы кезде пайда болған жадының жартылай өткізгіштік шағын сұлбасы. Қатты денелі дисктер магниттік дисктер жинақтағышын конструкциясы және интерфейстері бойынша толығымен алмастыра алады. Мұндай дисктерде қозғалатын бөліктері жоқ, сондықтан керекті қуат мөлшері төмен, дерекке қатынас құру уақыты төмен және сенімділігі жоғары. Сыйымдылығы 256 Гб÷1 Тб, жылдамдығы 50-200 нс.

Магнитті дисктер. Қазіргі ЕМ ең көп тараған сырт құрылғысы. Деректер жазық магниттік емес материалдардың (алюминий, металлдар қоспасы, шыны) беттеріне орналасқан магниттелетін материалдарда сақталады. Ақпараттар сыйымдылығы 512 Гб÷6 Тб, жылдамдығы 2,5-16 мс.

Оптикалық дисктер. Жылдамдығы 150-400 мс оптикалық дисктер үш типке бөлінеді. Олар компакт-диск (CD-compact disk), сыйымдылығы 700 МБ, DVD-диск (Digital Versatile Disk) – цифрлы әмбебап диск. Сыйымдылығы – 4,7Гб÷8,5Гб BD (Blye-ray) – көгілдір сәуле-диск. Оптикалық дисктердің кезекті буыны. Blye-ray технологиясында деректерді жазу және оқу үшін толқын ұзындығы 405 нм болатын көгілдір-күлгін түсті лазер қолданылады. Сыйымдылығы 25 Гб-128 Гб.

*Магниттік таспалар.*Деректер жұқа полистирал таспасының бетіне орналасқан магниттелетін материалдарда сақталады. Сыйымдылығы 5Тб-100Тб, жылдамдығы 500-1000 мс.

7.2. Жады құрылғысының процессормен байланысын ұйымдастыру.

Жады құрылғысының өлшемі есептеу машиналарындағы мекен-жай өрісінің разряд санымен анықталады. Мысалы, егер машина 16-разрядты мекен-жайды генерацияласа, онда ол сыйымдылығы 2^{16}=24 кбайт мекен-жайлы сақтау бірлігі бар жадымен жұмыс жасай алады, ал 40-разрядты мекен-жайы бар есептеу машиналарында қатынас құратын жадының сыйымдылығы 2^{32}=1 Тбайт мекенделетін сақтау бірлігімен анықталады. Жады құрылғысының мекенделетін сақтау бірлік санымен мекен-жай кеңістігінің өлшемі анықталады. Қазіргі көптеген машиналар жадыға байтпен мекенделеді.

Жады құрылғысының процессормен деректер алмасуын ұйымдастыру сұлбасы 7.1–суретте көрсетілген. Жады мен процессор арасында деректерді алмастыру екі регистрлер - дерек регистрі (МРг) және мекен-жай регистрі (МЖРг) арқылы жүргізіледі. Егер регистр МРг k биттерден, ал МЖРг – n биттерден тұрса, онда жады құрылғысы 2^k мекен-жайлы сақтау бірліктерінен тұрады. Қатынас құрудың бір циклында жады мен процессор n бит деректер алмасуды жүзеге асырады.

Деректер процессордың k мекен-жайы және n дерек арқылы беріледі. Процессорда жоғарыда аталған шиналардан басқа деректерді беру үдерісін басқаратын сигналдарды беретін басқару шинасы бар. Басқару шинасы арқылы R/$\overline{W}$ (Read/$\overline{Write}$) және MFC (Memory Function Compelled) сигналдары беріледі.

7.1-сурет. Жады мен процессор дерек алмасуын ұйымдастыру сұлбасы

Деректі жадыдан оқу үшін, процессор алдымен МЖРг регистріне мекен-жайын жүктеп R/$\overline{W}$шинасына 1 мәнін береді. Оған жауап ретінде дерек шинасына жады деректі беретін MFC сигналын береді. MFC сигналын алып

болған соң, процессор дерек шинасынан деректі МРг регистрге қабылдайды. Деректі жадыға жазу үшін процессор МЖРг регистріне мекен-жайды, ал МРг регистріне деректі жүктеп басқару шинасына R/$\overline{W}$=0 сигналын береді.

7.3. Жады құрылғыларының жіктелуі

Жады құрылғылары деректерге қол жеткізу тәсілдеріне байланысты үш түрге бөлінеді. Олар: мекен-жайлы, тізбекті (мекен-жайсыз) және ассоциативтік.

Мекен-жайлық қолжеткізімде жадының мекен-жайы кірісіндегі код оның дерекпен алмасатын ұяшығын көрсетеді.

Мекен-жайлық ЖҚ екі түрге RAM (Random Access Memory) және ROM (Read-Only Memory) болып бөлінеді. RAM терминінің қазақша синонимі: ЕҚЖҚ (еркін қатынасты жады құрылғы). Ол арқылы жедел жады (ЖЖ) және аралық жады (КЭШ жады) құрылады. Жедел жады ағымдағы орындалатын бағдарламаларды және оларға керекті деректерді сақтайды. Қазіргі ЖЖ энергияға тәуелді. ROM – (қазақша баламасы - ТЖ-тұрақты жады). Мұнда жазылған деректер мүлде өзгермейді немесе арнайы режимде өте сирек өзгереді. Жұмыс режимінде ТЖ-дан дерек тек оқылады, яғни ол код генераторы болып табылады. ТЖ-ға мәліметтер бағдарламалау арқылы жазылады. Бағдарламаланатын ТЖ екі түрге бөлінеді. Бірінші түрін Mask ROM (ROM (М)-деп белгіленеді) деп атайды. ТЖ мұндай түрін қазақша бүркемелі (маскалы) БТЖ деп атауға болады. Бүркемелі ТЖ деректер өндіріс орындарында бүркеме (маска, үлгі) арқылы технологиялық үдерістің аяқталу кезеңінде жазылады.

БТЖ жады элементтері ретінде диодтар, биполярлы транзисторлар және МОП-транзистор т.б. қолданылады. Бағдарламаланатын ТЖ екінші түріне пайдаланушылармен бағдарламаланатын ТЖ –PROM, EPROM-OTP, EPROM, EEPROM, FLASH жатады. Аталған ТЖ бірінші төртеуінің атауында Р әріпі Programmable - бағдарламаланатын деген сөздің бірінші әріпі. ТЖ PROM, EPROM-OTP – бір рет бағдарламаланатын ТЖ. Деректі пайдаланушылар зертханалық жағдайда қарапайым бағдарламалауыш арқылы жаза алады. EPROM, EEPROM және FLASH тұрақты жады түрлерінде жазылған деректерді өшіріп, олардың орнына жаңа деректерді жазуға мүмкіндіктері бар. Мұндай ТЖ түрлерінде деректі оқу үлкен жылдамдықпен орындалады.

EPROM шағын сұлбасында ескі деректер ультракүлгін сәулелермен өшіріледі, ал жазу қалған ТЖ сияқты электр сигналдарымен жазылады.

EEPROM (E^2PROM) шағын сұлбаларында деректерді өшіру де және жазу да электр сигналдарымен іске асырылады.

FLASH шағын сұлбасының сақтау элементтері EPROM және EEPROM ТЖ элементтеріне ұқсас, бірақ бұл сұлбаның құрылымдық және технологиялық өзгешеліктері бар. Алғашқы FLASH қайта бағдарламалау кезінде бұрынғы жазылған барлық деректерді өшіру керек болатын. Кейінгі микросхемаларда жады элементтері блок түрінде ұйымдастырылған. Бұл деректерді қайта бағдарламалау үдерісін жеңілдетуге алып келді.

RAM статикалық және динамикалық болып екі түрге бөлінеді. Статикалық RAM (SRAM) жады элементі ретінде триггерлер қолданылады. Триггерлер күйі оның кірісіне жазылатын жаңа дерек берілгенше немесе қорек көзінен ажыратылғанша сақталады. Динамикалық RAM (DRAM) деректер МОП-құрылым элементтерімен түзелген конденсатор зарядтары түрінде сақталады. Конденсатордың өзіндік разрядталуымен дерек бөлінеді, сондықтан олар белгілі бір уақыт аралығында (әрбір бірнеше миллисекундта) қайта қалпына келтіріліп тұру керек (регенерация). Бұл DRAM жады түрінің кемістігіне жатады. Бірақ DRAM элементтерін тығыз орналастыруға болады. Ол жады сыйымдылығын арттыруға мүмкіндік береді. Сондықтан мұндай жадының құны жылдамдығы жоғары статикалық жадыдан айтарлықтай төмен.

Статикалық ЖЖ асинхронды және синхронды деп екіге жіктелінеді. Асинхронды ЖЖ қатынас кез келген уақытта басталғаннан кейін дерек жады құрылғысының ішкі параметрлеріне байланысты белгілі бір уақыттан кейін процессордың параметріне байланыссыз шығарылады. Мұндай жағдайда ЖЖ мен процессордың деректермен алмасу кезінде қосымша кідірістер болуы мүмкін.

Синхронды ЖЖ жады жұмысының кезеңдерінің уақыт шамасы жүйе синхросигналдарымен қатаң байланысқан. Мұндай байланыс жады мен процессордың деректермен алмасу кезінде артық уақыт шығынын болдырмайды және деректі конвейермен өңдеуге мүмкіндік туғызады.

Статикалық ЖЖ бірпортты және көппортты түрде құрылады. Көппортты ЖЖ екі немесе одан көп ұяшықтарына бір уақытта қатынас құруға мүмкіндік бар. Мысалы, екі портты ЖЖ бір ұяшыққа дерек жазылса, сол уақытта екінші ұяшықтан дерек оқылуы мүмкін.

Динамикалық ЖЖ информациялық сыйымдылығы өте жоғары, ал құны төмен болғандықтан олар негізінен ЖЖ құруға арналған.

Статикалық ЖЖ динамикалық ЖЖ-дан 4-5 есе қымбат, ал сыйымдылығы жағынан сонша есе аз. Ең үлкен артықшылығы – жылдамдығы жоғары. Сондықтан статикалық ЖЖКЭШ-жадыларын, FIFO және LIFO тәртібінде жұмыс жасайтын аралық жады құруға қолданады. Бұлардан басқа статикалық

жады микроконтроллер құрылғыларында жылдамдығы жоғары коммуникациялық құрылғыларда т.с.с. кеңінен қолданылады.

Деректерге тізбекпен қолжеткізетін ЖҚ деректер ЖҚ ұяшықтарына тізбектеліп кезекпен жазылады. ЖҚ деректерді оқығанда жазылу тәртібімен не кері тәртіппен оқылады.

FIFO (First In-First Out) аралық жады «бірінші жазылған бірінші оқылады» тәртібімен жұмыс жасайды. FIFO тәртібімен жұмыс жасайтын ЖҚ-да файлдық және циклдық ЖҚ қолданылады.

Файлдық ЖҚ деректер тізбектің басында жазылып ЖҚ шығысында тізбек элементтерінің санымен анықталатын бірнеше қатынастан кейін пайда болады.

Циклдық ЖҚ сөздер бірінен кейін бірі жады сыйымдылығымен анықталатын периодпен қол жеткізіледі.

Кері тәртіппен оқылатын ЖҚ стектік ЖҚ-ға тиесілі. Стектік ЖҚ «соңынан келген бірінші шығады» тәртібімен жұмыс жасайды. Мұндай ЖҚ LIFO (Last in- First Out) *аралық жады* деп аталады.

Ассоциативтік қатынас құру жады құрылғыларында деректер олардың жадында орналасу тәртібіне байланысты емес (мекен-жайы немесе кезектегі орны емес) қайсыбір белгілер арқылы іріктеледі.

ЖҚ әрбір сөзі бір уақытта берілген белгілерге сәйкестігі тексеріледі. Мысал ретінде сөздің белгілі бір өрісі (тег-ағылшын тілінен tag) кіріс сөзімен берілетін (тег мекен-жайымен) белгісімен салыстырылады. ЖҚ сыртына белгілері шартына сәйкес келген сөздер таңдалып шығарылады. Ассоциативтік жады КЭШ-жадыны құруда қолданылады.

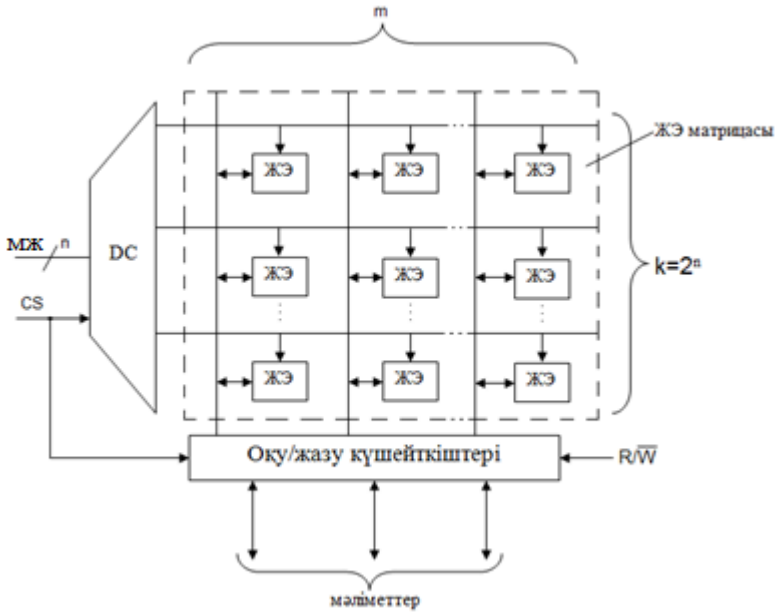
7.4. Жады құрылғыларының негізгі құрылымдары

Динамикалық жадының статикалық жадыдан ішкі құрылымының өзгешелігіне қарамастан (регенерациялау сұлбалары) оларды құратын жады элементтерін біріктіру жолдары бірдей. Статикалық және динамикалық ЖҚ шағын сұлбалары екі (2D), үш (3D) өлшемді (D-dimension-) және 2DM өлшемді түрінде құрылады.

2D ЖҚ құрылымы

Екі өлшемді ЖҚ жады элементтері (ЖЭ) тік бұрышты матрица түрінде орналасады. Оның өлшемі M=k*m, мұнда M - ЖҚ битпен өлшенетін ақпараттың сыйымдылығы, k-сақталатын сөз саны, m-сөз разрядтылығы (7.2-сурет).

ЖҚ контроллерінен келетін таңдау сигналы CS (Chip Select) белсенді болса, онда мекен-жай кодының дешифраторы DC матрица қатарын (ұяшықтарын) таңдап, оған қосылған жады элементтерінің барлығына бірдей қатынас құрады. Ұяшық мекен-жайы матрицадағы қатар нөмірімен анықталады. Матрицадағы баған элементтері тік желілерімен қосылып разряд (оқу/жазу) желілерін құрады. Әр баған бойында орналасқан элементте барлық сөздерінің бір разрядтарын сақтайды. Деректер алмасу бағыты R/$\overline{W}$ (Read-оқу,Write-жазу) сигналдарымен басқарылатын оқу/жазу күшейткіштерімен анықталады.



7.2-сурет. Екі өлшемді (2D) ЖҚ құрылымы

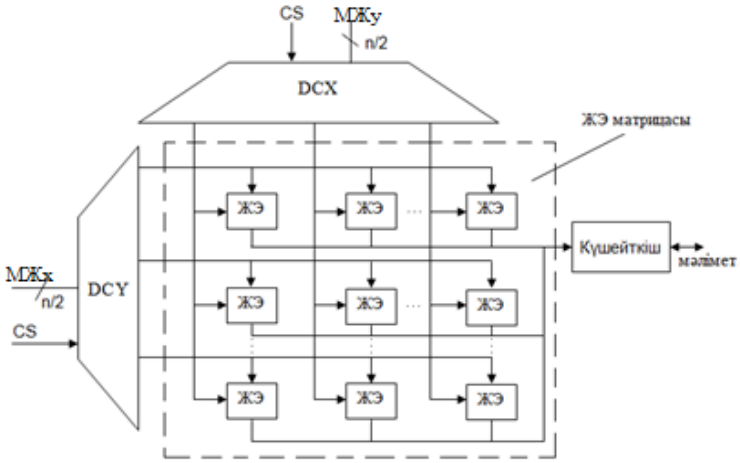
Екі өлшемді ЖҚ арқылы сыйымдылығы үлкен ЖҚ құрғанда мекен-жай кодының DC-ін құру қиынға соғады, өйткені DC-тің шығыс саны ЖҚ сақталатын сөз санына тең болуы керек. Сондықтан 2D жүйесі арқылы сыйымдылығы аз ЖҚ құрады.

3D ЖҚ құрылымы

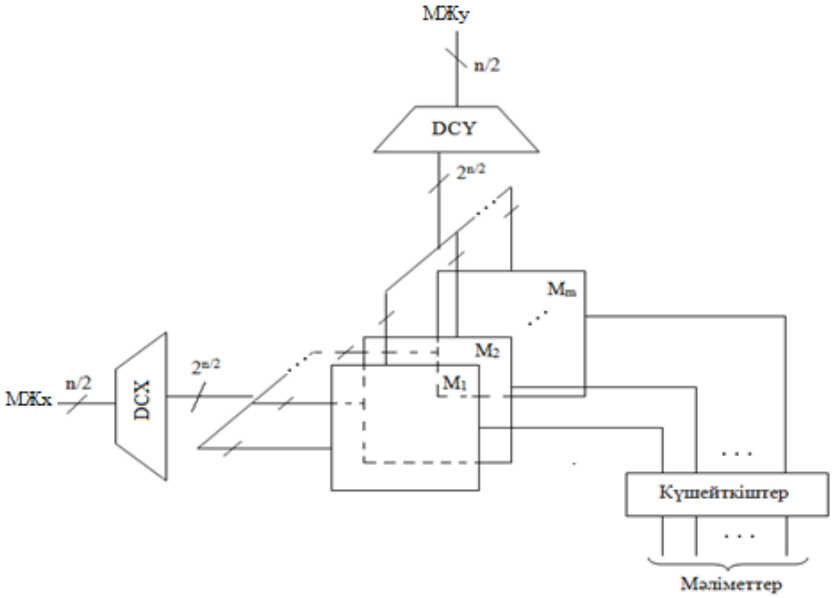
Сыйымдылығы үлкен ЖҚ құрғанда мекен-жай шифраторын ықшамдау үшін 3D жүйесі кеңінен қолданылады. 3D жүйесінде ЖЭ екі координатты іріктеу тәсілі арқылы іріктеледі. Бір разрядты 3D ЖҚ құрылымдық сұлбасы 7.3-суретте көрсетілген. Мұнда мекен-жай коды n екі бөлікке МЖХ (қатар мекен-жайы) және МЖҮ (бағана мекен-жайы) бөлінген. Әрбір бөліктің өз дешифраторлары бар (DCX, DCY).

Матрицадағы ЖЭ DCY және DCX шығыстарынан берілген сигналдар ЖЭ кірістеріне бір уақытта берілсе ғана таңдалынады. Таңдау саны $2^{n/2}x2^{n/2}=2^n$.DCY және DCX шығыс сандарының қосындысы $2^{n/2}+2^{n/2}=2^{n/2+1}$, формуладан $2^{n/2+1}<2^n$.

Яғни ұяшық саны n ЖҚ 2D жүйесімен құрсақ, дешифраторы шығыстарының саны 2^n, ал 3D жүйесімен құрғанда ол $2^{n/2+1}$. Мысалы n=10 болғанда 2D ЖҚ үшін дешифратор шығысының саны $2^n=2^{10}=1024$, ал 3D ЖҚ үшін $2^5+2^5=32+32=64$.



7.3-сурет. Бір разрядты 3D ЖҚ ұйымдастыру сұлбасы



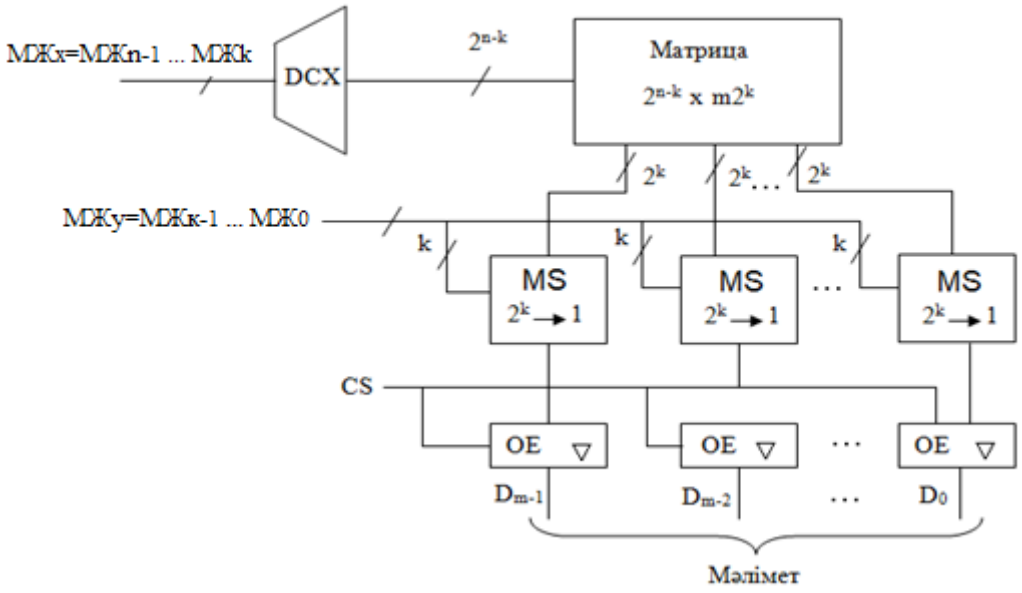
7.4-сурет. Көпразрядты 3D ЖҚ құрылымы

Бір разрядты 3D ЖҚ негізінде көп разрядты 3D ЖҚ құруға болады. Сондай ЖҚ құрылымы 7.4–суретте келтірілген. ЖҚ $M_1 \div M_m$ ЖЭ матрицасы бар. Матрица саны ЖҚ разрядтылығымен, әр матрицадағы ЖЭ саны ЖҚ ұяшық санымен анықталады. DCX дешифраторының әр шығысы әр матрицаның $2^{n/2}$ қатарының ішінен біреуін таңдайды, ал DCY дешифраторының әр шығысы әр матрицадан бір бағананы таңдайды. Нәтижесінде әр матрицадан бір ЖЭ таңдалып, деректі оқу кезінде m разрядты сөз оқылады.

2DM ЖҚ құрылымы

3D ЖҚ қолданылатын екі координатты ЖЭ құру 2D ЖҚ ЖЭ құрудан күрделі, ал 3D ЖҚ мекен-жай дешифраторы 2D ЖҚ мекен-жай дешифраторына қарағанда қарапайым. Қаралып отырған 2DM (2D-модификацияланған) жады құрылғысында 2D мен 3D жады құрылғыларының артықшылықтары ескерілген (7.5-сурет). DCX дешифраторының шығысына қосылған қатарлардағы ЖЭ 2D ЖҚ сияқты бір қатарды таңдайды. Бірақ әр қатардағы ЖЭ саны $m2^k$ ЖЭ элементтерінен тұрады. Яғни әр қатарда 2^k ЖЭ тұратын m топ болады. Осыған сәйкес матрицаның қатар саны азаяды. Ол DCX шығыс санын азайтатын мекен-жай кодының разряд санын $A_{n-1}…A_0$-дан $A_n…A_k$ санына алып келеді. Ал қалған $A_{к-1}$ -ден A_0 –ге дейінгі мекен-жай разрядтары DCX арқылы 2^k ЖЭ тұратын әр m топтың ішінен біреуін іріктеуге мүмкіндік береді. Соңғы іріктелу мультиплексор (MS) арқылы жүргізіледі. Мультиплексордың мекен-жай кірісіне $A_{n-1}…A_0$ кодын береміз. Мультиплексор таңдап алған биттер шығыс сөзін құрады.

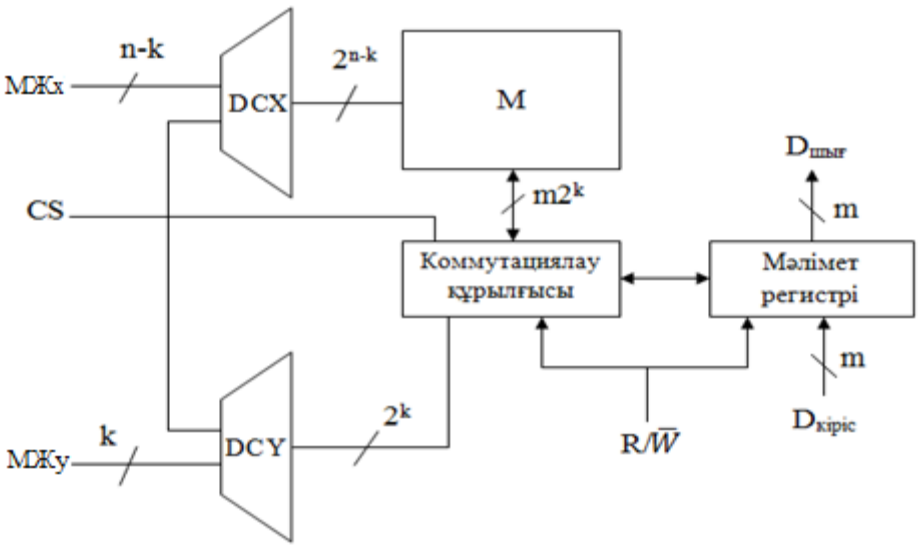
Шығыс сөзі CS сигналының рұқсатымен оны үш күйде бола алатын арнайы сұлба (OE) арқылы шығыс шинасына береді.



7.5-сурет. 2DM ЖҚ (ТЖ үшін)

7.5-суретте 2DM негізінде құрылған ЖЖ құрылымдық сұлбасы келтірілген. 7.6–суретте DCX, DCY дешифраторы мен матрицадан басқа оқылатын және жазылатын деректерді уақытша сақтайтын, CS, R/$\overline{W}$ сигналдары арқылы және DCY дешифраторының шығысымен басқарылатын коммутациялау құрылғысы (КҚ) және дерек регистрі (МРг) көрсетілген. Жазылатын деректер КҚ-ға беріледі. Оқылған деректер де осы сұлбадан

алынады. Коммутациялау сұлбасы мультипликациялау қызметінен басқа R/$\overline{W}$ сигналдарына байланысты деректердің алмасу бағытын өзгертіп отырады.



7.6-сурет. 2DM негізінде құрылған жедел жады құрылымы

7.5. Жады шағын сұлбасының құрылымы

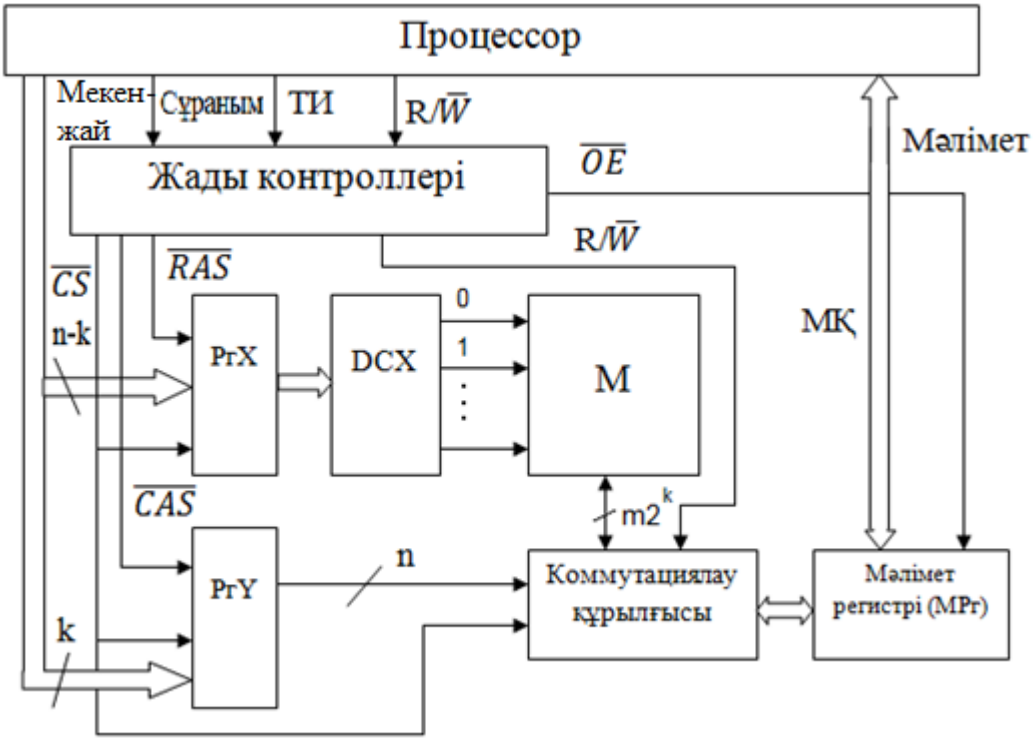
Жоғарыда ЖҚ ядросын құратын ЖЭ біріктіру және оларды таңдау тәсілдеріне байланысты құрылған 2D, 3D, 2DM жүйелерін қарадық.

Интегралды шағын сұлбаларда (ИШС) түйіспе санын азайту үшін қатар мекен-жайы мен бағана мекен-жайлары сұлба кірістеріне уақыт бойынша тізбектеле беріліп (мультиплекстеніп) сәйкес қатар мекен-жайлары регистріне және бағана мекен-жайларының регистріне жазылады.

7.7-суретте 2DM негізінде құрылған шағын сұлба құрылымы көрсетілген.

Процессордан жады құрылғысынан «сұраным» сигналы, операция коды (R/$\overline{W}$), такт импульстары (ТИ) және мекен-жай коды (А) беріледі. Жазу командасы орындалғанда жадының дерек регистріне (МРг) жазылатын дерек беріледі. Жады контроллері кристалл таңдау ($\overline{CS}$), $\overline{RAS}$, С$\overline{AS}$, R/$\overline{W}$ және $\overline{OE}$ сигналдарын тиісті уақытта шығарады. Қатар мекен-жайы RAS (Row Address Strobe) сигналымен қатар регистріне (РгХ) жазылады, ал бағана мекен-жайы келесі кезекте CAS (Column Address Strobe) сигналымен бағана регистріне (РгY) қабылданады. Шағын сұлба CS (Chip Select) сигналы арқылы іске қосылады. Егер жүйе бірнеше шағын сұлбалардан тұрса, онда CS сигналы арқылы олардың ішінен керектісін іске қоса аламыз. S/$\overline{W}$-сигналы арқылы орындалатын операцияның (оқу немесе жазу) түрін анықтаймыз. OE (Output Enable-шығыс сигналдарын сыртқа беруге рұқсат) сигналы арқылы дерек регистрінің шығысына қосылған үш күйде болатын сұлба арқылы дерек дерекшинасы (МҚ) арқылы процессорға беріледі.

Жазылатын дерек дерекшинасы арқылы дерек регистріне беріледі. Қатар регистрімен (РгХ) және қатар дешифраторы (DCX) арқылы жазылатын ұяшық таңдалады. Дерек ұяшықтың қай позициясына жазылу керектігін бағана мекен-жайында (РгY) жазылған k разрядты кодқа сәйкес коммутациялау құрылғысы анықтап жазады.



7.7-сурет. Жады шағын сұлбасының құрылымы

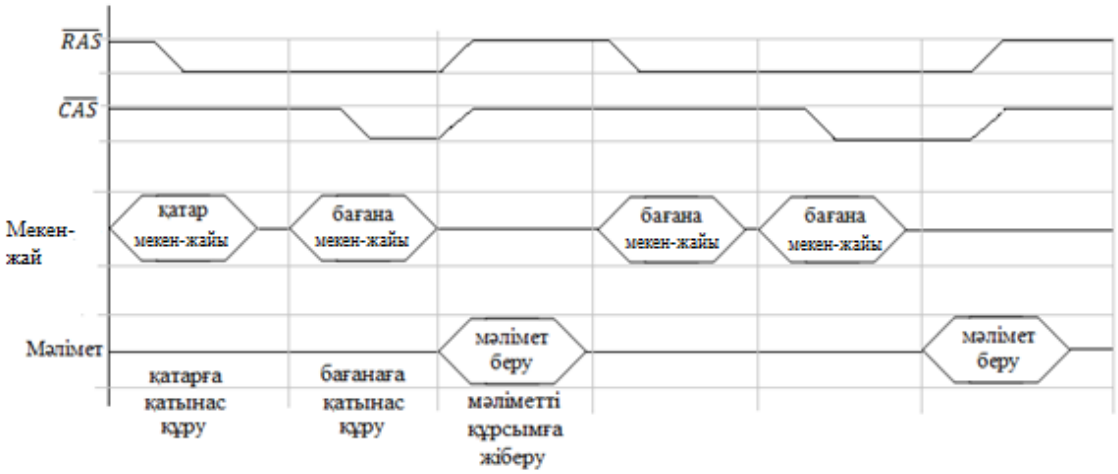
Деректі оқу кезінде ұяшық қатары қатар мекен-жай кодымен анықталса, ал оның ішіндегі шығарылатын биттер бағана мекен-жайының кодымен анықталады. Оқылған деректер КҚ арқылы дерек регистріне беріліп, одан ОЕ сигналы арқылы дерекшинасы арқылы процессорға жеткізіледі. Жоғарыда аталған басқару сигналдарының көпшілігінің белсенділігі сигналдардың төменгі деңгейінде болады, сондықтан 7.7–суретте олар терістелген түрінде көрсетілген.

7.7-суретте көрсетілген ЖҚ бір операцияны орындау үшін кем дегенде бес такт керек:

1. Орындалатын операцияның түрін көрсету және қатар мекен-жайын беру.
2. RAS сигналын қалыптастыру.
3. Бағана мекен-жайын беру.
4. CAS сигналын қалыптастыру.
5. RAS және CAS сигналдарын белсенді болмайтын қалыпқа келтіру.

Жоғарыда келтірілген бес тактың ішінде динамикалық жедел жадыда орындалатын регенерациялау әрекеттеріне керек такт сигналдары қаралмаған.

7.8-суретте қатар және бағана мекен-жайлары мультиплекстенетін интегралдық шағын сұлбаларда деректерді оқу үрдісінің уақыт диаграммасы келтірілген.



7.8-сурет. Оқу үрдісінің уақыт диаграммасы

Жадыдан оқу уақыт диаграммасы

Жады контроллері R/$\overline{W}$ сигналын оқу деңгейіне сәйкес қалыптастырған соң мекен-жай интегралды шағын сұлба (ИШС) түйіспелеріне қатар мекен-жайымен бірге RAS сигналын береміз. RAS сигналының артқы фронтымен мекен-жай қатар регистріне РгХ жазылып, ол DCX дешифраторына дешифрацияланады. Таңдалған қатар коммутациялау құрылғысына беріледі. Келесі тактта ИШС кірісіне бағана мекен-жайы CAS сигналымен беріліп, РгY регистріне жазылады. Бағана регистрінен k разрядты код коммутациялау сұлбасына беріледі. Коммутациялау құрылғысының құрамына кіретін m мультиплексорлар арқылы талданып алынған қатардың әр тобынан бір биттен оқылып, m күшейткіш арқылы күшейтіліп, дерек регистріне (МРг) жазылады. Дерек регистрі ОЕ сигналымен дерекшинасы (МҚ) арқылы процессорға беріледі.

ЖҚ жылдамдығын өсірудің екі жолы бар. Біріншісі - ИШС ядросын жетілдіру. Ол ЖЭ жетілдіру технологиясына байланысты. Екінші жолы - матрицаның таңдалынған қатарынан деректерді тиімді ретпен оқып, оларды дерекшинасына тиімді жолмен жеткізу. Осыған байланысты ЖҚ жылдамдығын арттырудың бірнеше іргелі тәсілдері бар:

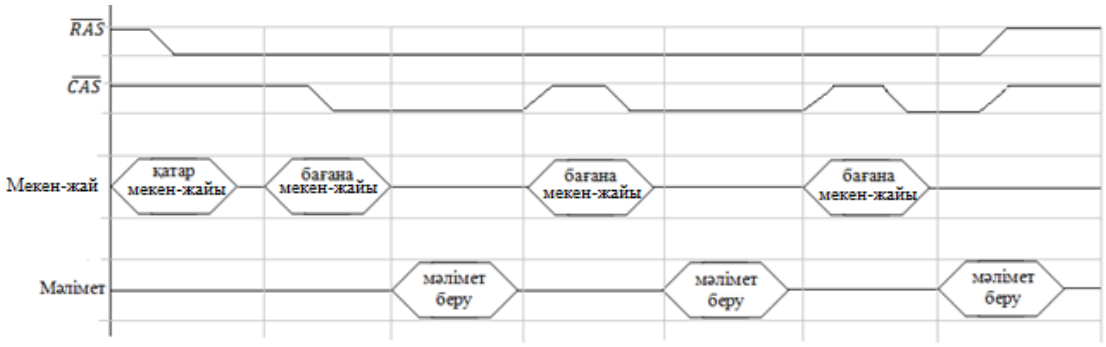
- Тізбекті;
- Регистрлік;
- Беттермен жылдам алмасу;

- Дестемен алмасу;
- Екі еселенген жылдамдық.

Тізбекті режим. Тізбекті режимде (Flow through Mode) мекен-жай мен басқару сигналдары микросұлба кірістеріне синхроимпульстан бұрын беріледі. Синхроимпульс берілген моментте оның алдыңғы фронтымен барлық деректер ЖҚ ішкі регистрлерінде сақталып, оқу циклы орындала бастайды. Белгілі бір уақыт өткеннен кейін осы оқу циклының ішінде деректер сыртқы шинаға беріледі. Деректің сыртқы шинаға берілу уақыты шағын сұлба электр тізбегінің жылдамдығымен анықталады.

Регистрлік режим. Реистрлік режим (Register to latch) тізбекті режимнен айырмашылығы – шағын сұлбаның шығысында аралық регистрдің болуы. Оқылған дерек осы аралық регистрінде синхросигнал келгенше сақталып, оның теріс фронтымен шығыс шинасына беріледі.

Беттермен жылдам алмасу режимі. 2DM жүйесінде құрылған ЖҚ әр қатар (матрица беті) қатар мекен-жайымен, қатар дешифраторымен RAS сигналының төменгі деңгейімен таңдалады. Әр матрица бетінде $m2^k$ ЖЭ бар. Мұнда әр m ұяшықтар көршілес мекен-жайларда орналасқан ұяшықтар қатары ішіндегі k мекен-жаймен анықталады. Кезекті ұяшыққа қатынас құру үшін шағын сұлба кірісіне жаңа ұяшық мекен-жайын және CAS сигналын берсе болғаны. Бірінші ұяшыққа қатынас құру стандартты түрде жүргізіледі – кезекпен қатар және бағана мекен-жайлары беріледі. Мұнда қатынас құру уақытын өзгерте алмаймыз. Келесі ұяшықтарға қатынас құрудан бастап, қатынас құру уақыты азая бастайды. Өйткені RAS сигналы барлық керекті ұяшықтардан оқылып болғанша өзгермейді. Мұндай режим *беттермен жылдам алмасу режимі* деп аталады (Page Mode). Динамикалық жедел жадыларда (DRAM) осы режимнің модификацияланған түрі кеңінен қолданылады. Оны беттермен жылдам алмасу режимі (FPM-Fast Page Mode) деп атайды. Оқу операциясының уақыт диаграммасы 7.9–суретте келтірілген.

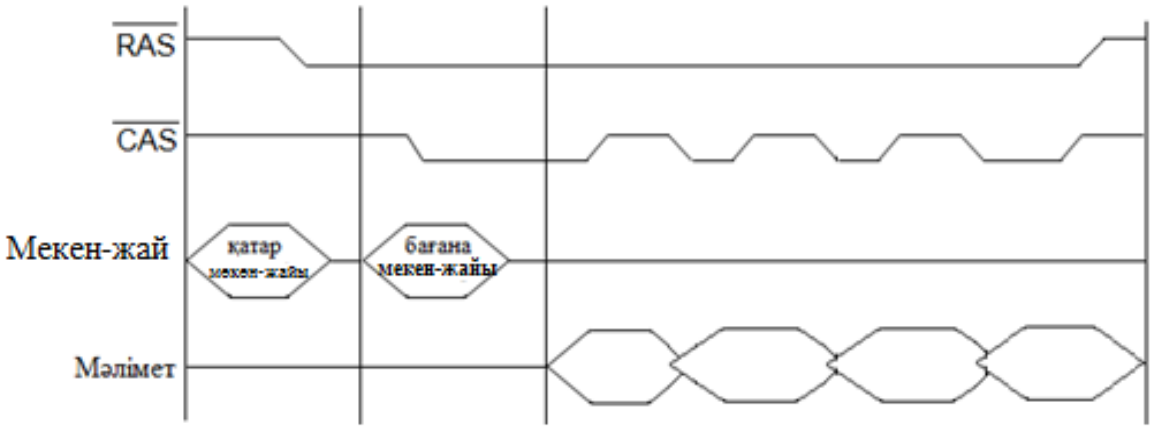


7.9-сурет. Беттермен жылдам алмасу режимінде оқу операциясының уақыт диаграммасы

Қаралған режимде оқу операциясының уақыты екі тактқа қысқарады, бірақ нақты шағын сұлбаның бір қатарында сақталған дерек блогын оқығанда байқауға болады. Егер бағдарлама ЖҚ әртүрлі беттерінен оқылса, онда қаралған тәсілдің артықшылығы жоққа шығады.

Беттермен алмасу режимін не оның модификациясы іске асыратын шағын сұлбаны х-у-у-у формуласымен сипаттауға болады. Бірінші сан х-бірінші ұяшыққа қатынас құруға кететін жүйелік шинаның такт саны, ал у-келесі ұяшықтарға қатынас құруға жұмасалатын такт саны. Мысалы 6-3-3-3 дегеніміз бірінші ұяшықтан деректі оқу үшін жүйелік шина 6 такт периодын жұмсайды, ал қалған сөздерді оқу үшін әр сөз үшін такт сигналдары керек.

Дестелік режим. Дестелік режимде (Burst Mode) ЖҚ белгілі бір мекен-жайға сұраным болғанда, сұранымда көрсетілген мекен-жайдан басқа келесі бірнеше мекен-жайлардан оқылатын дерек дестесі оқылады. Қазіргі есептеу машиналарында жады ұяшығының разряд саны байтқа тең. Егер шина төрт байттан тұрса, онда жадыға жасалынған бір сұраным үшін көрші төрт ұяшықтарға (дестеге) тізбекпен қатынас құру керек. Мұндай тәртіпте жұмыс жасайтын ЖҚ *дестелі режимді ЖҚ* деп аталады. Мұндай ЖҚ бағана мекен-жайында тек бірінші ұяшықтың мекен-жайы көрсетіледі, ал қалған ұяшықтардың мекен-жайлары шағын сұлбаның ішінде қалыптастырылады. Бұл төрт операцияның орнына шағын сұлбада орындалатын бір операцияны (бірінші ұяшық мекен-жайын беру) орындауға алып келеді. Дестелі режимде оқу операциясының уақыт диаграммасы 7.10–суретте келтірілген.

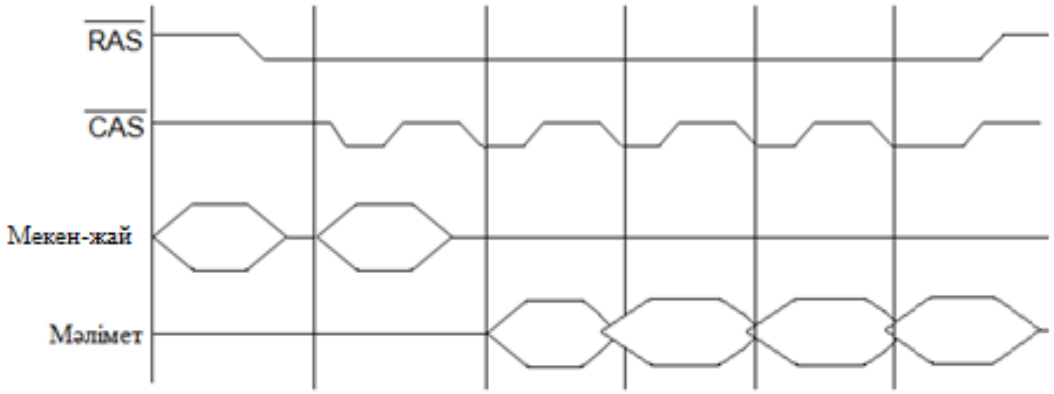


7.10-сурет. Дестелі режимде оқу операциясының уақыт диаграммасы

Конвейерлік режим. Конвейерлік режимде (Pipeline mode) үдеріс екі кезеңге бөлінеді. Бұрынғы циклда оқылған дерек сыртқы шинаға беріліп жатқан уақытта келесі оқу операциясының бірінші кезеңі орындала бастайды. Осындай жолмен екі оқу циклының кезеңдері уақыт бойынша қабат жеткізу

орындалады. Деректі сыртқы шинаға жеткізу сұлбасы күрделінеді. Ол оқу уақытын бір тактқа ұзартуға алып келеді. Сондықтан дерек ЖҚ шығысына келесі тактта беріледі. Бірақ мұндай кешігу тек бірінші сөзді оқуда ғана орын алады. Барлық қалған деректер бірінен кейін бірі сыртқа оқу сұранымынан бір тактқа кешігіп шығады.

7.11-суретте оқу операциясының конвейерленген дестелік режиміндегі жұмыс жасау уақытының диаграммасы келтірілген.



7.11-сурет. Оқу операциясының конвейерленген дестелік режиміндегі уақыт диаграммасы

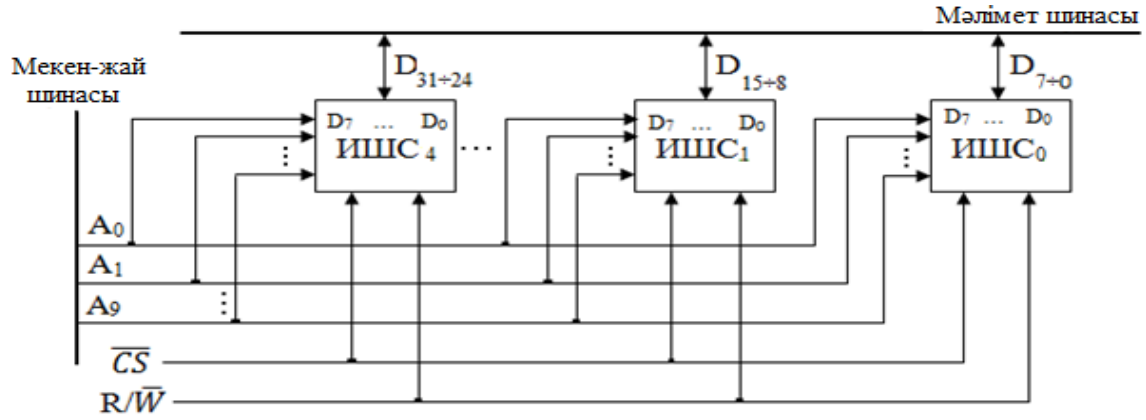
Екі еселенген жылдамдық режимі. Деректерді екі еселенген жылдамдықпен беру режимі DDR (Double Data Rate) синхронды ЖҚ-да кеңінен қолданылады. Мұндай тәсілде дерек синхросигналдың екі (алдыңғы және артқы) фронттарында беріледі, яғни синхросигнализациялау импульсінің бір периодында деректер екі рет беріледі. Сондықтан деректі жіберу мүмкіндігі екі есе өседі.

Жоғарыда қаралған ЖҚ жылдамдығын өсіру тәсілдерінен басқа да тәсілдер кеңінен қолданылады. Мысал ретінде шағын сұлба құрамына қосалқы КЭШ-жадыны енгізу жолын атауға болады және бір уақытта деректерді шинамен алмасу және ЖЭ матрицасына қатынас құру үшін тәуелсіз деректер трактарын іске қосу т.б. тәсілдерді атауға болады.

7.6. Негізгі жадыны блоктармен ұйымдастыру

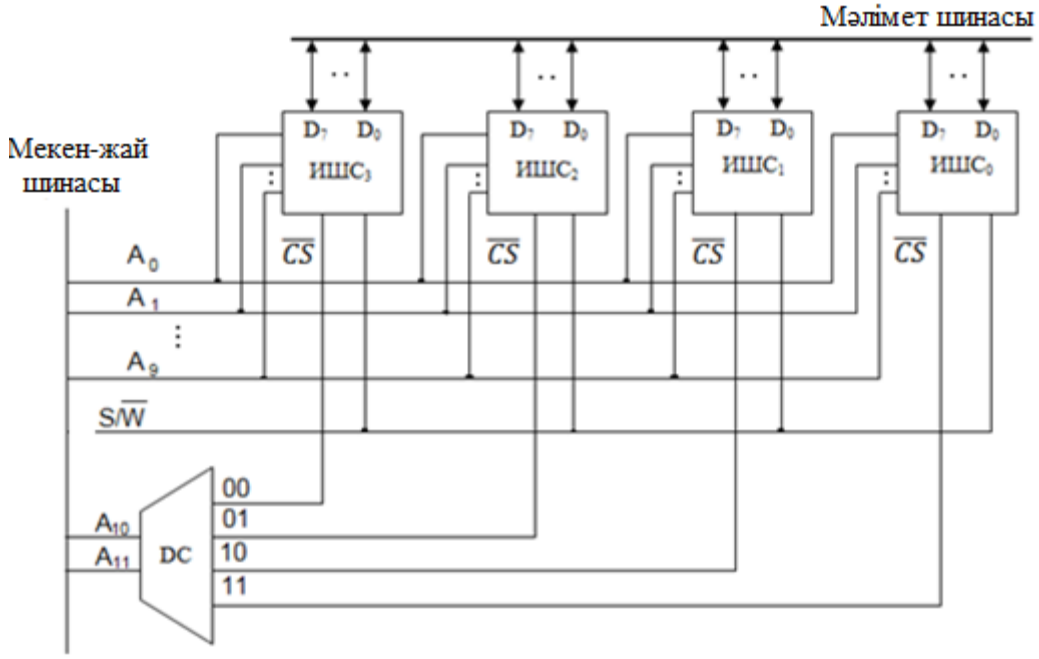
Қазіргі интегралды шағын сұлбалардың (ИШС) ақпараттық сыйымдылығы және разрядтылығы есептеу машинасына керекті сыйымдылығы мен разрядтылығы жағынан әлдеқайда кіші. Сондықтан бір ИШС арқылы керекті машина жадын құра алмаймыз. Ол үшін бірнеше ИШС біріктіру қажет. Біріктіру арқылы ЖҚ разряд не ұяшық санын немесе ұяшық саны мен

разрядтылығын бірдей өсіре аламыз. 1024 ұяшығы бар 8 разрядты шағын сұлбалар арқылы 1024 ұяшықты 32 разрядты (7.12– суретте көрсетілген) 4 ИШС-ның мекен-жай кірістерін біріктіреміз.



7.12-сурет. Жады разрядын өсіру сұлбасы.

ЖҚ ұяшық санын өсіру үшін ИШС санын өсірумен қатар олардың мекен-жай шинасының санын өсіру керек. Мысалы бізде 1024 ұяшық саны, сегіз разрядты ИШС бар делік. Осындай жады шағын сұлбалардан ұяшық саны 4096 болатын 8 разрядты ЖҚ құру керек. Мекен-жай арқылы өсірілген мұндай ЖҚ 7.13–суретте келтірілген. Суреттен көрініп тұрғандай ЖҚ құру үшін 4 ИШС және 2 кірісті 4 шығысты дешифратор керек.

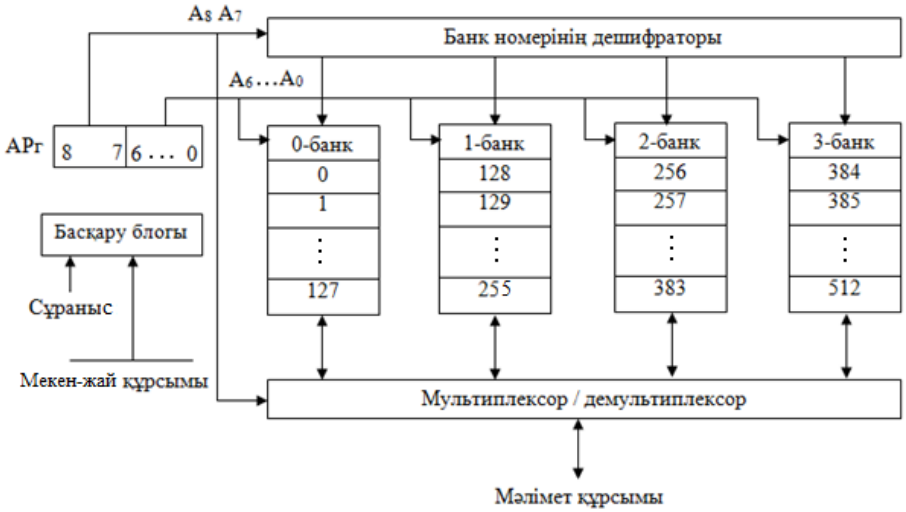


7.13-сурет. Жады ұяшық санын өсіру.

Бірнеше ИШС біріктіріліп алынған ЖҚ жады модулі деп аталады. Егер ИШС разрядтылығы ЖҚ құруға жеткілікті болса, оны да модуль деп атауға болады. Бір не бірнеше модульдерді қосу арқылы жады банкін құруға болады.

Керекті сыйымдылығы бар ЖҚ (негізгі жады) бірнеше банктерді біріктіру арқылы алынады. Олар блок түрінде құрылады. Блок түрінде құрылған негізгі жадыға мысал 7.14–суретте келтірілген. Мұнда әр банк 128 ұяшықтардан тұрады.

Мекен-жай регистрі екі бөлікке бөлінген. A_8 A_7 бөлігі арқылы банк нөмірі таңдалынады, ал A_8...A_9 мекен-жай бөлігі арқылы әр банктің ішінен бір ұяшық іріктеліп дерекшинасына беріледі. Әр банк 128 ұяшықтан тұрады. Мұндай ЖЖ *бір жады құрылғысы* деп қарауға болады. Ал жылдамдығы жеке банк жылдамдығымен анықталады.



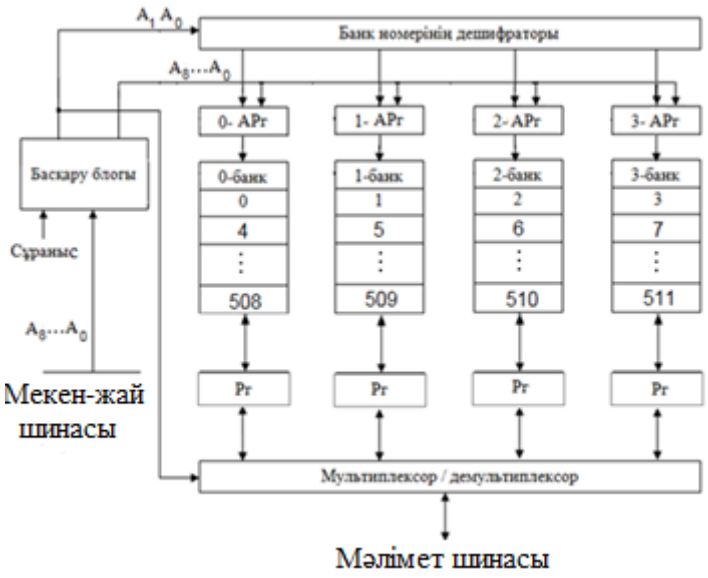
7.14-сурет. Блок түрінде құрылған негізгі жады

Бірнеше блоктардан құралған жедел жадыда бірнеше банктерге қатар қатынас құру арқылы құрылғының жылдамдығын арттыруға мүмкіндік туады. Ол үшін жадыны қатпарландыру тәсілі қолданылады. Қатпарландыру негізінде мекен-жайларды кезектестіру (address inter leaving) әдісі жатыр. Ол үшін жады банктерінде жады мекен-жайлары кезекпен орналастырылады (7.15-сурет).

Біздің мысалда банктердің бірі екі төменгі разрядтар (A_1 A_0) арқылы таңдалса, ал банктер ішінен тиісті ұяшық A_8...A_2 мекен-жайлары арқылы таңдаймыз.

Әр тактта мекен-жай шинасында тек бір ұяшық мекен-жайы болуы керек, сондықтан бірнеше банктерге қатар қатынас құру мүмкін емес.

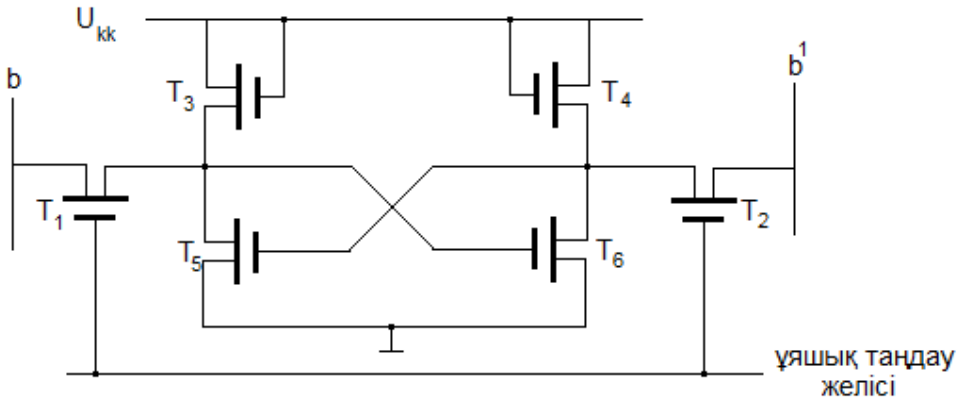
Қатар қатынас құруды іске асыру үшін ұяшық мекен-жайларын банктердің меншікті мекен-жайларында (0-МЖРг÷3-МЖРг) сақтап, әр банкке қатынас құру операциясын бір тактқа жылжыту керек. Мұндай жағдайда әр банктегі ұяшықтарға қатынас құру операцияларын қатар орындауға мүмкіндік береді.



7.15-сурет. Мекен-жайлары кезектестірілген блокты жады

7.7. Жады элементтері

Статикалық ЖҚ (SRAM) статикалық ЖЭ негізінде құрылады. Статикалық ЖЭ өз күйін оған берілген қорек берілгенше сақтап тұрады. 7.16–суретте мысал ретінде КМОП-транзисторлары негізінде құрылған ЖЭ (триггер сұлбасы) келтірілген.



7.16-сурет.КМОП-транзисторлары негізінде құрылған ЖЭ (триггер сұлбасы)

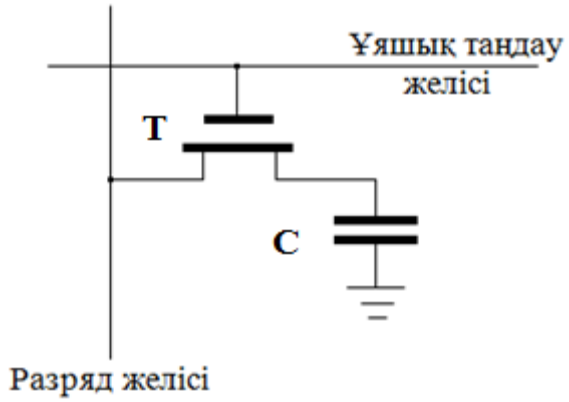
T_3,T_5 транзисторлары арқылы бірінші инвертор құрылады, ал екінші инверторды T_4,T_6 транзисторлары құрады. Екі инверторларды бір бірімен суретте көрсеткендей қоса отырып триггер сұлбасын аламыз. Жады ұяшығының күйін оқу үшін ұяшық таңдау желісін (мекен-жай желісіне) оқу сигналын береміз. Оның нәтижесінде T_1 және T_2 кілттері жабылады (олар арқылы ток жібермейді). Егерде ұяшық 1 күйінде болса, онда b желісінде жоғары деңгейлі сигнал пайда болады, ал b^1 желісінде сигнал деңгейі төмен болады. Егер ұяшықта «0» сақталған болса, онда сигналдар мәні кері болады.

Ұяшыққа дерек жазу үшін тиісті код b және b^1 желілеріне беріліп, ұяшық таңдау желісіне жазу сигналын берсе болғаны.

Статикалық ЖҚ жылдамдығы өте жоғары, осыған сәйкес оның құны да жоғары, өйткені жады ұяшығы бірнеше транзисторлардан құрылған. Бағасы арзан ЖҚ-лары бір транзисторлы ЖЭ арқылы жасалынады. Мұндай ЖЭ деректі өте ұзақ сақтай алмайды, сондықтан олардың күйін белгілі уақыттан кейін қайта қалпына келтіріп отыру керек. Мұндай ЖЭ құрылған ЖҚ динамикалық (Dynamic RAM, DRAM) деп атайды.

Динамикалық ЖЭ дерек заряд түрінде конденсаторда сақталады және бұл заряд бірнеше ондаған миллисекундқа ғана созылады. Оны ұзағырақ сақтау үшін белгілі бір уақытта конденсаторды қайта зарядтап отыру керек.

7.17-суретте динамикалық жады ұяшығына мысал келтірілген. Ұяшық С конденсаторы мен Т транзистордан тұрады.



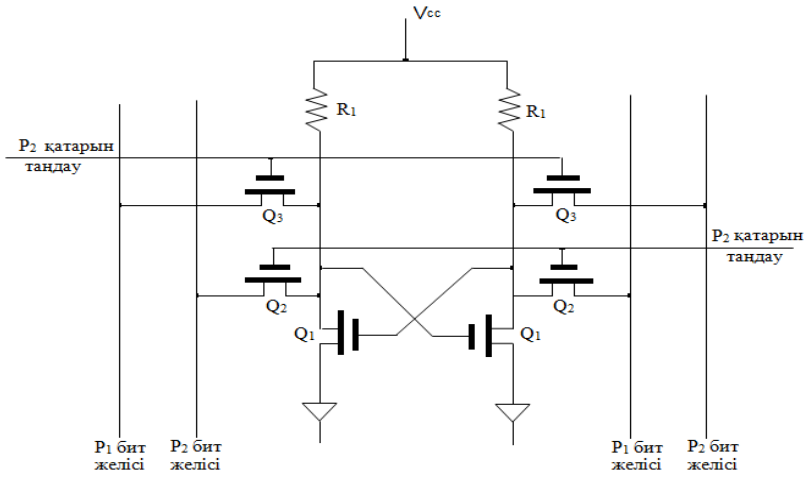
7.17-сурет. Динамикалық жадының бір транзисторлық ұяшығы

Дерек жазу үшін ұяшыққа Т транзисторын қосамыз және разряд желісіне тиісті дерек кодын береміз. Дерек кодасының деңгейіне байланысты конденсаторда белгілі заряд қалыптасады. Транзистор қосылмаған жағдайда конденсатордағы заряд бірте-бірте бәсеңделе бастайды. Зарядтың бәсеңдеу уақыты конденсатордың меншікті кедергісіне байланысты болады, оның үстіне сөндірілген транзистор аз мөлшерде болса да ток өткізеді. Егер де конденсатордағы заряд белгілі бір шамадан азаймай тұрғанда оны қайта зарядтасақ, онда оқылған деректерде ешбір қате болмайды. ЖЭ периодты түрде қайта зарядтау үрдісі *регенерациялау* деп аталады, ол әрбір 2-8 мс сайын қайталанып отырады. Динамикалық жадының матрицада орналасу тығыздығы өте жоғары. Сондықтан оның құны статикалық ЖҚ қарағанда 8-16 есе аз.

Жоғарыда қаралған ЖЭ бір портты ЖҚ элементтері болып табылады.

Қазіргі көп процессорлы және көп машиналы есептеу жүйелерінде көп портты ЖЖ кеңінен қолданылуда. Мұндай жүйелерде көп портты ЖЖ ортақ жады ретінде қолданылады.

Көп портты (n-портты) жедел жадында n мекен-жайлары, дерек және басқару шиналары болады. Олар арқылы жедел жадыға бір-бірімен қатар бірнеше қатынас құруға болады.



7.18-сурет. Екі портты статикалық ЖЖ жады элементі

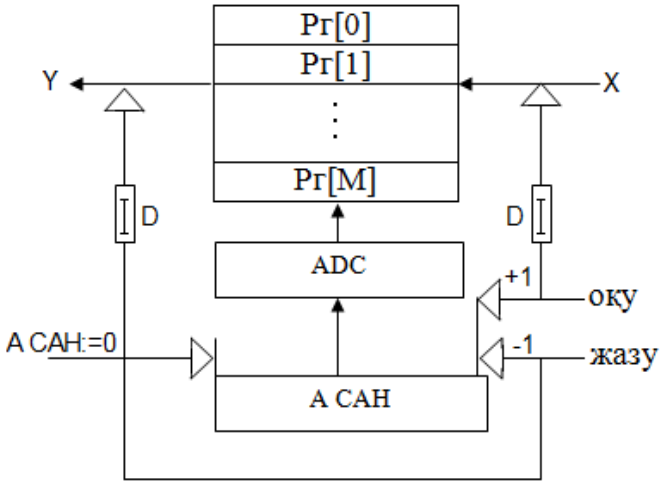
7.18 -суретте екі портты жады құрылғысының жады элементі келтірілген. ЖЭ алты транзисторлардан тұрады. Мұнда Q_3 және Q_2 транзисторлары екі бағытта қатынас құруға мүмкіндік береді. Екі мекен-жайлары, дерек және басқару шиналары арқылы ЖЭ ортақ массивіне екі бағытқа қатынас құруды іске асырады.

7.8. Стек жады

Стек жады «соңынан жазылған – бірінші оқылады» (LIFO-Last In First Out) режимінде жұмыс жасайды. Мұндай жады құрылғылары үзу жүйесінде үзу бағдарламасын орындау кезінде процессор регистрлерінің күйін сақтау үшін кеңінен қолданылады.

Стек жады разряд тізбектері арқылы бір бірімен байланысқан ұяшықтардан тұрады. Стек жады процессормен оның тек жоғарғы ұяшығы – стек шыңы арқылы алмасады. Жаңа дерек жазылғанда бұрынғы жазылған сөздер бір ұяшыққа төменге жылжытылады, ал жаңа сөз стек шыңына жазылады. Дерек тек стек шыңынан оқылады. Стек шыңынан оқылып болған соң, оның орнына стек шыңының төменгі ұяшығындағы дерек жазылады.

Стек жадының жұмыс жасау принципін 7.19–суретте көрсетілген сұлбамен түсіндіруге болады.



7.19-сурет. Стек жады

Стек Рг[0]÷Рг[M] регистрларынан, мекен-жай декодерінен (ADC) және мекен-жай санауышынан (А САН) тұрады. Мекен-жай санауышы реверсті режимде жұмыс жасайды және стек шыңы болып табылады. Жұмыс жасар алдында А САН:=0 микро командасымен А САН тазалаймыз, яғни ADC Рг[0] регистріне көрсетіп тұрады. ЖАЗУ сигналы арқылы дерек кіріс Х шинасы арқылы Рг[0]-ге жазылады. Дерек Рг[0] регистріне жазылып болғанша ЖАЗУ сигналы D элементінде кідіреді. Кідірістен кейін А САН бірге өсіріледі. Осылай бірнеше сөздерді жазатын болсақ, олар Рг[1], Рг[2]…регистрлеріне орналасады.

Стек жадынан деректі оқу ОҚУ сигналы арқылы жүзеге асырылады. ОҚУ сигналын берген кезде А САН күйі бірге кемиді, сол кезде Y шығыс сигналынан стекке жазылған соңғы сөз беріледі. Егер стекке сөз А1, А2, А3 ретімен жазылса, онда олар кері А3, А2, А1 тәртіппен оқылады.

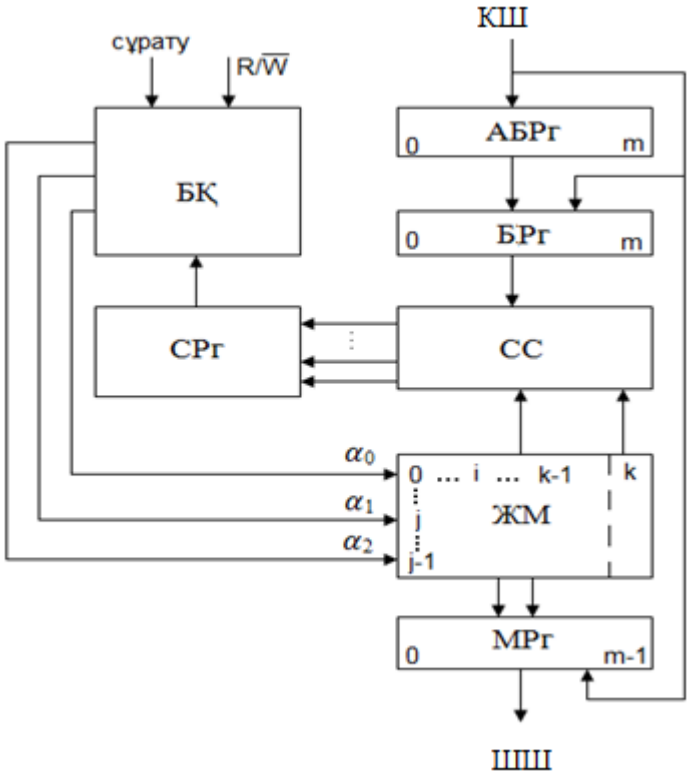
7.9. Ассоциативтік жады

Ассоциативтік жадында деректер мекен-жайлар арқылы емес олардың ассоциативтік белгілері арқылы (немесе осы белгінің жеке разрядтарымен параллель) ізделеді. Іздеу операциясы жады массивінің барлық ұяшықтарымен бір уақытта орындалады.

Көптеген жағдайда ассоциативтік іздеу деректерді өңдеуді жеңілдетеді, өйткені деректі оқу үстінде онымен қатар логикалық операциялар орындауға мүмкіндік болады.

Ассоциативтік жады құрылымы 7.20–суретте көрсетілген. Жады массиві (ЖМ) N (k+1) ұяшықтардан тұрады. Оның k-разряды j ұяшығының бос еместігін көрсетеді. Ақпараттың кіріс шинасы КШ арқылы ассоциативтік белгі регистріне (АБРг) k-разрядты ассоциативтік белгі қабылданады, ал бүркеме регистріне (БРг) оның коды беріледі. Ассоциативтік іздеу тек АБРг

разрядтарына сәйкес келетін 1 мәні жазылған БР разрядтармен (АБР$_г$ бүркемеленбеген разрядтарымен) жүргізіледі. Іздеу барысында СС салыстыру сұлбасы бір уақытта ЖМ барлық ұяшықтардың бүркемеленбеген биттерін АБРг бірінші битімен салыстырады.



7.20-сурет. Ассоциативтік жады құрылымы

Егер салыстыру нәтижесінде биттер сәйкес келмесе, онда сәйкестіктер регистрінің СР$_г$ бірінші разрядына «0» жазамыз (сәйкес келсе, «1» жазамыз). Осылай басқа қалқаланбаған биттер де салыстырылады. Сонымен СРг регистріне j-разряды мына формуламен анықталады:

$$СРг(j)=\bigwedge_{i=0}^{i=m-1}\{\overline{АБРг[i]\oplus ЖМ[j,i]}\vee\overline{БРг[i]}\}\ (0\le j\le N-1).$$

Мұнда АБРг[i], БРг[i], ЖМ[j,i]–АБРг, БРг регистрлерінің i-разряды және ЖМ-ің j-ұяшығы.

Сәйкестік регистрінің мазмұнына байланысты БҚ басқару құрылғысы

α_0, α_1, α_2сигналдарын шығарады. α_0 – ЖМ ассоциативтік белгілеріне сәйкес келетін дерек жоқ дегенді білдіреді. α_1=1 болса, онда ЖМ бір сөз табылғанын көрсетеді – ЖМ белгілері сәйкес келетін сөз оқылып, ол МРг дерек регистріне жазылады. Егер α_2=1 болса, онда берілген ассоциативтік белгіге екі

немесе одан да көп сөздер белгісі сәйкес келгені. Мұндай жағдайда ЖМ дерек регистріне сөз ең кіші нөмірі бар ұяшықтан жазылады.

Ассоциативтік жадыға дерек оның бос ұяшығына жазылады. Бос ұяшықтар ЖМ n-разрядын таңдау арқылы табылады.

7.10. КЭШ-жады

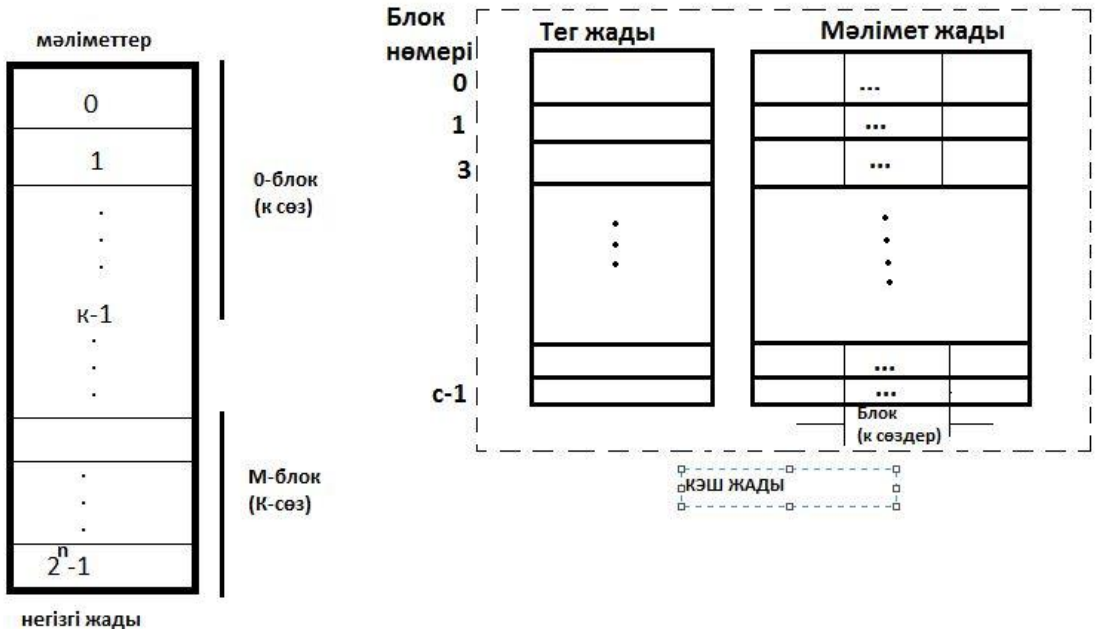
Есептеу жүйесінің өнімділігі негізінен оның құрамына кіретін процессор мен жедел жады жылдамдықтарының ғана емес олардың жылдамдық бойынша сәйкестіктерімен анықталады. Егер сыймдылығы жеткілікті жедел жадыны тек жылдамдығы жоғары статикалық жады шағынсұлбалары арқылы құратын болсақ, онда оның құны өте жоғары болар еді. Сондықтан есептеу машиналарында негізгі жады динамикалық шағынсұлбалар арқылы құрылады. Мұндай жедел жады бағасы төмен және ақпаратық сыйымдылығы жоғары. Оның негізгі кемістігі жылдамдығының төмендігінде. Егер мұндай динамикалық жадыны процессорға тікелей қоссақ, онда динамикалық жады жылдамдығымен анықталар еді. Мұндай жағдайда жедел жады процессорға аралық жады арқылы қосылады. Аралық жады сыйымдылығы үлкен емес, бірақ жылдамдығы процессор құрылғыларының жылдамдықтарына сәйкес келеді. Мұндай аралық жады ұяшықтарында процессордың ЖЖ қатынас құру кезінде анықтайтын сөздер тобының көшірмесі жазылады. Сөздер тобының ішінде бағдарламаның көптеген командалар мекен-жайлары және әртүрлі деректер болуы мүмкін. Егер процессорға керекті кезекті деректер тек аралық жадыдан оқылатын болса, онда ЖЖ ұяшықтарына қатынас құрудың керегі жоқ. Жоғарыда айтқанда аралық жадыға қатынас құру жылдамдығы жоғары болғандықтан, дерекке қатынас құру уақытын қысқартамыз.

Жоғарыда айтылған аралық жадыға КЭШ жады (ағылшын тілінен cache құпия, жасырын) жатады. КЭШ жадыға бағдарламалық жолмен қатынас құру мүмкін емес. Сондықтан бағдарламаушы ол туралы білмеуі де мүмкін.

Орталық процессор ЖЖ сөз оқу кезінде сөз көшірмесін ең алдымен КЭШ жадыдан іздейді. Егер ондай көшірме КЭШ жадыда жазылған болса, онда ЖЖ қатынас құрмайды, КЭШ жадыдан оқыған сөз орталық процессорға беріледі. Мұндай жағдайды *сәтті қатынас құру* немесе *КЭШ дәл түсуі* (cach hit) деп атайды. Егер КЭШ жадыда іздеген сөз жоқ болса, яғни қатынас құру сәтсіз болса, онда *КЭШ мүлт кетуі* (cache mis) орын алады. Мұндай жағдайда негізгі жадыдан құрамында ізделген сөз бар блоктың көшірмесі КЭШ жадыға жазылады.

7.21-суретте жедел жады мен КЭШ жадыдан тұратын жады жүйесінің құрамы көрсетілген. Сыйымдылығы 2^n сөзі бар ЖЖ әрқайсысы К сөздерден

тұратын М блоктардан тұрады. ЖЖ КЭШ жадымен К сөзден тұратын блоктармен алмасады. КЭШ жады да әрқайсысы К сөзден тұратын С блоктардан тұрады және С<<М. ЖЖ кез келген ұяшығында қатынас құрғанда осы ұяшық қай блоктың құрамына кірсе, сол блок көшірмесі КЭШ жадыға жазылады.



7.21-сурет . Негізгі жады және КЭШ жады жүйесінің құрылымы

Блокқа келесі қатынас құрғанда тек блоктың көшірмесіне қатынас құрамыз. КЭШ жады сыйымдылығы ЖЖ сыйымдылығынан аз, сондықтан КЭШ жадыға әр түрлі уақытта ЖЖ әртүрлі блоктарының көшірмелері жазылады. Әрбір көшірмелерге белгілер меншіктеледі. Мұндай белгілерді *ТЕГ* деп атайды. ТЕГ арқылы КЭШ жадыда қазіргі сақталып тұрған блок, ЖЖ қай блогының көшірмесіне жататындығын білуге болады.

Сонымен КЭШ жады екі түрлі есте сақтау құрылғыларынан тұрады. Біріншісінде ЖЖ деректерінің көшірмелері сақталады, екінші сақтау құрылғысында осы көшірмелердің ТЕГ-тері (ТЕГ жады) сақталады. Жадыда деректің әрбір блогына тег жадының бір ұяшығы сәйкес келеді.

Негізгі жадыны КЭШ жадында кескіндеу тәсілдері

Негізгі жадының КЭШ жадымен блоктар арқылы алмасу тәсілдерін түсіну үшін қарапайым мысалдар қарайық.

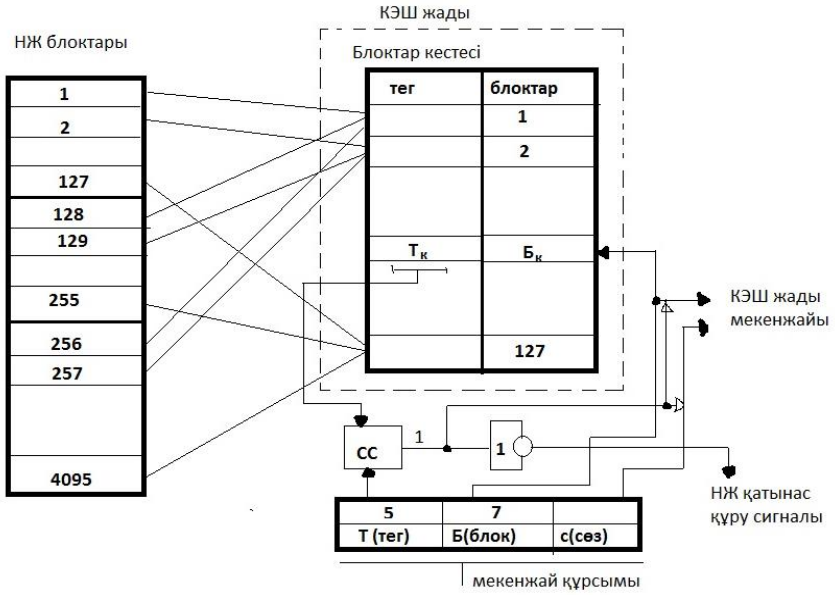
Бізде әрбір блогы 16 сөзден тұратын 128 блоктары бар КЭШ жады бар болсын делік. Мұндай жадыға қатынас құру үшін мекенжай разрядтылығы 11 болу керек (16сөз*128блок=2048сөз=2Ксөз=2^{11}сөз). Негізгі жады сыйымдылығы КЭШ жадыдағыдай 16 сөздерден тұратын 24К сөздер болсын.

24Ксөздер=2^{16}сөздер, яғни НЖ мекенжайының разрядтылығы (разряд саны)-16. Мұндай НЖ әрқайсысы 16 сөздерден құралған. 2К=2048 блоктардан тұратынына көз жеткізу қиын емес.

НЖ КЭШ жадыда кексіндеудің үш тәсілі кеңінен белгілі. Олар: тікелей, толығымен ассоциациялау және секторлы ассоциациялау арқылы кескіндеу.

Тура кескіндеу

Жоғырыда келтірілген мысалға сәйкес тура кескіндеу тәсілінде негізгі жадының j-блогы КЭШ жадының j-блогына 128 модульь бойынша кескінделеді. Осыған байланысты, егер НЖ блоктарының бірінен деректерді жүктегенде 0, 128, 256 т.б мекен-жайларынан басталатын сөздер КЭШ жадының 0-блогына, ал 1, 129, 257 т.б мекен-жайларынан басталатын сөздер КЭШ жадының 1-блогына т.с.с жүктеледі. Сонымен КЭШ жады блоктарының мекенжайы негізгі жады мекенжайымен анықталады. Негізгі жадының мекенжай кодын үш бөлікке бөліп қарауға болады. (7.22-сурет) Төменгі төрт разряд (сөз) арқылы әр блоктағы 16 сөздердің бірін анықтаймыз. КЭШ жадыға жазылатын жаңа блок орны 7 разрядты (Б блок) өріс арқылы анықталады. Жоғарғы 5 разрядтар (Т тег) жадыдағы блоктардың мекенжайын көрсетеді. Кезекті блокты КЭШ жадыға жазарда блоктармен қатар олардың белгісі тег арнайы (Т тег) өрісіне жазылады.



7.22-сурет. НЖ деректерін тура кескіндейтін КЭШ жадыда орындалатын мекенжайын қалыптастыру сұлбасы.

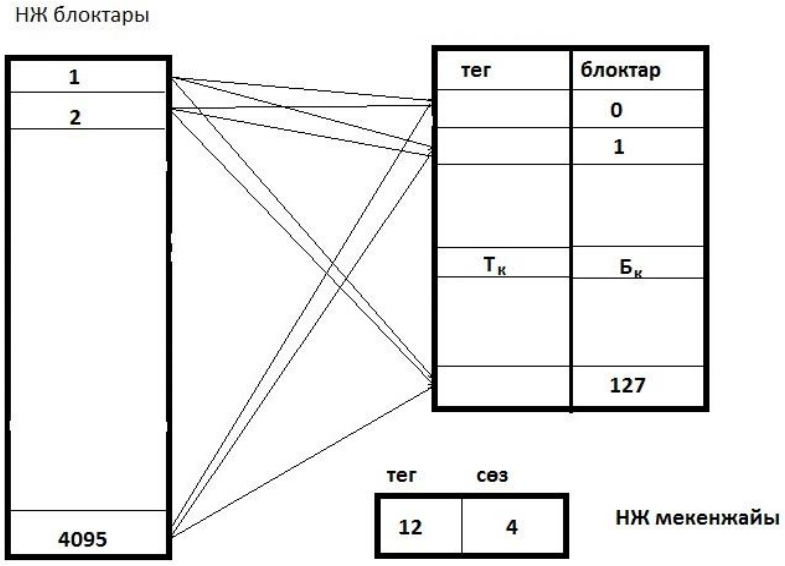
Бағдарламаны орындау үстінде процессор сөз мекенжайларын мекенжай шинасына генерациялайды. Олардың әрқайсысындағы 7 разрядты өріс (Б блок) КЭШ жадындағы блок нөмірін көрсетеді. Блок нөмірі (блок) арқылы блок кестесінен блок қатарын табамыз. Осы қатар ТЕГ-імен (Тк) процессордан

келген мекенжай кодының жоғарғы разрядымен (Т тег) салыстыру сұлбасы СС арқылы салыстырылады. Егер олар бір бірімен сәйкес келсе (СС=1), онда ізделген блоктың КЭШ жадыда сақталып тұрғандығын білдіреді. Мұндай жағдайда сұлба осы тегке сәйкес келетін блок нөмірін (Б) сыртқа шығарып, НЖ мекенжай кодындағы сөз кодымен қосылып, КЭШ жадыға қатынас құратын мекенжай кодын қалыптастырады. Егер СС=0 болса, онда ол ізделген блок КЭШ жады ұяшықтарында жазылмағанын көрсетеді.

Тура кескіндеу тәсілінің артықшылығы блоктар кестесінің жеңіл құрылуы, салыстыру операциясының жылдам орындалуы. Бірақ мұндай кескіндеу тәсілінде КЭШ жадыда оның ең үлкен блок санының мекенжай кодына еселі болып келетін НЖ мекенжайында орналасқан блоктарды бір уақытта сақтай алмаймыз.

Толық ассоциативтік кескіндеу

Мұндай тәсілде НЖ блоктары КЭШ жадының кез келген блоктарында орналаса алады. Біздің мысалымызға байланысты КЭШ жадыдағы блоктарды идентификациялау үшін 5 емес 12 бит керек. **(**7.23-сурет)



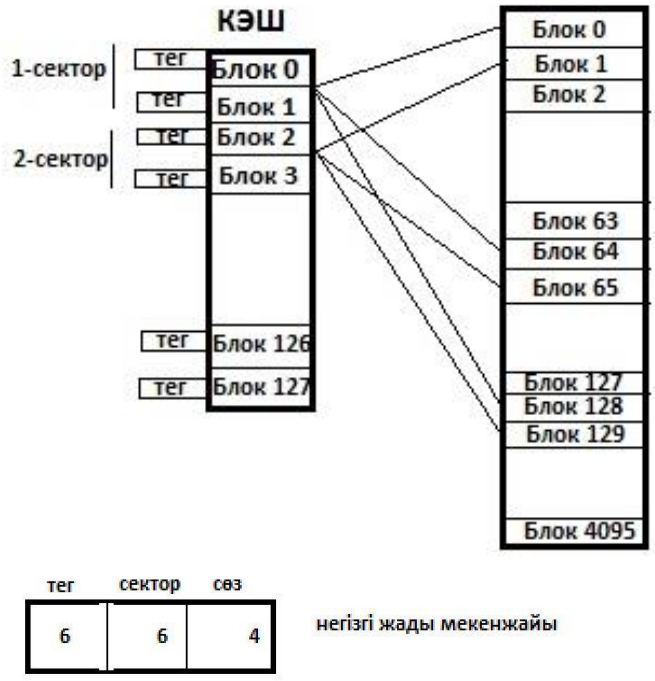
7.23-сурет. Толық ассоциативтік кескіндеу КЭШ жадысы

Бағдарлама орындалғанда процессормен генерацияланған тег биттері кезекпен КЭЩ жадының әр блоктарының ТЕГімен (Тк) салыстырылады. Егер олар бір бірімен сәйкес келсе, онда ізделген блок КЭШте бар болғаны. Мұндай технология *ассоциативтік кескіндеу* деп аталады. Ассоциативтік КЭШ жадының құны тура кескіндеу КЭШ жадыдан қымбат, өйткені мұнда 128 блоктарының тег-терін параллель салыстыру керек. Мұндай КЭШ жадыдағы блокты іздеу *ассоциативтік іздеу* деп аталады.

Секторлы ассоциативтік кескіндеу

Секторлы ассоциативтік кескіндеуде (set associative mapping) жоғарыда қаралған тура және толық ассоциативтік кескіндеулердің жақсы жақтарын ескереді. Мұндай технологияда КЭШ жады блоктары әрқайсысы К блоктардан тұратын секторларға бөлінеді және негізгі жадының әрқайсысы әр сектордың кез келген блогында орналаса алады. Секторлы ассоциативтік кескіндеудің іске асырылуы толық ассоциативтік КЭШ жадыға қарағанда қарапайым, өйткені мұнда ассоциативтік іздеу саны аз.

Мысал ретінде әрқайсысы 2 блоктан тұратын (k=2) 64 секторлы ассоциативтік КЭШ жадыны (7.24-сурет) қарайық. Мұндай КЭШ жадыда негізгі жадының 0,64,128,…,4031 блоктары 0-секторындағы екі блоктың (0,1 блоктарының) бірінде жазылады, ал 1,65,129,…,4031 блоктары 1-сектордың 2-не 3- блоктарының бірінде кескінделеді. НЖ қалған блоктары да жоғарыдағы көрсетілген тәртіппен басқа секторларда орналасады. КЭШ жадының 64 секторының бірі мекен-жай сөзінің құрамына кіретін 6 разрядты мекен-жай өрісімен анықталып, мекен-жай сөзінің тегі оқылған сектордың екі блогының тегтерімен ассоциативтік тәсілмен салыстырылады. Егер олар бір бірімен сәйкес келсе, онда керекті блок КЭШ жадыда сақталғандығын көрсетеді.



7.24-сурет. Әр секторда екі блогы бар секторлы ассоциативтік КЭШ жады.

Секторлардағы блок саны (k) әр машинада әртүрлі болуы мүмкін. Біздің мысалда қаралған НЖ мен КЭШ жадыда k=4 болса, онда жады секторларының бірін таңдау үшін 5 разрядты мекен-жай өрісі керек. Егер k=8 болса, мекен-жай өрісі 4 разрядты сан болады. Егер k=128 блоктан тұрса, онда тег биттерінің саны 12 разрядтан тұрады. Мұндай жағдайда біздер толық ассоциативтік жады

аламыз. Қарама қарсы жағдайда сектор бір блоктан тұрса, онда ол тура кескінделген КЭШ жады болып табылады. КЭШ жадының секторлары k блоктардан тұрса, онда мұндай жады К каналды секторлы ассоциативтік КЭШ жады деп аталады.

7.11. Ауани жадыны ұйымдастыру

Көптеген бағдарламаларды қатар орындайтын есептеу жүйелері жоғарғы өнімділік режимінде жұмыс істеу үшін барлық жұмыс бағдарламалары мен деректері жүйенің жедел жадында сақтаған дұрыс болар еді. Бірақ ЖЖ сыйымдылығы шектеулі болғандықтан барлық жұмыс бағдарламалардың ішінен тек белсенділерін ғана сақтап, қалғандарын сырт жадыда сақтасақ болады.

Әртүрлі есептерді шығару барысында кейбір белсенділігі төмен бағдарламалардың бөліктерін ЖЖ сырт жады құрылғысына (СЖҚ) шығарып, оның орнына бағдарламаның белсенді бөліктерін енгізу керек болады. Енгізілген бағдарлама көлемі ЖЖ сыртқа шығарылған бағдарламалардан көп орын алса, онда ЖЖ қалған бағдарламаларының орнын ауыстыру керек т.с.с осыған байланысты машиналарды көп бағдарламалық режимде істету үшін жадыны әртүрлі бағдарламалар арасында үлестіру мәселесі пайда болады.

Есептеу жүйелерінің жұмысы барысында жады (ЖЖ және СЖҚ) сыйымдылығын (мекенжай кеңістігін) әртүрлі бағдарламалар арасында үлестіру үдерісін *жадыны динамикалық үлестіру үдерісі* деп атайды. Динамикалық үлестіру негізінде жұмыс бағдарламасының шартты мекенжайы жазылады. Шартты мекенжайдан орындалатын мекенжайға түрлендіру бағдарламаны орындау кезінде жүзеге асырылады. Жадыны динамикалық үлестіру деректерді көп деңгейлі жадыға жазуға байланысты. Ол бір деңгейлі ауани жадыға негізделеді.

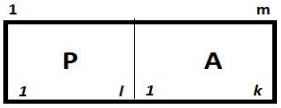
Ауани жадыда бағдарлама физикалық жедел жады мекен-жай кеңістігінде емес, ауани мекен-жай кеңістігінде жазылады. Ауани жады көлемі есептеу машинасының барлық жады түрлерінің мекен-жай кеңістігінің жиынтығынан тұрады.

Бағдарламаның ауани мекен-жайы физикалық мекен-жайға бағдарламаны орындау үстінде түрлендіріледі.

Ауани мекен-жайдан физикалық мекен жайға түрлендіру және деректерді жады құрылғыларында бірінен екіншісіне қотару үдерістерін беттік мекендеу тәсілімен орындау ыңғайлы.

Беттік мекендеу тәсілінде сөздер немесе ұяшықтар мекен-жайлары сегменттерге топталады.

Мысалы, $0,1,\ldots,Z^{k-1}$ мекен-жайларын 0-бетке топтаймыз, ал 2^k, 2^k+1, … 1-бетке т.с.с топтаймыз. Сонда ауани жады мекен-жайы екі өрістен тұрады (7.25- сурет).



7.25-сурет. Ауани жады мекен-жай құрамы.

Мұнда P - бет мекенжайы; A - P бетінің ішіндегі сөз (ұяшық) мекенжайы.

Ауани және физикалық жады беттері бірдей байттардан тұрады. Беттер нөмірленеді. Оларды ауани және физикалық беттер деп атайды.

Ауани беттер физикалық беттерге беттер кестесі арқылы түрлендіріледі. 7.1-кестеде беттер кестесіне мысал келтірілген. Кестенің әрбір қатары $P=0,1,\ldots,M$ беттерінің біріне қатысты деректер көрсетіледі. S_P - p бетінің физикалық мекен-жайын көрсетеді. dp беті p бетінің процессорға жеткізімділігін көрсетеді. Егер p беті ЖЖ орналасса, онда dp=1, ал dp=0 болса, онда p беті сырт жады құрылғысында орналасқан. T_p белгісі арқылы жедел жадының p бетіне керекті басқа деректерді жазуға болатындығы көрсетіледі.

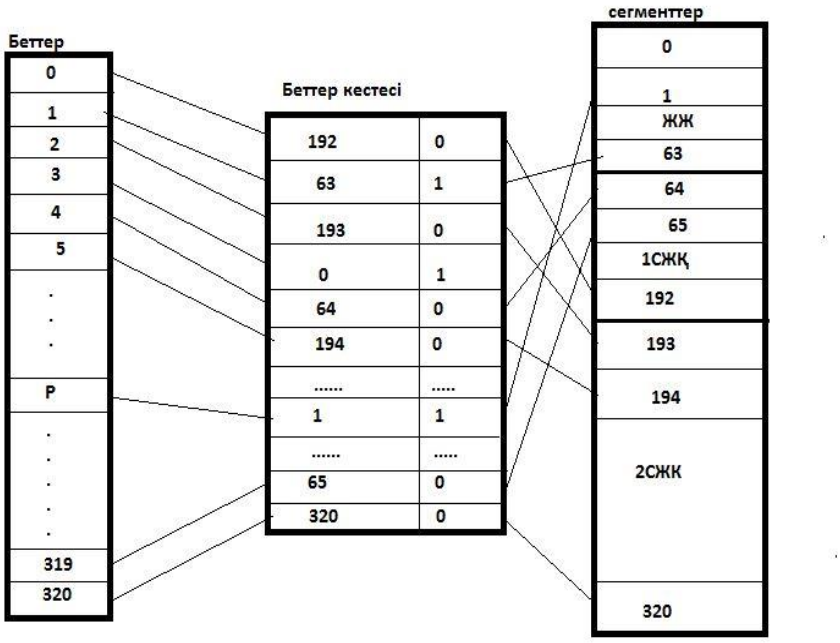
7.1-кесте. Беттер кестесіне мысал

P

0	S_0	d_0	T_0
1	S_1	d_1	T_1
⋮	⋮	⋮	⋮
P	S_P	d_P	T_P
⋮	⋮	⋮	⋮
M	S_M	d_M	T_M

Беттерді пайдалану тәртібін 7.26-суреттен көруге болады. Суреттен көрініп тұрғандай, ауани жады 320 беттерден тұрады. Олар ЖЖ 64 сегменттерінде және әрқайсысы 128 сегменттерден тұратын 1 және 2 - сырт жады құрылғыларында орналасқан.

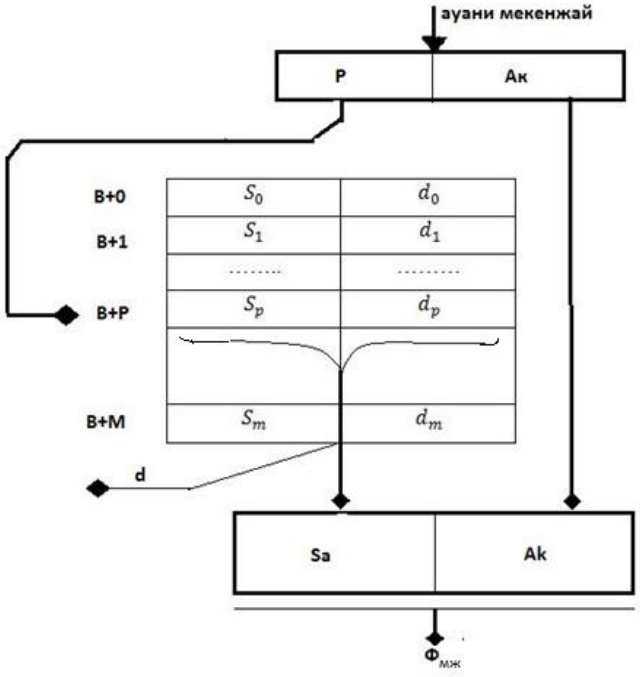
Әрбір 0,1,...,320 беттерге мекен-жайы бет кестесінде көрсетілген сегменттер сәйкес келеді, яғни әр бетке өзінің физикалық мекен-жайы меншіктелген. Жады құрылғыларында беттердің кез-келген жылжытылу кезінде бет кестесіндегі физикалық мекен-жайларға түзету енгізіледі.



7.26-сурет Беттер мен сегменттердің сәйкестігі

Ауани мекен-жайларды түрлендіру

Бағдарламаларда командалар мен деректер ауани мекен-жайлармен сәйкестендіріледі. Мұндай жағдайда сөзге қатынас құру үшін ауани мекен-жайды физикалық мекен-жайға түрлендіру қажет. Ол үшін процессор құрамына мекен-жай түрлендіргіш сұлбасын енгізу керек. Мұндай мекен-жай түрлендіргіш сұлбасы 7.27-суретте көрсетілген.

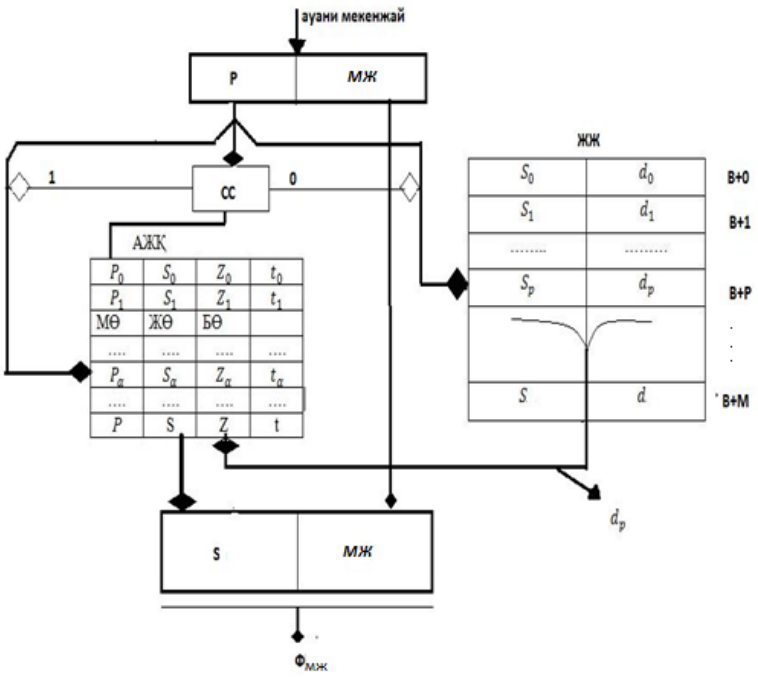


7.27-сурет. Беттер кестесіне қатынас құру арқылы ауани мекенжайларды түрлендіру

Ауани мекен-жай арқылы кезекті сөз сақталған ұяшық мекенжайын алу үшін ЖЖ В+Р ұяшығына қатынас құрып, одан Р бетін сипаттайтын параметрлерді S_p және d_p мәндерін оқимыз. Мұндағы В - бет кестесінің ЖЖ бекітілген базалық мекенжайы. Егер $d_p = 0$ болса, онда қазіргі уақыт мезетінде Р бетін өңдеуге мүмкіндік болмайды. Сондықтан процессор жұмысы үзіледі. Басқару бетті ауыстыру әрекеттерін жүргізетін жабдықтарға береді. Егер $d_p = 1$, онда S:=S_p. S команда мекен-жайы Ак-мен бірігіп ЖЖ физикалық мекен-жайын $Ф_{мж}$ қалыптастырады.

Мұндай тәсілде біздер ЖЖ-ға екі рет қатынас құрамыз: біріншісінде бет кестесінен S_p мен d_p оқимыз, екіншісінде $Ф_{мж}$ арқылы керекті деректі аламыз. Сондықтан машина жылдамдығы екі есе азаяды.

ЖЖ қатынас құру уақытын азайту үшін ауани мекенжайды түрлендіруге ассоциативтік жады құрылғысын (АЖҚ) пайдалануға болады. (7.28-сурет)



7.28-сурет. Ассоциативтік жадыны пайдаланып ауани мекенжайын түрлендіру.

Ассоциативтік жадының (АЖҚ) $\alpha = 0,1 \ldots, m$ ұяшықтарында ЖЖ-да сақталған $P_0, P_1, \ldots, P_m$ беттерінің параметрлері орналастырылады және оның мекенжай өрісінде (МӨ) $P_0, P_1, \ldots, P_m$ ауани мекенжайы орналасқан. Жұмыс өрісінде (ЖӨ) осы беттердің $S_0, S_1, \ldots, S_m$ физикалық мекен-жайлары

көрсетілген. Басқару өрісінде (БӨ) «бос емес» белгілері 0,1,…,m ұяшықтарының $Z_0, Z_1, \dots, Z_m$ бос еместік белгілерімен, ал осы ұяшықтардың белсенділік белгісі $t_0, t_1, \dots, t_m$ арқылы көрсетілген.

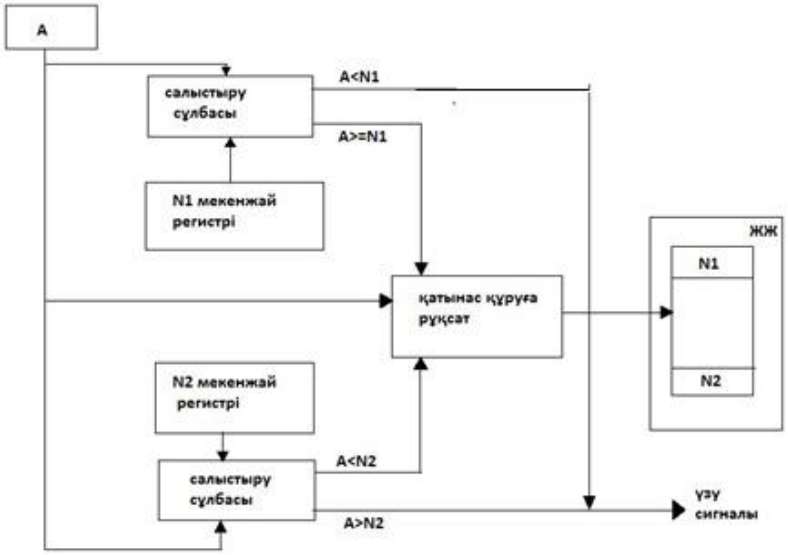
Егер $Z_\alpha = 0$ болса, онда АЖҚ α ұяшығы бос (бет параметрлері жазылмаған), ал $Z_\alpha = 1$, болса онда бет параметрі α ұяшығында сақталған. Егер $t_i = 0$, онда осы қатарда жазылған сөзді белсенді еместігін көрсетеді, ал $t_i = 1$, онда ол оның белсенділігін көрсетеді.

Беттер кестесі ЖЖ ұяшықтарында сақталады. Түрлендіру АЖҚ ұяшықтарындағы Р бетінің деректерінің бар екендігін іздестіреміз. Ол үшін салыстыру сұлбасы (СС) арқылы мекенжай өрістерін (МӨ) ауани мекенжайымен (Р) салыстырамыз. Егер олар сәйкес келсе (СС=1), онда түрлендіру АЖҚ α ұяшығында жазылған S_α физикалық мекен-жайін іріктеуге алып келеді. Егер олардың біреуі де сәйкес келмесе (СС=0), онда Р бетінің орнына ЖЖ-да орналасқан беттер кестесіне қатынас құрып, одан S_p физикалық мекенжайын және Р бетіне қолжеткізімдік белгісін d_p оқимыз. Егер d_p=1 болса, ол Р бетінің қолжеткізімділігін көрсетеді (р беті ЖЖ ұяшықтарында сақталған). Мұндай жағдайда оның параметрлері АЖҚ басқару өрісімен (БӨ) анықталатын σ ұяшығына жазылады. σ ұяшығы $Z_\sigma = 0$ белгісімен анықталатын бос ұяшықтарының бірі. Егер АЖҚ барлық ұяшықтары бос болмаса, онда қай ұяшықтың белсенділігі төмен болса ($t_\sigma = 0$) сол ұяшыққа Р бетінің параметрлерін жазамыз.

7.12. Жедел жадыны қорғау

Есептеу жүйелерінің процессорында әдетте бірнеше есептер қатар шығарылады. Олардың бағдарламалары жедел жадының әртүрлі мекенжай кеңістігінде орналасады. Мұндай жағдайда бір есептің бағдарламасын басқа бағдарламадан қорғау керек. Орындалып жатқан бағдарлама мекенжайына оған қатысы жоқ басқа бағдарлама қомандасы қатынас құратын болса, онда бірінші бағдарламаны қорғауға тура келеді. Қорғамаған жағдайда орындалып жатқан деректер бүлінуі мүмкін (бағдарлама командасының ұяшықтарына басқа деректер жазылуы немесе ол тіптеп өшірілуі мүмкін). Мұндай жағдай болмау үшін қорғау жүйесі жедел жадыға қатынас құратын мекенжайды тексеру керек. Егер ол басқа бағдарламаның мекенжайына жататын болса, онда қорғау сұлбасы үзу сигналын шығарады. Кері жағдайда ЖЖ қатынас құруға рұқсат беріледі. Жедел жадыны қорғаудың белгілі екі тәсілі бар. Оларға шекаралық мекенжайлар арқылы және кілттер арқылы қорғау жатады. Шекаралық қорғау тәсілінде әр бағдарламаға операциялық жүйе бағдарламалары арқылы нөмері

N_1 деп басталатын тұтас мекенжай кеңістігі бөлінеді. Мекенжай кеңістігінің соңғы нөмері N_2. Бағдарлама А мекенжайы арқылы ЖЖ қатынас құратын болса, оның N_1 мен N_2 аралығына жататындығы тексеріледі ($N_1 \leq A < N_2$).

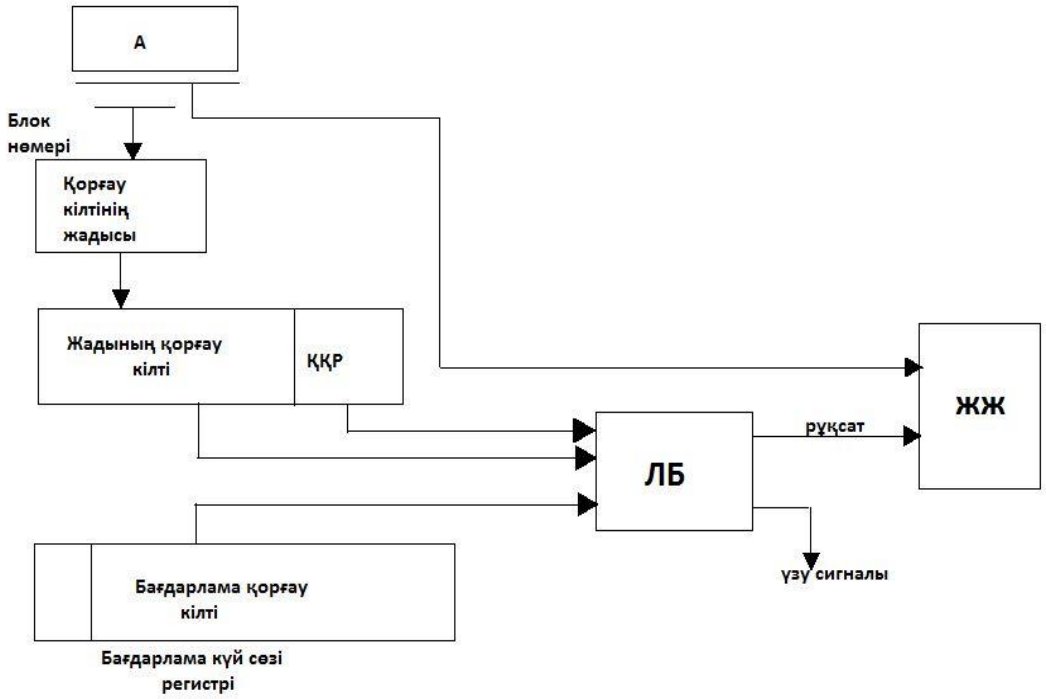


7.29-сурет. Шекаралық мекенжай арқылы қорғау

Егер A<N_1 және $A > N_2$ болса, онда қорғау сұлбасы үзу сигналын қалыптастырады, егер A$\leq N_1$ және A<N_2 болса, онда ЖЖ қатынас құруға рұқсат етіледі. Салыстыру операциялары салыстыру сұлбалары арқылы жүргізіледі, N_1 және N_2 мекенжайлары арнайы регистрлерде сақталады (7.29-сурет).

Шекаралық мекенжайлар N_1 және N_2 тиісті регистрлерге операциялық жүйе арқылы жазылады. Негізгі кеңістігі әр бағдарламаға тұтас мекенжай кеңістігі керек.

ЖЖ сақталған деректер мен бағдарламаларды қорғаудың екінші тәсіліне кілттер арқылы қорғау тәсілі жатады. ЖЖ ұяшықтары шартты түрде өлшемі бірдей блоктарға бөлінеді. Әрбір блокқа жадыны қорғау кілті (ЖҚҚ) деген екілік код меншіктеледі. Әрбір бағдарламаларға да қорғау кілті (БҚҚ) меншіктеледі. Бағдарламаның ЖЖ белгілі бір блогына қатынас құру үшін олардың қорғау кілттері сәйкес келу керек. Егер бағдарлама кілті нөльге тең болса, онда ол ЖЖ кез келген блогына қатынас құра алады. Мұндай нөльге тең кілт тек операциялық жүйенің бағдарламаларына меншіктеледі. Қорғау кілттері операциялық жүйе бағдарламасы арқылы үлестіріледі. Бағдарлама қорғау кілті (БҚҚ) процессордың құрамында болатын бағдарлама күй сөзі (БКС) арнайы регистрінің жеке өрісінде сақталады. Жады қорғау кілті ЖЖ қарағанда жылдамдығы жоғары арнайы жады құрылғысында сақталады (қорғау кілттерінің жады ҚКЖ). ЖЖ әрбір қатынас құру үстінде арнайы логикалық блок (ЛБ) жады қорғау кілті мен бағдалама кілттерін салыстырады (7.30-сурет).



7.30-сурет. Жедел жадыны кілт арқылы қорғау

Олар сәйкес келсе, онда ЖЖ-ға қатынас құруға рұқсат беріледі. Сәйкес келмеген жағдайда ЛБ ЖЖ қатынас құрудың тыйым салу түрін анықтайды.

Тыйым салу және оқуға да, жазуға да тыйым салу жатады. Олар қорғау кілті жадының қосымша разрядында сақталып, қатынас құру режимінің (ҚҚР) түрін көрсетеді. Егер ҚҚР=0 болса,онда ЖЖ тек жазу режимінде қорғалады. ҚҚР=1 болса, онда ол жазу және оқу режимдерінде қорғалады. Егер қатынас құру түріне тыйым салынса, онда үзу сигналы арқылы операциялық жүйенің тиісті бағдарламасы іске қосылады.

ЖЖ қатынас құрғанда орындалатын мекенжай кодының жоғарғы мекенжайы (блок нөмері) арқылы ҚКЖ жады қорғау кілті оқылады. Комбинациялық сұлба жадыны қорғау кілті мен бағдарлама қорғау кілттерін салыстырады. Қатынас құру режимі белгісін еске алып «қатынас құруға рұқсат» немесе «үзу» сигналдарын қалыптастырады. Кілт арқылы қорғау тәсілі бағдарлама блоктарын ЖЖ-ның кез келген мекенжай кеңістігінде сақтауға мүмкіндік береді.

7.13. Сырт жады құрылғылары (СЖҚ)

Жады құрылғыларының иерархиясында өзіндік орын алатын құрылғыларға әртүрлі жады құрылғыларынан тұратын сырт жады құрылғылары (СЖҚ) жатады. Олардың ішіндегі кең тарағандарына қатты денелі жартылай өткізгіш, магниттік және оптикалық жады құрылғылары жатады.

СЖҚ әдетте жеке конструктивтік блок түрінде есептеу машинасының корпусының ішінде орналастырылады. Машинаның орталық бөлігіне енгізу-шығару құрылғылары сияқты қосылады.

СЖҚ сыйымдылығы НЖ қарағанда өте үлкен, сондықтан олар мегабайтпен, гигабайтпен, терабайтпен өлшенеді. СЖҚ ақпаратқа қатынас құрудың (ҚҚ) екі тәсілдері қолданылады. Олар: тізбекті ҚҚ және тура ҚҚ тәсілдері.

Тізбекті қатынас. Жады құрылғыларында керекті сөзге немесе байтқа қол жеткізу үшін оның алдыңғы деректерінің барлығын оқу керек. Деректерге қол жеткізу уақыты керекті деректің тасымалдауышта жазылған орнына байланысты болады. Мұндай жады құрылғысына магниттік таспа негізінде құрылған жады жатады.

Тура қатынас құру тәсілінде әрбір жазудың бірегей мекенжайы болады. Ол дерек тасымалдауышында блоктардың орналасуына (позициясына) байланысты болады. Жадымен қатынасу кезінде жазудың бастапқы мекенжайы көрсетіледі. Одан соң жазудың ішінен тізбекпен оқылатын дерек бірлігі көрсетіледі. Тура қатынас тәсілі магниттік диск негізінде құрылған СЖҚ-да қолданылады.

Жылдамдық N биттерді оқу немесе жазу T_N уақыттарымен анықталады.

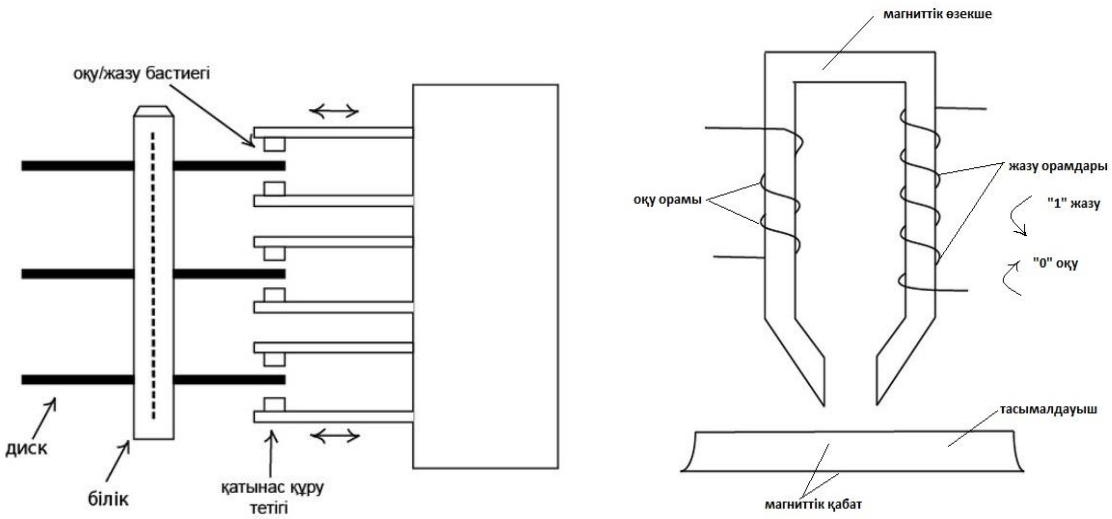
Тасымалдауыштары жылжымалы СЖҚ Тққ магнитті жазу/оқу бастиектері керекті позицияға орналастыру уақытымен анықталады.

N биттерді оқу немесе жазу уақыты $T_N = T_A + \frac{N}{R}$ формуласымен анықталады. Мұнда T_A орташа қатынас құруға кететін уақыт; R секундына жазылатын (оқылатын) бит саны.

7.13.1. Магниттік дисктер негізінде құрылған СЖҚ

Магниттік дисктері ортақ біліктерге орнатылған бір немесе бірнеше дисктерден тұрады (7.31, а-сурет)

Дисктер магниттелмейтін материалдардың (шыны немесе алюминий) екі бетіне магниттелетін материалдарды қаптау арқылы жасалынады. Білік пен дисктер электр моторы арқылы тұрақты жылдамдықпен айналады. Айналу кезінде магниттелген диск биттері жазу және оқу операцияларын орындайтын магниттік бастиектер арасында қорғалады. Магниттік бастиек орамалары бар өзекшеден тұратын электромагнит (7.31, б-сурет). Оқу және жазу кезінде орамалар арқылы әртүрлі бағытта ток өткізіледі.



а) б)

7.31-сурет. Магниттік диск: а) механикалық бөлігі; б) жазу/оқу магниттік бастиек

Маггниттік тасымалдауыш учаскілеріне 1 немесе 0 орауышқа берілетін токтың бағытына байланысты жазылады. (7.31, б-сурет) магниттік бастиек осы токтың бағытына сәйкес дискі бетінің (учаскесінің) магниттік векторларын өзгертеді. Деректі оқығанда осы учаске магниттік бастиектің тұсынан өткен кезде, ол оқу орамында кернеу индукциялайды. Басқару сұлбалары индукцияланған кернеуді таңдап оқылған деректің 0 не 1 екендігін анықтайды. Қазіргі есептеу жүйелеріндегі дискілі жады құрылғыларында дискілер мен оқу жазу бастиектегі жабық корпусқа біріктірілген. Мұндай қатқыл магниттік дискілерді (ҚМД) *винчестер* деп атайды. Мұндай ҚМД магниттік бастиектер тасымалдауыш беттеріне және бір бірімен өте жақын орналастырылады. Бұл диск жолақшаларында биттерді тығыз орналастыруға мүмкіндік береді және жолақшалар санын жақын орналастырып, дисктің ақпараттық сыйымдылығын арттыруға алып келеді. Оның үстіне жабық корпус диск беттерін шаң тозаңдардан қорғайды.

Оқу/жазу бастиектер саны диск беттерінің санымен анықталады. Бастиектер қозғалмалы болады. Дискінің екі шеттеріне дейін деректі кез келген орталық жолақшаларымен алмаса алады.

Дискілік жүйені үш бөлікке бөліп қарауға болады. Біріншісіне дискілік табақшалар жиыны жатады. Екіншісі диск жетегі жатпақ. Ол дискілерді айналдыратын және магниттік бастиектерді жылжытатын электромеханикалық құрылғылар. Үшінші бөлігіне дискілік жүйе жұмысын басқаратын электронды сұлбалар жатады. Электронды сұлбаларына басқару блогы, тұрақты жады, дискінің КЭШ жады, интерфейс блогы, сигналдарды цифрлық өңдейтін блоктар жатады.

Интерфейс блогы арқылы дискілік жүйе есептеу машинасының ядросымен байланыс құрайды.

Дискінің КЭШ жадысы дискілік жады мен ЖЖ жылдамдықтарын сәйкестендіреді. Мұндай жады статикалық ЖЖ микросхемалары арқылы құрылады. Басқару блогы жады құрылғысының барлық іс әрекеттерін басқарады.

Сигналдарды цифрлық өңдеу блогы оқылған аналогты сигналдарын декодалап, одан цифрлық сигналдар қалыптастырады. Тұрақты жады сигналдарды цифрлық өңдеу бағдарламаларымен қатар жүйенің қызмет көрсету бағдарламаларын сақтауға арналған.

ҚМД сипаттамаларына интерфейстер түрі, сыйымдылығы, қолжеткізім уақыт мөлшері т.б жатады. Есептеу машиналарына қосу тәсілдеріне байланысты ҚМД екіге бөлінеді: ішкі және сыртқы ҚМД. Бірінші түрі ЕМ корпусының ішіне бекітіледі. Мұндай дисктерді ЕМ-ге қосу үшін АТА стандарттарын (IDE,DATA),SATA, eSATA, SCSI, SAS, Fire Wire, SDIO және Fibre Channel пайдалануға болады. Сыртқы КМД есептеу машиналарының сырт құрылғылар тәрізді қосылады. ҚМД сыйымдылығы 2013 жылдың соңында 3,5 дюйімдік дискісі бар винчестерде 6Т байтқа, ал 2,5 дюйімдік дискілі ЖҚ 1,5Т байтқа жетті.

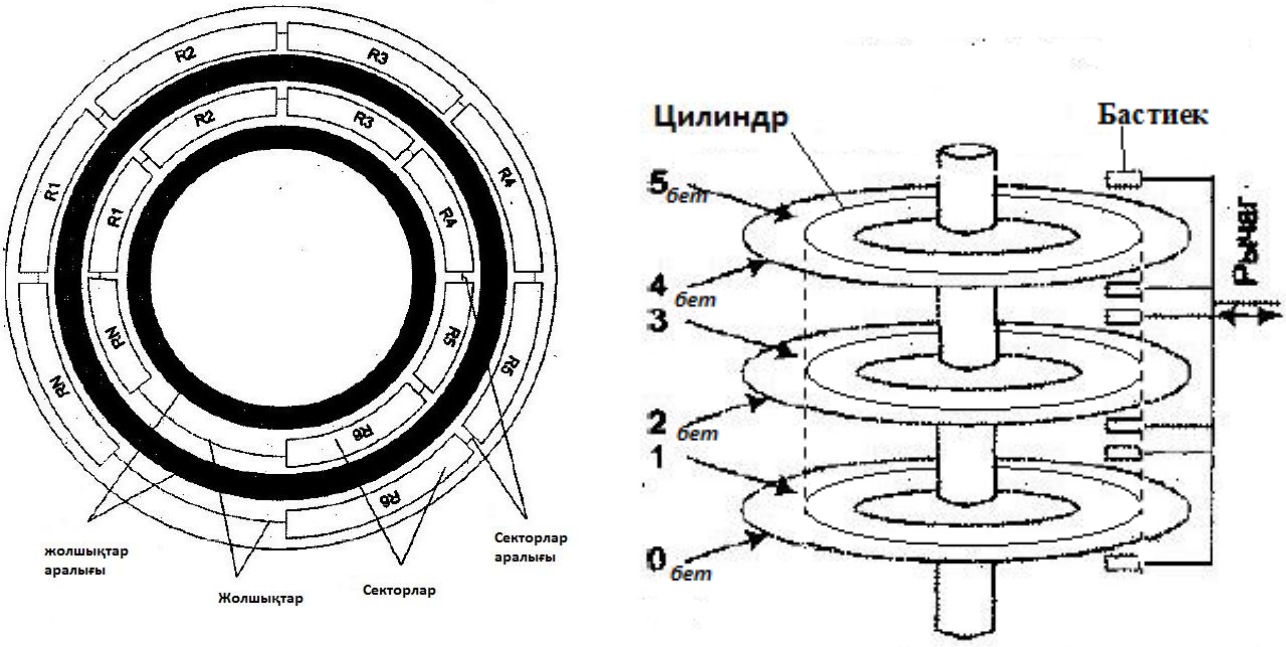
Қолжеткізім уақыты магниттік бастиектің керекті жолшыққа позициялауға кететін уақыт мөлшерімен анықталады. Бұл уақыт қазіргі КМД 2,5÷ 16 мс-ке тең.

Дискілердің айналу жылдамдығы диск білігінің минуттағы айналу санымен анықталады. Айналу жылдамдығы қолжеткізім уақыты мен деректерді беру жылдамдығына әсер етеді. Айналу жылдамдықтары ноутбуктерде - 5400,7200 а/мин, дербес компьютерлерде 5400÷10000 а/мин. Серверлер мен жұмыс стансаларда -10000,15000 а/мин.

Деректерді ұйымдастыру және дискілерді пішімдеу.

Дискілерде деректер жолшықтар деп аталатын орталықтанған шеңберлер арқылы ұйымдастырылады (7.32- сурет), әрбір жолшық кезегімен секторларға бөлінеді. Көрші жолшықтар секторлары бір бірімен аралықтар арқылы бөлінеді. Электрондық сұлбаларын ықшамдау үшін барлық жолшықтардағы дерек сыйымдылықтарын бірдей етіп орналастырылады. Ол үшін жолшықтарға жазылған деректер тығыздығы сыртқы шеңберден ішкі шеңберге қарай өсе түсуі керек.

Қазіргі магниттік дискілі жады құрылғылары бірнеше диск табақшаларынан тұрады.



7.32- сурет. Магниттік дискідегі деректерді орналастыру реттері.

Диск табақша беттеріндегі ортадан бірдей қашықтықта орналасқан жолшықтар жиынтығын цилиндр деп атайды. Әрбір жолшық секторларға бөлінеді. Файлдармен жұмыс жасау үшін бірнеше қатар орналасқан секторлар біріктіріліп кластер құрайды.

КМД 380 ÷700 секторлардан тұрады. Әр секторда - 4096 байттардан тұратын дерек жазылады.

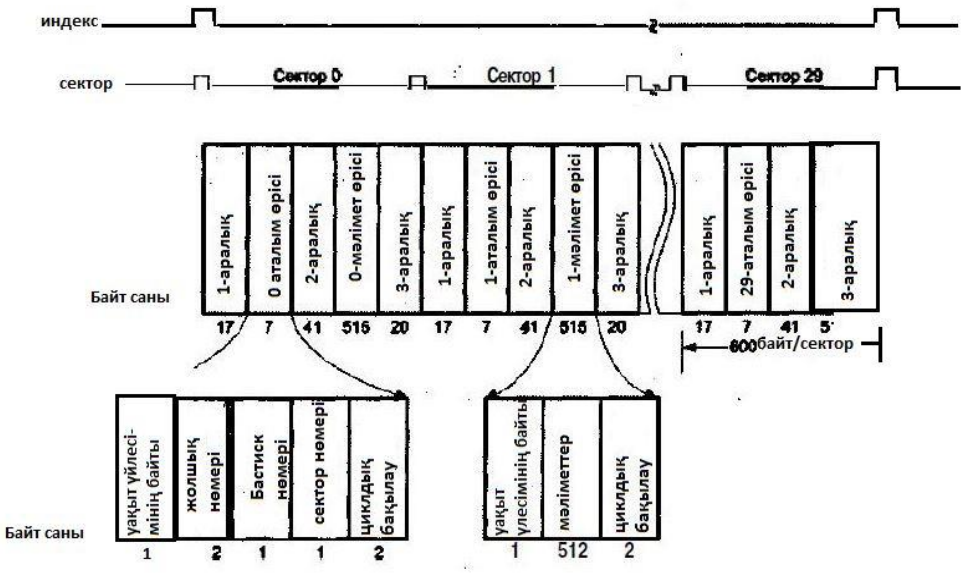
Көрші секторлар бір-бірімен секторлар аралығымен бөлінген. Мұндай аралықтарға сервистік ақпараттар жазылады. Сервистік ақпараттар жазу/оқу бастиектерінің дискінің бетінде орналасуын анықтауға мүмкіндік береді. Секторлар мекені оның дискіде орналасқан физикалық орнына байланысты: цилиндр нөмері, бастиек нөмері және сектор нөмері арылы анықталады. Бұл тәсілді CHS (cylinder head sector) деп атайды.

ҚМД пайдалану үшін диск табақшаларын пішімдеу керек. Пішімдеу кезінде оларға қызмет деректері жазылады.

7.33-суретте пішімдеуға мысал көрсетілген. Мұнда әрбір жолшық 30 сектордан тұрады. Әрбір секторға 600 байт жазылады. Оның 512 байттарында деректер жазылады, ал қалған байттарында диск жұмысын бақылайтын басқару деректері жазылады. Аталым өрісі арқылы сектор параметрлері анықталады. Уақыт үйлесімдіру байты арқылы дерек өрісінің басталуы анықталады. Келесі екі байт арқылы жолшық нөмері көрсетіледі. Егер жинақтағышта бірнеше табақша болса, онда бастиек нөмері арқылы керекті бет анықталады. Сектор нөмері арқылы таңдалған беттегі жолшықтың ішінен керекті сектор анықталады. Аталым өрісінің соңғы екі байтында жіберілген қателерді

бақылайтын циклдік код жазылады. Әдетте циклдік коды өрісіне жазылған байттары екі модулімен қосу арқылы анықталады.

Дерек өрістері уақыт үйлесімдіру байтынан 512 байттық деректерден және циклдік бақылау кодтарынан тұрады.



7.33- сурет. Винчестер (Seagate ST506) типті дискінің жолшықтар пішімі

Басы артық магниттік дискілер

Қазіргі компьютерлік жүйелерде магниттік дискілер арқылы құрылған жады құрылғылары негізгі сырт құрылғылары болып отыр. 1988 жылы Калифорния университетінің мамандары деректер бірнеше дискілерде сақталатын жүйе ұсынды. Ондай жүйені RAID (Redundant Array of Inexpensive Disks-арзан басы артық дискілер массиві) деп атады. RAID тұжырымдамасының негізінде сыйымдылығы үлкен бір физикалық магниттік дискіден параллель жұмыс жасайтын тәуелсіз диск массивтеріне өту жатады. Машина операциялық жүйесі мұндай массивті үлкен бір логикалық физикалық жады құрылғысы деп қарастырады. Дискі жадысын мұндай жолмен құрғанда оның жұмыс өнімділігін арттыруға алып келеді, өйткені мұндай жады жүйесінде оқу және жазу сұраныстарын қатар орындауға мүмкіндік береді. Ал сенімділік диск массивтерінде басы артық деректер арқылы мүмкін болған қателерді анықтап және оларды түзету арқылы арттырылады.

RAID массиві үшін алты әртүрлі пішін үйлесімін ұсынады.

RAID 0 - бұл базалық пішім, ол арқылы жүйе өнімділігі артады. Мұнда қатпарлау тәсілі қолданылады, оның негізінде дискілік кеңістікті жолақ деп аталатын сегменттерге бөледі. Жолақтар әртүрлі дискілер массивіне белгілі бір жүйемен бөлінеді. Мұндай қатпарлау тәсілі әртүрлі дискілерде жазылған

жолақтардан деректерді параллель оқуға және жазуға мүмкіндік береді. Өнімділік массивтегі дискілер санына байланысты өседі.

RAID 1 сәулетінде деректер бір дискіде емес екі дискіде жазылады. Бұл деректі сақтау сенімділігін арттырады. Мұндай дискілерді айналы дискілер деп атайды. Жұмыс үстінде егер олардың бірі істен шықса, онда оқу және жазу операциялары екіншісімен жалғастырылады. RAID 2, RAID 3 және RAID 4 пішімдері жүйе сенімділігін арттыруға арналған. Ол үшін әрбір жолақ топтарының Хэмминг коды есептеліп, түзету биттері қосымша дискіде сақталады немесе Хэмминг кодының орнына қарапайым бит паритеті (жұптығы) есептелінеді.

RAID 5 сәулетінде де қателерді анықтау бит жұптарын анықтау тәсіліне негізделген. Бірақ жұпқа тексеруге арналған деректер барлық дискілерге бөлінеді.

Иілгіш дискілер

Жоғарыда қаралған құрылғылар қатты дискілер (ҚД) деп аталады. Қатты дискілерден басқа қарапайым, арзан, алмалы-салмалы, өлшемі кіші, иілгіш магниттік материалмен қапталған пластик дискілер бар. Мұндай дискілер саңылауы бар пластикалық конвертке салынған. Саңылау арқылы дискі оқу/жазу магнитті бастиекпен түйіседі. Дискінің қақ ортасында дискетті айналдыруға арналған тиек бар. Деректерді оқу және жазу жоғарыда қаралған магниттік дискілердегі оқу/жазуға ұқсас. Стандартты дискінің диаметрі 3,25 дюйм. Сыйымдылығы 1,44 немесе 2Мбайт. Сыйымдылығы жоғары иілгіш дискілер де болады. Оларды ZIP дискілер деп атайды. Мұндай дискілерге 100 Мбайт дерек сақтауға болады.

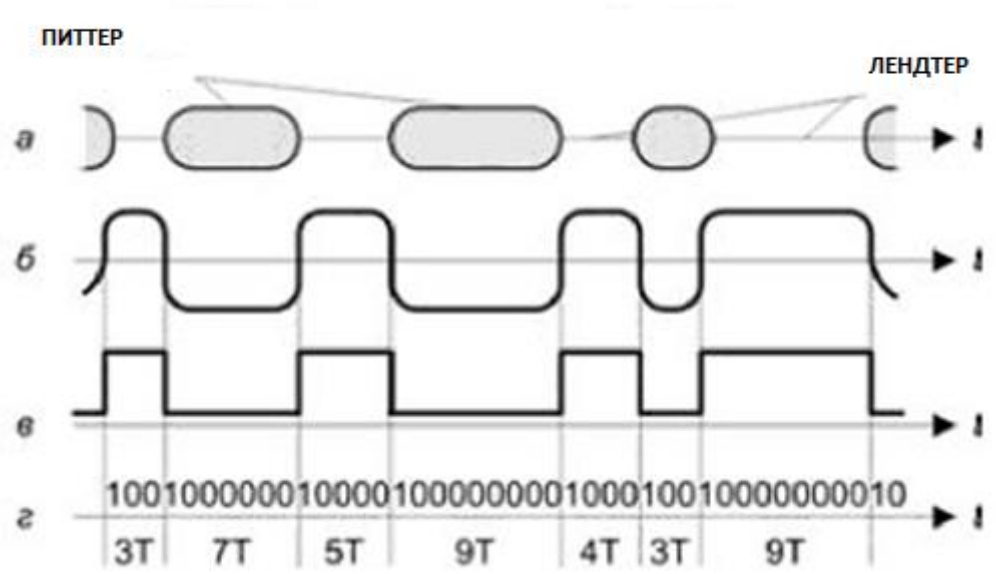
7.13.2. Оптикалық дискілер негізінде құрылған СЖҚ

Оптикалық дискілер алғашқы кезде дыбыстарды жазу үшін қолданылды, бірақ тез арада есептеу машиналарының сырт жадысы ретінде қолданыла бастады. Олар бірте-бірте жетілдіріліп, қазіргі кезде CD (Compact Disk-жинақы диск), DVD (Digital Versatile Disks-цифрлық әмбебап дискі), BD (Blue-ray Disks- көгілдір сәулелі дискі) түрлері компьютерлерде қолданыс табуда. Оптикалық дискісі жады құрылғылары (ОДЖҚ) екі бөліктен тұрады: алмалы-салмалы оптикалық дискіден және оны айналдыратын жетектен тұрады. Жетекте жазатын және оқитын лазерлер мен фотодетектор және электронды сұлбалар бар. Дерек тасымалдаушы ретінде сәуле шағылыштырғыш қабаты бар, диаметрі 12 немесе 8 см пластмассалық дискілер қолданылады. Деректерді оқу және жазу айналмалы бетке бағытталған лазерлер сәулелері арқылы орындалады. Деректерді жазу кезінде дискінің белгілі бір учаскілері лазер сәулесімен қызады. Сәулеленген жерлерде ойық пайда болады немесе учаскі

басқа күйге өтеді. Екеуінде де сәуле түскен учаскінің шағылысу көрсеткіші өзгереді. Күйі өзгерген учаскіні немесе ойықты «ПИТ» (pit) деп атайды. Пит аралығын «ЛЕНД» (land) деп атайды. Лазер сәулесінің траекториясы айналмалы дискінің бетінде спиральді жолшық түрінде түседі. Ол дискінің қақ ортасынан бастап оның шетіне қарай созылады.

Лазер толқын ұзындықтары бірдей синхронизацияланған когерентті сәуле шығарады. Егер екі бірдей сәулені бір фазамен біріктірсеңіз, онда бұрынғыдан да күшті жарық пайда болады. Егер сәулені жартылай фазаға жылжытсаңыз, олар бірін бірі өшіреді. Егер де осындай екі сәулені фотодетекторға бағыттасаңыз, онда бірінші жағдайда өте жарық дақ, ал екіншісінде қара дақ аласыз. Деректерді оқу кезінде лазер сәулесі ПИТ арқылы немесе ЛЕНД арқылы өтіп отырады, мұндай уақыттарда фотодетектор өте жарық сәуле қабылдайды. Сәуле ПИТ-пен ЛЕНД-қа немесе ЛЕНД-тан ПИТ-ке өткен кезде олардан шағылып келген сәулелер толқыны бір-бірімен 180^0 -қа жылжып бір бірін өшіреді. Сонымен ПИТ пен ЛЕНД шекарасында детекторға еш сәуле түспейді. Детектор тек қара дақты қабылдайды. Детекторға келіп түскен жарық дақты 0-ден алсақ, онда қара дақты 1-ден аламыз. Бірақ дискіден оқылған 0 мен 1 тізбектері дискіде жазылған деректерге сәйкес келмейді. Ол үшін күрделі кодтау жүйесі қолданылады. Әрбір байт кодалары қателерді анықтайтын және түзететін 14 разрядты 8421 кодасымен алмастырылады. 14 разрядты кодаға және үш разрядты біріктіру деп аталатын кодын қосады.

Мұндай 17 разрядты кодада екі бірлер арасында нөльдер саны екі нөльден аз және он нөльден көп болмайды. Дискідегі ПИТ және ЛЕНД-тер осындай кодаларда көрсетіледі. ЛЕНД-тер мен ПИТ-тердің ұзындықтары әртүрлі болады, осыған байланысты фотодетектор мен байланысқан электр сұлбасы такт жиілігінің санымен анықталатын уақыт интервалы санын анықтайды. Әрбір ЛЕНД пен ПИТ-те уақыт интервалдарының саны 3-ке (3Т интервал) тең, 11 (интервал 11)-ге дейін өзгеруі мүмкін. Уақыт интервалының ішіне 1-ге тең бит те кіреді. ПИТ-тен ЛЕНД-ке (немесе кері) өту логикалық 1-ге сәйкес келеді. Ал 0 - мұндай өтулердің жоқ екендігін көрсетеді (7.34-сурет).



7.34-сурет. Оптикалық дискіде ақпаратты көрсету: а) ПИТ-тер мен ЛЕНД-тер; б) фотодетектордан алынған сигналдар; в) түрленген ақпараттық сигналдар; г) интервалдар.

Фотодетектордан алынған сигналдар тікбұрышты сигналдарға қалыптастырылады. Соңғы сигналдардың ұзындығы оқылатын кодалар тізбегінің биттерімен анықталады. Әрбір оқылған 14 биттер тізбегі бір байттың мәнін анықтайды (біріктіру кодын шығарып тастағанда). Сонымен оптикалық дискіде ақпараттар спираль бойында орналасқан ПИТ-тер мен ЛЕНД-терден тұрады. Әртүрлі ОДЖҚ бір бірінен олардың лазер толқын ұзындықтарымен ерекшеленеді. Толқын ұзындығы кіші болған сайын фокусталған лазер сәулесінің өлшемі аз болады, ол ПИТ пен ЛЕНД өлшемін азайтуға алып келеді, жолшық екі және спираль орамының арақашықтығы азаяды. Соның салдарынан дискінің ақпараттық сыйымдылығы артады.

Жазу тәсілдеріне байланысты ОДЖҚ үш түрге бөлінеді: бірінші түріне деректі дискіні жасау кезінде жазатын прессталған дискілер жатады. Мұнда жазылған деректер тек оқылады. Сондықтан оны CD-ROM деп атайды. Мұндай дискілердің жұмыс бетінің түсі ақ болады. Екінші түріне дерек қолданушылар арқылы бір рет жазылады. Пайдалану уақытында деректер тек оқылады. Сондықтан оларды CD-R немесе DVD-R (R-Recovdale-жазылатын) деп атайды. Дискілер жасылша немесе көкшіл түсті болады. CD-R архив материалдарын және файлдарды сақтауға өте ыңғайлы. Үшінші түріне қайта жазылатын дискілер жатады. Оларға RW (Rewritable) белгісі жазылады. Мысалы, CD-RW немесе DVD-RW. Мұндай оптикалық дискілерде ақпарат магниттік дискілерге ұқсап қайта-қайта жазылатын мүмкіндікке ие. Мұндай дискілерде фазалық өту эффектісі қолданылады. Ол үшін дискіні жасауға екі түрлі фазалық күйде бола алатын материалдар қолданылады. Бірінші түрде материал аморфты күйде

болады, оның молекулалары кездейсоқ бағытта болады. Сондықтан ол өзіне түскен сәулені нашар шағылыстырады. Екінші күйіне кристалдық күйі жатады. Мұндай күйде материалдық шағылыстыру мүмкіндігі жоғары. Лазер сәулесінің әсерімен оптикалық дискінің белсенді қатпары өз күйін кристалдық күйінен аморфты күйге және кері күйіне өте алады.

Фазалық күйге өтетін оптикалық дискілердің ең үлкен кемістігіне материалдардың уақыт өте керекті қасиетін жоғалтуы жатады. RW дискілерді қайта жазу мүмкіндігіне байланысты магниттік дискілерді қосалқы жады ретінде қолдануға мүмкіндік береді.

CD оптикалық дискілері

Жинақы диск (CD-compact disk) бір жақты екілік деректерді сақтайтын диск түріне жатады. Типтік варианттарында спираль тәрізді жолшықтар арасы 1,6 микрон. Оның орама саны 20344, яғни спираль жолшық ұзындығы 5,27км. Диск тұрақты түзу сызықты 1,2 м/сек жылдамдықпен айналады. Жазылған деректерді оқу үшін 72,3 милпут керек. Деректер дискіден 176,4 Кбайт/с жылдамдығымен оқылатын болғандықтан оның сыйымдылығы 774,57 Мбайт.

CD-ROM-да деректер блоктармен тізбектеле орналасады. Блоктың типтік пішімі 7.35-суретте көрсетілген. Блокта мынадай өрістер бар:

- Синхронизация өрісі. Өріс блоктың басын идентификациялайды. Өрісте нөльдерден тұратын байттан және бірлік жазылған он байттардан және бір нөльдерден тұратын байт бар;
- Идентификатор өрісі. Блок мекенжайы және режим байтынан тұратын дерек аты. 0-режим дерек өрісінің бос екендігін көрсетеді. 1-режимі түзету кодын пайдаланатындығын көрсетеді. 2-режим 2336 байт деректердің барын көрсетеді және түзету жоқ екендігін көрсетеді;
- Деректер өрісі. Пайдаланушылар деректері
- түзету кодтары (КК). Пайдаланушылардың қосымша деректерін 2 режимде сақтауға арналған, ал 1-режимде қате түзету 288 байт коды сақталады.

7.35-сурет. Блоктың типтік пішімі.

DVD оптикалық дискілері

Кейінгі жылдары компьютерлерде DVD дискілер (Digital Versatile Disk цифрлық әмбебап дискі) кеңінен қолданыс тауып отыр. CD дискіге қарағанда DVD сыйымдылығы айтарлықтай жоғары. Қолданылатын лазердің толқын ұзындығы 650 мкм. Осының арқасында DVD сыйымдылығы өсіп 682 Мбайттан 4,7 Гбайтқа дейін жетті.

DVD-де ПИТ пен ЛЕНД үшін бірінші қатпар үстінде екінші қатпар орнатылды. Екі қатпарлы DVD-де сәуле шағылыстырғыш қатпардың үстінен жартылай шағылыстырғыш екінші қатпар орналасқан. Фокустарды өзгерту арқасында дерек әр қатпардан жеке оқылады. Мұндай технология диск сыйымдылығын екі есеге - 8,5 Гбайтқа дейін арттыруға мүмкіндік берді. Екі қатпарлы дискі алтын түсті болады, ал бір қатпарлысы жинақ диск сияқты күміс түсті. DVD-ROM екі жақты болуы мүмкін.

BD оптикалық дискілер

Blue Ray (көгілдір сәуле) дискі немесе қысқаша BD кезекті оптикалық дискілер буынына жатады. Blue Ray технологиясында деректерді оқу және жазу үшін күлгін түсті толқын, ұзындығы 405нм-дік лазер қолданылады. Мұндай толқын ұзындығы жолшық енін екі есе қысқартуға алып келеді (0,32 микрон). Оқу жылдамдығы 430-435 Мбит/с-ке жоғарылады. Қазіргі уақытта диаметрлері 12 және 8 см-лік BD шығарылады. Екеуі де бір және екі қатпарлы, 12 сантиметрлік дискілер сыйымдылығы 27 Гбайт (бір қатпарлы вариант) немесе 54 Гбайт (екі қатпарлы диск). 8 сантимерлік диск сыйымдылығы - 7,8 Гбайт және 15,6 Гбайт.

Қазіргі кезде HVD (Holographic Versatile disk) голографиялық әмбебап дискі зерттеліп жатыр. Мұндай дискінің сыйымдылығы 3,9 Тбайт болады деп күтілуде. Жоспарланған деректерді беру жылдамдығы -1Гбайт/с.

7.13.3. Магниттік таспа негізінде құрылған СЖҚ

Магниттік таспа көлемі үлкен деректерді сақтауға арналған. Көбінесе олар қатты дискі деректерінің көшірмелерін сақтауға қолданылады. Жазу, оқу және сақтау принциптері магнитті дискіге ұқсас. Бір символға сәйкес келетін 9 бит таспаға орау бағытына перпендикуляр көлденең тізбекпен жазылады (сызықтық жазу). Әрбір биттік позициясына бір жазу/оқу сәйкес келеді, сондықтан барлық биттер параллель жазылады және оқылады. Символдың бір биті жұп сандарын бақылауға арналған.

Сызықтық жазудың екінші түрінде, жазу/оқу операциясы таспа бірінші бағытта жылжытылып орындалғаннан кейін, бастиектер бір позицияға жылжытылады. Одан соң таспа екінші бағытта жылжытылып, оқу/жазу

операциясы таспаның келесі жолшығымен орындалады. Мұндай сызықтық тәсілді *серпантиндік жазу тәсілі* деп атайды. Жазудың басқа түрінде (сызықты көлбеу) бірнеше жазу/оқу бастиектері таспаға көлбеу орналасқан барабанға бекітіледі (7.36-сурет).

Магнит таспаларын кассеталарда не картриждерде орналастыруы мүмкін. Кассетада магнит таспалары оралатын екі бобина болады. Картрижде бір магнит таспасы оралған бір бобина болады, ал қабылдау бобинасы картриж орналасатын жинақтағышта болады. Жады құрылғысының типіне байланысты таспа екі түрге бөлінеді - 0,38см-ден (0,15 дюйм), 1,27 см (0,5 дюйм). Таспа қалыңдығы -0,025мм.

7.36-сурет. Сызықты көлбеу жазу принципі: а) жазу принципі; б) барабанның таспаны қапсыруы

Жазу үстінде таспа баяу бір бағытта жылжып отырады, ал магниттік бастиектер үлкен жылдамдықпен айналады. Жылдам айналатын барабан мен баяу жылжитын таспа жазу тығыздығын өсіруге мүмкіндік береді. Таспалы жинағыштар әдетте жазылатын деректер сығу операциясын орындайтын электронды сұлбаларымен жабдықталады.

Қазіргі кезде таспа негізіндегі төрт түрлі технологияда құрылған жинақтағыштар кеңінен тараған. Оларға: AIT, SAIT, SDLT және LTO жатады.

7.2-кестеде аты аталған жинағыштарының параметрлері келтірілген

7.2-кесте. Таспалық жинағыштарының сипаттамалары.

Сипаттамалар	ATT - 3	SALT - 1	SDLT 320	LTO 2
Жазу тәсілі	Сызықты- көлбеу	Сызықты- көлбеу	Серпантинды	Серпантинды
Картриж түрі	2 бобина	1 бобина	1 бобина	1 бобина
Таспа ені	88 мм	12,5 мм	12,5 мм	12,5 мм
Таспа ұзындығы	230 м	600 м	600 м	600 м
Сыймдылығы	100 Гбайт	500 Гбайт	160 Гбайт	200 Гбайт
Сығудан кейінгі сыйымдылығы	260 Гбайт	1,3 Гбайт	320 Гбайт	400 Гбайт
Жүктеу уақыты	10с	23с	12с	15с
Орташа іздеу уақыты	27с	70с	70с	46с
Ең үлкен қайта орау уқыты	<150с	40с	140с	80с
Буфер сиымдылығы	18мб	72мб	64мб	64мб

Жинақтағыштың келесі модельдерінде сыйымдылық сипаттамасы төмендегідей: ALT-6-800 байт; SAIT-4-4 Тбайт; SDLT 2400-1,2 Тбайт; LTO 4-800 байт.

Бақылау сұрақтары

1. ЖҚ сыйымдылығы қандай өлшеммен анықталады?
2. ЖҚ қолданылатын қандай қатынас құру тәсілдерін білесіз?
3. Деректерді іріктеу уақыты мен ЖҚ қатынас құру циклінің айырмашылығы неде?
4. Негізгі жады қандай жады құрылғыларынан тұрады?
5. Тұрақты жады қандай түрлерге бөлінеді?
6. 2D және 2DM жады құрылғысының қандай ерекшеліктері бар?
7. 3D ЖҚ ерекшеліктері неде?
8. Блок түрінде құрылған негізгі жадының ұйымдастыру тәсілдері?
9. Мекен-жайлары кезектестірілген блокты жадының ерекшеліктері неде?
10. Статикалық және динамикалық ЖҚ қолдану аясын атаңыз?
11. FIFO жады қай жерде қолданылады?
12. СТЕК жады қандай тәртіппен жұмыс жасайды?
13. СТЕК көрсеткіштерінде қандай дерек сақталады?
14. Ассоциативтік ЖҚ-да маска не үшін қажет?
15. КЭШ-жады қандай қызмет атқарады?
16. Негізгі жадыны КЭШ-жадыда кескіндеудің қандай түрлері бар?
17. Ауани мекен-жайды физикалық мекен-жайға түрлендірудің қандай тәсілдері бар?
18. Жедел жадыны кілт арқылы қорғаудың артықшылығы неде?

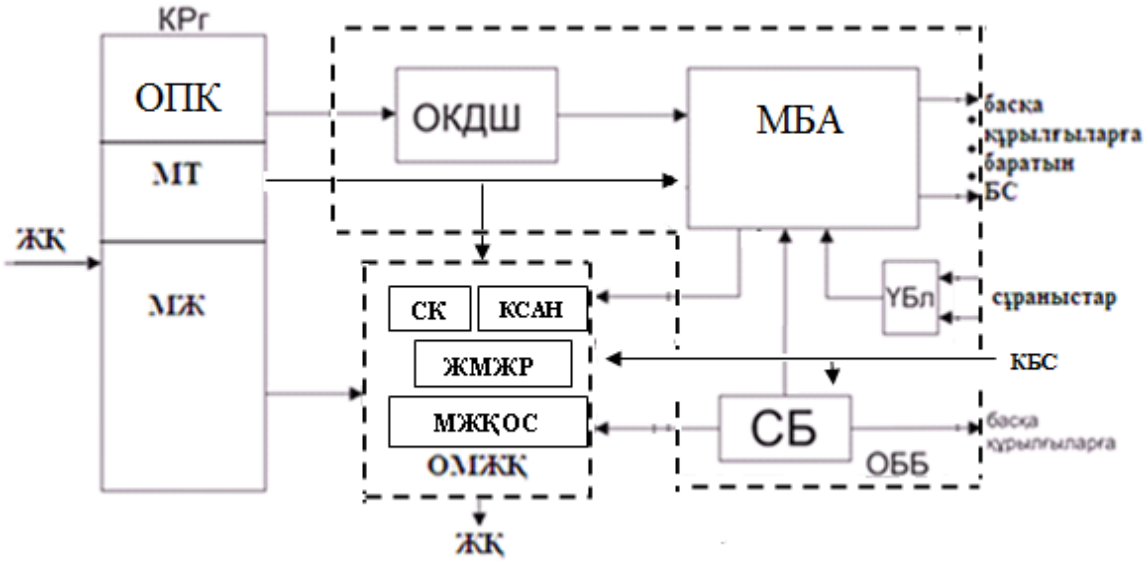
8. БАСҚАРУ ҚҰРЫЛҒЫЛАРЫ

8.1. Басқару құрылғыларының атқаратын қызметі және құрылымдық сұлбасы

Бағдарлама командалары автоматты түрде есептеу үдерісін басқару арқылы орындалады. Есептеу машиналарында бағдарламаны орындау үрдісі әр команданың тізбектеліп орындалатын машина циклімен анықталады. Машиналық циклді орындау кезеңдерінде басқару құрылғысы (БҚ) мынадай басқару қызметін атқарады(5.4):

- Команданы іріктеу және оны декодтау;
- Орындау мекен-жайын анықтау және операндтарды жадыдан оқу;
- Операцияны орындау;
- Келесі команда мекен-жайын анықтау.

Әрбір орындау кезеңдерінде машинаның әртүрлі түйіндерінде көптеген элементар әрекеттер орындалады. Бір тактта орындалатын элементар әрекеттерді микрооперациялар (МО) деп атаймыз. Бір уақытта орындалатын микрооперацияларды орындауға алып келетін басқару сигналдарының (БС) жиынтығын микрокоманда (МК) дейміз. Команда циклін орындайтын микрокомандалар тізбегін микробағдарлама (МП) деп атаймыз. БС басқару құрылғысының орталық бөлігі болып табылатын операцияларды басқару блогы (ОББ) арқылы шығарылады. Басқару құрылғысының жалпылама құрылымы 8.1-суретте келтірілген.



8.1-сурет. Басқару құрылғысының жалпылама құрылымы

БҚ жалпылама сұлбасынан көрініп тұрғандай құрылғы үш бөліктерден тұрады: орындалатын команда мекен-жай қалыптастырғышынан (ОМЖҚ), операцияларды басқару блогынан (ОББ) және команда регистрінен (КРг).

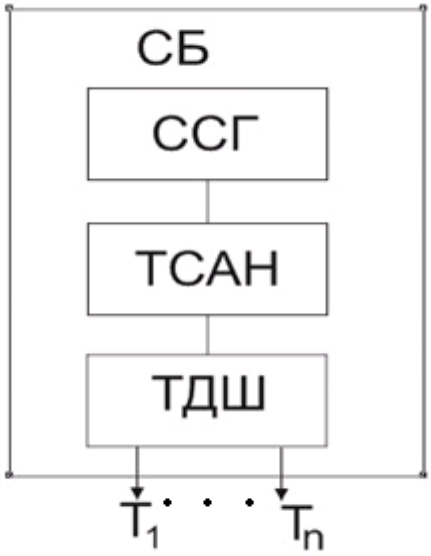
Команда регистрі кезекті команданы жады құрылғысынан қабылдап оны команда циклі аяқталғанша сақтайды. Оның операциялық бөлігінде операция коды (ОПК), мекендеу тәсілінің коды (МТ) және мекен-жай бөлігінде мекен-жай коды (МЖБ) сақталады.

Операцияларды басқару блогына (ОББ) операция кодының дешифраторы (ОКДШ), микробағдарламалық автомат (МБА), үзу блогы (ҮБл) және синхронизациялау блогы (СБ) жатады. Операция кодының дешифраторы операция түрін анықтайды.

МБА операция кодын декодтау нәтижесіне, мекендеу тәсілдерінің кодасына (МТ) байланысты және басқа құрылғыларынан келетін кері байланыс сигналдарына (КБС) байланысты микрокомандаларды (микробағдарламаны) орындауға керек,басқару сигналдарын (БС) шығарады.

Үзу блогы (ҮБл) машинаның ішінен немесе сырттан келетін сұратым сигналдарын қабылдайды, оларды өңдеп, керек болған жағдайда орындалып отырған команданы үзіп, басқа бағдарлама командаларды орындауға көшеді.

Синхронизациялау блогы (СБ) синхросигналдар генераторынан келетін сигналдарынан такт сигналдарын (Т) қалыптастырады. СБ құрамына синхросигналдар генераторы (ССГ), такт санауышы (ТСАН) және такт дешифраторы (ТДШ) кіреді. (8.2-сурет)



8.2-сурет.Синхронизациялау блогының құрамы.

Әрбір синхросигнал такт санауышының кірісіне берілгенде ол өз күйін 1-ден n-ге дейін өзгертеді. Санауыштың әр күйі такт дешифраторы арқылы $T_1 \dots \dots \dots T_n$ такт сигналдарын қалыптастырады.

Орындалатын команда мекен-жайын қалыптастыру блогы (ОМЖҚ) команданың мекен-жай бөлігінен және мекендеу тәсіліне (МТ) байланысты орындалатын операндтың мекен-жайын қалыптастырады және көшу командаларын орындағанда келесі команданың мекен-жайын анықтайды. Блоктың құрамына мекен-жай қосындылауышы (МЖҚОС), команда санауышы (КСАН), жады мекен-жай регистрі (ЖМЖР), стек көрсеткіші (СК) кіреді.

Мекен-жай қосындылауышы (МЖҚОС) арқылы мекендеу тәсілдеріне байланысты команда мекен-жайынан орындалатын мекен-жай алынады. Жады мекен-жайының регистрі (ЖМЖР) операндтардың орындалатын мекен-жайын сақтайды, ал команда санауышы - команда мекен-жайын қалыптастыру үшін және оны сақтау үшін қолданылады. ЖМЖР арқылы операндар іріктеледі, ал КСАН арқылы - жады құрылғысынан командалар оқылады. Стек көрсеткіші (СК) стек төбесінің мекен-жайын сақтайды. Ол арқылы стекпен амалдар орындалады.

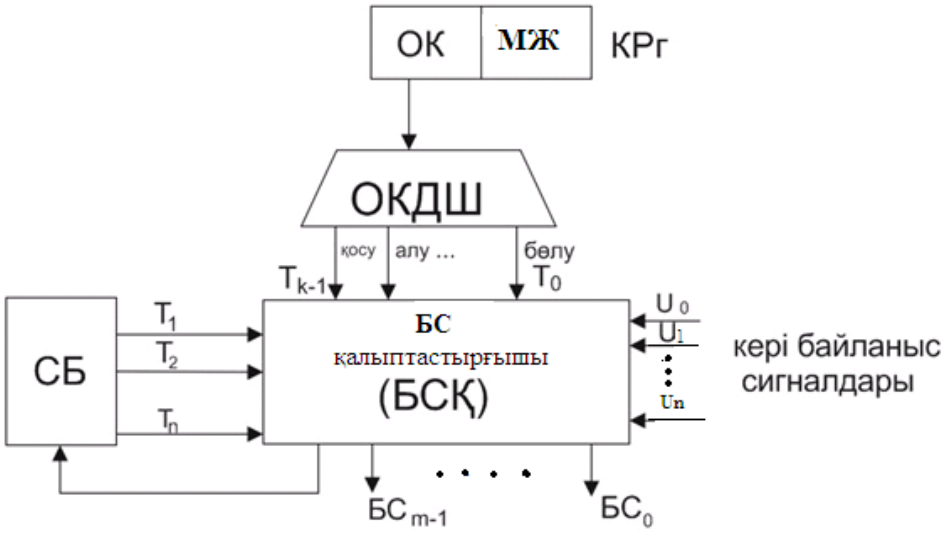
8.2.Микробағдарламалық автоматтар

Микробағдарламалық автоматтар (МБА) есептеу үрдісін басқаруға керекті барлық басқару сигналдарын қалыптастырады. МБА автомат кірісіне мынадай деректер беріледі:

- Операция коды. Ол арқылы МБА осы команданы орындауға керекті микрооперацияларды анықтайды;
- СБ шығарылатын такт сигналдары. Ол арқылы басқару сигналдарының шығу реті анықталады;
- Кері байланыс сигналдары. Оларға операциялық құрылғылардан, операциялар орындалып болған соң қалыптасатын белгі сигналдары, жады құрылғыларынан, үзу жүйесінен келетін сигналдар жатады.

МБА шығатын басқару сигналдары (БС) есептеу машинасының барлық құрылғыларына беріледі.

МБА жалпылама құрылысы 8.3-суретте келтірілген.



8.3-сурет. МБА жалпылама сұлбасы

МБА басқару сигналдарын қалыптастырғышын іске қосу тәсіліне байланысты олар екі түрге бөлінеді:

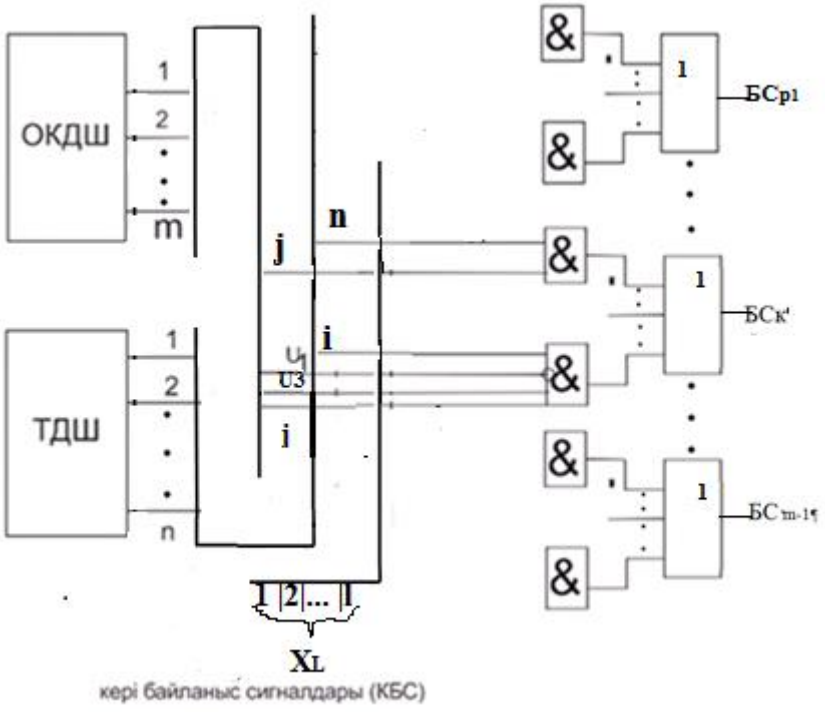
- Аппаратты немесе «қатаң» логикалы, МБА;
- Бағдарламаланған логикалы (жадыда сақталған логика) МБА.

Аппаратты логикалы МБА

Аппаратты логикалық МБА басқару сигналын қалыптастырғышы логикалық сұлбалар арқылы құрылады. Мұндай сұлбалар басқару сигналдарын орындалатын командалар түріне және кері байланыс сигналдарына байланысты қоздырады.

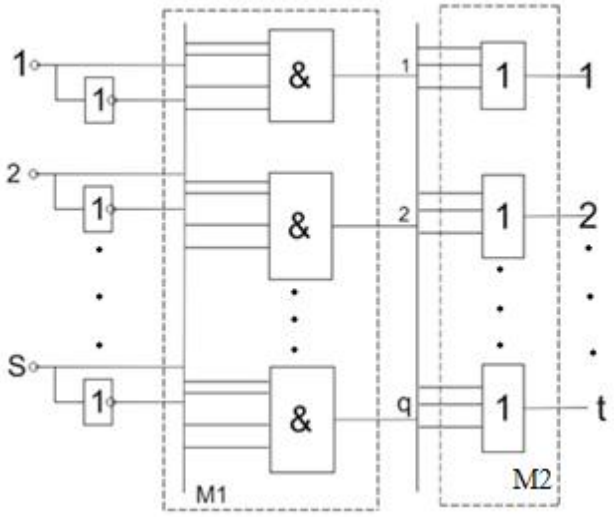
Басқару сигналдарын логикалық сұлбаларда құру принципін - 8.4 суретпен түсіндіруге болады. 8.4-суретте кері байланыс сигналына (КБС), орындалатын J-шы командаға байланысты n такт сигналының і-ші тактысында шығарылатын басқару сигналын ($БС_m$) қалыптастырғыш құрылғысының фрагменті көрсетілген.

Басқару сигналдарын қалыптастыру сұлбасын бағдарламаланатын логикалық матрицалар (БЛМ) арқылы да құруға болады. БЛМ құрамында жады элементтерінің болуы не болмауына байланысты олар: жадылы БЛМ және жадысыз БЛМ болып екіге бөлінеді.



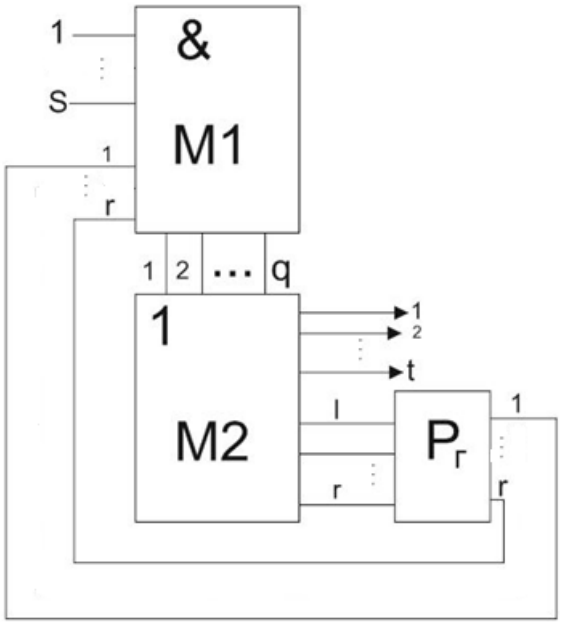
8.4-сурет. Басқару сигналдарын логикалық сұлбалар арқылы қалыптастыру құрылғысының фрагменті

Жадысыз БЛМ (8.5-сурет) екі матрицадан тұрады. Бірінші матрица (М1) S кірісі бар q ЖӘНЕ элементтерінен тұрса, екіншісі (М2) әрқайсысы q кірісті t НЕМЕСЕ элементтерінен тұрады



8.5-сурет. Жадысыз БЛМ

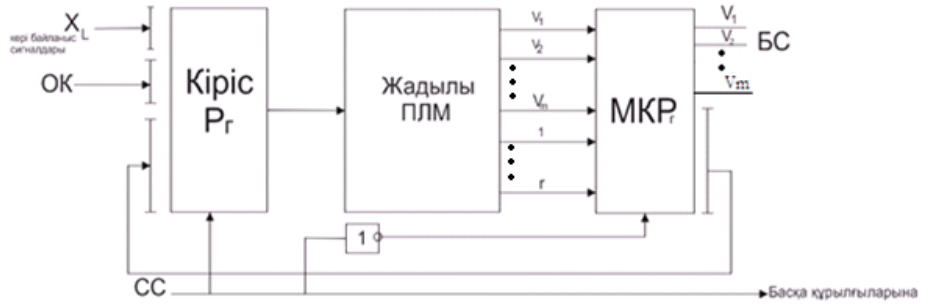
Құрамында жады бар БЛМ 8.6-суретте көрсетілген. Суретте көрініп тұрғандай жады ретінде r – разрядты регистр қолданылады.



8.6-сурет. Құрамында жады бар БЛМ

Жады ретінде r-разрядты регистр қолданылады. Регистрдің шығысы М1 матрицасының кірісіне кіріс сигналы ретінде беріледі. Сонда М1 матрицасының кіріс саны S+r санымен анықталады.

МБА жадылы БЛМ-де құрғанда БЛМ-нің кіріс регистріне операция коды, кері байланыс сигналдары және БЛМ жадының күй коды беріледі. Басқару сигналдары және БЛМ жады күйі бір регистрг е- микрокоманда регистріне (МКРг) біріктіріледі. Яғни автоматтың күйі МКРг бір бөлігінен КРг кірісіне беріледі. Жадылы БЛМ-де құрылған МБА 8.7-суретте көрсетілген.



8.7-сурет. БЛМ арқылы құрылған МБА

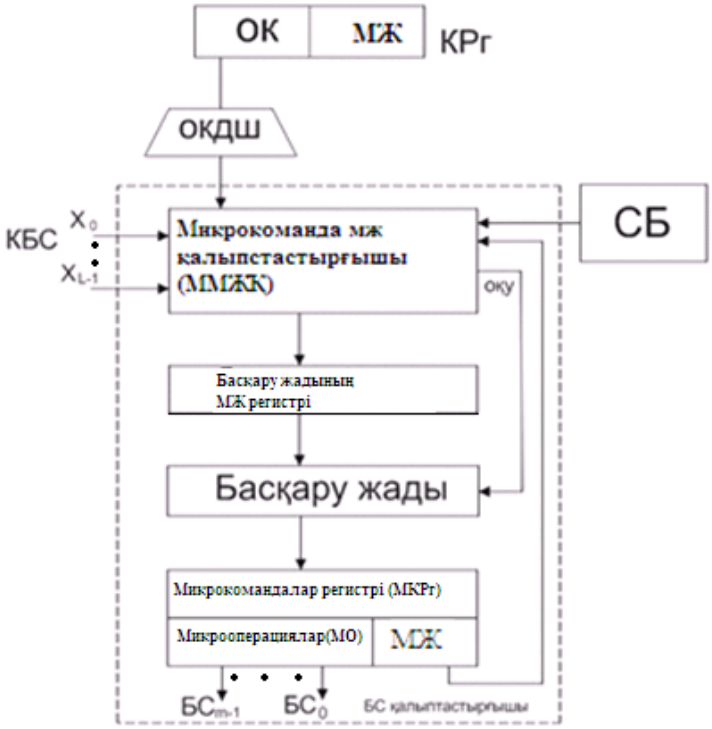
8.3 Бағдарламаланған логикалы МБА

Бағдарламаланған логикалы МБА құру жолын 50-жылдардың басында Британ ғалымы М.Уилкс ұсынған [39].

Жоғарыда айтылғандай әр команда өзінің микробағдарламасы арқылы орындалады. Микробағдарлама бір такт ішінде параллель орындалатын микрооперациялар жиынтығынан тұратын микрокомандалардан тұрады. Микрокомандалар саны команда түріне байланысты әртүрлі болады. Кейбір микрокомандалар тізбегі қайта орындалуы мүмкін. М.Уилкс осы микрокомандаларды жады құрылғысының ұяшықтарына жазып қоюды ұсынды.

Мұндай жадының ұяшық саны командаларға қатысатын микрокомандалар санымен анықталса, ал жады разряд саны микрооперациялар санымен анықталуы мүмкін (микрооперацияларды горизонталды тәсілмен кодаласақ).

Микробағдарламаның микрокомандалары іргелес мекен-жайларында жазылса, онда келесі микрокоманда мекен-жайы алдыңғы микрокоманда мекен-жайын бірге өсіру арқылы анықталады. Бағдарламаланған логикалы МБА жалпылама құрылымдық сұлбасы 8.8-суретте көрсетілген.



8.8-сурет. Бағдарламаланған логикалы МБА жалпылама сұлбасы

Кезекті микрокоманда мекен-жайын қалыптастыру

Командалардың микробағдарламаларын орындау үстінде микрокомандалардың орындалу тәртібі АЛҚ құрылғысының құрамында болатын белгілер регистрінің (БРг) күйімен анықталады. Мұндай жағдайда микрокоманданың кезекті мекен-жайы операция кодымен немесе микрокоманданың келесі мекен-жайымен, не өту мекен-жайымен анықталады.

МБА кезекті микрокомандасы мекен-жай қалыптастырғышынан (ММЖҚ), басқару жадысы мекен-жай регистрінен (БЖМЖР), басқару жадысынан (БЖ) және микрокомандалар регистрінен (МКР) тұрады. МЖҚ келесі микрокоманда мекен-жайын қалыптастырып оны БЖМЖР-ға жазады. Ол үшін орындалатын команда коды, СБ такт сигналдары, кері байланыс сигналдары (X_L) және МКР-інен келетін келесі микрокоманды мекен-жайын көрсететін мекен-жай (МЖ) қатынасады. Егер келесі микрокоманда шектес мекен-жайда орналасса, онда МЖҚ ағымды мекен-жайды бірге өсіру арқылы келесі микрокоманда мекен-жайын қалыптастырады. Егер келесі микрокоманда басқа мекен-жайда орналасса, онда келесі мекен-жай микрокоманда регистрінде көрсетілген МЖ-ге байланысты және X_L сигналдарына байланысты қалыптасады. МЖҚ-да келесі мекен-жайды анықтау МКР-нен оқылатын арнайы басқару сигналы арқылы тоқтатылады.

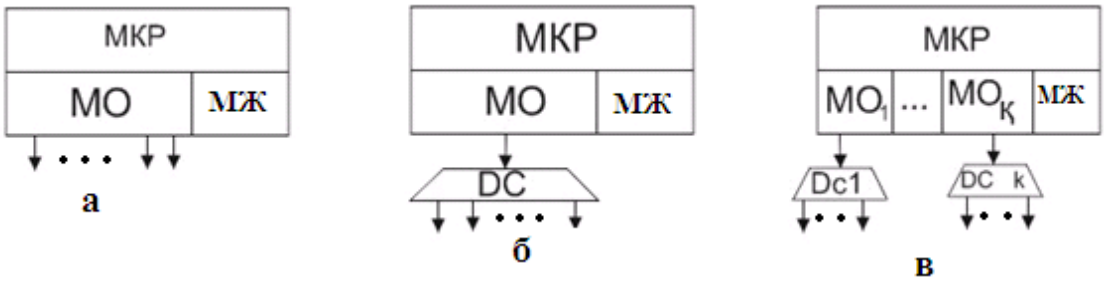
Басқару жадысынан оқылған микрокомандалар МКР-ге жазылады. Олар екі бөліктен - микрооперациялардан (МО) және мекен-жай бөлігінен тұрады. Мекен-жай бөлігі МЖҚ-ға беріледі, МО бөлігінен шығатын басқару сигналдары $БС_{m-1}, \ldots, БС_0$ ЕМ қалған құрылғыларына беріледі. Басқару жадының разрядтылығы МО санымен және жады ұяшықтар санымен анықталады. МО саны есептеу машиналарында өте көп болады. Сондықтан БЖ разрядтылығын қысқарту үшін оларды әртүрлі жолмен кодтайды. Олардың ішіндегі ең көп қолданатын түрлеріне жоғарыда каралған: тік кодтау, көлденең және аралас кодтаулар жатады.

Көлденең кодтау тәсілінде (8.9, а-сурет) микрокоманданың МО бөлігіндегі әр разряд микрооперацияның біріне сәйкес келеді. Егер басқару сигналының барлық саны m болса, онда МК операциялық бөлігінің разрядтылығы m болады. Егер m_i=1 болса, онда оған сәйкес БСm сыртқа шығарылып тиісті түйінге беріледі. Көлденең кодтаудың кемтістігі - МО ұзындығында. Мұндай жағдайда басқару жадысының разряд саны үлкен болуы мүмкін.

Микрооперацияларды тік кодтау тәсілінде (8.9, б-сурет) әрбір микрооперацияға бірегей код меншіктеледі, микрокоманданың микрооперациясында көрсетіледі. Бірегей кодтың разрядтылығы (МО ұзындығы) $\log_2 m$ өрнегімен анықталатын бүтін сан. Бұл басқару жадысының сыйымдылығына (разряд санының азаюына) әсер етеді. Бірақ бұл тәсілде бірегей кодтарда түрлендіріп, олардан басқару сигналдарын қалыптастыру үшін дешифратор керек. Оның үстіне микрооперацияларды тігінен кодтағанда МК-да тек бір басқару сигналын ғана көрсете аламыз. Егер көптеген

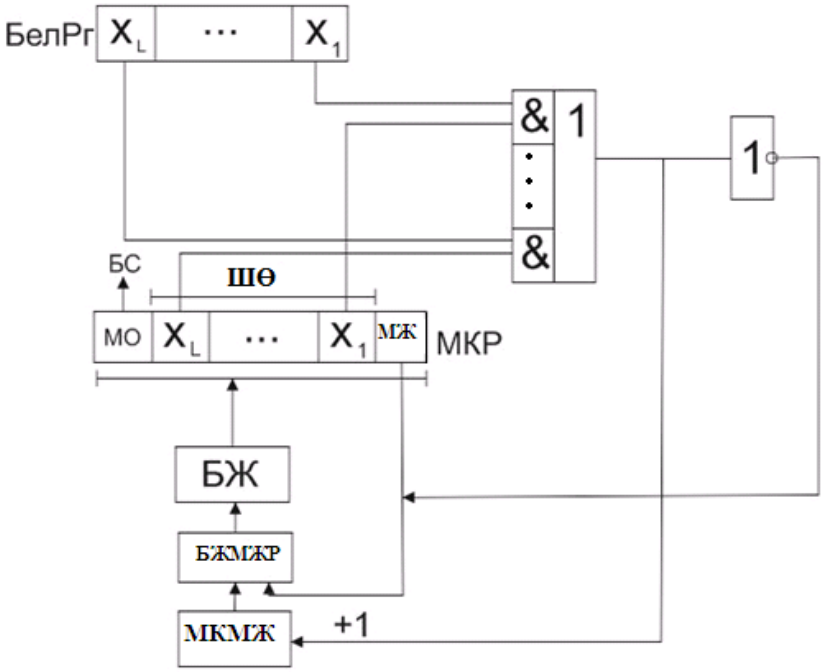
микрооперацияларды қатар орындау керек болса, онда бірнеше такт жұмсап оларды басқару жадысынан оқимыз. Бұл микробағдарламаның ұзындығының өсуіне алып келеді, яғни оны орындау уақыты ұзарады.

Микробағдарламаларды аралас (8.9, в-сурет) кодтауда әртүрлі m басқару сигналдары К топқа бөлінеді. Әр топтың шығысынан DC,…,DCk дешифратор арқылы басқару сигналдары қалыптасады, яғни микрооперациялар ішінен бір тактта k басқару сигналдары белсенділікке ие бола алады.



8.9-сурет. Микрооперацияларды кодтау тәсілдері: а-көлденең, б-тік, в-аралас

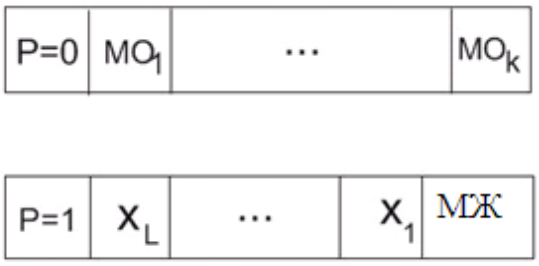
Микрокомандаңың мекен-жай коды операция кодымен әрбір командалық циклде тек бір рет анықталады. Ол операция кодын осы операция микробағдарламасының бірінші микрокоманда мекен-жайына түрлендіретін операция дешифраторы анықтайды. Микробағдарламаның кезекті мекен-жайын қалыптастыратын келесі тәсілінде әрбір микрокоманда құрамына келесі микрокоманда мекен-жайы көрсетіледі (мәжбүрлі мекендеу) немесе ағымды микрокоманда мекен-жайын автоматты түрде бірге өсіру (табиғи мекендеу) арқылы анықталады. Соңғы екі тәсілде де мекендеу белгілер регистрінде БРг сақталатын сигналдар тексеріледі. Қандай шарттарды орындау керектігін тексеру үшін микрокоманда құрамында шарт өрісі (ШӨ) енгізіледі. Мұнда әр белгілерге өз разрядтары сәйкестендіріледі. ШӨ микробағдарламаның кезекті микрокомандасының мекен-жайын анықтауға қай белгінің күйін талдау керектігін анықтайды. Егер ШӨ барлық разрядтары 0-ге тең болса, онда ешбір белгілер тексерілмейді, ал келесі МК мекен-жайы ағымды микрокомандаңың мекен-жай бөлігімен анықталады. Егер ШӨ і-ші разрядында «1» жазылса, онда белгі регистрінің і-разряды тексеріледі. Егер ол разряд бір күйінде болса келесі МК мекен-жайы ағымды микрокоманда мекен-жайын бірге өсіру арқылы анықталады. Мәжбүрлік мекендеу арқылы келесі микрокоманда мекен-жайын қалыптастыру сұлбасы 8.10-суретте көрсетілген.



8.10-сурет. Мәжбүрлеу арқылы келесі МЖ-ны қалыптастыру

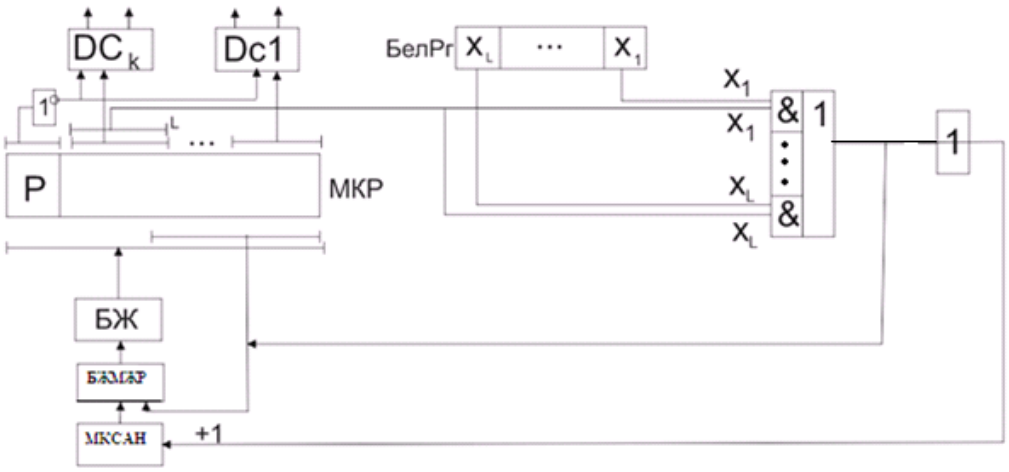
Суретте МКР шарт өрісінің (ШӨ) әрбір разряды белгілер регистрінің (БелРг) разрядтарымен салыстырылады. ЖӘНЕ-НЕМЕСЕ сұлбасының шығысында ноль қалыптасса, онда микрокоманда регистрінен А мекен-жайы БЖМЖР-іне жазылады. Егер тексерілген шарттардың нәтижесінде ЖӘНЕ-НЕМЕСЕ сұлбасының шығысында бір қалыптасса, онда мекен-жай санауышына «1» қосылып ол БЖМЖР-іне беріледі.

Табиғи мекендеу тәсілінде әрбір микрокоманданың құрамында мекен-жай бөлігін көрсету шарт емес. Табиғи мекендеуде микрокоманданың екі типі: операциялық және басқару түрлері болады. МК типін оның жоғарғы разряды (Р) көрсетеді. Егер Р=0 болса, онда оқылған микрокоманда - операциялық МК болады, ал Р=1 болса, онда ол басқару МК болып табылады. (8.11-сурет)



8.11-сурет. Операциялық және басқару микрокомандалары.

Табиғи мекендеу арқылы құрылған МБА 8.12-суретте келтірілген.



8.12-сурет. Табиғи мекендеу арқылы құрылған МБА

Р=0 мәнінде БЖ-дан оқылған микрокоманда операциялық болып табылады. Басқару сигналдары DC,…,DCk арқылы басқа құрылғыға беріледі. Келесі мекен-жай (МКМЖСАН) ағынды мекен-жайды +1-ге өсіру арқылы анықталады.

Р=1 болғанда БЖ басқару МК-ы оқылады. ЖӘНЕ-НЕМЕСЕ сұлбасы арқылы X_i сигналдары тексеріледі, сұлба шығысында «1» болса, онда БЖМЖ-ға $МКР_г$ мекен-жай бөлігі жазылады, сұлба шығысы «0» болса, келесі мекен-жай МКСАН бірге өсіру арқылы қалыптасады.

8.4 Бағдарламаны үзу жүйесі

Есептеу машинасы ағымды бағдарламаны орындап жатқан кезде оның ішкі құрылғыларында немесе онымен байланысқан сырт құрылғыларында машинаның жедел «назар» аударуын керек ететін әртүрлі оқиғалар болады.

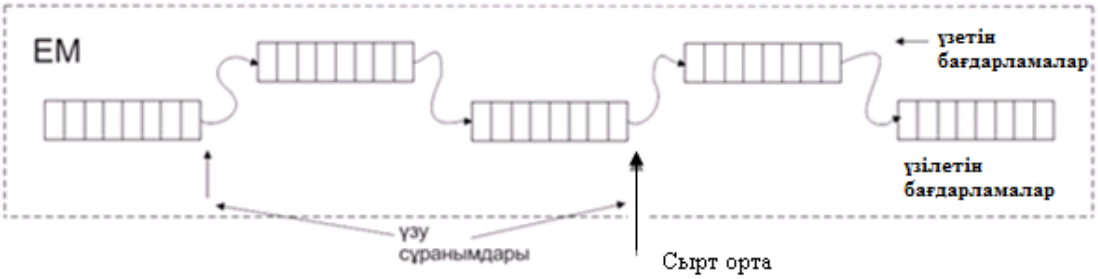
Назар аудару кезінде машина ағымды бағдарламаны үзіп, сол оқиғаға байланысты болатын басқа бағдарламаны орындауға өтеді. Бағдарламаны орындап болған соң үзілген бағдарламаға қайта оралады.

Қаралған үдерісті бағдарламаны *үзу үдерісі* деп атаймыз. Бағдарламаны үзу үдерісі 8.13-суретте көрсетілген.

Үзуді керек ететін әрбір оқиға машинаға сұратым деп аталатын үзу сигналдарын жібереді. Үзу сұратымымен орындалатын бағдарламаны үзетін бағдарлама деп атаймыз.

Үзу сұратымдары екіге бөлінеді: ішкі және сыртқы сұратымдар. Сыртқы сұратымдар көзіне машинамен байланысқан сырт жады құрылғылары, басқа машиналар, не процессорлар немесе басқа объекттер жатады. Ішкі үзу

сұратымдарына машина құрылғыларының ішіндегі бақылау сұлбаларынан келетін сигналдар жатады. Мысалдар: 0-ге бөлу, арифметикалық аса толу, қате команда т.с.с.



8.13-сурет. Бағдарламарды үзу

Сұратым бойынша бір бағдарламаны үзіп екіншісіне өту үшін және үзілген бағдарламаға қайта оралу үшін әртүрлі аппараттық және бағдарламалық жабдықтар керек. Мұндай жабдықтар жиынын бағдарламаны үзу жүйесі деп аталады.

Үзу жүйесінің негізгі міндеттеріне мыналар жатады:

- Сұраным сигналдарының ішінен бірінші кезекте өңделетін сұратымды анықтау;
- Үзілетін бағдарламаның күйін сақтап, үзетін бағдарламаны орындауға өту;
- Үзілген бағдарламаның күйін қалпына келтіру және оған өту.

Сұратым көздерінен үзу сигналдары бір уақытта келуі мүмкін. Олардың ішінен біреуін таңдап алу керек. Сондықтан, әртүрлі сұратымдарға артықшылық нөмері немесе басымдылығы белгіленуі керек. Басымдылығы жоғары сұратым басқа сұратымдарға қарағанда бірінші кезекте өңделуі керек.

Үзу жүйесі оның кірісіне келіп түскен әрбір сұратымға назар аударуы керек. Бірақ та, егер процессор орындалып жатқан бағдарламаның басымдылығы жаңадан келген сұратым басымдылығынан жоғары болса, онда процессор оған назар аудармауы керек. Сондықтан көптеген ЕМ үзу сұратымының бүркемеленбейтін және бүркемеленетін түрлері болады.

Бүркелеменбейтін үзу сұратымдары бағдарлама арқылы басқарылмайды. Мұндай сұратымдар жүйеге жеке шиналар арқылы беріледі, олардың басымдылығы өте жоғары болады. Мұндай сұратымдар қорек көздерінен немесе бақылау сұлбаларынан т.с.с түседі.

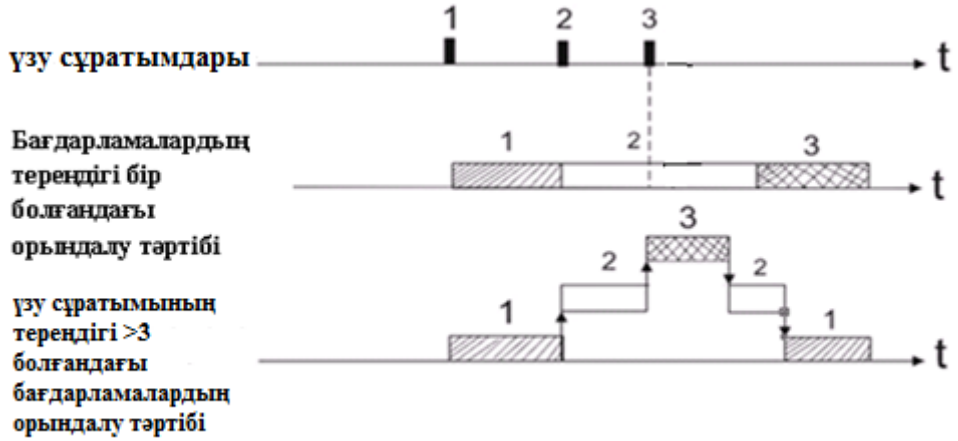
Бүркемеленетін үзу сұратымдары бағдарлама командаларымен басқарылады. Сол арқылы бағдарламашы есептеу үрдісін тиімді түрде басқара алады. Орындалып жатқан бағдарламаны үзу не үзбеу туралы дерек үзу бүркемесімен анықталады. Үзу бүркемесі дегеніміз – n разрядты екілік сөз. Егер бүркеменің і-разрядында 0 жазылса, онда і-сұратым орындалып жатқан бағдарламаны үзе алмайды, ал і=1 болса, онда бағдарламаны сұратым үзе алады.

Үзу жүйесінің сипаттамалары.

Үзу жүйесі жұмысының тиімділігі төмендегі сипаттамалар арқылы анықталады:

- Үзу сұранымдар саны;
- Реакция уақыты (t_p) – үзу сұранымының пайда болуы мен үзетін бағдарламаның басталу аралығындағы уақыт;
- Бір бағдарламадан екіншісіне өтуге жұмсалатын уақыт. Ол екі уақыт бөлігінен тұрады: t_c - үзілген бағдарламаның күйін сақтау уақытынан және $t_{қо}$- үзілген бағдарламаға қайта оралуға кететін уақыттан тұрады;
- Үзу тереңдігі - бірін-бірі үзе алатын бағдарлама саны. Егер үзетін бағдарламаға өткеннен соң ол аяқталғанша басқа сұранымдарға тиым салынса, онда жүйенің үзу тереңдігі 1 болады. Үзу тереңдігі n болған кезде n бағдарлама бірін-бірі тізбектеп үзе алады.

8.14-суретте тереңдігі әртүрлі үзу жүйесі келтірілген.



8.14-сурет. Тереңдігі әртүрлі үзу жүйесі

Үзу жүйесінің қанығуы. Егер сұратымға осы сұратым көзінен келген жаңа сұратымға дейін қызмет көрсетілмесе, онда үзу жүйесі қанығу күйіне жетеді.

Мұндай жағдайда бұрынғы келген сұратымды машина жоғалтады. Оған жол бермеу үшін үзу жүйесінің сипаттамасы, машинаның жылдамдығы, үзу көзінің саны мен олардан келетін сұратымдар жиілігін қанығу күйі орын алмайтындай етіп бір-біріне сәйкестендіру керек.

Бағдарламаны үзудің рұқсат етілген моменттері. Көбінесе үзуге ағымдағы кез келген команда аяқталған соң рұқсат етіледі. Мұндай жағдайда үзу жүйесінің реакция уақыты t_p бір команданың орындалу уақытымен анықталады. Нақты уақыт масштабында жұмыс жасайтын машиналар үшін бұл уақыт өте үлкен болуы мүмкін. Сондықтан мұндай машиналарда үзу команданың кез келген такты аяқталған соң рұқсат етіледі. Алайда мұндай жағдайда үзілген бағдарламаға қатысты сақталатын деректер саны күрт көбейеді.

Үзу жүйесіндегі сұратым көздері

Сұраным көздері шолу сұлбалары арқылы немесе векторлық үзу арқылы айқындалады. Циклдік шолу аппарат құрамы жағынан қарапайым, бірақ реакция уақыты t_p сұраным көздері көп болғанда t_p өте үлкен болады. Сондықтан қазіргі есептеу машиналарында көбінесе векторлық үзу жүйесі кеңінен қолданылады.

Векторлық үзу жүйесінде үзу көзі процессорға сұратым сигналымен қатар машина жадында үзу векторы туралы деректі бірге жібереді. Үзу векторында үзетін бағдарламаның мекен-жай көрсетіледі. Осы мекен-жай арқылы үзу бағдарламасы орындала бастайды.

Әрбір сұраным көздерінің үзу векторлары негізгі жады құрылғысының бекітілген мекен-жайларында сақталады. Сондықтан үзу векторы сұратым нөмерінің кодымен (СНК) анықталады.

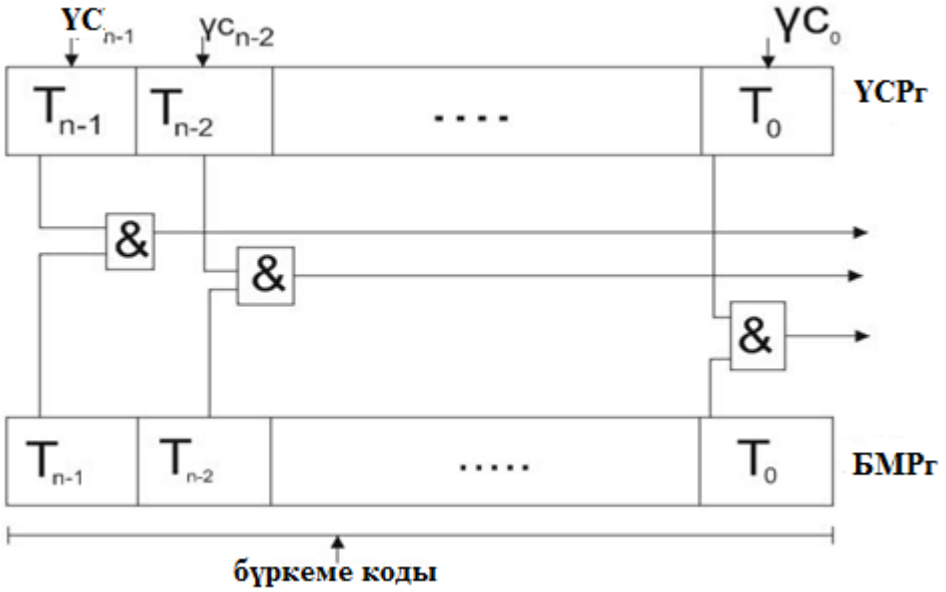
Сұратым векторлары негізгі жадының бірнеше ұяшықтарында сақталып, үзу векторының кестесін құрады. Кесте әдетте негізгі жадының 0-мекен-жайынан бастап орналасады.

Векторлық үзу контроллерінің құрылымы

Жоғарыда көрсетілген үзу жүйесінің міндеттерін векторлық үзу арқылы орындайтын векторлық үзу контроллерінің жұмысына тоқтайық (8.17-сурет).

Контроллер бірнеше блоктан тұрады. Бірінші блоктың міндеті - үзу сұратымдарының ішінен бүркеленбеген сұратымды іріктеу. Ол үшін блоктың құрамында үзу сұратымдарын қабылдап, оны сақтайтын үзу сұранымдарының

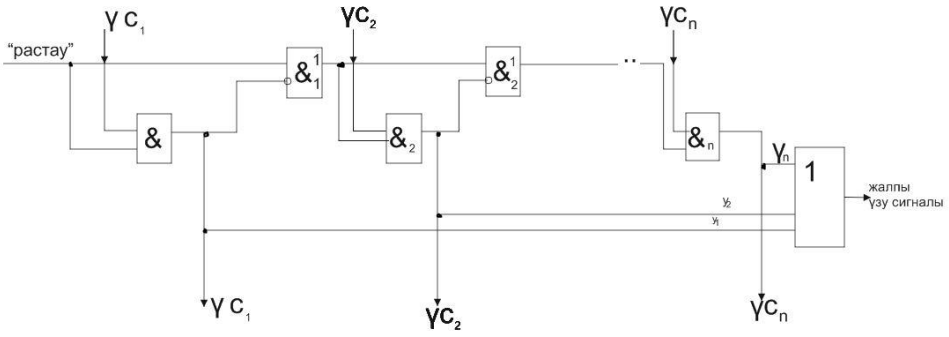
регистрі (ҮСРг), бүркеме регистрі (БРг) және логикалық сұлбалар керек. Регистрлердің разряд саны сұраным сандарымен анықталады (8.15-сурет).



8.15-сурет. Үзу сигналдарын бүркememен іріктеу

Бүркеме регистріне бүркеме коды бағдарламалық жолмен жүктеледі. Бүркеме регистрінің і-разряды 1 күйінде тұрса, ол ҮСі ары қарай өңдеуге рұқсат етеді, ал 0 күйінде тұрса, онда ҮСі өңдеуге рұқсат бермейді (бүркемеленеді).

Екінші блок арқылы бүркемеленбеген үзу сұратымдарының ішінен басымдылығы жоғары сұратым анықталады. Оны бір тактілі тізбекте сұлба («дейзи-тізбегі») арқылы анықталады. Мұндай тізбекті сұлба 8.16-суретте көрсетілген.



8.16-сурет. Біртактілі тізбекті сұлба

Сұратымдар басымдылығын анықтау процессордан келетін «растау» (РСТ) сигналынан басталады. РСТ сигналы тізбектеп қосылған $\&_i$ элементтеріне беріледі. Егер үзу сұратымдары жоқ болса, онда ол барлық & элементтерінен өтіп жалпы үзу сигналын қалыптастырмайды. Егер келіп түскен

сұратымдарының ішінде і-ші сұратымның басымдылығы жоғары болса, (YCi=1, $\bigvee_{j=1}^{i-1} YC = 0$) онда РСТ сигналының $\&_i$ элементінің оң шығысына берілуі тоқтатылады. Тізбекті сұлбаны і- шығысында y_i=1, ал қалған шығыстарында-0.

Контроллердің үшінші блогында басымдылығы жоғары үзу сұратымның нөмірі (үзу векторының сұратым нөмірінің коды (СНК)) шифратор арқылы қалыптасады.

Контроллердің төртінші блогында СНК коды үзу табалдырығы кодымен салыстырады. Егер СНК табалдырық кодынан жоғары болса, онда салыстыру сұлбасы үзуге рұқсат беру сигналын (УРС) шығарып СНК арқылы процессор үзетін бағдарлама жұмысын бастайды. Үзу табалдырығы коды бағдарлама командасымен анықталып табалдырық регистріне жазылады.

8.17-сурет. Векторлық үзу контроллерінің блок-сұлбасы

Ағымды бағдарламаны үзгенде үзілген бағдарламаның күйін - бағдарлама контекстін сақтау керек. Бағдарлама контекстінің құрамына команда санауышында сақталатын мекен-жай командасы (осы мекен-жай арқылы үзілген бағдарлама жұмысын қайта бастайды), аккумулятордағы (Акк) және белгілер регистріндегі деректер де кіреді. Бағдарлама контекстіндегі деректер аппараттық стекте сақталады. Мұндай стекте бірінші жазылған бағдарлама контексті соңынан немесе соңғы жазылған бағдарлама контекстісі бірінші болып оқылады.

Бақылау сұрақтары

1. Басқару құрылғысының негізгі міндеті неде?
2. Басқару құрылғысы қандай блоктардан тұрады?
3. Микробағдарламалық автомат (МБА) қандай блоктардан тұрады?
4. Бағдарламаланған логикалық МБА негізгі құрылғыларын атаңыз?
5. Микрооперацияларды кодтау тәсілдері қалай аталады?
6. Үзу жүйесінің міндеттеріне не жатады?

7. Үзу тереңдігі деген не?
8. “Дейзи” тізбегі не үшін керек?
9. Векторлық үзу контроллері қандай блоктардан тұрады?
10. Бүркеме регистрі не үшін керек?

8. ШИНАЛАРДЫ ҰЙЫМДАСТЫРУ

Шина деп есептеу машиналарының әртүрлі құрылғыларын бір-бірімен байланыстыратын арнажол жиынтығын айтамыз. Оларды атқаратын қызметтеріне байланысты ішкі және сыртқы шиналар деп екіге бөлуге болады.

Ішкі шиналар арқылы деректермен процессордың ішкі құрылғылары алмасады немесе олар арқылы деректер процессордан тыс басқа құрылғылармен алмасады.

Сырт шиналары процессордың жады құрылғыларымен немесе енгізу-шығару құрылғыларымен байланыс құруға арналған.

Әртүрлі шиналардың атқаратын қызметтері әртүрлі, сондықтан оларға қойылатын талаптар да әртүрлі. Әрбір шина олардың сигналдық желі жиынтығымен, физикалық, механикалық және электрлік қажеттерімен, шиналарға төрелік қызмет ететін, олардың күйін білдіретін, басқару және уақыт үйлесімдіру тәсілдерімен және алмасу тәсілдерімен (шина хаттамаларымен) ерекшеленеді.

Шиналар байланыс желілер жиынтығы арқылы құрылады. Байланыс желісі – бұл сигналдарды (көбінесе 0 немесе 1 екілік цифрлары) беретін физикалық орта. Олар арқылы сигналдар тізбекті немесе параллель беріледі. Физикалық түрде желі сым немесе өткізгіш материалдары арқылы құрылған желілер, жолақтар, оптоталшықтар. Желіні радио және инфрақызыл арналары да құра алады.

Шиналар арқылы орындалатын операцияларды транзакциялар деп атайды. Транзакциялар екі түрге бөлінеді: оқу транзакциясы және жазу транзакциясы. Егер алмасу операциясына енгізу-шығару құрылғылары қатынасса, онда енгізу немесе шығару транзакциялары орын алады. Шиналық транзакция екі құрамнан тұрады. Оларға мекен-жай кодын және деректі беру (немесе қабылдау) жатады.

Егер екі құрылғы шина арқылы ақпаратпен алмасатын болса, онда екеуінің бірі алмасу операциясын бастап оны басқаруы керек. Мұндай құрылғыны *басқарушы* немесе бастаушы (bus master) деп атаймыз. Компьютерлік жүйелерде басқарушы ретінде шинамен дерек беруді ұйымдастыра алатын кез келген құрылғы бола алады. Дерек беруді қолға алмаған құрылғыларды *орындаушы* құрылғылар деп атаймыз (bus slave).

9.1 Шиналардың төрелік механизмі

Шинаға оны басқаруға мүмкіндігі бар бірнеше құрылғылар қосылуы мүмкін. Егер осылардың ішінен бір уақытта шинаны басқаруға екі немесе одан көп құрылғылардан сұраныс түсірсе, онда олардың арасында қақтығыс орын

алады. Мұндай жағдайда шинаны басқаруға үміткер құрылғыларының ішінен біреуін анықтау керек, өйткені шинаға бір уақыт мезетінде ақпарат тек бір құрылғыдан берілуі керек, кері жағдайда берілген ақпарат бүлінеді, немесе қате орындалады. Мұндай қақтығысты шешу төрелік қызметін атқаратын механизмге жүктеледі. Шинаның төрелік механизмі басқарушы міндетін атқарушылыққа үміткер құрылғыларына меншіктелген артықшылық (приоритет) құқығына негізделеді. Шинаны басқаруға үміткер құрылғыларының ішінен қайсысының меншіктелген артықшылық деңгейі жоғары болса, сол құрылғы басқарушы міндетіне ие болады.

Төрелік алгоритмдері

Әрбір болашақ басқарушыға белгілі бір артықшылық деңгейін меншіктейді. Мұндай артықшылық деңгейі тұрақты түрде меншіктелсе, онда оны *статикалық немесе бекітілген артықшылық* деп атаймыз. Меншіктелген артықшылық белгілі бір алгоритммен өзгертіліп отырса, онда *динамикалық артықшылық* орын алады. Барлық құрылғылар артықшылықтары деңгейі бірдей болуы мүмкін.

Статикалық артықшылықпен жұмыс жасайтын жүйеде артықшылығы жоғары құрылғылар басқа құрылғылардың шинаға қол жеткізуге тосқауыл қоюы мүмкін. Бұл аталған жүйенің кемістігіне жатады.

Динамикалық артықшылықты жүйеде әрбір құрылғы шинаны басқару мүмкіндігіне ертелі-кеш ие бола алады. Артықшылықты динамикалық өзгерту әртүрлі алгоритмдер арқылы жүргізіледі. Олардың ең көп тараған түрлеріне:

- артықшылықтың қарапайым циклдік өзгертілуі;
- артықшылықты соңғы сұранымды ескере отырып, циклдік өзгертілуі;
- артықшылықты кездейсоқ заңдылығымен өзгертілуі;
- артықшылық деңгейін құрылғылардың шинаға қатынас құру уақыт аралығын еске ала өзгертілуі.

Артықшылықты қарапайым циклдік жолмен өзгертуде әр құрылғы бірегей артықшылық деңгейіне ие. Әрбір арбитраж циклінен кейін құрылғылардың артықшылық деңгейлері циклдік түрде бір деңгейге төмен түсіріледі. Одан соң ең артықшылығы төмен құрылғы ең жоғарғы артықшылыққа ие болады.

Артықшылықты соңғы сұранымды ескере отырып, циклдік өзгерту алгоритмінде шинаның төрелік қызметі шинаға сақина тәрізді құрылғы барлық құрылғылармен үлестірілген. Кезекті сұранымды өңдеп болған соң соңғы қызмет көрсетілген құрылғыға төменгі артықшылық меншіктеледі де, оған шинаның келесі цикліне төрелік ету жүктеледі.

Қалған құрылғыларға артықшылық деңгейі төрелік етуші құрылғының оларға сақина бойынша орналасу қашықтығына байланысты меншіктеледі. Төрелік етуші құрылғыға ең жақын құрылғыға деңгейі жоғары артықшылық меншіктелсе, одан кейін орналасқан құрылғыға келесі артықшылық деңгейі меншіктеледі, т.с.с. Жоғарыда қаралған екі алгоритмде де әрбір құрылғының басқарушы болуына мүмкіндігі бар.

Артықшылықты кездейсоқ заңдылығымен өзгерту тәсілінде төреліктің кезекті циклінен кейін жалған сандар генераторы арқылы әрбір құрылғыларға кездейсоқ артықшылық деңгей меншіктеледі.

Ең соңғы алгоритм бойынша артықшылық деңгейін құрылғылардың шинаға қатынас құру уақыты аралығына сай меншіктейді. Қай құрылғының соңғы қатынас құру уақыт аралығы үлкен болса, сол құрылғыға үлкен деңгейлі артықшылық белгіленеді.

Төрелік алгоритмінің барлық құрылғыларының артықшылық деңгейі бірдей болғанда әрбір құрылғыға циклдік түрде кезекпен белгілі уақыт мөлшері белгіленеді. Сол уақыт аралығында құрылғы шинаны басқара алады. Егер құрылғы осы уақыт аралығында шинаны керек етпесе, онда бөлінген уақыт аралығы пайдаланылмайды.

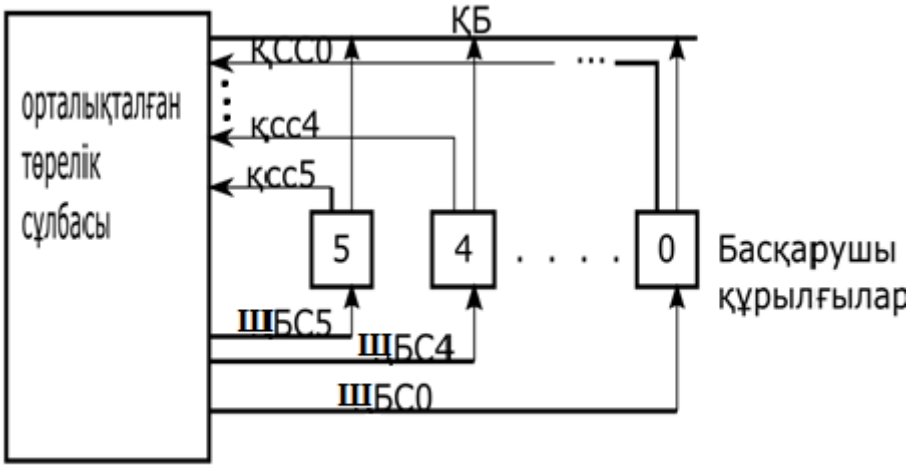
Төрелік сұлбалары.

Төрелік сұлбалары орталықтандырылған немесе үлестірілген (орталықтандырылмаған) деп екі түрге бөлінеді.

Орталықталған төрелік сұлбалары жүйедегі барлық сұраным жіберген басқарушы құрылғылардың ішінен тек біреуіне шинаға қатынас құру мүмкіндігін береді. Мұндай құрылғыны шинаның орталық контроллері деп те атайды. Ол жеке модуль немесе процессордың бір бөлігі болуы мүмкін. Басқарушы құрылғылардың орталық төрелік сұлбасына қосылуына байланысты орталықтандырылған төрелік сұлбасы екі түрге бөлінеді: параллель және тізбекті. Төрелік сұлбасының параллель нұсқасында әрбір басқарушы бола алатын құрылғы орталық төрелік сұлбалармен екі желіден тұратын арнажолмен байланысты болады. Сұлбаның мұндай нұсқасында орталық төрелік сұлбаға сұранымдар бір-бірінен тәуелсіз параллель түрде беріледі, сондықтан мұндай сұлбаны орталықтандырылған параллель төрелік сұлбасы деп атайды.

Әрбір шинаны басқаруға үміткер құрылғылар орталықтандырылған төрелік сұлбасына сұратым сигналдарын (ҚСС) кез келген уақытта жеке желі арқылы түсіреді. Төрелік сұлбасы арқылы таңдалған і-құрылғысына жеке желі арқылы шинаны басқару сигналы (ШБС) беріледі. Жаңа і-басқарушы құрылғы шинаның бос екендігіне (сигнал ҚБ) көз жеткізу керек. Егер шина бос болса, онда жаңа басқарушы құрылғы шинаны басқаруға кірісе алады.

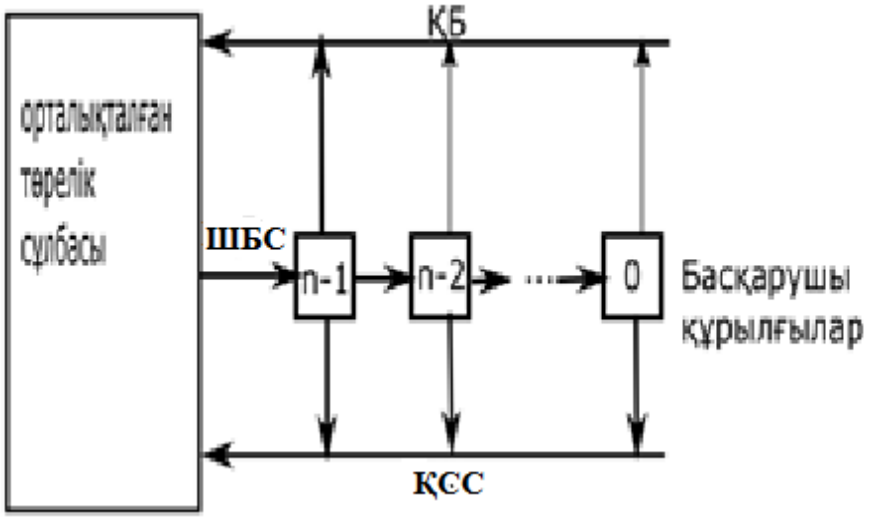
9.1-суретте орталықтандырылған параллель төрелік сұлбасы көрсетілген.



9.1-сурет. Орталықтандырылған төрелік сұлбасы

Орталықтандырылған төрелік сұлбасының екінші түріне орталықтандырылған тізбекті төрелік сұлбасы жатады.

Тізбекті төрелік сұлбасында артықшылық деңгейі жоғары құрылғыны іріктеу үшін барлық басқарушы құрылғылар тізбегі арқылы кезекпен өтетін сигнал қолданылады. Сондықтан мұндай төрелік сұлбасын тізбекті төрелік сұлба деп те атайды. Жүйедегі басқарушы құрылғылар тізбегінің артықшылық деңгейі сол жақтан оңға қарай төмендей береді. 9.2-суретте орталықтандырылған тізбекті төрелік сұлбасы көрсетілген.



9.2-сурет. Орталықтандырылған тізбекті төрелік сұлбасы.

Басқару құрылғыларынан шығатын сұраным сигналдары (ҚСС) бір желіге НЕМЕСЕ логикасымен біріктіріледі. Осыған ұқсас шинаның бос еместігін хабарлайтын сигналдары (ҚБ) бір желіге біріктірілген. Бір немесе бірнеше

басқарушылардан шығатын сұраным сигналдары (ҚСС) орталықтандырылған төрелік сұлбаға беріледі. ҚСС сигналдарын алып төрелік сұлба шинаның бос екендігіне көз жеткізеді. Егер шина бос болса, онда шинаны басқару сигналын (ҚБС) қалыптастыратын оны басқарушы құрылғыларына береді. ШБС басқару құрылғылары арқылы тізбекпен өтеді. Егер ШБС сигналы түскен құрылғы сұраным сигналын бермесе, онда ШБС сигналын ары қарай өткізеді. Ал құрылғы сұраным сигналын берген болса, онда ШБС сигналы келесі басқару құрылғысына беруді бұғаттайды. Тізбекпен төрелік ету сұлбасы статикалық артықшылық тәсілін іске асырады. Ең жоғарғы деңгей артықшылығын төрелік сұлбасына жақын құрылғы иеленеді. Ары қарай артықшылық деңгейі кеми береді.

Орталықтандырылған төрелік сұлбасында әрбір басқарушының өз құрамында шинаға қатынас құру блогы болады (шина контроллері). Мұндай блоктар бір-бірімен әрекеттесе отырып артықшылығы жоғары басқарушыны анықтап оған шинаны басқаруды тапсырады.

Мұндай төрелік сұлбаларда шина құрамында НЕМЕСЕ логикасы бойынша ұйымдастырылған арнайы желілер болады. Арнайы желілер арқылы әрбір басқарушы құрылғы қалған құрылғылардан шығатын сигналдарды бақылап отырады. Әрбір басқарушы құрылғының артықшылық деңгейіне сәйкес бірегей нөмірі болады. Шинаға сұратым жасаған құрылғы төрелік шинаға өз нөмірін береді. Әрбір сұратым жасаған басқарушы төрелік шинаның артықшылығынан жоғары артықшылығын көріп, төрелік желісінен өз нөмірінің төменгі биттерін алып тастайды. Ең соңында төрелік желіде артықшылық деңгейі жоғары тек бір нөмірді қосады. Осы нөмірге сәйкес келетін құрылғы басқарушы болып тағайындалады. Мұндай сұлбаларда әр басқарушы құрылғы өзінің басқарушы құрылғы екендігін өзі анықтайды.

9.2. Шиналар хаттамалары

Басқарушы құрылғыдан орындаушы құрылғы биттерін бір уақытта (параллель) берген кезде мекен-жай кодының биттері орындаушы жаққа барлығы бір уақытта келе бермейді. Өйткені биттерді беретін желілердің ұзындығы әртүрлі, олардың әртүрлі физикалық параметрлері т.б себептері болуы мүмкін. Берілген деректер дұрыс қабылдану үшін әртүрлі тәсілдер қолданылады. Мекенжай кодын, деректер басқару сигналын, ақпараттың күй сигналын дұрыс қабылдау үшін қолданылатын тәсілді шина хаттамасы деп атайды. Әдетте екі түрлі хаттамалар *синхронды* және *асинхронды хаттамалар* қолданылады.

Синхронды хаттамада дерекпен алмасу белгілі бір уақыт аралығында жүргізіледі. Уақыт аралығы генератордан шығаратын такт сигналдарымен беріледі. Хаттама сигналдарының барлығы осы такт импульсіне сәйкестендіріледі. Синхронды алмасу хаттамасы өте қарапайым, бірақ орындаушы құрылғыдан өте жоғарғы жылдамдықты керек етеді

Асинхронды хаттамада шина арқылы берілетін әрбір алмасу сигналына жауап ретінде «растау» сигналы шығарылады.

Асинхронды хаттаманың жұмыс сенімділігі жоғары, бірақ барлық орындаушы құрылғыларынан «растау» сигналдары қалыптасуы керек. Оларды қалыптастыру үшін қосымша шығын керек етеді.

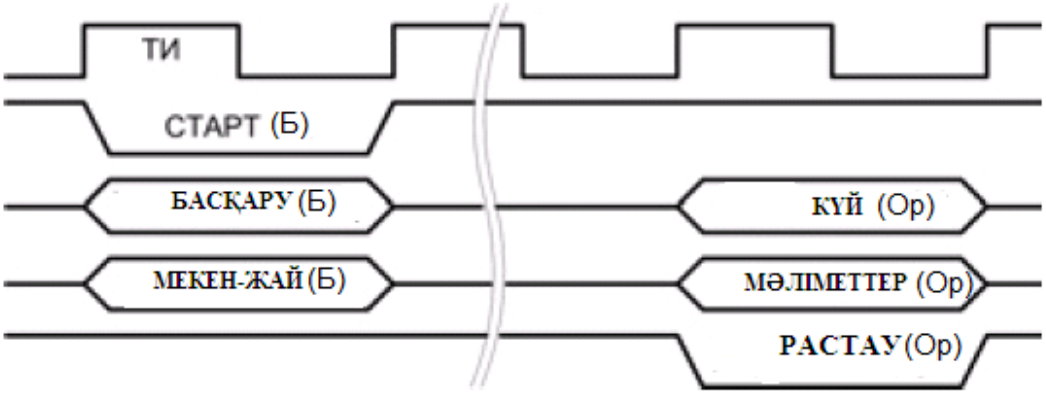
Синхронды хаттама

Синхронды шиналарда такт импульстары (ТИ) арнайы сигналдық желі арқылы таралады. ТИ тізбектірінің бір периодын шинаның такт периоды деп атайды. Такт периодымен шинаға бөлінетін уақыт аралығы анықталады. Шинаға қосылған құрылғылар алмасу әрекеттерін осы уақыт аралығында орындауы керек. Шинадағы басқару сигналдарының өзгерулері такт импульсінің алдыңғы немесе артқы фронттарына сәйкес келеді.

9.3-суретте қарапайым синхронды хаттама арқылы шинадағы «оқу» транзакциясы келтірілген (Б-басқарушы құрылғы сигналы, ал Ор – орындаушы құрылғысының сигналы).

Шинадағы сигналдардың өзгеру моменттері такт импульсінің алдыңғы фронтымен анықталады. Такт сигналының артқы фронты шина сигналдарының дұрыс қалыптасқандығын көрсетеді.

Старт сигналы шинада мекен-жай немесе басқару сигналдарының берілгенін көрсетеді. Орындаушы құрылғысы өз мекен-жайын анықтаған соң деректерді және құрылғы күй кодын шинаға орналастырып, осы туралы растау сигналын шығарады.



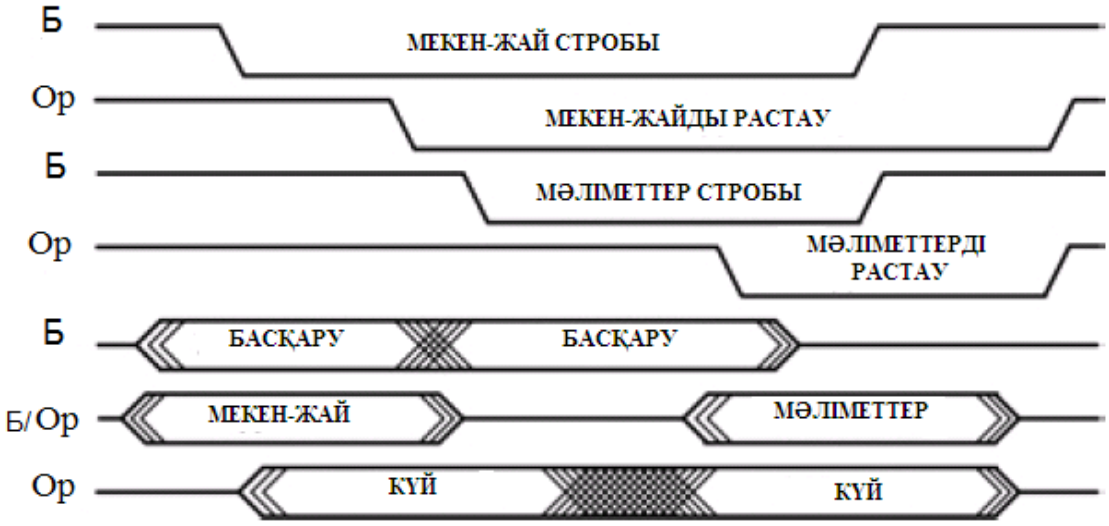
9.3-сурет. Синхронды шинада оқу транзакциясын орындау

Жазу операциясы осыған ұқсас түрде орындалады. Тек мұнда деректер басқарушы құрылғы алдыңғы такт периодында берілген мекен-жайы мен күй сөзі туралы орындаушы құрылғыдан растау сигналы келгеннен кейінгі такт периодында беріледі.

Асинхронды хаттама.

Асинхронды хаттамада басқарушы құрылғы кез келген ақпаратты орналастыру уақыт үйлестіргіш сигналымен- строб сигналымен қоса береді. Орындаушы құрылғы асинхронды хаттаманы іске асыру үшін өзінің сигналын-түбіртіктеуіш сигналын (handshakes) немесе хабарды растау (acknowledges) сигналын шығарады. 9.4–суретте асинхронды оқу операциясының орындалуы көрсетілген.

Асинхронды хаттамада деректі оқу үшін басқарушы құрылғы шинасына мекен-жай кодын береді және басқару сигналдарын шинаға беріп, барлық сигналдар мәреге жетті-ау дегенде мекен-жай стробын жібереді. Орындаушы құрылғылар мекен-жай шинасын бақылай отырып, берілген мекен-жай өзінікі болса, онда ол өзінің күйін мекен-жай кодын жіберу арқылы растайды. Расталған мекен-жай кодын алған басқарушы құрылғы байланыс орнының алғандығына көз жеткізіп күй сөзін талдауға кіріседі.



9.49.4-сурет. Асинхронды шинада оқу транзакциясын орындау

Ары қарай басқарушы құрылғы басқару сигналдарын өзгертіп, шинаға дерек стробын береді. Егер де жазу транзакциясы орындалып жатса, онда басқарушы басқару ақпараттарымен қоса шинаға жазатын деректерді береді.

Орындаушы құрылғы керекті деректі дайындап болған соң ол оны шинаға құрылғының жаңа күйімен бірге беріп, деректі растау сигналын қалыптастырады. Басқарушы құрылғы деректі растау сигналын көріп шинадан деректі оқып, дерек стробын алып тастайды. Осы арқылы деректерді алу

операциясының аяқталғанын білдіреді. Біздің мысалда басқарушы құрылғы мекен-жай стробын да алып тастайды.

Жоғарыдан көрініп тұрғандай, асинхронды шина циклінде операцияның дұрыс өткендігін көрсету үшін екі бағытты басқару сигналдары қолданылады. Мұндай процедура қол алысу (handshake) процедурасы деп аталады.

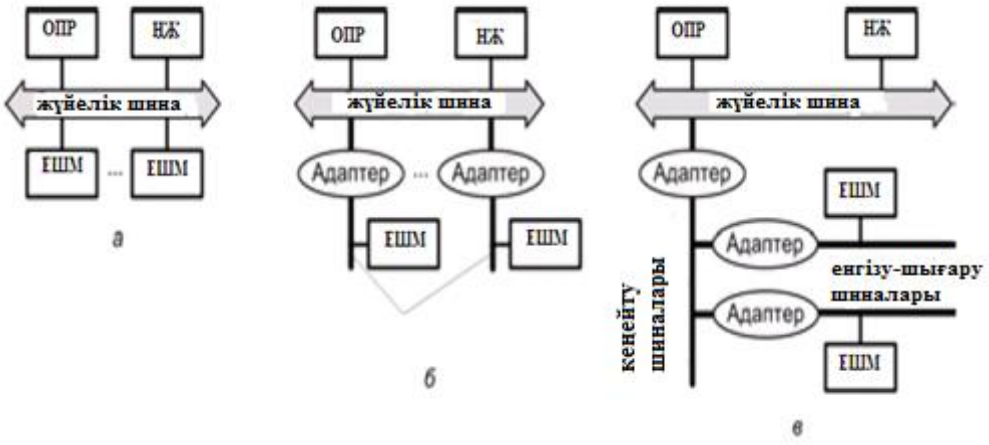
Әдетте енгізу/шығару шиналары асинхронды хаттамамен жұмыс жасайды.

9.3. Шина түрлері

Атқаратын қызметтеріне байланысты есептеу жүйесінің шиналарын мынадай түрлерге бөлуге болады:

- жүйелік шиналар;
- «процессор-жады»шиналары;
- кеңейту шиналары;
- сырт құрылғылар шиналары.

Егер есептеу машинасы тек жүйелік шиналар арқылы құрылса, ондай машина біршиналы машина деп аталады. (9.5, а-сурет) мұнда машинаның құрамына кіретін барлық құрылғылар жүйелік құрылғыға қосылған, сондықтан барлық деректер алмасуы тек бір шина арқылы орындалады. Мұндай машиналардың құрылымы өте қарапайым. Бірақ бір шиналы машиналарда транзакция жылдамдықтары төмен. Ол машина жылдамдығына әсер етеді.



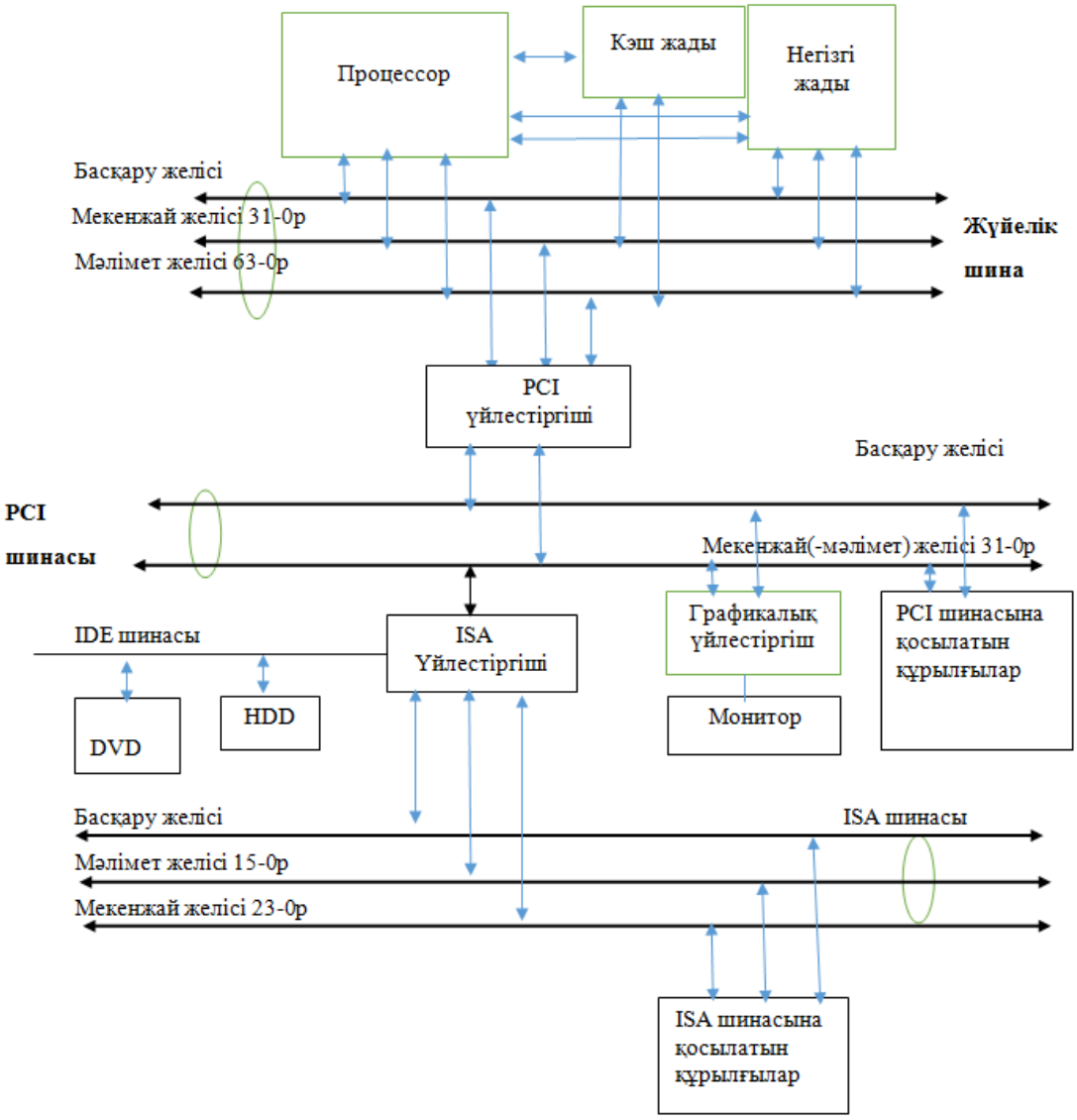
9.5-сурет. Шиналар санымен анықталатын есептеу машиналары: а-бір шиналы; б-екі шиналы; в-үш шиналы.

Суретте: ОПр - орталықпроцессор, НЖ – негізгі жады, ЕШМ - енгізу/шығару модулі.

ЕМ өнімділігін арттыру үшін жүйелік шина жүктемесін азайту мақсатында ЕШМ жүйелік шинасына енгізу/шығару шиналары арқылы

қосылады. Мұндай жүйеде енгізу/шығару шиналары жүйелік шинаға шина үйлестіргіштер адаптері арқылы қосылады. Үйлестіргіштер берілетін деректер үлгілерін сәйкестендіреді, шиналарда деректер алмасуын басқарады. Мұндай есептеу машиналарын екі шиналы машиналар деп атайды (9.5, б-сурет).

Жылдамдығы үлкен сырт құрылғыларын (магниттік, оптикалық дискілер) қосу үшін шиналар жүйесіне жылдамдығы жоғары кеңейту шинасын қосады. Мұндай жүйелерде енгізу/шығару шиналары үйлестіргіштер арқылы кеңейту шинасына қосылады, ал кеңейту шинасы жүйелік шинаға өзінің үйлестіргіші арқылы қосылады. Мұндай көп сатылы шинасы бар ЕМ көпшиналы ЕМ деп аталады. Көп шиналы ЕЖ жүйелік шина жүктемесін айтарлықтай азайтады. 9.6-суретте көп шиналы компьютер құрылымының иерархиясы көрсетілген.



9.6-сурет. Көп шиналы компьютер құрылымы

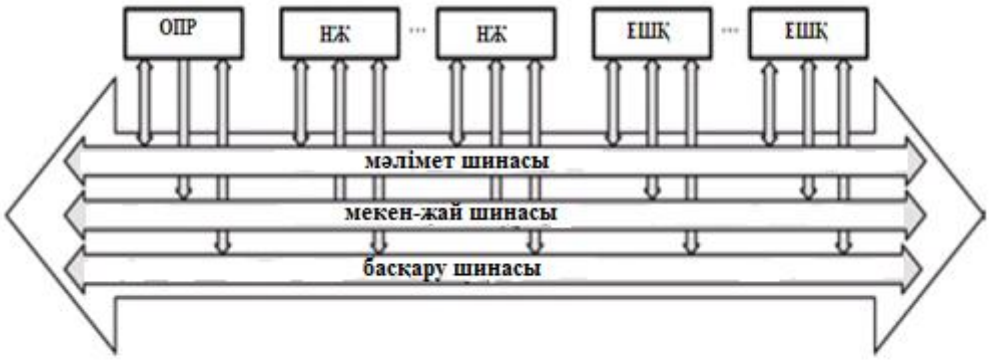
Шиналар иерархиясының бірінші деңгейіне жүйелік шина, және «процессор-жады»шиналары, екінші деңгейіне PCI шинасы, үшінші деңгейіне

IDE шинасы жатады. Ең төменгі деңгейге ISA енгізу/шығару құрылғыларының шиналары жатады.

Жүйелік шина.

Бір шинады есептеу машиналарында процессор, негізгі жады және енгізу/шығару құрылғылары ортақ шиналарға қосылады. Бұл ЕМ құнын азайтуға алып келеді. Мұндай шиналар жүйелік шиналар деп аталады.

Жүйелік шиналар әдетте жүздеген желілерден тұрады. Олар үш топқа бөлінеді: дерекшинасы, мекенжай және басқару шиналары. Соңғы шина тобына қорек желісі кіреді (9.7- сурет).



9.7- сурет. Жүйелік шина құрылымы.

Жүйелік шинада кез келген транзакция жетекші құрылғының мекендік ақпаратын шинаға қоюдан басталады. Мекенжай коды орындаушы құрылғыны таңдап онымен байланыс құруға мүмкіндік береді.

Мекенжай кодын беру үшін мекенжай шинасы (МЖШ) деп аталатын сигналдық желінің белгілі бөлігі пайдаланылады.

МЖШ жады ұяшығының мекен-жайы, ОПр регистрлерінің нөмірі, енгізу/шығару порттарының мекен-жайы т.б беріледі.

Мекенжай кодын беретін сигналдық желілер санымен (мекенжай шинасының енімен) мүмкін болатын мекенжай кеңістігінің ең үлкен өлшемі анықталады. Мекенжай шинасының енімен мекенденетін жады құрылғысының сыйымдылығы және жүйеге қосылатын енгізу/шығару порттарының саны анықталады. Қазіргі МЖК 36 сигналдық желілерден тұрады. Мұндай желі саны арқылы мекенжай кеңістігі 64 Г байтқа жетеді.

Модульдер арасында деректерді беруге арналған желілер жиынтығын деректер шинасы (МҚ) деп атайды. МШ негізгі сипаттасына шина ені мен деректерді енгізу қабілеттігі жатады. Дерекшинасының ені бір транзакция (шина циклі) кезінде берілетін дерек биттерінің санымен анықталады. Бір транзакцияны орындауға бірнеше такт периодтары керек болуы мүмкін. Қазіргі

есептеу машиналарында дерекшинасының ені 32, 64 немесе 128 биттерден тұрады. Әдетте машина сөзінің разрядтылығы шина енінің разрядтылығымен анықталады. Кейбір ЕМ сөз разрядтылығы шина еніне сәйкес келе бермейді.

Кейбір ЕМ мекенжай кодасы мен деректері бір желілер арқылы әртүрлі уақыт мезетінде кезекпен беріледі. Аталған берілуде мекенжай мен дерек желілері бір мультиплекстенген мекен-жай/дерекшиналарына біріктірілген. Мұндай шиналар желілерінде арнайы жолдың бір басында мультиплексор болса, екінші басында демультиплексор болады. Олар шина коммутаторлары қызметін атқарады.

Мультиплекстеу шина желілерінің санын қысқартуға мүмкіндік береді, бірақ жүйе өнімділігін төмендетуге алып келеді.

Деректер мен мекенжай кодаларын беретін арнажолдан басқа басқару ақпараттарын, транзакцияға қатысы бар құрылғылардың күйін беретін шина желілері бар. Мұндай желілер жиынтығын басқару шиналары (БҚ) деп атайды. Басқару шиналарының сигнал желілері бірнеше топқа бөлінеді.

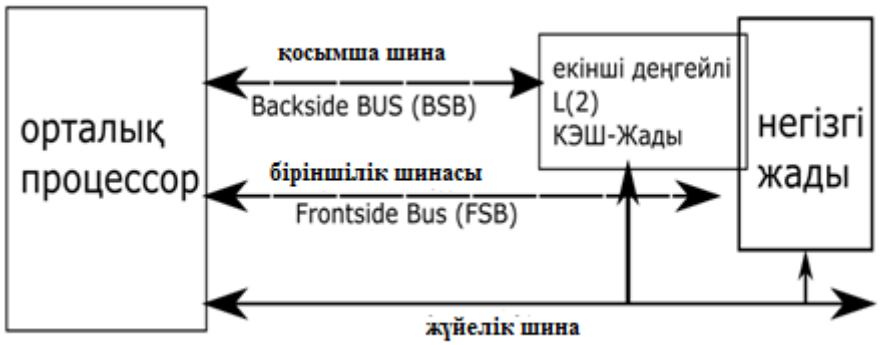
- Бірінші топқа төменде келтірілген транзакцияны басқару сигналдарын беретін сигналдар желілері жатады:
  - транзакция типі (оқу немесе жазу);
  - дерекшинасы арқылы берілетін байт саны және егер сөз бөлігі берілетін болса, онда байт нөмірі берілетін мекенжай түрі;
  - беру хаттамасының түрі.
- Екінші топқа басқарушы құрылғыларының күйін немесе қате кодын беретін сигналдар желілері жатады.
- Үшінші топқа төрелік желілері жатады.
- Төртінші топ үзілім желілерін құрады. Желілер арқылы орындаушы құрылғылардан келетін сұратым сигналдары беріледі.
- Бесінші топқа есептеу желілерін ұйымдастыруға керекті сигнал желілері жатады.
- Алтыншы топқа жататын сигнал желілері арқылы ЕМ құрылғылар жұмысын реттейтін, уақыт бойынша үйлестіретін сигналдары беріледі.

Басқару шинасының құрамына қорек көзін қосатын желілерді де қосуға болады.

«Процессор-жады»шинасы.

Қазіргі микропроцессорларда орталық процессорды негізгі жадымен байланыстыру үшін жүйелік шиналарымен қатар «процессор-жады» шиналары арқылы тікелей байланысыруды да қолдануға болады (9.8- сурет)

Орталық процессорды негізгі жадымен тікелей байланыстыратын шинаны біріншілік шинасы деп атайды FSB (Front-Side BUS). Бұл шинаның деректерді өткізгіш қасиеті өте жоғары: «процессор жады» шинасына процессорды екінші деңгейлі (L2) КЭШ жадымен байланыстыратын қосымша шина –BSB (Bach-Side BUS) жатады.



9.8-сурет.“Процессор-жады”шиналары.

BSB шинасы процессормен КЭШ жады дерек алмасуын жеделдетеді. Егер орталық процессор негізгі жадымен BSB және FSB шиналары арқылы алмасу операциясын іске асырса, онда мұндай архитектураны DIB (Dual Independent) қос тәуелсіз шина архитектурасы деп атайды. Мұндай қос шина арқылы орталық процессор негізгі жады және (L2) КЭШ жадыға қатар қатынас құру арқылы ЕМ жалпы өнімділігін арттыруға мүмкіндік алады.

Кеңейту шиналары.

ЕМ-ның орталық құрылғыларына ЕШМ арқылы көптеген сырт құрылғыларын қосуға тура келеді. Әртүрлі сырт құрылғыларымен алмасатын ақпараттар қарқыны әртүрлі. Сырт құрылғыларын ақпараттық қарқындылығына қарай топтап, әр топты арнайы шиналар арқылы орталық құрылғыларға қосады. Мұндай шиналарды кеңейту шиналарына ЕШМ модульдері арқылы әртүрлі сырт құрылғылары қосылады және төменгі деңгейдегі шина өз үйлестіргіші арқылы жоғары деңгейлі шинаға қосылады. Мұндай кеңейту шиналарына 9.6 - суретте көрсетілген ISA және PCI шиналары жатады.

ISA шинасы (Industry Standart Architecture) IBM PC машинасының бірінші моделінен бастап қолданылған. Ол өндірістік стандарт болып табылады. XT компьютерінде дерекшинасының разрядтылығы 8 бит, ал мекенжай разрядтылығы 20 бит болатын. PC/AT компьютерінен бастап дерек разрядтылығы 16 битке, ал мекенжай разрядтылығы 24 битке дейін өсірілді. ISA шинасының деректі беру жиілігі 8,33 МГц. Ол арқылы бір цикл ішінде екі байт бере аламыз, сондықтан оның деректі өткізу жылдамдығы 16,11 байт.

ISA шинасы 32 разрядқа дейін кеңейтілген. Мұндай кеңейтілген шина EISA (Extended ISA) деген атпен белгілі. EISA шинасы арқылы бір циклде 4 байт беруге болады. Оның деректерді өткізгіш қабілеті 33,3 Мбайт/с.

ISA және EISA шиналары үшін әртүрлі сырт құрылғыларын жүйеге қосуға мүмкіндік беретін көптеген кеңейту карталары өндірістік жолмен шығарылған және шығарылып отыр.

Кеңейту шиналарының екінші түріне PCI (Peripheral Component Interconnect перифериалдық компоненттерді қосу) шинасы жатады. PCI шинасы қазіргі компьютерлердің негізгі шинасы болып табылады. Өйткені PCI шинасы өте сенімді, өнімділігі жоғары стандартталған шина болып табылады.

PCI шинасының алғашқы түрінде бір циклдің ішіне 32 бит беріп, 33 МГц жиілігінде жұмыс жасады. 1993 жылы PCI 2.0, ал 1995 жылы PCI 2.1 пайда болды. Қазіргі PCI шинасы 66МГц жиілігінде жұмыс істеп, бір циклде 64 бит береді, оның жалпы өткізу қабілеті 528 Мбайт/с-қа өсті.

Сырт құрылғылар шиналары.

PCI шинасына әртүрлі жылдам жұмыс жасайтын сырт құрылғылары қосылады. PCI шинасына графикалық үйлестіргіш арқылы тікелей мониторды, SCSI және USB шиналары арқылы әртүрлі сырт құрылғыларын қосуға мүмкіндік бар, ISA үйлестіргіш арқылы ISA шинасын PCI шинасына қосуға болады.

PCI шинасына ATA (IDE) параллель шинасы ISA үйлестіргіші арқылы қосылады. IDE (Integrated Drive Electronics) шинасына оптикалық (DVD) және магниттік дискілер (HDD) қосылады. ATA параллель шинасы оның разрядтылығына, канал сандары мен оларға қосылатын құрылғылар санына т.б байланысты көптеген түрлерге бөлінеді.

Intel фирмасы PCI 2.1 негізінде графикалық мониторды PCI шинасына тікелей қосатын үйлестіргіш –AGP (Accelerated Graphic - Port жедел графикалық порт) үйлестіргішін іске қосты.

AGP порты 32-разрядты шина. Такт жиілігі - 66 МГц. Порттың жедел жұмыс жасауы мынадай факторларға байланысты:

- жады құрылғысына қатынас құруды конвейерлеу;
- деректі беруді екі (2x) немесе төрт (4x) есе жиіліктерімен беру;
- дерекшинасын және мекенжай шинасын мультиплекстеу.

Жоғарыда аталған IDE шинасынан басқа PCI кеңейту шинасымен қосылатын шинаға SCSI (Small Computer System Interface - шағын компьютерлердің жүйелік интерфейсі) шинасы жатады. SCSI интерфейсі арқылы әртүрлі құрылғылар: тура қатынас құратын дискілер (HDD), тізбекті

қатынас құратын магниттік таспалар (стриммерлер), оптикалық дискілер, принтерлер, сканерлер т.б. сырт құрылғылары компьютерге PCI шинасы арқылы қосылады. Интерфейс 8-битті параллель шинадан тұрады. Такт жиілігі 5МГц. Шинаға 8 құрылғыны қатар қосуға болады. Шинаның алғашқы нұсқасында деректі беру жылдамдығы 5 Мбайт/с болған. Кейінгі нұсқасында (Fast SCSI-2) такт жиілігі 10 МГц болса, Ultra SCSI-2 -20 МГцке жеткен. Дерек разрядтылығы 16 бит. 16-битті шина 16 құрылғымен қатар жұмыс жасай алады.

SCSI-2 стандартында интерфейс разрядтылығы 32 разряд.

SCSI-3 стандарты компьютерге қосылатын сырт құрылғыларының санын өсіруге бағытталған.

PCI шинасына қосылатын жоғарыдағы аталған IDE, SCSI шиналары сырт құрылғылардың параллель шиналарының қатарына жатады.

Сырт құрылғыларының параллель шиналары басқа тізбекті шиналар (1-2 жұп сандар) арқылы компьютердің орталық құрылғыларына қосылады. Ондай шиналардың қатарына USB (Universal Serial BUS-әмбебап тізбекті шина), FireWire (Hige Performance Serial Bus-өнімділігі жоғары тізбекті шина), деректі сымсыз беретін технологиямен жұмыс жасайтын интерфейс Bluetooth (көк тіс), инфрақызыл интерфейс IRDA (Infra red Assotiation) жатады.

USB шинасы жылдамдығы төмен және орта жылдамдықты сырт құрылғыларын ЕМ-ге қосуға арналған. Сырт құрылғыларды ЕМ-ге қосу үшін төрт сымды кәбіл қолданылады. Олардың екеуі арқылы деректі беруге және қабылдауға болады, ал қалған екеуі сырт құрылғысына берілетін қорек сым болып табылады. USB бір контроллеріне 127 құрылғыларды «жұлдызша» топологиясымен қосуға болады. USB-2 стандартының жылдамдығы 480 Мбит/с, ал USB 3 жылдамдығы 4,8Гбит/с.

Wireless шина *USB* нұсқасын сымсыз байланыс түрінде іске асырады. Арақашықтығы 3м болғанда дерек 480 Мбит/с жылдамдықпен, ал 10 метрде 110Мбит/с жылдамдықпен беріледі.

FireWire шинасы арқылы сырт құрылғы ЕМ-нің 6 сым желісімен байланысады. Олардың төртеуі деректі беруге арналса, екеуі сырт құрылғыға қорек жеткізеді. Шина арқылы 800 Мбит ақпарат жеткізе алады. Деректер десте режимінде беріледі. Өнімділігі жөнінен FireWire USB шинасынан асып түседі. *Bluetooth* интерфейсі 2,4ГГц жиілігінде жұмыс жасайды. Ол үшін қосылатын құрылғылар бір бірімен көрінетін зонада болуы керек. Деректі беру жылдамдығы 2-3 Мбит/с.

Bluetooth 3.0 стандарты, ол 24 Мбит/с-ке ұлғайтылған.

IrDA интерфейсі байланыс толқын ұзындығы 880 нм инфрақызыл сәулесі арқылы жасалынады. Қосылатын құрылғылар бір бірінен бір метрге дейінгі қашықтықта болу керек. IrDA құрудың негізгі мақсаты - энергияны аз пайдалану және арзан интерфейс құру.

Бақылау сұрақтары

1. Транзакция дегеніміз не?
2. Төрелік шинаның атқаратын қызметі?
3. Артықшылықты динамикалық өзгерту қандай алгоритмдер арқылы іске асырылады?
4. Төрелік сұлбаларының қандай түрін білесіз?
5. Синхронды шинада оқу транзакциясы қалай орындалады?
6. Асинхронды шинада оқу транзакциясы қалай орындалады?
7. Шиналар санымен анықталатын есептеу машиналарын атаңыз?
8. Қазіргі процессорларда орталық процессорды негізгі жадымен байланыстыру үшін жүйелік шиналармен қатар қандай шиналар қолданылады?
9. Жүйелік шиналар үш топқа бөлінеді. Оларды атап және атқаратын қызметін айтыңыз?
10. ISA, EISA және PCI шиналары қандай қызмет атқарады?
11. Сырт құрылғылар шиналарын атаңыз?
12. Қандай шиналар сымсыз байланысты іске асырады?

10. ЕНГІЗУ-ШЫҒАРУ ЖҮЙЕЛЕРІН ҰЙЫМДАСТЫРУ

Компьютерлік жүйелер құрамына жүйе ядросын құратын процессордан (процессорлардан) және негізгі жады құрылғыларынан басқа жұмыс жасау принциптері әртүрлі болып келетін алуан түрлі сырт құрылғылар (СҚ) кіреді.

Деректерді СҚ-дан ЕМ ядросына беру операциясын енгізу, ал ЕМ ядросынан СҚ-ға беру операциясын – шығару операциясы деп айтамыз. ЕМ пайдалану тиімділігі және оның өнімділігі машина процессорының мүмкіндігімен немесе НЖ құрылғысының көрсеткіштерімен ғана емес, СҚ құрамымен, олардың техникалық мүмкіндігімен және ЕМ ядросымен өзара дерек алмасуды ұйымдастыру тәсілдерімен де анықталады.

ЕМ құрылымдары бір-бірімен есептеу техникасында интерфейс деп аталатын үйлестіргіштер арқылы байланыс құрады.

Интерфейс деп машина құрылғыларының бір-бірімен деректер алмасуын қамтамасыз ететін деректердің мекен-жай кодтарын, басқару сигналдарын тасымалдайтын шиналар жиынтығын және электрондық сұлбалар мен алгоритмдерін (хаттамаларын) айтамыз.

Техникалық тұрғыдан енгізу/шығару операциялары енгізу/шығару модульдері (ЕШМ) арқылы іске қосылады. ЕШМ-ға екі түрлі міндет жүктеледі. Олар:

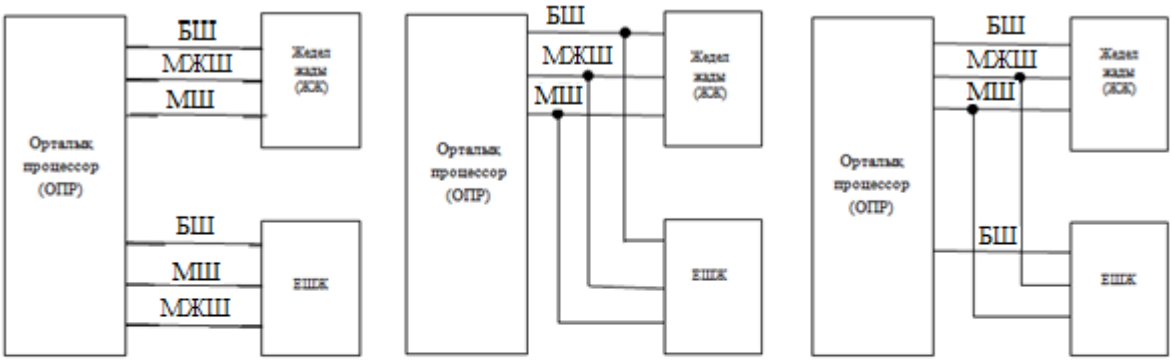
- сырт құрылғыларының машина ядросымен интерфейс міндетін атқару («үлкен» интерфейс);
- машина ядросының сырт құрылғыларының бірімен немесе бірнешеуімен интерфейс міндетін атқару («кіші» интерфейс);

Енгізу/шығару модулін және оған қосылған СҚ жиынтығын енгізу/шығару жүйесі (ЕШЖ) деп атайды.

10.1 Енгізу/шығару жүйесін орталық процессорға қосу тәсілдері

ЕШЖ-ні орталық процессорға (ОПР) әртүрлі жолмен қосуға болады. Олардың үш тәсілі 10.1– суретте көрсетілген.

Бірінші тәсілде – (10.1, а-сурет) ОПР жедел жадымен және ЕШЖ тәуелсіз шиналар арқылы алмасады. Мұнда ОПР, ЖЖ және ЕШЖ енгізу/шығару операцияларын қатар орындауға мүмкіндік алады. Оның үстіне ЕШМ байланыстыратын шина құрылымы, СҚ алмасу алгоритмдері мен дерекипішімдеріне және деректерді беру жылдамдықтарына байланысты аппараттық шығындар төмен болуы мүмкін. Бірақ мұндай тәсілді іске асыру үшін орталық процессорда қосу нүктелері көп болуы керек. Бұл орталық процессор сұлбасын күрделілендіруге алып келеді.



10.1 – сурет. ЕШЖ орталық процессорға қосу тәсілдері. а – ОПР, ЕШЖ және ЖЖ жеке шиналар арқылы қосу; б – ОПР, ЕШЖ және ЖЖ ортақ шиналар арқылы қосу; в – ОПР, ЕШЖ және ЖЖ мекен-жай құрсымдары (МЖШ) және дерекшиналары (МШ) арқылы қосу.

ЕШМ ОПР қосудың екінші түрінде (10.1, б– сурет) ЖЖ және ЕШМ ОПР қосу үшін ортақ шиналар қолданады. Мұндай тәсілде шина саны, ОПР қосу нүктелері 1-тәсілге қарағанда екі есе аз. Оның кемістігі – шиналар арқылы алмасу ЕШМ мен ЖЖ кезекпе-кезек жүргізіледі. Бұл жүйе өнімділігін азайтуға алып келеді. Үшінші тәсілде (10.1, в– сурет) ЖЖ және ЕШЖ енгізу/шығару операциялары ортақ мекен-жай шиналары және дерекшиналары арқылы орындалады.

ЖЖ мен ЕШЖ басқару сигналдары бөлек басқару желілері арқылы басқарылады. Мұндай бөлек басқару тәсілінде ЖЖ және ЕШЖ енгізу/шығару модульдерінің жұмыс ерекшеліктері еске алынады.

Орталық процессор ЖЖ ұяшықтарына қатынас құру үшін мекен-жай кодтарын пайдаланады. Сол сияқты енгізу/шығару командаларын орындағанда да мекендеу жүйесі қолданылады. Мекен-жай арқылы көптеген енгізу/шығару модульдерінің бірі анықталып, модульге қосылған СҚ бірін таңдайды.

ЕШМ мекен-жай кеңістігі ЖЖ мекен-жай кеңістігінің құрамдас бір бөлігі, не ол ЖЖ мекен-жай кеңістігінен бөлек кеңістікте болуы мүмкін.

Бірінші жағдайда ЕШМ мекендеу үшін ЖЖ мекен-жай кеңістігінен белгілі бір мекен-жай аймағы бөлінеді. Бөлінген аймақта ЕШ модульдерінің регистрлері: басқару, дерек регистрлері орналасады. Мұндай жағдайда енгізу/шығару процедурасында деректер модулінің бір регистрінде жазылып, ал оны оқу басқа регистрлермен жүргізіледі. Мұндай енгізу/шығару мекен-

жайларының біріктірілген түрінде әр ЕШМ регистрлерін ЖЖ ұяшықтары ретінде қарауға болады. Сондықтан орталық процессорда арнайы енгізу/шығару командалары керек болмайды.

Бөлек кеңістікті енгізу/шығару мекен-жайында ЕШМ-ге қатынас құру үшін оның арнайы командалары және мекендеу жүйелері болады. Жады арналары енгізу-шығару арналарынан бөлек құрылады. Соның арқасында арнажолдарда алмасу операцияларын қатар орындауға мүмкіндік туады.

Әрбір сырт құрылғылары өз ЕШМ-не жеке шиналар арқылы қосылады. ЕШМ мен СҚ әрекеттестіктер кіші интерфейс арқылы іске асырылады. Жеке шина арқылы деректер, басқару сигналдары және алмасуға қатысатын құрылғылар күйлері беріледі. Деректер шинасы арқылы ЕШ модуліне берілетін, не одан алынатын биттер жиынтығы берілсе, басқару сигналдары арқылы СҚ атқарылатын әрекеттер анықталады. Оған стандартты әрекет - деректерді ЕШМ-ге беру немесе одан алу, немесе әр СҚ-ның кезекті арнайы әрекет түрлері жатады, мысалы, магниттік бастиекті позициялау, магниттік таспаны қайта орау т.с.с. Күй сигналдары құрылғалардың күйін сипаттайды, мысалы, СҚ іске қосулы ма, жоқ па? СҚ құрамына жергілікті басқару құрылғысы және аралық жады кіреді. Аралық жадыға деректер түрлендіргіш арқылы келуі мүмкін. Түрлендіргіштер СҚ келетін аналог сигналдарын цифрлық деректерге, немесе СҚ берілетін цифрлық сигналдарды аналог сигналдарға түрлендіруге арналған.

10.2. Енгізу-шығару модульдері

ЕШМ өзіне қосылған бір немесе бірнеше сырт құрылғыларын негізгі жады құрылғысымен, немесе орталық процессордың регистрлерімен дерек алмасуын басқарады.

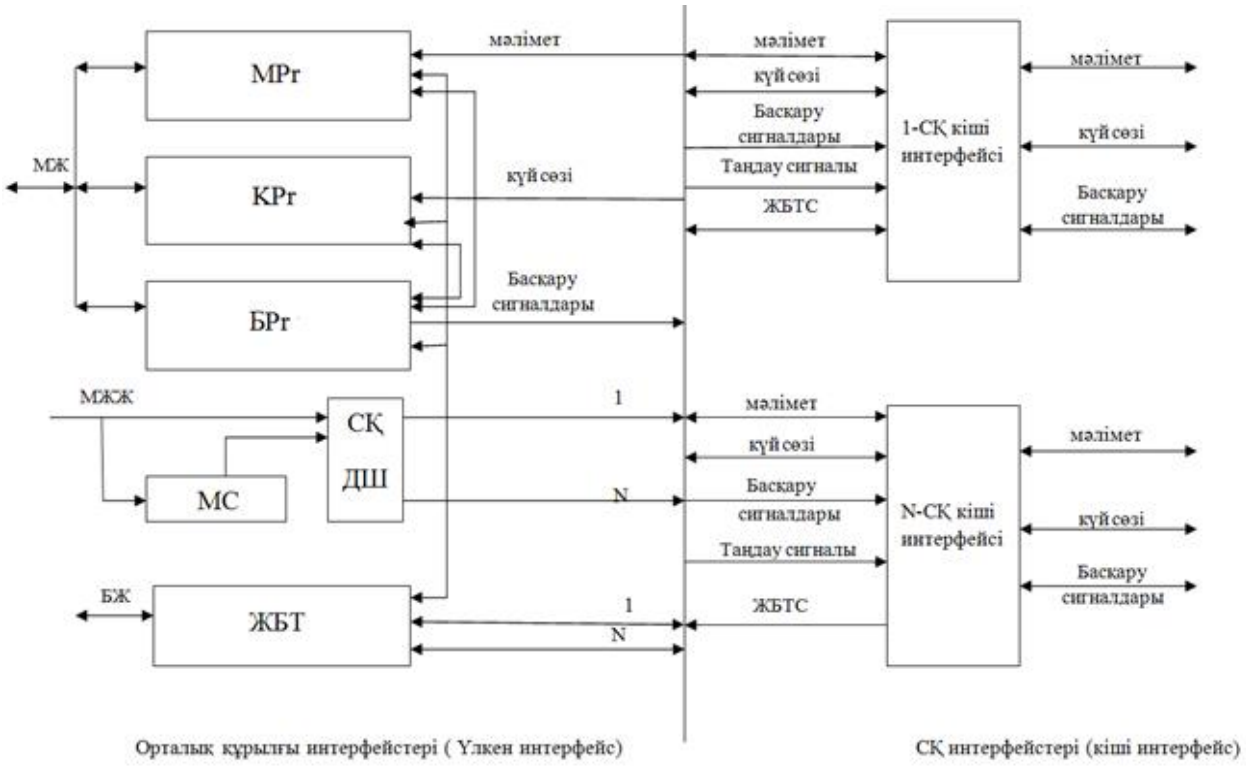
ЕШМ атқаратын негізгі міндеттеріне мыналар жатады:

- ЕШМ-ге қосылған СҚ ішінен бірін таңдап, құрылға жады мекен-жайын анықтау;
- енгізу/шығару операциясын басқару;
- ақпарат алмасуды қамтамсыз ету;
- деректерде буферлеу;
- қателерді анықтау.

Сырт құрылғы мекен-жайы енгізу/шығару командаларының мекен-жай бөлігінде көрсетіледі. Әрбір ЕШМ-ге бірнеше СҚ қосылуы мүмкін, сондықтан әрбір модульге белгілі мекен-жай ауқымы бөлінеді. Командадағы мекен-жай кодының жоғарғы разрядтары арқылы ЕШМ бірі таңдалады. Ал төменгі разрядтары арқылы модульге қосылған сырт құрылғының регистрлері

анықталады. Қарапайым СҚ, олардың регистрлері және ондағы деректері тікелей мекен-жай кодының төменгі разрядтары арқылы анықталса, ал сырт жады құрылғыларына қатынас құру үстінде сақталған деректі табу күрделірек. Мысалы, магниттік диск үшін цилиндр нөмірін т.б. көрсету керек. Мұндай мекендеу ақпараты ЕШМ мекен-жай шинасы емес, ақпарат шинасы арқылы қызметтік хабар ретінде беріледі.

Енгізу/шығару операциясын басқару түйіні ЕШМ машинасын ішкі құрылғылары мен СҚ арасында деректермен алмасуды қамтамасыз етеді. Орталық процессор бір уақытта бірнеше СҚ-мен байланысады және ЕШМ қосылған СҚ жылдамдықтары әртүрлі (секундына бірнеше байттан бастап, ондаған миллион байтқа жетеді) болады. Осыған байланысты егер орталық процессор негізгі жадымен синхронды тәртібінде жұмыс жасаса, онда сырт құрылғылармен олардың алмасу жылдамдықтарына байланысты синхронды тәртібінде жұмыс жасай алмайды. Сырт құрылғыларымен кезекті дерек алмасу тек олар дайын болғаннан кейін ғана жүргізіледі.



10.2-сурет. Енгізу-шығару модулінің құрылымдық сұлбасы

ЕШМ дерек алмасу кезінде, «үлкен» интерфейс жағынан – ол орталық процессормен, ал «кіші» интерфейс жағынан – ол СҚ алмасады. Алмасу кезінде ЕШМ енгізу/шығару командасына байланысты өзіне қосылған СҚ ішінен таңдап, оны орталық процессорға қосады; модуль мен оған қосылған СҚ күйлерін талдап, егер олар алмасуға дайын болса, алмасу басталады. Дерекпен алмасу аяқталған соң СҚ процессордан ажыратылады.

Процессор мен жедел жады құрылғыларының алмасу жылдамдықтары әртүрлі, олардың алмасу жылдамдықтарын сәйкестендіру үшін ЕШМ құрамына аралық жады кіреді.

10.3. Енгізу – шығару модулінің құрылымы

ЕШМ құрылымдық сұлбасы 10.2-суретте көрсетілген. ЕШМ есептеу машинасының ядросымен жүйелік немесе арнайы шиналар арқылы байланысады. Бұл жағынан ЕШМ «үлкен» интерфейсті құрады. ЕШМ мен ЕМ ядросы аралық жадысын құратын дерек регистрі (МРг) арқылы алмасады. Әдетте «үлкен» интерфейс жағында шина желісінің саны 4 немесе 8 байт болады, ал көптеген СҚ ақпарат бір байтпен алмасады. Сондықтан, «кіші» интерфейсте дерек регистрі құрамында бір байтты деректерді жинақтап төрт немесе 8 байт құратын (енгізу үстінде), немесе 4 немесе 8 байтты сөзді бір байттан шығаратын арнайы түйін болады.

ЕШМ келесі регистрі - басқару регистрі (БРг). Басқару регистрінде орталық процессордан келетін модуль мен оған қосылған СҚ басқаратын команда қабылданады. Регистрдің жеке разрядтары жеке командалар болуы мүмкін, мысалы, ЕШМ регистрін тазалау, СҚ бастапқы күйге көшіру, оқуды не жазуды бастау т.с.с командаларды көрсетуі мүмкін. Өте күрделі ЕШМ бірнеше басқару регистрлері болады. Күй регистрінде (КРг) ЕШМ мен оған қосылған СҚ күйлерін көрсететін биттерді сақтап, күй сөзін құрады.

Күй регистрінің жеке разрядтары құрылғы кезекті дерек бірлігін «қабылдауға дайын» немесе «құрылғы бос емес», «құрылғыға қорек көзі қосылмаған» т.с.с құрылғы күйлері көрсетеді.

МРг, КРг, БРг орталық құрылғылармен дерек желілері арқылы алмасады. Жоғарыда айтылғандай негізгі жадының мекен-жай кеңістігінде әрбір ЕШМ бірегей мекен-жай бекітіледі. Мұндай мекен-жай саны ЕШМ мекендейтін СҚ санымен анықталады. Орталық процессордан келетін мекен-жай осы ЕШМ жататындығын мекен-жай сұрыптауыш (МС) арқылы тексереді. Егер келген мекен-жай, осы ЕШМ мекен-жай құрамына кіретін болса, онда процессордан келетін мекен-жай кодының төменгі разрядтары МС арқылы СҚ дешифраторы беріліп, СҚ бірі таңдалып, оған тиісті таңдау сигналы жіберіледі. Орталық процессордан келетін мекен-жай магистралдық шинасының желілері (МЖЖ) арқылы беріледі.

Деректермен алмасу және ЕШМ жұмысын жергілікті басқару түйіні (ЖБТ) жүргізеді. Егер ЕШМ сырт құрылғыларымен барлық басқару қызметін өз міндетіне толық алса, онда орталық процессормен тек жоғары деңгейлі командамен алмасатын болады. Мұндай ЕШМ *енгізу/шығару процессоры* немесе *енгізу/шығару арнасы* деп атаймыз. Егер СҚ басқару негізін орталық

процессор арқылы жүргізілсе, онда ондай қарапайым ЕШМ *енгізу/шығару контроллері* немесе *құрылғы контроллері* деп атаймыз.

10.4 Енгізу/шығару операцияларын ұйымдастыру тәсілдері

Енгізу/шығару операцияларын басқарудың үш тәсілі белгілі. Олар:

- сұратым арқылы енгізу/шығару;
- үзу сигналдары арқылы енгізу/шығару;
- жадыға тікелей қатынас құру.

Бірінші екі тәсілде енгізу/шығару операциясы толығымен орталық процессор арқылы басқарылады, тіптен СҚ мен орталық жады арасынды берілетін дерек алдымен процессорға алынып, содан кейін ғана ол керекті құрылғыға беріледі.

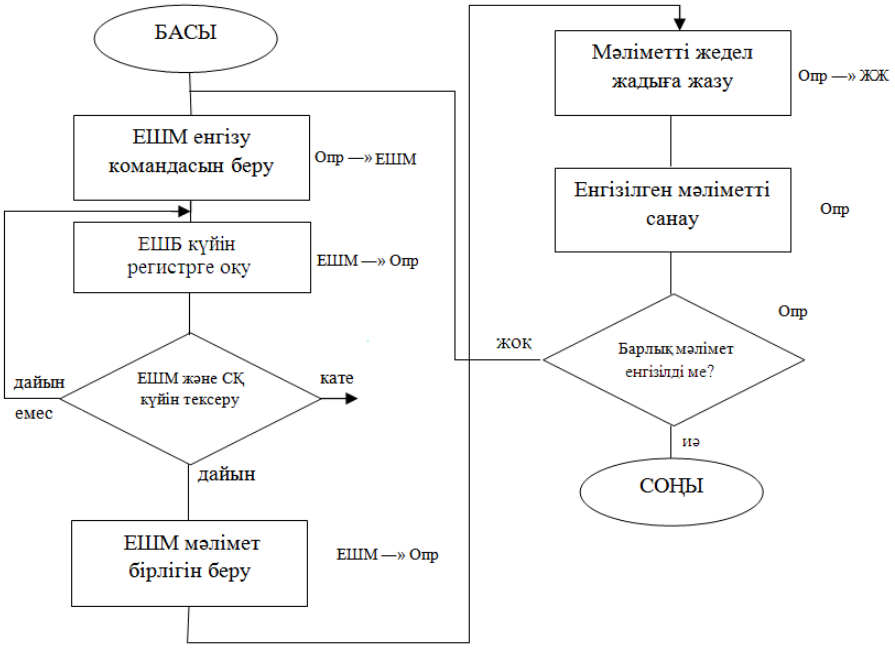
Сұратым арқылы енгізу/шығару.

Мұндай тәсілде орталық процессор енгізу/шығару командасы арқылы ЕШМ СҚ операция орындалу керектігін хабарлайды. Команда құрамында ЕШМ және оған қосылған СҚ мекен-жайы көрсетіледі. Сырт құрылғы дерекпен алмасуға дайын болса, онда модуль күй регистрінің белгілі разрядына дайын екендігін білдіретін «1» жазып қояды. Егер СҚ операцияға дайын болмаса, онда күй регистрінің разряды «0» күйінде қалады.

Процессор дерекпен алмасу алдында ЕШМ күй регистрларына сұраным жіберіп, одан дерек алып оны талдап отыруға тиіс. 10.3-суретте сұраным арқылы орындалатын енгізу операциясының орындалу тәртібі келтірілген.

Енгізу/шығару операциясына төрт түрлі командалар кіреді: басқару, тексеру, оқу және жазу командалары. Басқару командалары арқылы нақты СҚ-ның қандай әрекеттер жасау керектігін көрсетеді, мысалы, магниттік таспаға таспаны кері орау керектігі немесе таспаны ары қарай бір жазбаға жылжыту керектігі көрсетіледі.

Тексеру командасы ЕШБ мен СҚ әртүрлі күйлерін тексереді. Олардың жұмысқа дайын еместігі немесе одан бұрын басталған операциялардың аяқталғандығы, алмасу үстінде пайда болған қателіктер т.б осыған ұқсас деректер белгісі ЕШМ күй регистрінің разрядтарына жазылады.



10.3-сурет. Сұратым арқылы орындалатын енгізу операциясының орындалу тәртібі

Оқу командасы мен ЕШМ СҚ-дан дерек элементін алып, оны дерек регистріне жазады. Орталық процессор ЕШМ-ге қатынас құру үстінде осы дерек элементін дерекшинасына орналастырады.

Жазу командасы арқылы модульь дерек элементтерін дерекшинасынан алып оны СҚ-ға беру үшін дерек регистріне жазады.

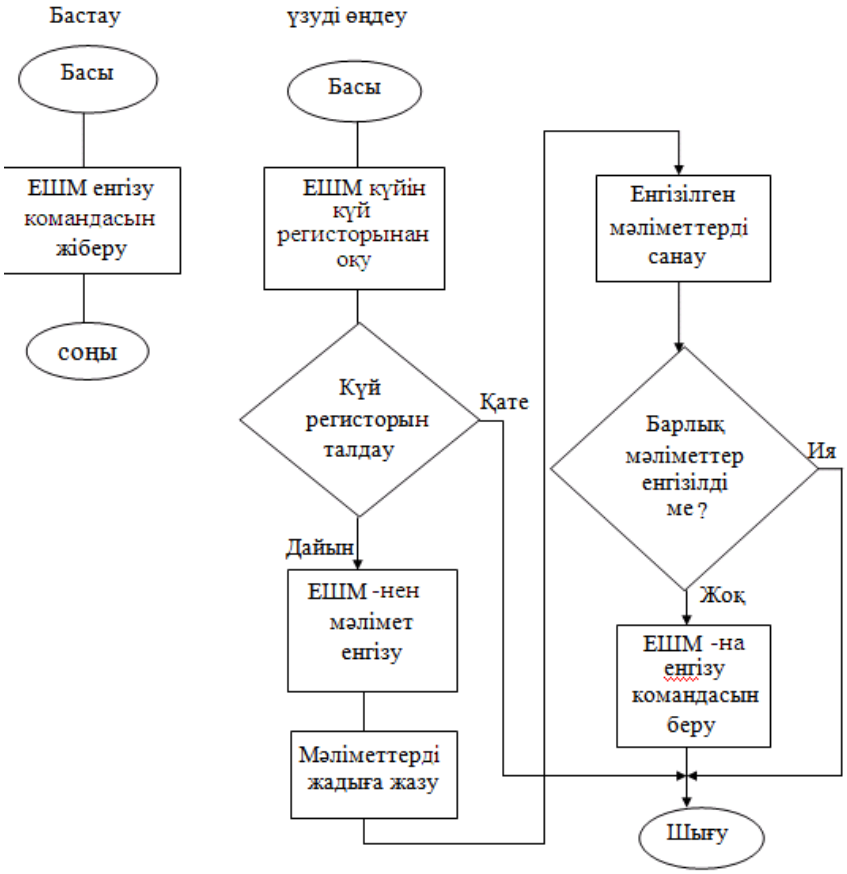
Сұраным арқылы басқарылатын енгізу/шығару операциясында орталық процессордың қолданылуы өте тиімсіз. Ол арқылы ЕШМ дайындығы тұрақты түрде сұраным сигналдары арқылы бақылану керек. Жазу, оқу командаларын орындау үшін де процессорға бірнеше қарапайым командаларды орындауға тура келеді. Оның үстіне ЕШМ күй регистрлерінің разрядтарын талдау да процессорға жүктеледі. Бірақ сұраным арқылы басқарудың өз артықшылықтары бар. Енгізу/шығару операциясын процессор өзі басқаратын болған соң, ЕШМ құрамы өте қарапайым; жүйеге қосылатын ЕШМ сандарын жеңіл өзгертуге болады.

Енгізу/шығару операциясын үзу арқылы басқару.

Мұндай басқаруға орталық процессор ЕШМ тек енгізу/шығару командасын береді де, ол өзінің негізгі жұмысын жалғастыра береді (СҚ алмасу дайындығын тексермейді). СҚ алмасуға дайын болған жағдайда, ол ЕШМ арқылы орталық процессорға үзу сұратымын жібереді. Орталық процессор сұратымды алып кезекті дерек элементін СҚ-ға жіберіп, одан соң үзген бағдарлама командаларын орындауға кіріседі.

10.4–суретте үзу арқылы енгізу операциясының орындалу тәртібі келтірілген. Суреттен көрініп тұрғандай, орталық процессор ЕШМ-ге енгізу командасын беріп өзі басқа бағдарламаны орындауға кірісіп кетеді. ЕШМ процессор командасын алып СҚ-дан деректер енгізуге кіріседі. СҚ-дан оқылған дерек модульдің PrM регистріне қабылданған соң, модуль басқару желісі арқылы үзу сигналын қалыптастырады. Одан соң ЕШМ үзу сұратымын процессорға жіберіп, деректі оның шинасына жіберіп келесі енгізу-шығару операциясы дайын болады. Орталық процессор әр команда циклінің соңында үзу сигналдары түскендігін тексеріп отырады. ЕШМ мұндай сигнал келгенде орталық процессор ағымды бағдарламаның контекстін стекте сақтап, үзу сигналын өңдеуге кіріседі: ЕШМ және СҚ күй регистріндегі деректер талданады. Егер оларда қате болмаса, онда орталық процессор деректі модулььден оқып, оны жедел жадыға жазады.

Одан соң орталық процессор үзілген бағдарламаның контекстін қайта орнына келтіріп оны орындауға кірісіп кетеді.



10.4-сурет. Үзу арқылы деректі енгізу.

Енгізу/шығару операциясын үзу арқылы басқару тәсілінде модульге берілген енгізу/шығару командасынан кейін орталық процессор СҚ-дан келетін сигналдарды күтіп босқа уақыт жоғалтпайды. Бұл тәсілде де жедел жадыдан

модульге жіберетін немесе кері берілетін дерек бірінші тәсілдегі сияқты орталық процессор регистрлері арқылы өтеді.

Жадыға тікелей қатынас құру.

Енгізу/шығару операциясын жедел орындау үшін және енгізу/шығару операцияларына процессорға түсетін жүктемені азайту үшін жадыға тікелей қатынас құру (ЖТҚҚ) тәсілі кеңінен қолданылады. Мұндай тәсілді іске қосу үшін жүйелік шинаға қосымша модуль – жадыға тікелей қатынас құратын контроллерді (ЖТҚҚ) қосу керек. ЖТҚҚК процессордың енгізу/шығару операцияларын орындауға керекті барлық функцияларын өзіне алып, деректерді ЖЖ мен СҚ тікелей алмасуына мүмкіндік туғызады. Дерек блогын оқу немесе жазу үшін ЖТҚҚК-ге керекті ақпараттар берілуі керек. Контроллерге берілетін ақпараттарға мынадай төрт параметрлер кіреді.

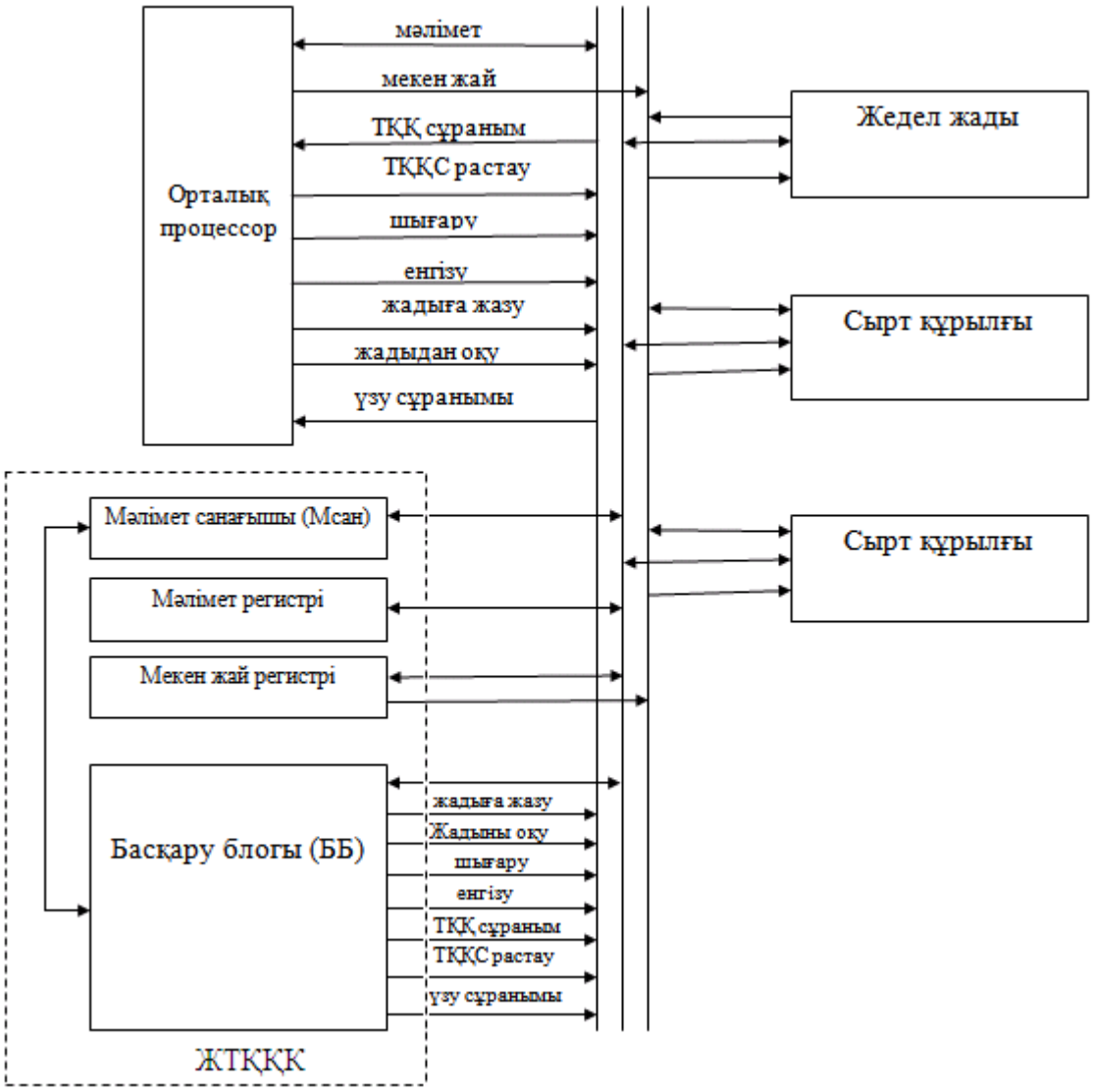
- сұрату түрі;
- енгізу/шығару құрылғысының мекен жайы;
- деректі оқуға немесе жазуға көрсетілетін жады блогының бастапқы ұяшық нөмірі;
- оқуға және жазуға тиісті сөз саны.

Сұратым түрі алмасу бағытын көрсетеді. Оқу сұратымында НЖ құрылғысынан дерек оқылып СҚ-ға берілсе (жадыдан оқу) жазу сұранымында СҚ дерек НЖ беріледі (жадыға жазу). Сұраным түрі ЖТҚҚ басқару блогында (ББ) сақталады. ТҚҚК-не әдетте бірнеше СҚ қосылады. Алмасу операциясына қатысатын СҚ мекен жайы контроллердің басқару блогында сақталады (10.5-сурет).

Бастапқы мекен-жай контроллердің мекен-жай регистрінде сақталады. Алмасу үстінде әрбір дерек бірлігі (сөз) PrA автоматты түрде бірге өсіріледі, яғни мұнда НЖ келесі мекен-жай қалыптасады. Алмасудағы блок сөздер саны контроллердің дерексанауышына (МСАН) беріледі. Деректің әр сөзін бергеннен кейін дерексанауыштың мәні бірге азайтылады. Дерексанауыш«0» мәнін алғанда дерек блогын беру аяқталады. Дерек алмасу орталық процессордан немесе сырт құрылғылардың бірінен бастау алуы мүмкін. Ол туралы контроллерге СҚ бірінен сигнал келеді. Сигналды алып контроллер орталық құрылғыға тікелей қатынас құру сұратымын (ТҚҚС) шығарады. Оған жауап ретінде орталық процессор мекенжай, дерек басқару шиналарын босатады. Келесі циклде орталық процессор ТҚҚ контроллеріне ТҚҚ растау сигналын (ТҚҚС РАСТАУ) шығарады. Бұл сигнал арқылы жүйелік шина толығымен сұратым жіберген құрылғының бақылауына өтеді. СҚ дерекпен алмасуына мүмкіндік алады.

Блоктың әрбір сөзін беру екі кезеңнен тұрады. Оқу операциясын орындағанда (ЖЖ–СҚ) бірінші кезекте контроллер мекен-жай регистрінен ЖЖ ағымды ұяшығынан оқылған сөз дерекшинасына орналастырлады. СҚ мекен

жайы беріп шығару сигналын қалыптастырады. Шығару сигналы дерекшинасынан сөзді СҚ беруді қамтамасыз етеді.



10.5– сурет. Жадыға тікелей қатынас құруды ұйымдастыру.

Жазу операциясын орындағанда (СҚ-ЖЖ) контроллер мекен-жай шинасына алдымен СҚ мекен-жайын беріп, ЖАЗУ сигналын қалыптастырады. ЖАЗУ сигналы арқылы деректер дерекшинасына беріледі. Екінші кезекте контроллер мекен-жай шинасына ЖЖ ұяшық мекен-жайын орналастырып ЖЖАЗУ сигналын шығарады. ЖЖАЗУ сигналы арқылы дерекшинасынан мекен-жай шинасында көрсетілген код арқылы ЖЖ ұяшығына жазылады.

Деректер алмасу үстінде дерек регистрі (МРг) ЖЖ мен СҚ арасында аралық жады қызметін атқарып, екі жақтан келген деректер жылдамдығын сәйкестендіреді.

Әрбір сөз берілген соң басқару блогы мекен-жай регистрінің мәнін бірге өсіріп, жедел жадының келесі ұяшығының мекен-жайын қалыптастырады. Одан басқа басқару блогы дерексанауыш (МСАН) мәнін бірге азайтып отырады

(берілген сөздерді санап отырады). Деректер алмасып болған соң (МСАН=0) контроллер ТҚК сұратым сигналын алып тастайды, оған жауап ретінде орталық процессор ТҚҚС РАСТАУ сигналын алып тастап, жүйелік шинасы басқаруды өз қолына алады. Сонымен орталық процессор енгізу/шығару операциясына тек оның басы мен соңында ғана араласады.

Жадыға тікелей қатынас құру тиімділігі жүйелік шина арқылы деректерді беру режиміне байланысты болады. Олар: блокпен беру, циклді өткізіп беру және ашық режимде беру.

Деректерді блокпен беру режимінде контроллер жүйелік шинаны толығымен басқарады, оны дерек блогының барлық сөздерін беріп болғаннан кейін босатады. Процессор блоктың барлық сөздері беріліп болғанша шинаға қатынас құра алмайды.

Циклді өткізіп беру режимінде жүйелік шинаны контроллер мен орталық процессор цикл сайын кезекпен пайдаланады.

Ашық режимінде контроллер жүйелік шинаны тек орталық процессордан бос циклдарында ғана басқара алады.

10.5.Енгізу/шығару арналары мен процессорлары

Енгізу/шығару процессорлары өнімділігі жоғары әмбебап есептеу жүйелерінде қолданылады. Мұндай жүйелерде енгізу/шығару амалдары орталықтандырылып, бағдарламамен басқарылатын арнайы енгізу/шығару процессорлары арқылы ұйымдастырылады.

Енгізу/шығару бағдарламалары деректер алмасуды жеңілдететін әртүрлі арнайы командалар жиынтығынан тұрады. Егер осы бағдарламалар орталық процессорының негізгі жадында сақталса, онда ондай ЕШМ енгізу/шығару арнасы (ЕША) деп аталады. Егер ЕШМ қосылған барлық СҚ енгізу/шығару бағдарламалары арнаның жергілікті жадында сақталса, онда ЕШМ енгізу/шығару процессоры (ЕШП) деп аталады.

Есептеу жүйесінің құрамына қосылған енгізу/шығару арнасы өте күрделі процедураларды іске асыра отырып, орталық процессорды енгізу/шығару операциясына қатысуынан толығымен босата алады. Енгізу/шығару операциясын орындау кезінде орталық процессор, ЕША енгізу-шығару бастау, тоқтату, арна күйін және оған қосылған СҚ күйлерін тексеру деген тек бірнеше (4-тен 8-ге дейін) командалар береді.

ЕША енгізу/шығару операцияларын арналық бағдарламалар арқылы іске асырды. Әртүрлі сырт құрылғыларының дерекпен алмасу ерекшеліктеріне байланысты құрылғыдан бағдарламалар есептеу жүйесінің негізгі жадында

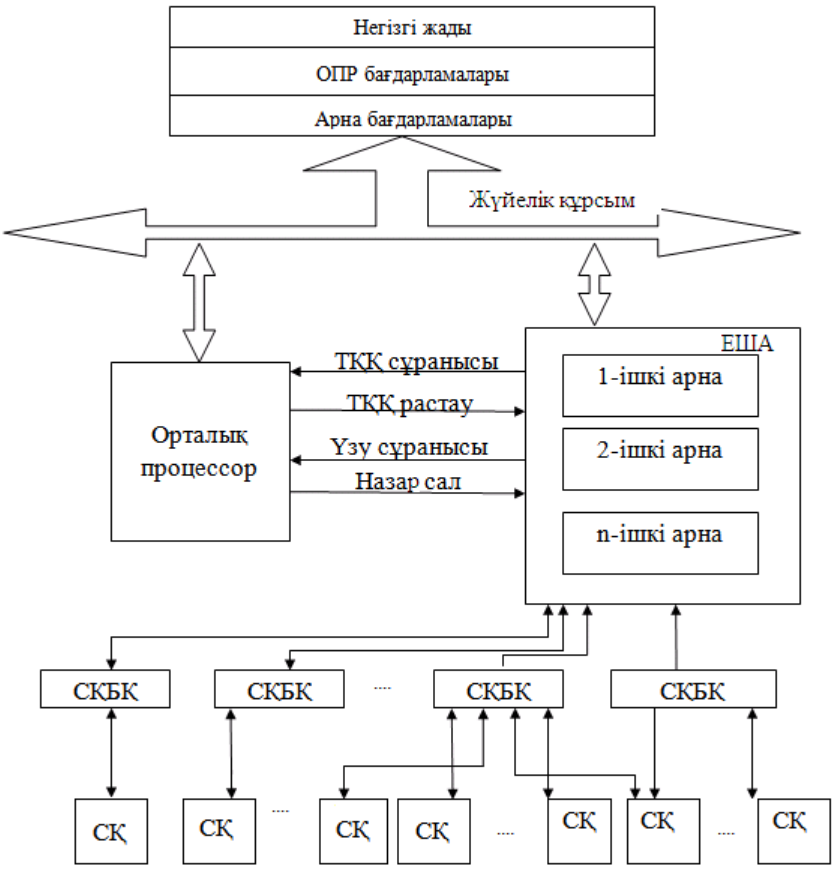
немесе арнаның жергілікті жадында сақталады. Арналық бағдарламаларда команда міндетін арнаның басқару сөзі (АБС) атқарады. АБС құрамы кәдімгі машинаның құрамынан өзгеше болады.

Типтік АБС құрамына негізінен операция коды, көрсеткіштер, дерек мекен-жайы, дерексанауышы т.б кіреді:

- операция коды ЕША мен оған қосылған СҚ орындалатын операциялардың түрін анықтайды: ЖАЗУ (НЖ →СҚ), ОҚУ (СҚ → НЖ), БАСҚАРУ (магниттік диск, таспа, бастиегін жылжыту т.б.);
- көрстекіштер - күрделі операциялар тізбегін орындауға арналған қосымша тағайындаулар. Көрсеткіштер арқылы енгізу операциясы кезінде жазбалар іріктеліп енгізіледі;
- дерек мекен-жайы операцияға қатысатын жады аймағы көрсетеді;
- дерексанауышы берілетін дерек блогының мәнін санауға арналған.

Деректермен алмасу процессордың бұйрығымен ЕША мен сырт құрылғылар арасында өтеді. Әр құрылғыға өзінің басқару ақпараттары: процессорға енгізу/шығару командасы, арнаға арнаның басқару сөзі (АБС), сырт құрылғыларға бұйрықтар сәйкес келеді. Одан басқа енгізу/шығаруды басқару үшін арна күйінің коды (арна күйінің сөзі) және СҚ күй сөздері қатынасады. Дерек негізгі жады мен ЕША жүйелік шина арқылы (10.6-сурет) алмасады. Сырт құрылғылары арнаға сырт құрылғыларды басқару контроллері (СҚБҚ) арқылы қосылады. СҚБҚ арнадағы әртүрлі бұйрықтарды (оқу, жазу, дерек тасымалдауыштарды немесе магниттік бастиектерді жылжыту т.б) алып, оны СҚ түрлеріне байланысты әртүрлі сигналдарға түрлендіреді. Әдетте бір СҚБК бірнеше СҚ басқарады. Егер СК өте жылдам жұмыс жасайтын болса, онда олардың әрқайсысы өзінің контроллерімен басқарылады.

Бір ЕША ішінде СҚ өзінің ішкі арнасына қосылады. Әрбір ішкі арнаның бірегей логикалық нөмері болады. Осы нөмір арқылы бағдарлама сырт құрылғыларының біріне мекенделеді. Физикалық тұрғыдан ішкі канал сырт құрылғының бірімен орындалатын енгізу/шығару операциясының параметрлерін сақтайтын жобаның бөлігі болып табылады. Мұндай параметрлерге ағынды мекенжай дерексанауыш мәндері, енгізу/шығару операциясының коды және көрсеткіштері, келесі АБС т.б. жатады. Параметрлерді сақтау үшін арнайы жергілікті жады пайдаланылады. СҚ мен НЖ жадыға тікелей қатынас құру режимінде алмасады. Ол үшін ТҚҚ сұранымы мен ТҚҚ растау сигналдары шығарылады.



10.6-сурет. Е/Ш арналы есептеу жүйесі.

Орталық процессор орталық бағдарламаның аяқталғаны, немесе пайда болған қателіктер туралы деректерін ЕША үзу сұранымы сигналы арқылы хабарлайды. Өз кезегінде орталық процессор өзіне назар салу үщін «назар сал» сигналын шығара алады.

Алмасу жылдамдығы тұрғысынан барлық СҚ-ны екі топқа бөлуге болады. Бірінші топқа алмасу жылдамдығы жоғары магниттік дискілі және магниттік таспалы жинақтағыштар жатса, екінші топқа алмасу жылдамдығы төмен дисплейлер, принтерлер т.б сырт құрылғылары жатады.

Алмасу жылдамдықтарына байланысты ЕШ арнайы сырт құрылғыларымен екі түрлі режимде - мультиплекстік және монополдық режимдерінде жұмыс жасайды. Мультиплекстік режимде арна жұмыс уақыты бірнеше СҚ-ға бөлінеді, яғни арна мүмкіндігін сырт құрылғылар кезекпен пайдаланады. Мұндай ЕШ арнасын *ЕШ мультиплекстік арнасы* деп атайды. Егер байланыс сеанс кезеңінде бір немесе бір машина сөзін құратын бірнеше байттармен алмаса алса, ондай арнаны *байт-мультиплекстік арна* деп атайды. Егер байланыс сеансы кезінде дерек блокпен алмасса, онда арна блок *мультиплекстік* деп аталады.

Монополды жұмыс тәртібінде арна мен СҚ байланысқаннан кейін арна толығымен осы СҚ-мен жұмыс жасайды. Басқа СҚ байланысу мүмкіндігі тек

байланысқан СҚ алмасу операциясы аяқталған соң ғана туады. Мұндай арнаны *селекторлық арна* деп атайды.

Енгізу/шығару арналары қазіргі 64-разрядты есептеу жүйелері сәулетінде ары қарай дамып отыр. Оған мысал ретінде IBM 370 есептеу жүйелерін және басқа да мэинфреймдерді мысалға келтіруге болады. Мұндай жүйелерде көптеген арналарды біріктіріп жеке енгізу/шығару арналарының ішкі жүйесін құрып отыр. Мұндай ішкі жүйеде бір немесе бірнеше арнайы енгізу/шығару процессорлары бар. СҚБҚ бірнеше арналарға қосылу арқасында алмасу арналары олардың жүктемелік деңгейіне байланысты динамикалық түрде өзгеріп отырады. Қазіргі есептеу жүйелерінің ішкі арналарына 256 арналық тракттар бар. Олардың әрқайсысына 256 СҚ қосуға болады.

Бақылау сұрақтары

1. Енгізу/шығару жүйесін процессорға қосудың үш тәсілдерінің артықшылығы мен кемістіктерін атаңыз?
2. ЕШМ негізгі атқаратын қызметтерін атаңыз?
3. Мекен жай сұрыптауышы қандай қызмет атқарады?
4. Сұраным тәсілімен басқарылатын және үзу арқылы басқарылатын енгізу/шығару операцияларын салыстырып, олардың артықшылықтары мен кемшіліктерін айтыңыз
5. Жадыға тікелей қатынас құру үдерісінде дерексанауышы не үшін керек?
6. Жадыға тікелей қатынас құру үдерісінде контроллерге қандай деректер беріледі?
7. Енгізу/шығару арналары мен процессорларында енгізу/шығару операциялары қандай бағдарламалар арқылы басқарылады?
8. Енгізу/шығару арналары мен процессорларында сырт құрылғыларымен қандай режимде жұмыс жасайды?

11.ПРОЦЕССОРЛАРДЫ ҰЙЫМДАСТЫРУ

Жоғарыда есептеу машиналарының операциялық құрылғылары, басқару құрылғылары, ішкі жадылар және т.б. орталық процессордың құрамдас бөліктері болып табылатын құрылғыларға тоқтадық.

Бұл тарауда орталық процессор жұмысының өнімділігін арттыруға алып келетін процессордың құрамдас бөліктерін әртүрлі жолмен ұйымдастыру тәсілдері қаралады. Олардың қатарына командаларды конвейерлеу, суперскалярлық процессорларды құру, CISC, RISC, VLIW архитектурасын қолдану, көпядролы процессорларды ұйымдастыру тәсілдері жатады.

11.1 Командалар конвейері

Командаларды конвейерлеу идеясын 1956 ж. академик С.Л.Лебедев ұсынған.

Жоғарыда қаралғандай, команда циклі әртүрлі кезеңдер тізбегінен тұрады. Егер команданың әр кезеңдері жеке дара құрылғылар арқылы орындалатын болса, онда мұндай құрылғыларды тізбектеп қосу арқылы команда конвейерін аламыз. 11.1-суретте жеті кезеңнен тұратын команда цикліне сәйкес құрылған 7 сатылы конвейер келтірілген:

1. КІ - команданы іріктеу;
2. КД- команданы декодтау;
3. ОМЖА - орындалатын мекен-жайын анықтау;
4. ОІ - операндтарды іріктеу;
5. ОО - Операцияны орындау;
6. НЖ - нәтижені жазу;
7. ККМЖҚ - келесі команда МЖ қалыптастыру;

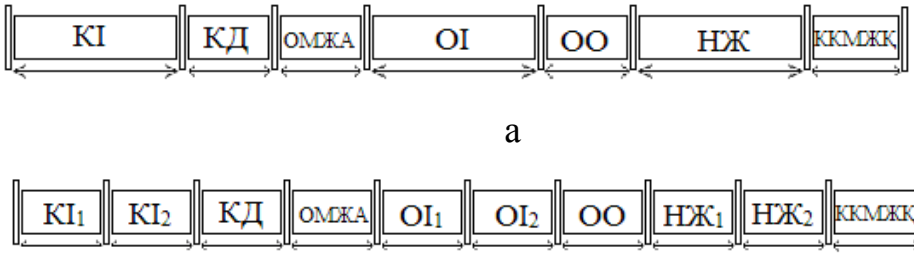
	1	2	3	4	5	6	7	8	9	10	11
1 команда	КІ	КД	ОМ ЖА	ОІ	ОО	НЖ	ККМ ЖҚ				
2 команда		КІ	КД	ОМ ЖА	ОІ	ОО	НЖ	ККМ ЖҚ			
3 команда			КІ	КД	ОМ ЖА	ОІ	ОО	НЖ	ККМ ЖҚ		
4 команда				КІ	КД	ОМ ЖА	ОІ	ОО	НЖ	ККМ ЖҚ	
5 команда					КІ	КД	ОМ ЖА	ОІ	ОО	НЖ	ККМ ЖҚ

11.1-сурет. Жеті кезеңнен тұратын команда цикліне сәйкес құрылған 7 сатылы конвейер

Конвейерсіз орындалатын 5 команда үшін 5*7=35 уақыт бірлігі керек. Ал конвейер арқылы орындалғанда 5 командаңың орындалуы үшін 11 уақыт бірлігі жеткілікті екендігіне көз жеткізу қиын емес.

Конвейер тиімділігі оның кірісіне берілетін деректер жиілігіне тәуелді болады. Әдетте конвейер кірісіне берілген деректер жиілігі конвейер тізбегіндегі ең “баяу” жұмыс жасайтын конвейер сатысымен шектеледі. Сондықтан деректер жиілігін жоғарылату үшін “баяу” жұмыс жасайтын конвейер сатысын анықтап, оны бірнеше сатыға бөліп, олардың әрқайсысының жылдамдықтарын қалған конвейер сатыларының жылдамдықтарымен теңестіреді.

Мұндай жолмен құрылған конвейерлерді *суперконвейерлер* деп атайды. 11.1-суретте 7 сатыдан тұратын командалар конвейері көрсетілген. Суреттен көрініп тұрғандай, КІ, ОІ, НЖ сатыларындағы орындалатын әрекеттерге қарағанда екі есе “баяу” (бұл жады құрылғысына қатынас құру уақытымен түсіндіріледі). Суретте әрбір конвейер сатыларының арасында орналасқан аралық регистрі көрсетілген. Процессордың такт жиілігі “баяу” жұмыс жасайтын КІ, ОІ, НЖ сатыларының жылдамдығымен анықталады.



11.2-сурет. а - “баяу” сатылары бар конвейер; б – суперконвейер

Конвейер сатыларының жылдамдығын өсіру арқылы процессордың өнімділігін арттыруға мүмкіндік туады. Суперконвейердің көрсеткішіне команда конвейерінің саты саны жатады. Бірінші суперконвейерлі процессорға MIPS R4000 процессоры жатады. Мұндай процессорда команда конвейері 8 сатыдан тұрады. Суперконвейерлеу кезінде команданы іріктеу (КІ) және операндтарды іріктеу (ОІ) кезеңдері бөлінеді.

Суперконвейерлеу ұғымымен қатар “гиперконвейерлеу” ұғымы пайда болды. Гиперконвейерлік процессорға PentiumIV процессоры жатады. Мұндай процессорда команда конвейері 20 сатыдан тұрады. Конвейер саты саны көбейген сайын олардың әрекеттесу логикасы күрделенеді. Конвейер сатыларының әрекеттесуінің логикасы күрделіленеді. Сондықтан Intel фирмасының соңғы модельдерінде (Core 2 процессоры) конвейерлік саты саны 14-ке дейін қысқартылған.

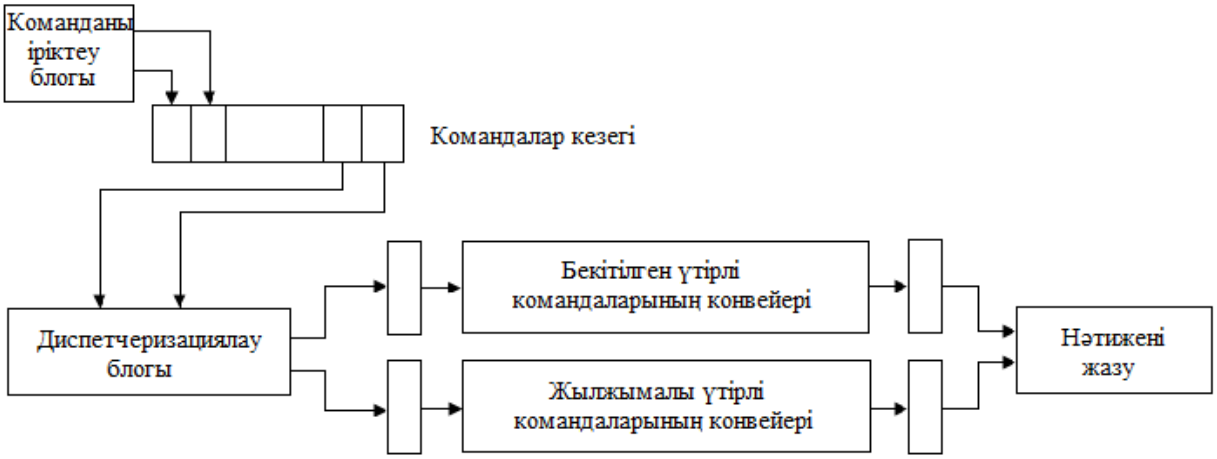
11.2. Суперскалярлық процессорлар

Суперскалярлық процессорда бірнеше командалар жеке операндтармен (скалярлық операндтармен) параллель орындалады. Ол үшін орталық процессордың құрамына бірнеше функционалдық блоктар енгізіледі. Мұндай блоктарда белгілі бір операциялар тобы орындалады және олар процессорда бірнеше дана болуы мүмкін. Мысалы, Intel Nehalem процессоры 64 разрядты бүтін санды өңдейтін 3 блоктардан (ALU), жылжыту (Shift) жане салыстыру (LEA) блоктарынан, жылжымалы үтірлі сандарды өңдейтін (FADD, FMUL, FPShuttes) 3 блоктардан, 128- разрядты сандарды (SSE) өңдейтін 3 блоктардан, өту операциясын (BRANCH) орындайтын бір блоктан, бөлу блогынан (Divide) және бүтін сандармен күрделі операциялар (Complex Integer) орындайтын блоктардан тұрады.

Суперскалярлық процессорларда бірнеше функционалдық блоктар параллель жұмыс жасап, бірнеше скалярлық командалар орындалады.

Суперскалярлық процессорлардың құрамына бірнеше командалық конвейерлер болады. Мұндай жағдайда барлық конвейерге ортақ команданы іріктеу сатысында әрбір конвейерлік тактта жады құрылғысынан бірнеше командалар оқылады. Команданы іріктеу сатысынан кейін диспетчерлеу блогы орындалады. Диспетчерлеу блогы командаларда конвейерлік блоктарға үлестіріледі.

11.3-суретте құрамында 2 конвейері бар процессор құрылымы келтірілген. Конвейердің біріншісінде бекітілген үтірлі сандарды өңдейтін құрылғы орналасқан, екіншісінде жылжымалы үтірлі сандарды өңдейтін конвейер көрсетілген.



11.3-сурет. 2 конвейерлі суперскалярлық процессор.

Командаларды іріктеу блогы бағдарлама командаларын алдын ала оқып, командалар регистріне сақтап отырады. Диспетчерлеу блогы осы

командалардың ішінен екеуін (жылжымалы үтірлі командасын және бекітілген үтірлі командасын) таңдап, оларды бір уақытта конвейерлерге жүктейді.

Қазіргі процессорлар құрамы функционалдық блоктар кешенінен тұрады. Олардың әрқайсысы әртүрлі операцияларды орындауға арналған. Мұндай жағдайда командалар конвейерге олардың түріне қарамай жүктеледі, ал диспетчер командаларды тиісті функционалдық блоктарға үлестіреді.

11.3. Процессорлар архитектурасы

Қазіргі есептеу машиналарында архитектурасы әртүрлі процессорлар қолданылады. Олардың қатарына CISC (Complex Instruction Set Computer-команда жиынтығы, толық компьютер), RISC (Reduced Intruction Set-Computer-команда жиынтығы, ықшамдалған компьютер) және VLIW (Very Long Instruction Word- командалық сөзі өте ұзын) процессорлары жатады. Олар бір-бірінен төмендегі көрсеткіштерімен өзгешеленеді:

- процессор регистрлерінің разрядтылығымен және олардың атқаратын қызметтерімен;
- жадыға қатынас құру түрлерімен;
- бір команда арқылы орындалатын операция санымен;
- команда ұзындығымен (тұрақты, өзгермелі);
- деректер типінің санымен өзгешелінеді.

CISC процессорлары.

CISC архитектурасының процессорлар алғашқы компьютердің процессорына жатады. Мұндай архитектурада деректерді өңдеу кезінде орындалатын әрбір типтік амалдарға жеке машиналық команда меншіктелген. Процессор жетілдірілген сайын команда саны да көбейе түсті. Ол бағдарлама құруды жеңіл еткенімен, оның процессор құрылымын күрделілендірді. Нәтижесінде CISC процессорларында мынадай жағдай қалыптасты:

- әрбір машиналық команданы орындау үшін бірнеше процессорлық тактар керек болды;
- басқару құрылғысы бағдарламаланатын логикамен ұйымдастырылды;
- жалпы мақсатты регистрлер саны үлкен болмады;
- ұзындығы әртүрлі және әртүрлі командалар пішімі орын алды;
- негізінен екі мекен-жайлы командалар орын алды;
- әртүрлі мекендеу (әсіресе жанама мекендеу) тәсілдері орын алды.

CISC – процессорда кейбір командалар таза аппараттық тәсілмен орындалу мүмкін емес. Олар бірнеше қарапайым командалар тізбегімен орындалады. Көптеген күрделі командалар бағдарлама құру кезінде пайдаланылмайды. Күрделі командалардың орындалуын басқару үшін баяу жұмыс жасайтын тұрақты жады негізінде құрылатын бағдарламалық логикалық басқару құрылғысы қолданылды. Мұндай тұрақты жады процессордың такт жиілігін өсіруге шек қояды. Осы айтылған жайттар RISC процессорлады дамытуға алып келді. Осымен қатар CISC процессордың бағдарлама құрудағы артықшылығын ескере отырып, көптеген өндіруші фирмалар (Intel, AMD, IBM, т.б.) CISC процессорын ары қарай дамытып отыр.

Бұл бағытта CISC процессорларындағы күрделі командаларды декодтау кезінде олар бірнеше бөліктерге бөлініп RISC процессордың қарапайым командаларымен алмастырылады. Ал қарапайым RISC командалары апараттық тәсілмен орындалады.

RISC процессорлары

RISC процессорларының CISC процессордан негізгі айырмашылықтарына төмендегі көрсетілген ерекшеліктер жатады:

- RISC процессордың қарапайым командалары негізінен бір циклде орындалады;
- Барлық командалар конвейерлік тәсілмен орындау үшін олардың ұзындықтары стандартты бір сөзді, ал деректер шиналарының ені бірдей;
- Командалар саны, олардың пішімдері, командалардың мекендеу тәсілдері шектеулі;
- “жазу”, “оқу” командаларынан басқа алмасулар процессордың ішкі регистрлерінде жүргізіледі;
- Басқару құрылғысы жоғары жиіліктегі апараттық логикамен іске қосылған;
- Процессордың регистрлер саны (жалпы мақсатты регистрлер, регистрлік файлдар) жоғары.

VLIW процессоры

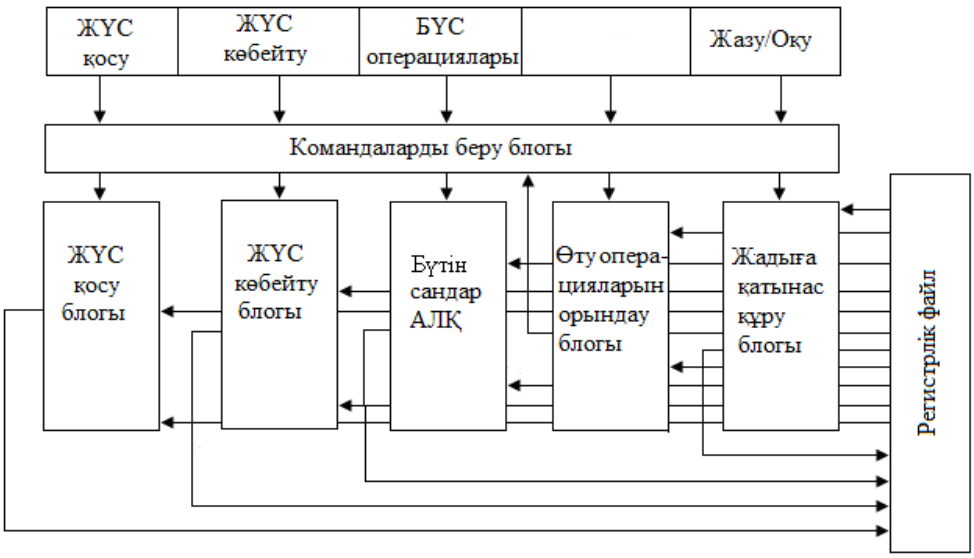
VLIW (Very Long Instruction Word - командалық сөзі өте ұзын) процессорының негізінде орындалатын командаларды тиімді ретпен жоспарлап, оларды параллель орындауға даярлау үдерісі жатыр. Мұндай үдерісті жүргізу “ақылды” бағдарлама құрастырғышына жүктеледі. Мұндай құрастырғыш бағдарлама орындалатын бағдарлама командаларын алдын ала талдайды. Талдау үстінде командалардың ішінен қатар орындауға болатын командалар анықталады. Келесі кезеңде бір біріне қақтығыспайтын, яғни қатар

орындауға болатын командалар бір дестеге (байламға) біріктіріледі. Мұнда деста ұзындығы үлкен бір команда ретінде қаралады. Қарапайым командаларды ұзындығы үлкен бір командаға біріктіру төмендегі ереже бойынша сақталады:

- Ұзындығы өте үлкен команда пішімі арқылы команда өрістерінде көрсетілген операцияларды орындау блоктарымен өзара байланыстырады;
- Бір командаға біріктірілген қарапайым командалар саны процессордың құрамына кіретін функционалдық блоктардың (ФБ) санымен анықталады;
- Ұзындығы үлкен команда құрамына функционалдық блоктарымен орындалатын қарапайым командалар ғана кіреді. Ұзындығы үлкен команда құрамына кіретін барлық қарапайым командалар қатар орындалады.

Ұзындығы үлкен команда әдетте 256-дан 1024-ке дейінгі биттерден тұрады. Мұндай мета команда қарапайым командалар санымен анықталатын бірнеше өрістерден тұрады. 11.4-суретте ұзындығы үлкен командаға мысал және оның өрістерінің функционалдық блоктарымен байланысы көрсетілген.

Ұзындығы үлкен команданың құрамына кіретін қарапайым команда ретінде RISC командаларын қолдануға болады. Ұзындығы үлкен команданың өрістер саны функционалды блоктар санымен анықталады. Әдетте олардың саны 3-тен 20-ға дейін болады. Барлық функционалдық блоктар ортақ көп портты регистрлік файлдағы деректерге қатынас құра алады.



11.4-сурет.Ұзындығы өте үлкен команда пішімі.

VLIW процессорын статикалық суперскалярлық процессер деп қарауға болады. Олардың айырмашылығы: VLIW процессорда кодты параллельдеу динамикалық орындау кезінде емес, құрастыру кезеңінде жүргізіледі.

VLIW процессорында регистрлік файлды құру күрделі. Әрбір есептеу блоктары файлдармен байланыста болу керек. Екіншіден, бір-бірімен тәуелді емес командаларды табатын құрастырғыш бағдарламасын құру керек.

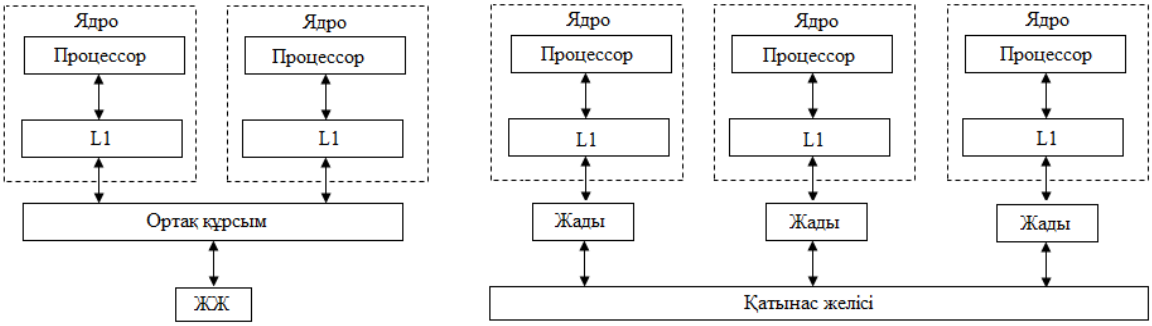
11.4. Көпядролық процессорлар

Есептеу машинасының өнімділігін арттыру үшін кейінгі кезде көпядролық процессорлар (КЯПР) дүниеге келді.

Көпядролы процессор бір кристалда немесе бір тақтайшада орналасқан екі немесе одан да көп есептеу ядроларының жиыны. Ядро процессор мен КЭШ жадыдан (әдетте бірінші деңгейлі L1-КЭШ жады) тұрады.

Бірнеше ядро ортақ негізгі жадымен ортақ шина арқылы немесе қатынас желісі арқылы байланысады. (11.5-сурет) Оларды таңдау негізгі жадыны ұйымдастыру тәсіліне байланысты болады. Егер негізгі жады барлық процессорларға ортақ болса, онда процессорлар негізгі жадымен ортақ шина арқылы байланысады.

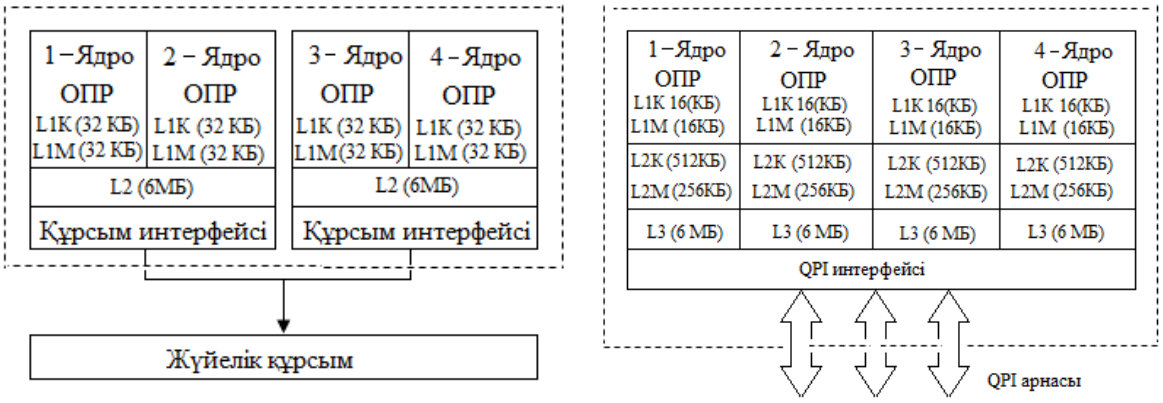
Егер негізгі жады үлестірілген болса, онда процессор ядролары бір-бірімен қатынас желісі арқылы байланысады. Ортақ шина арқылы байланыс тәсілінде көпядролы процессордағы ядро саны 32-ден аспайды, өйткені ядро санын одан артық көбейтсек копядролы процессордың өнімділігі күрт төмендейді.



11.5-сурет. Көпядролы процессорлар (КЯПР): а - ортақ жадылы; б - үлестірілген жадылы.

Көпядролы процессорларды құрудың әр түрлі жолдары бар.

Олар бір-бірінен құрамындағы КЭШ-жадыларының түрлеріне - бірінші деңгейлі команда КЭШ-жады (L1K), бірінші деңгейлі дерек КЭШ-жады (L1M), екінші деңгейлі команда (L2K) және дерек (L2M) КЭШ жадылары және үшінші деңгейлі L3 - байланысты бөлінеді. КЭШ жадылары пайдалану тәсілдеріне байланысты әртүрлі болып келеді. 11.6-суреттерде мысал ретінде Intel Core 2 Quad (а) және Intel Itanium 3 Tukwila (б) КЯПР құрамдары келтірілген.



а) б)

11.6-сурет. Көпядролы процессорлар құрылымдарына мысал.

Бақылау сұрақтары.

1. Суперконвейерлеу идеясын түсіндіріңіз. Суперконвейердің артықшылығы мен кемшілігі неде?
2. Суперскалярлық конвейердің артықшылығы неде?
3. CISC – процессордың пайда болуы неге байланысты болды? Қандай көрсеткіштерімен өзгешеленеді?
4. RISC – процессордың пайда болу себептерін түсіндіріңіз. Қандай көрсеткіштермен ерекшеленеді?
5. VLIW – процессорда қарапайым командаларды бір ұзындығы үлкен командаға біріктіру ерекшелігі қандай?
6. Көпядролы процессорларда ядроның құрамы қандай?
7. Көпядролы процессорларда ядролар саны немен шектеледі?
8. Көпядролы процессорларда ядролар санын арттыру үшін бір бірімен қандай орта арқылы байланысу керек?
9. Көпядролы процессорларда КЭШ жадының қандай түрлері қолданылады?

12. ПАРАЛЛЕЛЬ ӨҢДЕУ - ӨНІМДІЛІКТІ ӨСІРУДІҢ АРХИТЕКТУРАЛЫҚ ТӘСІЛІ

Фон Нейман машиналарында бағдарламалар командалары тізбекпен орындалатыны белгілі. Командалары тізбекпен орындалатын мұндай машиналар арқылы олардың өнімділігін күрт өзгерту мүмкіндігі шектеулі.

Есептеу машиналарының өнімділігін арттырудың тиімді жолына бір немесе көппроцессорлар (көпмашиналар) көлемінде бағдарлама командаларын пареллель орындау тәсілдері жатады. Көптеген командаларды параллель орындауға мүмкіндіктері бар есептеу машиналарын *есептеу жүйелері* деп атаймыз. Есептеу жүйесінің ерекшеліктерінде есептерде кездесетін параллель тармақтарын параллель өңдеуге мүмкіндік туғызатын құралдар мен амалдар ескерілген. Мұндай құралдар мен тәсілдер классикалық машиналар құрамында алдын ала ескерілмеген.

Параллель тәсілдері мен құралдары әртүрлі деңгейде іске асырылуы мүмкін.

Микродеңгейде командалар әртүрлі фазаларға бөлінетіні белгілі. Оларды конвейер арқылы параллель өңдеу тиімді. Микродеңгей бір процессорлы ЕЖ іске қосылады.

Команда деңгейінде бірнеше командалар параллель орындалады. Командалар суперскалярлық процессор құрамында орналасқан бірнеше конвейерлер арқылы параллель өңделеді.

Ағындар деңгейінде орындалатын есеп параллель орындауға келетін бөліктерге (ағындарға) бөлінеді. Ағындар деңгейі параллель ЕЖ арқылы іске асырылады.

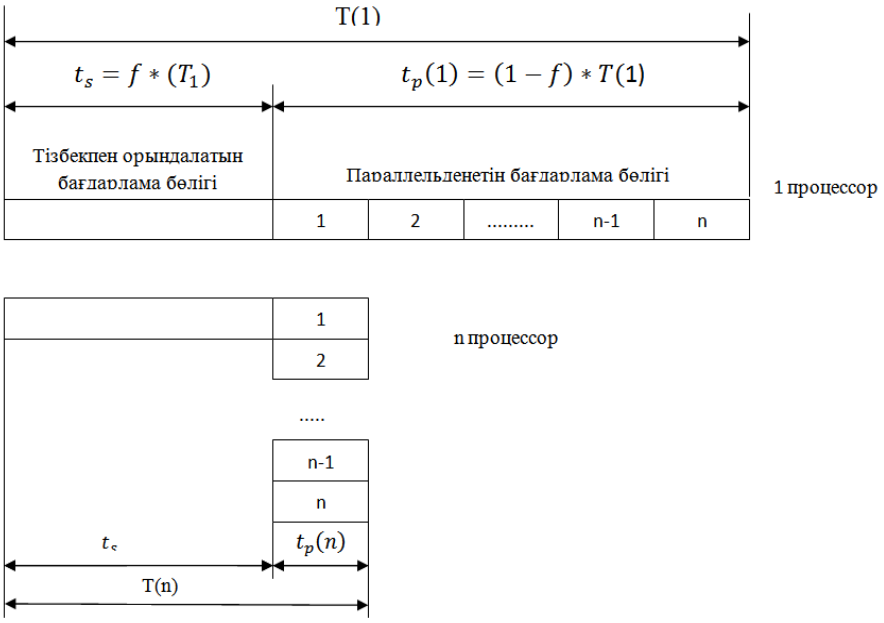
Тапсырмалар деңгейінде бірнеше бір бірімен байланыспаған (тәуелсіз) тапсырмалар параллель әртүрлі процессорларда орындалады. Тапсырмалар деңгейі көппроцессорлы және көпмашиналы ЕЖ-де іске асырылады.

12.1 Параллель есептеулер заңдылықтары

Параллель есептеу жүйесін қолданушылар өз есептерін жүйе құрамындағы параллель жұмыс жасайтын процессорларға жүктеу арқылы есептің шығару жылдамдығын арттыра алады.

Бір қарағанда n процессордан тұратын жүйе арқылы есептеу жылдамдығын n есе өсіруге мүмкіндік аламыз деп ойлаймыз. Бірақ іс жүзінде мұндай нәтижеге жету өте қиын. Ешбір есеп толық параллельдеуге келмейді, өйткені әрбір бағдарламада бір процессормен тізбекпен орындалатын кодалар

фрагменттері міндетті түрде болады. Мұндай фрагменттерге деректерді орналастыруда орын алатын индекстеу есептері, деректерді процессорға үлестіру, енгізу-шығару операциясын орындау т.с.с. жатады. Тізбекпен орындалатын бағдарламалардан қалған бағдарламалар параллельденуі мүмкін. Сонымен орындалатын есептің бағдарламалық коды екі бөліктен: тізбекпен орындалатын және параллельденетін бөліктен тұрады. Бір процессормен тізбекті түрде орындалатын операциялар бөлігін f арқылы белгілейміз, мұнда $0 \leq f \leq 1$. Операциялардың параллельденетін бөлігі 1- f арқылы белгіленеді. Егер $f = 0$ болса, онда операциялар толығымен параллельденетін операциялардан тұрады, ал $f = 1$ болса, онда операциялар толығымен тізбекпен орындалады (12.1-сурет). Суретте t_s −бір процессор арқылы тізбекпен орындалатын бағдарламаның өңделу уақыты; $t_p(1)$– бір процессор арқылы параллельденетін бағдарлама бөлігіне жұмсалатын уақыт; $t_p(n)$–n прцессорлар арқылы параллельденетін бағдарламалық өңдеу уақыты.



12.1-сурет. n-процессорлы есептеу жүйесінде бағдарламаны параллельдеу

Есептеу жүйелерінде есептеу уақытын қысқарту өлшемі ретінде «үдеу» өлшемі алынған. Үдеу (S_n) бір процессорлы жүйе арқылы өңдеуге (тізбекпен орындауға жұмсалатын T (1) уақыттың көппроцессорлар жүйесі арқылы (параллель орындалатын) өңдеуге жұмсалатын уақытқа T (n) қатынасын айтамыз.

$$S_n = \frac{T(1)}{T(n)};$$

Нақты үдеу пайдаланушылардың көппроцессорлы жүйені пайдалану мақсатына байланысты әртүрлі болады. Егер жүйені пайдаланудағы мақсат есеп шығару уақытын қысқарту болса, онда үдеу Амдал заңымен анықталады.

Ал, егер берілген уақыт ішінде шығарылатын есеп көлемін ұлғайтқыңыз келсе, онда оны Густафсон заңдылығымен анықтауға болады. Сана және Ная заңы Густафсон заңына ұқсас, тек мұнда есеп көлемін ұлғайтуда жүйе жады сыйымдылығы еске алады.

Амдал заңы

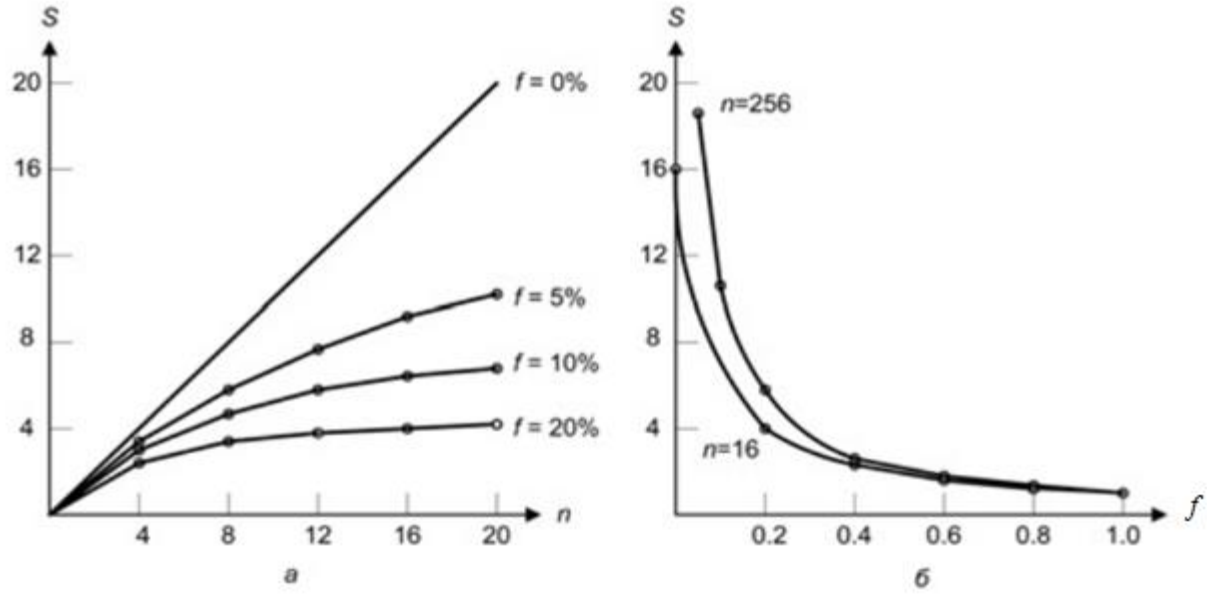
Джин Амдал – 1967 жылы өз жұмысында көппроцессорлы ЕЖ үшін есептеу үдеуінің (S (n)) процессор санына және есеп бағдарламасындағы тізбекті және параллельденетін бөліктерінің ара қатынасына байланысты болатын тәуелділікті көрсететін формуласын ұсынды. Формулада процессор саны өзгергенімен ЕЖ-де шығарылатын есеп көлемі өзгермейді деп қаралған. Формула берілген есеп көлеміне байланысты оның орындалу жылдамдығын анықтауға мүмкіндік береді. Амдал заңының формуласы:

$$S(n) = \frac{T\,(1)}{T\,(n)} = \frac{T\,(1)}{fT(n) + \frac{(1-f)T_1}{n}} = \frac{1}{f + \frac{1-f}{n}}$$

Процессор саны шексіз өскенде

$$\lim_{n\to\infty} S = \frac{1}{f}$$

Формуладан, егер бағдарлама командалары 25% тізбекпен орындалатын болса (f=0.25), онда процессор санын қанша өсіргенмен үдеу төрт еседен артық өспейтінін байқауға болады. 12.2-суретте процессор санына және тізбекпен орындалатын бағдарлама бөліктеріне байланысты үдеу мәндерінің өзгеру графигі келтірілген.



12.2-сурет.Үдеудің тізбекпен орындалатын бағдарлама бөлігіне (а) және процессор санына байланысты өзгеру графигі (б)

Густафсон заңы

Амдал заңы процессор санын көбейту кезінде шығарылатын есеп көлемі өзгермеген жағдайда орын алады. Бірақ іс жүзінде пайдаланушы өз қарамағындағы күрделі жүйе арқылы белгілі бір есептеу уақыты аралығындағы шығарылатын есеп көлемін ұлғайтуға тырысады (мысалы, есептеу дәлдігін жоғарылатады). Бұл жерде ұлғайтылатын есеп бағдарламасы негізінен параллельденетін бөлікке жатады екен. Бұл f мәнін қысқартуға алып келеді екен.

Мұндай жағдайға байланысты Густафсон есептеу үдеуін анықтау үшін мына формуланы ұсынған:

$$S(n) = f + (1 - f)n$$

Густафсон заңы Амдал заңына қайшы келмейді. Айырмашылығы тек ЕЖ процессор санымен анықтап, қосымша ресурстарын пайдалануы өзгешелігімен анықталады.

Сана-Най заңы

Есептеу жүйесінің кейбір түрлерінде әрбір процессордың сыйымдылығы шектеулі жергілікті жады болады. ЕЖ-нің жалпы жады сыйымдылығы осы жергілікті жады құрылғыларының сыйымдылығымен анықталады. Мұндай ЕЖ-де шығарылатын есеп ішкі есептерге бөлініп, әр процессордың жергілікті жадысына жүктеледі. Густафсон заңындағыдай процессорлар саны өскен сайын, орындалатын есеп көлемі де өсе түседі. Есеп көлемі тек жергілікті жады көлемімен шектеледі. Мұндай есептеу жүйелері үшін Ксиан-Хе Сан мен Лайонел Най үдеу заңын ұсынған. Мұндай заңды жады көлемі шектелмелі ЕЖ үдеу заңы деп атайды.

Аталған үдеу заңы төмендегі формуламен анықталады:

$$S(n) = \frac{f + (1 - f)G\ (n)}{f + \ (1 - f)\frac{G\ (n)}{n}}$$

Формулада $G(n)$ параметрлік процессор санының өсуімен анықталатын жұмыс жүктемесімен яғни n есе өскен жергілікті жады санымен анықталады.

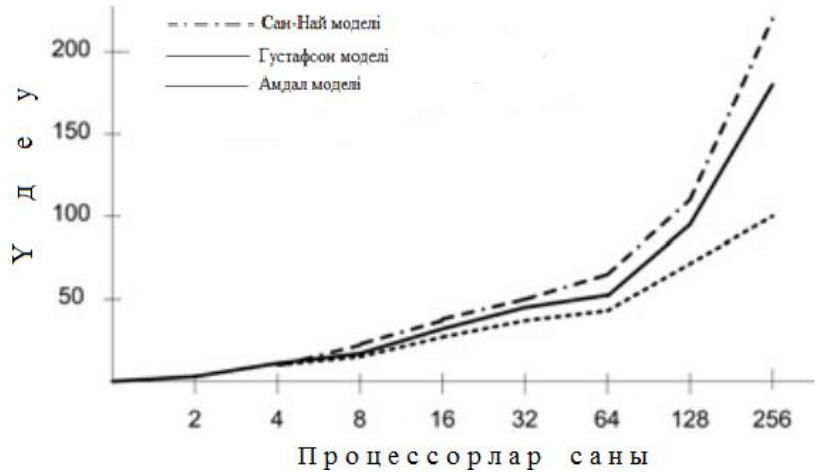
Жоғарыда келтірілген өрнек Амдал және Густафсон заңдарының қорытпасы болып табылады. $G(n) = 1$ болғанда есеп көлемі өзгертілмейді, онда Сан-Най формуласынан аламыз:

$$S(n) = \frac{f + (1-f)n}{f + (1-f)} = \frac{1}{f + \frac{1-f}{n}}$$

$G(n) = n$ мәнін алғанда, яғни жады сыйымдылығы n есе өскенде жұмыс жүктемесі де n есе өседі. Бұдан Густафсон формуласын аламыз:

$$S(n) = \frac{f + (1-f)n}{f + (1-f)} = f + (1-f)n$$

12.3-суретте қаралған үш түрлі үдеу модельдерін бір-бірімен салыстыру графигі келтірілген.

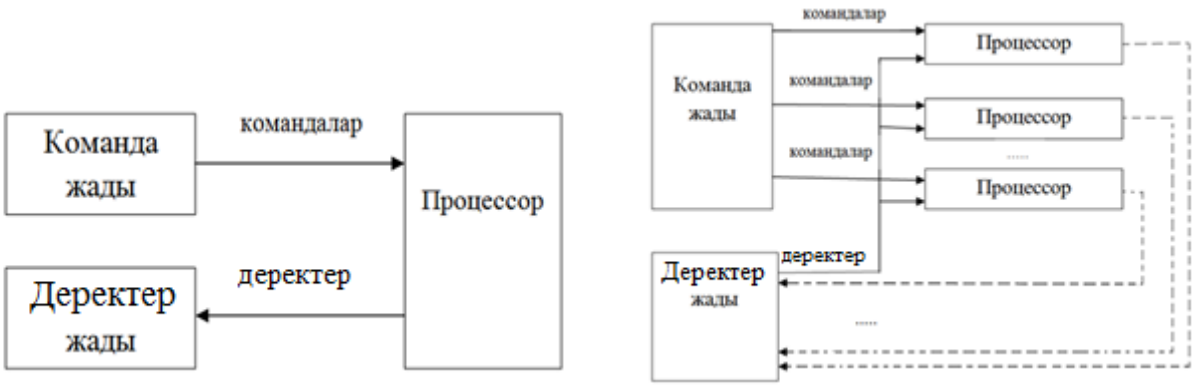


12.3-сурет. Үдеудің үш моделін салыстыру

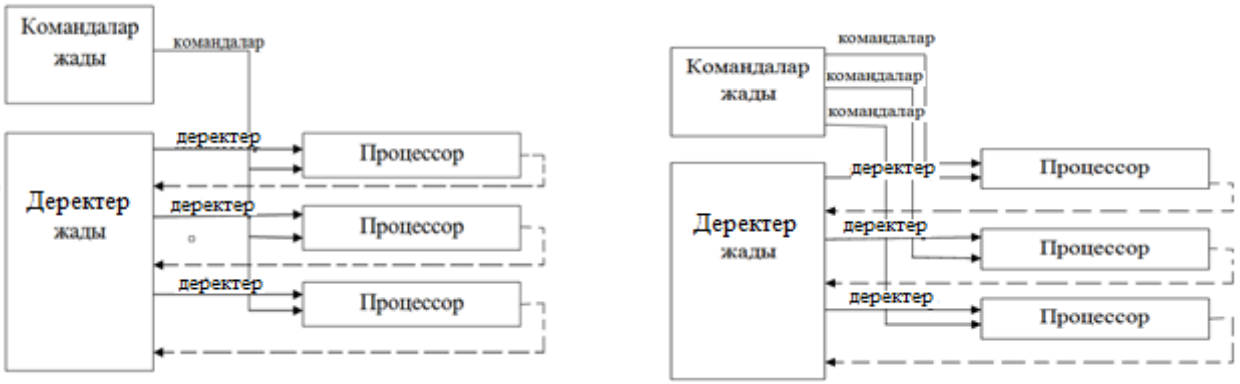
12.2. Параллель есептеу жүйелерін топтау

Параллель есептеу жүйелерінің түрлері өте көп. Олардың ішіндегі ең танылған жүйесіне 1966 жылы ұсынған М.Флинни ұсынған жүйесі жатады. Оның негізінде ағын ұғымы жатыр. *Ағын* деп процессорда өңделетін командалар мен деректер элементтерінің тізбегін айтамыз.

Флинни жүктеуі бойынша командалар (К) мен деректер (Д) саны бойынша ЕЖ төрт архитектуралық класқа бөлінеді. (12.4-сурет).



а б



в г

12.4-сурет.Флинни жүйесі бойынша ЕЖ сәулеттері: а - ЖКЖД (SISD); б-ККЖД (MISD); в - ЖККД (SIMD); г – КККД (MIMD).

ЖКЖД (SISD)

ЖКЖД – жеке командалар ағынды-жеке деректер ағынды (SISD–Single Instruction Stream/Single Date Stream) жүйесі (12.4, а-сурет).

ЖКЖД жүйесіне Фон Нейман машинасы жатады, мұнда тек бір команда ағыны бар. Олар тізбекпен бір дерек ағынымен өңделеді. Өңдеу жылдамдығын арттыру үшін конвейерлеу тәсілі қолданылуы мүмкін. ЖКЖД жүйесіне скалярлық және конвейерлік құрығылары бар CDC-6600, CDC-7600 есептеу машиналарын жатқызуға болады.

ККЖД (MISD)

ККЖД - көптік командалар ағынды-жеке ағынды деректер (MISD - Multiple Instruction Stream/Single Date Stream) жүйесі (12.4, б-сурет).. Анықтамаға сәйкес ЕЖ архитектурасында бір дерек ағынан өңдейтін көптеген процессорлар бар. Жүйеге мысал ретінде Флиннидің өзі де және басқа мамандар да осы уақытқа дейін ешбір есептеу жүйесін атай алмай отыр.

ЖККД (SIMD)

ЖККД - жеке команда ағынды-көптік деректер ағынды (SIMD Single Instruction Stream/ Multiple Date Stream) жүйесі (12.4, в-сурет).

Есептеу жүйесінің мұндай архитектурасы бір арифметикалық операцияны көптеген деректермен - вектор элементтерімен орындауға мүмкіндік береді. ЖККД класында барлық процессорлық элементтер басқару құрылғысынан жеке команда алып, оны өзінің жеке деректерімен орындайды. Аталған классқа векторлы-конвейерлі ЕЖ-н де қосуға болады, егер вектор элементтің дерек ағынының жеке элементі болса.

ЖККД жүйелеріне 12.4, б–суреттен көрініп тұрғандай, векторлық, ассоциативтік, систоликалық және матрицалық есептеу жүйелерін жатқызуға болады.

ККЖД (MIMD)

ККЖД-көптік командалар ағынды-көптік деректер ағынды (MIMD-Multiple Instruction Stream/ Multiple Date Stream) жүйесі (12.4, г-сурет). Есептеу жүйесінің мұндай түрі командаларды өңдейтін көптеген құрылғылар арқылы есептеу кешенін құрып, олардың әрқайсысы өз командалары және деректер ағынымен жұмыс жасайды.

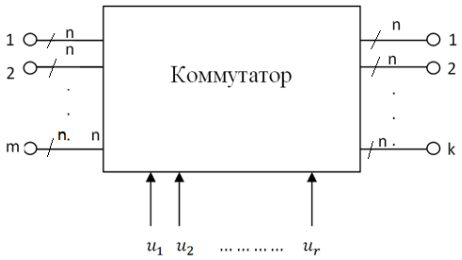
ККЖД класына көптеген көппроцессорлы және көпмашиналы жүйелер жатады.

Бақылау сұрақтары:

1. Параллель деңгейлерін салыстырыңыз.
2. Амдал заңының ерекшелігі неде?
3. Густафсон заңы Амдал заңының қандай моделін шешеді? Густафсон заңы қандай жағдайда қолданылады?
4. Сан-Най заңынан Амдал заңы қалай алынады?
5. Сан-Най заңынан Густафсон заңы қалай алынады?
6. Флинни топтау жүйесінің артықшылығы мен кемшіліктерін атаңыз?

13. ЕСЕПТЕУ ЖҮЙЕЛЕРІНІҢ ІШКІ БАЙЛАНЫСТАРЫН ҰЙЫМДАСТЫРУ

Кез-келген көппроцессорлы жүйеде (КПЖ) оның құрамдас бөліктері (процессорлар, жады модульдері, енгізу-шығару модулььдері) деректермен арналар құратын жабдықтар арқылы алмасады. Мұндай жабдықтар деректерді беру тәсілдеріне байланысты екі топқа бөлінеді. Олар: коммутаторлар және байланыс желілері. Коммутаторлар – деректерді беру каналдарын жедел байланыстыратын электрондық сұлбалар. Коммутатор m,n-разрядты ақпараттық кірістерден және k,n-разрядты ақпараттық шығыстардан және r (разрядты) басқару кірістерінен тұрады (13.1-сурет).



13.1-сурет. Коммутатор

Басқару сигналдары $(u_1, u_2, \dots, u_r)$ арқылы кіріс сигналдарын коммутатордың кез-келген шығысына қоса алады.

Байланыс желісі пайдаланылған кезде КПЖ әрбір процессоры тек көрші процессормен байланыса алады (нүкте-нүкте принципі), ал көрші емес басқа процессорлармен («алыс» процессормен) аралық процессорлар тізбегі арқылы байланысады. Соңғы жағдайда берілетін деректер келесі пунктке қосатын канал босағанға дейін аралық процессордың жадысында сақталады. Бұл жерде каналдарды қарапайым коммутатор арқылы байланыстыру емес, дестелерді коммутациялау орын алады. Егер берілетін деректердің ұзындығы үлкен болса, онда оларды бірнеше пакеттерге бөліп тиісті мекен-жайға бөлек-бөлек жіберуі мүмкін. Деректі алған жоқ. Оларды біріктіріп хабар құрады. Әр процессор жүйенің басқа құрамдастары (жады модульдері интерфейс блоктарын т.б) сияқты КПЖ-нің бір түйіні ретінде қарастырылады.

ЕЖ әртүрлі ішкі байланыстарын салыстыру үшін мынадай ұғымдар пайдаланылады: топология, байланысу уақыты, өткізу қабілеттілігі, құны т.б. Топология – құрылым элементтері және олардың бір-бірімен байланыс құру тәсілдері (ЕЖ ішкі байланыстарының ұйымдастырылуы). Желінің ішкі байланыс топологиясы әртүрлі параметрлермен анықталады. Олардың негізгілеріне мыналар жатады:

- *желі өлшемі* – N. Желіні құратын барлық түйіндер саны;
- *байланыс саны* – I. Желінің түйіндерін бір-бірімен байланыстыратын арналар саны

- *желі диаметрі* –D. Бір-бірінен алшақ орналасқан екі түйіндері қысқа жолмен қосылатын қабырғалар саны. Диаметр арқылы деректерді беру уақытын анықтауға болады;
- *желінің өткізгіштік қасиеті* – бір уақыт өлшеміндегі желі арқылы берілетін деректер мөлшері (Мб/сек, Гбайт/сек);
- *желі кідірісі* – хабардың желі арқылы өту уақыты. Ол хабарды беру маршрутына байланысты орта, максимальді және минимальді болады.

КПЖ ішкі байланыстарының топологиялық топталуы 13.2–суретте келтірілген.

13.2-сурет. Ішкі байланыстар топологиясының топталуы

13.1 Деректерді маршруттау функциялары

МПЖ топологиясын анықтау кезінде хабарды бір түйіннен екіншісіне тасымалдау үшін кезекті түйінді белгілі бір тәртіппен таңдауымыз керек. Осы тәртіп (ереже) арқылы деректерді маршруттау функциялары анықталады. Ол үшін жүйенің әрбір түйіндеріне бірегей мекен-жай кодтары (екілік кодтар) меншіктеледі. Осы мекен-жай кодтары арқылы түйіндер бір-бірімен байланыстырылады. Төменде КПЖ-де қолданылып жүрген деректерді маршруттаудың кейбір функцияларын қараймыз. Мысалдарда желі өлшемі – N, мекен-жай разрядтылығы – m (m=$\log_2 N$, мекен-жай биттер саны – b_i.

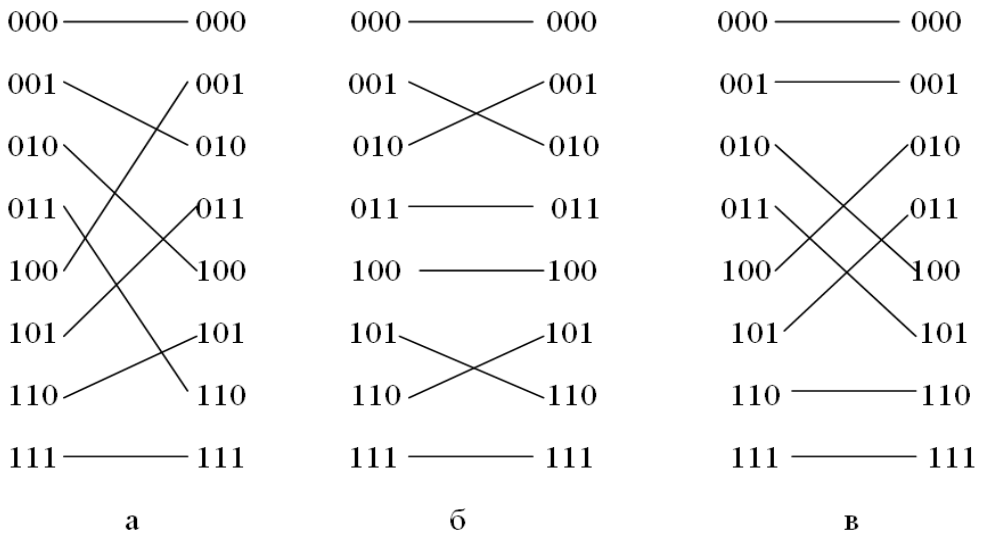
Алмастыру

Алмастыру функциясы төменде келтірілген формуламен анықталады.

$$S(b_m, b_{m-1}, \dots, b_1) = (b_{m-1}, b_{m-2}, \dots., b_1, b_m).$$

Формуладан көрініп тұрғандай і мекен-жайлы А түйіннен j мекен-жайлы В түйініне тікелей байланысу үшін А түйінінің і кодын солға қарай циклмен бір разрядқа жылжыту арқылы В түйінінің j екілік кодын табамыз. А түйінінің коды i=101 (m=3) болсын. Оны солға қарай бір разрядқа циклмен жылжытсақ, онда В түйінінің j=011 мекен-жай кодын табамыз. Егер i=101 кодын екі разрядқа солға қарай циклмен жылжытсақ, онда j=011 мекен-жай кодын аламыз. Онда ол алмасу функциясының басқа түріне жатады (екінші бит бойынша супер алмастыру функциялы, 13.3, в-сурет). Төменде алмастыру функцияларының кейбір түрлері келтірілген:

- і-бит бойынша субалмастыру:
  $S(b_m, b_{m-1}, \dots, b_i \dots, b_1) = (b_m, \dots., b_{i+1}, \dots., b_{i-1}, b_i).$
- і-бит бойынша алмастыру:
  $S(b_m, b_{m-1}, \dots, b_i \dots, b_1) = (b_{m-1}, \dots., b_{m-1+1}, \dots., b_m, b_{m-1}, .. b_1).$



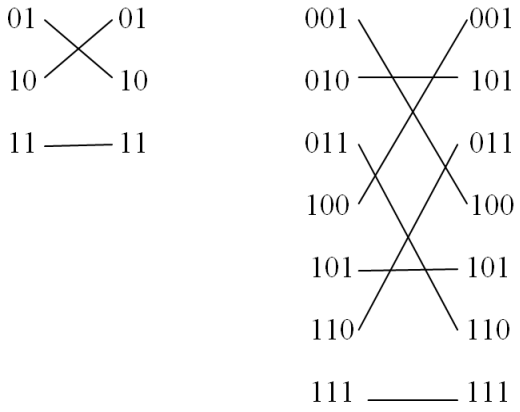
13.3-сурет. m=3 алмастыру топологиясына мысалдар. а-идеалды алмастыру; б-екінші бит бойынша субалмастыру; в-екінші бит бойынша супералмастыру.

Баттерфлей

«Баттерфлей» функциясы бойынша алынған пішін көбелек қанатын елестететін болғандықтан, оған «көбелек» деген ат берілген.

Оның математикалық функциясы төменде келтірілген (13.4-сурет):

B ($b_m, b_{m-1}, \dots., b_1 = (b_1, b_{m-1}, \dots, b_2, b_m)$.



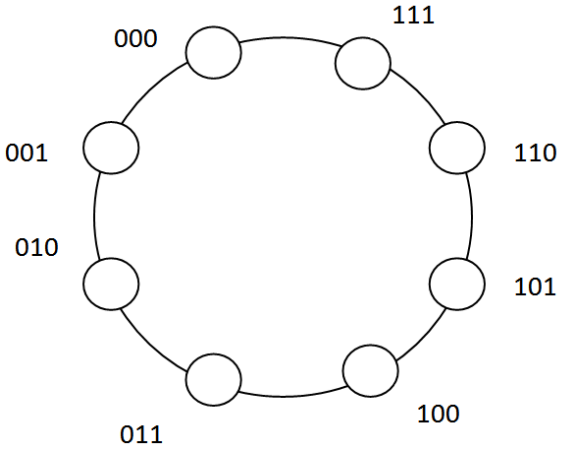
13.4-сурет.«Баттерфлей» топологиясына мысалдар: а – m=2 үшін; б – m=3 үшін

Жылжыту

Жылжыту алгоритмі үшін маршруттау функциясы төменде келтірілген:

SH (x)= (x+1)mod N; мұнда N= 2^m.

m=3 болғанда функция сақина топологиясына сәйкес келеді (13.5-сурет).



13.5-сурет. Жылжыту функциясының сақина тәрізді топологиясы.

ILLIAC IV

Жылжыту функциясының бірнеше түрлерін құрастыру арқылы күрделі маршруттау функциясын алуға болады. Солардың біріне ILLIAC IV желісі жатады. Ол ILLIAC IV есептеу жүйесінде қолданылған:

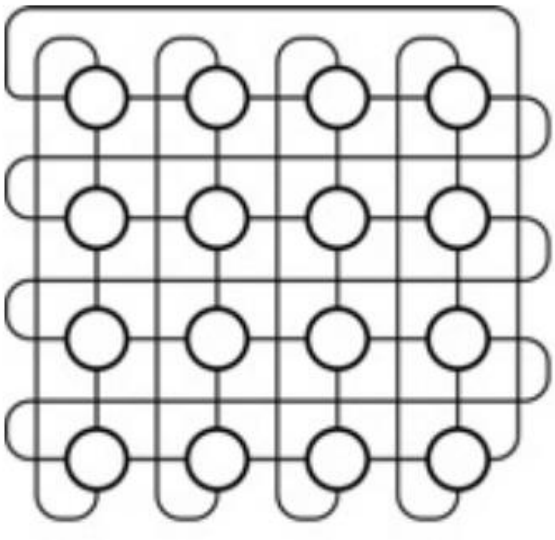
$$R_{+1} = (i+1) mod\ N;$$

$$R_{-1} = (i-1) mod\ N;$$

$$R_{+r} = (i+r) mod\ N\ (0 \le i \le N-1);$$

$$R_{-r} = (i-r) mod\ N\ (r = \sqrt{N}).$$

Формулада N=4 болғандағы топология кескіні 13.6-суретте көрсетілген.



13.6-сурет. Тор түрінде көрсетілген ILLIAC IV желі топологиясы

13.2. Статикалық топологиялар

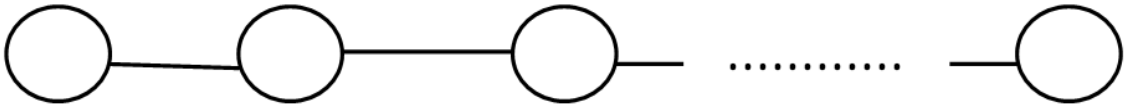
Статикалық топологияларда КПЖ кез-келген екі түйіндерді тікелей және тұрақты болатын бір жол арқылы қосады.

Оларға:

- бірөлшемді (сызықтық массив)
- екіөлшемді (сақина, жұлдыз, бұтақ, тор, систоликалық массив);
- үшөлшемді (толық байланысқан топология хордалы сақина);
- гиперкуб топологиялары жатады.

Сызықтық топология

Бір-бірімен байланысқан бір өлшемді массив (13.7-сурет).



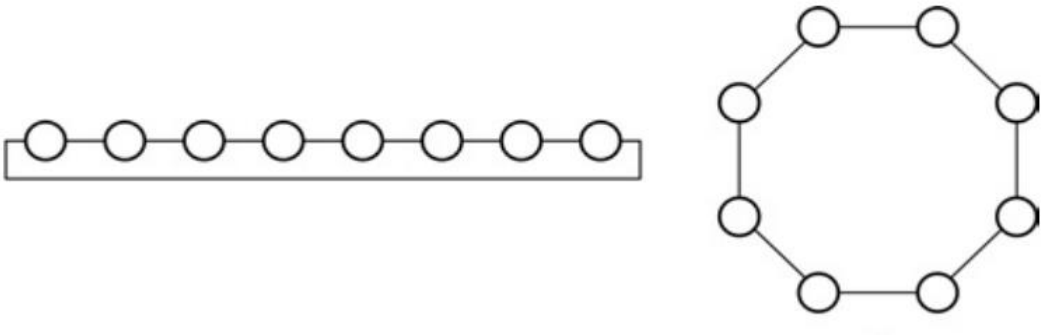
13.7-сурет. Сызықтық топология.

Сызықтық топологияның параметрлері: D=N-1; d=2; I=N-1.

Сақиналық топология

Сызықтық топологиядағы массивтің соңғы элементін бастапқы элементке байланыстырсақ, онда сақиналық топология аламыз (13.8-сурет). Көрші

түйіндер арасындағы арналар санына (бір немесе екі) байланысты сақина бір бағытты не екі бағытты деген атпен бөлінеді. Сақиналық топологиядағы D=min$\left\{\frac{N}{2}\right\}; d = 2,\ I = N.$



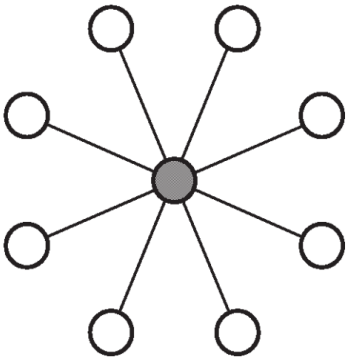
а *б*

13.8-сурет. Сақиналық топология: а- стандарттық көрсету; б- альтернативтік көрсету.

Сақиналық топологияға Token Ring желісі және KSR-1 және SCI есептеу жүйелерінің топологиясы мысал бола алады.

Жұлдыз тәрізді топология

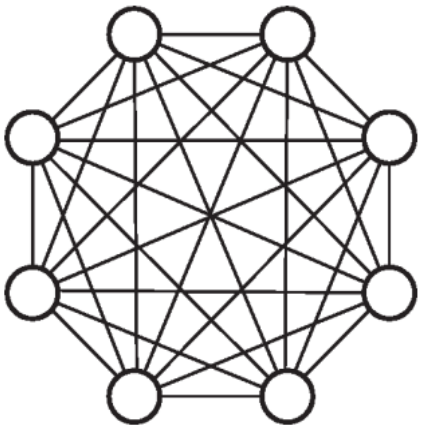
Жұлдыз тәрізді желіде көптеген түйіндер арнайы орталық түйін – концентратор арқылы біріктіледі (13.9-сурет). Топология праметрлері: D=2, d=N-1, I=N-1.



13.9-сурет. Жұлдыз тәрізді топология

Толық байланысқан топология

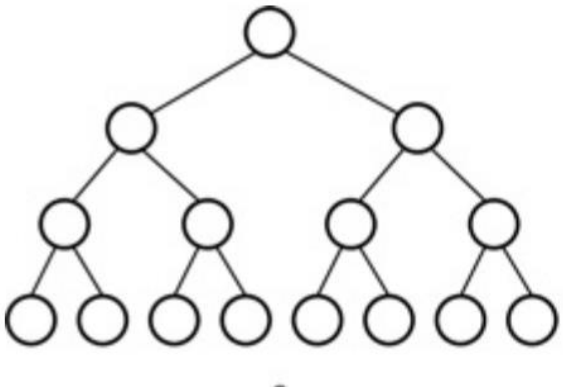
Толық байланысқан топологияда (13.10-сурет) әрбір түйін желінің қалған түйіндерімен тікелей байланысқан. N түйіннен тұратын желінің параметрлері: D=1, d=N-1, I= $\frac{N\,(N-1)}{2}$;



13.10-сурет. Толық байланысқан топология

Бұтақ тәрізді топология

Желі құрылымының сұлбасы 13.11-суретте көрсетілген. Егер h арқылы бұтақтың биіктігін (бұтақтың деңгейлер санын) белгілесек, онда желінің парметрлері төмендегідей болады: D=2 (n-1); d=3; I=N-1. Екілік бұтақ топологиясына Колумбия университетінде жасалған 1023 түйінді DADO жүйесі жатады.

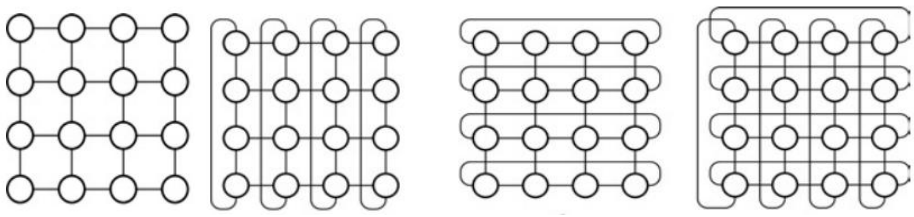


13.11-сурет. Стандартты бұтақ тәрізді топология

Торлы топология

Көптеген ғылыми-техникалық есептерде деректер массивтері кеңінен өңделеді. Деректер массивін өңдеу ерекшеліктері КПЖ құрғанда еске алынады. Мұндай топологияға тор архитектурасы жатады. Олардың пішіні өңделетін деректер массивінің түрлеріне және өлшемдеріне байланысты болады. Екі өлшемді деректер массивін өңдеуге жазық тікбұрышті көрші түйіндерінің әрқайсысы көрші түйіндермен байланысқан матрица сәйкес келеді (13.12, а-сурет). Өлшем m*m (m=$\sqrt{n}$) болғанда желі параметрлері былай анықталады: D=2 (m-1); d=4; I=2N-2m.

Жазық екі өлшемді матрицадан цилиндр тәрізді (13.12, б, в-сурет) топология, торойд тәрізді (13.12, г-сурет) желі топологияларын алуға болады.



13.12-сурет. Торлы топология: а–жазық; б– бағана бойынша цилиндр тәрізді; в– қатар бойынша цилиндр тәрізді; г–тороид тәрізді

Гиперкуб топологиясы

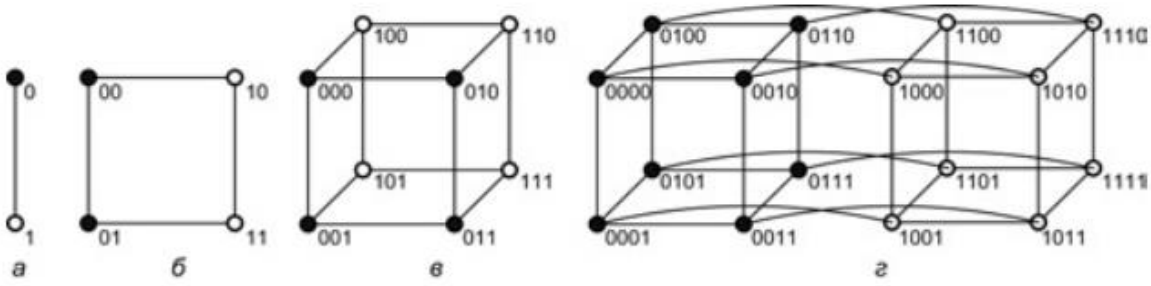
Параллель жұмыс істейтін процессорларды біріктірілгенде өте көп қолданылатын топологияға гиперкуб топологиясы жатады (13.13-сурет).

Екі түйінді байланыстыратын сызық (13.13, а-сурет) бірөлшемді гиперкуб құрады. Төрт түйіндер арқылы (13.13, б-сурет) құрылған квадрат екі өлшемді гиперкуб, ал 8 түйіннен құрылған куб (13.13, в-сурет) –үш өлшемді гипркуб құрады.

Өлшемі m (N= 2^m) Гиперкубта D=m, d=m, I=$\frac{mN}{2}$.

Гиперкуб өлшемін бірге өсірсек, онда түйін саны екіге өседі, ал түйін реті және желі диаметрі бірге өседі.

Желі түйіндерін нөмірлегенде көрші түйіндер нөмірлерінің екілік кодтары тек бір позициямен ерекшеленуі керек: мысалы кодтары 0010 және 0110 түйіндер – көрші; ал 0110 және 0101 кодты түйіндер көрші емес.



13.13- сурет. Гиперкуб топологиясы. а–бірөлшемді; б–екіөлшемді; в–үш өлшемді; г–екі үш өлшемді гиперкубтан құрастырылған төтрөлшемді гиперкуб.

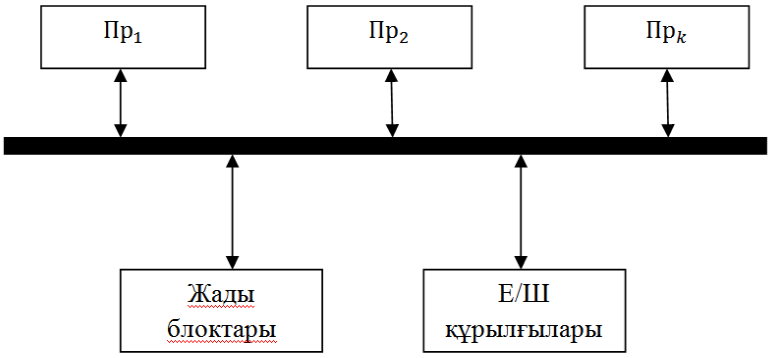
13.3. Динамикалық топологиялар

КПЖ динамикалық топологиясында түйіндер бір-бірімен басқарылатын электронды кілттер арқылы қосылады. Электронды кілттердің (коммутациялау элементтерінің) күйлерін өзгерте отырып желі топологиясын өзгерте аламыз.

Динамикалық топологиялық желілерін екі класқа бөліп қарауға болады. Олар: шиналар негізінде құрылған желілер және коммутаторлар негізінде құрылған желілер. Шиналы топология екі түрге – бір шиналы және көп шиналы топологияға бөлінеді.

Бір шиналы топология

Жүйенің барлық түйіндері (процессорлар, жады блоктары, енгізу-шығару құрылғылары) бір шинаға қосылады. Мұндай динамикалық желі қарапайым әрі арзан (13.14-сурет).

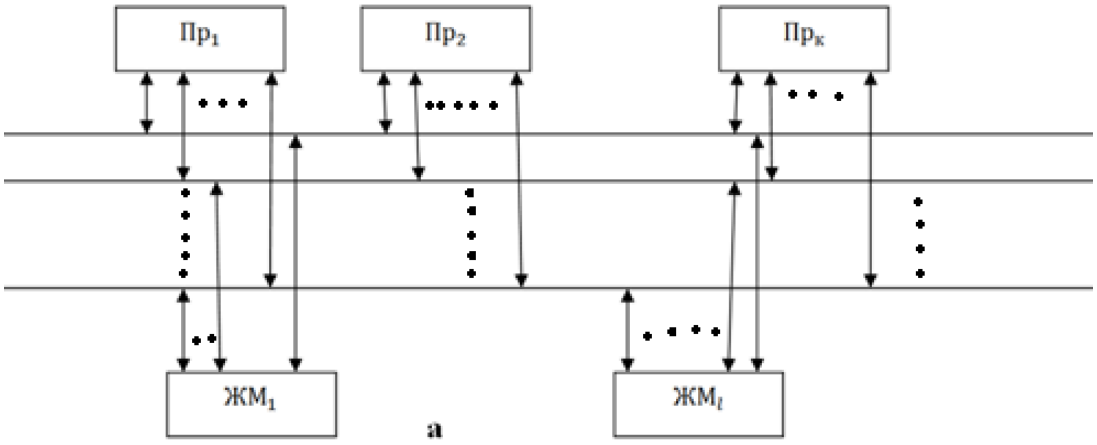


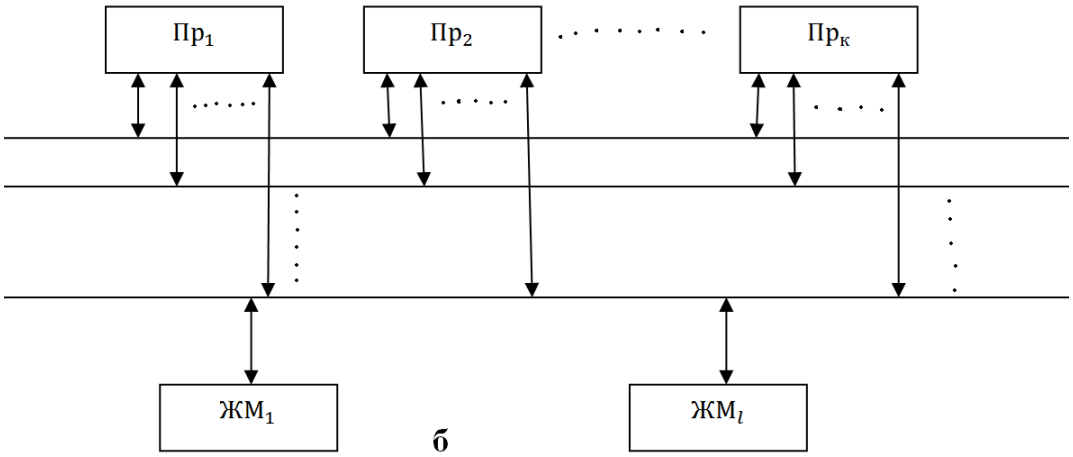
13.14-сурет. Бір шиналы жүйе

Барлық процессорлар жады және енгізу-шығару құрылғылары ортақ пайдаланылатын бір шинаға қосылған. Бір шиналы топологияда рет саны d=1, D=1. Бір шиналы біренеше түйіндерді бір топқа (кластерге) біріктіріп, олар арқылы басқа топологиялы желі құруға болады.

Көпшиналы топология

Көпшиналы топологиялы есептеу жүйелерінде көптеген процессорлар мен жады модульдері бірнеше параллель шиналар арқылы біріктіріледі. ЕЖ процессорлары әрбір шиналарға қосылса, жады модульдері (ЖМ) бір шинаға немесе қатар бірнеше шиналарға қосыла алады (13.15-сурет).





13.15-сурет. Көпмашиналы ЕЖ жады модульдерін (ЖМ) қосу сұлбаларына мысал. а–ЖМ барлық шиналарына қосылған; б–ЖМ бір шинаға қосылған.

Көпшиналы топология өнімділігі жоғары әртүрлі ЕЖ-де қолданыс тауып отыр.

Бұғатталған және қайта пішімделетін топологиялар

Дианмикалық жүйелерде терминалдық түйіндер арасындағы саражол коммутаторлар арқылы қалаптасады. Коммутаторлар желінің әрбір кірісін оның әрбір шығысымен байланыстыру міндетін атқарады. Байланыс кезінде n кірісті және n шығысты желіде электронды кілттер бірнеше терминалдар жұбын бір уақытта қақтығысқа алып келтірмей коммутациялауды қамтамасыз ету керек. Қақтығыс бір коммутатор арқылы бір уақытта екі немесе одан көп трафиктеу арқылы деректер беру кезінде пайда болатын құбылыс. Осыған байланысты коммутаторлар ақылы құрылғы желілері үш түрге: бұғатталатын, бұғатталмайтын және қайта пішінделетін түрлеріне бөлінеді.

Бұғатталатын желілерде егер коммутаторлар арқылы терминалдар байланысып қойса, онда басқа байланыстар осы коммутатор арқылы байланыстырылмайды, олар бұғатталады.

Бұғатталмайтын желілерде кез-келген кіріс терминалдары кез-келген бос шығыс терминалдарымен одан бұрын орнатылған байланыстарға қарамай жаңа байланыс құра алады. Бұғатталмайтын желілдер екі түрге бөлінеді. Бірінші түрінде жүйеде қолданылып отырған желі топологиясының ерекшелігіне байланысты бұғаттау орын алмайды. Екінші түрінде желіде белгілі маршруттау алгоритмін іске қосу нәтижесінде желі қақтығыстарына жол берілмейді.

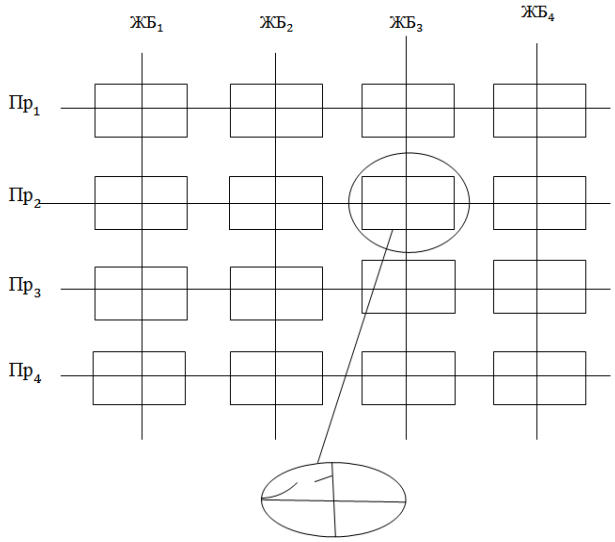
Қайта пішінделетін желілерде коммутаторларды тиімді ұйымдастыру арқылы көптеген кіріс және шығыс түйіндері бір уақытта деректермен алмасуға мүмкіндік алады. Желілердің бұл түрінде деректерді жеткізетін тракттар бір

уақытта қалыптастырылуы керек. Қайта пішінделетін желі топологиясы матрицалық есептеу жүйелерін құруда кеңінен қолданылады.

«Кроссбор» топологиясы

Мұндай топологиясында матрицалық коммутатор қолданылады.13.16–суретте n кірісті және m шығысы бар «Кроссбор» келтірілген. Кроссбор арқылы (n*m) терминалдар жұбы деректермен қатар алмаса алады.

Мұндай топологияда қақтығыстар орын алмайды. Жаңа байланыс бос түйіндер пайда болғанда кез-келген уақытта орын алады. Желі бұғатталмайтын желі түріне жатады. Деректерді беру жылдамдығы қалған топологияға қарағанда жоғары. Топология Fujitsu, VPP500 224×224 және т.б жүйелерде қолданылған.

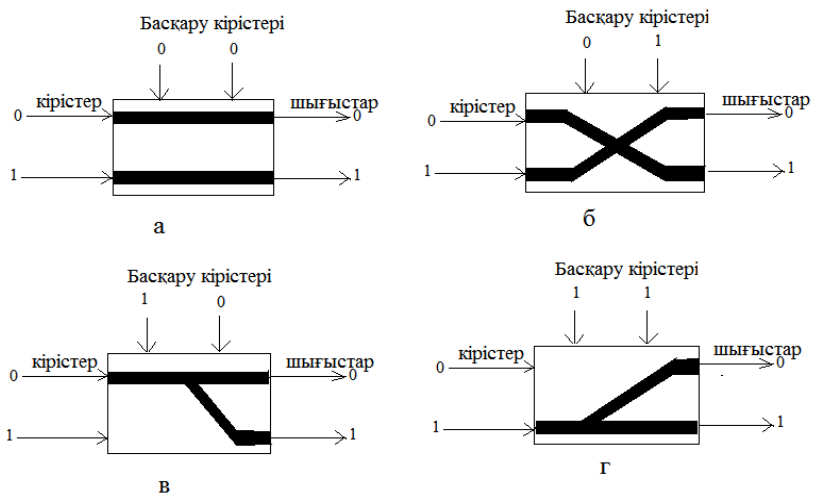


13.16-сурет. n*m өлшемді матрицалық коммутатор

Динамикалық топология желісінің коммутациялау элементтері

Желі түйіндерін көптеп қосу үшін көпсатылы топологиялар кеңінен қолданылады. Көпсатылы топологиялар толық байланысқан коммутациялау матрицалары арқылы және базалық коммутациялағыш элементтер арқылы құрылады.

Бірінші топқа жататын желілерде базалық коммутациялау элементтері негізінде n*m кроссбар пайдаланылады. Желінің екінші түрінде іс жүзінде 2*2 кросбардының екі разрядты кодымен басқарылатын түрі қолданылады (13.17-сурет). Мұндай кроссбарды базалық коммутациялау элементі (БКЭ) немесе β-элементі деп аталады. БКЭ төрт күйінің алғашқы екеуі негізгі күйлеріне жатады.



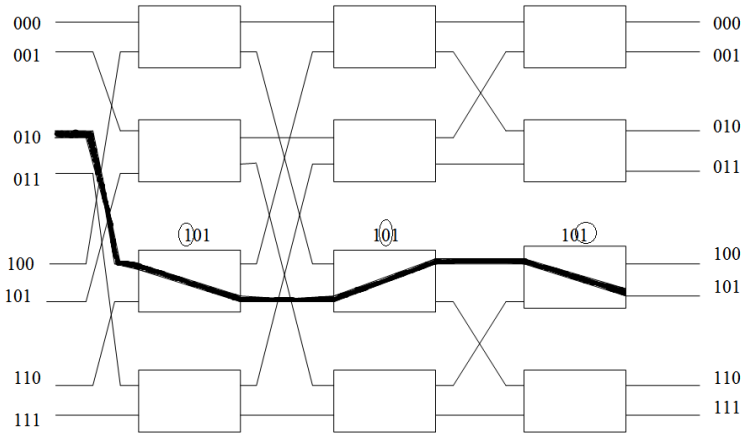
13.17-сурет. β-элементінің күйлері: а–тура; б–ерсілі-қарсылы; в–жоғарғы кірістен барлық шығыстарға; г–төменгі кірістен барлық шығыстарға

Оларда кіріс деректері тура немесе ерсілі-қарсылы беріледі. Келесі екі күйлерінде β-элементтерінде кіріс деректері элемент шығыстарына қосылған түйіндердің бәріне беріледі. БКЭ басқару кірістеріне сигнал желіні басқару құрылғысынан беріледі. Бета-элементінің күрделі түрлерінде басқару сигналдары БКЭ элементтерінің ішінде хабарда көрсетілген дерек көзінің және деректі алушы түйіннің мекен-жай кодына байланысты қалыптастырылады.

Бұғатталатын көпсатылы желі

«Баньян» топологиясымен құрылған желі.

Баньян желісі бұғатталатын көпсатылы желіге жатады. Мұнда кез-келген кіріс пен шығыс өзінің тек бір жолы арқылы байланысады. Оған мысал ретінде 13.18-суретінде көрсетілген 8*8 өлшемді «Баньян» желісі келтірілген.

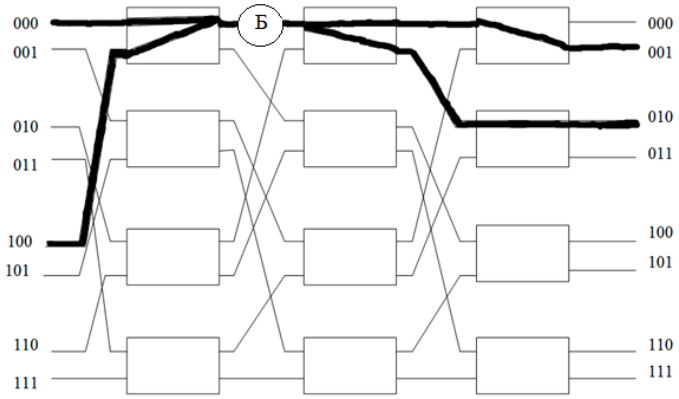


13.18-сурет. Баньян типті топология

Берілетін десте тақырыбында деректі қабылдайтын түйіннің 3 разрядты коды көрсетіледі. Келтірілген желі *өзі маршруттаматын* желілердің түріне жатады, өйткені деректі қабылдау түйінінің мекен-жайы арқылы хабар

маршруты анықталады және басқарылады. Әрбір БКЭ пакеттегі мекен-жай кодының бір разрядын талдап, оның мәніне байланысты хабарды БКЭ элементінің бірінші не екінші шығысына береді. 13.19–суретте келтірілген желі үш сатылы. Деректермен алмасатын түйіндер саны сегіз болғандықтан олардың мекен-жай кодтары үш разрядты. Деректі беру үстінде желінің әр сатысында қойылған элемент өзіне тиісті биттерді талдап деректің келесі сатыға беру бағытын басқарады.

«Омеге» желісіндегі бұғаттауға мысал 13.19-суретте келтірілген.



13.19-сурет. 8*8 өлшемді желіні бұғаттауға мысал

Желіде мекен-жай коды 000_2 терминал шығысынан мекен-жай коды 001_2 шығыс терминалына және мекен-жай коды 100_2 кіріс терминалынан мекен-жай коды 010_2 шығыс терминалына хабарларды беру үстінде бұғаттау орын алады. Сол жақтағы жоғарғы коммутациялау элементінің жоғарғы шығысы арқылы екі бағыттан келетін деректер бір уақытта берілуі мүмкін емес.

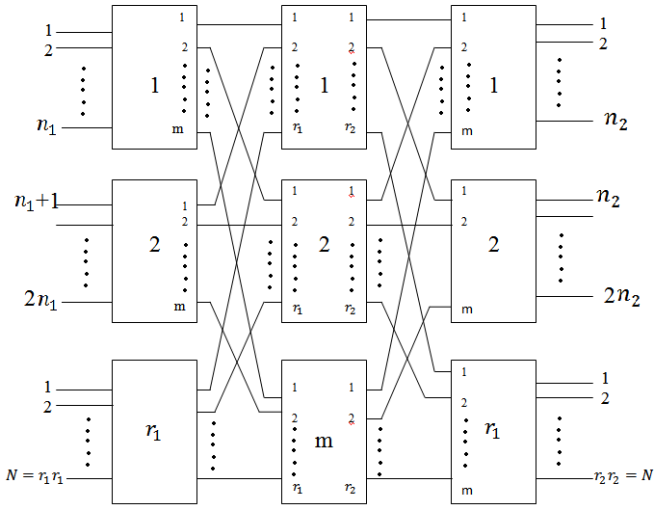
Бұғатталмайтын көпсатылы желі

Мысал ретінде Клоз топологиясын қараймыз. Клоз 1953 жылы Кроссбар негізінде құрылған көпсатылы желі арқылы бұғатталмайтын желі құруға болатындығын көрсетті.

Желінің кіріс сатысы r_1 кроосбардан, екінші сатысы m, ал үшінші сатысы r_1 кроссбарлардан тұрады. Кіріс сатысының әрбір коммутаторы n_1 кірістері және m шығыстары бар. Кіріс коммутаторларының әрбір шығысы екінші саты коммутарторларының тек бір кірісімен байланысқан. Мысалы, 1-кіріс коммутаторларының 1-шығысы екінші саты коммутаторларының біріншісінің бірінші кірісімен, ал 1-кіріс коммутаторларының 2-шығысы екінші саты коммутаторларының 2-екіншісінің 1-кірісімен, ал бірінші саты коммутаторының m-шығысы екінші саты m-коммутаторларының 1-кірісімен байланысқан. Соған ұқсас бірінші саты коммутаторларының қалғандарының шығыстары да екінші саты коммутаторларының бірімен байланысқан. Екінші

саты коммутаторларының шығыстары жоғарыда қаралған тәртіппен үшінші саты коммутаторларының кірістерімен байланысқан. Клоз желісінде кз-келген желі кірісінен кез-келген шығысымен байланысуға болады. Клоздың дәлелдері бойынша, егер $m \geq n_1, +n_2 - 1$ шарты орындалса, онда желі бұғатталмайтын желіге жатады. Егер $n_1 = n_2 = n$ болған жағдайда аралық саты саны $m \geq 2n - 1$ болған жағдайда Клоз желісі бұғатталмайтын желіге жатады. Егер $m \geq n_2$ болған жағдайда қайта үйлестіру арқылы құрылатын бұғатталмайтын желі топологиясын аламыз. Желі көрсеткіштерінің m, n_1 және n_2 басқаша ара қатынасында Клоз топологиясы бұғатталатын желіге айналады.

Үш сатылы Клоз желісі 13.20-суретте келтірілген.

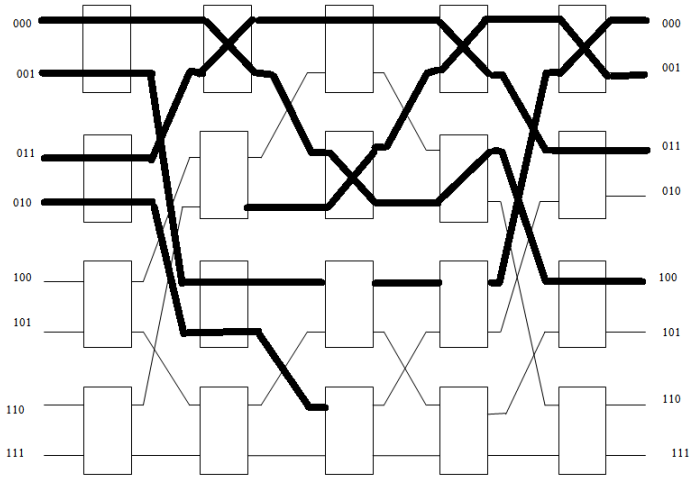


13.20-сурет.Үш сатылы Клоз топологиясы

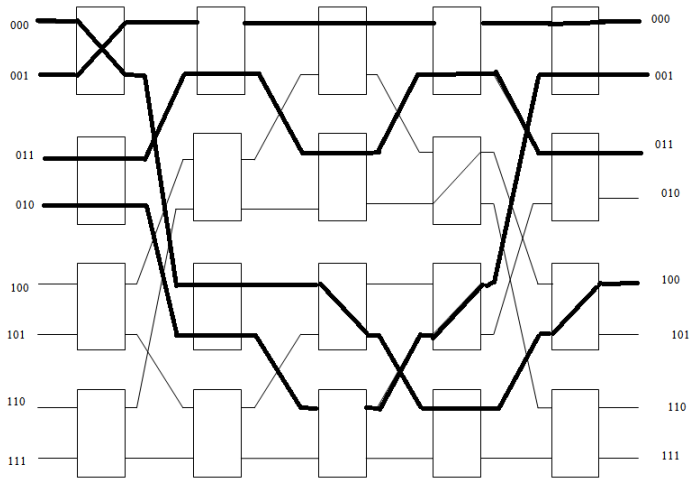
Егер $n_1 = r_1 = m_2$, онда ондай желіні «Мемфис» желісі деп атайды. Мұндай топологияны IBM фирмасының GF-11 жүйесінде қолданған. Клоз топологиясы көптеген фирмалардың ЕЖ-де қолданылады. Оған мысал: Fujitsu (FETEX-150), NIPPON ELECTRIC CMPANY (ATON), Hitachi.

Қайта пішімделетін көпсатылы желі

Мұндай желілерде желі коммутаторларын тиімді орналастыру арқылы көптеген кіріс және шығыс жұптары бірін-бірі бұғаттамайтындай етіп байланыс құруға мүмкіндік алады. Қайта пішіндеу кезінде желі түйіндерінің байланыстарын үзіп желі қайта құрылады. Қайта үлестірілетін желіге Бенеш желісі жатады. 13.21- суретте Бенеш желісіне мысал келтірілген.



а



б

13.21-сурет. Бенеш желісі. а–қайта пішімдеуге дейінгі желі; б–қайта пішімделген желі.

Бақылау сұрақтары

1. Статикалық және динамикалық қатынас желілері бір-бірімен қалай ажыратылады?

2. Маршруттау не үшін керек?

3. «Беттерфлей» топологиясы не үшін олай аталады?

4. Сызықтық топологиясының қандай жақсы және жаман жақтары бар?

5. «Жұлдыз» тәрізді топология мен «бұтақ» тәрізді топологияны салыстырыңыз

6. Динамикалық топологияның түрлерін атаңыз.

7. Бір шиналы және көп шиналы топологияларды салыстырыңыз

8. Клоз желісінің ерекшеліктерін түсіндіріңіз

9.Қайта пішімдеу қай уақытта орын алады?

14. ЕСЕПТЕУ ЖҮЙЕЛЕРІНІҢ ЖАДЫСЫН ҰЙЫМДАСТЫРУ

Көппроцессорлы немесе көпмашиналы жүйелерде жады құрылғыларының ұйымдастыруына байланысты жады ресурсын ортақ пайдаланатын және үлестірілген жадылы есептеу жүйелері деп екіге бөлінеді.

Ортақ жадылы жүйелерде жады құрылғысының кез-келген мекен-жайына әрбір процессор толық қатынас құра алады. ЕЖ ұйымдастыруының мұндай түрі ЖККД және КККД ЕЖ кластарында орын алады.

Үлестірілген жадылы есептеу жүйелерінде әр процессордың меншікті жадысы болады. Процессорлар желі арқылы біріктіріліп, деректермен хабар арқылы алмасып отырады. Ортақ жадылы есептеу жүйелерін - мультипроцессорлар, ал үлестірілген жадылы жүйелерін - мульти компьютерлер жүйесі деп те атайды.

Ортақ және үлестірілген жады негізінде есептеу жүйесінің үш түрлі жады сәулетін атап өтуге болады. Олар:

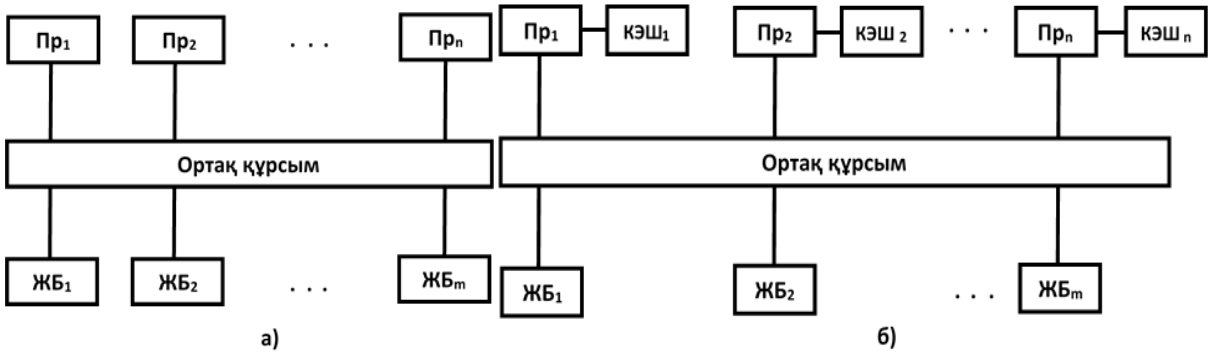
- ортақ жадылы сәулет;
- үлестірілген ортақ жадылы сәулет;
- үлестірілген жадылы сәулет.

14.1. Ортақ жадылы ЕЖ

Ортақ жадылы есептеу жүйесінде барлық процессорлар жады мекен-жай кеңістігіне қатынас құра алады. Әр процессорлардың ортақ жадыға қатынас құру уақыты бірдей, яғни қатынас құру біркелкі. Сондықтан мұндай есептеу жүйесін қысқартып UMA (Uniform Memory Access) деп атайды. UMA сәулеті ЕЖ құруда кеңінен таралған.

Ортақ жады бір блок немесе жеке-жеке бірнеше блок түрінде ұйымдастырылады

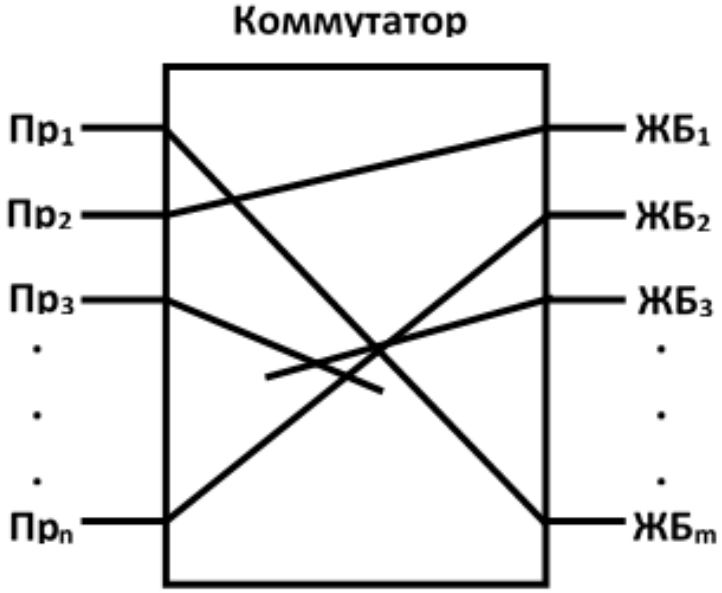
14.1-суретте жадыға біркелкі қатынас құратын ЕЖ (UMA) келтірілген.



14.1-сурет. Жадыға біркелкі қатынас құратын ЕЖ: а -КЭШ жадысы жоқ процессорлы жүйе; б - КЭШ жадылы процессорлы жүйе.

Суретте ЖБ - жады блогы; КЭШ - КЭШ-жады

14.1-суреттегі ортақ шинаны 14.2-суретте көрсетілген коммутатормен алмастыруға болады.



14.2-сурет. Коммутатор негізінде құрылған UMA жүйесі.

UMA архитектурасымен құрылған есептеу жүйелеріне Cray T90 супер машинасы және Inte SHU, Sun E10000, IBM R60 т.б. жүйелері жатады.

14.2. Үлестірілген ортақ жадылы ЕЖ

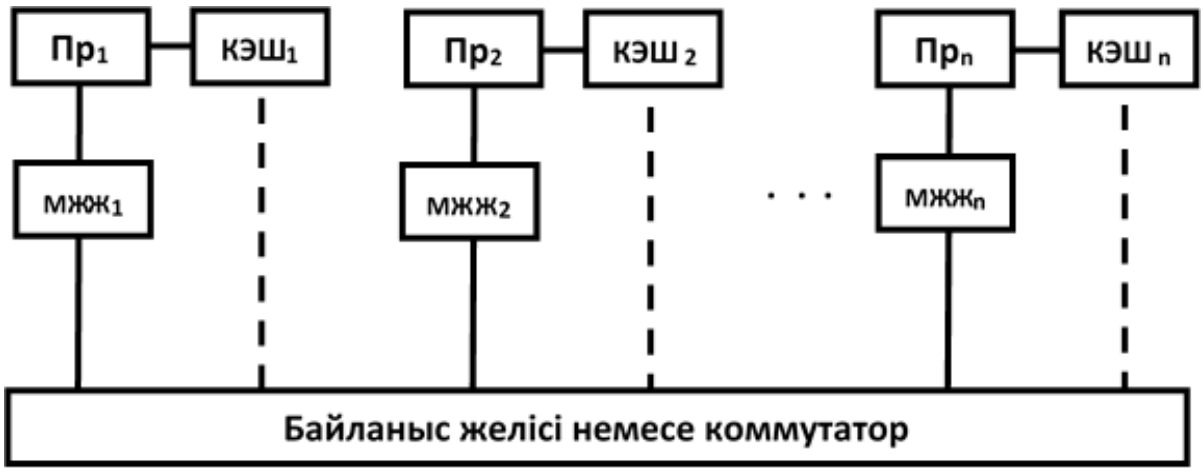
Үлестірілген ортақ жадылы ЕЖ-де әрбір процессордың жергілікті жадысы болады, ал олардың жиынтығы ЕЖ ортақ жадысын құрады. Мұндай жүйеде де мекен-жай кеңістігі ортақ және кез-келген процессор кез-келген мекен-жай кеңістігіне қатынас құра алады. Үлестірілген ортақ жадылы ЕЖ-ні іске асыру идеясын үш топқа бөліп қарайды. Олар: NUMA, COMA және DSM.

NUMA сәулеті.

NUMA (Non - Uniform Access - жадыға әртүрлі қатынас құру) NUMA жүйесінде ЕЖ ортақ жадының бір бөлігі болып табылатын әрбір процессордың меншікті жергілікті жадысы (МЖЖ) болады. МЖЖ мекен-жайы ЕЖ мекен-жай кеңістігінің бір бөлігін құрады.

Әр процессор өзінің МЖЖ қатынас құруымен қатар басқа «алыстағы» процессордың жергілікті жадысына да қатынас құра алады. Процессордың меншікті жергілікті жадысына қатынас құру уақыты «алыстағы» процессордың жергілікті жадысына қатынас құру уақытынан төмен, өйткені «алыс» процессор жадысына қатынас байланыс желісі немесе коммутатор арқылы жүргізіледі.

NUMA жады сәулетімен құрылған есептеу жүйесі 14.3-суретте көрсетілген.



14.3-сурет. Қатынас құруы әртүрлі NUMA ЕЖ

Ортақ жадылы ЕЖ-де әр процессор МЖЖ-дан басқа жергілікті КЭШ-жады болатындығын айтқанбыз. Мұндай жүйеде негізгі жадының кез-келген блогының көшірмесі барлық процессорлардың - КЭШ-жадысында сақталады. Жұмыс үдерісі кезінде егер процессорлардың бірі өз КЭШ-жадысындағы деректі өзгертсе, онда қалған процессорлардың КЭШ жадысындағы деректерін өзгерген дерекке сәйкестендіру керек.

Мұндай КЭШ-жадылардағы деректі сәйкестендіру мәселесін КЭШ-жады *когоренттігі* деп атайды. КЭШ-жадысын сәйкестендіру мәселесін шешу тәсілдеріне байланысты NUMA тұжырымдамасы екі түрге бөлінеді. Олар - ccNUMA және nccNUMA.

*ccNUMA (*Cahe Coherent Non - Uniform Memory Architecture - КЭШ когоренттілігі қамтамасыз етілген жадыға әртүрлі қатынас құру сәулеті шина негізінде іске асырылады.

Мұнда арнайы аппараттық жабдықтар шина арқылы КЭШ-жадыға деректердің жазылуы мен оқылуын бақылайды. Егер кез-келген процессорда КЭШ-жады деректерінде өзгеріс болса, онда аппараттық жабдықтар қалған процессорлардың КЭШ-жадына тиісті өзгерістер енгізіп отырады. SGI Origin 2000, Origin 3000, Cray, T3E, HP Exemplar, IBM/Sequent NUMA - Q2000 есептеу жүйелерінде nnNUMA типті сәулет жүйесі енгізілген.

*nccNUMA (*Non -Cahe Coherent Non - Uniform Memory Architecture - КЭШ когоренттілігі қамтамасыз етілмейтін жадыға әртүрлі қатынас құру) архитектурасында КЭШ-жадысындағы деректер сәйкестегі бағдарламалық жабдықтар арқылы жүргізіледі.

nccNUMA архитектурасымен супер компьютер Cray T3D және IBM SP3 жүйесінде іске қосылған.

COMA сәулеті

COMA (Cahe Only Memory Architecture - тек КЭШ-жадылы сәулеті). Мұнда әр процессордың КЭШ-жадысы үлкен КЭШ-жадысының бір бөлігі ретінде қаралады. COMA сәулетті ЕЖ құрылымы 14.4-суретте көрсетілген.

14.4-сурет. COMA сәулетті ЕЖ.

COMA сәулетінде деректер тұрақты бір жады модуліне бекітілмеген. Деректер соңғы сұрату жасаған КЭШ-жадыға ауыстырылады. Ол үшін операциялық жүйе қатынаспайды. Жады құрылғысының мұндай режимінде жұмыс жасау *анықтамалар* (АН) деп аталатын күрделі аппараттық жабдықтар пайдаланылады. Анықтамаларда барлық айнымалылар туралы деректер сақталады. Соңғы жазылған деректің көшірмесі КЭШ-жадыдан шығарылмайды.

COMA архитектурасы көптеген ЕЖ жүйелерінде іске асырылған. Олардың қатарына KSR және DDM жүйелері жатады.

DSM сәулеті.

ЕЖ үлестірілген ортақ жады түріне DSM (Distribute Shared Memory) сәулеті жатады. Мұнда әр процессордың жергілікті жадысы бар, бірақ NUMA-дан өзгешелігі оның басқа процессорлардың жергілікті жадына қатынас құруға мүмкіндігі жоқ. Ортақ жадылы режим операциялық жүйе арқылы ортақ мекен-жай кеңістігін керек процессорға қатынас құру арқылы іске асырылады. Процессорлар дерекпен хабарлар арқылы алмасады.

DSM архитектурасы КККД сыныбына жататын IBM SP2, DEC TruCluster және IntelTFLOPS ЕЖ-де кеңінен қолданылады.

14.3. Үлестірілген жадылы ЕЖ

Үлестірілген жадылы ЕЖ-да әрбір процессорлардың меншікті жадысы болады және тек осы жадыға мекендейді. Мұндай жүйені көпмашиналы ЕЖ деп те атайды.

Үлестірілген жадылы ЕЖ-н «алыстағы» жадыға тікелей қатынас құра алмайтын архитектура (NORMA, No Remote Memory Access) деп те атайды.

Үлестірілген жадылы ЕЖ 14.5-суретте келтірілген.

14.5-сурет.Үлестірмелі жадылы ЕЖ

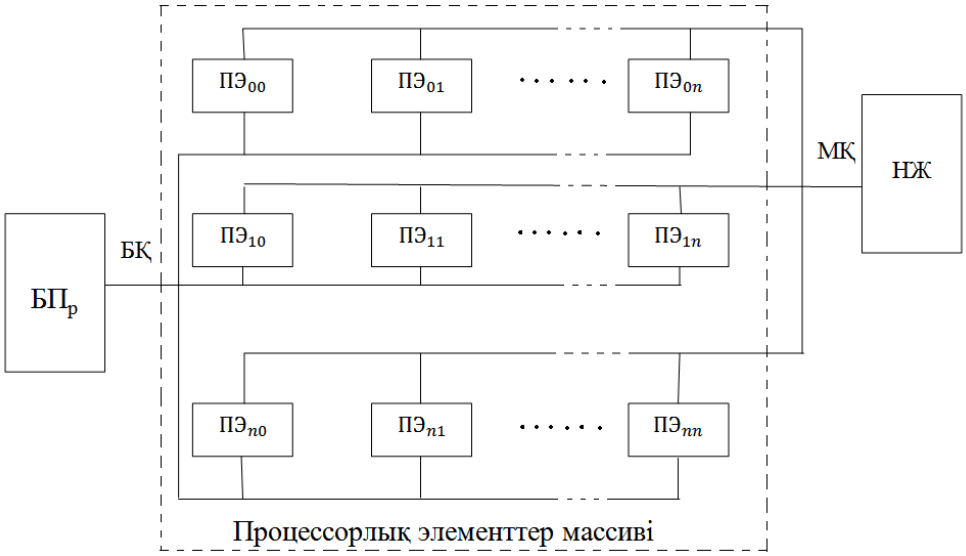
Әр процессорға қалған процессорлар сырт құрылғы ретінде қаралады. Басқа процессорға дерек беру үшін процессор өзінің жергілікті жадысында дерек блогын қалыптастырады. Одан соң хабар жергілікті контроллер арқылы хабар алушы процессордың жергілікті контроллеріне жіберіледі. Қабылдаушы процессордың контроллері хабарды алып оны өзінің жергілікті жадысына орналастырып, ол туралы хабарды жіберген процессорға ескертеді.

Бақылау сұрақтары

1. Ортақ жадылы ЕЖ жады блоктарының мекен-жай кеңістігі немен анықталады?
2. NUMA сәулетінде «алыстағы» процессордың жергілікті жадыға қатынас құруы қалай орындалады?
3. ccNUMA және nccNUMA архитектурасы немен өзгешеленеді?
4. COMA сәулетіндегі «анықтама» жабдықтары қандай қызмет атқарады?
5. NORMA архитектурасының NUMA архитектурасынан айырмашылығы неде?

15.ЖККД ЕСЕПТЕУ ЖҮЙЕЛЕРІ

ЖККД есептеу жүйелерінде бір типті операциялар көптеген деректер элементтерімен параллель орындалады. Мұндай көппроцессорлы жүйенің (КПЖ) жалпылама сұлбасы 15.1-суретте келтірілген.



15.1-сурет.ЖККД жүйесінің жалпылама сұлбасы

Жүйе бақылау процессорынан (БПр), негізгі жадыдан (НЖ) және процессорлық элементтер массивінен (ПЭМ) тұрады. Процессорлық элементтер (ПЭ) негізгі жадымен деректер шинасы (МҚ) және БПр-імен басқару шинасы (БҚ) арқылы байланысқан.

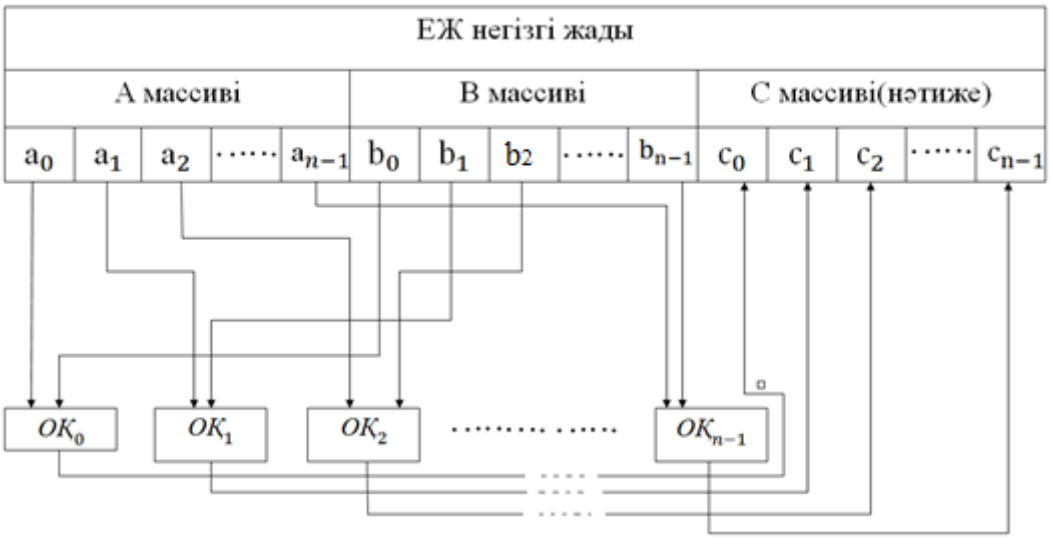
Басқару процессоры негізгі жады сақталатын ортақ команданы барлық процессорлық элементтерге жеткізеді. Әрбір ПЭ команданы өз деректерімен параллель өңдеу жүргізеді. ПЭ жұмысын разрядтылығы төмен процессор немесе қарапайым операциялық құрылғы орындай алады. ЖККД есептеу жүйелеріне векторлық, матрицалық, ассоциативтік және систоликалық есептеу жүйелері жатады.

15.1. Векторлық есептеу жүйелері

Көптеген есептерді, әсіресе әртүрлі үрдістерді және күрделі объекттерді модельдеу кездерінде жылжымалы үтірлі сандар массивін өңдеу керек. Мұндай санда массив матрица немесе векторлар түрінде беріледі. Олармен матрицалық операциялар жүргізуге тура келді.

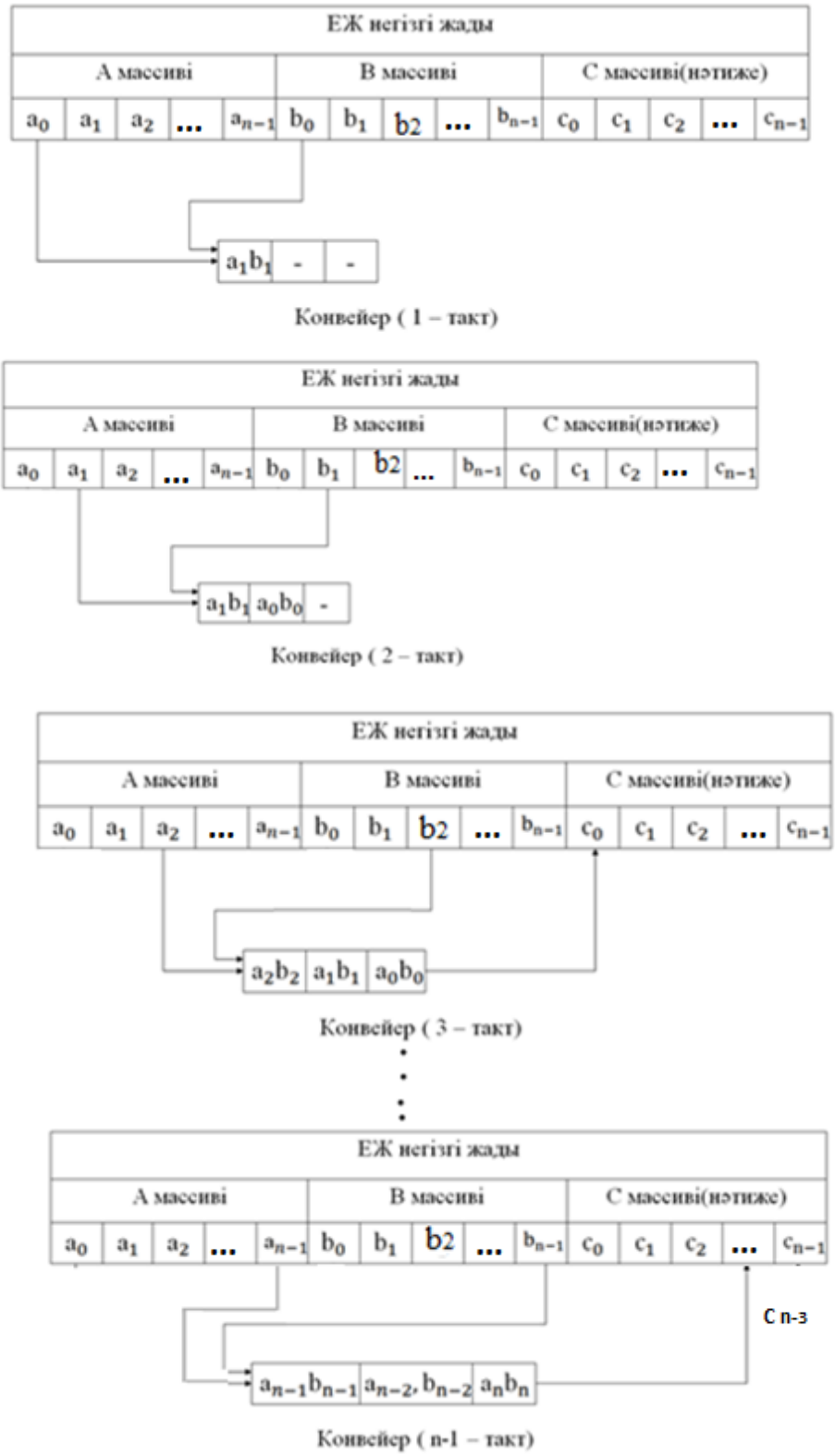
Дерек массивтерін өңдеу үшін бір команда арқылы массивтің барлық элементтерімен амалдар орындайтын есептеу жабдығы – векторлық өңдеуіш жабдығы қажет. Мұндай есептеу жабдығына векторлық процессор жатады. Векторлық процессорларда командалардың операндтарына деректер массиві -

векторлар жатады. Векторлық процессорлар векторлы-параллельді және векторлы-конвейерлік деп аталатын екі вариантпен құрылады. Векторлы-параллельдік процессорларда вектор элементтерімен (жылжымалы үтірлі сандармен) операция параллель бірнеше операциялық құрылғылар (ОҚ) арқылы жүргізіледі (15.2-сурет).



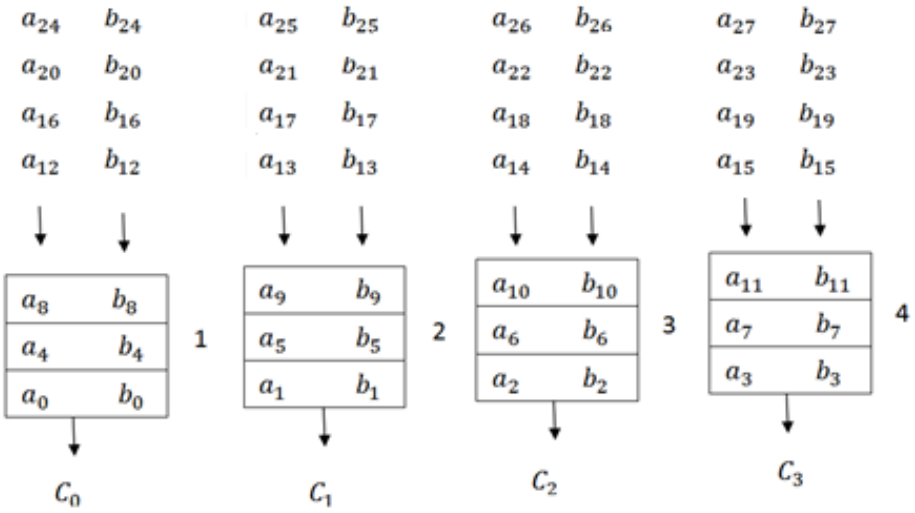
15.2-сурет. Векторлы-параллель өңдеу

Векторлы-конвейерлі процессорларда (15.3–сурет) вектор элементтері бір конвейерлі операциялық құрылғы арқылы жүргізіледі. Өңдеу кезінде жылжымалы үтірлі сандарды өңдеу бірнеше кезеңдерге бөлінеді. Конвейер саты саны осы кезең санымен анықталады. Кезекті операндтар жұбы конвейер кірісіне конвейердің бірнші сатысы босағанда беріледі.



15.3-сурет. Векторлы-конвейерлік өңдеу

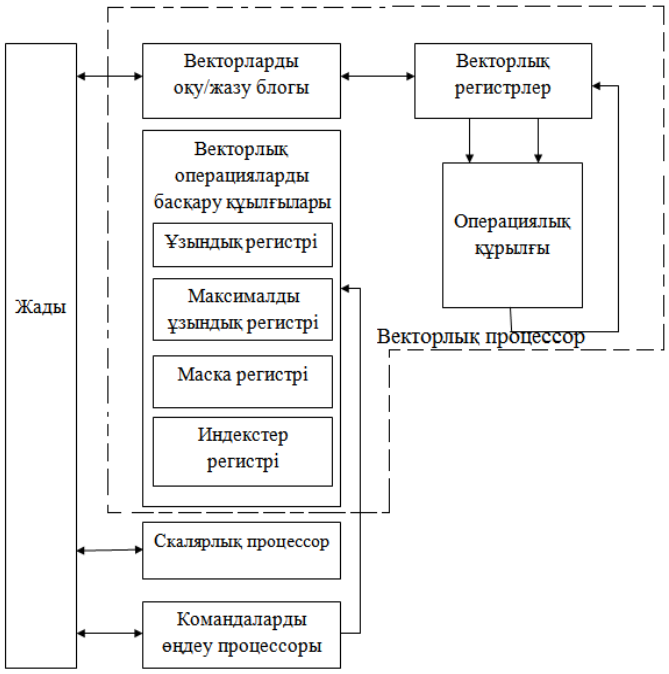
Бірнеше векторлық элементтерін қатар (бір уақытта) орындау үшін бірнеше конвейерлік операциялық құрылға қажет. 15.4-суретте $A = a_{27}a_{26} \ldots\ldots a_1a_0$ және $B = b_{27}b_{26} \ldots\ldots b_1b_0$ массивтерінің әрқайсысының үш сатылы төрт конвейер арқылы өңдеу реті көрсетілген. b_2



15.4-сурет. А және В векторларын үш сатылы төрт конвейер арқылы өңдеу

Деректер массиві операциялық құрылғыларға негізгі жадыдан немесе векторлық регистрлер арқылы беріледі. ЕЖ бірінші түрі «жады-жады» сәулетті векторлық процессорлары деп аталса, екінші түрін «регистр-регистр» сәулетті векторлық процессорлар деп атайды. Олар FIFO режимінде жұмыс жасайды. Нәтижелер векторлық регистрлер арқылы ЕЖ негізгі жадына жазылады.

«Жады-жады» сәулеті алғашқы КПЖ-ны қолданды. Қазіргі есептеу жүйелерінде негізінен «регистр-регистр» сәулеті қолданылады. Оған Cray ReSearch Inc., Fujitsu, Hitachi және NEC фирмаларының векторлық КПЖ жүйелері жатады. Қазірі векторлық процессорлардың цикл уақыты 2-2,5нс. Векторлық процессордың негізгі құрылымдары 15.5-суретте көрсетілген.



15.5-сурет. Векторлы-конвейерлік ЕЖ оңайлатылған сұлбасы

Операндтардың (векторлардың) n құрамдас элементтері бір векторлық командамен өңделеді. Операциялық құрылғы жылжымалы үтірлі сандармен кез-келген операндтарды орындай алатын көпфункцияоналды конвейер арқылы немесе әр операция әртүрлі блоктарда орындалатын конвейерлік блоктар арқылы ұйымдастырылады. Векторлық процессордың жұмысы векторлық операцияларды басқару құрылғысымен басқарылады. Векторлық процессордың командалар жүйесіне мынадай командалар кіреді:

- бірінші ұяшық мекен-жайы көрсетілген жады ұяшықтарынан векторлық регистрлерге мәлімттерді жазу;
- векторлық регистрлерде жазылған векторлардың барлық элементтерімен тиісті операцияларды орындау;
- векторлық регистрлердегі деректерді жады ұяшықтарына бірінші мекен-жайынан бастап жазу.

Векторлық операцияларды басқару құрылғысының құрамына бірнеше арнайы регистрлер: вектор ұзындығының регистрі, максимальді ұзындық регистрі, маска регистрі және индекстер регистрі кіреді.

Вектор ұзындығының регистрлерінде өңделетін вектор элементтерінің ұзындығы жазылады. Ол арқылы элементтермен қанша жеке операциялар орындалатындығы анықталады. Максимальді ұзындық регистрі процессорда бір уақытта орындалатын максимальді векторлар элементтерінің саны көрсетіледі. Регистрдегі дерек ұзын векторларды бірнеше сегменттерге бөлуге мүмкіндік береді. Ал сегменттер саны параллель жұмыс жасайтын өңдеу блоктарының санымен анықталады.

Маска регистрі арқылы орындалмайтын операцияларды «жасыра» аламыз. Жады құрылғысында векторлардың элементтері белгілі бір ретпен орналасады. Векторлық операцияларды орындағанда индекстеу қадам мәнін көрсету керек. Индекстердің мәні индекс регистрінде көрсетіледі. Әдетте ЖККД есептеу жүйелерінде өңделетін бағдарламаларда векторлық операциялармен қатар скалярлық операциялар орындалады. Сондықтан векторлы-конвейерлік ЕЖ құрамына скалярлық процессорлар да кіреді. Командалар командаларды өңдеу процессорымен орындалады.

15.2. Матрицалық есептеу жүйелері

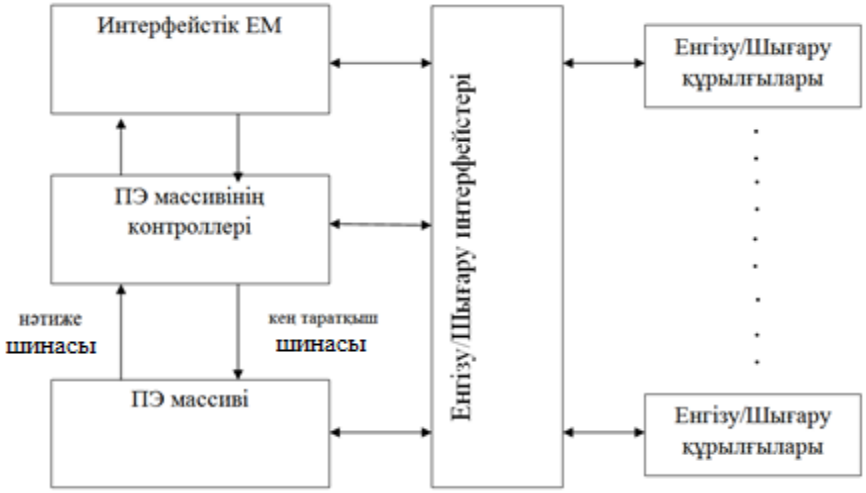
Матрицалық есептеу жүйелерінің негізін матрицалық процессор (array processor) құрады.

Матрицалық процессор деп ЖККД тәртібінде жұмыс жасайтын, біркелкі процессорлық элементтер массивінің белгілі бір логикалық матрицаға

біріктірілген күрделі құрылымын айтамыз. Мұндай құрылғыда барлық ПЭ синхронды режимде жұмыс жасайды, стандартты командаларды орындайды.

Матрицалық есептеу жүйесінің жалпы құрылымдық сұлбасы 15.6-суретте келтірілген.

Әртүрлі деректер элемнттерін параллель өңдеу ПЭ массиві арқылы жүргізіледі. ПЭ массивіне командалар, деректер және басқару сигналдары ПЭ массивінің контроллері (ПЭМК) арқылы генерацияланатын кең таратқыш шинасы арқылы беріледі. Өңдеу нәтижесі ПЭ массивімен ПЭМК кірісіне нәтиже шинасы арқылы жеткізіледі. Матрицалық ЕЖ құрамына бағдарламаларды құру және оларды жөндеу үшін интерфейстік ЕМ енгізіледі. Интерфейстік ЕМ ретінде әмбебап машиналар қолданылады. Интерфейстік ЕМ бағдарламаларды және деректерді ПЭМК-ге енгізу үшін де қолданылады. Енгізу/шығару құрылғаларды ЕЖ кез-келген құрылғыларымен енгізу/шығару интерфейстері арқылы алмасады.

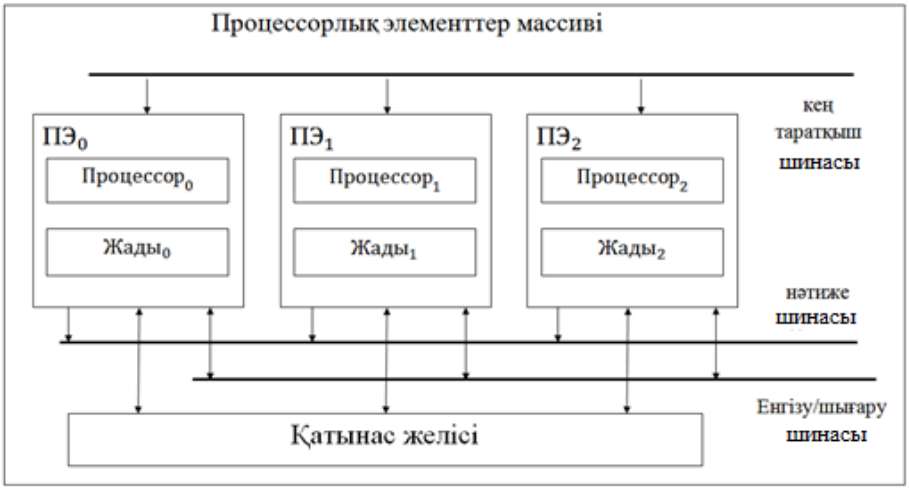


15.6-сурет. Матрицалық ЕЖ-нің жалпылама сұлбасы

Процессорлық элементтер массивінің құрамында көптеген процессорлық элементтерден басқа көптеген деректер жиынтығын сақтайтын жады модульдері болады. Оның үстіне ПЭ бір-бірімен және жады модульдерімен желі арқылы байланысады. Сонымен ПЭ массиві деп процессорлық элементтерінен басқа жады модульдері және оларды байланыстыратын қатынас желісінен тұратын күрделі құрылғыны айтамыз. 15.7–суретте ПЭ массивін ұйымдастырудың екі жолы көрсетілген.

Процессорлық элементтер массивін ұйымдастырудың бірінші түрінде (15.7, а-сурет) N процессорлық элементтер өзара қатынас желісі арқылы байланысады. Процессорлық элементтер кең таратқыш шиналары арқылы ПЭМК-дан командаларда алып өзінің жергілікті жадында сақталған немесе ПЭМК берілген деректерді өңдейді. ПЭ деректермен қатынас желісі арқылы, ал

енгізу/шығару құрылғыларымен енгізу/шығару шиналары арқылы алмасады. Нәтиже әрбір ПЭ-ден қатынас шинасы арқылы ПЭМК беріледі. ПЭМ ұйымдастырудың бірінші түрі MasPar MP-1, Connection Machine CM-2, GE-II, DAP, MPP, STARAH, PERE, ILLIAC-VI есептеу жүйелерінде қолданыс тапқан. Процессорлық элементтер массиві ұйымдастырудың екінші түрінде (15.7, б-сурет) N процессорлар жады модульдерімен екі бағытты қатынас желісі арқылы байланысады. Процессорлар басқару сигналдарын ПЭМК-дан кең таратқыш шинасы арқылы алады. Процессорлар дерекпен желі арқылы немесе жады модульдері арқылы алмасады. Жады модульдері бір-бірімен және енгізу/шығару құрылғыларымен енгізу/шығару шиналары арқылы байланыса алады. Нәтижелер әрбір жады модулінен нәтиже шинасы арқылы беріледі. ПЭМ ұйымдастырудың мұндай түрі Burroughs Scientific Processor (BSP), Texas Reconfigurable Array Computer TRAC жүйелерінде қолданыс тапқан.



а

б

15.7-сурет. ПЭ массивін ұйымдастыру: а–процессорлық элемент; б–«процессор жады»

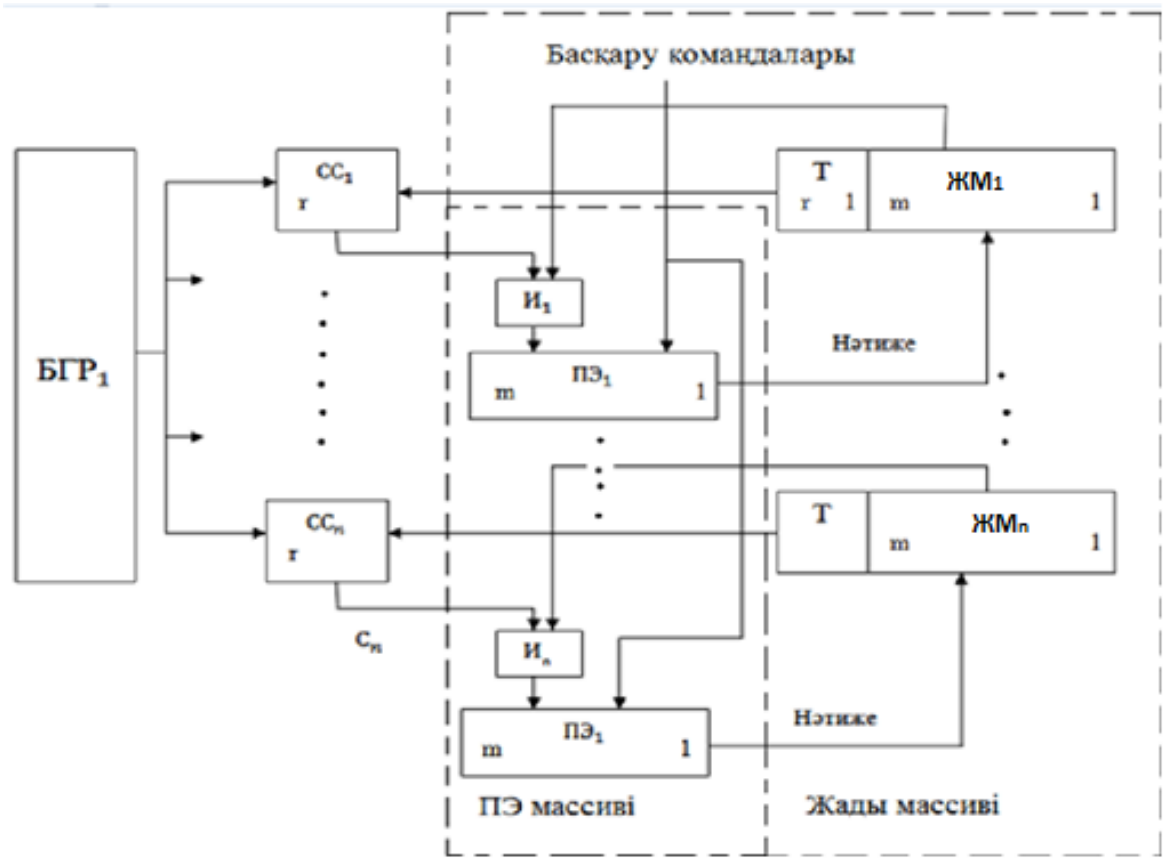
15.3. Ассоциативтік есептеу жүйелері

Ассоциативтік есептеу жүйелерінде матрицалық есептеу жүйелері сияқты, көптеген процессорлық элементтері бір команда арқылы көптеген деректерді параллель өңдейді. Матрицалық жүйеден айырмашылығы ассоциативтік ЕЖ-де мәліметтерге қатынас құру мекен-жай арқылы емес, операндтардың құрамындағы белгілер арқылы жүргізіледі.

Белгілер бойынша іріктелген деректер, ассоциативтік процессорларда (АП) өңделеді. АП негізінде құрылған ЕЖ-ні ассоциативтік есептеу жүйесі (АЕЖ) деп атаймыз.

Ассоциативтік процессор ассциативтік жады (АЖ) негізінде құрылады. Ассоциативтік процессордың құрамында ассоциативтік жадыдан басқа деректі өңдеу блогы және ассоциативтік белгілері сәйкес келетін барлық ұяшықтарға мәліметтерді параллель жазатын блогы болады.

Ассциативтік процессорлар құрамын 15.8-суретте көрсетілген сұлба арқылы көрсетуге болады.



15.8-сурет. Ассоциативтік процессорға түсініктеме сұлба

Ассоциативтік процессорлардың құрамына ПЭ массиві, жады массиві (ЖМ), салыстыру сұлбалары (СС) және белгілер регистрі (БГРr) кіреді. Жады массиві n ұяшықтардан құрылған. Әр ұяшық екі бөліктен – r разрядты белгіден (Т–тегтен) және m разрядты деректен (М) тұрады.

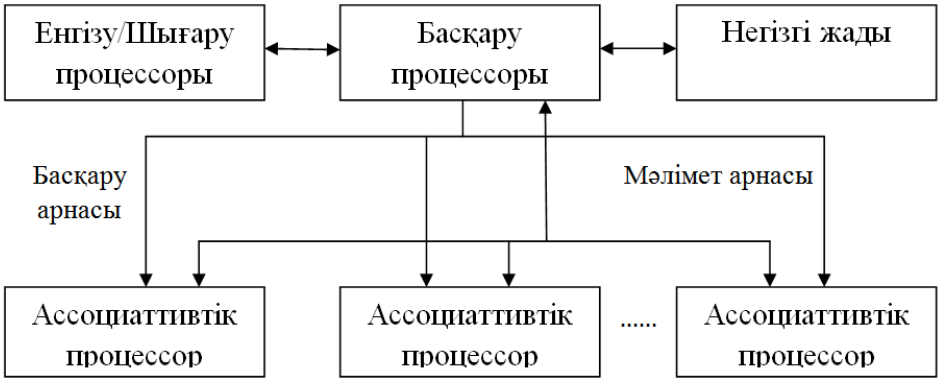
Суретте ПЭ саны салыстыру сұлбасының саны сияқты жады массивінің ұяшықтар санымен анықталады. ПЭ басқару процессорынан келетін командаларға сәйкес ұяшықтардан оқылатын m разрядты деректерді өңдейді.

Салыстыру сұлбалары r-разрядты ассоциативтік белгілер регистріндегі дерекпен әр жады ұяшығында жазылған r разрядты тег (Т) кодаларымен салыстырылып, $C_i = 1$ сәйкестік сигналын шығарады. C_i сигналдары арқылы ұяшықтардан m разрядты M_i деректер N_1/N_n сұлбалары арқылы ПЭ беріліп басқару командалары арқылы өңделеді. Алынған нәтижелер қайта жады массивінің ұяшықтарына жазылады. Процессордың элементтерде екілік сандармен әртүрлі арифметикалық және логикалық операциялар орындалады. Жоғарыда көрсетілген сұлбада процессорлық элементтердің операндтармен және нәтиже деректермен алмасу сұлбалары келтірілмеген.

Жады ұяшығындағы сөздерді өңдеу тәртібіне байланысты ассоциативтік процессорлар төрт түрге бөлінеді:

- параллельді (барлық сөздер және олардың разрядтары параллель ПЭ өңделеді);
- толық тізбекті (әр сөз және әр сөздің разрядтары тізбектеп бір разрядты ПЭ арқылы өңделеді);
- параллель тізбекті (әр сөз m разрядты ПЭ арқылы тізбекпен өңделеді);
- блокты (барлық сөздер топқа бөлініп блок құрады, әр блоктағы сөздер параллель өңделеді).

15.9-суретте көппроцессорлы ассоциативтік есептеу жүйесінің жалпы сұлбасы келтірілген.



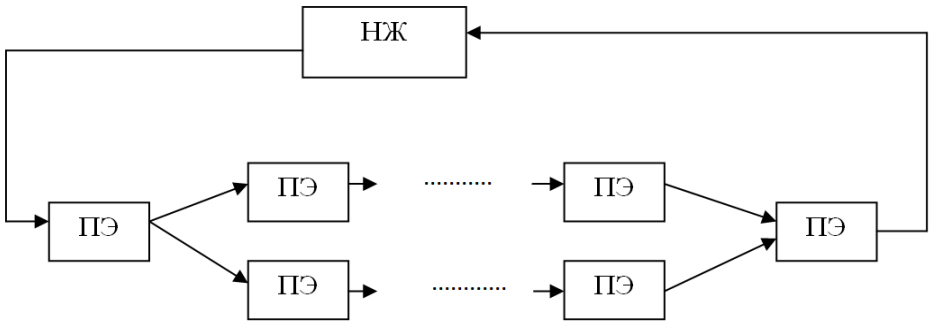
15.9-сурет. Ассоциативтік есептеу жүйесінің құрылымы

АЕЖ бағдарламасы негізгі жадыда сақталады. Бағдарламамен жұмыс жасау үстінде басқару процессоры НЖ-дан кезекті команданы алып оны декодтайды. Егер команда ассоциативтік процессорлар арқылы орындалу керек болса, онда ол басқару арнасы арқылы барлық ассоциативтік процессорларға беріледі. Басқару процессоры сонымен қатар негізгі жадыдан ассоциативтік процессорлардың жергілікті жадысына деректер арнасы арқылы тиісті деректерді беруді басқарады. АЕЖ сырт құрылғыларымен енгізу/шығару құрылғылары арқылы байланысады. Басқару процессоры бағдарламаларды трансляциялауға, оларды редакциялауға, өңделетін деректерді параллельдеуге қолданылады.

15.4. Систоликалық құрылымды есептеу жүйелері

Систоликалық құрылымдарда негізгі жадыдан оқылған деректер оған қайта оралудан бұрын көптеген процессорлық элементтер арқылы өңделеді (15.10-сурет).

Егер есептеу жүйесінің құрылымын тірі организмге ұқсататын болсақ, онда жады құрылғысын жүрек деп, көптеген ПЭ-лерді ағза деп, ал деректер ағынын айналасындағы қан деп қарауға болады. Осыдан систоликалық матрица (систола–жүректің қан қуысының жиырылуына байланысты қанның артериямен айналуы).



15.10- сурет. Систоликалық құрылымды есептеу жүйесіне деректерді өңдеу

Систоликалық құрылымдар матрицалық есептеулерді шығаруда, әртүрлі сигналдарды өңдеуде, деректерді сұрыптауда т.б өте тиімді. Систоликалық құрылымдар деп конвейерлік және матрицалық өңдеу мүмкіндігі бар біртекті ПЭ-ден құрылған есептеу ортасын айтамыз.

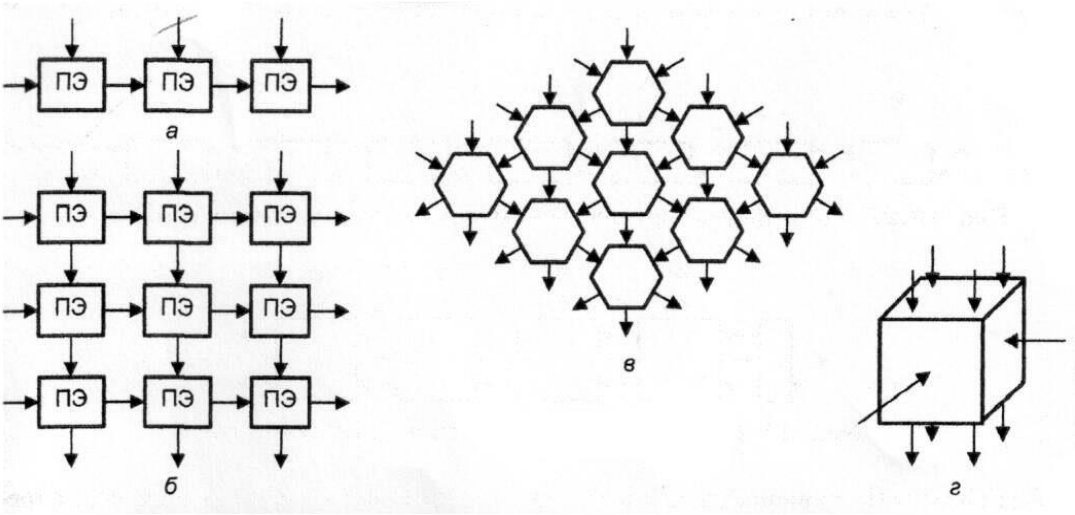
Систоликалық құрылымдардың негізгі ерекшеліктері:

- систоликалық құрылымдарда есептеу үрдісі кезінде деректер бір процессорлық элементтерден екіншісіне аралық нәтижелері сақталмай бір ырғақпен тізбекті түрде беріліп отырады;

- кіріс деректерінің әрбір элементі жады құрылғысынан бір рет орындалатын алгоритмге байланысты, олар бірнеше рет операцияға қатыса алады. Дерек шеткі ПЭ арқылы енгізіледі;
- жұмыс жасау алгоритмі деректерді конвейермен параллель өңдеуге мүмкіндік береді;
- матрица өнімділігін ПЭ санын көбейту арқылы өсіруге болады.

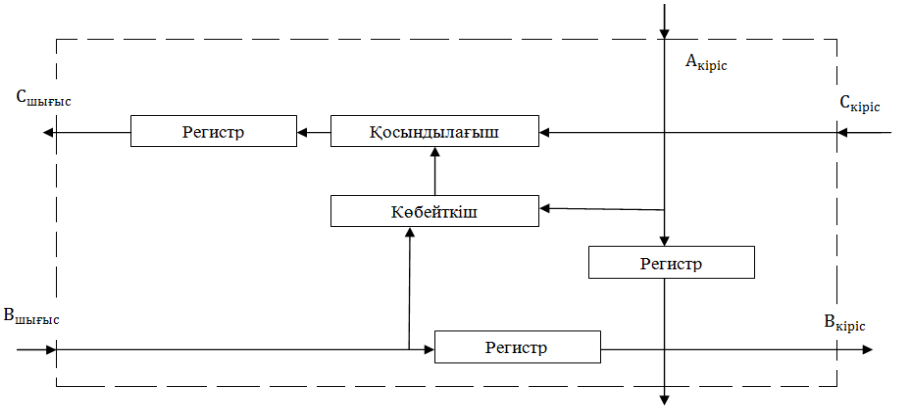
Қазіргі систоликалық процессорлардың өнімділігі – 1000 млрд операция/сек.

Қазіргі кезде систоликалық құрылымдардың әр түрі – сызықтық, квадратты, гексогоналды, үш өлшемді т.б геометриялық байланысы бар түрлері белгілі (15.11–сурет).



15.11-сурет. Систоликалық матрицалар конфигурациясы; а–сызықтық; б–тікбұрышты; в–гексогональді; г–үш өлшемді

Систоликалық матрицаның процессорлық элементінің функционалдық сұлбасы 15.12-суретте келтірілген.

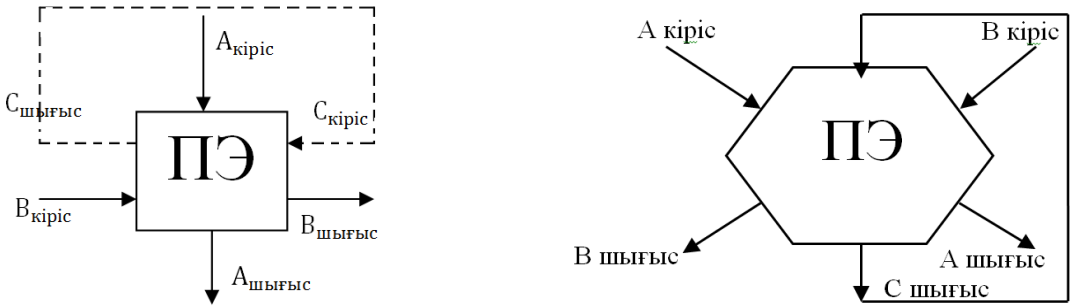


15.12-сурет. ПЭ функционалдық сұлбасы

Суреттен:

$$C_{шығыс} = C_{кіріс} + A_{кіріс} * B_{кіріс}$$

15.13-суретте көрсетілген ПЭ негізінде алынған тікбұрышты және гексоганальді ПЭ матрицаларының құрылымдары көрсетілген.



а б

15.13-сурет. ПЭ құрылымдары: а–тікбұрышты систоликалық матрица құрылымдары үшін, б–гексагональді систоликалық матрица құрылымдары үшін

Тікбұрышты систоликалық матрицалар құрылымы матрицаны векторға көбейтуді, ал гексогональді систоликалық матрица құрылғылары матрицаларды көбейтуге кеңінен қолданылады.

Мысал ретінде А матрицасын Х векторына тік бұрышты систоликалық матрица арқылы көбейтуді қарастырайық (n=4).

$$\begin{Vmatrix} a_{11} & a_{12} & a_{13} & a_{41} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{Vmatrix} * \begin{vmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{vmatrix} = \begin{vmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{vmatrix}$$

Мұнда

$$y_1 = a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4$$

$$y_2 = a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4$$

$$y_3 = a_{31}x_1 + a_{32}x_2 + a_{33}a_3 + a_{34}x_4$$

$$y_4 = a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4$$

Процессорлық элементтер шығысынан $y_1 \div y_4$ мәндерін алу үшін систоликалық матрицада процессорлық деректерді 15.14-суретте келтірілген тәртіппен орналастырамыз. Матрица элементтерін (a_{ij}) жоғарыдан төмен қарай, ал вектор элементтері (x_i) солдан оңға қарай белгілі бір ретпен 1 – ПЭ кірісіне беріледі. $y_1 \div y_3$ мәндері 1– ПЭ шығысында белгілі бір уақыт мезеттерінде қалыптасады.

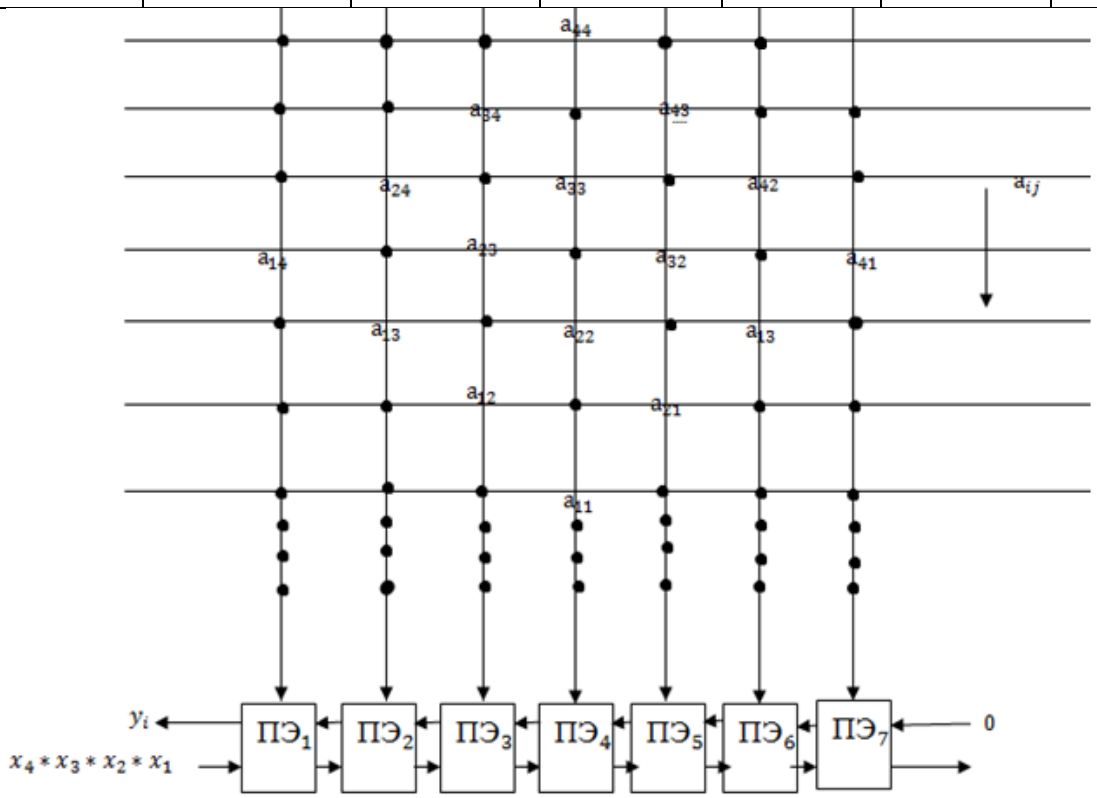
Матрица элементтері жоғарыдан төмен қарай ПЭ кірістеріне және x_1, x_2, x_3, x_4 мәндері $ПЭ_1$ төменгі кірістеріне такт сигналдары арқылы беріліп отырады. Әр такт сигналы аяқталған соң a_{ij} элементтері және x_{ij} сигналдары бір позицияға жылжытылып отырады. 4-такт сигналы аяқталған соң $ПЭ_4$ кірістеріне a_{11} және x_1 сигналы беріліп $ПЭ_4$ шығысында $y_1 = a_{11}x_1$ мәні қалыптасады. 5-такт сигналынан кейін $ПЭ_3$ элементінің шығысында $y_2 = a_{11}x_1 + a_{12}x_2$, ал $ПЭ_5$ элементінің шығысында $y_2 = a_{12}x_2$ мәні қалыптасады. $ПЭ_1$ элементінің шығысында $y_2 = a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4$ мәні 7-такт сигналынан кейін қалыптасады. $y_2 = a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4$ мәні 9-такт сигналынан кейін, ал y_3 және y_4 мәндері 11 және 13-такт сигналдарын бергеннен соң қалыптасады.

А матрицасын X векторына көбейту реті 15.1–кестеде көрсетілген.

15.1–кесте. А матрицасын X векторына көбейту кестесі

Такт сигналдары	**$ПЭ_1$**	**$ПЭ_2$**	**$ПЭ_3$**	**$ПЭ_4$**	**$ПЭ_5$**	**$ПЭ_6$**	**$ПЭ_7$**
1.	x_1						y_1
2.		x_1				y_1	
3.	x_2		x_1		y_1		y_2
4.		x_2		$y_1=a_{11}x_1$		y_2	
5.	x_3		$y_1=a_{11}x_1+a_{12}x_2$		$y_2=a_{21}x_1$		y_3
6.	$y_1=a_{11}x_1+a_{12}x_2+a_{13}x_3+a_{14}x_4$	$y_1=a_{11}x_1+a_{12}x_2+a_{13}x_3$		$y_2=a_{21}x_1+a_{22}x_2$		$y_3=a_{31}x_1$	
7.			$y_2=a_{21}x_1+a_{22}x_2+a_{23}x_3+a_{24}x_4$		$y_3=a_{31}x_1+a_{32}x_2$		$y_4=a_{41}x_1$
8.	$y_2=a_{21}x_1+a_{22}x_2+a_{23}x_3+a_{24}x_4$	$y_2=a_{21}x_1+a_{22}x_2+a_{23}x_3+a_{24}x_4$		$y_3=a_{31}x_1+a_{32}x_2+a_{33}x_3$		$y_4=a_{41}x_1+a_{42}x_2$	
9.			$y_3=a_{31}x_1+a_{32}x_2+a_{33}x_3+a_{34}x_4$		$y_4=a_{41}x_1+a_{42}x_2+a_{43}x_3$		x_2
10.	$y_3=a_{31}x_1+a_{32}x_2+a_{33}x_3+a_{34}x_4$	$y_3=a_{31}x_1+a_{32}x_2+a_{33}x_3+a_{34}x_4$		$y_4=a_{41}x_1+a_{42}x_2+a_{43}x_3+a_{44}x_4$		x_3	
11.			$y_4=a_{41}x_1+a_{42}x_2+a_{43}x_3+a_{44}x_4$		x_4		x_3

12.	$y_4=a_{41}x_1+a_{42}x_2+ a_{43}x_3+ a_{44}x_4$	$y_4=a_{41}x_1+a_{42}x_2+ a_{43}x_3+ a_{44}x_4$				x_4	
13.							x_4
14.							



15.14-сурет. А матрицасын Х векторына тікбұрышты систоликалық матрица арқылы көбейтуді ұйымдастыру сұлбасы

Бақылау сұрақтары:

1. ЖККД есептеу жүйелерінде деректерді өңдеу кезінде қандай параллелизм деңгейін іске асырады?
2. Векторлық өңдеу жүйесі қандай деректермен жұмыс жасауға бейімделген?
3. Векторлы-параллель жүйесінің векторлы конвейерлік жүйеден айырмашылығы неде?
4. Векторлық және матрицалық есептеу жүйелерінің айырмашылығын атаңыз.
5. Ассоциативтік жады мен ассоциативтік процессордың айырмашылығын түсіндіріңіз.

6. Систоликалық есептеу жүйесінің аталуын қалай түсіндіруге болады?
7. Өлшемдері n=3 матрицаны векторға көбейту үшін қанша ПЭ керек?
8. Өлшемдері n=5 А матрицасын В матрицасына көбейту үшін неше гексогональді конфигурациялы ПЭ керек?

16.КККД КЛАСТЫ ЕСЕПТЕУ ЖҮЙЕЛЕРІ

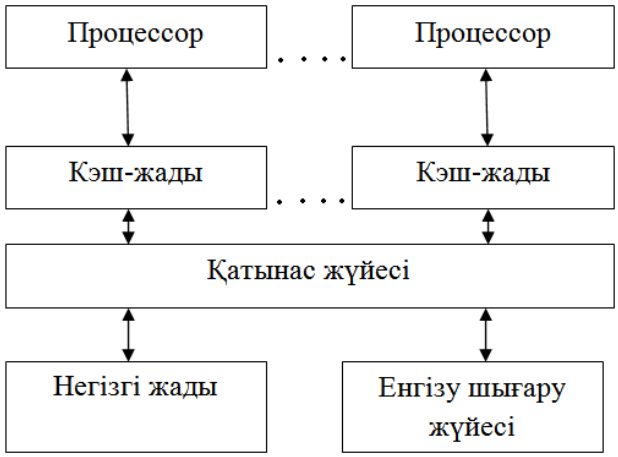
ККҚД класты ЕЖ-де әрбір процессорлық элементтер өз бағдарламасымен бір-бірінен тәуелсіз жұмыс жасайды. Процессорлық элементтер құрамын меншікті жадысы жоқ процессорлар немесе әрқайсысының өз жадысы бар есептеу машиналары құрады. ККҚД есептеу жүйесінің бірінші түрін «көппроцессорлы», ал екінші түрін «көпкомпьютерлі» есептеу жүйелері деп екіге бөледі.

16.1. Ортақ жадылы КККМ ЕЖ

Ортақ жадылы ККҚД есептеу жүйесіне симметриялық көппроцессорлы жүйе (SMP), параллель векторлық жүйе (PVP) және қолжеткізімділігі әртүрлі NUMA жүйесі жатады.

Симметриялы көппроцессорлық жүйе

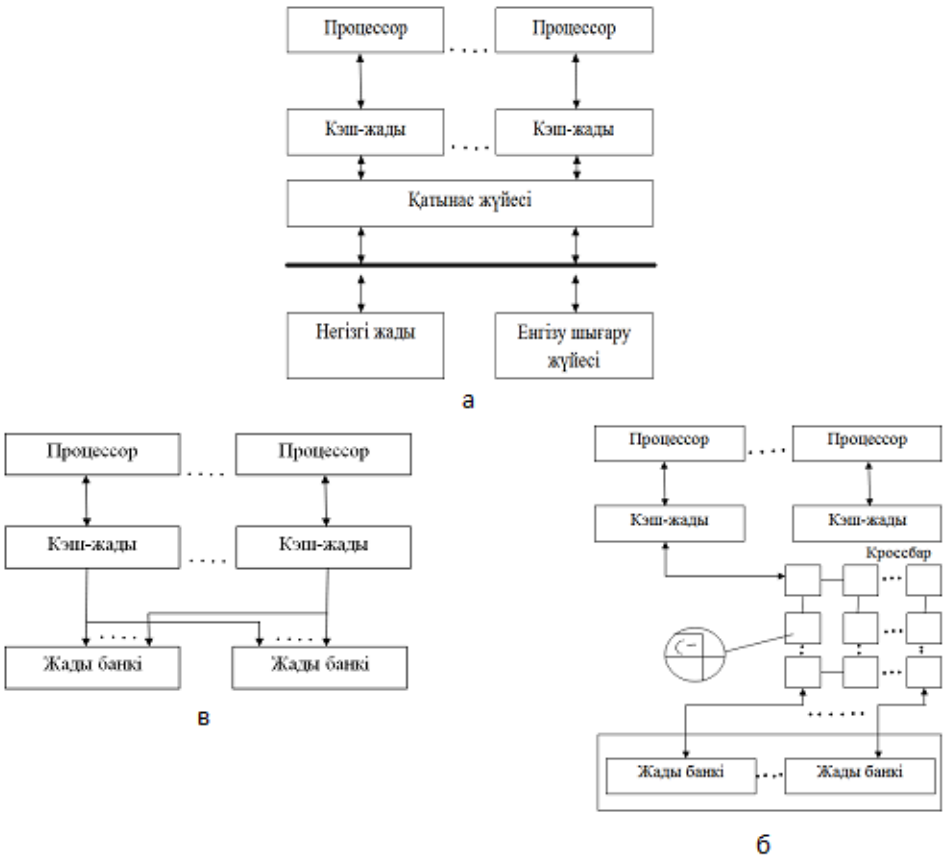
SMP жүйесі көптеген ПЭ-дан тұрады. Олардың әрқайсысы блок принципімен құрылған жадыға бірыңғай қатынас құрып, UMA концепциясын іске асырады. Барлық ПЭ мүмкіндігі бірдей процесссорлардан құрылған. Сондықтан мұндай жүйені симметриялы көппроцессорлы жүйе деп атайды. Барлық ПЭ операциялық жүйемен (ОЖ) басқарылады. ПЭ ретінде көбінесе RISC - процессорлар қолданылады. Жергілікті КЭШ-жадылы симметриялы көппроцессорлардың ұйымдастыру сұлбасы 16.1-суретте келтірілген.



16.1– сурет. SMP жүйесі құрылымының сұлбасы

SMP жүйесіндегі процессорлардың ортақ ресурстарымен алмасу тәсілдеріне байланысты SMP -жүйесінің мынадай архитектурасын атауға болады.

- ортақ шиналы (16.2, а- сурет)
- «кроссбар» типті коммутаторлы (16.2, б- сурет)
- көппортты жадылы (16.2, в- сурет)



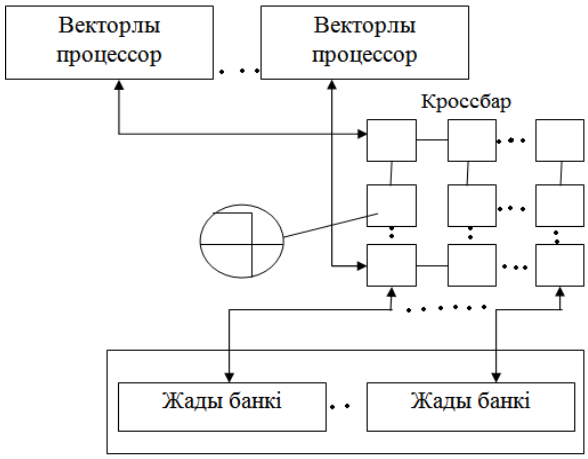
16.2-сурет. SMP жүйе сәулеті; а-ортақ шиналы, б-кроссбар, в-көппортты жадылы

Ортақ шиналы SOM Compaq Alpha Server GSMO және 4820 жүйелерінде 14 Alpha 2,264 процессорлары арқылы құрылған. HP N900 жүйесі 8 PA-8500 процессорлардан тұрады. Кроссбар типті коммутаторы бар SMP-ге Enterprise 10000 жүйесі жатады. Мұнда процессор саны - 64. IBM RS/6000 Enterprise Server Model S70 "кроссбар типті" коммутатор 12 RS64 процессор арқылы құрылған.

Көппортты жадылы SMP жүйесінде негізгі жадының блоктарына кез-келген процессорлар және енгізу/шығару модульдері тікелей қатынас құра алады (16.2, в-сурет).

Параллель векторлық жүйелер

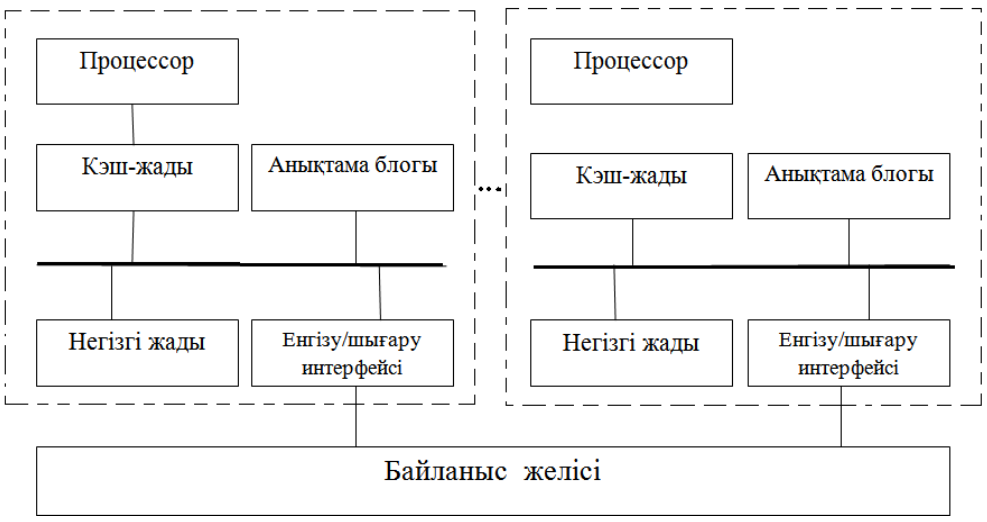
КККД үлестірілген ортақ жадылы жүйеге параллель векторлық жүйе (PVP, Parallel Vector Processor) жатады. PVP жүйесінде процессорлық элементтер ретінде векторлы-конвейер процессорлары қолданылады. Мұндай конвейерлердің өнімділігі 1 GELOPS-тан төмен емес. PVP-жүйесіне жылдамдығы өте жоғары «кроссбар» типті коммутатор қолданылады (16.3-сурет). PVP-жүйесіне жапон фирмаларының NEC, Fujitsu және Hitachi жүйелері жатады. Мысалы VPP жүйесінің құралы 8-ден 256-ға дейінгі процессорлық элементтерден тұрады. Мұндай ЕЖ-нің жоғарғы өнімділігі 4,9 TFLOPS.



16.3-сурет. PVP жүйесінің архитектурасы

Жады қолжетімдігі әртүрлі ЕЖ

Жады қолжетімділігі әртүрлі ЕЖ біріне ccNUMA сәулетті ЕЖ жатады. 16.4-суретте ccNUMA жүйесінің типтік сұлбасы келтірілген. Жүйе көптеген бір-бірінен тәуелсіз есептеу түйіндерінен тұрады. Есептеу түйіндері бір-бірімен кроссбар немесе сақина тәрізді желілер арқылы байланысады. Әр түйін КЭШ-жадымен жабдықталған процессордан, жүйе тұрғысынан глобальді кеңістігінде орналасқан мекен-жайы бар ортақ жады құрылғысының бір бөлігі болып табылатын жергілікті жадыдан тұрады. Әр түйінде түйін анықтама блогы (АБ) бар. Ол ішкі шинаға қосылған. Ішкі шина арқылы АБ есептеу түйінінің кез-келген құрылғысына және жүйе байланыс желісі арқылы кез-келген түйіннің АБ-на байланыса алады. АБ бір-бірімен әрекеттесе отырып кез-келген деректің кез-келген уақытта қай түйіннің КЭШ-жадысына немесе негізгі жадысында сақтаулы тұрғанын бақылай алады. Сол арқылы АБ деректерде бір түйінмен екіншісіне алмасуға және жады құрылғыларындағы сәйкессіздіктерді болдырмауға мүмкіндік алады.



16.4– сурет. ccNUMA типті жүйені ұйымдастыру сұлбасы

NUMA жүйесіне мысал ретінде 1024 R1000 процессорлардан құрылған Silicon Graphics Origin жүйесі, Segment NUMA-Q жүйелерін келтіруге болады.

16.2. Үлестірілген жадылы КККД жүйелері

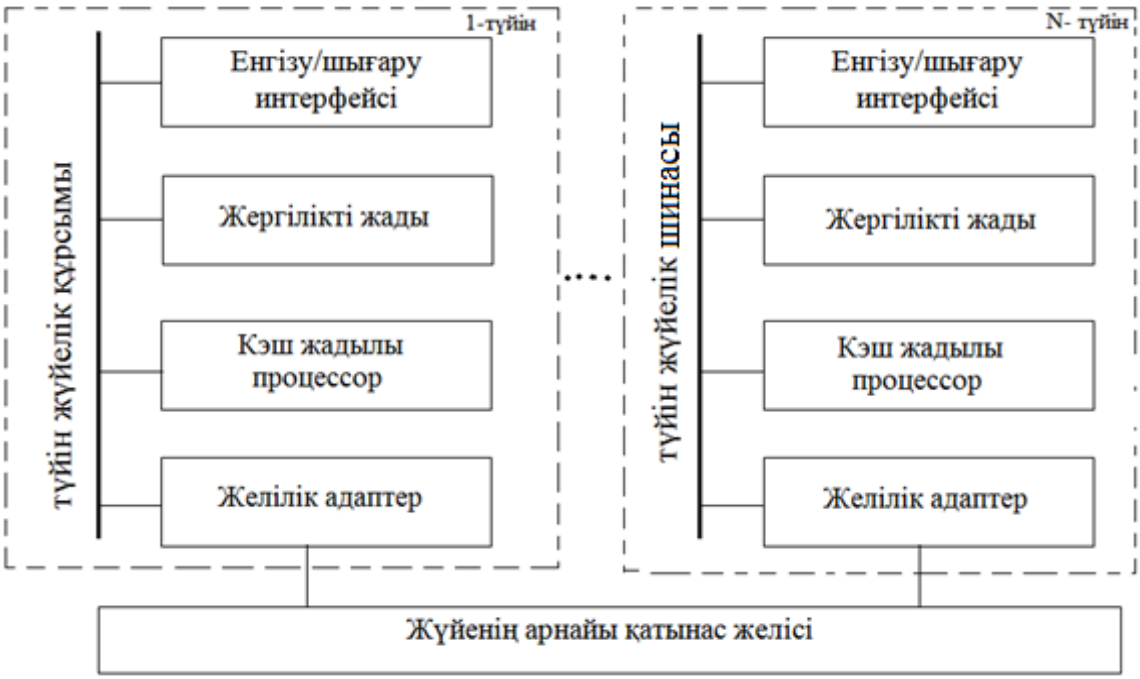
Үлестірілген жадылы КККД жүйелерінде процессорлар тек өздерінің жергілікті жадына тікелей қатынас құра алады. Басқа процессорларға қатынас құру арнайы қатынас желісі арқылы жүзеге асырылады. Үлестірілген жадылы КККД жүйелеріне MPP (Massively Parallel Processing–жаппай параллель өңдеу) жүйесі және кластерлік есептеу жүйелері мен үлкен SMP-жүйелер негізінде құрылған кластерлер жатады.

Жаппай параллель өңдеу жүйесі (MPP)

Өнімділігі аса үлкен жүйелерді жасауға арналаған зерттеулер MPP жүйелерінің пайда болуына алып келді. MPP жүйелерінде өнімділік оны құратын түйіндер санына байланысты болады. Ал түйіндер мөлшері оған жұмсалған қаражатқа байланысты болады.

MPP жүйесі мыңдаған біркелкі есептеу түйіндерінен тұрады. Есептеу түйіндері процессордан, жады құрылғысы мен енгізу/шығару құрылғысынан және қатынас құрылғыларынан тұрады, яғни толыққанды есептеу машинасын құрады. Түйіндер бір-бірімен жылдам жұмыс жасайтын қатынас құрылғыларымен байланысқан.

MPP жүйесінің жалпылама құрылымы 16.5-суретте келтіріліген.



16.5- сурет. MPP жүйесінің құрылымдық сұлбасы

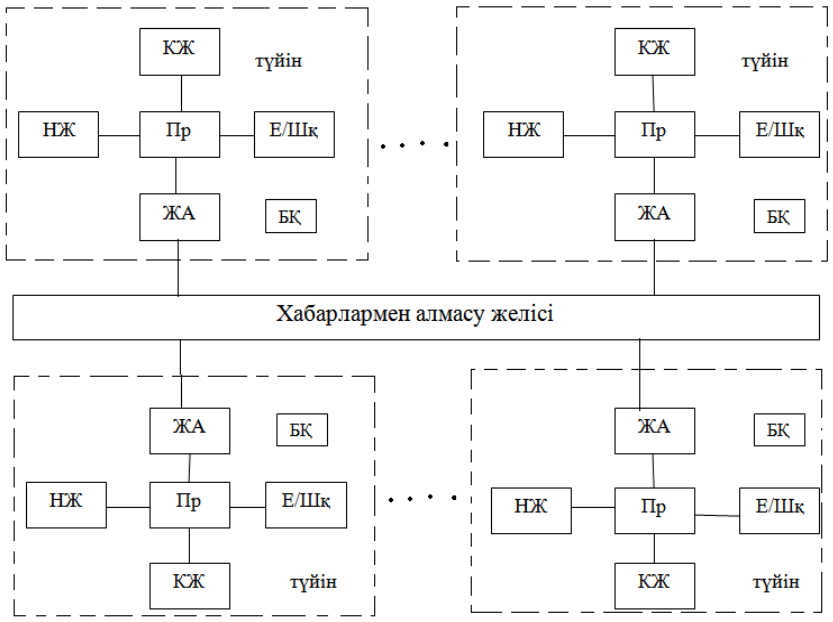
MPP түйіндерінің жұмыс реті бас басқару есептеу машинасымен (хост компьютерімен) басқарылады. Хост-компьютерлер ретінде Cray T3D жүйесіндегідей жеке есептеу машинасы болуы мүмкін, не жүйе түйіндерінің бірі болуы мүмкін.

Хост-компьютер немесе оны алмастыратын жүйе түйіндерінің бірі әртүрлі тапсырмаларды кезекпен босаған түйіндерге жүктеп отырады.

Егер жүйе құрамындағы процессорлар саны n≥128 болса, онда әдетте ондай жүйелерді MPP жүйесіне жатқызады. 16.6-суретте көрсетілген жүйе құрылымы мыңдаған процессорларды біріктіруге мүмкіндік береді. Мысалы Jaguar Cray XT-HE жүйесі 22.4256 есептеу ядросынан тұрады. Жады сыйымдылығы 6 Gb. бағдарлауыш қызметін атқаратын желілік құрылғы ЕЖ барлық түйіндерін біріктіріп, бір-бірімен хабар алмасуды ұйымдастырады. Мұнда есептелген жүйе өнімділігі 233 TFLOPS.

Кластерлік есептеу жүйелері

Қазіргі есептеу жүйесінің даму бағыттарының бірін кластерлеу деп атайды. Кластер деп бірлестірілген есептеу ресурстарын ұйымдастыра отырып, бір (жалғыз) машина не жүйе кейіпін құратын, не бір-бірімен байланысып жұмыс істейтін есептеу түйіндер тобы. Кластерлі түйін ретінде бір процессорлы есептеу машинасы, SMP немесе MPP тәрізді есептеу жүйелері бола алады. Сәулет тұрғысынан алғанда кластер - жылдамдығы жоғары желі арқылы біріктірілген бірнеше түйіндер жиынтығы (16.6–сурет).



16.6-сурет. Кластерлік жүйенің құрылымдық сұлбасы

Суретте КЖ- КЭШ жады, НЖ-негізгі жады, ЖА-желілік адаптер, БҚ-байланыстырушы қатпар.

Кластерлік есептеулер терминінен басқа гипересептеу (hyper computing), желі арқылы параллель есептеу (network-cased concurrent computing), ультраесептеу (ultracomputing) терминдері жиі қолданылады.

Кластерлер түйіндері ретінде біртипті (гомогенді кластер) және әртүрлі типті (гетерогенді кластер) есептеу жүйелері қолданылады. Кластерлік жүйені дамыту негізінен екі мәселеге байланысты: біріншісі - өнімділігі өте жоғары есептеу жүйесін құру; екіншісі - есептеу жүйесінің тоқтап қалмаушылығын өсіру.

Әртүрлі түйіндердегі аппараттық құрылғыларды ЕЖ кластеріне айналдыру байланыстырушы қатпар (БҚ) арқылы іске асырылады. БҚ-ның негізгі міндеті есептеу түйінін бір машина кейіпін құру және есептеу жүйесінің тоқтап қалмаушылық қасиетін қамтамасыз ету. Байланыстырушы қатпардың негізін арнайы бағдарламалық қамтама (БҚПҚ) құрады. Әр модуль БҚ әртүрлі функцияларын орындайды.

БҚПҚ операциялық жүйемен (ОЖ) қолданбалы бағдарламалардың аралығында орналасқан. ОЖ жоғарғы деңгейін кластерлік деңгей бағдарламалары бір-бірімен арнайы хабар арқылы (кластерлік хабар) әрекеттеседі. Алмасу қатынастық желі арқылы жүргізіледі.

Жалғыз машина кейіпін құру үшін байланыстырушы қатпар мынадай міндет атқаруы керек:

- сырттан жасалынатын қатынас кластердің тек бір кірісі арқылы құрылуы;
- файлдардың бір иерархиямен ұйымдастырылуы;
- үлестірілген ортақ жады архитектурасы арқылы бірыңғай мекен-жай кеңістігін қамтамасыз етілуі;
- енгізу/шығару кеңістігінің бірлігі;
- тапсырмаларды басқару бірлігі;
- пайдаланушылар интерфейстерінің бірлігін қамтамасыз ету.

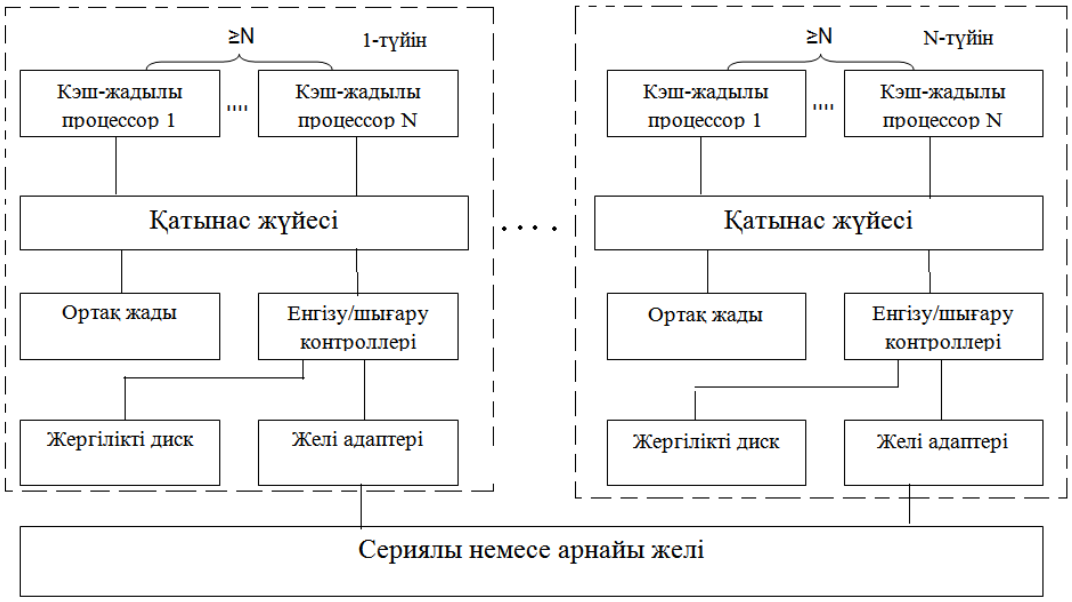
ЕЖ-нің тоқтап қалмаушылық мәселесін шешу кезінде, егер кластерлік түйіннің бір немесе бірнешеуі істен шықса, немесе шатасу орын алса, кластерлік деңгейдің бағдарламасы арқылы олардың жүктемелері қалған түйіндерге бөліне алады.

Кластерлік жүйе екі түрде құрылады. Бірінші түріне кіші жергілікті желі арқылы құрылған кіші кластерлік жүйе жатады. Мұндай жүйеде әрбір ЕМ жеке тәуелсіз машина режимінде немесе кластерлік жүйенің бір түйіні ретінде жұмыс жасай алады.

Екінші түрінде кластерлік жүйе өнімділігі өте жоғары есептеу жүйесін құруға арналған есептеу машиналарының жүйелік блоктарынан тұрады. Жүйе хост-компьютер арқылы басқарылады. Оған кластерге керекті дискілер мен әртүрлі сырт құрылғылары қосылған.

Ірі кластерлік жүйеге мысал ретінде Қытайдың Tianhe - жүйесін атауға болады. Жүйенің жоғарғы өнімділігі 33862 TFLOPS. Кластердің түйіні ретінде көпядролы Intel Xeon E5 процессорлары қолданылған. Tianhe-2 жүйесінің ядро саны 3120000.

Түйіндері SMP-жүйелері арқылы құрылған кластерлерін Constellations (шоқжұлдыз) жүйесі деп атайды (16.7–сурет).



16.7– сурет. Constellation жүйесі

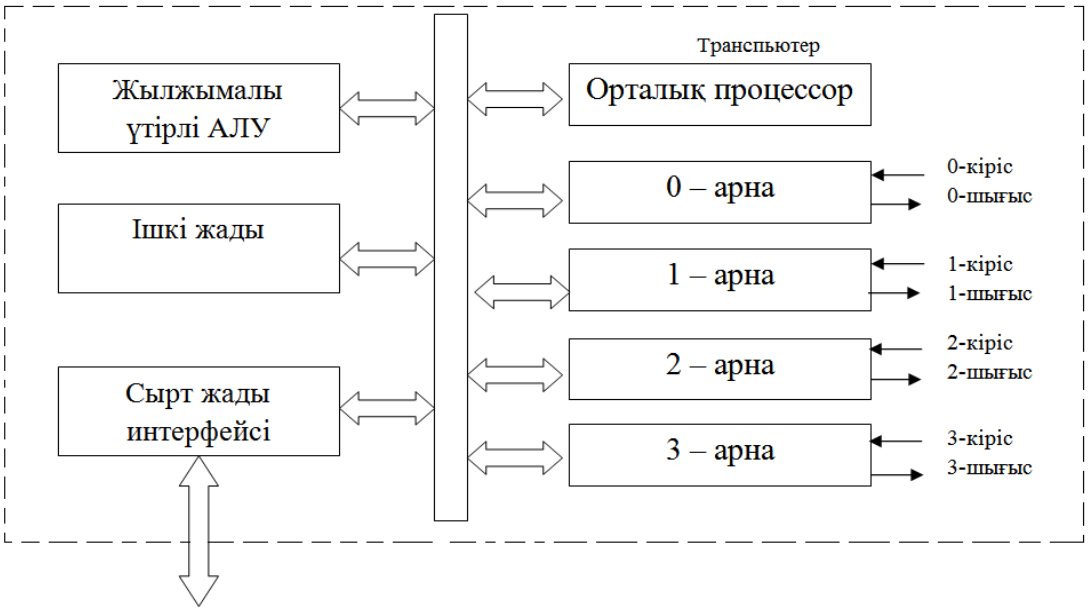
Constellation жүйесіне мысал ретінде Bull фирмасының Tera IO жүйесін келтіруге болады. Жоғарғы өнімділік 58,8 TFLOPS, кластер 544 түйіннен тұрады. Әрбір түйін 2-ядролық Itanium 2 процессорларынан тұратын SMP-жүйе.

Транспьютерлер негізінде құрылған есептеу жүйелері

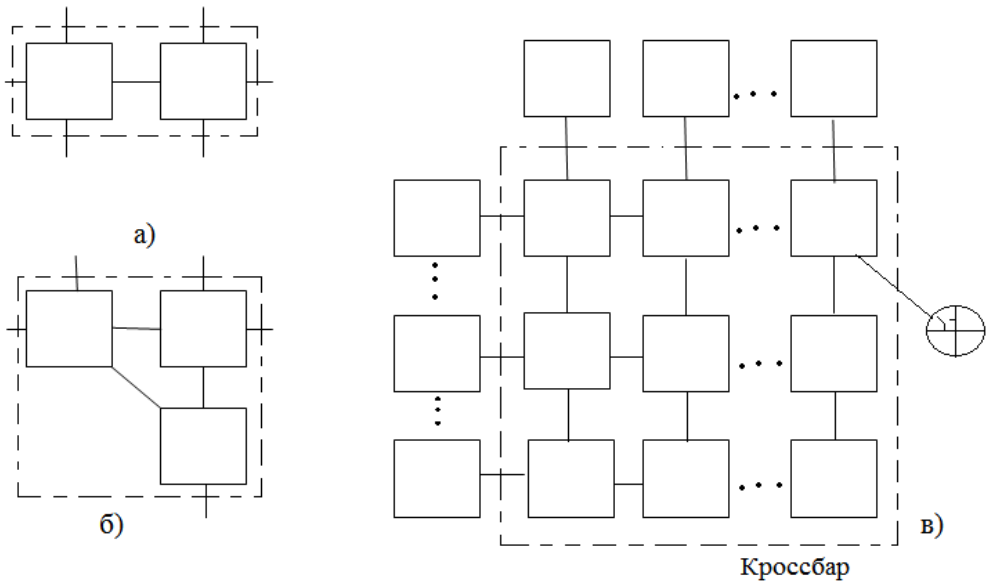
Транспьютерлер арқылы оларды бір-бірімен тікелей байланыстыра отырып өнімділігі әртүрлі жүйелерді құруға мүмкіндік туды. Транспьютер «транзистор» және «компьютер» деген екі сөзден тұрады. Транспьютерлер – шоғырлану дәрежесі өте үлкен шағынсұлба. Оның құрамы орталық процессор, жылжымалы үтірлі арифметикалық-логикалық құрылғы, статикалық жедел жады, сырт жадының интерфейсі және бірнеше алмасу арналарынан тұрады (16.8-сурет). Байланысу арнасы екі жақты алмасу үшін екі тізбекті желіден тұрады. Тізбекті желінің бірі арқылы дерек беріп, екіншісі арқылы берілген деректің дұрыстығы расталады. Транспьютерлер бір-бірімен тікелей біріктіріле

алады. Деректер пакет түрінде беріледі. Әрбір пакетті берген соң дерек алушы пакет арқылы оны алғандығын растайды.

Транспьютерлер негізінде әртүрлі ЕЖ құруға болады. Мысалы, транспьютердің төрт байланыс арнасы екі өлшемді массивті құруға мүмкіндік береді. Мұнда әрбір транспьютер жақын орналасқан төрт көрші транспьютерлерімен байланысқан.



16.8 -сурет. Транспьютер құрылымы



16.9- сурет. Толық байланысқан транспьютерлер, а–екі транспьютер, б–үш транспьютер, в–кроссбар негізінде құрылған жүйе

Егер транспьютерлер тобы екі транспьютерлерден тұрса, онда оларды басқа топқа қосу үшін байланыс арналары бос болады (16.9, а-сурет). Егер топ үш транспьютерден тұрса, онда да бос каналдар саны алты болады (16.9, б-

сурет). Транспьютерлі жүйені кроссбар негізінде де құруға болады (16.9, в-сурет). Транспьютерлермен жұмыс жасау ерекшеліктерін ескере отырып оларға арнайы OCCAM деп аталатын бағдарламалық тілі жасалынған. OCCAM тілі қарапайым екі нүкте арасында деректі жіберуді және бағдарламаны бірнеше транспьютерлер арқылы орындауды көрсетуге мүмкіндік береді. OCCAM тілінің негізгі ұғымы - үдеріс (процесс), бағдарламаның бір немесе бірнеше тізбекті не параллель орындалатын операторларынан тұрады. Үдерістер ЕЖ транспьютерлеріне бөлінуі мүмкін, бөлінген жағдайда транспьютерлер құрылғыларында бір немесе бірнеше үдерістер орындалуы мүмкін.

Бірінші транспьютер (T212) 16-разрядты арифметикалық процессордан тұрды. Келесі транспьютерлердің құрамында 32-разрядты бүтін санды процессор және жылжымалы үтірлі процессорлар (T700-T9000) болды. T9000 транспьютерінің құрамына КЭШ-жады мен арна процессоры қосылды. Басқару құрылғысы - микробағдарламалық қарапайым операциялар (қосу, алу) бір циклда, ал күрделі операциялар бірнеше циклда орындалады.

Бақылау сұрақтары:

1. Симметриялы есептеу жүйелерінде қандай параллель өңдеу деңгейі іске асырылады?
2. SMP жүйелерінің КЭШ жадында деректерді сәйкестендіру (когеренттілікті қамтамасыз ету) қандай жолдармен іске асырылады?
3. Матрицалық есептеу жүйесі мен симметриялық көппроцессорлар немен айрықшаланады?
4. Есептеу жүйелерін кластерлік жолмен ұйымдастыру арқылы қандай екі мәселені шешуге болады?
5. Кластерлік есептеу жүйесінде кластерлік бағдарламаларға қандай міндет жүктеледі?
6. MPP жүйелерінде процессорларға тапсырыстар қалай үлестіріледі және олардың жұмыстары қалай реттеледі?
7. MPP жүйесін құруда қолданылатын ең тиімді топология түрін атаңыз?
8. Транспьютердің стандартты бір кристалды есептеу машинасынан айырмашылығы неде?
9. Транспьютерлік есептеу жүйесінде көрші транспьютердің өзара әрекеттесуі қандай аппараттық және бағдарламалық қамтамалар арқылы іске асырылады?

Әдебиеттер тізімі

1. Авдеев В.А. Периферийные устройства: Интерфейсы, схемотехника, программирование-М.: ДМК Пресс, 2009.-848с.:ИЛ
2. Айтхожаева Е.Ж, Тынымбаев С.Т. Цифрлық автоматтардың қолданбалы теориясы: оқулық, Рауан, Алматы, 1990
3. Буза М.К. Архитектура компьютеров. Минск. Новое издание, 2006. 559 с
4. Гук М. Шины DCI, USB, Fire Wre, СПБ.: Питер, 2005. 540 с
5. Джурунтаев ДЖ. З. Сұлбатехника: оқулық. Алматы, 2011
6. Каган Б.М. Электронные вычислительные машины и системы. М.: Энергоатомиздат, 1991. 592с
7. Королев Л. Н. Архитектура ЭВМ. М.:Научный мир, 2005. 272с.
8. Крайзмер Л.П, Бородаев Д.А, Гутенмахер Л.Н, Кузьмин Б.Н, Смелянский И.П. Ассоциативные запоминающие устройства, Л.: Энергия, 1967
9. Лехин С.Н. Схематехника ЭВМ. СПб.: БХВ- Петербург, 2010-672с.:ИЛ
10. Майоров С.А., Новиков Г.И. Структура электронных вычслительных машин. Л.: Машиностроение, 1979, 384 с.
11. Максимов Н.В., Партыка Т.Л., Попов И.И. Архитектура ЭВМ и вычислительных систем: учебник. М.: Форум, 2008. 512 с.
12. Мелехин В. Ф., Павловский Е.Г. Вычислительные машины, системы и сети. М.: Издательский центр, «Академия», 2007.
13. Орлов С.А., Б.Я. Цилькер. Организация ЭВМ и систем – для бакалавров и магистров, СПб.:Прогресс книга ПИТЕР, 2018.-688с.:ИЛ
14. Таненбаум Э. Архитектура компьютера. 5-е изд. СПб.: ПИТЕР, 2019, 816с.
15. Тұрым А.Ш. Ақпараттану және есептеу техникасының түсіндірме сөздігі. – Ы. Алтынсарин атындағы Қазақтың білім академиясының Республикалық баспа кабинеті. – Алматы: 2000, 192б.
16. Тұрым А. Ш. Есептеу жүйелері мен желілері. – Алматы: ҚазҰТУ, 2006. -331б.
17. Тынымбаев С.Т. Вычислительные машины, системы, комплексы и сети. – Алматы: Рауан, 1997-366с: ИЛ.
18. Тынымбаев С.Т., Шайкулова А.А., Жандыбаева М.А. Көбейткішті – қосындылағыш блоктарын синтездеу және олардың негізінде ұлғаймалы көбейткіштер құру, Состояние, проблемы и задачи информатизации в Казахстане. Сб. трудов Третьей Международной конференции. г.Алматы, 20-22 ноября 2014- Алматы: КазНТУ; МАИН, 2014. -4.1-104-110 стр.
19. Тынымбаев С.Т., Мырзабакова К. Синтез умножителей с повышенный разрядностью. Труды международного форума «Инженерное

образование и наука в XXI веке»: проблемы и перспективы, посвященные 80-летию КазНТУ им. К.И. Сатпаева 20-24 октября 2014г.
20. Хамахер., Вранешич З., Заки С. Организация ЭВМ, 5-е издание. СПб.: ПИТЕР, 2003, 848с.
21. Хорошевский В.Г. Архитектура вычислительных систем. М.: Изд. МГТУ им. Н.Э. Баумана, 2005. 512с.
22. Цилькер Б.Я., Орлов С.А. Организация ЭВМ и систем СПб.: ПИТЕР, 2004. 668с.
23. Baugh, C.R., Wooley, B.A., A Two's Complement Parallel Array Multiplication Algorithm, IEEE Transactions on Computers, C-22, Dec.1973, pp 1045-1047.
24. Booth, A.D. Signed Binary Multiplication Technique Quart. J. mech. Appl. Math. Y. part 2, 1951, pp. 236÷240
25. Chen, T.C., HO, I.T., Storage- Efficient Representation of Decimal Data, CACM, 18(1), January 1975, pp 49-52
26. Cocke. J., Sweeney D.W., High Speed Arithmetic in a Parallel Device, Technical Report IBM. 1957.
27. Cowlishaw, M.F., Densely Packed Decimal Encoding, IEE Proceedings-Computer and Digital Techniques, 149(3), May 2002, pp 102-104
28. Flynn, M.J., Very High-Speed Computing System, Proceeding IEEE, N54, 1966. pp 1901-1909
29. Flynn, M.J., Some Computer Organization and their Effectiveness, IEEE Transactions on Computers, Vol.24, Sep.1972, pp.867-884
30. Kung S.Y. On Supercomputing with Systolic/Wavetront Array Processors, Proceeding IEEE, Vol.72, N47, 1984, pp. 867-884
31. Lehman, M., High-Speed Digital Multiplication, IRE Transaction on Electronic Computers, Vol.EC-6.6, N3, 1957
32. Lu, M., Arithmetic and Logic in Computer Systems, John Wiley & Sons, 2004.
33. Parhami, B., Computer Arithmetic: Algorithms and Hardware Design, Oxford University Press, 2000
34. Robertson, J.E., A New Class of Digital Division Methods, IEEE Transactions on Computers, Electronic Computers, EC – 7, Sep.1958, pp 218-222
35. Tocher, K.D., Techniques of Multiplication and Division for Automatic Binary Computers, Quart. J. mech. Appl. Math., 11, Jul./Sep.1958, pp 304-384
36. Stallings , W., Computer Organization and Architecture, 9th Education, Pearson/Prentice – Hall, 2013

37. Von Neumann, J., First Draft of a Report on the EDVAC, Moore School University of Pennsylvania, 1945.
38. Wilkes, M., The Best Way to Design on Automatic Calculating Machine, Proceedings Manchester University Computer Inaugural Conference, Jul.1951

Мазмұны