СӘТБАЕВ УНИВЕРСИТЕТІ

SATBAYEV UNIVERSITY

Векторлық процессорлар Олардың функционалдану ерекшеліктері мен үйымдастырылуы Cray суперкомьютерінің мысалы негізінде векторлы-конвейерлік процессордың архитектурасы Векторлық компьютерлерді қолдану кезіндегі тиімді аспап ретінде циклдарды векторизациялау түсінігі

Оқытушы: Жекамбаева М.Н.
«Программалық Инженерия»
кафедрасының профессоры
m.zhekambayeva@satbayev.university

Процессорлар

Қазірі замануи жоғарыөнімділікті есептеу жүйелері соның ішінде супер-ЭЕМ, өзінің құрамында бірнеше процессордан тұрады, жиі кездесетін келесі типтер:

- Скалярлы
- Векторлы

Бұл кез келген қиын тапсырмаларды тәжірибе жүзінде екі жиынтық ретінде көрсетуге болатындығына байланысты, олар: тізбекті және параллельді.

Тізбекті бөлік – біртипті операцияларды бір рет орындайтын жиынтық

Параллельді бөлік – параллельді орындалатын блоктар. Айқын параллелизм болуы мүмкін, егер бұл тәуелсіз программа тармақтары болса, немесе бір ағынның көптеген элементтерімен жүргізілетін операциялар кезіндегі деректер ағынын өңдеу параллелизмі (әдетте циклдар)

Кіріспе

Циклдарды жоғарыөнімділікті **скалярлы процессорларда** тарату жалпы максималды есептеу жылдамдығын шектейтін бірқатар факторламен байланысты болады:

- Әрбір скалярлы операция кезінде скалярлы команданы шақыру және декодтау керек болады.
- Әрбір команда үшін деректер адресін есептеу керек болады
- Деректер жадыдан шақырылып сол жадыға нәтижені ораластыру керек болады. Ол кезде әрбір команда өзінің жеке сұранысын тудырады да жадыға қатынау кезіндегі конфликтілер пайда болуына әкеледі
- Циклды құру командасын тарату (санағыш, ауысу) бақа да шығындарды тудырады

Сол себепті 1–3 факторарды болдырмау үшін: команда конвейерін қолданып әрі жеке команда және деректер буферлерін қолдану керектігі туындайды

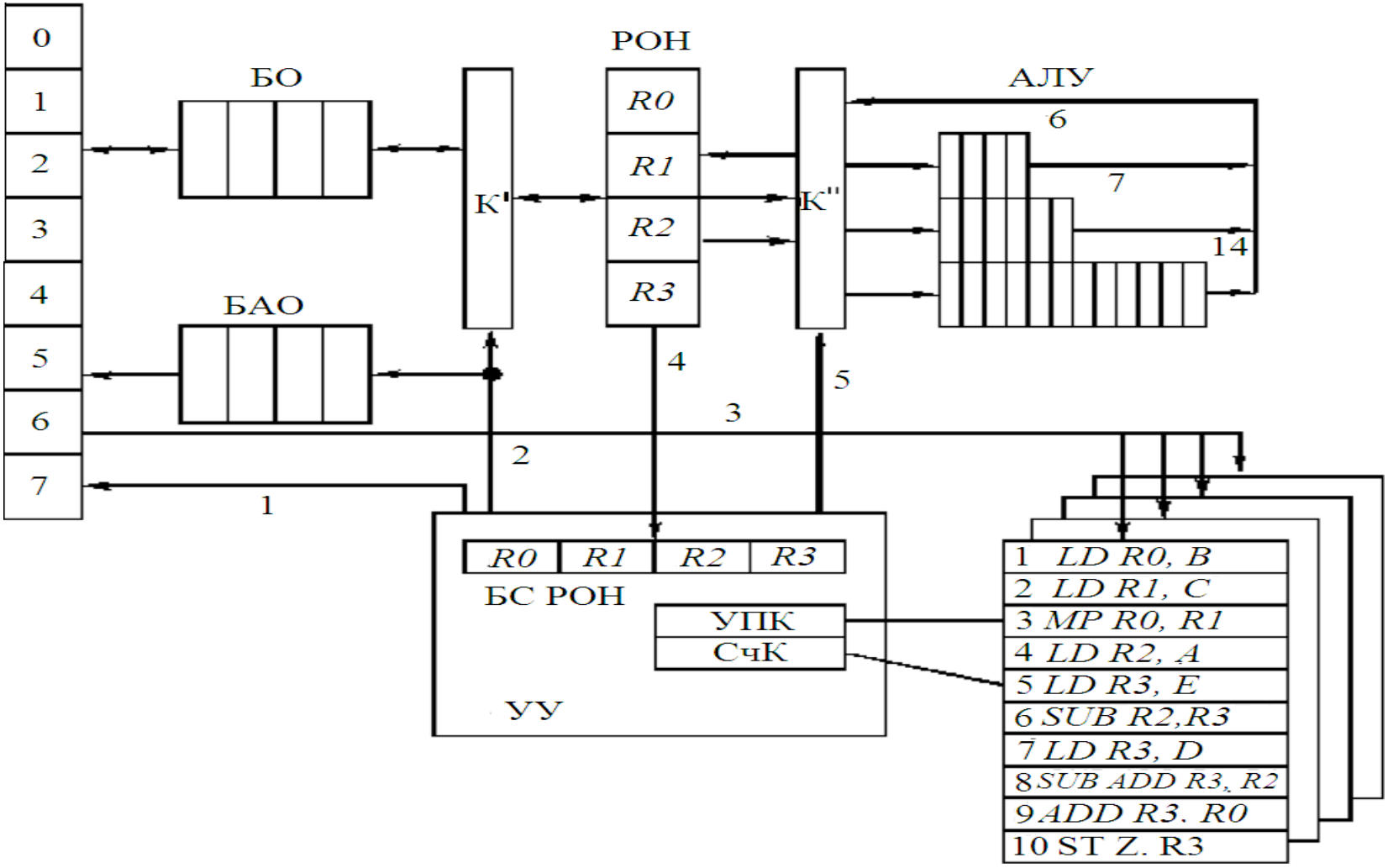
Ал барлық факторларды болдырмау үшін әрі ЕЖ өнімділігін арттыру мақсатында – векторлық процессорларды қолданады.

Ұйымдастырылу және функционалдану ерекшеліктері

Процессордың жылдам орындалуын арттырудың қазіргі кезде таратылған әрі қарапайым әдістерінің бірі ол есептеу процесін конвейеризациялау әдісі болып табылады. ***Конвейерлік ЭЕМнің параллельді ЭЕМ алдындағы үлкен артықшылығы ол тізбекті ЭЕМдерге арналған программалар пакетін қолдана алу мүмкіндігінің болуы болып табылады.***

Сол себепті *скалярлы конвейерлік процессордың* құрылымы мен функционалдануын толығымен қарастырайық

Скалярлы конвейерлі процессорды ұйымдастыру



Ұйымдастырылу және функционалдану ерекшеліктері

Мұнда процессор бірнеше конвейерлік АЛҚ тұрады. Бұл біруақытта аралық арифметикалы-логикалық операцияларды орындауға мүмкіндік береді, ол өз кезегінде тек параллельді қызметтік операцияларды ғана емес, сонымен қатар локальды параллелизмді тарауға мүмкіндік береді. Әртүрлі операциялар үшін АЛҚ әртүрлі конвейер ұзындығы қолданылады (суретте ұзындық 6, 7 және 14 позицияларға тең). Процессорда екі класс командадары қолданылады: жадыға қатынау командалары және ЖМР(РОН) регистрлік командалары. Команда буфері көпбетті құрылымды қолданады, ол БҚ (УУ) жұмыс кезінде алдын ала беттерді ауыстыруды жүргізуге мүмкіндіктер береді.

Скалярлы конвейерлі процессорды ұйымдастыру

ЦМП (ОКЖ) – Орталық көпблокты жады;
БАО (ОБА) – Операнд адрестерінің буфері;
АЛУ (АЛҚ) – конвейерленген АЛҚ, қосу, көбейту және жылжымады сандарды бөлу үшін қолданылады;
К', К'' – Жады және сәйкесінше АЛҚ коммутаторлары;
БС РОН (ЖМР КБ)– ЖМР күйлер блогы;
УПК (ЖКН) – жіберіліп қойған команда көрсеткішінің номері;
СчК (КСн)– команда санағышы;
1 – команда адресінің шинасы;
2 – жадыға қатынаған командаңы орындауды басқару шинасы;
3 – КБ толтыру шинасы;
4 – ЖМР күйін ауыстыру шинасы;
5 – регистрлік командаларды орындауды басқару шинасы
Суреттегі кез келген операция сәйкесінше операндтар дайын болған кезде ғана жіберіледі. ЖМР бос екендігі анықталмайынша шектеу қойылады. ЖМР күйі (БС) РОН анықталады.

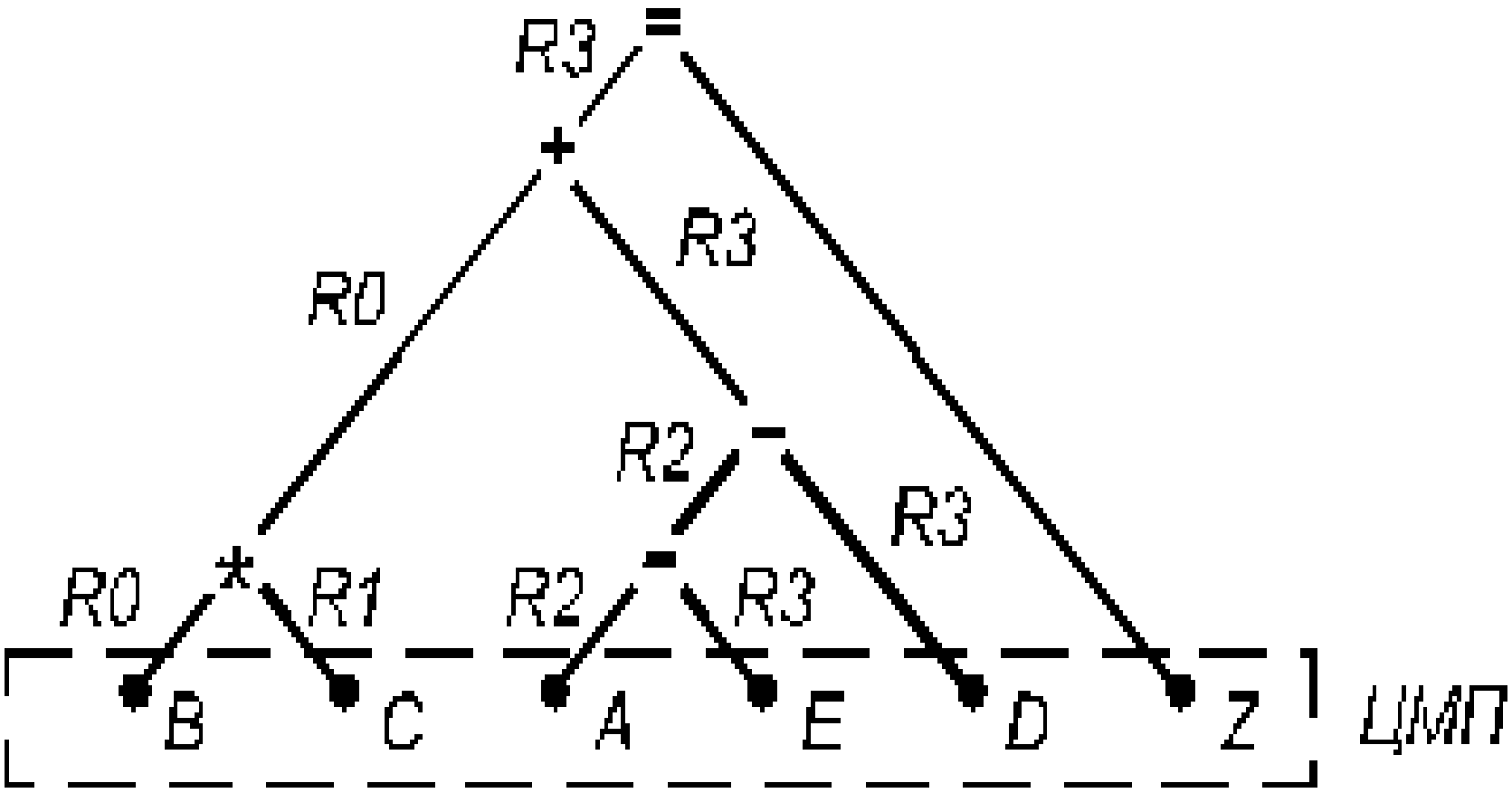
Скалярлы конвейерлі процессорды ұйымдастыру мысалы

Келесі өрнекті шешу керек:

*Z = A - (B * C + D) - E.*

Программады: *LD* – регистрден жадыға операндты жүктеу командасы; *MP*, *SUB*, *ADD* – көбейту, азайту және қосу командалары; *ST* – регистрден жадыға операндты жазу командасы.

Скалярлы конвейерлі процессорды ұйымдастыру мысалы орындалу реті



Скалярлы конвейерлі процессорды ұйымдастыру мысалы бойынша кесте

Мұндағы кейбір регистрлердің программаны орындау кезіндегі күйлері келесі кестеде келтірілген. Соңғы баған команданы орындауға жіберу мен ретін көрсетеді. Көріп отырғандарыңыздай кейбір командалар команданы жіберуден бұрын орындалуы мүмкін. Мысалы, 4 және 5 командалар 3 командадан бұрын орындалып тұр. Бұл программада локальды параллелизмнің және процессор құрылымында бірнеше АЛҚ болуна байланысты жүзеге асырылып отыр.

Бірақ мұндай "обгондар" ақпараттық графпен берілетін программа логикасының орындалуын бұзбай шартты орындауы керек.

Скалярлы конвейерлі процессорды ұйымдастыру мысалы бойынша кесте

Такт NN	ЖМР күйі				Команда нөмірі
	R0	R1	R2	R3	
1	4	–	–	–	1
2	3	4	–	–	2
3	2	3	4	–	4
4	1	2	3	4	5
5	x	1	2	3	–
6	7	–	1	2	3
7	6	–	x	1	–
8	5	–	6	x	6
9	4	–	5	4	7
10	3	–	4	3	–

Скалярлы конвейерлі процессорды ұйымдастыру мысалы бойынша кестеге түсініктеме

Такт 1. БС РОН талдау, баолық РОН бос екендігін көрсетіп тұр, сол себепті 1 команда ЦМП орындауға жібереді. *R*0 бағанға 4 жазылады, ол: *R*0 төрт такт бос болмайтындығы. Әрбір орындау сайын ол кішірейіп отырады. Процессор құрылымында *R*0 бос болмауы БС РОН *R*0 1 орнату арқылы белгіленеді, одан кейін *R*0 0ге 4 шина сигналы арқылы орнатылады, ол 4 тактіде жадыдан операндты алған кезде жүргізіледі.

Такт 2. 2 команда жіберіледі және *R*1 регистр блоктанады.

Такт 3. 3 команда қарастырылады, ол орындалмайды, себебі оны орындау үшін БС РОН оны орындауға *R*0 және *R*1 регистрлар керек ал олар блокта. Сол себепті 3 команда бос жіберіледі, ал оның номері УПК жазылады. Талдау жасалып келесі (СчК күйі бойынша) команда жіберіледі. 4 команданы орындауға болады.

Такт 4. Тағы 3 команда бойынша УПК тексеру жүргізіледі, әлі жүргізуге болмайды, сол себепті 5 команда жіберіледі.

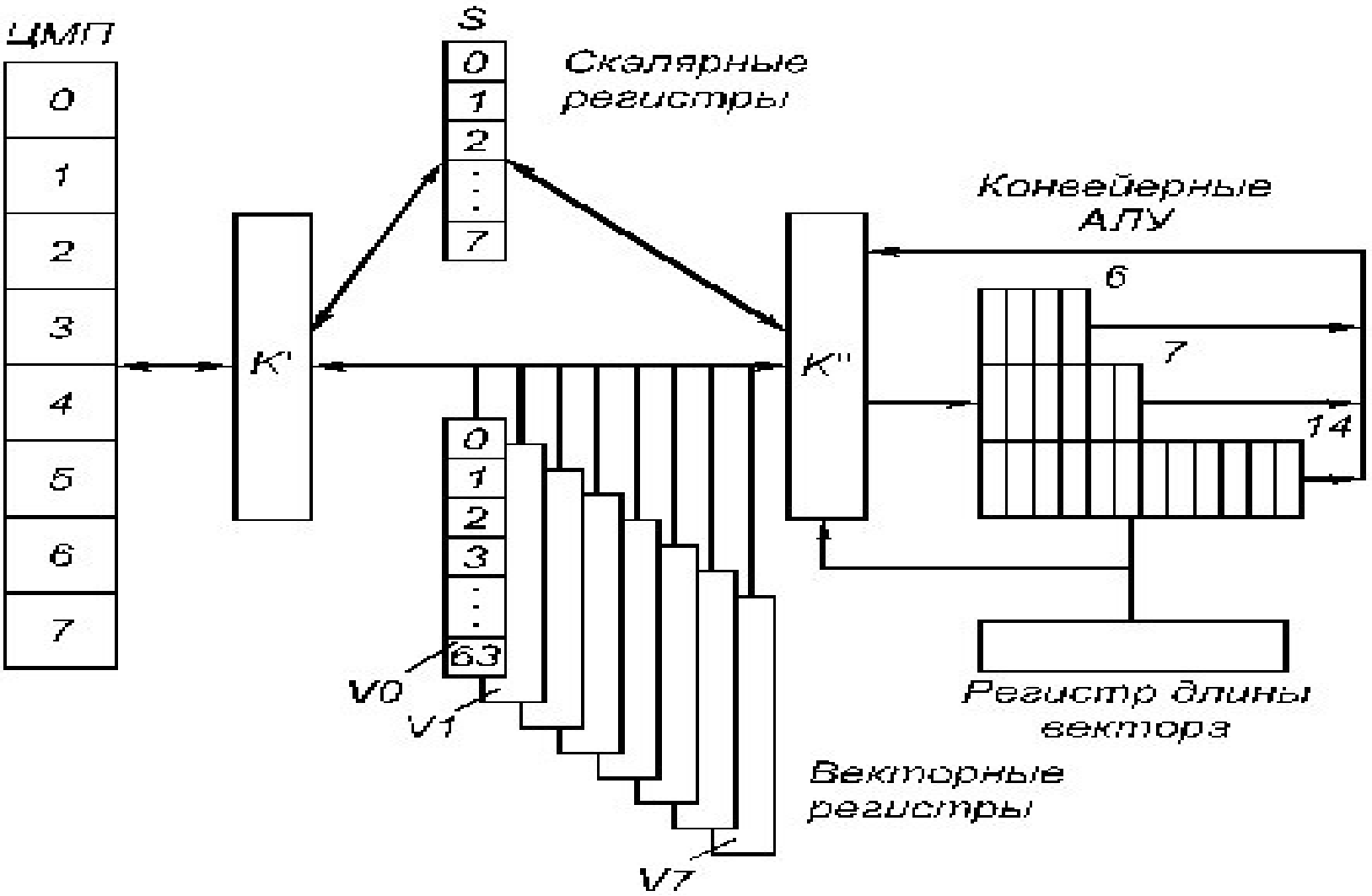
Скалярлы конвейерлі процессорды ұйымдастыру мысалы бойынша кестеге түсініктеме

Такт 5. 3 команда жіберілмейдіне, себебі *R*1 регистр бос емес, бірақ *R*0 регистр босатылды және 3 командамен қолданылуы мүмкін, бірақ ол қайта блокталды (**x** символы). Одан кейінгі келесі төрт команданың орындауы мүмкін емес, солс себепті 5 тактіде жаңа команда таңдалады.

Такт 6. 3 команда жіберіледі.

Одан кейін бұл үрдіс қайталанады. Көріп отырғандарыңыздай кестеде көрсетілгендей 10 такт ішінде, процессорда жеті команда орындалды, ол сәйкесінше 10/7 = командаға 1,5 такт бөлінгендігін көрсетеді. Айталық, процессор такті 10 нс тең болсын. Онда бір команданы орындау үшін 15 нс бөлінеді, ол сәйкесінше *V* = 70 млн оп/с тең болады.

Векторлы конвейерлік процессор құрылымы



Векторлы конвейерлік процессор құрылымы

Векторлық процессордың басқару құрылғысының құрылымы скалярлы конвейерлік процессордың БҚ ұқсас болып келеді. Басты ерекшелігі тек векторлық командаларды орындау кезінде командаңың операциясының коды өзгермейді.

Тағы бір векторлық процессордың ерекшелігі – бірқатар *векторлық регистрлердің V* (әдетте 8ге дейін) бар болуы, олардың әрқайсысы ұзындығы 64 сөзге тек векторды сақтай алады. Оны ЖМР деп қарауға болады, бірақ ол векторлық процессордың жұмысын біршама жылдамдатады. Ал қалған түйендердің тағайындалуы алдынғы суретте келтірілген скалярлы конвейерлік процессордағыдай болып келеді.

Векторлы конвейерлік процессор құрылымы

Векторлық процессорда (*) орындау мысалын қарастырайық. Шартты ассемблер тілінде программа келесі түрде беріледі:

LD L, Vi, A
LD L, Vj, B
MP Vk, Vi, Vj (**)
SUM Sn, Vk

1 және 2 операнторлар жадыдан сөздерді бастапқы A және B адрестерімен *Vi*, *Vj* регистрлеріне жүктеуді орындайды; 3 оператор векторларды элемент бойынша көбейтуді орындай отырып нәтижесін *Vk* регистріне орналастырады; 4 оператор – *Vk* векторларды қосып, нәтижесін *Sn* орналастырады.

Осы программаға сәйкес, векторлық регистрлер бастапқыда такт бойынша ЦМП дан толтырылады, одан кейін векторлық регистрлердегі сөздер такт бойынша (бір такт екі сөз) конвейерлік АЛҚ беріледі, онда бір такт бір нәтижеге тең.

Векторлы конвейерлік процессор құрылымы

Векторлық процессордың жылдамдығының сипатамаларын MP Vi, Vj, Vk командасының орындауы негізінде қарастыратын болсақ.

Команданы орындау үшін керекті болатын тактілердің саны:

r = m* + L, мұндағы m* – көбейту конвейерінің ұзындығы.

Себебі екі оперантды көбейту үшін k = (m* + L)/L такті жұмсалады онда мұндай процессордың жылдамдығы

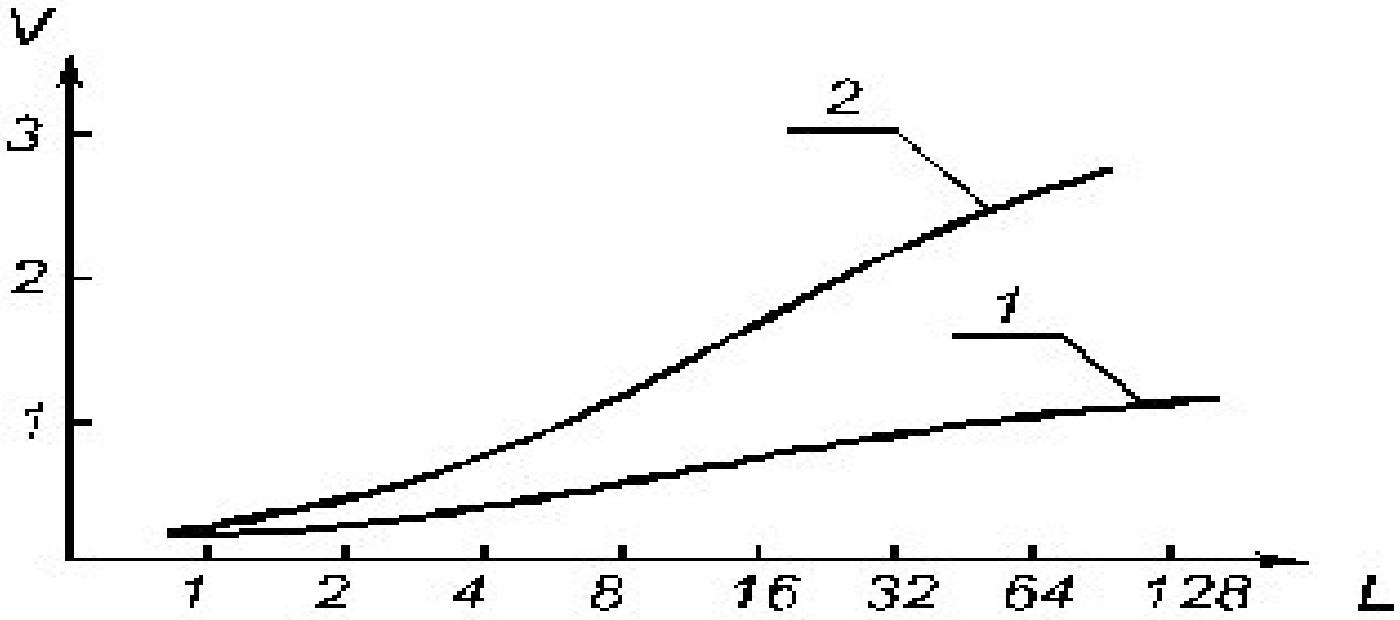
$$V = 1/(K\Delta t) = \frac{L}{(m_* + L)\Delta t},$$

мұндағы Δt – конвейердің бір тактісінің жұмыс уақыты.

Конвейердің жылдамдығы L такт өлшеміне тәуелді болады (келесі суретті қараңыз, 1 қисық)

Векторлы конвейерлік процессор құрылымы

L>m* кезінде K ~ 1 және V ~ 1/Δt өлшемдерін, әдетте векторлық процессорлар үшін Δt аз қылуға тырысады, мысалы 10...20 нс, себебі векторлық операцияларды орындау кезіндегі жылдамдық 50...10*106 оп/с жетуі мүмкін.



Векторлық процессорлар

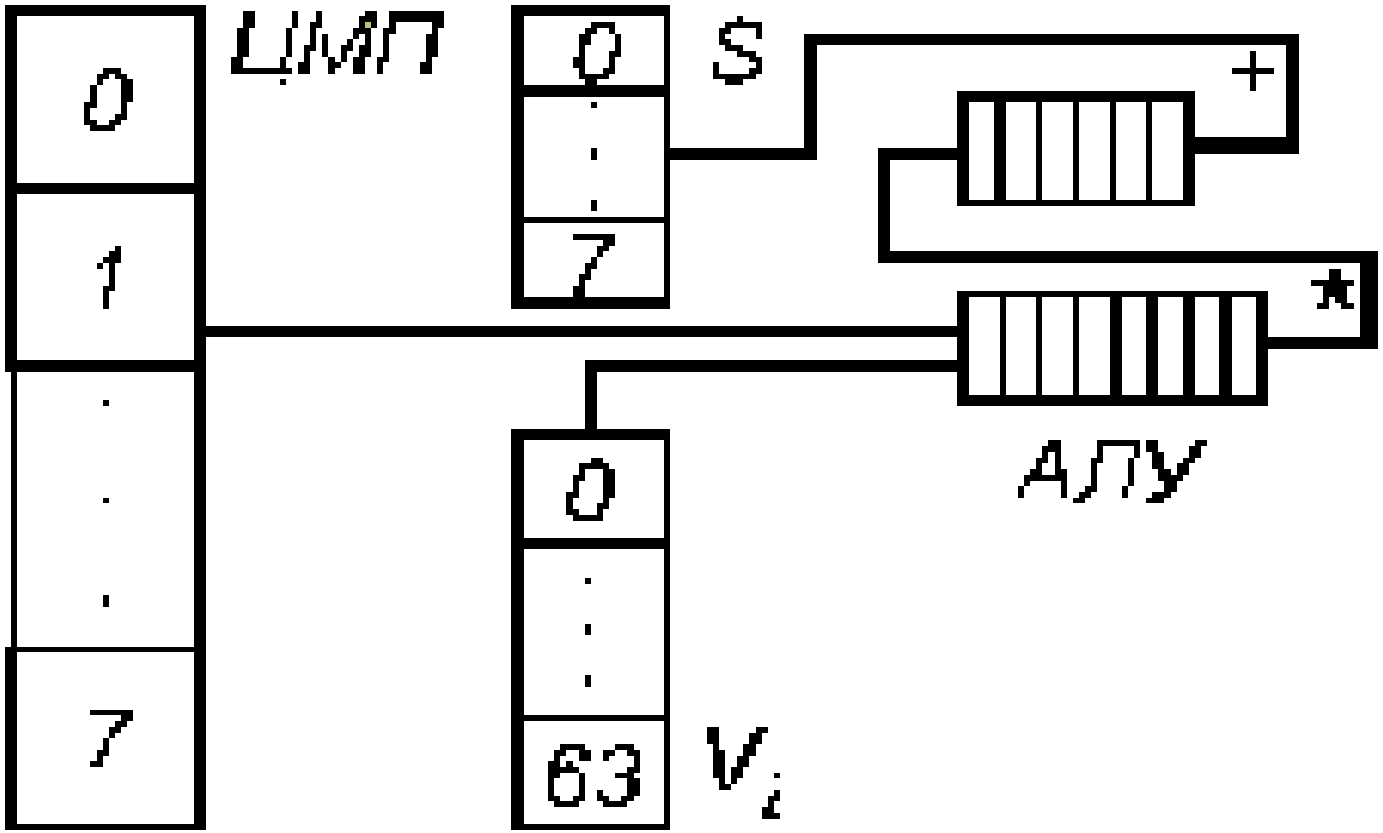
Векторлық конвейерлік процессорлардағы есептеу жылдамдығын арттырудың маңызды ерекшелігі, ол *зацепления* механизмі қолдану болып табылады. Зацепление – дегеніміз біруақытта векторлармен бірнеше операция түрін орындау алу әдісі. Соның ішінде, программада біруақытта ЦМП векторды таңдау, векторларды көбейту, векторлардың элементтерін қосу. Сол себепті программаны келесідегідей өзгертуге болады:

LD L, Vi, A

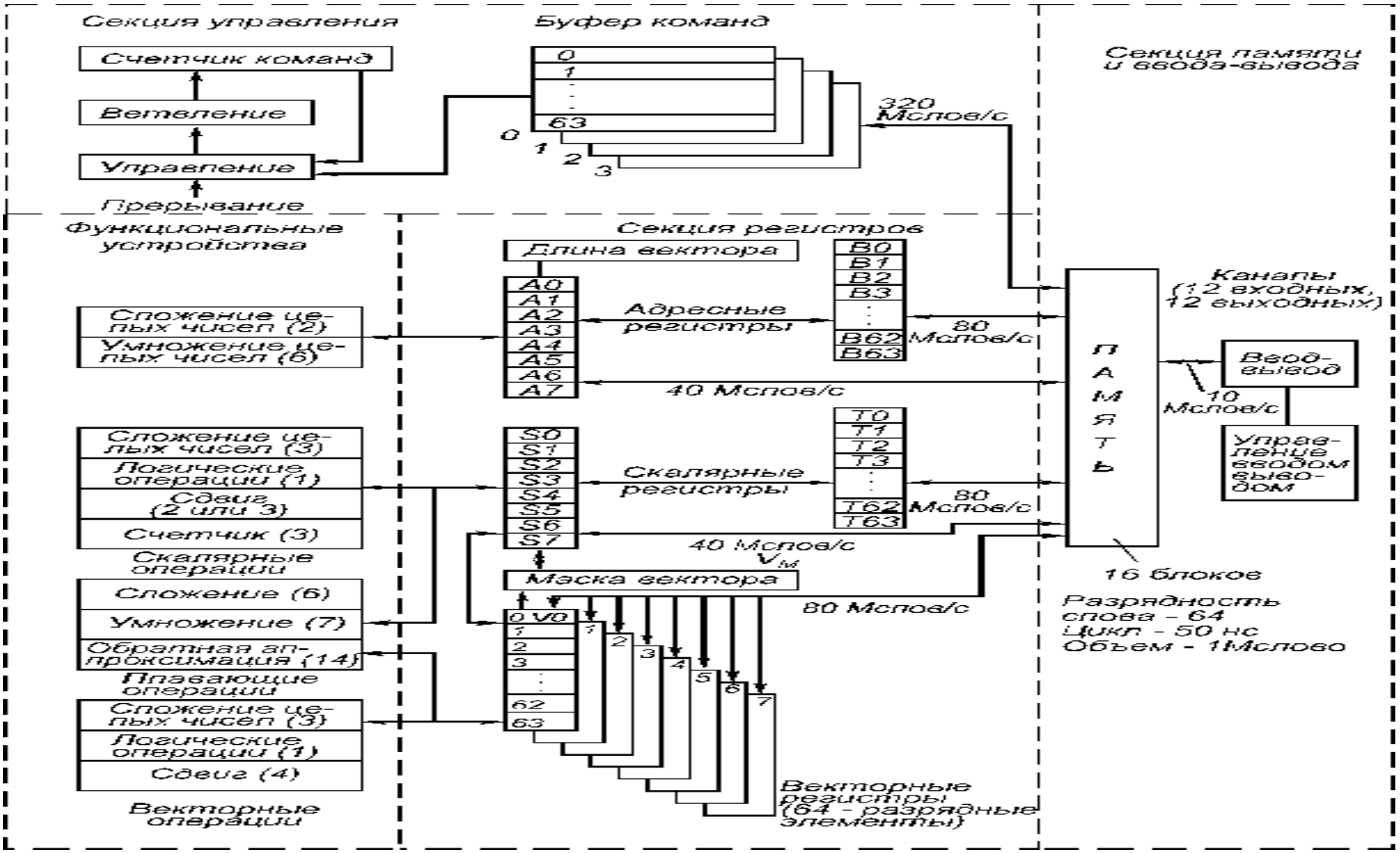
ЗЦ Sn, Vi, B

Мұнда зацепления (ЗЦ) командасы біруақытта келесі сұлбадағыдай байланыс арқылы операцияларды орындай алады

Тіркесу операциясын векторлы конвейерлік процессорда орындау



Cray суперкомпьютерінің архитектурасының мысалындағы векторлы-конвейерлік процессор



Cray суперкомпьютерінің архитектурасы

- Процессордың 8 векторлық регистрі бар, олардың әрқайсысы 64 биттен 64 сөзді сақтай алады.
- Сондай ақ 8 скалярлы регистрлері бар олар 64-биттік сөз және
- 20-биттік сөзге арналған 8 адресация регистрінен тұрады.
- Кэш-жадының орнына арнайы регистрлерді қолданады, оларды басқару программа арқылы орындалатын программадан жүргізіледі.

Барлығы Cray-1 компьютері 12 дейінгі конвейеризацияланған процессорлардан тұрады, олардың әрқайсысында әртүрлі арифметикалық және логикалық операцияларды орындау конвейерлері бар. Прцессордың жұмыс жасау жылдамдығы қатал жедел жады жұмысының жыладмдығымен байланысты.

Cray суперкомпьютерінің архитектурасы

CRAY-1, 1976 ж., бірінші ЭЕМ бірі ретінде пайда болды, онда *векторлы-конвейерлік машина* сипаттамалары көрсетілде. CRAY-1 кез келген ЭЕМ стандартты секциялардан тұрады: жадыны басқару және енгізу-шығару, регистлік секция және фукнциональды құрылғылар тобы. Бірақ әрбір секция құралдары қарапайым тізбекті ЭЕМ құрылымынан ерешеленеді.

Жадысы командалар және векторлық деректер (конвейерлік режим) таңдауы үшін, әрі скалярлы деректер (ағынды режим) үшін жүргізіледі. Ағынды режим үшін адрестің буферлік регистрлерін (*B*0...*B*63) және деректер үшін (*T*0...*T*63) қолданады. Жадының жіберу қабілеттілігі өте жоғары, бірақ әртүрлі түйіндер үшін барлық қабілеттілік жұмсала бермейді. Максимальды жылдамдылық тек команда буферін толтыру үшін керек.

БҚ құрылымы: алдын ала командаларды қарау механизмі бар, сондай ақ регистрлерді және функциональды құрылғыларды резервтеу қарастырылған.

Cray суперкомпьютерінің архитектурасы

Адрестік операцияларды орындау үшін *A*0...*A*7 адрестік регистрлер тобы қолданылады және арнайы АЛҚ бүтін санды арифмеикаларды орындау үшін қолданылады.

Бұл АЛҚ, барлық басқа функциональды құрылғылар сияқты конвейерлік құрылымды қолданады.

Скалярлы операциялар *S*0...*S*7 регистрлерінің көмегімен жүзеге асырылады, ал векторлы – *V*0...*V*7 векторлық регистрлерімен орындалады.

Беріліп отырған жүйеде бірінші рет зацепление операциясы қолданылған. Сол себепті екі, кейде тіптім үш функциональды құрылғылар жұмысқа қосыла алады, ол жүйенің өнімділігін $160*10^6$ және кейде $240*10^6$ дейін жеткізе алады, нәтижелер секундына орындалатын жылжымалы нүктелі сандармен жұмыс кезінде қамтиды.

Векторлық компьютерлерді қолдану кезіндегі тиімді аспап ретінде циклдарды векторизациялау түсінігі

Векторлық ЭЕМ жұмыс істеудің ең ыңғайлы әдісі ол ***массив ұзындығы векторлық регистрдің ұзындығына тең болғанда*** деп есептеледі. Бірақ бұл жағдай барлық уақытта орындала бермейді. Әдетте массивтермен жұмыс жасаған кезде элементтердің ұзындығына көңіл беле де бермейміз, сол себепті олардың ұызындығын 128 етіп алған дұрыс деп есептелетін әдіс болып табылады. Векторизациялауға болатын қарапайым циклды қарастырайық.

Мысалы 128 ұзындықтағы *m* массиві берілсін, оны келесі алгоритм бойынша толтыру керек:

```
do i=1, 128
m(i) = i
enddo
```

Векторлық компьютерлерді қолдану кезіндегі тиімді аспап ретінде циклдарды векторизациялау түсінігі

Векторлық регистрларды қолданып, келтірілген циклды ассемблер тілінде келідегідей жазуға болады:

SETLEN #128 ; қолданылатын элементтердің санын векторлық регистрлерге орналастырыңыз (барлық)
SETINC #4 ; жадыдағы массив элементтеріне ығысу орнатыңыз (4 байт)
SETNUM v0 ; векторлық регистр v0 элементтеріне олардың нөмірлерін 0 бастап 127 дейін жазыңыз
ADD #1, v0 ; әрбір v0 регистрнің элементіне 1 қосыңыз
SAVE v0, m ; v0 элементтеріндегі сөздерді ЖЖ тізбекті m адресінен бастап жазып шығыңыз (ығысу = 4)

Векторлық компьютерлерді қолдану кезіндегі тиімді аспап ретінде циклдарды векторизациялау түсінігі

Мұнда 3 командаға көңіл бөліңіз – векторлық регистрдің әрбір элементі өзінің нқмірін «біледі". Бұл цикл айнымалыларының орындалуын қарапайым етеді: 1 қосқаннан кейін айнымалының мәні вектордық сәйкесінше эементіне жазылады. Барлығы векторлық процессордың 5 тізбекті командасы 128 қайталаудан тұратын циклды орындай алады. Мұнда салыстыру да шартты өту де берілмеген.

Векторлық компьютерлерді қолдану кезіндегі тиімді аспап ретінде циклдарды векторизациялау түсінігі

Енді массив өлшемін және қайталаулар санын *N дейін* арттырайық:

```
do i=1, N
m(i) = i
enddo
```

Процессордың дұрыс жұмыс жасауы үшін вектордың қолданатын элменттерінің санын 128 артық етіп қолдануға болмайды. Егер N өзбетінше таңдалатын болса, онда беріліп тұрған бір ретті циклды екілік қолданылатын циклға айналдыру керек болады:

```
1  do inc=0, N-1, 128
2  NN = min0(128, N-inc)
3  do i=1, NN
      m(inc+i) = inc+i
      enddo
      enddo
```

Векторлық компьютерлерді қолдану кезіндегі тиімді аспап ретінде циклдарды векторизациялау түсінігі

1 таңбадағы цикл массив элеметтерін ұзындыңы 128 элементке тең әшкә массивтерге «бөлуді" орындайды. *inc* айнымалысы массивтің бірінші элементінен кезекті ішкімассивтердің: 0, 128, 256... ығысуын көрсетеді. *NN* айнымалысы ішкімассивтің ұзындығын анықтайды. Әдетте ол 128 тең, бірақ соңғы ішкі массив егер *N* 128 еселік болмаса ұызындығы басқаша болған жағдайды ескеред. *min*0 функциясы тек соңғы ішкімассивке арналған 2 таңбадағы 128 болмаған кездегі минимальды мәнді таңдауды орындайды. 3 таңбадағы ішкі цикл алдыңғы мысалдағы циклға ұқсас.

Тек 3 ерекшелігі бар:

- векторлық регистрдегі элементтер саны *NN* тең;
- циклдың параметріне қосымша *inc* мәнін қосу керек;
- вектордың элементтерін жадыға жазу массивтің басынан бастап емес, ал *i*+*inc* нөмірлі элементтен басталады.

Ассембелр тіліндегі программа

- SETLEN NN ; векиорлық регистрдегі элементтер саны = NN
- SETINC #4 ; келесі массив элементіне ығысуы
- SETNUM v0 ; векторлық регистр v0 элементтеріне олардың нөмірлерін 0 бастап 127 дейін жазыңыз
- ADD #1, v0 ; v0 регистрінің әрбір элементіне 1 қосу
- ADD inc, v0 ; inc айнымалысына мән қосу
- MOVE inc, r0 ; r0 скалярлы регистрге inc анымалсының мәнін жазу – біріншіден ығысу
- SETLEN NN ; m(inc+1) массивінің элементі
- MUL #4, r0 ; 4 көбейту- ығысу байтпен
- ADD #m, r0 ; m массивінің адресн қосу, яғни m(inc+1) элементінің адресін қосу
- SAVE v0, @r0 ; тізбекті ЖЖ v0 элементін жазу, олар r0 регистрінде сақталатын адрестен басталатын сөздерден (ығысу = 4 қоса) жазылады

Қорытынды

Қарапайым цикл үшін **бір векторлы және үш скалярлы** команда қосылды. Бірақ бұл тек ішкі цикл. Осы циклды қамтитын 1 таңбасындағы және *NN есептеу қосымша код құрады*, олар да көрсетілген ретте ішкі цикл қанша рет орындалса олар да сонша қайталанады. Транслятор барлық уақытта қамтылған циклдарды құрып отырады.

Айтылғандарды қорытындылай отырып, ***векторлық программалардың тиімділігі ұзындығы белгісіз бірнеше массивтармен жұмыс кезінде жоғары болмайды.***
Еселенген циклдар (мысалы, үш қайталауы бар) векторлық режимде, скалаярлы машиналарға қарағанда баяулау жұмыс жасауы мүмкін.