ЛИТЕРАТУРА

1. Конноли Томас, Бегг Каролин Базы данных. Проектирование, реализация и сопровождение. Теория и практика = Database systems. A Practical Approach to Design, Implementation, and Management; Пер.с англ.3-е изд. - М. : Вильямс, 2003.
2. Леффингуэлл Д. Принципы работы с требованиями к программному обеспечению, унифицированный подход / Д.Леффингуэлл, Д Уидрит.- Изд. Дом «Вильямс», 2002.
3. Маклаков С.В. Создание информационных систем с AllFusing Modeling Suite. – М.: Диалог-МИФИ, 2003.
4. Семенов М.И. Автоматизированные информационные технологии. – М.: Наука, 2000.

**УДК 681.142.2**

**Мукашева Гульбану Ануаровна – оқытушы (Алматы, ҚазККА)**
**Хабидолда Еркежан – оқытушы (Алматы, ҚазККА)**
**Нурбакова Гулия Серикмухаметовна – оқытушы (Алматы, ҚазККА)**

## АССЕМБЛЕР-ПРОГРАММАСЫ ҮШІН DEBUG ЖҮЙЕЛІК УТИЛИТТЕРДІ ҚОЛДАНУ

Қазіргі таңда программалау тілдерінің көптігі, олардың дамыған және қолайлы құралдарымен танымалдылығы, осы тілдер арқылы қолданбалы программаларды жасап шығарушы адамдарға компьютерлік жүйелердегі аппараттық ерекшеліктерді әрі қарай оқып үйренуге қажетсіз екені көрінеді. Операциялық жүйелердің резиденттік программасының жұмысын функцияналдауды оқу немесе жеке қолданыстағы драйверлерді өз қолымен жасау немесе басқа да жүйелік программаларды жасау және т.б. қиыншылықтармен кездескенде ғана программисттер осы қателіктерді пәндер ішінде «компьютерлік темірді» оқып үйренушілер үшін толықтыра бастайды.

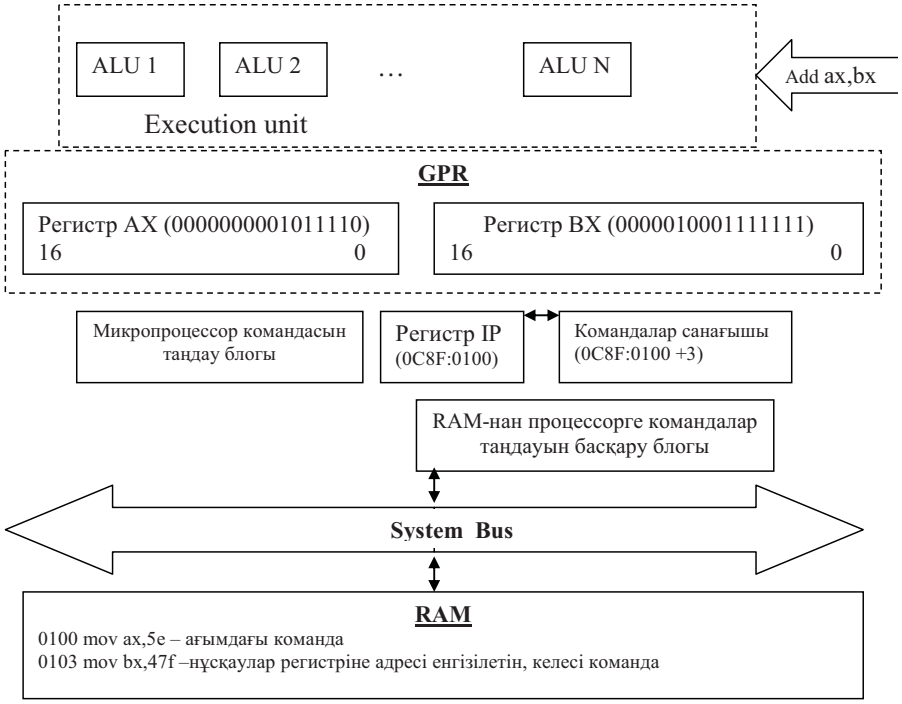
Микропроцессорлардың ұйымдастырылуы және құрылымы пәніндегі теориялық білімді бекітудің ең тиімді тәсілі, төмен деңгейлі тіл – ассемблер тілін оқып үйрену болып есептеледі. Әлемге танымал программист және ассемблер тілі туралы кітаптар авторы Владислав Пирогов айтқандай: ассемблер тілі не үшін керек? деп сұрағанда, берілетін ең қарапайым және сенімді жауап - Бұл процессор тілі және ол процессорлар бар болғанша керек болады. Таңқаларлық жағдайлардың бірі, студенттер үшін ақпараттың көптігі: бүкіл дүние жүзі торап ішінде, баспа құралдарында, электронды түрде таралатын материалдар сияқты ақпарат табу жолдары оңайлап бара жатқандықтан студенттің өз бетімен ақпарат іздеу технологиясы жоғалып бара жатыр. Қолдан программалау яғни кросс-ассемблерді қолданбай машина-бағытталған тілде программалау және интерпретаторларды қолданудың қиындығына қарамастан осы бағыттарда жұмыс істеу нәтижесінде қабылданған білімдер өте маңызды. «Есептеу техникасы және программалық қамтамасыз ету», «Ақпараттық жүйелер», «Автоматтандыру және басқару» және «Радиотехника, электроника және телекоммуникациялар» сияқты мамандықтардың студенттері ассемблер пәнін модуль көлемінде болсада оқуы қажет. ММУ-де (МГУ), мысалы, IT – мамандықтың студенттері Intel ассемблер процессорын семестр бойы оқиды.

Микропроцессор архитектурасын оқып білу және оперативті жады үшін, ассемблер тілінің қолдану тәсілін және орындалатын тапсырмалардың өзара әсерін

қарастырайық. Бірнеше жылдардың ішінде мақала авторлары debug.com утилиттарының көмегімен ассемблер оқыту тәсілін сынақтан өткізді. Мысалы, иллюстрация үшін резидентті тапсырманың жұмыс уақыты оперативті жады мен микропроцессордың өзара әрекетін келесі мысалмен қарастырайық.

```
-a
0C8F:0100 mov ax,5e
0C8F:0103 mov bx,47f
0C8F:0106 add ax,bx
0C8F:0108

-r
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0C8F  ES=0C8F  SS=0C8F  CS=0C8F  IP=0100   NV UP EI PL NZ NA PO NC
0C8F:0100 B85E00       MOV     AX,005E

-t
AX=005E  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0C8F  ES=0C8F  SS=0C8F  CS=0C8F  IP=0103   NV UP EI PL NZ NA PO NC
0C8F:0103 BB7F04       MOV     BX,047F

-t
AX=005E  BX=047F  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0C8F  ES=0C8F  SS=0C8F  CS=0C8F  IP=0106   NV UP EI PL NZ NA PO NC
0C8F:0106 01D8         ADD     AX,BX

-t
AX=04DD  BX=047F  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0C8F  ES=0C8F  SS=0C8F  CS=0C8F  IP=0108   NV UP EI PL NZ AC PE NC
0C8F:0108 C3           RET
```

Бұл мысалда студенттер біріншіден, қолдан командаларды жады ұяшықтарына «орналастырып» жатыр, бұл жерде бірінші команда ОЗУ-ды үш ұяшығын алып жатыр, ал үшінші команда – екіге ие. Екіншіден трассировка нәтижесі бойынша студент процессордың жалпы мағынасының регистрлерінің құрамы өзгеріп жатқанын көреді. Үшіншіден процессорға командаларды «беріп тұрған» санағыштың жұмысы көрініп тұр. Бұл программаның дәрісі шешілуі үшін компьютерлік жүйе архитектурасы көмегімен функционалды ядро түйіндерінің бір-бірімен әсерлесу сұлбасы құрылады.




1-сурет. Функционалды ядро түйіндерінің бір-бірімен әсерлесу сұлбасы

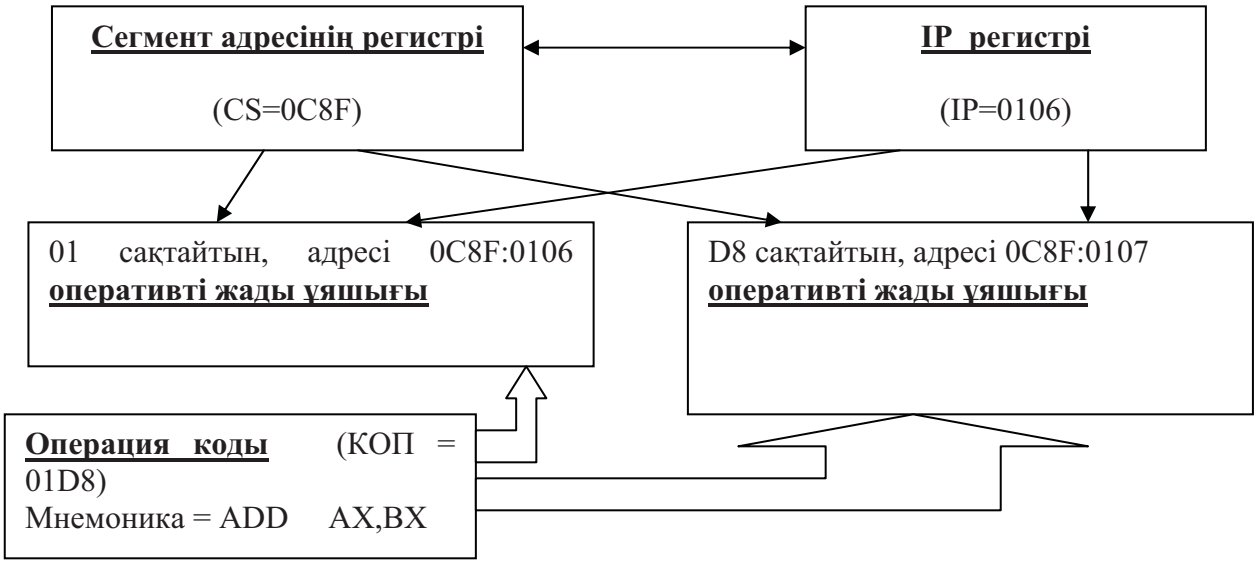
Бұл мысалда программаның орындалуына процессордың қарым-қатынасы мен жадыны жинақтау қажет. Олардың бірі алдыңғы команданың адресін анықтауда сегмент регистрі ескерілмейді.

debug тренингі жоғары деңгейдегі программалау тілінен айырмашылығы, ол қолданбалы программаны жасауға емес, процессор командасын өңдеуге бейімделген. Ассемблер тілінде программалау қажетті мағлұматтарды береді, ол компьютерлік жүйелердің түйіндері, ақпараттық және аппараттық қарым-қатынасты қамтамасыз етуде қажет. Программалау бойынша қарағанда, шығыс программамен есте сақтау жадысының ұяшықтары арасында семантикалық үзіліс болмайды.

Келтірілген мысалда тағы бір керекті жайт, методологиялық көзқараста. Оны кеңірек қарастырсақ. Белгілі болғандай debug трассировкасы кезінде процессор регистрінің мазмұнын және есте сақтау жадының алдыңғы ұяшығын көрсетеді.

```
AX=005E  BX=047F  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0C8F  ES=0C8F  SS=0C8F  CS=0C8F  IP=0106   NV UP EI PL NZ NA PO NC
0C8F:0106 01D8          ADD     AX,BX
```

Процессордың программаны өңдеудегі қарым қатынастарының жадысы бірбайтты ұяшықтан тұрады, оларға қарау өз номерлерімен беріледі (физикалық адресі). Берілгендерді есте сақтау жадысында байттармен оқуға немесе жазуға болады. Е debug командасын қолдана отырып студенттерге көрсетіп ғана қоймай, ұяшық жадысының адресациясын анықтауда тренинг ұйымдастыруға болады. Жады сегментациясы ұяшықтарының функционалдық адресі екі регистрде сақталады деп болжайды: сегмент адресінің регистрінде және сығу регистрінде. debug регистрінің соңғы жолындағы ақпаратты иллюстрация жасайды. Бұл мысалда жадыға операция кодын енгізу және команда адресін құрастыру сұлбасын иллюстрациялау түрінде жалпылауға болады.




Сурет 2. Команда адресін құрастыру сұлбасын.

Ұйымдастыру үшін есептеу микропроцессоры өзінің құрамына жалпы белгілердің өз тіркеуін, процессор күйінің аталуына және бағдарлама барысының өзгеруіне байланысты тіркеу флагтарының (FLAGS) деректерін уақытша сақтауға қолданады (орындалған команданың жалғастыру барысы). Флагтарды тіркеу жұмыстарының демонстрация әдісі «толықтырылған» тіркеудің анықталған командаларының трассировкасымен қорытындыланады. Екі тіркеудің бүтін біреуіне бөлу командаларына мысал қарастырайық:

```
-a
0C8F:0100 mov ax,ffff
0C8F:0103 mov bx,1111
0C8F:0106 add ax,bx
0C8F:0108
-r
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0C8F  ES=0C8F  SS=0C8F  CS=0C8F  IP=0100   NV UP EI PL NZ NA PO NC
0C8F:0100 B8FFFF       MOV     AX,FFFF
-t
AX=FFFF  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0C8F  ES=0C8F  SS=0C8F  CS=0C8F  IP=0103   NV UP EI PL NZ NA PO NC
0C8F:0103 BB1111       MOV     BX,1111
-t
AX=FFFF  BX=1111  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0C8F  ES=0C8F  SS=0C8F  CS=0C8F  IP=0106   NV UP EI PL NZ NA PO
0C8F:0106 01D8         ADD     AX,BX
-t
AX=1110  BX=1111  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0C8F  ES=0C8F  SS=0C8F  CS=0C8F  IP=0108   NV UP EI PL NZ AC PO CY
0C8F:0108 C3                RET
```

Келтірілген мысалдан трассировка нәтижесін сыртқы толысқан флаг жұмысынан (NA→AC) және ауыстырудан (NC→ CY) көріп тұрмыз.

**Қорытынды**

Микропроцессор архитектурасын оқып білу және оперативті жады үшін, ассемблер тілінің қолдану тәсілі және орындалатын тапсырмалардың өзара әсері қарастырылған. Мақалада debug.com утилиттасының көмегімен ассемблер тілін оқыту тәсілі аппробациядан өткізілген.

ӘДЕБИЕТ

1. http://asm.shadrinsk.net/ Сайт Владислава Пирогова. Ассемблер и не только.
2. Методические указания к лабораторным занятиям по дисциплине «Архитектура компьютерных систем» - Мукашева Г.А., АО «Казахская академия транспорта и коммуникаций имени М.Тынышпаева», 2009.
3. Методические указания к лабораторным занятиям по дисциплине «Цифровые устройства и микропроцессоры» - Мукашева Г.А., АО «Казахская академия транспорта и коммуникаций имени М.Тынышпаева», 2009.

**УДК.656.25(075)**

**Касимов Абдуразак Оразгельдиевич - преподаватель (Алматы, КазАТК)**
**Хамитова Дина Сейдуллаевна - преподаватель (Алматы, КазАТК)**

## СОПРЯЖЕНИЯ МИКРОПРОЦЕССОРНЫХ СИСТЕМ С ИСПОЛНИТЕЛЬНЫМИ ЭЛЕМЕНТАМИ

Наметившаяся в настоящее время тенденция в развитии средств автоматизации на железнодорожном транспорте предполагает широкое внедрение микропроцессорных систем. Микропроцессорные средства, предполагающие решение целого ряда задач на уровне программного обеспечения, позволяют существенно расширить функциональные возможности систем автоматики. Значительное уменьшение количества электромагнитных реле, применение перспективной элементной базы позволяют существенно снизить затраты на строительство, повысить надежность функционирования новых систем. Оборудование, не требующее периодического обслуживания, а также развитые возможности самодиагностики обеспечивают значительное снижение затрат на обслуживания новых систем автоматики, построенных на основе микропроцессорной техники. Одной из важнейших проблем при построении микропроцессорных систем управления является организация сопряжения управляющего вычислительного комплекса (УВК) с исполнительными объектами. Поиск технических решений, связанных с разработкой устройств сопряжения должен осуществляться с учетом современных достижений в области цифровой и аналоговой схемотехники. Для ввода-вывода информации из УВК обычно используются стандартные интерфейсы. Как правило, электрические параметры этих интерфейсов не могут обеспечить возможность непосредственного воздействия УВК на исполнительные объекты. Поэтому, под сопряжением, прежде всего, подразумевается энергетическое согласование электронных схем, осуществляющих вывод сигналов управления из УВК с исполнительными объектами, используемыми в конкретной системе управления.

Кроме того, существует задача временного согласования, предусматривающая приведение временных параметров сигналов к форме, необходимой для взаимодействия с тем или иным функциональным узлом. В частности, для вывода информации из ЭВМ все чаще используются интерфейсы, подразумевающие последовательную передачу данных (например, широко распространенный интерфейс RS-485). В этом случае возникает