Дәріс №11.
Тақырыбы: Жадыны бірігіп қолдану. Жадыны қорғау.

Көптеген стратегиялық шешімдер операциялық жүйе деңгейінде сияқты аппараттық деңгейде қайталанады. Мультипрограммалық режимде негізгі шарттардың бірі сақтауды қамтамасыз ету болып табылады. Бір рет қолданылатын операциялық жүйе сияқты файлдық жүйені қарастырайық. Бұл жағдайда мәліметтерді сақтау мәселесі болмайды, себебі осы операциямен жұмыс істеуші адам барлық файлдар иесі болып табылады. Бір рет қолдану жүйесінде мысалы, MS-DOS немесе Windows 95. Машинаны жүктеп басқа пайдаланушының барлық файлдарын жоюға болады. Көп қолданушы жүйесі көптеген пайдаланушылар жұмысын қамтамасыз етеді. MS-DOS мультипрограммалау тәртібінде жұмыс істей алады, бірақ жеткіліксіз, себебі бір үрдістегі қате көрші үрдістің және операциялық жүйенің өшіп қалуына алып келеді. Сондай-ақ Windows 95 ОЖ-де көптеген пайдаланушылар жұмыс істей алады, бірақ бұл жұмыс тиімді емес, себебі бұл ОЖ барлық сақтау құқықтарына ие емес. Сонымен көп қолданылатын жүйе санкцияланбаудан ақпаратты сақтауды қамтамасыз етеді. Негізінде, сақтау мәселесі файлдық жүйемен ғана байланысты емес. Операциялық жүйе барлық жерде мәліметтерді сақтау қабілетіне ие: бұл файлдар, үрдіс және қорлар.

Мұнда назарға осы фактке қарапайым, себебі файлдар үшін едәүір критикалық нүкте. Жедел жадыда орналасқан мәліметтер әдетте тегергіш жинақтауышы ролін атқаратын үшінші жадыда немесе екінші сақтау жабдықтарында орналасады. Үшінші жадыға жету уақыты ОЖ-ге жету уақытынан бірнеше рет жоғары және ОЖ-ң белсенді әрекет етуін талап етеді.

Unix операциялық жүйесінің жадыны басқару жүйесі үрдістер арасында жедел жадыдағы қорлардың тиімді тарауына жауап береді. Операция бөлімі операциялық жүйе басқаруымен үрдісті сақтаудың апараттық басқаруымен жүргізіледі.

Жады қорының менеджері. Жадыны бөлу стратегиялары. Қолданбалық интерфейстер және қабықшалар.

Компьютердің есте сақтау құрылғысы ең аз дегенде екі деңгейге бөлінеді: негізгі жады (бас, оперативті, физикалық) және ішкі жады.

Негізгі жады – бірбайтты реттелген ұяшық, ол ұяшықтың әрқайсысының нөмері (адресі) бар. Процессор негізгі жадыдан команданы алады да оны кодтайды, содан кейін оны орындайды. Командалар орындалу үшін тағы да негізгі жадыдағы бірнеше ұяшықтарға сұраныс жасау керек болады. Негізгі жады жартыөткізгіш технологиясын қолдану арқылы дайындалады және өзінің құрамындағыны токтан ажыратқан кезде жоғалтады.

Ішкі жадыны – (бұл дискінің негізі) көптеген байттардан құралған кеңістіктің бірөлшемді сызықтық адресіретінде қарастыруға болады. Оның оперативті жадыдан айырмашылығы ол токқа тәуелді емес, үлкен сиымдылығы бар және негізгі жадыны кеңейту кезінде қолданады.

Көпдеңгейлі схеманы былайша қолданады. Жадының жоғарғы деңгейінде орналасқан ақпарат үлкен нөмерлі деңгейде сақталады. Егер процессор керек ақпаратты і-ші деңгейден таба алмаса, онда ол келесі деңгейден іздей бастайды. Керек ақпаратты тапқаннан кейін ол тезірек орындалатын деңгейге беріледі.

Мұндай ұйымдастыру әдісі кезінде жады дегейінде болғанша қатынау жылдамдығы азайтылады және де оларға қатынас жасау жиілігі де азайтылады.

Бұл жерде негізгі рөлді шектелген уақыт бөлігінде жады адресінің кішкене жиынымен жұмыс істейтін нақты программа қасиетімен сипатталады. Бұл жергілікті қасиет.

*Логикалық жады.*Ұяшықтың сызықтық жиыны түріндегі жадының аппаратты ұйымдастырылуы программистердің программа мен деректерді сақтауды

ұйымдастыруымен сәйкес келмейді. Көпшілік программалар кеңістіктегі сызықтық адрестерді түзе отырып, бір-біріне тәуелсіз модуль жиынымен сипатталады. Бірақ модульдер әр аймақтағы жадыға әсер етеді және әр түрлі қолданылады.

Бұл көзқарасты қолдайтын жадыны басқару схемасы *сегментация* деп аталады. Сегмент – ішінде сызықтық адресті қолдайтын арнайы тағайындалатын жады аймағы. Сегмент процедураға, массивке, стекке және скалярлы шамаға ие болады, бірақ аралас типтегі ақпарат жоқ.

Алғашында жады сегменті программа кодтары фрагменттері (мәтіндік редактор, тригонометриялық кітапхана) әр процесс өзінің кеңістіктегі адресіне ақпараттың көшірмесін сақтау керек. Бұл бірнеше процесстер жадыны көрсететін жүйеде ақпаратты сақтайды жадының жеке бөлігі *сегмент* деген атқа ие болады. Жады, осылайша, сызықтық болмайтын болды және екі өлшемдіге айналып кетеді. Адрес екі компоненттен тұрады; сегмент нөмерінен, сегмент ішінде араласудан тұрады.

Оперативті жадыда нақты болатын адрестен процесс қатынас жасайтын адрестің айырмашылығы үлкен. Әрбір жанама жағдайда программа адресін қолдану әр түрлі адреспен беріледі. Мысалы, бастапқы мәтіннің адресі символды болады. Компилятор осы символды адрестерді араласқан адрестермен (модульдің басынан n байтты) байланыстырылады. Мұндай адрес логикалық (кеңістіктегі жады жүйесінде ол вртуальды деп аталады) адрес деп аталады. Барлық логикалық адрестердің жиынтығын *кеңістіктің* (виртуальды) *логикалық адресі* деп атайды.

Адрестерді байланыстыру. Кеңістіктің логикалық және физикалық адресі ұйымдастыру жағынан да, көлем жағынан да бір біріне сәйкес келмейді. Кеңістіктің логикалық адресінің ең жоғарғы көлемі, өлшемі процессор разрядын анықтайды және қазіргі жүйеде кеңістіктің физикалық адресінің өлшемін көбейтеді. ОЖ және процессор негізгі жадыда сәйкесінше ағымдағы программаның орналасуы нақты физикалық адрестегі программа кодына сілтемені көрсететін қабілеті болу керек. Адрестің мұндай түрі *адрес трансляциясы* немесе *адресті байланыстыру* деп аталады. Физикалық программа операторы көмегімен жасалған логикалық адресті байланыстыру операторлың орындалуына дейін немесе оның орындалып жатқан кезінде іске асырылуы керек. Осылайша, жадыға деректермен инструкция байланысы кем жағдайларда орындалады:

1. Компиляция кезеңі. (Compile time) Жадыда компиляция кезінде процесстің орналасатын орныаның белгілі болса, онда физикалық адрес сәйкесінше сол орында өндіріледі. Программаның бастапқы адресін өзгерту үшін оның кодын қайтадан компилирлеу керек. Мысал ретінде MS-DOS-та **.com* программасыналуға болады, ол компиляция стадиясында оны физикалық адреспен байланыстырады.

2. Жүктеу кезеңі. (Load time). Егер компиляция стадиясында программалардың орналасуы туралы ақпарат болмаса, онда компилятор араласқан кодты өндіреді. Бұл жағдайда байланыстыру жүктелген уақытқа дейін орындалмайды. Егер Бастапқы адрес ауыстырылса, онда өзгертілген шама есебінде кодты қайта жүктеу керек.

3. Орындалу кезеңі. (Execution time). Егер процесс жадының бір жағынан екіншісіне өту кезінде ауыстырылған болса, онда байланыстыру стадиясына дейін орындалмайды. Мұнда арнайы құрылғының бар болғанылдұрыс, мысалы ауыстырушы регистрдің болғаны дұрыс. Қазіргі ОЖ-нің көбінде «орындалу» кезеңінде адрес трансляциясы іске асыру арнайы аппарат механизмі үшін қолданылады.