O`ZBEKISTON RESPUBLIKASI AXBOROT TEXNOLOGIYALARI VA KOMMUNIKATSIYALARINI RIVOJLANTIRISH VAZIRLIGI

TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

Fakultet: Kompyuter injiniringi

Kafedra: Kompyuter tizimlari

“ Parallel kompyuter arxitekturasi va dasturlash ” fanidan

Kurs ishi.

Mavzu: Parallel qayta ishlash, ma’lumotlarni taqdim etish va buyruqlarni amalga oshirishda zamonaviy apparat vositalarning vazifalari. Murakkab ifodani ketma-ket va parallel hisoblash chizmasini (graf yordamida) ishlab chiqish va dasturini yaratish.

Guruh: 212-13

Bajardi: Ashuraliyev H.
Abdiyev M.

Rahbar: _____

Himoya qilindi: _____ (sana)
_____ (baho)
_____ (imzo)
_____ (imzo)
_____ (imzo)

O`ZBEKISTON RESPUBLIKASI AXBOROT TEXNOLOGIYALARI VA KOMMUNIKATSIYALARINI RIVOJLANTIRISH VAZIRLIGI

TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

Fakultet: KI

Kafedra: Kompyuter tizimlari.

Yo`nalish (Mutaxasislik): Kompyuter injiniringi.

Tasdiqlayman

Kafedra mudiri_____

<<_____>> _____ 2016 yil

Kurs ishiga

Topshiriq: Parallel qayta ishlash, ma’lumotlarni taqdim etish va buyruqlarni amalga oshirishda zamonaviy apparat vositalarning vazifalari. Murakkab ifodani ketma-ket va parallel hisoblash chizmasini (graf yordamida) ishlab chiqish va dasturini yaratish.

1) Kurs bo`yicha
2) 212-13 guruh talabalari
   Ashuraliyev H. “Parallel qayta ishlash, ma’lumotlarni taqdim etish va buyruqlarni amalga oshirishda zamonaviy apparat vositalarning o’rni va vazifalari (2 – 18 bet) ”
   Abdiyev M. “Murakkab ifodani ketma – ket va parallel hisoblash chizmasini (graf yordamida) ishlab chiqish va dasturini yaratish(19 – 27 bet)”
3) Rahbar:_____
4)

Kurs ishi mavzusi: Parallel qayta ishlash, ma’lumotlarni taqdim etish va buyruqlarni amalga oshirishda zamonaviy apparat vositalarning vazifalari. Murakkab ifodani ketma-ket va parallel hisoblash chizmasini (graf yordamida) ishlab chiqish va dasturini yaratish.

Topshiriq berilgan sana: _____

Mundarija:

Kirish

Kundalik hayotimizda, ofisda turli ilovalardan foydalanganimizda, ta’lim sohasida, ishlab chiqarish sohalarini boshqarishda, turli o‘yinlar o‘ynaganda biz personal kompyuterlardan foydalanamiz. Bularning barchasida kompyuterlardan foydalanish o‘ta qulay va osondek tuyiladi. Ammo, juda murakkab va qiyin masala va vazifalarni yechishda buning aksi bo‘ladi.

Keng foydalanishga bu atama birinchi kompyuterlar, berilgan masalani kerakli paytda yecha olishmagan, paydo bo‘lishi bilan kirib kelgan. Bir kompyuter berilgan vazifani bajara olmasa unda ko‘pgina kompyuterlarni bir paytda bir vazifani bajarishga undash g‘oyasi tug‘ilgan. G‘oya juda foydali edi, ammo birinchi kompyuterlar juda ham haybatli, noqulay va texnologik jihatdan birlashtirish imkoniyatini bermas edi. Keyinchalik texnologiyani rivojlanishi bilan bu imkoniyatlar amalga oshirila boshladi.

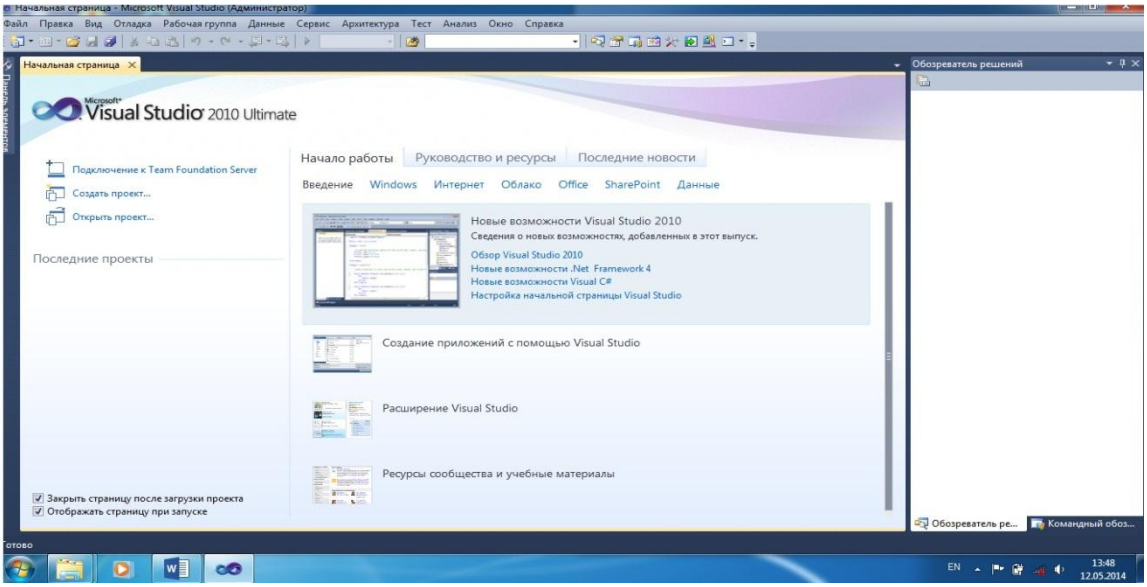
Ushbu kurs ishining maqsadi Parallellashtirish texnologiyalari asosida Parallel kompyuter arxitekturasi va dasturlash fani kesimida murakkab ifodani parallel va ketma – ket hisoblash va hisoblash chizmasi graflar yordamida ishlab chiqish va dasturini ishlab chiqish. Kurs ishini bajarish uchun quyidagi vazifalar qo‘yildi:

- Parallel qayta ishlash, ma’lumotlarni taqdim etish va buyruqlarni amalga oshirishda zamonaviy apparat vositalarning vazifalarini o’rganish;
- Murakkab ifodani ketma-ket va parallel hisoblashni graflar yordamida tushuntirish;
- Parallellashtirish texnologiyasida qo’llagan holda murakkab ifodani ketma-ket va parallel hisoblashn dasturiy ta’minotini yaratish;

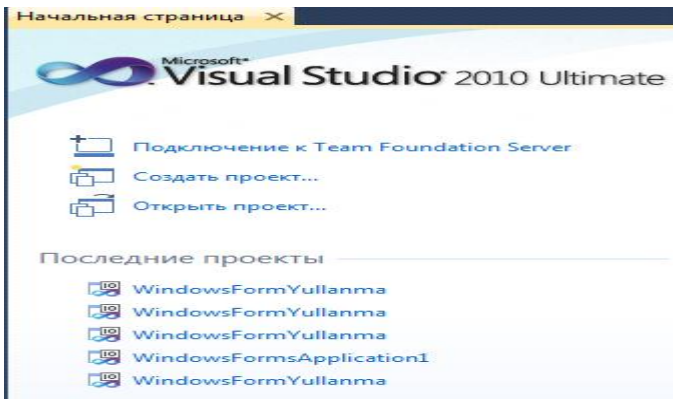
I.NAZARIY QISM

1.1 Microsoft Visual Studio haqida

Microsoft korporatsiyasi o’zining oynalar bilan ishlovchi va dunyoda eng ko’p tarqalgan va juda ham ko’p foydalanuvchilar tomonidan ma’qullangan operatsion tizimi Windows uchun ishlab chiqqan Visual Studio (VS) muhiti ham faqatgina oynalar orqaligina ishlaydi. Unda mavjud barcha dastrulash tillari oynalarga bo’lingan va shu oynalar yordamida bir biriga bog’langan. VS yuklash uchun **пуск->program->Microsoft Visual Studio->Microsoft Visual Studio** ketma ketligini bajarish kerak.VS yuklanganda odatda quyidagi ko’rinishda bo’ladi. Birinchi navbatda,VS yuklanganda Start Page muloqot oynasi ochiladi.



1-rasm. MS Visual Studio muloqot oynasi.



2-rasm. Bosh sahifa oynasi

1.Recent Projects – avval ishlangan loyihalardan 6 tasi ekranda ko'rsatilgan bo'ladi, ularni ustiga sichqonchaning chap tugmasini ikki marotaba bosish orqali ishga tushirish mumkin, shu bo'limning pastki qismida Open: Projects va Create: Projects bandlari bo'lib, open bandi orqali xotirada mavjud bo'lgan loyihani qayta ochish imkonini beradi, create bandi orqali esa yangi loyiha yaratish mumkin.

2. Getting started – loyiha yaratish va shu muhit haqida ma'lumot olish imkonini beradi;

3. Visual Studio Headlines – bu bo'limda sizning shu muhitda ishlayotganingiz uchun uning mualliflari tomonidan bildirilgan minnatdorligi ko'rsatilgan;

4. Visual Studio Developer News – bu asosiy qismda VS ustida qanday o'zgarishlar olib borilayotgani yoki olib borilgani haqida ma'lumot beradi.

Endi biz yangi foydalanuvchi bo'lganimiz uchun bu muloqot oynasini yopib, yangi oyna ochamiz. Buning uchun biz menyular paneli bilan tanishish kerak.

Menyular paneli bilan ishlash.

1.File

- **new** – yangi loyiha yaratish
- **open** – yaratilgan loyihani ochish
- **close** – joriy loyihani yopish
- **close Solution** – barcha loyihalarni yopish
- **save Selected Items** – belgilangan barcha loyiha qismlarini saqlash
- **save Selected Items As** - belgilangan qismlarni barchasini xohlagan tartibda saqlash

- **save All** – barcha dasturlarni saqlash
- **export Template** – dasturni arxivlash
- **page Setup** – sahifa sozlamasini o’zgartirish
- **print** – chop etish
- **recent files** – fayllarni qayta ochish
- **recent Projects** – loyihalarni qayta ochish
- **Visual Studio** dan chiqish

2. Edit

- **undo** – bajarilgan amallarni bir marotaba bekor qilish;
- **redo** – oxirgi bekor qilingan amalni qaytarish;
- **undo Last Global Action** – barcha qilingan amallarni bekor qilish;
- **redo Last Global Action** – bekor qilingan barcha amallarni qaytarish;
- **cut** – belgilangan qismni yoki komponentni qirqib, xotirada saqlab turish;
- **copy** – belgilangan qism yoki komponentaning nusxasini xotirada saqlab turish;
- **paste** – xotiraga qirqib yoki nusxalab olingan qism yoki komponentni belgilangan yoki ko’rsatilgan joyga qo’yish
- **select All** – barcha qismlarni belgilab olish;
- **Find and Replace** – ma’lum qism yoki komponentani butun muhit bo’ylab qidiradi va o’zgartiradi

3.View

- **open** – Joriy dasturni ochib beradi;

- open With – Joriy dasturni xohlagan dastur yordamida ishlov berish imkoniyatini yaratadi;

- server Explorer – Server oynasi bo'lib, unda asosiy komponentalar joylashadi;

- solution Explorer – Ob'yektlar oynasi;

- class View – Loyiha elementlarini ko'rsatib beradi, uning yordamida yangi yoki qo'shimcha ob'yektlarni qo'shish, olib tashlash, ko'rinishini, tartibibni o'zgartirish mumkin;

- code Definition Window – Ob'yekt kodlarini ko'rsatib beruvchi oyna;

- object Browser – Visual Studiodagi barcha ob'yektlar xususiyatlarini va ularni qo'llash usullari beriladigan oyna;

- Error List – xatoliklar oynasi bo'lib, dastur kompiliyatsiya qilinishidan oldin va keyin undagi xatoliklarni ko'rsatib beradi va unda namoyon bo'lgan har bir xatolikni ustida ikki marotaba sichqoncha chiqillatilsa xatolik bo'lgan joyga kursorni eltib qo'yadi;

-output –Dastur o'z ishlashi davomida qanday fayllarga murojaat qilishi, qanday fayllarni hosil qilishi, qaysi cs yordamida hosil bo'lishi va qanday qiymat qaytarishini ko'rsatuvchi oyna;

- properties Window – xususiyatlar oynasi bo'lib, qaysi ob'yekt faol bo'lsa, shu ob'yektning xususiyatlarini;

-toolbox – komponentalar joylashgan oyna bo'lib, unda VS da ishlatish mumkin bo'lgan barcha ob'yektlar joylashtirilgan;

- Find Result – natijani topish oynasi bo'lib, uning yordamida dastur ishlatilganda berilgan qiymatlarda qanday natija olinishini ko'rsatib beradi;

- toolbars – panellar qo’yish yoki olib tashlash bandi bo’lib, unda barcha turdagi panellarni izlash mumkin;

- full Screen – VS butun ekran hajmida kattalashadi;

4. Refactor

- rename – dasturni qayta nomlash;

- extract Method – metodni qayd etish;

- extract Interface – Interfeysni qayd etish;

-remove Parameters–parametrlarni qayta ko’chirish;

- reorder Parameters – parametrlarni qaytarish.

5. Project

- add Windows Form –Windows formasini qo’shadi;

- add User Control – foydalanuvchi boshqarishining boshqa usullarini qo’shadi;

- add Component – Yangi component qo’shadi (foydala-nuvchi tomonidan tayyorlangan bo’lishi ham mumkin);

- add Class – yangi sinf(ma’lumotlar yoki formalar sinfi)ni qo’shadi;

- add New Item – yangi elementini qo’shadi (ob’yektning);

- add Exiting Item – yangi chiquvchi elementini qo’shadi;

- show All Files – barcha fayllarni namoyish etadi;

- console - Properties – Consolning xususiyatlari;

6. Debug

- windows – Breakpoints,Output, Immedite Windows bandlarni o’z ichiga olgan bo’lib, bu oynalar yordamida ishlash imkonini beradi;

- start Debugging – Joriy loyihani komplyatsiya qilib, ishga tushiradi;

-start Without Debugging – Dasturni komplyatsiya qilib, bir necha komponentalarni birgalikda ishga tushiradi;

- new Breakpoints - Yangi to’xtash nuqtalarini hosil qiladi.Bunda hosil bo’lgan nuqtalarni Shift+F5 tugmasi orqali bekor qilish mumkin;

-Delete All Breakpoints – Barcha qo’yilgan to’xtash nuqtalarini bekor qilish uchun ishlatiladi.

7. Data

-Show Data Sources – Shu loyiha bilan bog’langan barcha ma’lumotlar bazalarini ko’rsatib beradi;

-Add New Data Sources – Yangi ma’lumotlar bazasini qo’shish uchun ishlatiladi;

-Schema Compare – Shu loyihada mavjud bo’lgan ma’lumotlar bazalari orasidagi barcha bog’liqliklarni ko’rsatib beradi;

-Preview Data – Shu loyihadagi barcha ma’lumotlar bazalarini qidirish uchun ishlatiladi;

-Refactor – Refaktor bandi bilan bog’liqlik o’rnatadi.

8. Format

-Align – Formaning joylashish koordinatlarini o’rnatish imkonini beradi. Bunda uning chap,o’ng,o’rta, past va yuqoridan chegaralash mumkin;

-Make Same Size – Formaning o'lchamini berish mumkin.Unda balandligi, eni qalinligi va chegara qismini berish mumkin;

New- Yangi loyiha yaratish VS ning eng asosiy amallaridan biri. Bu menyuda yangi loyiha, web-sayt, file (o'zida kodlarni saqlab turuvchi *.cs fayl), oddiy fayl yaratish mumkin. VS da bir necha tillar jamlangan.

1.2 Parallel qayta ishlash jarayonida kompyuter qurilmalarining roli. Ma'lumotlarni taqdim qilinishi va buyruqlarni bajarilishi

Muammolarni paydo bo'lish sababini tushunish uchun dastavval "oddiy" kompyuter qanday tuzilganligini ko'rib chiqamiz. Biz foydalanuvchi ko'zi bilan oddiy kompyuterni super kompyuter sifatida ko'ramiz va uning yutuq va kamchiliklarini ko'rib chiqamiz. Kompyuterdagi eng asosiy ma'lumotli element so'z hisoblanadi. Har bir so'z o'zida tartiblangan bitlar to'plamini ifodalaydi. So'z baytlarga bo'linishi mumkin. Bayt tartiblangan 8 bit ga teng. So'zdagi bitlar soni so'zning uzunligi deyiladi. Muayyan kompyuterlar bir xil so'zga va bir xil so'z uzunligiga ega bo'ladi. Turli xil kompyuterlar turli xil so'z uzunligiga ega bo'lishi mumkin. Misol uchun shaxsiy kompyuterlarda so'z bir baytdan iborat bo'lsa, Sgau-1 kompyuterida so'z 64 bitdan iborat. Agar so'zga qandaydir ma'lumot yozilsa, bu shuni anglatadiki, so'zning har bir biti qabul qilishi mumkin bo'lgan 0 yoki 1 bilan fikrsirlangan bo'ladi. So'zning barcha bitlarining to'plami so'zning tashkil etuvchisini aniqlaydi. So'zlarning to'plamini saqlaydigan qurilma xotira deb nomlanadi. U kompyuter bajaradigan vazifasi, yoki kompyuter tipidan kelib chiqqan holda oddiy yoki murakkab, bir xil qurilmali yoki turli xil qurilmali bo'lishi mumkin. Barcha so'zlar o'z nomiga ega bo'ladi. So'zning nomi adres bilan nomlanadi. Ma'no aniqlagan adres strukturasi xotira strukturasini aks ettiradi. Har xil so'zlar har xil adreslarga ega bo'ladi. Har bir adres muayyan fizik joy – xotira bilan bog'langan bo'ladi. Istalgan kompyuterning asosiy vazifasi xotirada saqlanadigan ma'lumotlarni qayta ishlashdan iborat. U alohida so'zdan tuzilgan, bir xil ma'noli, oddiy funksiya ketma – ketliklari bajarilishi kabi amalga oshiriladi. Qoida shunday, barcha funksiyalar ko'pi bilan 2 ta argumentdan iborat.

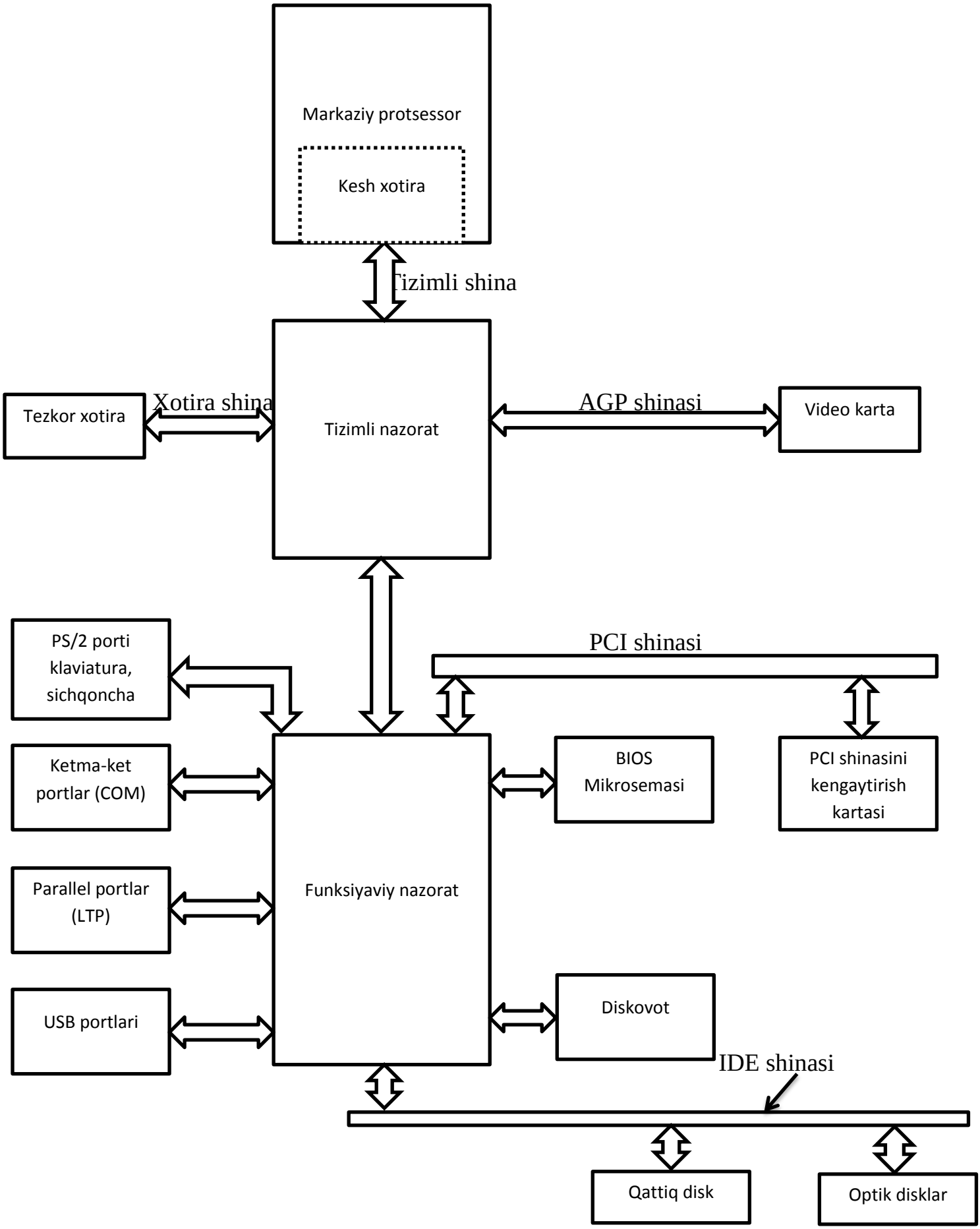
Funksiyalar so’zni to’laligicha yoki so’zning qismlarini ishlatishi yoki ularni o’zgartirishi mumkin. Umumiy holda aytilganda, har xil kompyuterlar har xil foydalanadigan funksiylar to’plamiga ega bo’loladi. Biroq ushbu to’plamlar ko’pincha funksional tomondan qisman yoki butunlay mos tushadi va faqatgina amalga oshirish texnikasi bilan farqlanadi. Ayniqsa, sonlar ustida oddiy arifmetik amallar (qo’shish, ayirish, ko’paytirish vahokazo), bitlar ustida mantiqiy operatsiyalar (konyunksiya, dizyunksiya vahokazo) keng tarqalgan funksiyalardir. Odatda kompyuter terminologiyasida barcha funksiyalar operatsiyalar deb nomlanadi, argument qiymati, ba’zan argumentning o’zi ham va hattoki argument ichidagi so’z adresi operandlar hisoblanadi.

Xotiradagi operatsiyalardan tashqari, kompyuter ma’lumotlarni qayta ishlovchi tashkiliy jarayonlar bilan bog’langan amallarni ham bajarishi kerak. Kompyuterning mumkin amallari tizimining mashina buyrug’ida yoziladi. Har qanday ma’lumot, mashina buyrug’i ham so’zlar kabi yoziladi. Buyruq ta’rifi operatsiya kodlari va operandlarni o’z ichiga oladi.

Bir qancha mashina buyruqlari qat’iy ko’rsatilgan joyda joylashgan (misol uchun fiksirlangan registr) so’zlar ustida amallar bajaradi. Bunday buyruqlar aniq ko’rsatilgan operandlar talab qilmaydi. Tizim komandasi quyidagicha ishlaydi, ya’ni bajarilgan buyruqdan keyin undan keyin keladigan buyruqlarni aniqlab boradi. Har qanday ma’lumotlarni qayta ishlovchi jarayonlar shu kompyuter uchun mumkin bo’lgan sonlar, ya’ni aniq mashina buyruqlari jamlanmasi yordamida ta’riflanadi. Bu jamlanma mashina kodi yoki ichki dasturi kodi deb nomlanadi. Ta’kidlash joizki, jarayonlar mashina tili buyruqlarida yoziladi, boshqacha bo’lishi mumkin emas.

Mashina kodining strukturasi doimo kompyuter strukturasiga mos bo’ladi va foydalanuvchi bajarishi uchun mo’ljallangan amallar strukturasiga umuman o’xshash bo’lmasligi mumkin. Qoida shunday,fondalanuvchi o’z amallarini yuqori darajali tillarda yozadi. Kompyuterlar ularni “tushunmaydi”. Shuning uchun, ular bajarilishi uchun ularning hammasi dastlab ekvivalent mashina kodiga o’girilishi

kerak. Kompliyator juda murakkab va u juda katta hajmdagi ishni bajaradi. Foydalanuvchi dasturining unumdorligini olinadigan mashina kodi va protsessorening masalani yechish unumdorligi belgilaydi. Mashina kodi bajarilgunga qadar barcha buyruqlar va kerakli ma’lumotlar xotiraga yuklangan bo’lishi kerak. Bu kiritish qurilmalari deb nomlangan maxsus buyruqlar orqali amalga oshiriladi. Ularga misol qilib ma’lumotli disklarni o’quvchi disketlar, lazerli disklar, skanerlar, klaviaturalar va boshqalarni olish mumkin. Kompyuterdagi natijalar maxsus buyruqlar yordamida chiqarish qurilmalari orqali xotiradan chiqariladi. Bularga misol qilib ma’lumotlarni diskka yozuvchi qurilmalar, printerlar, ekran va boshqalarni olish mumkin. Xotiraga mashina kodi va ma’lumotlar yuklangandan keyin komyuter o’z ishini boshlashi mumkin. Bularning barchasini boshqarish qurilmasi amalga oshiradi. U registrlar, hisoblagichlar va boshqa elementlardan tashkil topgan va u xotira, arifmetik mantiqiy qurilma, kiritish-chiqarish qurilmasi, kompyuterning boshqa qismalari orasidagi ma’lumotlar uzatilishini ta’minlaydi. Boshqarish qurilmasi 2 ta o’zaro zid topshiriqni bajarishi kerak. Bir tomondan boshqarish qurilmasi, topshiriqlarni yechish jarayonlari to’xtab qolmasligi uchun u yetarli darajada tez ishlashi kerak. Boshqa tomondan esa, boshqarish qurilmasi turli xil qurilmalarni boshqaradi, shu jumladan, bir vaqtda ishlaydigan va bir-biridan uzoqda ishlaydigan qurilmalarni ham boshqaradi. Shuning uchun boshqarish qurilmasi iyerarxik va taqsimalangan arxitektura yordamida tashkil qilinadi. Eng tezkor sath – bu elektron sxema hisoblanadi. Elektron sxema navbatdagi buyruqlarni rasshifrovka qiladi, operator adreslarini va operatsiya kodlarini belgilaydi, analiz uchun navbat turgan bitta yoki bir nechta buyruqlarni tanlaydi. Boshqarish qurilmasining boshqa sathi – bu, masalan, sxemalar, arifmetik-mantiqiy qurilmadagi jarayonlarni boshqaradi. Boshqarish qurilmasi, arifmetik-mantiqiy qurilma va tezkor xotira bloki jamlanmasi markaziy protsessor deb nomlanadi.



3-rasm.Kompyuterning umumiy sxemasi.

Qo’shish va ayirish operatsiyasini bajaradigan qurilma summator deb nomlanadi, ko’paytirish operatsiyasini bajaradigan qurilmaga esa ko’paytirgich deyiladi. Bu qurilmalar arifmetik-mantiqiy tarkibida bo’lib, mantiqiy operatsiyalarni bajaradi. Qo’shish operatsiyasi ko’paytirish operatsiyasiga nisbatan tezroq bajariladi. Istalgan mantiqiy operatsiya qo’shish/ayirish operatsiyasiga nisbatan tez bajariladi. Har bir kompyuterda standart operatsiyalarning bajarilish davomiyligi bir-biridan farq qiladi.

Keng qo’llaniladigan taqsimlangan resursli hisoblash tizimlari arxitekturasiga ko’p protsessorli tizimlar, klasterli hisoblash tizimlari, vektorli protsessorlar, VLIW protsessorlari, superskalyar protsessorlar kiradi va ular muayyan maqsadga yo’naltirilgan bo’ladi.

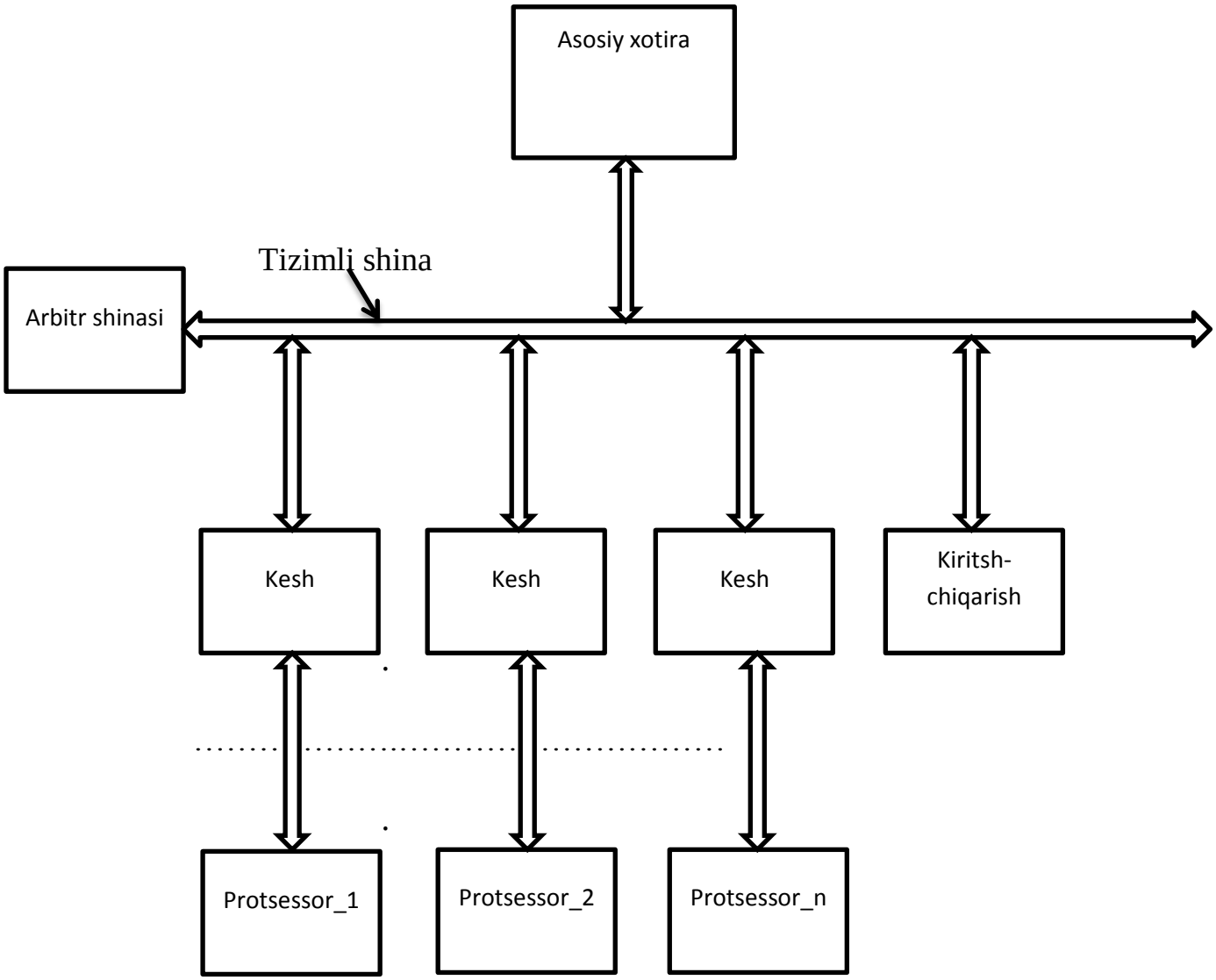
Ko’p protsessorli tizimlar

Ko’p protsessorli tizimlarning 2 xil turi bor:

1. Umumiy xotirali
2. Taqsimlangan xotirali

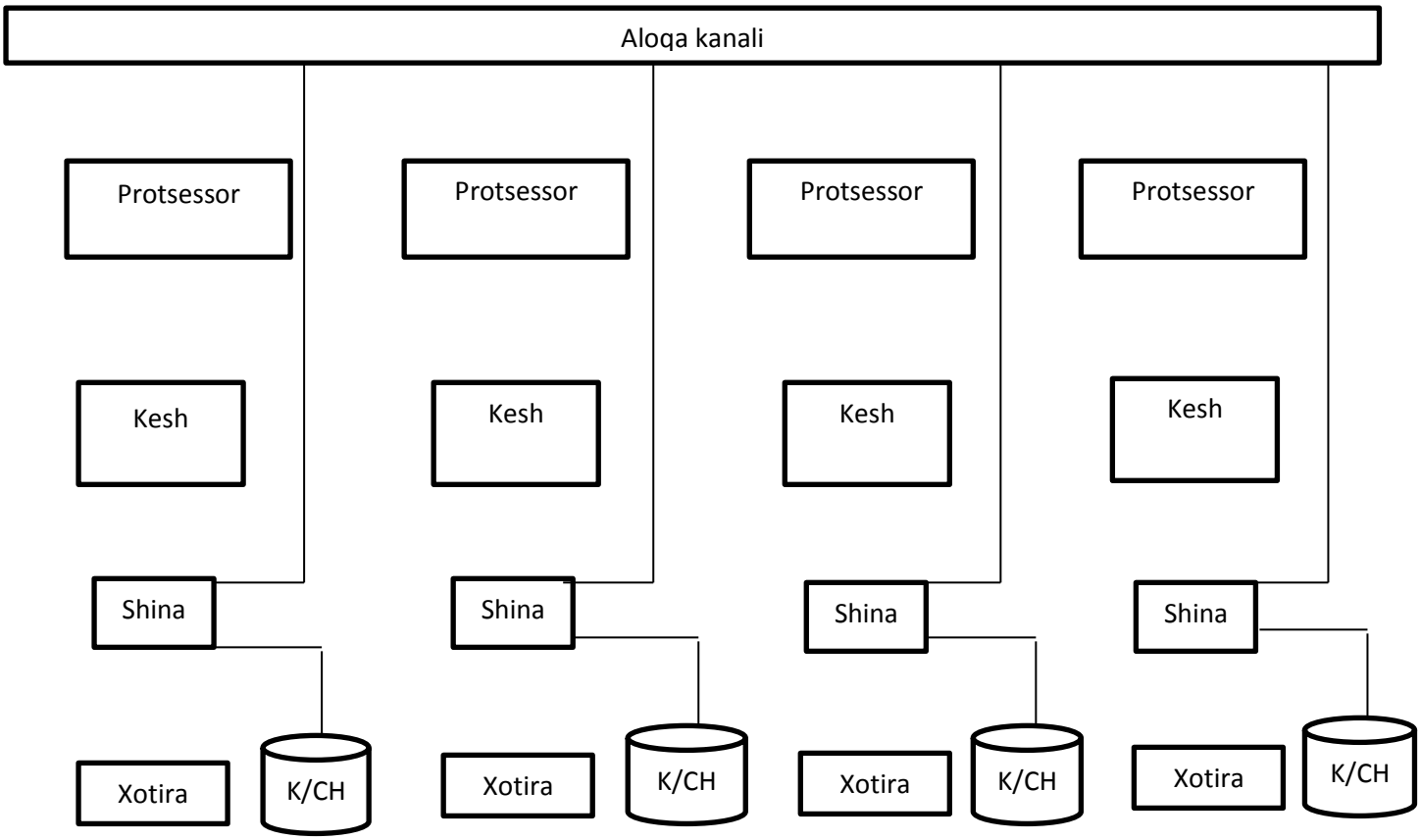
Umumiy xotirali tizimlarda protsessorlar soni ko’p bo’ladi ammo ular yagona, umumiy xotiraga ega bo’ladilar. Umumiy xotirali tizimlarning tipik vakili sifatida simmetrik protsessorli SMP(symmetric multiprocessors) tizimni olish mumkin. SMP tizimida har bir protsessor bir xil unumdorlikga ega va umumiy xotiraga barcha prosessorlarning murojaat huquqi hamda murojaat vaqti davomiyligi ham bir xil bo’ladi. Quyida SMP tiziming strukturasi keltirilgan.

Barcha zamonaviy kompyuterlar uchun operatsion sistema uning ajralmas qismi hisoblanadi va uning dasturiy davomchisi sanaladi. Bir kompyuterning o’zida bir nechta opertasion sistema bo’lishi mumkin. Masalan, shaxsiy kompyuterlar uchun MS-DOS, Windows, Unix operatsion sistemalari keng ommalashgan. Ularning har biri o’ziga xos xususiyatiga ega va mustaqil ravishda o’z vazifasini bajaradi.



4-rasm.SMP tizimi strukturasi.

Taqsimlangan xotirali tizimlar – parallel superkompyuterlar bo'lib, ularning tipik vakili MPP (massive parallel processing) hisoblanadi. MPP tizimi ko'p miqdorda protsessor va xotiraga ega. Bu tizim yuqori tezlik ega kanal orqali bog'langan, ajratilgan mashina (tugun) massivi qilib qurilgan, superkompyuter modeli sifatida foydalaniladi. Har bir mashina (tugun) faqatgina lokal xotiraga murojaat qiladi. Bunda har bir mashina ma'lumotlarni uzatish va qabul qilish yo'li orqali o'zaro bog'liq bo'lmagan jarayonlarni parallel ravishda bajaradi. Quyida MPP tizimining strukturasi keltirilgan.



5-rasm. MPP tizimining strukturasi.

Yuqoridagi ikkala sinf kompyuterlari ko'p hollarda ishchi sinf kompyuteri mikroprotsessorlari yordamida quriladi.

Hisoblash klasteri va hisoblash tarmog'i

Hisoblash klasteri kommunikatsion muhit bilan bog'langan, ko'plab tugunlardan tashkil topgan tizim sanaladi. Bunda tugunlar lokal yoki global tarmoqdagi kompyuter vazifasini bajaradi. Har bir tugun lokal xotiraga ega biroq umumiy operativ xotira mavjud emas. Klaster tarkibida turli xil unumdorlikdagi va har xil arxitektura asosida qurilgan tugunlar bo'ladi. Agar tugunlar unumdorligi va arxitekturasi bir xil bo'lsa bunday klasterlar bir jinsli deyiladi, aks holda esa har xil jinsli klasterlar deyiladi. Klasterlarning arxitekturasi MPP superkompyuterlar arxitekturasiga o'xshash bo'ladi.

1.3 OpenMP paketlarining ishlash prinsipi

OpenMP API (Application Program Interface) bu o'zida umumiy xotirali parallelashgan C, C++ va Fortran dasturlari uchun komplyator diektivalarini, kutubxonalarni va o'zgaruvchilar tavsifini jamlangan paket hisoblanadi.

C, C++ va Fortran tillarida direktivalar bitta dasturda bir nechta ma'lumotlar (SPMD-single program multiple data) tuzilmasi, vazifalar tuzilmasi, qurilma tuzilmasi, ish almashish tuzilmasi va moslashtirish tuzilmasini umumiy xotiraga o'zlashtirib beradi. Vazifasi ishlash vaqtini nazorat qilish, kutubxona va o'zgaruvchilar bilan ta'minlab berishdir.

OpenMP ning asosiy komponentalari 1-jadval.

Direktivalar	**O'zgaruvchan muhit**	**Ishlash vaqti muhiti**
Parallel maydon	Oqimlar soni	Oqimlar soni
Ishni taqsimlash	Jadvallar	Oqimlar ID
Moslashtirish	O'zgaruvchan oqimlarni tartibga solish	O'zgaruvchan oqimlarni tartibga solish
Ma'lumotlar ko'lami sifatlari(private, firstprivate, lastprivate, shared, reduction)	Parallelashtirishni qurish	Parallelashtirishni qurish
Umumlashtirish		Vaqtni hisoblash
		Bloklash uchun API

OpenMP paketining kamchiliklari.

OpenMP API yagona foydalanuvchi uchun mo'ljallangan. OpenMP dasturga bog'liq bo'lgan ma'lumotlar bog'liqligini, ma'lumotlar qarama-qarshiligini, muammoli holatlarni tekshirishni talab qilmaydi. Shu bilan birgalikda dasturdagi kodlar ketma-ketligini xam tekshirishni talab qilmaydi. Dasturchilar OpenMP API to'g'ri foydalanib dastur tuzishlari kerak. OpenMP API faqatgina murojaat qilgandan keyingina ishlaydi.

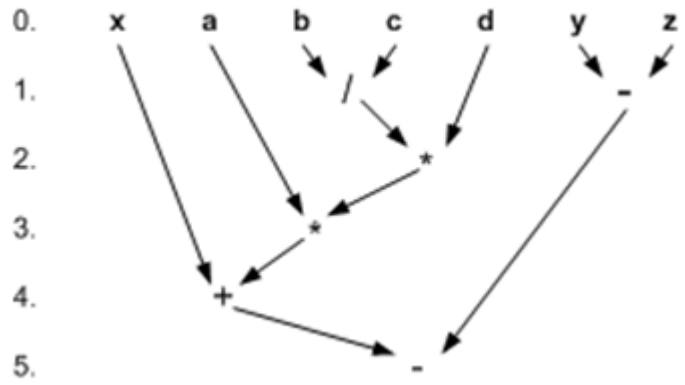
II Amaliy qism

2.1 Murakkab ifodani parallel hisoblash grafini ishlab chiqish

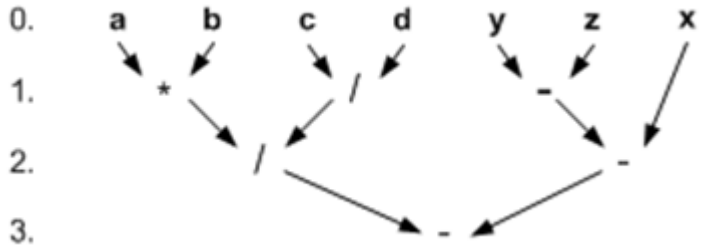
Bizga quyidagi murakkab ifodalar berilgan bo’lsin:

1) E=(x+(a*((b/c)*d)))-(y-z)
2) E'=((a*b)/(c/d)-((y-z)-x)

Bu murakkab ifodalarning parallel hisoblash grafini ishlab chiqamiz. Dastlab E=(x+(a*((b/c)*d)))-(y-z) ifodani parallel hisoblash grafini ishlab chiqamiz. Buning uchun vertikal ravishda parallel amallar ketma – ketligini gorzontal ravishda o’zgaruvchilarni yozib chiqamiz.Bunda parallel bajariladigan amallar bir – biriga bog’liq bo’lmasligi lozim.

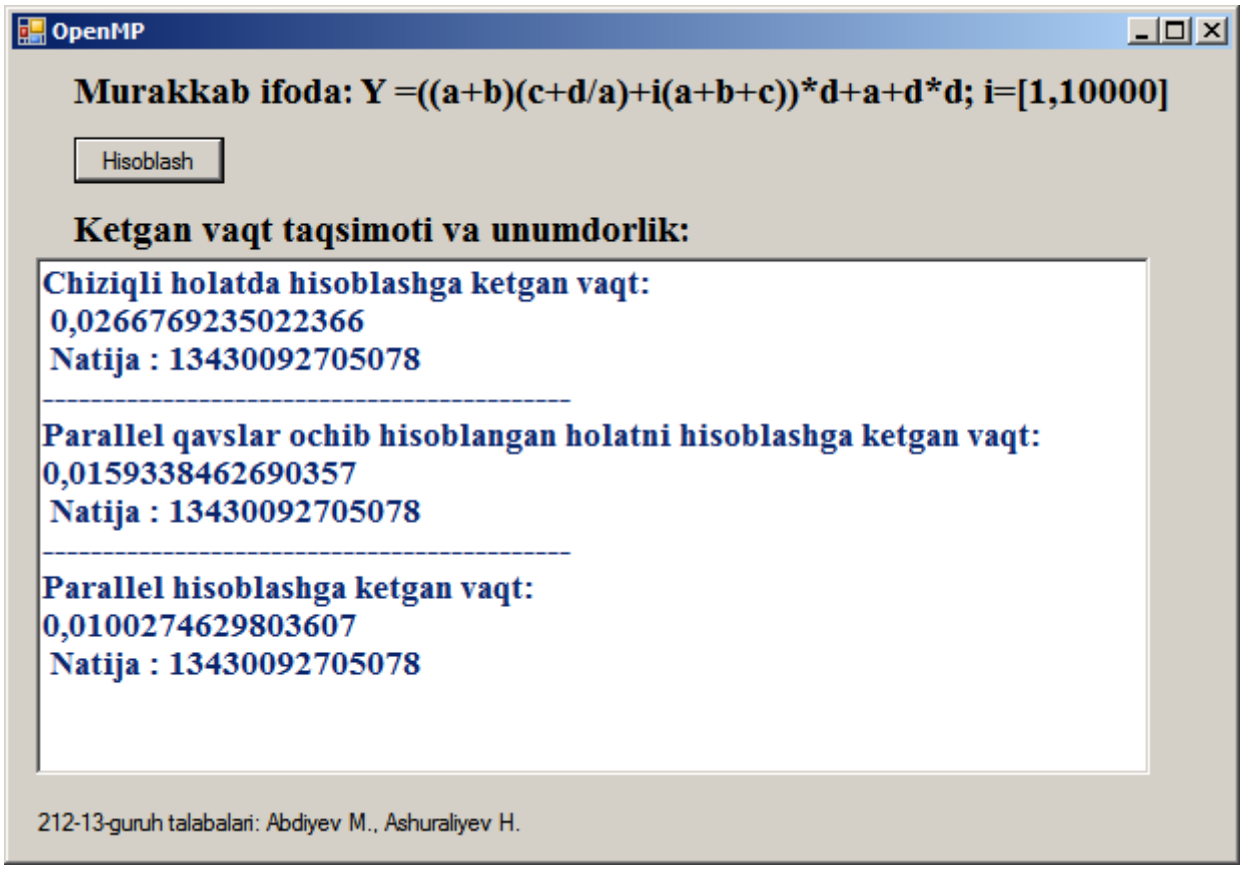


Xuddi shu kabi E'=((a*b)/(c/d)-((y-z)-x) ifoda uchun ham shu grafni tuzib chiqamiz.



Yuqoridagi E va E' ifodalarda amallar soni ham, o’zgaruvchilar soni ham teng. Biroq E ifodani hisoblashda 5 ta takt talab qilinmoqda, E' ifodada esa 3 ta taktning o’zi yetarli bo’lmoqda.

2.2 Murakkab ifodani parallel va ketma-ket hisoblash dasturiy ta'minotini ishlab chiqish



6-rasm.Berilgan ifodani parallel va ketma-ket hisoblashning dasturiy ta'minoti.

Dastur sodda va ixcham bo'lishi uchun faqatgina bitta oynadan foydalanildi. Yuqoridagi rasmdan ko'rish mumkin Y murakkab ifodani ketma-ket va parallel hisoblandi va hisoblash vaqti ham qayt qilingan. Bunda parallel hisoblashni avval qavslarni ochib chiqib hisoblandi, keyin esa o'z berilish holatida hisoblangan. Bu yerda a, b, c, d lar ixtiyoriy haqiqiy sonlar, i esa o'zida [1, 10000] oralig'idagi sonlarni saqlovchi massivdir.

Xulosa

Barcha yirik kompyuterlar va murakkab masalalar kabi iboralar bilan har doim “parallel” so‘zi hamnafas bo‘lib kelgan: parallel kompyuterlar, parallel hisoblash tizimi, parallel dasturlash tillari va h.k.

Keng foydalanishga bu atama birinchi kompyuterlar berilgan masalani kerakli paytda yecha olishmagani paydo bo‘lishi bilan kirib kelgan. Bir kompyuter berilgan vazifani bajara olmasa, unda ko‘pgina kompyuterlarni bir paytda bir vazifani bajarishga undash g‘oyasi tug‘ilgan. G‘oya juda foydali edi, ammo birinchi kompyuterlar juda ham haybatli, noqulay va texnologik jihatdan birlashtirish imkoniyatini bermas edi. Keyinchalik texnologiyani rivojlanishi bilan bu imkoniyatlar amalga oshirila boshlandi.

Ushbu kurs ishini yaratish jarayonida parallellashtirish bo’yicha bir qancha bilimlarini o’zlashtirildi. Kurs ishini tayyorlash davomida quyidagilar bajarildi va ular haqida bilimlarga ega bo’lindi:

- Visual Studio muhiti haqida ma’lumotlarga ega bo’lindi;
- Kompyuterni mutaxassis sifatida o’rganiladi;
- Parallel hisoblash tizimlari va ularning strukturasi ko’rildi;
- Parallel hisoblashni amalga oshiradigan paket (OpenMP) bilan tanishildi;
- Murakkab ifodalarni parallel hisoblash grafini ishlab chiqish o’rganildi;
- Visual Studio muhitida murakkab ifodani parallel va ketma – ket hisoblash dasturiy ta’minoti yaratildi;

Foydalanilgan adabiyotlar

1. M.M. Musayev. “Kompyuter tizimlari va tarmoqlari” Toshkent 2013.

2. A.C. Антоновv “Параллельное программирование с использованием технологии OpenMP”. Москва 2009.

3. V.V. Voyevodin “Parallel hisoblash”. Москва 2002.

4. Stolings U. “Kompyuter tizimlarini arxitekturasi”. Москва 2002.

5. David A. Patterson John L. Hennessy “COMPUTER ORGANIZATION AND DESIGN” 2012.

6. M. Sato “OpenMP. Parallel programming for multicore processors”. University of Tskuba 2012.

Internet resurslar:

7. https://ru.scribd.com

8. www.tenouk.com

9. https://en.wikipedia.org

ILOVA

```
#pragma once
#include <stdlib.h>
#include <math.h>
#include <omp.h>
#include <ctime>
namespace Forma {
using namespace System;
using namespace System::ComponentModel;
using namespace System::Collections;
using namespace System::Windows::Forms;
using namespace System::Data;
using namespace System::Drawing;
/// <summary>
/// Сводка для MyForm
/// </summary>
public ref class MyForm : public System::Windows::Forms::Form
	{
	public: MyForm(void)
		{   InitializeComponent();
			////TODO: добавьте код конструктора
			//}
protected:
	/// <summary>
/// Освободить все используемые ресурсы.
/// </summary>
~MyForm()
		{  if (components)
```

```
            {  delete components;}   }
        private: System::Windows::Forms::Button^  button1;
        protected:
private:
private: System::Windows::Forms::RichTextBox^  richTextBox1;
private: System::Windows::Forms::Label^  label3;
private: System::Windows::Forms::Label^  label4;
private: System::Windows::Forms::Label^  label1;
public:
public:
private:
private: System::ComponentModel::IContainer^  components;
private:
            /// <summary>
            /// Требуется переменная конструктора.
            /// </summary>
#pragma region Windows Form Designer generated code
            /// <summary>
            /// Обязательный метод для поддержки конструктора - не
изменяйте
            /// содержимое данного метода при помощи редактора кода.
            /// </summary>
            void InitializeComponent(void)
            {  this->button1 = (gcnew System::Windows::Forms::Button());
                this->richTextBox1 = (gcnew
System::Windows::Forms::RichTextBox());
                this->button1->Click += gcnew System::EventHandler(this,
&MyForm::button1_Click);
                // // richTextBox1
```

```
//
        this->richTextBox1->Font = (gcnew System::Drawing::Font(L"Times New
Roman", 13, System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(204)));
            this->richTextBox1->ForeColor =
System::Drawing::SystemColors::MenuHighlight;
            this->richTextBox1->Location = System::Drawing::Point(12,
103);
this->richTextBox1->Name = L"richTextBox1";
            this->richTextBox1->Size = System::Drawing::Size(556, 258);
            this->richTextBox1->TabIndex = 4;
            this->richTextBox1->Text = L"";
            //  // label3
            // this->label3->AutoSize = true;
            this->label3->Font = (gcnew System::Drawing::Font(L"Times
New Roman", 14.25F, System::Drawing::FontStyle::Bold,
System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(204)));
            this->label1->AutoSize = true;
            this->label1->Location = System::Drawing::Point(10, 378);
            this->label1->Name = L"label1";
            this->label1->Size = System::Drawing::Size(248, 13);
            this->label1->TabIndex = 11;
            this->label1->Text = L"212-13-guruh talabalari: Abdiyev M.,
Ashuraliyev H.";
            // // MyForm
            this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);
            this->AutoScaleMode =
System::Windows::Forms::AutoScaleMode::Font;
            this->ClientSize = System::Drawing::Size(609, 402);
```

```
                this->Controls->Add(this->richTextBox1);
                this->Controls->Add(this->button1);
                this->Name = L"MyForm";
                this->Text = L"OpenMP";
                this->Load += gcnew System::EventHandler(this,
&MyForm::MyForm_Load);
                this->ResumeLayout(false);
                this->PerformLayout(); }
#pragma endregion
        private: System::Void MyForm_Load(System::Object^  sender,
System::EventArgs^  e) {
        }
        private: System::Void button1_Click(System::Object^  sender,
System::EventArgs^  e) {
                long double s1 = 0, s2 = 0;
                double a = 1.3, b = 3.5, c = 3.1, d = 3.4;
                double t1, t2, k1, k2, k3;
//------------------------chiziqli-------------------------------------
                //t1 = omp_get_wtime();
                t1 = omp_get_wtime();
                for (int i = 0; i < 1000000; i++)
                        s1 += ((a + b)*(c + d / a) + i*(a + b + c))*d + a + d*d;
                t2 = omp_get_wtime();
k1 = (t2 - t1);
                richTextBox1->Text = "Chiziqli holatda hisoblashga ketgan vaqt:\n "
+ k1 + "\t\n Natija : " + s1;
                richTextBox1->Text += "\n----------------------------------------";
                //------------------Parallel no shaklan parallel emas-----------------------
------------------
```

```
t1 = omp_get_wtime();
        omp_set_num_threads(4);
#pragma omp for
        for (int i = 0; i < 1000000; i++){
            s2 += a*c*d + 2 * d*d + d*b*c + b*d*d / a + i*a*d + i*b*d +a
+ i*c*d; }
        t2 = omp_get_wtime();
        k2 = t2 - t1;
        richTextBox1->Text += "\nParallel qavslar ochib hisoblangan holatni
hisoblashga ketgan vaqt:\n" + k2 + "\t\n Natija : " + s2;
        richTextBox1->Text += "\n----------------------------------------";
        //-----------------Parallel----------------------------------------
        s2 = 0;
        t1 = omp_get_wtime();
#pragma omp for
        for (int i = 0; i < 1000000; i++)
            s2 += ((a + b)*(c + d / a) + i*(a + b + c))*d + a + d*d;
        t2 = omp_get_wtime();
        k3 = t2 - t1;
        richTextBox1->Text += "\nParallel hisoblashga ketgan vaqt:\n" + k3 +
"\t\n Natija : " + s2; }
private: System::Void label1_Click(System::Object^  sender, System::EventArgs^
e) {  }
private: System::Void button2_Click(System::Object^  sender,
System::EventArgs^  e) { }
private: System::Void label3_Click(System::Object^  sender, System::EventArgs^
e) { }}; }
```