ФАКУЛЬТЕТ АВТОМАТИКИ, ТЕЛЕМЕХАНІКИ ТА ЗВ'ЯЗКУ

Кафедра спеціалізованих комп'ютерних систем

В.В. Нарожний, О.В. Головко

СУЧАСНІ ПЕОМ: АПАРАТНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

Курс лекцій

Частина 2

Харків – 2014

УДК 004.41-43(075)

Нарожний В.В., Головко О.В. Сучасні ПЕОМ: апаратне та програмне забезпечення: Курс лекцій. – Харків: УкрДАЗТ, 2014. – Ч. 2. – 69 с.

У курсі лекцій викладено теоретичні відомості і схемотехніки, архітектури, устрою та функціонування сучасних обчислювальних систем. Для поліпшення якості знань наведено ряд прикладів мовою програмування Assembler, що дозволяють подивитись на устрій і функціонування сучасних обчислювальних систем з максимально глибоким дослідженням.

Курс лекцій призначений для студентів, що навчаються за спеціальностями 092507 «Спеціалізовані комп’ютерні системи», 091503 «Комп'ютерні інформаційно-управляючі системи», спеціалізаціями експлуатаційної спрямованості в рамках цих спеціальностей, а також для інших спеціальностей відповідних напрямків навчання. Курс лекцій містить лекційний і додатковий довідковий матеріал з дисциплін «Комп’ютерна схемотехніка», «Архітектура персонального комп’ютера» та «Периферійні пристрої обчислювальної техніки» може застосовуватись також при виконанні курсових проектів і робіт, розрахунково-графічних робіт, розділів дипломних проектів, присвячених обчислювальній техніці.

Іл. 24, бібліогр.: 7 назв.

Курс лекцій розглянуто та рекомендовано до друку на засіданні кафедри спеціалізованих комп’ютерних систем 20 грудня 2012 р., протокол № 6.

Рецензенти:
професори Г.Ф. Кривуля (ХНУРЕ),
С.В. Лістровий (УкрДАЗТ)

В.В. Нарожний, О.В. Головко

СУЧАСНІ ПЕОМ: АПАРАТНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

Курс лекцій

Частина 2

Відповідальний за випуск Нарожний В.В.

Редактор Ібрагімова Н.В.

Підписано до друку 24.12.12 р.

Формат паперу 60х84 1/16. Папір писальний.

Умовн.-друк.арк. 3,5. Тираж 60. Замовлення №

Видавець та виготовлювач Українська державна академія залізничного транспорту,
61050, Харків-50, майдан Фейєрбаха, 7.
Свідоцтво суб’єкта видавничої справи ДК № 2874 від 12.06.2007 р.

ЗМІСТ

ВСТУП

Центральним елементом кожної цифрової електронно-обчислювальної машини є центральний процесорний пристрій (ЦПП). Вже багато років термін «центральний процесор» є близьким терміну «мікропроцесор». Це сталося, коли майже всі електронні схеми ЦПП було вшито на кристалі однієї мікросхеми, яку назвали мікропроцесором. Перший мікропроцесор було розроблено корпорацією Intel у 1971 році. Саме цей рік є переворотним у розвитку обчислювальної техніки.

За останні 40 років технічний прогрес зробив великий стрибок. Особливо це стосується значного поширення цифрових електронно-обчислювальних систем майже в усі види діяльності людини. Все менше людина може жити без обслуговування цифровою електронно-обчислювальною технікою, іноді майже не підозрюючи про це. Цифрова електронно-обчислювальна техніка присутня в багатьох домашніх приладах і на виробництві. Усюди іде модернізація залізнично-дорожнього транспорту та станцій, метою якої є заміна пристроїв на сучасні комп'ютеризовані системи і це стосується всіх напрямків розвитку цивілізації. Мікропроцесор можна зустріти в багатьох сучасних приладах, таких як **телефони**, **пральні машини** та інші; вони відповідають за роботу **двигунів** і систем **гальмування** сучасних **автомобілів**, за їх допомогою створюються **системи контролю** і **системи збору інформації**. Шлях всесвітньої комп'ютеризації і далі буде продовжуватись і лише збільшувати темпи. Тому вивчення принципів роботи мікропроцесорів є найважливішим напрямком роботи вищої освіти.

У конспекті лекцій розглядається робота найпопулярнішої серії мікропроцесорів Intel, що є основою універсальної та найбільш популярної родини обчислювальних машин – **персонального комп'ютера**.

Мета конспекта лекцій – навчити студентів розуміти роботу всіх вузлів персонального комп'ютера та деяких зовнішніх периферійних пристроїв. Кожна лекція містить опис фрагментів лабораторних робіт, де застосовано мову програмування Assembler за допомогою безкоштовних програмних пакетів MyASM та FAR.

ЛЕКЦІЯ 1. Еволюція комп'ютерних технологій. Четверте покоління комп'ютерів (з 1971 року)

У 1965 році голова ради директорів компанії "Intel" Гордон Мур передбачив, що кількість елементів на інтегральних мікросхемах повинні подвоюватися кожні 18 місяців. Надалі це правило, відоме як закон, було застосовано до швидкості мікропроцесорів і досі не порушувалося.

У 1969 році компанія "Інтел" випустила майже найважливіший для розвитку обчислювальної техніки пристрій - **мікропроцесор**. Мікропроцесор являє собою інтегральну мікросхему, на якій зосереджений оброблювальний пристрій з власною системою команд. Конструкція мікропроцесора дозволяє застосовувати його для вирішення широкого кола завдань, створюючи при цьому різні функціональні пристрої. Використання мікропроцесорів значно спростило конструкцію комп'ютерів. Практично відразу мікропроцесори отримали широке застосування в різних системах управління: від космічних апаратів до побутових приладів.

Протягом наступних десятиріч, слідуючи закону Мура, триває збільшення швидкості та інтеграції мікропроцесорів. З'явилися надвеликі інтегральні схеми, що включають сотні тисяч і навіть мільйони елементів на один кристал. Це дозволило продовжити зменшення розмірів, вартості комп'ютерів і підвищити їхню продуктивність і надійність.

Практично одночасно з мікропроцесорами з'явилися мікрокомп'ютери, або персональні комп'ютери, відмітною особливістю яких є невеликі розміри і низька вартість. Завдяки своїм характеристикам персональні комп'ютери надали можливість практично будь-якій людині познайомитися з обчислювальною технікою. Комп'ютери перестали бути прерогативою крупних компаній і державних установ, а перетворилися на товар масового споживання.

Першою у виробництві персональних комп'ютерів була компанія Apple. Її фундатори Стів Джобс і Стів Возняк створили першу модель персонального комп'ютера у 1976 році і назвали її Apple I. У 1977 році вони представили свій комп'ютер членам комп'ютерного клубу в Каліфорнії і наступного дня отримали

замовлення на 50 подібних комп'ютерів. Вартість першого персонального комп'ютера складала лише 500 дол. У 1977 році компанія Apple представила наступну модель персонального комп'ютера – Apple II. У нової моделі був витончений пластиковий корпус із вбудованою клавіатурою. Вперше комп'ютер набув рис побутового приладу. Продажі персональних комп'ютерів різко зросли. Apple II остаточно зламав уявлення про комп'ютер як про величезного залізного монстра, у нього був витончений дизайн і доброзичливий інтерфейс взаємодії з користувачем.

ПК не цікавили крупні компанії до 1979 року, коли з'явився перший процесор електронних таблиць – VISICALC. Ідея VISICALC була запропонована студентом Гарварду Даном Бріськліном, якому довелося розв'язувати складні фінансові задачі, що вимагають великої кількості обчислень. Зі своїм другом Бобом Франкстоном вони написали VISICALC для комп'ютера Apple II. Програма виявилася настільки зручною для фінансових обчислень, що багато компаній стали купувати Apple II з VISICALC для своїх співробітників.

У 1981 році найбільша комп'ютерна компанія IBM представила свій перший персональний комп'ютер – IBM PC. Протягом двох років було продано більше п'яти мільйонів цих комп'ютерів. У той же час компанія Microsoft починає випуск ПО для IBM PC. З'являються клони IBM PC, але всі вони, так чи інакше, відображають стандарти, закладені IBM. Поява клонів IBM PC сприяла зростанню промислового виробництва ПК.

Протягом усього 50 років комп'ютери перетворилися з незграбних дивовижних електронних монстрів у могутній, гнучкий, зручний і доступний інструмент. Комп'ютери стали символом прогресу в XX столітті. Людині треба буде обробляти все більшу кількість інформації, а тому будуть удосконалюватися і засоби її обробки – комп'ютери.

ЛЕКЦІЯ 2. Intel Corporation — винахідник першого мікропроцесора

Intel Corporation (*Інтел Корпорейшн*) — корпорація США, що виробляє широкий спектр електронних пристроїв і комп'ютерних компонентів, включаючи напівпровідники, мікропроцесори, набори системної логіки (чипсет) та ін. Штаб-квартира — у місті Санта-Клара, штат Каліфорнія, США.

Компанію заснували Роберт Нойс і Гордон Мур у 1968 році. До них незабаром приєднався Енді Гроув. Після довгих роздумів засновники назвали компанію «**Intel**» (від слів «інтегральна електроніка» — **Integrated Electronics**). Бізнес-план компанії був роздрукований на друкарській машинці Робертом Нойсом і займав лише одну сторінку. Представивши його фінансисту, Intel отримала стартовий кредит в 2,5 млн дол.

Успіх до компанії прийшов у 1971 році, коли Intel почав співпрацю з японською компанією Busicom. Intel отримав замовлення на дванадцять спеціалізованих мікросхем, але за пропозицією інженера Теда Хоффа компанія розробила один універсальний мікропроцесор Intel 4004. Продуктивність цього процесора була порівнянна з продуктивністю наймогутніших комп'ютерів того часу.

Гордон Ерл Мур (*Gordon Earle Moore*; народився 3 січня 1929 році, Сан-Франциско, Каліфорнія) — почесний голова ради директорів і засновник корпорації Intel, основоположник «закону Мура».

На першому і другому курсах Мур навчався в державному університеті Сан-Хосе, де зустрів свою майбутню дружину Бетті.

У 1950 році отримав ступінь бакалавра хімії в Каліфорнійському університеті в Берклі, штат Каліфорнія, США. У 1954 році отримав ступінь доктора наук у галузі хімії та фізики в Каліфорнійському технологічному інституті.

З 1953 року працював у лабораторії прикладної фізики в Johns Hopkins University. З 1956 року працював в Shockley Semiconductor Laboratory в Пасло Альто, Каліфорнія, під керівництвом Вільяма Шоклі (William Shockley).

У вересні 1957 року Мур і ще сім талановитих інженерів («віроломна вісімка») йдуть з Shockley Semiconductor Laboratory

через розбіжності з Шоклі та засновують власну компанію Fairchild Semiconductor для роботи з кремнієвими транзисторами в Mountain View, Каліфорнія. З 1957 року працював у Fairchild Semiconductors начальником відділу, а з 1959 року Мур стає директором і керує групою з розроблення n-p-n транзистора.

У липні 1968 року Мур з Робертом Нойсом ідуть з Fairchild Semiconductors і стають засновниками корпорації Intel. Перший час Мур займав посаду виконавчого віце-президента. У 1975 році він був призначений президентом і головним виконавчим директором Intel. Ці посади він займав до квітня 1979 року, коли його обрали головою ради директорів і головним виконавчим директором корпорації. Гордон Мур залишався на посаді головного виконавчого директора корпорації до 1987 року, а в 1997 році у зв'язку з досягненням пенсійного віку йому було привласнено звання почесного голови ради директорів.

За станом на 2008 рік (за даними Форбс) статок Гордона Мура оцінюється в 3,7 млрд дол. (288-ма позиція в списку найбагатших людей планети (Форбс 400)). У 2004 році він посідав 75-ту позицію з 5,5 млрд дол.

Віроломна вісімка (англ. *Traitorous Eight*) — вісім молодих співробітників Shockley Semiconductor Laboratory, які звільнилися у вересні 1957 року через конфлікт з Уїльямом Шоклі та створили власну компанію – Fairchild Semiconductor. Ці вісім чоловік згодом зробили значний внесок у розвиток електроніки та «Силіконову долину» .

ЛЕКЦІЯ 3. Мікропроцесори Intel

3.1 Intel 4004 — перший доступний мікропроцесор у світі

Intel 4004 — 4-бітовий мікропроцесор, розроблений Intel Corp. і випущений 15 листопада 1971 року. Ця мікросхема вважається першим у світі комерційно доступним однокристальним мікропроцесором. Проте в 1970 році, більш ніж за рік до виходу чипа i4004, був виготовлений військовий мікропроцесор F14 CADC(en), який був засекречений до 1998 року.

У 1969 році невелика японська компанія *Nippon Calculating Machine, Ltd.* (згодом Busicom Corp.), що займалась виробництвом калькуляторів, замовила в Intel 12 мікросхем, які повинні були використовуватися в новому настільному калькуляторі. Такі мікросхеми завжди характеризувалися вузькоспеціалізованими функціями та призначалися для виконання строго певної роботи, тому для кожного нового застосування доводилося знову розробляти весь набір мікросхем. Такий підхід співробітникам Intel здався невигідним. 32-річний Маршіан Едвард (Тед) Хофф (Ted Hoff) пропонує керівництву Intel і Busicom зменшити число мікросхем, використовуючи центральний процесор, який повинен буде виконувати арифметичні та логічні функції – один замість декількох мікросхем. Ідея була ухвалена керівництвом обох фірм. Протягом осені 1969 року Тед Хофф за допомогою Стенлі Мейзор запропонував нову архітектуру мікросхем, число яких було скорочено до чотирьох, включаючи центральний процесор: 4-розрядний центральний процесор (ЦПУ), ПЗП для зберігання ПО та ОЗП для зберігання даних користувача.

Назва 4004 пішла від категорій. Кожній категорії продукції була привласнена своя цифра. Першими виробами Intel стали мікросхеми пам'яті (PMOS-чипи), якій була привласнена нумерація 1xxx. У серії 2xxx розроблялися мікросхеми NMOS. Біполярні мікросхеми були віднесені до серії 3xxx. 4-розрядні мікропроцесори отримали позначення 4xxx. Мікросхеми CMOS отримали позначення 5xxx, пам'ять на магнітних доменах — 7xxx, 8-ми та більш розрядні мікропроцесори та мікроконтролери належали до серії 8xxx. Серії 6xxx та 9xxx не використовувалися.

Друга цифра позначала тип продукції: 0 — процесори, 1— мікросхеми RAM, 2 — контролери, 3 — мікросхеми ROM, 4 — зсувні регістри, 5 — мікросхеми EPLD, 6 — мікросхеми PROM, 7 — мікросхеми EPROM, 8 — чипи спостереження та схеми синхронізації в генераторах імпульсів, 9 — чипи для телекомунікацій.

Третя і четверта цифра відповідали порядковому номеру виробу, а оскільки для роботи першого процесора були потрібні ще три спеціалізовані мікросхеми (мікросхеми ROM, RAM і

розширювач введення-виведення), які випущені раніше, ніж 4004, то мікропроцесор отримав ім'я 4004.

15 листопада 1971 року виходить мікросхема 4004 — перший мікропроцесор, який при вартості 200 дол. реалізовував на одному кристалі всі функції процесора великої ЕОМ. Перший у світі мікропроцесор був анонсований у листопаді 1971 року в журналі «Electronic News» (en:Electronic News).

Мікропроцесор 4004 випускався в 16-контактному корпусі типу DIP, розміри кристалу були менше 1 $см^2$. Процесор міг виконувати 60 000 інструкцій за секунду. (Для порівняння, один з перших повністю електронних комп'ютерів — американський ЭНИАК — виконував тільки 5000 інструкцій за секунду, займав площі в 278,7 $м^2$ важив 30 т.) Фірма Intel передбачила вирішальне значення мікропроцесорів у мініатюризації комп'ютерів і тому викупила за 60 тис. дол. у фірми Busicom авторські права на мікропроцесор 4004 та його вдосконалені версії.

Втім, у 1971 році процесор так і не став хітом продажів. Стратегія фірми Intel була направлена на те, що збут 4004 розширює ринок набагато більш популярних мікросхем пам'яті 1101/1103. Заслуженою популярністю став користуватися тільки мікропроцесор 8080, електронний «правнук» 4004.

Спеціалізовані мікросхеми серії 4xxx

Чип 4004 поставлявся з трьома спеціалізованими мікросхемами: мікросхеми ROM, RAM і розширювач введення-виведення. І хоча в цих мікросхем була своя система позначень (серії 1xxx, 2xxx і 3xxx), вони отримали друге найменування в категорії 4xxx, яке стало позначатися поряд з їхньою звичайною нумерацією.

4001. 256-байтове ПЗП (256 8-бітових програмних інструкцій) та один вбудований 4-битовий порт введення-виведення.

4002. 40-байтове ОЗУ (80 4-бітових комірок) та один вбудований 4-бітовий вихідний порт; RAM у чипі організована в 4 «регистри» з двадцяти 4-бітових комірок: 16 комірок даних (в оригінальному калькуляторі використовувалися для цифр мантиси), 4 комірок стану (в оригінальному калькуляторі використовувалися для цифр експоненти та знаків).

4003. 10-битовий «розширювач введення-виведення» (зсувний регістр, що перетворює послідовний код у паралельний).

Крім того, у сімействі 4xxx були випущені мікросхеми 4008 та 4009, які так само могли поставлятися з 4004.

4008*. 8-битовий фіксатор адреси для доступу до стандартних чипів пам'яті і один вбудований порт введення-виведення.

4009*. Перетворювач доступу введення-виведення до стандартної пам'яті та чипів введення-виведення.

Сімейство 400x також іменували як *MCS-4* (Micro Computer Set 4-bit).

Також фірма Інтел продавала *Intellec-4* (великі сині коробки) — систему розроблення і тестування програм для 4004. Фактично це була одна з перших мікро-ЕОМ, зібрана на основі серії 4xxx (чипи 4004, 4201, 4001x4 і 4002x2). Лише велика ціна (5 тис. дол.) не дозволяла назвати її персональним комп'ютером.

У колекціонерів Intel 4004, природно, є однією з найпопулярніших мікросхем у плані колекціонування. Найбільш високо цінуються біло-золоті мікросхеми Intel 4004 з видимими сірими слідами на білій частині (оригінальний тип корпусу). Так, у 2004 році така мікросхема на on-line аукціоні eBay оцінювалася приблизно в 400 дол.

Технічні характеристики:

- дата анонсу: 15 листопада 1971 року;
- тактова частота: 108 кГц (насправді 92,6 кГц згідно з документом, де сказано, що цикл інструкції триває 10,8 мікросекунд, звідси і плутанина маркетологів Intel — перша в історії помилка Intel);
- частота синхронізації: 740 кГц (ділиться на 8 до тактової, в описі указана як мінімальний період такту (clock period) 1,35 мікросекунд);
- гарвардська архітектура;
- внутрішній стек 3 рівні;
- розрядність шини: 4 біт;
- пам'ять команд (ПЗП): 4 кбайт;
- об'єм пам'яті, що адресується: 640 байт;

- кількість регістрів: 16 4-битових (можуть бути використані, як 8 8-бітових);

- кількість транзисторів: 2300;

- площа кристалу ($мм^2$): <100;

- техпроцес (нм): 10000 (10 мкм);

- напруга живлення: +15 В;

-рознімач: мікросхема безпосередньо впаювалася в друкарську плату або встановлювалася в спеціальний слот;

- корпус: 16-контактний керамічний DIP (CERDIP);

- підтримувані технології: 46 інструкцій (з яких 41 — 8-розрядні та 5 — 16-розрядні).

3.2 Intel 4040

Intel 4040 — 4-бітновий центральний процесор, розроблений Intel Corp. і випущений на початку 1972 року. Мікросхема є наступником Intel 4004 та попередником першого 8-битового мікропроцесора Intel 8008.

Мікросхема i4040 використовувалася, в основному, в іграшках, в управлінні обладнання, у тестових пристроях. Корпус i4040 удвічі ширше за корпус i4004 і має 24 виводи. У новий процесор добавили 14 (з інших джерел — 16) нових команд, збільшили глибину стека до восьми рівнів, також з'явилася підтримка переривань. Сімейство мікросхем i4040 так само іменували як MSC-40.

Особливості i4040:

- з’явилась підтримка переривань;

- кількість інструкцій розширено до 60 команд;

- число регістрів збільшено до 24;

- ПЗП розширено до 8 кбайт;

- глибина стеку складає 8 рівнів.

3.3 Intel 8008 – перший 8 розрядний

Intel 8008 — перший 8-бітовий (розрядний) центральний процесор, розроблений фірмою Intel Corp. і випущений 1 квітня 1972 року. Процесор позиціонувався як процесор для просунутих

калькуляторів загального призначення, терміналів введення-виведення та автоматів розливу рідини.

У 1969 році компанія Computer Terminal Corporation (CTC), згодом перейменована в Datapoint, замовляє в Intel новий процесор, який CTC мала намір інтегрувати в новий термінал Datapoint 2000. Спочатку CTC хотіла розмістити все на декількох мікросхемах, проте інженер компанії Intel — Тед Хофф (Ted Hoff), поглянувши на проект, запропонував розмістити всі компоненти нового процесора на одній мікросхемі. Отримавши згоду CTC, інженери Intel приступили до розроблення мікросхеми, яка отримала робочу назву 1201. У 1970 році, коли мікросхема була вже практично готова, CTC відмовляється від проекту. Причини були такі: мікросхему надали запізно, а характеристики мікросхеми не задовольняли CTC, у результаті вона не використовувалася в терміналах Datapoint 2200. Договір між Intel та CTC був розірваний, що дозволяло продавати цю мікросхему іншим компаніям. Мікросхемою 1210 зацікавилася японська компанія Seiko. Згодом, після деякої модифікації, 1210 перетворилася на мікропроцесор i8008 (випущений у 1972 році), який поклав початок новому сімейству MCS-8.

Мікропроцесор i8008 архітектурно був дуже схожий на i4004, багато рішень, використаних в i4004, також застосовувалися в i8008. Новий процесор успадкував систему позначень, що використовувалась у 4004, аналогічним чином у сімейство продукції «8xxx» увійшли всі мікросхеми RAM, ROM та EPROM, що підтримують мікропроцесор 8008.

Характеристики:

- розмір регістрів – 8 біт (1 байт);
- 14-бітов адресація пам’яті (процесор міг підтримувати до 16 кбайт зовнішньої пам’яті);
- 8-бітова шина даних;
- процесор міг звернутися до восьми портів введення та 24 портів виведення.

3.4 Intel 8080

Intel 8080 **-** 8-бітовий мікропроцесор, що розроблено компанією Intel у квітні 1974 року. Являє собою вдосконалену

версію процесора Intel 8008. За твердженнями Intel, цей процесор забезпечував десятиразовий приріст продуктивності порівняно з мікропроцесором Intel 8008 (580ВМ80 – клон, що випустив СРСР).

Новий процесор випускався за новітньою 6-мікронного NMOS технологією, що дозволило розмістити на кристалі 6000 транзисторів. Процесор, хоча й був побудований на архітектурі Intel 8008, але мав безліч відмінностей від свого попередника, завдяки яким і отримав велику популярність.

У новому процесорі порівняно з попередником була дуже розвинута система команд: 16 команд передачі даних, 31 команда для їхньої обробки, 28 команд для переходу (з прямою адресацією), 5 команд управління. У мікропроцесорі Intel 8080 не було команд множення та ділення (їх реалізовували за допомогою підпрограм або Intel пропонувала зовнішню мікросхему-співпроцесор). Завдяки 16-розрядній адресній шині процесор дозволяв проводити адресацію 64 кбайт пам'яті, яка не розділялася на пам'ять команд і даних. Хоча процесор і був 8-розрядним і містив сім 8-бітових регістрів (A, B, C, D, E, H, L), він мав обмежені можливості обробки 16-розрядних чисел, для чого регістри об'єднувалися в пари BC, DE, HL. У новому процесорі використовувався стек у зовнішній пам'яті (в Intel 8008 він був внутрішнім).

Існує невелика плутанина в позначеннях саме цього процесора. Первинний варіант i8080 мав 48-вивідний планарний корпус з кроком виведення 1/20 дюйма, максимальну тактову частоту 2 МГц та одну досить серйозну помилку, яка теоретично могла привести процесор у стан, з якого він виводився тільки сигналом reset. Поліпшений варіант 8080A, випущений через півроку, мав корпус DIP-40 з кроком виведення 1/10 дюйма, максимальну тактову частоту 2,5 МГц, а помилка в ньому була виправлена. Більшість авторів, використовуючи позначення 8080, мають на увазі насправді 8080A.

На базі мікропроцесора Intel 8080 фірмою MITS був випущений «перший у світі мінікомп'ютерний комплект, який може змагатися з промисловими зразками» Altair-8800, який мав неймовірно велику на ті часи популярність (MITS не встигала навіть вчасно обробляти замовлення).

Крім Altair-8800, мікропроцесор Intel 8080 також застосовувався в пристроях управління вуличним освітленням і світлофорами, станках з ЧПУ, а також в іншому обладнанні.

Технічні характеристики:

- дата анонсу: квітень 1974 року;
- тактова частота: 2 МГц (пізніше 2,5 та 3 МГц);
- розрядність регістрів: 8 біт;
- розрядність шини даних: 8 біт;
- розрядність шини адреси: 16 біт;
- об'єм пам'яті, що адресується: 64 кбайт;
- кількість транзисторів: 6000;
- техпроцес (нм): 6000 (6 мкм);
- необхідні джерела живлення: +5В, -5В, +12В;
- рознімач: мікросхема припаювалася до плати;
- корпус: 40-контактний керамічний DIP;
- підтримувані технології: 80 інструкцій.

3.5 Intel 8086 – початок історії сучасного персонального комп’ютера

Intel 8086 – Перший 16-бітовий процесор компанії Intel, що розроблено 8 червня 1978 року. Процесор мав набір команд, який застосовується в сучасних процесорах, саме від цього процесора бере свій початок відома на сьогодні архітектура x86.

Конкурентами мікропроцесора i8086 є такі розробки, як NEC V30, який був на 5% швидшим, ніж i8086, але при цьому був повністю з ним сумісний. Радянським аналогом є мікропроцесор К1810ВМ86, що входив до серії мікросхем К1810.

Ринок 8-розрядних мікропроцесорів наприкінці 1970-х років був переповнений. Intel залишає спроби на ньому закріпитися та випускає перший 16-бітовий (розрядний) процесор. Процесор i8086 являє собою модернізований процесор i8080. Хоча розробники не ставили за мету досягти повної сумісності на програмному рівні, більшість програм, написані для i8080, можуть працювати на i8086. Новий процесор несе в собі безліч змін, які дозволили значно (у 10 разів) збільшити продуктивність порівняно з попереднім поколінням процесорів.

Регістри. Всього у процесорі i8086 було 14 16-розрядних регістрів: 4 регістри загального призначення (AX, BX, CX, DX), 2 індексні регістри (SI, DI), 2 вказівних регістри (BP, SP), 4 сегментні регістри (CS, SS, DS, ES), програмний лічильник або покажчик команди (IP), регістр прапорів (FLAGS, включає 9 прапорів). При цьому регістри даних (AX, BX, CX, DX) припускали адресацію не тільки цілих регістрів, але і їхньої молодшої половини (регістри AL, BL, CL, DL) і старшої половини (регістри AH, BH, CH, DH), що дозволяло використовувати не тільки нове 16-розрядне ПО, але й зберігало сумісність зі старими програмами (правда, їх необхідно було, принаймні, перекомпілювати).

Шини. Розмір шини адреси був збільшений з 16 до 20 біт, що дозволило адресувати 1 Мбайт (1024 кбайт) пам'яті. Шина даних була 16-розрядною. Проте в мікропроцесорі шина даних і шина адреси використовували одні й ті самі контакти на корпусі (мультиплексування). Це призвело до того, що не можна одночасно подавати на системну шину адреси і дані. Мультиплексування адрес і даних у часі скорочує кількість контактів корпусу до 40, але уповільнює швидкість передачі даних.

Робота з пам'яттю. Для того щоб адресувати більший, ніж у i8080, об'єм пам'яті, знадобилось змінити спосіб адресації пам'яті. Адже якщо використовувати старі методи, коли адреса до елемента пам'яті містилася у вказівних регістрах, то довелося б збільшувати розмір цих самих регістрів, щоб мати можливість звертатися до більшого об'єму пам'яті. Тому для адресації 1 Мбайт пам'яті застосували таку схему. На шину адреси виводили фізичну адресу розміром 20 біт, яка формувалася шляхом складання вмісту одного з сегментних регістрів (16 біт), помноженого на 16, з вмістом вказівного регістру (рисунок. 3.1). Таким чином, адресація елемента пам'яті проводилася за номером сегмента та ефективною адресою комірки в сегменті (так званому зміщенні). Якщо результат складання виявлявся більше, ніж 1 Мбайт, то 21-й біт відкидався; така процедура називається «завертанням» адреси (англ. address wraparound). Цей метод назвали реальним режимом адресації процесора, такий режим дозволяє адресувати до 1 Мбайт пам'яті.

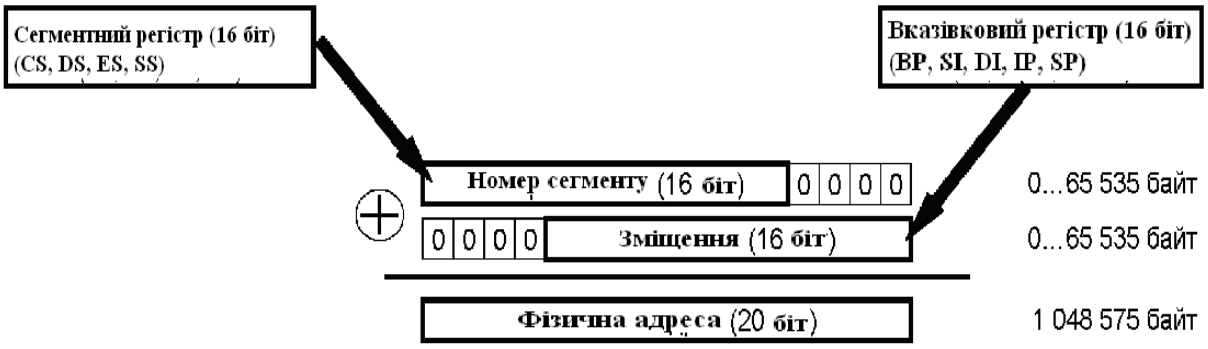

Рисунок 3.1 – Реальний режим адресації процесора

Таким чином, пам'ять поділяється на сегменти розміром 64 кбайт кожний, що починаються з адреси, кратної 16 (межа параграфу); пам'ять в 1 Мбайт поділялася, таким чином, на 16 сегментів. Ці 16 сегментів називають сторінками пам'яті. У комп'ютері, подібному IBM PC, останні 6 сторінок (A, B, C, D, E, F) пам'яті (верхня пам'ять – англ. upper memory) використовувалися для відеопам'яті та BIOS-а, це обмежувало пам'ять, доступну користувачу, об'ємом у 640 кбайт (звичайна пам'ять – англ. conventional memory; сторінки 0~9).

На той час такий режим адресації забезпечував безліч переваг: ємкість пам'яті могла складати до 1 Мбайт, хоча команди оперували 16-бітовими адресами; спрощувалося використання окремих областей пам'яті для програми, її даних і стеку; спрощувалося розроблення пристроїв, сумісних один з одним.

Система команд процесора i8086 складається з 98 команд (більше 3800 їхніх варіацій): 19 команд передачі даних, 38 команд обробки, 24 команди переходу та 17 команд управління процесором. Мікропроцесор не містив команд для роботи з числами з рухомого комою. Така можливість реалізовувалася окремою мікросхемою, так званим математичним співпроцесором, який встановлювався на материнській платі. Співпроцесор зовсім не обов'язково був вироблений Intel (модель i8087), приміром, деякі виробники мікросхем, такі як Weitek, випускали більш продуктивні співпроцесори, ніж Intel.

У мікропроцесорі i8086 була використана примітивна форма конвеєрної обробки. Блок інтерфейсу з шиною подавав потік команд до виконавчого пристрою через 6-байтову чергу команд.

Таким чином, вибірка та виконання нових команд могли відбуватися одночасно. Це значно збільшувало пропускну спроможність процесора та позбавляло необхідності прочитувати команди з повільної пам'яті.

Засновник сучасних персональних комп'ютерів IBM PC

Історія створення першого IBM PC (ай-бі-ем пі-сі), що поклав початок сімейству найпоширеніших сучасних персональних комп'ютерів, почалася в липні 1980 року та була завершена 12 серпня 1981 року представленням моделі IBM 5150. Модель коштувала 1565 дол.

Перший IBM PC був розроблений підрозділом IBM у м. Бока-Ратон, штат Флорида, у підрозділі працювало 12 співробітників (для порівняння: штат компанії Microsoft на той час налічував 32 людини).

У той час фірма IBM не надавала великого значення персональним комп'ютерам, тому в IBM PC було використано багато «чужих» компонентів (наприклад, використовувався процесор фірми Intel, а операційна система MS-DOS ліцензіювала Microsoft у Seattle Computer Products) — до цього IBM вважала за краще все робити самостійно. Більш того, всупереч жорсткій політиці IBM в області інтелектуальної власності ані ці компоненти, ані розроблена базова система введення-виведення не були заліцензовані, що дозволило стороннім фірмам, користуючись опублікованими специфікаціями, створити безліч клонів PC і відібрати в IBM левову частку цього ринку, що швидко розширяється.

Конфігурація першого **IBM PC.** Процесор Intel 8086 з частотою 4.77 МГЦ, ємкість ОЗП від 16 до 256 кбайт. Флопі-дисководи ємкістю 160 кбайт треба було отримати за окрему платню в кількості 1 або 2. Вінчестера не було.

Ключові технології:

- системна шина ISA зі стандартними рознімачами, що дозволяло вставляти в комп'ютер різноманітні плати розширення (відео, звукові, мережні та інші адаптери);

- BIOS – набір системних функцій, що дозволяє розробнику ПО абстрагуватися від деталей роботи апаратури та не залежати від конкретної конфігурації системи. (до цього все ПО

розроблялося тільки під конкретні машини та поставлялося разом з ними);

- в IBM PC можна було використовувати або монохромний відеоадаптер MDA (текст 80x25, розмір символу 9x14), або кольоровий відеоадаптер CGA (текст 80x25 або 40x25, розмір символу 8x8, або графіка 320x200/4 чи 640x200/2 кольори). Причому можна було навіть вставити обидва адаптери та підключити відразу два монітори, монохромний і кольоровий.

Вплив на подальший розвиток ПК

Процесор Intel 8088, що використали в IBM PC, у той час розглядався як тимчасове рішення в області 16-бітових процесорів, оскільки розроблення нового процесора вимагала багато часу. Операційну систему MS-DOS також вважали «тимчасовою»: під наступний процесор передбачалося створити вже операційну систему — OS/2.

Архітектура IBM PC прижилася настільки, що більшість наступних процесорів використовувалися в режимі сумісності з 8086 («реальний режим»), а MS-DOS надовго стала домінуючою операційною системою на IBM PC. Такому успіху IBM PC сприяла відкрита архітектура та поява величезної кількості клонів. Єдиним компонентом захищеної ліцензії було прошивання BIOS, яку втім все ж таки цілком легально скопіювала Compaq, скориставшись принципом зворотної розробки.

Відмінність між процесорами 8088 та 8086 полягає в ширині зовнішньої шини даних — процесор 8086 пересилає 16-розрядні, а 8088 — 8-розрядні дані. По-перше, на момент випуску IBM PC чип 8088 був значно дешевше, по-друге, використання 8-розрядної шини даних спрощувало апаратуру. 16-розрядний 8086 процесор з'явився вже в IBM PC/XT, хоча розроблений він був майже на рік раніше, ніж 8088.

Технічні характеристики:

- дата анонсу: 8 червня 1978 року;
- тактова частота (МГц): 5 (модель 8086), 8 (модель 8086-2), 10 (модель 8086-1) ;
- розрядність регістрів: 16 біт;
- розрядність шини даних: 16 біт;
- розрядність шини адреси: 20 біт;

- об'єм пам'яті, що адресується: 1 Мбайт;
- кількість транзисторів: 29 000;
- техпроцес (нм): 3000 (3 мкм);
- площа кристалу (мм^2): ~30;
- максимальне тепловиділення: 1,75 Вт;
- напруга живлення: +5 В;
- рознімач: немає (мікросхема припаювалася до плати);
- корпус: 40-контактний керамічний DIP;
- підтримувані технології: 98 інструкцій.

3.6 Intel 80186

Intel 80186 — 16-бітовий мікропроцесор, що випущений компанією Intel у другій половині 1982 року та являє собою вдосконалений варіант мікропроцесора i8086. До складу функцій нового мікропроцесора увійшли функції, які раніше реалізовувалися 10 окремими мікросхемами. Застосовувався 80186, головним чином, у роботі з керуючими додатками та у високоінтелектуальних периферійних адаптерах, наприклад мережних.

До розроблення нового мікропроцесора Intel приступила відразу після виходу процесорів i8086/i8088. Процесори i8086/i8088 вимагали великої кількості мікросхем підтримки. Intel вирішує розробити мікропроцесор, що вже містить на кристалі всі необхідні модулі. Новий процесор включав безліч компонентів, що раніше випускалися у вигляді окремих мікросхем, це дозволило різко скоротити кількість мікросхем у комп'ютері, що зменшило б його вартість. Крім того, була розширена система внутрішніх команд (інструкцій).

Нові технології:

- два контролери прямого доступу до пам'яті (DMA) зі схемами переривань;
- дешифратори адреси (програмовані схеми вибору кристалу);
- триканальний програмований таймер/лічильник;
- генератор синхронізації;
- програмований контролер переривань.

3.7 Intel 80286 – багатозадачність

Intel 80286 (також відомий як i286) — 16-бітовий x86-сумісний мікропроцесор другого покоління фірми Intel, випущений 1 лютого 1982 року. Процесор – вдосконалений варіант процесора Intel 8086, але в 3–6 раз швидше за нього. Процесор застосовувався, в основному, в IBM PC сумісних ПК.

Процесор i286 розроблявся одночасно з процесорами Intel 80186/80188, проте в ньому були відсутні деякі модулі, що були в процесорі Intel 80186. Процесор i286 випускався в точно такому корпусі, як i80186 — LCC, а також у корпусах типу PGA з 68 виводами. У новому процесорі було збільшено кількість регістрів, добавлено нові інструкції, добавлено новий режим роботи процесора — захищений режим. Процесор мав 6-байтову чергу (як Intel 8086). Шини адреси і даних тепер не мультиплексуються (тобто адреси і дані передаються по різних ніжках). Шина адреси збільшена до 24 біт, таким чином, об'єм ОЗУ може складати 16 Мбайт. Процесор не міг виконувати операцію над числами з рухомою комою, для виконання таких операцій був необхідний математичний співпроцесор, наприклад Intel 80287.

Регістри. До 14 регістрів процесора Intel 8086 було додано 11 нових регістрів, необхідних для реалізації захищеного режиму та інших функцій: регістр – слово стану машини –16 біт (MSW); регістр задач, 16 біт (TR); регістри дескрипторної таблиці, один 64-бітовий і два 40-бітових (GDTR, IDTR, LDTR), 6 регістрів розширення сегментних регістрів - 48 біт.

Інструкції. Процесор мав той самий набір інструкцій, що процесор Intel 80186, до якого додали 16 нових команд (LGDT, LIDT, LLDT, LMSW, LTR, SGDT, SIDT, SLDT, SMSW, STR, ARPL, CLTS, LAR, LSL, VERR, VERW), необхідних для роботи з засобами управління пам'яттю. Команда PUSH тепер могла зберігати в стеку константи. Інструкції в i286 виконуються в середньому за 4,5 такту.

Режими роботи процесора 80286. У процесорі i286 було реалізовано два режими роботи — захищений і реальний. У реальному режимі роботи процесор був повністю сумісний з процесорами x86, що були випущені до цього, тобто процесор міг

виконувати програми, призначені для Intel 8086/8088/80186, без повторного асемблювання або переасемблювання з мінімальними модифікаціями. У формуванні адреси брали участь тільки 20 ліній шини даних, тому максимальний об'єм пам'яті, що адресується, у цьому режимі залишився колишнім — 1 Мбайт. У захищеному режимі процесор міг адресувати до 1 Гбайт віртуальної пам'яті (при цьому об'єм реальної пам'яті складав не більше 16 Мбайт) за рахунок зміни механізму адресації пам'яті (рисунок 3.2). Перемикання з реального режиму в захищений відбувається програмно і відносно просто, проте для зворотного переходу необхідне апаратне скидання процесора, який в IBM PC-сумісних машинах здійснювався звичайно за допомогою контролера клавіатури. Для відстеження поточного режиму роботи процесора використовується регістр – слово стану машини (MSW). Програми реального режиму без модифікацій у захищеному режимі виконуватися не можуть, так само як і програми BIOS машини.

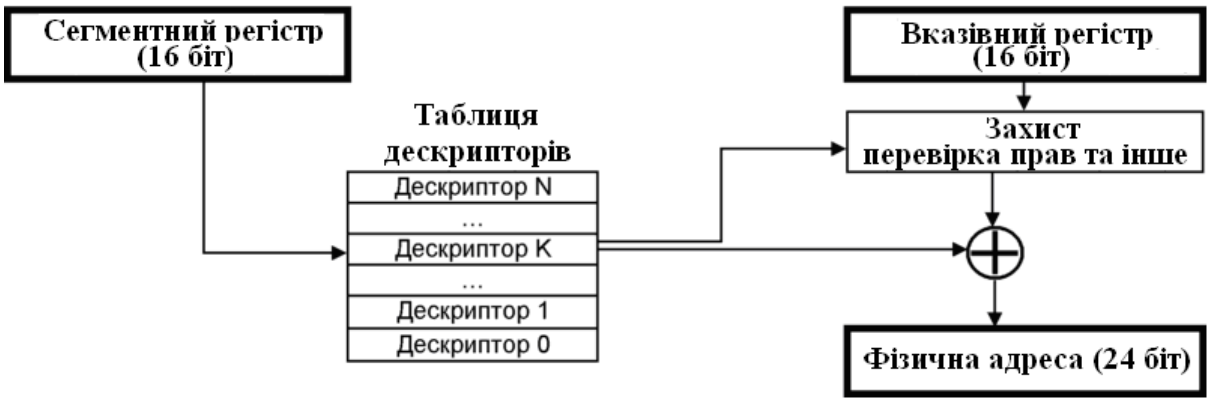

Рисунок 3.2 – Захищений режим адресації процесора 80286

Суть захищеного режиму полягає в такому. Програміст і програми, що розробляються ним, використовують логічний адресний простір (віртуальний адресний простір), розмір якого може складати 1024 Мбайт (для i286). Логічна адреса перетвориться на фізичну адресу автоматично за допомогою схеми управління пам’яттю (MMU). Завдяки захищеному режиму в пам'яті можна зберігати тільки ту частину програми, яка необхідна в даний момент, а інша частина могла зберігатися в зовнішній пам'яті (наприклад на жорсткому диску). У разі

звернення до тієї частини програми, якої немає в пам'яті в даний момент, операційна система може припинити програму, завантажити необхідну секцію коду з зовнішньої пам'яті та відновити виконання програми. Отже, стають допустимими програми, розмір яких більше об'єму пам'яті, що є на комп’ютері. Іншими словами, користувачу здається, що він працює з більшою пам'яттю, ніж насправді. Проте реалізація системи віртуальної пам'яті була ще далекою від досконалості. Для використання захищеного режиму необхідна багатозадачна операційна система, наприклад Microsoft Windows 3.0, IBM OS/2 або UNIX.

Фізична адреса формується таким чином. У сегментних регістрах зберігається селектор, що містить індекс дескриптора в таблиці дескрипторів (13 біт), 1 біт,що визначає, до якої таблиці дескрипторів буде проводитися звертання (до локальної або до глобальної) і 2 біт запрошуваного рівня привілеїв. Далі відбувається звернення до відповідної таблиці дескрипторів і відповідного дескриптора, який містив початковий 24-бітовий адрес сегмента, розмір сегмента та права доступу. Після цього обчислялася необхідна фізична адреса шляхом складання адреси сегмента з зсувом, зберігається у 16-розрядному вказівному регістрі.

Захищений режим у процесорі Intel 80286 мав деякі недоліки: несумісність з програмами, написаними для реального режиму MS-DOS: для переходу з захищеного режиму в реальний режим було потрібне апаратне скидання процесора.

Рівні захисту. Для захисту від виконання привілейованих команд, які можуть кардинально змінити стан всієї системи, для захисту доступу до даних і для захисту сегментів коду у процесорі i286 було введено захист по привілеях. Було виділено 4 рівні привілеїв, так званих кілець (Ring) захисту — від якнайбільш привілейованого 0 рівня (Ring 0), що призначено для ядра системи, до якнайменш привілейованого 3 рівня (Ring 3), що призначено для прикладних програм.

Комп’ютери на базі процесора 80286. У 1984 році фірма IBM представила свій ПК, заснований на процесорі i286 з частотою 6 МГц — IBM PC AT, який викликав великий інтерес до архітектури x86 взагалі та до ПК IBM PC зокрема. У 1987 році IBM випускає нові моделі ПК — IBM PS/2-50 та IBM PS/2-60. До

моменту випуску ПК IBM PS/2 IBM вже не монополіст ринку персональних комп'ютерів, багато фірм випускали аналогічні моделі, що були дешевшими.

Технічні характеристики:

- дата анонсу: 1 лютого 1982 року;
- тактова частота (МГц): 6, 8, 10, 12, 16, 20 ;
- розрядність регістрів: 16 біт;
- розрядність шини даних: 16 біт;
- розрядність шини адреси: 24 біт;
- об'єм пам'яті, що адресується: 16 Мбайт;
- об'єм віртуальної пам'яті: 1 Гбайт;
- кількість транзисторів: 134 000;
- техпроцес (нм): 1500 (1,5 мкм);
- напруга живлення: +5 В.

3.8 Intel 80386 – перший 32- розрядний

Intel 80386 (відомий як *i386* або просто *386*) — 32-бітовий x86-сумісний процесор третього покоління фірми Intel, випущений 17 жовтня 1985 року. Даний процесор був першим 32-розрядним процесором для IBM PC — сумісних ПК. Застосовувався, переважно, у настільних ПК і портативних ПК (ноутбуки).

Архітектура мікропроцесора Intel 80386. Процесор i386 повністю сумісний зі своїми попередниками — процесорами 8086-80286. Він виконує програми, призначені для них, без необхідності модифікації коду та перекомпіляції (або з мінімальними модифікаціями):

- затрачує на виконання меншу кількість тактів синхронізації;
- має більш високі тактові частоти за рахунок використання нових технологій;
- має збільшений порівняно з попередніми процесорами буфер передвибору команд — 16 байт (яких вистачає приблизно на п'ять команд); буфер передвибору забезпечує меншу кількість звернень за командами та виключає зайві звертання до пам'яті в коротких циклах і виконанні рядкових команд;
- першим з процесорів x86 має вбудовану кеш-пам'ять, що

забезпечує меншу кількість звертань у пам'ять при послідовному читанні даних, а також зверненні до одних й тих самих даних.

Разом з тим i386 є серйозною переробкою процесора 80286. За деякими оцінками, ані до, ані після i386 архітектура процесорів x86 жодного разу не перероблялася так кардинально. Сучасні операційні системи називають апаратну платформу PC-сумісних комп'ютерів ніяк не інакше як «i386», тому що не здатні працювати на процесорах нижче за цього.

Винахід процесорів Intel 80386 та Intel 80286 був великим кроком у поліпшенні архітектури та продуктивності процесорів фірми Intel, зараз ці процесори морально застаріли й в основному використовуються у контролерах, а також у побутовій техніці.

Основні нововведення. Всю архітектуру x86 було розширено до 32 біт — всі регістри (за винятком сегментних) встановили 32-бітовими, отримавши в назві префікс «E» (EAX, EBX, EIP, EFLAGS та ін.), із збереженням повного набору команд для роботи з ними. У тому числі:

- регістр прапорів, що отримав безліч нових прапорів для управління багатозадачністю;
- регістр управління MSW процесора 80286, названий в i386 «CR0».

32-бітовою стала адресація в захищеному режимі (з можливістю створення 16-бітових сегментів для сумісності з 80286). Це дозволило вперше з часу появи 8086 забути про *сегментацію*, а точніше обмеження розміру *сегмента* 64 кбайт (обмеження 16-бітової адреси), яке давно перестало влаштовувати програмістів.

До появи i386 програми та операційні системи використовували декілька головоломних *моделей організації пам'яті* (крихітна — tiny, мала — small, велика — large, величезна — huge), що розрізняються за організацією в пам'яті сегментів коду, стеку та даних. 32-бітова адреса дозволила використовувати замість них одну просту *плоску* модель (англ. *flat*) — 32-бітовий варіант *крихітної* моделі, в якій всі сегменти *задачі* знаходяться в одному й тому самому місці адресного простору пам'яті. Плоска модель забезпечує розмір такого «загального» сегмента до 4 Гбайт, яких вистачає для будь-якої мислимої задачі. Плоска модель має й недоліки:

- виникають проблеми *переміщування машинного коду*, які раніше легко вирішувалися сегментацією, забезпечення переміщуваної тепер покладалося на операційні системи з новими форматами дискового образу програми;

- плоска модель, практично, зводить нанівець управління пам'яттю в захищеному режимі (обмеження доступу та підтримка віртуальної пам'яті), яке до i386 могло виконуватися тільки на рівні сегментації.

Тільки поява нової моделі управління пам’яттю – *сторінкового перетворення* – забезпечило плоскій моделі її сьогоднішню популярність. Плоска модель увійшла до побуту так широко, що сучасні програмісти часто не підозрюють, що програми звертаються в пам'ять через сегменти.

Сторінкове перетворення. Захищений режим адресації процесора. У i386 був введений новий механізм управління пам'яттю (рисунок 3.3) — адресний простір, до якого звертається процесор за даними та кодом, в якому розташовуються сегменти (назване лінійним адресним простором), що може не відповідати реальній фізичній пам'яті.

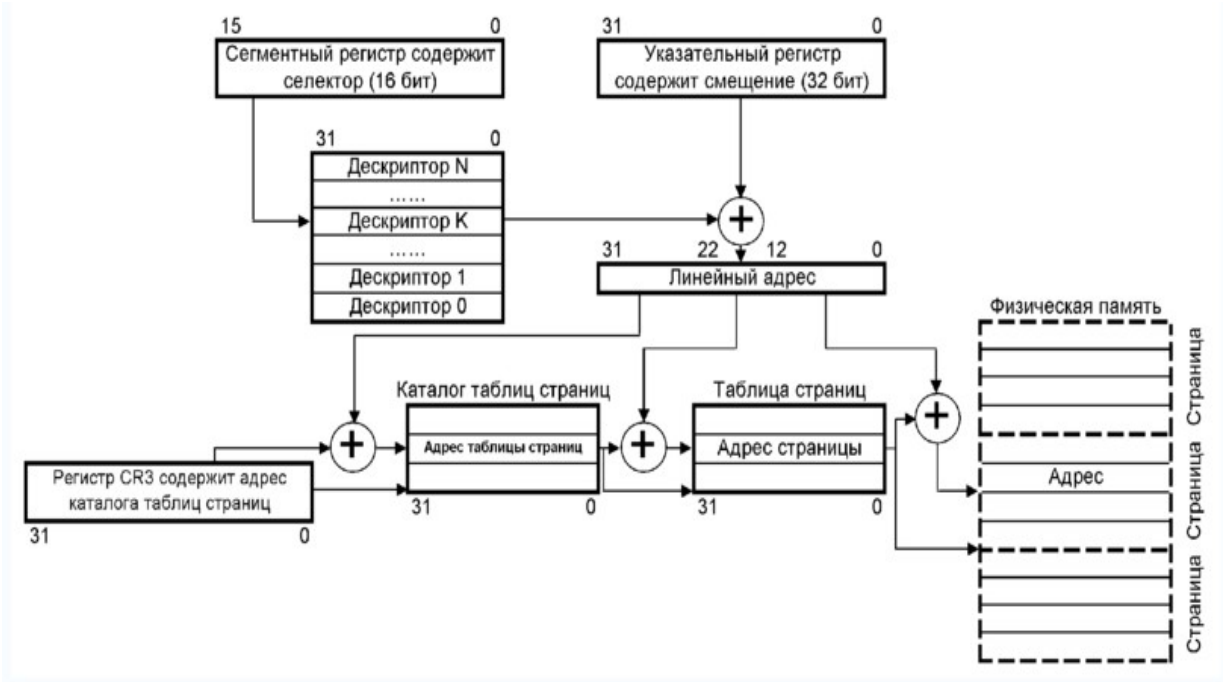

Рисунок 3.3 – Сторінкове перетворення пам’яті 80386

Фізична пам'ять (включаючи буфери зовнішніх пристроїв, наприклад відеобуфер) може відображуватися в лінійний адресний простір довільним образом – кожна сторінка (розміром 4 кбайт) лінійного простору може бути переадресована на будь-яку сторінку фізичної пам'яті через каталог сторінок, розташований в оперативній пам'яті (адреса каталогу сторінок визначається значенням нового регістру управління «CR3»).

Як і сегменти, сторінки лінійного адресного простору можуть бути оголошені неприсутніми (звернення до таких сторінок викликає обробник сторінкового порушення операційної системи). Неприсутні сторінки, передусім, використовуються для організації віртуальної пам’яті — обробник сторінкового порушення здійснює свопінг сторінок пам'яті з зовнішніми запам’ятовуючими пристроями. Також неприсутні сторінки використовуються в плоскій моделі пам'яті (розмір сегмента звичайно має розмір від 2 до 4 Гбайт, навіть якщо в комп'ютера нема стільки фізичної пам'яті) для маркування сторінок сегмента, в які операційна система не виділила пам'яті. У цьому випадку сторінкове порушення завершує задачу або починає її налагодження.

Завдяки сторінковому перетворенню i386 може адресувати до 4 Гбайт фізичної пам'яті та до 64 Тбайт віртуальної пам'яті.

Поліпшена підтримка багатозадачності та захисту. Підтримка багатозадачності у процесорах x86 означає апаратну підтримку «прозорого» перемикання з однієї звичайної програми (задачі) на іншу (Alt-Tab). При перемиканні процесор зберігає свій стан (включаючи адресу наступної команди, селектори сегментів) у сегменті стану (TSS; сегмент пам'яті, з селектором з регістра TR) однієї задачі, після чого відновлює стан іншої задачі з її сегмента стану (селектор сегмента стану нової задачі завантажується з дескриптора її сегмента коду).

Перемикання між задачами звичайно здійснюється:

- перериванням таймера – час, на який настроєний таймер, називається квантом часу для задачі;
- системним викликом (викликом функції операційної системи);
- вимиканням — наприклад, при спробі виконати неприпустиму команду або зверненні до неприсутньої пам'яті;

- налагодженням.

У i386 механізми захисту та багатозадачності були значно розширені та поліпшені. Залежно від характеру порушень вони можуть тихо ігноруватись (наприклад, деякі біти регістра EFLAGS неможна змінити завантаженням прапорів зі стеку), викликати обробника вимикання (операційної системи). Серйозні помилки на рівні операційної системи (або в реальному режимі) можуть довести процесор до режиму аварійної зупинки (наприклад, при виникненні в обробнику подвійного порушення), з якого можна вийти тільки апаратним скиданням (англ. *reset*) процесора.

Крім всього багатозадачність i386 повністю підтримує всі нові можливості — для 32-бітових задач сегмент стану містить все 32-бітов та необхідні нові регістри (наприклад, регістр CR3 с адресою каталогу сторінок для задачі).

*Віртуальний режим.*У процесорі i386 компанія Intel врахувала необхідність кращої підтримки реального режиму, тому що програмне забезпечення до часу його появи не було готове повністю працювати в захищеному режимі. Тому, наприклад, в i386 можливе перемикання з захищеного режиму назад до реального (при розробленні 80286 вважалося, що це не потрібно, тому на комп'ютерах з процесором 80286 повернення до реального режиму здійснюється схемно — через скидання процесора).

Як розширену підтримку реального режиму, i386 дозволяє одній або декільком задачам працювати у віртуальному режимі — режимі емуляції режиму реальної адреси.

Важливо розуміти, що «віртуальний режим», не дивлячись на схожість назви, не є «третім режимом роботи процесора» (тобто реальний, захищений і віртуальний), а лише режимом роботи *задачі* в багатозадачному оточенні захищеного режиму.

Віртуальний режим призначається для одночасного виконання програм реального режиму (наприклад, програми для DOS) під багатозадачною операційною системою захищеного режиму.

Виконання у віртуальному режимі практично ідентично реальному за декількома виключеннями, обумовленими тим, що віртуальна задача виконується в захищеному режимі:

- віртуальна задача не може виконувати привілейовані команди, тому що має найнижчий рівень привілеїв;

- всі переривання та вимикання обробляються операційною системою захищеного режиму (яка може ініціювати обробник переривання віртуальної задачі);

- при виконанні декількох задач віртуального режиму кожна з них може виконуватися цілком окремо одна від одної, чого не можна досягти в реальному режимі.

У задачі віртуального режиму можна використовувати:

- сторінкове перетворення (розширення пам'яті шляхом додавання сторінок у незайнятому адресному просторі, емуляція розширень з перемиканням банків, віртуальною розгорткою або згорткою буферів зовнішніх пристроїв);

- емуляцію зовнішніх пристроїв через емуляцію портів введення-виведення;

- налагодження.

Режим віртуального режиму 8086 підтримується і в подальших 32-бітових процесорах x86, аж до режиму сумісності в x86-64.

*Апаратне налагодження.*Як і в попередніх процесорах (починаючи з 8086), налагодження в i386 здійснюється викликом налагоджувального переривання, обробник якого передає управління програмі-наладнику. У попередніх процесорах налагодження могла бути викликана двома подіями:

- пошагове виконання;

- програмна точка зупинки.

У i386 налагодження, ще можуть почати:

- апаратна точка зупинки;

- пастка перемикання задачі.

Інщі зміни. Лінії даних та адрес у процесорі 80386, як і в процесорі 80286, не є мультиплексованими: є 32 лінії даних і 32 незалежні від них лінії адреси.

Регістри. У складі мікропроцесора є 8 32-битових регістрів загального призначення (EAX, EBX, ECX, EDX, ESI, EDI, EBP, ESP). Молодша, 16-бітова, половина кожного з цих регістрів відповідає регістрам AX, BX, CX, DX, SI, DI, BP, SP у попередніх процесорах сімейства x86. Як і раніше, можлива адресація молодших і старших 8-бітових половин молодшою 16-бітовою

половиною 32-розрядних регістрів даних (AL та AH, BL та BH, CL та CH, DL та DH). Регістр-покажчик команди та регістр прапорів також стали 32-бітовими (EIP та EFLAGS відповідно), причому в регістрі прапорів додано нову групу прапорців. До чотирьох 16-бітових сегментних регістрів (ES, CS, SS, DS) додалися ще два 16-бітових регістри (FS та GS). Додано декілька нових груп регістрів (кожний регістр розміром 32 біт): 3 регістри управління (CR0 (MSW), CR2, CR3), 8 регістрів налагодження (DR0, DR1, DR2, DR3, DR6, DR7), 2 тестові регістри (TR6, TR7).

Набір інструкцій. Набір інструкцій i386 був розширений в основному за рахунок додавання 32-бітових варіантів існуючих інструкцій (утворених 32-бітовими префіксами), включаючи ті, у яких 32-бітові варіанти мають інші мнемоніки (pushad/popad, pushfd/popfd, cwd*/cdq, movsd/cmpsd/scasd/lodsd/insd/outsd, iretd), а також команди mov для нових регістрів, інші нові інструкції.

Середня довжина інструкцій склала 3,2 байт.

Математичний співпроцесор. Спеціально для мікропроцесорів Intel 80386 були розроблені математичні співпроцесори 387SX та 387DX, об'єднувані єдиним найменуванням 80387. Ці співпроцесори сумісні з процесорами 386SX та 386DX відповідно; так, співпроцесор 387DX можливо підключити до 32-розрядної шини процесора. В іншому випадку обидві моделі аналогічні математичному співпроцесору 80287, який також можна використовувати разом з процесором 80386.

Моделі. З 1985 року було випущено безліч модифікацій процесорів i386, що розрізнялись між собою продуктивністю, споживанням потужності, рознімачами, корпусами та іншими характеристиками.

80386DX. Перший процесор сімейства 386, випущений 17 жовтня 1985 року, мав тактову частоту 16 МГц. Після випуску процесорів 386SX процесори цієї серії отримали індекс "DX" — "D"ouble-word e"X"ternal, що вказувало на його 32-розрядну зовнішню шину. Процесор позиціонувався як продуктивне рішення для настільних систем. Проводився процесор по CHMOS IV технології та споживав 400 мА, що значне менше, ніж Intel 8086. 16 лютого 1987 року анонсовано модель з частотою 20 МГц; 4 квітня 1988 року – з частотою 25 МГц, а 10 квітня 1989 року – з частотою 33 МГц. Процесор випускався в корпусах

PQFP-132 (такі процесори мали літери «NG» на початку назви, наприклад, NG80386DX25) або в керамічному PGA-132 (такі процесори мали літеру «A» на початку назви, наприклад, A80386DX25).

Перші процесори 386DX мали помилку, яка іноді призводила до неправильних результатів при роботі з 32-розрядними числами в таких програмах, як OS/2 2.x, UNIX/386 або Windows в розширеному режимі. Помилка призводила до того, що система зависала. Унаслідок деяких проблем, у тому числі за відсутності 32-бітових операційних систем, усунути помилку вдалося лише у квітні 1987 року. Процесори, що вже вийшли, пройшли перевірку, у результаті якої процесори, що не мають помилку, були марковані подвійним символом «сигма» і/або одним символом «IV».

80386SX. Перша модель цього сімейства була представлена 16 червня 1988 року та мала частоту 16 МГц, пізніше були представлені більш швидкі моделі: 20 МГц (25 січня 1989 року), 25 та 33 МГц (обидва процесори представлено 26 жовтня 1992 року). Процесори позиціонували як рішення для настільних ПК початкового рівня та портативних ПК. Сімейство SX відрізнялося від сімейства DX тим, що в нього розрядність зовнішньої шини даних складала 16 біт, а розрядність зовнішньої шини адреси 24 біт. Внаслідок цього процесор міг адресувати тільки 16 Мбайт фізичної пам'яті, що робило його сумісним із старими процесорами (Intel 80286). З іншого боку, процесор 386SX міг виконувати всі програми, написані для 386DX, та навпаки. Це й зробило його популярним для виготовлення «Турбо-плат», наприклад, Cumulus 386SX, Intel INBOARD або Orchid Tiny Turbo. У рамках серії SX були випущені процесори, що марковані 80386SXTA, 80386SXSA, 80386SXLP, які являли собою вбудовані (embedded) процесори (серія SXSA), процесори низького споживання (Low Power), а також використовувалися в інших цілях.

80386SL. Перша модель цього сімейства була представлена 15 жовтня 1990 року і мала частоту 20 МГц, пізніше була представлена модель 25 МГц (30 вересня 1991 року). Процесори позиціонували як перші мікропроцесори, спеціально призначені для портативних ПК. Сімейство SL відрізнялося від сімейства DX

тим, що мало в кристалі також контролери оперативної пам'яті, кеш-пам'яті до 64 кбайт і шини.

80386EX. Являє собою модифікацію процесора 386SX. Процесор призначався для вбудованих додатків з високою інтеграцією та малим споживанням потужності. Ключові особливості цього процесору — низьке енергоспоживання, знижена напруга живлення, розташовані на кристалі контролер переривань, мікросхема вибору чипу, лічильники й таймери, логіка тестування JTAG. Ця серія процесорів мала декілька модифікацій: EXSA, EXTA, EXTB, EXTC. Максимальний струм, споживаний процесорами, складає 320 мА для процесорів серії EXTC і 140 мА для процесорів серії EXTB.

Використовувався на борту різних орбітальних супутників і мікро супутників, а також у NASA проекті FlightLinux.

Комп'ютери на базі процесора Intel 80386. На основі мікропроцесорів 80386 фірмою IBM були створені персональні комп'ютери IBM AT 386 (сімейство PC) та IBM PS/2-80 (сімейство PS/2). У першому застосовувалася ОС PC-DOS, а у другому — OS/2.

Технічні характеристики для всіх процесорів:

- розрядність регістрів: 32 біт;
- об’єм віртуальної пам’яті, що адресується: 64 Тбайт;
- максимальний об’єм пам’яті: 4 Гбайт;
- тактова частота (МГц): від 16 до 40;
- розрядність шини адреси: 32 біт;
- кількість транзисторів: 275 000;
- техпроцес (нм): від 1500 до 800;
- напруга живлення: +5 В.

3.9 Intel 80486 – скалярний (конвеєрний)

Intel486 (також відомий як i486, Intel 80486 або просто 486-ий) – 32-бітовий скалярний x86-сумісний процесор четвертого покоління, побудований на гібридному CISC-RISC ядрі та випущений фірмою Intel 10 квітня 1989 року. Цей мікропроцесор є вдосконаленою версією процесора Intel 80386. Вперше він був продемонстрований на виставці Comdex Fall восени 1989 року. Це був перший мікропроцесор з вбудованим математичним

співпроцесором (FPU). Застосовувався, переважно, у настільних ПК, у високопродуктивних робочих станціях, серверах і портативних ПК (ноутбуки).

На той час Intel вже позбавилася прав власності на товарні знаки x86. Тепер подібні найменування використовували безліч виробників. Основне гасло конкурентів Intel у той час — «Майже те що в Intel, тільки за менші гроші». Тоді загострилася конкурентна боротьба між виробниками процесорів x86. При цьому Intel вирішила відмовитися від найменування процесорів за схемою Intel 80x86 і назвала процесор Intel486, одночасно позмінювали імена на аналогічні інші процесори, що випускали раніше.

У травні 2006 року Intel заявила, що виробництво 80486 припиниться наприкінці вересня 2007 року. І хоча для прикладних програм на персональних комп'ютерах цей чип вже довгий час був застарілим, Intel продовжувала випускати його для використання у виробництві.

Процесор базується на тій самій архітектурі, що застосовувалася, проте в ньому було декілька значущих удосконалень. Основні з яких:

- внутрішній КЕШ першого рівня;
- вбудований математичний співпроцесор;
- конвеєрна обробка інструкцій;
- вдосконалений модуль інтерфейсу шини (bus interface unit);
- укорочені цикли пам'яті (burst mode);
- використання буферів запису.

Процесор мав 32-бітові шини адреси та даних. Це вимагало наявності пам'яті у вигляді чотирьох 30-контактних або одного 72-контактного модулів SIMM.

КЕШ. Intel486 мав розташовану на кристалі кеш-пам'ять об'ємом 8 Кбайт, пізніше — 16 Кбайт, працюючу на частоті ядра. Наявність кеш дозволило істотно збільшити швидкість виконання операцій мікропроцесором. Спочатку кеш Intel486 працював за принципом крізного запису (англ. write-through, WT), але пізніше, у рамках сімейства Intel486, були випущені моделі з внутрішнім кеш, працюючим за принципом зворотного запису (англ. write-back, WB). Процесор міг використовувати зовнішній кеш, швидкість читання-запису якого була помітно нижче, ніж у

внутрішнього кеш. При цьому внутрішній кеш стали називати кешем першого рівня (Level 1 Cache), а зовнішній кеш, розташований на материнській платі -– кешем другого рівня (Level 2 Cache). Кеш мав 4-канальну набірно-асоціативну архітектуру та працював на рівні фізичних адрес пам'яті.

Проте в результаті використання інтегрованої кеш-пам'яті істотно зросла кількість транзисторів у процесорі, як наслідок, збільшилася площа кристалу. Збільшення кількості транзисторів призвело до істотного збільшення розсіюваної потужності. У середньому розсіювана потужність збільшилася у 2 рази порівняно з аналогічними моделями серії Intel386. Багато в чому цьому сприяла інтеграція кеш-пам'яті, хоча були інші чинники, але вони не так суттєві. З цієї причини процесори Intel486 старших моделей вже вимагали примусового (активного) охолоджування.

Математичний співпроцесор. В Intel486 був використаний вбудований математичний співпроцесор (англ. Floating Point Unit, FPU). Взагалі це був перший мікропроцесор сімейства x86 з вбудованим FPU. Вбудований FPU був програмно сумісним з мікросхемою Intel 80387 — математичним співпроцесором, що застосовувався в системах з процесором Intel386. Завдяки використанню вбудованого співпроцесора здешевлювалася та швидшала система за рахунок зменшення загальної кількості контактів і корпусів мікросхем.

Спочатку всі мікропроцесори Intel486, що випускалися, оснащувалися працюючим співпроцесором, ці процесори отримали ім'я Intel486DX. Пізніше, у 1991 році, Intel вирішує випустити процесори з відключеним співпроцесором, ці процесори отримали назву Intel486SX. Системи побудовані на цих процесорах могли оснащуватися окремим співпроцесором, наприклад Intel487SX, або співпроцесором інших виробників.

Конвеєрна обробка інструкцій. В Intel486 був вдосконалений механізм виконання інструкцій у декілька етапів. Конвеєр процесорів серії Intel486 складався з 5 рівнів: вибір інструкції, декодування інструкції, декодування адрес операндів інструкції, виконання команди, запис результату виконання інструкції. Використання конвеєра дозволило під час виконання однієї інструкції проводити підготовчі операції над іншою

інструкцією. Це дозволило значно збільшити продуктивність процесора.

Регістри та інструкції. У процесорі є той самий набір інструкцій, що в Intel386, до якого було додано декілька додаткових регістрів, а саме три 32-битових тестових регістри (TR5, TR4, TR3). Також були додані нові прапори в регістрі прапорів (EFLAGS) і в інших регістрах управління (CR0, CR3).

Унаслідок включення математичного співпроцесора у кристал процесора в Intel486 можна звертатися до регістрів FPU: регістри даних, регістр тегів, регістр стану, покажчики команд і даних FPU, регістр управління FPU.

Набір інструкцій не зазнав істотних змін, але були додані додаткові інструкції для роботи з внутрішньою кеш-пам'яттю (INVD, INVLPG, WBINVD), одна інструкція (BSWAP) для забезпечення сумісності з процесорами Motorola, дві інструкції для атомарних операцій з пам'яттю: CMPXCHG (для порівняння з обміном – нове значення записувалося тільки тоді, якщо старе співпадало з заданим, старе запам'ятовувалося) і XADD (інструкція для складання двох операндів з переміщенням результату в другий операнд, а не в перший, як в ADD). Інструкція CPUID дозволяла вперше в сімействі x86 напряму отримати детальну інформацію про версію та властивості процесора. Крім того, до набору інструкцій додалися 75 інструкцій FPU. Довжина черги інструкцій була збільшена до 32 байт.

Моделі. З моменту появи першого процесора Intel486DX було випущено безліч інших моделей сімейства 486 з суфіксами DX (вбудований математичний співпроцесор), SX (без математичного співпроцесора), SL (знижене енергоспоживання), DX2 (має подвоєну тактову частоту відносно зовнішньої шини), DX4 (має потрійну тактову частоту відносно зовнішньої шини). Вони відрізнялися функціональним призначенням та деякими технологічними параметрами (напруга живлення, тактова частота, розмір кеш-пам'яті, відсутність або наявність співпроцесора та інше), але всі були побудовані на одній архітектурі.

Процесори з індексом DX2 мали коефіцієнт множення 2, тобто, наприклад, при частоті системної шини 33 МГц робоча

частота самого процесора складала 66 МГц. Пізніше з'явилися процесори з індексом DX4 — проте коефіцієнт множення у них був не 4, а 3. Вже після відходу з масового ринку 486-процесорів виробництва Intel компанія AMD випустила процесори 486DX4-120 та 486DX4-133 (останній використовувався переважно в портативних системах). У результаті введення множників у широкий побут вперше ввійшло таке поняття, як розгін (англ. overclocking) — підвищення продуктивності процесора шляхом збільшення тактової частоти шини або коефіцієнта множення. Так, відомо, що в Росії навіть у відкритий продаж поступали системи, в яких процесори i486 працювали на частотах до 160 МГц.

Конкурентні рішення. 486-сумісні процесори виробляли й інші компанії, такі як IBM, Texas Instruments, AMD, Cyrix, UMC і Chips and Technologies. Деякі з них були майже точними копіями, як за продуктивністю, так і технічними характеристиками, інші ж, навпаки, відрізнялися від оригіналу.

Платформа 486. Спочатку системи на базі Intel486 були обладнані тільки 8- і/або 16-битовими шинами ISA. Пізніше материнська плата суміщала в собі повільну шину ISA з високошвидкісною шиною VESA (або VLB — англ. Vesa Local Bus), що призначалася перш за все для відеоплат і контролерів жорсткого диска. Остання материнська плата для процесорів i486 була обладнана шинами PCI та ISA, а іноді і VESA. Швидкодія шини ISA визначалася множниками, а робоча частота шин PCI та VLB дорівнювалася частоті шини процесора i486 (хоча деякі материнські плати мали множники також і для них).

Пізніше материнські плати для i486 знайшли підтримку технології PLUG-AND-PLAY, яка використовувалася у Windows 95, що дозволяла комп'ютерам автоматично виявляти та настроювати пристрої, встановлювані на комп'ютер, і встановлювати відповідні драйвери.

Спочатку процесор позиціонували як процесор для настільних ПК і серверів. Intel також випускала ПК, побудовані на процесорі Intel486DX, наприклад MICROCOMPUTER Model 401, MICROSYSTEM Series 4000 (робоча станція) та ін. ПК на базі Intel486 випускала й IBM: IBM AT 486, IBM PS/2-90/95. Пізніше, після виходу більш швидких процесорів (Intel486DX2, Intel486DX4, Pentium та ін.), процесор

застосовували в іншій техніці або, наприклад, як контролер у мережній платі.

Технічні характеристики для всіх процесорів:

- кодове ім’я: P4;
- розрядність регістрів: 32 біт;
- об’єм віртуальної пам’яті: 64 Тбайт;
- максимальний об’єм пам’яті: 4 Гбайт;
- тактова частота (МГц): від 22 до 50;
- розрядність шини адреси: 32 біт;
- кеш L1: 8 Кбайт;
- кількість транзисторів: 1,185 млн;
- техпроцес (нм): від 1000 до 800;
- напруга живлення: +3,3В, +5В;
- розсіювана потужність: 5Вт.

3.10 Pentium – суперскалярний

Pentium (вимовляється *пентіум*) — торгова марка декількох поколінь мікропроцесорів сімейства x86, що випускаються корпорацією Intel з 22 березня 1993 року. Pentium є процесором Intel п'ятого покоління та прийшов на зміну Intel 80486.

Історія. У червні 1989 року Вінодом Демом були зроблені перші напрацювання процесора під кодовою назвою P5. Він не підозрював, що саме цей продукт буде одним з головних досягнень корпорації Intel. Наприкінці 1991 року було завершено розроблення макета процесора. Інженери змогли запустити на ньому програмне забезпечення. Почався етап оптимізації топології та підвищення ефективності роботи. У лютому 1992 року проектування в основному було завершено, почалося всебічне тестування дослідної партії процесорів. У квітні 1992 року ухвалено рішення про початок промислового виробництва процесорів. Як основна промислова база була вибрана Орегонська фабрика № 5. Почалося промислове освоєння виробництва та остаточне доведення технічних характеристик. У жовтні 1992 року Intel оголосила, що процесори п'ятого покоління, що раніше носили кодове ім'я P5, будуть називатися Pentium, а не 80586, як передбачали. Це викликано

тим, що багато фірм, виробляючих процесори, активно освоїли виробництво «клонів» процесорів 386 та 486. Intel збиралася зареєструвати як торгову марку назву «586», щоб більше ніхто не зміг займатися виробництвом процесорів з такою назвою. Проте виявилося, що зареєструвати цифри як торгову марку не можна. Тому було ухвалено рішення назвати нові процесори «Pentium» (від «*pent-*» – п'ять), що також вказувало на покоління даного процесора. 22 березня 1993 року відбулася презентація нового мікропроцесора, через декілька місяців з'явилися і перші комп'ютери на їхній основі.

Альтернативна історія. Історія про Вінода Дема є офіційною версією Intel, проте є інша версія появи цього процесора. У 1980-х роках у СРСР над процесором «Эльбрус» працював Володимир Мстиславович Пентковський, який працював при РАН під керівництвом Бабаяна. Приблизно у 1989 році делегація від Intel відвідала лабораторії Обчислювальної техніки РАН. Пентковський отримав запрошення приїхати по обміну досвідом до США у дослідницький центр Intel. З цієї поїздки до СРСР Пентковський не повернувся, а через декілька місяців Intel офіційно заявила про розроблення принципово нового процесора під назвою Pentium (названий на честь розробника).

Основні відмінності від 486-го процесора:

- суперскалярна архітектура. Завдяки використанню суперскалярної архітектури процесор може виконувати 2 команди за 1 такт. Така можливість існує завдяки наявності двох конвеєрів – u- і v-конвеєр. u-конвеєр — основний, виконує всі операції над цілими та дійсними числами; v-конвеєр — допоміжний, виконує тільки прості операції над цілими та частково над речовинними. Щоб старі програми (для 486) повною мірою використовували можливості такої архітектури, необхідно було їх перекомпілювати. Pentium є першим CISC процесором, що використовує багатоконвеєрну архітектуру;

- 64-бітова шина даних. Дозволяє процесору Pentium обмінюватися вдвічі більшим об'ємом даних з оперативною пам'яттю, ніж 486 за один шинний цикл (при однаковій тактовій частоті);

- механізм прогнозу адрес галуження. Застосовується для скорочення часу простою конвеєрів, викликаного затримками вибору команд при зміні лічильника адреси під час виконання команд галуження. Для цього в процесорі використовується буфер адреси галуження BTB (Branch Target Buffer), що використовує алгоритми прогнозу адрес галуження;

- роздільний кеш програмного коду та даних. У процесорах Pentium використовується кеш-пам'ять першого рівня (кеш L1) об'ємом 16 кбайт, розділена на 2 сегмента: 8 кбайт для даних і 8 кбайт для інструкцій. Це поліпшує продуктивність та дозволяє робити подвійне кешування доступним частіше, ніж це було можливо раніше. Крім того, змінено механізм кеш;

- покращено блок обчислень з рухомого комою (FPU, співпроцесор);

- симетрична багатопроцесорна робота (SMP).

Моделі. 22 березня 1993 року було представлено тільки дві моделі, засновані на ядрі P5 з частотами 60 та 66 МГц. Проте пізніше були випущені більш продуктивні процесори Pentium, але засновані на вдосконалених ядрах. Крім того, були представлені мобільні версії процесорів і процесори Pentium Overdrive.

P5. Процесори Pentium першого покоління. Дві (єдині) моделі було анонсовано 23 березня 1993 року, працювали з тактовою частотою ядра 60 і 66 МГц, частота системної шини (FSB) дорівнювала частоті ядра, тобто множник ядра «1,0». Кеш другого рівня розміщувався на материнській платі та міг мати розмір до 1 Мбайт. Процесор випускався у 273-контактному корпусі CPGA та встановлювався в корпус Socket 4 і працював від напруги 5 В. Всі процесори Pentium належать до класу SL Enhanced. Це означає, що в них передбачена система SMM, що забезпечує зниження енергоспоживання. Ранні варіанти процесорів з частотами 60—100 МГц (ядра P5 та P54C) мали помилку в модулі FPU (математичний співпроцесор), яка в рідких випадках призводила до зменшення точності операції розподілу. Цей дефект був виявлений у Лінчберге (США, штат Вірджинія) у 1994 році і став відомий як «Pentium FDIV баг». Процесори на ядрі P5 виготовлялися з використанням 800 нанометрового техпроцесу за біполярною BiCMOS-технологією. Процесор

містить 3,1 млн транзисторів, а розмір кристалу ядра складає 294 мм^2. Процесор Pentium 66 споживає струм у 3,2 А, а споживана ним потужність дорівнює 16 Вт, що потребувало установлення додаткового вентилятора. Виробництво таких процесорів було дуже складним, відсоток виходу придатних кристалів виявився дуже малим. Багато фахівців вказували на численні недоліки процесорів Pentium першого покоління, не радили купувати ці моделі. Виробництво на якийсь час довелося зупинити. Проте незабаром почалося виробництво вдосконалених процесорів, заснованих на ядрі P54C.

P54C. 7 березня 1994 року були випущені процесори Pentium другого покоління. Спочатку були випущені моделі з тактовими частотами 90 та 100 МГц, проте потім була випущена модель з частотою 75 МГц. Процесори створювали за 600 нанометровою біполярною BiCMOS-технологією, що дозволило зменшити розмір кристалу до 148 мм^2 (ядро містило 3,2 млн транзисторів) і знизити споживану потужність до 10,1 Вт (для Pentium 100). Напруга живлення також була зменшена до 3,3 В, струм, споживаний процесором, складав 3,25 А. Процесор випускався у 296-контактному корпусі CPGA та встановлювався в Socket 5 або Socket 7, але був не сумісний з Socket 4. У цих процесорах поліпшена система SMM, додано вдосконалений програмований контролер переривань APIC. У процесорах Pentium другого покоління використовувався множення тактової частоти, вони працювали швидше за системну шину. Для зазначення, у скільки разів тактова частота ядра процесора більше частоти системної шини, використовувався множник. У всіх процесорах, заснованих на ядрі P54C, множник дорівнював 1,5.

P54CS. Перші процесори, засновані на даному ядрі, були випущені 27 березня 1995 року. За великим рахунком, це ядро являє собою ядро P54C, виготовлене з використанням 350 нанометрової біполярної BiCMOS-технології, що дозволило ще зменшити розмір кристалу ядра до 91 мм^2 (процесори Pentium 120 та 133), проте незабаром у результаті оптимізації ядра його розмір вдалося зменшити до 83 мм^2 при тій самій кількості транзисторів. При цьому Pentium 200 споживав струм у 4,6 А, а його максимальна розсіювана енергія (тепловиділення) складала 15,5 Вт.

P55C. 8 січня 1997 року були випущені процесори Pentium, засновані на ядрі P5 третього покоління (P55C). Центром Розробок та Досліджень Intel у Хайфі (Ізраїль) у ядро P55C було додано новий набір інструкцій, названий MMX, істотно збільшуючий продуктивність комп'ютера в мультимедіа-додатках (від 10 до 60 % залежно від оптимізації). Внаслідок цього нові процесори тепер називаються Pentium MMX technology (звичайно скорочується до Pentium MMX). Новий процесор включає пристрій MMX з конвеєрною обробкою команд, кеш L1 збільшений до 32 кбайт (16 кбайт для даних і 16 кбайт для інструкцій). Складався новий процесор з 4,5 млн транзисторів і вироблявся з вдосконаленою 350-нанометровою CMOS-технологією з використанням кремнієвих напівпровідників, працював на напрузі 2,8 В. Максимальний споживаний струм 6,5 А, а тепловиділення 17 Вт (для Pentium 233 MMX). Площа кристалу у процесорів Pentium MMX 141 мм^2. Процесори випускалися у 296-контактному корпусі типу CPGA або PPGA для Socket 7.

Pentium OverDrive. Було вироблено декілька поколінь Pentium OverDrive. У 1995 році вийшов перший Pentium OVERDRIVE (використовувалося ядро P24T). Він був призначений для установлення у гніздо типу Socket 2 або Socket 3 і працював з напругою живлення 5 В, тобто служив для модернізації систем, що використовували процесор 486 без заміни материнської плати. При цьому даний процесор володів всіма функціями процесора Pentium першого покоління (на ядрі P5). Було випущено дві моделі, що працювали на частотах 63 та 83 МГц, старша споживала струм у 2,8 А та розсіювала потужність 14 Вт. Через високу вартість даний процесор зник, не встигнувши з'явитися. І хоча через деякий час (4 березня 1996 року) на зміну цим процесорам прийшли Pentium ODP5V з частотами 120 та 133 МГц, засновані на ядрі P5T (по суті, являє собою ядро P54CS), вони також не стали популярними.

Tillamook. Процесори, засновані на даному ядрі, призначалися для портативних комп'ютерів. Ядро Tillamook (названо на честь міста в штаті Орегон, США) являє собою ядро P55C із зниженою напругою живлення — модель з частотою 300 МГц працювала з напругою 2,0 В, споживаючи при цьому

струм у 4,5 А, і тепловиділенням у 8,4 Вт. Старші моделі (з частотою 233, 266 та 300 МГц) випускалися з використанням 250 нм техпроцесу та мали кристал площею 90 мм^2, пізніше (133 та 150 МГц) з використанням 280 нм техпроцесу та мали кристал площею 140 мм^2, інші випускалися з використанням обох техпроцесів. Залежно від типу корпусу (TCP або CBGA) процесор відповідно або припаювався до материнської плати, або встановлювався у спеціальний модуль MMC1. Перші моделі були випущені в січні 1997 року.

Процесори Intel Pentium користувалися величезною популярністю. Intel вирішила не відмовлятися від марки Pentium, називаючи так наступні процесори, хоча вони сильно відрізнялися від перших Pentium та не належали до п'ятого покоління.

Технічні характеристики для всіх процесорів:

- кодове ім’я: P5;
- розрядність регістрів: 32 біт;
- об’єм віртуальної пам’яті: 64 Тбайт;
- максимальний об’єм пам’яті: 4 Гбайт;
- тактова частота: від 60 до 233 МГц;
- розрядність шини адреси: 32 біт;
- розрядність шини даних: 64 біт;
- кеш L1: 8 кбайт даних + 8 кбайт інструкцій;
- кількість транзисторів: від 3,11 до 4,5 млн;
- техпроцес (нм): від 800 до 350;
- напруга живлення: +2,8В, +3,3В, +5В;
- розсіювана потужність: від 10 до 16 Вт.

СПИСОК ЛІТЕРАТУРИ

1 Гук М.Ю. Аппаратные средства IBM PC: энциклопедия / М.Ю. Гук. – 3-е изд. – СПб.: Питер, 2006. – 1072 с. – ISBN 5-46901182-8.

2 Максимов Н.В. Архитектура ЭВМ и вычислительных систем: учебник / Н.В. Максимов, Т.Л. Партыка, И.И. Попов. – М.: Форум, 2006. – 512 с. – ISBN 5-8199-0160-6.

3 Мюллер С. Модернизация и ремонт ПК: пер. с англ. / С. Мюллер. – М.: Вильямс, 2007. – 1360 с. – ISBN 5-8459-0447-1.

4 Степаненко О.С. Практическая сборка и наладка ПК: пер. с англ. / О.С. Степаненко. – М.: Вильямс, 2007. – 336 с. – ISBN 978-5-8459-1193-3.

5 Томпсон Б. Ремонт и модернизация ПК: пер. с англ. / Б. Томпсон, Р. Томпсон. – М.: BHV, 2007. – 608 с. – ISBN 978-5-94157974-7.

6 Трасковский А.В. *BIOS.* – М.: BHV, 2006. –400 с. – ISBN 5-94157-859-8.

7 Крупник А. Асемблер. Самоучитель. – С.Пб.: Питер, 2005. – 235 с.

ДОДАТОК А

1 Робота мовою Assembler – створення програми

Теоретична підготовка

ЦП влаштований так, що йому необхідно давати команди, які він виконує. Кількість команд, які може виконувати ЦП, невелика, але, змінюючи їхню послідовність, можна створювати майже необмежені програмні продукти. Самі команди, які розуміє ЦП, називають машинними кодами. Програмувати людині в машинних кодах дуже складно. Тому для полегшення праці програмістів були придумані мови програмування. Першою мовою програмування була мова низького рівня Assembler. Її особливість полягає у тому, що кожна команда цієї мови трансформується в один машинний код. Така властивість мови дуже добре дозволяє вивчити особливості комп'ютерної схемотехніки та архітектури ПК. Інші мови програмування для цього не придатні, оскільки не можуть відображати всі особливості комп'ютера.

Структурно мова Assembler складається з компілятора, який автоматично переводить команди, зрозумілі програмісту (інструкції), у команди, зрозумілі комп'ютеру (машинні коди). Таким чином, Assembler є своєрідним посередником між програмістом і комп'ютером. Щоб Assembler міг правильно проводити компіляцію, необхідно ставити спеціальні інструкції, які повідомляють компілятор, що потрібно робити. Такі інструкції називають **директивами**. Після компіляції директиви не входять в основну програму.

У сучасних операційних системах (ОС) величезна кількість функцій (процедур) вже написана програмістами-розробниками операційних систем. Розумно використовувати ці процедури, а не писати їх самостійно. Це скорочує час роботи над програмами. Оскільки програми працюють під управлінням ОС, то для правильного запуску, забезпечення взаємодії з зовнішнім середовищем (виведення на екран, читання та запис з вінчестера та ін.) і правильного завершення необхідні спеціальні процедури ОС – API (Application Programming Interface – Інтерфейс прикладних програм). Кожній процедурі API передаються параметри, тобто відомості, необхідні їй для роботи.

Найпростіша процедура API – EXITPROCESS – процедура завершення програми.

Процедури ОС викликає спеціальна директива – **invoke**.

Приклад

Invoke ExitProcess. 0

Нижче приклад програми.

```
.386
.model  flat, stdcall
includelib \grup\stud\myasm\lib\kernel32.lib
ExitProcess proto :DWORD
.code
    start:
        mov eax, 2
        add eax, 3
        invoke ExitProcess, 0
    end start
```

У даній програмі лише дві реальної команди, які будуть переведені в машинні коди:

1) mov eax, 2; – помістити 2 в регістр eax;

2) add eax, 3; – добавити 3 до значення в регістрі eax.

Інші записи – директиви.

.386 – директива, що вказує на мінімальний тип процесора, з яким повинна працювати програма. Тут слід зазначити, що всі процесори сумісні з більш ранніми версіями.

.model flat, stdcall – дана директива вказує, для якої ОС написана програма. У цьому випадку – Windows.

includelib \grup\stud\myasm\lib\kernel32.lib – директива підключає бібліотеку kernel32.lib – бібліотеку, що містить готові машинні коди, які будуть вставлені в програму при компіляції. Дуже важливо правильно указати шлях до бібліотеки.

EXITPROCESS proto :DWORD – опис використовуваної в програмі процедури для компілятора. Даний рядок означає, що процедура має одну змінну розміром у DWORD (подвійне слово або чотири байти). Assembler перекладе програму мовою машинних кодів, коли параметри, описані директивою proto, відповідають опису процедури в бібліотеці, а також параметрам, указаним при виклику процедури директивою invoke.

.code – директива показує, що далі підуть саме команди

програми, які потрібно буде перевести машинною мовою.

start: – мітка (посилання) початку програми.

end start – директива закінчення програми.

Для роботи з мовою Assembler необхідно використовувати програму цієї мови – **MASM** фірми Microsoft. Дана програма є безкоштовною та може бути скачена з офіційного сайту фірми. Для набору програм, компіляції та відображення результатів їхньої роботи буде використовуватися програма **FAR.**

Практична частина

Інсталювати програму FAR

Для написання програми необхідно створити умови роботи. Необхідно встановити програму для зручної роботи з Assembler. Для установлення програми FAR необхідно володіти правами адміністратора. Необхідно встановити програму, якщо вона ще не встановлена. Якщо програма вже встановлена на комп'ютері, то просто необхідно запустити програму. На рисунку А.1 зображено вікно FAR.

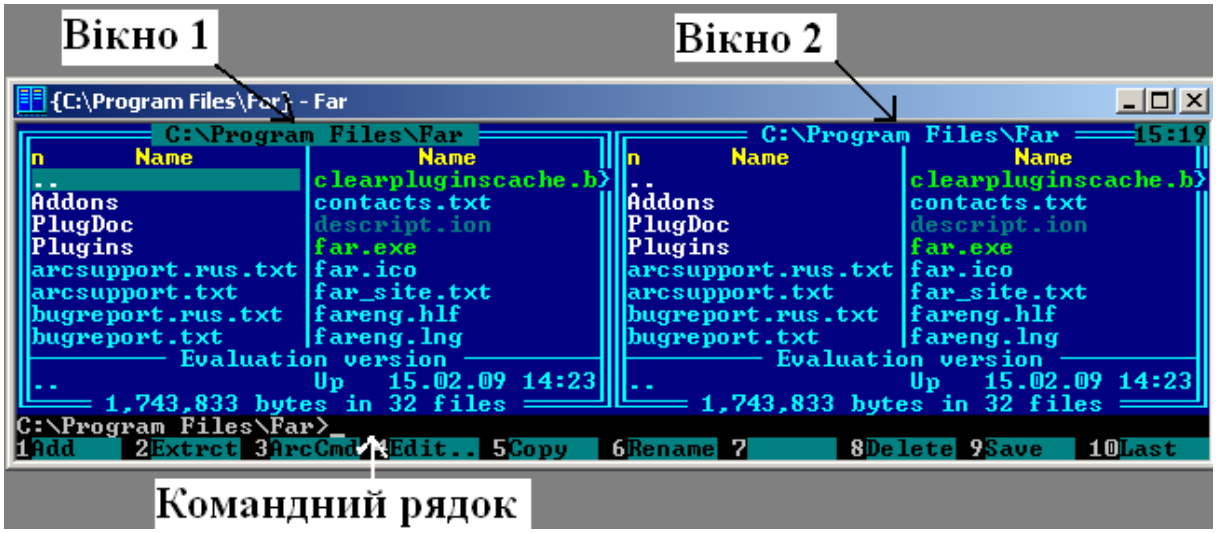

Рисунок А.1 – Зображення вікна програми FAR після запуску

Встановлення компілятора Assembler MASM фірми Microsoft

Для вибору потрібного диска необхідно натиснути комбінацію клавіш Alt+F1 (перше вікно) або Alt+F2 (друге вікно). У вікні (рисунок А.2), що з'явилося, необхідно вибрати диск, на якому буде проводитися робота.

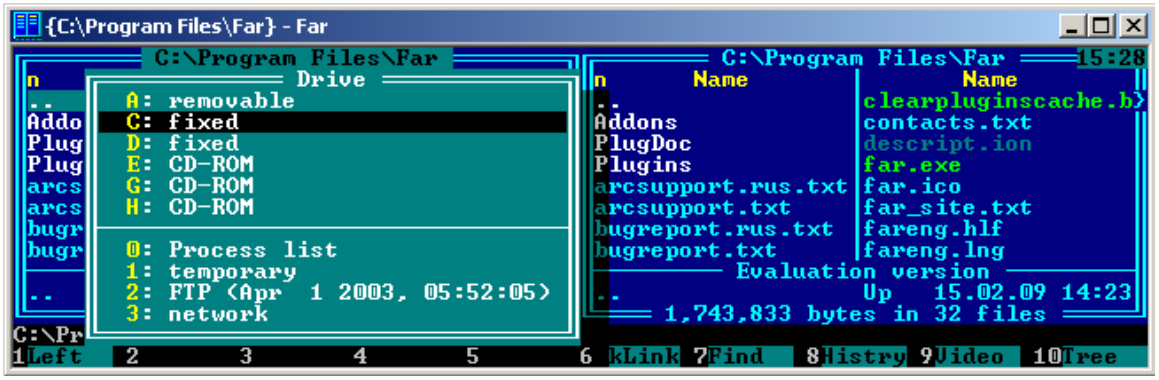

Рисунок А.2 – Зображення вікна програми FAR після натиснення комбінації клавіш Alt+F1

Після вибору диска необхідно створити папку для роботи. Ім'я папки необхідно створити у відповідності з номером групи, а всередині цієї папки створити папку з власним прізвищем. У наведеному вище прикладі-програми stud і grup відповідно. Для створення папки необхідно натиснути клавішу F7 та у вікні, що з'явилося, набрати ім'я папки (рисунок А.3). Після цього натиснути Enter. На диску буде створена папка stud. Необхідно навести на неї курсор і натиснути Enter. Після цього папка відкриється всередині неї аналогічним чином необхідно створити другу папку з власним прізвищем та увійти до цієї папки.

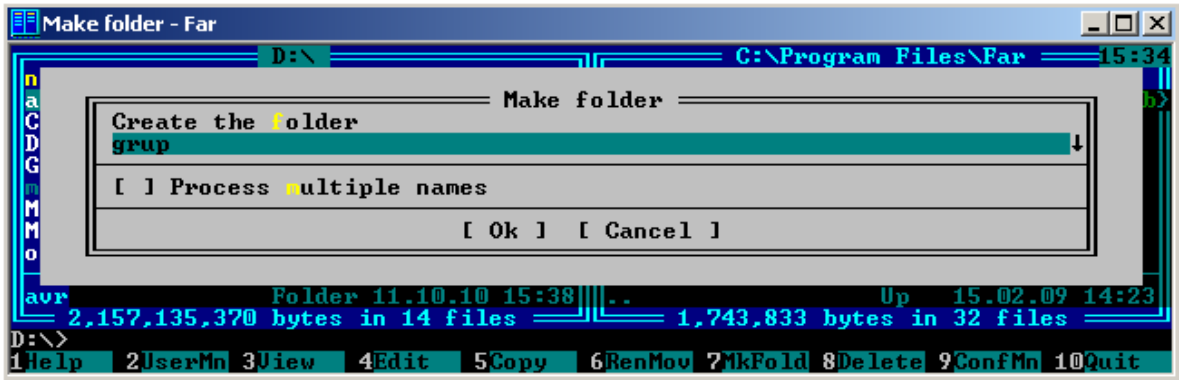

Рисунок А.3 – Зображення вікна програми FAR після натиснення клавіші F7 і введення імені папки

Тепер необхідно скопіювати в цю відкриту папку мову Assembler. Для цього потрібно скопіювати туди папку MYASM зі всім вмістом. Необхідно використовувати друге вікно (перехід по вікнах – клавіша Tab) і знайти цю папку. Потім навести на неї

курсор і натиснути клавішу F5 і далі Enter. Після цього буде здійснено копіювання та в створеній папці опиниться папка MYASM.

Набір програми

Для запропонованої до набору програми, в теоретичній частині, необхідно зайти глибше. Відкрити папку MYASM і далі папку BIN. У цій папці необхідно створити текстовий файл для набору програми. Для створення файла необхідно натиснути комбінацію клавіш Shift+F4. У вікні, що відкрилося, необхідно написати будь-яке ім'я програми, але обов'язково в кінці приписати чотири символи «**.asm**» без лапок (рисунок А.4). Після цього натиснути Enter і в результаті з'явиться вікно редактора для набору програми (рисунок А.5). Необхідно набрати програму та натиснути F2 для збереження тексту програми у файлі. Потім натиснути Esc і вийти з редактора.

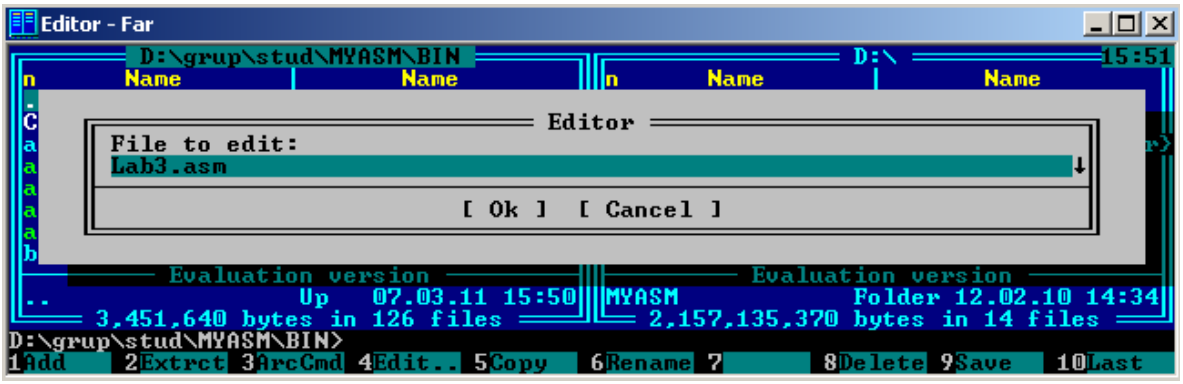

Рисунок А.4 – Приклад створення файла

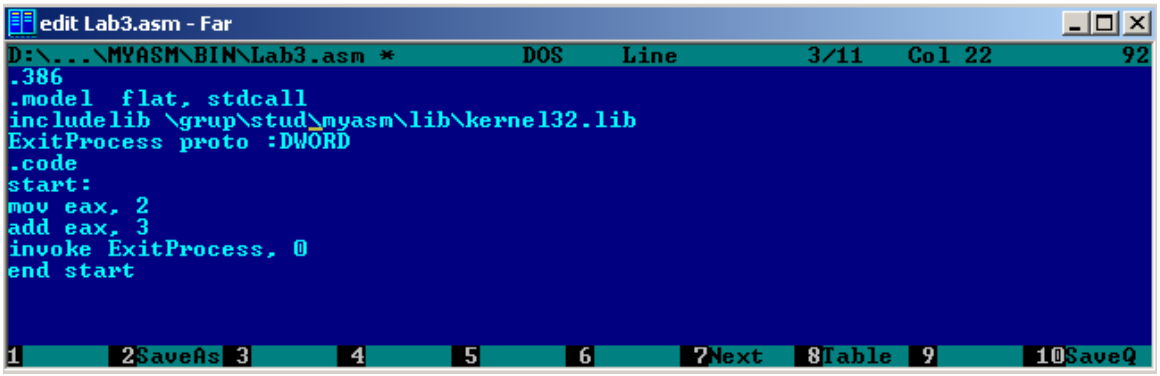

Рисунок А.5 – Набір програми в редакторі FAR

Компіляція програми

Набрати в командному рядку FAR запис «**amake lab3» без лапок** і натиснути ENTER. Після цього, якщо все правильно зроблено, у папці з'являться ще два файли: об'єктний (lab3.obj) і додаток або прикладна програма (lab3.exe). Обидва файли вийшли завдяки набору інструкцій, що перебувають у файлі **amake.bat.**

Якщо цих файлів після компіляції не з'явилося, то необхідно натиснути Ctrl+o. Там можна буде подивитися на помилки, допущені при написанні програми та які не дозволили компілятору створити файли додатку. Приклад наведено на рисунку А.6.

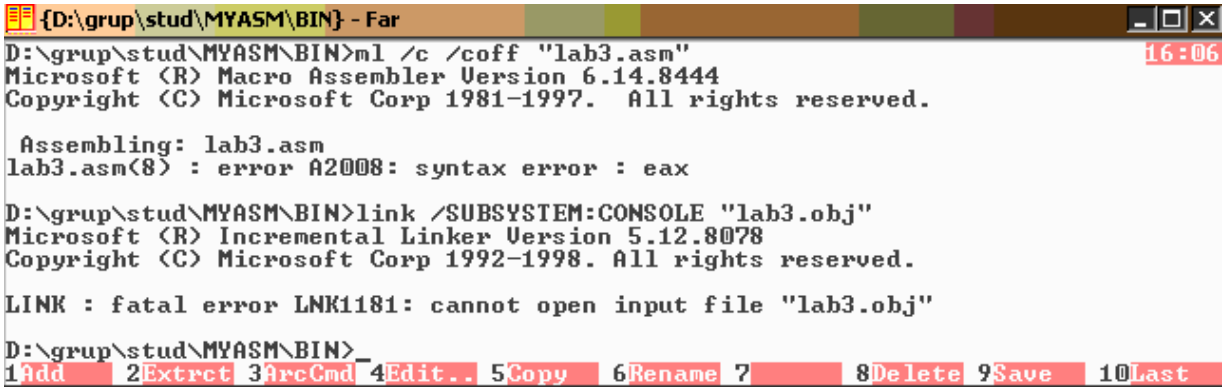

Рисунок А.6 – Приклад помилки, що виявлено компілятором

Необхідно натиснути Ctrl+o (повернення вікна). Увійти до програми (навести на Lab3.asm курсор та натиснути F4). Знайти помилку. Виправити програму та повторити дії з компіляції.

2 Дослідження процедур на Assembler

Теоретична підготовка

Процедура – відособлена частина програми, що виконує певне завдання. Процедури дозволяють спростити програму, зробити її зрозумілою та керованою. Тому досвідчені програмісти повинні уміти скласти програму будь-якої складності з окремих незалежних модулів – процедур (рисунок А.7). Завдяки цьому в разі виникнення помилки її пошук помітно спрощується. Адже спочатку можна визначити процедуру з помилкою, а потім шукати помилку в ній.

Найпростіша програма, що використовує процедуру **AddDigs**, наведену нижче, додає ва числа.

```
.386
.model  flat, stdcall
includelib \myasm\lib\kernel32.lib
ExitProcess proto :DWORD
option casemap:none
.code
start:
mov ax,2
mov bx,3
call AddDigs          ; виклик процедури
invoke ExitProcess, 0
AddDigs proc          ; початок процедури
add ax,bx
ret
AddDigs endp          ; кінець процедури
end start
```

У програмі додалося декілька нових директив і команд.

Директива **option casemap:none –** при її виконанні компілятор почне відрізняти рядкові та прописні букви.

Команда **call AddDigs** викликає процедуру з ім'ям **AddDigs**. Далі **AddDigs proc –** заголовок процедури, а **Adddigs endp** – закінчення процедури.

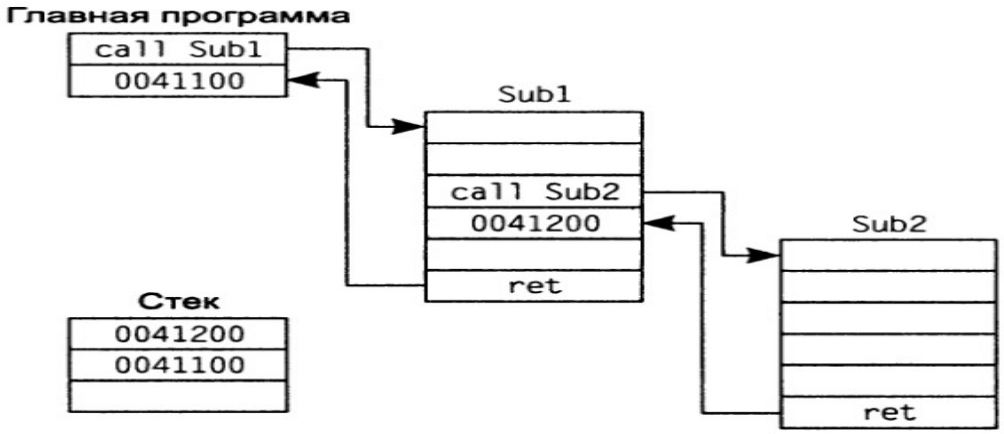

Рисунок А.7 – Структура визову процедур

У програмі OLLYDBG.exe на початку виконання програма матиме вигляд як на рисунку. А8:

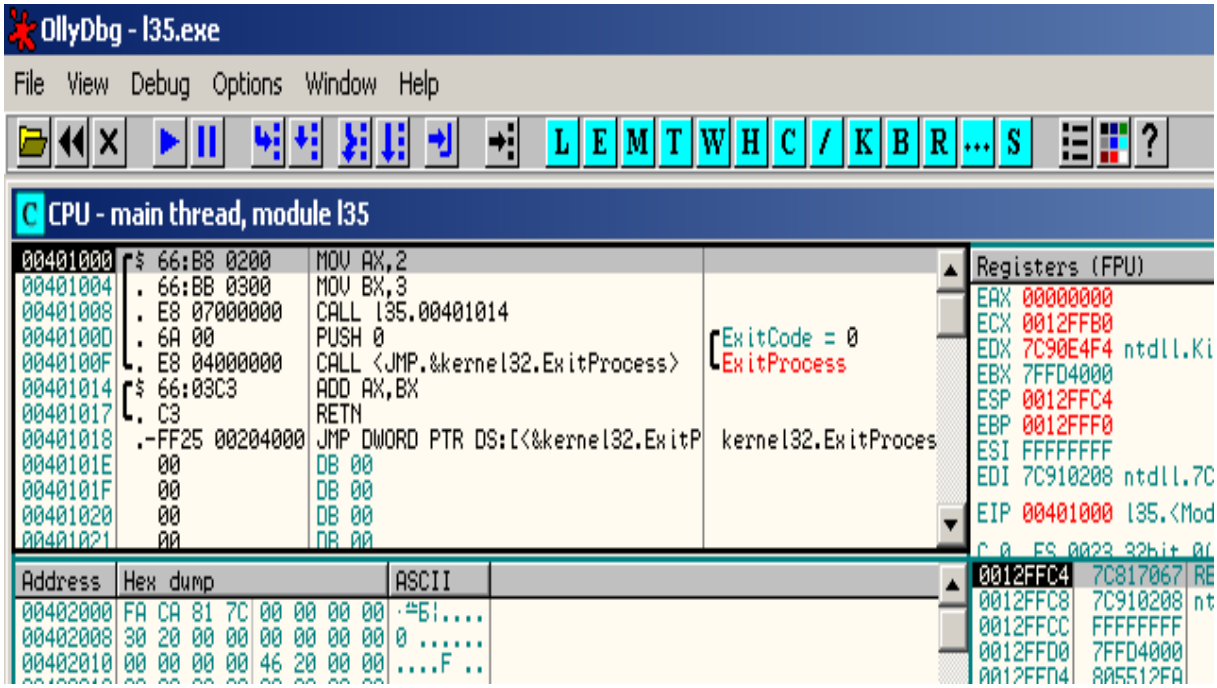

Рисунок А.8 – Вигляд програми та процедури у OLLYDBG.exe на початку виконання програми

ВАЖЛИВО. У момент виклику процедури командою **call** необхідно запам'ятати поточну адресу, куди програма повинна буде повернутися після виконання процедури. Команда **call** перед тим, як перейти до процедури, запам'ятовує адресу в стек, а команда **ret** витягає адресу зі стеку та передає управління тій частині програми, звідки була викликана поточна процедура. Адреса поточної команди зберігається в регістрі **EIP.** Саме в цей регістр завантажується адреса команди, що витягається зі стеку команда **ret.**

Якщо процедур декілька й одна викликається з іншої процедури (рисунок А.7), то адреси в стеку заповнюються та витягаються за звичайним правилом (першим прийшов – останнім пішов).

Процедур може бути незліченна безліч, а для їхнього правильного використання, можливо, знадобляться дані, яких буде більше, ніж регістрів. У цьому випадку дуже зручно

використовувати стек, поміщаючи в нього дані командою **Push** та витягаючи командою **Pop** або за допомогою непрямої адресації.

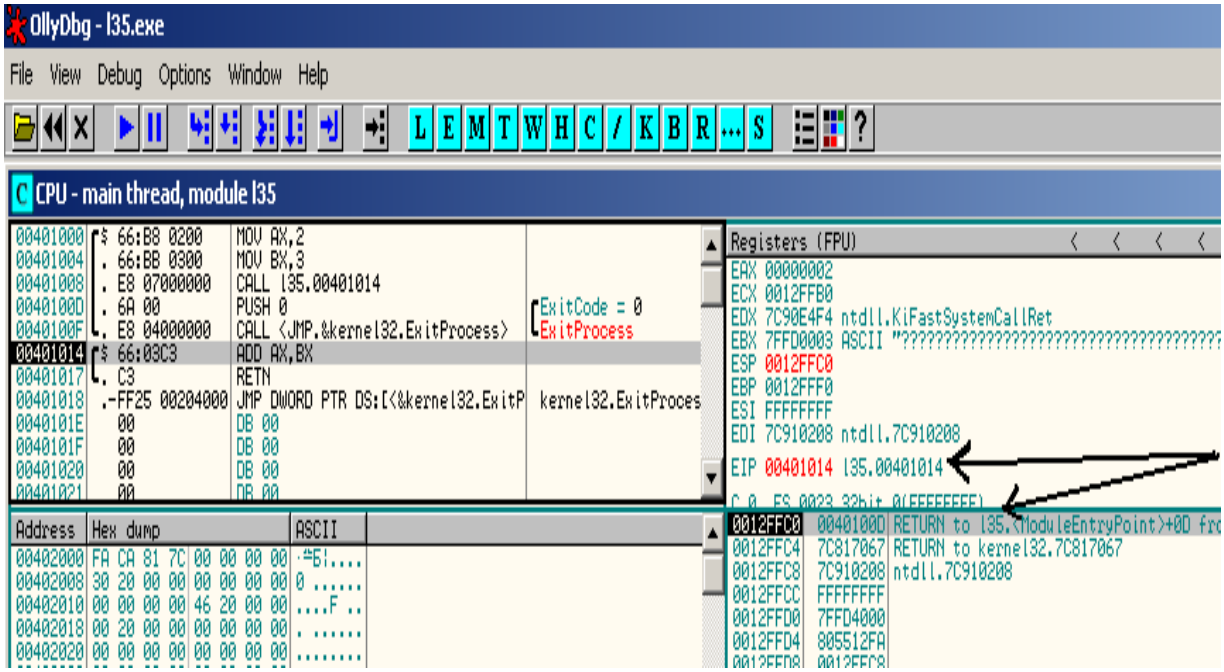

Рисунок А.9 – Вигляд програми та процедури у OLLYDBG.exe після виклику процедури

Програму можливо написати по-іншому.

```
start:
push DWORD PTR 2
push DWORD PTR 3
call AddDigs
invoke ExitProcess, 0
AddDigs proc
mov eax,[esp+8]       ; eax=2
add eax,[esp+4]       ; eax=5
ret 8
AddDigs endp
end start
```

Команда **push DWORD PTR 2 (стара команда з новими можливостями)** записує в стек подвійне слово (4 байт) з цифрою 2 (00 00 00 02) минаючи регістри. Те саме можна записати двома командами **mov eax, 2** та **push eax**.

У програмі OLLYDBG.exe на початку виконання програми буде такий вигляд, як на рисунку А.10.

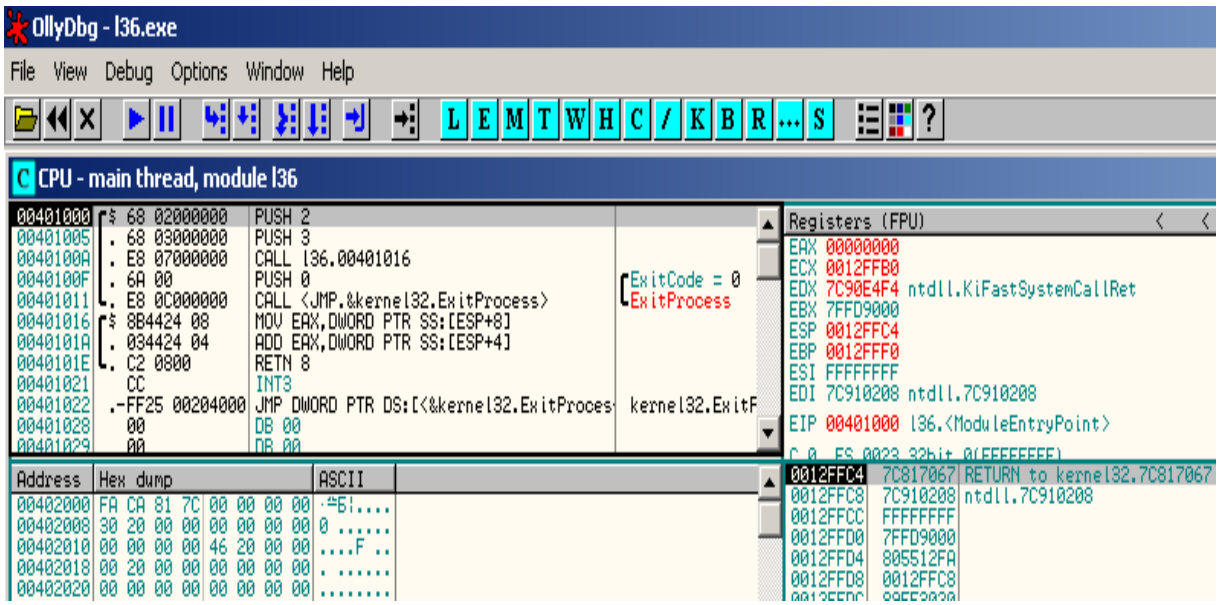

Рисунок А.10 – Вигляд програми та процедури у OLLYDBG.exe на початку виконання програми

Після двох команд **push** та команди **call** в стеку зберігається три чотири байтові числа (перше – 00 00 00 02; друге – 00 00 00 03; третє – адреса повернення з процедури). Так, щоб звернутися до стеку та витягти дані, необхідно внести зсув у регістр **esp,** а команді **ret** поставити посилання на 8 байт (не забуваємо, що рахунок йде від нуля, а отже фактично 8 байт – 4+4+1) (рисунок А.11).

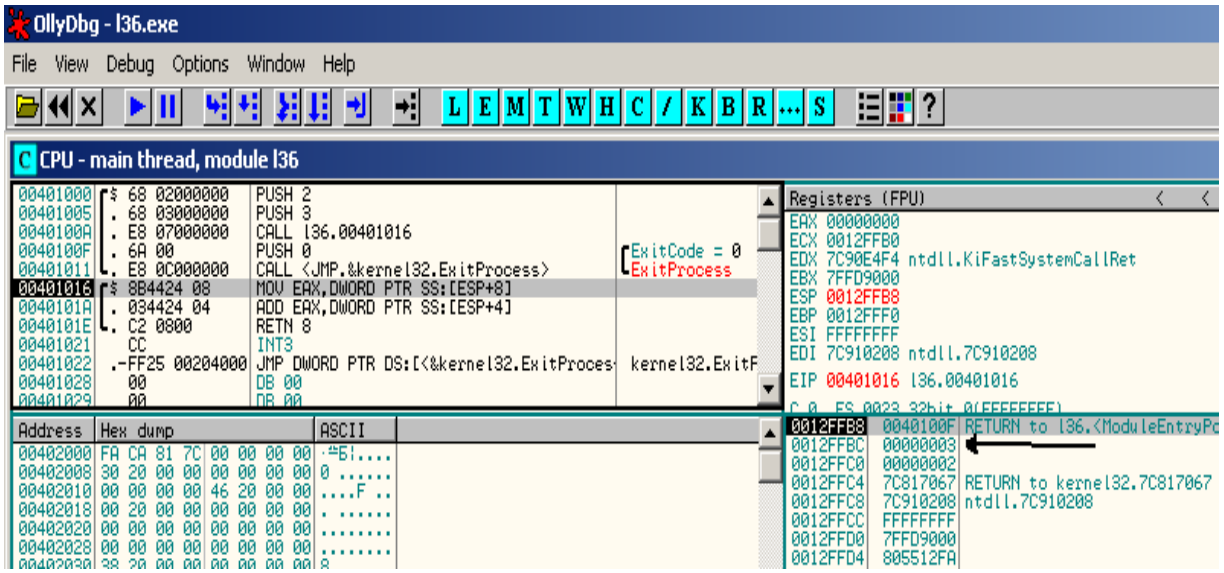

Рисунок А.11 – Вигляд програми та процедури у OLLYDBG.exe після виклику процедури

Важливо пам'ятати. Необхідно дуже ретельно працювати зі стеком. А саме адреса вершини стеку наприкінці роботи повинна залишитись такою, як була до початку роботи зі стеком.

Практична частина

Використовуючи досвід, отриманий у попередніх роботах (пункт 2 – додаток А.3) необхідно набрати та компілювати програми, наведені в теоретичній частині.

Використовуючи налагоджувач OLLYDBG.exe, можна спостерігати зміни, що відбуваються зі стеком в правому нижньому вікні, а вміст регістрів можна спостерігати в правому верхньому вікні. Для покрокового виконання програми необхідно скористатися клавішею F7 або F8.

Важливо пам'ятати, що команда POP збільшує адресу вершини стеку, але не стирає самі числа. Після команди POP числа, як і раніше, лежать у стеку та будуть там знаходитися до тих пір, поки чергова команда PUSH їх не зітре. Щоб їх побачити, необхідно натиснути стрілки вікна стеку, після чого вершина зміститься, відкривши значення. Якщо є необхідність, то можна витягати ці значення, але вже не за допомогою команди POP, а за допомогою непрямої адресації, що розглянуто в цьому додатку.

Самостійно змінити програми та обидва отримані варіанти. Досконало переглянути це у налагоджувачі.

Необхідно уважно поставитися до розташування даних у стеку. Для цього потрібно багато разів опрацювати зміни чисел у регістрах і подивитися на розташування їх у налагоджувачу, провести обчислення уручну та перевірити їх на комп'ютері.

Важливо! Непряма адресація не змінює адресу вершини стеку, що знаходиться у регістрі esp.

3 Дослідження можливості виведення інформаці на екран монітора

Теоретична підготовка

Процедури Windows API – процедури, написані для зручності програмістів, що розміщені в спеціальні бібліотеки.

Перш, ніж виводити що-небудь на екран монітора, порівняємо готові процедури ОС та написані нами раніше. Якщо подивитися у вікно налагоджувача, то виявилося, що Assembler у

процесі компіляції змінює і нашу процедуру **AddDigs,** та процедуру Windows API **ExitProcess** на команди push и call. Єдиною відмінністю є необхідність підключення бібліотеки (адже не ми написали процедуру **ExitProcess**), у якій Assembler повинен знайти відповідні інструкції процедурі Windows API. Тому процедури Windows API можна використати як свої власні. Нижче наведена програма, яка використовує процедуру **ExitProcess** інакше, ніж ми звикли. Наберіть програму та подивіться на неї в ollydbg. (рисунок А.12).

```
.386
.model  flat,stdcall
option casemap:none
includelib \myasm\lib\kernel32.lib
ExitProcess proto :DWORD
.code
start:
push 0
call ExitProcess
end  start
```

Програма передає параметр процедурі **ExitProcess** командою **push**, а потім викликає саму процедуру командою **call**.

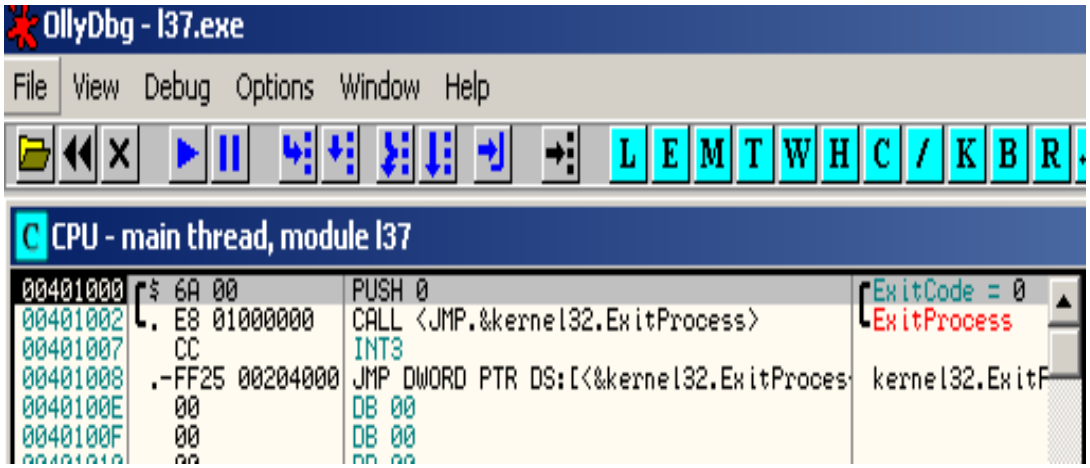

Рисунок А.12 – Вигляд програми у OLLYDBG.exe на початку виконання

Важливим елементом програмування є інтерфейс користувача та комп’ютера. Тому необхідно навчити комп’ютер виводити інформацію на екран монітора. Програма виведення на екран фрази.

```
.386
.model  flat, stdcall
option  casemap:none
ExitProcess             proto :DWORD
GetStdHandle  proto :DWORD
WriteConsoleA       proto :DWORD,:DWORD,\
:DWORD,:DWORD,:DWORD
includelib \myasm\lib\kernel32.lib
.data
stdout   DWORD ?
msg      BYTE "Первые слова",0dh,0ah
cWritten DWORD ?
.code
start:
invoke  GetStdHandle, -11
mov stdout, eax
invoke WriteConsoleA, stdout, ADDR msg, sizeof msg, \
ADDR cWritten, 0
invoke ExitProcess, 0
end start
```

Розглянемо роботу програми.

Директива **option casemap:none** примушує компілятор розрізнять в програмі рядкові та прописні букви.

Текст, що буде виведено на екран монітора, розміщується у змінній **msg BYTE "Первые слова",0dh,0ah (**рисунок А.13). Але слід пам’ятати, що після роботи компілятора всі змінні буде замінено на адреси сегмента даних **(**рисунок А.13), де і розташовується інформація.

За допомогою директиви **invoke** в програмі викликаются дві нові процедури Windows API: **GetStdHandle, WriteConsoleA.** Кажній процедурі на початку програми указуються їхні прототипи. У всіх трьох процедур размір параметрів дорівнює DWORD (34 байт). Прототип **WriteConsoleA** не уміщався на одному рядку, тому опис процедури було продовжено на іншому рядку. Перенесення оформляється оберненою косойю рискою – «\».

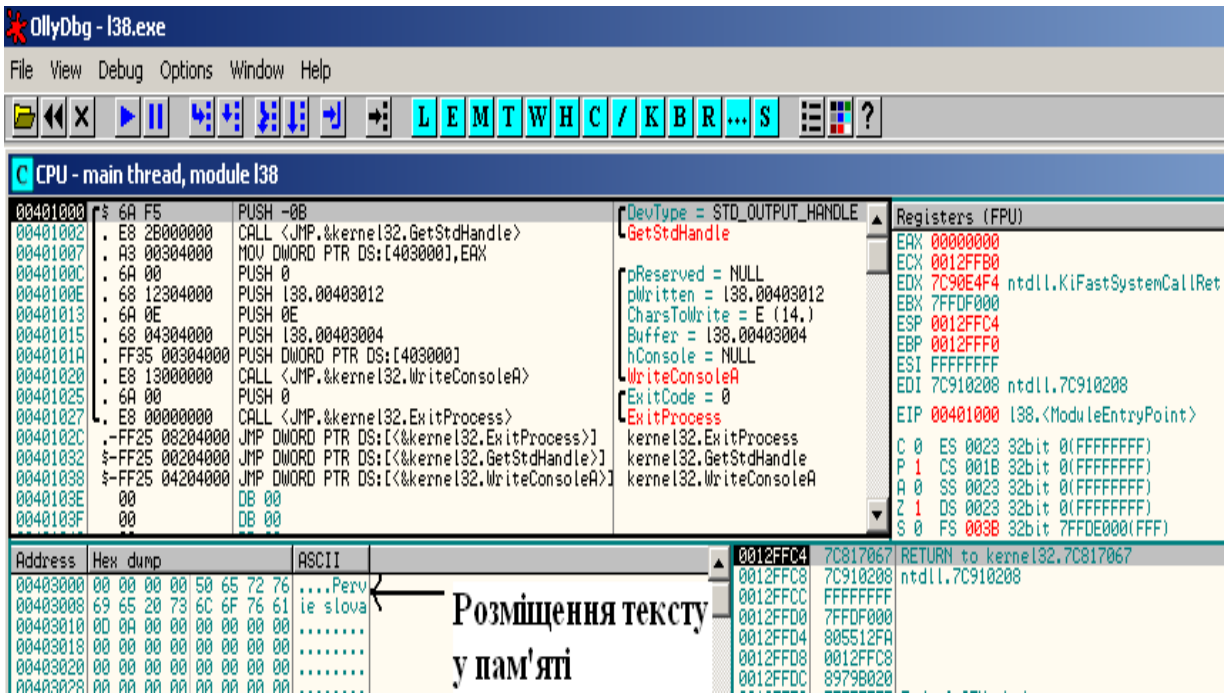

Рисунок А.13 – Вигляд програми у OLLYDBG.exe на початку виконання

Процедура **GetStdHandle** одержує дескриптор стандартного пристрою – число, яке потрібно вказувати іншим процедурам, що взаємодіють з цим пристроем. Її єдиний параметр показує, якого роду дескриптор потрібно отримати. Наприклад, дізнатися **дескриптор стандартного пристрою виведення (монітор)**, куди буде відправлена ваша фраза, параметр повинен бути рівний **«-11»** (або B у шіснадцятковому форматі).. Як і багато інших процедур, **GetStdHandle** поміщує результат роботи в регистр **eax.** (рисунок А.14), тому потрібна ще одна інструкція **mov stdout, eax**, щоб зберегти результат роботи процедури в пам'яті (змінна **stdout**), адже **eax** часто використовуємо та інформація може бути втрачена, якщо її не покласти у спеціально заявлену змінну (рисунок А.15).

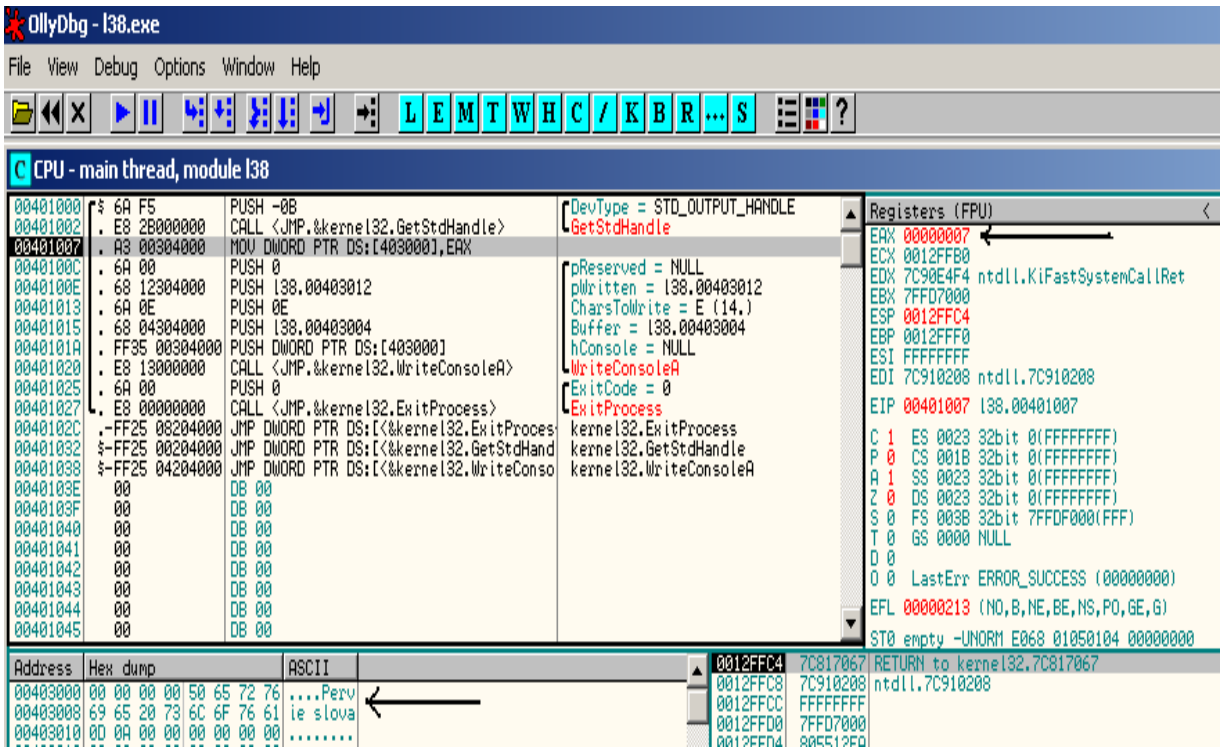

Рисунок А.14 – **GetStdHandle** розміщує результат своєї роботи в регістр загального призначення **eax**

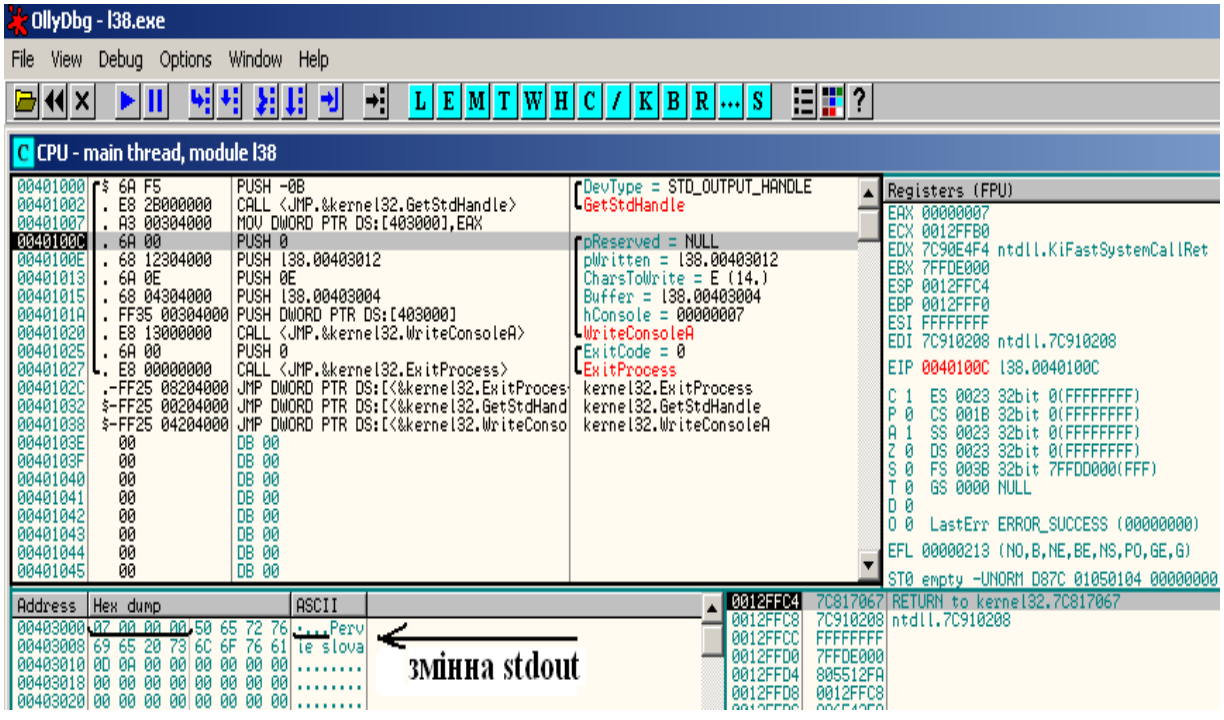

Рисунок А.15 – Результат роботи процедури **GetStdHandle перенесено у змінну** в пам'яті

Треба пам’ятати, що у ПК використовується система пам’яті little-endian (від молодшого значення до великого), тому запис у сегменті пам’яті починається з 07.

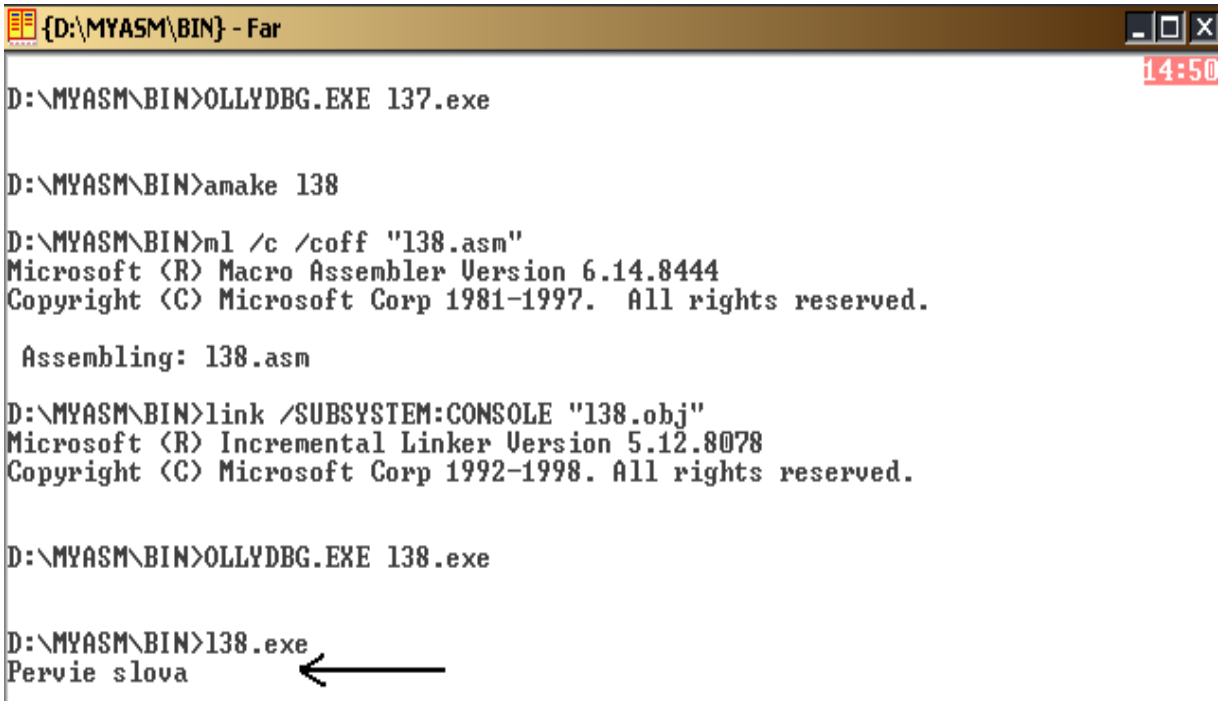

Рисунок А.16– Результати роботи програми

Для виведення інформації на екран використовують процедуру **WriteConsoleA**. Вона має п’ять параметрів (п’ятий завжди дорівнює нулю). **Stdout, msg, cWritten** – змінні, вони можуть мати різні імена:

1) **stdout** – дескриптор стандартного пристрою виведення, куди буде відправлена ваша фраза, був отриманим процедурою **GetStdHandle;**

2) **ADDR msg** – адреса початку повідомлення, що обчислюється директивою **ADDR**. Як і все в ПК, повідомлення представлені послідовністю чисел. Кожна буква повідомлення кодується певним числом. Так, наприклад, прописна російська буква «Н» кодується числом 8D (шіснадцятковим) або 141 (десятковим). Всього використовуються 256 символів, які кодуються, а отже, код будь-якої уміщується в одному байті. **0dh, 0ah – командують процедурою** переведення рядка;

3) **SIZEOF msg** – розмір повідомлення, тобто кількість байтів у ньому. Розмір обчислюється під час компіляції програми директивою **SIZEOF**;

4) **ADDR cWritten –** адреса ділянки пам'яті, де процедура збереже число виведених на екран символів.

Якщо навести курсор у FAR.EXE на програму l38.exe та натиснути ENTER, то результат роботи програми буде таким, як на рисунку А.16.

Практична частина

Використовуючи досвід отриманий у попередніх роботах (частина 2 – додаток А.1), необхідно набрати та компілювати програми, наведені в теоретичній частині.

Використовуючи налагоджувач OLLYDBG.exe, можна спостерігати зміни, що відбуваються зі стеком в правому нижньому вікні, а вміст регістрів можна спостерігати в правому верхньому вікні. Для покрокового виконання програми необхідно скористатися клавішею F7 або F8.

Важливо пам'ятати, що команда POP збільшує адресу вершини стеку, але не стирає самі числа. Після команди POP числа, як і раніше, лежать у стеку та будуть там знаходитися до тих пір, поки чергова команда PUSH їх не зітре. Щоб їх побачити, необхідно натиснути стрілки вікна стеку, після чого вершина зміститься, відкривши значення. Якщо є необхідність, то можна витягати ці значення, але вже не за допомогою команди POP, а за допомогою непрямої адресації, що розглянуто в цьому додатку.

Самостійно змініть програми та обидва отримані варіанти необхідно досконало переглянути у налагоджувачі.

Необхідно уважно поставитись до розташування даних у стеку. Для цього потрібно багато разів опрацювати зміни чисел у регістрах і подивитися на розташування їх у налагоджувачу, провести обчислення уручну та перевірити їх на комп'ютері.

4 Дослідження змісту бібліотек Assembler

Теоретична підготовка

Процедури Windows API – процедури, написані для зручності програмістів. Вони розташовані у спеціальних бібліотеках. Під час розвитку програмного забезпечення було

вирішено, що найбільш популярні процедури необхідно зробити доступними для всіх програмістів, бо це значно скорочує час розроблення програмного забезпечення. Наприклад, якщо хтось написав процедуру виведення на екран монітора, то навіщо її кожного разу писати наново, якщо можна розмістити її у спеціальному файлі (бібліотеці), тоді її зможе використовувати кожний. Також було розроблено файли-бібліотеки, які містили опис параметрів процедури та стандартні константи.

Модифікація програми виведення на екран монітора фрази, що було розглянуто раніше.

```
.386
.model  flat, stdcall
option  casemap:none
include    \myasm\include\windows.inc
include    \myasm\include\kernel32.inc
includelib \myasm\lib\kernel32.lib
.data
stdout     DWORD ?
msg        BYTE  "Первые слова",0dh,0ah
cWritten   DWORD ?
.code
start:
invoke  GetStdHandle, STD_OUTPUT_HANDLE
mov stdout, eax
invoke  WriteConsoleA, /
stdout, ADDR msg, sizeof msg, ADDR cWritten, NULL
invoke  ExitProcess, 0
end start
```

Розглянемо роботу програми.

Замість

```
ExitProcess          proto :DWORD
GetStdHandle      proto :DWORD
WriteConsoleA    proto
            :DWORD,:DWORD,:DWORD,:DWORD,:DWORD
```

З’явились **include \myasm\include\windows.inc,** де міститься константа **STD_OUTPUT_HANDLE equ -11** (рисунок А.17), **include \myasm\include\kernel32.inc,** де міститься опис прототипів процедур (рисунок А.18). Тоді Assembler, знайшовши

незнайомі в програмі змінні та назви, звернеться до указаних файлів, а знайшовши їх там, використає їх у процесі компіляції.

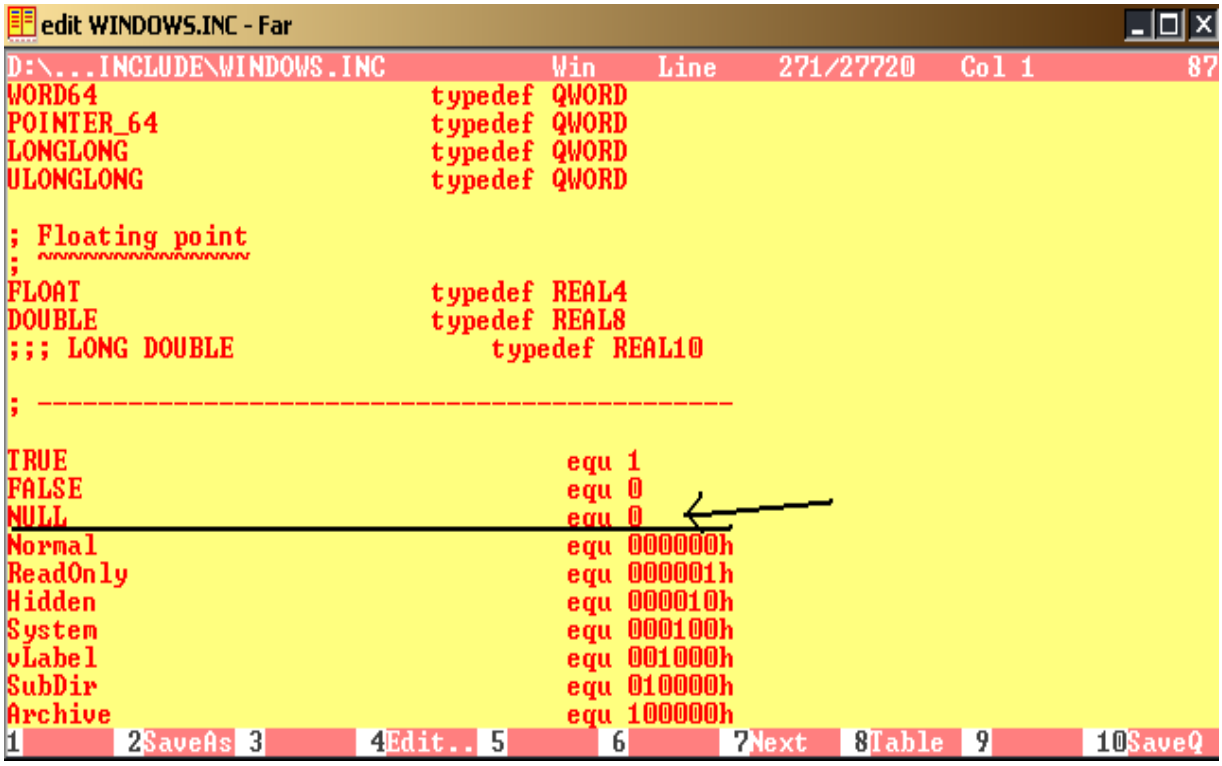

Рисунок А.17 – Розташування константи NULL

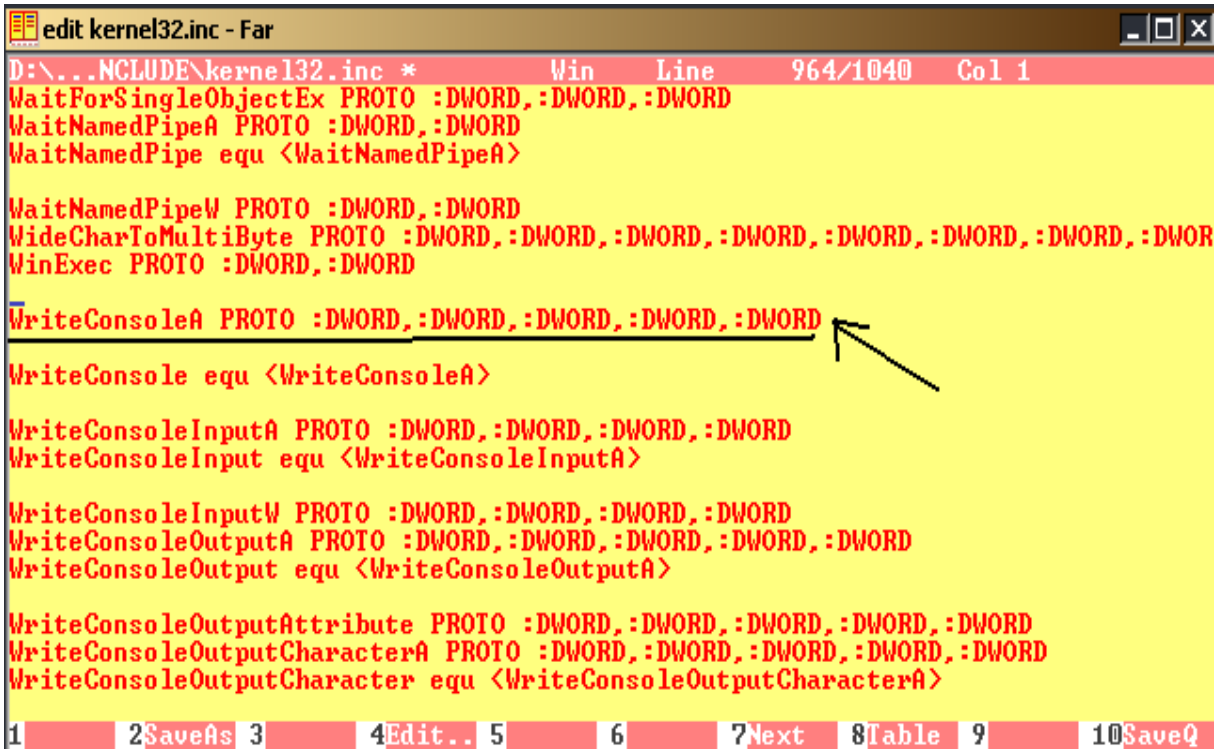

Рисунок А.18 – Розташування опису процедури WriteConsoleA

Використаємо можливості виклику процедур для нашої процедури**AddDigs:**

```
.386
.model  flat,stdcall
option casemap:none
includelib \myasm\lib\kernel32.lib
ExitProcess proto :DWORD
AddDigs proto :DWORD, :DWORD
.code
start:
invoke AddDigs,2,3
invoke ExitProcess,0
AddDigs proc arg1:DWORD,arg2:DWORD
mov eax,[esp+8]  ; eax=2
add eax,[esp+12] ; eax=5
ret 8
AddDigs endp
end start
```

Зверніть увагу на зсув цифр при непрямій адресації в процедурі. Тепер звернення до стеку йде не 4 та 8 (попередній пункт), а 8 та 12. Дивимося в налагоджувач.

```
PUSH EBP
MOV EBP,ESP
MOV EAX,DWORD PTR SS:[ESP+8]
ADD EAX,DWORD PTR SS:[ESP+C]; C=12 у десятковій системі
LEAVE          ; Еквівалент :      mov esp,ebp
               ;                   pop ebp
RETN 8
```

Фактично наша процедура опинилася всередині виділених інструкцій. Assembler зберігає покажчик стеку в регістрі **EBP.** Якщо регістр **EBP не з**мінюється всередині процедури, то адресу вершини стеку можна відновити в **ESP.** А для того щоб не втратити дані, що знаходяться в **EBP,** їх Assembler зберігає в стеку при вході у процедуру. Через те що **EBP** зберігається в стеку, змінюється положення параметрів процедури. Таким чином, спочатку в стек поміщається число 3, потім 2, потім

значення **EBP** та адреса повернення. У нашій процедурі **ESP не** змінюється, але Assembler не знає, змінюється покажчик стека або ні. Тому Assembler зберігає **ESP** про всяк випадок, бо він може змінитися, коли в стек заштовхують параметри процедур, що викликаються з даної, а також при виділенні місця для локальних змінних. Тому зворотне значення процедури збільшено на 4.

Якщо програміст упевнений, що йому не знадобиться втручання Assembler при роботі з процедурами, то достатньо застосувати директиви:

Option prologue:none

Option epilogue:none.

Локальні змінні – змінні, необхідні тільки у процесі виконання процедури, тому під них не варто виділяти місце в сегменті даних (директива .data), а зберігати їх у стеку, де вони зникнуть після закінчення виконання процедури.

Практична частина

Використовуючи досвід, отриманий у попередніх роботах (додаток А – пункт 1) необхідно набрати та компілювати програми, наведені у теоретичній частині.

Використовуючи налагоджувач OLLYDBG.exe, можна спостерігати зміни, що відбуваються зі стеком у правому нижньому вікні, а вміст регістрів можна спостерігати в правому верхньому вікні. Для покрокового виконання програми необхідно скористатися клавішею F7 або F8.

Важливо пам'ятати, що команда POP збільшує адресу вершини стеку, але не стирає самі числа. Після команди POP числа, як і раніше, лежать у стеку та будуть там знаходитися до тих пір, поки чергова команда PUSH їх не зітре. Щоб їх побачити, необхідно натиснути стрілки вікна стеку, після чого вершина зміститься, відкривши значення. Якщо є необхідність, то можна витягати ці значення, але вже не за допомогою команди POP, а за допомогою непрямої адресації, що розглянуто в цьому додатку.

Самостійно змініть програми та обидва отримані варіанти необхідно досконало переглянути у налагоджувачі.

Необхідно уважно поставитися до розташування даних у стеку. Для цього потрібно багато разів опрацювати зміни чисел у регістрах і подивитися на розташування їх у налагоджувачі, провести обчислення уручну та перевірити їх на комп'ютері.

5 Виведення чисел на екран

Теоретична підготовка

Найпрактичнішим підходом при програмуванні є виведення результатів роботи на екран монітора. У наведенному прикладі буде вивчено процес виведення чисел. До цих пір на екран монітора виводилися символи. Виведення чисел трохи складніше. Річ у тому, що комп'ютер і людина розуміють числа по-різному.

У зв'язку з цим необхідно дуже ретельно підійти до чіткої послідовності матеріалу, що буде отримано.

У попередніх прикладах для виведення на екран використовувалася процедура **WriteConsoleA.** Потрібно пам'ятати, що дана процедура виводить тільки готові до виведення символи. Так склалося, що весь список символів в ПК складається 256 символів – ASCII-код (рисунок А.19).

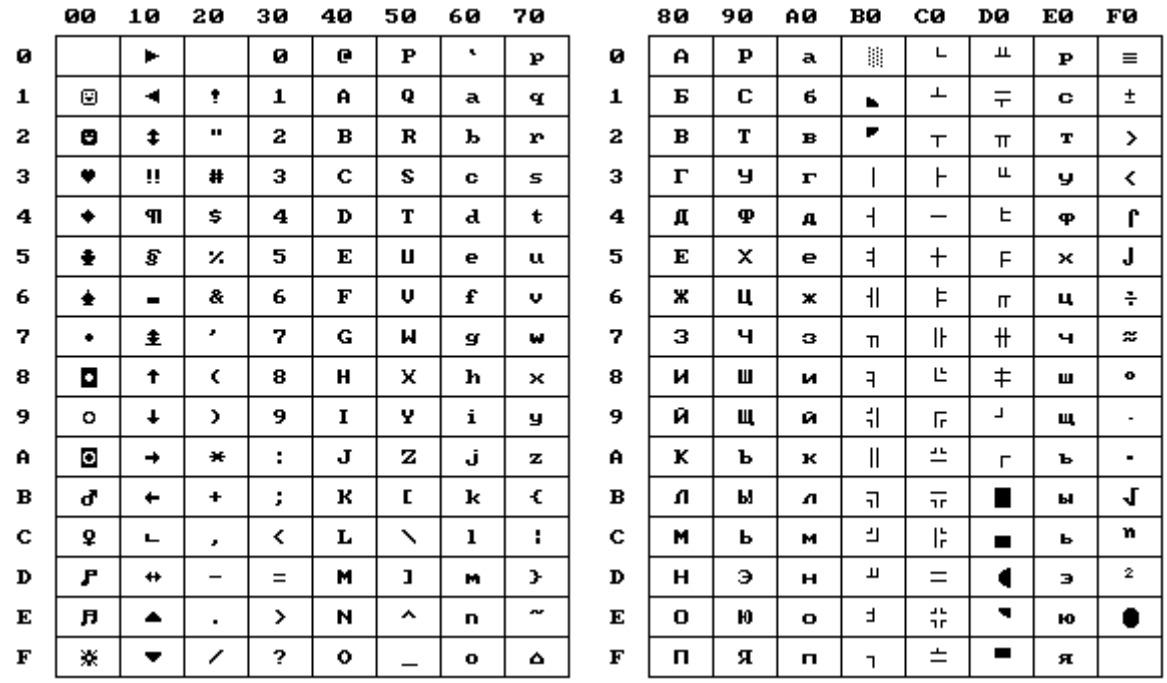

Рисунок А.19 – Таблиця ASCII

Цифри мають свій порядковий номер у цій таблиці. Так, число 0 знаходиться під номером 30 у шістнадцятковому форматі або 48 у десятковому, 1 – 31 або 49, 2 – 32 або 50 і так далі. Також потрібно пам'ятати, що число 1 може займати BYTE,

WORD, DWORD (01, 0100, 01000000 відповідно). А значить, перед виведенням числа на екран необхідно перетворити його у зрозумілий користувачу вигляд. Інакше ви ризикуєте видати замість 1234 – 30313233 або 01020304. Для вирішення цього завдання використовується спеціальна процедура **wsprintf**, яка готує числа до виведення їх на екран у зрозумілому для читання вигляді. У наступному коді приклад переведення числа в символи, які можна вивести у зрозумілому користувачеві вигляді.

```
.386
.model  flat, stdcall
option  casemap:none
include    \myasm\include\windows.inc
include    \myasm\include\user32.inc ;----------------------
include    \myasm\include\kernel32.inc
includelib \myasm\lib\user32.lib ;---------------------------
includelib \myasm\lib\kernel32.lib
BSIZE   equ 15;----------------------------
.data
ifmt       BYTE "%d",0;----------------------------
buf        BYTE BSIZE dup(?);----------------------------
dig        DWORD 123456;---------------------------
stdout     DWORD ?
cWritten   DWORD ?
.code
start:
invoke  GetStdHandle, STD_OUTPUT_HANDLE
mov stdout, eax
invoke  wsprintf, ADDR buf, ADDR ifmt, dig;--------------------
invoke  WriteConsoleA, stdout, ADDR buf, \
        BSIZE, ADDR cWritten, NULL
invoke  ExitProcess, 0
end start
```

Що відбувається в програмі? Багато моментів програми вже було розглянуто в попередніх прикладах, тому потрібно розглянути тільки основні відмінності.

На початку програми у пам’яті розташовується змінна ***dig DWORD 123456*** (рисунок А.20). Як можна бачити, у символах це незрозумілий набір символів.

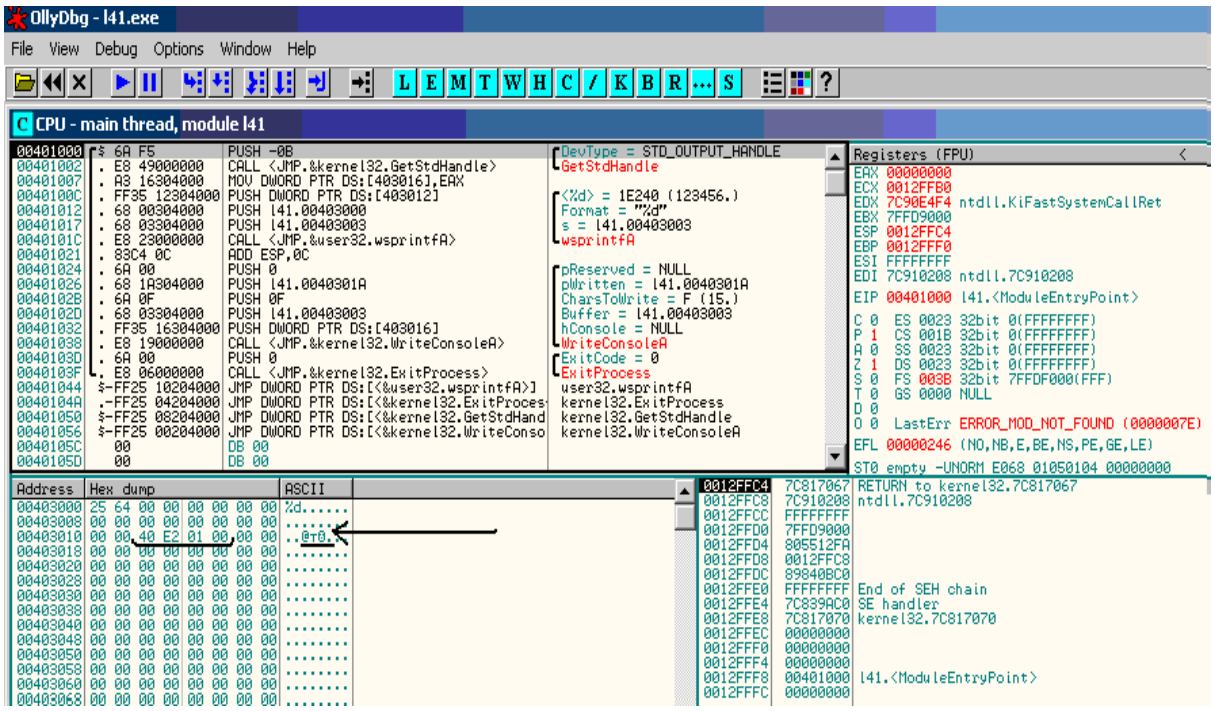

Рисунок А.20 – Програма на початку

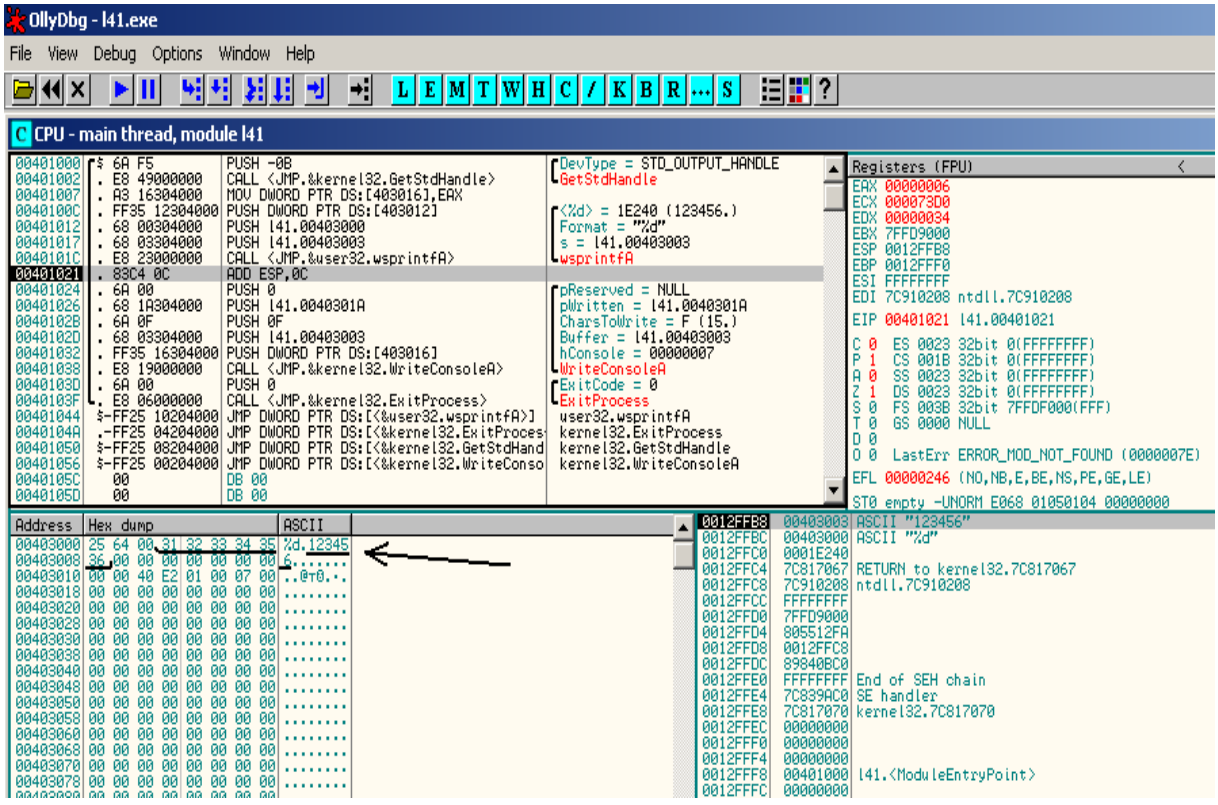

Рисунок А.21 – Після використання процедури **wsprintf**

Опис процедури **wsprintf** знаходиться у двох файлах ***user32.lib і user32.inc,*** тому їх довелося підключити стандартним

способом. Рядок ***BSIZE equ 15*** лише створює константу, яка нам дозволить надалі використовувати ім'я ***BSIZE*** замість того, щоб вишукувати кожне потрібне нам число 15, і за необхідності замінити його на інше число, не перегортаючи всю програму, де це число необхідно змінити (у нашій програмі через її простоту таких проблем не буде, але в подальших програмах це може стати в нагоді). Далі йдуть три змінні для забезпечення роботою **wsprintf**, які ми описуємо у відповідному розділі нашої програми. Процедура **wsprintf** має три параметри:

- buf BYTE BSIZE dup(?) **–** пам'ять для буфер, куди буде записана послідовність символів; ***BSIZE*** – кількість зарезервованих байтів; ***dup(?)*** – означає, що виділяється деякий байт, але їхня кількість наперед невідома і доведеться Assemblery це визначити самому;

- ifmt BYTE "%d",0 – тип формату, дозадає перетворення. У даному випадку перетворення одного цілого числа в послідовність символів (123456 у 1, 2, 3, 4, 5, 6). Форматів багато, але в нашому випадку використовуємо цей;

- dig DWORD 123456 – число, яке необхідно вивести на екран.

Скомпілювавши програму, Assembler знову виявив свавілля, додавши рядок **add esp,12**. Річ у тому, що **wsprintf** має змінну кількість параметрів, на відміну від вивчених раніше процедур. У процесі компіляції Assembler дізнався, що нам знадобляться три параметри та скоректував стек на 12 байт, тому кожний параметр процедури **wsprintf** має DWORD.

Практична частина

Використовуючи досвід, отриманий у попередніх роботах (додаток А – пункт 1) необхідно набрати та компілювати програми, наведені у теоретичній частині.

Використовуючи налагоджувач OLLYDBG.exe, можна спостерігати зміни, що відбуваються в лівому нижньому вікні, а вміст регістрів можна спостерігати в правому верхньому вікні. Для покрокового виконання програми необхідно скористатися клавішею F7 або F8.

Самостійно змініть програми та обидва отримані варіанти необхідно досконало переглянути у налагоджувачі.

Необхідно уважно поставитися до розташування даних у сегменті даних. Для цього потрібно багато разів опрацювати зміни чисел і подивитися на розташування їх у налагоджувачі, провести обчислення уручну та перевірити їх на комп'ютері.