DOI: https://doi.org/10.26565/2304-6201-2023-58-06

УДК (UDC) 004.942 : 371.388

Програмний емулятор навчальної моделі цифрового процесора

Рева С. М., Комеристий В. С.

Рева Сергій Миколайович
к.т.н., доцент кафедри ЕіУС; факультету комп'ютерних наук; Харківський національний університет імені В.Н. Каразіна, майдан Свободи, 6, Харків, Україна, 61022
e-mail: iec-lab@karazin.ua
https://orcid.org/0000-0002-2615-9226

Комеристий Владислав Сергійоич
бакалавр факультету комп'ютерних наук Харківського національного університету імені В.Н. Каразіна, майдан Свободи, 6, Харків, Україна, 61022
e-mail: xa12850420@student.karazin.ua
https://orcid.org/0009-0003-2375-5190

В роботі розглянуто проблеми організації практичних та лабораторних робіт в освітньому процесі у вищих навчальних закладах України в умовах проведення дистанційних занять. Причиною цих проблем є відсутність в нинішніх умовах можливості проведення аудиторних занять з використанням відповідного наочного обладнання та лабораторного устаткування. Як один із можливих методів вирішення цього питання розглядається розробка і створення програмних емуляторів та інтерактивних навчальних додатків.

Метою роботи є покращення якості освітнього процесу при вивченні основ мікропроцесорної техніки за рахунок розробки і використання комп'ютерної моделі навчального процесора.

В статті проведено короткий огляд та **аналіз** існуючих зразків програмних емуляторів цифрових процесорів, що працюють самостійно або у складі інтегрованих середовищ для розробки програмного забезпечення, визначено їх недоліки відносно використання в навчальному процесі, **запропоновано** новий підхід щодо дизайну та програмної реалізації, сформульовано основні технічні вимоги до комп'ютерної моделі. Розглянуто структуру створеного програмного додатку та його користувацький інтерфейс, описано особливості програмної реалізації. Головною відмінністю створеного емулятора від існуючих аналогів є розширене графічне представлення внутрішньої будови процесора, а також анімаційна індикація сигналів і процесів, що протікають під час його роботи.

Програмно **реалізована** комп'ютерна модель дозволяє виконувати розміщену у віртуальній пам'яті програму у одному із трьох режимів. Перший режим забезпечує покрокове виконання команд, наочно демонструючи за допомогою графічних засобів формування внутрішніх сигналів управління процесором та зміну станів його окремих вузлів протягом всього машинного циклу. Другий режим дозволяє поступово виконувати програму по одній команді, відображаючи стан процесора лише після завершення виконання кожної команди. Третій режим призначений для автоматичного безперервного виконання програми із попередньо встановленою швидкістю.

Також наведено **результати** тестування та пробної експлуатації створеного програмного емулятора в умовах дистанційного навчання на факультеті комп'ютерних наук Харківського національного університету імені В.Н. Каразіна.

Підведено **підсумки** та розглянуто **перспективи** подальшого вдосконалення і розширення можливостей використання даної моделі у навчальному процесі.

Ключові слова: *програмний емулятор, комп'ютерне моделювання, алгоритмічна модель, цифровий процесор, користувацький інтерфейс, навчальний процес.*

Як цитувати: Рева С. М., Комеристий В. С. Програмний емулятор навчальної моделі цифрового процесора. *Вісник Харківського національного університету імені В.Н. Каразіна, сер. «Математичне моделювання. Інформаційні технології. Автоматизовані системи управління»*. 2023. т. 58. С.54-63. https://doi.org/10.26565/2304-6201-2023-58-06

How to quote: S.M. Reva, V.S. Komerystyi "Software emulator of the educational model of the digital processor" *Bulletin of V.N. Karazin Kharkiv National University, series "Mathematical modeling. Information technology. Automated control systems"*, vol. 58, pp. 54-63, 2023. https://doi.org/10.26565/2304-6201-2023-58-06

1 Вступ

Фахівці, що працюють у галузі комп'ютерних наук, повинні знати та розуміти базові принципи роботи процесора, який присутній в усіх електронних обчислювальних машинах (ЕОМ), смартфонах, побутовій техніці та інших пристроях, без яких не можливо уявити собі сучасний світ. Особливо це необхідно для майбутніх системних програмістів, які створюють програмне забезпечення для взаємодії користувачів з апаратною частиною пристроїв (драйвери, утиліти, операційні системи тощо), а також для майбутніх розробників нових комп'ютерів і комп'ютерних систем. Ці знання допоможуть розробляти швидкодіючі та ефективні програми, що здатні працювати без використання зайвих ресурсів. Щоб не знищити бажання студентів, а навпаки — зацікавити їх у вивченні основ мікропроцесорної техніки, бажано знайомство розпочинати не із сучасних процесорів зі складною архітектурою, а з простої моделі, на прикладі якої можна пояснити функціонування процесорів та їх взаємодію з іншими компонентами комп'ютера. Викладачі та студенти факультету комп'ютерних наук свого часу створили таку модель [1], щоб надати можливість здобувачам вищої освіти докладно розібратися в будові та принципі дії основного компоненту ЕОМ. Зовнішній вигляд моделі показано на рисунку 1.1.

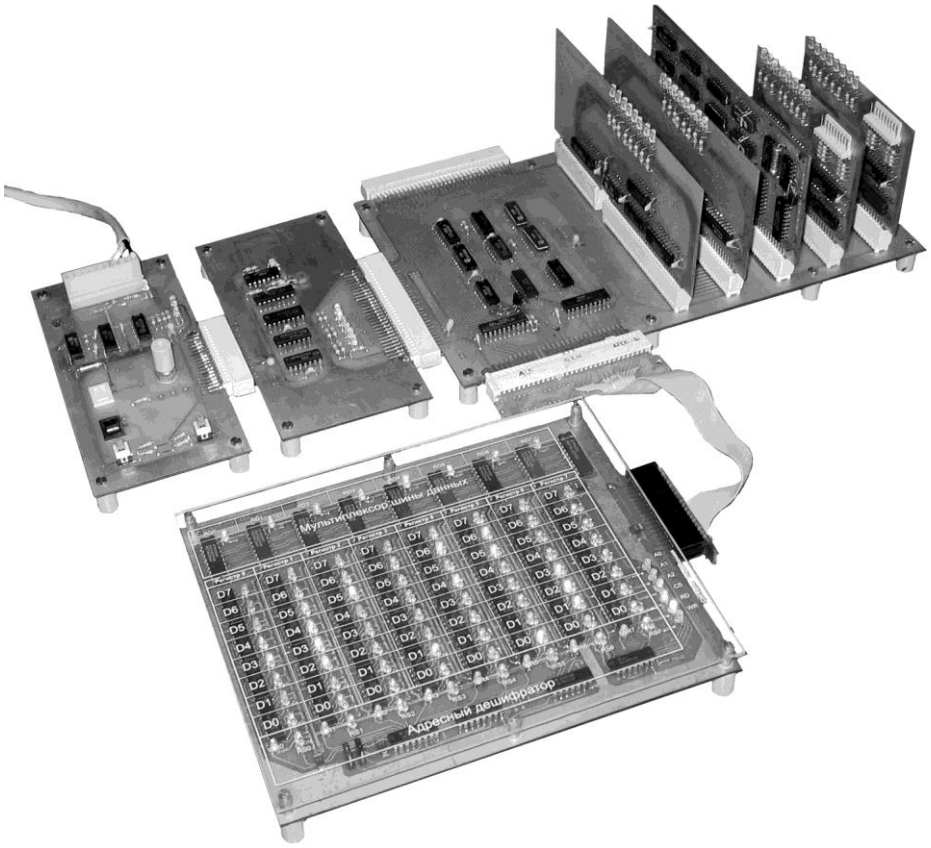

Рис 1.1 – Лабораторна модель цифрового процесора DPM-08

Останні роки стали справжнім життєвим випробуванням як для кожного із нас, так і для освітньої галузі України в цілому. Спочатку — два роки пандемії, які змусили викладачів та студентів відшуковувати нові форми освітнього процесу. Лабораторні роботи та практичні заняття проводилися з використанням додаткової відеокамери, що дозволяла доєднати до zoom-конференції ще одного «учасника» — лабораторну модель цифрового процесора. З початком повномасштабного вторгнення країни-агресора робота комп'ютерних лабораторій в стінах університету стала неможливою. Однак, всупереч усім складнощам, університет продовжує навчати студентів і готувати з них гідних спеціалістів. Проте, сьогоднішні умови не дають можливості здобувачам освіти працювати з реальним обладнанням і знову вимагають від викладачів пошуку нових підходів для формування навичок практичної роботи. Одним із методів вирішення цієї задачі може бути створення емуляторів, математичних і програмних моделей лабораторного обладнання, демонстраційних інтерактивних додатків і тому подібного.

Так з'явилася ідея проведення окремого дослідження та розробки програмного емулятора лабораторної моделі цифрового процесора, який буде доступним для кожного студента. Це дозволить як і раніше проводити лабораторний практикум у відповідності до навчальної програми, а також розширити використання емулятора для виконання студентами індивідуальних робіт, цікавих та пізнавальних домашніх завдань.

Метою даної роботи є покращення навчального процесу завдяки забезпеченню можливості дистанційного вивчення основ мікропроцесорної техніки та отримання практичного досвіду програмування на мовах низького рівня. Результатом дослідження має бути створення програмної моделі цифрового процесора із зручним графічним інтерфейсом, яка допоможе в проведенні дистанційних лекцій та практичних занять в навчальних курсах «Мікропроцесори», «Основи комп'ютерної схемотехніки» та інших. Програма має імітувати роботу реальної лабораторної моделі процесора.

Дана стаття містить кілька розділів. У розділі другому проведено короткий огляд та аналіз існуючих зразків програмних емуляторів цифрових процесорів. Третій розділ містить формулювання основних цілей та технічних вимог до створюваної моделі. В четвертому розділі мова піде про структурну організацію та користувацький інтерфейс програмної моделі процесора. П'ятий розділ присвячений розгляду деяких особливостей цієї розробки. У шостому розділі підведено підсумки та розглянуто перспективи подальшого вдосконалення і розширення можливостей використання даної моделі у навчальному процесі.

2 Короткий огляд та аналіз існуючих програмних рішень

На сьогоднішній день існує чимало програм-емуляторів, які створювалися програмістами від початку появи перших комп'ютерів. Всі вони певною мірою спрощують життя, надаючи можливість виконувати програмний код шляхом імітації роботи процесора та деяких інших важливих компонентів комп'ютера, не маючи при цьому реального обладнання. При цьому спеціалісти отримують ті самі результати, що і при роботі з фізичними пристроями.

Розпочати розгляд існуючих емуляторів варто з програми Turbo Debugger (TD.exe), яка була створена компанією Borland Software Corporation ще наприкінці восьмидесятих років минулого століття [2]. Свого часу це був досить потужний налагоджувач програмного забезпечення, який використовувався в основному з транслятором Borland Turbo Pascal. Емулятор працював в операційній системі DOS в текстовому повноекранному режимі. Після завантаження програмного коду Turbo Debugger забезпечує його відображення в мнемонічних позначеннях асемблера, надає можливість використовувати точки зупинки при виконанні програмного коду, демонструє поточний стан регістрів процесора та ділянки пам'яті, з якою він працює. Не зважаючи на екранний текстовий режим відображення емулятор досить чітко показує всю необхідну інформацію і фокусує увагу оператора на важливих подіях. І хоча цей продукт має досить солідний вік, він і зараз іноді використовується в навчальному процесі для демонстрації роботи процесорів x86 та відлагодження консольних додатків, написаних на асемблері та C++.

Один із можливих варіантів емуляції роботи процесора реалізований в програмі Proteus Design від компанії Labcenter Electronics [3]. Це програмний пакет, який являє собою систему схемотехнічного моделювання на основі математичних моделей окремих компонентів, в тому числі — мікроконтролерів та мікропроцесорних систем. Емулятор має чудові графічні можливості, але пакет орієнтований на розробку електронних пристроїв і систем, а емуляція роботи процесорів не надає можливості вивчати їх внутрішню будову та принцип дії.

Ще одним різновидом є емулятори, вбудовані в сучасні інтегровані середовища для розробки програмного забезпечення, характерним представником яких є програма Keil uVision, розроблена німецькою компанією Keil Elektronik GmbH [4]. Компанія вперше реалізувала компілятор C для мікроконтролерів 8051. Вбудований емулятор поєднує в собі можливості Turbo Debugger (візуалізація стану регістрів, пам'яті і т.п.) та деякі можливості схемотехнічного моделювання, наприклад, побудову осцилограм сигналів, що формуються на виходах портів мікроконтролера. Однак дослідити схемну конструкцію та роботу процесорного ядра в цьому середовищі також не можливо.

Підводячи підсумки цього короткого огляду, можна зробити висновок, що велика кількість досить потужних емуляторів, якісно створених відомими світовими компаніями, орієнтована в першу чергу на розробників програмного забезпечення та розробників електронних систем. Вони не володіють достатньою мірою наочності для демонстрації базових принципів будови та

функціонування мікропроцесорів і мікроконтролерів. Вони також не можуть бути налаштовані на роботу процесора з довільною системою команд і не можуть бути використані як заміна лабораторної моделі процесора DPM08 під час проведення занять у дистанційній формі. Саме це стало вагомим фактором у прийнятті рішення щодо створення власного програмного емулятора.

Розроблена програмна модель представляє собою застосунок зі зручним та зрозумілим графічним інтерфейсом. Він може працювати під будь-якою операційною системою сімейства Windows, починаючи з Windows XP і закінчуючи сучасними Windows 10 та Windows 11. Це дає можливість застосовувати додаток як на нових комп'ютерах, так і на застарілих ноутбуках, що особливо важливо в нинішніх умовах проведення освітнього процесу. Використання сумісно з UNIX-подібними системами поки що не розглядалось, але в майбутньому це цілком можливо.

Для розробки програми використовувалась мова програмування C++ [5] та набір бібліотек Qt [6]. Цей набір надає багато можливостей для створення якісних користувацьких додатків, а також дозволяє легко впроваджувати кросплатформеність, що і стане в нагоді при реалізації роботи емулятора під UNIX-подібними системами.

3 Основні цілі та завдання дослідження.

Виходячи з результатів проведеного аналізу автори прийшли до висновку, що розробка власної програмної моделі простого цифрового процесора допоможе вирішити цілий ряд питань, що виникли з переходом на дистанційну форму проведення занять, і які не можуть бути вирішені шляхом застосування існуючих програмних емуляторів. Для початку були визначені основні цілі розробки та технічні вимоги, яким повинна відповідати дана модель.

1. Програма повинна моделювати роботу лабораторної моделі цифрового процесора DPM08. Для цього вона має емулювати виконання всіх інструкцій, які входять до складу системи команд цього процесора. Закладати можливості гнучкого налаштування на довільну систему команд не варто, оскільки це призведе до ускладнення програми та знизить її швидкодію.
2. Програма має працювати під операційною системою Windows як найбільш поширеною у студентському середовищі. При цьому бажано забезпечити її сумісність як з останніми версіями, так і з попередніми, починаючи з Windows XP. Це надасть можливість використання програмної моделі навіть на застарілому апаратному обладнанні.
3. Для повної емуляції апаратних можливостей реального процесора бажано створити модель програмної пам'яті, яка може змінювати свій розмір, імітуючи роботу як з демонстраційним блоком пам'яті об'ємом вісім байт, так і з усім адресним простором, що доступний процесору.
4. Для зручності роботи з емулятором та ручного редагування програми безпосередньо в машинних кодах необхідно забезпечити можливість керування моделлю не лише за допомогою миші, але й з клавіатури з використанням «гарячих» кнопок оперативного управління.
5. Модель повинна наочно відображати формування основних внутрішніх та зовнішніх сигналів управління, поточний стан усіх регістрів та програмної пам'яті, а також графічними засобами акцентувати увагу на виконанні основних процесів машинного циклу.

4 Структура програмної моделі.

Процесор — це апаратний пристрій, який керує процесом виконання програми. Програма може зберігатися у спеціальній програмній пам'яті (гарвардська архітектура) або у програмному сегменті загальної оперативної пам'яті (архітектура фон Неймана). Але у будь якому випадку для емуляції роботи процесора потрібно мати дві комп'ютерні моделі: модель пам'яті для зберігання і редагування програмного коду та модель самого процесора, який буде послідовно зчитувати та виконувати цей код. Тому при розробці комп'ютерної моделі було вирішено створити два окремі компоненти, взаємодія між якими буде відбуватися через масив програмного коду та деякі змінні, що будуть забезпечувати певну їх синхронізацію. На рисунку 4.1 показано зовнішній вигляд робочого вікна створеної програмної моделі.

Для забезпечення високої якості відображення окремих графічних елементів додаток має фіксований розмір робочого вікна, яке умовно розділене на дві частини. Зліва знаходиться монітор емулятора процесора DPM08, на якому відображається поточний стан його основних складових частин і сигналів, а внизу — органи управління моделлю процесора. Права частина вікна містить інтерфейс моделі програмної пам'яті. При цьому користувач не може працювати одночасно з обома частинами: якщо він працює з модулем пам'яті, то управління модулем процесора заблоковано і навпаки. Далі окремо наведено опис кожної моделі.

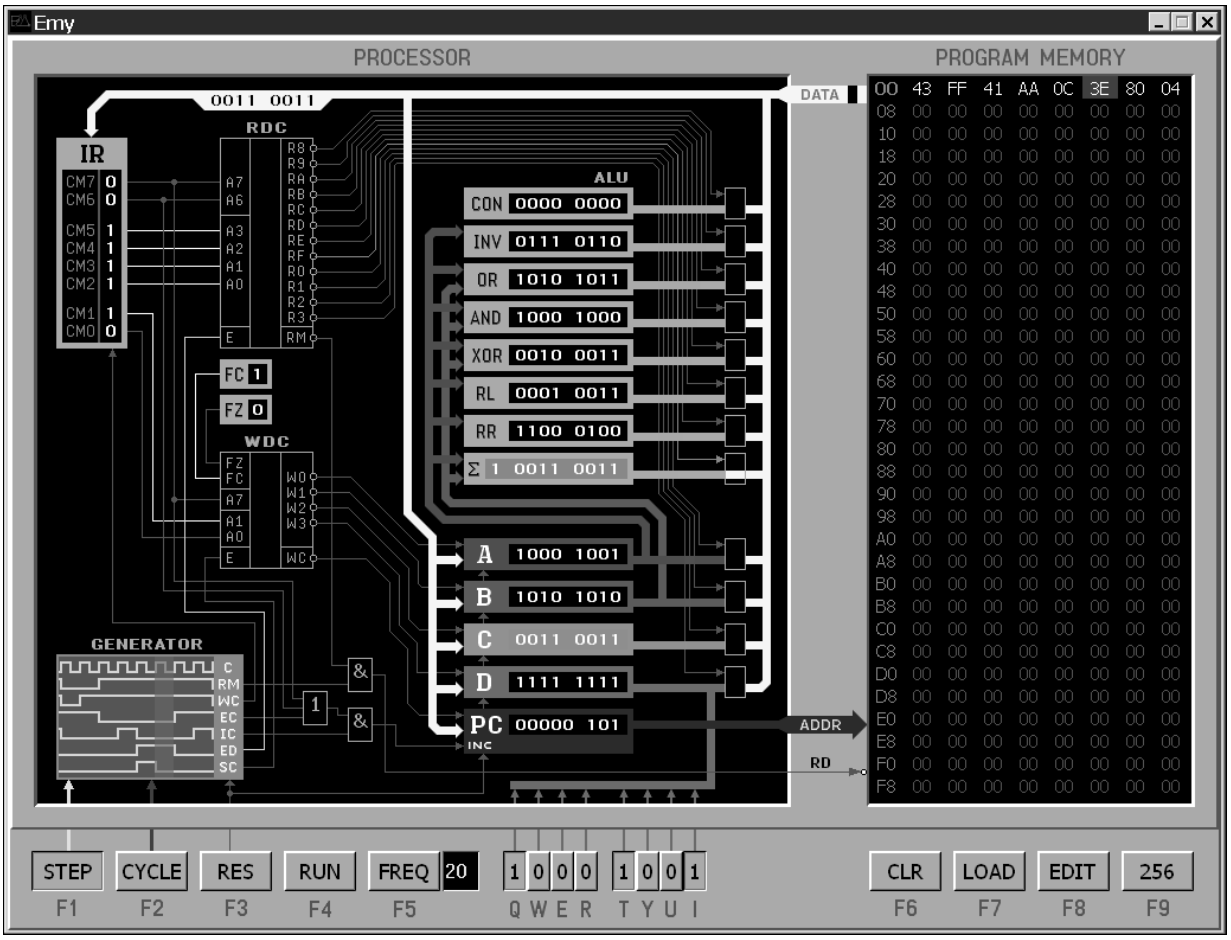

Рис 4.1 – Зовнішній вигляд інтерфейсного вікна програмної моделі процесора DPM-08

Модуль пам’яті представлений у вигляді таблиці, яка відображає масив із 256 однобайтних чисел, що знаходяться у пам’яті комп’ютера. Таблиця складається із 32 рядків, кожен із яких містить вісім числових значень окремих байт масиву та розпочинається з адреси — номера елемента масиву, що показаний першим у цьому рядку відразу після вказівника адреси (рис. 4.2). Для компактності запису та зручності зчитування дані та адреси представлені у шістнадцятковому форматі (від 00 до FF), значення адрес виділені зеленим кольором. Кожен елемент масиву відображає значення однієї лунки програмної пам’яті, з якою може працювати програмна модель процесора. Масив із 256 елементів моделює максимально доступний об’єм пам’яті, так як реальний процесор DPM08 має лише восьмирозрядну адресну шину.

Модель пам’яті передбачає три можливих стани кожної лунки (див. рис. 4.2):

– лунка недоступна для програмної моделі процесора та для редагування — значення байта відображається темно-сірим кольором;

– лунка доступна для використання, але не містить коду програми — значення байта відображається світло-сірим кольором;

– лунка містить байт програмного коду, що завантажений туди будь-яким доступним методом — значення відображається білим кольором.

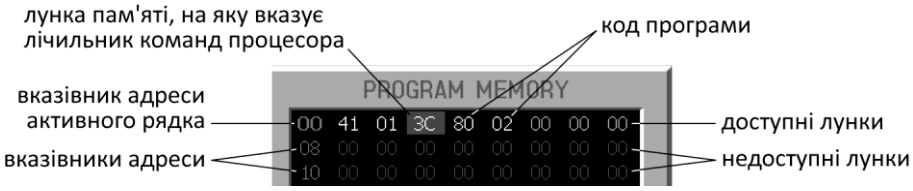

Рис 4.2 – Відображення елементів моделі пам’яті

Завдяки таким властивостям досить просто реалізується модель пам'яті об'ємом вісім байт, яка зазвичай використовується з процесором DPM08 при проведенні перших лабораторних робіт з програмування в машинних кодах. Для цього доступними для завантаження коду призначаються лише вісім перших лунок, а інші залишаються недоступними. Зміна максимального об'єму пам'яті здійснюється при натисканні на кнопку управління «256».

Інші кнопки управління, що розташовані під інтерфейсним вікном модуля програмної моделі пам'яті (див. рис. 4.1) виконують наступні функції:

«CLR» – натискання на кнопку очищає всі елементи масиву пам'яті;

«LOAD» – при натисканні на цю кнопку відкривається стандартне діалогове вікно Windows, яке дозволяє відшукати на диску та завантажити до масиву пам'яті вміст попередньо створеного файлу з програмним кодом у бінарному форматі або Intel-HEX форматі;

«EDIT» – вмикає режим ручного редагування пам'яті, дозволяючи вводити та змінювати дані, що знаходяться в доступних для редагування лунках (клітинках таблиці).

Кнопки управління модулем пам'яті продубльовані на клавіатурі комп'ютера функціональними кнопками F6 … F9. Також програмна модель пам'яті передбачає можливість виділення тим чи іншим кольором тих клітинок таблиці, на які в поточний момент часу вказує лічильник команд PC. Це додатково акцентує увагу користувача на взаємодії програмної моделі процесора та модуля пам'яті під час їх спільної роботи.

Таким чином, після завантаження програмного коду в блок пам'яті користувач може розпочинати роботу безпосередньо з емулятором процесора.

Модуль процесора — це основна частина програми, яка моделює роботу процесора під час виконання команд. Так само, як і блок пам'яті, модуль процесора має ряд кнопок управління, що розташовані під вікном монітора стану процесора (рис. 4.1). Активувати їх можна кліком миші або натисканням на клавіатурі відповідних функціональних кнопок F1 … F5. У вікні монітора користувач може спостерігати за послідовністю формування сигналів управління, що генеруються протягом машинного циклу, за поточним станом шин, за процесом дешифрації інструкцій та зміною станів окремих регістрів і флагів, за роботою вузлів арифметико-логічного пристрою (АЛП). Дана програмна модель підтримує три режими роботи:

– покрокове виконання команди;
– покомандне виконання програми;
– безперервний режим роботи із заданою частотою.

Перший режим активується натисканням кнопки «STEP». Однократне натискання на цю кнопку імітує надходження у процесор одного імпульсу тактової частоти. Режим має найбільшу ступінь демонстраційності та наочності і дозволяє детально відслідкувати виконання всіх етапів машинного циклу, підсвічуючи на кожному кроці сигнали, які задіяні в даний момент часу. Активовані сигнали мають різну кольорову окраску, що допомагає зрозуміти суть процесів, що відбуваються. Зеленим кольором відображаються сигнали читання, червоним — сигнали запису, білим — деякі інші сигнали управління, дешифратори коду операції в момент їх активації підсвічуються білим кольором. Модуль пам'яті в цей час також використовує кольорове виділення клітинок таблиці, з якими взаємодіє модуль процесора: лунки пам'яті, до яких звертається процесор, підсвічуються синім кольором, а в момент їх зчитування — зеленим. (як і самі сигнали читання). До складу монітора стану процесора входить невелике вікно, у якому відображаються сигнали генератора управління. На кожному етапі виконання команди відповідна фаза формування цих сигналів виділяється блакитним кольором, надаючи можливість зрозуміти їх призначення та відстежити всю послідовність виконання машинного циклу.

Другий режим здійснює покомандне виконання програми. Однократне натискання на кнопку «CYCLE» призводить до виконання однієї команди. В цьому режимі користувач не бачить послідовності формування окремих сигналів, адже весь машинний цикл проходить автоматично. Відображається лише оновлений стан лічильника команд та флагів, а також зміни в регістрах та на виходах функціональних вузлів АЛП, які відбулися в результаті виконання команди.

Індикація в модулі пам'яті також змінюється. Після виконання кожної команди фіолетовим кольором підсвічуються клітинки таблиці, в яких розміщується наступна команда. Якщо вона однобайтна, кольором виділяється одна клітинка, а якщо двобайтна — обидві. Цей режим дозволяє спостерігати за послідовністю виконання команд, відслідковувати алгоритм та відлагоджувати програму, завантажену у модуль пам'яті, якщо вона працює некоректно.

Третій режим активується натисканням кнопки «RUN». Він забезпечує безперервне виконання програми зі встановленою швидкістю. В цьому режимі користувач бачить зміни лише в регістрах оперативного призначення. Автори дійшли висновку, що в даному режимі відсутня потреба спостерігати за змінами АЛП та окремими сигналами, головне — отримати результат роботи програми. В третьому режимі в момент виконання кожної команди її програмний код підсвічується в таблиці пам'яті помаранчевим кольором. Це дає можливість спостерігати за загальним перебігом виконання машинного коду, за розміщенням у пам'яті замкнутих циклів та аналізувати загальну структурну організацію програми.

В цьому режимі користувач може обирати тактову частоту, з якою комп'ютерна модель процесора буде виконувати програму. Натисканням на кнопку «FREQ» він може переглянути доступні значення частоти, що відображаються на розташованому поряд із кнопкою дисплеї, та залишити бажане. Перелік частот знаходиться у файлі ***config.conf***, і за потреби його можна відредагувати та доповнити. Проте, варто пам'ятати, що програмна модель має певне обмеження по швидкодії, спричинене необхідністю формування відповідних графічних зображень за результатами виконання кожної команди. Тести показали, що максимальна швидкість роботи може сягати приблизно 30 команд за секунду. Такої продуктивності цілком достатньо для вивчення принципів дії процесора і навіть для створення анімаційних зображень, що програмним способом можуть формуватися відповідними цифровими значеннями в регістрах.

Для повернення програмної моделі в початковий стан достатньо натиснути кнопку «RES». При цьому всі регістри оперативного призначення (A, B, C і D) будуть очищені, на виходах всіх вузлів АЛП (окрім функціонального вузла інверсії) також встановляться нульові значення. До нульового стану буде очищений і лічильник команд, відповідно, в таблиці модуля пам'яті кольором виділиться клітинка, що відповідає нульовій адресі. Режим автоматичного виконання програми також буде вимкнуто.

Комп'ютерна модель процесора передбачає можливість ручного введення числових даних в межах одного байта. Для цього до регістра D, який побудований за принципом порту вводу-виводу, під'єднана віртуальна клавіатура, що складається із восьми кнопок з фіксацією положення. Їх стан алгоритмічно обробляється моделлю відповідно до схеми «монтажного І». Тобто, при зчитуванні даних з регістра D стан кожного розряду однобайтного числа, що надходить на шину даних, буде визначатися операцією кон'юнкції між значенням відповідного розряду числа, записаного в регістрі, та поточним станом відповідної кнопки віртуальної клавіатури. Це дозволяє користувачеві вводити зовнішні дані для програмної обробки моделлю процесора та змінювати їх при виконанні програми, якщо її алгоритм передбачатиме періодичний контроль стану віртуальної клавіатури. Змінити стан віртуальних кнопок моделі користувач може натисканням «гарячих кнопок» на клавіатурі комп'ютера («Q», «W», «E» і т.д. до «I») або за допомогою миші.

5 Особливості розробки.

Всі функціональні можливості, що описані у попередньому розділі, потребували розробки відповідних алгоритмів і написання зрозумілого та структурованого програмного коду. Весь код програми розділено на три великі класи або, інакше кажучи, три великі структури, а також включає в себе кілька допоміжних структур. На рисунку 5.1 зображена ієрархія класів програми.

Перший клас (EmyWindow), основний, відповідає за створення вікна програми та розміщення на ньому всіх необхідних компонентів. Він є найбільшим та найскладнішим, адже саме тут відбувається формування елементів графічного дизайну та надання функціональності таким компонентам, як кнопки управління. Також цей клас відповідає за основну логіку роботи програми. Він складається з функції обробки подій промальовування зображень, декількох слотів, що викликаються при натисканні кнопок (особливість QT [7]) та невеликої кількості допоміжних функцій, як наприклад, створення та читання файлу конфігурації.

Другий великий клас (DisplayWidget) відповідає за представлення модуля процесора. Його основними функціями є побудова графічних зображень всіх компонентів, написів, зберігання інформації про кольори. Тут же прописані функції, необхідні для аналізу командного слова та визначення подальших дій, які від цього залежать. Перший клас мав велику наповненість інтерфейсними компонентами для взаємодії з програмою, а цей клас містить більше інформації про поточний стан емулятора в цілому.

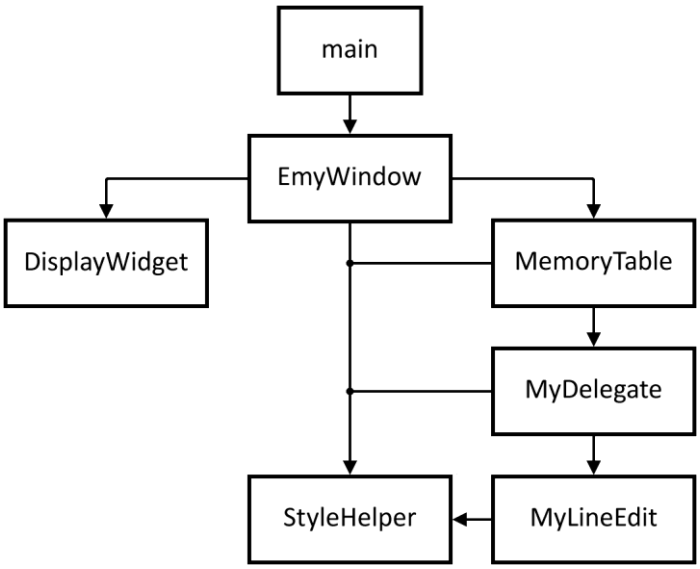

Рис 5.1 – Ієрархія класів програми

Третій великий клас (MemoryTable) відповідає за модуль пам'яті. Тут створювалась таблиця пам'яті, розроблявся її дизайн. Окрім цього, цей клас містить всі функції, які надають користувачеві можливість швидко і зручно працювати з пам'яттю, не витрачаючи час на зайві переміщення курсору, виправлення некоректно введених даних тощо. Для надання ергономічних якостей даному компоненту довелося створювати пару допоміжних класів (MyDelegate, MyLineEdit), які за розміром є невеликими, але досить важливими для того, щоб модуль пам'яті функціонував правильно і був зручним у роботі.

Окрім того, програма містить у своєму складі певні ресурси, а саме графічні зображення, які є статичними і не змінюються в процесі виконання програми, шрифт для написів та іконку самого застосунку. Це необхідно для коректного відображення графічних елементів на інших комп'ютерах та під різними операційними системами.

6 Висновки та плани на подальші дослідження

Розроблені комп'ютерні моделі процесора і програмної пам'яті, які реалізовані у створеному на їх основі застосунку, пройшли випробування навчальним процесом на факультеті комп'ютерних наук Харківського національного університету імені В.Н. Каразіна. Застосунок надавався студентам третього курсу під час проведення лабораторних робіт в режимі дистанційного навчання, а також для домашнього використання з метою проведення самостійних досліджень та виконання домашніх завдань. Робоче тестування програми здійснювалося протягом двох навчальних семестрів і дало хороші результати. За цей час були виявлені та виправлені певні недоліки, які в основному стосувалися роботи функцій графічного відображення деяких елементів та покращення ергономічних якостей застосунку. Студентами та викладачами робота створеної програми перевірена сумісно з операційними системами Windows XP, Windows 7, Windows 10 і Windows 11. Для тестування використовувалися як програми, що були написані раніше для реальної моделі процесора DPM08, так і нові програми, що створювалися студентами під час виконання лабораторних робіт. Зауважень щодо роботи функціональної частини емулятора не виявлено.

Таким чином, було досягнуто всіх поставлених цілей даного дослідження.

1. Створений застосунок повністю моделює роботу лабораторної моделі цифрового процесора DPM08, забезпечуючи виконання всіх інструкцій реального процесора.

2. Працездатність та коректна робота застосунку підтверджена всестороннім та тривалим тестуванням сумісно з усіма передбаченими технічним завданням операційними системами сімейства Windows. Продуктивності роботи емулятора цілком достатньо для навчальних цілей навіть при використанні застарілих комп'ютерів з одноядерними процесорами та тактовою частотою менше 1 ГГц.

3. Комп'ютерна модель програмної пам'яті допускає можливість зміни її максимального об'єму, що дозволяє використовувати емулятор для проведення усіх лабораторних робіт, передбачених навчальними програмами факультету.

4. Інтерфейс створеної програми дозволяє працювати з емулятором як з використанням миші, так і виключно за допомогою стандартної клавіатури комп'ютера. Емулятор надає можливість завантаження програмного коду в режимі ручного вводу, отриманого за рахунок трансляції асемблерної програми, або створеного іншими програмними засобами.

5. Побудовані комп'ютерні моделі надають можливість наочно демонструвати устрій та принцип дії простого процесора на рівні окремих регістрів та сигналів машинного циклу, відлагоджувати написані програми, акцентуючи при цьому увагу на важливих деталях та виконуваних процесах. Програма має зручний та зрозумілий інтерфейс управління. Застосунок активно використовується у навчальному процесі на кафедрі електроніки та управляючих систем Харківського національного університету імені В.Н. Каразіна під час проведення дистанційних лекцій та лабораторних робіт.

Отриманий досвід застосування створеного програмного забезпечення наштовхує на думку про подальше його вдосконалення. По-перше, варто забезпечити можливість роботи застосунку під UNIX-подібними операційними системами, оскільки деякі студенти саме їх використовують на своїх комп'ютерах. По-друге, варто реалізувати режим відображення завантажених у регістри даних не в числовому бінарному вигляді, а за допомогою імітації зображення ввімкнутих та вимкнутих світлодіодів, як це зроблено на реальній моделі. Без жодних сумнівів це додасть реалістичності програмній моделі, особливо, при створенні та відлагодженні анімаційних програм. І нарешті, деякі невеликі вдосконалення моделі процесора без суттєвого його ускладнення дозволять значно розширити функціональність моделі, наблизивши її за можливостями до промислових процесорів. При цьому програма лабораторного практикуму може бути суттєво розширена новими цікавими лабораторними роботами.

ЛІТЕРАТУРА

1. Журавель Ю. А. Модель цифрового процесора / Ю. А. Журавель, С. Н. Рева // Вісник Харківського національного університету імені В.Н. Каразіна. Серія : Математичне моделювання. Інформаційні технології. Автоматизовані системи управління. - 2011. - № 987, Вип. 18. - С. 5-18. URL: http://nbuv.gov.ua/UJRN/VKhIMAM_2011_987_18_3 (дата звернення: 15.02.2023)
2. UMD Computer Science Department. Turbo Debugger for Windows. URL: https://www.d.umn.edu/~rmaclin/cs1621/debugger.html (дата звернення: 15.02.2023)
3. Proteus: PCB Design & Simulation Made Easy. URL: https://www.labcenter.com/ (дата звернення: 15.02.2023)
4. ARM Keil: About µVision. URL: https://www.keil.com/support/man/docs/uv4cl/uv4cl_overview.htm (дата звернення: 15.02.2023)
5. Bjarne Stroustrup. The C++ Programming Language. Fourth Edition. Addison-Wesley. 2013. 1346 pages.
6. QT Framework Official Website. URL: https://www.qt.io/product/framework (дата звернення: 15.02.2023)
7. QT Documentation: Signals & Slots. URL: https://doc.qt.io/qt-6/signalsandslots.html (дата звернення: 15.02.2023)

REFERENCES

1. Zhuravel Y. Digital processor model / Y. Zhuravel, S. Reva // Bulletin of V. N. Karazin Kharkiv National University. Series : Mathematical Modeling. Information Technology. Automated Control Systems. - 2011. - № 987, Volume 18. - P. 5-18. [in Russian] URL: http://nbuv.gov.ua/UJRN/VKhIMAM_2011_987_18_3 (Last accessed: 15.02.2023)
2. UMD Computer Science Department. Turbo Debugger for Windows. URL: https://www.d.umn.edu/~rmaclin/cs1621/debugger.html (Last accessed: 15.02.2023)

3. Proteus: PCB Design & Simulation Made Easy. URL: https://www.labcenter.com/ (Last accessed: 15.02.2023)
4. ARM Keil: About µVision. URL: https://www.keil.com/support/man/docs/uv4cl/uv4cl_overview.htm (Last accessed: 15.02.2023)
5. Bjarne Stroustrup. The C++ Programming Language. Fourth Edition. Addison-Wesley. 2013. 1346 pages.
6. QT Framework Official Website. URL: https://www.qt.io/product/framework (Last accessed: 15.02.2023)
7. QT Documentation: Signals & Slots. URL: https://doc.qt.io/qt-6/signalsandslots.html (Last accessed: 15.02.2023)

Software emulator of the digital processor training model

Reva Sergiy *Candidate of Technical Science, Associate Professor of the Department of Electronics and Control Systems; Faculty of Computer Science; V. N. Karazin National University, Svobody Sq 6, Kharkiv, Ukraine, 61022*
e-mail: iec-lab@karazin.ua;
https://orcid.org/0000-0002-2615-9226

Komerystyi Vladislav *bachelor student V.N. Karazin National University, Svobody Sq 6, Kharkiv, Ukraine, 61022*
e-mail: xa12850420@student.karazin.ua
https://orcid.org/0009-0003-2375-5190

The paper deals with the problems of organizing practical and laboratory work in the educational process in higher educational institutions of Ukraine in the context of distance learning. The reason for these problems is the lack of the possibility of conducting classroom classes with the use of appropriate visual equipment and laboratory equipment in the current conditions. The development and creation of software emulators and interactive learning applications is considered as one of the possible methods of solving this issue.

The purpose of the work is to improve the quality of the educational process in the study of the basics of microprocessor technology through the development and use of a computer model of a learning processor.

The article provides a brief review and analysis of existing samples of software emulators of digital processors that work independently or as part of integrated software development environments, identifies their shortcomings in relation to the use in the educational process, proposes a new approach to design and software implementation, and formulates the main technical requirements for a computer model. The structure of the created software application and its user interface are considered, and the features of the software implementation are described. The main difference between the created emulator and existing analogues is an extended graphical representation of the internal structure of the processor, as well as animated indication of signals and processes occurring during its operation.

The software implemented computer model allows executing a program placed in virtual memory in one of three modes. The first mode provides step-by-step execution of commands, clearly demonstrating with the help of graphical means the formation of internal processor control signals and changes in the states of its individual nodes throughout the entire machine cycle. The second mode allows you to gradually execute the program one command at a time, displaying the processor status only after each command is completed. The third mode is designed for automatic continuous program execution at a preset speed.

The results of testing and trial operation of the created emulator within the framework of distance learning at the Faculty of Computer Science of V. N. Karazin Kharkiv National University are also presented.

The results are summarized and the prospects for further improvement and expansion of the possibilities of using this model in the educational process are considered.

Keywords: *software emulator, computer modeling, algorithmic model, digital processor, user interface, educational process.*

Надійшла у першій редакції 18.04.2023, в останній - 19.05.2024.