УДК 681.51

АДАПТАЦІЯ АЛГОСТРУКТУРНОГО МЕТОДУ ДЛЯ ПРОЕКТУВАННЯ ЦИФРОВИХ ПРИСТРОЇВ КОМП'ЮТЕРНИХ СИСТЕМ КЕРУВАННЯ

Г.М. Смірнов, викладач
Східноукраїнський національний університет ім. В. Даля

Розглядається питання адаптації існуючого алгоструктурного методу проектування систем керування з метою розширення галузі його застосування. Аналіз існуючого варіанта алгоструктурного методу дозволяє зробити висновки про наявність в ньому положень, що дозволяють ефективне проектування лише програмної складової комп'ютерних систем керування. Запропонований новий адаптований алгоструктурний метод базується на новому, протилежному погляді на побудову інформаційної моделі та потоки інформації і дозволяє проектувати як програмну, так і апаратну складові комп'ютерних систем керування. Запропонований порядок опрацювання модельних даних дозволяє вийти на новий рівень адекватності комп'ютерної моделі проектованої системи.

ВСТУП

Актуальність. Автоматичні системи керування значно підвищують продуктивність виробництва та забезпечують можливість реалізації таких технологічних процесів, які без застосування автоматизації взагалі не могли б існувати. Ефективний синтез автоматичних систем керування являє собою задачу, що рідко має тривіальне розв'язання. Через це розроблення нових методів та інструментальних систем, які забезпечують автоматизацію проектування апаратних засобів систем керування і алгоритмів функціонування таких систем, а також роблять цей процес ефективнішим, є актуальною.

Постановлення питання. Алгоструктурна технологія проектування у своєму первісному варіанті [1, 2] може використовуватися для автоматизації створення програмного забезпечення комп'ютерних систем керування. Нижче аналізуються основні властивості згаданої технології і висуваються пропозиції щодо її вдосконалення і спрямовані на розширення галузі застосування. Запропонований адаптований варіант алгоструктурної технології стає придатним для проектування цифрових пристроїв та логіки керування.

Короткі відомості про історію питання. Вперше такі поняття, як алгоструктура та алгоструктурний метод проектування, було використано в працях [1, 2]. У цих працях зазначено, що основною галуззю застосування алгоструктурного методу проектування є автоматизація процесу розроблення програм для комп'ютерних систем керування.

В основі алгоструктурного методу проектування лежить ідея замінити процес розроблення прикладного програмного забезпечення комп'ютерних систем керування, виконуваний за допомогою традиційних методів програмування, адекватним процесом вибору алгоструктур, їх комутації і подальшим перетворенням в тексти відповідного прикладного програмного забезпечення [1].

АНАЛІЗ КЛАСИЧНОГО АЛГОСТРУКТУРНОГО МЕТОДУ ТА ФОРМУЛЮВАННЯ ОСНОВНИХ ЗАСАД АДАПТОВАНОГО АЛГОСТРУКТУРНОГО МЕТОДУ

Під алгоструктурами в [1, 2] пропонується розуміти алгоелементи та алгоблоки. Алгоелемент – це певний гіпотетичний об'єкт, котрий має у

своєму складі входи і виходи та виконує визначені функції, пов'язані з перетворенням вхідних величин на вихідні. Алгоблок – це гіпотетичний об'єкт, побудований на базі декількох алгоелементів (алгоблоків) [1].

Для практичного застосування алгоструктурного методу проектування потрібен відповідний інструментарій, призначений забезпечувати процедуру проектування. Таким інструментом може бути комп'ютерна програма, здатна створювати віртуальне втілення гіпотетичних об'єктів - алгоструктур, та маніпулювання ними. Одну з можливих реалізацій згаданої програми наведено в [2], – це САПР AlgoCAD. САПР AlgoCAD забезпечує створення, редагування, налаштування, моделювання поведінки алгоструктур та трансляцію логіки, закладеної в алгоструктурі у програму мовою програмування високого рівня.

На практиці розроблення алгопроекту полягає в наступному. Створюється алгоблок (алгоструктура), який репрезентує проектовану керуючу програму вцілому. Певно, що керуюча програма має отримувати дані для обробки. Такими даними можуть бути, наприклад, оцифровані покази датчиків з об'єкта керування. Отже, створена алгоструктура повинна мати потрібну кількість інформаційних входів. Аналогічно, оскільки керуюча програма має генерувати значення величин, які мають втілитися у керувальні діяння, слід передбачити потрібну кількість виходів алгоблока. Залежно від обставин алгоструктура може містити і наладнувальні параметри. Наприклад, для програми, яка керує приземленням космічного корабля як наладнувальні параметри можуть розглядатися значення величин, що характеризують умови приземлення, такі, як щільність атмосфери, пришвидшення вільного падіння, швидкість вітру тощо. Наявність таких наладнувальних параметрів додасть керуючій програмі гнучкості та універсальності і дозволить використати програму під час посадки в різних умовах. Таким чином, створення будь-якої алгоструктури починається із створення головного структуротвірного алгоблока та визначення його інформаційного інтерфейсу.

Подальшим кроком у створенні програм керування за допомогою алгоструктур є визначення функціональності алгоструктури. Проектування функціональності виконується за компонентним методом. Для цього до складу алгоблока, створеного на попередньому етапі проектування, додаються алгокомпоненти з бібліотеки, за допомогою яких буде реалізовано поведінку алгоблока. Далі алгокомпоненти потрібним чином налаштовуються і поєднуються між собою та з головним блоком за допомогою інформаційних зв'язків.

Алгокомпоненти, які містяться у бібліотеці, поділяються на алгоелементи та алгоблоки. Алгоелементи – це неподільні компонентні утворення вбудовані в бібліотеку, кожен з яких виконує заздалегідь визначену функцію, хоча й припускає певне налаштування. Алгоблоки, що містяться в бібліотеці, – це багатоелементні утворення зі складною структурою, які репрезентують певні алгоструктурні рішення, попередньо створені і збережені у бібліотеці для можливого подальшого багаторазового використання.

Алгоелементи, в свою чергу, поділяються на обчислювальні та керуючі. До обчислювальних алгоелементів, в першу чергу, слід зарахувати алгокомпонент «Expression», призначений для виконання обчислень згідно з аналітичним виразом, який визначається проектувальником під час налаштовування конкретного екземпляра алгокомпонента «Expression». Як операнди тут можуть бути імена входів, виходів та наладнувальних параметрів алгокомпонента «Expression», літеральні константи, зазначені безпосередньо у аналітичному виразі, або виклики стандартних математичних функцій.

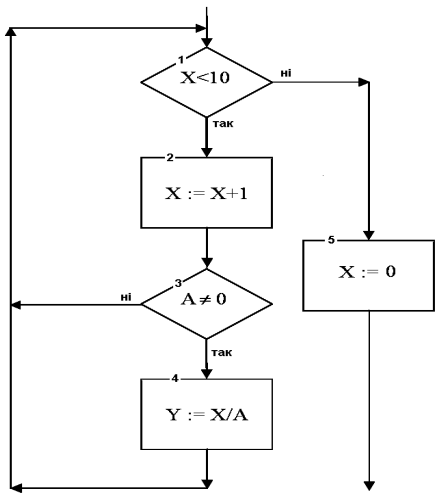
Крім алгокомпонента «Expression», у новіших версіях САПР AlgoCAD наявні ще декілька алгоелементів обчислювального типу, які реалізують певну специфічну функціональність, яку неможливо або важко реалізувати за допомогою алгокомпонента «Expression». Це такі алгокомпоненти як FIFO, RAM, ROM, TIMER, BUS_IN, BUS_OUT, PORT_IN, PORT_OUT. Докладніше про призначення та особливості застосування згаданих компонентів можна довідатися з [3].

Принагідно слід зазначити, що алгоблок, коли його повністю розроблено і збережено у бібліотеці, також може розглядатися як алгокомпонент обчислювального типу в разі його використання у проектах.

До групи керуючих належать алгоелементи, призначенням яких є керування порядком опрацювання процесором ЕОМ послідовності обчислювальних алгоелементів та алгоблоків. До цієї групи належать такі компоненти, як IF..ELSE..END, CASE..ELSE..END, WHILE..END, DO..WHILE, FOR..NEXT, EXIT, RETURN. Докладніше про призначення та особливості застосування зазначених компонентів можна довідатися з [1]. Зауважимо лише, що згадані компоненти повністю відображають бачення проблеми керування обчислювальним процесом з точки зору мов програмування високого рівня. Це й не дивно, якщо згадати, що метою розроблення алгоструктурної технології є автоматизація проектування прикладного програмного забезпечення комп'ютерних систем керування. В той самий час, тільки ця обставина й не дозволяє ефективно застосовувати традиційну алгоструктурну технологію до розв'язання задач проектування апаратної частини систем керування і цифрових пристроїв зокрема.

Адаптованість алгоструктурної технології до розв'язання задач проектування програмного забезпечення полягає у можливості майже прямого відображення елементів алгоструктури у конструкції певної мови програмування. Відображення алгоструктур у апаратні конструкції є значно складнішим, що дозволяє ставити питання про ефективність застосування алгоструктурної технології проектування (у її традиційному розумінні [1, 2, 3]) для цього класу задач.

Підтвердимо сказане на прикладі. Припустимо, що для реалізації процедури керування потрібно виконати обчислення за таким алгоритмом (фрагмент схеми алгоритму наведено нижче). Для реалізації наведеного алгоритмічного фрагменту алгоструктурними засобами слід створити алгоблок з входами «X» та «A», а також виходом «Y».




```
WHILE          //X < 10
EXPRESSION     //X := X + 1
IF             //A <> 0
EXPRESSION     //Y := X / A
END            //END IF
END            //END WHILE
EXPRESSION     //X := 0
```

Далі слід розмістити в алгоблоці алгоелементи, які забезпечать виконання потрібних обчислень. Відповідну послідовність алгоелементів (з необхідними коментарями) наведено поруч з фрагментом схеми алгоритму.

Далі, кожен з алгоелементів має бути відповідним чином налаштовано. Процедура налаштовування передбачає визначення потрібної кількості входів та виходів, а також визначення особливостей функціональності (визначається за допомогою аналітичного виразу). Наприклад, для алгоструктури, що розглядається, алгокомпонент WHILE має отримати один вхід – «X» і бути налаштованим на обчислення логічного виразу (умови повторювання) – «X<10»; другий з алгокомпонентів типу «EXPRESSION» повинен мати два входи «X» та «A», а також вихід «Y» і бути налаштованим на обчислення виразу «Y := X/A». Решта алгокомпонентів має бути налаштовано аналогічним чином. Налаштовування не потребують хіба що два алгокомпоненти «END», котрі просторово обмежують дію алгокомпонентів WHILE та IF і таким чином у неявний спосіб визначають алгоритмічні залежності між алгокомпонентами. Під алгоритмічною залежністю будемо розуміти той факт, що один з алгокомпонентів (у нашому випадку WHILE або IF) може «дозволити» або «заборонити» працювати іншому алгокомпоненту або групі алгокомпонентів. Крім того, наявність алгоритмічної залежності передбачає також цілковиту визначеність порядку, у якому алгокомпоненти опрацьовуються і загальна логіка функціонування алгоструктури, у загальному випадку залежить від цього порядку.

Подальшим кроком у створенні алгоструктури є визначення інформаційних зв'язків між алгокомпонентами. Тобто потрібно зазначити для кожного із входів кожного алгокомпонента, до якого з виходів його підключено.

На цьому процес створення алгоструктури можна вважати в основному завершеним. Лишається лише перевірити її правильність шляхом моделювання і виконати трансляцію у одну з алгоритмічних мов високого рівня.

Аналізуючи наведений фрагмент схеми алгоритму та послідовність алгокомпонентів, які реалізують означену функціональність, можна зауважити, що послідовність алгокомпонентів дуже нагадує послідовність операторів мови програмування високого рівня для реалізації зазначеного алгоритму, що полегшує трансляцію логіки, закладеної у алгоструктурі, до програм у термінах алгоритмічних мов. Множина алгокомпонентів дуже просто відображається на множину операторів, які є структурними елементами комп'ютерних програм.

Апаратні системи (системи керування, а також цифрові пристрої, які можуть належати до таких систем) теж можна розглядати як системи складені з певних структурних елементів, але тут структурні елементи вже не є операторами певної мови програмування, отже і поводитися з ними треба інакше. У разі, коли розглядається (проектується) логіка роботи системи керування, то найчастіше оперують такими поняттями, як елементарні динамічні ланки. Наприклад, такими ланками можуть бути: інтегруюча ланка, аперіодична ланка тощо. А на більш високому рівні оперують функціональними елементами, такими, як регулятор, підсилювач тощо. У разі, коли розглядається (проектується) логіка роботи цифрового пристрою, то маємо справу з елементарними логічними ланками, а на більш високому рівні – цифровими функціональними елементами: тригер, регістр, комбінаційна схема, процесор, пам'ять та інші.

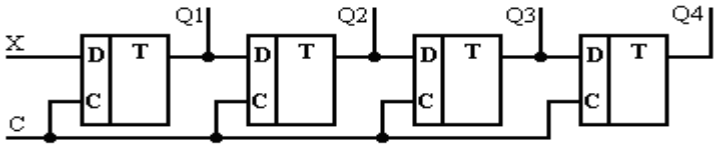
Вочевидь, елементи, з яких складаються системи керування та цифрові пристрої за своєю природою, властивостями та логікою застосування значно відрізняються від операторів мов програмування. Взяти хоча б порядок їх роботи. У мовах програмування оператори мають

виконуватися один за одним – послідовно. Крім того, у програмах існують поняття про поточний оператор, управління та передачу управління (зокрема так реалізуються циклічні алгоритми опрацювання інформації). Завдяки цьому різні оператори програми під час її прогону, можуть бути виконані різну кількість разів. Наприклад, наведена вище схема алгоритму (якщо на початку X=0) передбачає, що блок №5 обов'язково має виконатися один раз, блок №2 виконається десять разів, а блок №4 – або жодного разу не виконається, або виконається десять разів (залежно від значення «A»).

В апаратних пристроях під час їх роботи не існує поняття про поточний структурний елемент (чи то оператор, чи то блок). Так само не існує поняття про управління та його передачу в тому сенсі, в якому вони існують у програмних системах. Натомість, у більшості цифрових пристроїв робота синхронізується тактаовими імпульсами, і всі підсистеми (структурні одиниці, наприклад мікросхеми) спрацьовують паралельно і одночасно. Хоча функціональність конкретної мікросхеми і може суттєво різнитися в різні моменти часу (наприклад, в певні моменти на її функціональність може впливати сигнал вибору кристала «CS»), але мікросхема однаково в ці цикли спрацьовує, хоча її робота може зводитися до простого очікування. Реалізація циклічних алгоритмів може бути і апаратною, але в такому разі ітерації розподіляються між системними тактами, а решта елементів системи можуть тимчасом очікувати завершення цього процесу, знаходячись в неактивному стані (саме тут може бути простір для організації паралельних обчислень).

Через відмінність принципів організації обробки інформації в апаратних і програмних системах у разі використання алгоструктур для проектування цифрових пристроїв постає питання про можливу неадекватність отриманих алгоструктурних моделей. Наведемо приклад такої неадекватності.

Припустимо, що потрібно реалізувати алгоструктурну модель регістра зсуву. Відомо, що з точки зору схемотехніки регістр зсуву являє собою декілька тригерів, причому інформаційний вхід кожного з них підключається до інформаційного виходу попереднього, як це зображено на такій схемі:



У реальних пристроях всі тригери працюють паралельно в часі. При цьому кожен з тригерів спрацьовує не миттєво, а має скінчений час спрацьовування. Тобто результат на його інформаційному виході з'являється з деяким запізненням після отримання інформації на вході. Внаслідок цього кожен з тригерів, що входять до регістра, під час приходу синхроімпульсу у момент часу T приймає на вході інформацію з виходу попереднього тригера, вироблену на попередньому такті T-1. Нову інформацію, яка відповідає моменту часу T, тригери вироблять пізніше, паралельно і з деякою затримкою.

Тепер уявімо, що було побудовано алгоструктурну модель регістра зсуву, де кожен з тригерів репрезентовано як алгоелемент чи алгоблок. Якщо виконувати моделювання поведінки, отриманої алгоструктури за послідовною схемою організації обчислень, то отриманий результат не відповідатиме логіці роботи регістра зсуву. Оскільки алгокомпоненти моделі опрацьовуються послідовно, то на момент початку опрацювання поведінки i-го тригера, всі розрахунки, які стосуються попереднього (i-1)-го тригера вже повністю завершено і на інформаційному виході

цього (i-1)-го тригера вже міститься інформація, яка відповідає моменту часу T, а не T-1. Таким чином, фактично за один такт відбувається наскрізна передача інформації від першого тригера регістра до останнього. Зрозуміло, що таку модель важко вважати адекватною. Звісно, що в окремих випадках можна знайти можливість, щоб за допомогою певних перетворень досягти адекватності. Так, для розглянутого випадку можна призначити порядок опрацювання алгокомпонентів зворотний відносно порядку інформаційних зв'язків, тобто в першу чергу опрацьовувати останній тригер, а в останню чергу перший.

Думається, що для багатьох проектів таки можна знайти аналогічну можливість досягти адекватності, якщо добре обміркувати порядок обчислень алгокомпонентів. Проте в загальному випадку така можливість не гарантується, бо вона не підтримується самою алгоструктурною технологію проектування, і нема жодних гарантій, що в разі проектування системи значної складності правильне рішення щодо порядку опрацювання алгокомпонентів завжди буде легко знайти і обґрунтувати.

Таким чином, аналізуючи можливості алгоструктурної технології проектування слід зробити висновки, що зазначена технологія може успішно застосовуватися для проектування програмного забезпечення, в тому числі і для комп'ютерних систем керування. В той самий час існують певні особливості фундаментального характеру, які перешкоджають ефективному використанню алгоструктурного методу для проектування систем керування та цифрових пристроїв. Більше того, відомо, що до систем керування висуваються підвищені вимоги щодо надійності через потенційну небезпеку техногенних катастроф, яку несуть помилки в управлінні. В той самий час недоліки розглянутої алгоструктурної технології проектування можуть стати джерелом помилок, виявити які не завжди просто. Фактично, сама технологія провокує появу таких помилок.

Отже, постає питання про адаптацію алгоструктурної технології проектування з метою забезпечення ефективного проектування цифрових пристроїв для комп'ютерних систем керування. Фактично потрібно створити нову технологію проектування (оскільки йдеться про фундаментальні відмінності), яка, з одного боку, запозичувала б позитивні риси алгоструктурної технології, а з іншого боку, дозволила б ефективно створювати адекватні інформаційні моделі комп'ютерних систем керування. Таким чином, на відміну від розглянутого [1, 2] алгоструктурного методу проектування, який, для зручності, зватимемо «класичним», пропонується «адаптований» алгоструктурний метод проектування. Щоб розрізняти алгоструктури, створені за різними методами, будемо їх називати відповідно «класичними» і «адаптованими». Розглянемо адаптований алгоструктурний метод докладніше.

Адаптований алгоструктурний метод проектування також реалізує компонентний підхід до процедури проектування. Проект так само складається з алгокомпонентів, що мають входи, виходи і певну функціональність.

В той самий час з метою забезпечення можливості ефективно створювати адекватний інформаційний образ апаратної системи замість алгоритмічної залежності, яка мала місце між алгокомпонентами у класичній алгоструктурі і стала на заваді створенню адекватних моделей апаратних пристроїв, вводиться поняття структурної залежності.

Щоб визначити поняття структурної залежності, згадаємо про модульний принцип побудови. Модульний принцип – є загально-технічним і застосовується для побудови та аналізу складних систем.

Коли певну систему через складність важко створити безпосередньо, її будують з підсистем – модулів, проектування яких вже має бути простішим. В свою чергу, складні підсистеми також можуть проектуватися за модульним принципом. І так далі. Таким чином, маємо справу з деревоподібною структурою, де кожна нетермінальна вершина являє собою систему (підсистему) і має структурну побудову з підлеглих підсистем.

Зазначимо, що хоча сама по собі деревоподібна побудова не забороняє мати впорядкованість підлеглих модулів на кожному рівні ієрархії, але в нашому випадку (під час проектування логіки роботи апаратних модулів) ми свідомо вводимо можливість такого обмеження. Тобто оскільки в апаратних системах всі підсистеми в загальному випадку є рівнозначними, то множина алгокомпонентів одного рівня ієрархії є невпорядкованою.

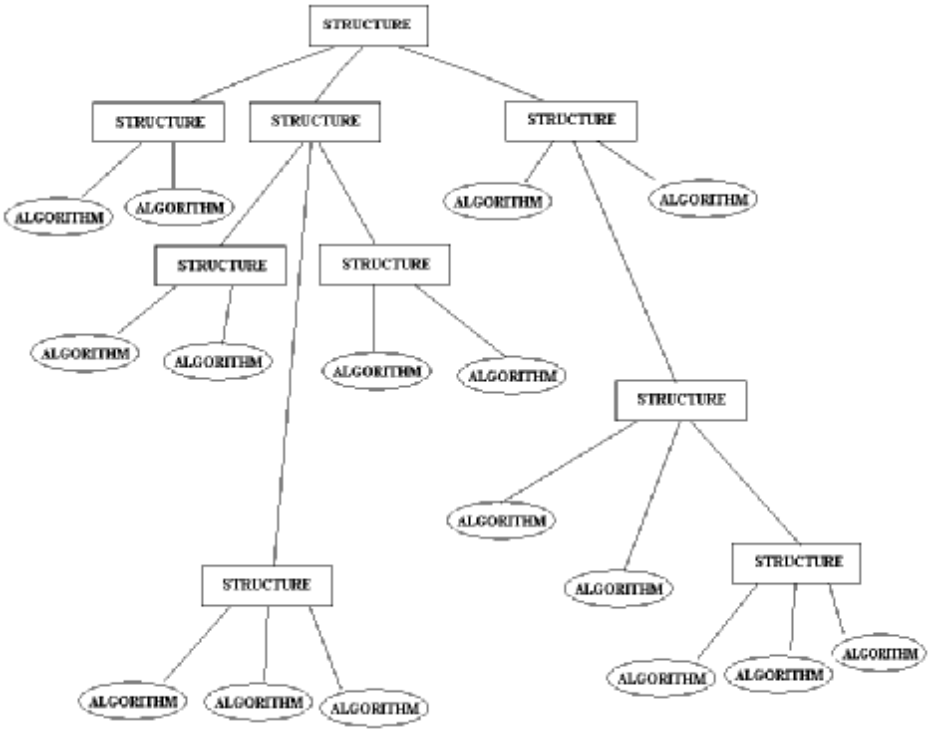
Таким чином, під структурною залежністю будемо розуміти таку залежність між алгокомпонентами, коли одні компоненти не керують (як у програмах), а володіють іншими – підлеглими алгокомпонентами. Структурна залежність визначає швидше структурну побудову проектованої системи (проектованого пристрою) і кожної з підсистем, ніж управління.

Отже, адаптований алгоструктурний метод проектування передбачає наявність не алгоритмічних, а структурних зв'язків між алгокомпонентами. А відповідно відпадає потреба у алгоелементах, котрі керують обчислювальним процесом («IF», «WHILE» тощо). Основним структуроутворним алгокомпонентом у адаптованій алгоструктурі є алгоблок. Тепер його роль значно підвищено (раніше він відігравав роль операторних дужок, які у мовах програмування визначають складний оператор). Тому є сенс дати цьому компоненту нову назву, яка більше відповідатиме його призначенню, а саме – «STRUCTURE». Проектування тепер відбуватиметься за методом «згори-донизу». В процесі проектування буде будуватися деревоподібна структура з алгокомпонентів. Екземплярами алгокомпонента «STRUCTURE» відтепер буде репрезентовано і саму проектовану систему та її структурні складові аж донизу за ієрархією (всі нетермінальні вершини дерева).

Окрема розмова про термінальні вершини. У класичному алгоструктурному методі термінальними (неподільними) були алгоелементи типу «EXPRESSION» та інші, додаткові. Відповідно до ролі, яку вони виконували (оператори мови програмування), їхня функціональність була розвинена недостатньо. Адаптований алгоструктурний метод передбачає розширення функціональних можливостей алгоелемента «EXPRESSION» з таким розрахунком, щоб він міг виконувати не лише прості обчислення за формулами, а й невеликі програмні фрагменти. Для написання таких програмних фрагментів розроблено нескладну мову програмування, яка є поєднанням певних підмножин мов «C» та «Pascal». Зокрема є можливість вводити нові (локальні щодо компонента) змінні та масиви, організовувати обчислення за допомогою операторів циклу та розгалуження, викликати стандартні арифметичні функції, звертатися до системних ресурсів (порти вводу-виводу, годинник). Вочевидь є сенс дати цьому компоненту нову назву, яка більше відповідатиме його теперішньому призначенню, а саме – «ALGORITHM». Схематично, адаптована алгоструктура виглядатиме наступним чином (див. наступний рисунок) .

Отже, адаптований алгоструктурний метод запроваджує принципово відмінний спосіб модельного відображення проектованого об'єкта. Якщо проект, побудований за допомогою класичного алгоструктурного методу можна кваліфікувати як алгоритм, який реалізується за допомогою структурних засобів, то проект, побудований за допомогою адаптованого

алгоструктурного методу, можна кваліфікувати як структуру, утворену багатьма алгоритмами. В обох випадках для позначення спроектованої сутності правомірним є вживання терміну «алгоструктура». В той самий час сенс, який стоїть за цим терміном, в кожному з випадків є відмінним.



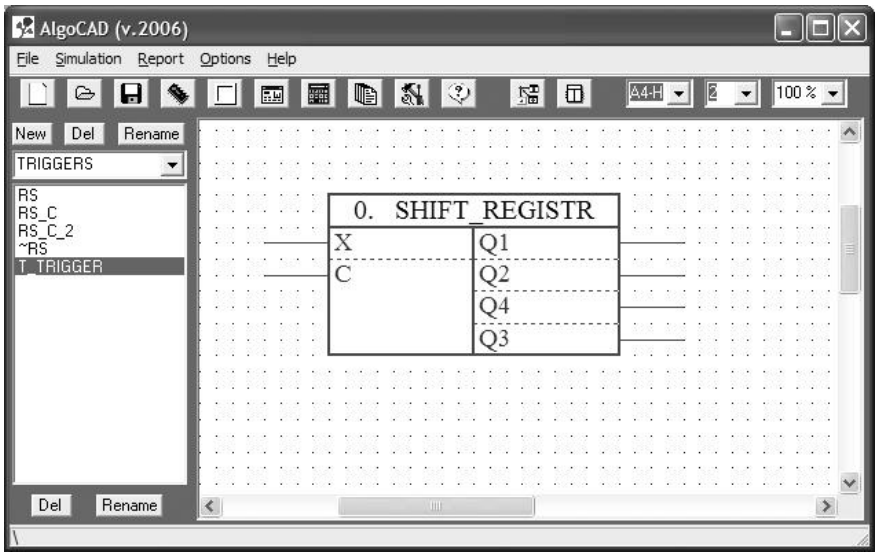
Нове визначення алгоструктури покликане значно змінити ідеологію проектування, що повною мірою відповідає розширенню галузі застосування. Якщо проектувальник програмного забезпечення, який користується класичним алгоструктурним методом, має міркувати в термінах побудови алгоритму, то проектувальник систем керування та цифрових приладів, який користується адаптованим алгоструктурним методом, має в першу чергу мати на думці структуру проектованої системи та її підсистем.

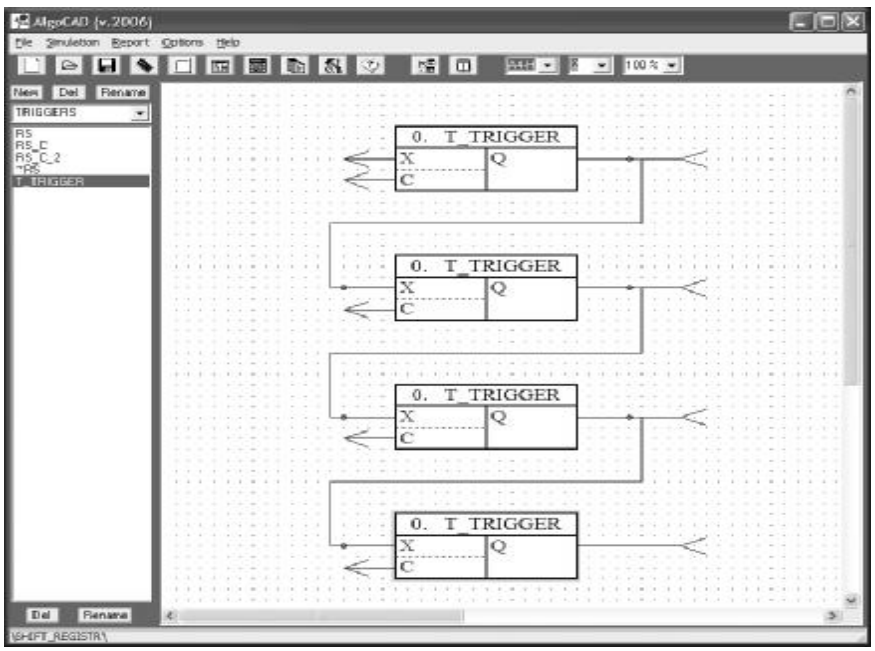
Крім того, нове визначення алгоструктури робить процес проектування значно гнучкішим. Мається на увазі той факт, що тепер проектувальник завжди має можливість вибору між структурним та алгоритмічним підходами до проектування. Одну і ту саму задачу можна тепер розв'язати багатьма способами. Саме проектувальник визначає, які частини проекту доцільно проектувати як структуру, а які частини проекту слід запрограмувати. Тобто сам проектувальник визначає баланс між структурною і алгоритмічною складовими проекту.

Для практичного застосування адаптованого алгоструктурного методу автором розроблено відповідний інструментарій, призначений забезпечувати процедуру проектування. Таким інструментом є комп'ютерна програма – САПР AlgoCAD-2006.

САПР AlgoCAD-2006 забезпечує створення, редагування, налаштовування та дослідження (шляхом моделювання) поведінки саме адаптованих алгоструктур, які можуть бути втіленням інформаційних моделей як апаратних засобів комп'ютерних систем керування, так і керуючих програм. Для збереження можливості проектування програмної складової згадана програма залишає користувачеві можливість в разі

потреби визначати впорядкованість компонентів на кожному (або деяких) з рівнів ієрархії. Цим зберігається наступність відносно попередніх версій системи AlgoCAD. Крім того, в нову версію AlgoCAD автором було внесено чимало вдосконалень і нових функціональних можливостей, відсутніх у попередніх версіях. Зокрема, було кардинальним чином змінено підхід до самої процедури проектування шляхом впровадження багатофункціонального графічного інтерфейсу, що забезпечує можливість візуального проектування, яка була відсутня у попередніх версіях. Це впровадження значно підвищило ефективність роботи проектувальника. Також значних змін зазнав і алгоритм роботи підсистеми моделювання. Відповідно до нового визначення поняття алгоструктури, підсистема моделювання здатна виконувати послідовне, паралельне та змішане опрацювання компонентів моделі, на відміну від лише послідовного у попередніх версіях.

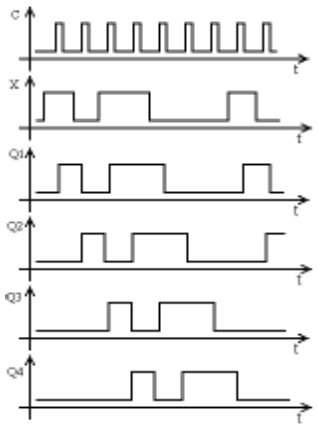


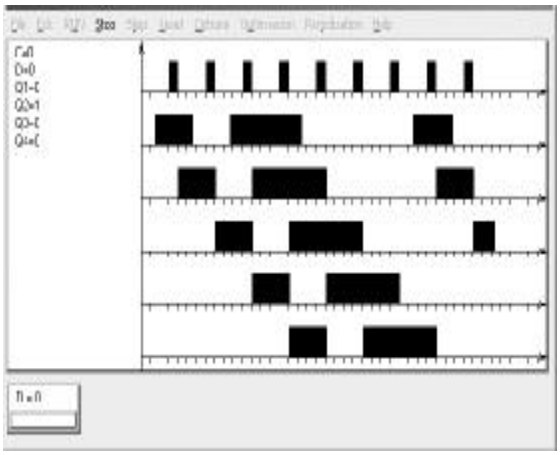


Ефективність запропонованого методу проектування підтверджується експериментально. Так, наступні два рисунки репрезентують процес створення алгоструктурної моделі регістра зсуву на чотири розряди. (Особливості моделювання його роботи було проаналізовано вище.)

На першому рисунку представлено умовне зображення титульного алгокомпонента (компонент «STRUCTURE»), а на другому – його структурна побудова (з компонентів «ALGORITHM»).

Адекватність зазначених адаптованих моделей підтверджується шляхом дослідження їх поведінки під час моделювання та побудови часових діаграм, що відображають цю поведінку. Так, на двох наступних рисунках зображено ідеальну часову діаграму роботи чотирирозрядного регістра зсуву і часову діаграму, отриману експериментально в системі AlgoCAD-2006 під час моделювання поведінки алгоструктури, яка репрезентує логіку роботи згаданого регістра.





ВИСНОВКИ

Таким чином, адаптований алгоструктурний метод, запропонований автором, забезпечує можливість ефективного застосування алгоструктур не лише для проектування програмної частини комп'ютерних систем керування, а й для проектування їх апаратної частини та логіки керування.

SUMMARY

The adaptation of existing algostructured control system designing method to enlargement it's usage area is proposed in this article. The analysis of original algostructured system designing method show some reasons witch limit the usage of this method by designing control programs area only. Offered new adapted algostructure method based on original view on the system model data structures and data streams flows and allow to design as software as hardware of computer-based control systems. Offered the different (in comparing with original algostructured method) order of model data processing which make able to achieve a much level identity between of a designed system and of it's algostructured model behaviour.

СПИСОК ЛІТЕРАТУРИ

1. А.Ф. Горбатюк, А.В. Бешкарев. Применение алгоструктурной технологии в компьютерных системах управления. – Луганск: ВУГУ, 2000.
2. Горбатюк А.Ф. САПР прикладного программного обеспечения компьютерных систем управления AlgoCAD. – Луганск: Восточноукр. гос. ун-т, 1999. – 189 с.
3. А.Ф. Горбатюк, С.А. Горбатюк, Г.Н. Смирнов. Алгоструктурная технология проектирования и программирования компьютерных систем управления // Вестник ХГТУ. - 2002. - №1(14)

Надійшла до редакції 26 жовтня 2006 р.