УДК 621.38.061(075.8)

Л. Ф. Мараховський
Ю. О. Романок

ПИТАННЯ ПОБУДОВИ МІКРОПРОЦЕСОРІВ, ЩО РЕКОНФІГУРУЮТЬСЯ

У статті розглянуті питання: принцип ієрархічного програмного управління; розширення структури команд за рахунок введення додаткових полів загального коду; етап логічного проектування пристрою управління, що реконфігурується; опис поліпрограм в ієрархічному автоматі, що дає можливість будувати на їхній основі нові реконфігуровані мікропроцесори.

В статье рассмотрены фундаментальные вопросы: принцип иерархического программного управления; расширение структуры команд за счет введения дополнительных полей общего кода; этап логического проектирования реконфигурируемого устройства управления; описание полипрограмм в иерархическом автомате, что дает возможность строить на их основе новые реконфигурируемые микропроцессоры.

The article describes the fundamental issues : the principle of hierarchical management software ; extension command structure , by introducing additional fields of common code ; reconfigurable logic design stage of the control; description poliprogram hierarchical machine that gives you the opportunity to build on their basis new reconfigurable microprocessors.

Ключові слова: принцип ієрархічного програмного управління, структура команд мікропроцесора, реконфігурований пристрій управління, опис поліграм для ієрархічних автоматів

Вступ. Алгоритмом називається система формальних правил, де чітко й однозначно визначається процес виконання заданної роботи [1, 2]. Обробка дискретної інформації передбачає попереднє складання алгоритмів (обчислень, моделювання, логічних перетворень тощо), що визначають послідовність перетворення цифрової інформації.

Це пов’язане з тим, що в сучасних комп’ютерних системах використовується принцип програмного управління, запропонований Ч. Беббіджем в 30-х роках XIX ст., а також застосовуються двійкові елементарні схеми пам’яті, у яких стани запам’ятовуються при одному вхідному сигналі, який в теорії алгоритмів називається «порожнім словом нульової довжини» [2].

Відповідно до специфіки чисельних методів програми записуються в запам'ятовуючому пристрої у вигляді послідовності три-, дво- або одноадресних команд (можливі і чотириадресні команди). Для запису кожної команди відводиться слово стандартної довжини (розрядності). Пам'ять комп'ютера, де розміщуються слова програми, поділяються на комірки – одна комірка для одної команди [3].

Команди мають такий вигляд:

Триадресна команда має в своєму складі код операції, код адреси першої комірки 1-го операнда, код адреси другої комірки 2-го операнда та код адреси третьої комірки, куди записується результат виконання команди.

Двоадресна команда має в своєму складі код операції, код адреси першої комірки 1-го операнда та код адреси другої комірки 2-го операнда, куди записується результат виконання команди.

Одноадресна команда має в своєму складі код операції, код адреси першої комірки 1-го операнда.

У чотириадресній команді використовується ще адреса коду команди.

У таких мовах програмування вищого рівня, як С++ широко застосовуються адресації команд [4].

У зв'язку з застосування програмного управління в запам'ятовуючому пристрої (ЗП) команди програми і дані розташовуються у різних частинах ЗП. Довжина слів команди і чисел, як правило, вибираються однаковими.

Важливо зазначити, що обчислювання, які виконує ЕОМ, визначаються під впливом послідовних команд програми. Заміна команд програми може призвести до зміни функцій мікропроцесора.

Ще в 60-х роках XX ст. висловлювалася думка, що двійкова система тригерної пам'яті обмежує розвиток обчислювальної техніки. З появою у 1971 р. першого мікропроцесора 4004 фірми «Intel» і наступних за ним мікропроцесорів на НВІС на час зняли обмеження елементної бази ЕОМ, а також з ними зупинили і розвиток фундаментальних основ обчислювальної техніки, до яких належать елементарні схеми пам'яті. Японська програма 1981 р. зі створення високоінтелектуальних машин п'ятого покоління до нинішнього часу не дала жаданих результатів, хоча дослідження у цьому напряму продовжуються у багатьох розвинутих країнах і досі. Рішення потрібно шукати не тільки на верхньому рівні (архітектурному або програмному), а і у фундаментальних основах елементної бази комп'ютерів і розв'язувати ці завдання у комплексній сукупності. Розробка фундаментальних основ комп'ютерної техніки, де обробка загальної і часткової частини команди оброблялася б одночасно (за 1 такт) є актуальним сучасним завданням при розробці комп'ютерів.

З цієї точки зору розглянемо узагальнені принципи програмного управління за рахунок використання можливостей багатофункціональних елементарних автоматів, який названий принципом ієрархічного програмного управління [4].

Принцип ієрархічного програмного управління. Принцип ієрархічного програмного управління [5] полягає у тому, що інформацію, яка є обробляючою і керуючою, можна розбити на часткову і загальну, котрі взаємозв'язані між собою так, що керуюча інформація зв'язана із входом обробляючої інформації, що визначає функцію належності станів до визначених блоків (рис. 1).

Використання паралельної багаторівневої роботи пристроїв управління, які мають можливість працювати не тільки в однозначному, а й у ймовірному і не-

чіткому режимах, розширюють функціональні можливості обчислювальних пристроїв і створюють передумови для збільшення рівня машинного інтелекту.

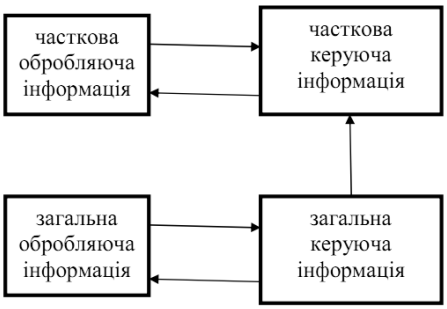

Рис. 1. **Принцип ієрархічного програмного управління**

При розробці типів адресних команд на основі принципу ієрархічного програмного управління потрібно в кожній адресній команді ввести додатковий код загальної команди, який дозволить визначити блок (підмножину) станів в якому буде працювати часткова керуюча інформація.

Розширення структури команд процесора. Команди з додатковим кодом загальної команди мають такий вигляд:

– триадресна команда має в своєму складі додатковий код загальної команди, код операції, код адреси першої комірки 1-го операнда, код адреси другої комірки 2-го операнда та код адреси третьої комірки, куди записується результат виконання команди (рис. 2).

Загальний код	Код операції	Код адреси 1-го операнда	Код адреси 2-го операнда	Код адреси 3-го операнда

Рис. 2. **Вигляд триадресної команди із загальним кодом**

– двоадресна команда має у своєму складі додатковий код загальної команди, код операції, код адреси першої комірки 1-го операнда та код адреси другої комірки 2-го операнда, куди записується результат виконання команди.

Загальний код	Код операції	Код адреси 1-го операнда	Код адреси 2-го операнда

Рис. 3. **Вигляд двоадресної команди із загальним кодом**

– одноадресна команда має у своєму складі додатковий код загальної команди, код операції, код адреси першої комірки 1-го операнда.

Загальний код	Код операції	Код адреси операнда

Рис. 4. **Вигляд одноадресної команди із загальним кодом**

У чотириадресній команді використовується ще адреса додаткового коду загальної команди та коду команд.

У команді адреса (номер комірки пам’яті, де вона зберігається) не вказується (за винятком чотириадресної, в якій четвертою адресою є адреса команди), він визначається тим, що команди програми розташовані у пам’яті послідовно. Адреса кожної послідовної команди подається в мікропроцесор за рахунок додавання 1 у програмному лічильнику мікропроцесора, який і визначає наступну адресу команди (*рис. 5*).

Команда програми за адресою, що подається із програмного лічильника (ПЛ), викликається із оперативної пам’яті і подається на регістр команд (РК) мікропроцесора і обробляється в пристрої управління (ПУ).

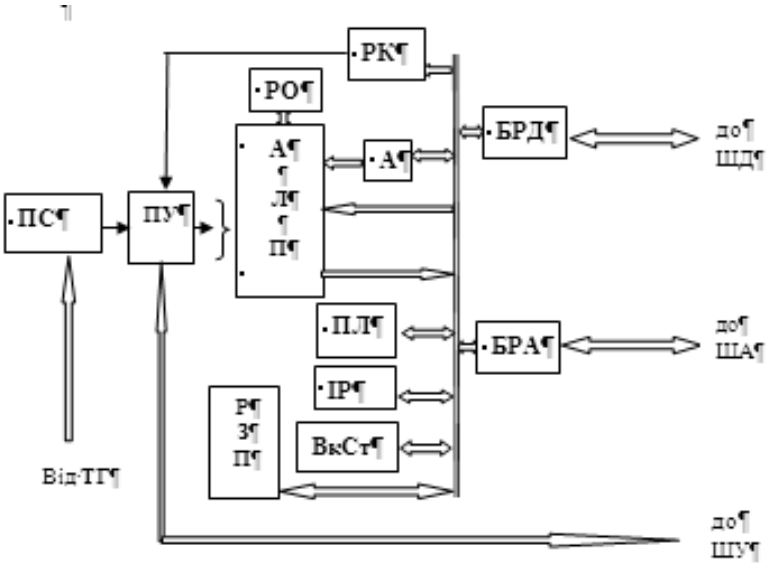

Рис. 5. **Структурна схема мікропроцесора**

Етап логічного проектування пристрою управління. На етапі логічного проектування дискретних систем найбільші труднощі виникають при проектуванні пристроїв управління (ПУ) у зв’язку з їхньою нерегулярністю і малою повторюваністю окремих вузлів. Починаючи з середині 70-х років XX ст. розв’язання цієї складної задачі здійснюється на програмуючих пасивних запам’ятовуючих пристроях (ПЗП) [5].

Матричні регулярні структури ПЗП реалізують комбінаційні схеми з деякою надмірністю. Але за рахунок своєї регулярності структури при проектуванні НВІС матрична структура реалізації функцій переключення спрощує процес проектування топології кристала НВІС, що значно зменшує час розробки і кількості помилок.

Розглянемо основні принципи побудови автоматів при використані комбінаційних схем на ПЗП і пам’яті автомата на багаторівневих схемах пам’яті (БРСП). Моделлю ПУ, що отримана за допомогою запропонованої методики, є кінцевий автомат, у якого пам’ять складається із БРСП, а комбінаційні схеми – із ПЗП. Багатопрограмний ПУ, що відображений на рис 6, є кінцевим автоматом. Схема пам’яті такого автомата складається із регістрових структур на БРСП. Узагальнені стани одного рівня пам’яті можна дешифрувати на окремому дешифраторі.

Дешифратор верхнього рівня DC_i (i= 1, 2, …, n) у рамках якого може бути організовані прямокутні матриці, реалізують притаманні тільки їм функції виходів автомата і функції переходів в БРСП, а отже, і функції збереження станів всередині БРСП. Дешифратор нижнього рівня DC_M дозволяє розширити число дешифраторів верхніх рівнів. Регістр $A_м$ своїми вихідними сигналами організує функції збереження станів регістра A_y.

Виходи регістра A_M^2дешифруються. На вихідних шинах дешифратора DC_M організується вихідна матриця переходів в регістрі A_M^1 і одночасно вихідні сигнали цього дешифратора разом з вихідними сигналами регістра A_M^2 організують дешифровку станів визначеного блока π_j станів регістра A_y^2.

На виходах цих дешифраторів організовуються дві матриці: матриці переходів регістра A_y^1 і матриця виходів. Ці матриці мають постійну прошивку відповідно до заданої поліграми і діють як набір логічних елементів АБО, котрий збуджує підмножину вихідних шин матриць. Матриці складаються із системи горизонтальних проводів (виходів дешифраторів) і системи вертикальних проводів.

Чорні точки (*рис. 6*) на перетині горизонтальних і вертикальних шин показують компонент, який зв'язує між собою ці дві системи проводів і сприяє розповсюдженню електричного струму (логічної одиниці) із горизонтальної шини у вертикальну. Горизонтальну шину можна уявляти як полікоманду, коли ця шина збуджена.

Відмінною рисою запропонованого поліпрограмного автомата від мікропрограмного автомата Уілкса [3] є те, що запропонований поліавтомат здатний функціонувати в різних підмножинах своїх станів залежно від станів регістра A_M, виконуючи різні режими своєї роботи. Такі поліавтомати здатні перебудовувати алгоритм своєї роботи у час його функціонування за один машинний такт.

Збільшення числа дешифраторів визначених блоків π_j станів регістра A_y^2 дозволяє збільшення можливостей півавтомата, що у свою чергу збільшує його функціональні можливості по реалізації додаткових робочих режимів.

При проектуванні автомата з «жорсткою» логікою на логічних елементах спочатку визначаємо потрібну кількість БРСП або кількість розрядів регістра на багатофункціональних схемах пам'яті (БФСП) з одним автоматом стратегії для кодування усіх станів регістра, використовуючи метод синтезу для побудови функціональних схем автомата.

При проектуванні автомата на програмуючих матрицях спочатку з'ясовуємо технологічні можливості виготовлення регістрів на БРСП і логічних матриць (ПЗП), а потім будуємо, виходячи із отриманих обмежень, функції схем автомата, що будується. Закон функціонування автомата залежить від відповідних поліграм, що містяться в матричних структурах ПЗП, тому що в поліавтоматі використовують двоступеневі пристрої пам'яті, запис інформації в яких синхронізується в різних моментах часових синхроімпульсів τ_1 і τ_2, що дає можливість довільного кодування станів автомата.

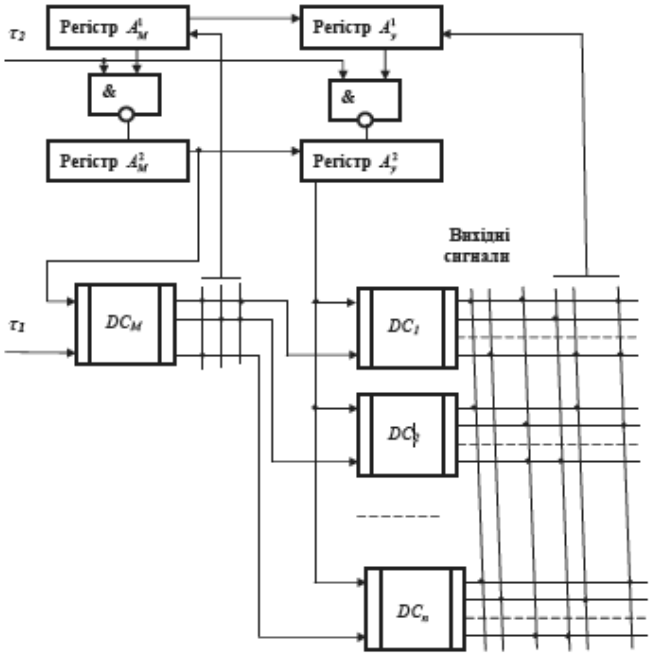

Рис. 6. **Схема пристрою управління на регістрах БРСП і ПЗУ**

Опис поліпрограм в ієрархічному автоматі. Автомат зручно описувати алгоритмічно, розглядаючи його функціонування послідовно при кожному переході із одного стану в інший за час одного зовнішнього такту T_i автоматного неперервного часу [5]. Наочним способом завдання автоматів із пам'яттю на тригерах є завдання їх у вигляді мікропрограм або більш в узагальненому вигляді як автограмма [6].

Для алгоритмічного опису ієрархічних автоматів (*ІА*) із пам'яттю на БФСП з узагальненим станом, що складається з окремих станів підавтоматів S_i, застосуємо термін «поліграма» з таких міркувань. По-перше, термін «мікропрограма», «автограма» й інші засоби завдання автоматів із пам'яттю на тригерах орієнтовані на опис функцій переходів в автоматах тільки у час тактового моменту t, що обмежує опис функціонування автомата під час дії внутрішнього тактового моменту Δ автоматного неперервного часу T. По-друге, поняття «мікропрограма», «автограма» й інші засоби завдання автоматів зі пам'яттю на тригерах орієнтовані на запам'ятовування стану в тригерних регістрах, тому терміни «мікропрограма» й «автограма» не дозволяють описувати узагальнений стан регістрів на БРСП. Термін «поліграма», що описаний в роботі [5], асоціюється зі схемною реалізацією управління на багаторівневих регістрах з багатофункціональною системою організації пам'яті, яку називають багаторівневою ієрархічною автоматною реалізацією. По-третє, поліграма орієнтована не тільки на перетворення інформаційних x_i вхідних сигналів у вихідні, але і на зміну області запам'ятовування станів автомата, яка визначається e_j вхідним сигналом. Ця характерна особливість дозволяє використовувати поліграму при описі ієрархічної структури, яка здатна перетворювати інформаційні x_i вхідні сигнали та ще змінювати внутрішню структуру перетвореної інформації всередині схеми пам'яті (*рис. 7*).

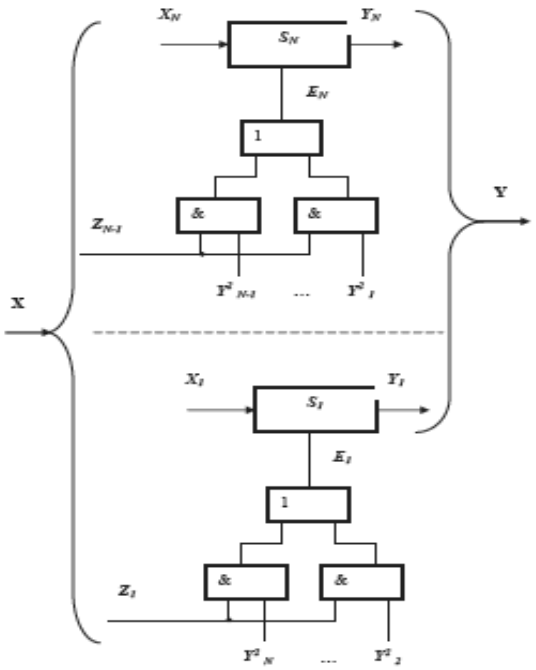

Рис. 7. **Структура ієрархічного абстрактного автомата**

Поліграма описує кожний стан *IA A* як об'єднання a_i станів підавтоматів S_i:

$$a_к = \cup\, a_i. \tag{1}$$

В кожному пункті поліграми описуються режими роботи підавтоматів S_i за один зовнішній такт T автоматного неперервного часу. За один зовнішній такт T *IA A* сприймає вхідне слово $R_k(T)$, яке складається із сукупності елементарних $p_i(T)$ вхідних слів підавтоматів S_i :

$$R_k = \cup\, p_i. \tag{2}$$

Під впливом вхідного слова $R_k(T)$ *IA A* здатний перейти в новий стан $a_к$ і видати вихідний сигнал Y_k, який складається із сукупності Y_i вихідних сигналів визначених типів підавтоматів S_i:

$$Y_K^1(t) = \cup\, Y_I^1(t); \tag{3}$$

$$Y_K^2(T) = \cup\, Y_I^2(T); \tag{4}$$

$$Y_K^3(\Delta) = \cup\, Y_I^3(\Delta). \tag{5}$$

де вихідні сигнали $Y_I^1(t)$, $Y_I^2(T)$, $Y_I^3(\Delta)$ визначаються відповідними по функціях виходу, що описується у рівняннях (3 – 5).

При незмінних $e_i(\Delta)$ вхідних сигналах підавтомати S_i функціонують у визначених блоках π_{ji} своїх станів, а *IA A* також функціонують у той же період часу у визначеному блоці $\pi_к$ своїх станів, що складає сукупність блоків π_{ji}:

$$\pi_k = \cup\, \pi_{ij}. \tag{6}$$

Кожний стан a_i підавтомата S_i запам'ятовує свій стан при відповідних e_j вхідних сигналах, які відображають загальну інформацію. Зміна e_j вхідного сигналу в підавтоматах *Si* здійснюється в тактовий момент t і залежно від вихідних сиг-

налів $Y_I^2(T)$, що надходять від інших підавтоматів *Si* (відповідно до алгоритму розв'язання задачі) під час внутрішнього такту Δ зберігаючий $e_j(\Delta)$ ($Y_I^2(T) = e_j(\Delta)$) вхідний сигнал визначає блок π_{ij} станів підавтомата S_i.

Зберігаючий $e_j(\Delta)$ ($Y_I^2(T) = e_j(\Delta)$) вхідний сигнал однозначно визначає блок π_{ij} станів, в яких працює підавтомат S_i, а сукупність $E_к$ зберігаючих $e_j(\Delta)$ вхідних сигналів однозначно визначає блок $\pi_к$ станів, в яких працює *ІА А*:

$$E_k = \cup e_j. \tag{7}$$

Пункт поліграми при детермінованому елементарному вхідному слові R_i записується в такому вигляді:

$$\begin{gathered} KE.R_1 \to Y_1^I - K_1E_1, \\ R_2 \to Y_2^I - K_2E_2 \\ R_m \to Y_m^I - K_mE_m, \end{gathered} \tag{8}$$

де *К* – узагальнений стан *ІА;*

Е – узагальнений зберігаючий вхідний сигнал *ІА;*

R_j *(j*=1, 2, ..., *m)* – узагальнене елементарне вхідне слово;

K_j *(j=1, 2, ..., m)* – узагальнений стан *ІА*, до якого здійснюється перехід від стану *К* при виконані рядка *j*;

E_j *(j*=1, 2, …, *m)* – узагальнений зберігаючий вхідний сигнал, при якому запам'ятовується стан K_j;

Y_j^i *(j*=1, 2, …, *m)* – узагальнений вихідний сигнал *ІА*.

Кожний рядок ($R_j \to Y_j^i - K_jE_j$) пункту *КЕ* поліграми розгортається в $S_к$ рядків і описується так:

$$\begin{gathered} a_1e_1 * p_{j1} \to Y_{j1`}^i - a_{j1}e_{j1}, \\ a_2e_2 * p_{j2} \to Y_{j2`}^i - a_{j2}e_{j2},.. \\ a_qe_q * p_{jq} \to Y_{jq`}^i - a_{jq}e_{jq}, \end{gathered} \tag{9}$$

де *q* – число рядків S_k, що є в опису рядка поліграми *j*;

p_{ji}– елементарне вхідне слово *j* рядка;

$Y_{ji`}^i$ – вихідний вектор *j* рядка;

a_{ji} – стан підавтомата S_i, до якого здійснюється перехід від стану a_i при виконані рядка S_k;

e_j – зберігаючий вхідний сигнал, який характеризує загальну інформацію зберігання підмножини станів півавтомата S_i.

Такий опис роботи *ІА А* в стані *КЕ* є пунктом опису ієрархічного алгоритму функціонування *ІА*.

Пункт поліграми *КЕ* має ієрархічну структуру, тобто спочатку описується узагальнені стани K_0, K_1, ..., K_m, що під впливом узагальнених зберігаючих E_0, E_1, ..., E_m вхідних сигналів запам'ятовують підмножини станів, узагальнені елементарні вхідні слова R_j *(j*=1, 2, …, *m),* а потім кожний рядок пункту поліграми розгортається в $S_к$ рядків, які реалізуються в підавтоматах S_i *ІА А*, здійснюючи перехід від одного стану в інший. Під час функціонування *ІА А* деякі (а можливо і усі) підавтомати S_i можуть не змінювати свої стани на якомусь відрізку часу. У цьому випадку, при описі поліграми *ІА А* в кінці рядків будуть помічатися стани, що збіжні з початковою поміткою пункту поліграми або пункту рядка поліграми.

Кожний початковий *j*-й рядок пункту *КЕ* поліграми складається із таких елементів, функціонування яких описується в автоматному неперервному часі: початкова помітка пункту *КЕ* поліграми, що відображає узагальнений стан *К*(Δ-1),

який запам'ятовується при узагальненому зберігаючому $E(\Delta\text{-}1)$ вхідному сигналі; $R_j(T)$ – узагальненого елементарного вхідного слова, що складається із узагальнених інформаційних $X_j(t)$ вхідних сигналів і узагальнених зберігаючих $E_j(\Delta)$ вхідних сигналів; узагальнених вихідних Y_j^i сигналів, що складаються із вихідних Y_k^i сигналів підавтоматів S_i і описуються рівняннями (1 – 5).

Змістовний смисл кожного *j*-го рядка поліграми полягає в установлені зв'язків між станами $K(\Delta\text{-}1)$ *ІА А* у попередньому зовнішньому такті $T_{i\text{-}1}$ і реакцією *ІА А* у наступному зовнішньому такті T_i при переході в узагальнений стан $K_j(\Delta)$.

Кожний *j*-й рядок поліграми можна подати як команду, яка складається із трьох частин. Перша частина задає вектор елементарного R_j вхідного слова, друга – вектор вихідних Y_j^i сигналів, а третя – визначає функцію вектора K_j станів при векторі зберігаючого E_j вхідного сигналу.

Вектор зберігаючого E_j вхідного сигналу задає визначений блок запам'ятовуючих станів автомата, в якому він діє у даний момент часу. Вхідний E_j сигнал забезпечує ієрархічний взаємозв'язок підавтоматів S_i в *ІА А* під час внутрішнього такту Δ. Логіка системи, у котрій внутрішній взаємозв'язок має таке або більше значення, ніж їхній зовнішній взаємозв'язок, найчастіше стикає нас з складністю формулювання взаємозв'язаних поліграм. Перетворення зовнішньої інформації може бути більш досконалим за рахунок нових переходів автоматів третього роду у більш складній і добре організованій структурі системи, яку воно супроводжує.

Зміна внутрішньої структури запам'ятовування станів в схемах автоматної пам'яті і перетворення зовнішньої інформації – це лише дві взаємозв'язані частини одного і того ж процесу перетворення інформації в *ІА А*. Ці дві частини і відображені у рядку поліграми.

Висновки. У статті розглянуті фундаментальні питання: принцип ієрархічного програмного управління; розширення структури команд, за рахунок введення додаткових полів загального коду; етап логічного проектування реконфігурованого пристрою управління; опис поліпрограм в ієрархічному автоматі, що дає можливість будувати на їхній основі нові реконфігуровані мікропроцесори.

ЛІТЕРАТУРА

1. *Глушков В.М.* Синтез цифровых автоматов. – М.: Физматгиз, 1962. – 476 с.
2. *Глушков В.М.* Теория алгоритмов. – К.: КВИРТУ, 1961. – 167 с.
3. *Справочник* по цифровой вычислительной технике (процессоры и память) / Б.Н. Малиновский, Е.И. Брюхович, Е.Л. Денисенко и др.; Под ред. Б.Н. Малиновского. – К.: Техніка, 1979. – 366 с.
4. *Ишкова Э.А.* С++ начала программирования.– М.: ЗАО «Издательство БИНОМ», 2001. – 351 с.
5. *Мараховский Л.Ф., Михно Н.Л.* Основы новой информационной технологии: Монографія.– Saarbrcken, Germany: i.melnic@lap-publishing.ru/www.lap-publishing.ru., 2013. – 369 с.
6. *Глушков В.М., Капитонова Ю, В., Мищенко А.Т.* Логическое проектирование дискретных устройств. – К.: Наук. думка, 1987. – 264 с.