УДК 62.503.5:[004.2+004.4]

АРХІТЕКТУРА ОБЧИСЛЮВАЛЬНОГО ВУЗЛА МЕХАТРОННОГО ПРИСТРОЮ

С.М. Ткаченко[1], Д.О. Бешта[2], А.В.Кожевніков[3]

[1] Національний технічний університет «Дніпровська політехніка», м.Дніпро,
tkachenko.s.m@nmu.one ORCID 0000-0003-1156-3151
[2] Національний технічний университет “Дніпровська політехніка”, Дніпро, Україна,
beshta.d.o@nmu.one ORCID 0000-0003-2848-2737
[3] Національний технічний университет “Дніпровська політехніка”, Дніпро, Україна,
kozhevnykov.a.v@nmu.one ORCID 0000-0002-0078-2546

ARCHITECTURE OF THE COMPUTER UNIT OF THE MECHATRONIC DEVICE

S. Tkachenko[1], D. Beshta[2], A. Kozhevnikov[3]

[1] Dnipro University of Technology, Dnipro, Ukraine
tkachenko.s.m@nmu.one ORCID 0000-0003-1156-3151
[2] Dnipro University of Technology, Dnipro, Ukraine,
beshta.d.o@nmu.one ORCID 0000-0003-2848-2737
[3] Dnipro University of Technology, Dnipro, Ukraine,
kozhevnykov.a.v@nmu.one ORCID 0000-0002-0078-2546

Мета роботи. Розробити архітектуру обчислювального вузла та керуючий алгоритм мехатронного пристрою, які дозволяють скоротити трудовитрати на програмування і відлагодження за рахунок уніфікації методу програмування.

Методика дослідження. Аналіз рішень, рекомендацій та алгоритмів керування від виробника станції маніпулятора. Ескізне проектування систем у частині функціональних структур. Синтез алгоритму та структури даних з використанням методів ситуаційного керування та методів організації зберігання даних у пам’яті комп’ютера.

Результати дослідження. Запропоновано загальну функціональну структуру обчислювального вузла мехатронного пристрою, обґрунтовано її властивості та переваги, запропоновано універсальний підхід до написання алгоритму основного керуючого блоку станції маніпулятора. Синтезовано мову ситуаційного керування для розробки структури даних у формі, що дозволяє гнучко змінювати технологічні задачі станції маніпулятора. Представлено алгоритм основного керуючого блоку програми станції маніпулятора, який реалізує вибірку керуючих впливів з масиву за вхідними інформаційними ознаками і є універсальним.

Наукова новизна. Побудовано маргінально індексовий масив продукцій вихідних впливів, який дозволяє замість алгоритму керування за графом використати простий алгоритм вибору з масиву без перевитрат оперативної пам’яті, які виникають у таком рішенні за умови використання інформаційних векторів вихідних впливів.

Практичні результати. Запропоновано метод задання дій маніпулятора, який дозволяє відмовитись від програмування його обчислювального вузла на користь задання структур даних безпосередньо програмістом або засобами автоматичних систем проектування чи налагодження. Метод дозволяє відокремити обчислювальний вузол від мехатронного пристрою і використовувати окремо або замінювати обчислювальні вузли під час експлуатації без перепрограмування.

Ключові слова: Маніпулятор, обчислювальний вузол, комптроллер, функціональна структура, станція маніпулятору FESTO, ситуаційне керування, маргінальне індексування, графсет, керуючий вплив, семантичний зв'язок.

Вступ. Об’єктами досліджень, з точки зору мехатроніки, є крани-маніпулятори, маніпулятори ліній збирання, сортування, бурильні установки та інші. Особливості впровадження та експлуатації мехатронних пристроїв можуть бути пов’язані з необхідністю частого внесення змін в обчислювальні вузли мехатронних пристоїв або необхідністю їх фізичної заміни. Не завжди просто, а іноді й неможливо організувати робоче місце програміста безпосередньо біля мехатронного пристрою. З іншого боку, віддалене програмування і налагодження такого пристрою чи лінії також складна задача, що потребує порівняно більше часу, підготовчих операцій та наявності, як мінімум, одного кваліфікованого асистента зі спеціалізованим обладнанням на об’єкті. Тому актуальна задача розробки архітектури керуючих обчислювальних ву-

злів мехатронних пристроїв спрямованої на зменшення трудовитрат на програмування і відлагодження під час їх впровадження і експлуатації.

Мета досліджень. Виходячи з аналізу існуючих підходів до створення промислових систем управління та методів організації доступу до даних розробити архітектуру обчислювального вузла та керуючий алгоритм мехатронного пристрою, які дозволяють скоротити трудовитрати на програмування і відлагодження за рахунок уніфікації методу програмування.

Основний матеріал досліджень. В умовах виробництва в якості обчислювальних вузлів, у тому числі й мехатронних пристроїв, використовують промислові контролери, або, як їх ще називають комптролери. На апаратному рівні комп'ютерної архітектури комптролери є уніфікованими пристроями, що складаються з центрального процесорного модуля, який також включає у свій склад оперативну та постійну пам'ять та модулів розширення. Модулі розширення можуть приймати або видавати дискретні чи аналогові сигнали, організовувати зв'язок з іншими вузлами промислової мережі, використовуючи відповідні протоколи. Окремо виконується блок живлення комптролера, як правило +24В. Центральні процесорні вузли також можуть мати інтегровані входи і виходи сигналів, інтерфейси промислових мереж, блоки живлення. Комптролери мають свої операційні системи, що включають засоби зв'язку, діагностики обладнання, інтерпретації програм [1]. Тому з точки зору програмування, впровадження і експлуатації мехатронних систем інтерес для дослідження представляє більш високий рівень архітектури – рівень прикладних програм [2].

Розглянемо підхід до написання програм для промислових контролерів, що пропонується визнаною фірмою FESTO в рамках курсу підготовки фахівців з мехатроніки та для змагань «Wordskills» з мехатроніки. Використаємо дані одного з лабораторних маніпуляторів FESTO[3], призначеного для переміщення деталей різного матеріалу і кольору в кілька кінцевих точок. Станція маніпулятору разом з комптролером Simatic S7-1214C являє собою мехатронний пристрій. Перелік використаних фізичних входів і виходів комптролеру узгоджено зі схемою підключення [2] та наведено в таблиці 1. Сюди ж, для зручності подання матеріалу, додано і деякі внутрішні змінні.

Таблиця 1.

Перелік вхідних і вихідних сигналів маніпулятора

Позначення сигналу	**Назва сигналу**	**Напрямок відносно контролера**	**Призначення**
1	2	3	4
X_0	Заготовка на приймальному лотку є	Вхід	Виявити заготовку і запустити алгоритм обслуговування
X_1	Каретка на місці захоплення	Вхід	Дозволити захоплення деталі
X_2	Каретка на місці сортування	Вхід	Дозволити вивантаження чорної деталі
X_3	Каретка на місці вивантаження	Вхід	Дозволити вивантаження деталі
X_4	Захоплювальний пристрій опущений	Вхід	Дозволити розкрити чи закрити захоплювальний пристрій в залежності від положення каретки
X_5	Захоплювальний пристрій піднятий	Вхід	Дозволити почати рух каретки в одному з напрямів чи розпочати відлік часу перед роботою з черговою деталлю
X_6	Колір заготовки не чорний	Вхід	Дозволити вивантаження у місці сортування. Датчик розміщено у захваті маніпулятора
X_7	Кнопка «start» на інтерфейсі управління	Вхід	Виставити ознаку «start», тобто «станція працює в автоматичному режимі»
X_7'	Кнопка «stop» на інтерфейсі управління	Вхід	Скинути ознаку «start».
X_8	Кнопка «reset» на інтерфейсі управління	Вхід	Виставити каретку у початкову позицію (на місце захоплення)
Y_0	Рух каретки у напряму місця завантаження	Вихід	Включення електроприводу руху у прямому напрямку.
Y_1	Рух каретки у напряму місця вивантаження	Вихід	Включення електроприводу руху у зворотньому напрямку.
Y_2	Опускання захоплювального пристою	Вихід	Включення одноходового пневмоциліндру опускання
Y_3	Розкриття захоплювального пристрою	Вихід	Включення одноходового пневмоциліндру розкриття
Y_4	Індикатор «reset»	Вихід	Включення лампочки на інтерфейсі управління
Y_5	Індикатор «start»	Вихід	–//–

Закінчення таблиці 1

1	2	3	4
Y_6	Індикатор роботи в автоматичному режимі	Вихід	–//–
Y_7	Сигнал «Станція зайнята»	Вихід	Сигнал для зовнішнього пристрою завантаження деталі на станцію.
Z_0	Затримка часу перед роботою з черговою деталлю пройшла	Внутрішня	Дозвіл роботи з черговою деталлю. Введено, щоб дати час не виведення органу пристрою завантаження з робочої зони станції
Z_2	Ознака «start», тобто «станція працює в автоматичному режимі»	Внутрішня	Програмний тригер, що використовується замість кнопок «start» і «stop»
Z_3	Ознака «появу заготовки на вході зафіксовано»	Внутрішня	Програмний тригер, що має використовуватись в запропонованому рішенні, оскільки захват маніпулятора не має відповідного датчика
Z_4	Ознака «станція готова до роботи»	Внутрішня	Програмний тригер, що означає готовність станції до запуску в автоматичному режимі

За рекомендаціями FESTO, та оргкомітету змагань, керування станцією, предсталеною на рисунку 1, може бути реалізовано за графсетом, показаним на рисунках 2 і 3 [3, 4].

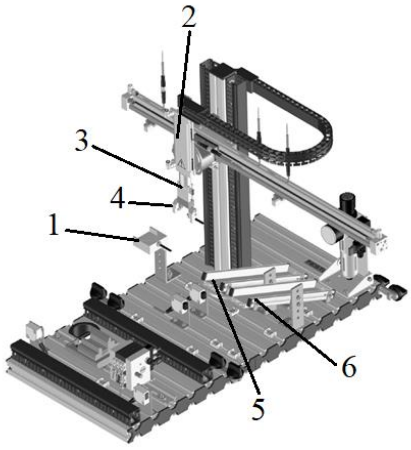

Рис. 1 Станція маніпулятору FESTO:
1 – місце захоплення (завантаження); 2 – каретка з електроприводом; 3 – шток пневмоциліндра підніймання і опускання захоплювального пристою; 4 –пневматичний захоплювальний пристрій; 5 – місце сортування (вивантаження чорної деталі); 6 – місце вивантаження (не чорних деталей)

В якості рішення для змагань, так ий проект програми цілком придатний. Але, з точки зору розробки і експлуатації програмного забезпечення рішення не структуроване. В одному алгоритмі тут об'єднані кілька задач – обслуговування таймеру, виконання графу управління, нормування і масштабування сигналів. Остання задача настільки тісно пов'язана з графом, що за умови використання дискретних сигналів дії з нормування у графі виявити не просто. Тим не менше, звернімо увагу, що, в залежності від технічних характеристик застосованих датчиків, сигнали можуть передаватись як логічним 0, так і як логічним 1. Граф управління, представлений на рисунках 2 і 3 жорстко орієнтований на значення входів. А значить, зміна характеристики вхідного сигналу, наприклад, через заміну датчика, призведе до необхідності переписування, зміни структури графу та переналагодження програми. Те ж саме можна спостерігати у випадку зміни затримки часу в алгоритмі чи зміни кількості вхідних інформаційних ознак. Таким чином, рішення, запропоноване в [3, 4], для виробництва непридатне.

Скористаємося іншими, відомими методами проектування систем [5]. Визначимо функціональну структуру програми керування маніпулятором. Пропонована структура показана на рисунку 4.

Алгоритми керування маніпулятором, враховуючи вимоги до стадійності проектування [5], повинні бути створені для тих блоків функціональної структури на рис. 4, які цього потребують. Переваги підходу очевидні. У випадку зміни типу сигналу дискретного датчика, або способу надання керуючого впливу на виконавчий орган, задача переписування і відлагодження програми локалізується в окремих функціональних блоках нормалізації та масшабування без втручання в керуючий алгоритм. Це значно спрощує і внесення змін у програму, і процес налагодження.

Винесення таймерів технологічних затримок часу з керуючого алгоритма також дозволяє їх коригувати без втручання в основний алгоритм. Більше того, прирівнювання груп таймерів до звичайних вхід-

них з обов'язковим пропусканням пов'язаних з ними інформаційних ознак через блоки нормування і масштабування (див. рис. 4) дозволяє вільно модифікувати алгоритми контроля часу та навіть, за необхідності, відмовитися від них на користь зовнішніх апаратних засобів контролю часу. Така необхідність виникає, коли продуктивності комптролера не вистачає для дотримання необхідної точності відліку часу.

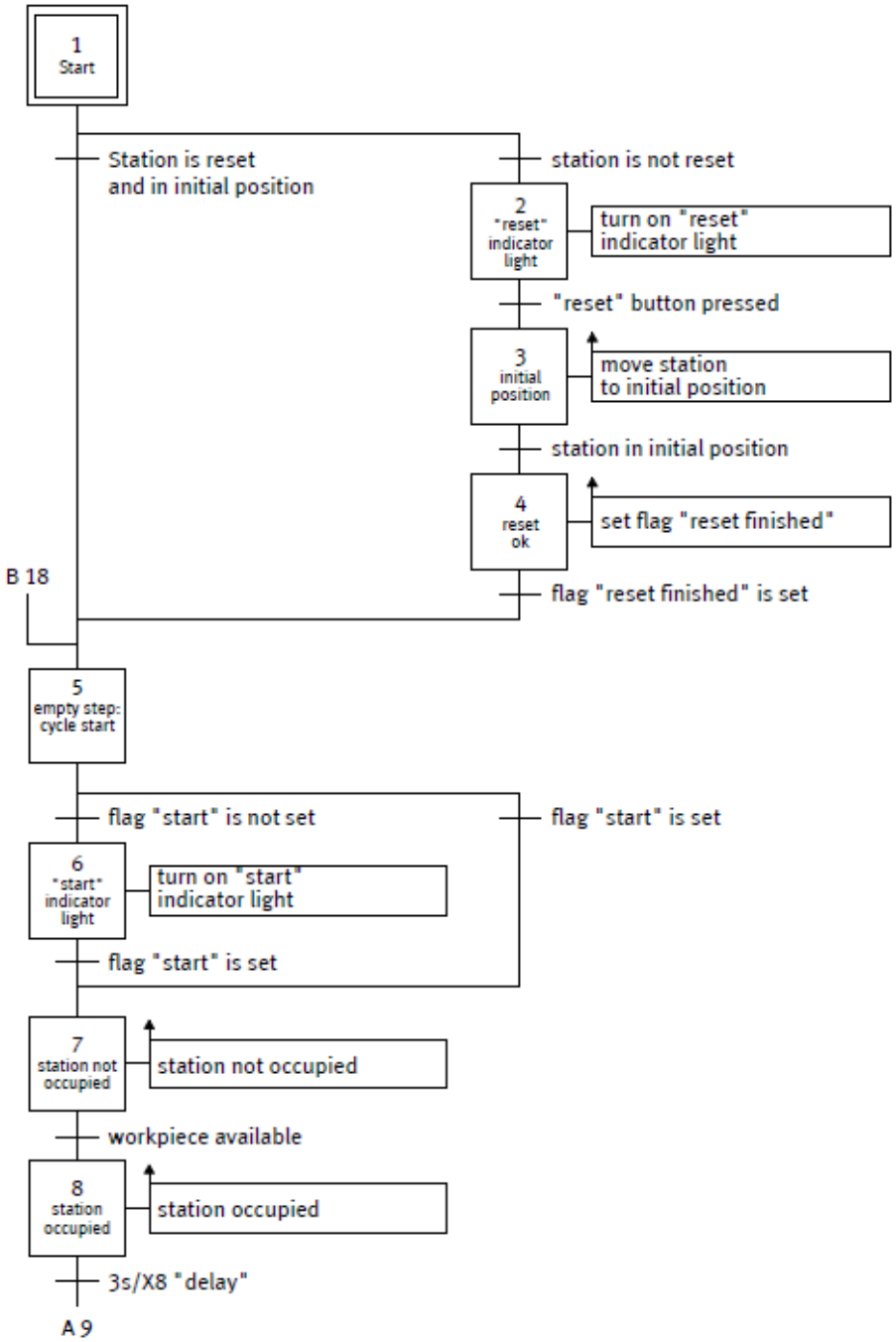

Рисунок 2 – Графсет програми роботи маніпулятора згідно DIN EN 60848 (IEC 60848)

Щодо інтерфейсу пульта управління. Зазвичай він присутній. Але з точки зору програми управління, як показано на рисунках 2 і 3, не має значення: надходять сигнали від датчиків, кнопок чи панелі HMI. Відповідно, і керуючі впливи однаково надходять на виконавчі органи чи на індикатори візуалізації.

Звернімо увагу на групу блоків нормування і масштабування вхідних сигналів. У наведеному прикладі маніпулятора все просто. Якщо доводиться замінити дискретний датчик з нормально розімкнутим контактом на нормально замкнутий, то програміст просто має виконати інверсію відповідного входу. Інша річ, коли мова йде про маніпулятор, наприклад, бурильної машини для буровибухових робіт. Тут мова піде про обробку сигналів положення від енкодерів, причому для деякої непостійної кількості шпурів

різної глибини, що визначається паспортом буровибухових робіт. Задача складніша, оскільки пов'язана з розрахунками переміщення кінематичних ланок.

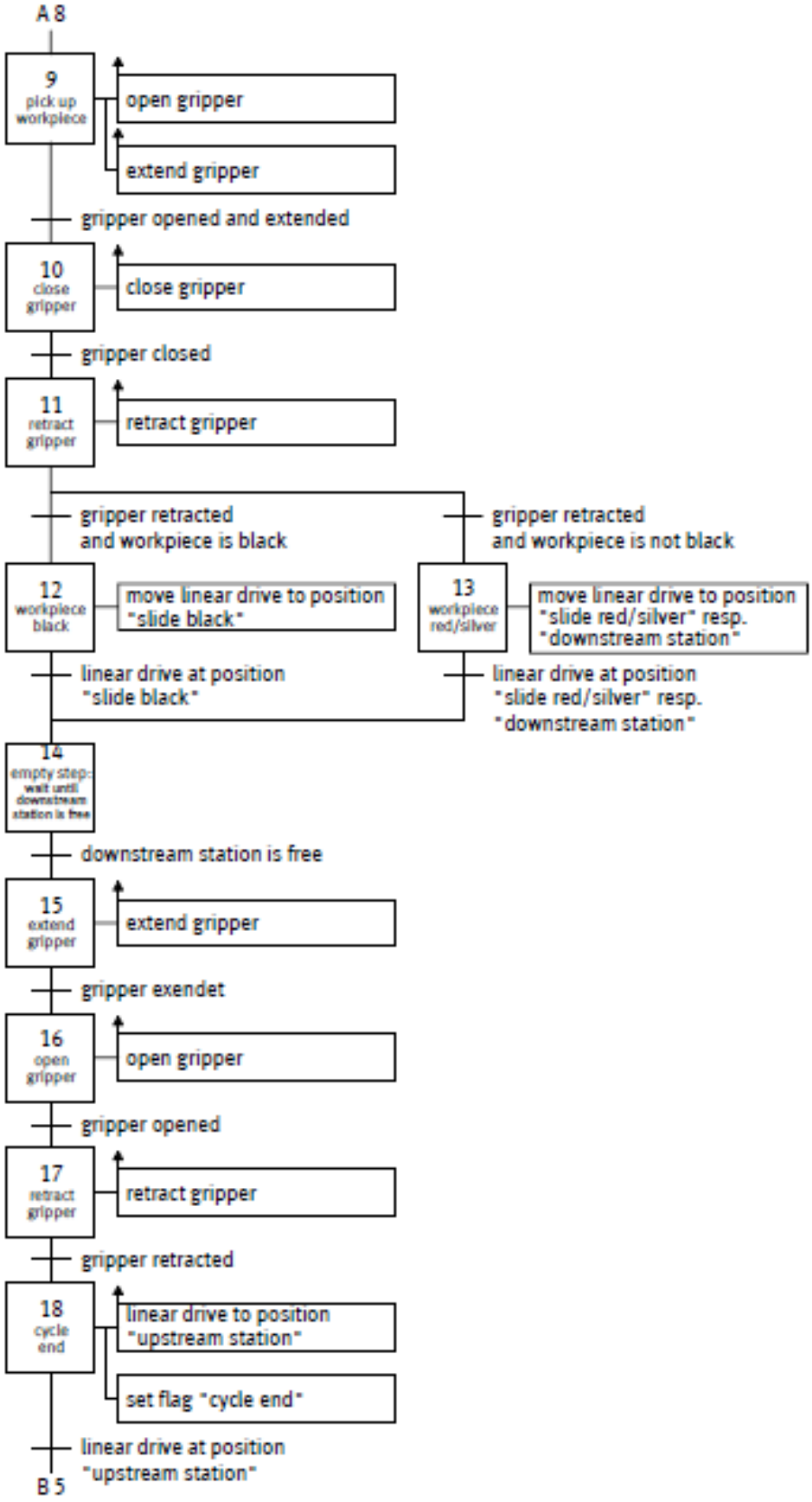

Рисунок 3 – Графсет програми роботи маніпулятора згідно DIN EN 60848 (IEC 60848). Продовження

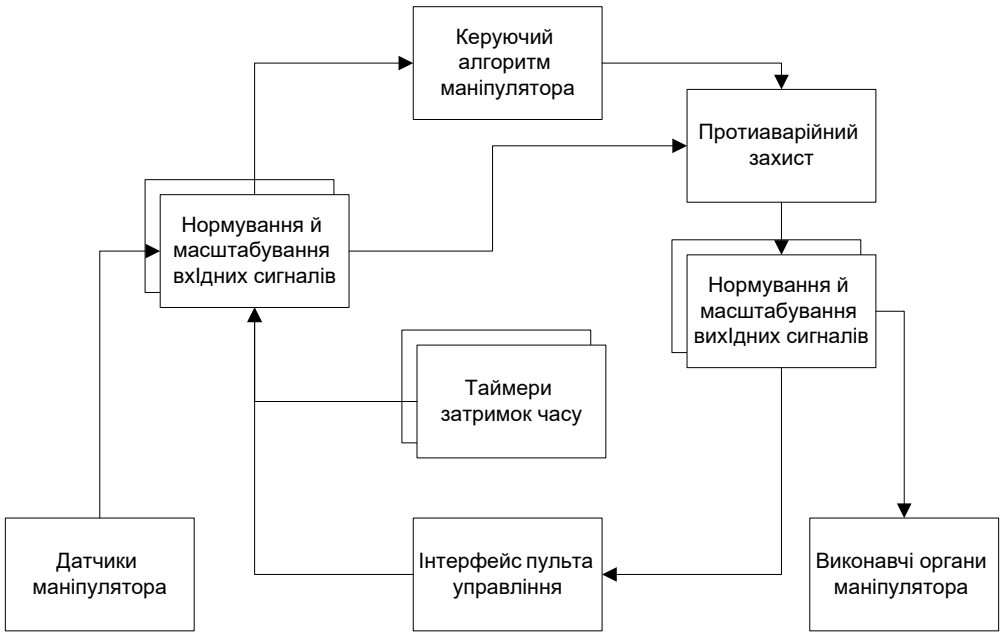

Рис. 4 Функціональна структура системи керування маніпулятором

Але і розрахунки для буріння локалізуються в рамках блоку нормування й масштабування. Власне, для керуючого алгоритму важливий лише факт досягнення маніпулятором заданого положення незалежно від способу отримання відповідної інформаційної ознаки. Проблема полягає в іншому. Якщо для лабораторного маніпулятора на 3 положення, показаного на рисунку 1, керуючий алгоритм описаний графсетом на 18 блоків, то у випадку з бурильною машиною, де кількість шпурів непостійна і складає близько 20..30, довжина алгоритму виростає на порядок.

Для вирішення проблеми зміни кількості інформаційних ознак на вході та виході функціонального блоку керуючого алгоритму можливо відійти від принципу написання керуючого алгоритму під конкретний пристрій на користь універсального алгоритму, що працює з групами інформаційних ознак.

Розглянемо входи маніпулятора згідно таблиці 1. Їх 9: інформаційних ознак, які надаються входами 8 та 1 внутрішня ознака затримки часу.

Можна представити ці 9 вхідних інформаційних ознак у вигляді звичайного двійкового числа виду $Z_0Z_1Z_2Z_4X_1X_2X_3X_4X_5X_6X_8$. Число можна використати як координату у одновимірному масиві груп вихідних керуючих впливів, яких буде 8, тобто вистачить змінної типу BYTE. Група керуючих впливів буде являти собою число, наприклад, вигляду $Y_0Y_1Y_2Y_3Y_4Y_5Y_6Y_7$.

Іншими словами, якщо поставити у відповідність 11-розрядному ключеві пошуку готову вихідну комбінацію керуючих впливів, то замість складного графу керування отримаємо просту вибірку готового впливу на групи входів та паралелізм у видачі керуючих впливів. Робота програміста у такому рішенні зводяться до заповнення масиву вихідних впливів.

Але, у чистому вигляді показаний підхід насправді не спрощує роботу з програмування, оскільки довжина масиву вихідних впливів складе 2^{11}=2048 комбінацій, причому більшість його елементів або будуть повторюватись, або не знадобляться, оскільки є технічно заборонені і апаратно неможливі комбінації. Наприклад, каретка маніпулятора не зможе одночасно знаходитись у кількох місцях. Між тим, програмістові доведеться передбачити всі 2048 комбінацій. А якщо, наприклад, кран-маніпулятор складу обслуговує 30 комірок, то йому може знадобитися вже до 40 вхідних інформаційних ознак. Передбачити й внести у інформаційний вектор 2^{40}= 1 099 511 627 776 вихідних комбінацій неможливо, а оперативна пам'ять порядка терабайт у комптролерах не застосовується.

Використаємо інший підхід. Розпишемо всі можливі комбінації вихідних сигналів $Y_0Y_1Y_2Y_3Y_4Y_5Y_6Y_7$ у таблиці 2.

Комбінації вихідних впливів, наведені у таблиці 2 для станції маніпулятору FESTO є достатніми. Їх можна зібрати у індексований одновимірний масив Q[0..n], де n=8. Масив для розміщення у пам’яті контролера може використовувати тривіальний метод інформаційного вектора. Що стосується згадуваного для порівняння крану-маніпулятору, то в цьому випадку кількість впливів також обмежена: лінійні рухи каретки; опускання й піднімання захвату; захват і відпускання вантажу. Щоправда, точок завантаження/вивантаження може близько 30, але на повну комбінацію вихідних впливів цей факт не впливає, оскільки чергова точка обирається шляхом впливів на обмежену кількість приводів.

Таблиця 2.

Допустимі комбінації вихідних впливів

Індекс n	Комбінація $Y_0Y_1Y_2Y_3Y_4Y_5Y_6Y_7$	Опис впливу
0	0000 1000	Маніпулятор не готовий працювати. Захват підняти і закрити. Каретку зупинити. Подати сигнал-запрошення натиснути «reset».
1	1001 0000	Маніпулятор не готовий працювати. Захват підняти і закрити. Каретку рухати у бік точки завантаження.
2	0001 0100	Маніпулятор готовий працювати. Захват підняти і відкрити. Каретку зупинити. Подати сигнал-запрошення натиснути «start».
3	0001 0010	Маніпулятор працює в автоматичному режимі. Захват підняти і відкрити. Каретку зупинити. Індикатор роботи в автоматичному режимі включити
4	0011 0010	Маніпулятор працює в автоматичному режимі. Захват відкрити і опустити. Каретку зупинити. Індикатор роботи в автоматичному режимі включити
5	0000 0010	Маніпулятор працює в автоматичному режимі. Захват закрити і підняти. Каретку зупинити. Індикатор роботи в автоматичному режимі включити
6	0100 0010	Маніпулятор працює в автоматичному режимі. Захват закрити і підняти. Каретку рухати у бік точки вивантаження. Індикатор роботи в автоматичному режимі включити
7	0010 0010	Маніпулятор працює в автоматичному режимі. Захват закрити і опустити. Каретку зупинити. Індикатор роботи в автоматичному режимі включити
8	1001 0010	Маніпулятор працює в автоматичному режимі. Захват підняти і відкрити. Каретку рухати у бік завантаження. Індикатор роботи в автоматичному режимі включити

Тепер повернімося до ідеї використання комбінації вхідних інформаційних ознак в якості координат елементу масиву вихідних впливів. Її можливо реалізувати, якщо перейти від методу представлення масиву у вигляді інформаційного вектора до методу маргінального індексування елементів у багатовимірному базисі [2].

Щоб визначити мірність масиву з маргінальним індексуванням, скористаємося підходами, що поширені у ситуаційному керуванні[6]. Пропонуємо для описання станції маніпулятора ввести елементи мови ситуаційного керування (МСК) з поняттями, наведеними у таблиці 3.

Таблиця 3.

Поняття й імена МСК станції маніпулятора

Позначення поняття	Опис	Інформаційні ознаки, що визначають імена у просторі поняття	Допустимі імена просторів понять у бінарній формі
a	Власний стан системи керування	Z_2Z_4	00 – станція не готова до роботи; 01 – нештатний режим; 10 – станція готова до роботи; 11 – станція працює в автоматичному режимі
b	Стан захоплювального пристрою	X_4X_5	00 – піднятий і закритий; 01 – опущений і закритий; 10 – піднятий і відкритий; 11 – опущений і відкритий
c	Положення каретки маніпулятора	$X_1X_2X_3$	000– збереження попереднього стану; 100 – каретка на місці захоплення; 010 – каретка на місці сортування; 001 – каретка на місці вивантаження; 011, 101, 110, 111 – нештатний режим
d	Класифікація заготовки	$Z_0Z_3X_6$	000 – заготовка відсутня; 001, 010, 011, 100 – збереження попереднього стану; 110 – заготовка не чорна; 111 – заготовка чорна; 101 – нештатний режим

Таким чином, отримаємо групи інформаційних векторів a[0..i], b[0..j], c[0..k], d[0..l], де i=j=3, k=l=7. Тобто, величина максимального індексу вектору на 1 менша за його розмір. Сутність маргінального індексування у багатовимірному базисі полягає в тому, що кожен з масивів включає в себе посилання на групи масивів нижчого рівня. Тобто, масив найвищого рівня a[0..i] містить i+1 посилань на i+1 масивів b[0..j]. В свою чергу, кожен з масивів b[0..j] містить по j+1 посилань на j+1 масивів c[0..k] і так далі, до рівня груп масивів Q[0..n]. На перший погляд, це здається ускладненням яке не принесе позитивного результату. Але багатовимірна матриця з маргінальним індексуванням застосована тут як засіб реалізації

мови ситуаційного керування, де імена нижчих рівнів понять співставлені за допомогою зв'язків приналежності з іменами понять вищих рівнів, формуючи семантичну форму a→b→c→d→Q для звернення до обмеженого набору вихідних керуючих впливів, зібраних у векторі Q[0..n]. За умови, що групи інформаційних векторів [0..i], b[0..j], c[0..k] та d[0..l] містять лише посилання, на найнижчому рівні досить одного єдиного вектора Q[0..n], що значно спрощує задачу заповнення даними.

Тепер поглянемо на таблицю 3. Серед допустимих імен різних просторів часто зустрічаються комбінації, що означають «збереження попереднього стану» і «нештатний режим». Що стосується стану об'єкту «збереження попереднього стану», немає сенсу його обробляти методом вибірки з матриці вихідних впливів. Якщо замінити всі коди імен «збереження попереднього стану» на недопустиме, -1, то в функціональному блоці «керуючий алгоритм маніпулятора» можна легко відмовитись від такої виборки за допомогою блоку умовного переходу. Керуючись принципом «не нашкодь», усі індекси новостворених груп масивів a[0..i], b[0..j], c[0..k], d[0..l] необхідно за замовчуванням виставляти у -1, то тих пір, поки їх не перевизначить програміст, налагоджувальник, чи зовнішня система конфігурування.

Стани понять «нештатний режим» за аналогією можна також закодувати -1 і не обробляти, довірившись контуру протиаварійного захисту(ПАЗ). Але в умовах реальної експлуатації можуть виникати ситуації, не передбачені у ПАЗ. З цієї причини варто залишити можливість протиаварійного впливу на рівні керуючого алгоритму. Як правило, це один, чітко визначений вплив, спрямований на відключення всіх чи деяких виконавчих органів. У даному випадку це елемент Q[0], хоча як нештатний вплив, він може бути представлений в окремому місці, не в масиві. Введемо ще одне недопустиме ім'я, яке оброблюється алгоритмічно, -2. У випадку її виявлення у індексних масивах, на виходи подається комбінація Q[0]. Окрім зупинки виконавчих органів нештатний вплив, наприклад, може включати подачу сигналу аварії.

Звернімо увагу на ім'я a[0] «станція не готова до роботи». Тут однозначна ситуація, коли жоден виконавчий орган не може бути задіяний, тому можна замінити його на -2, як для нештатного режиму. Зменшивши, таким чином, кількість зв'язків належності, можна звернути увагу на різноманітність допустимих конструкцій МСК для об'єкта «станція маніпулятора».

В області понять a для встановлення зв'язків залишися елементи a[2] і a[3]. Область b залишилася без змін. В області c – c[1], c[2], c[4]. В області d – d[0], d[5], d[6], d[7].

Щоб перейти до стану готовності до роботи зі стану неготовності, станція повинна, за графсетом на рисунку 2, переміститись із піднятим і відкритим захватом у місце завантаження. Тому маємо такі мовні конструкції:

1) a[2]→b[0]→c[1, або 2, або 4]→d[0, або 5, або 6, або 7]→Q[1];
2) a[2]→b[0]→c[1, або 2, або 4]→d[0, або 5, або 6, або 7]→Q[2].

Варіантів, окрім a[2]→b[0] та a[2]→b[1] для роботи чи підготовки до роботи не існує. Інакше система знаходиться у нештатному режимі і потребує втручання з наступним перезапуском.

Насправді, незалежно від кольору заготовки на місці завантаження, від факту одночасної наявності чи відсутності деякої заготовки у захваті, від положення каретки маніпулятора, на момент натискання кнопки «reset» на панелі керування маніпулятор повинен прийняти вихідне положення, а потім подати сигнал-запрошення «start». Тому практичного сенсу реалізовувати таку поліваріативність команд мовою ситуаційного управління немає. Вихід – у ситуації неготовності станції маніпулятора до роботи у блоці нормування і масштабування вхідних сигналів у момент натискання кнопки «reset» замінити інформаційні ознаки сигналів для понять c і d, надавши імена-індекси масивів 1 і 0 відповідно. Перекриття ознак для поняття c повинне протриматись до спрацювання датчика позиції каретки у місці завантаження. Перекриття ознак для поняття d – до переходу системи маніпулятора у стан «працює в автоматичному режимі». Тоді отримаємо:

1) a[2]→b[0]→c[1]→d[0]→Q[1] – захват підняти і відкрити. Каретку рухати у бік точки завантаження;

2) a[2]→b[0]→c[4]→d[0]→Q[2] – захват підняти і відкрити. Каретку зупинити. Подати сигнал-запрошення натиснути «start».

Якщо поглянути на останні мовні конструкції стає очевидним, що для їх реалізації засобами маргінального індексування достатньо використати не повну множину масивів, а лише один ланцюг із чотирьох масивів-індексаторів та одного масиву вихідних впливів, оскільки відмінність виявляється тільки у кінцевому елементі. Але, оскільки впливи Q у ситуації a[2]→b[0]→c[k] різні, а елементи d[0] однакові, знадобиться 2 окремих масиви d[0..7], тобто всього буде 5 масивів. Структура цієї частини масиву вихідних впливів а також інших частин показана на рисунку 4.

Тепер розглянемо режим, заданий ім'ям a[3]. Йому потрібен окремий масив b[0..3]. Розглянемо стан d[1] – «опущений і закритий». Він виникає для трьох випадків:

1) a[3]→b[1]→c[4]→d[6]→Q[5], a[2]→b[1]→c[4]→d[7]→Q[5] – каретка на місці завантаження. Подається група впливів: захват закрити і підняти; каретку зупинити;

2) a[3]→b[1]→c[2]→d[7]→Q[3] – каретка на місці сортування. Подається група впливів: захват підняти і відкрити; каретку зупинити;

3) a[3]→b[1]→c[1]→d[6]→Q[3] – каретка на місці вивантаження. Подається група впливів, аналогічна пункту 2).

Для опису ситуації a[3]→b[1]→c[k] може знадобитися 3 різних масиви d, оскільки для різних положень каретки маніпулятора колір заготовки призводить до різних рішень. Так, для ситуації 1) колір заготовки не має значення, але її відсутність на місці подачі є нештатною ситуацією, тобто тут d[0]=-2. Для ситуації 2) d[6]=-2, оскільки це значить, що не чорна заготовка попала у лоток сортування, чого не повинно бути. Відповідно, для ситуації 3) d[7]=-2.

Тепер розглянемо початок руху каретки у бік вивантаження. Стан «піднятий і закритий». Під час руху змін з точки зору управління не відбувається, оскільки маємо c[0]=-1. Тут існує стан b[0], оскільки маніпулятор несе заготовку:

1) a[3]→b[0]→c[4]→d[6]→Q[6], a[2]→b[2]→c[4]→d[7]→Q[6] – каретка на місці завантаження і маніпулятор підняв заготовку. Подається група впливів: захват закрити і підняти; каретку рухати у бік точки вивантаження;

2) a[3]→b[0]→c[2]→d[7]→Q[7] – каретка на місці сортування, заготовка піднята. Подається група впливів: захват закрити і опустити; каретку зупинити;

3) a[3]→b[0]→c[1]→d[6]→Q[7] – каретка на місці вивантаження, заготовка піднята. Подається група впливів, аналогічна пункту 2).

Аналогічно, для опису Для опису ситуації a[2]→b[0]→c[k] може знадобитися 3 різних масиви d.

Стан «опущений і відкритий» може мати 3 ситуації:

1) a[3]→b[3]→c[4]→d[7]→Q[5] – каретка на місці завантаження. Подається група впливів: захват закрити і підняти; каретку зупинити;

2) a[3]→b[3]→c[2]→d[7]→Q[3] – каретка на місці сортування. Подається група впливів: захват підняти і відкрити; каретку зупинити;

3) a[3]→b[3]→c[1]→d[6]→Q[3] – каретка на місці вивантаження. Подається група впливів, аналогічна пункту 2).

Як бачимо, для ситуацій a[3]→b[1]→c[k] і a[2]→b[3]→c[k] пункти 1), 2), 3) в частині c→d→Q повністю співпали. Тому можна зорієнтувати елементи-вказівники масиву так, щоб використати вже існуючі гілки c→d→Q не вводячи нових. Так отримуємо ефект скорочення витрат пам'яті за рахунок технології маргінального індексування.

Стан «піднятий і відкритий» може мати наступні ситуації:

1) a[3]→b[2]→c[4]→d[0]→-1 – заготовка відсутня. Зводимо цю ситуацію до необроблюваної, вказавши d[0]=-1;

2) a[3]→b[2]→c[4]→d[6]→Q[4] – каретка на місці завантаження і з'явилась нова деталь у місці захоплення. Пройшла витримка часу за ознакою Z_0. Колір заготовки ще не визначено. Подається група впливів: захват відкрити і опустити; каретку зупинити;

3) a[3]→b[2]→c[2]→d[7]→Q[8] – каретка на місці сортування. Подається група впливів: захват підняти і відкрити; каретку рухати в бік завантаження;

4) a[3]→b[2]→c[1]→d[6]→Q[8] – каретка на місці вивантаження. Подається група впливів, аналогічна пункту 3).

Пункти 2), для a[3]→b[2]→c[k] оригінальний за ознакою виходу. Для нього необхідно ввести окремі масиви c[0…7] та d[0..7]. А пункти 3) і 4) еквівалентні за виходами і потребують окремого, але одного масиву d[0..7].

Результати розробки масиву вихідних впливів за сукупністю вхідних інформаційних ознак a[0..i], b[0..j], c[0..k], d[0..l] показано на рисунку 5. Для роботи з елементами Q[i;j;k;l;n] маргінально індексованого масиву, необхідно і достатньо використовувати алгоритм, наведений на рисунку 6.

Висновки. Представлена архітектура обчислювального вузла надає переваг для написання програмного забезпечення для мехатронних пристроїв гірничовидобувної галузі порівняно з рішеннями, рекомендованими розробниками стендів з мехатроніки:

1. Розробка алгоритмів на основі представленої функціональної структури дозволяє структурувати програмне забезпечення вузла. Відділення процесів підготовки вхідної, вихідної інформації та обробки часових величин від основного керуючого алгоритму дозволяє полегшити відлагодження програм при зміні датчиків, засобів інтерфейсу, їх фізичних та інформаційних характеристик. Оскільки основний алгоритм працює с абстрактними інформаційними ознаками, він таких випадках не потребує втручання;

2. Організація роботи з групами інформаційних ознак і вихідних керуючих впливів як з цільними машинними словами обчислювальної системи дозволяє отримати паралельне обчислення і видачу керуючих впливів, що підвищує продуктивність та надійність алгоритму.

3. Використання груп вхідних інформаційних ознак у якості координат масиву готових вихідних керуючих впливів скорочує основний керуючий алгоритм до простого вибору елемента з масиву. У випадку зміни завдання мехатронного пристрою замість зміни алгоритму будуть змінюватись дані масиву керуючих впливів, що на сьогодні може бути виконано не лише програмістом безпосередньо, але й за допомогою системи підготовки даних;

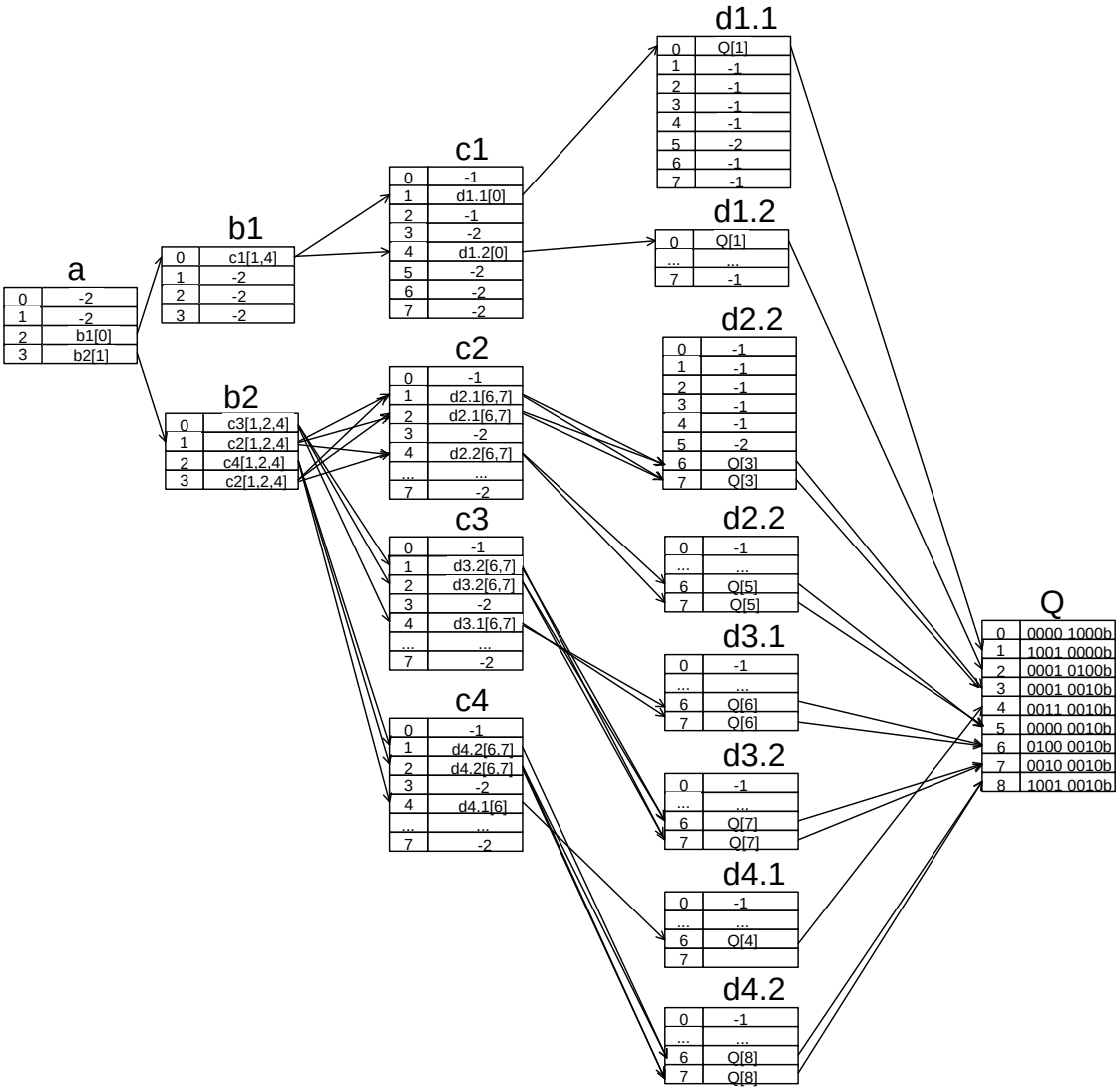

Рисунок 5 – Структура 5-вимірного маргінально індексованого масиву вихідних керуючих впливів станції маніпулятора

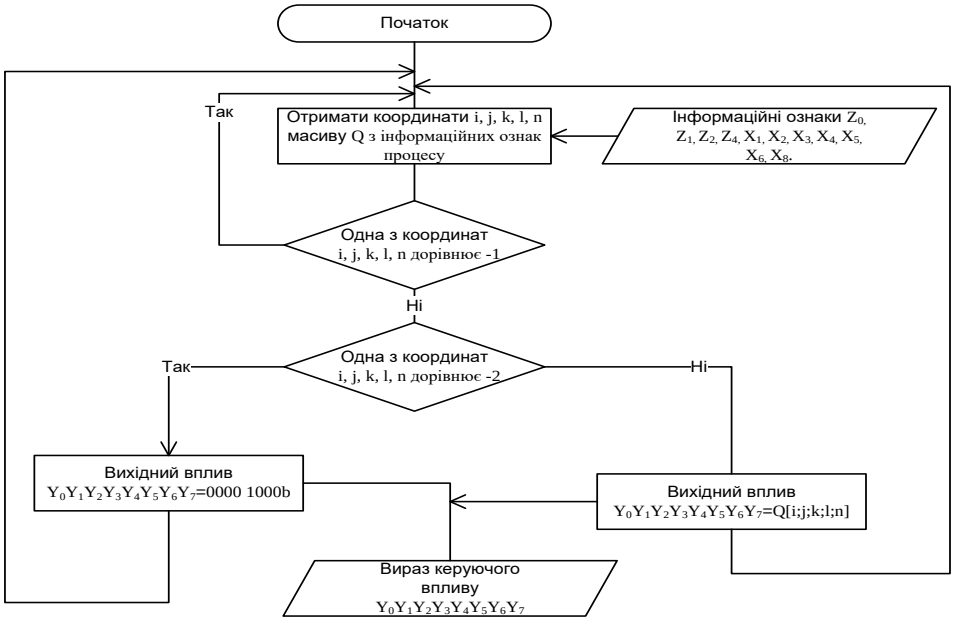

Рисунок 6 – Керуючий алгоритм маніпулятора

4. Організація масиву вихідних впливів як маргінально індексованого дозволяє економити оперативну пам'ять і значно спростити процес редагування самого масиву. Наприклад, кількість елементів кінцевого масиву Q можна збільшити без необхідності будь-яких змін масивів індексації a[0..i], b[0..j], c[0..k], d[0..l]. Оскільки масиви індексації – самостійні елементи, їх також можна додавати, змінювати зв'язки, видаляти в залежності від зміни окремих операцій без шкоди іншим.

Наукова новизна. На основі виразів мови ситуаційного керування, складених з інформаційних ознак стану об'єкту керування, побудовано маргінально індексовий масив продукцій вихідних впливів, який дозволяє замість алгоритму керування за графом використати простий алгоритм вибору з масиву без перевитрат оперативної пам'яті, які виникають у таком рішенні за умови використання інформаційних векторів вихідних впливів.

Практичні результати. Запропоновано метод задання дій маніпулятора, який дозволяє відмовитись від програмування його обчислювального вузла на користь задання структур даних безпосередньо програмістом або засобами автоматичних систем проектування чи налагодження. Метод дозволяє відокремити обчислювальний вузол від мехатронного пристрою і використовувати окремо або замінювати обчислювальні вузли під час експлуатації.

Подальше дослідження планується вести у напрямку розробки алгоритмів підготовки даних для основного керуючого функціонального блоку. Оскільки у реального промислового обладнання можуть бути існують не лише дискретні датчики, але й вимірювальні прилади (ваги, положення, вуглу повороту та ін.), їх дані підлягають обробці і підготовці до надання керуючому алгоритму.

Перелік посилань

1. Программируемый контроллер S7-1200. Системное руководство. – Siemens: 11/2009, A5E02669003-02. – 397 с.
2. Эндрю Таненбаум. Многоуровневая организация ЭВМ. - М.: "Мир", 1979. - 552 с.
3. Station Handhaben. Handbuch. – Festo Didactic GmbH & Co: 655633 DE/EN 06/10 R2.3. – 116 p.
4. FESTO MPS STATIONS. Технічна документація (Керівництво для користувачів). – Wordskills Ukraine 2020: Version 1.0 - 17.02.2020. – 64 p.
5. ГОСТ 34.601-90 Інформаційна технологія. Комплекс стандартів на автоматизовані системи. Автоматизовані системи. Стадії створення.
6. Поспелов Д.А. Ситуационное управление. Теория и практика. – М.: Наука. – Гл. ред. физ. мат. лит., 1986. -288 с.

АННОТАЦИЯ.

Цель работы. Разработать архитектуру вычислительного узла и управляющий алгоритм мехатронного устройства, которые позволяют сократить трудозатраты на программирование и отладки за счет унификации метода программирования.

Методика исследования. Анализ решений, рекомендаций и алгоритмов управления от производителя станции манипулятора. Эскизное проектирование систем в части функциональных структур. Синтез алгоритма и структуры данных с использованием методов ситуационного управления и методов организации хранения данных в памяти компьютера.

Результаты исследования. Предложена общая функциональную структуру вычислительного узла мехатронного устройства, обоснованно ее свойства и преимущества, предложен универсальный подход к написанию алгоритма основного управляющего блока станции манипулятора. Синтезирован язык ситуационного управления для разработки структуры данных в форме, позволяющей гибко менять технологические задачи станции манипулятора. Представлен алгоритм основного управляющего блока программы станции манипулятора, который реализует выборку управляющих воздействий из массива по входным информационными признаками и является универсальным.

Научная новизна. Построен маргинально индексированный массив продукций выходных воздействий, который позволяет вместо алгоритма управления на основе графа использовать простой алгоритм выбора из массива без перерасхода оперативной памяти, которые возникают в таком решении при использовании информационных векторов выходных воздействий.

Практические результаты. Предложен метод задания действий манипулятора, который позволяет отказаться от программирования его вычислительного узла в пользу задания структур данных непосредственно программистом или средствами автоматических систем проектирования или отладки. Метод позволяет отделить вычислительный узел от мехатронного устройства и использовать отдельно или заменять вычислительные узлы во время эксплуатации без перепрограммирования.

Ключевые слова: *Манипулятор, вычислительный узел, комптроллер, функциональная структура, станция манипулятору FESTO, ситуационное управление, маргинальное индексирования, графсет, управляющее воздействие, семантическая связь.*

ABSTRACT

The purpose of the work. Develop the architecture of the computing node and the control algorithm of the mechatronic device, which reduce labor costs for programming and debugging by unifying the programming method.

Research methodology. Analysis of solutions, the recommendations and control algorithms from the manufacturer of the manipulator station. Sketch design of systems in terms of functional structures. Synthesis of algorithm and data structure using situational control methods and methods of organizing data storage in computer memory.

Research results. The general functional structure of the computing unit of the mechatronic device is offered, its properties and advantages are substantiated, the universal approach to writing of algorithm of the basic control block of station of the manipulator is offered. The situational control language has been synthesized for the development of a data structure in a form that allows to flexibly change the technological tasks of the manipulator station. The algorithm of the main control unit of the manipulator station program is presented, which implements the sampling of control influences from the array according to the input information features and is universal.

Scientific novelty. A marginally indexed array of output products is constructed, which allows using a simple algorithm of selection from the array instead of graphical control without overuse of RAM, which arises in such a solution when using information vectors of output effects.

Practical results. A method for setting the actions of the manipulator is proposed, which allows to abandon the programming of its computing node in favor of setting data structures directly by the programmer or by means of automatic design or debugging systems. The method allows to separate the computing node from the mechatronic device and to use it separately or to replace the computing nodes during operation without reprogramming.

Keywords: *Manipulator, computing node, comptroller, functional structure, FESTO manipulator station, situational control, marginal indexing, graphset, control effect, semantic connection.*

Рекомендовано до друку: д-ром техн. наук, професором Ткачевив В.В.