**1.Завдання та обґрунтування варіанту**

Необхідно записати номер залікової книжки у двійковій системі та відповідно до номеру визначити варіант, а також записати два числа

 і ,

де  - двійкові цифри номера залікової книжки у двійковій системі числення ( - молодший розряд).

Над цими числами потрібно виконати 8 операцій у форматі з плаваючою комою. Для обробки мантис в кожній операції потрібно подати:

1) теоретичне обґрунтування способу;

2) операційну схему;

3) змістовний мікроалгоритм;

4) таблицю станів регістрів (лічильника), довжина яких забезпечує одержання 15 основних розрядів мантиси результату;

5) функціональну схему з відображенням управляючих сигналів;

6) граф управляючого автомата Мура з кодами вершин;

7) обробку порядків (показати у довільній формі);

8) форму запису нормалізованого результату з плаваючою комою в пам’ять.

Операцію додавання до етапу нормалізації результату можна проілюструвати у довільній формі. Вказані пункти виконати для етапу нормалізації результату з урахуванням можливого нулевого результату.

Також для операції з кодом  необхідно побудувати управляючий автомат Мура на тригерах (тип обрати самостійно) та елементах булевого базису.

Визначення варіанту завдання:

Номер залікової книжки №9123. У двійковій системі: 10 0011 1010 0011

Х2=-1111 1011, 0000 111 Y2=+1111 1,0110000111

=011, що відповідає четвертому способу множення.

**2.Виконання завдання**

2.1. Запишемо числа X2 та Y2 у вигляді з плаваючою комою, як вони зберігаються у пам'яті (з мантисою, порядком та характеристикою):

Mx=1,1111 1011 0000 111 Px=0.0001000 Ex=Px+28=1 0000 1000

My=0,1111 1011 0000 111 Py=0.0000101 Ey=Py+25=100101

При множенні двох чисел з плаваючою комою потрібно виконати обробку порядку результату, склавши результати початкових чисел, а мантису результату отримуємо перемноживши мантиси початкових чисел як два цілих числа.

2.2. Перший спосіб множення чисел.

За першим способом множення здійснюється молодшими розрядами множника, сума часткових добутків при цьому зсувається вправо. В першому такті і-того циклу аналізується значення RG2(1) – молодшого розряду регістру RG2, в якому знаходиться чергова цифра множника. Вміст RG3 додається до суми часткових добутків, що знаходяться в регістрі RG1, якщо молодший розряд регістру множника =1, якщо 0 – не додається. В другому такті здійснюється зсув праворуч в регістрах RG1 i RG2. За зсуву цифра молодшого розряду регістру RG1 записується у старший розряд RG2.

Операційна схема:

Змістовний мікроалгоритм:



Таблиця станів регістрів та лічильника:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № ц | R1(z) | R2(y) | R3(x) | CT | Коментар |
| П.с. | 0000000000000000 | 111110110000111 | 0111110110000111 | 1111 | Встановлення п.с. |
| 1 | 0111110110000111  0011111011000011 | 011111011000011 |  | 1110 | RG1=RG1+RG3  RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 2 | 1011110001001010  0101111000100101 | 001111101100001 |  | 1101 | RG1=RG1+RG3  RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 3 | 1101101110101100  0110110111010110 | 000111110110000 |  | 1100 | RG1=RG1+RG3  RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 4 | 0011011011101011 | 000011111011000 |  | 1011 | RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 5 | 0001101101110101 | 100001111101100 |  | 1010 | RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 6 | 0000110110111010 | 110000111110110 |  | 1001 | RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 7 | 0000011011011101 | 011000011111011 |  | 1000 | RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 8 | 1000010001100100  0100001000110010 | 001100001111101 |  | 0111 | RG1=RG1+RG3  RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 9 | 1011111110111001  0101111111011100 | 100110000111110 |  | 0110 | RG1=RG1+RG3  RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 10 | 0010111111101110 | 010011000011111 |  | 0101 | RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 11 | 1010110101110101  0101011010111010 | 101001100001111 |  | 0100 | RG1=RG1+RG3  RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 12 | 1101010001000001  0110101000100000 | 110100110000111 |  | 0011 | RG1=RG1+RG3  RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 13 | 1110011110100111  0111001111010011 | 111010011000011 |  | 0010 | RG1=RG1+RG3  RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 14 | 1111000101011010  0111100010101101 | 011101001100001 |  | 0001 | RG1=RG1+RG3  RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |
| 15 | 1111011000110100  0111101100011010 | 001110100110000 |  | 0000 | RG1=RG1+RG3  RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |

Z=1111011000110100011101001100001

Функціональна схема:

Закодований мікроалгоритм:

Для побудови закодованого алгоритму потрібно позначити мікрооперації та логічні умови управляючими сигналами.

|  |  |
| --- | --- |
| Зміст мікрооперації | Управляючий сигнал |
| R1:=0 | Rz |
| R2:=Y | Wy |
| R3:=X | Wx |
| CT:=n | W |
| R1:=R1+R3 | Wct |
| R1:=0.r[R1] | SRz |
| R2:=R1(1).r[R2] | SRy |
| CT:=CT-1 | Dct |

|  |  |
| --- | --- |
| Логічна умова | Сигнал |
| R2(1) | x1 |
| CT=0 | x2 |

Побудуємо закодований мікроалгоритм, замінивши сукупності операцій, які не повторюються окремо одна від одної символами A, B, C.



а1

а4

а3

а2

а1

На основі закодованого алгоритму побудуємо граф автомату Мура. Оскільки у графі матимемо петлю, що замикається на ту ж саму вершину, то добавимо додаткову вершину, щоб був перепад сигналу.



Обробимо порядок результату Pz:

Pz= Px+ Py=0.0001101

Оскільки у пам’ять можна записати лише 16 розрядів разом зі знаковим, то округлюємо результат. Похибки не буде.

Отже, у пам’ять результат запишеться у вигляді:

Mz=1, 111 1011 0001 1010 Pz= Px+ Py=0.0001101

2.3. Другий спосіб множення чисел.

Множення виконується молодшими розрядами. Вміст RG2 під час множення зсувається вправо, для накопичення суми часткових добутків застосовується регістр подвійної довжини. Молодші n розрядів RG3 – x, старші – нулі. RG3 зсувається вліво. Закінчення операції множення визначається за нульовим вмістом RG2.

Операційна схема:

Змістовний мікроалгоритм:



Таблиця станів регістрів(див.наступ.стор).

Функціональна схема:

Закодований мікроалгоритм:

Для побудови закодованого алгоритму потрібно позначити мікрооперації та логічні умови управляючими сигналами.

|  |  |
| --- | --- |
| Зміст мікрооперації | Управляючий сигнал |
| R1:=0 | Rz |
| R2:=Y | Wy |
| R3:=X | Wx |
| R1:=R1+R3 | Wct |
| R2:=0.r[R2] | SRy |
| R3:=l[R3].0 | SLx |

|  |  |
| --- | --- |
| Логічна умова | Сигнал |
| R2(1) | x1 |
| R2=0 | x2 |

Побудуємо закодований мікроалгоритм, замінивши сукупності операцій, які не повторюються окремо одна від одної символами A, B, C.



а1

а4

а3

а2

а1

На основі закодованого алгоритму побудуємо граф автомату Мура. Оскільки у графі матимемо петлю, що замикається на ту ж саму вершину, то добавимо додаткову вершину, щоб був перепад сигналу.



Обробимо порядок результату Pz:

Pz= Px+ Py=0.0001101

Оскільки у пам’ять можна записати лише 16 розрядів разом зі знаковим, то округлюємо результат. Похибки не буде.

Отже, у пам’ять результат запишеться у вигляді:

Mz=1, 111 1011 0001 1010 Pz= Px+ Py=0.0001101

2.4. Третій спосіб множення

Множення складається з n циклів. Цей спосіб подібний першому, але має більшу складність. Перевагою можна вважати сумісність апаратури з пристроями ділення першим способом.

Операційна схема:

Змістовний мікроалгоритм:



Таблиця станів регістрів та лічильника:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № ц | R1(x) | R2(z) | R3(y) | CT | Коментар |
| П.с. | 1111101100001110 | 000000000000000 | 111110110000111 | 1111 | Встановлення п.с. |
| 1 | 1111101100001110  1111011000011101 | 111110110000111  111000100101010 |  | 1110 | R1.R2:=R1.R2+0..0.R3  R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 2 | 1111011000011110  1110110000111101 | 111100010010101  111000100101010 |  | 1101 | R1.R2:=R1.R2+0..0.R3  R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 3 | 1110110000111110  1101100001111101 | 110111010110001  101110101100010 |  | 1100 | R1.R2:=R1.R2+0..0.R3  R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 4 | 1101100001111110  1011000011111101 | 101101011101001  011010111010010 |  | 1011 | R1.R2:=R1.R2+0..0.R3  R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 5 | 1011000011111110  0110000111111100 | 011001101011001  110011010110010 |  | 1010 | R1.R2:=R1.R2+0..0.R3  R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 6 | 1100001111111001 | 100001111101100 |  | 1001 | R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 7 | 1100001111111010  1000011111110101 | 100101011101011  001010111010110 |  | 1000 | R1.R2:=R1.R2+0..0.R3  R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 8 | 1000011111110110  0000111111101100 | 001001101011101  010011010111010 |  | 0111 | R1.R2:=R1.R2+0..0.R3  R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 9 | 0001111111011000 | 100110101110100 |  | 0110 | R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 10 | 0011111110110001 | 001101011101000 |  | 0101 | R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 11 | 0111111101100010 | 011010111010000 |  | 0100 | R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 12 | 1111111011000100 | 110101110100000 |  | 0011 | R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 13 | 1111111011000101  1111110110001011 | 110100100100111  101001001001110 |  | 0010 | R1.R2:=R1.R2+0..0.R3  R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 14 | 1111110110001100  1111101100011001 | 100111111010101  001111110101010 |  | 0001 | R1.R2:=R1.R2+0..0.R3  R1.R2:=l[R1.R2].0  CT:=CT-1 |
| 15 | 11110 11000 11010 | 001110100110001 |  | 0000 | RG1=RG1+RG3  RG1=0.r[RG1]  RG2=RG1(1).r[RG2] |

Функціональна схема:

Закодований мікроалгоритм:

Для побудови закодованого алгоритму потрібно позначити мікрооперації та логічні умови управляючими сигналами.

|  |  |
| --- | --- |
| Зміст мікрооперації | Управляючий сигнал |
| R1:=X.0 | Wx |
| R2:=0 | Rz |
| R3:=Y | Wy |
| R1.R2:=R1.R2+0..0R3 | Wct |
| R1.R2:=l[R1.R2].0 | SLz |
| CT:=n | W |
| CT:=CT-1 | Dct |

|  |  |
| --- | --- |
| Логічна умова | Сигнал |
| R1(0) | x1 |
| CT=0 | x2 |

Побудуємо закодований мікроалгоритм, замінивши сукупності операцій, які не повторюються окремо одна від одної символами A, B, C.



а1

а4

а3

а2

а1

На основі закодованого алгоритму побудуємо граф автомату Мура. Оскільки у графі матимемо петлю, що замикається на ту ж саму вершину, то добавимо додаткову вершину, щоб був перепад сигналу.



Обробимо порядок результату Pz:

Pz= Px+ Py=0.0001101

Оскільки у пам’ять можна записати лише 16 розрядів разом зі знаковим, то округлюємо результат. Похибки не буде.

Отже, у пам’ять результат запишеться у вигляді:

Mz=1, 111 1011 0001 1010 Pz= Px+ Py=0.0001101

2.5. Четвертий спосіб множення

Множення відбувається із старших розрядів множника, сума часткових добутків нерухома, множник зсувається праворуч. Множення відбувається в n циклах.

Операційна схема:

Змістовний мікроалгоритм:



Таблиця станів регістрів(див.наступ.стр).

Функціональна схема:

Закодований мікроалгоритм:

Для побудови закодованого алгоритму потрібно позначити мікрооперації та логічні умови управляючими сигналами.

|  |  |
| --- | --- |
| Зміст мікрооперації | Управляючий сигнал |
| R1:=0 | Rz |
| R2:=X | Wx |
| R3:=Y | Wy |
| R1:=R1+R3 | Wct |
| R2:=l[R2].0 | SLx |
| R3:=0.r[R3] | SRy |

|  |  |
| --- | --- |
| Логічна умова | Сигнал |
| R2(1) | x1 |
| R2=0 | x2 |

Побудуємо закодований мікроалгоритм, замінивши сукупності операцій, які не повторюються окремо одна від одної символами A, B, C.



а1

а4

а3

а2

а1

На основі закодованого алгоритму побудуємо граф автомату Мура. Оскільки у графі матимемо петлю, що замикається на ту ж саму вершину, то добавимо додаткову вершину, щоб був перепад сигналу.



Обробимо порядок результату Pz:

Pz= Px+ Py=0.0001101

Оскільки у пам’ять можна записати лише 16 розрядів разом зі знаковим, то округлюємо результат. Похибки не буде.

Отже, у пам’ять результат запишеться у вигляді:

Mz=1, 111 1011 0001 1010 Pz= Px+ Py=0.0001101

2.6. Перший спосіб ділення чисел.

При діленні чисел з плаваючою комою також окремо визначається порядок результату и мантиса. При діленні мантис Z=X/Y потрібно враховувати, що мантиса Х повинна бути меншою за мантису Y. Для нормалізації мантису Х зсувають на один розряд праворуч, а порядок збільшують на 1. Можливе округлення результату ділення.

Нормалізація модуля числа Х(оскільки спочатку рахуємо модуль результату, а потім визначаємо його знак):

Mx=0,01111 1011 0000 111 Px=0.0001001

Перший спосіб ділення апаратно ефективно суміщається із третім способом множення. Зсувається вліво залишок, а дільник залишаються нерухомим.

Операційна схема:

Змістовний мікроалгоритм:



Таблиця станів регістрів:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №ц | R1 | R2 | R3 | Коментар |
| П.С. | 00000000000000000 | 000111110110000111 | 000111110110000111 | Встановлення п.с. |
| 1 | 00000000000000001 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 2 | 00000000000000011 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 3 | 00000000000000111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 4 | 00000000000001111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 5 | 00000000000011111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 6 | 00000000000111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 7 | 00000000001111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 8 | 00000000011111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 9 | 00000000111111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 10 | 00000001111111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 11 | 00000011111111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 12 | 00000111111111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 13 | 00001111111111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 14 | 00011111111111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 15 | 00111111111111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 16 | 01111111111111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |
| 17 | 11111111111111111 | 001111101100001110  000111110110000111 |  | R1:=l[R1].!R2(n+2)  R2:=l[R2].0  R2:=R2+!R3+CI |

Результатом буде Z=1, оскільки мантиси чисел однакові.

Функціональна схема:

Закодований мікроалгоритм:

Для побудови закодованого алгоритму потрібно позначити мікрооперації та логічні умови управляючими сигналами.

|  |  |
| --- | --- |
| Зміст мікрооперації | Управляючий сигнал |
| R1:=0 | Rz |
| R2:=X | Wx |
| R3:=Y | Wy |
| R1:=l[R1].!R2(n+2) | SLz |
| R2:=l[R2].0 | SLx |
| R2:=R2+!R3+CI | Wx1 |
| R2:=R2+R3 | Wx2 |

|  |  |
| --- | --- |
| Логічна умова | Сигнал |
| R2(n+2) | x1 |
| R1(n+1) | x2 |

Побудуємо закодований мікроалгоритм, замінивши сукупності операцій, які не повторюються окремо одна від одної символами A, B, C,D.



А1

А5

А4

А3

А2

А1

На основі алгоритму побудуємо граф автомата Мура:



Обробимо порядки:

Pz=Px-Py=0.0000100

Отже, у пам’ять результат запишеться у вигляді:

Mz=1,100000000000000 Pz=0.0000100

2.7.Ділення другим способом.

Другий спосіб ділення програє першому у часі; його вигідно використовувати з четвертим способом множення. Під час ділення зсувається дільник.

Операційна схема:

Змістовний мікроалгоритм:



Таблиця станів регістрів(див.наступ.стр).

Функціональна схема:

Закодований мікроалгоритм:

Для побудови закодованого алгоритму потрібно позначити мікрооперації та логічні умови управляючими сигналами.

|  |  |
| --- | --- |
| Зміст мікрооперації | Управляючий сигнал |
| R1:=0..01 | Rz |
| R2:=00X | Wx |
| R3:=00Y | Wy |
| R2:=R2+!R3+CI | Wx1 |
| R2:=R2+R3 | Wx2 |
| R3:=0.r[R3] | SRy |
| R1:=l[R1].sm(p) | SLz |

|  |  |
| --- | --- |
| Логічна умова | Сигнал |
| R2(2n+1) | x1 |
| R1(n+1) | x2 |

Побудуємо закодований мікроалгоритм, замінивши сукупності операцій, які не повторюються окремо одна від одної символами A, B, C,D.



На основі алгоритму побудуємо граф автомата Мура:



Обробимо порядки:

Pz=Px-Py=0.0000100

Отже, у пам’ять результат запишеться у вигляді:

Mz=1,100000000000000 Pz=0.0000100

2.8. Обчислення кореня

Будемо обчислювати корінь з модуля X2. Для обчислення кореня порядок числа з плаваючою комою повинен ділитися на 2 без остачі. Порядок цього числа дорівнює 8 у десятинному коді, а отже ділиться на 2 без остачі. Корінь можна обчислити, якщо результат формувати згідно з виразом

0<=(X-Zi2)2i-1<Zi+2-i-1

Обчислювати корінь можна з відновленням та без відновлення від’ємного залишку.

Операційна схема:

Змістовний мікроалгоритм:



Таблиця станів регістрів:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № ц | R1(z) | R2(y) | R3(x) | CT | Коментар |
| П.с. | 000000000000000 | 00000000000000000 | 111110110000111 | 1111 | Встановлення п.с. |
| 1 | 000000000000001 | 00000000000000011  +11111111111111111  00000000000000010 | 111011000011100 | 1110 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+!R1+11  R1:=l[R1].!R2(n+2) |
| 2 | 000000000000011 | 00000000000001011  +11111111111111011  00000000000000110 | 101100001110000 | 1101 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+!R1+11  R1:=l[R1].!R2(n+2) |
| 3 | 000000000000111 | 00000000000011010  +11111111111110011  00000000000001101 | 110000111000000 | 1100 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+!R1+11  R1:=l[R1].!R2(n+2) |
| 4 | 000000000001111 | 00000000000110111  +11111111111100011  00000000000011010 | 000011100000000 | 1011 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+!R1+11  R1:=l[R1].!R2(n+2) |
| 5 | 000000000011111 | 00000000001101000  +11111111111000011  00000000000101011 | 001110000000000 | 1010 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+!R1+11  R1:=l[R1].!R2(n+2) |
| 6 | 000000000111111 | 00000000010101100  +11111111110000011  00000000000101111 | 111000000000000 | 1001 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+!R1+11  R1:=l[R1].!R2(n+2) |
| 7 | 000000001111110 | 00000000010111111  +11111111100000011  11111111111000010 | 100000000000000 | 1000 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+!R1+11  R1:=l[R1].!R2(n+2) |
| 8 | 000000011111101 | 11111111100001010  +00000000111111011  00000000100000101 | 000000000000000 | 0111 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+R1+11  R1:=l[R1].!R2(n+2) |
| 9 | 000000111111011 | 00000010000010100  +11111110000001011  00000000000011111 | 000000000000000 | 0110 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+!R1+11  R1:=l[R1].!R2(n+2) |
| 10 | 000001111110110 | 00000000001111100  +11111100000010011  11111100010001111 | 000000000000000 | 0101 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+!R1+11  R1:=l[R1].!R2(n+2) |
| 11 | 000011111101100 | 11110001000111100  +00000111111011011  11111001000010111 | 000000000000000 | 0100 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+R1+11  R1:=l[R1].!R2(n+2) |
| 12 | 000111111011000 | 11100100001011100  +00001111110110011  11110100000001111 |  | 0011 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+R1+11  R1:=l[R1].!R2(n+2) |
| 13 | 001111110110000 | 11010000000111100  +00011111101100011  11101111110011111 |  | 0010 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+R1+11  R1:=l[R1].!R2(n+2) |
| 14 | 011111101100000 | 10111111001111100  +00111111011000011  11111110100111111 |  | 0001 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+R1+11  R1:=l[R1].!R2(n+2) |
| 15 | 111111011000001 | 11111010011111100  +01111110110000011  01111001001111111 |  | 0000 | R2:=l2[R2].R3(n).R3(n-1)  R3:=l2[R3].00  R2:=R2+R1+11  R1:=l[R1].!R2(n+2) |

Z=111111011000001

Функціональна схема:

Закодований мікроалгоритм:

Для побудови закодованого алгоритму потрібно позначити мікрооперації та логічні умови управляючими сигналами.

|  |  |
| --- | --- |
| Зміст мікрооперації | Управляючий сигнал |
| R1:=0 | Rz |
| R2:=0 | W2 |
| R3:=00x | Wx |
| CT:=0 | W |
| R2:=l2[R2].R3(n).R3(n-1) | SL2 |
| R3:=l2[R3].00 | SLx |
| R2:=R2+!R1+11 | Wr1 |
| R2:=R2+R1+11 | Wr2 |
| CT:=CT-1 | Dct |
| R1:=l[R1].!R2(n+2) | SLz |

|  |  |
| --- | --- |
| Логічна умова | Сигнал |
| R2(2n+2) | x1 |
| CT=0 | x2 |

Побудуємо закодований мікроалгоритм, замінивши сукупності операцій, які не повторюються окремо одна від одної символами A, B, C,D,E.



На основі закодованого алгоритму побудуємо граф автомата Мура:



Порядок результату визначається діленням порядку аргумента на 2:

Pz=Px/2=8/2=410=0.00001002

Отже, у пам’ять результат запишеться у вигляді:

Mz=0,111111011000001 Pz=0.0000100

2.9. Додавання чисел.

Додавання чисел виконується в оберненому та доповнювальному кодах. Виконаємо додавання в оберненому коді. Для цього числа потрібно перевести в обернений код. Оскільки можливе переповнення розрядної сітки, то переведемо їх в модифікований код, тобто код, в якому для знаку відведено два розряди. При додаванні чисел з плаваючою комою потрібно вирівняти їх порядки, додати мантиси та за необхідності нормалізувати результат. Можливе округлення результату.

Pz=max(Px;Py)=810=0.00010002

Потрібно зсунути Y вправо на три розряди та збільшити його порядок на 3:

My=0,000111110110000111 Py=0.0001000

Переведемо числа в модифікований обернений код:

Mx=11.000001001111000 My=00.000111110110000111

Mz=11.000001001111000000

+ 00.000111110110000111

11.001001000101000111+1=11.001001000101001000

Додаємо 1 в молодший розряд для корекції. Переведемо мантису у прямий код та запишемо у вигляді як вона запишеться у пам’ять:

Mz=11.110110111010110111

Округлюємо результат до 15 розрядів після коми:

Mz=1,110110111010110 Pz=0.0000100

**3. Побудова автомату Мура для вказаної операції (четвертий спосіб множення).**

Згідно з графом автомату та таблицею переходів для D-тригера побудуємо структурну таблицю.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Перехід | Старий стан  Q3Q2Q1 | Новий стан  Q3Q2Q1 | Вхідні сигнали  x2x1 | Вихідні сигнали  ABC | Функції тригерів  D3D2D1 |
| a1-a2 | 000 | 001 | -- | 000 | 001 |
| a2-a3 | 001 | 011 | -1 | 100 | 011 |
| a2-a4 | 001 | 010 | -0 | 100 | 010 |
| a3-a4 | 011 | 010 | -- | 010 | 010 |
| a4-α | 010 | 110 | 00 | 001 | 110 |
| a4-a3 | 010 | 011 | 01 | 001 | 011 |
| a4-a1 | 010 | 000 | 1- | 001 | 000 |
| α-a4 | 110 | 010 | -- | 000 | 010 |

Робимо мінімізацію функцій тригерів та вихідних сигналів:



В результаті отримуємо:

A=!Q3!Q2Q1; B=!Q3Q2Q1; C=!Q3Q2!Q1; D1=!Q3Q2!Q1!x2!x1; D2=!Q3Q1 !Q3Q2!x1!x2 Q3Q2!Q!; D3=!Q3!Q2 !Q3Q2!Q1!x2!x1

Будуємо схему автомата: