|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления» (ИУ)

КАФЕДРА «Информационная безопасность» (ИУ8)

Домашнее задание №3

По дисциплине: «Аппаратные средства вычислительной техники»

**Тема: «Разработка рабочей модели процессора»**

**Вариант №2**

Выполнил: Анищенков В. А.,

Студент группы ИУ8-61

Проверил: Рафиков А. Г.,

Старший преподаватель кафедры ИУ

**Задание и условия выполнения ДЗ №3**

Разработать модель специализированного процессора, использовать ИС малой и средней степени интеграции из библиотеки элементов Proteus VSM. УА необходимо синтезировать, тип УА для своего варианта определить из приведенных ниже таблиц, в качестве ОА использовать1804ВС1 или 589 ИК02, или синтезировать оригинальный ОА.

Управляющий автомат – МПА с принудительной адресацией (ПА).

Алгоритм для моделирования – разработать микропрограмму, которая выполняет операции с разрядами РОНi и РОНj, нумерация разрядов регистров i и j - сквозная: определяет разряды с 0 в коде регистров РОНi и РОНj и записывает А в регистры с номерами, соответствующими номерам единичных разрядов.

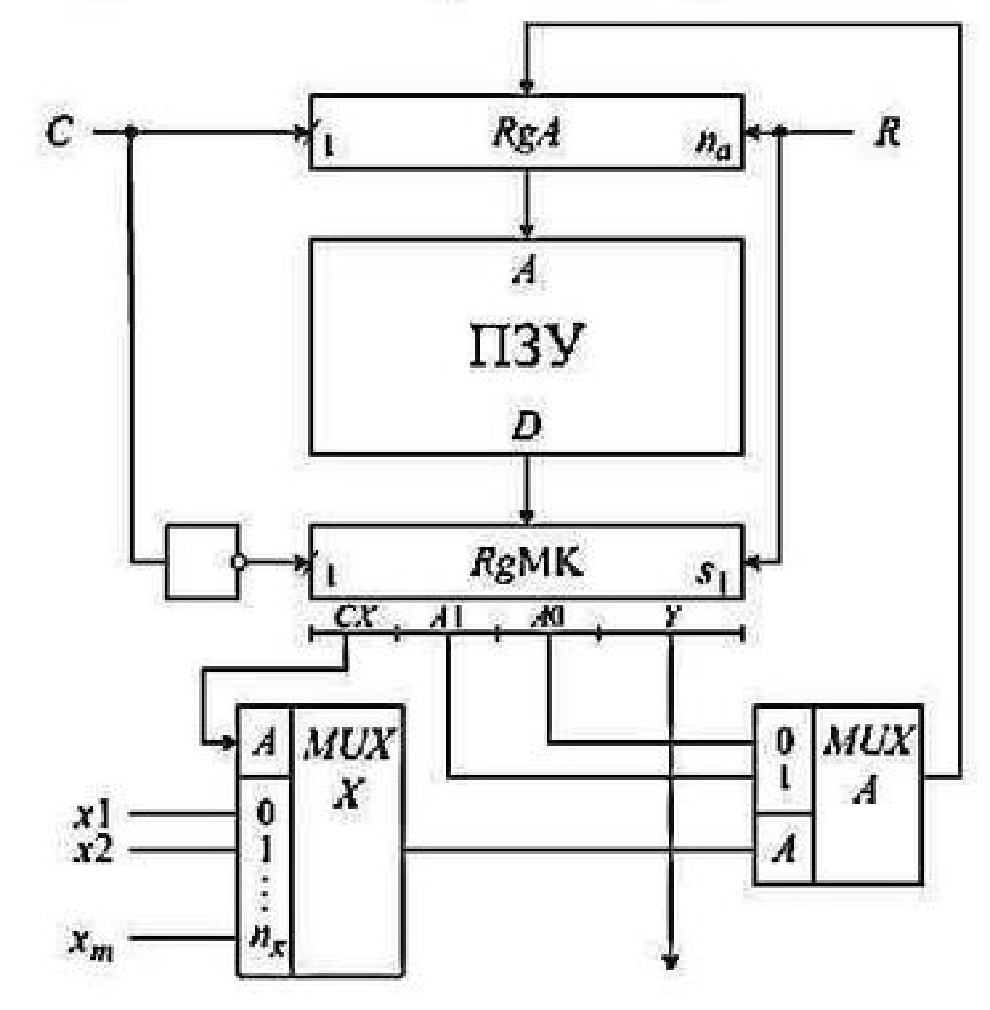


Рисунок 1. Схема МПА с принудительной адресацией.

В качестве ОА используем схему КМ1804. Особенность МПА с принудительной адресацией заключается в том, что адрес следующей микрокоманды в ПЗУ определяется путем выбора соответствующего флага на мультиплексоре флагов.

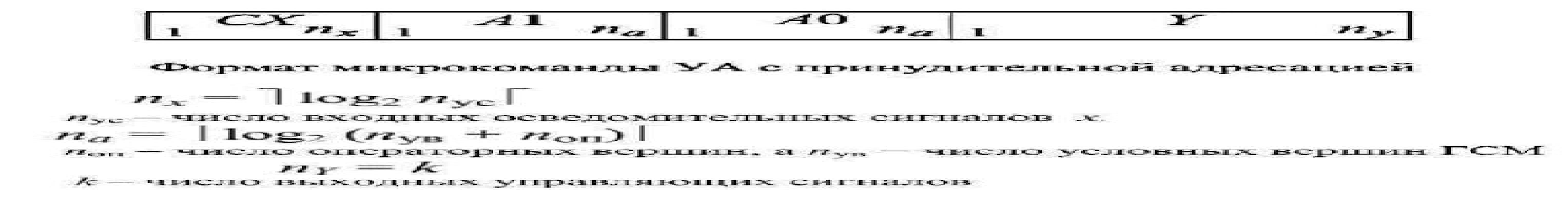


Рисунок 2. Формат микрокоманд МПА с принудительной адресацией

*Таблица 1 – Осведомительные сигналы*

|  |  |  |  |
| --- | --- | --- | --- |
| CA2 | CA1 | CA0 | Flag |
| 0 | 0 | 0 | NZ |
| 0 | 0 | 1 | CONST1 |
| 0 | 1 | 0 |  |
| 0 | 1 | 1 | LDM |
| 1 | 0 | 0 | Z |
| 1 | 1 | 1 | ENDP |

В качестве источников операндов выступают входы D0-D3 регистры внутри микросхемы, адреса которых задаются входами А0-А3 и В0-В3. Сами операции задаются на входах I0-I8. Результаты операций выводится на выходы Y0-Y3. Данной конфигурацией обладает каждый из использованных KM1804.

Для реализации были прошиты элементы EPROM и использована внешняя библиотека для операционного автомата.

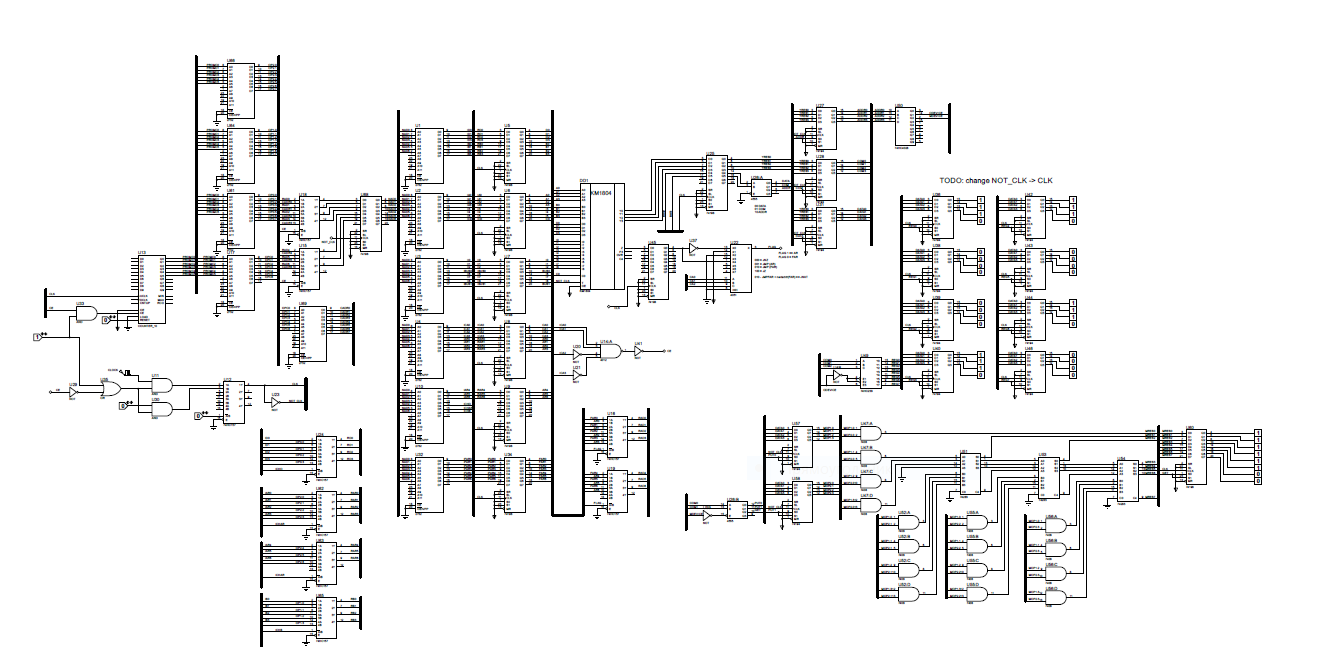


Рисунок 3. Схема процессора

Максимальная тактовая частота, на которой схема продолжает стабильно работать – 1.2 Мгц.

**Описание алгоритма выполнения вычислительного задания:**

Задача **–** разработать микропрограмму, которая выполняет операции с разрядами РОНi и РОНj, нумерация разрядов регистров i и j – сквозная: определяет разряды с 0 в коде регистров РОНi и РОНj и записывает А в регистры с номерами, соответствующими номерам единичных разрядов;

Алгоритм выполнения вычислительной задачи c комментариями представлен на рисунке 4.

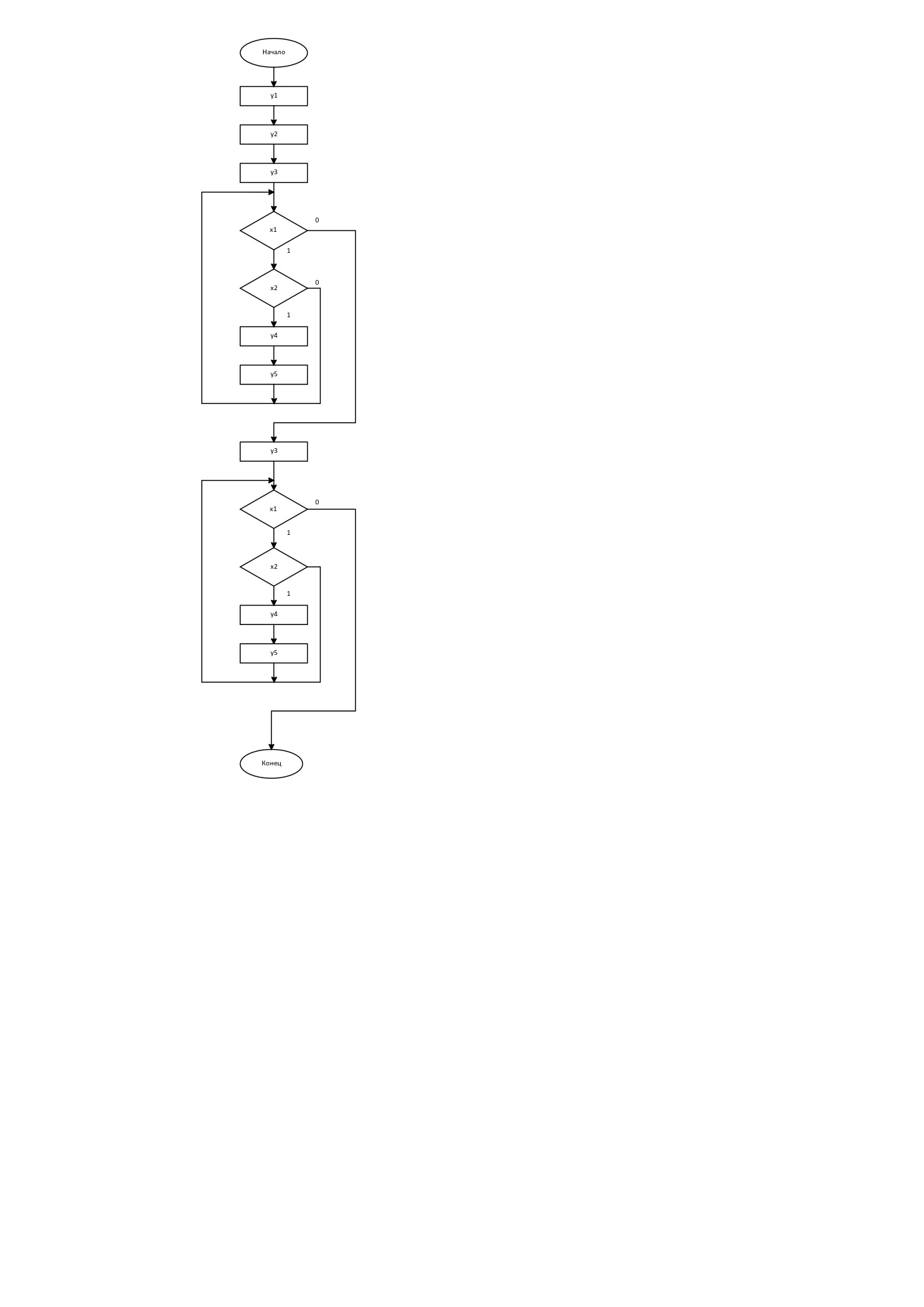


Рисунок 4. Алгоритм работы программы для задания 1

**Техническая реализация**

Переданные из ПЗУ команды выполняют КМ1804 – DD1. В результате выполнения данных команд на каждом такте работы получается некоторый набор флагов, наиболее существенными из которых, для выполнения данной задачи, являются JZ и JNZ – флаги, сохраняющиеся в регистр флагов U22.

В зависимости от выведенного значения флага выбирается адрес перехода на следующую микрокоманду.

Для реализации алгоритма используется разработанный набор команд, для каждой из которых в микропрограммной памяти хранится соответствующая микропрограмма. Содержание микропрограммной памяти представлено на рисунке 5.

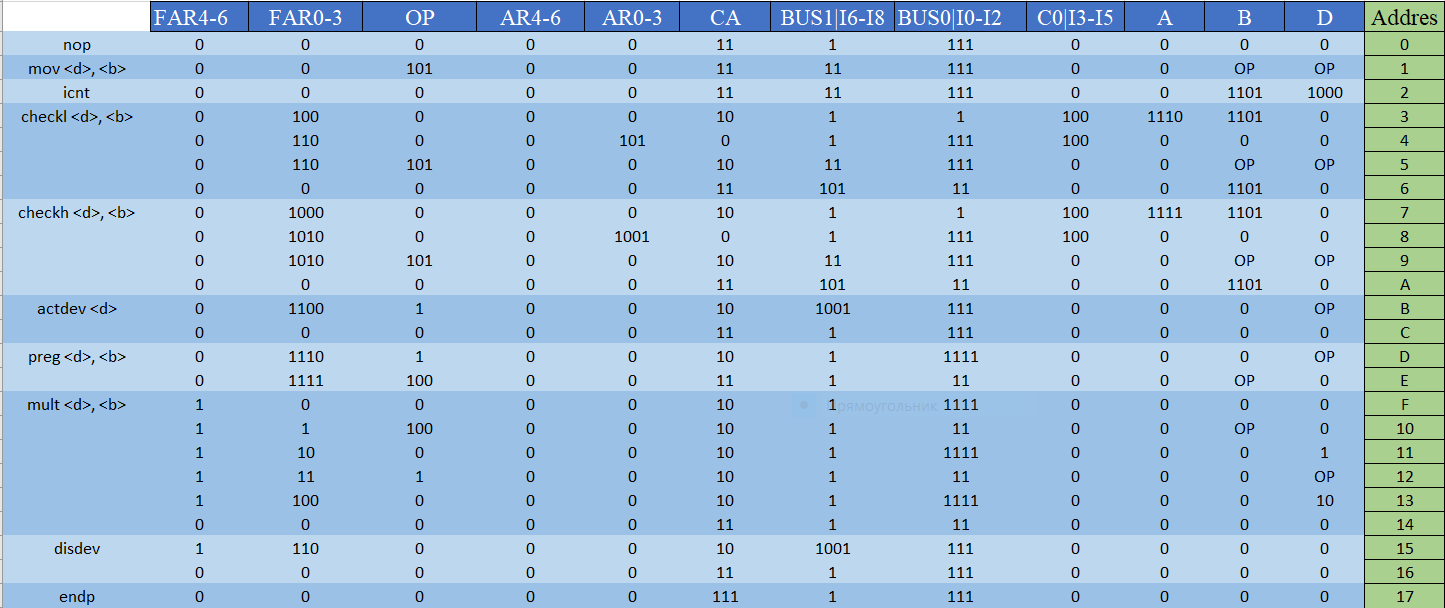


Рисунок 5. Содержимое микропрограммной памяти

Соответственно, коды команд и адреса реализации имеют следующее соответствие:

*Таблица 2 – Коды и адреса команд*

|  |  |  |
| --- | --- | --- |
| Семантика команды | Код команды | Адрес команды |
| NOP | 0000 | 0 |
| MOV <d>, <b> | 0001 | 1 |
| ICNT | 0010 | 10 |
| CHECKL <d>, <b> | 0011 | 11 |
| CHECKE <d>, <b> | 0100 | 111 |
| ACTDEV <d> | 0101 | 1011 |
| PREG <d>, <b> | 0110 | 1101 |
| MULT <d>, <b> | 0111 | 1111 |
| DISDEV | 1000 | 10101 |
| ENDP | 1001 | 10111 |

Для проверки работоспособности программы достаточно наблюдать значения на выходах Y во время работы алгоритма. Кроме того, значения внутренних регистров КМ для каждой команды отражены в логах Proteus.

Демонстрация работы схемы:

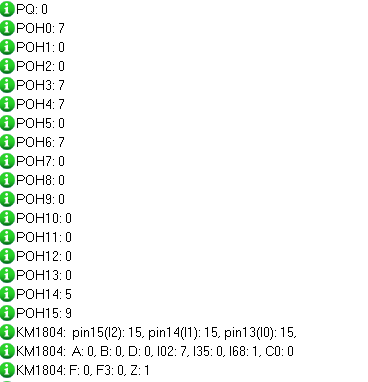


Рисунок 4. Логгирование операционной части

В данном случае регистры i=14 (старшие 4 бита) и j=15 (младшие 4 бита), принимающие значения, проверка которых и происходит в алгоритме были заполнены 0101 1001 для DD1. Значение, записываемое в регистры: A=0111.

В регистры, соответствующие номерам единичных разрядов записано переданное из программной памяти число А в качестве операнда. Программа для выполнения задания 1 представлена в приложении 1;

Для вывода результата используется внешнее устройство с кодом 1, активируемое дешифратором внешних устройств по выходу ODEVICE. Программа для вывода на внешнее устройство представлена в приложении 1 и совмещена с программой для задания 1.

**Задание №2**

Необходимо реализовать аппаратный умножитель 4x4 и умножить с его помощью два числа.

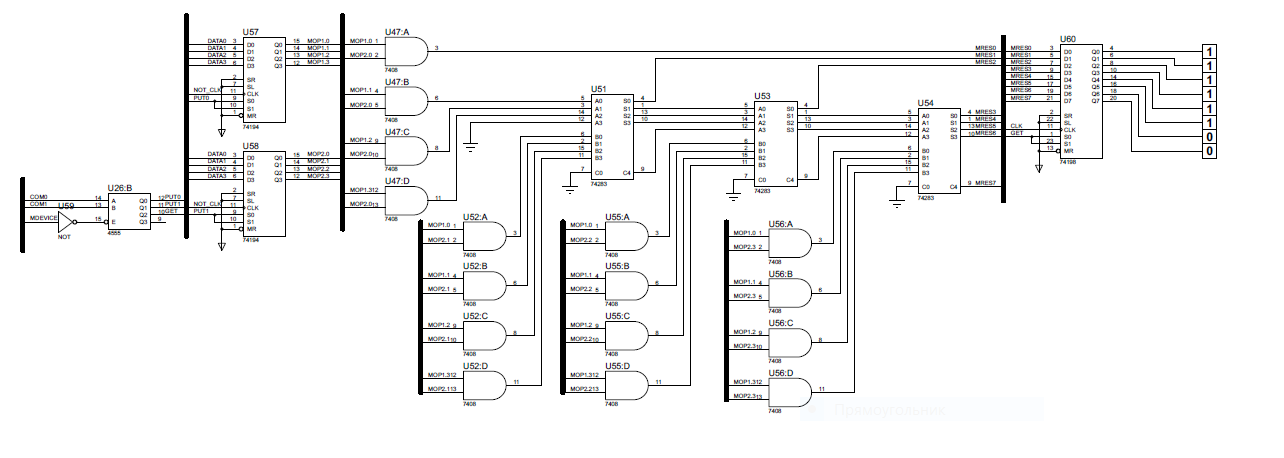


Рисунок 5. Схема аппаратного умножителя 4x4 для задания 2

**Реализация алгоритма**

Алгоритм исполнения заключается в подаче на умножиель двух чисел поочередно, дальше на умножитель отправляется команда для выдачи результата работы устройства на регистр результата умножения. Программа для выполнения задания 2 представлена в приложении 2.

Аппаратный умножитель устроен следующим образом: он имеет два регистра входных операндов, дешифратор команды, операционную часть, и регистр результата.

С помощью соответствующих команд в два регистра операндов необходимо положить два операнда. Дальше необходимо отправить на умножитель команду GET, которая положит результат умножения в регистр результата. Весь алгоритм инкапсулирован в одной команде MULT.

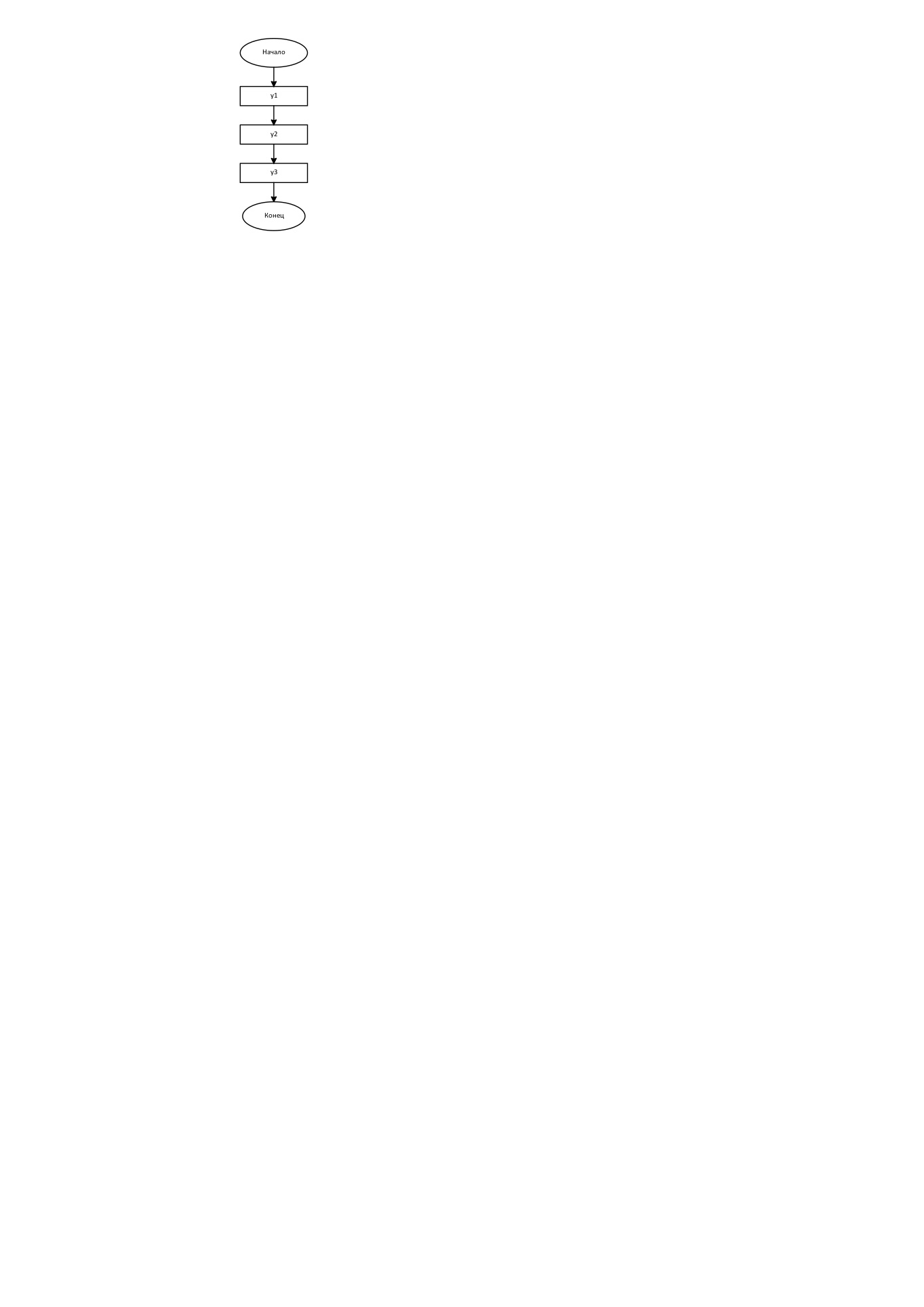


Рисунок 6. Алгоритм для задания 2

**Описание библиотеки MT1804.dll**

Данная библиотека необходима для использования элемента КМ1804 (см. рис. 9).

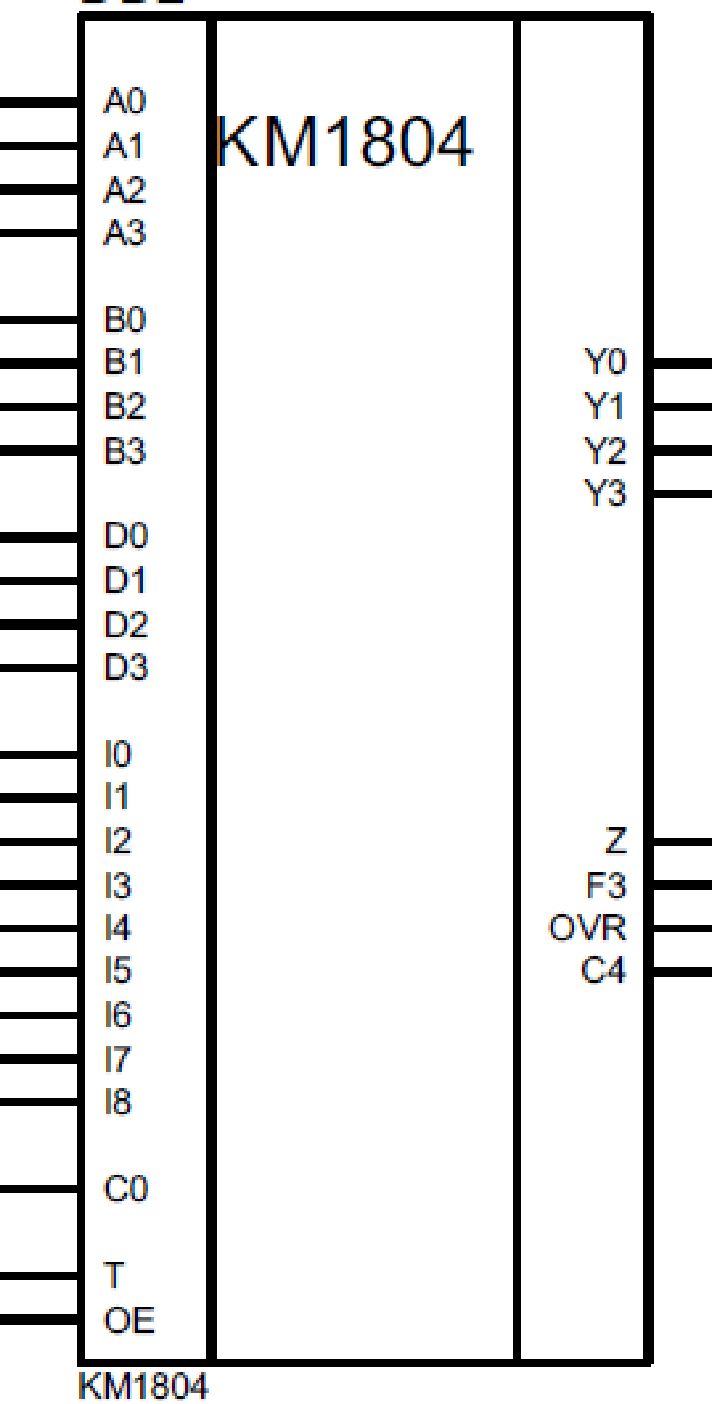


Рисунок 7. Изображение КМ1804

Операционный автомат в данной работе состоит из двух таких элементов. В качестве входных данных данный элемент принимает 22 бита данных:

Значение операнда A (4 бита).

Значение операнда B (4 бита).

Значение с внешней шины данных D (4 бита).

Значение переноса C0 (1 бит).

Код операции I8-0 (9 бит).

В качестве выходных данных элемент выдает 8 бит данных:

Результат работы КМ1804 Y (4 бита).

Флаги, полученные в результате работы КМ1804 Z, F3, OVR, C4 (4 бита).

**Приложение 1**

NOP; # Программа начинает свое исполнение

MOV 9, 15; # Помещает в РОН 15 значение 9

MOV 5, 14; # Помещает в РОН 14 значение 5

ICNT; # Инициализирует счетчик-маску значением 8 в регистре 13

CHECKE 7, 7; # Проверяет масочный бит старшей тетрады и помещает 7 в РОН 7 в случае истинного значения, уменьшает значение счетчика-маски в 2 раза

CHECKE 7, 6;

CHECKE 7, 5;

CHECKE 7, 4;

ICNT;

CHECKL 7, 4; # Проверяет масочный бит младшей тетрады и помещает 7 в РОН 4 в случае истинного значения, уменьшает значение счетчика-маски в 2 раза

CHECKL 7, 3;

CHECKL 7, 2;

CHECKL 7, 1;

NOP;

ACTDEV 1; # Активирует устройство с кодом 1 (8 регистров для вывода результата)

PREG 0, 0; # Помещает значение из РОН 0 КМ1804 в регистр 0 внешнего устройства

PREG 1, 1;

PREG 2, 2;

PREG 2, 2;

PREG 3, 3;

PREG 4, 4;

PREG 5, 5;

PREG 6, 6;

PREG 7, 7;

DISDEV; # Отключает ранее подключенное внешнее устройство

**Приложение 2**

ACTDEV 2;

MULT 5, 15; # Умножает значение 5 на значение из РОН 15 КМ1804

DISDEV;