# Формулировка задания

Написать 5 простейших программ на языке ассемблера, осуществляющих вычисление выражений в соответствии с вариантом и вывод результатов на цифровые порты ввода-вывода PORTA-PORTD. Скомпилировать программы в среде Atmel Studio, прошить программу в ПЗУ МК и проверить корректность выполненных расчётов.

В программах №№1-4 число *a* состоит из трёх байт R16:R17:R18, число *b* из двух байт R19:R20, число *с* из трёх байт R21:R22:R23. В результате вычислений необходимо обеспечить следующий вывод:

PORTA:PORTB:PORTC = expr1 (24-разрядное число)

PORTD =

Многоразрядные числа *a*, *b* и *c* должны интерпретироваться:

Программа №1 – числа без знака

Программа №2 – числа со знаком в прямом коде

Программа №3 – числа со знаком в обратном коде

Программа №4 – числа со знаком в дополнительном коде

В программе №5 число *x* состоит из трёх байт R24:R25:R26, *y* из двух байт R27:R28, *z* из трёх байт R29:R30:R31, причём числа в регистрах R24-R31 интерпретируются как двоично-десятичный код. В результате вычислений необходимо обеспечить следующий вывод:

PORTA:PORTB:PORTC = expr2 (24-разрядное число в двоично-десятичном коде)

PORTD =

Значения регистров R16-R31 выбираются самостоятельно, при этом необходимо, чтобы хотя бы в одном из чисел *a*, *b* старший разряд был равен единице (для программ 2-4), а в числах *x*, *y*, *z* не было запрещённых комбинаций

# Схема лабораторной установки

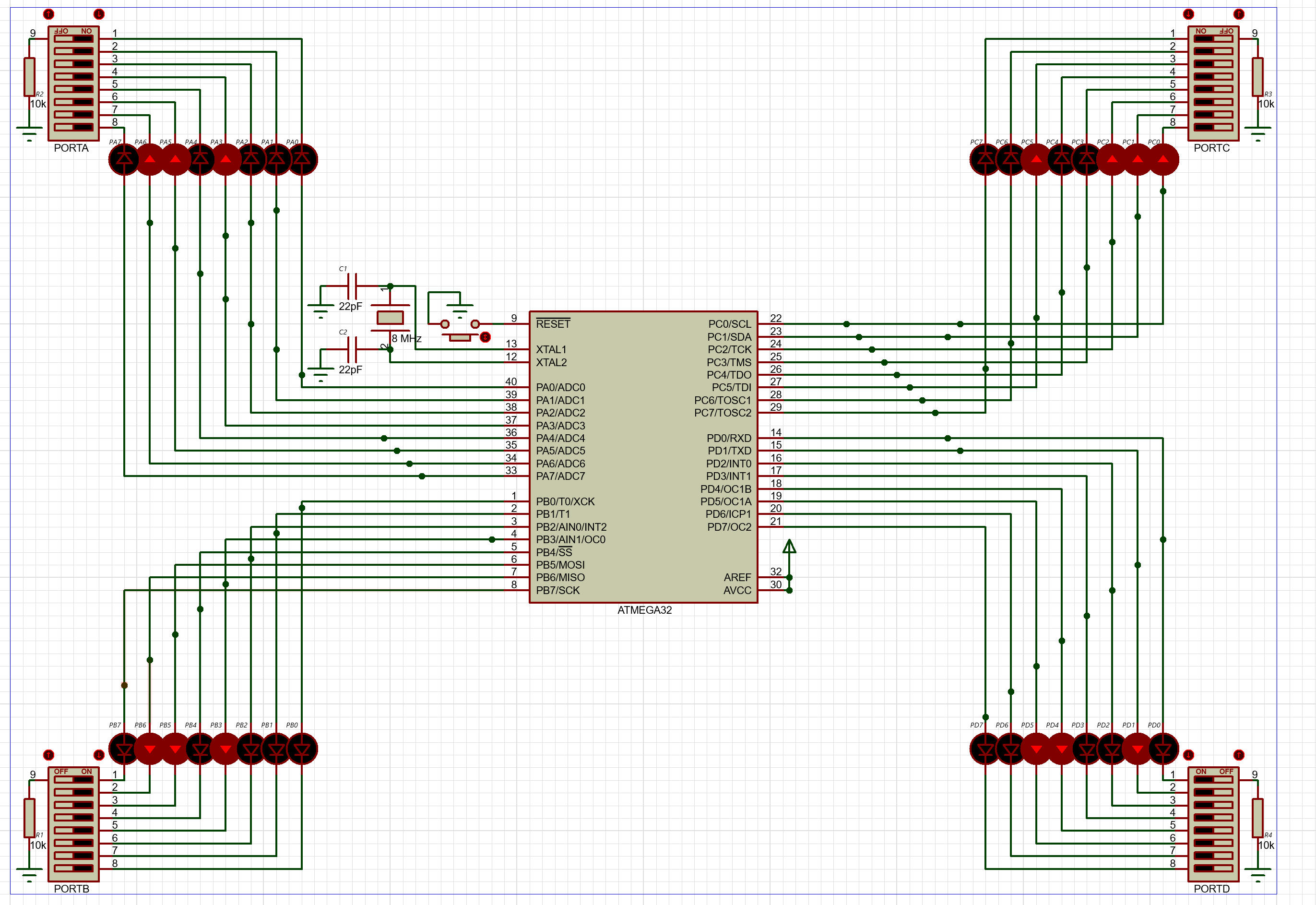


Рисунок 1 – Схема установки

# Блок-схемы алгоритмов работы программ (ключевые моменты)

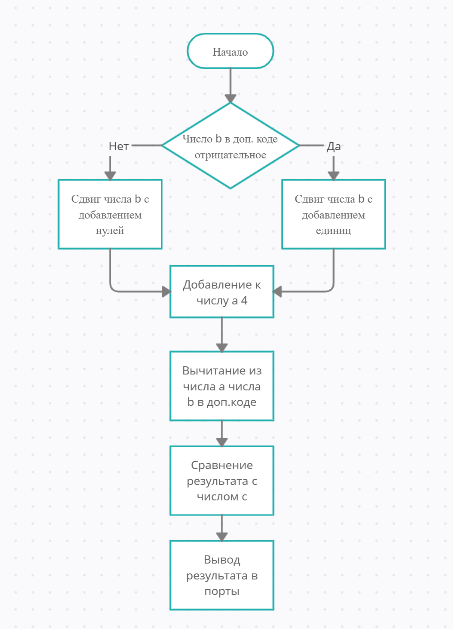


Рисунок 2 – Блок схема алгоритмов 1-4

В программах со знаковым прямым кодом и обратным кодом производится перевод в дополнительный код, после вычислений перевод обратно.



Рисунок 3 – Блок схема алгоритма 5

# Алгоритм выполнения задействованных команд (конструкций) ассемблера

SER – Установка всех битов регистра

OUT – Запись значения регистра в порт

LDI – Загрузка константы

CP – Сравнение

BRCS – Переход при установленном флаге C

BREQ – Переход при установленном флаге Z

AND – Логическое и

OR – Логическое или

ROR – Циклический сдвиг вправо через флаг переноса

LSR – Логический сдвиг вправо

RJMP – Безусловный переход

ADD – Сложение

ADC – Сложение с учётом флага переноса

SUB – Вычитание

SBC – Вычитание с учётом флага переноса

EOR – Исключающее или

# Результаты работы

Были написаны 5 программ на языке ассемблера, осуществляющих вычисление выражений в соответствии с вариантом в прямом, прямом знаковом, обратном, дополнительном и двоично-десятичном коде и вывод результатов на цифровые порты ввода-вывода PORTA-PORTD

# Ответы на контрольные вопросы

1. Какие возможности предоставляет МК Atmega32 для обработки

многоразрядных чисел?

Сложение, вычитание, сдвиги, сравнению многоразрядных чисел

2. Как можно выполнить циклический сдвиг влево 8-разрядного числа без использования флага Т?

LDI R18, 0b10000000

AND R18, R17; R17 – 8-разрядное число

LSL R18

ROR R17

3. Как можно выполнить сложение 24-разрядного числа с константой при помощи трёх машинных команд?

Число R26:R25:R24, константа R15, R14 = 0;

SUBI R24, 256 - k

SBCI R25, 256 - k

SBCI R26, 256 - k

4. Для каких целей можно использовать команду SBIW?

SBIW – Вычитание непосредственного значения (0-63) из пары регистров и размещение результата в паре регистров

Возможно использование для работы с 16-битными адресами.

5. Чем различаются арифметический и логический сдвиги вправо и какого их назначение?

При логическом сдвиге уходящий бит исчезает, а на освободившееся место записывается ноль. Эквивалентен делению на 2^n.

При арифметическом сдвиге на освободившемся месте появляется бит, соответствующий знаку, что позволяет проводить сдвиги вправо чисел, записанных в дополнительном коде. Эквивалентен делению на 2^n для положительных или беззнаковых чисел.