Хартов В.Я.

Методические указания
Для проведения лабораторных работ по курсу
Микропроцессорные системы

Лабораторная работа №2. Арифметическая обработка данных

Цель работы - изучение способов представления числовых данных в микроконтроллерах, алгоритмов арифметических операций, программирование арифметических процедур.

Теоретическая часть
1 Представление двоичных чисел в микроконтроллерах

При обработке числовой информации в микроконтроллерах обычно полагают, что целые числа имеют формат с фиксированной точкой справа $D = d_{n-1}d_n{}_{-2}\ldots d_1d_0$,
дробные числа меньше 1 имеют формат с точкой слева $D = d_{-1}d_{-2}\ldots d_{-(n-1)}d_{-n}$,
где n – число разрядов, равное 8 или 16. Обрабатываемые числа могут быть числа со знаком и числа без знака.

При целочисленном представлении разряд d_0 – младший разряд числа с весом 2^0, старший разряд d_{n-1} используется для представления знака («0» - положительный, «1» - отрицательный). Старший цифровой разряд - разряд d_{n-2} с весом 2^{n-2}. При обработке чисел без знака разряд d_{n-1} является цифровым с весом 2^{n-1}.

Дробь без знака имеет старший цифровой разряд d_{-1} с весом 2^{-1}. Для дробных чисел со знаком разряд.d_{-1} отводится под знак, старший цифровой разряд в этом случае - разряд d_{-2} с весом 2^{-1}.

Отрицательные числа, как целые так и дробные, обычно представляют в виде дополнений до основания системы счисления. Для двоичных целых чисел это будет дополнение до 2^n, для дробных – дополнение до двух.

В общем случае дополнение любого целого n – разрядного числа D до основания b системы счисления можно получить путем вычитания D из b^n. Если D находится в пределах от 1 до b^n –1, то при вычитании получается другое число в тех же пределах. Если D=0, то результат вычитания равен b^n и имеет вид 100…0 при общем числе разрядов, равном (n+1). Отбросив цифру старшего разряда, получим «0». Следовательно, в системе представления чисел дополнением до основания системы счисления существует только одно представление 0. В системе, где отрицательные числа представлены в дополнительном коде, число является положительным, если значение старшего разряда $d_{n-1} = 0$, и отрицательным, если $d_{n-1} = 1$.
Десятичный эквивалент двоичного числа, представленного дополнительным кодом, вычисляется так же, как и для числа без знака, за исключением того, что вес старшего разряда равен $-2^{(n-1)}$, а не $+2^{(n-1)}$. Представляемые числа находятся в диапазоне от $-2^{(n-1)}$ до $+2^{(n-1)}-1$.
Для дробных чисел: дробь положительна, если разряд $d_{-1} = 0$, и отрицательна, если $d_{-1} = 1$. Диапазон дробных чисел: от – 1 до $+(1 - 2^{-n})$.

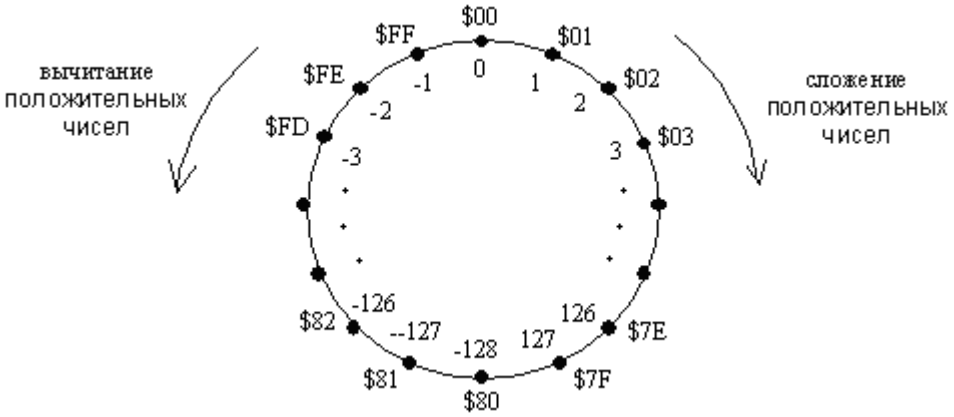
2 Сложение и вычитание чисел в дополнительном коде

Графическое представление 8–разрядных двоичных чисел со знаком в дополнительном коде приведено на рис.1,а (внутри круга указаны десятичные значения, снаружи их шестнадцатеричные изображения). Сложение с положительным числом N легко интерпретировать, перемещая указатель по часовой стрелке на N позиций; вычитание (-N), перемещая указатель против часовой стрелки, или перемещая по часовой стрелке на (256–N) позиций, что равносильно замене вычитания сложением с дополнением числа до $2^8 = 256$. Если

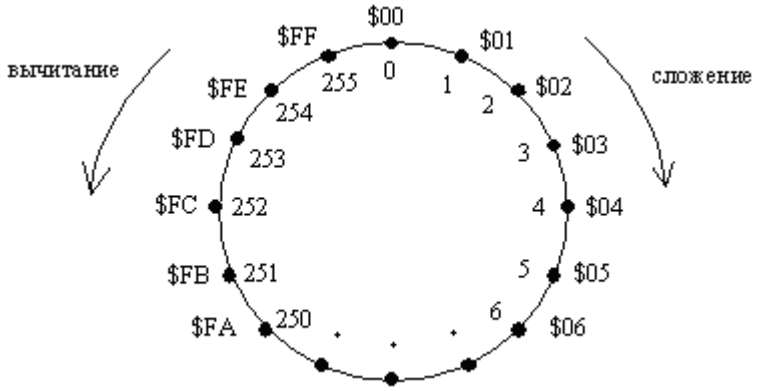
при сложении (или вычитании) получают результат, который выходит за пределы диапазона представляемых чисел от -128 до +127, то имеет место переполнение.

Правило выявления переполнения. При сложении переполнение происходит только в том случае, если слагаемые имеют одинаковые знаки, а знак суммы отличается от знака слагаемых. При вычитании переполнение происходит только в том случае, если операнды имеют разные знаки, а знак разности отличается от знака уменьшаемого. Правило переполнения можно сформулировать иначе, используя понятие переносов, возникающих при сложении/вычитании. Переполнение OVR возникает, если значения переносов в знаковый разряд P_7 и из знакового разряда P_8 различны (OVR= $P_8 \oplus P_7$). Из анализа рис.1,а следует, что переполнение возникает при сложении в случае, если указатель перейдет границу между +127 и –128.

Числа в дополнительном коде складываются и вычитаются так же, как числа без знака той же длины. Поэтому при сложении (вычитании) чисел со знаком и чисел без знака необходима одна и та же команда сложения (вычитания). Различие заключается лишь в том, что результаты операций интерпретируются по-разному в зависимости от того, какими числами оперирует пользователь: числами со знаком (то есть от –128 до +127) или без знака (от 0 до 255).



а)



б)

Рис.1. Круговые диаграммы

На рис.1,б приведено графическое представление 8–разрядных двоичных чисел без знака. Из него видно, что двоичные кодовые комбинации занимают те же позиции, что и на рис.1,а, а сложение и вычитание можно осуществить, перемещая указатель на N позиций в том или ином направлении. При сложении чисел без знака результат выходит за пределы диапазона представления при переходе границы между 255 и 0. В этом случае говорят о возникновении переноса из старшего разряда. При вычитании чисел без знака результат выходит за пределы диапазона при переходе границы между 0 и 255. В этом случае возникает заем. Но так как вычитание N можно заменить сложением с дополнительным кодом числа N, равным (256 – N), то из диаграммы видно, что заем возникает без переноса из старшего разряда. Тот же вывод

следует при выполнении операции в машинном коде. Действительно, вычитая из числа $05 число $07, имеем: $05 – $07 = $05 + $F9 = $FE = -2 Переноса нет. Заем, определяемый по отсутствию переноса при операции вычитания, есть. И наоборот, вычитая из $07 число $05, имеем: $07 – $05 = $07 + $FB = $102 = $100 (перенос) + $02 Перенос есть, что при вычитании соответствует отсутствию заема.

3 Умножение чисел без знака

Наиболее просто умножение целых чисел можно выполнить по схеме алгоритма, изображенной на рис.2. После загрузки множимого A и множителя B в регистры общего назначения и обнуления регистра произведения C производится анализ содержимого регистра множителя. Если $B \neq 0$, то к сумме частичных произведений C прибавляется множимое A.

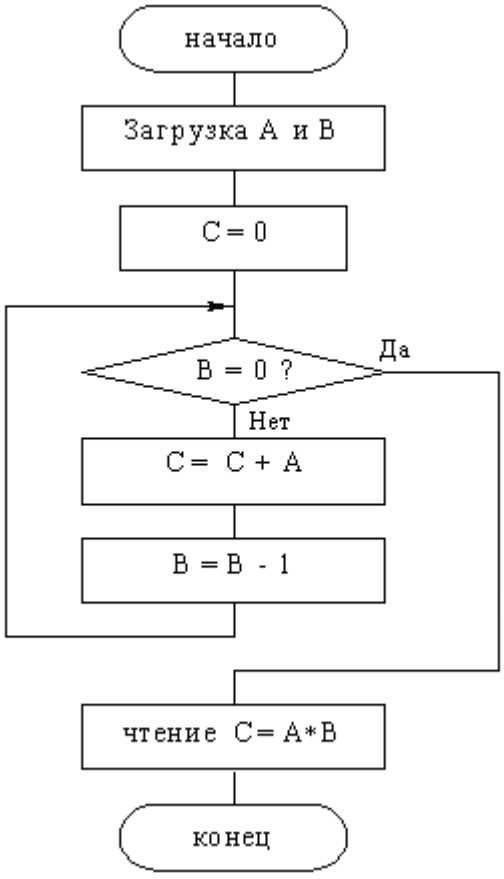


Рис.2. Схема простейшего алгоритма умножения

Затем содержимое регистра множителя уменьшается на 1 и цикл умножения повторяется до тех пор, пока содержимое регистра множителя не окажется равным 0. При умножении n – разрядных сомножителей 2n – разрядное произведение размещают в двух регистрах. Данный метод умножения находит ограниченное применение в тех приложениях, где время умножения не является критичным (при 8-разрядных сомножителях максимальная продолжительность операции умножения может составить 255 циклов сложения).

На практике больше распространены методы умножения путем сложения ряда частичных произведений $C = \sum 2^i A * b_i$, где b_i - значение разряда множителя (i=0,1,...,n-1). Один из алгоритмов умножения, начиная с младших разрядов множителя, со сдвигом вправо суммы частичных произведений приведен на рис.3.

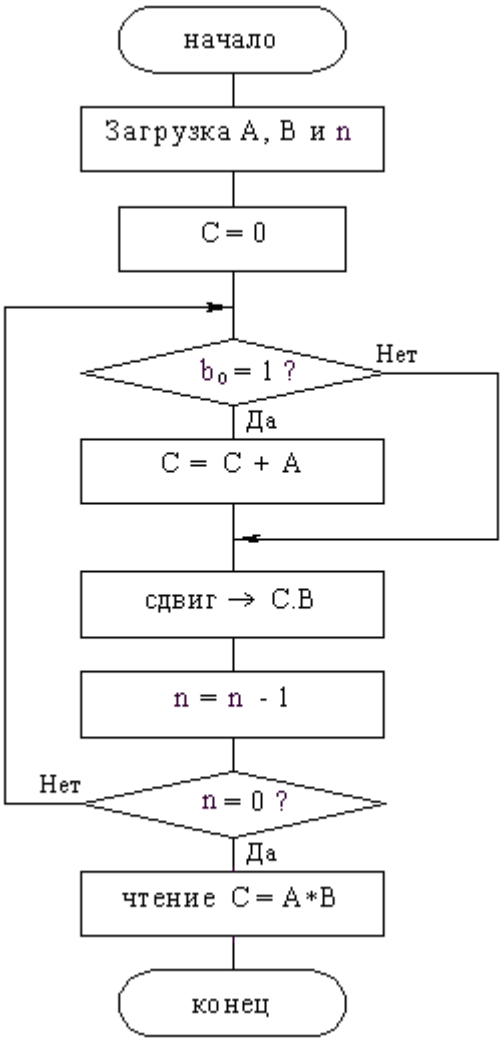


Рис.3. Схема алгоритма умножения, начиная с младших разрядов множителя

Этот алгоритм может быть использован для получения произведения двух двоичных чисел без знака. Количество итераций умножения N определяется числом разрядов множителя. Поскольку в процессе умножения на каждой итерации осуществляется сдвиг множителя B на 1 разряд вправо, на место освобождаемого разряда можно записать выталкиваемый при сдвиге вправо разряд произведения C. Таким образом, 2n – разрядное произведение можно получить, объединив содержимое n – разрядного регистра C, в котором формируется старшая часть произведения, и регистра B, в котором после выполнения умножения окажется младшая часть произведения.

4 Деление целых чисел

Для типичного алгоритма целочисленного деления делимым является двойное слово, а делителем – одинарное; частное и остаток получаются в виде одинарных слов. При выполнении деления необходимо исключить возможность деления на 0. Если для представления частного потребуется более одного слова, то имеем переполнение. Поэтому перед выполнением деления необходимо проверить условие – делитель должен быть больше старшего слова делимого.

При делении целых чисел можно использовать алгоритм деления без восстановления остатка и алгоритм с восстановлением остатка.

Схема алгоритма с восстановление остатка приведена на рис.4. Алгоритм деления представляет собой итерационную процедуру. На каждой итерации производится удвоение делимого (на первой итерации) или остатка (на всех последующих) путем сдвига влево на один разряд, вычитание делителя и определение цифры частного по знаку разности. Если разность

положительная, определяемая на данной итерации цифра частного C_i =1, если разность отрицательная – цифра частного C_i=0. Восстановление остатка выполняется путем сложения делителя с остатком после вычитания на текущей итерации деления. Деление выполняется до получения всех цифр частного.

Деление чисел со знаком можно выполнить различными способами. Если исходные операнды заданы в прямых кодах, то путем сложения по модулю 2 знаковых разрядов можно определить знак частного. Модули делимого и делителя можно разделить, используя один из вышеописанных алгоритмов. Для выяснения переполнения необходимо выполнить пробное вычитание $A – 2^{(n-1)}*B$, резервируя один разряд n-разрядного частного для знака.

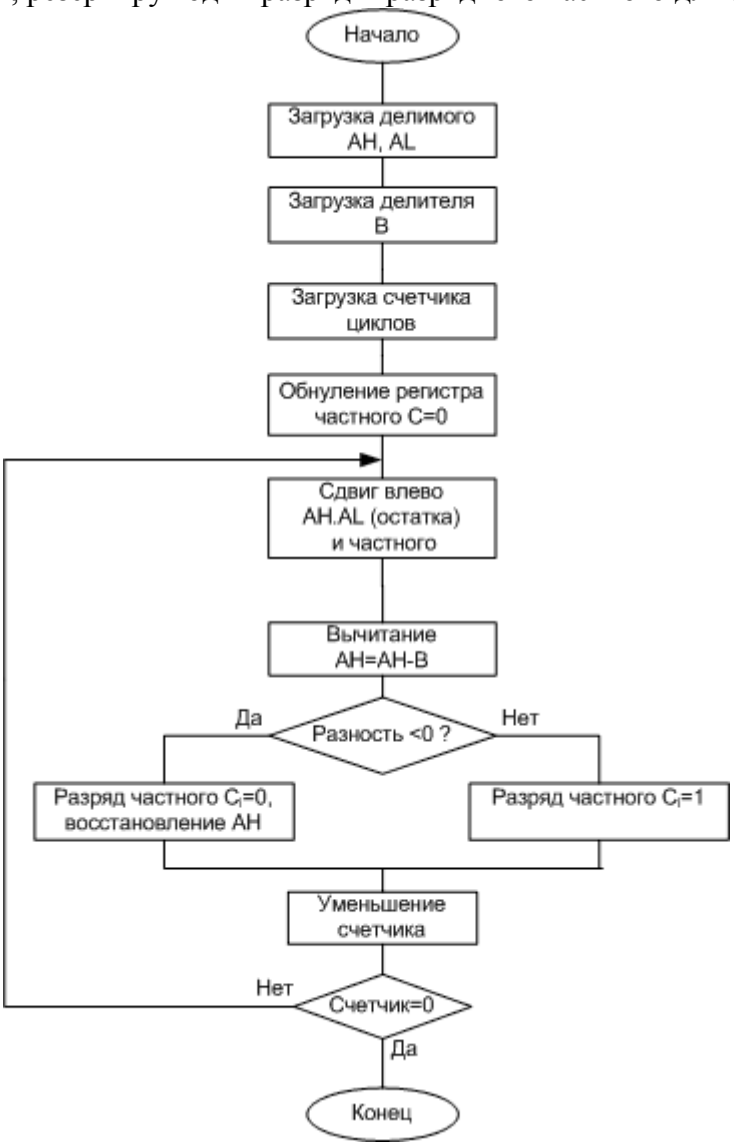


Рис.4. Схема алгоритма деления с восстановлением остатка

Ниже приведен пример деления 16 – разрядного числа A на 8–разрядное число B с восстановлением остатка.

A = 1024 = 00000100.00000000, B = 10 = 00001010, $-B = [-10]_{доп}$ = 11110110,

C = $c_7c_6c_5c_4c_3c_2c_1c_0$ – частное, x – бит, свободно определяемый при сдвиге.

+	00000100.00000000	делимое A (AH.AL)
	11110110	пробное вычитание B
	11111010	так как разность < 0, переполнения нет
+	00001000.0000000x	сдвиг A влево
	11110110	вычитание B
	11111110	1-ый остаток <0, разряд частного c_7 = 0

+	00010000.000000xx	сдвиг влево восстановленного АН
	11110110	вычитание В
	00000110	2-ой остаток >0, разряд частного c_6 =1
+	00001100.00000xxx	сдвиг остатка
	11110110	вычитание В
	00000010	3-й остаток, c_5 = 1
+	00000100.0000xxxx	сдвиг остатка
	11110110	вычитание В
	11111010	4-й остаток, c_4 = 0
+	00001000.000xxxxx	сдвиг восстановленного АН
	11110110	вычитание В
	11111110	5-й остаток, c_3 = 0
+	00010000.00xxxxxx	сдвиг восстановленного АН
	11110110	вычитание В
	00000110	6-ой остаток, c_2 = 1
+	00001100.0xxxxxxx	сдвиг остатка
	11110110	вычитание В
	00000010	7-й остаток, c_1 = 1
+	00000100.xxxxxxxx	сдвиг остатка
	11110110	вычитание В
	11111010	8-й остаток, c_0 = 0
+	11111010	
	00001010	прибавление В
	00000100	восстановлен остаток АН = 4

C = 01100110 = 102

Практическая часть
5 Программирование арифметических операций

Задание 1. Изучить программу для исследования арифметических операций в стартовом наборе STK500, приведенную ниже.

Программой предусмотрен ввод кода операции, 8- и 16- разрядных операндов, выполнение заданной операции и показ результатов.

В стартовом наборе STK500 всего восемь кнопок общего назначения (SW7…SW0). При тестировании арифметических операций эти кнопки используются следующим образом: кнопки SW0…SW2 - для ввода младшего (AL) и старшего байта (AH) первого операнда и одного байта второго операнда (BL), SW3…SW6 – для выполнения операций сложения, вычитания, умножения и деления, SW7 – для просмотра.

Программа 3.2

```
;*****************************************************************
;Программа тестирования в STK500 двоичных арифметических операций
; сложения, вычитания, умножения, деления
```

```
;Порт PD - порт управления для выбора операндов и операций
;Порт PB - порт индикации исходных операндов и результатов операции
;Соединения шлейфами: порт PB-LED, порт PD-SW
;*********************************************************************
.include "m8515def.inc"    ;файл определений для ATmega8515
;Выводы порта PD
.equ SW_op_AL = 0     ;кнопка выбора операнда op_AL
.equ SW_op_AH = 1     ;кнопка выбора операнда op_AH
.equ SW_op_BL = 2     ;кнопка выбора операнда op_BL
.equ SW_ADD = 3       ;кнопка операции сложения res=op_AL+op_BL
.equ SW_SUB = 4       ;кнопка операции вычитания res=op_AL-op_BL
.equ SW_MUL = 5       ;кнопка операции умножения shov.res=op_AL x op_BL
.equ SW_DIV = 6       ;кнопка операции деления res=op_AH.op_AL/op_BL
.equ SW_SHOW = 7      ;кнопка для просмотра признаков сложения-вычитания,
                      ;старшего байта произведения или остатка при делении

.def op_AL = r16      ;1-й операнд AL
.def op_AH = r17      ;старший байт делимого AH
.def op_BL = r18      ;2-й операнд BL
.def res = r1         ;результат операции (сумма, разность,
                      ; младший байт произведения или частное)
.def show = r31       ;регистр признаков сложения-вычитания,
                      ; старшего байта произведения или остатка при делении
.def mul_l = r21      ;младший байт произведения
.def mul_h = r22      ;старший байт произведения
.def copy_AH = r23    ;копия старшего байта делимого
.def copy_AL = r24    ;копия младшего байта делимого
.def copy_BL = r25    ;копия множителя
.def temp = r26       ;временный регистр
.def sw_reg = r27     ;регистр состояния кнопок
.def count = r28      ;число операндов в таблице операндов
.def c_bit = r29      ;счетчик циклов умножения (деления)
.def temp = r30
.macro vvod                ;ввод операнда
     lpm                   ;считывание байта из flash-памяти в r0
     mov @0,r0             ; и пересылка в регистр операнда
     mov res, r0
     adiw zl, 1            ;увеличение указателя адреса на 1
     dec count
     brne exit
     ldi ZL,low(tabl_op*2)  ;перезагрузка начала таблицы операндов
     ldi ZH,high(tabl_op*2) ; в регистр Z
     ldi count, 16
exit: nop
.endmacro

.org $000
;Инициализация
     ldi temp,low(RAMEND)       ;установка
     out SPL,temp               ; указателя стека
     ldi temp,high(RAMEND)      ; на последнюю
     out SPH,temp               ; ячейку ОЗУ
     ser temp                   ;настройка
     out DDRB,temp              ; порта PB
     out PORTB,temp             ; на вывод
     clr temp                   ;настройка
     out DDRD,temp              ; порта PD
     ser temp                   ; на
```

```
      out PORTD,temp             ; ввод
      ldi ZL,low(tabl_op*2)      ;загрузка адреса таблицы операндов
      ldi ZH,high(tabl_op*2)     ; в регистр Z
      ldi count,16               ;число операндов 16
;Опрос кнопок и выполнение заданных действий
LOOP: in sw_reg,PIND
      sbrs sw_reg,0
      rjmp f_op_AL
      sbrs sw_reg,1
      rjmp f_op_AH
      sbrs sw_reg,2
      rjmp f_op_BL
      sbrs sw_reg,3
      rjmp add_bin
      sbrs sw_reg,4
      rjmp sub_bin
      sbrs sw_reg,5
      rjmp mul_bin
      sbrs sw_reg,6
      rjmp div_bin
      sbrc sw_reg,7
      rjmp loop
      mov res,show
      rjmp outled

;Выборка 1-го операнда из таблицы операндов
f_op_AL:   vvod op_AL
           rjmp outled
;Выборка старшего байта 1-го операнда (при делении)
f_op_AH:   vvod op_AH
           rjmp outled
;Выборка 2-го операнда
f_op_BL:   vvod op_BL
           rjmp outled
;Сложение 8-разрядных операндов
add_bin:   mov res,op_AL
           add res,op_BL
           in show,SREG    ;выборка из регистра SREG
           rjmp outled
;Вычитание 8-разрядных операндов
sub_bin:   mov res,op_AL
           sub res,op_BL
           in show,SREG    ;выборка из регистра SREG
           rjmp outled
;Умножение  8-разрядных операндов
mul_bin:   clr mul_l       ;очистка младшего
      clr mul_h            ; и старшего байта произведения
      ldi c_bit,8          ;счетчик циклов
      mov copy_BL,op_BL
L1:   clc                  ;очистка флага C
      sbrc copy_BL,0       ;проверка младшего бита множителя
      add mul_h,op_AL      ;прибавление множимого AL
L2:   ror mul_h            ;сдвиг вправо
      ror mul_l            ; 2-х байтов произведения
      lsr copy_BL          ;сдвиг множителя вправо
L3:   dec c_bit            ;уменьшение счётчика циклов
      brne L1              ;если не 0, продолжаем умножение
      mov res,mul_l        ;выводимые значения - младший
```

```
      mov show,mul_h        ; и старший байты произведения
      rjmp outled
;Деление 16-разрядного числа на 8-разрядное
div_bin: sbrc op_AH,7       ;ошибки исходных данных
      rjmp error
      sbrc op_BL,7
      rjmp error
      tst op_BL             ;ошибка при делении на 0
      brsh error
      cp op_AH,op_BL        ;ошибка при переполнении
      brge error
      clr res               ;обнуляем частное
      ldi c_bit,8           ; число итераций
      mov copy_AH,op_AH
      mov copy_AL,op_AL
L4:   clc
      rol copy_AL           ;сдвиг
      rol copy_AH           ; делимого
      lsl res               ;сдвиг частного влево
      sub copy_AH,op_BL     ;вычитание делителя
      brcs recov            ;если остаток < 0,переход
      inc res               ; иначе добавить 1 в частное
      rjmp L5
recov: add copy_AH,op_BL    ;восстановление остатка
L5:   dec c_bit
      brne L4
      mov show,copy_AH      ;пересылка остатка
      rjmp outled
error: clr show
      out PORTB,show
      rcall delay
      ser show
      out PORTB,show
      rjmp wait

outled: com res
      out portb,res
      rcall delay
wait: in sw_reg,PIND        ;ждать, пока кнопка не отпущена
      com sw_reg
      brne wait
      rjmp loop

; Задержка
DELAY: ldi r21,8
m1:   ldi r19,250
m3:   ldi r20,250
m2:   dec r20
      brne m2
      dec r19
      brne m3
      dec r21
      brne m1
      ret
; Таблица операндов
tabl_op: .db 0xE5,0x10,0x1E,0xAA,0x6C,0xC7,0x1D,0xE2
         .db 0x15,0xD6,0x0E,0xB6,0x5D,0xB7,0x03,0xB2
```

Сложение/вычитание двоичных чисел

Задание 3. Выполнить ряд примеров на сложение и вычитание, выбирая операнды слагаемых AL и BL нажатием кнопок SW0 и SW2. Объяснить результаты операций при нажатиях кнопки SW3 (сложение) и SW4 (вычитание), рассматривая операнды как беззнаковые числа, затем как числа со знаком. В последнем случае вводимые отрицательные числа, содержащие единицу в старшем разряде, следует рассматривать в дополнительном коде. Нажатие кнопки SW7 показывает признаки результата операции, формируемые в регистре SREG (табл.3.1): C – перенос при сложении (заем при вычитании), Z – признак нулевого результата, N – знак результата при операциях с числами со знаком, V – переполнение разрядной сетки, S=N⊕V – знак результата вне зависимости от переполнения, H – межтетрадный перенос (заем).

Таблица 3.1. **Байт признаков результата**

№ разряда	7	6	5	4	3	2	1	0
Флаг			H	S	V	N	Z	C

Результаты наблюдений (исходные операнды, результаты операций и признаки) привести в таблице в двоичном и десятичном виде.

Таблица 3.2. **Результаты**

Число A_2/A_{10}	Число B_2/B_{10}	A+B / A-B	Признаки: HSVNZC
11000001/193 Беззнаковое	01111111/127 Беззнаковое	01000000/64 01000010/66	1 - - - 0 1 1 - - - 0 0
11000001/-63 Со знаком	01111111/+127 Со знаком	01000000/+64 01000010/+66	1 0 0 0 0 1 1 1 1 0 0 0

Умножение и деление целых чисел

Задание 4. Выполнить ряд примеров умножения 8-разрядных двоичных чисел. Нажатие кнопки SW5 показывает младший байт произведения, SW7 – старший байт.

Задание 5. Выполнить деление беззнаковых чисел, 16-разрядного делимого на 8-разрядный делитель, с восстановлением остатка при условиях, что делитель не равен 0 и его значение не вызовет переполнения, а также делимое и делитель заданы с нулевыми значениями старших разрядов. Если деление невозможно, выводится предупреждение путем зажигания и гашения всех светодиодов. Нажатие кнопки SW6 показывает частное, SW7 – остаток. Выполнить ряд примеров на деление двоичных чисел.