ПРИЛОЖЕНИЕ 1

Система команд микроконтроллеров MCS-51

Команда ACALL

По команде ACALL осуществляется вызов подпрограммы, расположенной в текущем банке памяти команд (2 Кбайта), следующим образом

- содержимое счетчика команд PC (т е полный адрес следующей команды) записывается в вершину стека (младший байт адреса записывается первым),
- содержимое указателя стека SP увеличивается на два,
- в разряды 0–10 счетчика команд записывается длинный (11-битный) адрес подпрограммы, хранящийся в коде команды Содержимое разрядов 11–15 счетчика команд не изменяется

Таким образом, точка назначения команды ACALL должна размещаться в том же банке памяти программ, что и ее первый байт

Синтаксис	Байт	Циклов	Тактов
ACALL addr11	2	12	24

Код	a10 a9 a8 1	0 0 0 1	a7 a6 a5 a4	a3 a2 a1 a0

Операция ACALL

(PC) ← (PC) + 2
(SP) = (SP) + 1
((SP)) = (PC_{7-0})
(SP) = (SP) + 1
((SP)) = (PC_{15-8})
(PC_{10-0}) = AAA A A

Команда ADD

По команде ADD выполняются следующие действия

- операнд-источник складывается с содержимым аккумулятора, результат записывается в аккумулятор,
- устанавливаются биты признаков результата (C), (AC), (OV), (P)

Синтаксис	Байт	Циклов	Тактов
ADD A, Rn	1	1	12

Операция ADD

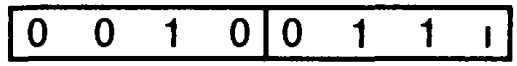
(A) ← (A) + (Rn)

Синтаксис	Байт	Циклов	Тактов
ADD A, direct	2	1	12

Операция ADD

(A) ← (A) + (direct)

Синтаксис	Байт	Циклов	Тактов
ADD A,@ Ri	1	1	12



Операция ADD

(A) ← (A) + ((Ri))

Синтаксис	Байт	Циклов	Тактов
ADD A,#data	2	1	12

Операция ADD

(A) ← (A)+ #data

Команда ADDC

По команде ADDC выполняются следующие действия

- складываются операнд-источник, бит признака переноса C и содержимое аккумулятора,
- результат записывается в аккумулятор,
- устанавливаются биты признаков результата (C),(AC),(OV),(P)

Синтаксис	Байт	Циклов	Тактов
ADDC A, Rn	1	1	12

Код:

Операция ADDC

(A) ← (A) + (C) + (Rn)

Синтаксис	Байт	Циклов	Тактов
ADDC A, direct	2	1	12

Код.

Операция ADDC

(A) ← (A) +(C) + (direct)

Синтаксис	Байт	Циклов	Тактов
ADDC A,@ Ri	1	1	12

Код.

Операция ADDC

(A) ← (A) + (C) + ((Ri))

Синтаксис	Байт	Циклов	Тактов
ADDC A,#data	2	1	12

Код

Операция ADDC

(A) ← (A)+ (C) + #data

Команда AJMP

По команде AJMP осуществляется переход в точку назначения, расположенную в текущем банке памяти программ, следующим образом

- в разряды 0-10 счетчика команд записывается длинный (11-битный) адрес точки назначения, представленный 7-15 разрядами первого байта AAA (старшие три разряда) и вторым (младшие восемь разрядов) байтом A A кода команды
- содержимое разрядов 11-15 счетчика команд соответствует адресу следующей команды

Таким образом, точка назначения команды AJMP должна размещаться в том же банке памяти программ, что и ее первый байт

Синтаксис	Байт	Циклов	Тактов
AJMP addr11	2	2	24

Операция AJMP

(PC) ← (PC) + 2

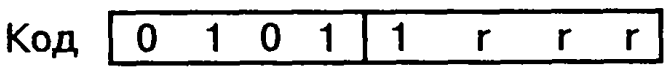
$(PC_{10\text{-}0})$←page address

Команда ANL

По команде ANL выполняются следующие действия

- операнд-источник поразрядно логически умножается на операнд-приемник,
- результат записывается на место операнда-приемника
- если в качестве операнда-приемника используется содержимое аккумулятора, то устанавливается бит четности

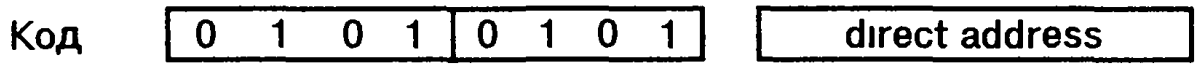
Синтаксис	Байт	Циклов	Тактов
ANL A,Rn	1	1	12



Операция ANL

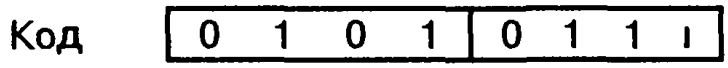
(A) ∧ ((Rn))

Синтаксис	Байт	Циклов	Тактов
ANL A,direct	2	1	12



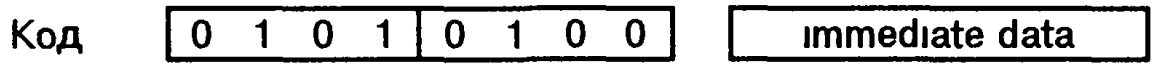
Операция ANL
(A) ← (A) ∧ (direct)

Синтаксис	Байт	Циклов	Тактов
ANL A,@ Ri	1	1	12



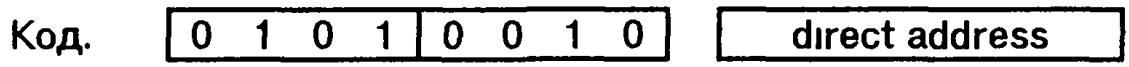
Операция ANL
(A) ← (A) ∧ ((Ri))

Синтаксис	Байт	Циклов	Тактов
ANL A,#data	2	1	12



Операция ANL
(A) ← (A) ∧ #data

Синтаксис	Байт	Циклов	Тактов
ANL direct, A	2	1	24

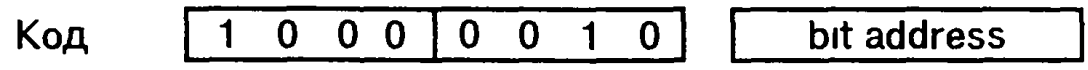


Операция ANL
(direct)←(direct) ∧ (A)

Синтаксис	Байт	Циклов	Тактов
ANL direct, #data	3	2	24

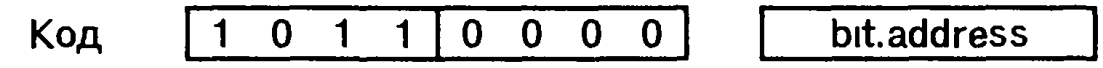
Операция ANL
(direct)←(direct)∧#data

Синтаксис	Байт	Циклов	Тактов
ANL C, bit	2	2	24



Операция ANL
(C) ← (C)∧(bit)

Синтаксис	Байт	Циклов	Тактов
ANL C,/bit	2	2	24



Операция ANL
(C) ←(C)∧/(bit)

Команда CJNE

По команде CJNE осуществляется ветвление в программе следующим образом

- если операнд-приемник не равен операнду-источнику, то управление передается в точку назначения (к текущему содержимому счетчика команд прибавляется позиционно независимое смещение, представленное третьим байтом кода команды S S),
- в противном случае управление передается следующей команде

Синтаксис	Байт	Циклов	Тактов
CJNE A, direct, rel	3	2	24

Операция (PC)← (PC)+3
IF (A) < >(direct)
THEN
(PC)← (PC)+relative offset
IF (A) < (direct) THEN
(C) ← 1,
ELSE (C) ← 0

Синтаксис	Байт	Циклов	Тактов
CJNE A, #data, rel	3	2	24

Операция (PC)← (PC)+3
IF (A) < >(data)
THEN
(PC) ← (PC)+relative offset
IF (A) < (data) THEN
(C) ← 1,
ELSE (C) ← 0

Синтаксис	Байт	Циклов	Тактов
CJNE Rn, #data, rel	3	2	24

Операция (PC)← (PC)+3
IF ((Rn)) < >data
THEN
(PC) ← (PC)+relative offset
IF (Rn) < (data) THEN
(C) ← 1,
ELSE (C) ← 0

Синтаксис	Байт	Циклов	Тактов
CJNE @Ri, #data,rel	3	2	24

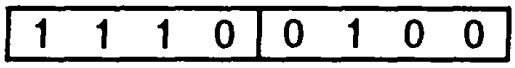
Операция (PC)← (PC)+3
IF (Ri) < >data
THEN
(PC) ← (PC)+relative offset
IF ((Ri)) < data THEN
(C) ← 1,
ELSE (C) ← 0

Команда CLR

По команде CLR операнд-приемник устанавливается в состояние «нуль» Если в качестве операнда-приемника используется содержимое аккумулятора, то устанавливается бит четности

Синтаксис	Байт	Циклов	Тактов
CLR A	1	1	12

Код



Операция CLR
(A) ← 0

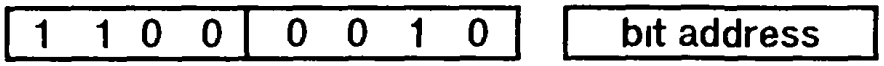
Синтаксис	Байт	Циклов	Тактов
CLR C	1	1	12

Код

Операция CLR
(C) ← 0

Синтаксис	Байт	Циклов	Тактов
CLR bit	2	1	12

Код



Операция CLR
(bit)← 0

Команда CPL

По команде CPL операнд-приемник инвертируется

Синтаксис	Байт	Циклов	Тактов
CPL A	1	1	12

Код

Операция CPL
(A) ← (A)

Синтаксис	Байт	Циклов	Тактов
CPL C	1	1	12

Код

Операция CPL
(C) ←/(C)

Синтаксис	Байт	Циклов	Тактов
CPL bit	2	1	12

Код.

1	0	1	1	0	0	1	0

Операция CPL
(bit)← /(bit)

Команда DA

По команде DA результат двоичного сложения упакованных двоично-десятичных чисел в аккумуляторе (сложение выполняется командой ADD или ADDC) преобразуется в упакованное двоично-десятичное число следующим образом

- если число в младших 4-х разрядах аккумулятора больше 9 (т е XXXX1010B XXXX1111B) или бит признака вспомогательного переноса равен 1, то к содержимому аккумулятора прибавляется число 6 При переносе из 3-го разряда аккумулятора, признак переноса устанавливается в 1,
- если число в старших 4-х разрядах аккумулятора больше 9 (т е 1010XXXXB 1111XXXXB) или бит признака переноса равен 1, то к содержимому аккумулятора прибавляется число 60H При переносе из 7-го разряда аккумулятора, признак переноса устанавливается в 1

Данная команда может быть использована для программной реализации процедуры сложения многобайтных упакованных двоично-десятичных чисел

Синтаксис	Байт	Циклов	Тактов
DA A	1	1	12

Код

1	1	0	1	0	1	0	0

Операция DA
если (A3-0) > 9 или (AC) = 1,
то если ((A3-0) + 6) > OFH,
то (C) = 1,
(A) =(A) + 6
если (A7-4) > 9 или (C) = 1,
то если ((A7-4) + 6) > OFH,

то (C) = 1,
(A) =(A) + 60H
(P) = 0\1

Команда DEC

По команде DEC операнд-приемник уменьшается на 1 Если в качестве операнда-приемника используется содержимое аккумулятора, то устанавливается бит четности (P) Из состояния 0 0B операнд-приемник переходит в состояние 1 1B

Синтаксис	Байт	Циклов	Тактов
DEC A	1	1	12

Код

0	0	0	1	0	1	0	0

Операция DEC
(A) = (A) − 1

Синтаксис	Байт	Циклов	Тактов
DEC Rn	1	1	12

Код:

0	0	0	1	1	r	r	r

Операция DEC
(Rn) ← (Rn) − 1

Синтаксис	Байт	Циклов	Тактов
DEC @Ri	1	1	12

Код.

0	0	0	1	0	1	1	i

Операция DEC
((Ri)) ← ((Ri)) − 1

Синтаксис	Байт	Циклов	Тактов
DEC direct	2	1	12

Код

0	0	0	1	0	1	0	1	direct address

Операция DEC
(direct) ← (direct) − 1

Команда DIV

По команде DIV выполняются следующие действия

- содержимое Acc (делимое) делится на содержимое регистра B (делитель),
- целая часть результата (частное) записывается в аккумулятор (Acc), остаток от деления в регистр B,
- признак переноса устанавливается в состояние «0»,
- признак переполнения устанавливается в состояние «1», если делимое в регистре B равно нулю, или в состояние «0», в противном случае,
- устанавливается признак четности

Замечание

Операнды команды рассматриваются как порядковые числа Если делитель равен нулю, то результат выполнения команды не определен

Синтаксис	Байт	Циклов	Тактов
DIV AB	1	4	48

Код

1 0 0 0	0 1 0 0

Операция DIV
если (B) = 0 то OV = 1,
иначе OV 0
(TMP) =
(B) =(A) MOD (B)
(A) = (TMP)
(C) = 0
(P) = 0\1

Команда DJNZ

По команде DJNZ осуществляется ветвление в программе следующим образом

- содержимое операнда-приемника уменьшается на единицу, затем, если операнд-приемник не равен нулю, то управление передается в точку назначения (к текущему содержимому счетчика команд прибавляется позиционно независимое смещение, представленное третьим байтом кода команды S S),

- в противном случае управление передается следующей команде

При уменьшении операнд-приемник из состояния 00H переходит в состояние 0FFH

Синтаксис	Байт	Циклов	Тактов
DJNZ Rn, rel	2	2	24

Операция DJNZ
(PC)← (PC)+2
(Rn)← (Rn) - 1,
IF Rn > 0 or Rn < 0
THEN PC← PC+rel,

Синтаксис	Байт	Циклов	Тактов
DJNZ direct,rel	3	2	24

Операция DJNZ
(PC)← (PC)+2
(direct)←(diret)-1
IF (direct) > 0 or
(direct) < 0, THEN
(PC)← (PC) + rel

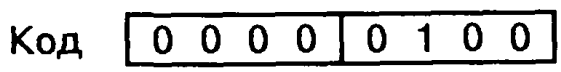
Команда INC

По команде INC операнд-приемник увеличивается на единицу

Если в качестве операнда-источника используется содержимое аккумулятора, то устанавливается бит четности (P)

Из состояния 1 1B операнд-приемник переходит в состояние 0 0B

Синтаксис	Байт	Циклов	Тактов
INC A	1	1	12



Код | 0 0 0 0 | 0 1 0 0 |

Операция INC
(A) ← (A) +1

Синтаксис	Байт	Циклов	Тактов
INC Rn	1	1	12

Код: | 0 0 0 0 | 1 r r r |

Операция INC
(Rn) ← (Rn) + 1

Синтаксис	Байт	Циклов	Тактов
INC direct	2	1	12

Код: | 0 0 0 0 | 0 1 0 1 | direct address |

Операция INC
(direct) ← (direct) + 1

Синтаксис	Байт	Циклов	Тактов
INC @Ri	1	1	12

Код | 0 0 0 0 | 0 1 1 i |

Операция INC
((Ri)) ← ((Ri)) + 1

Синтаксис	Байт	Циклов	Тактов
INC DPTR	1	2	24

Код | 1 0 1 0 | 0 0 1 1 |

Операция INC
(DPTR) ← (DPTR) + 1

Команда JB

По команде JB осуществляется ветвление в программе следующим образом

- если прямо адресуемый бит ячейки внутренней памяти данных или регистра специальной функции равен единице, то управление передается в точку назначения (к текущему содержимому счетчика команд прибавляется позиционно независимое смещение, представленное третьим байтом кода команды S S)
- в противном случае управление передается следующей команде

Прямой адрес анализируемого бита данных представляется вторым байтом кода команды I I

Синтаксис	Байт	Циклов	Тактов
JB bit,rel	3	2	24

Код	0 0 1 0	0 0 0 0	bit address	rel address

Операция JB
(PC)← (PC)+3
IF (bit)=1
THEN
(PC)← (PC) + rel,

Команда JBC

По команде JBC осуществляется ветвление в программе следующим образом

- если прямо адресуемый бит ячейки внутренней памяти данных или регистра специальной функции равен единице, то этот бит устанавливается в состояние «нуль», а управление передается в точку назначения (к текущему содержимому счетчика команд прибавляется позиционно независимое смещение, представленное третьим байтом кода команды S S)
- в противном случае управление передается следующей команде

Прямой адрес анализируемого бита данных представляется вторым байтом кода команды I I

Синтаксис	Байт	Циклов	Тактов
JBC bit,rel	3	2	24

Код	0 0 0 1	0 0 0 0	bit address	rel address

Операция JBC
(PC)← (PC)+3
IF (bit)=1
THEN (bit) ← 0
(PC)← (PC) + rel

Команда JC

По команде JC осуществляется ветвление в программе следующим образом

- если бит признака переноса C равен единице, то управление передается в точку назначения (к текущему содержимому счетчика команд прибавляется позиционно независимое смещение, представленное вторым байтом кода команды S S)
- в противном случае управление передается следующей команде

Синтаксис	Байт	Циклов	Тактов
JC rel	2	2	24

Код: | 0 1 0 0 | 0 0 0 0 | | rel.address |

Операция JC
(PC)← (PC)+2
IF (C)=1
THEN
(PC)← (PC) + rel

Команда JMP

По команде JMP осуществляется косвенный переход

Содержимое счетчика команд устанавливается равным сумме содержимого аккумулятора и содержимого регистра DPTR Сумма вычисляется по модулю 2**16, содержимое аккумулятора интерпретируется как целое число без знака в диапазоне 0 255

Синтаксис	Байт	Циклов	Тактов
JMP @A+DPTR	1	2	24

Код | 0 1 1 1 | 0 0 1 1 |

Операция JMP
(PC)← (A) + (DPTR)

Команда JNB

По команде JNB осуществляется ветвление в программе следующим образом

- если прямо адресуемый бит ячейки внутренней памяти данных или регистра специальной функции равен нулю, то управление передается в точку назначения (к текущему содержимому счетчика команд прибавляется позиционно независимое смещение, представленное третьим байтом кода команды S S)
- в противном случае управление передается следующей команде

Прямой адрес анализируемого бита данных представляется вторым байтом кода команды I I

Синтаксис	Байт	Циклов	Тактов
JNB bit,rel	3	2	24

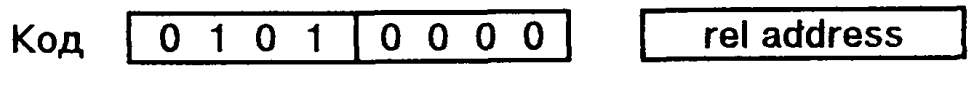
Операция JNB
(PC)← (PC)+3
IF (bit)=0
THEN
(PC)← (PC) + rel

Команда JNC

По команде JNC осуществляется ветвление в программе следующим образом

- если бит признака переноса C равен нулю, то управление передается в точку назначения (к текущему содержимому счетчика команд прибавляется позиционно независимое смещение, представленное вторым байтом кода команды S S)
- в противном случае управление передается следующей команде

Синтаксис	Байт	Циклов	Тактов
JNC rel	2	2	24



Операция JNC
(PC)← (PC)+2
IF (C)=0
THEN
(PC)← (PC) + rel

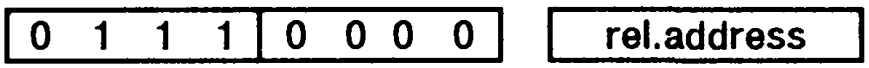
Команда JNZ

По команде JNZ осуществляется ветвление в программе следующим образом

- если содержимое аккумулятора не равно нулю, то управление передается в точку назначения (к текущему содержимому счетчика команд прибавляется позиционно независимое смещение, представленное вторым байтом кода команды S S),
- в противном случае управление передается следующей команде

Синтаксис	Байт	Циклов	Тактов
JNZ rel	2	2	24

Код:



Операция JNZ
PC)← (PC)+2
IF (A) ≠0
THEN (PC)← (PC)+rel,

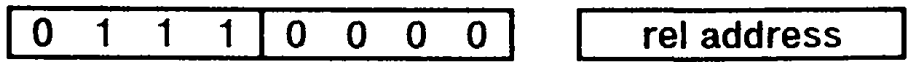
Команда JZ

По команде JZ осуществляется ветвление в программе следующим образом.

- если содержимое аккумулятора равно нулю, то управление передается в точку назначения (к текущему содержимому счетчика команд прибавляется позиционно независимое смещение, представленное вторым байтом кода команды S S),
- в противном случае управление передается следующей команде

Синтаксис	Байт	Циклов	Тактов
JZ rel	2	2	24

Код



IF (A) =0
THEN (PC)← (PC)+rel,

Команда LCALL

По команде LCALL осуществляется вызов подпрограммы, расположенной в любой точке памяти программ, следующим образом

- содержимое счетчика команд PC (т е полный адрес следующей команды) записывается в вершину стека (младший байт адреса записывается первым),
- содержимое указателя стека SP увеличивается на два,
- в счетчик команд записывается полный (16-битный) адрес подпрограммы, представленный вторым (старшим) A A и третьим (младшим) A A байтами кода команды

Синтаксис	Байт	Циклов	Тактов
LCALL addr16	3	2	24

Код:	0 0 0 1 | 0 0 1 0	addr15-addr8	addr7- addr0

Операция LCALL

(PC) ← (PC) + 3
(SP) ← (SP) +1
((SP)) ← (PC_{7-0})
(SP) ← (SP) + 1
((SP)) ← (PC_{15-8})
(PC) ← $addr_{15-0}$

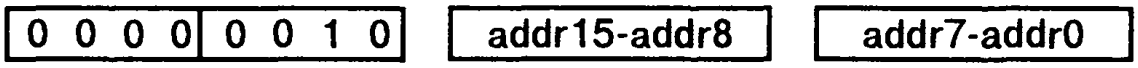
Команда LJMP

По команде LJMP осуществляется безусловный переход в точку назначения, расположенную в любом месте памяти команд, следующим образом

- в счетчик команд записывается полный (16-битный) адрес точки назначения, представленный вторым (старшим) A A и третьим (младшим) A A байтами кода команды

Синтаксис	Байт	Циклов	Тактов
LJMP addr16	3	2	24

Синтаксис	Байт	Циклов	Тактов
LJMP addr16	3	2	24

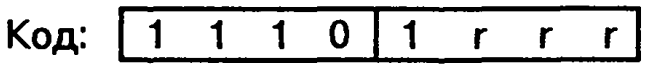


Операция LJMP
(PC) ←$addr_{15\text{-}0}$

Команда MOV

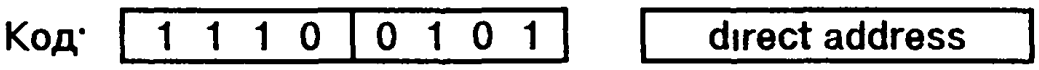
По команде MOV операнд-источник пересылается на место операнда-приемника Если местом операнда-приемника является аккумулятор, то устанавливается бит четности

Синтаксис	Байт	Циклов	Тактов
MOV A, Rn	1	1	12



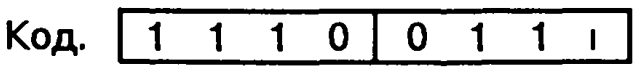
Операция MOV
(A) ← (Rn)

Синтаксис	Байт	Циклов	Тактов
MOV A, direct	2	1	12



Операция MOV
(A) ← (direct)

Синтаксис	Байт	Циклов	Тактов
MOV A, @Ri	1	1	12



Операция MOV
(A) ← ((Ri))

Синтаксис	Байт	Циклов	Тактов
MOV A, #data	2	1	12

Код 

Операция MOV
(A) ←# data

Синтаксис	Байт	Циклов.	Тактов
MOV Rn, A	1	1	12

Код 

Операция MOV
(Rn) ←(A)

Синтаксис	Байт	Циклов	Тактов
MOV Rn, direct	2	2	24

Код. 

Операция MOV
(Rn) ← (direct)

Синтаксис	Байт	Циклов	Тактов
MOV Rn,#data	2	1	12

Код 

Операция MOV
(Rn) ←# data

Синтаксис	Байт	Циклов	Тактов
MOV direct,A	2	1	12

Код 

Операция MOV
(direct) ← (A)

Синтаксис	Байт	Циклов	Тактов
MOV direct, Rn	2	2	24

Код 

Операция MOV
(direct) ←(Rn)

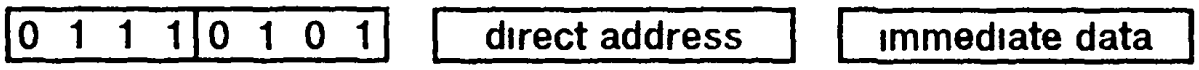
Синтаксис	Байт	Циклов	Тактов
MOV direct, direct	3	2	24

Операция MOV
(direct) ← (direct)

Синтаксис	Байт	Циклов	Тактов
MOV direct, @Ri	2	2	24

Операция MOV
(direct) ←((Ri))

Синтаксис	Байт	Циклов	Тактов
MOV direct, #data	3	2	24



Операция MOV
(direct) ←#data

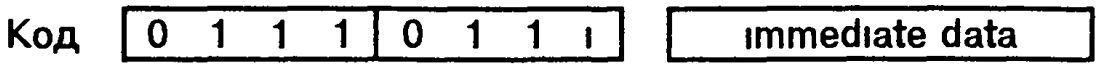
Синтаксис	Байт	Циклов	Тактов
MOV @Ri, A	1	1	12

Операция MOV
((Ri)) ←(A)

Синтаксис	Байт	Циклов	Тактов
MOV @Ri, direct	2	2	24

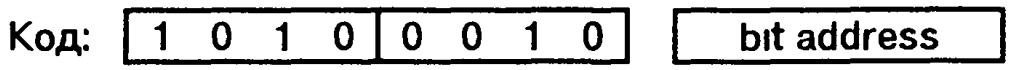
Операция MOV
((Ri)) ← (direct)

Синтаксис	Байт	Циклов	Тактов
MOV @Ri, #data	2	1	12



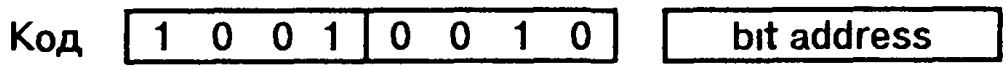
Операция MOV
((Ri)) ← (data)

Синтаксис	Байт	Циклов	Тактов
MOV C, bit	2	1	12



Операция MOV
(C) ← (bit)

Синтаксис	Байт	Циклов	Тактов
MOV bit, C	2	2	24



Операция MOV
(bit) ← (C)

Синтаксис	Байт	Циклов	Тактов
MOV DPTR,data 16	3	2	24

Операция MOV
(DPTR) ← #$data_{15-0}$
DPH ← #$data_{15-8}$
DPL ← #$data_{7-0}$

Команда MOVC

По команде MOVC операнд-источник (байт кода команды или 8-битная константа) из памяти программ пересылается в аккумулятор Имеется два способа формирования адреса операнда-источника

- как сумма содержимого аккумулятора и регистра указателя данных DPTR,
- как сумма содержимого аккумулятора и регистра счетчика команд PC

Примечания

- После выборки текущей команды счетчик команд увеличивается на число, равное количеству байт, занимаемых этой командой (для команды MOVC – на единицу)
- Старший байт адреса передается через порт #2, младший – через порт #0

Устанавливается признак четности

Синтаксис	Байт	Циклов	Тактов
MOVC A, @A+DPTR	1	2	24

Код.

1 0 0 1	0 0 1 1

Операция MOVC
(A) ← ((A)+(DPTR))

Синтаксис	Байт	Циклов	Тактов
MOVC A, @A+PC	1	2	24

Код

1 0 0 0	0 0 1 1

Операция MOVC
(PC) ←(PC)+1
(A) ←((A)+(PC))

Команда MOVX

Имеется два вида команды MOVX

По команде первого вида операнд-источник из внешней памяти данных пересылается в аккумулятор

По команде второго вида содержимое аккумулятора пересылается на место операнда-приемника во внешнюю память данных

В качестве адреса ячейки памяти данных может использоваться

1) содержимое рабочего регистра с кодом R (т е R0 R1) В этом случае адрес и данные мультиплексно передаются через порт Размер адресуемой области данных – 256 байт С целью увеличения размера адре-

суемой области данных до 64 Кбайт, рекомендуется использовать порт #2 для предварительной передачи старшего байта адреса,

2) содержимое регистра указателя данных DPTR Младший байт адреса мультиплексно передается через порт #0, старший байт - через порт #2 В фиксаторе порта #2 (т е P2) восстанавливается его предварительно сохраненное значение В буфер порта #2 записываются пересылаемые данные Размер адресуемой области данных – 64 Кбайт

Если в качестве операнда-приемника используется аккумулятор, то устанавливается признак четности

Синтаксис	Байт	Циклов	Тактов
MOVX A, @Ri	1	2	24

Код.

Операция MOVX
(A) ←((Ri))

Синтаксис	Байт	Циклов	Тактов
MOVX A, @DPTR	1	2	24

Код

Операция MOVX
(A) ←((DPTR))

Синтаксис	Байт	Циклов	Тактов
MOVX @Ri, A	1	2	24

Код.

Операция MOVX
((Ri)) ←(A)

Синтаксис	Байт	Циклов	Тактов
MOVX @DPTR, A	1	2	24

Код

Операция MOVX
(DPTR) ←(A)

Команда MUL

По команде MUL выполняются следующие действия

- содержимое аккумулятора умножается на содержимое регистра B, результат записывается в регистровую пару AB (младший байт в регистр A, старший – в регистр B),
- признак переноса устанавливается в состояние «0»,
- признак переполнения – в состояние «0», если старший байт результата равен нулю, или в состояние «1», в противном случае,
- устанавливается признак четности

Операнды команды рассматриваются как порядковые числа

Синтаксис	Байт	Циклов	Тактов
MUL AB	1	4	48

Код | 1 0 1 0 | 0 1 0 0 |

Операция MUL
$(A)_{7\text{-}0} \leftarrow (A)X(B)$
$(B)_{15\text{-}8}$,

Команда NOP

По команде NOP выполняется холостая операция микроконтроллера В результате содержимое всех регистров, кроме счетчика команд, не изменяется Счетчик команд наращивается единицу

Синтаксис	Байт	Циклов	Тактов
NOP	1	1	12

Код | 0 0 0 0 | 0 0 0 0 |

Операция NOP
(PC)← (PC) + 1

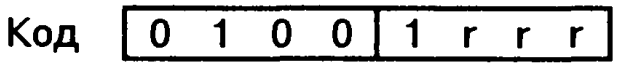
Команда ORL

По команде ORL выполняются следующие действия

- операнд-источник поразрядно логически умножает на операнд-приемник,
- результат записывается на место операнда-приемника

Если в качестве операнда-приемника используется содержимое аккумулятора, то устанавливается бит четности

Синтаксис	Байт	Циклов	Тактов
ORL A, Rn	1	1	12



Операция ORL

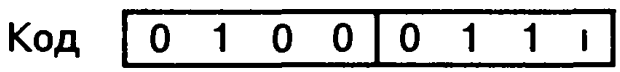
(A) ← (A)∨ (Rn)

Синтаксис	Байт	Циклов	Тактов
ORL A, direct	2	1	12

Операция ORL

(A) ← (A)∨ (direct)

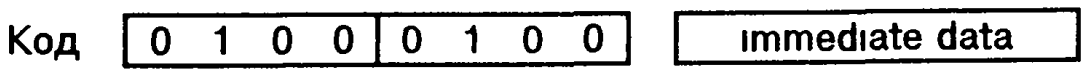
Синтаксис	Байт	Циклов	Тактов
ORL A, @Ri	1	1	12



Операция ORL

(A) ← (A)∨ ((Ri))

Синтаксис	Байт	Циклов	Тактов
ORL A, #data	2	1	12



Операция ORL

(A) ← (A)∨ #data

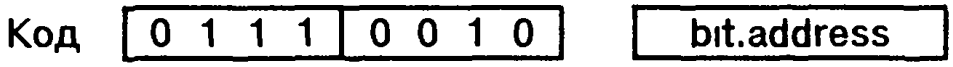
Синтаксис	Байт	Циклов	Тактов
ORL direct, A	2	1	12

Операция ORL
(direct)← (direct)∨(A)

Синтаксис	Байт	Циклов	Тактов
ORL direct,#data	3	2	24

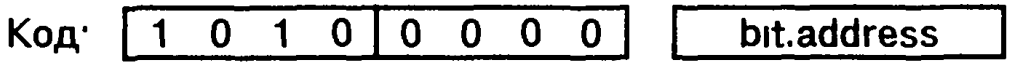
Операция ORL
(direct)←(direct)∨#data

Синтаксис	Байт	Циклов	Тактов
ORL C, bit	2	2	24



Операция ORL
(C) ← (C)∨(bit)

Синтаксис	Байт	Циклов	Тактов
ORL C,/bit	2	2	24



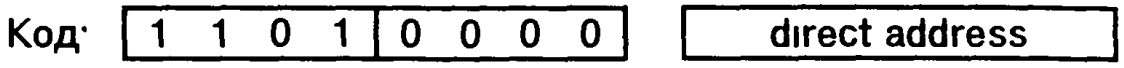
Операция ORL
(C)←(C) ∨/(bit)

Команда POP

По команде POP выполняются следующие действия

- содержимое стека по адресу, находящемуся в регистре указателя стека SP, пересылается в ячейку внутренней памяти данных или в регистр специальной функции, прямо адресуемые вторым байтом кода команды B B,
- содержимое регистра указателя стека SP уменьшается на единицу

Синтаксис	Байт	Циклов	Тактов
POP direct	2	2	24



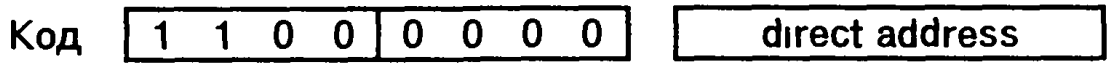
Операция POP
(direct)←((SP))
(SP) ← (SP-1)

Команда PUSH

По команде PUSH выполняются следующие действия

- содержимое регистра указателя стека SP увеличивается на единицу,
- содержимое ячейки внутренней памяти данных или содержимое регистра специальной функции, прямо адресуемое вторым байтом кода команды B B, записывается в стек по адресу, находящемуся в регистре указателя стека SP

Синтаксис	Байт	Циклов	Тактов
PUSH direct	2	2	24



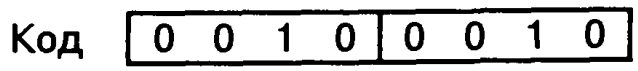
Операция PUSH
(SP) ←(SP) + 1
((SP)) ← (direct)

Команда RET

По команде RET осуществляется возврат из подпрограммы, которая была вызвана командой ACALL или LCALL, следующим образом

- содержимое вершины стека (т е полный адрес возврата из подпрограммы) пересылается в счетчик команд PC,
- содержимое указателя стека SP уменьшается на два

Синтаксис	Байт	Циклов	Тактов
RET	1	2	24



Операция RET
(PC_{15-8})← ((SP))

(SP) ← (SP) - 1
(PC_{7-0})← ((SP))
(SP) ← (SP) - 1

Команда RETI

По команде RETI осуществляется возврат из подпрограммы обслуживания прерывания следующим образом

- содержимое вершины стека (т е полный адрес возврата в точку прерывания) пересылается в счетчик команд PC Точка прерывания размещается за командой, во время выполнения которой пришел запрос на прерывание,
- содержимое указателя стека SP уменьшается на два

Разрешается прерывание равного или меньшего уровня

Примечание

Если во время выполнения команды RETI пришел запрос на прерывание равного или меньшего уровня, то его обработка начинается только после выполнения следующей команды

Синтаксис	Байт	Циклов	Тактов
RETI	1	2	24

Код. | 0 0 1 1 | 0 0 1 0 |

Операция RETI
(PC_{15-8})← ((SP))
(SP) ← (SP) - 1
(PC_{7-0})← ((SP))
(SP) ← (SP) - 1

Команда RL

По команде RL содержимое аккумулятора циклически сдвигается влево на один двоичный разряд

Синтаксис	Байт	Циклов	Тактов
RL A	1	1	12

Код· | 0 0 1 0 | 0 0 1 1 |

Операция. RL

(An+1) ← (An), n=0 6

(A0) ← (A7)

Команда RLC

По команде RLC содержимое аккумулятора и бит признака переноса циклически сдвигается влево на один двоичный разряд

Устанавливается бит признака четности

Синтаксис	Байт	Циклов	Тактов
RLC A	1	1	12

Код	0 0 1 1	0 0 1 1

Операция RLC

(An+1) ← (An), n=0 6

(A0) ← (C)

(C) ← (A7)

Команда RR

По команде RR содержимое аккумулятора циклически сдвигается вправо на один двоичный разряд

Синтаксис	Байт	Циклов	Тактов
RR A	1	1	12

Код	0 0 0 0	0 0 1 1

Операция RR

(An) ← (An+1), n=0 6

(A7) ← (A0)

Команда RRC

По команде RRC содержимое аккумулятора и бит признака переноса циклически сдвигается вправо на один двоичный разряд

Устанавливается бит признака четности

Синтаксис	Байт	Циклов	Тактов
RRC A	1	1	12

По команде RRC содержимое аккумулятора и бит признака переноса циклически сдвигаются вправо на один двоичный разряд Устанавливается бит признака четности

Синтаксис	Байт	Циклов	Тактов
RRC A	1	1	12

Код: | 0 0 0 1 | 0 0 1 1 |

Операция RRC
(An) ← (An+1), n=0 6
(A7) ← (C)
(C) ← (A7)

Команда SETB

По команде SETB операнд-приемник устанавливается в состояние «1» Если в качестве приемника используется бит PSW 0, PSW 2, PSW 6 или PSW 7, то соответствующий флаг признака результата (P), (OV), (AC) или (C) устанавливается в состояние «1»

Синтаксис.	Байт·	Циклов.	Тактов
SETB C	1	1	12

Код: | 1 1 0 1 | 0 0 1 1 |

Операция. SETB
(C) ← 1

Синтаксис	Байт	Циклов·	Тактов:
SETB bit	2	1	12

Код: | 1 1 0 1 | 0 0 1 0 | bit.address |

Операция SETB
(bit) ← 1

Команда SJMP

По команде SJMP осуществляется безусловный переход в точку назначения, расположенную в пределах от 128 байт, предшествующих команде, до 127 байт, следующих за командой, путем сложения текущего содержимого счетчика команд PC и позиционно независимого смещения, представленного вторым байтом когда команда S S

Байт позиционно независимого смещения интерпретируется как целое число со знаком в пределах от -128 до +127

Синтаксис	Байт	Циклов	Тактов
SJMP rel	2	2	24

Код	1 0 0 0	0 0 0 0	rel address

Операция SJMP
(PC) ← (PC) + 2
(PC) ← (PC) + rel

Команда SUBB

По команде SUBB выполняются следующие действия

- операнд-источник и бит признака переноса вычитаются из содержимого аккумулятора,
- результат записывается в аккумулятор,
- устанавливаются биты признаков результата (C), (AC), (OV) и (P)

Синтаксис	Байт	Циклов	Тактов
SUBB A, Rn	1	1	12

Код.	1 0 0 1	1 r r r

Операция SUBB
(A) ← (A) – (C) – (Rn)

Синтаксис	Байт	Циклов	Тактов
SUBB A,direct	2	1	12

Код	1 0 0 1	0 1 0 1	direct address

Операция SUBB
(A) ← (A) – (C) – (direct)

Синтаксис	Байт	Циклов	Тактов
SUBB A, @Ri	1	1	12

Код.	1 0 0 1	0 1 1 i

Операция SUBB
(A) ← (A) – (C)– ((Ri))

Синтаксис	Байт	Циклов	Тактов
SUBB A, #data	2	1	12

Операция. SUBB

(A) ← (A) − (C)− #data

Команда SWAP

По команде SWAP разряды 0-3 и 4-7 аккумулятора обмениваются содержимым (т е содержимое аккумулятора циклически сдвигается на четыре двоичных разряда)

Синтаксис	Байт	Циклов	Тактов
SWAP A	1	1	12

Операция SWAP

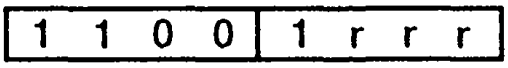
$(A_{3\text{-}0}) \leftarrow (A_{7\text{-}4})$

Команда XCH

По команде XCH в аккумулятор пересылается операнд-источник и одновременно на место операнда-источника пересылается первоначальное значение содержимого аккумулятора

Устанавливается признак четности

Синтаксис	Байт	Циклов	Тактов
XCH A, Rn	1	1	12



Операция XCH

(A) ↔ ((Rn))

Синтаксис	Байт	Циклов	Тактов
XCH A, direct	2	1	12

Операция XCH

(A) ↔ (direct)

Синтаксис	Байт	Циклов	Тактов
XCH A, @Ri	1	1	12

Код:

Операция XCH

$(A) \leftrightarrow ((R_i))$

Команда XCHD

По команде XCHD разряды 0-3 ячейки внутренней памяти данных, адрес которой находится в рабочем регистре R0 или R1, и аккумулятора обмениваются содержимым (содержимое 4-7 разрядов ячейки памяти и аккумулятора не изменяется)

Устанавливается признак четности

Синтаксис	Байт	Циклов	Тактов
XCHD A, @Ri	1	1	12

Код

Операция. XCHD

$(A_{3\text{-}0}) \leftrightarrow ((R_{i3\text{-}0}))$

Команда XRL

По команде XRL выполняются следующие действия

- операнд-источник поразрядно логически умножается на операнд приемник,
- результат записывается на место операнда-приемника

Если в качестве операнда-приемника используется содержимое аккумулятора, то устанавливается бит четности

Синтаксис	Байт	Циклов	Тактов
XRL A, Rn	1	1	12

Код

Операция XRL

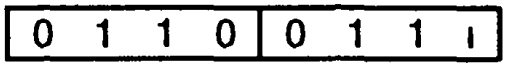
$(A) \leftarrow (A) \forall (Rn)$

Синтаксис	Байт	Циклов	Тактов
XRL A, direct	2	1	12

Код: 

Операция XRL
(A) ← (A) ∀ (direct)

Синтаксис	Байт	Циклов	Тактов
XRL A, @Ri	1	1	12

Код. 

Операция XRL
(A) ← (A) ∀ ((Ri))

Синтаксис.	Байт	Циклов	Тактов
XRL A, #data	2	1	12

Код. 

Операция XRL
(A) ← (A) ∀ #data

Синтаксис	Байт	Циклов	Тактов
XRL direct, A	2	1	12

Код: 

Операция XRL
(direct)← (direct)∀(A)

Синтаксис	Байт	Циклов	Тактов
XRL direct, #data	3	2	24

Код 

Операция XRL
(direct)←(direct)∀#data

ПРИЛОЖЕНИЕ 2

Система команд микроконтроллеров AVR

Команда ADC

Сложение двух регистров с учетом значения флага C, результат помещается в регистр Rd

Операция

Rd = Rd + Rr + C

Синтаксис	Операнды	PC
ADC Rd,Rr	0≤d≤31, 0≤r≤31	PC = PC+1

Признаки H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда ADD

Сложение двух регистров, результат помещается в регистр Rd

Операция

Rd = Rd + Rr

Синтаксис	Операнды	PC
ADC Rd,Rr	0≤d≤31,0≤r≤31	PC = PC+1

Признаки H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда ADIW

Сложение непосредственного значения (0-63) с содержимым регистровой пары, результат размещается в регистровой паре В команде может быть использована одна из четырех регистровых пар Команда удобна для операций с использованием регистра-указателя

Операция

Rdh Rdl = Rdh Rdl + K

Синтаксис	Операнды	PC
ADIW Rdl,K	dl ∈ [24,26,28,30], 0≤K≤63	PC = PC+1

Признаки S,V,N,Z,C
Слов 1 (2 байта)
Циклов 2

Команда AND

Выполняется операция AND с содержимым регистров Rd и Rr, результат размещается в регистр Rd

Операция

Rd = Rd * Rr

Синтаксис	Операнды	PC
AND Rd,Rr	0≤d≤31,0≤r≤31	PC = PC+1

Признаки V(0),S,N,Z
Слов 1 (2 байта)
Циклов 1

Команда ANDI

Выполняется операция AND между содержимым регистра Rd и константой, результат размещается в регистре-приемнике Rd

Операция

Rd = Rd * K

Синтаксис	Операнды	PC
ANDI Rd,K	16≤d≤31,0≤K≤255	PC = PC+1

Признаки V(0),S,N,Z
Слов 1 (2 байта)
Циклов 1

Команда ASR

Сдвиг всех битов регистра Rd на один разряд вправо Бит 7 не изменяется Бит 0 загружается во флаг C регистра SREG

Операция

$Rd(i) = Rd(i+1)$ $0 \le i < 7$
$Rd(7) = Rd(7)$
$C = Rd(0)$

Синтаксис	Операнды	PC
ASR Rd	0≤d≤31	PC = PC+1

Признаки S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда BCLR

Очистка отдельного флага регистра SREG

Операция

SREG(s) = 0

Синтаксис	Операнды	PC
BCLR s	0≤s≤7	PC = PC+1

Признаки I,T,H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда BLD

Копирует флаг T регистра SREG в бит b регистра Rd

Операция

Rd(b) = T

Синтаксис	Операнды	PC
BLD Rd,b	0≤d≤31, 0≤b≤7	PC = PC+1

Слов 1 (2 байта)
Циклов 1

Команда BRBC

Условный переход Если бит регистра SREG очищен, то выполняется переход относительно PC

Операция

(1) **if** SREG(s) = 0 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRBC s,k	0≤s≤7, -64≤k≤63	PC = PC + k + 1 PC = PC + 1, если SREG(s) ≠ 0

Слов 1 (2 байта)

Циклов 1, если условие = false
2, если условие = true

Команда BRBS

Условный переход Если бит регистра SREG установлен, то выполняется переход относительно PC

Операция

(i) **if** SREG(s) = 1 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRBS s,k	0≤s≤7, -64≤b≤63	PC = PC + k + 1 PC = PC + 1, если SREG(s) ≠ 1

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRCC

Переход, если флаг переноса C очищен

Операция

(1) **if** C = 0 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRCC k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если C ≠ 0

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRCS

Переход, если флаг переноса C установлен

Операция

(1) **if** C = 1 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRCS k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если C ≠ 1

Слов 1 (2 байта)

Циклов 1, если условие = false
2, если условие = true

Команда BREQ

Условный переход, если установлен флаг Z Если команда выполняется непосредственно после какой-либо из команд CP, CPL, SUB или SUBI, условный переход будет только в том случае, если содержимое Rd равно содержимому Rr (Эквивалентна команде BRBS 1,k)

Операция

if Rd = Rr (Z = 1) **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BREQ k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRGE

Условный переход, если сброшен флаг знака S Если команда выполняется непосредственно после какой-либо из команд CP, CPL, SUB или SUBI, условный переход будет только в том случае, если число со знаком в Rd, больше или равно числа со знаком в Rr (Эквивалентна команде BRBC 4,k)

Операция

if Rd ≥ Rr (N **xor** V = 0) **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRGE k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRHC

Условный переход, если сброшен флаг полупереноса H (Эквивалентна команде BRBC 5,k)

Операция

if H = 0 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRHC k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRHS

Условный переход, если установлен флаг полупереноса H (Эквивалентна команде BRBS 5,k)

Операция

if H = 0 **then** PC = PC + k + 1,
else PC .= PC + 1

Синтаксис	Операнды	PC
BRHS k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRID

Условный переход, если сброшен флаг общего прерывания I (Эквивалентна команде BRBC 7,k)

Операция

if I = 0 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRID k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRIE

Условный переход, если установлен флаг общего прерывания I (Эквивалентна команде BRBS 7,k)

Операция

if I = 1 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRIE k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)

Циклов 1, если условие = false
2, если условие = true

Команда BRLO

Условный переход, если установлен флаг переноса C Если команда выполняется непосредственно после какой-либо из команд CP, CPL, SUB или SUBI, условный переход будет только в том случае, если число без знака в Rd меньше числа без знака в Rr (Эквивалентна команде BRBS 0,k)

Операция

if Rd < Rr (C = 1) **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRLO k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)

Циклов 1, если условие = false
2, если условие = true

Команда BRLT

Условный переход, если установлен флаг знака S Если команда выполняется непосредственно после какой-либо из команд CP, CPL, SUB или SUBI, условный переход будет только в том случае, если число со знаком в Rd меньше числа со знаком в Rr (Эквивалентна команде BRBS 4,k)

Операция

if Rd < Rr (N **xor** V = 1) **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRLT k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRMI

Условный переход, если установлен флаг отрицательного результата N (Эквивалентна команде BRBS 2,k)

Операция

if N = 1 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRMI k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRNE

Условный переход, если сброшен флаг нуля Z Если команда выполняется непосредственно после какой-либо из команд CP, CPL, SUB или SUBI, условный переход будет только в том случае, если содержимое в Rd неравно содержимому в Rr (Эквивалентна команде BRBC 1,k)

Операция

if Rd ≠ Rr (Z = 0) **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRNE k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRPL

Условный переход, если сброшен флаг отрицательного результата N (Эквивалентна команде BRBC 2,k)

Операция

if N = 0 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRPL k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRSH

Условный переход, если сброшен флаг переноса C Если команда выполняется непосредственно после какой-либо из команд CP, CPL, SUB или SUBI, условный переход будет только в том случае, если число без знака в Rd больше или равно числу без знака в Rr (Эквивалентна команде BRBC 0,k)

Операция

if Rd ≥ Rr (C = 0) **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRSH k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRTC

Условный переход, если сброшен флаг передачи T (Эквивалентна команде BRBC 6,k)

Операция

if T = 0 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRTC k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)
Циклов 1, если условие = false
2, если условие = true

Команда BRTS

Условный переход, если установлен флаг передачи T (Эквивалентна команде BRBS 6,k)

Операция

if T = 1 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRTS k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)

Циклов 1, если условие = false
2, если условие = true

Команда BRVC

Условный переход, если сброшен флаг переполнения V (Эквивалентна команде BRBC 3,k)

Операция

if V = 0 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRVC k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)

Циклов 1, если условие = false
2, если условие = true

Команда BRVS

Условный переход, если установлен флаг переполнения V (Эквивалентна команде BRBS 3,k)

Операция

if V = 1 **then** PC = PC + k + 1,
else PC = PC + 1

Синтаксис	Операнды	PC
BRVS k	-64≤k≤63	PC = PC + k + 1 PC = PC + 1, если условие = false

Слов 1 (2 байта)

Циклов 1, если условие = false
2, если условие = true

Команда BSET

Установка флага в регистре SREG

Операция

SREG(s) = 1

Синтаксис	Операнды	PC
BSET s	0≤s≤7	PC = PC + 1

Признаки I,T,H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда BST

Копирует бит b регистра Rd во флаг T регистра SREG

Операция

T = Rd(b)

Синтаксис	Операнды	PC
BST Rd,b	0≤d≤31,0≤b≤7	PC = PC + 1

Признаки T
Слов 1 (2 байта)
Циклов 1

Команда CALL

Вызов подпрограммы, расположенной по любому адресу пространства памяти программ Адрес возврата (команды, следующей за CALL) запоминается в стеке (см также RCALL)

Операция

PC = k

Синтаксис	Операнды	PC	Стек
CALL k	0≤k≤64K	PC = k	STACK = PC + 2 SP = SP - 2

Слов 2 (4 байта)
Циклов 4

Команда CBI

Очищает определенный бит в регистре ввода/вывода Команда определена для младших 32 регистров (0 31)

Операция

I/O(P,b) = 0

Синтаксис	Операнды	PC
CBI P,b	0≤P≤31, 0≤b≤7	PC = PC + 1

Слов 1 (2 байта)
Циклов 2

Команда CLC
Очищает флаг переноса C в регистре SREG
Операция
C = 0

Синтаксис	Операнды	PC
CLC		PC = PC + 1

Признаки C(0)
Слов 1 (2 байта)
Циклов 1

Команда CLH
Очищает флаг полупереноса H в регистре SREG
Операция
H = 0

Синтаксис	Операнды	PC
CLH		PC = PC + 1

Признаки H(0)
Слов 1 (2 байта)
Циклов 1

Команда CLI
Очищает флаг прерываний I в регистре SREG
Операция
I = 0

Синтаксис	Операнды	PC
CLI		PC = PC + 1

Признаки I(0)
Слов 1 (2 байта)
Циклов 1

Команда CLN

Очищает флаг отрицательного результата N в регистре SREG

Операция

N = 0

Синтаксис	Операнды	PC
CLN		PC = PC + 1

Признаки N(0)
Слов 1 (2 байта)
Циклов 1

Команда CLR

Очищает регистр Команда выполняет операцию ИСКЛЮЧАЮЩЕЕ ИЛИ над регистром и этим же регистром Все биты регистра сбрасываются

Операция

Rd = Rd **xor** Rd

Синтаксис	Операнды	PC
CLR Rd	0≤d≤31	PC = PC + 1

Признаки S(0),V(0),N(0),Z(1)
Слов 1 (2 байта)
Циклов 1

Команда CLS

Очищает флаг знака S в регистре SREG

Операция

S = 0

Синтаксис	Операнды	PC
CLS		PC = PC + 1

Признаки S(0)
Слов 1 (2 байта)
Циклов 1

Команда CLT

Очищает флаг T в регистре SREG

Операция

T = 0

Синтаксис	Операнды	PC
CLT		PC = PC + 1

Признаки T(0)
Слов 1 (2 байта)
Циклов 1

Команда CLV
Очищает флаг переполнения V в регистре SREG
Операция

V = 0

Синтаксис	Операнды	PC
CLV		PC = PC + 1

Признаки V(0)
Слов 1 (2 байта)
Циклов 1

Команда CLZ
Очищает флаг нулевого результата Z в регистре SREG
Операция

Z = 0

Синтаксис	Операнды	PC
CLZ		PC = PC + 1

Признаки Z(0)
Слов 1 (2 байта)
Циклов 1

Команда COM
Выполняет операцию дополнения до единицы регистра Rd
Операция

Rd = $FF - Rd

Синтаксис	Операнды	PC
COM Rd	0≤d≤31	PC = PC + 1

Признаки S,V(0),N,Z,C(1)
Слов 1 (2 байта)
Циклов 1

Команда CP

Сравнение регистров Rd и Rr содержимое регистров не меняется

Операция

Rd - Rr

Синтаксис	Операнды	PC
CP Rd,Rr	0≤d≤31,0≤r≤31	PC = PC + 1

Признаки H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда CPC

Сравнение регистров Rd и Rr с учетом предыдущего переноса Содержимое регистров не меняется

Операция

Rd - Rr - C

Синтаксис	Операнды	PC
CPC Rd,Rr	0≤d≤31, 0≤r≤31	PC = PC + 1

Признаки H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда CPI

Сравнение регистра Rd с константой Содержимое регистра не меняется

Операция

Rd - K

Синтаксис	Операнды	PC
CPI Rd,K	16≤d≤31, 0≤K≤255	PC = PC + 1

Признаки H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда CPSE

Сравнение регистров Rd и Rr и пропуск следующей команды, если Rd = Rr

Операция

if Rd = Rr **then** PC = PC + 2 (или 3)
else PC = PC + 1

Синтаксис	Операнды	PC
CPSE Rd,Rr	0≤d≤31,0≤r≤31	PC = PC + 1, условие false PC = PC + 2, пропуск однословной команды PC = PC + 3, пропуск двухсловной команды

Слов 1 (2 байта)
Циклов 1

Команда DEC

Вычитает 1 из содержимого регистра Rd и размещает результат в регистре Rd Флаг C не изменяется

Операция

Rd = Rd - 1

Синтаксис	Операнды	PC
DEC Rd	0≤d≤31	PC = PC + 1

Признаки S,V,N,Z
Слов 1 (2 байта)
Циклов 1

Команда EOR

Выполняет операцию ИСКЛЮЧАЮЩЕЕ ИЛИ над содержимым регистров Rd и Rr Результат размещается в регистре Rd

Операция

Rd = Rd **xor** Rr

Синтаксис	Операнды	PC
EOR Rd,Rr	0≤d≤31, 0≤r≤31	PC = PC + 1

Признаки S,V(0),N,Z
Слов 1 (2 байта)
Циклов 1

Команда ICALL

Косвенный вызов подпрограммы через указатель в регистре Z Регистр-указатель Z является 16-битным и позволяет адресовать 64 Кслов (128 Кбайт) памяти программ

Операция

PC(15-0) = Z(15-0)

Синтаксис	Операнды	PC
ICALL		см Операция

Стек STACK = PC + 1
SP = SP - 2
Слов 1 (2 байта)
Циклов 3

Команда IJMP

Косвенный переход по адресу в регистре-указателе Z Регистр-указатель Z является 16-битным и позволяет адресовать 64 Кслов (128 Кбайт) памяти программ

Операция

PC = Z(15-0)

Синтаксис	Операнды	PC	Стек
IJMP		см Операция	Не изменяется

Слов 1 (2 байта)
Циклов 2

Команда IN

Загружает данные из пространства ввода/вывода (порты, таймеры, регистры конфигурации) в регистр Rd регистрового файла

Операция

Rd = P

Синтаксис	Операнды	PC
IN Rd,P	0≤d≤31, 0≤P≤63	PC = PC + 1

Слов 1 (2 байта)
Циклов 1

Команда INC

Добавляет 1 к содержимому регистра Rd и размещает результат в регистре Rd Флаг C не изменяется

Операция

Rd = Rd + 1

Синтаксис	Операнды	PC
INC Rd	0≤d≤31	PC = PC + 1

Признаки S,V,N,Z
Слов 1 (2 байта)
Циклов 1

Команда JMP

Переход по адресу в пространстве 4 Мбайт памяти программ
Операция

PC = k

Синтаксис	Операнды	PC	Стек
JMP k	0≤k≤4M	PC = k	не изменяется

Слов 2 (4 байта)
Циклов 3

Команда LD

Загружает байт из ячейки SRAM в регистр Адрес SRAM содержится в одном из 16-битных регистров-указателей X, Y или Z регистрового файла Доступ к памяти осуществляется внутри текущей страницы SRAM (64 Кбайт) Для доступа к другой странице SRAM необходимо изменить номер страницы в регистре RAMPX, RAMPY или RAMPZ (в зависимости от выбранного регистра-указателя)

Операция	Комментарий
Rd = (R)	R - X/Y/Z Не изменяется
(ii)Rd = (R), R = R+1	R - X/Y/Z
	После инкрементирования
(iii)R = R-1, Rd = (R)	R - X/Y/Z
	До декрементирования

Синтаксис	Операнды	PC	Обозначения
LD Rd,R	0≤d≤31	PC = PC + 1,	R - X/Y/Z
(ii)LD Rd,R+	0≤d≤31	PC = PC + 1,	R - X/Y/Z
(iii)LD Rd,-R	0≤d≤31	PC = PC + 1,	R - X/Y/Z

Слов 1 (2 байта)
Циклов 2

Команда LDD

Загружает байт из ячейки SRAM в регистр Адрес SRAM находится в одном из 16-битных регистров-указателей Y или Z регистрового файла Доступ к памяти осуществляется внутри текущей страницы SRAM (64 Кбайт) Для доступа к другой странице SRAM необходимо изменить номер страницы в регистре RAMPY или RAMPZ (в зависимости от выбранного регистра-указателя)

Операция	Комментарий
Rd = (R+q)	R - Y/Z Не изменяется, q - Смещение

Синтаксис	Операнды	PC	Обозначения
LDD Rd,R+q	0≤d≤31, 0≤q≤63	PC = PC + 1	R - Y/Z

Слов 1 (2 байта)
Циклов 2

Команда LDI

Загружает 8-битную константу в один из регистров R16-R31

Операция

Rd = K

Синтаксис	Операнды	PC
LDI Rd,K	16≤d≤31, 0≤K≤255	PC = PC + 1

Слов 1 (2 байта)
Циклов 1

Команда LDS

Загружает байт из ячейки SRAM в регистр 16-битный адрес задается непосредственно в команде Доступ к памяти ограничен текущей страницей (64 Кбайта) Для доступа к памяти выше 64 Кбайт используется регистр RAMPZ

Операция

Rd = (k)

Синтаксис	Операнды	PC
LDS Rd,k	0≤d≤31, 0≤K≤65535	PC = PC + 2

Слов 2 (4 байта)
Циклов 3

Команда LPM

Загружает байт, адресуемый регистром Z, в регистр 0 (R0) Команда обеспечивает доступ к любому байту памяти программ, организованной как 16-битные слова Команда адресует первые 64 Кбайта (32 Кслов) памяти программ Старший бит регистра Z определяет, осуществляется ли доступ к младшему байту слова (0) или старшему (1)

Операция	Комментарий	
R0 = (Z)	Z указывает адрес памяти программ	
Синтаксис	Операнды	PC
LPM		PC = PC + 1

Слов 1 (2 байта)
Циклов 3

Команда LSL

Сдвиг всех битов в регистре Rd на один разряд влево Бит 0 обнуляется, бит 7 загружается во флаг C регистра SREG Команда эффективна для умножения беззнаковых значений на 2

Операция

Rd(i+1) = Rd(i), i = 0 6
Rd(0) = 0
C = Rd(7)

Синтаксис	Операнды	PC
LSL Rd	0≤d≤31	PC = PC + 1

Признаки H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда LSR

Сдвиг всех битов в регистре Rd на один разряд вправо Бит 7 обнуляется, бит 0 загружается во флаг C регистра SREG Команда эффективна для деления беззнаковых значений на 2

Операция

Rd(I - 1) = Rd(i), i = 1 7
Rd(7) = 0
C = Rd(0)

Синтаксис	Операнды	PC
LSR Rd	0≤d≤31	PC = PC + 1

Признаки S,V,N(0),Z,C
Слов 1 (2 байта)
Циклов 1

Команда MOV

Содержимое регистра Rr копируется в регистр Rd Содержимое регистра Rr не изменяется

Операция

Rd = Rr

Синтаксис	Операнды	PC
MOV Rd,Rr	0≤d≤31, 0≤r≤31	PC = PC + 1

Слов 1 (2 байта)
Циклов 1

Команда MUL

Перемножаются два 8-битных числа, множимое и множитель находятся в двух 8-битных регистрах 16-битный результат размещается в регистры R1 (старший байт) и R0 (младший байт) Если множимое и множитель выбираются из R0 или R1, их содержимое изменится после операции умножения

Операция

R1 R0 = Rr * Rd

Синтаксис	Операнды	PC
MUL Rd,Rr	0≤d≤31, 0≤r≤31	PC = PC + 1

Слов 1 (2 байта)
Циклов 2

Команда NEG

Заменяет содержимое регистра Rd его двоичным дополнением

Операция

Rd = $00 - Rd

Синтаксис	Операнды	PC
NEG Rd	0≤d≤31	PC = PC + 1

Признаки H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда NOP

Команда не выполняет операции, программный счетчик инкрементируется

Синтаксис	Операнды	PC
NOP		PC = PC + 1

Слов 1 (2 байта)
Циклов 1

Команда OR

Выполняется логическая операция ИЛИ над содержимым регистров Rd и Rr, результат размещается в регистре Rd

Операция

Rd = Rd **or** Rr

Синтаксис	Операнды	PC
OR Rd,Rr	0≤d≤31, 0≤r≤31	PC = PC + 1

Признаки S,V(0),N,Z
Слов 1 (2 байта)
Циклов 1

Команда ORI

Выполняется логическая операция ИЛИ над содержимым регистра Rd и константой, результат размещается в регистре Rd

Операция

Rd = Rd **or** K

Синтаксис	Операнды	PC
ORI Rd,K	16≤d≤31, 0≤K≤255	PC = PC + 1

Признаки S,V(0),N,Z
Слов 1 (2 байта)
Циклов 1

Команда OUT

Сохраняет данные регистра Rr в регистре пространства ввода/вывода (порты, таймеры, регистры конфигурации и др)

Операция

P = Rr

Синтаксис	Операнды	PC
OUT P,Rr	0≤r≤31, 0≤P≤63	PC = PC + 1

Слов 1 (2 байта)
Циклов 1

Команда POP

Загружает байт данных из стека в регистр Rd

Операция

Rd = STACK

Синтаксис	Операнды	PC	Стек
POP Rd	0≤d≤31	PC = PC + 1	SP = SP+1

Слов 1 (2 байта)
Циклов 2

Команда PUSH

Сохраняет содержимое регистра Rr в стеке
Операция

STACK = Rr

Синтаксис	Операнды	PC	Стек
PUSH Rr	0≤r≤31	PC = PC + 1	SP = SP-1

Слов 1 (2 байта)
Циклов 2

Команда RCALL

Вызов подпрограммы, находящейся в диапазоне адресов [- 2 Кслов + 2 К слов] относительно текущего Адрес возврата (См также CALL)
Операция

PC = PC + k + 1

Синтаксис	Операнды	PC
RCALL k	-2K≤k≤2K	PC = PC + k + 1

Стек
STACK = PC + 1
SP = SP - 2

Слов 1 (2 байта)
Циклов 3

Команда RET

Возврат из подпрограммы Адрес возврата загружается из стека
Операция

PC(15 – 0) = STACK

Синтаксис	Операнды	PC	Стек
RET		PC(15 – 0) = STACK	SP = SP + 2

Слов 1 (2 байта)
Циклов 4

Команда RETI

Возврат из прерывания Адрес возврата загружается из стека Устанавливается общий флаг прерываний I

Операция

PC(15 – 0) = STACK

Синтаксис	Операнды	PC	Стек
RETI		PC(15 – 0) = STACK	SP = SP + 2

Признаки I(1)
Слов 1 (2 байта)
Циклов 4

Команда RJMP

Переход по адресу в диапазоне [PC – 2K PC+2K] слов В ассемблере вместо относительных адресов используются символические метки

Операция

PC = PC + k + 1

Синтаксис	Операнды	PC
RJMP k	-2K≤k≤2K	PC = PC + k + 1

Стек Не изменяется
Слов 1 (2 байта)
Циклов 2

Команда ROL

Сдвигает все биты регистра Rd на один разряд влево Флаг C сдвигается в бит 0 регистра Rd Бит 7 регистра Rd сдвигается во флаг C

Операция

C⇐b7⇐b6⇐ ⇐b0⇐C

Синтаксис	Операнды	PC	Признаки
ROL Rd	0≤d≤31	PC = PC + 1	H,S,V,N,Z,C

Слов 1 (2 байта)
Циклов 1

Команда ROR

Сдвигает все биты регистра Rd на один разряд вправо Флаг C сдвигается в бит 7 регистра Rd Бит 0 регистра Rd сдвигается во флаг C

Операция

C⇒b7⇒b6⇒ ⇒b0⇒C

Синтаксис	Операнды	PC	Признаки
ROR Rd	0≤d≤31	PC = PC + 1	S,V,N,Z,C

Слов 1 (2 байта)
Циклов 1

Команда SBCI

Вычитает из регистра константу с учетом значения флага C, размещает результат в регистре Rd

Операция

Rd = Rd - K - C

Синтаксис	Операнды	PC
SBCI Rd,K	16≤d≤31, 0≤K≤255	PC = PC + 1

Признаки H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда SBI

Устанавливает определенный бит в регистре ввода/вывода Команда работает с нижними 32 регистрами ввода/вывода (с адресами в диапазоне 0 – 31)

Операция

I/O(P,b) = 1

Синтаксис	Операнды	PC
SBI P,b	0≤P≤31, 0≤b≤7	PC = PC + 1

Слов 1 (2 байта)
Циклов 2

Команда SBIC

Тестирует отдельный бит в регистре ввода/вывода и пропускает следующую команду, если бит сброшен Команда работает с нижними 32 регистрами ввода/вывода (адреса 0 – 31)

Операция

if I/O(P,b) = 0 **then** PC = PC + 2 (or 3)
else PC = PC + 1

Синтаксис	Операнды
SBIC P,b	0≤P≤31, 0≤b≤7

PC
PC = PC + 1, если условие false
PC = PC + 2, если следующая команда однословная
PC = PC + 3, если следующая команда двухсловная

Слов 1 (2 байта)
Циклов 2 если условие false
3 если условие true

Команда SBIS

Тестирует отдельный бит в регистре ввода/вывода и пропускает следующую команду, если бит установлен Команда работает с нижними 32 регистрами ввода/вывода (адреса 0-31)

Операция

if I/O(P,b) = 1 **then** PC = PC + 2 (or 3)
else PC = PC + 1

Синтаксис	Операнды
SBIS P,b	0≤P≤31, 0≤b≤7

PC
PC = PC + 1, если условие false
PC = PC + 2, если следующая команда однословная
PC = PC + 3, если следующая команда двухсловная

Слов 1 (2 байта)
Циклов 2 если условие false
3 если условие true

Команда SBIW

Вычитает непосредственное значение (0 – 63) из регистровой пары и размещает результат в регистровой паре Команда работает с верхними четырьмя регистровыми парами и удобна для операций с регистрами-указателями

Операция

Rdh Rdl = Rdh Rdl - K

Синтаксис	Операнды	PC
SBIW Rdl,K	dl ∈ [24,26,28,30] 0≤K≤63	PC = PC + 1

Признаки S,V,N,Z,C
Слов 1 (2 байта)
Циклов 2

Команда SBR

Устанавливает определенные биты регистра Rd Выполняет операцию логическое ИЛИ над содержимым регистра Rd и константой K, размещает результат в регистре Rd

Операция

Rd = Rd **or** K

Синтаксис	Операнды	PC
SBR Rd,K	16≤d≤31, 0≤K≤255	PC = PC + 1

Признаки S,V(0),N,Z
Слов 1 (2 байта)
Циклов 1

Команда SBRC

Тестирует отдельный бит регистра и пропускает следующую команду, если бит сброшен

Операция

if Rr(b) = 0 **then** PC = PC + 2 (or 3)
else PC = PC + 1

Синтаксис	Операнды
SBRC Rr,b	0≤r≤31, 0≤b≤7

PC
PC = PC + 1, если условие false
PC = PC + 2, если следующая команда однословная
PC = PC + 3, если следующая команда двухсловная

Слов 1 (2 байта)
Циклов 1 если условие false
2 если условие true

Команда SBRS

Тестирует отдельный бит регистра и пропускает следующую команду, если бит установлен

Операция

if Rr(b) = 1 **then** PC = PC + 2 (or 3)
else PC = PC + 1

Синтаксис	Операнды
SBRS Rr,b	0≤r≤31, 0≤b≤7

PC
PC = PC + 1, если условие false
PC = PC + 2, если следующая команда однословная
PC = PC + 3, если следующая команда двухсловная

Слов 1 (2 байта)
Циклов 1 если условие false
2 если условие true

Команда SEC

Устанавливает флаг переноса (C) в регистре SREG

Операция

C = 1

Синтаксис	Операнды	PC	Признаки
SEC		PC = PC + 1	C(1)

Слов 1 (2 байта)
Циклов 1

Команда SEH

Устанавливает флаг полупереноса (H) в регистре SREG

Операция

H = 1

Синтаксис	Операнды	PC	Признаки
SEH		PC = PC + 1	H(1)

Слов 1 (2 байта)
Циклов 1

Команда SEI

Устанавливает общий флаг прерываний I в регистре SREG

Операция

I = 1

Синтаксис	Операнды	PC	Признаки
SEI		PC = PC + 1	I(1)

Слов 1 (2 байта)
Циклов 1

Команда SEN

Устанавливает флаг отрицательного результата N в регистре SREG

Операция

N = 1

Синтаксис	Операнды	PC	Признаки
SEN		PC = PC + 1	N(1)

Слов 1 (2 байта)
Циклов 1

Команда SER

Загружает $FF непосредственно в регистр Rd

Операция

Rd = $FF

Синтаксис	Операнды	PC
SER Rd	16≤d≤31	PC = PC + 1

Слов 1 (2 байта)
Циклов 1

Команда SES

Устанавливает флаг знака S в регистре SREG

Операция

S = 1

Синтаксис	Операнды	PC	Признаки
SES		PC = PC + 1	S(1)

Слов 1 (2 байта)
Циклов 1

Команда SET
Устанавливает флаг T в регистре SREG
Операция
T = 1

Синтаксис	Операнды	PC	Признаки
SET		PC = PC + 1	T(1)

Слов 1 (2 байта)
Циклов 1

Команда SEV
Устанавливает флаг переполнения V в регистре SREG
Операция
V = 1

Синтаксис	Операнды	PC	Признаки
SEV		PC = PC + 1	V(1)

Слов 1 (2 байта)
Циклов 1

Команда SEZ
Устанавливает флаг нулевого результата Z в регистре SREG
Операция
Z = 1

Синтаксис	Операнды	PC	Признаки
SEZ		PC = PC + 1	Z(1)

Слов 1 (2 байта)
Циклов 1

Команда SLEEP

Устанавливает «спящий» режим в зависимости от состояния регистра управления MCUCR При появлении прерывания контроллер «просыпается», выполняет одну команду после SLEEP и затем отрабатывает прерывание

Синтаксис	Операнды	PC
SLEEP		PC = PC + 1

Слов 1 (2 байта)
Циклов 1

Команда ST

Загружает один байт из регистра в SRAM Адрес ячейки SRAM находится в одном из 16-битных регистров-указателей X, Y или Z регистрового файла Доступ к памяти осуществляется внутри текущей страницы SRAM (64 Кбайта) Для доступа к другой странице SRAM необходимо изменить номер страницы в регистре RAMPX, RAMPY или RAMPZ (в зависимости от выбранного регистра-указателя)

Операция	Комментарий
(i) (R) = Rr	R - X/Y/Z Не изменяется
(ii) (R) = Rr, R = R+1	R - X/Y/Z После инкрементирования
(iii) R = R-1, (R) = Rr	R - X/Y/Z До декрементирования

Синтаксис	Операнды	PC	Обозначения
ST R,Rr	0≤r≤31	PC = PC + 1	R - X/Y/Z
(ii)ST R+,Rr	0≤r≤31	PC = PC + 1	R - X/Y/Z
(iii)ST -R,Rr	0≤r≤31	PC = PC + 1	R - X/Y/Z

Слов 1 (2 байта)
Циклов 2

Команда STD

Загружает один байт из регистра в SRAM Адрес ячейки SRAM находится в одном из 16-битных регистров-указателей Y или Z регистрового файла Доступ к памяти осуществляется внутри текущей страницы SRAM (64K байта) Для доступа к другой странице SRAM необходимо изменить номер страницы в регистре RAMPY или RAMPZ (в зависимости от выбранного регистра-указателя)

Операция	Комментарий
(R+q) = Rr	R - Y/Z Не изменяется, q - Смещение

Синтаксис	Операнды	PC	Обозначения
STD R+q, Rr	0≤r≤31, 0≤q≤63	PC = PC + 1	R - Y/Z

Слов 1 (2 байта)
Циклов 2

Команда STS

Запоминание байта из регистра в ячейке SRAM 16-битный адрес должен быть корректен Доступ к памяти ограничен текущей страницей SRAM (64 Кбайта) Для доступа к памяти выше 64 Кбайт используется регистр RAMPZ

Операция

(k) = Rr

Синтаксис	Операнды	PC
STS k,Rr	0≤r≤31, 0≤k≤65535	PC = PC + 2

Слов 2 (4 байта)
Циклов 3

Команда SUB

Вычитание содержимого двух регистров Результат размещается в регистре Rd

Операция

Rd = Rd − Rr

Синтаксис	Операнды	PC
SUB Rd,Rr	0≤d≤31, 0≤r≤31	PC = PC + 1

Признаки H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда SUBI

Вычитание константы из содержимого регистра Результат размещается в регистре Rd Команда работает с регистрами R16-R31

Операция

Rd = Rd − K

Синтаксис	Операнды	PC
SUBI Rd,K	16≤d≤31, 0≤K≤255	PC = PC + 1

Признаки H,S,V,N,Z,C
Слов 1 (2 байта)
Циклов 1

Команда SWAP

Меняет местами старший и младший тетрады регистра

Операция

R(7-4) = Rd(3-0), R(3-0) = Rd(7-4)

Синтаксис	Операнды	PC
SWAP Rd	0≤d≤31	PC = PC + 1

Слов 1 (2 байта)
Циклов 1

Команда TST

Тестирует регистр на нулевой или отрицательный результат Выполняет операцию AND регистра с самим собой Регистр остается неизмененным

Операция

Rd **and** Rd

Синтаксис	Операнды	PC	Признаки
TST Rd	0≤d≤31	PC = PC + 1	S,V(0),N,Z

Слов 1 (2 байта)
Циклов 1

Команда WDR

Перезапуск Watchdog таймера

Бродин В.Б., Калинин А.В.

Системы на микроконтроллерах и БИС программируемой логики

Главный редактор *Н В Григорьева*
Технический директор *Е В Новиков*
Главный художник *О В Будко*

Подписано в печать 24 12 2001 Формат 70×100 $^{1}/_{16}$
Печать офсетная 25 печ л
Тираж 5000 экз Заказ № 5121

«Издательство ЭКОМ», лицензия ЛД № 065036 от 28 02 1997
ПБОЮЛ Тараев Сергей Павлович, *лицензия ИД № 01612 от 19 04 2000*
117342 Москва, ул Бутлерова, д 17, оф 105
Телефон для оптовых покупателей (095) 330-68-65

Отпечатано в ОАО «Можайский полиграфический комбинат»
143200 г Можайск, ул Мира, 93