Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Севастопольский государственный университет»

ИССЛЕДОВАНИЕ СИСТЕМЫ КОМАНД МИКРОКОНТРОЛЛЕРОВ AVR

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

по выполнению лабораторных работ

для студентов, обучающихся по направлению
09.03.02 «Информационные системы и технологии»

Севастополь
2019

Исследование системы команд микроконтроллеров AVR. Методические указания по проведению лабораторных работ для студентов, обучающихся по направлению 09.03.02 «Информационные системы и технологии». Сост. Чернега В.С. – Севастополь, СевГУ, 2019 – 18 с.

Цель указаний: оказать помощь студентам в изучении архитектуры и системы команд микроконтроллеров семейства AVRmega и особенностей программирования арифметических и логических операций на языке ассемблер, а также операций ввода/вывода информации.

Методические указания рассмотрены и утверждены на методическом семинаре и заседании кафедры информационных систем
(протокол № _____ от «____» _____ 2019 г.)

Допущено учебно-методическим центром СевГУ в качестве методических указаний.

Рецензент: Кудрявченко И.В., канд. техн. наук, доцент кафедры ИС

СОДЕРЖАНИЕ

ЛАБОРАТОРНАЯ РАБОТА №2

Исследование системы команд микроконтроллеров AVR

1. Цель работы

Изучить основные команды микроконтроллеров семейства AVR и исследовать влияние команд различного типа на флаги регистра состояния и регистры управления портами ввода/вывода, а также на работу функциональных узлов устройства. Приобрести практические навыки разработки программ средней сложности на языке Ассемблера и отладки их в специализированной инструментальной среде.

2. Особенности архитектуры микроконтроллеров семейства ATmega

2.1 Краткая характеристика системы команд

Система команд микроконтроллеров ATMEL семейства AVR очень большая и в то же время эффективная. Одной из отличительных особенностей микроконтроллеров AVR является то, что почти все команды выполняются за 1 тактовый цикл. Исключение составляют команды перехода. Это существенно увеличивает производительность микроконтроллера даже при относительно невысокой тактовой частоте.

Все команды можно классифицировать на 5 типов:

1) арифметические команды и команды сдвига;
2) логические команды;
3) команды перехода (передачи управления);
4) команды передачи (пересылки) данных;
5) побитовые команды и команды тестирования битов.

Обозначение операндов команд приведены в табл.2.1.

Таблица 2.1 – Обозначение операндов ассемблера

Rd	Регистр - приемник, место, куда заносится результат выполнения команды
Rs	Регистр - источник в двухоперандных командах. Его значение после выполнения команды не изменяется.
I/O	Регистр ввода-вывода. Это порты, таймеры и т.д.
K	8-ми разрядная константа в операциях со "старшими" регистрами общего назначения (R16-R31)
b	Номер бита в операциях с регистрами ввода-вывода
A	16-ти разрядный адрес при работе с памятью данных (прямая адресация)
q	6-ти разрядное смещение при работе с памятью данных
X	Регистровая пара X. Состоит их регистров **XL** (R26) и **XH** (R27)

Y	Регистровая пара Y. Состоит их регистров **YL** (R28) и **YH** (R29)
Z	Регистровая пара Z. Состоит их регистров **ZL** (R30) и **ZH** (R31)

Значения битов (флагов) регистра состояния микроконтроллера приведено в таблице 2.2. Адрес регистра 0x3F (либо 0x5F).

Таблица 2.2 – Значения флагов регистра состояния МК

I	**SREG.7**	Бит разрешения прерывания. Если он = 0, то все прерывания в МК запрещены. Если он =1, то разрешением прерываний будут управлять соответствующие биты периферии.
T	**SREG.6**	Битовый аккумулятор. С этим битом работают команды BST и BLD
H	**SREG.5**	Флаг переноса из младшей тетрады
S	**SREG.4**	Sign - ислючающее ИЛИ битов **N** и **V**
V	**SREG.3**	oVerflow - переполнение
N	**SREG.2**	Negative - Результат операции < 0
Z	**SREG.1**	Zero - Результат операции равен нулю
C	**SREG.0**	Carry - Флаг переноса

Основные команды микроконтроллеров семейства AVR даны в приложении А. Наиболее часто используемые команды приведены в таблице 2.3.

Таблица 2.3 – Наиболее часто используемые команды МК семейства AVR

	Мнемоника	Описание
	Команды передачи данных	
	MOV Rd, Rr	; Move between Registers Rd ← Rr
	LDI Rd, K	; Load Immediate Rd ← K
	LD Rd, X	; Load Indirect Rd ← (X)
	LD Rd, X+	; Load Indirect and Post-inc. Rd ← (X), X ← X + 1
	LD Rd, -X	; Load Indirect and Pre-dec. X ← X - 1, Rd ← (X)
	CLR Rd	; Clear Register Rd ← Rd ⊕ Rd
	SER Rd	; Set Register Rd ← $FF
	ST X, Rr	; Store Indirect (X)=← Rr
	ST X+, Rr	; Store Indirect and Post-inc. (X)=← Rr, X ← X + 1
	IN Rd, P	; Input Port Rd ← P
	OUT P, Rr	; Output Port P ← Rr
	PUSH Rr	; Push Register on Stack STACK ← Rr
	POP Rd	; Pop Register from Stack Rd ← STACK
	Команды с битами	
	SBI P, b	; Set Bit in I/O Register I/O(P,b) ← 1
	CBI P, b	; Clear Bit in I/O Register I/O(P,b) ← 0
	Команды арифметические и сдвига	
	ADD Rd, Rr	; Add Two Registers Rd ← Rd + Rr Z,C,N,V,H 1
	ADC Rd, Rr	; Add with Carry Two Registers Rd ← Rd + Rr + C

	SUBI Rd, K	; Subtract Constant from Register Rd ← Rd - K
	LSL Rd	; Logical Shift Left Rd(n+1) ← Rd(n), Rd(0) ← 0
	ROR Rd	; Rotate Right through Carry Rd(7) ←=C, Rd(n) ← Rd(n+1)
	ASR Rd	; Arithmetic Shift Right Rd(n) ← Rd(n+1)
	Команды перехода	
	SBRC Rr, b	; Skip if Bit in Register Cleared if (Rr(b) = 0) PC ← PC + 2 or 3
	SBRS Rr, b	; Skip if Bit in Register is Set if (Rr(b) = 1) PC ← PC + 2 or 3
	SBIC P, b	; Skip if Bit in I/O Register Cleared if (P(b) = 0) PC ← PC + 2 or
	SBIS P, b	; Skip if Bit in I/O Register is Set if (P(b) = 1) PC ← PC + 2 or 3
	RJMP k	Relative Jump PC ← PC + k + 1
	IJMP	Indirect Jump to (Z) PC ← Z
	RCALL k	Relative Subroutine Call PC ← PC + k + 1
	ICALL	Indirect Call to (Z) PC ← Z
	RET	Subroutine Return PC ← STACK
	RETI	Interrupt Return PC ← STACK I
	CPSE Rd, Rr	Compare, Skip if Equal if (Rd = Rr) PC=← PC + 2 or 3
	CP Rd, Rr	Compare Rd − Rr
	CPC Rd, Rr	Compare with Carry Rd − Rr − C
	BRBS s, k	Branch if Status Flag Set if (SREG(s) = 1) then PC ← =PC + k + 1
	BRBC s, k	Branch if Status Flag Cleared if (SREG(s) = 0) then PC ← =PC + k + 1
	BREQ k	Branch if Equal if (Z = 1) then PC ← PC + k + 1
	BRNE k	Branch if Not Equal if (Z = 0) then PC ← PC + k + 1
	BRCS k	Branch if Carry Set if (C = 1) then PC ← PC + k + 1
	BRCC k	Branch if Carry Cleared if (C = 0) then PC ← PC + k + 1
	Прочие, часто используемые команды	
	SEI	Global Interrupt Enable I ← 1 I 1
	CLI	Global Interrupt Disable I=← 0
	NOP	No Operation

Примеры применения некоторых команд:

```
ldi   R16, 0b00001001      ; загрузка константы в РОН  (R16 - R32)
out  PORTD, R16            ; запись значения регистра в порт D
in    R25, PORTB           ; считать значения Port B в регистр R25
cpi   R25, 4               ; сравнить считанное значение с константой =4
breq   exit                ; переход на метку если было равно
…                          ;
exit:                      ; метка
nop                        ; пустой (холостой) такт
sbi   PORTD, PD4           ; записать в 4-й бит порта D лог. "1"
rcall  my_delay            ; вызов подпрограммы задержки
cbi  PORTD, PD4            ; сброс 4-го бита порта D
....                       ;
my_delay:                  ; подпрограмма задержки (4 такта)
nop                        ; холостой такт
nop                        ;
nop                        ;
nop                        ;
ret                        ; возвращение из подпрограммы (3 такта)
                           ; вызов и выполнение подпрограммы my_delay
                           ; займёт 4+ 5+3 = 12 тактов
```

2.2. Особенности программирования регистров и портов ввода/вывода AVR-микроконтроллеров

Одним из важных аспектов программирования микроконтроллеров является работа с регистрами и портами. У микроконтроллеров серии AVR имеется несколько регистров ввода/вывода и 32 рабочих регистра общего назначения. Регистры ввод/вывода имеют номера, например от $0 до $3F. На схемах порты ввода/вывода имеют буквенные обозначения PortB, PinB, PortD, PinD и др. Причем, в разных моделях микроконтроллеров их адреса могут отличаться. Для того чтобы связать адреса регистров ввода/вывода и других регистров микроконтроллера с их именами, следует в начале программы подключить заголовочный файл, соответствующий используемому микроконтроллеру. Так, например, если программа предназначена для микроконтроллера AT90s8515, то нужно с помощью специальной директивы подключить соответствующий файл:

```
.include   "8515def.inc"
```

Для микроконтроллеров ATmega8 в ATmega16 в директиву .include записывается “m8def.inc “ и “m16def.inc“соответственно, для МК типа Tiny – “tn12def.inc” либо “tn2313def.inc”, в зависимости от модели микроконтроллера.

При программировании микроконтроллеров нельзя непосредственно записать число в регистр ввода/вывода. Вместо этого нужно записать число в один из рабочих регистров общего назначения, а затем скопировать значение этого регистра в регистр ввода/вывода. Рабочие регистры обозначаются как R0, R1, R2, ... , R31.

Для упрощения написания программ очень удобно давать регистрам имена. Целесообразно давать имена, соответствующие хранимой информации. Например, если регистр R16 используется для хранения временной информации, то его можно назвать temp. Это выполняется с помощью директивы def:

```
.def   temp   = R16
```

Микроконтроллеры серии Atmel Mega имеют несколько портов Ввода/Вывода, обозначающихся буквами (PORTA, PORTB и т.д.). Эти порты являются двунаправленными 8-разрядными, т.е. каждый такой порт имеет 8 выводов (ножек или пинов, от англ. *pin*) позволяет читать или записывать до 8 двоичных сигналов. Каждая линия порта может быть запрограммирована на вход или на выход. Выходные усилители (драйверы) обеспечивают токовую нагрузочную способность 20 мА на линию порта (максимальное значение 40 мА), что позволяет, например, непосредственно подключать к микроконтроллеру светодиоды и биполярные транзисторы. Общая токовая нагрузка на все линии одного порта не должна превышать 80 мА (все значения приведены для напряжения питания 5 В). Для управления каждым из портов ВВ имеются три 8-разрядных регистра в/в: DDRx, PORTx и PINx. Упрощенная схема ввода/вывода изображена на рисунке 2.1. Биты регистра DDR задают режим ввода (лог. 0) или вывода (лог.1). Регистр PINx позволяет считать текущее значение

порта, независимо от того, какой режим ввода/вывода установлен. Регистр PORTx в режиме вывода задает выходные сигналы на выводах порта, а в режиме ввода задает включение (лог. 1) или отключение (лог. 0) «подтягивающих» резисторов для выводов порта.

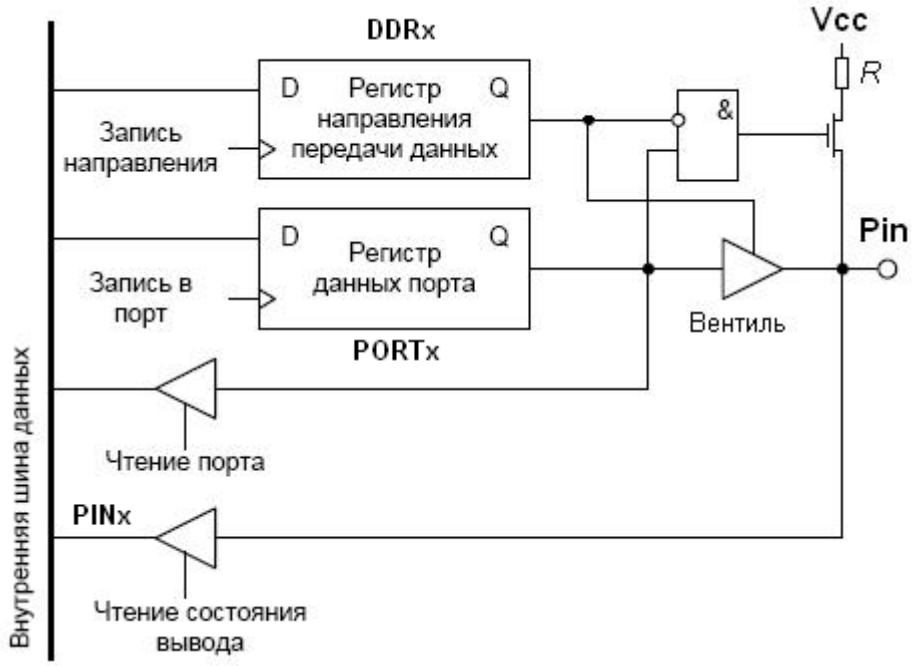

Рисунок 2.1 – Схема порта ввода/вывода микроконтроллеров Atmel

Для того чтобы сделать пины выходами, необходимо записать в соответствующие биты регистра DDRx "1". Например, если требуется сделать пин PB7 порта B входом, а остальные ножки выходами, то для этого необходимо записать в регистр DDRB значение 0b01111111. Приставка 0b означает, что число записано в двоичном виде. При начальном запуске регистры DDRx обнулены, т.е. все ножки (пины) микроконтроллера являются входами. Рекомендуется неиспользуемые ножки в устройстве делать входами и «потянуть» их потенциал к источнику питания.

«Подтягивание» внешнего вывода к высокому потенциалу осуществляется путем подключения вывода через транзисторный ключ и подтягивающий резистор R. Как видно из схемы, «подтягивание» реализуется только тогда, когда внешний вывод (Pin) работает в режиме ввода данных, т.е., когда бит регистра направления данных сброшен в 0, а бит регистра данных порта установлен в единицу.

Для побитного обращения к линиям порта, можно использовать команды SBI, CBI, SBIS, SBIC... Команда SBI устанавливает соответствующий бит порта в "1", а команда CBI делает обратное — очищает соответствующий бит т.е. записывает 0 в этот бит. Например:

```
1. ldi R16, 0xFF
2. out DDRB, R16        ; все ножки (пины) — выходы
3. sbi PortB, 5         ; установить на ножке лог. "1"
```

```
4. cbi PortB, 5          ; сбросить, на ножке появится лог. "0"
```

3. Программа и методика выполнения лабораторной работы

4.1. Повторить теоретический материал, касающийся архитектуры микроконтроллеров AVR и схеме организации памяти и портов ввода/вывода микроконтроллера.
4.2. Ознакомиться с системой команд микроконтроллеров AVR и изучить наиболее часто используемые команды.
4.3. Составить алгоритм программы согласно заданному варианту (см. Приложение А).
4.4. Составить текст программы на языке ассемблера и откомпилировать программу.
4.5. Исправить ошибки, при их обнаружении компилятором.
4.6. Провести исследование процесса выполнения программы в среде AVR Studio.
4.7. Составить отчет по результатам исследований.

4. Описание лабораторной установки

Лабораторная установка для создания и отладки приложений и исследования процессов функционирования микроконтроллеров типа AVR Atmel представляет собой персональный компьютер, на котором установлена интегрированная среда разработки AVR Studio 4 либо AVR Studio старших версий. Описание установки приведено в лабораторной работе №1.

5. Содержание отчета

5.1. Цель работы.
5.2. Программа исследований.
5.3. Перечень программно доступных регистров и схема портов ввода/вывода.
5.4. Тексты исследуемой программы на языке ассемблера и языке Си.
5.5. Описание результатов исследований.
5.6. Выводы по работе.

6. Контрольные вопросы

6.1.Расскажите о структуре регистров микроконтроллера и их назначении.
6.2.В чем состоит особенность регистров X, Y, Z?

6.3. Какие флаги определяют состояние микроконтроллера?
6.4. Проведите анализ системы команд микроконтроллера и приведите примеры их применения в конкретных задачах.
6.5. Проведете сравнительный анализ команд перехода микроконтроллера.
6.6. Начертите упрощенную схему регистров ввода/вывода и поясните принцип задания режимов работы.
6.7. Что означает термин «подтягивающий» резистор и каково его назначение?
6.8. Зачем в программу включается директива include “ “ и что содержится во включаемом файле?
6.9. В какой области памяти размещается стек и как осуществляется инициализация указателя стека?

7. Список рекомендованной литературы

7.1. Бальзамов А. Ю. Программирование на ассемблере для AVR-микроконтроллеров: Лаб. практикум по основам микропроцессорной техники / А. Ю. Бальзамов. – Саранск: Изд-во Мордов. ун-та, 2012. – 108 с.
7.2. Баранов В.Н. Применение микроконтроллеров AVR: схемы, алгоритмы, программы. – М.: Издательский дом «Додека-XXI», 2004. – 288 с.
7.3. Белов А.В. Создаем устройства на микроконтроллерах. – СПб. Наука и техника, 2007. – 304 с.
7.4. Мортон Дж. Микроконтроллеры AVR. Вводный курс. / Пер. с англ.- М.: Издательский дом «Додека-XXI», 2006. – 272 с.
7.5. Ревич Ю.В. Практическое программирование микроконтроллеров Atmel AVR на языке ассемблера. – СПб., БХВ-Петербург, 2008. – 384 с.
7.6. Хартов В. Я. Микроконтроллеры AVR. Практикум для начинающих: учеб. пособие / В. Я. Хартов. – 2-е изд., испр. и доп. – М. : Изд-во МГТУ им. Н. Э. Баумана, 2012. – 280 с.
7.7. Программирование с нуля в AVRStudio 5. http://chipmk.ru/index.php/component/k2/item/118-programmirovanie-c-nulya-v-avrstudio-5-ch-1

Приложение А. Варианты задания

1. Составить программу сложения 5 однобайтных беззнаковых чисел, находящихся в ОЗУ. Если в результате сложения происходит переполнение разрядной сетки, вывести в порт B число $FF. Если переполнения нет, то выводить непрерывно в порт D поочередно числа $55 и $AA.

2. Составить программу сложения 8 однобайтных беззнаковых чисел, находящихся в ОЗУ. Затем из полученной суммы вычесть число 220. Если в результате вычитания происходит переполнение разрядной сетки, вывести в порт B число $FF. Если переполнения нет, то выводить непрерывно в порт D поочередно числа $0F и $F0.

3. Составить программу сложения 2 однобайтных беззнаковых чисел, находящихся в ОЗУ. Затем полученную сумму умножить на 7. Если в результате сложения происходит переполнение разрядной сетки, вывести в порт B число $FF. Если переполнения нет, то выводить непрерывно в порт D поочередно числа $5A и $CA.

Приложение Б. Команды микроконтроллеров семейства AVR

Таблица 1. Арифметические и сдвиговые команды

Мнемоника	Описание	Операция	Флаги
ADD Rd, Rr	Сложение двух РОН	Rd = Rd + Rr	Z,C,N,V,H
ADC Rd, Rr	Сложение двух РОН с переносом	Rd = Rd + Rr + C	Z,C,N,V,H
ADIW Rd, K **	Сложение регистровой пары с константой	Rd+1:Rd = Rd+1:Rd + K	Z,C,N,V,S
SUB Rd, Rr	Вычитание двух РОН	Rd = Rd - Rr	Z,C,N,V,H
SUBI Rd, K *	Вычитание константы из РОН	Rd = Rd - K	Z,C,N,V,H
SBC Rd, Rr	Вычитание двух РОН с заемом	Rd = Rd – Rr - C	Z,C,N,V,H
SBCI Rd, K *	Вычитание константы из РОН с заемом	Rd = Rd – K - C	Z,C,N,V,H
SBIW Rd, K **	Вычитание константы из регистровой пары	Rd+1:Rd = Rd+1:Rd - K	Z,C,N,V,S
DEC Rd	Декрементирование РОН	Rd = Rd - 1	Z,N,V
INC Rd	Инкрементирование РОН	Rd = Rd + 1	Z,N,V
ASR Rd	Арифметический сдвиг вправо	Rd(n) = Rd(n+1), n = 0..6	Z,C,N,V
LSL Rd	Логический сдвиг влево	Rd(n+1) = Rd(n), Rd(0) = 0	Z,C,N,V
LSR Rd	Логический сдвиг вправо	Rd(n) = Rd(n+l), Rd(7) = 0	Z,C,N,V
ROL Rd	Сдвиг влево через перенос	Rd(0) = C, Rd(n+l) = Rd(n), C = Rd(7)	Z,C,N,V
ROR Rd	Сдвиг вправо через перенос	Rd(7) = C, Rd(n) = Rd(n+l), C = Rd(0)	Z,C,N,V
MUL Rd, Rr	Умножение беззнаковых чисел	R1:R0 = Rd*Rr	Z,C
MULS Rd, Rr *	Умножение чисел со знаком	R1:R0 = Rd*Rr	Z,C
MULSU Rd, Rr ***	Умножение беззнакового числа на число со знаком	R1:R0 = Rd*Rr	Z,C
FMUL Rd, Rr ***	Умножение дробных беззнаковых чисел	R1:R0 = (Rd*Rr) << l	Z,C
FMULS Rd, Rr ***	Умножение дробных чисел со знаком	R1:R0 = (Rd*Rr) << l	Z,C
FMULSU Rd, Rr ***	Умножение дробного беззнакового числа и дробного числа со знаком	R1:R0 = (Rd*Rr) << l	Z,C
$0 \le d \le 31$, $0 \le r \le 31$, $0 \le K \le 255$ (* в командах SUBI, SBCI, MULS: $16 \le d \le 31$, $16 \le r \le 31$; ** в командах ADIW, SBIW: d = {24, 26, 28, 30}, $0 \le K \le 63$; *** в командах MULSU, FMUL, FMULS, FMULSU $16 \le d \le 23$, $16 \le r \le 23$)			

Таблица 2. Логические команды

Мнемоника	Описание	Операция	Флаги
AND Rd, Rr	«Логическое И» двух РОН	Rd = Rd • Rr	Z,N,V
ANDI Rd, K *	«Логическое И» РОН и константы	Rd = Rd • K	Z,N,V
EOR Rd, Rr	«Исключающее ИЛИ» двух РОН	Rd = Rd⊕Rr	Z,N,V
OR Rd, Rr	«Логическое ИЛИ» двух РОН	Rd = Rd v Rr	Z,N,V
ORI Rd, K *	«Логическое ИЛИ» РОН и константы	Rd = Rd v K	Z,N,V
COM Rd	Перевод в обратный код	Rd = $FF - Rd	Z,C,N,V
NEG Rd	Перевод в дополнительный код	Rd = $00 - Rd	Z,C,N,V,H
CLR Rd	Сброс всех битов РОН	Rd = Rd⊕Rd	Z,N,V
SER Rd *	Установка всех битов РОН	Rd = $FF	-
TST Rd	Проверка РОН на отрицательное или нулевое значение	Rd • Rd	Z,N,V
SWAP Rd	Обмен местами полубайтов в РОН	Rd(3...0) = Rd(7...4), Rd(7...4) = Rd(3...0)	-
0 ≤ d ≤ 31, 0 ≤ r ≤ 31, 0 ≤ K ≤ 255 (* в командах ANDI, ORI и SER: 16 ≤ d ≤ 31)			

Таблица 3. Команды передачи управления

Мнемоника	Описание	Операция	Флаги
RJMP k **	Относительный безусловный переход	PC = PC + к + 1	-
IJMP	Косвенный безусловный переход	PC = Z	-
JMP k ***	Абсолютный переход	PC = k	-
RCALL k **	Относительный вызов подпрограммы	STACK = PC, PC = PC + k + 1	-
ICALL	Косвенный вызов подпрограммы	STACK = PC, PC = Z	-
CALL k ***	Абсолютный вызов подпрограммы	STACK = PC, PC = k	-
RET	Возврат из подпрограммы	PC = STACK	-
RETI	Возврат из п/п обработки прерывания	PC = STACK	I
CP Rd, Rr	Сравнение РОН	Rd - Rr	Z,N,V,C,H
CPC Rd, Rr	Сравнение РОН с учетом переноса	Rd – Rr - C	Z,N,V,C,H
CPI Rd, K *	Сравнение РОН с константой	Rd - K	Z,N,V,C,H
CPSE Rd, Rr	Сравнение и пропуск следующей команды при равенстве	Если Rd = Rr, то PC = PC + 2(3)	-
SBRC Rr, b	Пропуск следующей команды, если бит РОН сброшен	Если Rr.b = 0, то PC = PC + 2(3)	-
SBRS Rr, b	Пропуск следующей команды, если бит РОН установлен	Если Rr.b= 1, то PC = PC + 2(3)	-
SBIC P, b	Пропуск следующей команды, если бит РВВ сброшен	Если P.b = 0, то PC = PC + 2(3)	-
SBIS P, b	Пропуск следующей команды, если бит РВВ установлен	Если P.b = 1, то PC = PC + 2(3)	-
BRBC s, k	Переход, если флаг s регистра SREG сброшен	Если SREG.s = 0, то PC = PC + k + 1	-

Таблица 3 Команды передачи управления (продолжение)

Мнемоника	Описание	Операция	Флаги
BRBS s, k	Переход, если флаг s регистра SREG установлен	Если SREG.s = l, то PC = PC + k + 1	-
BRCS k	Переход по переносу	Если C = 1, то PC = PC + k + 1	-
BRCC k	Переход, если нет переноса	Если C = 0, то PC = PC + k + 1	-
BREQ k	Переход по «равно»	Если Z = 1, то PC = PC + k + 1	-
BRNE k	Переход по «не равно»	Если Z = 0, то PC = PC + k + 1	-
BRSH k	Переход по «больше или равно»	Если C = 0, то PC = PC + k + 1	-
BRLO k	Переход по «меньше»	Если C =1, то PC = PC + k + 1	-
BRMI k	Переход по «отрицательное значение»	Если N = 1, то PC = PC + k + 1	-
BRPL k	Переход по «положительное значение»	Если N = 0, то PC = PC + k + 1	-
BRGE k	Переход по «больше или равно» (числа со знаком)	Если (N⊕V) = 0, то PC = PC + k + 1	-
BRLT k	Переход по «меньше нуля» (числа со знаком)	Если (N⊕V) = l, то PC = PC + k + 1	-
BRHS k	Переход по половинному переносу	Если H = 1, то PC = PC + k + 1	-
BRHC k	Переход, если нет половинного переноса	Если H = 0, то PC = PC + k + 1	-
BRTS k	Переход, если флаг T установлен	Если T = 1, то PC = PC + k + 1	-
BRTC k	Переход, если флаг T сброшен	Если T = 0, то PC = PC + k + 1	-
BRVS k	Переход по переполнению дополнительного кода	Если V = 1, то PC = PC + k + 1	-
BRVC k	Переход, если нет переполнения дополнительного кода	Если V = 0, то PC = PC + k + 1	-
BRID k	Переход, если прерывания запрещены	Если I = 0, то PC = PC + k + 1	-
BRIE k	Переход, если прерывания разрешены	Если I = 1, то PC = PC + k + 1	-
$0 \le d \le 31$, $0 \le r \le 31$, $0 \le b \le 7$, $0 \le s \le 7$, $0 \le K \le 255$, $-64 \le k \le 63$, $0 \le P \le 31$ (* в команде CPI: $16 \le d \le 31$; ** в командах RJMP, RCALL: $-2048 \le k \le +2047$; *** в командах JMP, CALL: $0 \le k \le 4M$)			

Таблица 4. Команды пересылки данных

Мнемоника	Описание	Операция	Флаги
MOV Rd, Rr	Пересылка между РОН	Rd = Rr	-
MOVW Rd, Rr	Пересылка 2-байтных значений	Rd+l:Rd = Rr+l:Rr	-
LDI Rd, K *	Загрузка константы в РОН	Rd = K	-
LD Rd, X	Косвенное чтение	Rd = [X]	-
LD Rd, X+	Косвенное чтение с постинкрементом	Rd = [X], X = X + l	-
LD Rd, -X	Косвенное чтение с преддекрементом	X = X - l, Rd = [X]	-
LD Rd, Y	Косвенное чтение	Rd = [Y]	-
LD Rd, Y+	Косвенное чтение с постинкрементом	Rd = [Y], Y = Y + l	-
LD Rd, -Y	Косвенное чтение с преддекрементом	Y = Y - l, Rd = [Y]	-
LDD Rd, Y+q	Косвенное относительное чтение	Rd= [Y + q]	-
LD Rd, Z	Косвенное чтение	Rd = [Z]	-
LD Rd, Z+	Косвенное чтение с постинкрементом	Rd = [Z], Z = Z +l	-
LD Rd, -Z	Косвенное чтение с преддекрементом	Z = Z - l, Rd = [Z]	-
LDD Rd, Z+q	Косвенное относительное чтение	Rd = [Z + q]	-
LDS Rd, k	Непосредственное чтение из ОЗУ	Rd = [k]	-
ST X, Rr	Косвенная запись	[X] = Rr	-
ST X+, Rr	Косвенная запись с постинкрементом	[X] = Rr, X = X + l	-
ST -X, Rr	Косвенная запись с преддекрементом	X = X - l, [X] = Rr	-
ST Y, Rr	Косвенная запись	[Y] = Rr	-
ST Y+, Rr	Косвенная запись с постинкрементом	[Y] = Rr, Y = Y + l	-
ST -Y, Rr	Косвенная запись с преддекрементом	Y = Y - l, [X] = Rr	-
STD Y+q, Rr	Косвенная относительная запись	[Y + q] = Rr	-
ST Z, Rr	Косвенная запись	[Z] = Rr	-
ST Z+, Rr	Косвенная запись с постинкрементом	[Z] = Rr, Z = Z + l	-
ST -Z, Rr	Косвенная запись с преддекрементом	Z = Z - l, [Z] = Rr	-
STD Z+q, Rr	Косвенная относительная запись	[Z + q] = Rr	-
STS k, Rr	Непосредственная запись в ОЗУ	[k] = Rr	-
LPM	Загрузка данных из памяти программ	R0 = {Z}	-
LPM Rd, Z	Загрузка данных из памяти программ	Rd = {Z}	-
LPM Rd, Z+	Загрузка данных из памяти программ	Rd = {Z}, Z = Z + 1	-
SPM	Запись в память программ	{Z} = R1:R0	-
IN Rd, P	Пересылка из РВВ в РОН	Rd = P	-
OUT P, Rr	Пересылка из РОН в РВВ	P = Rr	-
PUSH Rr	Сохранение байта в стеке	STACK = Rr	-
POP Rd	Извлечение байта из стека	Rd = STACK	-
$0 \le d \le 31$, $0 \le r \le 31$, $0 \le K \le 255$, $0 \le k \le 65535$, $0 \le q \le 63$, $0 \le P \le 63$ (* в команде LDI: $16 \le d \le 31$); [ ] – содержимое памяти данных; {} – содержимое памяти команд			

Таблица 5. Команды битовых операций

Мнемоника	Описание	Операция	Флаги
CBR Rd, K *	Сброс бита(ов) РОН	Rd = Rd • ($FF - K)	Z, N, V
SBR Rd, K *	Установка бита(ов) РОН	Rd = Rd v K	Z, N, V
CBI P, b	Сброс бита РВВ	P.b = 0	-
SBI P, b	Установка бита РВВ	P.b = 1	-
BCLR s	Сброс флага	SREG.s = 0	SREG.s
BSET s	Установка флага	SREG.s = 1	SREG.s
BLD Rd, b	Загрузка бита РОН из флага T (SREG)	Rd.b = T	-
BST Rr, b	Запись бита РОН в флаг T (SREG)	T = Rr.b	T
CLC	Сброс флага переноса	C = 0	C
SEC	Установка флага переноса	C = 1	C
CLN	Сброс флага отрицательного числа	N = 0	N
SEN	Установка флага отрицательного числа	N = 1	N
CLZ	Сброс флага нуля	Z = 0	Z
SEZ	Установка флага нуля	Z = 1	Z
CLI	Общее запрещение прерываний	I = 0	I
SEI	Общее разрешение прерываний	I = 1	I
CLS	Сброс флага знака	S = 0	S
SES	Установка флага знака	S = 1	S
CLV	Сброс флага переполнения дополнительного кода	V = 0	V
SEV	Установка флага переполнения дополнительного кода	V = l	V
CLT	Сброс флага T	T = 0	T
SET	Установка флага T	T = l	T
CLH	Сброс флага половинного переноса	H = 0	H
SEH	Установка флага половинного переноса	H = l	H
$0 \le d \le 31$, $0 \le r \le 31$, $0 \le b \le 7$, $0 \le s \le 7$, $0 \le K \le 255$ (* в командах CBR, SBR: $16 \le d \le 31$)			