گروه فنی مهندسی جوش و برش مقدم



اعتماد از شما کیفیت و تخصص از ما

 \bigcirc

09153223758 051-37581400 https://www.moghadamwelding http://instagram.com/moghadam https://t.me/moghadamwelding https://whatsapp.com/channel https://rubika.ir/moghadamwelding

مشهد خیام شمالی 63 خیابان پردیس 3

برای کسب اطلاعات بیشتر بر روی لینک ها کلیک کنید

- 7 سال سابقه آموزش تعمیرات تخصصی دستگاه های جوش اینورتری تک فاز و 3 فاز
- 7 سال سابقه فروش قطعات الكترونيكي دستگاه جوش
 تك فاز و 3 فاز
 - آموزش تخصصی تحلیل دستگاه های جوش اینورتری مختص ابراز فروشان
 - آموزش تخصصی ابراز آلات شارژی

Features

- High-performance, Low-power Atmel®AVR® 8-bit Microcontroller
- Advanced RISC Architecture
 - 131 Powerful Instructions Most Single-clock Cycle Execution
 - 32 x 8 General Purpose Working Registers
 - Fully Static Operation
 - Up to 16MIPS Throughput at 16MHz
 - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory segments
 - 32KBytes of In-System Self-programmable Flash program memory
 - 1024Bytes EEPROM
 - 2KByte Internal SRAM
 - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
 - Data retention: 20 years at 85°C/100 years at 25°C(1)
 - Optional Boot Code Section with Independent Lock Bits
 - In-System Programming by On-chip Boot Program
 - True Read-While-Write Operation
 - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 Compliant) Interface
- Boundary-scan Capabilities According to the JTAG Standard
 - Extensive On-chip Debug Support
- Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
 - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
 - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and CaptureMode
 - Real Time Counter with Separate Oscillator
 - Four PWM Channels
 - 8-channel, 10-bit ADC
 - 8 Single-ended Channels
 - 7 Differential Channels in TQFP Package Only
 - 2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
 - Byte-oriented Two-wire Serial Interface
 - Programmable Serial USART
 - Master/Slave SPI Serial Interface
 - Programmable Watchdog Timer with Separate On-chip Oscillator
 - On-chip Analog Comparator
- Special Microcontroller Features
 - Power-on Reset and Programmable Brown-out Detection
 - Internal Calibrated RC Oscillator
 - External and Internal Interrupt Sources
 - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standbyand Extende Standby
- I/O and Packages
 - 32 Programmable I/O Lines
 - 40-pin PDIP, 44-lead TQFP, and 44-pad QFN/MLF
- Operating Voltages
 - 2.7 5.5V
 - Speed Grades
 - 0 16MHz
- Power Consumption at 1 MHz, 3V, 25°C
 - Active: 0.6mA
 - Idle Mode: 0.2mA
 - Power-down Mode: < 1µA



8-bit **AVR**[®] Microcontroller with 32KBytes In-System Programmable Flash

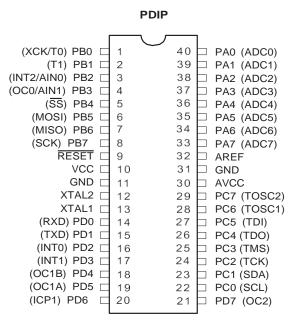
ATmega32A

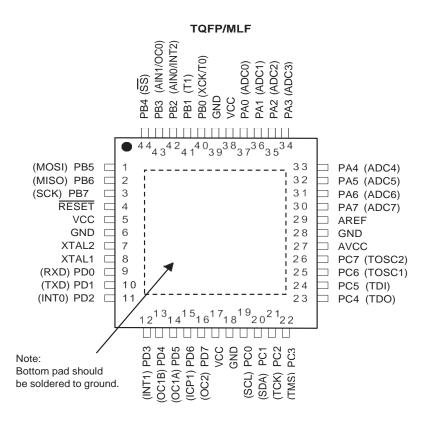
Summary



1. Pin Configurations

Figure 1-1.	Pinout ATmega32A



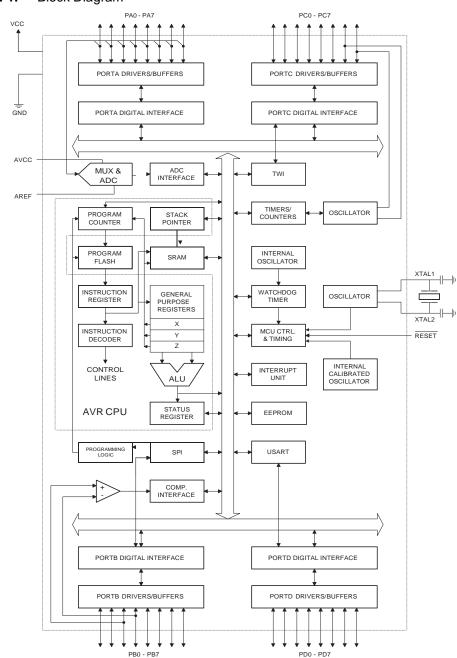




2. Overview

The Atmel[®]AVR[®] ATmega32A is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega32A achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram







The Atmel[®]AVR[®] core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega32A provides the following features: 32K bytes of In-System Programmable Flash Program memory with Read-While-Write capabilities, 1024 bytes EEPROM, 2K byte SRAM, 32 general purpose I/O lines, 32 general purpose working registers, a JTAG interface for Boundaryscan, On-chip Debugging support and programming, three flexible Timer/Counters with compare modes, Internal and External Interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain (TQFP package only), a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the USART, Two-wire interface, A/D Converter, SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next External Interrupt or Hardware Reset. In Power-save mode, the Asynchronous Timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

The device is manufactured using Atmel's high density nonvolatile memory technology. The Onchip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega32A is a powerful microcontroller that provides a highly-flexible and costeffective solution to many embedded control applications.

The ATmega32A AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

2.2 Pin Descriptions

2.2.1 VCC

Digital supply voltage.

2.2.2 GND

Ground.

2.2.3 Port A (PA7:PA0)

Port A serves as the analog inputs to the A/D Converter.

Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have sym-



metrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

2.2.4 Port B (PB7:PB0)

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B also serves the functions of various special features of the ATmega32A as listed on page 59.

2.2.5 Port C (PC7:PC0)

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs.

The TD0 pin is tri-stated unless TAP states that shift out data are entered.

Port C also serves the functions of the JTAG interface and other special features of the ATmega32A as listed on page 62.

2.2.6 Port D (PD7:PD0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port D also serves the functions of various special features of the ATmega32A as listed on page 64.

2.2.7 RESET Reset Input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in Table 27-1 on page 299. Shorter pulses are not guaranteed to generate a reset. 2.2.8 XTAL1

Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

2.2.9 XTAL2 Output from the inverting Oscillator amplifier.



2.2.10 AVCC

AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to V_{CC} , even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter.

2.2.11 AREF

AREF is the analog reference pin for the A/D Converter.

3. Resources

A comprehensive set of development tools, application notes and datasheets are available for download on http://www.atmel.com/avr.

4. Data Retention

Reliability Qualification results show that the projected data retention failure rate is much less than 1 PPM over 20 years at 85°C or 100 years at 25°C.



5. Register Summary

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
\$3F (\$5F)	SREG	I	Т	Н	S	V	N	Z	С	8
\$3E (\$5E)	SPH	_	-	_	-	SP11	SP10	SP9	SP8	11
\$3D (\$5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	11
\$3C (\$5C)	OCR0	Timer/Counter	0 Output Compai	e Register						86
\$3B (\$5B)	GICR	INT1	INT0	INT2	-	-	-	IVSEL	IVCE	48, 71
\$3A (\$5A)	GIFR	INTF1	INTF0	INTF2	-	-	-	-	-	71
\$39 (\$59)	TIMSK	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	87, 117, 136
\$38 (\$58)	TIFR	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	87, 117, 136
\$37 (\$57)	SPMCR	SPMIE	RWWSB	-	RWWSRE	BLBSET	PGWRT	PGERS	SPMEN	264
\$36 (\$56)	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE	202
\$35 (\$55)	MCUCR	SE	SM2	SM1	SM0	ISC11	ISC10	ISC01	ISC00	36, 69
\$34 (\$54)	MCUCSR	JTD	ISC2	-	JTRF	WDRF	BORF	EXTRF	PORF	42, 70, 251
\$33 (\$53)	TCCR0	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	84
\$32 (\$52)	TCNT0	Timer/Counter	, ,							86
\$31 ⁽¹⁾ (\$51) ⁽¹⁾	OSCCAL		bration Register							32
. ,	OCDR	On-Chip Debu								232
\$30 (\$50)	SFIOR	ADTS2	ADTS1	ADTS0	-	ACME	PUD	PSR2	PSR10	66,90,137,206,226
\$2F (\$4F)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	112
\$2E (\$4E)	TCCR1B	ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10	114
\$2D (\$4D)	TCNT1H		r1 – Counter Regi							116
\$2C (\$4C) \$2B (\$4B)	TCNT1L OCR1AH		r1 – Counter Regi		ah Buto					116 116
\$2B (\$4B) \$2A (\$4A)	OCR1AH OCR1AL			are Register A Hi	• •					116
\$2A (\$4A) \$29 (\$49)	OCR1AL OCR1BH			are Register A Lo are Register B Hi						116
\$29 (\$49)	OCR1BL			are Register B Lo						116
\$27 (\$47)	ICR1H			Register High By						116
\$26 (\$46)	ICR1L			Register Low By						116
\$25 (\$45)	TCCR2	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	132
\$24 (\$44)	TCNT2	Timer/Counter		001121	001120		0022	0021	0020	135
\$23 (\$43)	OCR2		r2 Output Compa	e Register						135
\$22 (\$42)	ASSR	-		_	_	AS2	TCN2UB	OCR2UB	TCR2UB	135
\$21 (\$41)	WDTCR	-	-	-	WDTOE	WDE	WDP2	WDP1	WDP0	43
	UBRRH	URSEL	-	-	-		UBR	R[11:8]		171
\$20 ⁽²⁾ (\$40) ⁽²⁾	UCSRC	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL	170
\$1F (\$3F)	EEARH	-	-	-	-	-	-	EEAR9	EEAR8	20
\$1E (\$3E)	EEARL	EEPROM Add	Iress Register Lov	v Byte						20
\$1D (\$3D)	EEDR	EEPROM Data	a Register							21
\$1C (\$3C)	EECR	-	-	-	-	EERIE	EEMWE	EEWE	EERE	21
\$1B (\$3B)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	66
\$1A (\$3A)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	66
\$19 (\$39)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	66
\$18 (\$38)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	67
\$17 (\$37)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	67
\$16 (\$36)	PINB	PINB7 PORTC7	PINB6 PORTC6	PINB5	PINB4	PINB3	PINB2 PORTC2	PINB1 PORTC1	PINB0	67
\$15 (\$35)	PORTC	I PORICZ		PORTC5	PORTC4	PORTC3		PORICI	PORTC0	67
\$14 (\$34) \$13 (\$33)	DDDO				DDO1				DDOO	67
, ,	DDRC	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	67
	PINC	DDC7 PINC7	DDC6 PINC6	DDC5 PINC5	PINC4	DDC3 PINC3	DDC2 PINC2	DDC1 PINC1	PINC0	67
\$12 (\$32) \$11 (\$31)	PINC PORTD	DDC7 PINC7 PORTD7	DDC6 PINC6 PORTD6	DDC5 PINC5 PORTD5	PINC4 PORTD4	DDC3 PINC3 PORTD3	DDC2 PINC2 PORTD2	DDC1 PINC1 PORTD1	PINC0 PORTD0	67 67
\$11 (\$31)	PINC PORTD DDRD	DDC7 PINC7 PORTD7 DDD7	DDC6 PINC6 PORTD6 DDD6	DDC5 PINC5 PORTD5 DDD5	PINC4 PORTD4 DDD4	DDC3 PINC3 PORTD3 DDD3	DDC2 PINC2 PORTD2 DDD2	DDC1 PINC1 PORTD1 DDD1	PINC0 PORTD0 DDD0	67 67 67
\$11 (\$31) \$10 (\$30)	PINC PORTD DDRD PIND	DDC7 PINC7 PORTD7 DDD7 PIND7	DDC6 PINC6 PORTD6 DDD6 PIND6	DDC5 PINC5 PORTD5	PINC4 PORTD4	DDC3 PINC3 PORTD3	DDC2 PINC2 PORTD2	DDC1 PINC1 PORTD1	PINC0 PORTD0	67 67 67 68
\$11 (\$31) \$10 (\$30) \$0F (\$2F)	PINC PORTD DDRD PIND SPDR	DDC7 PINC7 PORTD7 DDD7 PIND7 SPI Data Reg	DDC6 PINC6 PORTD6 DDD6 PIND6 ister	DDC5 PINC5 PORTD5 DDD5	PINC4 PORTD4 DDD4	DDC3 PINC3 PORTD3 DDD3	DDC2 PINC2 PORTD2 DDD2	DDC1 PINC1 PORTD1 DDD1	PINC0 PORTD0 DDD0 PIND0	67 67 67 68 145
\$11 (\$31) \$10 (\$30) \$0F (\$2F) \$0E (\$2E)	PINC PORTD DDRD PIND SPDR SPSR	DDC7 PINC7 PORTD7 DDD7 PIND7	DDC6 PINC6 PORTD6 DDD6 PIND6	DDC5 PINC5 PORTD5 DDD5	PINC4 PORTD4 DDD4 PIND4	DDC3 PINC3 PORTD3 DDD3	DDC2 PINC2 PORTD2 DDD2 PIND2 -	DDC1 PINC1 PORTD1 DDD1 PIND1	PINC0 PORTD0 DDD0 PIND0 SPI2X	67 67 68 145 145
\$11 (\$31) \$10 (\$30) \$0F (\$2F) \$0E (\$2E) \$0D (\$2D)	PINC PORTD DDRD PIND SPDR SPSR SPCR	DDC7 PINC7 PORTD7 DDD7 PIND7 SPI Data Reg SPIF SPIE	DDC6 PINC6 PORTD6 DDD6 PIND6 ister WCOL SPE	DDC5 PINC5 PORTD5 DDD5 PIND5 -	PINC4 PORTD4 DDD4	DDC3 PINC3 PORTD3 DDD3 PIND3	DDC2 PINC2 PORTD2 DDD2	DDC1 PINC1 PORTD1 DDD1	PINC0 PORTD0 DDD0 PIND0	67 67 68 145 145 143
\$11 (\$31) \$10 (\$30) \$0F (\$2F) \$0E (\$2E)	PINC PORTD DDRD PIND SPDR SPSR	DDC7 PINC7 PORTD7 DDD7 PIND7 SPI Data Reg SPIF	DDC6 PINC6 PORTD6 DDD6 PIND6 ister WCOL SPE	DDC5 PINC5 PORTD5 DDD5 PIND5 -	PINC4 PORTD4 DDD4 PIND4	DDC3 PINC3 PORTD3 DDD3 PIND3	DDC2 PINC2 PORTD2 DDD2 PIND2 -	DDC1 PINC1 PORTD1 DDD1 PIND1	PINC0 PORTD0 DDD0 PIND0 SPI2X	67 67 68 145 145
\$11 (\$31) \$10 (\$30) \$0F (\$2F) \$0E (\$2E) \$0D (\$2D) \$0C (\$2C)	PINC PORTD DDRD PIND SPDR SPSR SPCR UDR	DDC7 PINC7 PORTD7 DDD7 PIND7 SPI Data Reg SPIF SPIE USART I/O D	DDC6 PINC6 PORTD6 DDD6 PIND6 ister WCOL SPE ata Register	DDC5 PINC5 PORTD5 DDD5 PIND5 - DORD	PINC4 PORTD4 DDD4 PIND4 - MSTR	DDC3 PINC3 PORTD3 DDD3 PIND3 - CPOL	DDC2 PINC2 PORTD2 DDD2 PIND2 - CPHA	DDC1 PINC1 PORTD1 DDD1 PIND1 - SPR1	PINC0 PORTD0 DDD0 PIND0 SPI2X SPR0	67 67 68 145 145 143 167
\$11 (\$31) \$10 (\$30) \$0F (\$2F) \$0E (\$2E) \$0D (\$2D) \$0C (\$2C) \$0B (\$2B)	PINC PORTD DDRD PIND SPDR SPSR SPCR UDR UCSRA	DDC7 PINC7 DDD7 PIND7 SPI Data Reg SPIF SPIE USART I/O D RXC RXCIE	DDC6 PINC6 PORTD6 DDD6 PIND6 ister WCOL SPE ata Register TXC	DDC5 PINC5 PORTD5 DDD5 PIND5 - DORD UDRE UDRE UDRIE	PINC4 PORTD4 DDD4 PIND4 - MSTR FE	DDC3 PINC3 PORTD3 DDD3 PIND3 - CPOL DOR	DDC2 PINC2 PORTD2 DDD2 PIND2 - CPHA PE	DDC1 PINC1 PORTD1 DDD1 PIND1 - SPR1 U2X	PINC0 PORTD0 DDD0 PIND0 SPI2X SPR0 MPCM	67 67 68 145 145 143 167 168
\$11 (\$31) \$10 (\$30) \$0F (\$2F) \$0E (\$2E) \$0D (\$2D) \$0C (\$2C) \$0B (\$2B) \$0A (\$2A)	PINC PORTD DDRD PIND SPDR SPSR SPCR UDR UCSRA UCSRB	DDC7 PINC7 DDD7 PIND7 SPI Data Reg SPIF SPIE USART I/O D RXC RXCIE	DDC6 PINC6 PORTD6 DDD6 PIND6 ister WCOL SPE ata Register TXC TXCIE	DDC5 PINC5 PORTD5 DDD5 PIND5 - DORD UDRE UDRE UDRIE	PINC4 PORTD4 DDD4 PIND4 - MSTR FE	DDC3 PINC3 PORTD3 DDD3 PIND3 - CPOL DOR	DDC2 PINC2 PORTD2 DDD2 PIND2 - CPHA PE	DDC1 PINC1 PORTD1 DDD1 PIND1 - SPR1 U2X	PINC0 PORTD0 DDD0 PIND0 SPI2X SPR0 MPCM	67 67 68 145 145 143 167 168 169
\$11 (\$31) \$10 (\$30) \$0F (\$2F) \$0E (\$2E) \$0D (\$2D) \$0C (\$2C) \$0B (\$2B) \$0A (\$2A) \$09 (\$29)	PINC PORTD DDRD PIND SPDR SPSR SPCR UDR UCSRA UCSRB UBRRL	DDC7 PINC7 DDD7 PIND7 SPI Data Reg SPIF SPIE USART I/O D RXC RXCIE USART Baud	DDC6 PINC6 DDD6 PIND6 ister WCOL SPE ata Register TXC TXCIE Rate Register LC	DDC5 PINC5 DDD5 PIND5 DORD UDRE UDRE UDRIE w Byte	PINC4 PORTD4 DDD4 PIND4 - MSTR FE RXEN	DDC3 PINC3 PORTD3 DDD3 PIND3 - CPOL DOR TXEN	DDC2 PINC2 PORTD2 DDD2 PIND2 - CPHA PE UCSZ2	DDC1 PINC1 PORTD1 DDD1 PIND1 - SPR1 U2X RXB8	PINC0 PORTD0 DDD0 PIND0 SPI2X SPR0 MPCM TXB8	67 67 68 145 145 143 167 168 169 171
\$11 (\$31) \$10 (\$30) \$0F (\$2F) \$0E (\$2E) \$0D (\$2D) \$0C (\$2C) \$0B (\$2B) \$0A (\$2A) \$09 (\$29) \$08 (\$28)	PINC PORTD DDRD PIND SPDR SPSR SPCR UDR UCSRA UCSRB UBRRL ACSR	DDC7 PINC7 DDD7 PIND7 SPI Data Reg SPIF SPIE USART I/O D RXC RXCIE USART Baud ACD	DDC6 PINC6 PORTD6 DDD6 PIND6 ister WCOL SPE ata Register TXC TXCIE Rate Register LC ACBG	DDC5 PINC5 DDD5 PIND5 DORD DORD UDRE UDRE UDRIE w Byte ACO	PINC4 PORTD4 DDD4 PIND4 - MSTR FE RXEN ACI	DDC3 PINC3 PORTD3 DDD3 PIND3 - CPOL DOR TXEN ACIE	DDC2 PINC2 PORTD2 DDD2 PIND2 - CPHA PE UCSZ2 ACIC	DDC1 PINC1 PORTD1 DDD1 PIND1 - SPR1 U2X RXB8 ACIS1	PINC0 PORTD0 DDD0 PIND0 SPI2X SPR0 MPCM TXB8 ACIS0	67 67 68 145 145 143 167 168 169 171 206
\$11 (\$31) \$10 (\$30) \$0F (\$2F) \$0E (\$2E) \$0D (\$2D) \$0C (\$2C) \$0B (\$2B) \$0A (\$2A) \$09 (\$29) \$08 (\$28) \$07 (\$27)	PINC PORTD DDRD PIND SPDR SPSR SPCR UDR UCSRA UCSRA UCSRB UBRRL ACSR ADMUX	DDC7 PINC7 PORTD7 DDD7 PIND7 SPI Data Reg SPIF SPIE USART I/O D RXC RXCIE USART Baud ACD REFS1 ADEN	DDC6 PINC6 PORTD6 DDD6 PIND6 ister WCOL SPE ata Register TXC TXCIE Rate Register LC ACBG REFS0	DDC5 PINC5 PORTD5 DDD5 PIND5 - DORD UDRE UDRE UDRIE w Byte ACO ADLAR	PINC4 PORTD4 DDD4 PIND4 - MSTR FE RXEN ACI MUX4	DDC3 PINC3 PORTD3 DDD3 PIND3 - CPOL DOR TXEN ACIE MUX3	DDC2 PINC2 PORTD2 DDD2 PIND2 - CPHA PE UCSZ2 ACIC MUX2	DDC1 PINC1 PORTD1 DDD1 PIND1 - SPR1 U2X RXB8 ACIS1 MUX1	PINC0 PORTD0 DDD0 PIND0 SPI2X SPR0 MPCM TXB8 ACIS0 MUX0	67 67 68 145 145 143 167 168 169 171 206 222
\$11 (\$31) \$10 (\$30) \$0F (\$2F) \$0E (\$2E) \$0D (\$2D) \$0C (\$2C) \$0B (\$2B) \$0A (\$2A) \$09 (\$29) \$08 (\$28) \$07 (\$27) \$06 (\$26)	PINC PORTD DDRD PIND SPDR SPSR SPCR UDR UCSRA UCSRA UCSRB UBRRL ACSR ADMUX ADCSRA	DDC7 PINC7 DDD7 PIND7 SPI Data Reg SPIF SPIE USART I/O D RXC RXCIE USART Baud ACD REFS1 ADEN ADC Data Reg	DDC6 PINC6 PORTD6 DDD6 PIND6 ister WCOL SPE ata Register TXC TXCIE Rate Register LC ACBG REFS0 ADSC	DDC5 PINC5 PORTD5 DDD5 PIND5 - DORD UDRE UDRE UDRIE w Byte ACO ADLAR	PINC4 PORTD4 DDD4 PIND4 - MSTR FE RXEN ACI MUX4	DDC3 PINC3 PORTD3 DDD3 PIND3 - CPOL DOR TXEN ACIE MUX3	DDC2 PINC2 PORTD2 DDD2 PIND2 - CPHA PE UCSZ2 ACIC MUX2	DDC1 PINC1 PORTD1 DDD1 PIND1 - SPR1 U2X RXB8 ACIS1 MUX1	PINC0 PORTD0 DDD0 PIND0 SPI2X SPR0 MPCM TXB8 ACIS0 MUX0	67 67 67 68 145 145 145 143 167 168 169 171 206 222 224
\$11 (\$31) \$10 (\$30) \$0F (\$2F) \$0D (\$2D) \$0C (\$2C) \$0B (\$2B) \$0A (\$2A) \$09 (\$29) \$08 (\$28) \$07 (\$27) \$06 (\$26) \$05 (\$25)	PINC PORTD DDRD PIND SPDR SPSR SPCR UDR UCSRA UCSRA UCSRB UBRRL ACSR ADMUX ADCSRA ADCH	DDC7 PINC7 DDD7 PIND7 SPI Data Reg SPIF USART I/O D RXC RXCIE USART Baud ACD REFS1 ADEN ADC Data Reg ADC Data Reg	DDC6 PINC6 PORTD6 DDD6 PIND6 ister WCOL SPE ata Register TXC TXCIE Rate Register Loc ACBG REFS0 ADSC gister High Byte	DDC5 PINC5 PORTD5 DDD5 PIND5 - DORD UDRE UDRIE w Byte ACO ADLAR ADATE	PINC4 PORTD4 DDD4 PIND4 - MSTR FE RXEN ACI MUX4	DDC3 PINC3 PORTD3 DDD3 PIND3 - CPOL DOR TXEN ACIE MUX3	DDC2 PINC2 PORTD2 DDD2 PIND2 - CPHA PE UCSZ2 ACIC MUX2	DDC1 PINC1 PORTD1 DDD1 PIND1 - SPR1 U2X RXB8 ACIS1 MUX1	PINC0 PORTD0 DDD0 PIND0 SPI2X SPR0 MPCM TXB8 ACIS0 MUX0	67 67 67 68 145 145 145 143 167 168 169 171 206 222 224 224 225



Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
\$01 (\$21)	TWSR	TWS7	TWS6	TWS5	TWS4	TWS3	-	TWPS1	TWPS0	203
\$00 (\$20)	TWBR	Two-wire Serial Interface Bit Rate Register							201	

Notes: 1. When the OCDEN Fuse is unprogrammed, the OSCCAL Register is always accessed on this address. Refer to the debugger specific documentation for details on how to use the OCDR Register.

2. Refer to the USART description for details on how to access UBRRH and UCSRC.

- 3. For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.
- 4. Some of the Status Flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O Register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers \$00 to \$1F only.



6. Instruction Set Summary

ARTHMEND USCH INTERVIEW Vertical Section Rel = Mar NC Z.X.V.M Section ADD Rel RD Add the Registres Rel = Mar NC Z.X.V.M 1 ADC Rel RD Add the Registres Rel = Mar NC Z.X.V.M 1 ADD Rel RD Schere The Registres Rel = Mar NC Z.X.V.M 1 Stable Rel RD Schere The Registres Rel = Mar NC Z.X.V.V.M 1 Stable Rel RD Schere The Registres Rel = Mar NC Z.X.V.V.M 1 Stable Rel RD Schere The Registres Rel = Mar NC Z.X.V.V.M 1 Stable Rel RD Schere The Registres Rel RD Z.X.V.V.M 1 Stable Rel RD Schere The Registres Rel RD Z.X.V.V.M 1 OR Rel RD Schere The Rel RD Z.X.V.V.M 1 CR Rel RD Schere The Rel RE Z.X.V.M 1 CR Rel RD Schere The Rel RE Z.X.V.M 1 CR <th>Mnemonics</th> <th>Operands</th> <th>Description</th> <th>Operation</th> <th>Flags</th> <th>#Clocks</th>	Mnemonics	Operands	Description	Operation	Flags	#Clocks
ADCRd, Fr.Ads incrigence NursiaRd = RA HA = C.CANNASASUBRd, KNSubtrace Constant from RegisterRd = Rd = Rd = Rd = Rd = C.CANNA1SUGRd, KNSubtrace Constant from RegisterRd = Rd	ARITHMETIC AND L	OGIC INSTRUCTIONS				
ADCRd, Fr.Ads incrigence NursiaRd = RA HA = C.CANNASASUBRd, KNSubtrace Constant from RegisterRd = Rd = Rd = Rd = Rd = C.CANNA1SUGRd, KNSubtrace Constant from RegisterRd = Rd	ADD	Rd, Rr	Add two Registers	$Rd \leftarrow Rd + Rr$	Z,C,N,V,H	1
ADNRed.KAdd humorabo spagnenRed.s. Au.Bubbl HZCANUS1SUBBRd, KrSubhrat vo SoppenRed.s. Au.C.ZCANUH1SUBBRd, KrSubhrat vo SoppenRed.s. Au.C.ZCANUH1SUBCRd, KrSubhrat vo Cargo trons RegionRed.s. Au.C.ZCANUH1SBCRd, KrSubhrat vo Cargo trons Reg.Red.s. Au.C.ZCANUH1SBCRd, KrSubhrat vo Cargo trons Reg.Red.s. Rd. K.ZCANUH1SBCRd, KrLogial ADA Region rot CarabartRed.s. Rd. KLZCANUH1ORRd, KrLogial ADA Region rot CarabartRed.s. Rd. KLZNV1ORRd, KrLogial ADA Region rot CarabartRed.s. Rd. KLZNV1DRRd, KrLogial ADA Region rot CarabartRed.s. Rd. KLZNV1EDGRd, KrSeptem rot CarabartRed.s. Rd. KLZNV1EDGRdDorava CarabartRed.s. Rd. KLZNV1EDGRdDorava CarabartRed.s. Rd. KLZNV1EDGRdDorava CarabartRed.s. Rd. KLZNV1EDGRdDorava CarabartRd. Rd. Rd. Rd. Rd. ZNV1SRRd.Set Bigin fraginerRd. Rd. Rd. Rd. Rd. ZNV1EDGRdDorava CarabartRd. Rd. Rd. Rd. Rd. Rd. ZNV1SRRd.Set Bigin fraginerRd. Rd. Rd. Rd. Rd. Rd. Rd. Rd. Rd. ZNV1SRRd.Set Bigin fra	ADC	Rd, Rr	Add with Carry two Registers	1		1
BJBI R4, R Submrd with Carry be Registern R4 - R0 + K - C ZCR.VH 1 SBC R4, R Submrd with Carry Constant from Reg. R4 - R0 + K - C ZCR.VH 1 SBC R4, K Submrd with Carry Constant from Reg. R4 - R0 + K - C ZCR.VH 1 SBC R4, K Submrd with Carry Constant from Reg. R4 - R0 + K - C ZCR.VH 1 AND R4, K Logical AND Registers and Constant R4 - R0 + K - C ZCR.VH 1 CR R4, K Logical CR Registers and Constant R4 - R0 + K - C ZNV 1 CR R4, K Logical CR Register and Constant R4 - R0 + K - C ZNV 1 CR R4 Constant Register and Constant Register R4 - R0 + R1 ZNV 1 CR R4 Constant Register and Constant Register R4 - R0 + R1 ZNV 1 CR R4 Descreter R4 - R0 + R1 R4 - R0 + R1 ZNV 1 CR R4 R4 - R0 + R1 R4 - R0 + R1 ZNV 1 <td>ADIW</td> <td>Rdl,K</td> <td>Add Immediate to Word</td> <td>$Rdh:Rdl \leftarrow Rdh:Rdl + K$</td> <td>Z,C,N,V,S</td> <td>2</td>	ADIW	Rdl,K	Add Immediate to Word	$Rdh:Rdl \leftarrow Rdh:Rdl + K$	Z,C,N,V,S	2
SBC R.M. Bubaras with Carry non Regulary. R.J. R.J. C.C. Z.C.N.V.H 1 SBCI R.M.K Bubaras with Carry Constantion Regulary. R.J. R.R.+CC. Z.C.N.V.H 1 SBNV R.B.R Logical NAD Register and Constant R.d. R.R.Fr. Z.N.V.V 1 AND R.B. R Logical NAD Register and Constant R.d. R.R.Fr. Z.N.V.V 1 CAR B.B., R Logical OR Register and Constant R.d. R.R.Fr. Z.N.V.V 1 CAR B.B., R Logical OR Register and Constant R.d. Rel N Z.N.V.V 1 COM R.G. C.R.S.L.G.R.R.R.R.R.R.R.R.R.R.R.R.R.R.R.R.R.R	SUB	Rd, Rr	Subtract two Registers	$Rd \leftarrow Rd - Rr$	Z,C,N,V,H	1
SBC Rds // Subtract immediator tom Word Rd+Rd+ RoHkaH / ZCAN /S 2 AND Rds // Logical AND Register and Constant Rd+Rd+ RoHkaH / ZNN // 1 AND Rds // Logical AND Register and Constant Rd+-Rd+ Rd+ ZNN // 1 OR Rds // Logical AND Register and Constant Rd+-Rd+ Rd+ ZNN // 1 OR Rds // Logical OR Register and Constant Rd+-Rd+ Rd+ ZNN // 1 COM Rds // Exclusive OR Register and Constant Rd+-Rd+ Rd+ ZNN // 1 ECR Rds // Exclusive OR Register and Constant Rd+-Rd+ Rd+ ZNN // 1 Rds Gas Enterginergister and Constant Rd+-Rd+ Rd+ ZNN // 1 DEC Rd Decement Rd+-Rd+ Rd -1 ZNN // 1 DEC Rd Decement Rd+-Rd+Rd -1 ZNN // 1 DEC Rd Decement Rd+-Rd+Rd -1 ZNN // 1 DEC Rd Decement Rd+	SUBI	Rd, K	Subtract Constant from Register	$Rd \leftarrow Rd - K$	Z,C,N,V,H	1
SNVRd.KSubscriptionRdR.BR. Hor.BK.Z.R.V.S22ANDRd. R.Logial AND Register and ConstantRdR.B.K.Z.N.V1ANDRd.KLogial AND Register and ConstantRdR.B.K.Z.N.V1ORRd.KLogial OR Register and ConstantRdR.B.K.Z.N.V1ORIRd.KLogial OR Register and ConstantRdR.B.V.K.Z.N.V1ORIRd.KLogial OR Register and ConstantRdR.B.V.K.Z.N.V1COMRdOne ComplementRd.+S.F.R.B.Z.N.V.V1COMRdOne ComplementRd.+S.F.R.B.Z.N.V.V1SRRd.KSet Rish In RegisterRd.+R.B.F.H.Z.N.V.V1DRRd.KDesementRd.+R.B.F.H.Z.N.V.V1INCRdDesementRd.+R.B.F.H.Z.N.V.V1RdDesementRd.+R.B.F.H.Z.N.V.V1RdRd.DesementRd.+R.B.F.H.Z.N.V.V1RdDesementRd.+R.B.F.H.Z.N.V.V1RdDesementRd.+R.B.F.H.Z.N.V.V1RdDesementRd.+R.B.F.H.Z.N.V.V.1RdDesementRd.+R.B.F.H.Z.N.V.V.1RdDesementRdR.B.F.H.Z.N.V.V.1RdDesementRdR.B.F.H.Z.N.V.V.1RdDesementRdR.B.F.H.Z.N.V.V.1RdDesementRd.F.R.F.R	SBC	Rd, Rr	Subtract with Carry two Registers	$Rd \leftarrow Rd - Rr - C$	Z,C,N,V,H	1
AND Rs.K Logial AND Regulars and Constant Rd - Rd + K Z.N.V 1 OR Rd, K' Logial CN Registers and Constant Rd - Rd + K Z.N.V 1 OR Rd, K' Logial CN Registers and Constant Rd - Rd + K Z.N.V 1 ECR Rd, R' Exclusive OR Registers Rd - Rd + K Z.N.V 1 ECR Rd, R' Exclusive OR Registers Rd - Rd + Rd + N Z.N.V 1 RGR Rd Constructure Rd - Rd + Rd + N Z.N.V 1 NEG Rd Constructure Rd + Rd + Rd Z.N.V 1 DEC Rd Constructure Rd + Rd + Rd Z.N.V 1 DEC Rd Decrement Rd + Rd + Rd Z.N.V 1 DEC Rd Decrement Rd + Rd + Rd Z.N.V 1 DLL Rd Rd Rd Rd Multiply Grad orthuns Rd + Rd + Rd Z.N.V 1 DLL Rd Rd Rd Rd Rd Multiply Grad orthuns Rd + Rd + Rd Z.N.V 1 DLL Rd Rd Rd Multiply Grad orthuns	SBCI	Rd, K	Subtract with Carry Constant from Reg.	$Rd \leftarrow Rd - K - C$	Z,C,N,V,H	1
AND Rat. R Logical AND Regular and Constant Rd – Rd vR Z.N.V 11 OR Rd, R Logical OR Register and Constant Rd – Rd vR Z.N.V 11 ORI Rd, R Exclusive OR Register and Constant Rd – Rd vR Z.N.V 11 COM Rd, Oren Scoregement Rd – Rd vR Z.N.V 11 COM Rd Oren Scoregement Rd – SPF – Rd Z.N.V 11 SR Statistic St	SBIW	Rdl,K	Subtract Immediate from Word	Rdh:Rdl ← Rdh:Rdl - K	Z,C,N,V,S	2
OR Rs.K Logical OR Register and Constant Rd – Rd v Kn ZAV 1 DOR Rs.K.Y Excluse OR Register and Constant Rd – Rd v K ZAV 1 EOR Rs.A.Y Excluse OR Register and Constant Rd – Rd v K ZAV 1 COM Rd Onvis Complement Rd – Rd v K ZAV 1 NEG Rd K Sea Biblijn Register Rd – Rd v K ZAV 1 CBR Rd K Class Register Rd – Rd + G ZAV 1 DEC Rd Decomment Rd – Rd + Rd ZAV 1 DEC Rd Decomment Rd – Rd + Rd ZAV 1 DER Rd R Decomment Rd – Rd + Rd ZAV 1 DER Rd R Decomment Rd – Rd + Rd ZAV 1 DER Rd R Mailysiggerd on Multa Rd – Rd + Rd ZAV 2 DRLS Rd RT Mailysiggerd with Unsigned Rt 100 – Rd x Rd <11	AND	Rd, Rr	Logical AND Registers	$Rd \leftarrow Rd \bullet Rr$	Z,N,V	1
ORI Rd. R Logical OR Register and Constant Rd – Rd vK Z.N.V 1 EOR Rd. R Exclusive OR Registers Rd – Rd VK Z.N.V 1 COM Rd One Complement Rd – SPT – Rd Z.N.V 1 SR Rd K Set Birglin Register Rd – Rd vK Z.N.V 1 SR Rd K Gener Singlin Register Rd – Rd vK Z.N.V 1 SR Rd K Gener Singlin Register Rd – Rd vK Z.N.V 1 DCG Rd Incorment Rd – Rd vK Z.N.V 1 DCG Rd Clear Register Rd – Rd vK Z.N.V 1 SER Rd Gener Register Rd – Rd vR Z.N.V 1 SER Rd Set Register Rd - Rd vR Z.N.V 1 SER Rd Multiply Usingend Rt 180 – Rd vR Z.C 2 MULS Rd, RT Multiply Generation Multiply Usingend Rt 180 – Rd vR rd Z.C 2	ANDI	Rd, K	Logical AND Register and Constant	$Rd \leftarrow Rd \bullet K$	Z,N,V	1
FOR Rd. Rd. Rd. Rd. Rd. Rd. Rd. No. 1 NGO Rd One's Complement Rd.+S00Rd 2.GN.V 1 NGR Rd.K Set Brigh in Register Rd.+A Kd. 2.GN.V.H 1 SRR Rd.K Complex Bright in Register Rd.+A Rd.+G.F.A.V. 2.N.V. 1 CBR Rd.K Complex Bright in Register Rd.+A Rd.+G.F.A.V. 2.N.V. 1 DEG Rd Decement Rd.+A Rd.+B.R. 2.N.V. 1 CLR Rd Come Register Rd.+A Rd.+B.R. 2.N.V. 1 SER Rd Set Register Rd.+S.R. 2.D.V. 1 NULS Rd.Rr Multry Signed R180R9.RR 2.C. 2 2 NULS Rd.Rr Fractoral Multry Signed with Unsigned R180R18.RN 2.C. 2 2 FMLL Rd.Rr Fractoral Multry Signed with Unsigned R18.RD - R18.RN 1 2 2 2 <t< td=""><td>OR</td><td>Rd, Rr</td><td>Logical OR Registers</td><td>$Rd \leftarrow Rd v Rr$</td><td>Z,N,V</td><td>1</td></t<>	OR	Rd, Rr	Logical OR Registers	$Rd \leftarrow Rd v Rr$	Z,N,V	1
COM Rd Ones Complement Rd + SFF - Rd Z.D.N.V 1 SBE Rd K Set Big)in Register Rd + Rd × K Z.N.V 1 SBR RdK Set Big)in Register Rd + Rd × K Z.N.V 1 CR Rd K Class Biglin Register Rd + Rd × K Z.N.V 1 DEC Rd Class Biglin Register Rd + Rd × K Z.N.V 1 DEC Rd Determent Rd + Rd × Rd Z.N.V 1 DEC Rd Comment Rd + Rd × Rd Z.N.V 1 DEC Rd Comment Rd + Rd × Rd Z.N.V 1 DEC Rd Comment Rd + Rd × Rd Z.N.V 1 DEC Rd Nd register Rd + Rd × Rd Z.C 2 MULS Rd, Rr Minifely Signed R1 × Ro + Rd × Rd Z.C 2 PULL Rd, Rr Fractional Mulphy Signed with Unsigned R1 × Ro + Rd × Rd Z.C 2 PULLSU	ORI	Rd, K	Logical OR Register and Constant	$Rd \leftarrow Rd \vee K$	Z,N,V	1
NEG Rd // Two Complement Rd + 603 + K Z.C.N.VH 1 CBR Rd K Care Bridy in Register Rd + 603 + K Z.N.V 1 CBR Rd K Care Bridy in Register Rd + 603 + K Z.N.V 1 INC Rd Increment Rd + 603 + K Z.N.V 1 INC Rd Decement Rd + 603 + K Z.N.V 1 TST Rd Care Register Rd - 603 + Rd Z.N.V 1 SER Rd Set Register Rd - 603 + Rd Z.N.V 1 SER Rd Set Register Rd - 603 + Rd Z.C. 2 MULS Rd, Rr Multply Unsigned R1180 - 603 rR Z.C. 2 MULSU Rd, Rr Fractional Multply Signed with Unsigned R1180 - (Rd RR) (x Z.C. 2 FMULSU Rd, Rr Fractional Multply Signed with Unsigned R1180 - (Rd RR) (x<1	EOR	Rd, Rr	Exclusive OR Registers	$Rd \leftarrow Rd \oplus Rr$	Z,N,V	1
SBR RdK Set Bitty in Register Rd - Rd v K ZNV 1 CBR RdK Clear Bitly in Register Rd - Rd + Rd + I ZNV 1 INC Rd Decrement Rd - Rd + Rd + I ZNV 1 INC Rd Decrement Rd - Rd + Rd + Rd ZNV 1 DEC Rd Decrement Rd - Rd + Rd + Rd ZNV 1 DEC Rd Set Register Rd - Rd +	COM	Rd	One's Complement	Rd ← \$FF – Rd	Z,C,N,V	1
COBR RXK Close Bir(s):n Register Rd - Ed + EF + (s) Z.N.V 11 DEC Rd Increment Rd - Ed + I Z.N.V 11 DEC Rd Test for Zero or Minus Rd - Ed + I Z.N.V 11 DET Rd Test for Zero or Minus Rd - Rd - Rd - Rd - Z.N.V 11 SER Rd Ges Register Rd - AG + Rd - Rd - Z.N.V 11 SER Rd Ser Register Rd - SF P None 11 MULS Rd, Rr Multply Unaigned RTR 0 - Rd x Rr Z.C 2 MULS Rd, Rr Finctional Multply Unaigned RTR 0 - Rd x Rr Z.C 2 MULS Rd, Rr Finctional Multply Signed RTR 0 - Rd x Rr Z.C 2 FMULS Rd, Rr Finctional Multply Signed RTR 0 - Rd x Rr Z.C 2 FMULS Rd, Rr Finctional Multply Signed with Unaigned RTR 0 - Rd x Rr <	NEG	Rd	Two's Complement	Rd ← \$00 – Rd	Z,C,N,V,H	1
INC Rd Increment Rd - Rd -1 ZNV 11 DEC Rd Descrement Rd + Rd -1 ZNV 1 TST Rd Test for Zero or Minus Rd - Rd + Rd - Rd ZNV 1 CLR Rd Cear Register Rd - Rd + Rd - Rd ZNV 1 SR Rd Sar Register Rd - Rd + Rd + Rd None 1 MUL Rd, Rr Multply Usigned Rt Ro + Rd x Rr Z.C 2 MULS Rd, Rr Multply Signed with Unsigned Rt Ro + Rd x Rr Z.C 2 FMUL Rd, Rr Fractional Multply Signed Rt Ro + Rd x Rr Z.C 2 FMULS Rd, Rr Fractional Multply Signed Rt Ro + Rd x Rr Z.C 2 FMULS Rd, Rr Redore Jump PC + PC + k + 1 None 2 IMP k Redore Jump (Z) PC + PC + k + 1 None 3 ICALL k Redore Jump (Z) PC + C + k + 1 None 3 <tr< td=""><td>SBR</td><td>Rd,K</td><td>Set Bit(s) in Register</td><td>$Rd \leftarrow Rd \vee K$</td><td>Z,N,V</td><td>1</td></tr<>	SBR	Rd,K	Set Bit(s) in Register	$Rd \leftarrow Rd \vee K$	Z,N,V	1
DEC Rd Decrement Rd + Rd + Rd ZNV 1 TST Rd Test for Zero or Minus Rd + Rd + Rd ZNV 1 C.R. Rd Caur Register Rd + Rd + Rd Rd ZNV 1 SER Rd Set Register Rd + Rd / Rd ZNV 1 MUL Rd, R Multiply Unsigned R1:R0 + Rd x Rr Z.C 2 MULS Rd, Rr Multiply Signed with Unsigned R1:R0 + Rd x Rr Z.C 2 FMUL Rd, Rr Fractional Multiply Unsigned R1:R0 + Rd x Rr <-1	CBR	Rd,K	Clear Bit(s) in Register	$Rd \leftarrow Rd \bullet (\$FF - K)$	Z,N,V	1
TST Rd Test for Zero or Munus Rd + Rd Rd Z,N/V 1 CLR Rd Case Register Rd + Rd Rd Z,N/V 11 CLR Rd Set Register Rd + Rd Rd Z,N/V 11 MUL Rd, Rr Muliply Unsigned R1 R0 + Rd x Rr Z,C 2 MULS Rd, Rr Muliply Signed With Unsigned R1 R0 + Rd x Rr Z,C 2 MULS Rd, Rr Fractional Muliply Signed with Unsigned R1 R0 + (Rd x R) <<1	INC	Rd	Increment	$Rd \leftarrow Rd + 1$	Z,N,V	1
CLR Rd Clear Register Rd + AB Rd ZNV 1 SER Rd Ste Register Rd - SFF None 1 SER Rd, R Muliply Unsigned R1R0 - Rd x Rr Z.C 2 MULS Rd, Rr Muliply Signed with Unsigned R1R0 - Rd x Rr Z.C 2 MULS Rd, Rr Fractional Muliply Signed with Unsigned R1R0 - Rd x Rr Z.C 2 FMULS Rd, Rr Fractional Multiply Signed with Unsigned R1R0 - (Rd x Rr) <<1	DEC	Rd	Decrement	$Rd \leftarrow Rd - 1$	Z,N,V	1
SER Rd Sel Register Rd - SFF None 1 MUL Rd, Rr Multiply Signed R1:R0 - Rd x Rr Z.C 2 MULSU Rd, Rr Multiply Signed with Unsigned R1:R0 - Rd x Rr Z.C 2 MULSU Rd, Rr Fractional Multiply Unsigned R1:R0 - (Rd x Rr) <	TST	Rd	Test for Zero or Minus	$Rd \leftarrow Rd \bullet Rd$	Z,N,V	1
NUL Rd, Rr Multiply Unsigned R1R0 + Rd x Rr Z.C 2 MULS Rd, Rr Multiply Signed with Unsigned R1R0 + Rd x Rr Z.C 2 MULU Rd, Rr Fractional Multiply Unsigned R1R0 + Rd x Rr Z.C 2 FMUL Rd, Rr Fractional Multiply Signed with Unsigned R1R0 - (Rd x Rr) <<1	CLR	Rd	Clear Register	$Rd \leftarrow Rd \oplus Rd$	Z,N,V	1
NULSRd, RrMultiply Signed with LangendR1:R0 - Rd x RrZ.C2MULSURd, RrMultiply Signed with LangendR1:R0 - Rd x RrZ.C2FMULRd, RrFractional Multiply UnsignedR1:R0 - (Rd x R) <<1	SER	Rd	Set Register	$Rd \leftarrow \$FF$	None	1
MULSU Rd, Rr Multiply Signed with Unsigned R1:R0 - Rd x Rr Z.C 2 FMUL Rd, Rr Fractional Multiply Signed R1:R0 - (Rd x R) <<1	MUL	Rd, Rr	Multiply Unsigned	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
FMULRd, RrFractional Multiply UnsignedR1:R0 \leftarrow (Rd x R) $<<1$ Z.C2FMULSURd, RrFractional Multiply Signed with UnsignedR1:R0 \leftarrow (Rd x R) $<<1$ Z.C2FMULSURd, RrFractional Multiply Signed with UnsignedR1:R0 \leftarrow (Rd x R) $<<1$ Z.C2BRANCH INSTRUCTIONSZ.C2LIMPkRelative JumpPC \leftarrow PC $+$ k+1None2JMPkDirect Jump to (Z)PC \leftarrow ZNone3RCALLkRolative Subroutine CallPC \leftarrow PC $+$ k+1None3ICALLkDirect Subroutine CallPC \leftarrow ZNone3ICALLkDirect Subroutine CallPC \leftarrow KNone4RETSubroutine RatumPC \leftarrow StackI4CPSERd,RrCompareRd $-$ RrZ, NV,C,H1CPCRd,RrCompare Skip if EqualII (Rd $=$ R) PC \leftarrow PC $+$ 2 or 3None1/2/3CPRd,RrCompare Register with ImmediateRd $-$ Rr $-$ CZ, NV,C,H1CPCRd,RrCompare Register ClearedII (Rto)=0) PC \leftarrow PC $+$ 2 or 3None1/2/3SBRCRr, bSkp1 Bit in Register ClearedII (Rto)=0) PC $+$ PC $+$ 2 or 3None1/2/3SBRSP, bSkp1 Bit in Rigister ClearedII (Rto)=0) PC $+$ PC $+$ 2 or 3None1/2/3SBRSP, bSkp1 Bit in Rigister ClearedII (Rto)=0) PC $+$ PC $+$ 2 or 3None1/2/3S	MULS	Rd, Rr	Multiply Signed	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
FMULSURd, RrFractional Multiply SignedR1:R0 \leftarrow (Rd x R) $<<1$ Z.C2FMULSURd, RrFractional Multiply Signed with UnsignedR1:R0 \leftarrow (Rd x R) $<<1$ Z.C2BRANCH INSTRUCTIONSFractional Multiply Signed with UnsignedRC \leftarrow PC $+$ k+1None2JMPkRelative Jump (2)PC \leftarrow PC $+$ k+1None2JMPkDirect Jump (2)PC \leftarrow kNone3RCALLkRelative Subroutine CallPC \leftarrow kNone3CALLkDirect Call (2)PC \leftarrow kNone4RETSubroutine CallPC \leftarrow kNone4RETSubroutine ReturnPC \leftarrow StackNone4CPSERd,RrCompare, Skip I Equalif (Rd - Rr) PC \leftarrow PC $+$ 2 or 3None11/2/3CPRd,RrCompare Register with ImmediateRd $-$ Rr $-$ CZ, NV, C, H1CPRd,RrCompare Register with ImmediateRd $-$ Rr $-$ CZ, NV, C, H1SBRCRr, bSkip I Bit in Register Clearedif (Rr(b) =0) PC $-$ PC $+$ 2 or 3None11/2/3SBRSRr, bSkip I Bit in Register Clearedif (Rr(b) =0) PC $-$ PC $+$ 2 or 3None11/2/3SBRSP, bSkip I Bit in Register Clearedif (Rr(b) =0) PC $-$ PC $+$ 2 or 3None11/2/3SBRSRr, bSkip I Bit in Register I Setif (Rr(b) =0) PC $-$ PC $+$ 2 or 3None11/2/3SBRSP, bSkip I Bit in Register I Setif (Rr(MULSU	Rd, Rr	Multiply Signed with Unsigned	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
FMULSURd, RrFractional Multiply Signed with UnsignedR1:R0 \leftarrow (Rd x Rr) < 1 Z,C2BRANCH INSTRUCTIONSUMPkRelative JumpPC \leftarrow PC $+$ k + 1None2JMPkDirect JumpPC \leftarrow ZNone3GALLkRelative Subroutine CallPC \leftarrow PC $+$ k + 1None3ICALLkDirect Subroutine CallPC \leftarrow PC $+$ k + 1None3ICALLkDirect Subroutine CallPC \leftarrow CNone4RETSubroutine ReturnPC \leftarrow StackNone4RETSubroutine ReturnPC \leftarrow StackNone1CPSRd,RrCompare, Skip If EqualIf (Rd = Rr) PC \leftarrow PC $+$ 2 or 3None1/2/3CPRd,RrCompare Mic CarryRd $-$ Rr $-$ CZ,NV,C,H1CPCRd,RrCompare Register with ImmediateRd $-$ Rr $-$ CZ,NV,C,H1CPCRd,KCompare Register ClearedIf (Rr0)=0) PC \leftarrow PC $+$ 2 or 3None1/2/3SBRSRr, bSkip If Bit in Kegister ClearedIf (Rr0)=0) PC \leftarrow PC $+$ 2 or 3None1/2/3SBRSP, bSkip If Bit in IOR Register Is SetIf (Rr0)=0) PC \leftarrow PC $+$ 2 or 3None1/2/3SBRSP, bSkip If Bit in IOR Register Is SetIf (Rr0)=0) PC \leftarrow PC $+$ 2 or 3None1/2/3SBRSP, bSkip If Bit in IOR Register Is SetIf (Rr0)=0) PC \leftarrow PC $+$ 2 or 3None1/2/3SBRSP, bSkip If Bit in IOR Re	FMUL	Rd, Rr	Fractional Multiply Unsigned	$R1:R0 \leftarrow (Rd \times Rr) << 1$	Z,C	2
BRANCH INSTRUCTIONSRUMPkRelative JumpPC \leftarrow PC + k + 1None2JMPindirect Jump (Z)PC \leftarrow CZNone2JMPkDirect Jump (Z)PC \leftarrow CANone3RCALLkRelative Subroutine CallPC \leftarrow PC + k + 1None3ICALLIndirect Call to (Z)PC \leftarrow PC + k + 1None3CALLkDirect Subroutine CallPC \leftarrow StackNone4RETSubroutine ReturnPC \leftarrow StackNone4RETInterrupt ReturnPC \leftarrow Stack14CPERd,RrCompare, Skip if EgualIf (Rd = Rr) PC \leftarrow PC + 2 or 3None1/2/3CPRd,RrCompare with CarryRd $-$ Rr $-$ CZ, N.V.C.H1CPCRd,KrCompare with CarryRd $-$ Rr $-$ CZ, N.V.C.H1CPLRd,KCompare Register with ImmediateRd $-$ KZ, N.V.C.H1CPLRd,KCompare Register with ImmediateRd $-$ KZ, N.V.C.H1SBRCRr, bSkip if Bit in Register is SetIf (RTD)=0) PC \leftarrow PC + 2 or 3None1/2/3SBISP, bSkip if Bit in IOR Register is SetIf (RTD)=0) PC \leftarrow PC + 2 or 3None1/2/3SBISP, bSkip if Bit in IOR Register is SetIf (RD)=0) PC \leftarrow PC + 2 or 3None1/2/3SBISP, bSkip if Bit in IOR Register is SetIf (RD)=0) PC \leftarrow PC + 2 or 3None1/2/3SBRSS, k </td <td>FMULS</td> <td>Rd, Rr</td> <td>Fractional Multiply Signed</td> <td>$R1:R0 \leftarrow (Rd x Rr) << 1$</td> <td>Z,C</td> <td>2</td>	FMULS	Rd, Rr	Fractional Multiply Signed	$R1:R0 \leftarrow (Rd x Rr) << 1$	Z,C	2
RJMPkRelative Jump $PC \leftarrow PC + k + 1$ None2LIMPIndirect Jump to (2) $PC \leftarrow PC + k + 1$ None2JMPkDirect Jump $PC \leftarrow k$ None3RCALLkRelative Subroutine Call $PC \leftarrow PC + k + 1$ None3ICALLkDirect Call to (2) $PC \leftarrow PC + k + 1$ None3CALLkDirect Subroutine Call $PC \leftarrow PC + k + 1$ None4RETSubroutine Return $PC \leftarrow stack$ None4CPERd,RrCompare, Skip if Equalif (Rd = Rr) PC $\leftarrow PC + 2 \text{ or 3}$ None1/2/3CPRd,RrCompare Mic GaryRd $-Rr - C$ Z, N, V, C, H1CPCRd,RrCompare Nagister with ImmediateRd $-Rr - C$ Z, N, V, C, H1CPIRd,KCompare Nagister with ImmediateRd $-Kr - C + 2 \text{ or 3}$ None1/2/3SBRCRr, bSkip if Bit in Register Iclearedif (Rr(b)-1) PC $+ PC + 2 \text{ or 3}$ None1/2/3SBRCP, bSkip if Bit in Negister Iclearedif (Rr(b)-1) PC $+ PC + 2 \text{ or 3}$ None1/2/3SBRSs, kBranch if Status Flag Setif (P(D)-1) PC $+ PC + 2 \text{ or 3}$ None1/2/3SBRSs, kBranch if Status Flag Setif (P(D)-1) PC $+ PC + 2 \text{ or 3}$ None1/2/3SBRSs, kBranch if Status Flag Setif (P(D)-1) PC $+ PC + 2 \text{ or 3}$ None1/2/3SBRSs, kBranch if Status Flag Setif (P(D)-1) PC $+ PC + 2$	FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	$R1:R0 \leftarrow (Rd x Rr) << 1$	Z,C	2
$ \begin{array}{c c c c c c c c c c c c c c c c c c c $	BRANCH INSTRUCT	TIONS		·		
JMPkDirect JumpPC +-kNone3RCALLkRelative Subroutine CallPC +-PC + k+1None3ICALLIndirect Call to (2)PC +-ZNone3CALLkDirect Subroutine CallPC +-KNone4RETSubroutine ReturnPC +-StackNone4RETInterrupt ReturnPCStackI4CPSERd,RrCompare, Skip if Equalif (Rd = Rr) PC +- PC + 2 or 3None1/2/3CPRd,RrCompare Mit CarryRd - Rr - CZ, NV,C,H1CPCRd,RrCompare Register with ImmediateRd - Rr - CZ, NV,C,H1CPIRd,KCompare Register Gearedif (R(b)=0) PC +-PC + 2 or 3None1/2/3SBRCRr, bSkip if Bit in Register IS Setif (R(b)=0) PC +-PC + 2 or 3None1/2/3SBICP, bSkip if Bit in I/O Register IS Setif (P(b)=1) PC +-PC + 2 or 3None1/2/3SBISP, bSkip if Bit in I/O Register IS Setif (P(b)=1) PC +-PC + 2 or 3None1/2/3SBRSs, kBranch if Status Flag Setif (SREG(s) = 0) then PC +-PC + k + 1None1/2BRBSs, kBranch if Status Flag Setif (C = 0) then PC + PC + k + 1None1/2BRCGkBranch if Carry Clearedif (C = 0) then PC + PC + k + 1None1/2BRCGkBranch if Gard Clearedif (C = 0) then PC + PC + k + 1None1/2BRRDk </td <td>RJMP</td> <td>k</td> <td>Relative Jump</td> <td>$PC \leftarrow PC + k + 1$</td> <td>None</td> <td>2</td>	RJMP	k	Relative Jump	$PC \leftarrow PC + k + 1$	None	2
RCALLkRelative Subroutine CallPC +- PC + k + 1None3ICALLIndirect Call to (2)PCZNone3CALLkDirect Subroutine CallPCZNone4RETSubroutine ReturnPC +-StackNone4RETIInterrupt ReturnPCStackI4CPSERd,RrCompare, Skip if Equalif (Rd = Rr) PCPC + 2 or 3None1/2/3CPRd,RrCompare QRd - RrZ, NV,C.H1CPCRd,RrCompare Register with ImmediateRd - RrZ, NV,C.H1CP1Rd.KCompare Register with ImmediateRd - KZ, NV,C.H1SBRCRr, bSkip if Bit in Register Clearedif (Rt(b)=1) PCPC + 2 or 3None1/2/3SBRSRr, bSkip if Bit in Register Clearedif (Rt(b)=1) PCPC + 2 or 3None1/2/3SBRSP, bSkip if Bit in UC Register Clearedif (Rt(b)=1) PCPC + 2 or 3None1/2/3SBRSP, bSkip if Bit in Register Clearedif (Rt(b)=1) PCPC + 2 or 3None1/2/3SBRSP, bSkip if Bit in Register Setif (P(b)=1) PCPC + 2 or 3None1/2/3SBRSS, kBranch if Status Flag Clearedif (Rt(b)=0) then PC -PC + 2 or 3None1/2/3BRBSS, kBranch if Status Flag Clearedif (SREG(s)=0) then PCPC + k + 1None1/2BRCQkBranch if Cary Setif (C = 1) then PC -PC + k + 1None <td< td=""><td>IJMP</td><td></td><td>Indirect Jump to (Z)</td><td>$PC \leftarrow Z$</td><td>None</td><td>2</td></td<>	IJMP		Indirect Jump to (Z)	$PC \leftarrow Z$	None	2
$\begin{array}{c c c c c c c c c c c c c c c c c c c $	JMP	k	Direct Jump	PC ← k	None	3
CALLkDirect Subroutine CallPC \leftarrow kNone4RETSubroutine ReturnPC \leftarrow StackNone4RETIInterrupt ReturnPC \leftarrow StackI4CPSRd,RrCompare, Skp if Equalif (Rd = Rr) PC \leftarrow PC + 2 or 3None1/2/3CPRd,RrCompare of Rd = RrZ, NV,C,H11CPCRd,RrCompare with CarryRd = Rr - CZ, NV,C,H1CPIRd,KCompare Register with ImmediateRd = KrZ, NV,C,H1SBRCRr, bSkip if Bit in Register IS Setif (Rt(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBRSRr, bSkip if Bit in Register Clearedif (P(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBRSP, bSkip if Bit in I/O Register Clearedif (P(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBRSS, kBranch if Status Flag Setif (SREG(s) = 1) then PC \leftarrow PC + 2 or 3None1/2/3SBRSS, kBranch if Status Flag Setif (SREG(s) = 0) then PC \leftarrow PC + k + 1None1/2/3BRBSs, kBranch if Status Flag Clearedif (Z = 0) then PC \leftarrow PC + k + 1None1/2BREQkBranch if Carry Setif (C = 0) then PC \leftarrow PC + k + 1None1/2BRNEkBranch if Carry Setif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Game Or Higherif (C = 0) then PC \leftarrow PC + k + 1None1/2BRNEkBranch if Game Or Hig	RCALL	k	Relative Subroutine Call	$PC \leftarrow PC + k + 1$	None	3
RETSubroutine ReturnPC \leftarrow StackNone4RETIInterrupt ReturnPC \leftarrow StackI4CPSERd,RrCompare, Skip if Equalif (Rd = Rr) PC \leftarrow PC + 2 or 3None1/2/3CPRd,RrCompareRd - RrZ, N, V, C, H1CPCRd,RrCompare Register with ImmediateRd - Rr - CZ, N, V, C, H1CPIRd,KCompare Register with ImmediateRd - KZ, N, V, C, H1SBRCRr, bSkip if Bit in Register Setif (Rr(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBRSRr, bSkip if Bit in Register Setif (Rr(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBISP, bSkip if Bit in I/O Register Clearedif (P(b)=1) PC \leftarrow PC + 2 or 3None1/2/3SBISP, bSkip if Bit in I/O Register Setif (P(b)=1) PC \leftarrow PC + 2 or 3None1/2/3BRBSs, kBranch if Status Flag Setif (SREG(s) = 0) then PC \leftarrow PC + k + 1None1/2BRBCs, kBranch if Status Flag Clearedif (SREG(s) = 0) then PC \leftarrow PC + k + 1None1/2BREQkBranch if Caruy Setif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCkBranch if Caruy Setif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCkBranch if Gared or regula, Signedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCkBranch if Gared or Equal, Signedif (N = 0) then PC \leftarrow PC + k + 1None1/2 <td>ICALL</td> <td></td> <td>Indirect Call to (Z)</td> <td>$PC \leftarrow Z$</td> <td>None</td> <td>3</td>	ICALL		Indirect Call to (Z)	$PC \leftarrow Z$	None	3
RETIInterrupt Return $PC \leftarrow Stack$ I4CPSERd,RrCompare, Skip if Equalif (Rd = Rr) PC \leftarrow PC + 2 or 3None $1/2/3$ CPRd,RrCompareCompareRd - RrZ, N,V,C,H1CPCRd,RrCompare with CarryRd - Rr - CZ, N,V,C,H1CPIRd,KCompare Register with ImmediateRd - KZ, N,V,C,H1SBRCRr, bSkip if Bit in Register Clearedif (Rr(b)=0) PC ← PC + 2 or 3None $1/2/3$ SBRSRr, bSkip if Bit in Register is Setif (Rr(b)=1) PC ← PC + 2 or 3None $1/2/3$ SBRSP, bSkip if Bit in I/O Register is Setif (PD)=1) PC ← PC + 2 or 3None $1/2/3$ SBRSP, bSkip if Bit in I/O Register is Setif (PD)=1) PC ← PC + 2 or 3None $1/2/3$ SBRSs, kBranch if Status Flag Setif (PD)=1) PC ← PC + 2 or 3None $1/2/3$ BRBSs, kBranch if Status Flag Clearedif (RREG(s) = 1) then PC ← PC + k + 1None $1/2$ BREQkBranch if Status Flag Clearedif (Z = 0) then PC ← PC + k + 1None $1/2$ BRCSkBranch if Not Equalif (Z = 0) then PC ← PC + k + 1None $1/2$ BRCSkBranch if I Garry Clearedif (C = 0) then PC ← PC + k + 1None $1/2$ BRCSkBranch if Garry Clearedif (C = 0) then PC ← PC + k + 1None $1/2$ BRCSkBranch if Garry Clearedif (C = 0) then PC ← PC + k + 1 </td <td>CALL</td> <td>k</td> <td>Direct Subroutine Call</td> <td>PC ← k</td> <td>None</td> <td>4</td>	CALL	k	Direct Subroutine Call	PC ← k	None	4
CPSERd,RrCompare, Skip if Equalif (Rd = Rr) PC \leftarrow PC + 2 or 3None1/2/3CPRd,RrCompareRd - RrZ, NV,C,H1CPCRd,RrCompare with CarryRd - Rr - CZ, NV,C,H1CPIRd,KCompare Register with ImmediateRd - KZ, NV,C,H1SBRCRr, bSkip if Bit in Register Clearedif (Rr(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBRSRr, bSkip if Bit in Register is Setif (Rr(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBICP, bSkip if Bit in I/O Register Clearedif (P(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBISP, bSkip if Bit in I/O Register Clearedif (P(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBRSs, kBranch if Status Flag Setif (SREG(s) = 1) then PC \leftarrow PC + k + 1None1/2/3BRBCs, kBranch if Status Flag Clearedif (SREG(s) = 0) then PC \leftarrow PC + k + 1None1/2BRPQkBranch if Grupaif (Z = 1) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Carry Setif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Carry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Carry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Game or Higherif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCLkBranch if Game or Equal, Signedif	RET		Subroutine Return	PC ← Stack	None	4
CPRd,RrCompareRd - RrZ, N, V, C, H1CPCRd,RrCompare with CarryRd - Rr - CZ, N, V, C, H1CPIRd,KCompare Register with ImmediateRd - KZ, N, V, C, H1CBIRr, bSkip if Bit in Register Clearedif (Rr(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBRCRr, bSkip if Bit in Register Clearedif (Rr(b)=1) PC \leftarrow PC + 2 or 3None1/2/3SBRCP, bSkip if Bit in Register Clearedif (P(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBICP, bSkip if Bit in I/O Register Clearedif (P(b)=1) PC \leftarrow PC + 2 or 3None1/2/3SBRSs, kBranch if Status Flag Setif (SREG(s) = 1) then PC- \leftarrow PC+k+1None1/2/3BRBSs, kBranch if Status Flag Setif (SREG(s) = 0) then PC- \leftarrow PC+k+1None1/2BREQkBranch if Status Flag Clearedif (Z = 0) then PC \leftarrow PC+k+1None1/2BRCSkBranch if Cary Setif (C = 1) then PC \leftarrow PC+k+1None1/2BRCCkBranch if Carry Setif (C = 0) then PC \leftarrow PC+k+1None1/2BRSHkBranch if Same or Higherif (C = 0) then PC \leftarrow PC+k+1None1/2BRSHkBranch if Same or Higherif (C = 0) then PC \leftarrow PC+k+1None1/2BRSHkBranch if Gareg Setif (N = 1) then PC \leftarrow PC+k+1None1/2BRSHkBranch if Gareg Setif (N = 1) then PC \leftarrow PC+k+1N	RETI		Interrupt Return	PC ← Stack	1	4
CPCRd.RrCompare with CarryRd - Rr - CZ, N, V, C, H1CPIRd,KCompare Register with ImmediateRd - KZ, N, V, C, H1SBRCRr, bSkip if Bit in Register Clearedif (Rr(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBRSRr, bSkip if Bit in Register Clearedif (Rr(b)=1) PC \leftarrow PC + 2 or 3None1/2/3SBRCP, bSkip if Bit in 1/O Register Clearedif (P(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBISP, bSkip if Bit in 1/O Register Clearedif (P(b)=1) PC \leftarrow PC + 2 or 3None1/2/3BRSs, kBranch if Status Flag Setif (SREG(s) = 1) then PC \leftarrow PC + k + 1None1/2BREQs, kBranch if Status Flag Clearedif (Z = 0) then PC \leftarrow PC + k + 1None1/2BREQkBranch if Act Qualif (Z = 0) then PC \leftarrow PC + k + 1None1/2BRCSkBranch if Carry Setif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Carry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Same or Higherif (C = 0) then PC \leftarrow PC + k + 1None1/2BRSHkBranch if Garry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Mame or Higherif (C = 0) then PC \leftarrow PC + k + 1None1/2BRHkBranch if Garry Clearedif (N = 0) then PC \leftarrow PC + k + 1None1/2BRHkBranch	CPSE	Rd,Rr	Compare, Skip if Equal	if (Rd = Rr) PC \leftarrow PC + 2 or 3	None	1/2/3
CPIRd,KCompare Register with ImmediateRd – KZ, N,V,C,H1SBRCRr, bSkip if Bit in Register Clearedif (Rr(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBRSRr, bSkip if Bit in Register is Setif (Rr(b)=1) PC \leftarrow PC + 2 or 3None1/2/3SBICP, bSkip if Bit in I/O Register Clearedif (P(b)=1) PC \leftarrow PC + 2 or 3None1/2/3SBISP, bSkip if Bit in I/O Register is Setif (P(b)=1) PC \leftarrow PC + 2 or 3None1/2/3BRSSs, kBranch if Status Flag Setif (SREG(s) = 1) then PC \leftarrow PC + k + 1None1/2/3BRBCs, kBranch if Status Flag Setif (SREG(s) = 0) then PC \leftarrow PC + k + 1None1/2/3BREQkBranch if Status Flag Clearedif (Z = 0) then PC \leftarrow PC + k + 1None1/2/3BREQkBranch if Ott Equalif (Z = 0) then PC \leftarrow PC + k + 1None1/2BRNEkBranch if Carry Setif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Carry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRLOkBranch if Carry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRMIkBranch if Ilowerif (N = 0) then PC \leftarrow PC + k + 1None1/2BRMIkBranch if Ilowerif (N = 0) then PC \leftarrow PC + k + 1None1/2BRMIkBranch if Ilowerif (N = 0) then PC \leftarrow PC + k + 1None1/2BRHSk <td< td=""><td>CP</td><td>Rd,Rr</td><td>Compare</td><td>Rd – Rr</td><td>Z, N,V,C,H</td><td>1</td></td<>	CP	Rd,Rr	Compare	Rd – Rr	Z, N,V,C,H	1
SBRCRr, bSkip if Bit in Register Clearedif (Rr(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBRSRr, bSkip if Bit in Register is Setif (Rr(b)=1) PC \leftarrow PC + 2 or 3None1/2/3SBICP, bSkip if Bit in I/O Register Clearedif (P(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBISP, bSkip if Bit in I/O Register is Setif (P(b)=0) PC \leftarrow PC + 2 or 3None1/2/3SBISP, bSkip if Bit in I/O Register is Setif (P(b)=0) PC \leftarrow PC + 2 or 3None1/2/3BRBSs, kBranch if Status Flag Setif (P(b)=1) PC \leftarrow PC + 2 or 3None1/2/3BRBCs, kBranch if Status Flag Setif (SREG(s) = 0) then PC \leftarrow PC + k + 1None1/2BREQkBranch if Equalif (Z = 0) then PC \leftarrow PC + k + 1None1/2BRNEkBranch if Carry Setif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Carry Setif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Carry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Same or Higherif (C = 0) then PC \leftarrow PC + k + 1None1/2BRCkBranch if Minusif (C = 0) then PC \leftarrow PC + k + 1None1/2BRL0kBranch if Minusif (N = 1) then PC \leftarrow PC + k + 1None1/2BRPLkBranch if Greater or Equal, Signedif (N \oplus V = 0) then PC \leftarrow PC + k + 1None1/2B	CPC	Rd,Rr	Compare with Carry	Rd – Rr – C	Z, N,V,C,H	1
SBRSRr, bSkip if Bit in Register is Setif $(R(b)=1) PC \leftarrow PC+2 \text{ or } 3$ None $1/2/3$ SBICP, bSkip if Bit in I/O Register Clearedif $(P(b)=0) PC \leftarrow PC+2 \text{ or } 3$ None $1/2/3$ SBISP, bSkip if Bit in I/O Register is Setif $(P(b)=1) PC \leftarrow PC+2 \text{ or } 3$ None $1/2/3$ BRSs, kBranch if Status Flag Setif $(P(b)=1) PC \leftarrow PC+2 \text{ or } 3$ None $1/2/3$ BRBCs, kBranch if Status Flag Setif $(SREG(s)=1)$ then $PC \leftarrow PC+k+1$ None $1/2$ BREQkBranch if Status Flag Clearedif $(Z=0)$ then $PC \leftarrow PC+k+1$ None $1/2$ BREQkBranch if Gary Setif $(Z=0)$ then $PC \leftarrow PC+k+1$ None $1/2$ BRCSkBranch if Carry Setif $(C=1)$ then $PC \leftarrow PC+k+1$ None $1/2$ BRCCkBranch if Same or Higherif $(C=0)$ then $PC \leftarrow PC+k+1$ None $1/2$ BRLDkBranch if Jourg Clearedif $(C=1)$ then $PC \leftarrow PC+k+1$ None $1/2$ BRLDkBranch if Jourg Clearedif $(C=0)$ then $PC \leftarrow PC+k+1$ None $1/2$ BRLDkBranch if Jourg Clearedif $(N=0)$ then $PC \leftarrow PC+k+1$ None $1/2$ BRLDkBranch if Jourg Clearedif $(N=0)$ then $PC \leftarrow PC+k+1$ None $1/2$ BRLDkBranch if Jourg Clearedif $(N=0)$ then $PC \leftarrow PC+k+1$ None $1/2$ BRLDkBranch if Jourg Clearedif $(N=0)$ then $PC \leftarrow PC+k+1$ None $1/2$ BRHK <td>CPI</td> <td>Rd,K</td> <td>Compare Register with Immediate</td> <td>Rd – K</td> <td>Z, N,V,C,H</td> <td>1</td>	CPI	Rd,K	Compare Register with Immediate	Rd – K	Z, N,V,C,H	1
SBICP, bSkip if Bit in I/O Register Clearedif $(P(b)=0) PC \leftarrow PC + 2 \text{ or } 3$ None $1/2/3$ SBISP, bSkip if Bit in I/O Register is Setif $(P(b)=1) PC \leftarrow PC + 2 \text{ or } 3$ None $1/2/3$ BRBSs, kBranch if Status Flag Setif $(SREG(s) = 1) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRBCs, kBranch if Status Flag Clearedif $(SREG(s) = 0) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRRCkBranch if Equalif $(Z = 1) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRREkBranch if Not Equalif $(Z = 0) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRCSkBranch if Carry Setif $(C = 1) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRCCkBranch if Carry Clearedif $(C = 0) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRSHkBranch if Same or Higherif $(C = 0) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRLOkBranch if Ilowerif $(C = 0) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRMIkBranch if Ilowerif $(N = 0) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRMIkBranch if Greater or Equal, Signedif $(N \oplus V = 0) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRCEkBranch if Greater or Equal, Signedif $(N \oplus V = 0) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRHSkBranch if Half Carry Flag Setif $(H = 1) \text{ then } PC \leftarrow PC + k + 1$ None $1/2$ BRHSkBranch if Half	SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBISP, bSkip if Bit in I/O Register is Setif (P(b)=1) PC \leftarrow PC + 2 or 3None1/2/3BRBSs, kBranch if Status Flag Setif (SREG(s) = 1) then PC \leftarrow PC+k+1None1/2BRBCs, kBranch if Status Flag Clearedif (SREG(s) = 0) then PC \leftarrow PC+k+1None1/2BREQkBranch if Equalif (Z = 1) then PC \leftarrow PC+k+1None1/2BRNEkBranch if Not Equalif (Z = 0) then PC \leftarrow PC+k+1None1/2BRCSkBranch if Carry Setif (C = 1) then PC \leftarrow PC+k+1None1/2BRCCkBranch if Garry Clearedif (C = 0) then PC \leftarrow PC+k+1None1/2BRCDkBranch if Same or Higherif (C = 0) then PC \leftarrow PC+k+1None1/2BRLDkBranch if Minusif (N = 1) then PC \leftarrow PC+k+1None1/2BRMIkBranch if Greater or Equal, Signedif (N = 1) then PC \leftarrow PC+k+1None1/2BRCEkBranch if Greater or Equal, Signedif (N = 0) then PC \leftarrow PC+k+1None1/2BRCEkBranch if Greater or Equal, Signedif (N = 0) then PC \leftarrow PC+k+1None1/2BRHSkBranch if Jargenges Than Zero, Signedif (N = 0) then PC \leftarrow PC+k+1None1/2BRHSkBranch if Half Carry Flag Setif (H = 1) then PC \leftarrow PC+k+1None1/2BRHSkBranch if Half Carry Flag Setif (H = 0) then PC \leftarrow PC+k+1None1/2BRHCkBranch if Half Carry F	SBRS	Rr, b	Skip if Bit in Register is Set	if (Rr(b)=1) PC ← PC + 2 or 3	None	1/2/3
BRBSs, kBranch if Status Flag Setif (SREG(s) = 1) then PC ← PC+k + 1None1/2BRBCs, kBranch if Status Flag Clearedif (SREG(s) = 0) then PC ← PC+k + 1None1/2BREQkBranch if Equalif (Z = 1) then PC ← PC+k + 1None1/2BRNEkBranch if Not Equalif (Z = 0) then PC ← PC+k + 1None1/2BRCSkBranch if Carry Setif (C = 1) then PC ← PC+k+1None1/2BRCCkBranch if Carry Clearedif (C = 0) then PC ← PC+k+1None1/2BRSHkBranch if Same or Higherif (C = 0) then PC ← PC+k+1None1/2BRLOkBranch if Lowerif (C = 1) then PC ← PC+k+1None1/2BRMIkBranch if Mussif (N = 1) then PC ← PC+k+1None1/2BRELkBranch if Greater or Equal, Signedif (N = 0) then PC ← PC+k+1None1/2BRGEkBranch if Greater or Equal, Signedif (N ⊕ V= 0) then PC ← PC+k+1None1/2BRHSkBranch if Less Than Zero, Signedif (N ⊕ V= 1) then PC ← PC+k+1None1/2BRHSkBranch if Half Carry Flag Setif (H = 1) then PC ← PC+k+1None1/2BRHCkBranch if Half Carry Flag Clearedif (H = 0) then PC ← PC+k+1None1/2	SBIC	P, b	Skip if Bit in I/O Register Cleared	if (P(b)=0) PC \leftarrow PC + 2 or 3	None	1/2/3
BRBCs, kBranch if Status Flag Clearedif (SREG(s) = 0) then PC \leftarrow PC + k + 1None1/2BREQkBranch if Equalif (Z = 1) then PC \leftarrow PC + k + 1None1/2BRNEkBranch if Not Equalif (Z = 0) then PC \leftarrow PC + k + 1None1/2BRCSkBranch if Carry Setif (C = 1) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Carry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRSHkBranch if Same or Higherif (C = 0) then PC \leftarrow PC + k + 1None1/2BRLOkBranch if Lowerif (C = 1) then PC \leftarrow PC + k + 1None1/2BRMIkBranch if Lowerif (N = 1) then PC \leftarrow PC + k + 1None1/2BRLDkBranch if Minusif (N = 1) then PC \leftarrow PC + k + 1None1/2BRL1kBranch if Minusif (N = 0) then PC \leftarrow PC + k + 1None1/2BRE2kBranch if Greater or Equal, Signedif (N ⊕ V = 0) then PC ← PC + k + 1None1/2BRL1kBranch if Less Than Zero, Signedif (N ⊕ V = 1) then PC ← PC + k + 1None1/2BRH2kBranch if Half Carry Flag Setif (H = 1) then PC ← PC + k + 1None1/2BRH5kBranch if Half Carry Flag Clearedif (H = 0) then PC ← PC + k + 1None1/2BRHCkBranch if Half Carry Flag Clearedif (H = 0) then PC ← PC + k + 1None1/2	SBIS	P, b	Skip if Bit in I/O Register is Set	if (P(b)=1) PC \leftarrow PC + 2 or 3	None	1/2/3
BREQkBranch if Equalif (Z = 1) then PC \leftarrow PC + k + 1None1/2BRNEkBranch if Not Equalif (Z = 0) then PC \leftarrow PC + k + 1None1/2BRCSkBranch if Carry Setif (C = 1) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Carry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRSHkBranch if Same or Higherif (C = 0) then PC \leftarrow PC + k + 1None1/2BRLOkBranch if Lowerif (C = 0) then PC \leftarrow PC + k + 1None1/2BRMIkBranch if Minusif (N = 1) then PC \leftarrow PC + k + 1None1/2BRPLkBranch if Minusif (N = 0) then PC \leftarrow PC + k + 1None1/2BREEkBranch if Greater or Equal, Signedif (N = 0) then PC \leftarrow PC + k + 1None1/2BRLTkBranch if Less Than Zero, Signedif (N \oplus V = 0) then PC \leftarrow PC + k + 1None1/2BRHSkBranch if Half Carry Flag Setif (H = 1) then PC \leftarrow PC + k + 1None1/2BRHCkBranch if Half Carry Flag Clearedif (H = 0) then PC \leftarrow PC + k + 1None1/2	BRBS	s, k	Branch if Status Flag Set	if $(SREG(s) = 1)$ then $PC \leftarrow PC+k + 1$	None	1/2
BRNEkBranch if Not Equalif (Z = 0) then PC \leftarrow PC + k + 1None1/2BRCSkBranch if Carry Setif (C = 1) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Carry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRSHkBranch if Same or Higherif (C = 0) then PC \leftarrow PC + k + 1None1/2BRLOkBranch if Lowerif (C = 1) then PC \leftarrow PC + k + 1None1/2BRMIkBranch if Minusif (N = 1) then PC \leftarrow PC + k + 1None1/2BRPLkBranch if Minusif (N = 0) then PC \leftarrow PC + k + 1None1/2BRGEkBranch if Greater or Equal, Signedif (N = 0) then PC \leftarrow PC + k + 1None1/2BRLTkBranch if Less Than Zero, Signedif (N \oplus V = 0) then PC \leftarrow PC + k + 1None1/2BRHSkBranch if Half Carry Flag Setif (H = 1) then PC \leftarrow PC + k + 1None1/2BRHCkBranch if Half Carry Flag Clearedif (H = 0) then PC \leftarrow PC + k + 1None1/2	BRBC	s, k	Branch if Status Flag Cleared	if $(SREG(s) = 0)$ then $PC \leftarrow PC+k + 1$	None	1/2
BRCSkBranch if Carry Setif (C = 1) then PC \leftarrow PC + k + 1None1/2BRCCkBranch if Carry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRSHkBranch if Same or Higherif (C = 0) then PC \leftarrow PC + k + 1None1/2BRLOkBranch if Lowerif (C = 1) then PC \leftarrow PC + k + 1None1/2BRMIkBranch if Minusif (N = 1) then PC \leftarrow PC + k + 1None1/2BRPLkBranch if Minusif (N = 0) then PC \leftarrow PC + k + 1None1/2BRGEkBranch if Greater or Equal, Signedif (N = 0) then PC \leftarrow PC + k + 1None1/2BRLTkBranch if Less Than Zero, Signedif (N \oplus V = 0) then PC \leftarrow PC + k + 1None1/2BRHSkBranch if Half Carry Flag Setif (H = 1) then PC \leftarrow PC + k + 1None1/2BRHCkBranch if Half Carry Flag Clearedif (H = 0) then PC \leftarrow PC + k + 1None1/2	BREQ	k	Branch if Equal	if (Z = 1) then PC \leftarrow PC + k + 1	None	1/2
BRCCkBranch if Carry Clearedif (C = 0) then PC \leftarrow PC + k + 1None1/2BRSHkBranch if Same or Higherif (C = 0) then PC \leftarrow PC + k + 1None1/2BRLOkBranch if Lowerif (C = 1) then PC \leftarrow PC + k + 1None1/2BRMIkBranch if Minusif (N = 1) then PC \leftarrow PC + k + 1None1/2BRPLkBranch if Plusif (N = 0) then PC \leftarrow PC + k + 1None1/2BRGEkBranch if Greater or Equal, Signedif (N \oplus V = 0) then PC \leftarrow PC + k + 1None1/2BRLTkBranch if Less Than Zero, Signedif (N \oplus V = 1) then PC \leftarrow PC + k + 1None1/2BRHSkBranch if Half Carry Flag Setif (H = 1) then PC \leftarrow PC + k + 1None1/2BRHCkBranch if Half Carry Flag Clearedif (H = 0) then PC \leftarrow PC + k + 1None1/2	BRNE	k	Branch if Not Equal	if (Z = 0) then PC \leftarrow PC + k + 1	None	1/2
BRSHkBranch if Same or Higherif $(C = 0)$ then $PC \leftarrow PC + k + 1$ None1/2BRLOkBranch if Lowerif $(C = 1)$ then $PC \leftarrow PC + k + 1$ None1/2BRMIkBranch if Minusif $(N = 1)$ then $PC \leftarrow PC + k + 1$ None1/2BRPLkBranch if Plusif $(N = 0)$ then $PC \leftarrow PC + k + 1$ None1/2BRGEkBranch if Greater or Equal, Signedif $(N \oplus V = 0)$ then $PC \leftarrow PC + k + 1$ None1/2BRLTkBranch if Less Than Zero, Signedif $(N \oplus V = 1)$ then $PC \leftarrow PC + k + 1$ None1/2BRHSkBranch if Half Carry Flag Setif $(H = 1)$ then $PC \leftarrow PC + k + 1$ None1/2BRHCkBranch if Half Carry Flag Clearedif $(H = 0)$ then $PC \leftarrow PC + k + 1$ None1/2	BRCS	k	Branch if Carry Set	if (C = 1) then PC \leftarrow PC + k + 1	None	1/2
BRLOkBranch if Lowerif $(C = 1)$ then $PC \leftarrow PC + k + 1$ None1/2BRMIkBranch if Minusif $(N = 1)$ then $PC \leftarrow PC + k + 1$ None1/2BRPLkBranch if Plusif $(N = 0)$ then $PC \leftarrow PC + k + 1$ None1/2BRGEkBranch if Greater or Equal, Signedif $(N \oplus V = 0)$ then $PC \leftarrow PC + k + 1$ None1/2BRLTkBranch if Less Than Zero, Signedif $(N \oplus V = 0)$ then $PC \leftarrow PC + k + 1$ None1/2BRHSkBranch if Half Carry Flag Setif $(H = 1)$ then $PC \leftarrow PC + k + 1$ None1/2BRHCkBranch if Half Carry Flag Clearedif $(H = 0)$ then $PC \leftarrow PC + k + 1$ None1/2	BRCC	k	Branch if Carry Cleared	if (C = 0) then PC \leftarrow PC + k + 1	None	1/2
BRMIkBranch if Minusif (N = 1) then PC \leftarrow PC + k + 1None1/2BRPLkBranch if Plusif (N = 0) then PC \leftarrow PC + k + 1None1/2BRGEkBranch if Greater or Equal, Signedif (N \oplus V= 0) then PC \leftarrow PC + k + 1None1/2BRLTkBranch if Less Than Zero, Signedif (N \oplus V= 1) then PC \leftarrow PC + k + 1None1/2BRHSkBranch if Half Carry Flag Setif (H = 1) then PC \leftarrow PC + k + 1None1/2BRHCkBranch if Half Carry Flag Clearedif (H = 0) then PC \leftarrow PC + k + 1None1/2	BRSH	k	Branch if Same or Higher	if (C = 0) then PC \leftarrow PC + k + 1	None	1/2
BRMIkBranch if Minusif $(N = 1)$ then $PC \leftarrow PC + k + 1$ None1/2BRPLkBranch if Plusif $(N = 0)$ then $PC \leftarrow PC + k + 1$ None1/2BRGEkBranch if Greater or Equal, Signedif $(N \oplus V = 0)$ then $PC \leftarrow PC + k + 1$ None1/2BRLTkBranch if Less Than Zero, Signedif $(N \oplus V = 1)$ then $PC \leftarrow PC + k + 1$ None1/2BRHSkBranch if Half Carry Flag Setif $(H = 1)$ then $PC \leftarrow PC + k + 1$ None1/2BRHCkBranch if Half Carry Flag Clearedif $(H = 0)$ then $PC \leftarrow PC + k + 1$ None1/2	BRLO	k	Branch if Lower	if (C = 1) then PC \leftarrow PC + k + 1	None	1/2
BRGEkBranch if Greater or Equal, Signedif $(N \oplus V=0)$ then $PC \leftarrow PC + k + 1$ None1/2BRLTkBranch if Less Than Zero, Signedif $(N \oplus V=1)$ then $PC \leftarrow PC + k + 1$ None1/2BRHSkBranch if Half Carry Flag Setif $(H=1)$ then $PC \leftarrow PC + k + 1$ None1/2BRHCkBranch if Half Carry Flag Clearedif $(H=0)$ then $PC \leftarrow PC + k + 1$ None1/2	BRMI	k	Branch if Minus	if (N = 1) then PC \leftarrow PC + k + 1	None	
BRGEkBranch if Greater or Equal, Signedif $(N \oplus V=0)$ then $PC \leftarrow PC + k + 1$ None1/2BRLTkBranch if Less Than Zero, Signedif $(N \oplus V=1)$ then $PC \leftarrow PC + k + 1$ None1/2BRHSkBranch if Half Carry Flag Setif $(H=1)$ then $PC \leftarrow PC + k + 1$ None1/2BRHCkBranch if Half Carry Flag Clearedif $(H=0)$ then $PC \leftarrow PC + k + 1$ None1/2	BRPL	k				
BRLTkBranch if Less Than Zero, Signedif $(N \oplus V=1)$ then PC \leftarrow PC + k + 1None1/2BRHSkBranch if Half Carry Flag Setif $(H=1)$ then PC \leftarrow PC + k + 1None1/2BRHCkBranch if Half Carry Flag Clearedif $(H=0)$ then PC \leftarrow PC + k + 1None1/2						
BRHS k Branch if Half Carry Flag Set if (H = 1) then PC ← PC + k + 1 None 1/2 BRHC k Branch if Half Carry Flag Cleared if (H = 0) then PC ← PC + k + 1 None 1/2	BRLT	k			None	
BRHC k Branch if Half Carry Flag Cleared if (H = 0) then PC ← PC + k + 1 None 1 / 2						
	BRTS	k	Branch if T Flag Set	if (T = 1) then PC \leftarrow PC + k + 1	None	1/2



Mnemonics	Operands	Description	Operation	Flags	#Clocks
BRTC	k	Branch if T Flag Cleared	if (T = 0) then PC \leftarrow PC + k + 1	None	1/2
BRVS	k	Branch if Overflow Flag is Set	if (V = 1) then PC \leftarrow PC + k + 1	None	1/2
BRVC	k	Branch if Overflow Flag is Cleared	if (V = 0) then PC \leftarrow PC + k + 1	None	1/2
BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC \leftarrow PC + k + 1	None	1/2
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC \leftarrow PC + k + 1	None	1/2
DATA TRANSFER I	NSTRUCTIONS				
MOV	Rd, Rr	Move Between Registers	$Rd \leftarrow Rr$	None	1
MOVW	Rd, Rr	Copy Register Word	$Rd+1:Rd \leftarrow Rr+1:Rr$	None	1
LDI	Rd, K	Load Immediate	$Rd \leftarrow K$	None	1
LD	Rd, X	Load Indirect	$Rd \leftarrow (X)$	None	2
LD	Rd, X+	Load Indirect and Post-Inc.	$Rd \leftarrow (X), X \leftarrow X + 1$	None	2
LD	Rd, - X	Load Indirect and Pre-Dec.	$X \leftarrow X - 1, Rd \leftarrow (X)$	None	2
LD	Rd, Y	Load Indirect	$Rd \leftarrow (Y)$	None	2
LD	Rd, Y+	Load Indirect and Post-Inc.	$Rd \leftarrow (Y), Y \leftarrow Y + 1$	None	2
LD	Rd, - Y	Load Indirect and Pre-Dec.	$Y \leftarrow Y - 1, Rd \leftarrow (Y)$	None	2
LDD	Rd,Y+q	Load Indirect with Displacement	$Rd \leftarrow (Y + q)$	None	2
LD	Rd, Z	Load Indirect	$Rd \leftarrow (Z)$	None	2
LD	Rd, Z+	Load Indirect and Post-Inc.	$Rd \leftarrow (Z), Z \leftarrow Z+1$	None	2
LD	Rd, -Z	Load Indirect and Pre-Dec.	$Z \leftarrow Z - 1, Rd \leftarrow (Z)$	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	$Rd \leftarrow (Z + q)$	None	2
LDS	Rd, k	Load Direct from SRAM	$Rd \leftarrow (k)$	None	2
ST	X, Rr	Store Indirect	$(X) \leftarrow Rr$	None	2
ST	X+, Rr	Store Indirect and Post-Inc.	$(X) \leftarrow Rr, X \leftarrow X + 1$	None	2
ST	- X, Rr	Store Indirect and Pre-Dec.	$X \leftarrow X - 1, (X) \leftarrow Rr$	None	2
ST	Y, Rr	Store Indirect	$(Y) \leftarrow Rr$	None	2
ST	Y+, Rr	Store Indirect and Post-Inc.	$(Y) \leftarrow Rr, Y \leftarrow Y + 1$	None	2
ST	- Y, Rr	Store Indirect and Pre-Dec.	$Y \leftarrow Y - 1, (Y) \leftarrow Rr$	None	2
STD	Y+q,Rr	Store Indirect with Displacement	$(Y + q) \leftarrow Rr$	None	2
ST	Z, Rr	Store Indirect	$(Z) \leftarrow Rr$	None	2
ST	Z+, Rr	Store Indirect and Post-Inc.	$(Z) \leftarrow Rr, Z \leftarrow Z + 1$	None	2
ST	-Z, Rr	Store Indirect and Pre-Dec.	$Z \leftarrow Z - 1, (Z) \leftarrow Rr$	None	2
STD	Z+q,Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store Direct to SRAM	$(k) \leftarrow Rr$	None	2
LPM		Load Program Memory	$R0 \leftarrow (Z)$	None	3
LPM	Rd, Z	Load Program Memory	$Rd \leftarrow (Z)$	None	3
LPM	Rd, Z+	Load Program Memory and Post-Inc	$Rd \leftarrow (Z), Z \leftarrow Z+1$	None	3
SPM		Store Program Memory	(Z) ← R1:R0	None	-
IN	Rd, P	In Port	Rd ← P	None	1
OUT	P, Rr	Out Port	P ← Rr	None	1
PUSH	Rr	Push Register on Stack	Stack ← Rr	None	2
POP	Rd	Pop Register from Stack	Rd ← Stack	None	2
BIT AND BIT-TEST		Oct Dit in 1/O De nictor		Maria	-
SBI	P,b	Set Bit in I/O Register	$I/O(P,b) \leftarrow 1$	None	2
CBI	P,b	Clear Bit in I/O Register	$I/O(P,b) \leftarrow 0$	None	2
LSL	Rd	Logical Shift Left	$Rd(n+1) \leftarrow Rd(n), Rd(0) \leftarrow 0$	Z,C,N,V	1
LSR	Rd	Logical Shift Right Rotate Left Through Carry	$Rd(n) \leftarrow Rd(n+1), Rd(7) \leftarrow 0$	Z,C,N,V	1
ROL	Rd		$Rd(0) \leftarrow C, Rd(n+1) \leftarrow Rd(n), C \leftarrow Rd(7)$	Z,C,N,V Z,C,N,V	
ROR	Rd	Rotate Right Through Carry	$Rd(7) \leftarrow C, Rd(n) \leftarrow Rd(n+1), C \leftarrow Rd(0)$ $Rd(n) \leftarrow Rd(n+1), n=0:6$		1
ASR SWAP	Rd Rd	Arithmetic Shift Right		Z,C,N,V	1
		Swap Nibbles	Rd(3:0)←Rd(7:4),Rd(7:4)←Rd(3:0)	None	
BSET BCLR	s	Flag Set	$SREG(s) \leftarrow 1$	SREG(s)	1
BST	S	Flag Clear	$SREG(s) \leftarrow 0$	SREG(s)	1
BLD	Rr, b Rd, b	Bit Store from Register to T Bit load from T to Register	$\begin{array}{c} T \leftarrow Rr(b) \\ Rd(b) \leftarrow T \end{array}$	None	1
SEC	Ku, D	Set Carry	$C \leftarrow 1$	C	1
				C	1
CLC SEN		Clear Carry Set Negative Flag	$C \leftarrow 0$ N $\leftarrow 1$	N	1
CLN		Clear Negative Flag	$N \leftarrow 1$ $N \leftarrow 0$	N	1
SEZ	1	Set Zero Flag		Z	1
CLZ	+	Clear Zero Flag	$Z \leftarrow 1$ $Z \leftarrow 0$	Z	1
SEI		Clear Zero Flag Global Interrupt Enable		2	1
				1	
CLI SES		Global Interrupt Disable		S	1
		Set Signed Test Flag	S ← 1		
CLS		Clear Signed Test Flag	S ← 0	S V	1
SEV		Set Twos Complement Overflow.	$V \leftarrow 1$		1
CLV		Clear Twos Complement Overflow	$V \leftarrow 0$	V	1



Mnemonics	Operands	Description	Operation	Flags	#Clocks
SET		Set T in SREG	T ← 1	Т	1
CLT		Clear T in SREG	T ← 0	Т	1
SEH		Set Half Carry Flag in SREG	H ← 1	Н	1
CLH		Clear Half Carry Flag in SREG	H ← 0	Н	1
MCU CONTROL	INSTRUCTIONS		•		
NOP		No Operation		None	1
SLEEP		Sleep	(see specific descr. for Sleep function)	None	1
WDR		Watchdog Reset	(see specific descr. for WDR/timer)	None	1
BREAK		Break	For On-Chip Debug Only	None	N/A



7. Ordering Information

Speed (MHz)	Power Supply	Ordering Code ⁽²⁾	Package ⁽¹⁾	Operational Range
16	2.7V - 5.5V	ATmega32A-AU SATmega32A-AUR ⁽³⁾ ATmega32A-PU ATmega32A-MU ATmega32A-MUR ⁽³⁾	44A 44A 40P6 44M1 44M1	Industrial (-40°C to 85°C)
		ATmega32A-AN SATmega32A-ANR ⁽³⁾ ATmega32A-MN ATmega32A-MNR ⁽³⁾	44A 44A 44M1 44M1	Extended (-40°C to 105°C) ⁽⁴⁾

Notes: 1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.

2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.

3. Tape & Reel

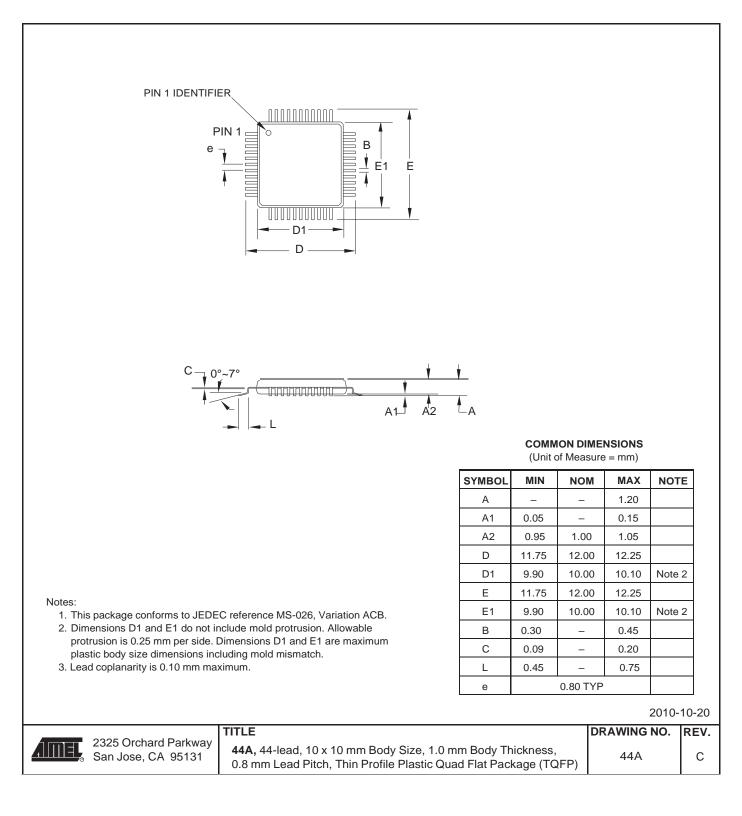
4. See Appendix A ATmega32A 105°C

	Package Type
44A	44-lead, 10 x 10 x 1.0 mm, Thin Profile Plastic Quad Flat Package (TQFP)
40P6	40-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)
44M1	44-pad, 7 x 7 x 1.0 mm, Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)



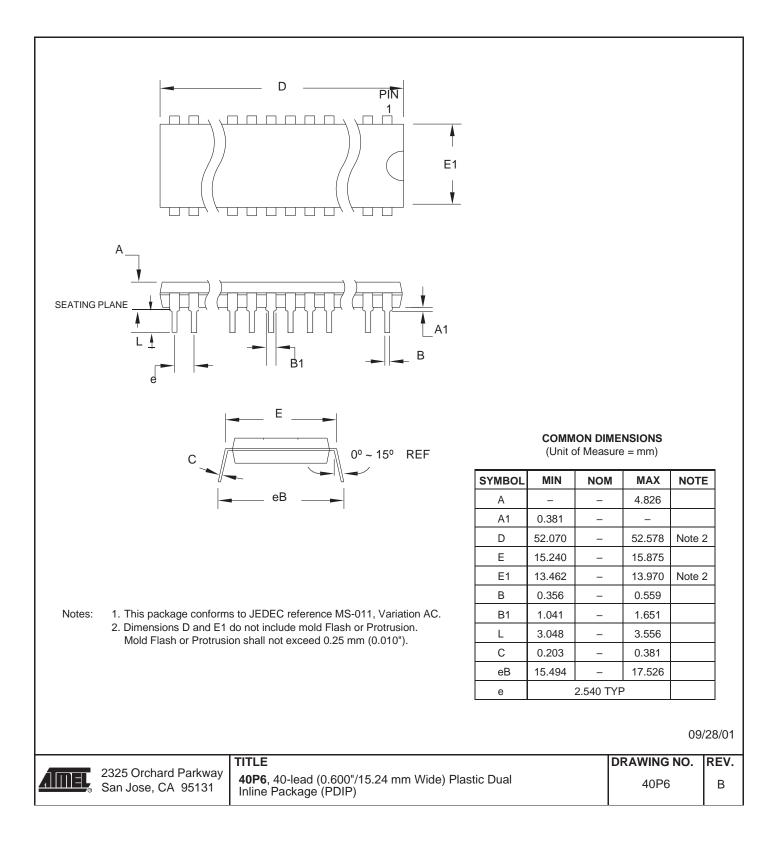
8. Packaging Information

8.1 44A



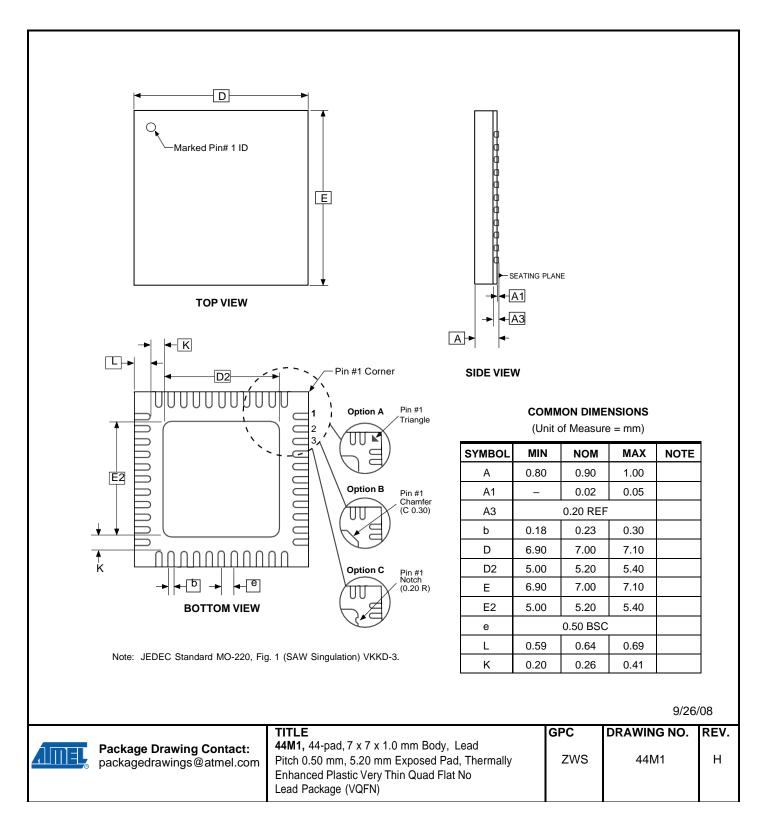


8.2 40P6





8.3 44M1





9. Errata

9.1 ATmega32A, rev. G to rev. I

- First Analog Comparator conversion may be delayed
- · Interrupts may be lost when writing the timer registers in the asynchronous timer
- IDCODE masks data from TDI input
- Reading EEPROM by using ST or STS to set EERE bit triggers unexpected interrupt request.

1. First Analog Comparator conversion may be delayed

If the device is powered by a slow rising V_{CC} , the first Analog Comparator conversion will take longer than expected on some devices.

Problem Fix/Workaround

When the device has been powered or reset, disable then enable the Analog Comparator before the first conversion.

2. Interrupts may be lost when writing the timer registers in the asynchronous timer The interrupt will be lost if a timer register that is synchronous timer clock is written when the

The interrupt will be lost if a timer register that is synchronous timer clock is written when the asynchronous Timer/Counter register (TCNTx) is 0x00.

Problem Fix/Workaround

Always check that the asynchronous Timer/Counter register neither have the value 0xFF nor 0x00 before writing to the asynchronous Timer Control Register (TCCRx), asynchronous Timer Counter Register (TCNTx), or asynchronous Output Compare Register (OCRx).

3. IDCODE masks data from TDI input

The JTAG instruction IDCODE is not working correctly. Data to succeeding devices are replaced by all-ones during Update-DR.

Problem Fix / Workaround

- If ATmega32A is the only device in the scan chain, the problem is not visible.
- Select the Device ID Register of the ATmega32A by issuing the IDCODE instruction or by entering the Test-Logic-Reset state of the TAP controller to read out the contents of its Device ID Register and possibly data from succeeding devices of the scan chain. Issue the BYPASS instruction to the ATmega32A while reading the Device ID Registers of preceding devices of the boundary scan chain.
- If the Device IDs of all devices in the boundary scan chain must be captured simultaneously, the ATmega32A must be the fist device in the chain.

4. Reading EEPROM by using ST or STS to set EERE bit triggers unexpected interrupt request.

Reading EEPROM by using the ST or STS command to set the EERE bit in the EECR register triggers an unexpected EEPROM interrupt request.

Problem Fix / Workaround

Always use OUT or SBI to set EERE in EECR.



10. Datasheet Revision History

Please note that the referring page numbers in this section are referred to this document. The referring revision in this section are referring to the document revision.

10.1 Rev. 8155C – 02/11

- 1. Updated the datashee according to the Atmel new brand style guide (new logo, last page, etc).
- 2. Inserted note in "Performing Page Erase by SPM" on page 259.
- 3. Note 6 and Note 7 below Table 27-2, "Two-wire Serial Bus Requirements," on page 301 have been removed.
- 4. Updated "Ordering Information" on page 340 to include Tape & Reel and 105°C devices.
- 5. Updated all "Typical Characteristics"

10.2 Rev. 8155B - 07/09

- 1. Updated "Errata" on page 343.
- 2. Updated the last page with Atmel's new addresses.

10.3 Rev. 8155A - 06/08

1. Initial revision (Based on the ATmega32/L datasheet 2503N-AVR-06/08)

Changes done compared ATmega32/L datasheet 2503N-AVR-06/08:

- Updated description in "Stack Pointer" on page 11.
- All Electrical characteristics is moved to "Electrical Characteristics" on page 296.
- Register descriptions are moved to sub sections at the end of each chapter.
- Test limits of Reset Pull-up Resistor (R_{RST}) in "DC Characteristics" on page 296.
- New graphs in "Typical Characteristics" on page 306.
- New "Ordering Information" on page 339.





Atmel Corporation

2325 Orchard Parkway San Jose, CA 95131 USA Tel: (+1)(408) 441-0311 Fax: (+1)(408) 487-2600 www.atmel.com

Atmel Asia Limited Unit 1-5 & 16, 19/F BEA Tower, Millennium City 5 418 Kwun Tong Road Kwun Tong, Kowloon HONG KONG Tel: (+852) 2245-6100 Fax: (+852) 2722-1369

Atmel Munich GmbH Business Campus Parkring 4 D-85748 Garching b. Munich GERMANY Tel: (+49) 89-31970-0 Fax: (+49) 89-3194621

Atmel Japan

9F, Tonetsu Shinkawa Bldg. 1-24-8 Shinkawa Chuo-ku, Tokyo 104-0033 JAPAN Tel: (+81)(3) 3523-3551 Fax: (+81)(3) 3523-7581

© 2011 Atmel Corporation. All rights reserved. / Rev. CORP0XXXX

Atmel[®], Atmel logo and combinations thereof, AVR[®] and others are registered trademarks or trademarks of Atmel Corporation or its subsidiaries. Other terms and product names may be trademarks of others.

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN THE ATMEL TERMS AND CONDITIONS OF SALES LOCATED ON THE ATMEL WEBSITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRCT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS AND PROF-ITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.