



Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



## Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

Email & Skype: info@chipsmall.com Web: www.chipsmall.com

Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China



## Introduction

The Atmel® 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 close to 1MIPS per MHz. This empowers system designer to optimize the device for power consumption versus processing speed.

## Features

- High-performance, Low-power Atmel AVR 8-bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions - Most Single-clock Cycle Execution
  - 32 × 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
  - 2Kbytes Internal SRAM
  - Write/Erase cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C<sup>(1)</sup>
  - 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
- Atmel QTouch® library support

- Capacitive touch buttons, sliders and wheels
- Atmel QTouch and QMatrix acquisition
- Up to 64 sense channels
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - 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 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, Standby, and Extended 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 1MHz, 3V, 25°C
  - Active: 0.6mA
  - Idle Mode: 0.2mA
  - Power-down Mode: < 1µA

## Table of Contents

---

Introduction.....	1
Features.....	1
1. Description.....	9
2. Configuration Summary.....	10
3. Ordering Information.....	11
4. Block Diagram.....	12
5. Pin Configurations.....	13
5.1. V <sub>CC</sub> .....	14
5.2. GND.....	14
5.3. PortA (PA7:PA0).....	14
5.4. Port B (PB7:PB0).....	15
5.5. Port C (PC7:PC0).....	15
5.6. Port D (PD7:PD0).....	15
5.7. RESET.....	15
5.8. XTAL1.....	16
5.9. XTAL2.....	16
5.10. AV <sub>CC</sub> .....	16
5.11. AREF.....	16
6. Resources.....	17
7. Data Retention.....	18
8. About Code Examples.....	19
9. Capacitive Touch Sensing.....	20
10. AVR CPU Core.....	21
10.1. Overview.....	21
10.2. ALU – Arithmetic Logic Unit.....	22
10.3. Status Register.....	22
10.4. General Purpose Register File.....	24
10.5. Stack Pointer.....	25
10.6. Instruction Execution Timing.....	26
10.7. Reset and Interrupt Handling.....	27
11. AVR Memories.....	29
11.1. Overview.....	29
11.2. In-System Reprogrammable Flash Program Memory.....	29
11.3. SRAM Data Memory.....	30
11.4. EEPROM Data Memory.....	31

11.5. I/O Memory.....	32
11.6. Register Description.....	32
<b>12. System Clock and Clock Options.....</b>	<b>39</b>
12.1. Clock Systems and their Distribution.....	39
12.2. Clock Sources.....	40
12.3. Default Clock Source.....	41
12.4. Crystal Oscillator.....	41
12.5. Low-frequency Crystal Oscillator.....	42
12.6. External RC Oscillator.....	43
12.7. Calibrated Internal RC Oscillator.....	43
12.8. External Clock.....	44
12.9. Timer/Counter Oscillator.....	45
12.10. Register Description.....	45
<b>13. Power Management and Sleep Modes.....</b>	<b>47</b>
13.1. Sleep Modes.....	47
13.2. Idle Mode.....	48
13.3. ADC Noise Reduction Mode.....	48
13.4. Power-down Mode.....	48
13.5. Power-save Mode.....	48
13.6. Standby Mode.....	49
13.7. Extended Standby Mode.....	49
13.8. Minimizing Power Consumption.....	49
13.9. Register Description.....	51
<b>14. System Control and Reset.....</b>	<b>53</b>
14.1. Resetting the AVR.....	53
14.2. Reset Sources.....	53
14.3. Internal Voltage Reference.....	57
14.4. Watchdog Timer.....	57
14.5. Register Description.....	58
<b>15. Interrupts.....</b>	<b>62</b>
15.1. Interrupt Vectors in ATmega32A.....	62
15.2. Register Description.....	66
<b>16. External Interrupts.....</b>	<b>69</b>
16.1. Register Description.....	69
<b>17. I/O Ports.....</b>	<b>74</b>
17.1. Overview.....	74
17.2. Ports as General Digital I/O.....	75
17.3. Alternate Port Functions.....	78
17.4. Register Description.....	88
<b>18. Timer/Counter0 and Timer/Counter1 Prescalers.....</b>	<b>102</b>
18.1. Overview.....	102
18.2. Internal Clock Source.....	102

18.3. Prescaler Reset.....	102
18.4. External Clock Source.....	102
18.5. Register Description.....	103
<b>19. 16-bit Timer/Counter1.....</b>	<b>105</b>
19.1. Features.....	105
19.2. Overview.....	105
19.3. Accessing 16-bit Registers.....	107
19.4. Timer/Counter Clock Sources.....	110
19.5. Counter Unit.....	110
19.6. Input Capture Unit.....	111
19.7. Output Compare Units.....	113
19.8. Compare Match Output Unit.....	115
19.9. Modes of Operation.....	116
19.10. Timer/Counter Timing Diagrams.....	123
19.11. Register Description.....	124
<b>20. 8-bit Timer/Counter2 with PWM and Asynchronous Operation.....</b>	<b>140</b>
20.1. Features.....	140
20.2. Overview.....	140
20.3. Timer/Counter Clock Sources.....	141
20.4. Counter Unit.....	141
20.5. Output Compare Unit.....	142
20.6. Compare Match Output Unit.....	144
20.7. Modes of Operation.....	145
20.8. Timer/Counter Timing Diagrams.....	149
20.9. Asynchronous Operation of the Timer/Counter.....	150
20.10. Timer/Counter Prescaler.....	152
20.11. Register Description.....	152
<b>21. 8-bit Timer/Counter0 with PWM.....</b>	<b>162</b>
21.1. Features.....	162
21.2. Overview.....	162
21.3. Timer/Counter Clock Sources.....	163
21.4. Counter Unit.....	163
21.5. Output Compare Unit.....	164
21.6. Compare Match Output Unit.....	166
21.7. Modes of Operation.....	167
21.8. Timer/Counter Timing Diagrams.....	171
21.9. Register Description.....	172
<b>22. SPI – Serial Peripheral Interface.....</b>	<b>180</b>
22.1. Features.....	180
22.2. Overview.....	180
22.3. $\overline{SS}$ Pin Functionality.....	183
22.4. Data Modes.....	184
22.5. Register Description.....	185

23. USART - Universal Synchronous and Asynchronous serial Receiver and Transmitter.....	190
23.1. Features.....	190
23.2. Overview.....	190
23.3. Clock Generation.....	192
23.4. Frame Formats.....	195
23.5. USART Initialization.....	196
23.6. Data Transmission – The USART Transmitter.....	197
23.7. Data Reception – The USART Receiver.....	199
23.8. Asynchronous Data Reception.....	203
23.9. Multi-Processor Communication Mode.....	205
23.10. Accessing UBRRH/UCSRC Registers.....	206
23.11. Register Description.....	208
23.12. Examples of Baud Rate Setting.....	217
24. TWI - Two-wire Serial Interface.....	221
24.1. Features.....	221
24.2. Overview.....	221
24.3. Two-Wire Serial Interface Bus Definition.....	223
24.4. Data Transfer and Frame Format.....	224
24.5. Multi-master Bus Systems, Arbitration and Synchronization.....	227
24.6. Using the TWI.....	228
24.7. Multi-master Systems and Arbitration.....	245
24.8. Register Description.....	246
25. AC - Analog Comparator.....	253
25.1. Overview.....	253
25.2. Analog Comparator Multiplexed Input.....	253
25.3. Register Description.....	254
26. ADC - Analog to Digital Converter.....	258
26.1. Features.....	258
26.2. Overview.....	258
26.3. Starting a Conversion.....	260
26.4. Prescaling and Conversion Timing.....	261
26.5. Changing Channel or Reference Selection.....	264
26.6. ADC Noise Canceler.....	265
26.7. ADC Conversion Result.....	269
26.8. Register Description.....	271
27. JTAG Interface and On-chip Debug System.....	282
27.1. Features.....	282
27.2. Overview.....	282
27.3. TAP – Test Access Port.....	283
27.4. TAP Controller.....	284
27.5. Using the Boundary-scan Chain.....	285
27.6. Using the On-chip Debug System.....	285
27.7. On-chip Debug Specific JTAG Instructions.....	286

27.8. Using the JTAG Programming Capabilities.....	286
27.9. Bibliography.....	287
27.10. IEEE 1149.1 (JTAG) Boundary-scan.....	287
27.11. Data Registers.....	288
27.12. Boundry-scan Specific JTAG Instructions.....	290
27.13. Boundary-scan Chain.....	291
27.14. ATmega32A Boundary-scan Order.....	301
27.15. Boundary-scan Description Language Files.....	307
27.16. Register Description.....	308
<b>28. BTLDR - Boot Loader Support – Read-While-Write Self-Programming.....</b>	<b>311</b>
28.1. Features.....	311
28.2. Overview.....	311
28.3. Application and Boot Loader Flash Sections.....	311
28.4. Read-While-Write and No Read-While-Write Flash Sections.....	312
28.5. Boot Loader Lock Bits.....	314
28.6. Entering the Boot Loader Program.....	315
28.7. Addressing the Flash During Self-Programming.....	316
28.8. Self-Programming the Flash.....	317
28.9. Register Description.....	324
<b>29. Memory Programming.....</b>	<b>327</b>
29.1. Program and Data Memory Lock Bits.....	327
29.2. Fuse Bits.....	328
29.3. Signature Bytes.....	330
29.4. Signature Bytes.....	330
29.5. Calibration Byte.....	330
29.6. Parallel Programming Parameters, Pin Mapping, and Commands.....	330
29.7. Parallel Programming.....	333
29.8. Serial Downloading.....	341
29.9. Serial Programming Pin Mapping.....	341
29.10. Programming Via the JTAG Interface.....	345
<b>30. Electrical Characteristics.....</b>	<b>359</b>
30.1. DC Characteristics.....	359
30.2. Speed Grades.....	362
30.3. Clock Characteristics.....	362
30.4. System and Reset Characteristics.....	363
30.5. Two-wire Serial Interface Characteristics.....	363
30.6. SPI Timing Characteristics.....	365
30.7. ADC Characteristics.....	367
<b>31. Typical Characteristics.....</b>	<b>371</b>
31.1. Active Supply Current.....	371
31.2. Idle Supply Current.....	374
31.3. Power-down Supply Current.....	377
31.4. Power-save Supply current.....	378
31.5. Standby Supply Current.....	379
31.6. Pin Pull-up.....	379



31.7. Pin Driver Strength.....	381
31.8. Pin Thresholds and Hysteresis.....	383
31.9. BOD Thresholds and Analog Comparator Offset.....	386
31.10. Internal Oscillator Speed.....	388
31.11. Current Consumption of Peripheral Units.....	394
31.12. Current Consumption in Reset and Reset Pulsewidth.....	397
<b>32. Register Summary.....</b>	<b>399</b>
<b>33. Instruction Set Summary.....</b>	<b>401</b>
<b>34. Packaging Information.....</b>	<b>406</b>
34.1. 44-pin TQFP.....	406
34.2. 40-pin PDIP.....	407
34.3. 44-pin VQFN.....	408
<b>35. Errata.....</b>	<b>409</b>
35.1. ATmega32A, rev. J to rev. K.....	409
35.2. ATmega32A, rev. G to rev. I.....	410
<b>36. Datasheet Revision History.....</b>	<b>412</b>
36.1. 8155I - 08/2016.....	412
36.2. 8155H - 08/2016.....	412
36.3. 8155G - 10/2015.....	412
36.4. 8155F - 08/2015.....	412
36.5. 8155E - 02/2014.....	412
36.6. 8155D – 10/2013.....	412
36.7. 8155C - 02/2011.....	412
36.8. 8155B – 07/2009.....	413
36.9. 8155A – 06/2008.....	413

## 1. Description

The 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: 32Kbytes of In-System Programmable Flash Program memory with Read-While-Write capabilities, 1024bytes EEPROM, 2048bytes SRAM, 32 general purpose I/O lines, 32 general purpose working registers, a JTAG interface for Boundary-scan, 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 On-chip 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 cost-effective solution to many embedded control applications.

The Atmel AVR ATmega32A 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. Configuration Summary

Features	ATmega32A
Pin count	44
Flash (KB)	32
SRAM (KB)	2
EEPROM (KB)	1
General Purpose I/O pins	32
SPI	1
TWI (I <sup>2</sup> C)	1
USART	1
ADC	10-bit, up to 76.9ksps (15ksps at max resolution)
ADC channels	8
AC propagation delay	Typ 400ns
8-bit Timer/Counters	2
16-bit Timer/Counters	1
PWM channels	4
RC Oscillator	+/-3%
VREF Bandgap	
Operating voltage	2.7 - 5.5V
Max operating frequency	16MHz
Temperature range	-55°C to +125°C
JTAG	Yes

### 3. Ordering Information

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

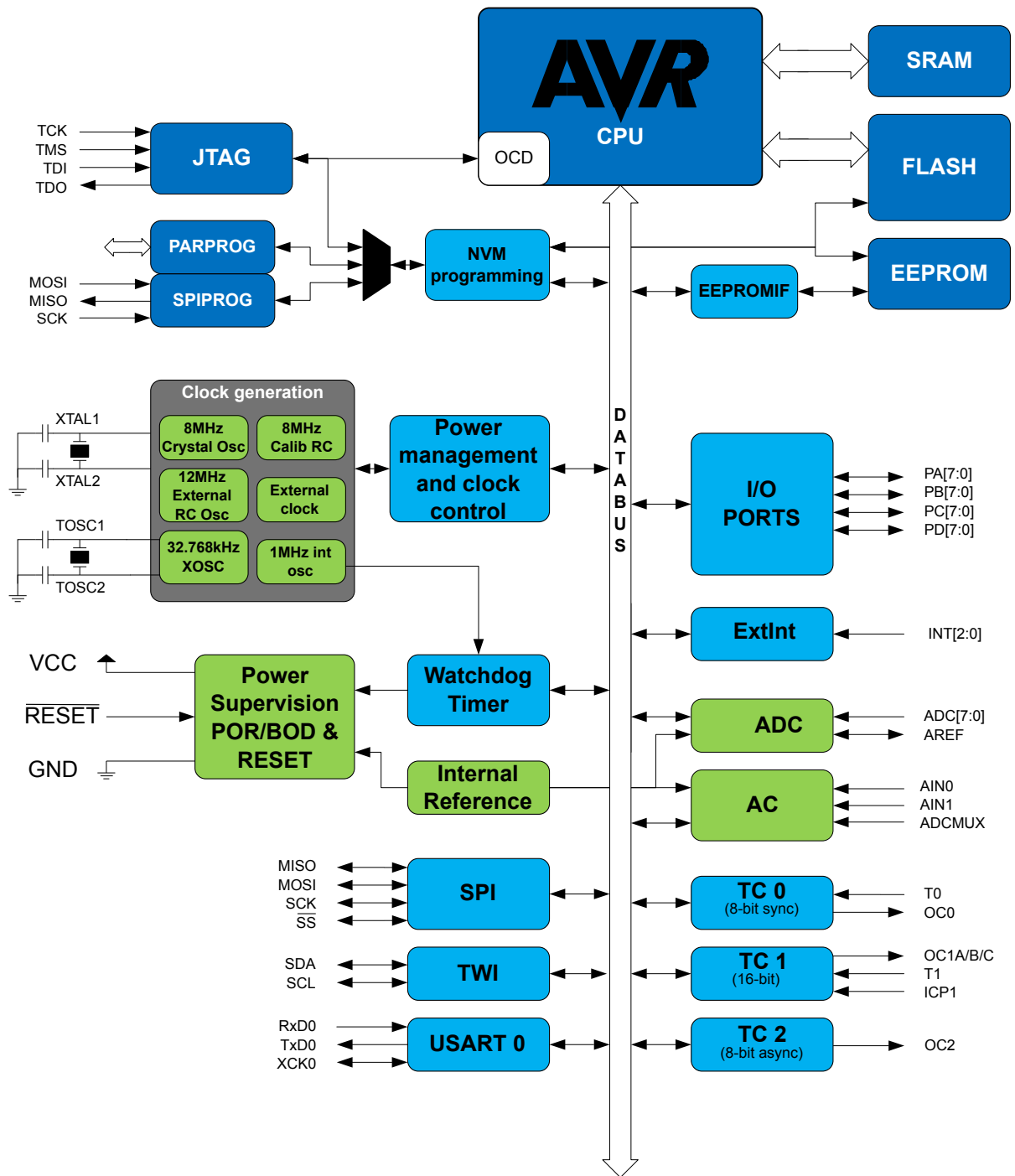
**Note:**

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 and Reel
4. See characterization specifications at 105°C

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

## 4. Block Diagram

Figure 4-1. Block Diagram



# 5. Pin Configurations

Figure 5-1. Pinout TQFP ATmega32A

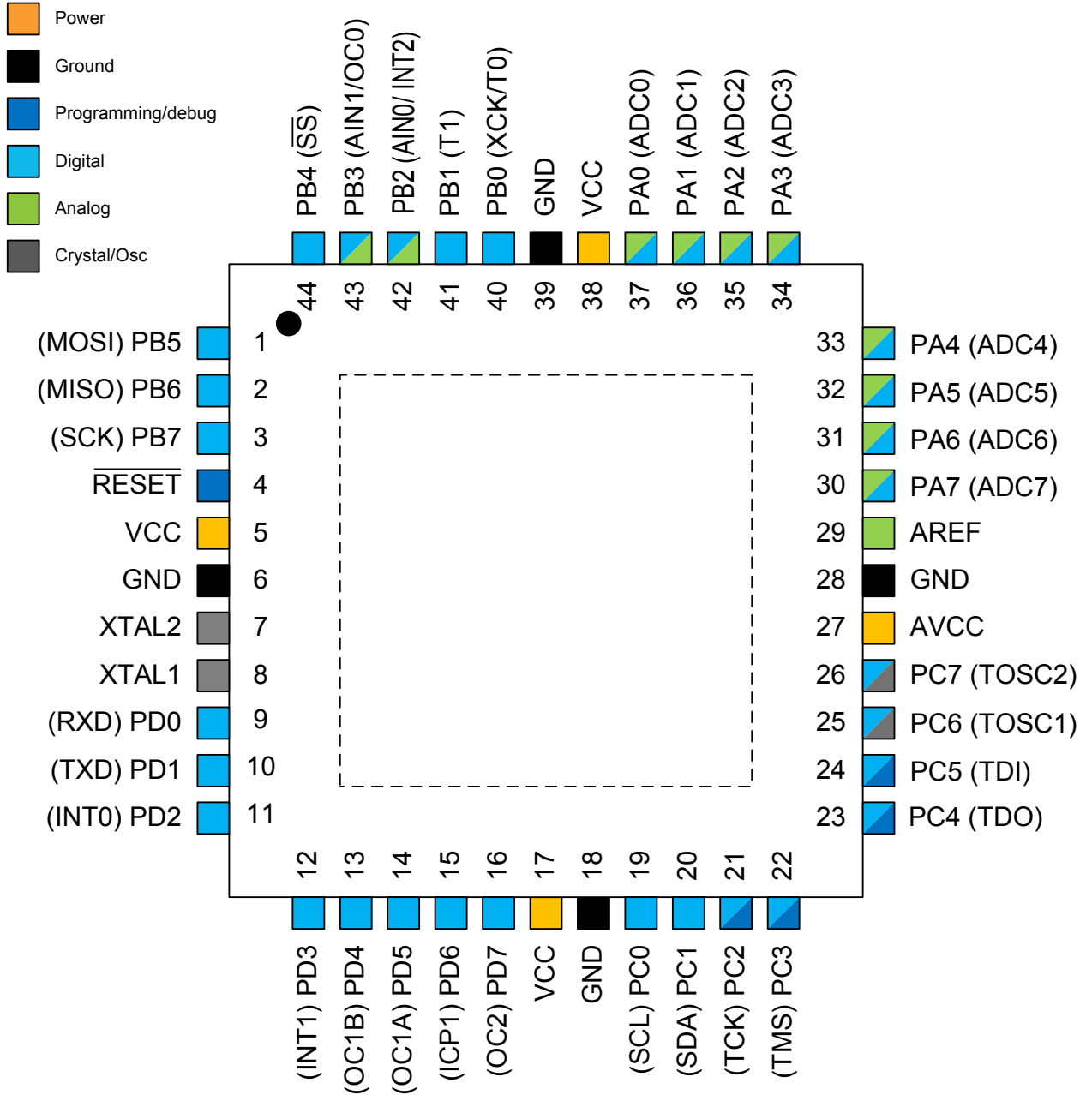
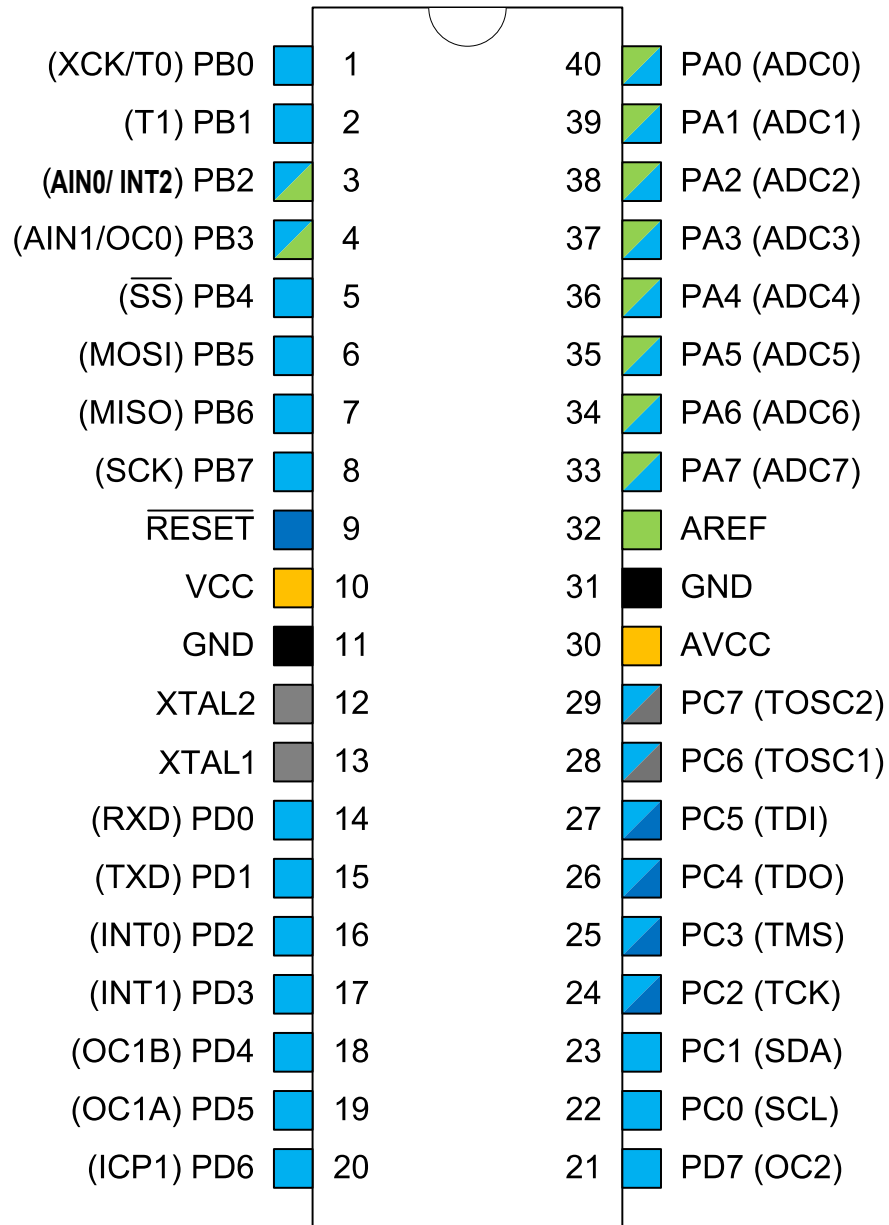


Figure 5-2. Pinout PDIP ATmega32A



### 5.1. V<sub>CC</sub>

Digital supply voltage.

### 5.2. GND

Ground.

### 5.3. PortA (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 symmetrical 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 tristated when a reset condition becomes active, even if the clock is not running.

#### 5.4. 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 tristated 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 in *Alternate Functions of Port B*.

##### Related Links

[Alternate Functions of Port B](#) on page 81

#### 5.5. 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 tristated 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 in *Alternate Functions of Port C*.

##### Related Links

[Alternate Functions of Port C](#) on page 84

#### 5.6. 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 tristated 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 in *Alternate Functions of Port D*.

##### Related Links

[Alternate Functions of Port D](#) on page 86

#### 5.7. 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 *System and Reset Characteristics*. Shorter pulses are not guaranteed to generate a reset.



## Related Links

[System and Reset Characteristics](#) on page 363

### 5.8. XTAL1

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

### 5.9. XTAL2

Output from the inverting Oscillator amplifier.

### 5.10. AV<sub>CC</sub>

AV<sub>CC</sub> is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to V<sub>CC</sub>, even if the ADC is not used. If the ADC is used, it should be connected to V<sub>CC</sub> through a low-pass filter.

### 5.11. AREF

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

## 6. Resources

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

## 7. 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.

## 8. About Code Examples

This datasheet contains simple code examples that briefly show how to use various parts of the device. These code examples assume that the part specific header file is included before compilation. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C compiler documentation for more details.

## 9. Capacitive Touch Sensing

The Atmel QTouch Library provides a simple to use solution to realize touch sensitive interfaces on most Atmel AVR microcontrollers. The QTouch Library includes support for the QTouch and QMatrix<sup>®</sup> acquisition methods.

Touch sensing can be added to any application by linking the appropriate Atmel QTouch Library for the AVR Microcontroller. This is done by using a simple set of APIs to define the touch channels and sensors, and then calling the touch sensing API's to retrieve the channel information and determine the touch sensor states.

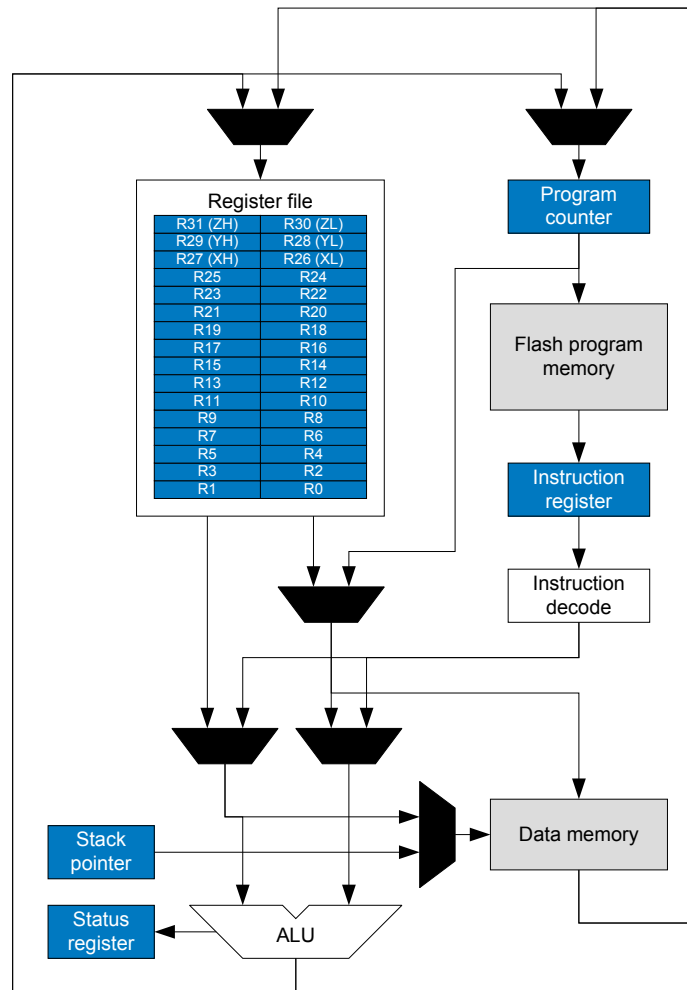
The QTouch Library is FREE and downloadable from the Atmel website at the following location: [www.atmel.com/qtouchlibrary](http://www.atmel.com/qtouchlibrary). For implementation details and other information, refer to the [Atmel QTouch Library User Guide](#) - also available for download from the Atmel website.

## 10. AVR CPU Core

### 10.1. Overview

This section discusses the Atmel AVR core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

Figure 10-1. Block Diagram of the AVR MCU Architecture



In order to maximize performance and parallelism, the AVR uses a Harvard architecture – with separate memories and buses for program and data. Instructions in the Program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the Program memory. This concept enables instructions to be executed in every clock cycle. The Program memory is In-System Reprogrammable Flash memory.

The fast-access Register File contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the Register File, the operation is executed, and the result is stored back in the Register File – in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of these address pointers can also be used

as an address pointer for look up tables in Flash Program memory. These added function registers are the 16-bit X-, Y-, and Z-register, described later in this section.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After an arithmetic operation, the Status Register is updated to reflect information about the result of the operation.

The Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most AVR instructions have a single 16-bit word format. Every Program memory address contains a 16- or 32-bit instruction.

Program Flash memory space is divided in two sections, the Boot program section and the Application program section. Both sections have dedicated Lock Bits for write and read/write protection. The SPM instruction that writes into the Application Flash memory section must reside in the Boot program section.

During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the Stack. The Stack is effectively allocated in the general data SRAM, and consequently the Stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the reset routine (before subroutines or interrupts are executed). The Stack Pointer SP is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.

The memory spaces in the AVR architecture are all linear and regular memory maps.

A flexible interrupt module has its control registers in the I/O space with an additional global interrupt enable bit in the Status Register. All interrupts have a separate Interrupt Vector in the Interrupt Vector table. The interrupts have priority in accordance with their Interrupt Vector position. The lower the Interrupt Vector address, the higher the priority.

The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, SPI, and other I/O functions. The I/O Memory can be accessed directly, or as the Data Space locations following those of the Register File, 0x20 - 0x5F.

## 10.2. ALU – Arithmetic Logic Unit

The high-performance Atmel AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit-functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See the “Instruction Set” section for a detailed description.

## 10.3. Status Register

The Status Register contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. Note that the Status Register is updated after all ALU operations, as specified in the Instruction Set Reference. This will in many cases remove the need for using the dedicated compare instructions, resulting in faster and more compact code.

The Status Register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt. This must be handled by software.

### 10.3.1. SREG – The AVR Status Register

When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O Registers as data space using LD and ST instructions, 0x20 must be added to these offset addresses.

**Name:** SREG

**Offset:** 0x3F

**Reset:** 0x00

**Property:** When addressing I/O Registers as data space the offset address is 0x5F

Bit	7	6	5	4	3	2	1	0
	I	T	H	S	V	N	Z	C
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

#### Bit 7 – I: Global Interrupt Enable

The Global Interrupt Enable bit must be set for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the Global Interrupt Enable Register is cleared, none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts. The I-bit can also be set and cleared by the application with the SEI and CLI instructions, as described in the Instruction Set Reference.

#### Bit 6 – T: Bit Copy Storage

The Bit Copy instructions BLD (Bit Load) and BST (Bit Store) use the T-bit as source or destination for the operated bit. A bit from a register in the Register File can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the Register File by the BLD instruction.

#### Bit 5 – H: Half Carry Flag

The Half Carry Flag H indicates a Half Carry in some arithmetic operations. Half Carry is useful in BCD arithmetic. See the “Instruction Set Description” for detailed information.

#### Bit 4 – S: Sign Bit, $S = N \oplus V$

The S-bit is always an exclusive or between the Negative Flag N and the Two’s Complement Overflow Flag V. See the “Instruction Set Description” for detailed information.

#### Bit 3 – V: Two’s Complement Overflow Flag

The Two’s Complement Overflow Flag V supports two’s complement arithmetics. See the “Instruction Set Description” for detailed information.

#### Bit 2 – N: Negative Flag

The Negative Flag N indicates a negative result in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.

#### Bit 1 – Z: Zero Flag

The Zero Flag Z indicates a zero result in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.



### Bit 0 – C: Carry Flag

The Carry Flag C indicates a Carry in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.

## 10.4. General Purpose Register File

The Register File is optimized for the Atmel AVR Enhanced RISC instruction set. In order to achieve the required performance and flexibility, the following input/output schemes are supported by the Register File:

- One 8-bit output operand and one 8-bit result input.
- Two 8-bit output operands and one 8-bit result input.
- Two 8-bit output operands and one 16-bit result input.
- One 16-bit output operand and one 16-bit result input.

The following figure shows the structure of the 32 general purpose working registers in the CPU.

**Figure 10-2. AVR CPU General Purpose Working Registers**

	7	0	Addr.	
General Purpose Working Registers	R0		0x00	
	R1		0x01	
	R2		0x02	
	...			
	R13		0x0D	
	R14		0x0E	
	R15		0x0F	
	R16		0x10	
	R17		0x11	
	...			
	R26		0x1A	X-register Low Byte
	R27		0x1B	X-register High Byte
	R28		0x1C	Y-register Low Byte
	R29		0x1D	Y-register High Byte
	R30		0x1E	Z-register Low Byte
	R31		0x1F	Z-register High Byte

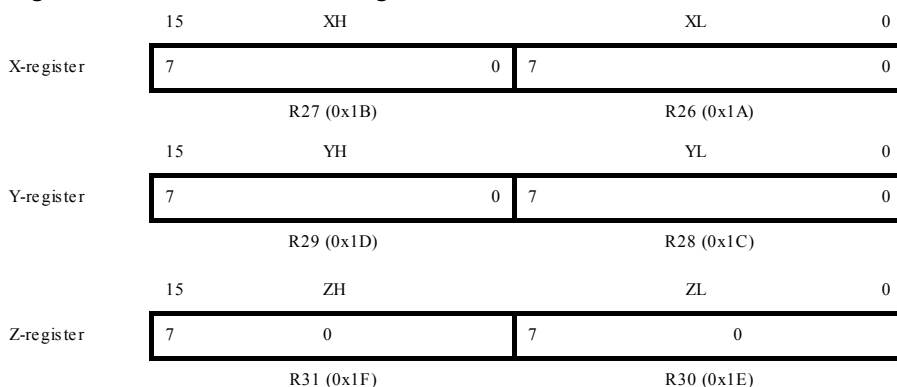
Most of the instructions operating on the Register File have direct access to all registers, and most of them are single cycle instructions.

As shown in the figure above, each register is also assigned a Data memory address, mapping them directly into the first 32 locations of the user Data Space. Although not being physically implemented as SRAM locations, this memory organization provides great flexibility in access of the registers, as the X-, Y-, and Z-pointer Registers can be set to index any register in the file.

### 10.4.1. The X-register, Y-register and Z-register

The registers R26:R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the Data Space. The three indirect address registers X, Y and Z are defined as described in the following figure.

**Figure 10-3. The X-, Y- and Z-Registers**



In the different addressing modes these address registers have functions as fixed displacement, automatic increment, and automatic decrement (see the Instruction Set Reference for details).

## 10.5. Stack Pointer

The Stack is mainly used for storing temporary data, for storing local variables and for storing return addresses after interrupts and subroutine calls. Note that the Stack is implemented as growing from higher to lower memory locations. The Stack Pointer Register always points to the top of the Stack. The Stack Pointer points to the data SRAM Stack area where the Subroutine and Interrupt Stacks are located. A Stack PUSH command will decrease the Stack Pointer.

The Stack in the data SRAM must be defined by the program before any subroutine calls are executed or interrupts are enabled. Initial Stack Pointer value equals the last address of the internal SRAM and the Stack Pointer must be set to point above start of the SRAM, refer to figure *Data Memory Map* in *SRAM Data Memory*.

The following table contains Stack Pointer details.

**Table 10-1. Stack Pointer instructions**

Instruction	Stack pointer	Description
PUSH	Decrement by 1	Data is pushed onto the stack
CALL ICALL RCALL	Decrement by 2	Return address is pushed onto the stack with a subroutine call or interrupt
POP	Increment by 1	Data is popped from the stack
RET RETI	Increment by 2	Return address is popped from the stack with return from subroutine or return from interrupt

The AVR Stack Pointer is implemented as two 8-bit registers in the I/O space. The number of bits actually used is implementation dependent. Note that the data space in some implementations of the AVR architecture is so small that only SPL is needed. In this case, the SPH Register will not be present.