



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® picoPower® ATmega1284P 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 ATmega1284P achieves throughputs close to 1MIPS per MHz. This empowers system designer to optimize the device for power consumption versus processing speed.

Feature

High Performance, Low Power Atmel®AVR® 8-Bit Microcontroller Family

- Advanced RISC Architecture
 - 131 Powerful Instructions
 - Most Single Clock Cycle Execution
 - 32 x 8 General Purpose Working Registers
 - Fully Static Operation
 - Up to 20 MIPS Throughput at 20MHz
 - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
 - 128KBytes of In-System Self-Programmable Flash Program Memory
 - 4KBytes EEPROM
 - 16KBytes 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
- Atmel QTouch® Library Support
 - Capacitive Touch Buttons, Sliders and Wheels
 - QTouch and QMatrix acquisition
 - Up to 64 Sense Channels

- 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 Prescaler and Compare Mode
 - Two 16-bit Timer/Counters with Separate Prescaler, Compare Mode, and Capture Mode
 - Real Time Counter with Separate Oscillator
 - Eight PWM Channels
 - 8-channel 10-bit ADC
 - Differential Mode with Selectable Gain at 1×, 10× or 200×
 - One Byte-oriented 2-wire Serial Interface (Philips I²C compatible)
 - Two Programmable Serial USART
 - One Master/Slave SPI Serial Interface
 - Programmable Watchdog Timer with Separate On-chip Oscillator
 - On-chip Analog Comparator
 - Interrupt and Wake-up on Pin Change
- 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
 - 44-pad VQFN/QFN
- Operating Voltage:
 - 1.8 - 5.5V
- Speed Grades
 - 0 - 4MHz @ 1.8V - 5.5V
 - 0 - 10MHz @ 2.7V - 5.5V
 - 0 - 20MHz @ 4.5 - 5.5V
- Power Consumption at 1MHz, 1.8V, 25°C
 - Active Mode: 0.4mA
 - Power-down Mode: 0.1µA
 - Power-save Mode: 0.6µA (Including 32kHz RTC)

Note:

1. Refer to *Data Retention*

Related Links

[Data Retention](#) on page 18

Table of Contents

Introduction.....	1
Feature.....	1
1. Description.....	9
2. Configuration Summary.....	10
3. Ordering Information	11
4. Block Diagram.....	12
5. Pin Configurations.....	13
5.1. Pinout.....	13
5.2. Pin Descriptions.....	14
6. I/O Multiplexing.....	16
7. General Information.....	18
7.1. Resources.....	18
7.2. Data Retention.....	18
7.3. About Code Examples.....	18
7.4. Capacitive Touch Sensing.....	18
8. AVR CPU Core.....	19
8.1. Overview.....	19
8.2. ALU – Arithmetic Logic Unit.....	20
8.3. Status Register.....	20
8.4. General Purpose Register File.....	22
8.5. Stack Pointer.....	23
8.6. Accessing 16-bit Registers.....	25
8.7. Instruction Execution Timing.....	26
8.8. Reset and Interrupt Handling.....	26
9. AVR Memories.....	29
9.1. Overview.....	29
9.2. In-System Reprogrammable Flash Program Memory.....	29
9.3. SRAM Data Memory.....	30
9.4. EEPROM Data Memory.....	31
9.5. I/O Memory.....	32
9.6. Register Description.....	33
10. System Clock and Clock Options.....	42
10.1. Clock Systems and Their Distribution.....	42
10.2. Clock Sources.....	43
10.3. Low Power Crystal Oscillator.....	45

10.4.	Full Swing Crystal Oscillator.....	46
10.5.	Low Frequency Crystal Oscillator.....	47
10.6.	Calibrated Internal RC Oscillator.....	48
10.7.	128kHz Internal Oscillator.....	49
10.8.	External Clock.....	50
10.9.	Timer/Counter Oscillator.....	51
10.10.	Clock Output Buffer.....	51
10.11.	System Clock Prescaler.....	51
10.12.	Register Description.....	52
11.	PM - Power Management and Sleep Modes.....	56
11.1.	Overview.....	56
11.2.	Sleep Modes.....	56
11.3.	BOD Disable.....	57
11.4.	Idle Mode.....	57
11.5.	ADC Noise Reduction Mode.....	57
11.6.	Power-Down Mode.....	58
11.7.	Power-Save Mode.....	58
11.8.	Standby Mode.....	59
11.9.	Extended Standby Mode.....	59
11.10.	Power Reduction Registers.....	59
11.11.	Minimizing Power Consumption.....	59
11.12.	Register Description.....	61
12.	SCRST - System Control and Reset.....	68
12.1.	Resetting the AVR.....	68
12.2.	Reset Sources.....	68
12.3.	Power-on Reset.....	69
12.4.	External Reset.....	70
12.5.	Brown-out Detection.....	70
12.6.	Watchdog System Reset.....	71
12.7.	Internal Voltage Reference.....	71
12.8.	Watchdog Timer.....	72
12.9.	Register Description.....	74
13.	Interrupts.....	78
13.1.	Overview.....	78
13.2.	Interrupt Vectors in ATmega1284P.....	78
13.3.	Register Description.....	81
14.	External Interrupts.....	84
14.1.	EXINT - External Interrupts.....	84
15.	I/O-Ports.....	96
15.1.	Overview.....	96
15.2.	Ports as General Digital I/O.....	97
15.3.	Alternate Port Functions.....	100
15.4.	Register Description.....	113

16. TC0 - 8-bit Timer/Counter0 with PWM.....	128
16.1. Features.....	128
16.2. Overview.....	128
16.3. Timer/Counter Clock Sources.....	130
16.4. Counter Unit.....	130
16.5. Output Compare Unit.....	131
16.6. Compare Match Output Unit.....	133
16.7. Modes of Operation.....	134
16.8. Timer/Counter Timing Diagrams.....	138
16.9. Register Description.....	140
17. TC1/3 - 16-bit Timer/Counter1/3 with PWM.....	153
17.1. Overview.....	153
17.2. Features.....	153
17.3. Block Diagram.....	153
17.4. Definitions.....	154
17.5. Registers.....	155
17.6. Accessing 16-bit Timer/Counter Registers.....	155
17.7. Timer/Counter Clock Sources.....	158
17.8. Counter Unit.....	158
17.9. Input Capture Unit.....	159
17.10. Output Compare Units.....	162
17.11. Compare Match Output Unit.....	164
17.12. Modes of Operation.....	165
17.13. Timer/Counter Timing Diagrams.....	173
17.14. Register Description.....	174
18. Timer/Counter 0, 1, 3 Prescalers.....	199
18.1. Internal Clock Source.....	199
18.2. Prescaler Reset.....	199
18.3. External Clock Source.....	199
18.4. Register Description.....	200
19. TC2 - 8-bit Timer/Counter2 with PWM and Asynchronous Operation.....	202
19.1. Features.....	202
19.2. Overview.....	202
19.3. Timer/Counter Clock Sources.....	204
19.4. Counter Unit.....	204
19.5. Output Compare Unit.....	205
19.6. Compare Match Output Unit.....	207
19.7. Modes of Operation.....	208
19.8. Timer/Counter Timing Diagrams.....	212
19.9. Asynchronous Operation of Timer/Counter2.....	213
19.10. Timer/Counter Prescaler.....	215
19.11. Register Description.....	215
20. SPI – Serial Peripheral Interface.....	228
20.1. Features.....	228

20.2. Overview.....	228
20.3. SS Pin Functionality.....	232
20.4. Data Modes.....	232
20.5. Register Description.....	233
21. USART - Universal Synchronous Asynchronous Receiver Transceiver.....	238
21.1. Features.....	238
21.2. Overview.....	238
21.3. Block Diagram.....	238
21.4. Clock Generation.....	239
21.5. Frame Formats.....	242
21.6. USART Initialization.....	243
21.7. Data Transmission – The USART Transmitter.....	244
21.8. Data Reception – The USART Receiver.....	246
21.9. Asynchronous Data Reception.....	249
21.10. Multi-Processor Communication Mode.....	252
21.11. Examples of Baud Rate Setting.....	253
21.12. Register Description.....	256
22. USARTSPI - USART in SPI Mode.....	266
22.1. Features.....	266
22.2. Overview.....	266
22.3. Clock Generation.....	266
22.4. SPI Data Modes and Timing.....	267
22.5. Frame Formats.....	267
22.6. Data Transfer.....	269
22.7. AVR USART MSPIM vs. AVR SPI.....	270
22.8. Register Description.....	271
23. TWI - 2-wire Serial Interface.....	272
23.1. Features.....	272
23.2. Two-Wire Serial Interface Bus Definition.....	272
23.3. Data Transfer and Frame Format.....	273
23.4. Multi-master Bus Systems, Arbitration, and Synchronization.....	276
23.5. Overview of the TWI Module.....	278
23.6. Using the TWI.....	280
23.7. Transmission Modes.....	283
23.8. Multi-master Systems and Arbitration.....	301
23.9. Register Description.....	303
24. AC - Analog Comparator.....	311
24.1. Overview.....	311
24.2. Analog Comparator Multiplexed Input.....	311
24.3. Register Description.....	312
25. ADC - Analog to Digital Converter.....	316
25.1. Features.....	316
25.2. Overview.....	316
25.3. Starting a Conversion.....	318

25.4. Prescaling and Conversion Timing.....	319
25.5. Changing Channel or Reference Selection.....	322
25.6. ADC Noise Canceler.....	324
25.7. ADC Conversion Result.....	328
25.8. Register Description.....	330
26. JTAG Interface and On-chip Debug System.....	340
26.1. Features.....	340
26.2. Overview.....	340
26.3. TAP – Test Access Port.....	341
26.4. TAP Controller.....	342
26.5. Using the Boundary-scan Chain.....	343
26.6. Using the On-chip Debug System.....	343
26.7. On-chip Debug Specific JTAG Instructions.....	344
26.8. Using the JTAG Programming Capabilities.....	344
26.9. Bibliography.....	345
26.10. IEEE 1149.1 (JTAG) Boundary-scan.....	345
26.11. Data Registers.....	346
26.12. Boundry-scan Specific JTAG Instructions.....	347
26.13. Boundary-scan Chain.....	349
26.14. ATmega1284P Boundary-scan Order.....	352
26.15. Boundary-scan Description Language Files.....	354
26.16. Register Description.....	354
27. BTLDR - Boot Loader Support – Read-While-Write Self-Programming.....	359
27.1. Features.....	359
27.2. Overview.....	359
27.3. Application and Boot Loader Flash Sections.....	359
27.4. Read-While-Write and No Read-While-Write Flash Sections.....	360
27.5. Entering the Boot Loader Program.....	362
27.6. Boot Loader Lock Bits.....	363
27.7. Addressing the Flash During Self-Programming.....	364
27.8. Self-Programming the Flash.....	365
27.9. Register Description.....	373
28. MEMPROG- Memory Programming.....	376
28.1. Program And Data Memory Lock Bits.....	376
28.2. Fuse Bits.....	377
28.3. Signature Bytes.....	380
28.4. Calibration Byte.....	380
28.5. Serial Number.....	380
28.6. Page Size.....	380
28.7. Parallel Programming Parameters, Pin Mapping, and Commands.....	381
28.8. Parallel Programming.....	383
28.9. Serial Downloading.....	390
28.10. Programming Via the JTAG Interface.....	395
29. Electrical Characteristics.....	409
29.1. Absolute Maximum Ratings.....	409

29.2. DC Characteristics.....	409
29.3. Speed Grades.....	412
29.4. Clock Characteristics.....	412
29.5. System and Reset Characteristics.....	413
29.6. External interrupts characteristics.....	414
29.7. SPI Timing Characteristics.....	415
29.8. Two-wire Serial Interface Characteristics.....	416
29.9. ADC characteristics.....	418
29.10. Parallel Programming Characteristics.....	422
30. Typical Characteristics.....	424
30.1. Active Supply Current.....	424
30.2. Idle Supply Current.....	426
30.3. Supply Current of I/O Modules.....	428
30.4. Power-down Supply Current.....	429
30.5. Power-save Supply Current.....	430
30.6. Standby Supply Current.....	431
30.7. Pin Pull-Up.....	431
30.8. Pin Driver Strength.....	434
30.9. Pin Threshold and Hysteresis.....	436
30.10. BOD Threshold.....	438
30.11. Internal Oscillator Speed.....	440
30.12. Current Consumption of Peripheral Units.....	442
30.13. Current Consumption in Reset and Reset Pulse Width.....	445
31. Register Summary.....	447
32. Instruction Set Summary.....	451
33. Packaging Information.....	456
33.1. 40-pin PDIP.....	456
33.2. 44-pin TQFP.....	457
33.3. 44-pin VQFN.....	458
34. Datasheet Revision History.....	459
34.1. Rev. C – 10/2016.....	459
34.2. Rev. B – 08/2016.....	459
34.3. Rev. A – 05/2016.....	459
35. Errata.....	460
35.1. Rev. B.....	460

1. Description

The Atmel® ATmega1284P 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 ATmega1284P achieves throughputs close to 1MIPS per MHz. This empowers system designer to optimize the device for power consumption versus processing speed.

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 a 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 ATmega1284P provides the following features: 128Kbytes of In-System Programmable Flash with Read-While-Write capabilities, 4Kbytes EEPROM, 16Kbytes SRAM, 32 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), three flexible Timer/Counters with compare modes and PWM, two serial programmable USARTs, one byte-oriented 2-wire Serial Interface (I2C), a 8-channel 10-bit ADC with optional differential input stage with programmable gain, a programmable Watchdog Timer with internal Oscillator, an SPI serial port, IEEE std. 1149.1 compliant JTAG test interface, also used for accessing the On-chip Debug system and programming and six software selectable power saving modes. The Idle mode stops the CPU while allowing the 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 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.

Atmel offers the QTouch® library for embedding capacitive touch buttons, sliders and wheels functionality into AVR microcontrollers. The patented charge-transfer signal acquisition offers robust sensing and includes fully debounced reporting of touch keys and includes Adjacent Key Suppression® (AKS™) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop and debug your own touch applications.

The device is manufactured using Atmel's high density non-volatile 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 ATmega1284P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega1284P 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

The table below compares the device series of feature and pin compatible devices, providing a seamless migration path.

Table 2-1. Configuration Summary and Device Comparison

Features	ATmega164PA	ATmega324PA	ATmega644PA	ATmega1284P
Pin Count	40/44/49	40/44/49	40/44	40/44
Flash (Bytes)	16K	32K	64K	128K
SRAM (Bytes)	1K	2K	4K	16K
EEPROM (Bytes)	512	1K	2K	4K
General Purpose I/O Lines	32	32	32	32
SPI	1	1	1	1
TWI (I ² C)	1	1	1	1
USART	2	2	2	2
ADC	10-bit 15ksps	10-bit 15ksps	10-bit 15ksps	10-bit 15ksps
ADC Channels	8	8	8	8
Analog Comparator	1	1	1	1
8-bit Timer/Counters	2	2	2	2
16-bit Timer/Counters	1	1	1	2
PWM channels	6	6	6	8
Packages	PDIP TQFP VQFN/QFN DRQFN VFBGA	PDIP TQFP VQFN/QFN DRQFN VFBGA	PDIP TQFP VQFN/QFN	PDIP TQFP VQFN/QFN

3. Ordering Information

Speed [MHz] ⁽³⁾	Power Supply [V]	Ordering Code ⁽²⁾	Package ⁽¹⁾	Operational Range
20	1.8 - 5.5	ATmega1284P-AU	44A	Industrial (-40°C to 85°C)
		ATmega1284P-AUR ⁽⁴⁾	44A	
		ATmega1284P-PU	40P6	
		ATmega1284P-MU	44M1	
		ATmega1284P-MUR ⁽⁴⁾	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. Refer to *Speed Grades* for Speed vs. V_{CC}
4. Tape & Reel.

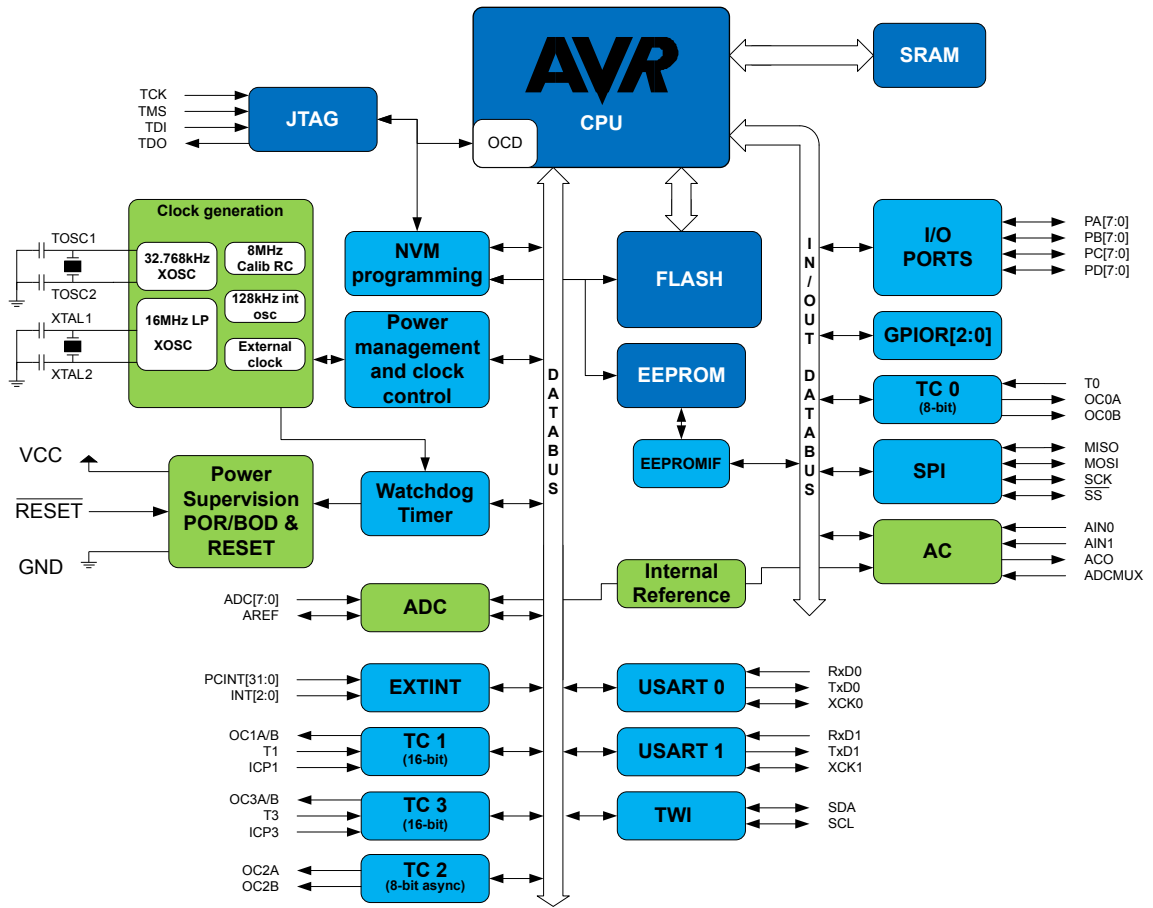
Package Type	
40P6	40-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)
44A	44-lead, Thin (1.0mm) Plastic Quad Flat Package (TQFP)
44M1	44-pad, 7 × 7 × 1.0mm body, lead pitch 0.50mm, Thermally Enhanced Plastic Very Thin Quad Flat No-Lead (VQFN)

Related Links

[Speed Grades](#) on page 412

4. Block Diagram









































Figure 4-1. Block Diagram



5. Pin Configurations

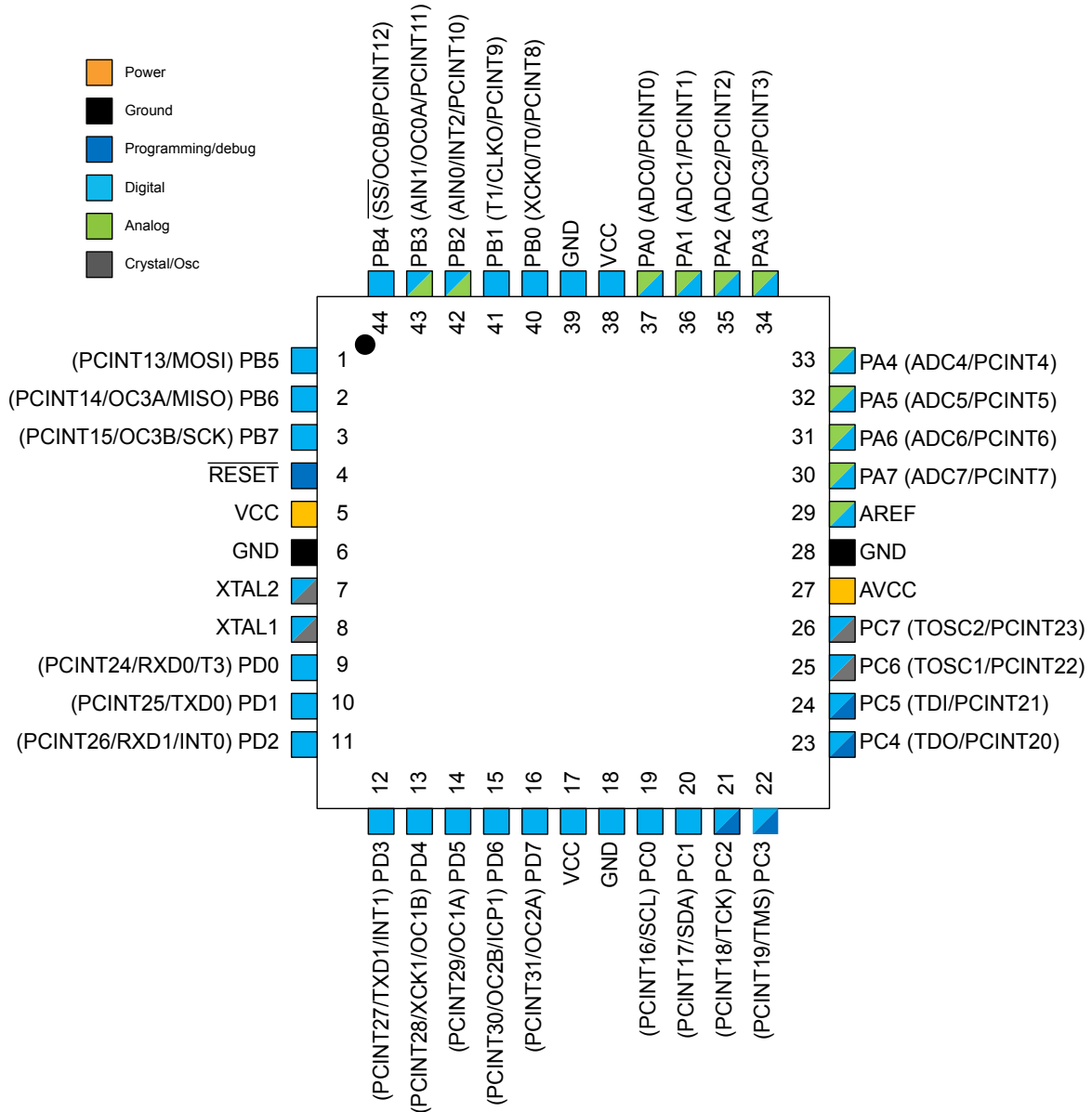
5.1. Pinout

5.1.1. PDIP

(PCINT8/XCK0/T0) PB0		1	40		PA0 (ADC0/PCINT0)
(PCINT9/CLKO/T1) PB1		2	39		PA1 (ADC1/PCINT1)
(PCINT10/INT2/AIN0) PB2		3	38		PA2 (ADC2/PCINT2)
(PCINT11/OC0A/AIN1) PB3		4	37		PA3 (ADC3/PCINT3)
(PCINT12/OC0B/ \overline{SS}) PB4		5	36		PA4 (ADC4/PCINT4)
(PCINT13/MOSI) PB5		6	35		PA5 (ADC5/PCINT5)
(PCINT14/OC3A/MISO) PB6		7	34		PA6 (ADC6/PCINT6)
(PCINT15/OC3B/SCK) PB7		8	33		PA7 (ADC7/PCINT7)
\overline{RESET}		9	32		AREF
VCC		10	31		GND
GND		11	30		AVCC
XTAL2		12	29		PC7 (TOSC2/PCINT23)
XTAL1		13	28		PC6 (TOSC1/PCINT22)
(PCINT24/RXD0/T3) PD0		14	27		PC5 (TDI/PCINT21)
(PCINT25/TXD0) PD1		15	26		PC4 (TDO/PCINT20)
(PCINT26/RXD1/INT0) PD2		16	25		PC3 (TMS/PCINT19)
(PCINT27/TXD1/INT1) PD3		17	24		PC2 (TCK/PCINT18)
(PCINT28/XCK1/OC1B) PD4		18	23		PC1 (SDA/PCINT17)
(PCINT29/OC1A) PD5		19	22		PC0 (SCL/PCINT16)
(PCINT30/OC2B/ICP1) PD6		20	21		PD7 (OC2A/PCINT31)

	Power
	Ground
	Programming/debug
	Digital
	Analog
	Crystal/Osc

5.1.2. TQFN and QFN



5.2. Pin Descriptions

5.2.1. VCC

Digital supply voltage.

5.2.2. GND

Ground.

5.2.3. Port A (PA[7:0])

This port serves as analog inputs to the Analog-to-digital Converter.

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

5.2.4. Port B (PB[7:0])

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

This port also serves the functions of various special features.

5.2.5. Port C (PC[7:0])

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

This port also serves the functions of the JTAG interface, along with special features.

5.2.6. Port D (PD[7:0])

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

This port also serves the functions of various special features.

5.2.7. $\overline{\text{RESET}}$

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. Shorter pulses are not guaranteed to generate a reset.

5.2.8. XTAL1

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

5.2.9. XTAL2

Output from the inverting Oscillator amplifier.

5.2.10. AVCC

AVCC is the supply voltage pin for Port A and the Analog-to-digital 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.

5.2.11. AREF

This is the analog reference pin for the Analog-to-digital Converter.

6. I/O Multiplexing

Each pin is by default controlled by the PORT as a general purpose I/O and alternatively it can be assigned to one of the peripheral functions.

The following table describes the peripheral signals multiplexed to the PORT I/O pins.

Table 6-1. PORT Function Multiplexing

32-pin TQFP/ QFN/ MLF Pin #	40-pin PDIP Pin #	PAD	EXTINT	PCINT	ADC/AC	OSC	T/C # 0	T/C # 1	USART	I2C	SPI	JTAG
1	6	PB[5]		PCINT13							MOSI	
2	7	PB[6]		PCINT14							MISO	
3	8	PB[7]		PCINT15							SCK	
4	9	RESET										
5	10	VCC										
6	11	GND										
7	12	XTAL2										
8	13	XTAL1										
9	14	PD[0]		PCINT24					RxD0			
10	15	PD[1]		PCINT25					TxD0			
11	16	PD[2]	INT0	PCINT26					RxD1			
12	17	PD[3]	INT1	PCINT27					TxD1			
13	18	PD[4]		PCINT28				OC1B	XCK1			
14	19	PD[5]		PCINT29				OC1A				
15	20	PD[6]		PCINT30			OC2B	ICP1				
16	21	PD[7]		PCINT31			OC2A					
17	-	VCC							RxD2		MISO1	
18	-	GND							TxD2		MOSI1	
19	22	PC[0]		PCINT16						SCL		
20	23	PC[1]		PCINT17						SDA		
21	24	PC[2]		PCINT18								TCK
22	25	PC[3]		PCINT19								TMS
23	26	PC[4]		PCINT20								TDO
24	27	PC[5]		PCINT21								TDI
25	28	PC[6]		PCINT22		TOSC1						
26	29	PC[7]		PCINT23		TOSC2						
27	30	AVCC										
28	31	GND										
29	32	AREF			AREF							
30	33	PA[7]		PCINT7	ADC7							
31	34	PA[6]		PCINT6	ADC6							
32	35	PA[5]		PCINT5	ADC5							
33	36	PA[4]		PCINT4	ADC4							
34	37	PA[3]		PCINT3	ADC3							

32-pin TQFP/ QFN/ MLF Pin #	40-pin PDIP Pin #	PAD	EXTINT	PCINT	ADC/AC	OSC	T/C # 0	T/C # 1	USART	I2C	SPI	JTAG
35	38	PA[2]		PCINT2	ADC2							
36	39	PA[1]		PCINT1	ADC1							
37	40	PA[0]		PCINT0	ADC0							
38	-	VCC								SDA1		
39	-	GND								SCL1		
40	1	PB[0]		PCINT8			T0		XCK0			
41	2	PB[1]		PCINT9		CLKO		T1				
42	3	PB[2]	INT2	PCINT10	AIN0							
43	4	PB[3]		PCINT11	AIN1		OC0A					
44	5	PB[4]		PCINT12			OC0B				SS	
-	-	GND										
-	-	GND										
-	-	GND										
-	-	GND										
-	-	GND										

7. General Information

7.1. Resources

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

7.2. 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.

7.3. About Code Examples

This documentation 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. Confirm with the C compiler documentation for more details.

For I/O Registers located in extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”.

7.4. Capacitive Touch Sensing

7.4.1. QTouch Library

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 Atmel QTouch and Atmel QMatrix® 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.

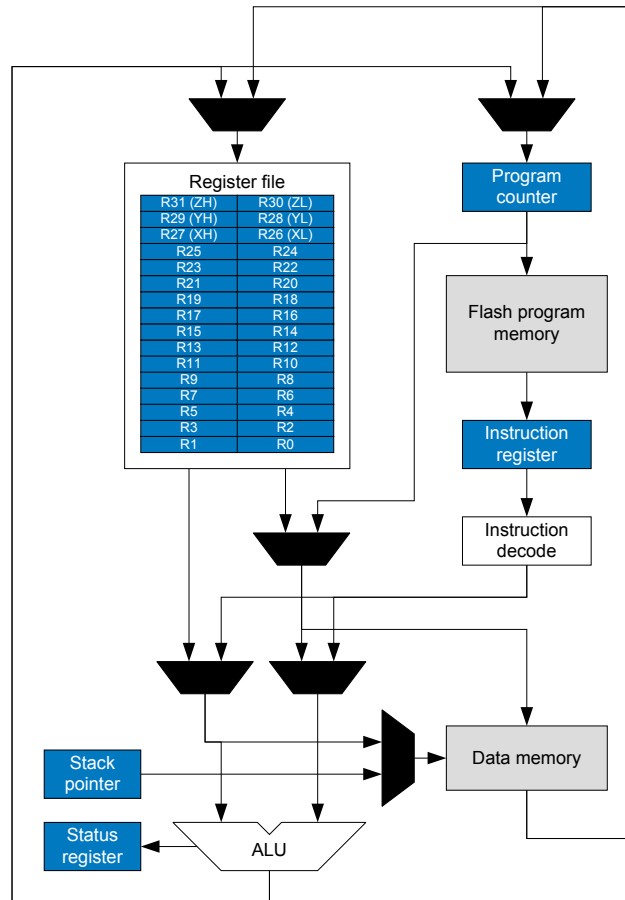
The QTouch Library is FREE and downloadable from the Atmel website at the following location: <http://www.atmel.com/technologies/touch/>. For implementation details and other information, refer to the [Atmel QTouch Library User Guide](#) - also available for download from the Atmel website.

8. AVR CPU Core

8.1. Overview

This section discusses the 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 8-1. Block Diagram of the AVR 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.

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. In addition, this device has Extended I/O space from 0x60 - 0xFF in SRAM where only the ST/STS/STD and LD/LDS/LDD instructions can be used.

8.2. ALU – Arithmetic Logic Unit

The high-performance 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 *Instruction Set Summary* section for a detailed description.

8.3. 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. 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.

8.3.1. Status Register

When addressing I/O Registers as data space using LD and ST instructions, the provided offset must be used. When using the I/O specific commands IN and OUT, the offset is reduced by 0x20, resulting in an I/O address offset within 0x00 - 0x3F.

Name: SREG

Offset: 0x5F

Reset: 0x00

Property: When addressing as I/O Register: address offset is 0x3F

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: 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 Flag is useful in BCD arithmetic. See the *Instruction Set Description* for detailed information.

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

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 arithmetic. 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.

8.4. General Purpose Register File

The Register File is optimized for the 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

Figure 8-2. 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, 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.

8.4.1. 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 figure.

Figure 8-3. 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).

8.5. 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. 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. See the table for Stack Pointer details.

Table 8-1. 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.

8.5.1. Stack Pointer Register Low and High byte

The SPL and SPH register pair represents the 16-bit value, SP. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + 0x01. For more details on reading and writing 16-bit registers, refer to [Accessing 16-bit Registers](#).

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. The device is a complex microcontroller with more peripheral units than can be supported within the 64 locations reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

Name: SPL and SPH

Offset: 0x5D

Reset: 0x10FF

Property: When addressing I/O Registers as data space the offset address is 0x3D

Bit	15	14	13	12	11	10	9	8
			SP13	SP12	SP11	SP10	SP9	SP8
Access	R	R	RW	RW	RW	RW	RW	RW
Reset	0	0	1	0	0	0	0	0
Bit	7	6	5	4	3	2	1	0
	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 – SPn: Stack Pointer Register

SPL and SPH are combined into SP.

8.5.2. Extended Z-pointer Register for ELPM/SPM

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. The device is a complex microcontroller with more peripheral units than can be supported within the 64 locations reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

Name: RAMPZ

Offset: 0x5B

Reset: 0x0

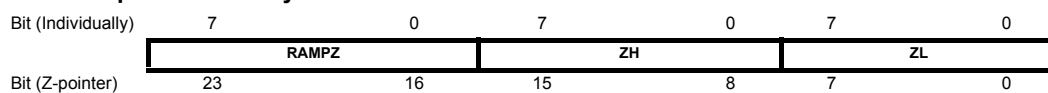
Property: When addressing I/O Registers as data space the offset address is 0x3B

Bit	7	6	5	4	3	2	1	0
	RAMPZ7	RAMPZ6	RAMPZ5	RAMPZ4	RAMPZ3	RAMPZ2	RAMPZ1	RAMPZ0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0

Bits 0, 1, 2, 3, 4, 5, 6, 7 – RAMPZn: Extended Z-pointer Register for ELPM/SPM

For ELPM/SPM instructions, the Z-pointer is a concatenation of RAMPZ, ZH, and ZL, as shown in the below figure. Note that LPM is not affected by the RAMPZ setting.

Figure 8-4. The Z-pointer used by ELPM and SPM



The actual number of bits is implementation dependent. Unused bits in an implementation will always read as zero. For compatibility with future devices, be sure to write these bits to zero.

8.6. Accessing 16-bit Registers

The AVR data bus is 8 bits wide, and so accessing 16-bit registers requires atomic operations. These registers must be byte-accessed using two read or write operations. 16-bit registers are connected to the 8-bit bus and a temporary register using a 16-bit bus.

For a write operation, the low byte of the 16-bit register must be written before the high byte. The low byte is then written into the temporary register. When the high byte of the 16-bit register is written, the temporary register is copied into the low byte of the 16-bit register in the same clock cycle.

For a read operation, the low byte of the 16-bit register must be read before the high byte. When the low byte register is read by the CPU, the high byte of the 16-bit register is copied into the temporary register in the same clock cycle as the low byte is read. When the high byte is read, it is then read from the temporary register.

This ensures that the low and high bytes of 16-bit registers are always accessed simultaneously when reading or writing the register.

Interrupts can corrupt the timed sequence if an interrupt is triggered and accesses the same 16-bit register during an atomic 16-bit read/write operation. To prevent this, interrupts can be disabled when writing or reading 16-bit registers.

The temporary registers can also be read and written directly from user software.