# ATmega328/P

## AVR® Microcontroller with picoPower® Technology

## Introduction

The picoPower® ATmega328/P 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 ATmega328/P achieves throughputs close to 1 MIPS per MHz. This empowers system designers to optimize the device for power consumption versus processing speed.

## Feature

High Performance, Low-Power 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 20 MHz
  - On-chip 2-cycle multiplier
- High Endurance Nonvolatile Memory Segments
  - 32K Bytes of in-system self-programmable Flash program memory
  - 1K Bytes EEPROM
  - 2K Bytes 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
- QTouch Library Support
  - Capacitive touch buttons, sliders and wheels
  - QTouch and QMatrix acquisition
  - Up to 64 sense channels
- Peripheral Features
  - Two 8-bit Timer/counters with separate prescaler and Compare mode
  - One 16-bit Timer/counter with separate prescaler, Compare mode, and Capture mode
  - Real time counter with separate oscillator
  - Six PWM channels

  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
    - Temperature measurement
  - 6-channel 10-bit ADC in PDIP package
    - Temperature measurement
  - Two master/slave SPI serial interface
  - One programmable serial USART
  - One byte-oriented 2-wire serial interface (Philips $I^2C$ compatible)
  - Programmable watchdog timer with separate on-chip oscillator
  - One on-chip analog comparator
  - Interrupt and wake-up on pin change
- Special Microcontroller Features
  - Power-on Reset and programmable Brown-out Detection
  - Internal calibrated 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
  - 23 Programmable I/O lines
  - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage:
  - 1.8 - 5.5V
- Temperature Range:
  - -40°C to 105°C
- Speed Grade:
  - ATmega328/P: 0 - 4 MHz @ 1.8 - 5.5V, 0 - 10 MHz @ 2.7 - 5.5V, 0 - 20 MHz @ 4.5 - 5.5V
- Power Consumption at 1 MHz, 1.8V, 25°C
  - Active mode: 0.2 mA
  - Power-Down mode: 0.1 µA
  - Power-Save mode: 0.75 µA (Including 32 kHz RTC)

# Table of Contents


Introduction....1

Feature....1

1. Description....9

2. Configuration Summary....10

3. Ordering Information....11
   - 3.1. ATmega328....11
   - 3.2. ATmega328P....11

4. Block Diagram....13

5. Pin Configurations....14
   - 5.1. Pinout....14
   - 5.2. Pin Descriptions....17

6. I/O Multiplexing....19

7. Resources....21

8. Data Retention....22

9. About Code Examples....23

10. Capacitive Touch Sensing....24
    - 10.1. QTouch Library....24

11. AVR CPU Core....25
    - 11.1. Overview....25
    - 11.2. Arithmetic Logic Unit (ALU)....26
    - 11.3. Status Register....26
    - 11.4. General Purpose Register File....29
    - 11.5. Stack Pointer....30
    - 11.6. Instruction Execution Timing....32
    - 11.7. Reset and Interrupt Handling....33

12. AVR Memories....36
    - 12.1. Overview....36
    - 12.2. In-System Reprogrammable Flash Program Memory....36
    - 12.3. SRAM Data Memory....37
    - 12.4. EEPROM Data Memory....38
    - 12.5. I/O Memory....39
    - 12.6. Register Description....40

13. System Clock and Clock Options..... 49
13.1. Clock Systems and Their Distribution..... 49
13.2. Clock Sources..... 50
13.3. Low-Power Crystal Oscillator..... 52
13.4. Full Swing Crystal Oscillator..... 54
13.5. Low-Frequency Crystal Oscillator..... 55
13.6. Calibrated Internal RC Oscillator..... 56
13.7. 128 kHz Internal Oscillator..... 57
13.8. External Clock..... 58
13.9. Timer/Counter Oscillator..... 59
13.10. Clock Output Buffer..... 59
13.11. System Clock Prescaler..... 59
13.12. Register Description..... 60

14. Power Management and Sleep Modes..... 64
14.1. Overview..... 64
14.2. Sleep Modes..... 64
14.3. BOD Disable..... 65
14.4. Idle Mode..... 65
14.5. ADC Noise Reduction Mode..... 65
14.6. Power-Down Mode..... 66
14.7. Power-Save Mode..... 66
14.8. Standby Mode..... 67
14.9. Extended Standby Mode..... 67
14.10. Power Reduction Register..... 67
14.11. Minimizing Power Consumption..... 67
14.12. Register Description..... 69

15. System Control and Reset..... 74
15.1. Resetting the AVR..... 74
15.2. Reset Sources..... 74
15.3. Power-on Reset..... 75
15.4. External Reset..... 76
15.5. Brown-out Detection..... 76
15.6. Watchdog System Reset..... 77
15.7. Internal Voltage Reference..... 77
15.8. Watchdog Timer..... 78
15.9. Register Description..... 80

16. Interrupts..... 84
16.1. Interrupt Vectors in ATmega328/P..... 84
16.2. Register Description..... 86

17. EXTINT - External Interrupts..... 89
17.1. Pin Change Interrupt Timing..... 89
17.2. Register Description..... 90

18. I/O-Ports..... 99

18.1. Overview ... 99
18.2. Ports as General Digital I/O ... 100
18.3. Alternate Port Functions ... 103
18.4. Register Description ... 115

19. 8-bit Timer/Counter0 (TC0) with PWM ... 127
19.1. Features ... 127
19.2. Overview ... 127
19.3. Timer/Counter Clock Sources ... 129
19.4. Counter Unit ... 129
19.5. Output Compare Unit ... 130
19.6. Compare Match Output Unit ... 132
19.7. Modes of Operation ... 134
19.8. Timer/Counter Timing Diagrams ... 138
19.9. Register Description ... 140

20. 16-bit Timer/Counter1 (TC1) with PWM ... 152
20.1. Overview ... 152
20.2. Features ... 152
20.3. Block Diagram ... 152
20.4. Definitions ... 153
20.5. Registers ... 154
20.6. Accessing 16-bit Timer/Counter Registers ... 154
20.7. Timer/Counter Clock Sources ... 157
20.8. Counter Unit ... 157
20.9. Input Capture Unit ... 158
20.10. Output Compare Units ... 160
20.11. Compare Match Output Unit ... 162
20.12. Modes of Operation ... 163
20.13. Timer/Counter 0, 1 Prescalers ... 171
20.14. Timer/Counter Timing Diagrams ... 171
20.15. Register Description ... 173

21. Timer/Counter 0, 1 Prescalers ... 186
21.1. Internal Clock Source ... 186
21.2. Prescaler Reset ... 186
21.3. External Clock Source ... 186
21.4. Register Description ... 188

22. 8-bit Timer/Counter2 (TC2) with PWM and Asynchronous Operation ... 190
22.1. Features ... 190
22.2. Overview ... 190
22.3. Timer/Counter Clock Sources ... 192
22.4. Counter Unit ... 192
22.5. Output Compare Unit ... 193
22.6. Compare Match Output Unit ... 195
22.7. Modes of Operation ... 196
22.8. Timer/Counter Timing Diagrams ... 200

22.9. Asynchronous Operation of Timer/Counter2 ..... 201
22.10. Timer/Counter Prescaler ..... 203
22.11. Register Description ..... 203

23. Serial Peripheral Interface (SPI) ..... 218
23.1. Features ..... 218
23.2. Overview ..... 218
23.3. $\overline{SS}$ Pin Functionality ..... 222
23.4. Data Modes ..... 222
23.5. Register Description ..... 223

24. Universal Synchronous Asynchronous Receiver Transceiver (USART) ..... 228
24.1. Features ..... 228
24.2. Overview ..... 228
24.3. Block Diagram ..... 228
24.4. Clock Generation ..... 229
24.5. Frame Formats ..... 232
24.6. USART Initialization ..... 233
24.7. Data Transmission – The USART Transmitter ..... 234
24.8. Data Reception – The USART Receiver ..... 236
24.9. Asynchronous Data Reception ..... 240
24.10. Multi-Processor Communication Mode ..... 243
24.11. Examples of Baud Rate Setting ..... 243
24.12. Register Description ..... 246

25. USART in SPI (USARTSPI) Mode ..... 256
25.1. Features ..... 256
25.2. Overview ..... 256
25.3. Clock Generation ..... 256
25.4. SPI Data Modes and Timing ..... 257
25.5. Frame Formats ..... 257
25.6. Data Transfer ..... 259
25.7. AVR USART MSPIM vs. AVR SPI ..... 260
25.8. Register Description ..... 261

26. Two-Wire Serial Interface (TWI) ..... 262
26.1. Features ..... 262
26.2. Two-Wire Serial Interface Bus Definition ..... 262
26.3. Data Transfer and Frame Format ..... 263
26.4. Multi-Master Bus Systems, Arbitration, and Synchronization ..... 266
26.5. Overview of the TWI Module ..... 268
26.6. Using the TWI ..... 270
26.7. Transmission Modes ..... 273
26.8. Multi-Master Systems and Arbitration ..... 291
26.9. Register Description ..... 292

27. Analog Comparator (AC) ..... 300
27.1. Overview ..... 300

27.2. Analog Comparator Multiplexed Input....300
27.3. Register Description....301

28. Analog-to-Digital Converter (ADC)....305
28.1. Features....305
28.2. Overview....305
28.3. Starting a Conversion....307
28.4. Prescaling and Conversion Timing....308
28.5. Changing Channel or Reference Selection....310
28.6. ADC Noise Canceler....312
28.7. ADC Conversion Result....315
28.8. Temperature Measurement....316
28.9. Register Description....316

29. debugWIRE On-chip Debug System....325
29.1. Features....325
29.2. Overview....325
29.3. Physical Interface....325
29.4. Software Breakpoints....326
29.5. Limitations of debugWIRE....326
29.6. Register Description....326

30. Boot Loader Support – Read-While-Write Self-programming (BTLDR)....328
30.1. Features....328
30.2. Overview....328
30.3. Application and Boot Loader Flash Sections....328
30.4. Read-While-Write and No Read-While-Write Flash Sections....329
30.5. Boot Loader Lock Bits....331
30.6. Entering the Boot Loader Program....332
30.7. Addressing the Flash During Self-Programming....333
30.8. Self-Programming the Flash....334
30.9. Register Description....342

31. Memory Programming (MEMPROG)....345
31.1. Program And Data Memory Lock Bits....345
31.2. Fuse Bits....346
31.3. Signature Bytes....348
31.4. Calibration Byte....349
31.5. Serial Number....349
31.6. Page Size....349
31.7. Parallel Programming Parameters, Pin Mapping, and Commands....349
31.8. Parallel Programming....351
31.9. Serial Downloading....359

32. Electrical Characteristics....364
32.1. Absolute Maximum Ratings....364
32.2. Common DC Characteristics....364
32.3. Speed Grades....367

32.4. Clock Characteristics....368
32.5. System and Reset Characteristics....369
32.6. SPI Timing Characteristics....370
32.7. Two-Wire Serial Interface Characteristics....371
32.8. ADC Characteristics....373
32.9. Parallel Programming Characteristics....374

33. Typical Characteristics ($T_A$ = -40°C to 85°C)....377
33.1. ATmega328 Typical Characteristics....377

34. Typical Characteristics ($T_A$ = -40°C to 105°C)....402
34.1. ATmega328P Typical Characteristics....402

35. Register Summary....427
35.1. Note....429

36. Instruction Set Summary....431

37. Packaging Information....436
37.1. 32-pin 32A....436
37.2. 32-pin 32M1-A....437
37.3. 28-pin 28M1....438
37.4. 28-pin 28P3....438

38. Errata....440
38.1. Errata ATmega328/P....440

39. Datasheet Revision History....441
39.1. Rev. A – 2/2018....441
39.2. Pre Microchip Revisions....441

The Microchip Web Site....442

Customer Change Notification Service....442

Customer Support....442

Microchip Devices Code Protection Feature....442

Legal Notice....443

Trademarks....443

Quality Management System Certified by DNV....444

Worldwide Sales and Service....445

# 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 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 ATmega328/P provides the following features: 32Kbytes of in-system programmable Flash with read-while-write capabilities, 1Kbytes EEPROM, 2Kbytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), three flexible timer/counters with Compare modes and PWM, 1 serial programmable USARTs , 1 byte-oriented 2-wire Serial Interface ($I^2C$), a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages), 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 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.

Microchip 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 Microchip'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 ATmega328/P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega328/P 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 | ATmega328/P |
|---|---|
| Pin Count | 28/32 |
| Flash (Bytes) | 32K |
| SRAM (Bytes) | 2K |
| EEPROM (Bytes) | 1K |
| General Purpose I/O Lines | 23 |
| SPI | 2 |
| TWI ($I^2C$) | 1 |
| USART | 1 |
| ADC | 10-bit 15 kSPS |
| ADC Channels | 8 |
| 8-bit Timer/Counters | 2 |
| 16-bit Timer/Counters | 1 |

# 3. Ordering Information

## 3.1 ATmega328

| Speed [MHz](3) | Power Supply [V] | Ordering Code(2) | Package(1) | Operational Range |
|---|---|---|---|---|
| 20 | 1.8 - 5.5 | ATmega328-AU<br>ATmega328-AUR(5)<br>ATmega328-MMH(4)<br>ATmega328-MMHR(4)(5)<br>ATmega328-MU<br>ATmega328-MUR(5)<br>ATmega328-PU | 32A<br>32A<br>28M1<br>28M1<br>32M1-A<br>32M1-A<br>28P3 | Industrial<br>(-40°C to 85°C) |

**Note:**

1. This device can also be supplied in wafer form. Please contact your local Microchip 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. Please refer to *Speed Grades* for Speed vs. $V_{CC}$
4. Tape & Reel.
5. NiPdAu Lead Finish.

| Package Type | |
|---|---|
| 28M1 | 28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF) |
| 28P3 | 28-lead, 0.300” Wide, Plastic Dual Inline Package (PDIP) |
| 32M1-A | 32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF) |
| 32A | 32-lead, Thin (1.0mm) Plastic Quad Flat Package (TQFP) |

## 3.2 ATmega328P

| Speed [MHz](3) | Power Supply [V] | Ordering Code(2) | Package(1) | Operational Range |
|---|---|---|---|---|
| 20 | 1.8 - 5.5 | ATmega328P-AU<br>ATmega328P-AUR(5)<br>ATmega328P-MMH(4)<br>ATmega328P-MMHR(4)(5)<br>ATmega328P-MU | 32A<br>32A<br>28M1<br>28M1<br>32M1-A | Industrial<br>(-40°C to 85°C) |

| Speed [MHz](3) | Power Supply [V] | Ordering Code(2) | Package(1) | Operational Range |
|---|---|---|---|---|
| | | ATmega328P-MUR(5)<br>ATmega328P-PU | 32M1-A<br>28P3 | |
| | | ATmega328P-AN<br>ATmega328P-ANR(5)<br>ATmega328P-MN<br>ATmega328P-MNR(5)<br>ATmega328P-PN | 32A<br>32A<br>32M1-A<br>32M1-A<br>28P3 | Industrial<br>(-40°C to 105°C) |

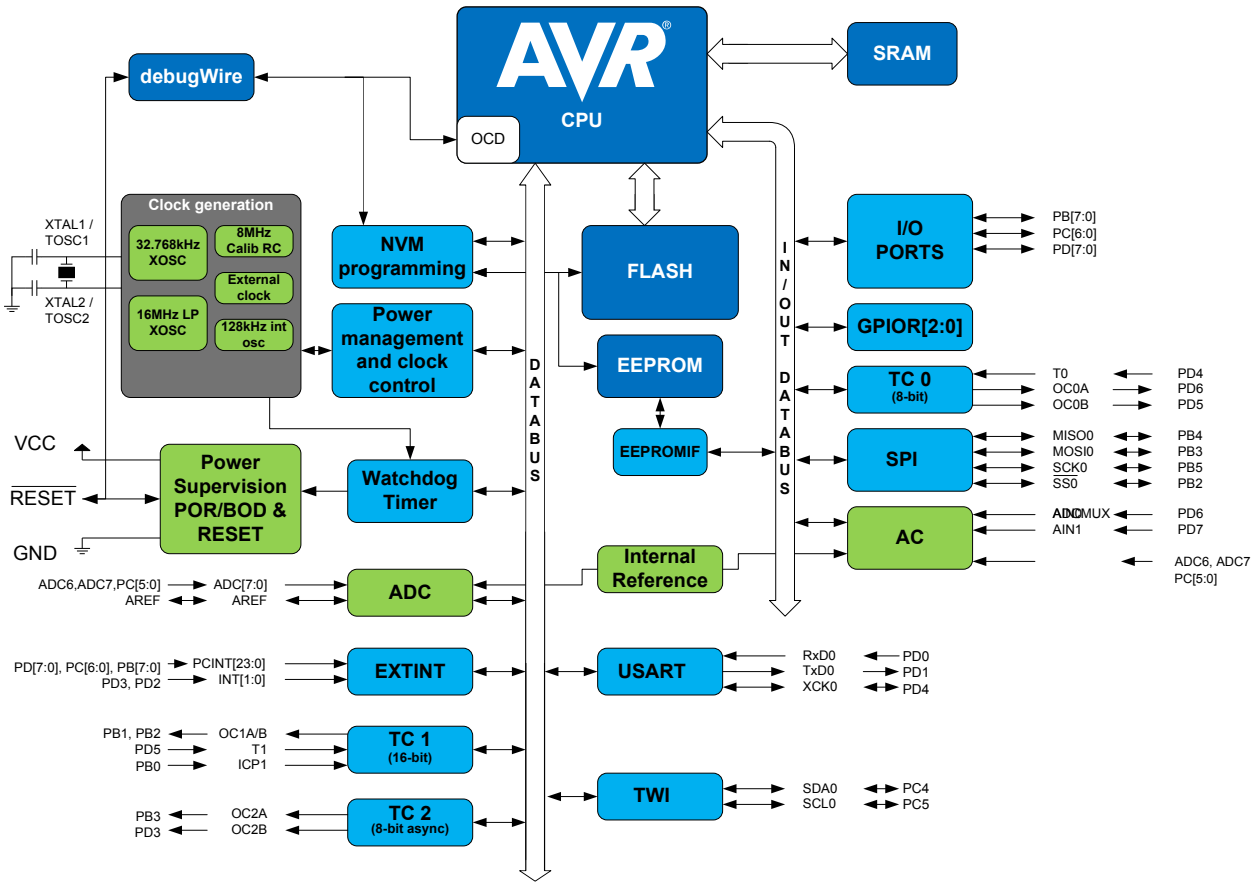
**Note:**

1. This device can also be supplied in wafer form. Please contact your local Microchip 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. Please refer to *Speed Grades* for Speed vs. $V_{CC}$
4. Tape & Reel.
5. NiPdAu Lead Finish.

| Package Type | |
|---|---|
| 28M1 | 28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF) |
| 28P3 | 28-lead, 0.300” Wide, Plastic Dual Inline Package (PDIP) |
| 32M1-A | 32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF) |
| 32A | 32-lead, Thin (1.0mm) Plastic Quad Flat Package (TQFP) |

# 4. Block Diagram

**Figure 4-1. Block Diagram**



# 5. Pin Configurations

## 5.1 Pinout

**Figure 5-1. 28-pin PDIP**

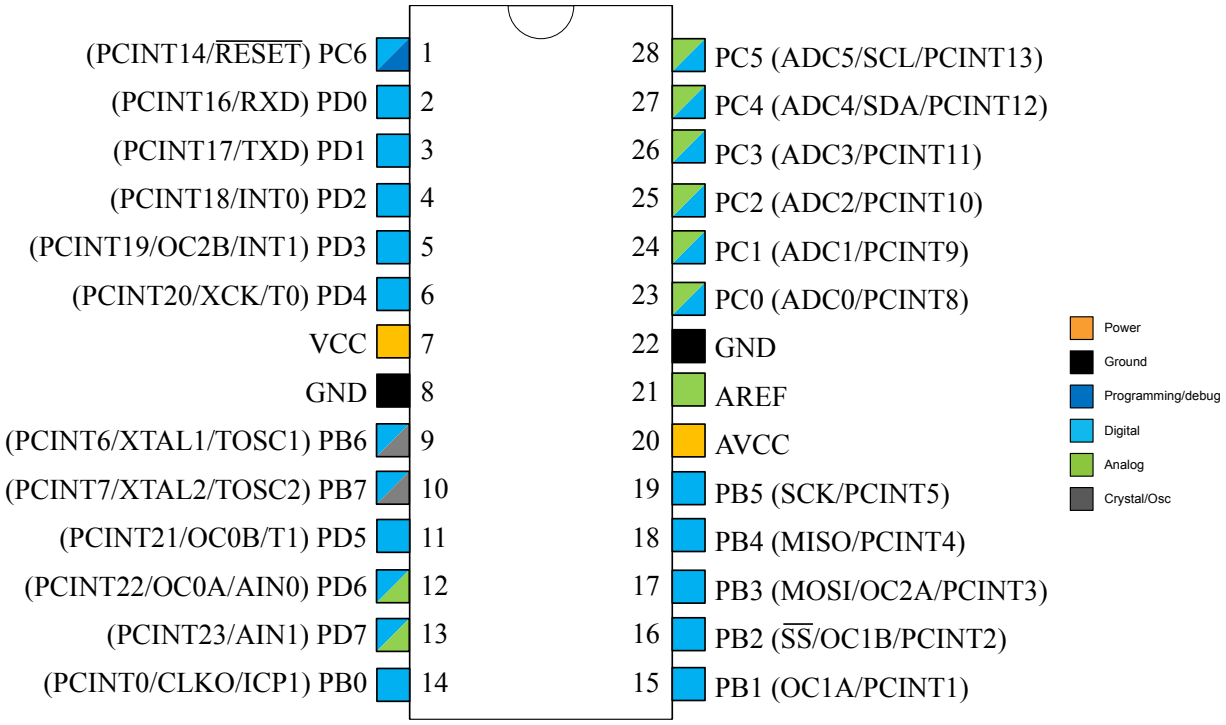


**Figure 5-2. 28-pin MLF Top View**

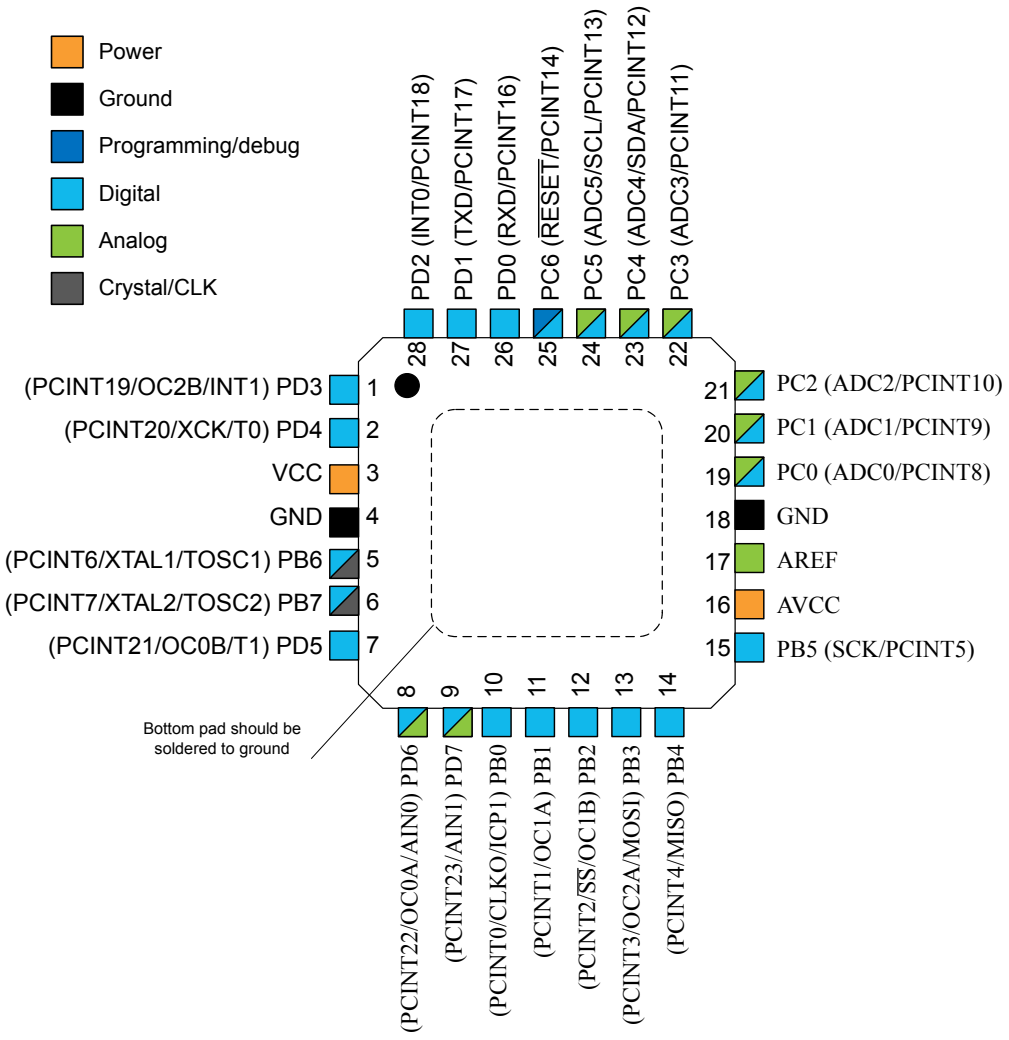


**Figure 5-3. 32-pin TQFP Top View**

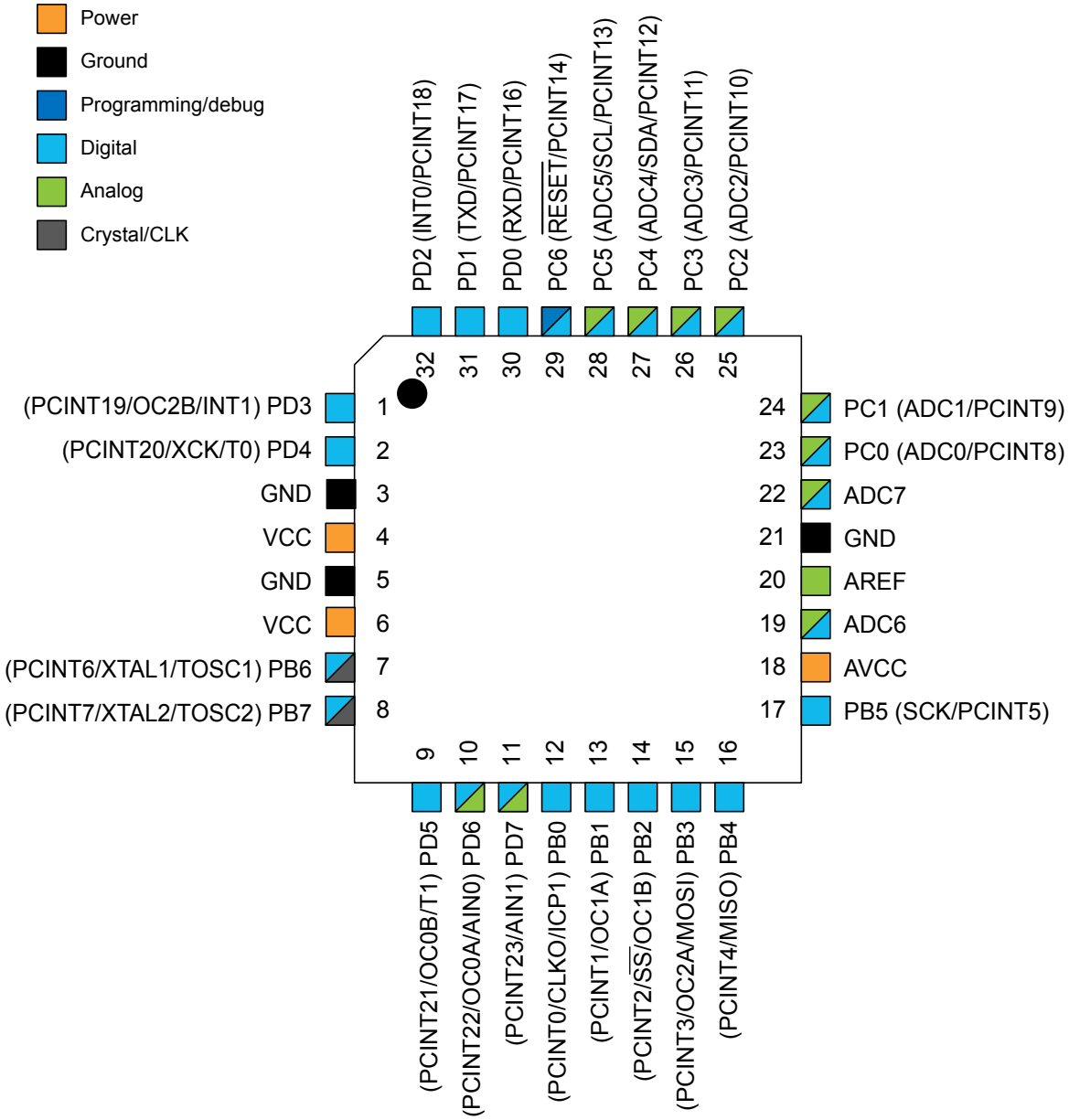
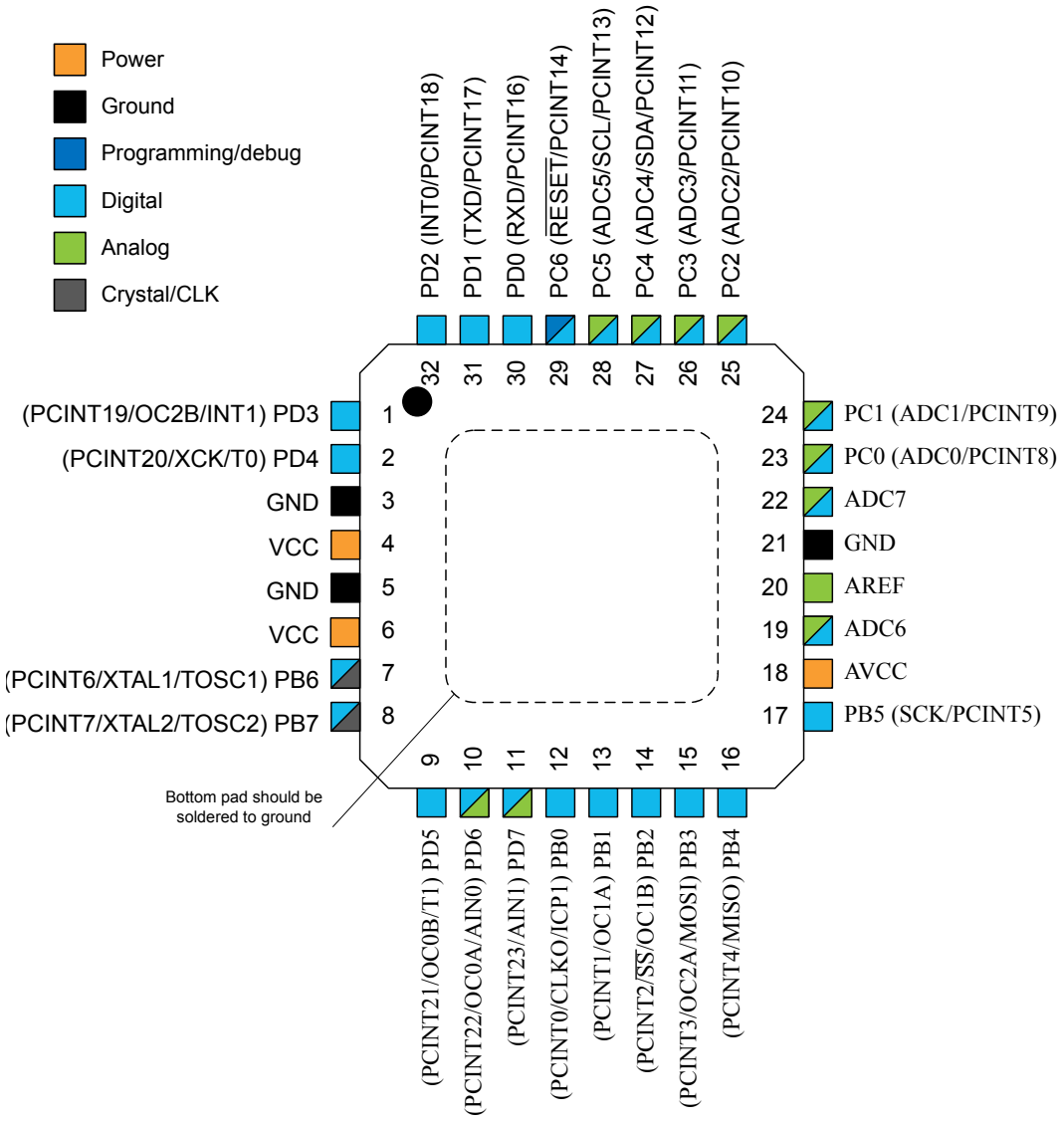


**Figure 5-4. 32-pin MLF Top View**




## 5.2 Pin Descriptions

### 5.2.1 VCC

Digital supply voltage pin.

### 5.2.2 GND

Ground.

### 5.2.3 Port B (PB[7:0]) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each pin). 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 during a Reset condition even if the clock is not running.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting oscillator amplifier and input to the internal clock operating circuit.

Depending on the clock selection fuse settings, PB7 can be used as output from the inverting oscillator amplifier.

If the internal calibrated RC oscillator is used as chip clock source, PB[7:6] is used as TOSC[2:1] input for the asynchronous timer/counter2 if the AS2 bit in ASSR is set.

### 5.2.4 Port C (PC[5:0])

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each pin). The PC[5:0] 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 during a Reset condition even if the clock is not running.

### 5.2.5 PC6/$\overline{\text{RESET}}$

If the RSTDISBL fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C.

If the RSTDISBL fuse is unprogrammed, PC6 is used as a 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.

The various special features of Port C are elaborated in the *Alternate Functions of Port C* section.

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

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each pin). 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 during a Reset condition even if the clock is not running.

### 5.2.7 $AV_{CC}$

$AV_{CC}$ is the supply voltage pin for the A/D Converter (ADC), PC[3:0], and PE[3:2]. 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. Note that PC[6:4] use digital supply voltage, $V_{CC}$.

### 5.2.8 AREF

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

### 5.2.9 ADC[7:6]

In the TQFP and VFQFN package, ADC[7:6] serve as analog inputs to the A/D converter. These pins are powered by the analog supply and serve as 10-bit ADC channels.

# 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 MLF/TQFP) Pin# | (28-pin MLF) Pin# | (28-pin PIPD) Pin# | PAD | EXTINT | PCINT | ADC/AC | OSC | T/C #0 | T/C #1 | USART 0 | $I^2C$ 0 | SPI 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 5 | PD3 | INT1 | PCINT19 | | | OC2B | | | | |
| 2 | 2 | 6 | PD4 | | PCINT20 | | | T0 | | XCK0 | | |
| 4 | 3 | 7 | VCC | | | | | | | | | |
| 3 | 4 | 8 | GND | | | | | | | | | |
| 6 | - | - | VCC | | | | | | | | | |
| 5 | - | - | GND | | | | | | | | | |
| 7 | 5 | 9 | PB6 | | PCINT6 | | XTAL1/ TOSC1 | | | | | |
| 8 | 6 | 10 | PB7 | | PCINT7 | | XTAL2/ TOSC2 | | | | | |
| 9 | 7 | 11 | PD5 | | PCINT21 | | | OC0B | T1 | | | |
| 10 | 8 | 12 | PD6 | | PCINT22 | AIN0 | | OC0A | | | | |
| 11 | 9 | 13 | PD7 | | PCINT23 | AIN1 | | | | | | |
| 12 | 10 | 14 | PB0 | | PCINT0 | | CLKO | ICP1 | | | | |
| 13 | 11 | 15 | PB1 | | PCINT1 | | | OC1A | | | | |
| 14 | 12 | 16 | PB2 | | PCINT2 | | | OC1B | | | | $\overline{SS0}$ |
| 15 | 13 | 17 | PB3 | | PCINT3 | | | OC2A | | | | MOSI0 |
| 16 | 14 | 18 | PB4 | | PCINT4 | | | | | | | MISO0 |
| 17 | 15 | 19 | PB5 | | PCINT5 | | | | | | | SCK0 |
| 18 | 16 | 20 | AVCC | | | | | | | | | |
| 19 | - | - | ADC6 | | | ADC6 | | | | | | |
| 20 | 17 | 21 | AREF | | | | | | | | | |
| 21 | 18 | 22 | GND | | | | | | | | | |
| 22 | - | - | ADC7 | | | ADC7 | | | | | | |
| 23 | 19 | 13 | PC0 | | PCINT8 | ADC0 | | | | | | |
| 24 | 20 | 24 | PC1 | | PCINT9 | ADC1 | | | | | | |
| 25 | 21 | 25 | PC2 | | PCINT10 | ADC2 | | | | | | |
| 26 | 22 | 26 | PC3 | | PCINT11 | ADC3 | | | | | | |
| 27 | 23 | 27 | PC4 | | PCINT12 | ADC4 | | | | | SDA0 | |
| 28 | 24 | 28 | PC5 | | PCINT13 | ADC5 | | | | | SCL0 | |

| (32-pin MLF/TQFP) Pin# | (28-pin MLF) Pin# | (28-pin PIPD) Pin# | PAD | EXTINT | PCINT | ADC/AC | OSC | T/C #0 | T/C #1 | USART 0 | $I^2C$ 0 | SPI 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 29 | 25 | 1 | PC6/$\overline{\text{RESET}}$ | | PCINT14 | | | | | | | |
| 30 | 26 | 2 | PD0 | | PCINT16 | | | | | RXD0 | | |
| 31 | 27 | 3 | PD1 | | PCINT17 | | | | | TXD0 | | |
| 32 | 28 | 4 | PD2 | INT0 | PCINT18 | | | | | | | |

# 7. Resources

A comprehensive set of development tools, application notes, and datasheets are available for download on http://www.microchip.com/design-centers/8-bit/microchip-avr-mcus.

# 8. 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.

# 9. 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`.

# 10. Capacitive Touch Sensing

## 10.1 QTouch Library

The QTouch® library provides a simple to use solution to realize touch sensitive interfaces on most AVR® microcontrollers. The QTouch library includes support for the QTouch and QMatrix™ acquisition methods.

Touch sensing can be added to any application by linking the appropriate 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.

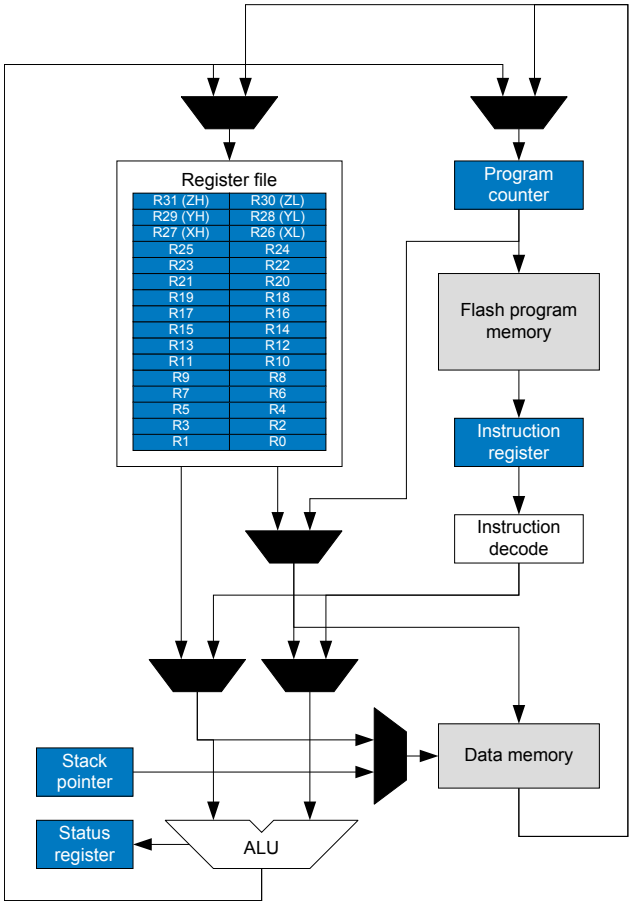
The QTouch library is FREE and downloadable from QTouch Library . For implementation details and other information, refer to the QTouch Library User Guide, also available for download from the Microchip website.

# 11. AVR CPU Core

## 11.1 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 11-1. 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 be used as an

address pointer for lookup 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 into 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 Stack Pointer (SP) in the Reset routine (before subroutines or interrupts are executed). The 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.

## 11.2 Arithmetic Logic Unit (ALU)

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 provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See *Instruction Set Summary* section for a detailed description.

**Related Links**

Instruction Set Summary

## 11.3 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.

# 24. Universal Synchronous Asynchronous Receiver Transceiver (USART)

## 24.1 Features

- Full Duplex Operation (Independent Serial Receive and Transmit Registers)
- Asynchronous or Synchronous Operation
- Master or Slave Clocked Synchronous Operation
- High-Resolution Baud Rate Generator
- Supports Serial Frames with 5, 6, 7, 8, or 9 data bits and 1 or 2 stop bits
- Odd or Even Parity Generation and Parity Check Supported by Hardware
- Data OverRun Detection
- Framing Error Detection
- Noise Filtering Includes False Start Bit Detection and Digital Low Pass Filter
- Three Separate Interrupts on TX Complete, TX Data Register Empty, and RX Complete
- Multi-processor Communication Mode
- Double Speed Asynchronous Communication Mode

## 24.2 Overview

The Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) is a highly flexible serial communication device.

The USART can also be used in Master SPI mode. The Power Reduction USART bit in the Power Reduction Register (PRR.PRUSARTn) must be written to '0' in order to enable USARTn.

**Related Links**

USART in SPI (USARTSPI) Mode

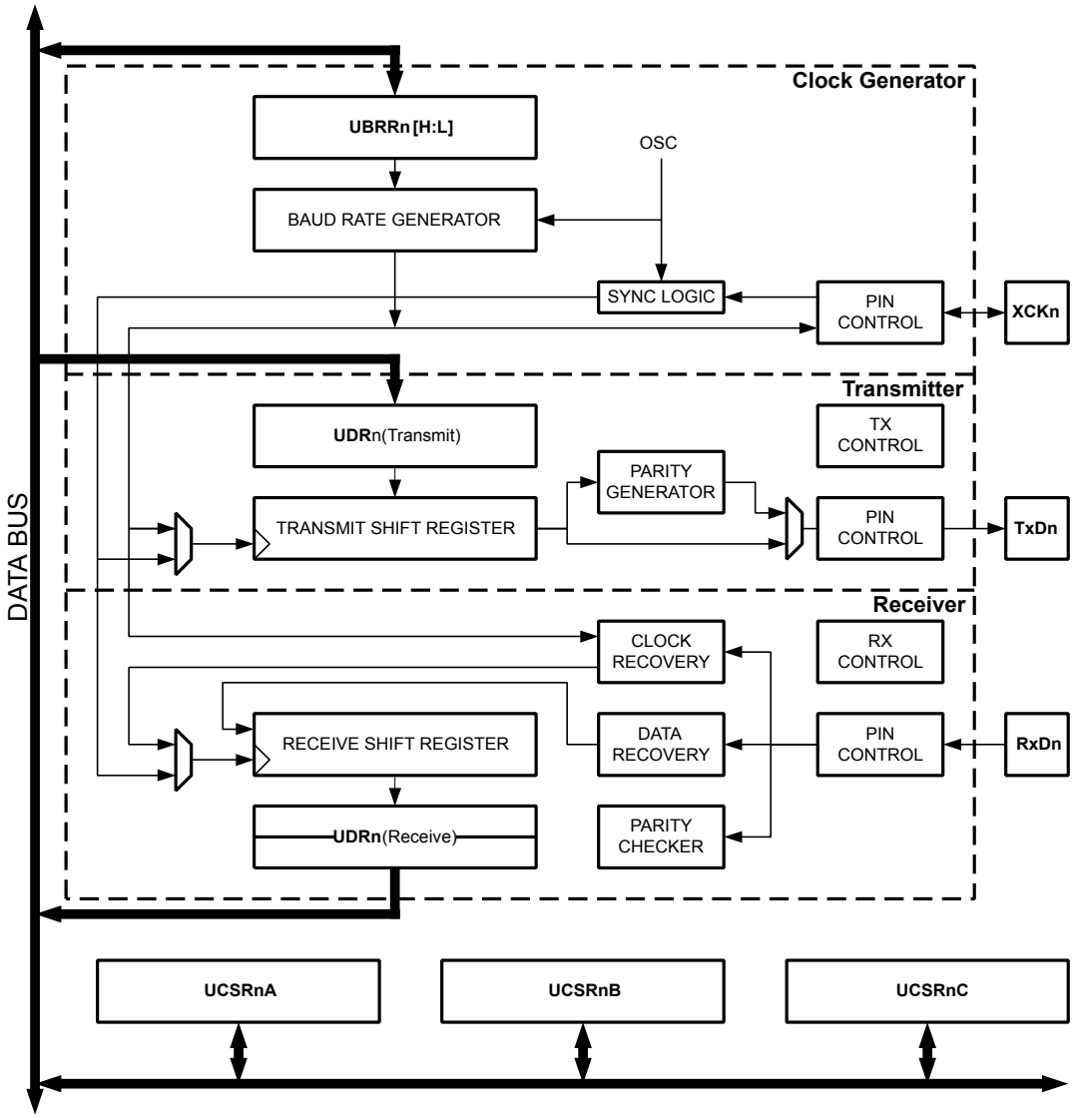
I/O-Ports

PRR

## 24.3 Block Diagram

In the USART block diagram, the CPU accessible I/O registers and I/O pins are shown in bold. The dashed boxes in the block diagram separate the three main parts of the USART (listed from the top): Clock generator, transmitter, and receiver. Control registers are shared by all units. The clock generation logic consists of synchronization logic for external clock input used by synchronous slave operation, and the baud rate generator. The XCKn (Transfer clock) pin is only used by synchronous transfer mode. The transmitter consists of a single write buffer, a serial Shift register, parity generator, and control logic for handling different serial frame formats. The write buffer allows a continuous transfer of data without any delay between frames. The receiver is the most complex part of the USART module due to its clock and data recovery units. The recovery units are used for asynchronous data reception. In addition to the recovery units, the receiver includes a parity checker, control logic, a Shift register, and a two-level receive buffer (UDRn). The receiver supports the same frame formats as the transmitter and can detect frame error, data overrun, and parity errors.

**Figure 24-1. USART Block Diagram**



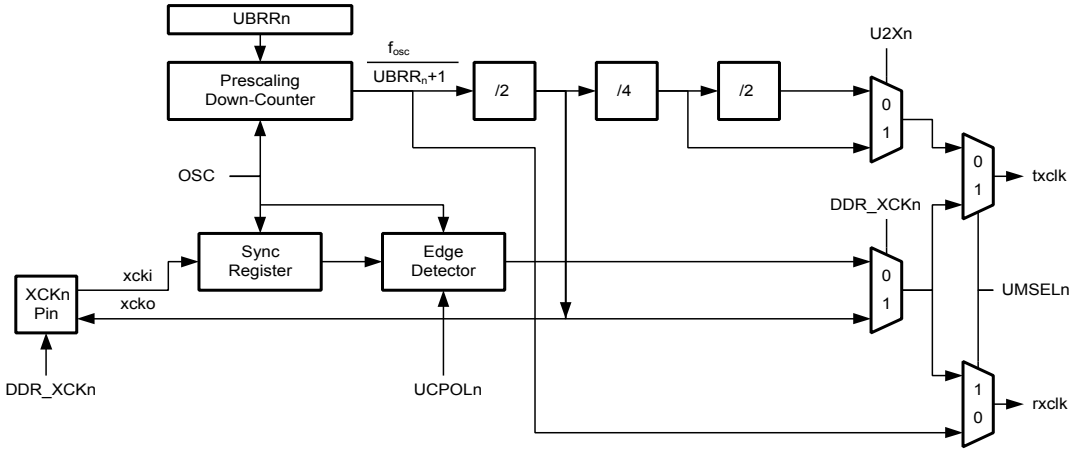

**Note:** Refer to the *Pin Configurations* and the *I/O-Ports* description for USART pin placement.

## 24.4 Clock Generation

The clock generation logic generates the base clock for the transmitter and receiver. The USART supports four modes of clock operation: Normal asynchronous, Double Speed asynchronous, Master synchronous, and Slave synchronous mode. The USART mode select bit 0 in the USART Control and Status Register n C (UCSRnC.UMSELn0) selects between asynchronous and synchronous operation. Double speed (asynchronous mode only) is controlled by the U2Xn found in the UCSRnA register. When using synchronous mode (UMSELn0=1), the data direction register for the XCKn pin (DDR_XCKn) controls whether the clock source is internal (Master mode) or external (Slave mode). The XCKn pin is only active when using Synchronous mode.

Below is a block diagram of the clock generation logic.

**Figure 24-2. Clock Generation Logic, Block Diagram**




Signal description:

- txclk: Transmitter clock (internal signal).
- rxclk: Receiver base clock (internal signal).
- xcki: Input from XCKn pin (internal signal). Used for synchronous slave operation.
- xcko: Clock output to XCKn pin (internal signal). Used for synchronous master operation.
- $f_{osc}$: System clock frequency.

### 24.4.1 Internal Clock Generation – The Baud Rate Generator

Internal clock generation is used for the Asynchronous and the Synchronous Master modes of operation. The description in this section refers to the clock generation logic block diagram in the previous section.

The USART Baud Rate Register (UBRRn) and the down-counter connected to it function as a programmable prescaler or baud rate generator. The down-counter, running at system clock ($f_{osc}$), is loaded with the UBRRn value each time the counter has counted down to zero or when the UBRRnL register is written. A clock is generated each time the counter reaches zero. This clock is the baud rate generator clock output (= $f_{osc}$/(UBRRn+1)). The transmitter divides the baud rate generator clock output by 2, 8, or 16 depending on the mode. The baud rate generator output is used directly by the receiver's clock and data recovery units. However, the recovery units use a state machine that uses 2, 8, or 16 states depending on the mode set by the state of the UMSEL, U2Xn and DDR_XCK bits.

The table below contains equations for calculating the baud rate (in bits per second) and for calculating the UBRRn value for each mode of operation using an internally generated clock source.

**Table 24-1. Equations for Calculating Baud Rate Register Setting**

| Operating Mode | Equation for Calculating Baud Rate(1) | Equation for Calculating UBRRn Value |
|---|---|---|
| Asynchronous Normal mode (U2Xn = 0) | $\mathrm{BAUD} = \frac{f_{\mathrm{OSC}}}{16(\mathbf{UBRR}n + 1)}$ | $\mathbf{UBRR}n = \frac{f_{\mathrm{OSC}}}{16\mathrm{BAUD}} - 1$ |
| Asynchronous Double Speed mode (U2Xn = 1) | $\mathrm{BAUD} = \frac{f_{\mathrm{OSC}}}{8(\mathbf{UBRR}n + 1)}$ | $\mathbf{UBRR}n = \frac{f_{\mathrm{OSC}}}{8\mathrm{BAUD}} - 1$ |
| Synchronous Master mode | $\mathrm{BAUD} = \frac{f_{\mathrm{OSC}}}{2(\mathbf{UBRR}n + 1)}$ | $\mathbf{UBRR}n = \frac{f_{\mathrm{OSC}}}{2\mathrm{BAUD}} - 1$ |

**Note:** 1. The baud rate is defined to be the transfer rate in bits per second (bps)

| | |
|---|---|
| **BAUD** | Baud rate (in bits per second, bps) |
| **$f_{osc}$** | System oscillator clock frequency |
| **UBRRn** | Contents of the UBRRnH and UBRRnL registers, (0-4095). Some examples of UBRRn values for some system clock frequencies are found in Examples of Baud Rate Settings. |

### 24.4.2 Double Speed Operation (U2Xn)

The transfer rate can be doubled by setting the U2Xn bit in UCSRnA. Setting this bit only has effect on the asynchronous operation. Set this bit to zero when using synchronous operation.

Setting this bit will reduce the divisor of the baud rate divider from 16 to 8, effectively doubling the transfer rate for asynchronous communication. However, in this case, the Receiver will only use half the number of samples (reduced from 16 to 8) for data sampling and clock recovery, and therefore a more accurate baud rate setting and system clock are required when this mode is used.

For the transmitter, there are no downsides.

### 24.4.3 External Clock

External clocking is used by the synchronous slave modes of operation. The description in this section refers to the clock generation logic block diagram in the previous section.

External clock input from the XCKn pin is sampled by a synchronization register to minimize the chance of meta-stability. The output from the synchronization register must then pass through an edge detector before it can be used by the transmitter and receiver. This process introduces a two CPU clock period delay and therefore the maximum external XCKn clock frequency is limited by the following equation:

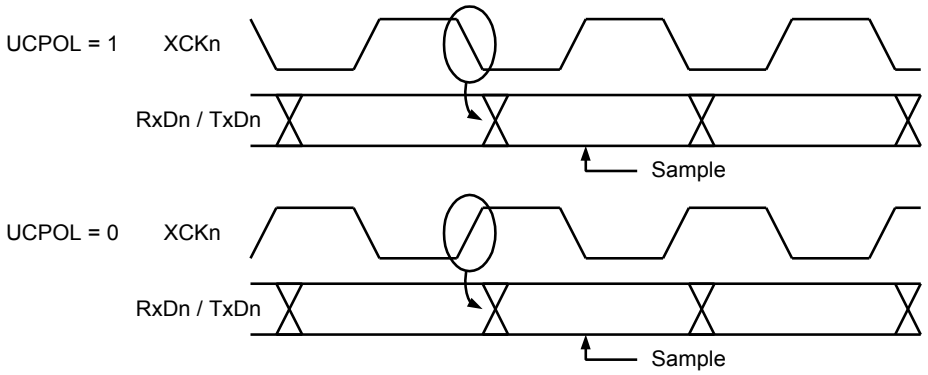
$$f_{XCKn} < \frac{f_{OSC}}{4}$$

The value of $f_{osc}$ depends on the stability of the system clock source. It is therefore recommended to add some margin to avoid possible loss of data due to frequency variations.

### 24.4.4 Synchronous Clock Operation

When synchronous mode is used (UMSEL = 1), the XCKn pin will be used as either clock input (slave) or clock output (master). The dependency between the clock edges and data sampling or data change is the same. The basic principle is that data input (on RxDn) is sampled at the opposite XCKn clock edge of the edge the data output (TxDn) is changed.

**Figure 24-3. Synchronous Mode XCKn Timing**



The UCPOL bit UCRSC selects which XCKn clock edge is used for data sampling and which is used for data change. As the above timing diagram shows, when UCPOL is zero, the data will be changed at rising XCKn edge and sampled at falling XCKn edge. If UCPOL is set, the data will be changed at falling XCKn edge and sampled at rising XCKn edge.

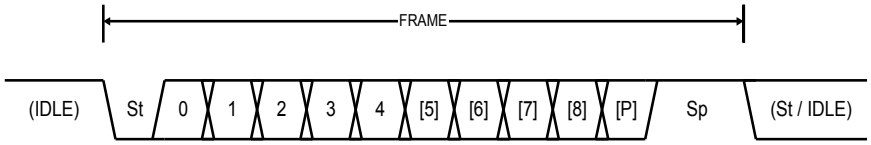
## 24.5 Frame Formats

A serial frame is defined to be one character of data bits with synchronization bits (start and stop bits), and optionally a parity bit for error checking. The USART accepts all 30 combinations of the following as valid frame formats:

- 1 start bit
- 5, 6, 7, 8, or 9 data bits
- no, even or odd parity bit
- 1 or 2 stop bits

A frame starts with the start bit, followed by the data bits (from five up to nine data bits in total): first the least significant data bit, then the next data bits ending with the most significant bit. If enabled, the parity bit is inserted after the data bits, before the one or two stop bits. When a complete frame is transmitted, it can be directly followed by a new frame, or the communication line can be set to an idle (high) state. the figure below illustrates the possible combinations of the frame formats. Bits inside brackets are optional.

**Figure 24-4. Frame Formats**




| | |
|---|---|
| St | Start bit, always low. |
| (n) | Data bits (0 to 8). |
| P | Parity bit. Can be odd or even. |
| Sp | Stop bit, always high. |
| IDLE | No transfers on the communication line (RxDn or TxDn). An IDLE line must be high. |

The frame format used by the USART is set by:

- Character Size bits (UCSRnC.UCSZn[2:0]) select the number of data bits in the frame.
- Parity Mode bits (UCSRnC.UPMn[1:0]) enable and set the type of parity bit.
- Stop Bit Select bit (UCSRnC.USBSn) select the number of stop bits. The Receiver ignores the second stop bit.

The receiver and transmitter use the same setting. Note that changing the setting of any of these bits will corrupt all ongoing communication for both the receiver and transmitter. An FE (Frame Error) will only be detected in cases where the first stop bit is zero.

### 24.5.1 Parity Bit Calculation

The parity bit is calculated by doing an exclusive-or of all the data bits. If odd parity is used, the result of the exclusive or is inverted. The relation between the parity bit and data bits is as follows:

$$P_{\text{even}} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 0$$