AVR Family of Micro-Controllers

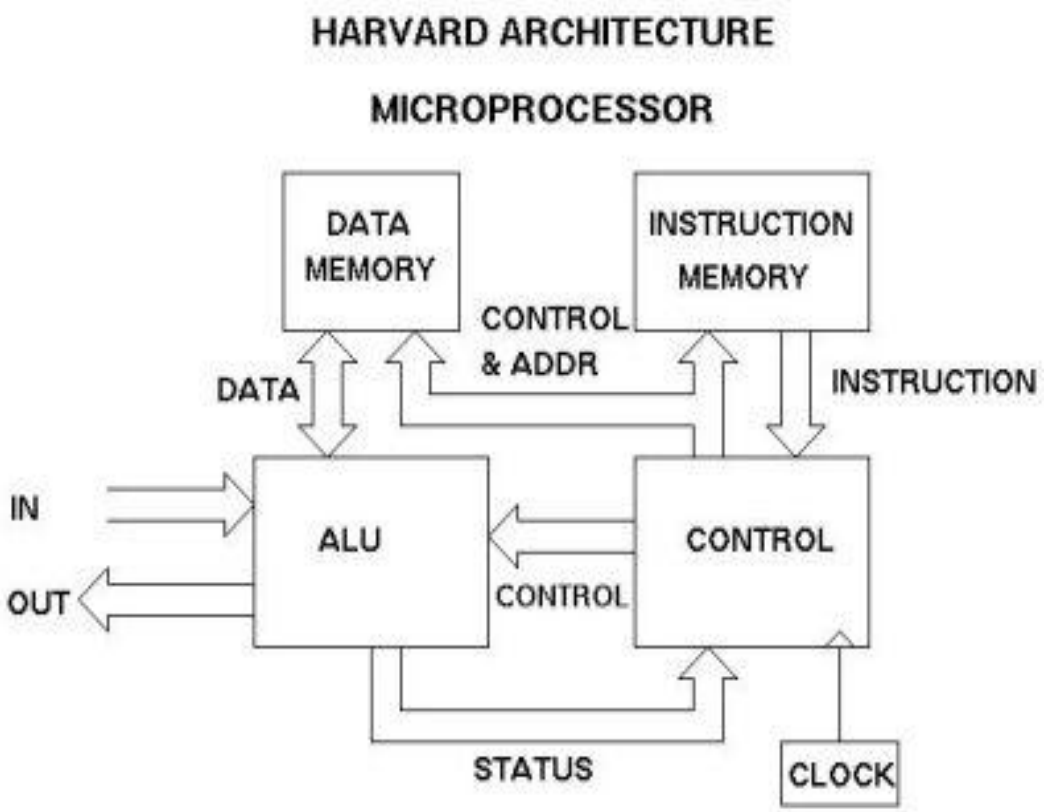
What is AVR ?

- Modified Harvard architecture 8-bit RISC single chip microcontroller
- Complete System-on-a-chip
  - On Board Memory (FLASH, SRAM & EEPROM)
  - On Board Peripherals
- Advanced (for 8 bit processors) technology
- Developed by Atmel in 1996
- First In-house CPU design by Atmel

AVR Family

- 8 Bit tinyAVR
  - Small package – as small as 6 pins
- 8 Bit megaAVR
  - Wide variety of configurations and packages
- 8 / 16 Bit AVR XMEGA
  - Second Generation Technology
- 32 Bit AVR UC3
  - Higher computational throughput

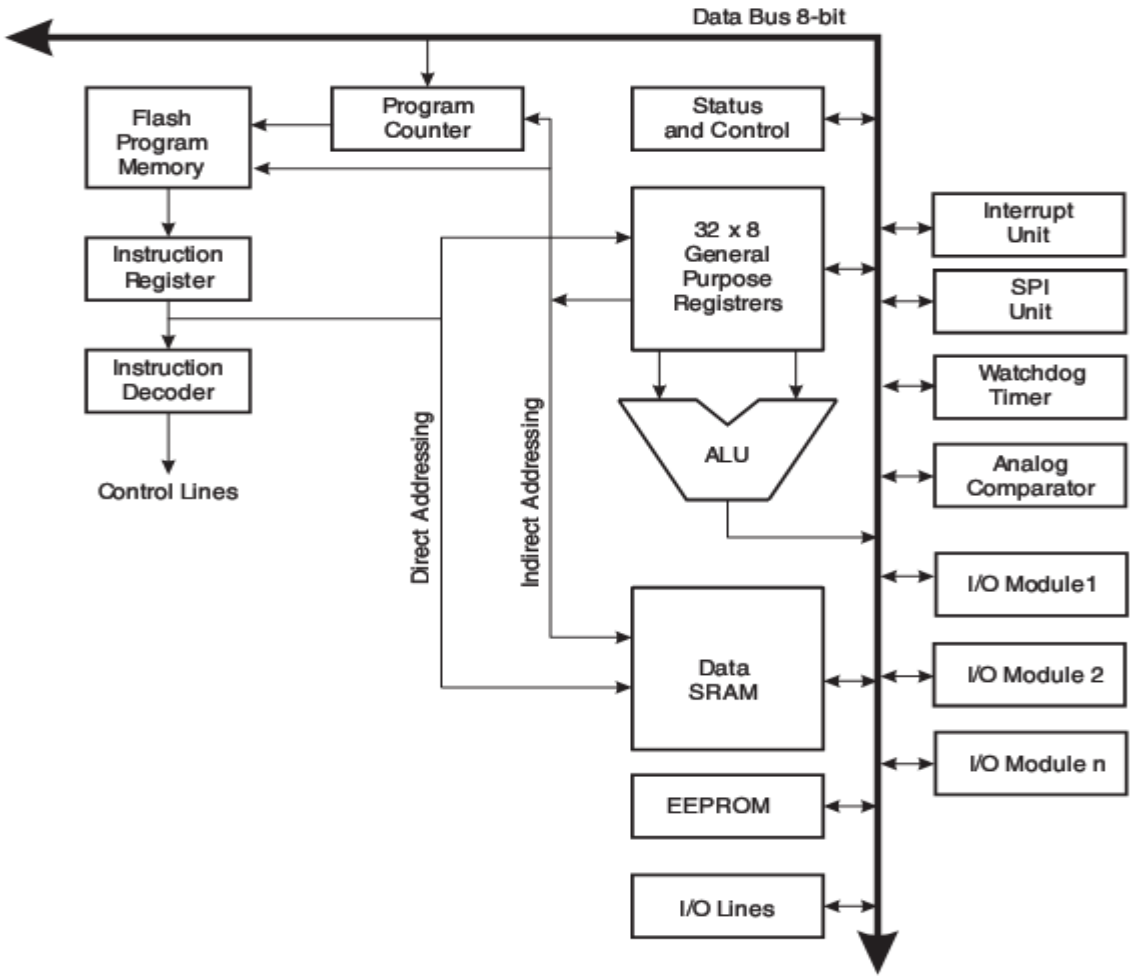
Harvard Architecture



Harvard Architecture Advantages

- Separate instruction and data paths
- Simultaneous accesses to instructions & data
- Hardware can be optimized for access type and bus width.

AVR Architecture



Modified Harvard Architecture

- Special instructions can access data from program space.
- Data memory is more expensive than program memory
- Don’t waste data memory for non-volatile data

What is RISC?

- Reduced Instruction Set Computer
- As compared to Complex Instruction Set Computers, i.e. x86
- Assumption: Simpler instructions execute faster
- Optimized most used instructions
- Other RISC machines: ARM, PowerPC, SPARC
- Became popular in mid 1990s

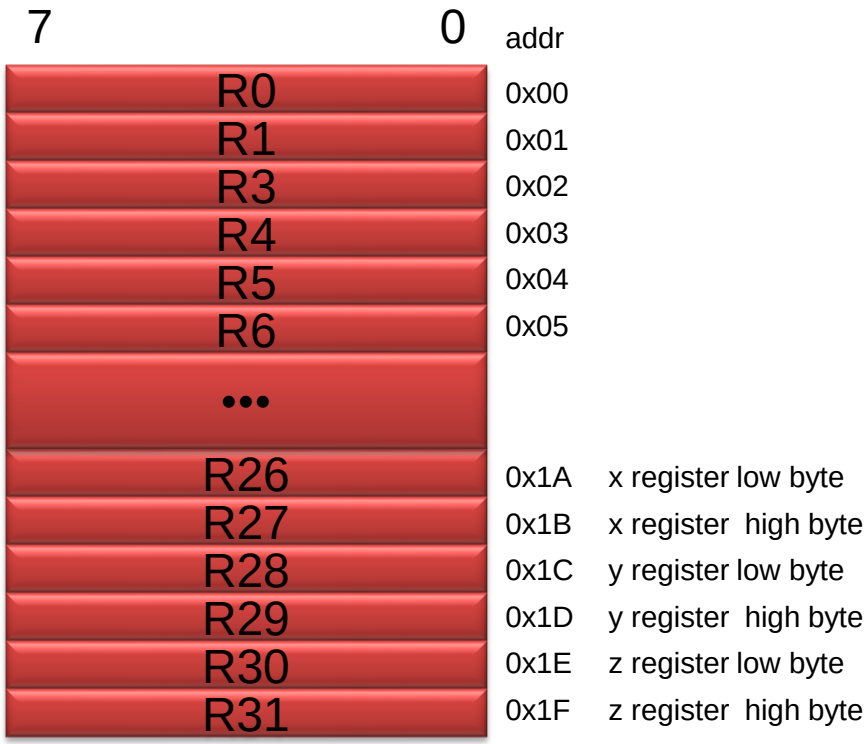
Characteristics of RISC Processors

- Faster clock rates
- Single cycle instructions (20 MIPS @ 20 MHz)
- Better compiler optimization

AVR Register File

- 32 8 Bit registers
- Mapped to address 0-31 in data space
- Most instructions can access any register and complete in one cycle
- Last 3 register pairs can be used as 3 16 bit index registers
- 32 bit stack pointer

Register File



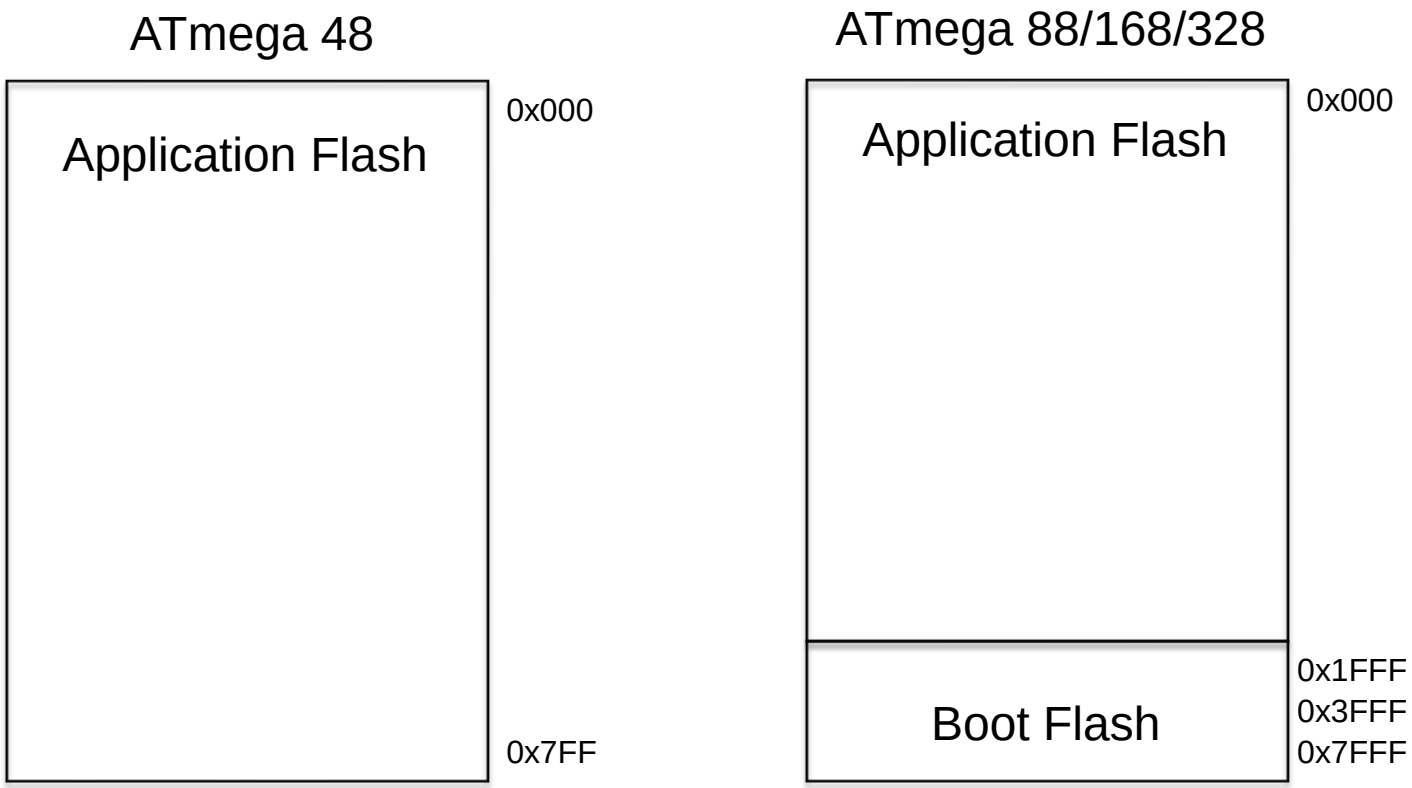
AVR Memory

FLASH

- Non-volatile program space storage
- 16 Bit width
- Some devices have separate lockable boot section
- At least 10,000 write/erase cycles

AVR Memories

FLASH – Memory Map

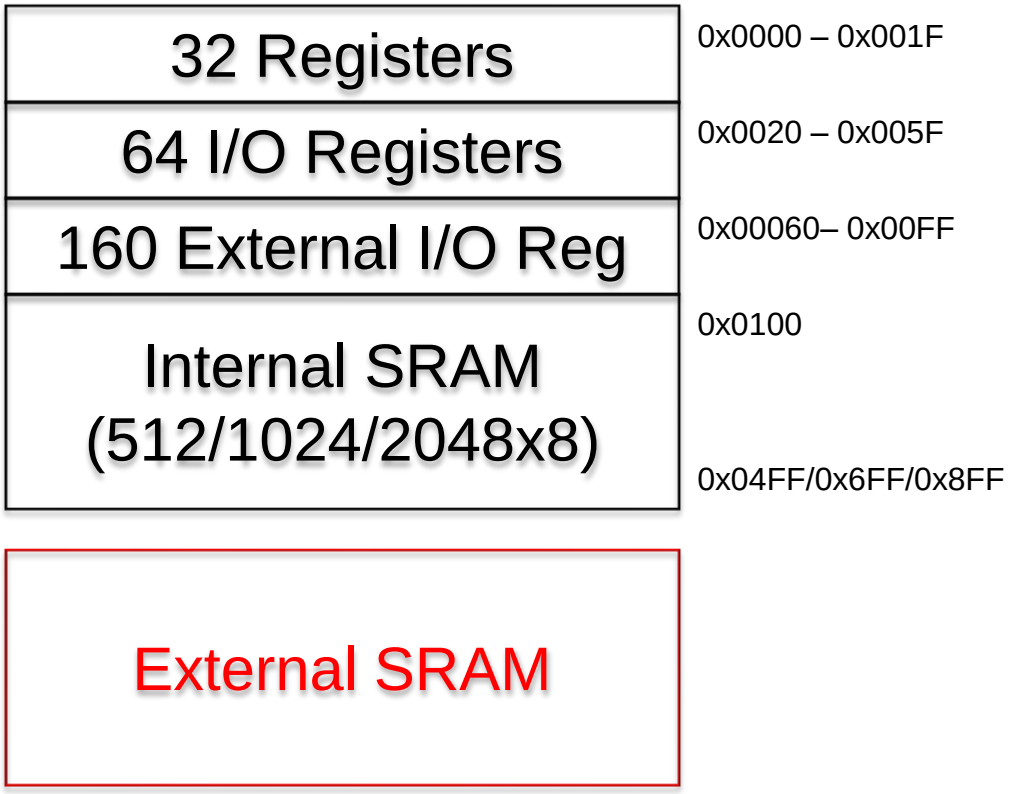


AVR Memories
SRAM

- Data space storage
- 8 Bit width

AVR Memories

SRAM - Memory Map



AVR Memories

EEPROM

- Electrically Erasable Programmable Read Only Memory
- 8 bit width
- Requires special write sequence
- Non-volatile storage for program specific data, constants, etc.
- At least 100,000 write/erase cycles

AVR Memories

DEVICE	FLASH	EEPROM	SRAM
ATmega48A	4K Bytes	256 Bytes	512 Bytes
ATmega48PA	4K Bytes	256 Bytes	512 Bytes
ATmega88A	8K Bytes	512 Bytes	1K Bytes
ATmega88PA	8K Bytes	512 Bytes	1K Bytes
ATmega168A	16K Bytes	512 Bytes	1K Bytes
ATmega168PA	16K Bytes	512 Bytes	1K Bytes
ATmega328	32K Bytes	1K Bytes	2K Bytes
ATmega328P	32K Bytes	1K Bytes	2K Bytes

Memory Mapped I/O Space

- I/O registers visible in data space
  - I/O can be accessed using same instructions as data
  - Compilers can treat I/O space as data access
- Bit manipulation instructions
  - Set/Clear single I/O bits
  - Only work on lower memory addresses

ALU – Arithmetic Logic Unit

- Directly connected to all 32 general purpose registers
- Operations between registers executed within a single clock cycle
- Supports arithmetic, logic and bit functions
- On-chip 2-cycle Multiplier

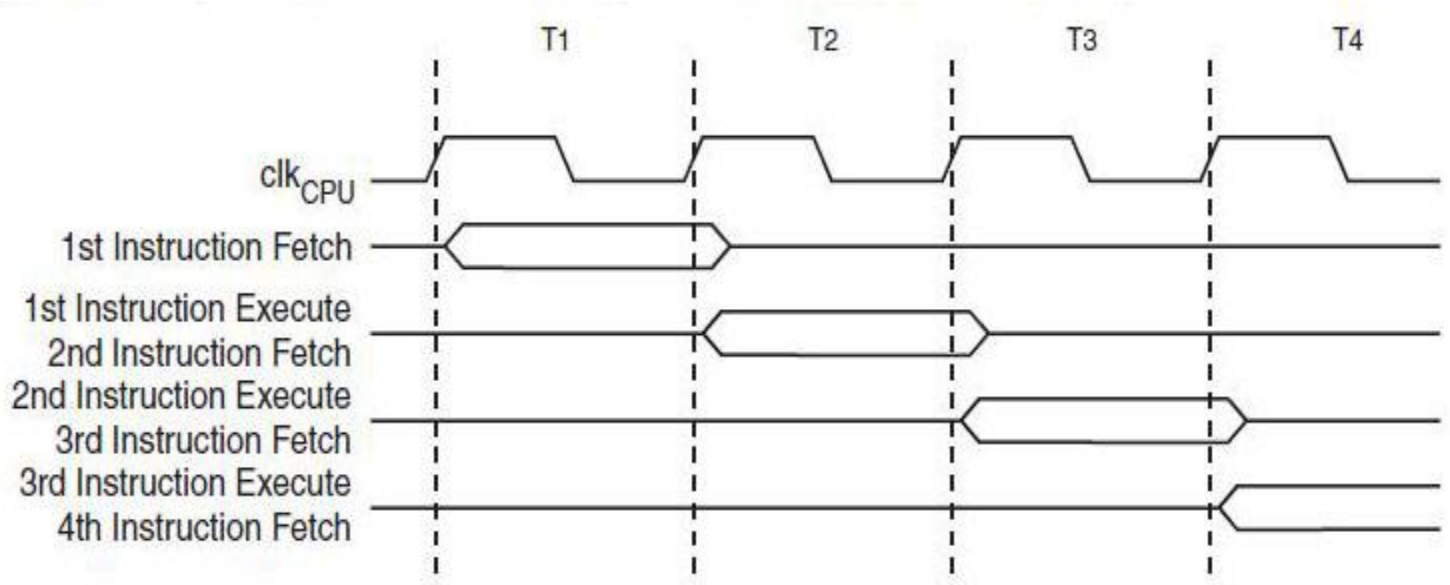
Instruction Set

- 131 instructions
  - Arithmetic & Logic
  - Branch
  - Bit set/clear/test
  - Data transfer
  - MCU control

Instruction Timing

- Register ↔ register in 1 cycle
- Register ↔ memory in 2 cycles
- Branch instruction 1-2 cycles
- Subroutine call & return 3-5 cycles
- Some operations may take longer for external memory

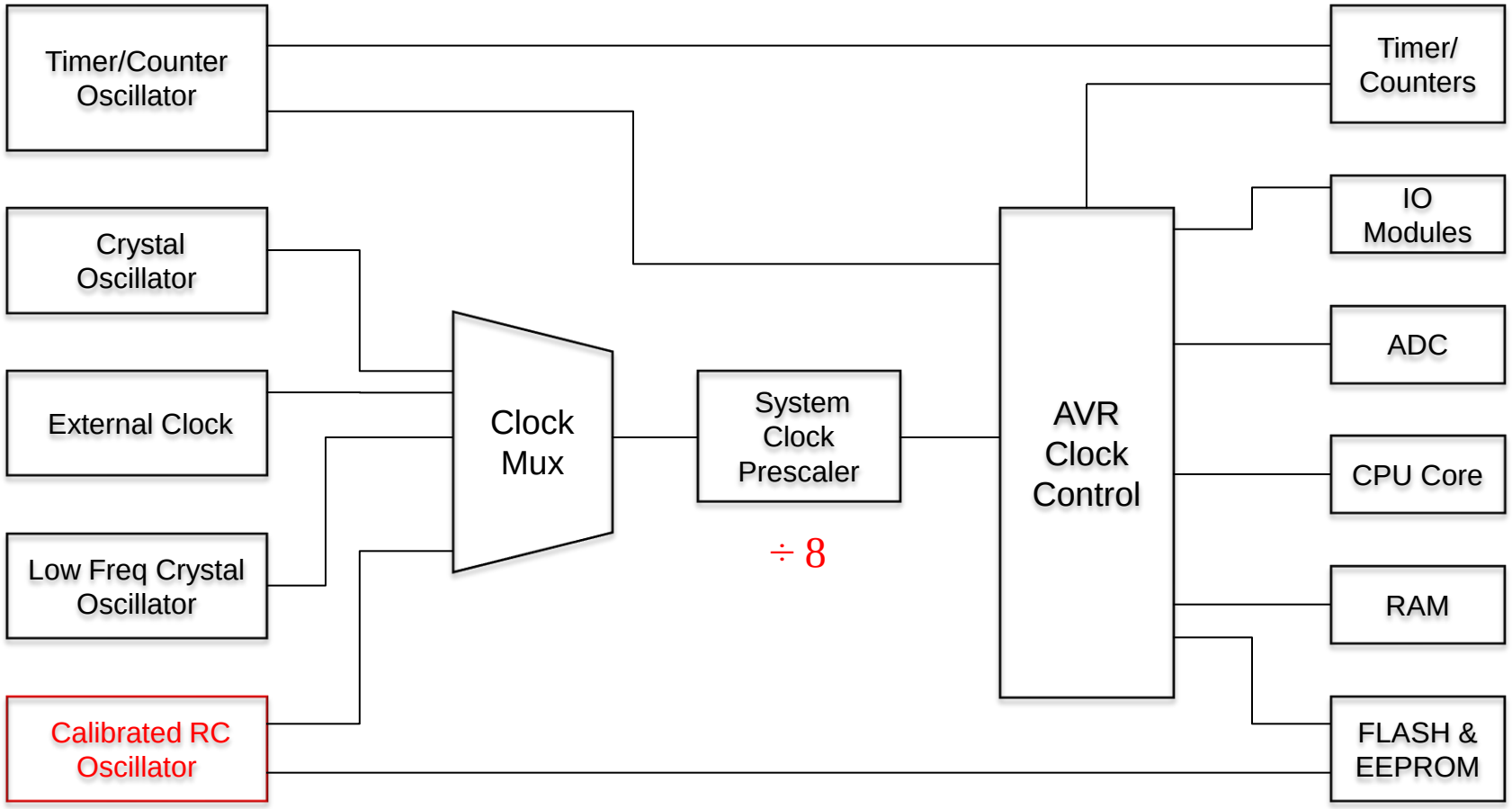
Pipelined Execution



AVR Clock System

- Clock control module generates clocks for memory and IO devices
- Multiple internal clock sources
- Provisions for external crystal clock source (max 20 MHz)
- Default is internal RC 8 MHz oscillator with ÷ 8 prescale yielding 1 MHz CPU clock

Clock Sources



Power Management

- Multiple power down modes
  - Power down mode
    - Wake on external reset or watchdog reset
  - Power save mode
    - Wake on timer events
  - Several standby modes
- Unused modules can be shut down

Reset Sources

- Power on reset
- External reset
- Watchdog system reset
- Brown out detect (BOD) reset

ATmega Peripherals

- 23 General Purpose IO Bits
- Two 8 bit & one 16 bit timer/counters
- Real time counter with separate oscillator
- 6 PWM Channels
- 6 or 8 ADC channels (depends on package)
- Serial USART
- SPI & I^2C Serial Interfaces
- Analog comparator
- Programmable watchdog timer

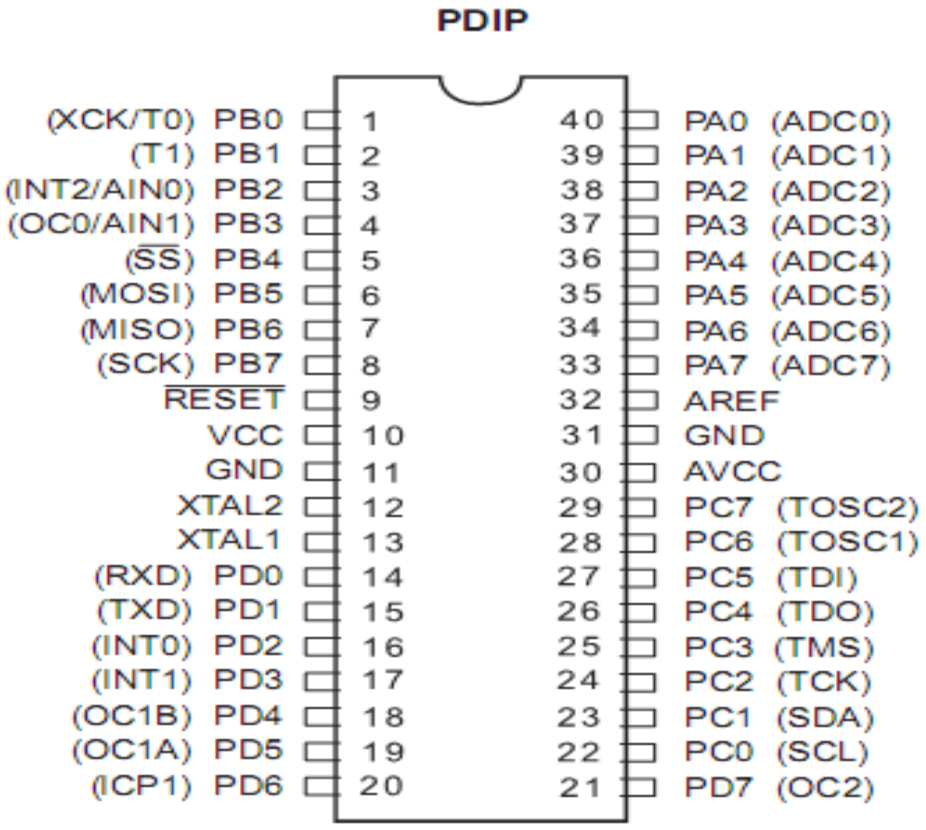
General Purpose IO Ports

- Three 8 Bit IO Ports
  - Port B, Port C & Port D
  - Pins identified as PBx, PCx or PDx (x=0..7)
- Each pin can be configured as:
  - Input with internal pull-up
  - Input with no pull-up
  - Output low
  - Output high

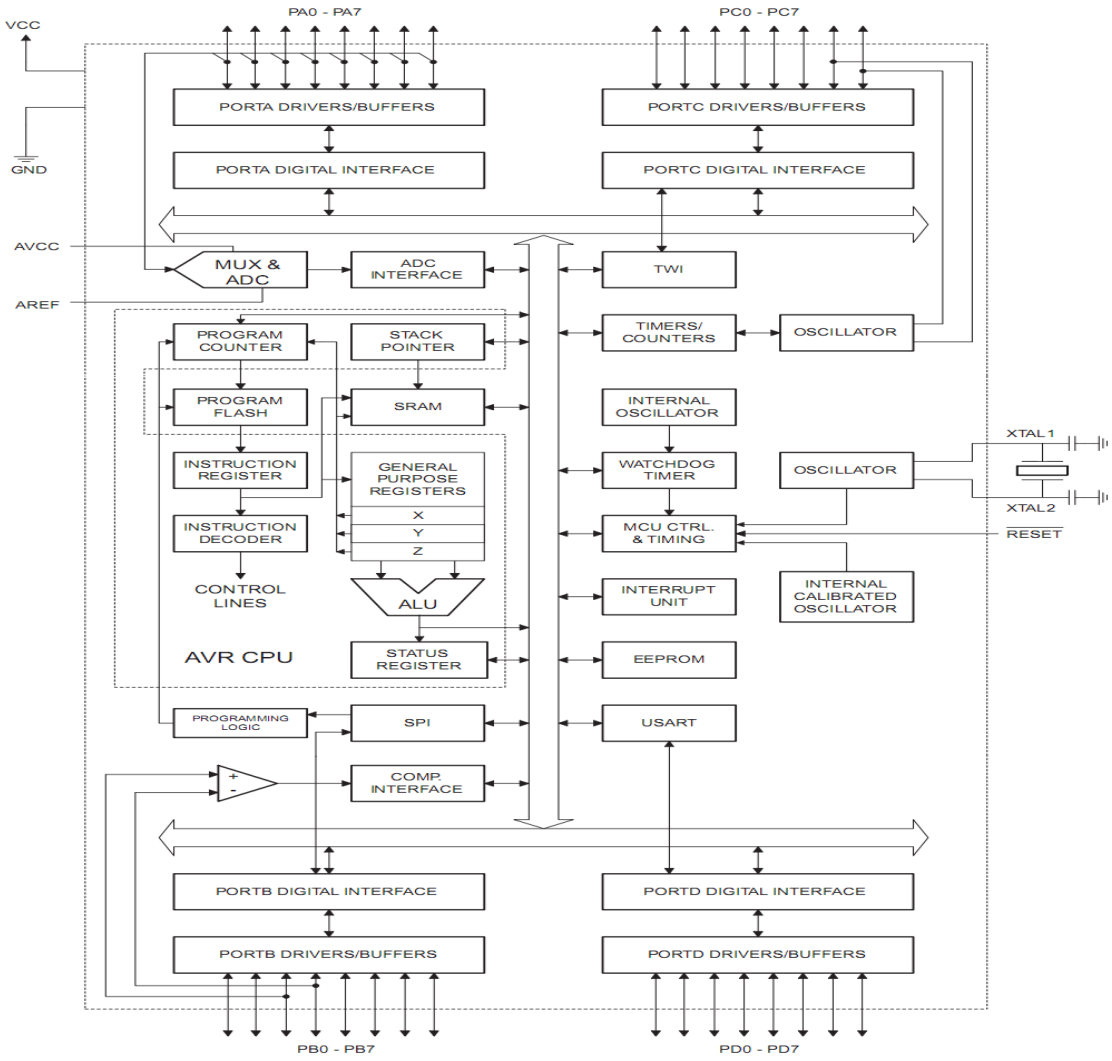
Alternate Port Functions

- Most port pins have alternate functions
- Internal peripherals use the alternate functions
- Each port pin can be assigned only one function at a time

Alternate Pins for PDIP Package



Atmega 16 Block Diagram



Instruction Set

Mnemonics	Operands	Description	Operation	Flags	#Clocks
ARITHMETIC AND LOGIC INSTRUCTIONS					
ADD	Rd, Rr	Add two Registers	Rd ← Rd + Rr	Z,C,N,V,H	1
ADC	Rd, Rr	Add with Carry two Registers	Rd ← Rd + Rr + C	Z,C,N,V,H	1
ADIW	Rdl,K	Add Immediate to Word	Rdh:Rdl ← Rdh:Rdl + K	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract two Registers	Rd ← Rd - Rr	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Constant from Register	Rd ← Rd - K	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract with Carry two Registers	Rd ← Rd - Rr - C	Z,C,N,V,H	1
SBCI	Rd, K	Subtract with Carry Constant from Reg.	Rd ← Rd - K - C	Z,C,N,V,H	1
SBIW	Rdl,K	Subtract Immediate from Word	Rdh:Rdl ← Rdh:Rdl - K	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND Registers	Rd ← Rd • Rr	Z,N,V	1
ANDI	Rd, K	Logical AND Register and Constant	Rd ← Rd • K	Z,N,V	1
OR	Rd, Rr	Logical OR Registers	Rd ← Rd v Rr	Z,N,V	1
ORI	Rd, K	Logical OR Register and Constant	Rd ← Rd v K	Z,N,V	1
EOR	Rd, Rr	Exclusive OR Registers	Rd ← Rd ⊕ Rr	Z,N,V	1
COM	Rd	One's Complement	Rd ← $FF – Rd	Z,C,N,V	1
NEG	Rd	Two's Complement	Rd ← $00 – Rd	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	Rd ← Rd v K	Z,N,V	1
CBR	Rd,K	Clear Bit(s) in Register	Rd ← Rd • ($FF - K)	Z,N,V	1
INC	Rd	Increment	Rd ← Rd + 1	Z,N,V	1
DEC	Rd	Decrement	Rd ← Rd – 1	Z,N,V	1
TST	Rd	Test for Zero or Minus	Rd ← Rd • Rd	Z,N,V	1
CLR	Rd	Clear Register	Rd ← Rd ⊕ Rd	Z,N,V	1
SER	Rd	Set Register	Rd ← $FF	None	1
MUL	Rd, Rr	Multiply Unsigned	R1:R0 ← Rd x Rr	Z,C	2
MULS	Rd, Rr	Multiply Signed	R1:R0 ← Rd x Rr	Z,C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	R1:R0 ← Rd x Rr	Z,C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULS	Rd, Rr	Fractional Multiply Signed	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2