CP316
Introduction to AVR Architecture

Terry Sturtevant

Wilfrid Laurier University

June 19, 2019

What this course is and is not

What this course is and is not

This is *not* an exhaustive course about the AVR microcontroller.

What this course is and is not

This is *not* an exhaustive course about the AVR microcontroller.

This is *not* an exhaustive course about the AVR assembly language.

What this course is and is not

This is *not* an exhaustive course about the AVR microcontroller.

This is *not* an exhaustive course about the AVR assembly language.

This is *not* an exhaustive course about the Arduino UNO.

What this course is and is not

This is *not* an exhaustive course about the AVR microcontroller.

This is *not* an exhaustive course about the AVR assembly language.

This is *not* an exhaustive course about the Arduino UNO.

This is *not* an exhaustive course about real-time programming.

What this course is and is not

This is *not* an exhaustive course about the AVR microcontroller.

This is *not* an exhaustive course about the AVR assembly language.

This is *not* an exhaustive course about the Arduino UNO.

This is *not* an exhaustive course about real-time programming.

This *is* an introduction to techniques for real-time hardware interfacing.

Microprocessors, digital signal processors and microcontrollers

Microprocessors, digital signal processors and microcontrollers

- μPU vs. DSP vs. μCU

Microprocessors, digital signal processors and microcontrollers

- μPU vs. DSP vs. μCU

  μPU is for *general purpose* processing

Microprocessors, digital signal processors and microcontrollers

- μPU vs. DSP vs. μCU

  μPU is for *general purpose* processing

  DSP is for *math intensive* processing

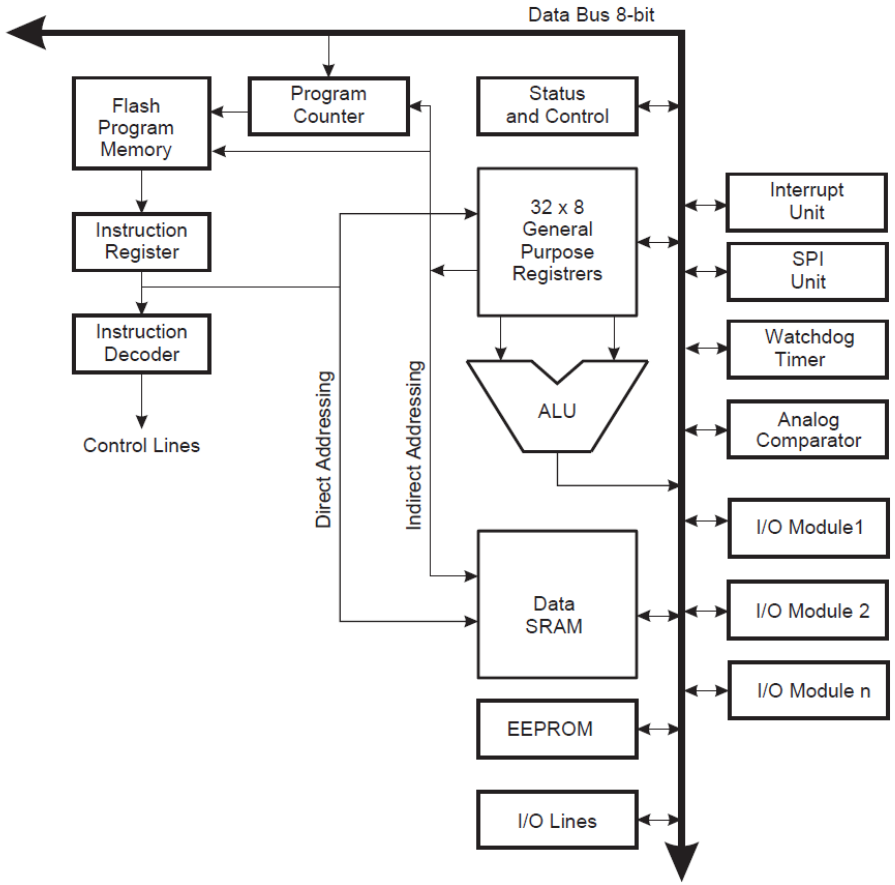
Microprocessors, digital signal processors and microcontrollers

- μPU vs. DSP vs. μCU

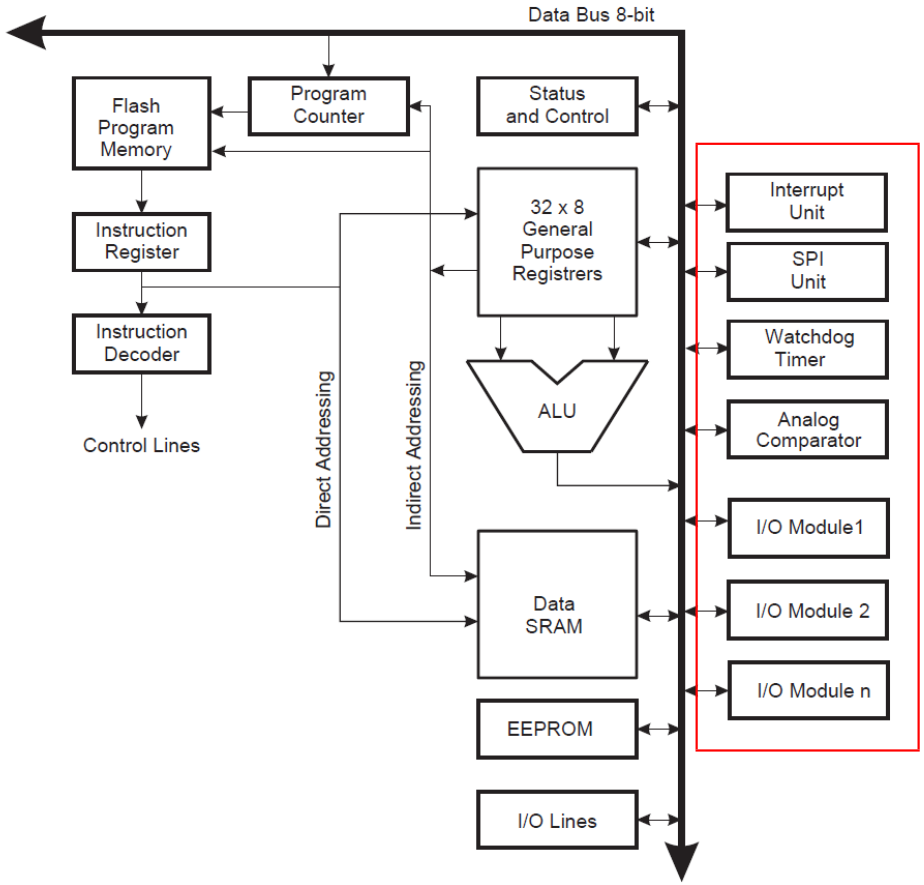
  μPU is for *general purpose* processing

  DSP is for *math intensive* processing

  μCU is for *I/O intensive* processing


Data Bus 8-bit
Flash Program Memory
Program Counter
Status and Control
Instruction Register
Instruction Decoder
Control Lines
32 x 8 General Purpose Registrers
ALU
Direct Addressing
Indirect Addressing
Data SRAM
EEPROM
I/O Lines
Interrupt Unit
SPI Unit
Watchdog Timer
Analog Comparator
I/O Module1
I/O Module 2
I/O Module n

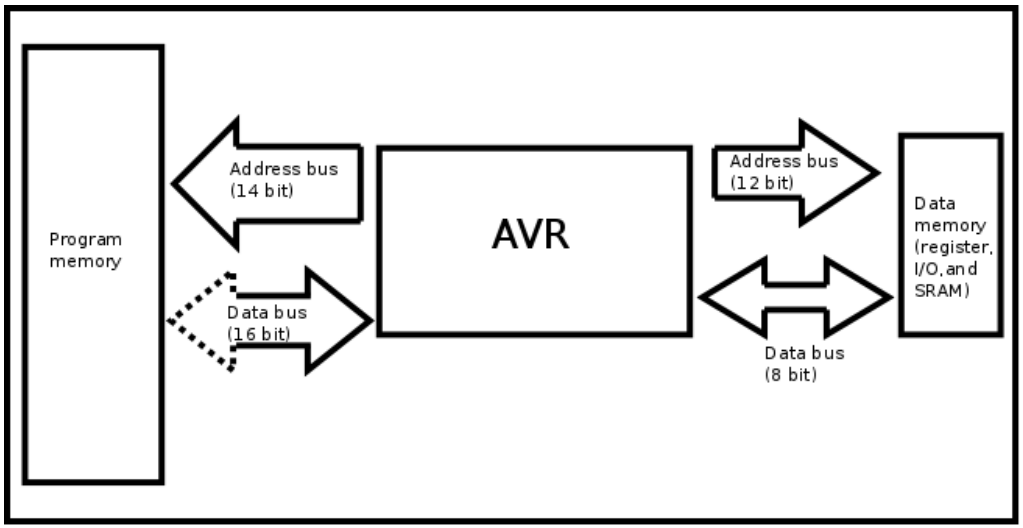
AVR core



I/O modules identify a micro*controller*

Section 8

Section 8



Memory organization - **Harvard** architecture

Harvard Architecture

Harvard Architecture

separate memory space for data and code

Harvard Architecture

separate memory space for data and code

address and data buses can be different widths

Harvard Architecture

separate memory space for data and code

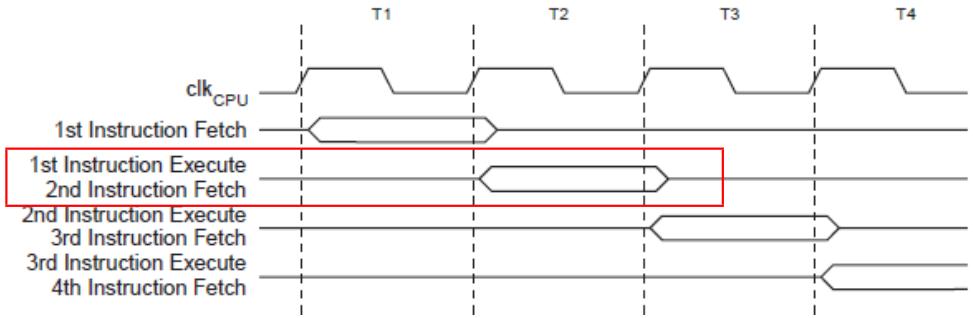
address and data buses can be different widths

program execution is more efficient since *execution* of current instruction and *fetch* of next instruction can happen simultaneously

Figure 7-4

Figure 7-4 shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast-access Register File concept. This is the basic pipelining concept to obtain up to 1 MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks, and functions per power-unit.

Figure 7-4. The Parallel Instruction Fetches and Instruction Executions

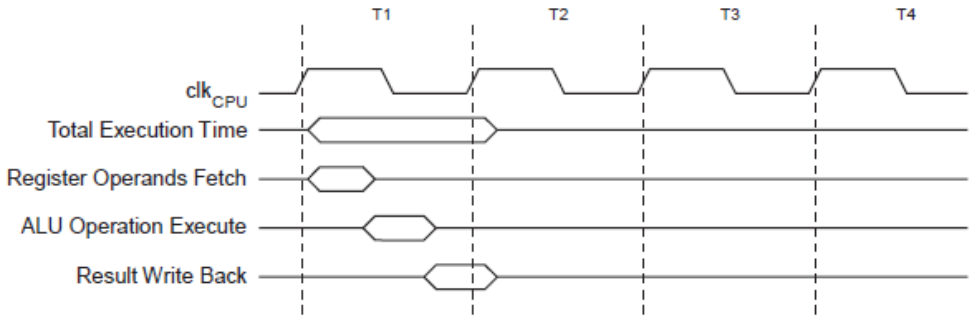


Instruction timing

Figure 7-5

Figure 7-5 shows the internal timing concept for the Register File. In a single clock cycle an ALU operation using two register operands is executed, and the result is stored back to the destination register.

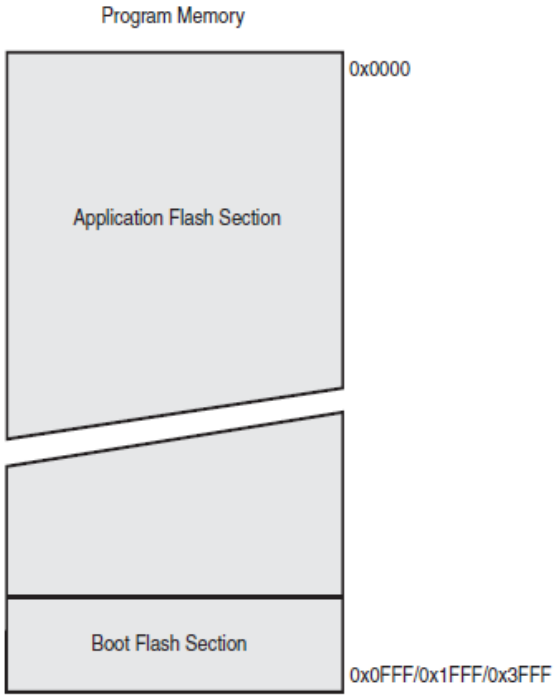
Figure 7-5. Single Cycle ALU Operation



ALU register timing

Figure 8-2

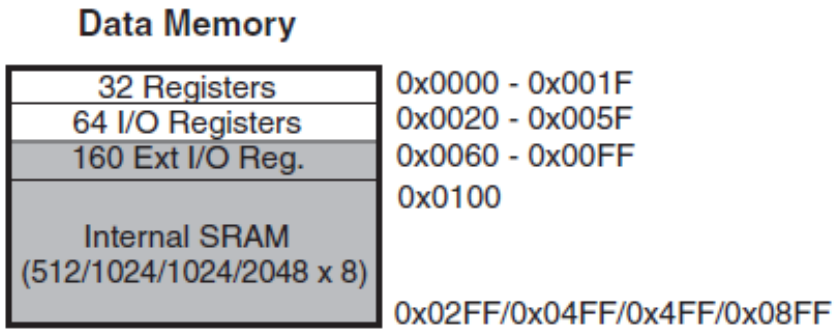
Figure 8-2. Program Memory Map ATmega88A, ATmega88PA, ATmega168A, ATmega168PA, ATmega328 and ATmega328P



Program memory

Figure 8-3

Figure 8-3. Data Memory Map



Data and I/O memory

Registers accessible as data or I/O

Registers accessible as data or I/O

LD/LDS/LDD and ST/STS/STD for *all* I/O registers

Registers accessible as data or I/O

LD/LDS/LDD and ST/STS/STD for *all* I/O registers

I/O registers 0x00-0x1F

SBI and CBI instructions as well

Registers accessible as data or I/O

LD/LDS/LDD and ST/STS/STD for *all* I/O registers

I/O registers 0x00-0x1F

SBI and CBI instructions as well

LD and ST instructions require 0x20 added to address

Section 14

14.4.2 PORTB – The Port B Data Register

Bit	7	6	5	4	3	2	1	0	
0x05 (0x25)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

14.4.3 DDRB – The Port B Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x04 (0x24)	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

14.4.4 PINB – The Port B Input Pins Address(1)

Bit	7	6	5	4	3	2	1	0	
0x03 (0x23)	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Note: 1. Writing to the pin register provides toggle functionality for IO (see "Toggling the Pin" on page 76)

Access as I/O *or* data (addresses +0x20)

Section 14

14.4.2 PORTB – The Port B Data Register

Bit	7	6	5	4	3	2	1	0	
0x05 (0x25)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

14.4.3 DDRB – The Port B Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x04 (0x24)	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

14.4.4 PINB – The Port B Input Pins Address(1)

Bit	7	6	5	4	3	2	1	0	
0x03 (0x23)	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Note: 1. Writing to the pin register provides toggle functionality for IO (see "Toggling the Pin" on page 76)

Access as I/O *or* data (addresses +0x20)

Section 14

14.4.2 PORTB – The Port B Data Register

Bit	7	6	5	4	3	2	1	0	
0x05 (0x25)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

14.4.3 DDRB – The Port B Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x04 (0x24)	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

14.4.4 PINB – The Port B Input Pins Address(1)

Bit	7	6	5	4	3	2	1	0	
0x03 (0x23)	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Note: 1. Writing to the pin register provides toggle functionality for IO (see "Toggling the Pin" on page 76)

Access as I/O *or* data (addresses +0x20)