CSSE2010/CSSE7201
Lecture 11

AVR Introduction

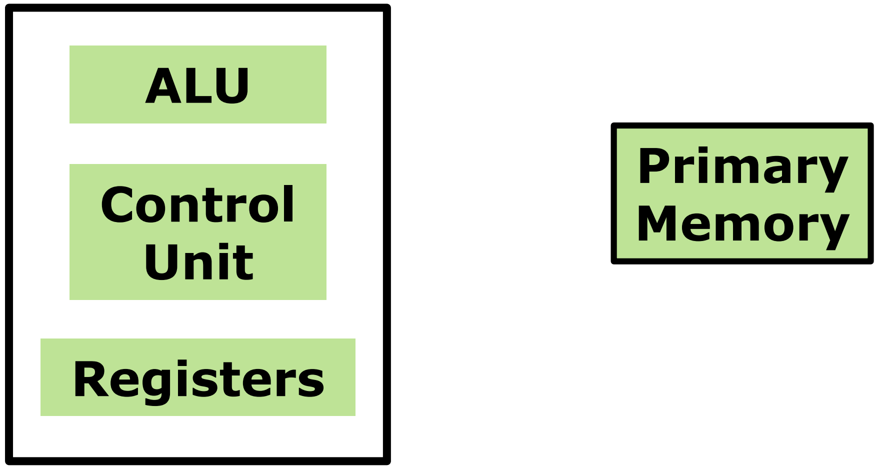
School of Information Technology and Electrical Engineering
The University of Queensland

Today

- Machine Code
- Introduction to AVR
  - Hardware
  - Assembly Language

Last Time

- Basic computer organisation
  - ALU
  - Control Unit
  - Registers
  - Primary memory



Machine Instructions (Machine Code)

- Just 1’s and 0’s
  - The 1’s and 0’s specify the instruction to be executed
- PowerPC Example:

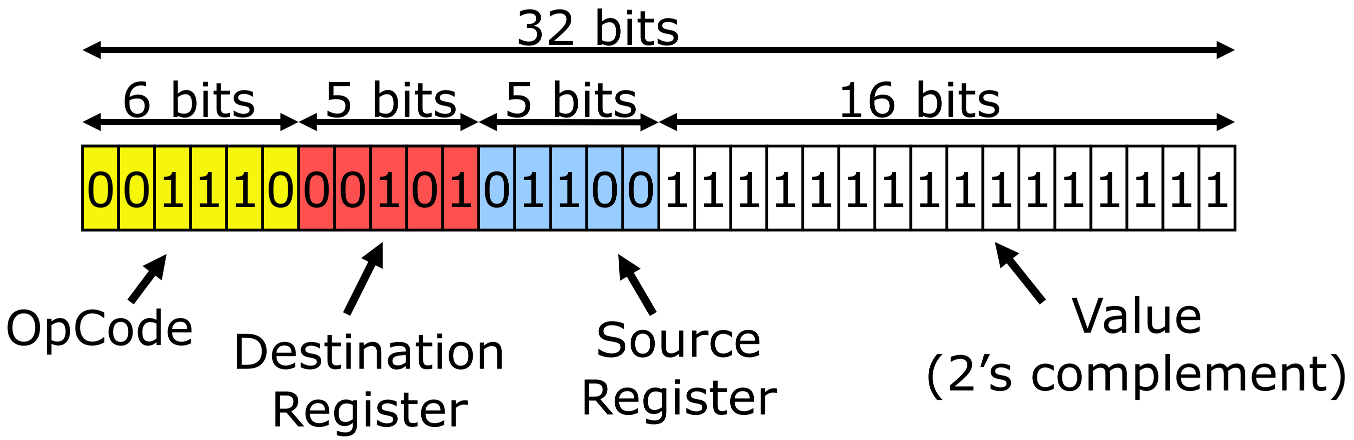
0	0	1	1	1	0	0	0	1	0	1	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- What does this example mean?

Instructions

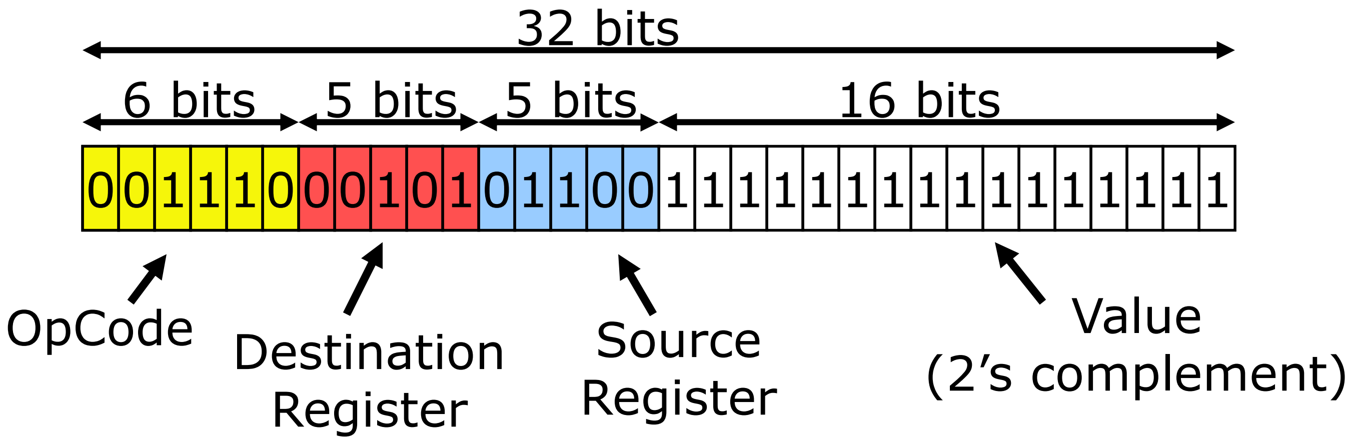
- Instructions typically consist of
  - **Opcode** (Operation code)
    - defines the operation (e.g. addition)
  - **Operands**
    - what's being operated on (e.g. particular registers or memory address)

PowerPC Instruction Example



- How do we know this?
- A machine's **Instruction Set Architecture (ISA)** defines how binary instructions are interpreted.

Instruction Set Architecture (ISA)



- OpCode (001110_2 or 14) tells us that this instruction is an integer addition and has the format shown:

destination-register = source-register + value

r5 = r12 + (-1)

Instruction Set Architecture (ISA)

- Is a specification
  - i.e. a document
- Specifies the interface between hardware and software
- We return to ISA in lectures 12 and 13

Microarchitecture

- **Microarchitecture** = The architecture of the control unit
    - i.e. the state machine (gates and flip-flops etc) that implements the control unit

ISA vs. Microarchitecture

- An Instruction Set Architecture (ISA) can be implemented by many different microarchitectures
- Examples
  - 8086 ISA is implemented by many processors – in different ways
  - Pentium ISA is implemented by
    - Pentium … Pentium IV … Core2Duo … i7
      - in different ways
    - Various AMD devices …
    - Other manufacturers also…

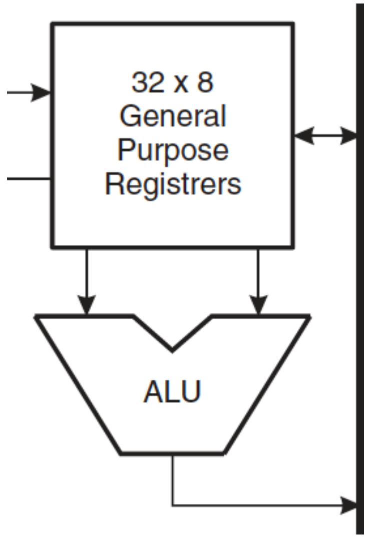
Assembly Language

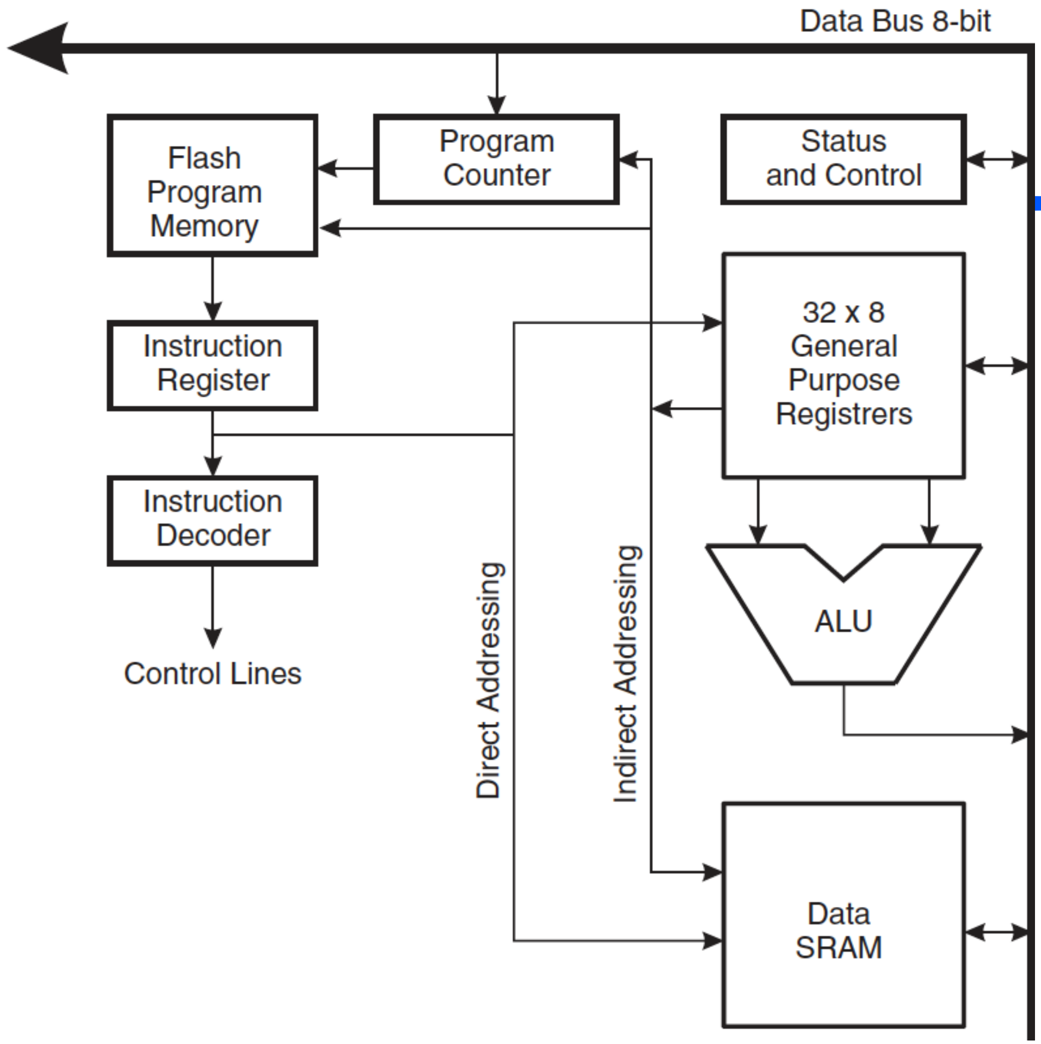
- Machine code is too hard to program in
- Use assembly language instead
  - Human readable (just!)
  - Converted into machine code by software called an assembler
- Atmel AVR examples
  - **`add r3,r4`** (means r3 ← r3 + r4)
  - **`inc r22`** (means r22 ← r22 + 1)

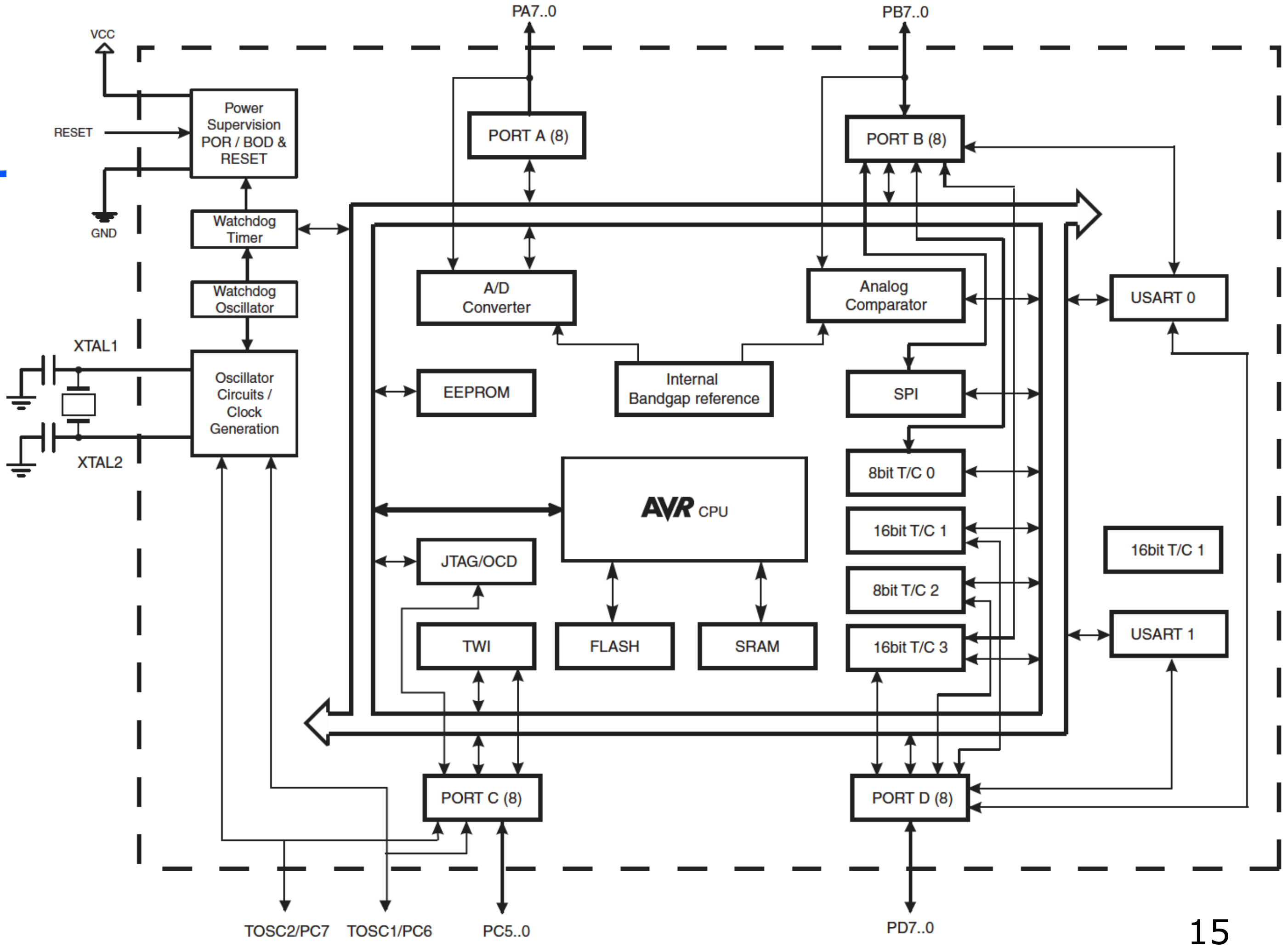
Introduction to Atmel AVR

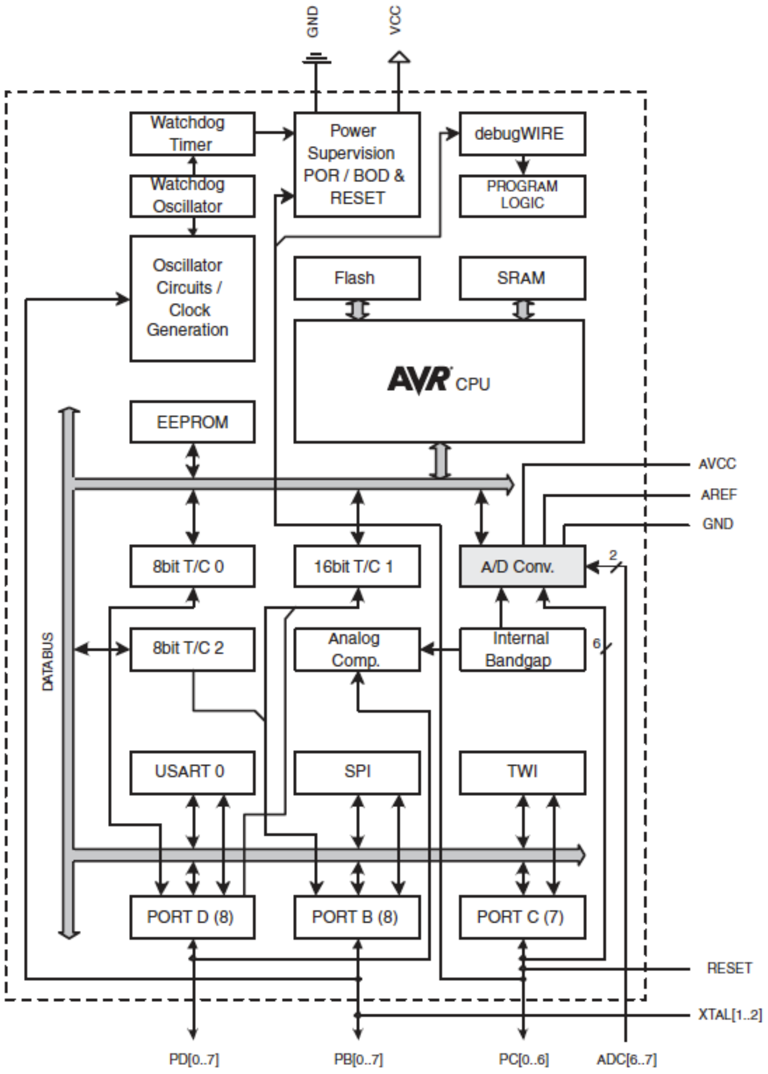
- Atmel AVR devices are **microcontrollers**
- Microcontrollers include lots of components on one chip, e.g.,
  - CPU
  - Memory (RAM)
  - Program Memory
  - Input/Output devices
- Some datasheet extracts follow
- To be discussed in class

ATmega324A Architecture




Data Bus 8-bit
Flash Program Memory
Program Counter
Status and Control
Instruction Register
32 x 8 General Purpose Registrers
Instruction Decoder
Control Lines
Direct Addressing
Indirect Addressing
ALU
Data SRAM


PA7..0
PB7..0
VCC
RESET
GND
Power Supervision POR / BOD & RESET
Watchdog Timer
Watchdog Oscillator
XTAL1
XTAL2
Oscillator Circuits / Clock Generation
PORT A (8)
PORT B (8)
A/D Converter
Analog Comparator
Internal Bandgap reference
EEPROM
SPI
USART 0
8bit T/C 0
AVR CPU
16bit T/C 1
16bit T/C 1
JTAG/OCD
8bit T/C 2
16bit T/C 3
USART 1
TWI
FLASH
SRAM
PORT C (8)
PORT D (8)
TOSC2/PC7
TOSC1/PC6
PC5..0
PD7..0


GND
VCC
Watchdog Timer
Watchdog Oscillator
Power Supervision POR / BOD & RESET
debugWIRE
PROGRAM LOGIC
Oscillator Circuits / Clock Generation
Flash
SRAM
AVR CPU
EEPROM
AVCC
AREF
GND
8bit T/C 0
16bit T/C 1
A/D Conv.
2
DATABUS
8bit T/C 2
Analog Comp.
Internal Bandgap
6
USART 0
SPI
TWI
PORT D (8)
PORT B (8)
PORT C (7)
RESET
XTAL[1..2]
PD[0..7]
PB[0..7]
PC[0..6]
ADC[6..7]

AVR Instructions

AVR assembly language instructions - examples

- **`MOV rd, rr`**
  - "move" (but actually means copy) contents of register rr (source) to register rd (destination)
  - Example: **`mov r3, r14`**

AVR Instructions (cont.)

- **`ADD rd, rr`**
  - Add contents of register rr to register rd
  - Example: **`add r5,r6`**
- **`AND rd, rr`**
- **`OR  rd, rr`**
- **`EOR rd, rr`**
  - Bitwise operations on given two registers, result goes into rd

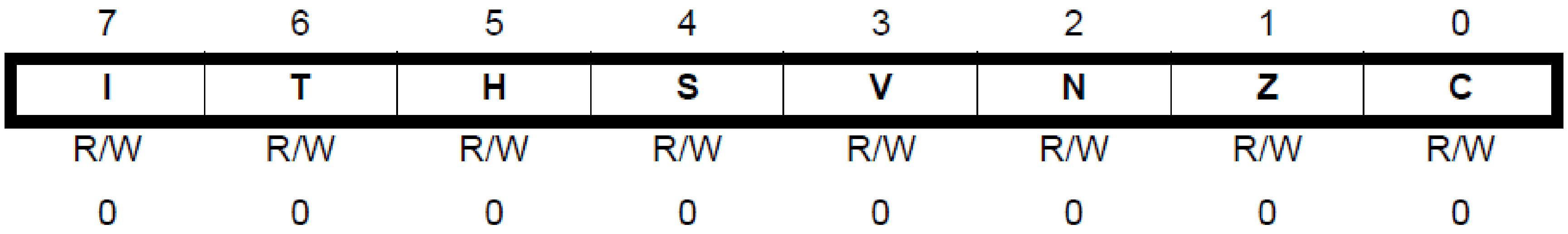
AVR Instructions (cont.)

- `LDI rd, number`
  - "load immediate" – put the specified number into the given register
  - Registers r16 to r31 only
  - 8-bit numbers (00000000 to 11111111)
    - If write in decimal can be in range -128 to 255
  - Example: `ldi r16,73`

- We'll see many more instructions in coming weeks

AVR Status Register

- 8-bits



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

AVR Machine Code

- ADD instruction
  - `0000 11rd dddd rrrr`

- Example: `add r23, r7`

AVR Machine Code (cont.)

- COM instruction
  - Flip-all bits (one's complement)
  - **`1001 010d dddd 0000`**
- Example: **`com r19`**

AVR Machine Code (cont.)

- LDI instruction
  - Load immediate
  - Only for registers r16 to r31
  - Machine code: **`1110 KKKK dddd KKKK`**
    - K=8-bit constant
    - d=register number - 16

Coming Up

- Instruction Set Architecture
  - More examples of AVR instructions