

---

# Introduction to Atmel AVR

---

## Learning Objectives

After this lecture and associated exercises and pracs, you should

- have an understanding of the basic architecture of the Atmel AVR microcontroller family

---

## Textbook References

This material is not covered by the textbook – it uses a different microprocessor. You should refer to the AVR 90S8515 datasheet (particularly the first 10 pages which give an overview of the architecture). Further Atmel AVR resources are available on the course website (within the Pracs section).

---

## Exercises

1. How can we perform 16-bit wide arithmetic operations with an ALU that supports only 8-bit wide operations? Specifically, if two 8-bit registers of a CPU contain two halves of a 16-bit number and another two registers contain two halves of another 16-bit number, what sequence of operations do we need to undertake in order to
  - (a) Add the two numbers together
  - (b) Subtract one number from the other
  - (c) Negate the first number(Think of this question in terms of any 8-bit CPU, not specifically the Atmel AVR<sup>1</sup>. The question is essentially about how to build 16-bit operations out of 8-bit operations, and importantly, what to do with carry bits.)
2. Using the datasheet as a reference, explain what the following peripherals might be used for:
  - (a) Timers/Counters
  - (b) Watchdog timer
  - (c) SPI
  - (d) UART
  - (e) Analog comparator
3. Draw a flowchart to calculate the first 15 Fibonacci numbers. Write an AVR assembly program to implement the flowchart (Write the numbers to PORTB. Assume the port has been configured correctly). (The Fibonacci sequence is given by  $F(n+2) = F(n+1) + F(n)$ . The first two are defined as being 0 and 1.)

---

<sup>1</sup> The Atmel AVR is an 8-bit microcontroller – i.e. the width of its datapath (registers and ALU) is 8 bits.