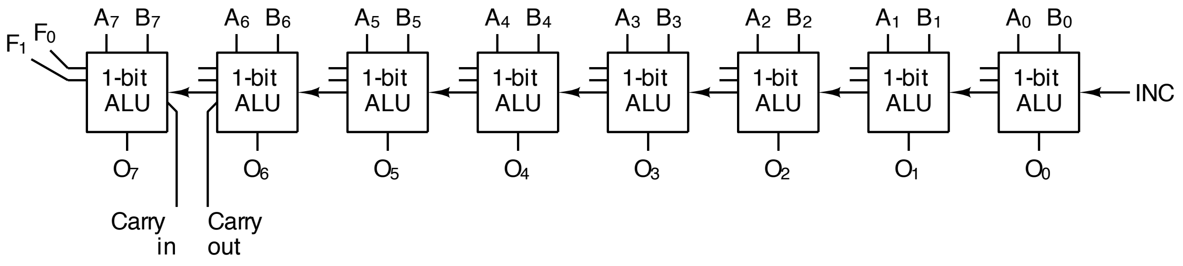
CSSE2010 / CSSE7201 – Introduction to Computer Systems Exercises – Week Six Control Unit, AVR Introduction

Exercises

Some of the questions below are taken from or based on questions in Tanenbaum, Structured Computer Organisation, 5th edition.

1. Consider the 8-bit ALU below (from Tanenbaum, also shown in lectures) where each 1-bit ALU corresponds to the bit-slice shown in lectures. The ALU has six control inputs (F1,F0,ENA, ENB, INVA, INC (carry-in)) which (except for the carry-in) are common to all the bit slices. Show the logic circuitry that can be used to generate status register bits Z, N, C, V (as defined in the lecture).



2. After each of the following operations, what would be the values of the Z (zero), V (overflow), C (carry) and N (negative) bits? (Numbers given are decimal.)
(a) 7 + 9 (in a 5 bit ALU)
(b) 7 + 9 (in an 8 bit ALU)
(c) -16 – 16 (in a 5 bit ALU)
(d) -16 – 16 (in an 8 bit ALU)

3. 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) Negate the first number
(Think of this question in terms of any 8-bit CPU, not specifically the Atmel AVR. The question is essentially about how to build 16-bit operations out of 8-bit operations, and importantly, what to do with carry bits.)

4. Write down Atmel AVR instructions (in assembly language and machine code form) which will perform the following operations (not all of these instructions are covered in lecture – use the AVR instruction reference to attempt to locate these operations and also see the machine code equivalent; some of the concepts will be covered in future weeks):
(a) increment register 5
(b) load the value 24 into register 17
(c) logically AND registers 23 and 5 and put the result in register 5
(d) increment register 7
(e) load the value 35 into register 18
(f) set all the bits of register 18 to 1
(g) set all the bits of register 18 to 0
(h) negate register 10
(i) invert register 10
(j) move the contents of register 3 to register 4