CS1021 Tutorial 5

Logic and Shift Instructions

Q1 Calculate, in hexadecimal, the results of the following 8 bit expressions

(i) \(0x96 \& 0xF\)
(ii) \(0x96 \mid 0x0F\)
(iii) \(0xAA ^ 0xF\)
(iv) \(~0xA5\)
(v) \(0x96 >> 2\)

and 32 bit expressions

(vi) \(0x0123 << 2\)
(vii) \(0x12345678 >> 24\)
(viii) \(0x12345678 >> 16\)
(ix) \((0x12345678 >> 16) \& 0xFF\)
(x) \((0x12345678 \& ~0xFF00) \mid 0x4400\)

Q2 Write ARM Assembly Language instructions to perform the following operations (assume the LSB of a register is bit 0).

(i) clear bits 4 to 7 of R0
(ii) clear the first and last bytes of R0
(iii) invert the most significant bit of R0
(iv) set bits 2 to 4 of R0
(v) swap the most and least significant bytes of R0
(vi) replace bits 8 to 15 in R0 with the value 0x44
(vii) \(R0 = R1*10\) (don’t use a multiply instruction)
(viii) \(R0 = R1*100\) (don’t use a multiply instruction)
(ix) \(R0 = R1/256\)
(x) \(R0 = R1 \% 256\) (mod operator - remainder on division)

Q3 Write an ARM assembly language program to calculate, in R0, the \((\text{sum} \% 256)\) of the 4 bytes in R1. For example, if \(R1 = 0x12345678\), \(R0 = (0x12 + 0x34 + 0x56 + 0x78) \% 256 = 0x14\).

Q4 Write and ARM assembly language program to calculate, in R0, the number of one bits in R1. For example, if \(R1 = 0x12345678\), \(R0 = 13\).