SHEET 3

Q1. Embedded systems always require the user to manipulate bits in registers or variables. Given an integer variable a, write two code fragments in C. The first should set bit 3 of a. The second should clear bit 3 of a. In both cases, the remaining bits should be unmodified.

Q2. Develop a sequence of instructions that sets the rightmost four bits of R3, clears the leftmost three bits of R3, and inverts bit positions 7,8 and 9 of R3. Assuming that R3 is 16-bit register.

Q3. When does the LR have to be pushed on the stack?

Q4. Show the SP value and the content of stack after executing this instruction PUSH {R4, R6-R8} assuming the SP initially equals 0x2000.1000 and R4=1, R6=2, R7=3, R8=4. What will be the values of the registers R0-R4 after executing this instruction POP{R0-R3}?

Q5. Explain how does the return from subroutine work in these two functions?

```
Function PUSH {R4,LR}          Function2
    ;stuff                         ;stuff
    POP  {R4,PC}                   BX LR
```

Q6. Write assembly code that pushes registers R1, R3, and R5 onto the stack.

Q7. What are the addressing modes used in each of the following instructions?

```
LDR R0, [R1]
LDR R2, [R1, #4]
MOV   R3, #100
BL function
MOV R0, #1
LDRB    R0, [PC, #0x30]
LDR R0, =1234567
```

Q8. Write a complete ARM assembly program for the procedure func2. The procedure func2 calculates this C expression ((X+Y)>>3) – Z and stores its value in R0. Assume X, Y, Z are 32-bit signed numbers. X, Y, Z are defined in the memory as shown

```
        AREA   mydata, DATA, READONLY
X       DCD    -20
Y       DCD    -60
Z       DCD    -20
```

Q9. Write a complete ARM assembly program that calls the procedure func1 which in turn calls a procedure func2. The procedure func2 is defined in Q8 of Sheet 3.