1. Assume the values in memory addresses 0x40000030, 0x40000034, 0x40000038, 0x4000003C are respectively 0x12345678, 0x87654321, 0xBEEFFACE, 0xFACEBEEF **Write a program** in the following 8 steps

(1) Use LDMIA (2) Use LDMIB (3) Use LDMDA (4) Use LDMDB

to load the values in addresses 0x40000030, 0x40000034, 0x40000038, 0x4000003C into r0, r1, r2, r3 respectively.

(5) Use STMIA (6) Use STMIB (7) Use STMDA (8) Use STMDB

to store r0, r1, r2, r3 into memory with addresses 0x40000058, 0x4000005C, 0x40000060, 0x40000064 respectively.

2. (1) **Write a program** to copy the integers from TABLE to memory space starting from address 0x4000000, **reversing the integer order** in the process. Assume the program has the following declaration initially.

TABLE DCD 0xFEBBAAAA, 0x12340000, 0x88881111, 0x22227777

DCD 0x00000013, 0x80808080, 0xFFFF0000, 0x1111FFFF

(2) **Write a program** to copy a string of characters from STRING to memory space starting from address 0x4000050, **reversing the string** in the process. Assume the program has the following declaration initially.

STRING DCB “The (source) string”, 0

(3) **Write a program** to **reverse** the bits in R0 so that the register R0 containing b31b30b29…b1b0 now contains b0b1…b29b30b31, assuming R0 contains the value 0xFACEBEEF initially.

3. Assume the word at memory address 0x40000000 = 0xCDEF6543, the word at memory address 0x40000004 = 0x8765ABCD, the word at memory address 0x40000008 = 0x87654321, the word at memory address 0x4000000C = 0xFACEBEEF. **Write a program** that includes **6 subroutines** **sub1**, **sub2**, **sub3**, **sub4**, **sub5, sub6** and calls to the 6 subroutines one by one.

(1) **sub1** puts even parity for bits 4, 5, 6, 10 and 12 of the word at memory address 0x40000000 into bit 9 of the word at memory address 0x40000004 (**Note: Be sure to use TST and do not use EOR**.)

(2) **sub2** puts odd parity for bits 5, 8, 12, 15 and 18 of the word at memory address 0x40000000 into bit 19 of the word at memory address 0x40000004. (**Note: Do not use TST and do not use EOR**.)

(3) **sub3** puts bits 30, 29, 28, 27, 15, 14, 12, 11, 10 of the word at memory address 0x40000000 into bits 24~16 of the word at memory address 0x40000004 respectively,

(4) **sub4** puts the number of different bits between the word at memory address 0x40000008 and the word at memory address 0x4000000C into r0 **using TST**.

(5) **sub5** puts the number of different bits between the word at memory address 0x40000008 and the word at memory address 0x4000000C into r1 **without using TST**.

(6) **sub6** puts the different bit numbers between the word at memory address 0x40000008 and the word at memory address 0x4000000C one by one in the words started from memory address 0x40000070.

(Do not forget to use **STMDB** with **SP!** and the corresponding **LDM** in each subroutine to avoid the side effect and **highlight** **the memory locations with the related registers stored using STM and the registers restored using LDM**. **Assume SP = 0x40000020 initially.**)

4. Assume the word at memory address 0x40000030 = 0xFACEBEEF, the word at memory address 0x40000034 = 0xCDEF5678, the word at memory address 0x40000038 = 0x5678CDEF, the word at memory address 0x4000003C = 0xBEEFFACE initially.

(1)Write **a subroutine ParChk** **to check parity**. **Subroutine ParChk** gets the input word in R7, check the parity of the word in R7 and write the value 0xFACEBEEF/0xBEEFFACE if odd parity/even parity into R8.(**Note: Do not use TST**.)

(2) **Call subroutine ParChk to check what parity (odd or even)** the 4 words at addresses 0x40000030, 0x40000034, 0x40000038 and 0x4000003C have and write the value 0xFACEBEEF/0xBEEFFACE if odd parity/even parity into the word at addresses 0x40000060, 0x40000064, 0x40000068 and 0x4000006C respectively.

(Do not forget to use **STMIA** with **SP!** and the corresponding **LDM** in each subroutine to avoid the side effect and **highlight** **the memory locations with the related registers stored using STM and the registers restored using LDM**. **Assume SP = 0x40000020 initially.**)

**Note:** Please

1. put necessary **Keil Tool DEBUG window screenshots** to show your **program** and **execution results** including **highlighted necessary initial assumptions and subsequent memory and register changes**,
2. **comment student ID+your English name in every screenshots**, and
3. put reports into one word file named by student\_ID+your\_name.