**Introduction to Computer Organisation and Architecture**

**Tutorial-7**

1. Write the following procedure with ARM assembly language. Assuming that the variable A and B are initialized with the value of 15 and 12, respectively, complete the code below

|  |
| --- |
| procedure CALC is  A,B: 32 bit unsigned integer;  begin  C := (A+B)\*(A-B)  end CALC; |

|  |
| --- |
| TTL \_\_\_\_CALC C\_\_\_\_\_\_\_\_\_\_\_\_\_\_  AREA \_\_\_\_\_\_\_Program, CODE, READONLY\_\_\_\_\_\_\_\_\_\_\_  ENTRY \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Main  \_\_\_\_\_ LDR R1, A\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ; Load A  \_\_\_\_\_ LDR R2, B\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ; Load B  \_\_\_\_\_ ADD R3, R1, R2\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ; Perform A + B  \_\_\_\_\_ SUB R4, R1, R2\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ; Perform A - B  \_\_\_\_\_ MUL R5, R3,R4\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ; Perform (A+B)\*(A-B)  \_\_\_\_\_ STR R3, A\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ; Save result to A  \_\_\_\_\_ BAL HERE\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ; End Statement    AREA \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ; start of data  \_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ; Define A  \_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ; Define B  \_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ; Define C  END |

1. The program in Q1 may have overflow problem if the value of A and B is large. For example, when variable A=0x00020002 and B=0x00010001 where the result requires the spaces of two words = 0x0000000300060003.

Explore the command UMULL (given below) to overcome the problem. UMULL performs unsigned multiplication of two registers (32 bit) and the result is stored in two separate result registers (64 bits). Assume a Big Endian memory scheme where the more significant word is stored at a lower address as follows:

Sample Problems:

*Input*:

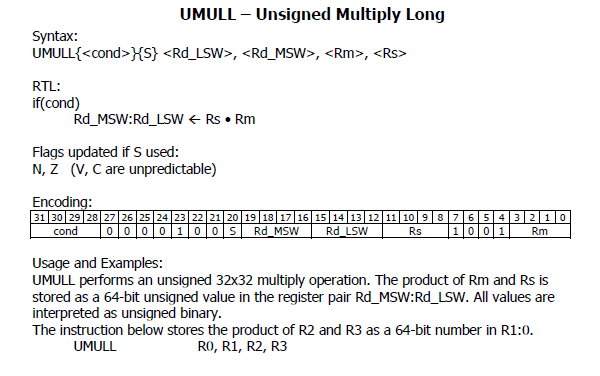
[A] = 0x00020002

[B] = 0x00010001

*Output*:

[RESULT] = 0x0000000300060003

Modify the code in Q1 to cater for this requirement.



1. The distance traveled by a car undergoing constant acceleration is given by the following formula:

Write an assembly program which performs the above computation. Assume that the parameters and variables have been loaded into the memory as follows:

|  |  |
| --- | --- |
| *Velocity* | 10 |
| *Acceleration* | 6 |
| t | 5 |
| s |  |

LDR R1, VEL

LDR R2, ACC

LDR R3, t

MUL R4, 41, R3

MUL R5, R2, R3

MUL R6, R5, R3

ADD R7, R6, LSR #1

VEL DCD 10

ACC DCD 6

T DCD 5

S DCD 0

1. Combine an array of **four bytes** into one 16-bit halfword. The first item should go into the most significant nibble of the result. Store the result in the 32-bit variable result.

Sample Problems:

*Input*:

[LIST] = 0xC, 0x2, 0x6, 0x9

*(Each item is a byte, Use DCB to define a variable of type byte)*

*Output*:

[RESULT] = 0x0C0206209

LDR R0, =LIST

LDRB R1, [R0], #1

LDRB R2, [R0], #1

LDRB R3, [R0], #1

LDRB R4, [R0]

LDRB R0, =RESULT

STRB R1,[R0], #1

STRB R2, [R0] #1

STRB R3, [R0]#1

STRB R4, [R0]

BAL HERE

1. Write an assembly language which performs the following:

*Description:*

Disassemble a word into four bytes. For example, given a word 0xFE13F7AA, the program will produce 0x000000FE, 0x00000013, 0x000000F7 and 0x000000AA. Use the logical operation AND and Shifter Operand to extract the byte.

*Input:*

[WORD] = 0xFE13F7AA

*Output:*

[B3] = 0x000000FE

[B2] = 0x00000013

[B1] = 0x000000F7

[B0] = 0x000000AA

**EXTRA QUESTIONS**

1. Write an assembly language which performs the following:

Description:

Given an array of items (in bytes), combine the bytes in the odd position of the array into a single word in a reverse manner.

Restriction:

Only one memory write is allowed for the operation

*Input (Each item is a byte, Use DCB to define a variable of type byte):*

[ARR] = 0x01, 0xA2, 0x03, 0xFC, 0x2C 0x07, 0x1C, 0x2B

*Output:*

[RESULT] = 0x2B07FCA2