**Name of the student: Komal Shrikant Jainar**

**USN: 01FE23BEC298**

**Div: E**

**Roll No: 564**

**EXPERIMENT NO. 1**

**32 - bit two number addition / subtraction**

|  |  |  |
| --- | --- | --- |
| **Programs** | **Time in microseconds** | **Memory space in bytes** |
| **Code 1**  AREA ADDITION,CODE,READONLY  ENTRY  ADD R0,R0,R1;  STOP B STOP  END | 0.333 | 8 |
| **Code 2**  AREA ABC,CODE,READONLY  ENTRY  LDR R0,=0x12345678;  LDR R1,=0x87654321;  ADD R0,R0,R1;  STOP B STOP  END | 0.833 | 24 |
| **Code 3**  TTL ADD32  AREA AA, CODE, READONLY  ENTRY  MAIN LDR R3, =0x40000000 ;load the starting address  LDR R0, [R3] ;get the first data  LDR R1, [R3,#0x04] ;get the second data  ADDS R2, R0, R1; add the data & store in reg r2  STOP B STOP  END | 0.916 | 20 |
| **Code 4**  TTL ADD32  AREA AA, CODE, READONLY  ENTRY  MAIN LDR R3, =VAL1 ;load the starting address  LDR R0, [R3] ;get the first data  LDR R3, =VAL2 ;load the second address  LDR R1, [R3] ;get the second data  ADDS R2, R0, R1 ;add the data & store in reg r2  STOP B STOP  AREA CODE, READONLY  VAL1 DCD 0x00000010  VAL2 DCD 0x00000020  END | 1.333 | 40 |

**64 – bit two numbers addition / subtraction**

|  |  |  |
| --- | --- | --- |
| **Programs** | **Time in microseconds** | **Memory space in bytes** |
| **Code 1**  AREA ADDITION,CODE,READONLY  ENTRY  ADD R4,R1,R0;  ADD R5,R3,R2;  STOP B STOP  END | 0.416 | 12 |
| **Code 2**  TTL ADD64  AREA AA, CODE, READONLY  ENTRY  MAIN LDR R0, =0X00000020 ;load 32-bit data  LDR R1, =0X00000080 ;load 32-bit data  ADDS R2, R0, R1 ;add 32-bit data  LDR R3, =0X00000050 ;load 32-bit data  LDR R4, =0X00000020 ;load 32-bit data  ADCS R5, R3, R4 ;add 32-bit data along with carry  STOP B STOP  END | 0.748 | 28 |
| **Code 3**  TTL ADD64  AREA AA, CODE, READONLY  ENTRY  MAIN LDR R4, =0x40000000  LDR R0, [R4]  LDR R1, [R4,#0x04]!;  ADDS R5, R1, R0    LDR R2, [R4,#0x04]!;  LDR R3, [R4,#0x04]!;  ADCS R6, R3, R2  STOP B STOP  END | 1.499 | 32 |
| **Code 4**  TTL ADD64  AREA AA, CODE, READONLY  ENTRY  MAIN LDR R4, =VAL1 ;load the starting address  LDR R0, [R4] ;get the first data  LDR R4, =VAL2 ;load the second address  LDR R1, [R4] ;get the second data  ADDS R5, R1, R0;add the data & store in reg r2  LDR R4, =VAL3 ;load the starting address  LDR R2, [R4] ;get the first data  LDR R4, =VAL4 ;load the second address  LDR R3, [R4] ;get the second data  ADCS R6, R3, R2 ;add the data & store in reg r2  STOP B STOP  AREA CODE, READONLY  VAL1 DCD 0x00000010  VAL2 DCD 0x00000020  VAL3 DCD 0X00000030  VAL4 DCD 0X00000040  END | 2.416 | 76 |

**Multiplication of two 32 bit numbers**

|  |  |  |
| --- | --- | --- |
| **Program** | **Time in microseconds** | **Memory space in bytes** |
| **Code 1**  TTL MUL32  AREA ML, CODE, READONLY  ENTRY  MULS R2,R1,R0;  STOP B STOP  END  **Code 2**  TTL MUL32  AREA ML, CODE, READONLY  ENTRY  MAIN LDR R0, =0X00000005 ; load 32-bit data  LDR R1, =0X00000009 load 32-bit data  MULS R2, R1, R0 ;multiply the 32-bit data  STOP B STOP  END | 0.417  0.583 | 8  16 |
| **Code 3**  TTL MUL32  AREA ML, CODE, READONLY  ENTRY  LDR R3,=0x40000000;  LDR R0,[R3];  LDR R1,[R3,#0x04]!;  MULS R2,R1,R0;  STOP B STOP  END | 1 | 20 |
| **Code 4**  TTL MUL32  AREA ML, CODE, READONLY  ENTRY  MAIN LDR R3, =val1  LDR R0,[R3]  LDR R3, =val2  LDR R1,[R3]  MULS R2, R1, R0  STOP B STOP  val1 DCD 0x00004321  val2 DCD 0x00001234  END | 1.500 | 40 |

**Division of two 32 bit numbers**

|  |  |  |
| --- | --- | --- |
| **Program** | **Time in microseconds** | **Memory space in bytes** |
| **Code 1**  TTL DIV32  AREA DIV, CODE, READONLY  ENTRY  MAIN LDR R0, =0X08 ;load divisor into r0  LDR R1, =0X02 ;load dividend into r1  BACK SUBS R2, R0, R1  ADD R3, R3, #1  CMP R1, R2  BHI STOP  MOV R0, R2  b BACK  STOP B STOP  END | 2.917 | 36 |
| **Code 2**  TTL DIV32  AREA DIV, CODE, READONLY  ENTRY  MAIN LDR R4, =dividend  LDR R5, =divisor  LDR R0, [R4]  LDR R1, [R5]  MOV R3, #0  BACK CMP R0, R1  BLT STOP  SUB R0, R0, R1  ADD R3, R3, #1  B BACK  STOP B STOP  dividend DCD 0x00000036  divisor DCD 0x00000004  END | 9.250 | 60 |

**EXPERIMENT NO. 2**

**Sum of N 16bit numbers in a given array**

|  |  |  |
| --- | --- | --- |
| **Program** | **Time in microseconds** | **Memory space in bytes** |
| **Code 1**  TTL SUM16  AREA SUM, CODE, READONLY  ENTRY  MAIN LDR R0, =VAL1;initialise the sum to zero  LDR R1, =0X00 ;initialize the counter  LDR R2, =0X05  LOOP LDRH R3, [R0], #0x02 ; pointed by r0  ADD R1, R1, R3 ;add the numbers  SUBS R2, R2, #0X01 ;decrement the counter  BNE LOOP  STOP B STOP  AREA VAL, CODE  VAL1 DCW 10, 20,30,40,50,60  END | 3.833 | 48 |
| **CODE 2**  AREA SUM16,CODE,READONLY  ENTRY  LDR R0,=0X40000000;  MOV R3,R4; // No. of values to sum  LOOP LDRH R1,[R0],#0X02;  ADD R2,R2,R1;  SUBS R3,R3,#0X01;  BNE LOOP  STOP B STOP  END | 3.583 | 28 |

**Maximum/ Minimum number in given array**

|  |  |  |
| --- | --- | --- |
| **Program** | **Time in u.sec.** | **Memory space in bytes** |
| **Code 1**  AREA LARGESTNUM,CODE,READONLY  ENTRY  LDR R0,=ARRAY ; Load the address of the array into R0  MOV R1,#0x05 ; Set number of elements in the array  LDR R2,[R0]; Load the first element of the array into R2 (assume it's the largest)  ADD R0,R0,#0x04 ; Move to the next element    REPEAT SUBS R1,R1,#0x01 ; Decrement the counter  BEQ STOP ; If counter is zero, stop  LDR R3,[R0],#0x04 ; Load the next element of the array into R3 and post-increment R0  CMP R2,R3 ; Compare R2 with R3  MOVLT R2,R3 ; If R3 is greater than R2, move R3 to R2  BNE REPEAT ; Repeat the loop  STOP B STOP ; Infinite loop to stop execution  ARRAY DCD 0x11111111,0x22222222,0x33333333,0x55555555,0x44444444  END | 4.583 | 68 |
| **Code 2**  TTL MAX  area largest , code, readonly  entry  start  mov r5,#6 ; intialise counter to 6(i.e. n=7)  ldr r1,=value1 ; loads the address of first value  ldr r2,[r1],#4 ; word align t0 array element  loop  ldr r4,[r1],#4 ; word align t0 array element  cmp r2,r4; compare numbers  bhi loop1 ; if the first number is > then goto loop1  mov r2,r4; if the first number is < then mov content r4 to r2  loop1  subs r5,r5,#1 ; decrement counter  cmp r5,#0 ; compare counter to 0  bne loop ; loop back till array ends  ldr r4,=result ; loads the address of result  str r2,[r4] ; stores the result in r2  stop b stop    value1  dcd 0x44444444,0x22222222,0x11111111,0x33333333,0xaaaaaaaa,0x88888888,0x99999999  area data2,data,readwrite  result dcd 0x0  end | 7 | 92 |

**Factorial of a given number**

|  |  |  |
| --- | --- | --- |
| **Program** | **Time in u.sec.** | **Memory space in bytes** |
| **Code 1**  AREA FACTORIAL,CODE,READONLY  ENTRY  LDR R0,=0x00000001;  LDR R1,=0x00000005;  LOOP MUL R2,R1,R0;  MOV R0,R2;  SUBS R1,R1,#0x01;  BNE LOOP  STOP B STOP  END | 3.167 | 28 |
| **Code 2**  area factorial,code  entry  main  mov r0,#5  mov r1,r0  mov r3,r0  loop subs r1,r1,#01  mulne r2,r1,r0  mov r0,r2  bne loop  stop b stop  end | 3.167 | 32 |
| **Code 3**  TTL FACTORIAL  AREA FACT, CODE, READONLY  ENTRY  MAIN LDR R0, =dd;load address from lookup table  LDR R4, =VAL ;load the input  LDR R1, [R4];  SUBS R1,R1,#0X01;  MOV R1, R1, LSL #0x02 ;move the shifted data to reg  ADD R0, R0,R1;add the data in r0 with shifted data  LDR R2, [R0]  LDR R3, =0X40000000  STR R2, [R3]  STOP B STOP  AREA dd, CODE  DCD 1  DCD 2  DCD 6  DCD 24  DCD 120  DCD 720  DCD 5040  VAL DCB 5  END | 1.750 | 80 |
| **Code 4**  AREA SUBFACT,CODE,READONLY  ENTRY  MOV R0,#0X05;  MOV R1,R0;  BACK SUBS R1,R1,#0X01;  BEQ STOP;  BL FACTORIAL;  B BACK;  STOP B STOP  FACTORIAL MUL R2,R1,R0;  MOV R0,R2;  MOV PC,LR; //also BX LR can be replaced  END | 5.417 | 40 |