;64 bit substraction

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT main

; Start of CODE area

main

LDR r0,=0x00000001 ; X\_Low(X = 0x0000000100000001)

LDR r1,=0x00000001 ; X\_High

LDR r2,=0x00000003 ; Y\_Low(Y = 0x0000000000000003)

LDR r3,=0x00000000 ; Y\_High

SUBS r0,r0,r2 ; lower 32-bit

SBCS r1,r1,r3 ; upper 32-bit

END

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT \_\_main

; Start of CODE area

\_\_main

LDR r2,=0x00000080;

ASRS r0,r2,#04

LDR r2,=0x80000000;

ASRS r0,r2,#04

stop B stop

END

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT \_\_main

; Start of CODE area

\_\_main

LDR r3,=0x20000100

LDR r0,=0x20000050

LDMIA r3!,{r1,r2}

mov SP,r0

PUSH {r1,r2}

POP {r4,r5}

stop B stop

END ; End of file

extrating data

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT main

; Start of CODE area

main

ldr r1,=0xffffffff

movs r2, #0xff

lsls r2,r2,#16

bics r1,r1,r2

movs r2, #0xc0

lsls r2,r2,#16

orrs r1,r1,r2

str r1,[r0]

stop b stop

end

;program to use compare and branch less than

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT main

; Start of CODE area

main

LDR R0,=0X00000001

LDR R1,=0X00000002

strt3 cmp r0,#10

ble start2

movs r0,#0

b stop

start2 adds r0,r0,#1

b strt3

stop B stop ;stop program

END

;CASE BRANCHING

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT main

; Start of CODE area

main

LDR R0, =0

CMP R0, #3 ; Compare input to maximum valid choice

BHI default\_case ; Branch to default case if higher than 3

MOVS R2, #4 ; Multiply branch table offset by 4

MULS R0, R2, R0 ; (size of each entry)

LDR R1, =BranchTable ; Get base address of branch table(0x284)

LDR R2,[R1,R0] ; Get the actual branch destination

BX R2 ; Branch to destination

ALIGN 4 ; Alignment control. The table has

BranchTable ; to be word aligned to prevent unaligned read

; table of each destination address

DCD Dest0

DCD Dest1

DCD Dest2

DCD Dest3

default\_case

stop B stop

; Instructions for default case

Dest0 ldr r0, =10

stop1 B stop1

; Instructions for case ‘0’

Dest1 ldr r0, =20

stop2 B stop2

; Instructions for case ‘1’

Dest2 ldr r0, =30

stop3 B stop3

; Instructions for case ‘2’

Dest3 ldr r0, =40

stop4 B stop4

; Instructions for case ‘3’

stop5 B stop5

END

;Copy hyundered locations

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT main

; Start of CODE area

main

LDR r0,=0x00000000 ; Source address

LDR r1,=0x20000000 ; Destination address

LDR r2, =100 ; number of bytes to copy

copy\_loop

LDRB r3, [r0] ; read 1 byte

ADDS r0, r0, #1 ; increment source pointer

STRB r3, [r1] ; write 1 byte

ADDS r1, r1, #1 ; increment destination pointer

SUBS r2, r2, #1 ; decrement loop counter

BNE copy\_loop ; loop until all data copied

stop B stop

END

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT main

; Start of CODE area

main MOVS R6,#05 ; factorial of no

MOVS R4, R6

SUBS R4,R4,#1

LOOP

MOVS R7,R4

MULS R7,R6,R7

MOVS R6,R7

SUBS R4,R4,#1

BNE LOOP

stop B stop ; R7 ANSWER

END

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT main

; Start of CODE area

main

ldr r1, =0x50004080

ldr r2, =0x08

ldr r3, =0x01

lsls r4, r3, #30

str r4, [r1]

ldr r4, =0x0f

adds r1, r1,r2

str r4, [r1]

stop b stop

end

;Program using loop

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT main

; Start of CODE area

main

MOVS R0, #5 ; Loop counter

loop

SUBS R0, R0, #1 ; Decrement loop counter

BNE loop ; if result is not 0 then branch to loop

END

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT \_\_main

; Start of CODE area

\_\_main

LDR r0,=0x80000001;

LSLS r2,r0,#01

LDR r0,=0x80000001;

LSLS r2,r0,#02

stop B stop

END

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT \_\_main

; Start of CODE area

\_\_main

LDR r0,=0x80000001;

LSRS r2,r0,#31

LDR r0,=0x80000001;

LSRS r2,r0,#30

stop B stop

END

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT \_\_main

; Start of CODE area

\_\_main

LDR r3,=0x20000100

LDR r0,=0x20000050

LDMIA r3!,{r1,r2}

mov SP,r0

PUSH {r1,r2}

POP {r4,r5}

stop B stop

END ; End of file

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT \_\_main

; Start of CODE area

\_\_main

LDR r0,=0x80000001;

MOVS r2,#31;32-1

RORS r0,r2

LDR r0,=0x80000001;

MOVS r2,#30

RORS r0,r2

stop B stop

END

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT main

; Start of CODE area

main

LDR R0,=0XF631024C

LDR R1,=0X17539ABD

EORS R0,R0,R1 ;R0^R1

EORS R1,R0,R1 ;R1^R0

EORS R0,R0,R1 ;R0^R1

stop B stop ;stop program

END

PRESERVE8 ; Indicate the code here preserve

; 8 byte stack alignment

THUMB ; Indicate THUMB code is used

AREA |.text|, CODE, READONLY

EXPORT \_\_main

; Start of CODE area

\_\_main

LDR r0,=0xF0000000;

LDR r2,=0xF0000000;

TST r0,r2;

MRS r3,XPSR;updating only N and Z flags not C

LDR r0,=0x70000000;

LDR r2,=0x70000000;

TST r0,r2;

MRS r4,XPSR;updating only N and Z flags not C

stop B stop

END