CS1021 Tutorial 8

Stacks and Subroutines

Q1 If SP = 0x40010000, R4 = 4, R5 = 5 and R6 = 6 (1) draw a diagram of the stack after the following instructions are executed and (2) what are the contents of R4, R5, and R6?

(i) PUSH {R4}
PUSH {R5}
POP  {R6}

(ii) PUSH  {R4, R5}
POP  {R4, R5}

(iii) PUSH  {R4, R5}
POP  {R5, R4}
(iv)  \[ \text{PUSH \{R4, R5, R6\}} \]
\[ \text{POP \{R6\}} \]
\[ \text{POP \{R5\}} \]
\[ \text{POP \{R4\}} \]

... 4 5
0x4000FFF0
0x4000FFF4
...
0x4000FFFC
0x4000FFF8
SP no RAM
no RAM
0x40010000
0x40010004
...
6
R4 = 6, R5 = 5, R6 = 4

(v)  \[ \text{PUSH \{R4, R5, R6\}} \]
\[ \text{POP \{R6, R4\}} \]
\[ \text{POP \{R5\}} \]

... 4 5
0x4000FFF0
0x4000FFF4
...
0x4000FFFC
0x4000FFF8
SP no RAM
no RAM
0x40010000
0x40010004
...
6
R4 = 4, R5 = 6, R6 = 5

Q2  If SP = 0x40010000 and R4 = 0, what do the following instructions do?

\[ \text{PUSH \{R4\}} \quad ; \text{PC = 0 (branch to 0)} \]
\[ \text{POP \{PC\}} \]

Q3  Write suitable entry and exit code for a leaf subroutine XXXX which modifies R4, R5, R6 and R7.

\\begin{verbatim}
XXXX PUSH \{R4, R5, R6, R7\}; push R4, R5, R6 and R7
... ;
... ;
POP \{R4, R5, R6, R7\}; pop R4, R5, R6 and R7
BX LR; return
\\end{verbatim}

Q4  Write suitable entry and exit code for a non-leaf subroutine YYYY which modifies R4, R5, and R7.

\\begin{verbatim}
YYYY PUSH \{R4, R5, R7, LR\}; push R4, R5, R7 and return address
... ;
... ;
POP \{R4, R5, R7, PC\}; pop R4, R5, R7 and return
\\end{verbatim}
Q5 Write a subroutine STRLEN which returns the length of NUL terminated ASCII string in R0. The address of the string is passed to the subroutine in R0.

; leaf subroutine

STRLEN  MOV  R1, R0  ; R1->str
      MOV  R0, #0  ; R0 = 0
STREN0  LDR  R2, [R1], #1  ; R2 = ch AND R1 = R1 + 1
      CMP  R2, #0  ; ch == 0?
      BEQ  STREN1  ; finished
      ADD  R0, R0, #1  ; R0 = R0 + 1
      B  STREN0  ; next ch
STREN1  BX  LR  ; return

Q6 Write a subroutine LEN that computes $\sqrt{x^2 + y^2}$. Assume x is passed to the subroutine in R0, y in R1 and that the result is returned in R0. Assume also that you can call a subroutine SQRT which the returns the integer square root of R0 in R0.

If a is stored @ 0x40000000, b @ 0x40000004 and c @ 0x40000008 respectively, write code, using subroutine LEN, to compute $c = \sqrt{a^2 + b^2}$.

; non-leaf subroutine (calls SQRT)

LEN  PUSH  {LR}  ; push return address
      MUL  R2, R0, R0  ; R2 = x*x
      MUL  R0, R1, R1  ; R0 = y*y
      ADD  R0, R2, R0  ; R0 = x*x + y*y
      BL  SQRT  ; R0 = sqrt(x*x + y*y)
      POP  {PC}  ; return

MAIN  LDR  R4, =0x40000000  ; R4 -> a (use R4 as it will not be modified by LEN)
      LDR  R0, [R4], #4  ; R0 = a AND R4 -> b
      LDR  R1, [R4], #4  ; R1 = b AND R4 -> c
      BL  LEN  ; R0 = sqrt(a*a + b*b)
      STR  R0, [R4]  ; c = sqrt(a*a + b*b)