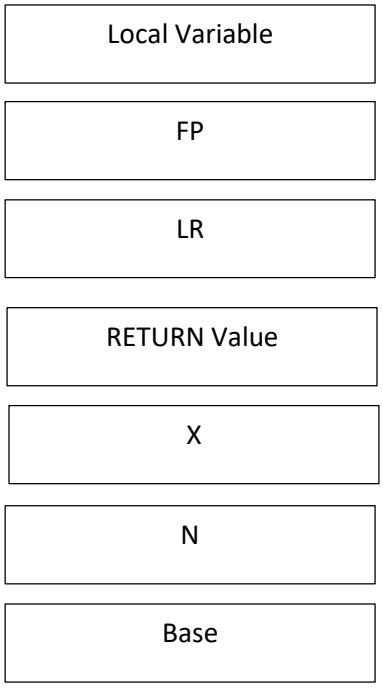
CS2208b Assignment 5
April 4, 2018
Jacob Su 250844774

Source Code

```
        AREA    power, CODE, READONLY
x               EQU     3
n               EQU     6
                ENTRY

MAIN    ADR             sp, stack               ;Define the stack
                MOV     r0, #x                  ;Store x into r0 from data
                MOV     r1, #n                  ;Store n into r1 from data
                STMFD   sp!,{r0-r1}             ;Store x and n on the stack
                SUB     sp, sp, #4              ;reserve a place in the stack for return value
                BL              POW             ;Branch to power function

                LDR             r0,[sp],#4      ;Load the result in r0 and pop it from the stack
                ADD             sp,sp,#8        ;Remove the parameters from the stack
                ADR             r1, result      ;Get the address of the result variable
                STR             r0,[r1]         ;Store the final result in the result variable

LOOP    B               LOOP                    ;Infinite loop

POW             STMFD   sp!,{r0,r1,r2,fp,lr}    ;Push general registers, as well as fp and lr on
                                                ;the stack
                MOV     fp, sp                  ;Set the fp for this call
                SUB             sp, sp, #4      ;Create space for local variable y
                LDR     r1, [fp, #0x1C]         ;Get n parameter from the stack
                LDR             r0, [fp, #0x18] ;Get x parameter from the stack

                CMP             r1, #0          ;Check base case (if n == 0)
                MOVEQr1, #1                     ;Store 1 into r1
                STREQ   r1,[fp,#0x14]           ;Store return value in stack
                BEQ             RET             ;Branch to return function

                TST             r1,#1           ;Check if n is odd
                BEQ             EVEN            ;Branch to even subroutine if it is even

                SUBNE   r1,#1                   ;Decrement n by 1
                STMFD   sp!,{r0-r1}             ;Store x and n on the stack
                SUBNE   sp,sp,#4                ;Reserve a space in the stack for the return
value
                BLNE    POW                     ;Branch to POW subroutine

                LDR             r1, [sp], #4    ;Load the return value into r1
                ADD             sp, sp, #8      ;Remove the parameters from the stack
```

```
                MUL         r2, r1, r0          ;Multiply the return value by x and store it in r2
                STR         r2, [fp,#0x14]      ;Store r2 on the stack
                B           RET                 ;Branch to RET subroutine

EVEN    LSR             r1, #1                  ;Divide n by 2
                STMFD   sp!,{r0-r1}             ;Store x and n on the stack
                SUB         sp, sp, #4          ;Reserve a place in the stack for the return
value
                BL          POW                 ;Branch to POW subroutine

                LDR         r1, [sp], #4        ;Load return value
                ADD         sp, sp, #8          ;Clear stack space
                MUL     r2, r1, r1              ;Multiply return value with itself and store in r2
                STR         r2,[fp,#0x14]       ;Store value into memory
                B           RET                 ;Branch to return function

RET             MOV         sp,fp               ;collapse all working spaces for this function call
                LDMFD   sp!,{r0,r1,r2,fp,pc}    ;Load all registers and return to the caller

                AREA    power, DATA, READWRITE
result  DCD             0x00                    ;The final result
                space   0xFF                    ;Declare the space for the stack
stack   DCD             0x00                    ;Initial stack position (FD model)
                END
```

Structure of the Stack Frame



How many stack frames are needed to calculate x^n, when n = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12?

- N = 0: 8 frames
- N = 1: 9 frames
- N = 2: 10 frames
- N = 3: 11 frames
- N = 4: 11 frames
- N = 5: 12 frames
- N = 6: 12 frames
- N = 7: 13 frames
- N = 8: 13 frames
- N = 9: 14 frames
- N = 10: 14 frames
- N = 11: 15 frames
- N = 12: 15 frames