Source Code

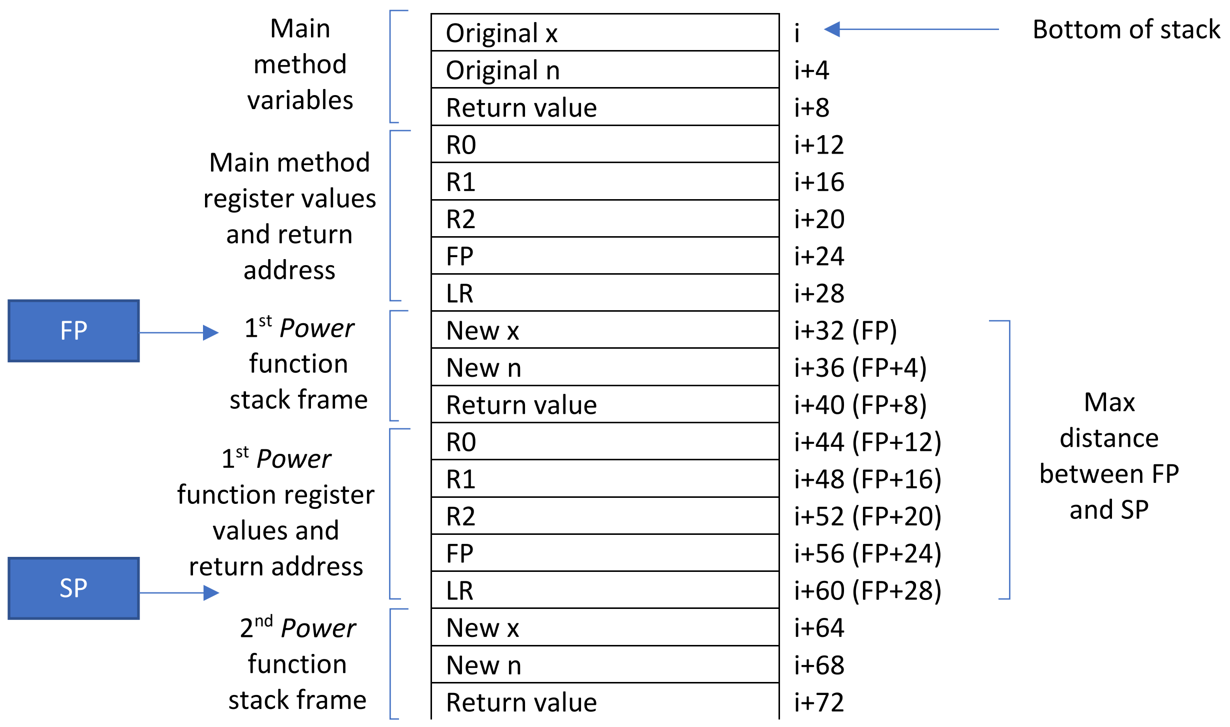
```
          ;CS2208 Assignment 5
          ;Author: Fengyi Zhu
          ;Student#: 250903071
          ;Date: 4/3/2018
          ;Program Description: This program calculates x^n using a recursive function,
          ;  x and n are passed-by-value through the stack. The returned value is then
          ;  stored in a local variable called "result".
          ;Length: 31 lines

          AREA power, CODE, READWRITE
          ENTRY
Main      ADR fp, stack                 ;store the address of the bottom of the stack in fp
          ADR sp, stack                 ;store the address of the bottom of the stack in sp
          MOV r0, #_X                   ;get the value of x through r0
          MOV r1, #_N                   ;get the value of n through r1
          STMIA sp!, {r0-r2}            ;store r0 to r2 in stack
          BL Power                      ;call function Power
          LDMDB sp!, {r0-r2}            ;after returned from call, pop the parameters and returned value from the function
          STR r2, result                ;store the returned value in r2 to "result"
Finished  B Finished                    ;end of program

Power     STMIA sp!, {r0-r2, fp, lr}    ;store r0 to r2, fp and lr in stack
          LDMIA fp, {r0, r1}            ;load parameters into r0 and r1
          MOV fp, sp                    ;set fp to top of stack to make new stack frame
          CMP r1, #0                    ;check if n=0
Zero      MOVEQ r2, #1                  ;if n=0, set return value (in r2) to 1
          BEQ Return                    ;go to code to return from function
          TST r1, #1                    ;if n!=0, check if n is even
          BEQ Even                      ;if n is even, go to code for even n
Odd       SUB r1, #1                    ;if n is odd, decrement n
          STMIA sp!, {r0-r2}            ;store x and new n in stack as arguments for next function call
          BL Power                      ;call Power function
          LDR r2, [fp, #LDRValLoc]      ;after returned from call, load the returned value into r2
          MUL r2, r0, r2                ;multiply returned value by x
          B Return                      ;go to code to return from function
Even      LSR r1, #1                    ;if n is even, divide n by 2
          STMIA sp!, {r0-r2}            ;store x and new n in stack as arguments for next function call
          BL Power                      ;call Power function
          LDR r0, [fp, #LDRValLoc]      ;after returned from call, load the returned value into r0
          MUL r2, r0, r0                ;square returned value and store in r2
Return    STR r2, [fp, #STRValLoc]      ;store the return value in appropriate location
          MOV sp, fp                    ;collapse stack frame
          LDMDB sp!, {r0-r2, fp, pc}    ;restore r0 to r2, fp and pc to values before function call

result    SPACE 4                       ;local variable result of main function
stack     DCD 0x00                      ;bottom of stack
_X        EQU 6                         ;value of x
_N        EQU 12                        ;value of n
LDRValLoc EQU 8                         ;offset to fp when loading returned value
STRValLoc EQU -24                       ;offset to fp when storing return value
          END
```

Structure of the Stack Frame



Main method variables	Original x	i
	Original n	i+4
	Return value	i+8
Main method register values and return address	R0	i+12
	R1	i+16
	R2	i+20
	FP	i+24
	LR	i+28
1st *Power* function stack frame	New x	i+32 (FP)
	New n	i+36 (FP+4)
	Return value	i+40 (FP+8)
1st *Power* function register values and return address	R0	i+44 (FP+12)
	R1	i+48 (FP+16)
	R2	i+52 (FP+20)
	FP	i+56 (FP+24)
	LR	i+60 (FP+28)
2nd *Power* function stack frame	New x	i+64
	New n	i+68
	Return value	i+72

Stack growth: Ascending
Class: Empty
Stack suffix: EA
Load suffix: DB (decrement before)
Store suffix: IA (increment after)