```
; This program is for calculating X to the power n
; MZHOU272  MingCong, Zhou   250945414
;-------------------------------------------------------------------------------
                AREA  power, CODE, READONLY
x               EQU 2
n               EQU 4
                ENTRY
Main            ADR sp,stack     ;define the stack

                MOV r0, #x                      ;prepare the parameter x
                MOV r1, #n       ;prepare the parameter n

                STR r1,[sp,#-4]!  ;push the parameter on the stack
                SUB sp,sp,#4     ;reserve a place in the stack for the return value

                BL Pow          ;call the Fact subroutine

                LDR r1,[sp],#4   ;load the result in r0 and pop it from the stack
                ADD sp,sp,#4     ;also remove the parameter from the stack
                ADR r2,result    ;get the address of the result variable

                STR r1 ,[r2]      ;store the final result in the result variable
Loop    B Loop       ;infinite loop
;-------------------------------------------------------------------------------
                AREA  power, CODE, READONLY
Pow             STMFD sp!,{r0,r1,r2,fp,lr} ;push general registers, as well as fp and lr
                MOV fp,sp        ;set the fp for this call
                                                ;r1 already have the parameter LDR r1,[fp,#0x18]
                CMP r1,#0        ; this is the base case of the recursion if (n == 0)
                MOVEQ r1,#1       ;{ prepare the value to be returned
                STREQ r1,[fp,#0x14] ;  store the returned value in the stack
                BEQ ret          ;  branch to the return section ;}

                ANDS r2, r1, #1  ;check odd or even, odd would be z clear NE even would be z set EQ

                BEQ even                        ; if it is even branch to even
                SUBNE r1, r1, #1     ; it is odd prapare the parameter n-1 power(x, n - 1);
                STRNE r1,[sp,#-4]! ; and push the parameter to the memory
                SUBNE sp,sp,#4     ; reserve a place in the stack for the return value

                BLNE Pow              ; call the Pow subroutine

even            LDR r1,[fp,#0x18]    ; load r1 again
                ANDS r2,r1,#1                   ; check n again if it is even or not

                LSREQ r1, r1, #1                ; if it is even n/2 divide n by 2
                STREQ r1, [sp, #4]!             ; push the parameter to the stack
                SUBEQ sp, sp, #4                ; reserve a place in the stack for the return value

                BLEQ Pow                        ; call the Pow subroutine power(x, n >> 1);

                LDR r1,[fp,#0x18]               ; because of the branch of subroutine the flag might change
                ANDS r2,r2,#1                           ; therefore check odd or even again

                LDR r1, [sp], #4                ; load the result in r0 and pop it from the stack
                ADD sp, sp, #4                          ; remove also the parameter from the stack
                MULNE r2,r0, r1                         ; if it is odd simpley x * power(x, n - 1);
                MULEQ r2, r1, r1                ; if it is even y * y;
                STR r2, [fp, #0x14]             ; store the returned value in the stack

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 0xA8       ;declare the space for stack
stack           DCD   0x00       ;initial stack position (FD model)
;-------------------------------------------------------------------------------
                END
```

Stack Frame for 2 to the 3

		push para and sub sp	1St Pow STM	odd 3become 2 Second Pow	third call pow n/2
Address =	0x 0108				02
Address =	0x 010C				01
Address =	0x 0110				00
Address =	0x 0114			02	114
Address =	0x 0118			02	74
Address =	0x 011C			00	02
Address =	0x 0120			28	01
Address =	0x 0124			74	00
Address =	0x 0128		02	02	28
Address =	0x 012C		04	02	74
Address =	0x 0130		00	00	02
Address =	0x 0134		00	00	02
Address =	0x 0138		18	18	00
Address =	0x 013C				00
Address =	0x 0140	04	04	04	18
SP value		013C	0128	0114	04

third call pow n/2

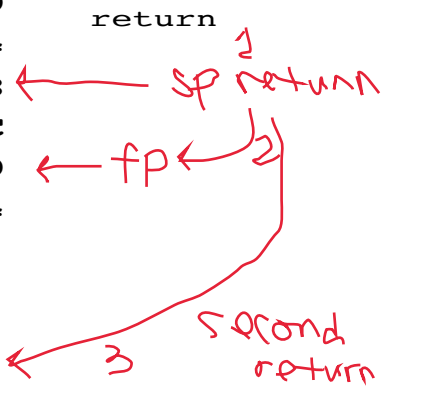
02
01
00
114
74
02
01
00
28
74
02
02
00
00
18
04
100

forth call pow
base case reach r1 = 0

02	0x 00E8
00	0x 00EC
00	0x 00F0
01	0x 00F4
0100	0x 00F8
5C	0x 00FC
	0x 0100
00	0x 0104
02	0x 0108
01	0x 010C
00	0x 0110
114	0x 0114
74	0x 0118
02	0x 011C
01	0x 0120
00	0x 0124
00	0x 0128
28	0x 012C
74	0x 0130
02	0x 0134
02	0x 0138
00	0x 013C
00	0x 0140
18	
10	
04	
E4	

store the value

0x098	10



sp 3rd return

sp store 10

finish recursion