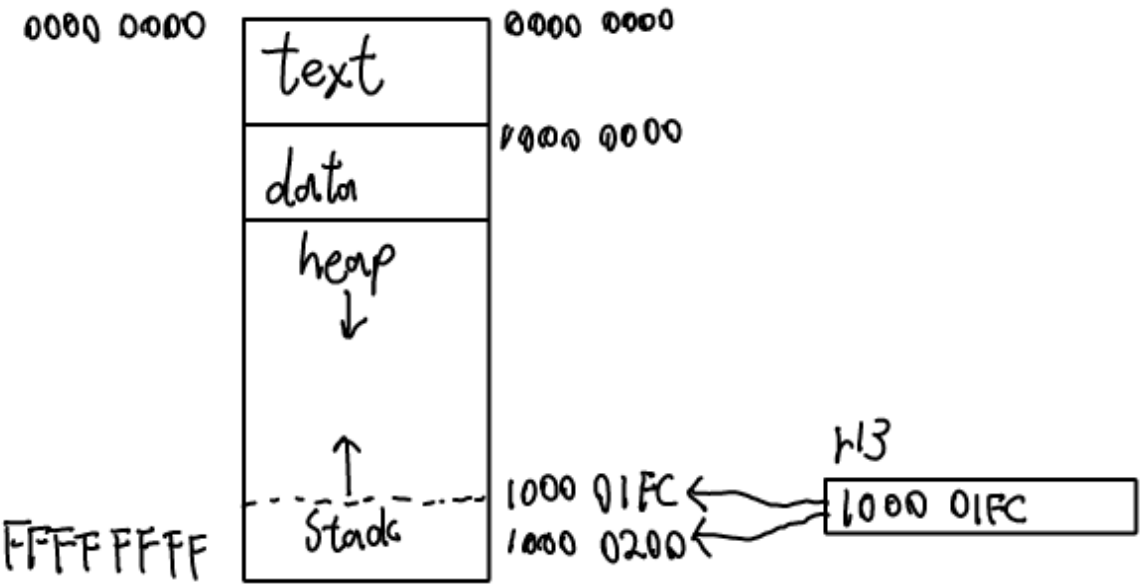
Subroutines

1) The Stack

- LIFO
- each thread (process) has a call stack growing from high to low memory addresses
- used for local variables and parameter passing.
- typical memory map (32 bit addressing)



- r13 (alias sp) is the stack pointer.
- push [r0] onto stack:

```
STR r0, [sp, #-4]!   (pre-indexed)
```

- pop from stack onto r[0]:

```
LDR r0, [sp], #4   (post-indexed)
```

2) Calling

- branch to subroutine instructions store return address in the link register r14 (alias lr)
- branch and link: BL{cond} label (invocation)

```
// pc ← [pc] + offset,   // offset ∈ [-16MB, +16MB]
r14 ← <return address>
        ↑ pc of next instruction
```

- branch, link, and exchange: BLX{cond} Rn (invocation)

  e.g. BLX r1

  //pc ← [r1], r14 ← [pc]

  - greater range than BL [-2GB, +2GB]

- branch exchange: BX{cond} Rn (return)

  e.g. BX lr

  //pc ← [r14]

3) Load/Store Multiple

- push/pop multiple register values to/from stack
- syntax: <op>{mode}{cond} Rn{!}, reglist

  op = LDM Load Multiple

  STM Store Multiple

  mode = IA increment after

  IB increment before

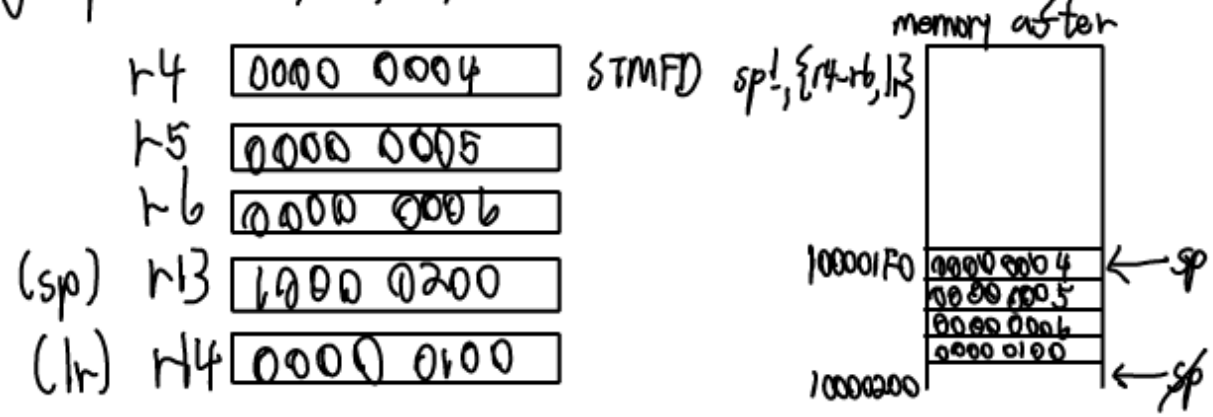
  ! = write back (update sp reg.)

  reglist = comma separated list of regs. or reg. ranges

- STMFD is synonym for STMDB (push)
- LDMFD is synonym for LDMIA (pop)

  FD = Full descending stack

e.g. push r4, r5, r6, and r14 onto stack



e.g. pop from stack into r4, r5, r6, pc
(restore r4-r6, and subroutine return)

LDMFD sp!, {r4-r6, pc}

4) AAPCS. ARM Architecture Procedure Call Standard

registers	synonyms	callee preserved	function
r0 - r3	a1 - a4	no	result/argument/scratch
r4 - r11	v1 - v8	yes	local var
r12	ip	no	intra-procedure/scratch
r13	sp	yes	stack pointer
r14	lr	no	link register
r15	pc	no	program counter

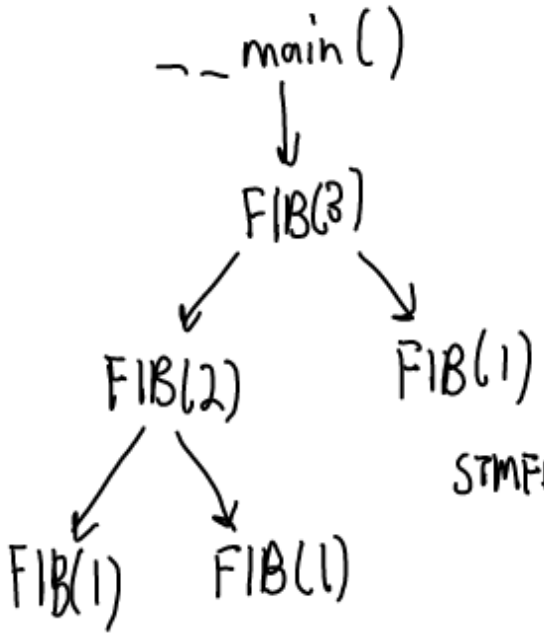
- guidelines
  - preserve and restore v1-v8 (r4-r11) if you modify them
  - anything pushed on the stack must be popped
  - return values are in r0 (and r1-r3 as needed)
  - pass parameters via registers first (faster)
  - pass additional parameters via stack

- demo4.s (Array example as a subroutine with args on the stack)

- demo5.s (fibonacci sequence)

$$f(n) = \begin{cases} f(n-1)+f(n-2), & n \geq 2 \\ 1, & n = 1 \\ 0, & n = 0 \end{cases}$$

- recursive
- n is passed in r0




```
ldmfd sp!, {r4, r5, lr}
bx    lr
```

```
ldmfd sp! {r4, r5, pc}
```

		r0	r4	r5	r13	r14
__main()						
mov r0, #3		0	0	0	10000200	0
bl fib		3				00000108
fib(3)						
cmp r0, #1						
bxle lr						
STMFD	sp!, {r4, r5, lr}				100001F4	
sub r0, r0, #1		2				
sub r4, r4, #1			1			
bl fib						00000126
fib(2)						
⋮						
fib(1)						
cmp r0, #1						
bxle lr						
fib(2)						
mov r5, r0				1		
mov r0, r4		0				
bl fib						0000012E
fib(0)						
⋮						