Questions

find_zeros_index

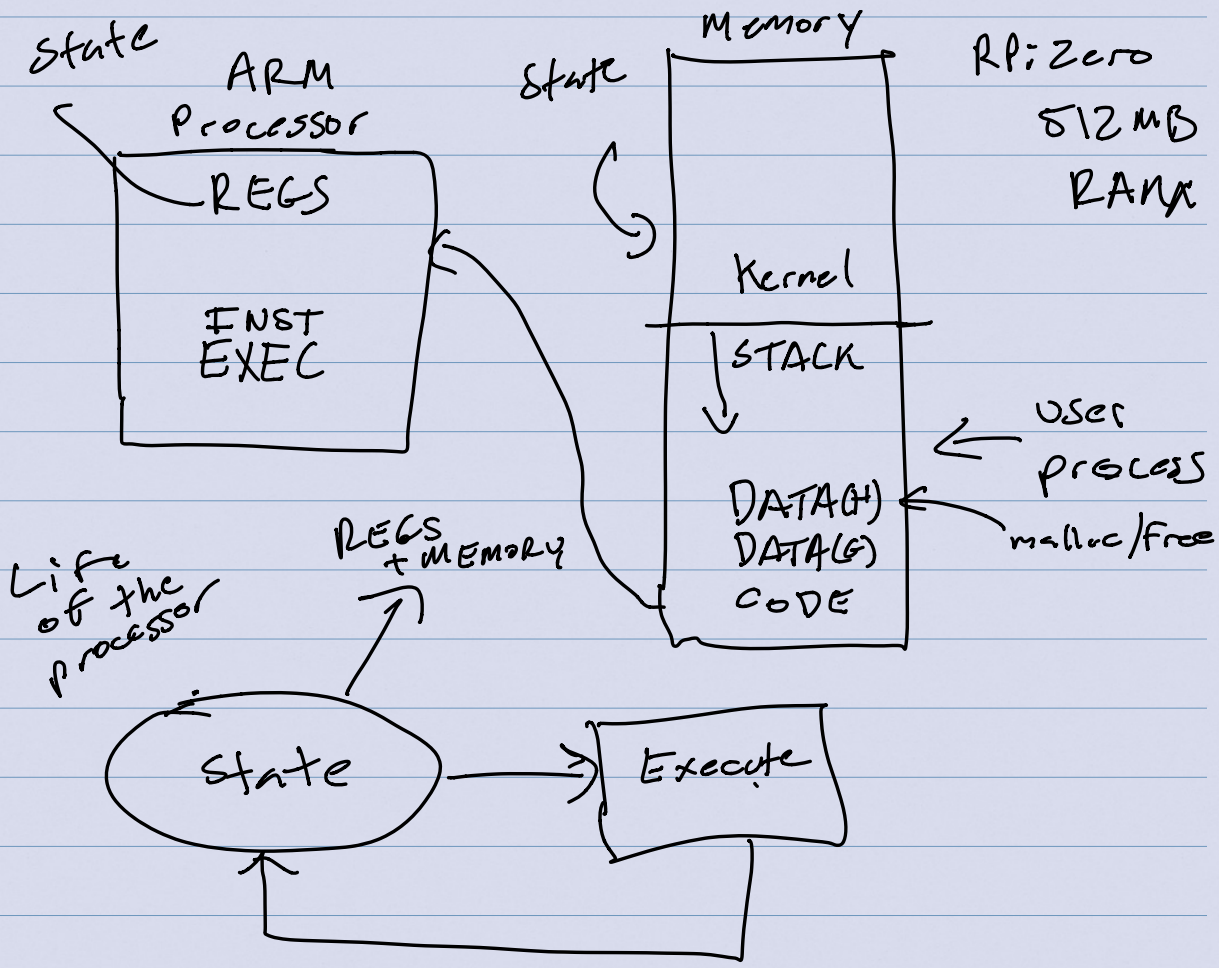
ARM Produre Calling Conventions

strings and strlen

hello world

Recursion - factorial

---



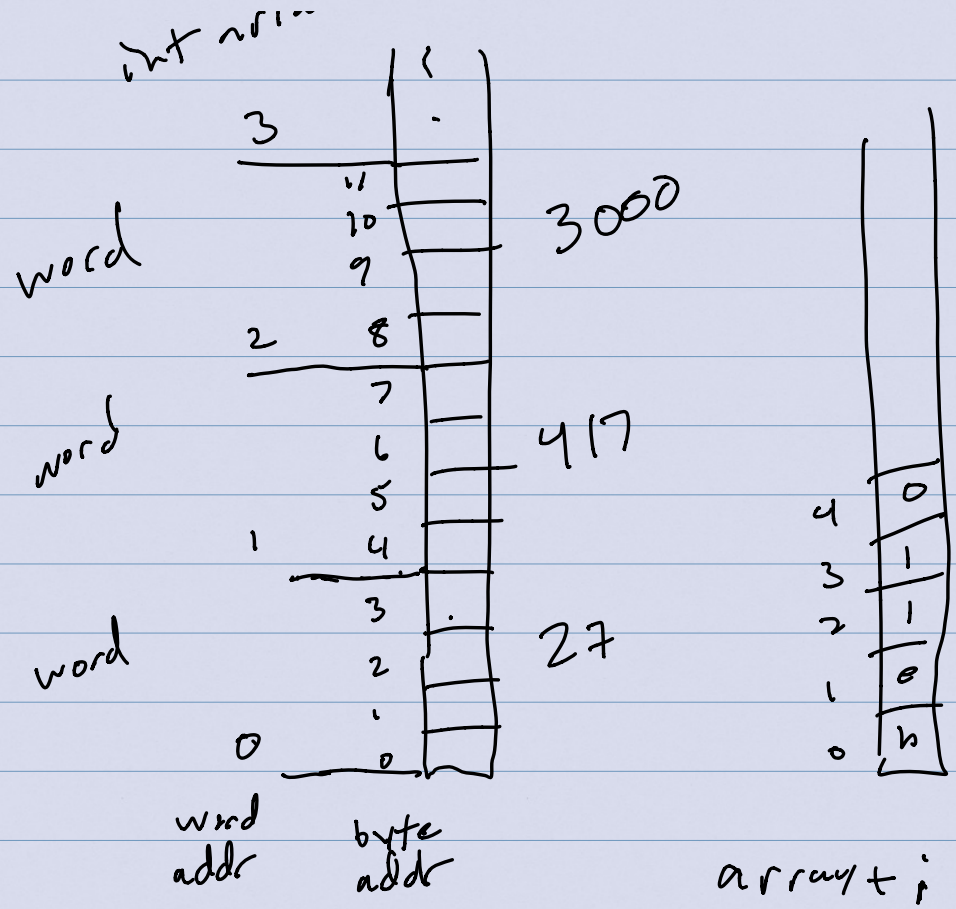
# ARM Instruction Types

Data Processing : add, mul, sub, mov, cmp
Memory : ldr, str
Control : b, beq, bl, bx

```
add r0, r1, r2
ldr r0, [r1]      C: r0 = *r1      -> 32 bit word
str r0, [r1]      *r1 = r0

ldr r0, [r1, #4]      addr = r1 + 4
ldr r0, [r1, r2]      addr = r1 + r2
ldr r0, [r1, r2 LSL #2]

r2 << 2 = r2 * 4
addr = r1 + (r2 * 4)
```

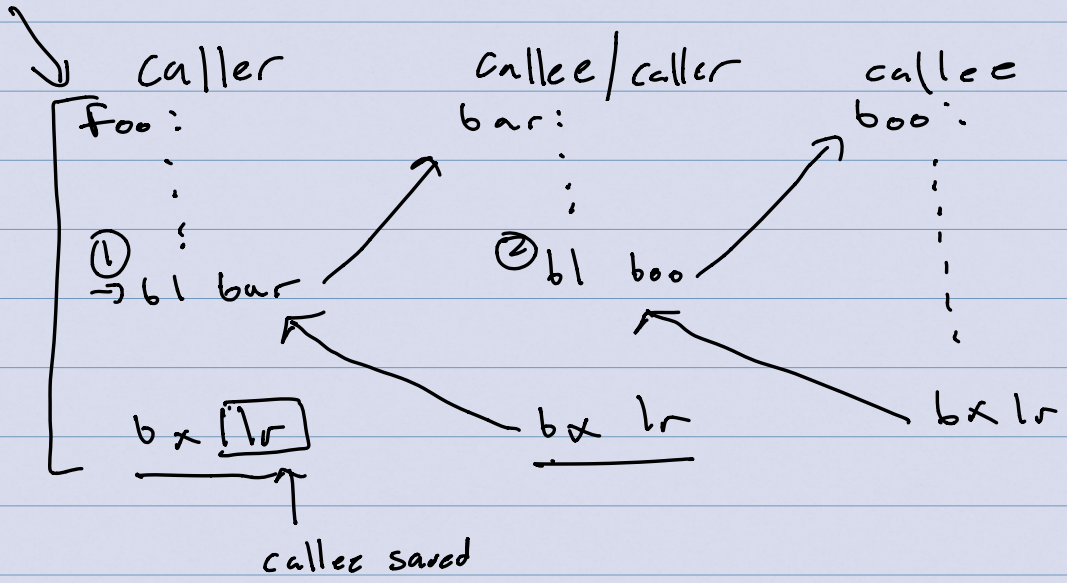
```
x = array[i]

int a[5];
char c[5];
```

array is base point
array + (i * 4) int
char




---

# ARM Procedure Calling Conventions



bl  branch and link

bl foo  PC = foo

add  LR = Addr after bl

↑ link resister return address

bx lr  branch and exchange

Caller-saved (non preserved)

Callee-saved (preserved)

CPSR holds the cmp bits

Caller-saved

[r0, r1, r2, r3
r12, CPSR]

Callee-saved

r4-r11, r13, r14
SP  LR

Must be a multiple of 8

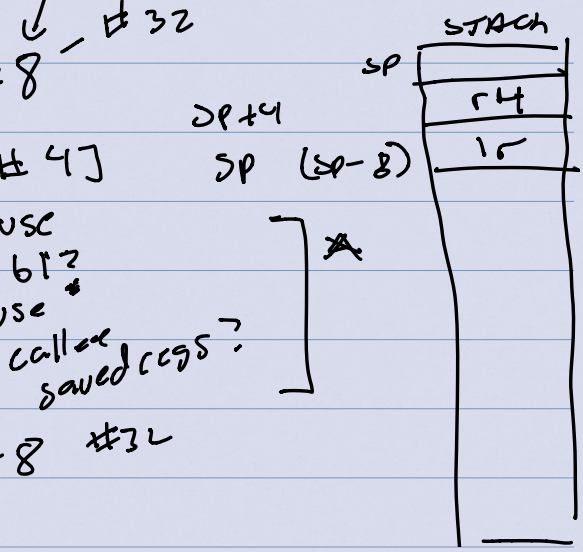
foo:

entry
```
sub sp, sp, #8      ; #32
str lr, [sp]
str r4, [sp, #4]
```

```
bl
```

exit
```
ldr r4, [sp, #4]
ldr lr, [sp]
add sp, sp, #8      #32
bx lr
```

use blr ∈ use callee saved regs?



full function call form

foo:

entry
```
sub sp, sp, #8
str lr, [sp]
str r4, [sp, #4]
```
allocate local vars on stack

we may need stack space for caller saved regs (r3)

```
bl bar
```
restore caller saved

exit
```
ldr r4, [sp, #4]
ldr lr, [sp]
add sp, sp, #8
```
bx lr

---

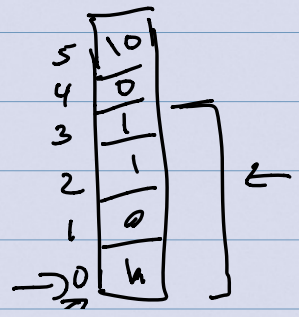
strings

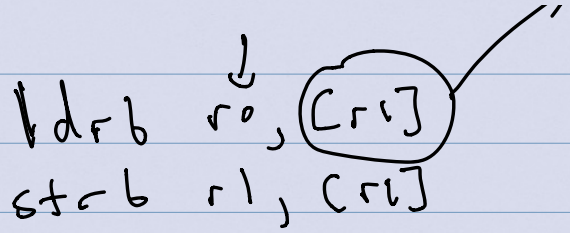
r0
r1
:

} all regs are 32 bits = 4 bytes

ldr = word

lower 8 bits




ldrb r0, [r1]
strb r1, [r1]