Questions

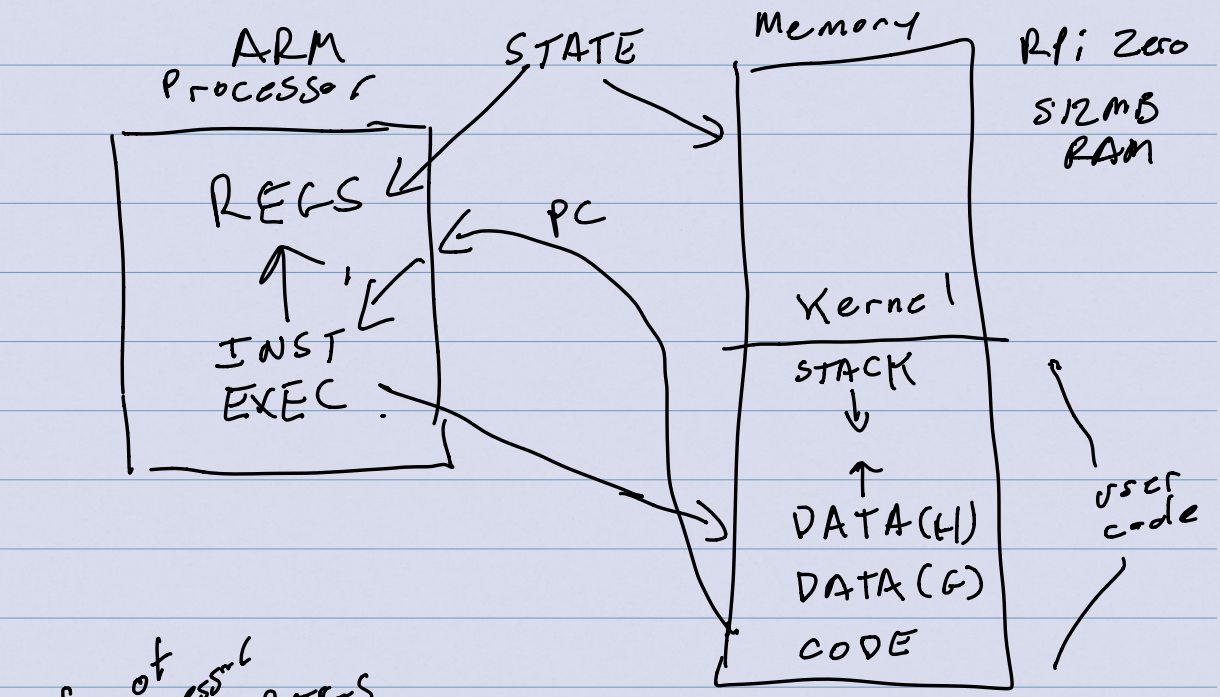
Find_zeros_index

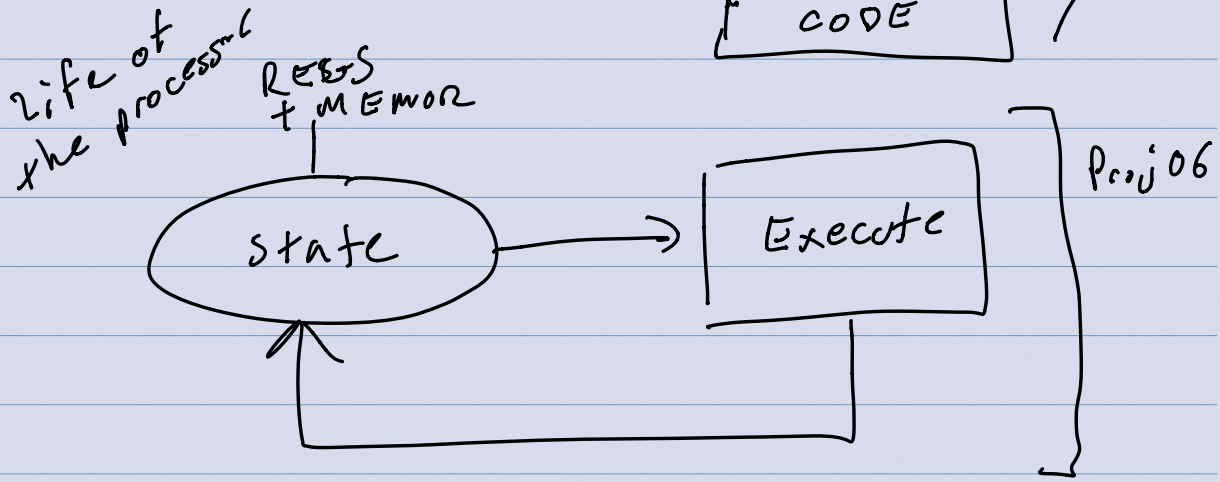
ARM Procedure Calling Conventions

strings and strlen

hello_s.s hello world

Recursion - factorial





ARM Instruction Types

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

c

```
ldr r0, [r1]    r0 = *r1
str r0, [r1]    *r1 = r0
```

```
ldr r0, [r1]
```

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

logical shift left

r2 << 2 = r2 * 4

addr = r1 + (r2 * 4)

```
int array[5]
x = array[i]
```

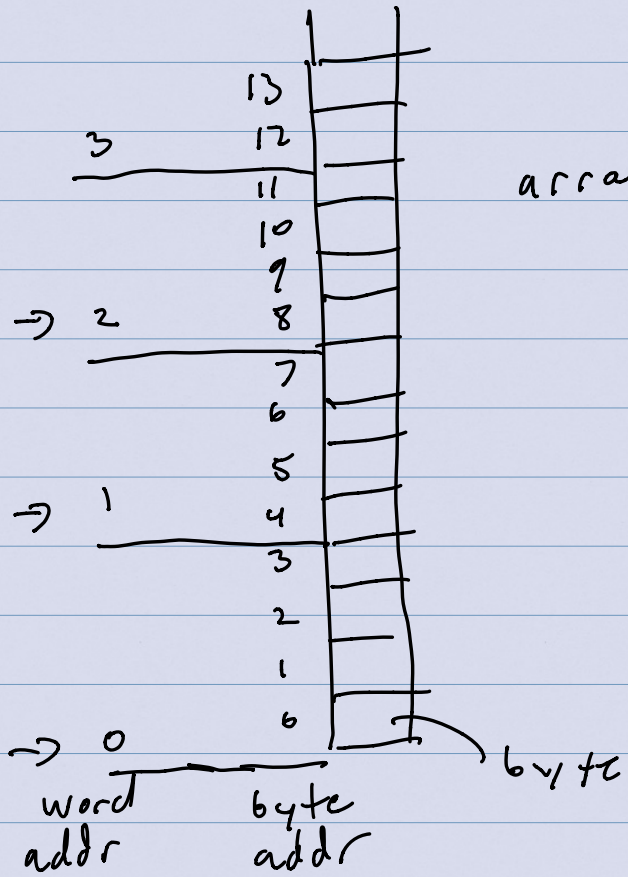
array is base pointer

array + (i * 4)

size of the data type of the array

```
char ca[5]      ca[i] = ca + (i * 1)
```

Memory is byte addressable



array + (i * 4)

v = 0000 0110 ← 6

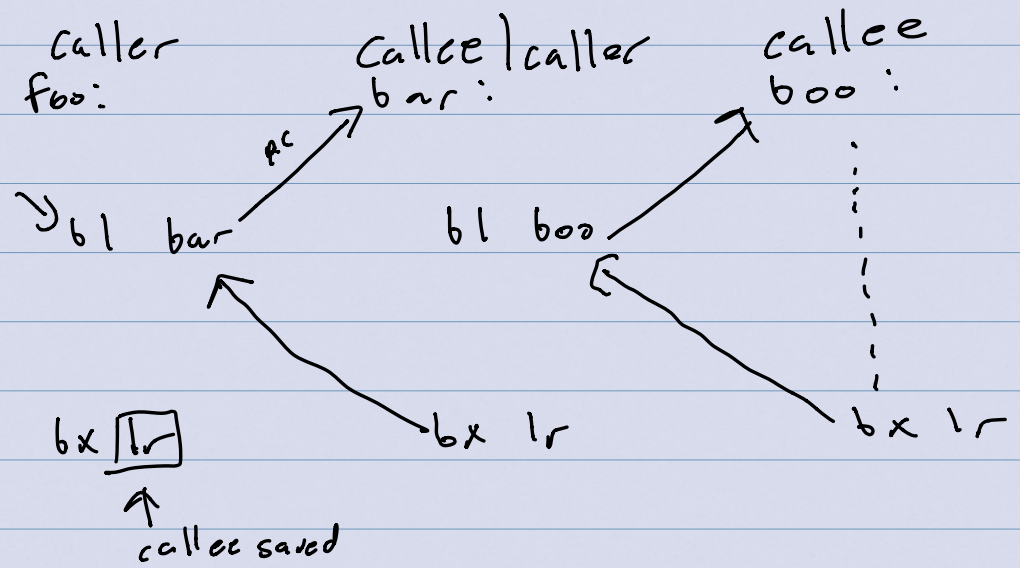
v << 2 × 4

0000 1000 ← 24

16 8

r3, LSL #2

ARM Procedure Calling Conventions



bl branch and link

bl foo PC = foo

⇒ add LR = addr of the inst after bl foo

↑ link register
return address

bx lr branch and exchange

caller-saved (non preserved)

callee-saved (preserved)

caller-saved

r0, r1, r2, r3

r12, CPSR

callee-saved

r4-r11, r13, r14

SP LR

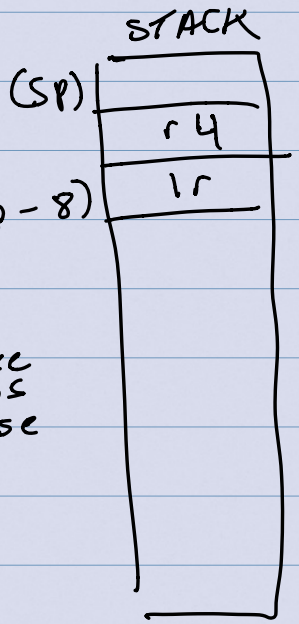
PC special

foo:

sp must be a multiple of 8

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

sp (sp-8)



body
```
bl bar
```
r4

↖ use bl? save lr

↖ use callee save regs save these

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

---

full function call form

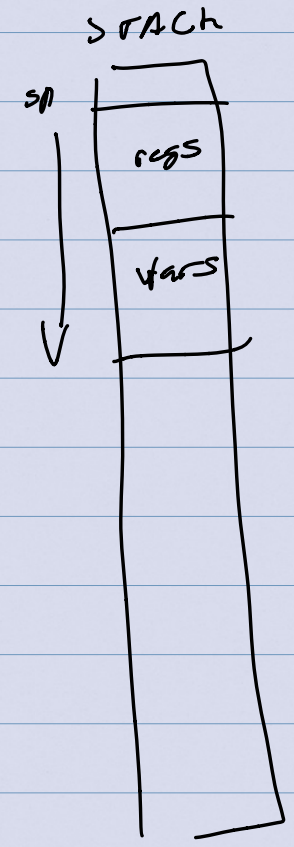
foo:

entry
```
sub sp,sp,#8
str lr,[sp]
str r4,[sp,#4]
```
[allocating variable on the stack]

[we may need stackspace for caller saved regs]

save caller saved

body
```
bl bar
```
restor caller saved



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

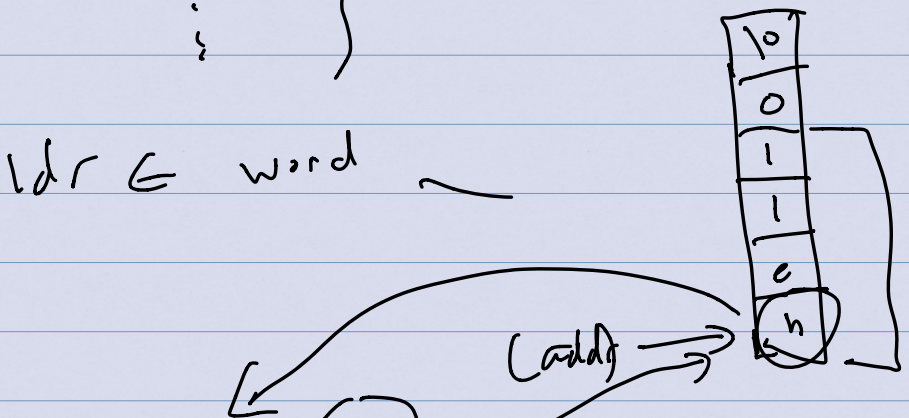
---

strings

r0
r1
:

} 32 bits
4 bytes

ldr ← word




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