

ARM Programming Structure

Young W. Lim

June 8, 2016

Copyright (c) 2011-2016 Young W. Lim. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Making Decision

```
cmp reg1, reg2  
beq l1
```

```
cmp reg1, reg2  
bne l1
```

```
if (i==j) f = g+h; else f = g-h;
```

```
cmp r3, r4  
bne l2  
add r0, r1, r2  
b exit
```

```
else:  
sub r0, r1, r2  
exit:
```

Loop

```
while (save[i] == k) i += 1;
```

```
loop:
```

```
    add r12, r6, r3, lsl #2
```

```
    ldr r0, [r12, #0]
```

```
    cmp r0, r5
```

```
    bne exit;
```

```
    add r3, r3, #1
```

```
    b  loop
```

```
exit:
```

Simple C Procedure (1)

```
int func (int g, int h, int i, int j) {  
    int f;  
    f = (g+h) - (i+j);  
    return f;  
}
```

```
sub sp, sp, #12  
str r6, [sp, #12]  
str r5, [sp, #8]  
str r4, [sp, #4]
```

; r4, r5, r6 are used

```
ldr r4, [sp, #0]  
ldr r5, [sp, #4]  
ldr r6, [sp, #8]  
add sp, sp, #12
```

Simple C Procedure (2)

```
sub sp, sp, #12
str r6, [sp, #12]
str r5, [sp, #8]
str r4, [sp, #4]
```

```
add r5, r0, r1
add r6, r2, r3
sub r4, r5, r6
```

```
mov r0, r4
```

```
ldr r4, [sp, #0]
ldr r5, [sp, #4]
ldr r6, [sp, #8]
add sp, sp, #12
```

```
mov pc, lr
```

Reference

- [1] D. Harris, "Digital Design and Computer Architecture", 2nd ed.
- [2] D.A. Patterson & J.H. Hennessy, "Computer Organization and Design (ARM ed)"