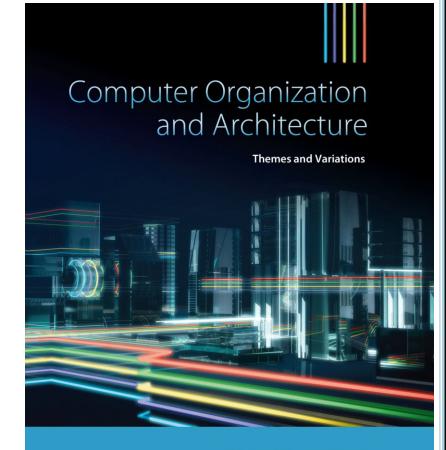
## Part D

### CHAPTER 3

# Architecture and Organization



Alan Clements

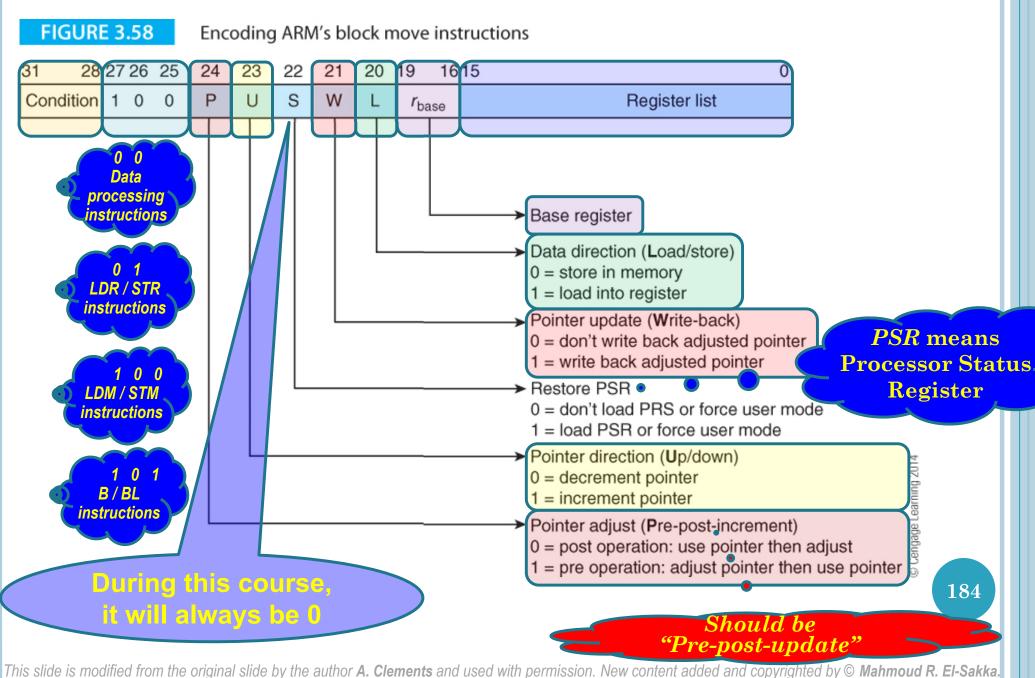
1

These slides are being provided with permission from the copyright for CS2208 use only. The slides must not be reproduced or provided to anyone outside of the class.

All download copies of the slides are for personal use only.
Students must destroy these copies within 30 days after receipt of final course evaluations.



#### **Block Move Instructions Encoding/Decoding**



0 = post operation: use pointer then adjust

1 = pre operation: adjust pointer then use pointer

#### **Block Move Instructions Encoding Example**

```
ARM Instruction: STMFD
                                               r13!, {r0-r4, r10}
Condition = 1110 (always - unconditional)
      P = 1 (DB: adjust pointer then use pointer)
      U = 0 (DB: decrement)
      S = 0 (user mode)
      W = 1 (write-back adjusted pointer)
      L = 0 (store)
      r_{\text{base}} = 1101 (r13)
      Register list (r15, r14, ..., r2, r1, r0) = 0000 0100 0001 1111
      1110 1001 0010 1101 0000 0100 0001 1111
                                                FIGURE 3.58
                                                           Encoding ARM's block move instructions
0xE92D041F
                                                              23 22 21 20 19 16 15
                                                   28 27 26 25
                                               Condition 1 0 0
                                                            Ρ
                                                              U
                                                                 S
                                                                                          Register list
           Stack full descending
                                    Free | n - 12
                                                                                  Base register
                Occupied

    Data direction (Load/store)

                                   Item 3 | n - 8
                                                                                   0 = store in memory
                 memory
                                                                                   1 = load into register
                                   Item 2 \mid n-4
                                                                                  Pointer update (Write-back)
                                                                                   0 = don't write back adjusted pointer
                                   Item 1 | n
                                                                                   1 = write back adjusted pointer
                        SP
Grows up
                                                                                  Restore PSR
                                                                                   0 = don't load PRS or force user mode
                      Stack grows towards low memory
                                                                                   1 = load PSR or force user mode
                                                                                  ➤ Pointer direction (Up/down)
                      Stack pointer points at top of stack
                                                                                   0 = decrement pointer
                                                                                   1 = increment pointer
                                                                                  Pointer adjust (Pre-post-increment)
```

This slide is modified from the original slide by the author A. Clements

Pointer adjust (Pre-post-increment)
 0 = post operation: use pointer then adjust

1 = pre operation: adjust pointer then use pointer

#### **Block Move Instructions Encoding Example**

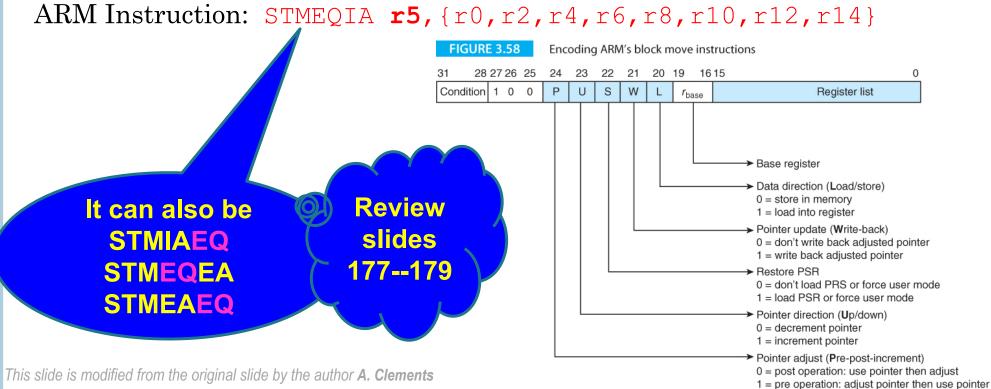
r13!, {r0-r4,r10} ARM Instruction: LDMFD Condition = 1110 (always - unconditional) P = 0 (IA: use pointer then adjust) U = 1 (IA: increment) S = 0 (user mode) W = 1 (write-back adjusted pointer) L = 1 (load)  $r_{\text{base}} = 1101 (r13)$ Register list (r15, r14, ..., r2, r1, r0) = 0000 0100 0001 11111110 1000 1011 1101 0000 0100 0001 1111 FIGURE 3.58 Encoding ARM's block move instructions 0xE8BD041F 23 22 21 20 19 16 15 28 27 26 25 Condition 1 0 0 Ρ U S Register list Stack full descending Free | *n* – 12 Base register Occupied Data direction (Load/store) Item 3 | n - 80 = store in memory memory 1 = load into register Item  $2 \mid n-4$ Pointer update (Write-back) 0 = don't write back adjusted pointer Item 1 | n 1 = write back adjusted pointer SP **Grows up** Restore PSR 0 = don't load PRS or force user mode Stack grows towards low memory 1 = load PSR or force user mode ➤ Pointer direction (Up/down) Stack pointer points at top of stack 0 = decrement pointer 1 = increment pointer

This slide is modified from the original slide by the author A. Clements

#### **Block Move Instructions Decoding Example**

Decode the ARM machine language 0x08855555

```
0000 1000 1000 0101 0101 0101 0101 0101
Condition = 0000 (EQ)
   P = 0 (IA: use pointer then adjust)
   U = 1 (IA: increment)
   S = 0 (user mode)
   W = 0 (do not write-back adjusted pointer)
   L = 0 (store)
   r_{\text{base}} = 0101 (r5)
   Register list (r15, r14, ..., r2, r1, r0) = 0101 0101 0101 0101
```



0 = post operation: use pointer then adjust

1 = pre operation: adjust pointer then use pointer

#### **Block Move Instructions Decoding Example**

Decode the ARM machine language 0x99922222

This slide is modified from the original slide by the author A. Clements

Encoding ARM's block move instructions 24 23 22 21 20 19 16 15 Condition 1 0 0 US Register list Base register Data direction (Load/store) 0 = store in memory It can also be Review 1 = load into register Pointer update (Write-back) slides LDMIBLS 0 = don't write back adjusted pointer 1 = write back adjusted pointer **177--179** LDMLSED Restore PSR 0 = don't load PRS or force user mode LDMEDLS 1 = load PSR or force user mode Pointer direction (Up/down) 0 = decrement pointer 1 = increment pointer Pointer adjust (Pre-post-increment)