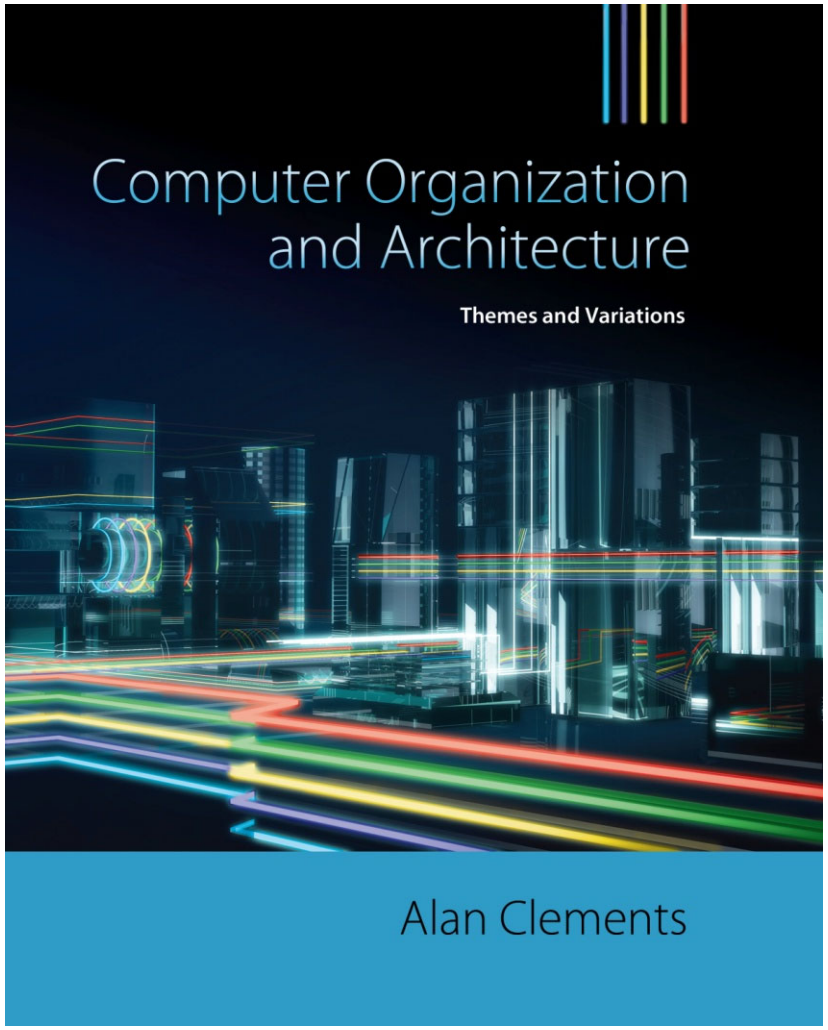
Part D

Chapter 3

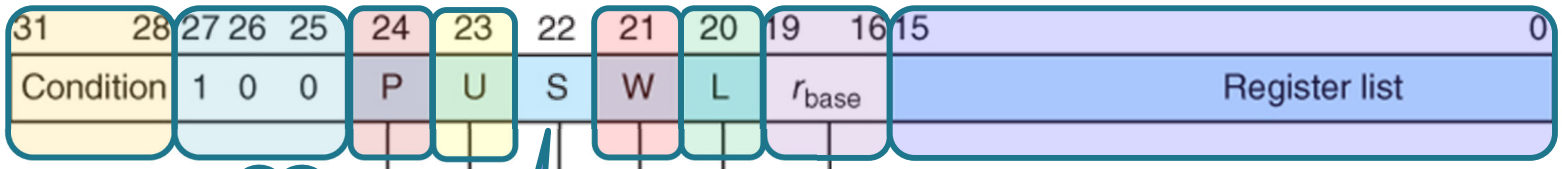
Architecture and Organization



CENGAGE Learning

Block Move Instructions Encoding/Decoding

FIGURE 3.58 Encoding ARM's block move instructions



0 0 Data processing instructions

0 1 LDR / STR instructions

1 0 0 LDM / STM instructions

1 0 1 B / BL instructions

Base register

Data direction (**L**oad/store)
0 = store in memory
1 = load into register

Pointer update (**W**rite-back)
0 = don't write back adjusted pointer
1 = write back adjusted pointer

Restore PSR
0 = don't load PRS or force user mode
1 = load PSR or force user mode

Pointer direction (**U**p/down)
0 = decrement pointer
1 = increment pointer

Pointer adjust (**P**re-post-increment)
0 = post operation: use pointer then adjust
1 = pre operation: adjust pointer then use pointer

PSR means Processor Status Register

During this course, it will always be 0

Should be "Pre-post-update"

STM r1! {r2-r10}：将r2-r10的数据存到r1所处的内存地址，并更新SP

LDM r1! {r2-r10}：将r1开头的数据从内存读取到r2-r10这9个register中.

ST：store to memory

LD：load to register.

LDR R0, [SP]

LDM SP!, {R0, R1, R2}

方向不同！

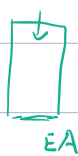
IB：increasement before

FA：full-ascending

IB
IA
DB
DA
=> register上操作 => 数据读取/储存.

FD
FA
ED
EA
. Register -> Memory => stack上操作.

STMDB <=> LDMIA.



IB => EA
IA => FA
DB => ED
DA => FD

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)

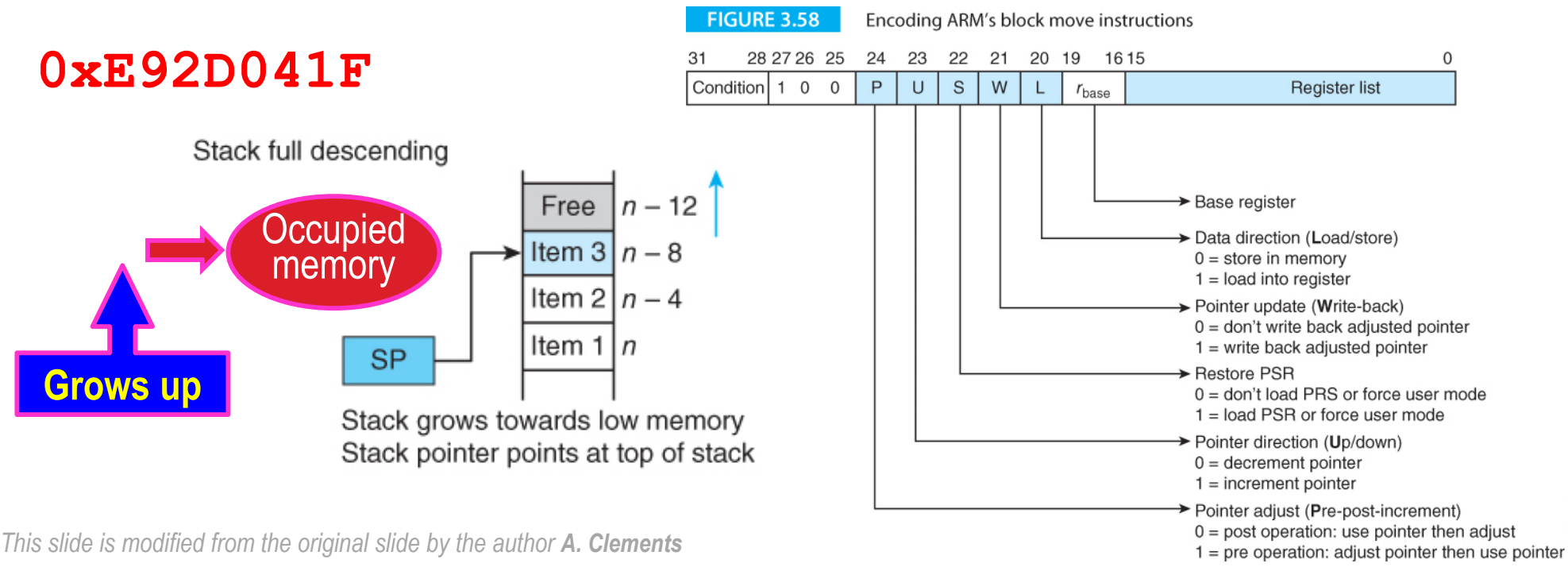
L = 0 (store)

r_{base} = 1101 (r13)

Register list (r15, r14, …., r2, r1, r0) = 0000 0100 0001 1111

1110 1001 0010 1101 0000 0100 0001 1111

0xE92D041F



Block Move Instructions Encoding Example

ARM Instruction: LDMFD r13!,{r0-r4,r10}

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)

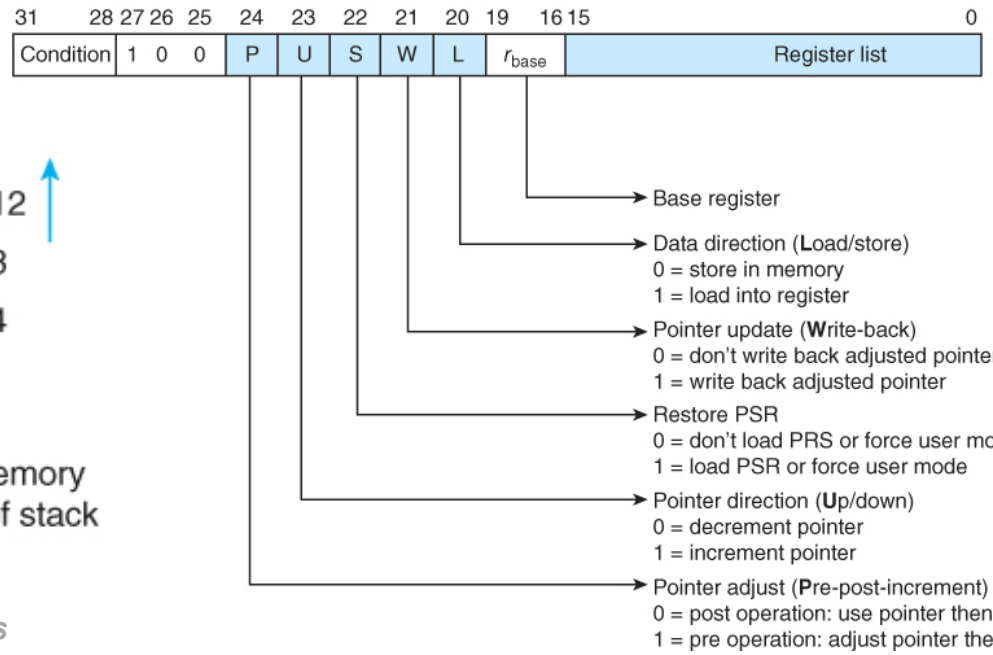
r_{base} = 1101 (r13)

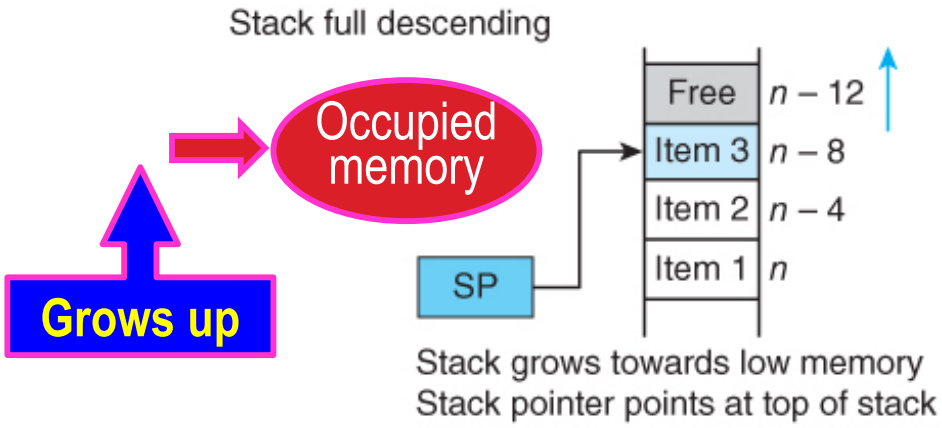
Register list (r15, r14, …., r2, r1, r0) = 0000 0100 0001 1111

1110 1000 1011 1101 0000 0100 0001 1111

0xE8BD041F

FIGURE 3.58 Encoding ARM's block move instructions





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_{base} = 0101 (r5)

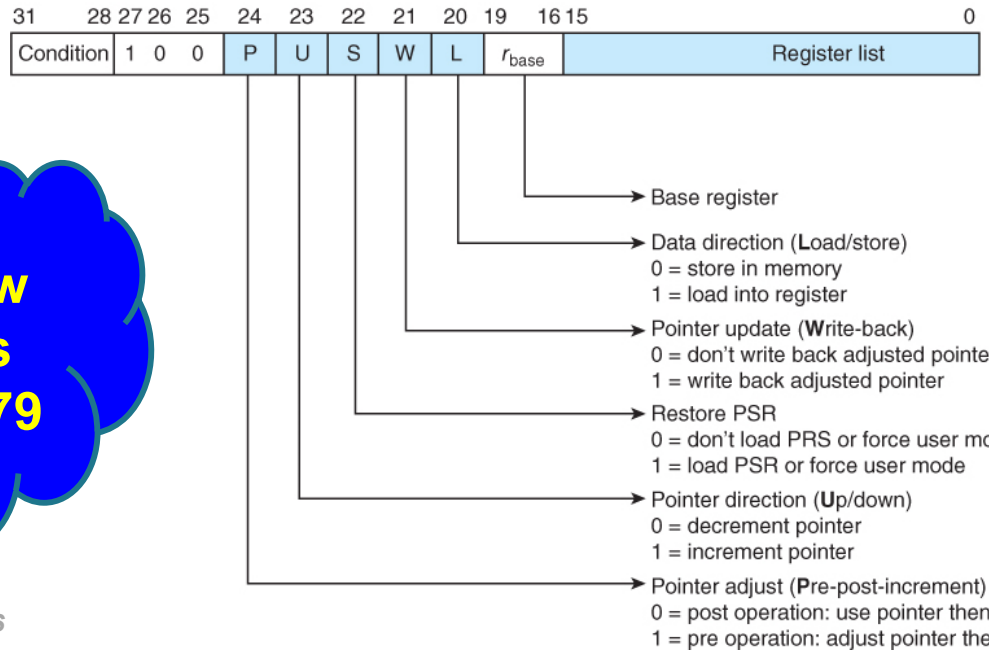
Register list (r15, r14, …., r2, r1, r0) = 0101 0101 0101 0101

ARM Instruction: STMEQIA **r5**,{r0,r2,r4,r6,r8,r10,r12,r14}

It can also be
STMIAEQ
STMEQEA
STMEAEQ

Review slides 177--179

FIGURE 3.58 Encoding ARM's block move instructions



Block Move Instructions Decoding Example

Decode the ARM machine language **0x99922222**

1001 1001 1001 0010 0010 0010 0010 0010

Condition = 1001 (LS)

P = 1 (IB: adjust pointer then use pointer)

U = 1 (IB: increment)

S = 0 (user mode)

W = 0 (do not write-back adjusted pointer)

L = 1 (load)

r_{base} = 0010 (r2)

Register list (r15, r14, …., r2, r1, r0) = 0010 0010 0010 0010

ARM Instruction: LDMLSIB **r2**,{r1,r5,r9,r13}

It can also be
LDMIBLS
LDMLSED
LDMEDLS

Review slides 177--179

FIGURE 3.58 Encoding ARM's block move instructions

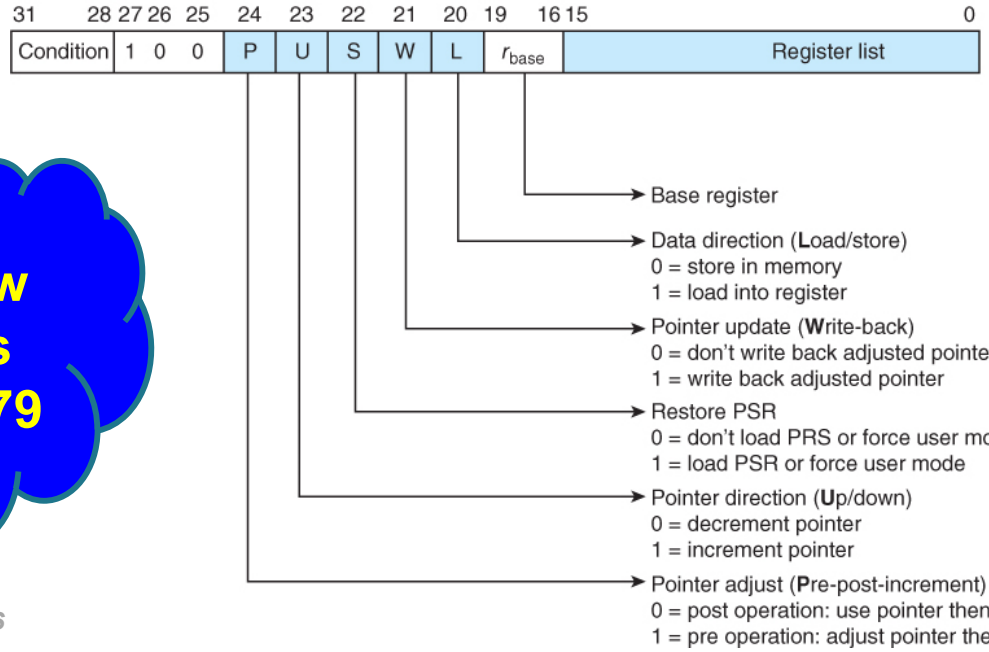
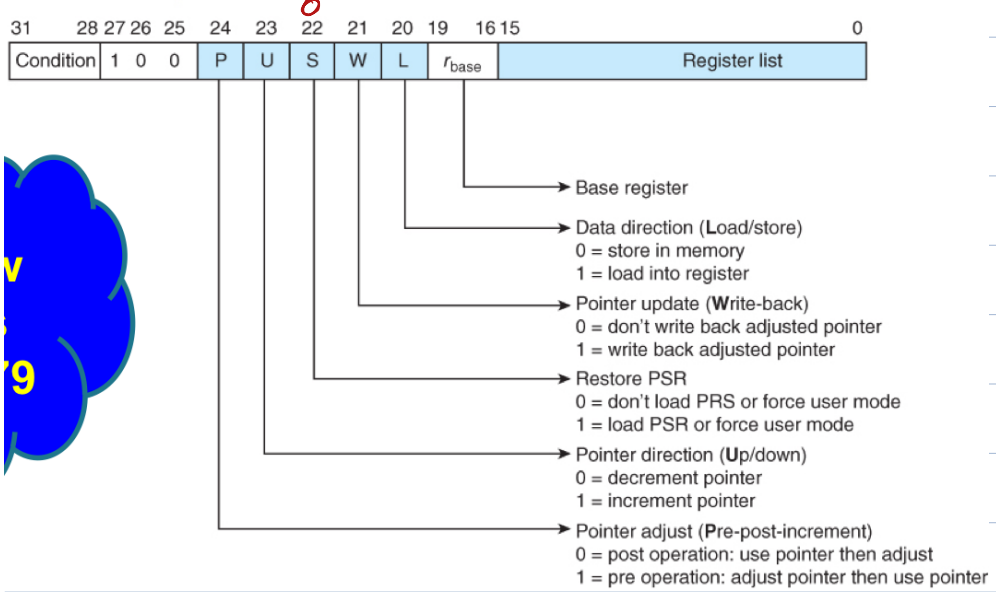


FIGURE 3.58 Encoding ARM's block move instructions



SP after operation

LDMEA :10.

10 => 1010.

LDMFD r0! {r1-r5, r10}.

1110 100 0 1 0 1 1 0000 0000 0100 0000

F/B => P=1 FD:01

0xE8B0043E

LDMFD r0! {r1-r5, r10} 1110 100 0 1 0 1 1 0000 00000 10000 11110

0xE9300436

EA 1 0 0 1 1

0xE9B0043E

IB 1 1

0xE830043E

DA 0 0

0xE9200436

STM FD 1 0 0 1 0

0xE8A0042E

EA 0 1 0 1 0

0xE9A0043E

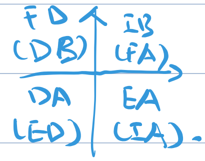
IB 1 1 0

0xE8200 43E

DA. 00 0

FD

EA



(DB).
LDMFD => 01.

STM		LDM	
FD/DB	10	FD/IA	01
FA/IB	11	FA/DA	00
ED/DA	00	ED/IB	11
EA/IA	01	EA/DB	10