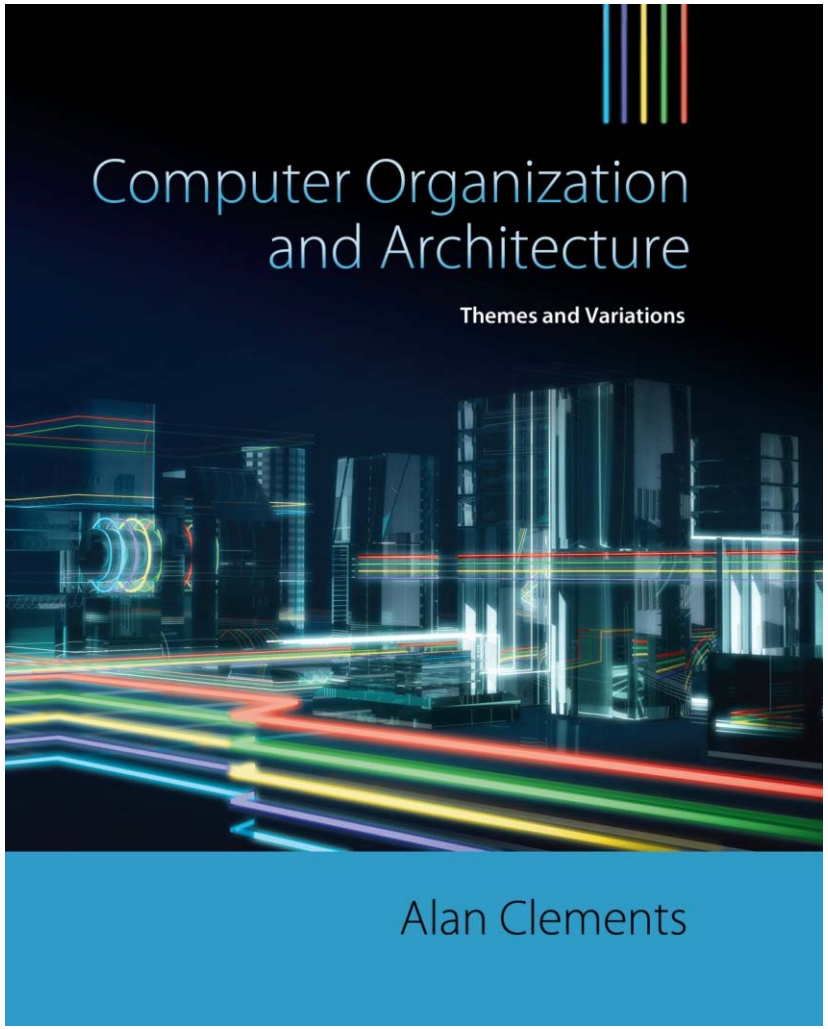
Part 0xA

Chapter 3

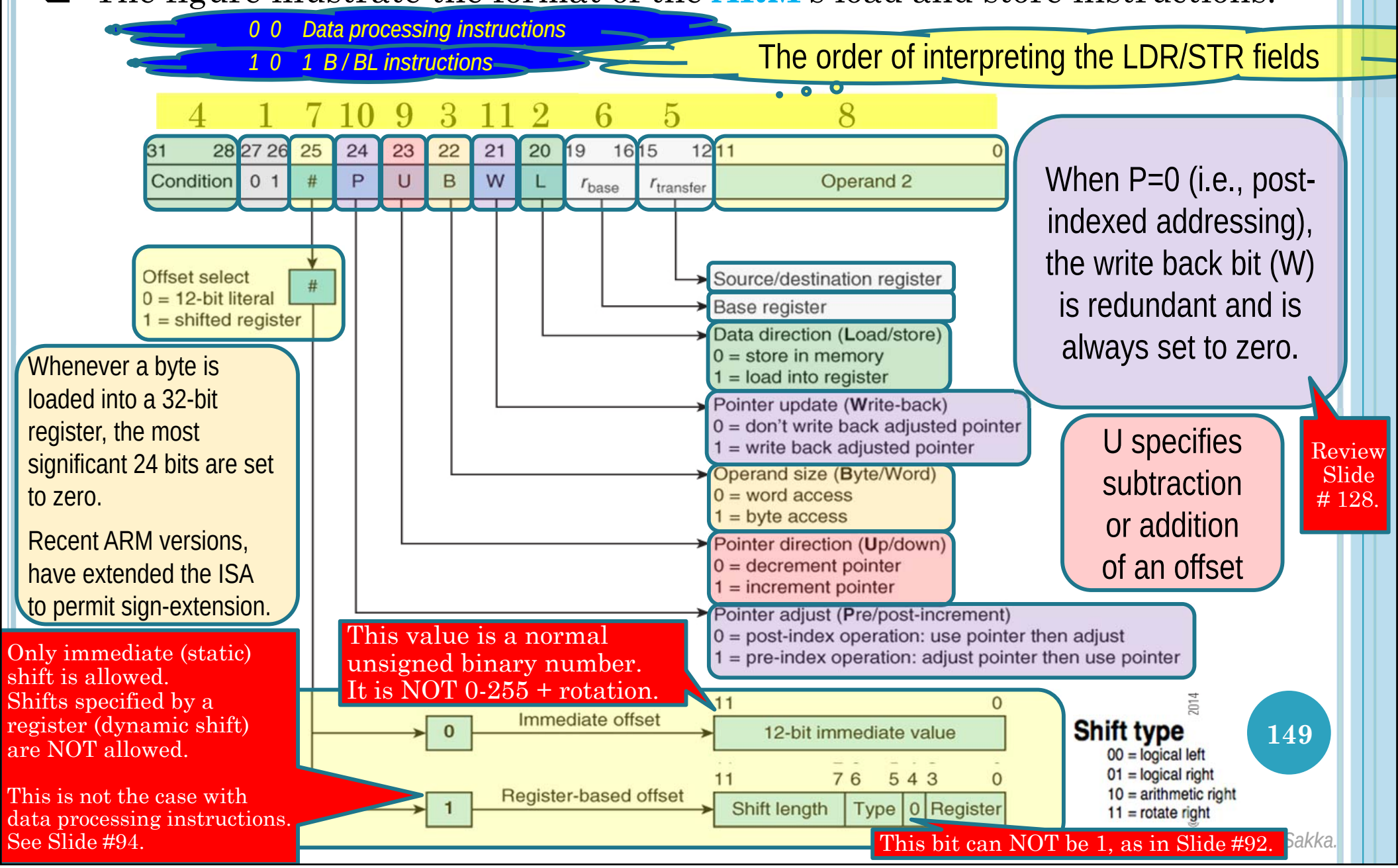
Architecture and Organization



CENGAGE Learning

ARM Load and Store Encoding

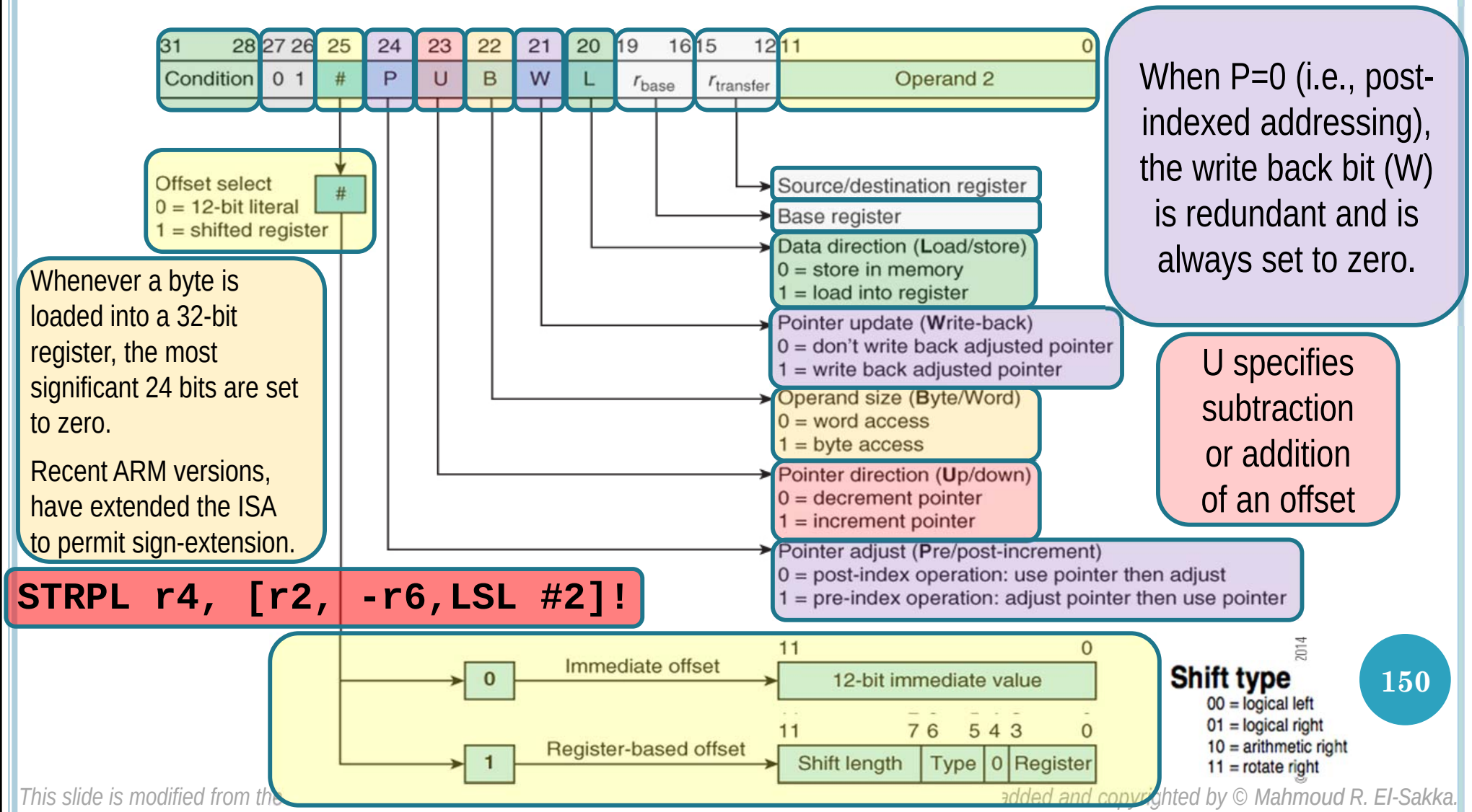
- The figure illustrate the format of the **ARM**'s load and store instructions.



ARM Load and Store Encoding

- Decode the following ARM machine code instruction **0x57224106**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	0	1	1	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	1	0



ARM Load and Store Encoding

Decoding the ARM Instruction **STRPL r4,[r2,-r6,LSL#2]!**

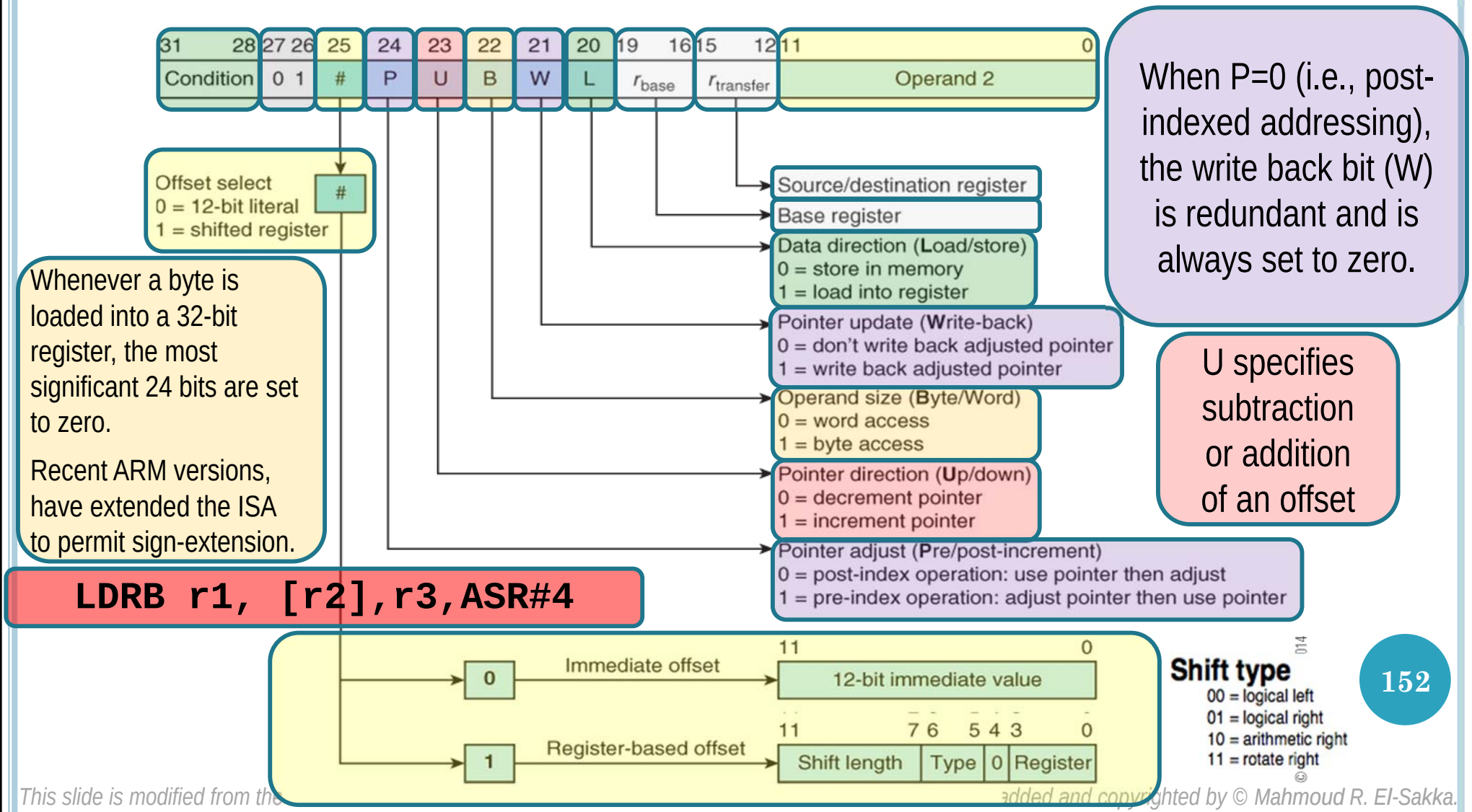
Field Name	Value	Action	Interpretation
Condition	0101	PL	Execute on positive
OP-code	01		Defines load/store instruction
#	1	Operand 2 format	Operand is a shifted register
P	1	Pre/post adjust	Adjust pointer before using
U	0	Pointer direction	Decrement pointer
B	0	Byte/word	This is a word access
W	1	Pointer write back	Update pointer after use
L	0	Load/store	Store data in memory
r_{base}	0010	Base register	r2 is the base (pointer) register
$r_{transfer}$	0100	Source/destination	r4 is the source in this store instruction
Shift length	00010	Shift length	Shift the register 2 places
Shift type	00	Logical shift left	Logical shift left the offset in r6
Op-code	0		
Shift register	0110	Specified register to be shifted	r6 is shifted twice

Operand 2

ARM Load and Store Encoding

- Decode the following ARM machine code instruction **0xE6D21243**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
1	1	1	0	0	1	1	0	1	1	0	1	0	0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	1	1



ARM Load and Store Encoding

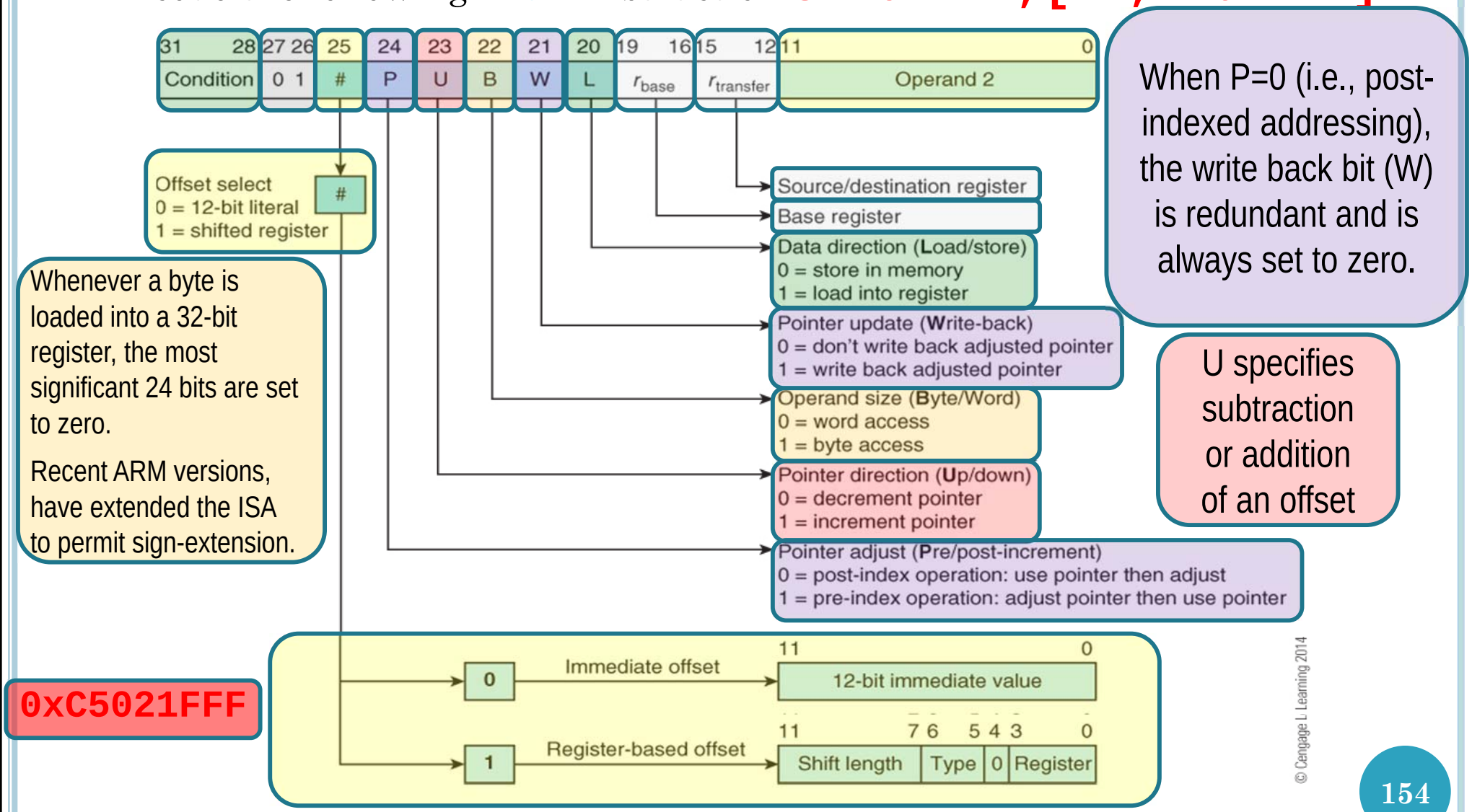
Decoding the ARM Instruction **LDR r1, [r2],r3,ASR#4**

Field Name	Value	Action	Interpretation
Condition	1110	AL	Always (default)
OP-code	01		Defines load/store instruction
#	1	Operand 2 format	Operand is a shifted register
P	0	Pre/post adjust	Adjust pointer after using
U	1	Pointer direction	Increment pointer
B	0	Byte/word	This is a word access
W	0	Pointer write back	As P=0, W is redundant and always=0
L	1	Load/store	Load data from memory
r_{base}	0010	Base register	r2 is the base (pointer)register
$r_{transfer}$	0001	Source/destination	r1 is the destination in this load instruction
Shift length	00100	Shift length	Shift the register 4 places
Shift type	10	Arithmetic shift right	Arithmetic shift right the offset in r3
Op-code	0		
Shift register	0011	Specified register to be shifted	r3 is shifted four times

Operand 2

ARM Load and Store Encoding

- Encode the following ARM instruction **STRGT r1,[r2,#-0xFFF]**



When P=0 (i.e., post-indexed addressing), the write back bit (W) is redundant and is always set to zero.

U specifies subtraction or addition of an offset

Whenever a byte is loaded into a 32-bit register, the most significant 24 bits are set to zero.

Recent ARM versions, have extended the ISA to permit sign-extension.

0xC5021FFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

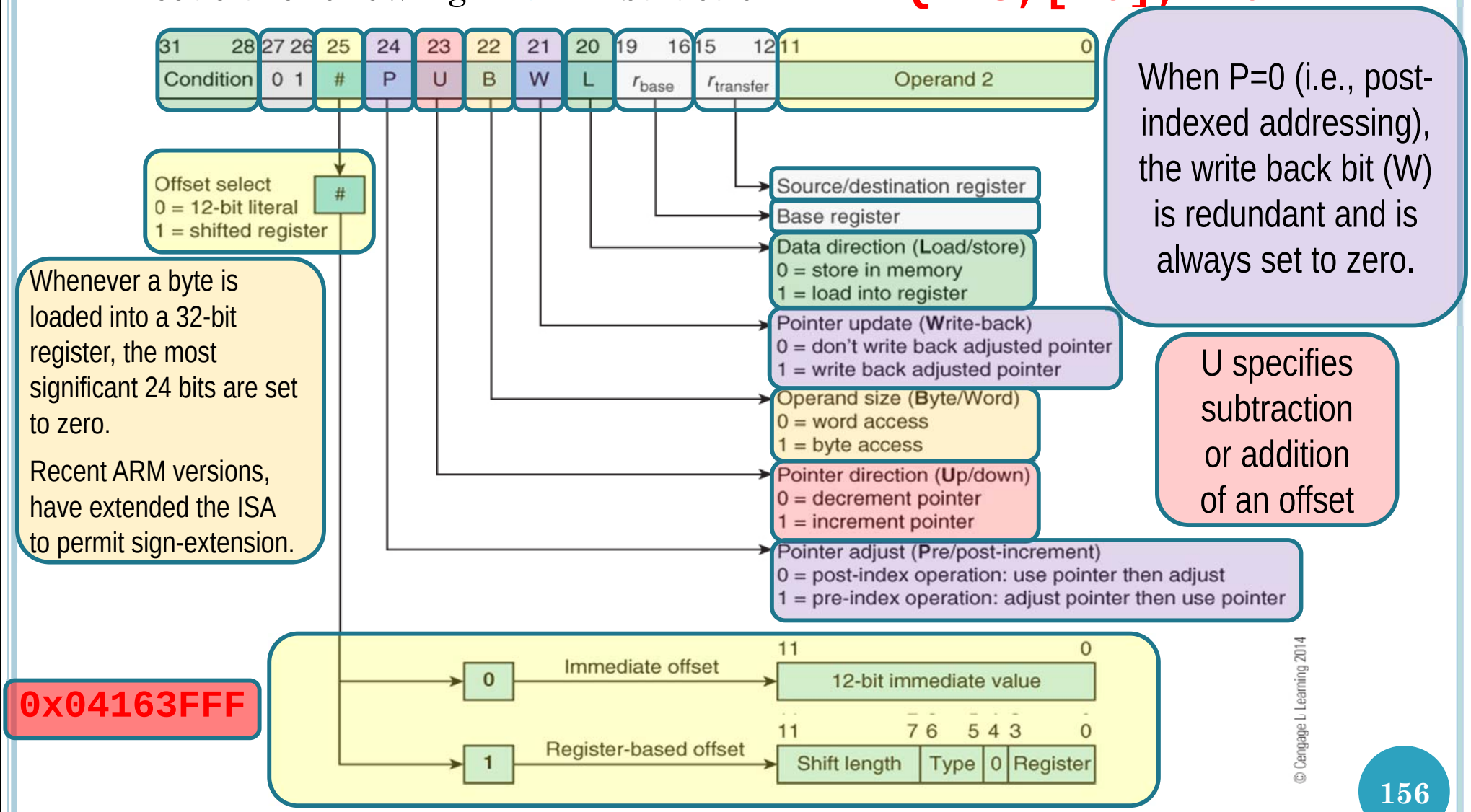
ARM Load and Store Encoding

Decoding the ARM Instruction **STRGT r1,[r2,#-0xFFF]**

Field Name	Value	Action	Interpretation
Condition	1100	GT	Execute on greater than
OP-code	01		Defines load/store instruction
#	0	Operand 2 format	Operand is immediate
P	1	Pre/post adjust	Adjust pointer before using
U	0	Pointer direction	Decrement pointer
B	0	Byte/word	This is a word access
W	0	Pointer write back	Update pointer before use
L	0	Load/store	Store data in memory
r_{base}	0010	Base register	r2 is the base (pointer) register
$r_{transfer}$	0001	Source/destination	r1 is the source in this store instruction
Immediate offset	111111111111	Shift length	Offset value = 0xFFF

ARM Load and Store Encoding

- Encode the following ARM instruction **LDREQ r3,[r6],#-0xFFF**



When P=0 (i.e., post-indexed addressing), the write back bit (W) is redundant and is always set to zero.

U specifies subtraction or addition of an offset

Whenever a byte is loaded into a 32-bit register, the most significant 24 bits are set to zero.

Recent ARM versions, have extended the ISA to permit sign-extension.

0x04163FFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	0	0	0	0	1	0	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1

ARM Load and Store Encoding

Decoding the ARM Instruction **LDREQ r3,[r6],#-0xFFF**

Field Name	Value	Action	Interpretation
Condition	0000	EQ	Execute on equal
OP-code	01		Defines load/store instruction
#	0	Operand 2 format	Operand is immediate
P	0	Pre/post adjust	Adjust pointer after using
U	0	Pointer direction	Decrement pointer
B	0	Byte/word	This is a word access
W	0	Pointer write back	Update pointer before use
L	1	Load/store	Load data from memory
r_{base}	0110	Base register	r6 is the base (pointer) register
$r_{transfer}$	0011	Source/destination	r3 is the destination in this load instruction
Immediate offset	111111111111	Shift length	Offset value = 0xFFF

ARM Load and Store Encoding

- Encode the following ARM instructions.

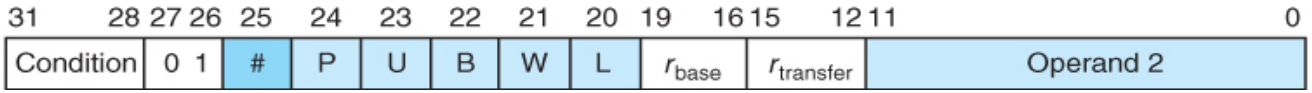
```
LDR R1,[R2]
LDR R1,[R2],#0
LDR R1,[R2,#0]
LDR R1,[R2,#0]!

STR R1,[R2]
STR R1,[R2],#0
STR R1,[R2,#0]
STR R1,[R2,#0]!
```

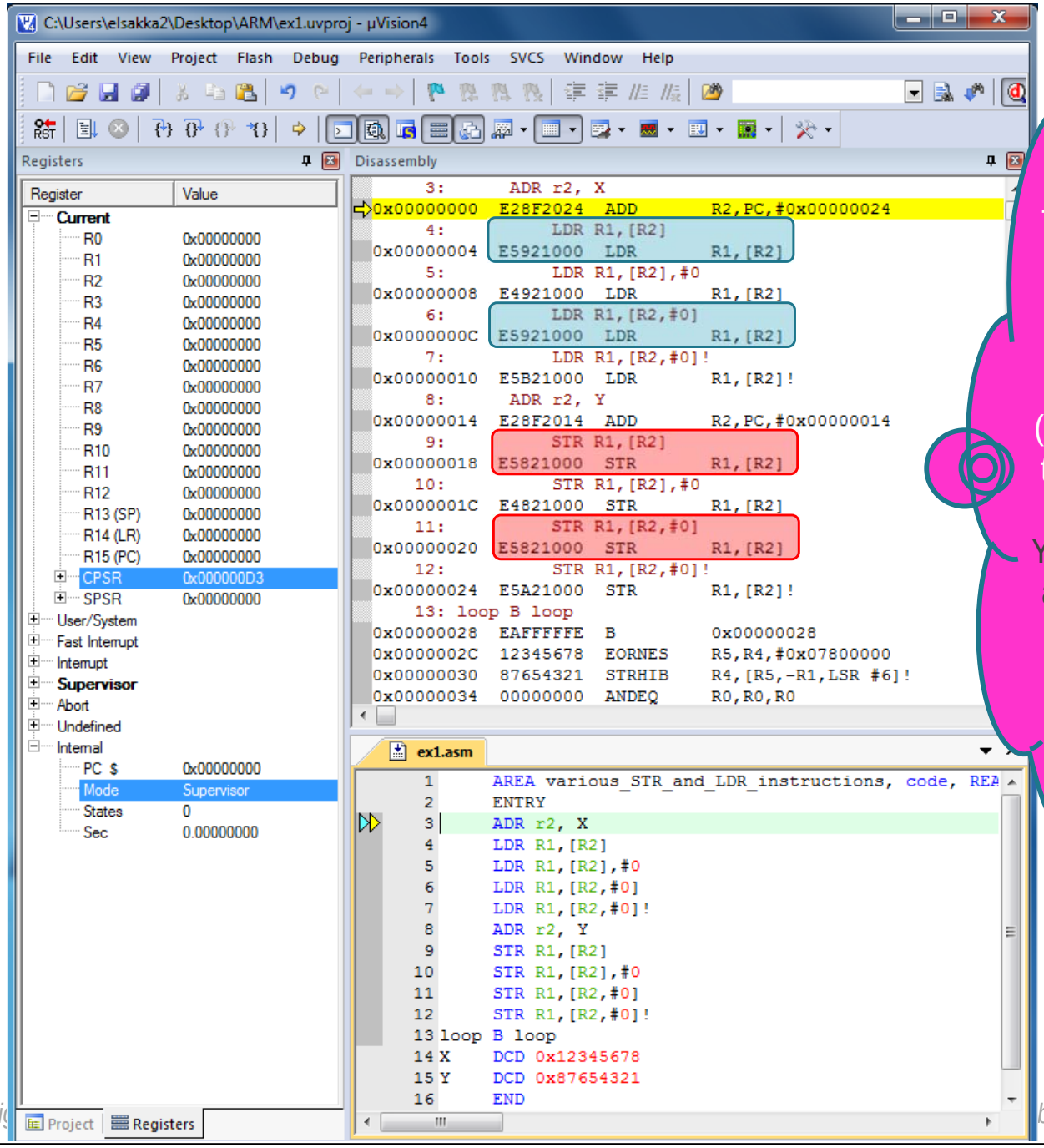
- Is there any *effective* difference between the 4 LDR instructions?
- Is there any *effective* difference between the 4 LDR instructions?

ARM Load and Store Encoding

```
      AREA various_STR_and_LDR_instructions, code, READONLY
      ENTRY
      ADR r2, X
      LDR R1,[R2]
      LDR R1,[R2],#0
      LDR R1,[R2,#0]
      LDR R1,[R2,#0]!
      ADR r2, Y
      STR R1,[R2]
      STR R1,[R2],#0
      STR R1,[R2,#0]
      STR R1,[R2,#0]!
loop  B loop
X     DCD 0x12345678
Y     DCD 0x87654321
      END
```



ARM Load and Store Encoding



To test this program, you need to change the permission of memory-locations from 0x30 to 0x33 (i.e., the location of Y) to make it read/write.

You also need to open a memory window to see the effect of the STR instructions.