LDR and STR Addressing Modes

Mode	Pseudo-code Operation *	Example
Offset modes – the base address register *is not* modified		
[Rn]	EA = Rn	`LDR R0, [R1]`
[Rn, #offset]	EA = Rn + offset	`LDR R0, [R1, #4]`
[Rn, Rm]	EA = Rn + Rm	`LDR R0, [R1, R2]`
[Rn, Rm, LSL #shift]	EA = Rn + (Rm << shift)	`LDR R0, [R1, R2, LSL #2]`
[Rn, Rm, LSR #shift]	EA = Rn + (Rm >> shift)	`LDR R0, [R1, R2, LSR #2]`
Pre-indexed modes – the base address register *is* modified *before* accessing memory		
[Rn, #offset]!	Rn = Rn + offset EA = Rn	`LDR R0, [R1, #4]!`
[Rn, Rm]!	Rn = Rn + Rm EA = Rn	`LDR R0, [R1, R2]!`
[Rn, Rm, LSL #shift]!	Rn = Rn + (Rm << shift) EA = Rn	`LDR R0, [R1, R2, LSL #2]!`
[Rn, Rm, LSR #shift]!	Rn = Rn + (Rm >> shift) EA = Rn	`LDR R0, [R1, R2, LSR #2]!`
Post-indexed modes – the base address register *is* modified *after* accessing memory		
[Rn], #offset	EA = Rn Rn = Rn + offset	`LDR R0, [R1], #4`
[Rn], Rm	EA = Rn Rn = Rn + Rm	`LDR R0, [R1], R2`
[Rn], Rm, LSL #shift	EA = Rn Rn = Rn + (Rm << shift)	`LDR R0, [R1], R2, LSL #2`
[Rn], Rm, LSR #shift	EA = Rn Rn = Rn + (Rm >> shift)	`LDR R0, [R1], R2, LSR #2`

***** EA is the *Effective Address*, which is the memory address to which the load or store operation is applied.

Note: only a subset of the above addressing modes can be used to load or store halfwords, signed-halfwords or signed-bytes. See ARM Architecture Reference Manual section A5.3.

STM and LDM – STore and LoaD Multiple

Instruction	Example	
Base register Rn <u>is not</u> modified (no !)		
STM*mode* Rn, {list}	`STMIA R12, {R0-R3}`	Store the contents of R0-R3 in memory at the address contained in R12
LDM*mode* Rn, {list}	`LDMIA R12, {R5,R7,R10}`	Load R5, R7 and R10 with the contents of memory at the address contained in R12
Base register Rn <u>is</u> modified (Rn!)		
STM*mode* Rn!, {list}	`STMFD SP!, {R4-R12,R14}`	Push R4-R12 and R14 on to the system stack, updating the system stack pointer
LDM*mode* Rn!, {list}	`LDMFD SP!, {R4-R12,R14}`	Pop 10 words off the top of the system stack into R4-R12 and R14, updating the system stack pointer

mode

STM – STore Multiple	
Instruction	Stack-Oriented Synonym
STMDB (decrement before)	**STMFD / PUSH** (full descending)
STMIB (increment before)	**STMFA** (full ascending)
STMDA (decrement after)	**STMED** (empty decending)
STMIA (increment after)	**STMEA** (empty ascending)

LDM – LoaD Multiple	
Instruction	Stack-Oriented Synonym
LDMIA (increment after)	**LDMFD / POP** (full descending)
LDMDA (decrement after)	**LDMFA** (full ascending)
LDMIB (increment before)	**LDMED** (empty decending)
LDMDB (decrement before)	**LDMEA** (empty ascending)