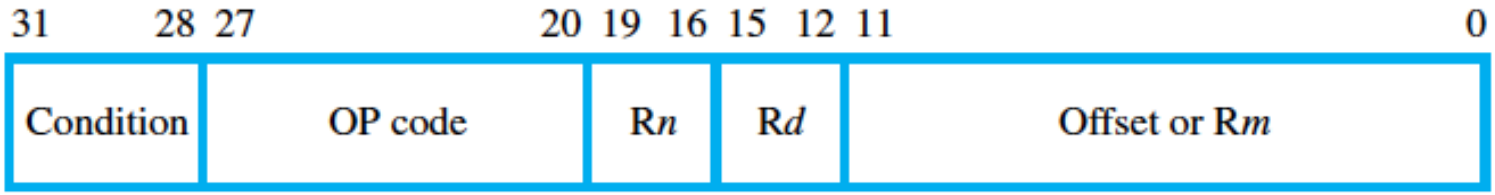
ARM Addressing Modes

- Basic Indexed Mode
  - Pre-indexed – effective address (EA) = contents of base register plus a signed offset
  - Load (LDR) & store (STR) instructions will be used to illustrate addressing modes



Format for Load and Store instructions.

High 4 bits in all instructions specify a condition that determines whether the instruction is executed.

ARM Addressing Modes

- Pre-indexed mode
  - A bit within the opcode indicates whether the offset is in the low 12 bits or whether the offset is in a register indicated by the low 4 bits of the instruction.
  - Offset is treated as an unsigned value (a bit within the opcode indicates sign for offset).
  - Examples:
    - LDR Rd,[Rn, #offset] ; Rd ← [[Rn] + offset]
    - LDR Rd,[Rn, Rm] ; Rd ← [[Rn]+[Rm]]
    - LDR Rd,[Rn] ; Rd ← [[Rn] + 0]

ARM Addressing Modes

- Relative Addressing Mode
  - The PC is used as the base register.
  - Programmer uses label and assembler determines offset
  - Offset = operand address - PC+8
  - Examples:
    - LDR R1,ITEM ; loads contents of memory location ITEM into R1.

ARM Addressing Modes

- Pre-indexed with writeback
  - Computed EA overwrites Rn
- Post-indexed
  - EA = [Rn]
  - Rn is then overwritten with EA + offset
- Offset in register may be scaled by power of 2
  - shifted right or left a specified amount (0 – 31)
  - direction & shift amount are encoded in Rm field

ARM Addressing Modes

- Example:
  - LDR R0,[R1, -R2,LSL #4]!
    - R0 ← [[R1] – 16*[R2]]
    - R1 is overwritten by EA (! specifies writeback)

ARM Addressing Modes

ARM indexed addressing modes.

Name	Assembler syntax	Addressing function
With immediate offset:		
Pre-indexed	[R*n*, #offset]	EA = [R*n*] + offset
Pre-indexed with writeback	[R*n*, #offset]!	EA = [R*n*] + offset; R*n* ← [R*n*] + offset
Post-indexed	[R*n*], #offset	EA = [R*n*]; R*n* ← [R*n*] + offset
With offset magnitude in R*m*:		
Pre-indexed	[R*n*, ± R*m*, shift]	EA = [R*n*] ± [R*m*] shifted
Pre-indexed with writeback	[R*n*, ± R*m*, shift]!	EA = [R*n*] ± [R*m*] shifted; R*n* ← [R*n*] ± [R*m*] shifted
Post-indexed	[R*n*], ± R*m*, shift	EA = [R*n*]; R*n* ← [R*n*] ± [R*m*] shifted
Relative (Pre-indexed with immediate offset)	Location	EA = Location = [PC] + offset

EA = effective address
offset = a signed number contained in the instruction
shift = direction #integer
where direction is LSL for left shift or LSR for right shift; and
integer is a 5-bit unsigned number specifying the shift amount
±R*m* = the offset magnitude in register R*m* can be added to or subtracted from the contents of base register R*n*

ARM Addressing Modes

- Register mode
  - Used for arithmetic & logic instructions
    - 2 source registers and a result register
- Absolute mode
  - If base register contains 0, 12-bit offset = absolute address

ARM Addressing Modes

- Immediate mode
  - Provided via pseudo-instructions
  - Format: LDR Rd,=value
  - Equal sign indicates immediate value
    - Examples:
    - LDR R2,=127 ; replaced by MOV R2,#127
    - LDR R2,=&ABCD3456 ;replaced by LDR R2,MEMLOC
      where MEMLOC contains hex ABCD3456
      "&" prefix denotes a hex value

ARM Addressing Modes

- Load 32-bit addresses
  - Examples:
  - ADR Rd,LOCATION ; loads 32-bit address into Rd
    - Assembler computes offset from current PC value
    - If LOCATION is in forward direction

      ADD Rd,R15,#offset ;is substituted
    - If LOCATION is in backward direction

      SUB Rd,R15,#offset ; is substituted
  - In either case, offset is an unsigned 8-bit number
  - Rotating the 8-bit number can give larger offsets