Registers:

1. R0-R15 register according to number
2. R0-R12 general purpose registers
3. R13 SP stack pointer
4. R14 LR link register
5. R15 PC program counter

ARM Address Modes:

1. Register Indirect Addressing:
    LDR r0,[r1]
        If r1 = 0x100
        Set r0 = mem[0x100]
    STR r0,[r1]
        If r1 = 0x100
        Set mem[0x100] = r0
2. Base-Plus-Offset Addressing
    Pre-indexing
        LDR r0, [r1,#4]; r0:=mem[r1+4]
        The offset (like #4 above) range: -4095 to +4095
    Post-indexing
        LDR r0,[r1],#4; r0:=mem[r1] then r1:=r1+4
    Auto-indexing
        LDR r0,[r1,#4]!; r0:=mem[r1+4] then r1:=r1+4

Single Register Load and Store Instructions

1. LDR: load word
    LDR r0, [r1,#1000]; retrieve a word(32 bits) from address (r1+1000)
2. LDRH: load halfword unsigned
    LDRH r0, [r1,#1000]; retrieve a halfword(16 bits) unsigned from address (r1+1000)
3. LDRB: load byte unsigned
    LDRB r0, [r1,#1000]; retrieve a byte(8 bits) from address (r1+1000)
4. LDRSH: load halfword sign extend
    LDRSH r0, [r1,#1000]; retrieve a halfword(16 bits) sign extend from address (r1+1000)
5. LDRSB: load byte sign extend
    LDRSB r0, [r1,#1000]; retrieve a byte(8 bits) sign extend from address (r1+1000)
6. STR: store word
    STR r0, [r1,#1000]; store all 32 bits of r0 to MEM[r1+1000]
7. STRH: store half-word
    STRH r0, [r1,#1000]; store 16 least significant bits of r0 to MEM[r1+1000]
8. STRB: store byte
    STRB r0, [r1,#1000]; store 8 least significant bits of r0 to MEM[r1+1000]

ARM Arithmetic Instructions

1. ADD: add
   ADD r0,r1,r2; r0 = r1+r2
2. SUB: subtract
   SUB r0, r1,r2; r0 = r1-r2
3. RSB: reverse subtract
   RSB r0,r1,r2; r0 = r2-r1
4. MUL: multiply
   MUL r0,r1,r2; r0 = r1*r2
5. AND: Bit-wise and
   AND r0,r1,r2; r0 = r1&r2
6. ORR: Bit-wise or
   ORR r0,r1,r2; r0 = r1|r2
7. EOR: Bit-wise exclusive-or
   EOR r0,r1,r2; r0 = r1 XOR r2
8. BIC: bit clear
   BIC r0,r1,r2; r0 = r1 & NOT(r2)
9. LSL: logical shift left
   Fills with zeroes
   ADD r0, r1, r2, LSL, #1; r0 = r1 + (r2 << 1)
10. LSR: logical shift right
   ADD r0, r1, r2, LSR, #1; r0 = r1 + (r2 >> 1)
11. ASR: arithmetic shift right
   Shift in the most significant bit
12. ROR: rotate right

ARM Move Instructions

1. Register-to-register
2. MOV: move
   MOV r0,r1; r0 = r1
3. MVN: move(negated)
   MVN r0,r1; r0 = NOT(r1)

ARM Comparison Instructions

(Do NOT modify general registers. Set the values of PSR register.
selected PSR bits: N(negative), Z(zero), C(carry).)

1. CMP: compare
   CMP r0, r1; compute (r0-r1) and set PSR
2. CMN: negated compare
   CMN r0, r1; compute (r0+r1) and set PSR

ARM Flow of Control

1. Branches use the status register set by the CMP instructions.
2. Branch instruction:
    B #100; add 400 to the current PC value
    B LABEL; branch a LABEL
3. Branch and Link instruction (BL)
    Used for subroutine call
    The return address is saved in r14
    Example:
        BL foo
        Save the next PC value to r14 and then jump to foo
    To return from subroutine: MOV r15 r14

ARM Condition Code

Most instructions can have any one of the condition codes after the instruction code, which means to only execute under such condition (decided based on PSR)

EQ	equal
NE	nonequal
GE	signed greater than or equal
GT	signed greater than
LE	signed less than or equal
LT	signed less than
AL	always(normally omitted)