ARM Instruction Set

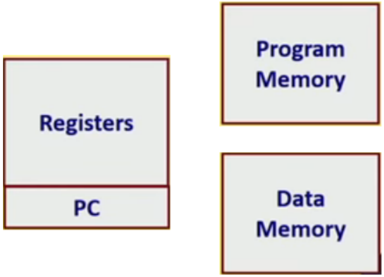
Dr. Munesh Singh

Indian Institute of Information Technology Design and Manufacturing
Kancheepuram, Chennai, Tamil Nadu

January 13, 2020

The ARM Instruction Set

- ARM instruction can be categorized into three groups:
  1. Data processing instructions
     - Operate on value in registers
  2. Data transfer instructions
     - Move values between registers and memory
  3. Control flow instructions
     - Change the value of the program counter (PC)



Data Processing Instructions

- All operands are 32-bits in size:
  - Either registers
  - Or literals (immediate values) specified as part of the instruction
- The result, if any,is also 32-bit in size and goes into a specified register.
  - One exception: long multiply, that generates 64-bit results.
- All operand and result registers are independently specified as part of the instruction.

Arithmetic Instructions:

- ADD r0,r1,r2; r0=r1+r2
- ADC r0,r1,r2; r0=r1+r2+c (c is carry bit)
- SUB r0,r1,r2; r0=r1-r2
- SBC r0,r1,r2; r0=r1-r2+c-1
- SBC r0,r1,r2; r0=r2-r1
- RSC r0,r1,r2; r0=r2-r1+c-1
- All operations can be viewed as either unsigned or 2's complement signed
    - Means the same thing.

Bit-wise logical instructions:

- AND r0,r1,r2; r0=r1 and r2
- ORR r0,r1,r2; r0=r1 or r2
- EOR r0,r1,r2; r0=r1 xor r2
- BIC r0,r1,r2; r0=r1 and not r2
- BIC is the acronym for "bit clear"
    - Each 1-bit in r2 clears the corresponding bit in r1.

Register-register move instructions:

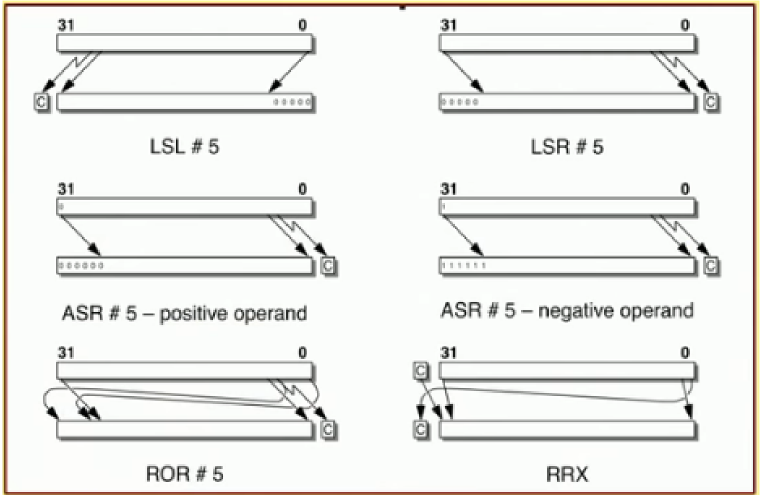
- MOV r0,r2;   r0=r2
- MVN r0,r2;   r0=not r2
- MVN is the acronym for "move negated"
  - Each 1-bit in r2 clears the corresponding bit in r0.
- In the instruction encoding, the first operand r1 is not specified, as these are unary operations.

Specifying immediate operands:

- ADD r1,r2,#2;   r1=r2+2
- SUB r3,r3,#1;   r3=r3-1
- AND r6,r4,#&0f;   r6=r4[3:0]
- Notations:
  - # indicates immediate value
  - & indicates hexadecimal notation
- Allowed immediate values:
  - 0 to 255 (8 bits), rotated by any number of bit positions that is multiple of 2

Shifted register operands:

- The second source operand may be shifted either by a constant number of bit positions, or by a register-specified number of bit positions.
- ADD r1,r2,r3,LSL #3; r1=r2+(r3<< 3)
- ADD r1,r2,r3,LSL r5; r1=r2+(r3<< r5)
- Various shift and rotate options:
  - LSL: logical shift left
  - LSR: logical shift right
  - ROR: rotate right
  - RRX: rotate right extended by 1 bit
  - ASL: Arithmetic shift left
  - ASR: Arithmetic shift right


31
0
C
LSL # 5
LSR # 5
ASR # 5 – positive operand
ASR # 5 – negative operand
ROR # 5
RRX

Multiplication Instructions

Multiplication instruction

- MUL r1,r2,r3; $r1=(r2\times r3)[31:0]$
- Only the least significant 32-bits are returned.
- Immediate operands are not supported.

Multiply-accumulate instruction:

- MLA r1,r2,r3,r4; $r1=(r2\times r3+r4)[31:0]$
- Required in digital signal processing (DSP) applications.
- Multiplication with 64-bit results is also supported.

Thank You