SPRING 2016
CMPE 364

Microprocessor Based Design

Dr. Ryan Riley
(Slides adapted from Dr. Mohamed Al-Meer)

Introduction to the ARM Instruction Set

DATA PROCESSING
INSTRUCTION

Data Processing Instruction Format

- Most Data Processing Instructions follows:
  MNEMONIC DST, SRC1, SRC2
- Mnemonic is a short name for the operation
- Like ADD, SUB, UMUL, EOR
- Destination is first register listed (must GP Register = R0 – R15)
- will represent hexadecimal numbers with the prefix **0x** and binary numbers with the prefix **0b**.
- Memory will denoted as **mem<data_size>[address]**
- Where data size bits of memory starting at the given byte address

Data Processing Instruction Format

- Most data processing instructions can process one of their operands using the **barrel shifter**.
- If you use the **S suffix** on a data processing instruction, then it updates the flags in the **CPSR**.
- Move and logical operations update the carry flag *C*, negative flag *N*, and zero flag *Z.*
- The carry flag is set from the result of the barrel shift as the last bit shifted out.
- The *N* flag is set to bit 31 of the result. The *Z* flag is set if the result is zero.

MOV Instruction

- It copies *N* into a destination register *Rd*, where *N* is a register or immediate value.
- useful for setting initial values and transferring data between registers

MOV DST, SRC

```
Syntax: <instruction>{<cond>}{S} Rd, N
```

MOV	Move a 32-bit value into a register	$Rd = N$
MVN	move the NOT of the 32-bit value into a register	$Rd = \sim N$

MOV Instruction

- Operand N is usually it is a register *Rm* or a constant preceded by #.
- Example-1
  given the instruction shown below, get R3 and R9 after execution. Assume R3 = 0xFEEA082C and R9 = 0x8000AC40?

MOV R3, R9

- Solution
  R3 = 0x8000AC40 and R9 remains the same.

MOV Instruction

- Example 2
- Determine R1 after execution on next instruction assuming it contains = 0x2E059401?

MOV R1, #0x0000009C

- Solution

R1 = 0x0000009C

MOV Instruction

- Example 3
- Determine R10 after execution on next instruction assuming it contains = 0x2E059401?

MOV R10, #384

- Solution

R10 = 0x00000180

MOV Instruction

This example shows a simple move instruction. The MOV instruction takes the contents of register *r5* and copies them into register *r7*, in this case, taking the value 5, and overwriting the value 8 in register *r7*.

```
PRE     r5 = 5
        r7 = 8
        MOV     r7, r5     ; let r7 = r5
POST    r5 = 5
        r7 = 5
```

MVN Instruction

- Move Negative Instruction.
- Has same effect of MOV instruction but copies **the one's complement** of the source to destination.
- Source: general purpose register or immediate

MVN DST, SRC

Example

MVN Instruction

- Example: Determine the content of R2 and R5 after the execution of the following instruction. Assume R2 = 0xA6E9F004, R5 = 0xCE00A824?

  MVN R2, R5

- Solution:

  R2 = 0x31FF57DB, R5 = SAME.

MVN Instruction

- Example: Determine the content of R4 after the execution of the following instruction?

  MVN R4, #24

- Solution:

BEFORE: #24 = $0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0001\ 1000_2$.

#24 = **0x00000018**.

AFTER: R4 = $1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1110\ 0111_2$.

R4 = **0xFFFFFFE7**.

Arithmetic Instructions

- The arithmetic instructions implement addition and subtraction of 32-bit signed and unsigned values.

Syntax: <instruction>{<cond>}{S} Rd, Rn, N

ADC	add two 32-bit values and carry	$Rd = Rn + N +$ carry
ADD	add two 32-bit values	$Rd = Rn + N$
RSB	reverse subtract of two 32-bit values	$Rd = N - Rn$
RSC	reverse subtract with carry of two 32-bit values	$Rd = N - Rn -$!(carry flag)
SBC	subtract with carry of two 32-bit values	$Rd = Rn - N -$!(carry flag)
SUB	subtract two 32-bit values	$Rd = Rn - N$

N is the result of the shifter operation. The syntax of shifter operation is shown in Table 3.3.

ADD Instruction

- ADD instruction adds 2 source operands SRC1 and SRC2, and stores the result in destination register DST.
- The first operand must be general purpose register, while the other can be a register or an immediate operand.

ADD DST, SRC1, SRC2

ADD Examples

• Example:

Get R0, R3, and R9 after the next instruction, if R0 =0x00014009, and R3 = 0x00326018?

ADD R9, R0, R3

Answer:

R9 = **0x0033A021**

R0 and R3 No Change

ADD Examples

• Example:

Get R1and R6 after the next instruction, IF:

R6 = 0xFFFFFFFE (-2?)

ADD R1, R6, #24

Answer:

R1 = **0x00000016 = 22**.

SUB Instruction

- The SUB instruction subtracts the 2nd Source from the 1st Operand and replaces the result in destination register.

  SUB DST, SRC1, SRC2

- EXAMPLE: Get R4 if R2 = 0x000006A0, R1 = 0x000003C4

  SUB R4, R2, R1

- SOLUTION: R4 = **0x000002DC**

SUB Instruction

- EXAMPLE: Get R4 if R2 = 0x000008E0?

  SUB R4, R2, #0xFFFFFFFE (-1)

- SOLUTION:

  R4 = 0x000008E0 – 0xFFFFFFFE = **0x000008E2**

RSB Instruction

- RSB is Reverse Subtract Instruction.
- Subtracts SRC1 from SRC2

RSB DST, SRC1, SRC2

- EXAMPLE: Show how the subtracts occur? If R3 = 0x000006BE, R9 = 0x000009AC?

RSB R8, R3, R9

- SOLUTION: R8 = R9 – R3 = **0x000002EE**

RSB Instruction

- EXAMPLE:
  - Write Assembly instruction to subtract R7 from 1000 and replace result in R7?
- SOLUTION:

RSB R7, R7, #1000

- EXAMPLE
  - Write a single code to convert the sign of R1 register without knowing its content?
- SOLUTION:

RSB R1, R1, #0