

Chapter 4
ARM Arithmetic and Logic Instructions
Exercises

Z. Gu

Fall 2025

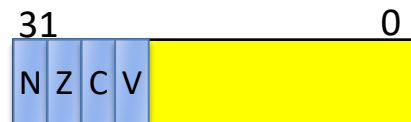
Summary of Carry and Overflow Flags

Bit	Name	Meaning after add or sub
N	negative	result is negative
Z	zero	result is zero
V	overflow	signed overflow
C	carry	unsigned overflow

Carry flag C = 1 upon an **unsigned** addition if the answer is wrong (true result $> 2^n - 1$)

Carry flag C = 0 (Borrow flag = 1) upon an **unsigned** subtraction if the answer is wrong (true result < 0)

Overflow flag V = 1 upon a **signed** addition if the answer is wrong (true result $> 2^{n-1} - 1$ or true result $< -2^{n-1}$)



CPSR (Current Program Status Register)

References

- ▶ Lecture 2: Carry flag for unsigned addition and subtraction
 - ▶ <https://www.youtube.com/watch?v=MxGW2WurKuM&list=PLRJhV4hUhlymmp5CCeIFPyxbknsdcXCc8&index=2>
- ▶ Lecture 3: Overflow flag for signed addition and subtraction
 - ▶ <https://www.youtube.com/watch?v=BlN6iyYIGio&list=PLRJhV4hUhlymmp5CCeIFPyxbknsdcXCc8&index=3>



Flags ANDS

- ▶ What are value of r2, and NZCV flags after execution, assuming all flags are initially 0.

```
LDR r0, =0xFFFFFFFF00  
LDR r1, =0x00000001  
ANDS r2, r1, r0, LSL #1
```

Flags ADDS

- ▶ What are value of r2, and NZCV flags after execution, assuming all flags are initially 0.

```
LDR r0, =0xFFFFFFFF00  
LDR r1, =0x00000001  
ADDS r2, r1, r0, LSL #1
```

r0	0xffffffff
r1	0x00000001
r2	0x00000003
r3	0xffffffff0

Flags

- ▶ Suppose registers have the following values:
- ▶ What are value of r4, and NZCV flags after execution, assuming all flags are initially 0. (Each instruction runs individually.)
- ▶ (a) ADD r4, r0, r2, ASR #3
- ▶ (b) ADDS r4, r0, r1
- ▶ (c) LSRS r4, r0, #1
- ▶ (d) ANDS r4, r0, r3
- ▶ (e) CMP r2, #3

Barrel Shifter: Explanations

- ▶ LSL (logical shift left): **shifts left, fills zeros on the right**; C gets the last bit shifted out of bit 31. This is multiply by 2^n for non-overflowing values.
- ▶ LSR (logical shift right): **shifts right, fills zeros on the left**; C gets the last bit shifted out of bit 0. This is unsigned division by 2^n .
- ▶ ASR (arithmetic shift right): **shifts right, fills the sign bit on the left** to preserving the sign; C gets the last bit shifted out of bit 0. This is signed division by 2^n with sign extension
- ▶ ROR (rotate right): **rotates bits right with wraparound**; bits leaving bit 0 re-enter at bit 31, and C receives the bit that wrapped. This is a pure rotation without data loss.
- ▶ RRX (rotate right extended): **rotates right by one through the carry flag**, treating C as a 33rd bit; new bit 31 comes from old C, and C receives old bit 0.

Arithmetic with Shifts

- ▶ Assuming 32-bit registers:
- ▶ Q1:
 - ▶ LDR r0, =0x00000007
 - ▶ MOV r0, r0, LSL 7
- ▶ Q2:
 - ▶ LDR r0, =0x00000400
 - ▶ MOV r0, r0, LSR 2
- ▶ Q3:
 - ▶ LDR r0, =0xFFFFC000
 - ▶ MOV r0, r0, LSR 2
- ▶ Q4:
 - ▶ LDR r0, =0xFFFFC000
 - ▶ MOV r0, r0, ASR 2
- ▶ Q5:
 - ▶ LDR r0, =0x00000007
 - ▶ MOV r0, r0, ROR 2

Assembly Programming

- ▶ Write ARMv7 assembly for pseudocode
 - ▶ $r1 = (r0 \gg 4) \& 15$