DATE
October 19, 2025

SIGNED INTEGERS COMPARISON IN ARM ARCHITECTURE

CSCI 6461 Computer Systems Architecture

Delivered by:
Rufat Guliyev
rguliyev24988@ada.edu.az

Instructor:
Dr. John Thomas Burns

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

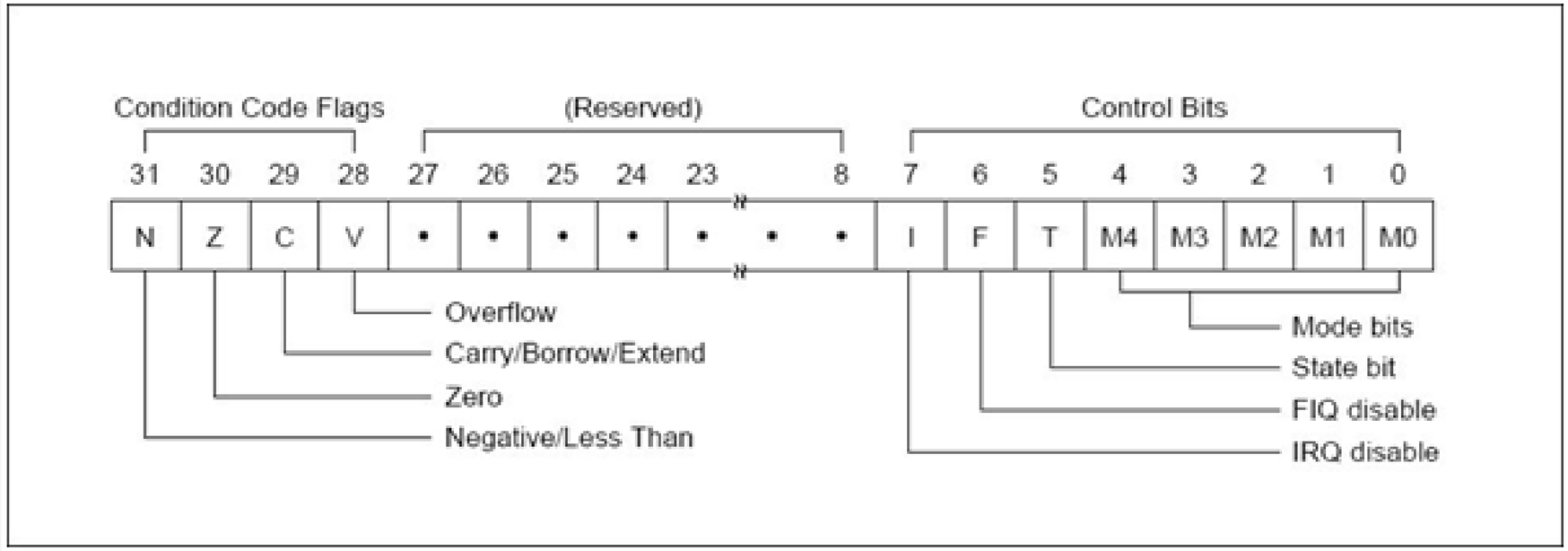
UNIVERSITET ◦ UNIVERSITY ADA

CPSR

What is the CPSR?

The Current Program Status Register (CPSR) is a crucial control register found in ARM architecture CPUs. It's essentially a special 32-bit register that tracks the overall operational state of the processor at any given moment.

Figure 1. CPSR register structure



Key Functions

The CPSR is divided into several fields, but for our purposes today, we focus entirely on the top four bits, which are the Condition Code Flags (or status flags):

Unsigned Comparison Recap

The Principle: Simple Magnitude

- Unsigned numbers treat the full 8-bit pattern as a positive magnitude (0 to 255).
- The CPU compares A and B by performing the subtraction: A−B.
- We only check the Carry (C) and Zero (Z) flags—the N and V flags are ignored.

Figure 2. Unsigned comparison

Flag	Role	Logic in Subtraction ($\mathbf{A} - \mathbf{B}$)
$\mathbf{Z}$ **(Zero)**	Checks for equality.	$\mathbf{Z} = 1$ if $\mathbf{A} = \mathbf{B}$. (Result is 0)
$\mathbf{C}$ **(Carry)**	Checks for Higher/Lower.	Acts as the **'No Borrow'** indicator.

Figure 3. Role of C borrow flag

Unsigned conditional codes rely only on the state of C and Z. We ignore N and V, which is why they cannot handle signed negative numbers or overflow

Comparison	Condition	$\mathbf{C}$ Flag Status
$\mathbf{A} \geq \mathbf{B}$ (Higher or Same)	No borrow required.	$\mathbf{C} = 1$ **(Set)**
$\mathbf{A} < \mathbf{B}$ (Lower)	A borrow is required.	$\mathbf{C} = 0$ **(Cleared)**

Why Unsigned Logic Fails for Signed CMP

The Core Problem: The Sign Bit (N)

- Signed numbers use the Most Significant Bit (MSB) as the sign indicator:
  - 0 = Positive
  - 1 = Negative
- The Negative (N) flag mirrors this MSB. In most cases, N accurately reflects the sign of the result of A−B.

The Unsigned C flag is irrelevant for signed comparison because of Signed Overflow.
C only tracks carries/borrows out of the final bit—it ignores signed magnitude.
If one considers a 8-bit CPU then subtraction occasionally produces a result outside its range (−128 to +127).

The Effect of Overflow
When signed overflow occurs:

1. The mathematical result is too large or too small for the register.
2. The resulting Sign Bit (N) is inverted and is now WRONG.

Example: Calculating 127−(−1) should be +128. But in 8-bit signed math, +128 overflows and is stored as −128.
The N flag incorrectly indicates a negative result.
Conclusion: We need a new flag to detect this N-bit lie. We need V (Overflow).

The Solution: N⊕V - The Corrected Sign

The Necessary Flags

To overcome the sign inversion caused by overflow, we shift our focus from C to N and V:

- **N (Negative)** - The primary indicator of the result's sign.
- **V (Overflow)** - The crucial "Mistake Detector" flag. V=1 only when N is lying.

The Logic: N⊕V

We combine N and V using the Exclusive OR (⊕) logic gate. This combination yields the mathematically correct sign of the subtraction A−B.

Correct Sign: **N⊕V**

- N⊕V=0: The result of A−B is Positive (or Zero).
- N⊕V=1: The result of A−B is Negative.

The Universal Rule for Signed GE

This N⊕V result is the universal determinant for the signed condition Greater than or Equal (GE).

A≥B IF **N⊕V=0**

V as the Correction Factor

The V Flag: The Sign Inverter

Think of the Overflow flag (V) as a sign inverter that only "turns on" when the Negative flag (N) is giving a mathematically incorrect sign. The Exclusive OR (⊕) operation performs this correction:

$$N \oplus V = \text{Corrected Sign}$$

The pattern N⊕V ensures we read the mathematically correct sign of A−B by using V to nullify or correct the misleading N whenever a signed overflow error occurs. This core concept allows us to analyze all four specific signed conditional codes

Signed Greater Than or Equal

N⊕V or **(NOT(N XOR V))**

The Requirement
The GE condition is met if the result of the subtraction, A−B, is Zero or Positive.

Logic Breakdown
This condition uses the core corrected sign indicator, N⊕V:

1. Positive Check (A>B):
    - A positive result requires the Corrected Sign to be positive.
    - Since N⊕V=1 means Negative, a positive result requires N⊕V to be 0.
    - (Example: 5−3=2. Flags: N=0,V=0. Check: 0⊕0=0. Pass.)
2. Zero Check (A=B):
    - If A=B, the result is zero, which sets Z=1.
    - A zero result always yields N=0 and V=0, meaning N⊕V is 0.

Signed Less Than

N⊕V or **(N XOR V)**

The Requirement
The LT condition is met if the result of the subtraction, A−B, is a strictly Negative number.

Logic Breakdown
This condition is the direct inverse of the GE condition, relying entirely on the corrected sign indicator, N⊕V:

1. Negative Check (A<B):
    - A negative result requires the Corrected Sign to be negative.
    - Since N⊕V=1 is the indicator for Negative, the condition is met when N⊕V evaluates to 1.
    - (Example: −10−20=−30. Flags: N=1,V=0. Check: 1⊕0=1. Pass.)
2. Zero Check (A not equal to B):
    - Since LT requires the result to be strictly less than, the Z flag must be 0.
    - If A=B, Z=1, but N⊕V would be 0, causing the LT condition to fail, which is correct.

Signed Greater Than

$Z \text{ AND } (N \oplus V)$

The Requirement

The GT condition is met if the result of the subtraction, A−B, is a strictly Positive number. This requires two things to be true simultaneously:

1. The result must be positive (the GE condition).
2. The result must not be zero (it must exclude the case A=B).

Logic Breakdown

This condition combines the Zero flag (Z) with the Corrected Sign ($N \oplus V$) using an AND operation:

1. Positive Check (Must be GE):
   - The corrected sign must be positive. This means $N \oplus V$ must evaluate to 0. We use the NOT of this: $(N \oplus V)$.
2. Not Zero Check (Must be A not equal to B):
   - The Z flag must be 0. We use the NOT of the Z flag: Z.

Signed Less Than or Equal

The Requirement
The LE condition is met if the result of the subtraction, A−B, is either Zero or a Negative number. This means the condition is the direct logical opposite of GT (A>B).

Logic Breakdown
Since either a negative result OR a zero result satisfies the condition, we combine the checks using the logical OR operation:

1. Negative Check (A<B):
    - The corrected sign must be negative. This requires N⊕V to evaluate to 1. We use the term (N⊕V).
2. Zero Check (A=B):
    - The result can be zero. This requires the Z flag to be 1. We use the term Z.

DATE
October 19, 2025

THANK YOU FOR ATTENTION

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

UNIVERSITET • UNIVERSITY
ADA

References

- CPSR register structure illustration
  https://www.ques10.com/p/41159/arm7-a-32-bit-microcontrollerpart-2-1/
- Figures 2 and 3
  Generated by Google Gemini