In computer processors, the **overflow flag** (sometimes called V flag) is usually a single bit in a system status register used to indicate when an arithmetic overflow has occurred in an operation, indicating that the signed two's-complement result would not fit in the number of bits used for the operation (the ALU width). Some architectures may be configured to automatically generate an exception on an operation resulting in overflow.

Although not very precise, the overflow flag could be considered a two's complement form of a carry flag, but the typical usage is quite different.

An illustrative example is what happens if we add 127 and 127 using 8-bit registers. 127+127 is 254, but using 8-bit arithmetic the result would be 1111 1110 binary, which is -2 in two's complement, and thus negative. A negative result out of positive operands (or vice versa) is an overflow. The overflow flag would then be set so the program can be aware of the problem and mitigate this or signal an error. The overflow flag is thus set when the most significant bit (here considered the sign bit) is changed by adding two numbers with the same sign (or subtracting two numbers with opposite signs). Overflow never occurs when the sign of two addition operands are different (or the sign of two subtraction operands are the same).

Internally, the overflow flag is usually generated by an exclusive or of the internal carry *into* and *out of* the sign bit. As the sign bit is the same as the most significant bit of a number *considered* unsigned, the overflow flag is "meaningless" and normally ignored when unsigned numbers are added or subtracted.

The overflow flag is typically changed by all arithmetic operations, including compare instructions (equivalent to a subtract instruction without storing the result). In many processor architectures, the overflow flag is cleared by bitwise operations (and, or, xor, not), possibly including shifts and rotates, but it may also be left undefined by these. Instructions such as multiply and divide often leave the flag undefined, or affected by the last partial result.