What is a “register”?

It is a special location of a fixed size of bits, where binary (base 2) arithmetic takes place.

For example: Java has a 32 bit, signed integer register
(1 byte = 8 bits) (32 bits = 4 bytes) (So, what’s a nybble?4 bits)

This is an example of a 16 bit register (size = 16 bits):

| R: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

How do we note a “register”?

R:

The values listed after the “R” are assumed to be binary (base 2), and DO NOT require a subscript.
(“R” implies a binary value to follow.)

You must include all leading zero’s (0) in a register as “padding”.

The decimal value 4 is illustrated in the above 16 bit register.

How many values can we store in a register?
$2^n$, where n = size of the register.
Thus, a 16 bit register can store $2^{16}$ values or 65,536 values

Hence the range of values 0 to 65,535 (for an unsigned register) or $2^n - 1$ (the minus 1 representing the value 0)

How do we represent – 4 (negative 4)?

| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

Let’s see why this is the correct answer….

# Signed Integer Registers

How do they work?
(2's complement method)

| How many values can be stored in a 4 bit register?<br>**$2^4 = 16$** | Binary #<br>(or all possible bit combinations) | Decimal Value Of Binary Number | Integers are Positive ( 0 ) Or Negative ( 1 ) | R: □□□□<br>4-bit Register "R"<br>**Uses *Signed* integers**<br>(The bit pattern represents this decimal value.) |
|---|---|---|---|---|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |
| 6 | | | | |
| 7 | | | | |
| 8 | | | | |
| 9 | | | | |
| 10 | | | | |
| 11 | | | | |
| 12 | | | | |
| 13 | | | | |
| 14 | | | | |
| 15 | | | | |
| 16 | | | | |

Finding 2's complement representation of a binary value within a register:

1. Find the positive value
2. Flip the bits (0's become1's, and 1's become 0's)
3. Add One (+1)
4. Result is your answer.

More Issues With Integer Registers…

Detecting Overflow – when the sign bit of the result value is negative after adding 2 positive values.

Adding 2 positive values always results in a positive value.

Why isn't this a problem for mixed sign addition, or subtraction?

You will always remain within the range of the register!

So what is the "range" of a signed integer register?

$-2^{n-1}$ to $+ (2^{n-1} -1)$

Let's look at the 16 bit register again…this time as a "signed integer register":

n=16

- 32,768 to +32,767

Note: Range is always expressed from negative to positive value.