FLOATING POINT

<--->

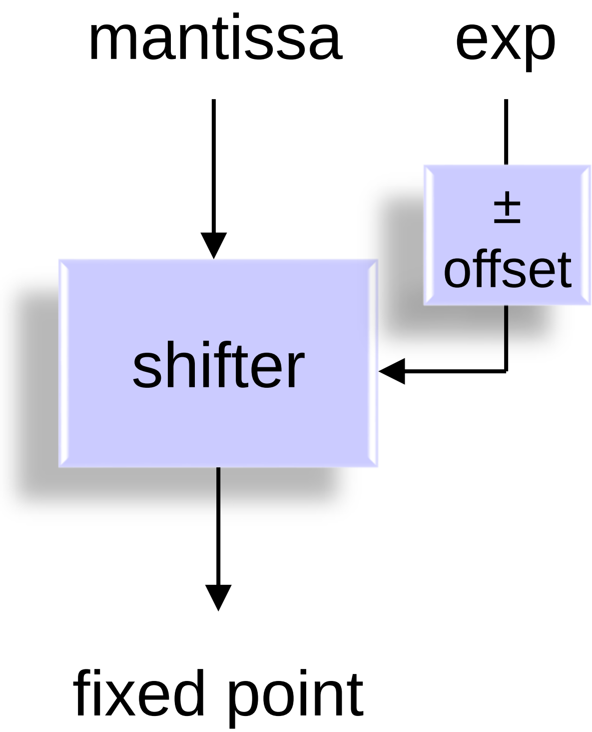
FIXED POINT

CONVERSION

Floating Point → Fixed Point Conversion

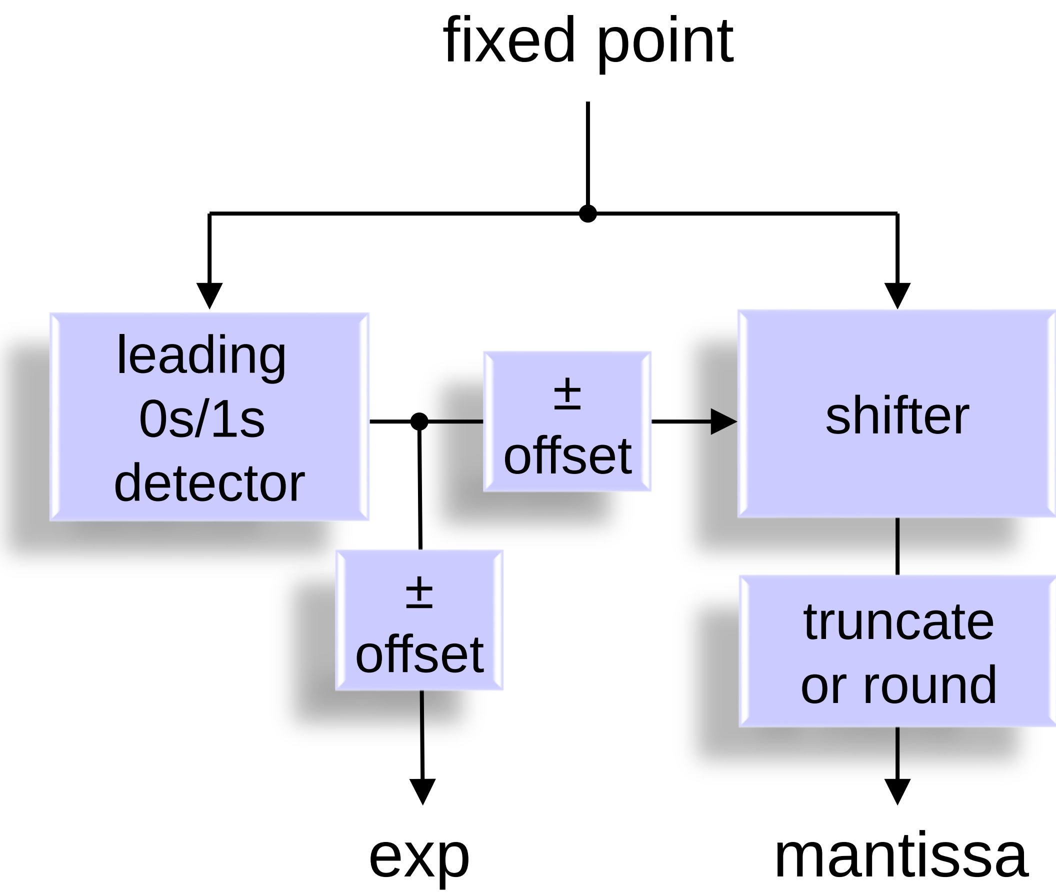
- If the *exp* is unsigned, the shifter shifts only to the left
- If the *exp* is signed, the shifter must shift to the left and right
- Example:

```
     01011. * 2^2
     01011. << 2
   000101100.
```



Fixed Point → Floating Point Conversion

- Leading 0s/1s detector finds the optimum place to begin selecting bits for the mantissa
- Common pitfall: If the *mantissa* is signed, its sign bit(s) must be maintained!



Fixed Point → Floating Point Conversion

- Fixed-to-float conversion example (*positive* input)
  - Input: 8-bit 2's complement (signed) integer
    Output: 4-bit 2's complement (signed) mantissa

a) integer mantissa

```
 s        4
        ┌───┐
0 0 0 0 1 1 0 0.   →   0 1 1 0. * 2^(001)     % 2^1
        12         =      6     *   2^1
```

b) fractional "0.4 format" mantissa

```
 s        4
        ┌───┐
0 0 0 0 1 1 0 0.   →  .0 1 1 0  * 2^(101)     % 2^(5)
        12         =    0.375   *    2^5
```

Fixed Point → Floating Point Conversion

- Fixed-to-float conversion example (*negative* input)
  - Input: 8-bit 2's complement (signed) integer
    Output: 4-bit 2's complement (signed) mantissa

a) integer mantissa

```
s   4
1 1 0 1 0 0 0 1.   →   1 0 1 0. * 2^(011)     % 2^3
      -47          =      -6    *  2^3
```

b) fractional "2.2 format" mantissa

```
s   4
1 1 0 1 0 0 0 1.   →   1 0.1 0  * 2^(101)     % 2^(5)
      -47          =      -1.5  *  2^5
```

Fixed Point → Floating Point Conversion Special Cases

- Example 1: converting a fixed-point zero

  `00000000`

  – Clearly, the selection of mantissa bits does not matter → it will be all zeros

  – But then what should the exponent be?
    - In absolute terms it does not matter
    - Choose whatever makes the hardware **more regular** and simpler

- Example 2: converting a string of 1's to FloatPt with a 4-bit mantissa

  `111111111`

  – We have at least two main approaches to selecting the mantissa bits which in general do not affect accuracy
    - 1) Choose mantissa after removing the max number of redundant sign bits
      1000. x 2^(–3) = –8 x (1/8) = –1
    - 2) Choose mantissa to preserve as many bits as possible while removing the max number of redundant sign bits
      1111. x 2^0 = –1 x 1 = –1
    - Choose whichever method makes the hardware **more regular** and simpler