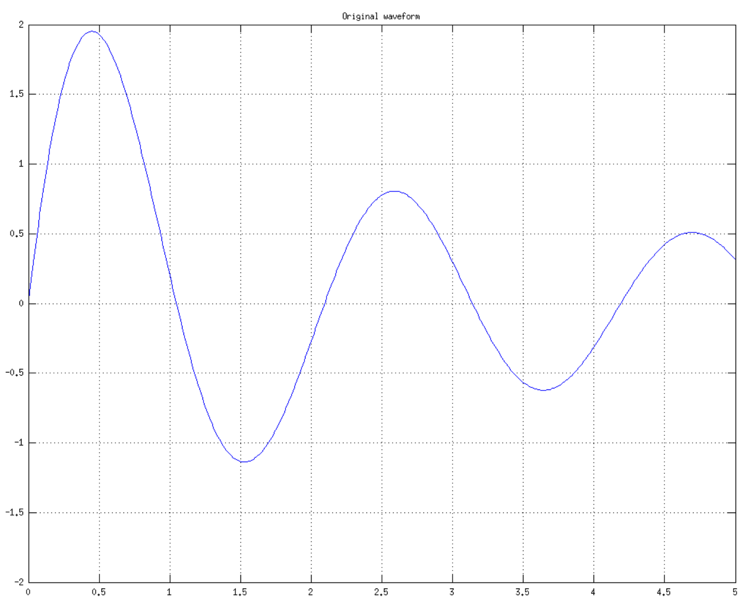
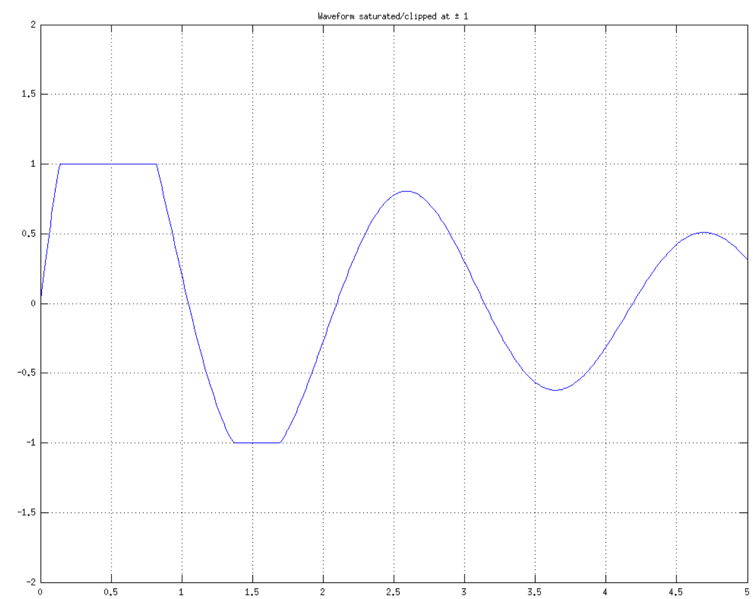
SATURATION

Saturation (or Clipping)

- Eliminates MSB bits
- It is common to saturate a signal after an operation which will or *may* cause the magnitude of a signal to increase (e.g., addition, subtraction, multiplication, (almost any operation), etc.)



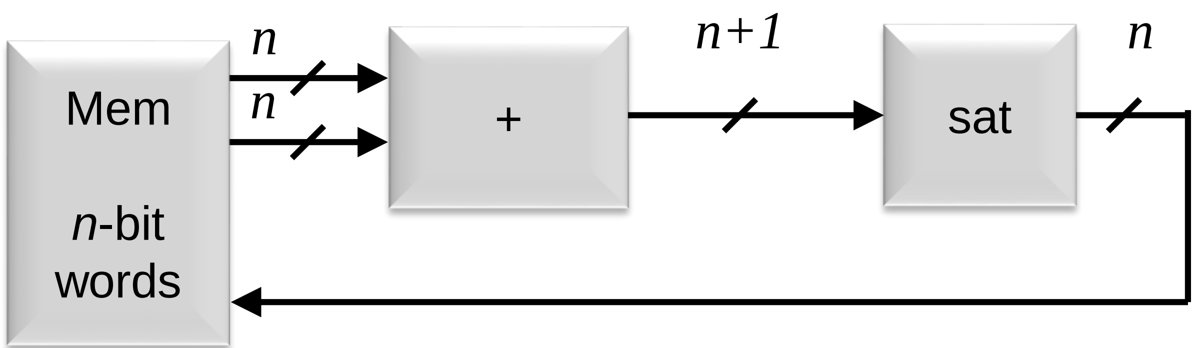
Original waveform



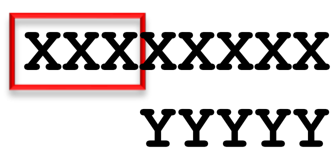
Saturated/clipped waveform

Saturation (or Clipping)

- Saturation is a fundamental method to reduce the size of a word, such as after arithmetic operations
  - For example to maintain the word width for memory storage



- When *saturated*, bits are removed from the *MSB end* of the input word



Saturation (or Clipping)

- Saturation is actually a 2-step process:
  1. Saturate the input to a maximum SAT_HI value and to a minimum SAT_LO value
  2. While it makes a lot of sense to choose SAT_HI and SAT_LO such that there are redundant MSB bits that can be dropped (shorten the word), this need not always be the case. When the saturation operation creates redundant and therefore unnecessary MSB bits, they should be dropped.
- It is often efficient to perform both steps simultaneously
- Ex: Output is saturated to a reduced-word size
  - input 4-bit 2's complement — Range is [–8, +7]
  - output 3-bit 2's complement — Range is [–4, +3]
- Ex: Output word size is not reducible
  - input 4-bit 2's complement — Range is [–8, +7]
  - output saturated to [–5, +5] — Requires 4 bits in the output word, so no word width reduction is possible

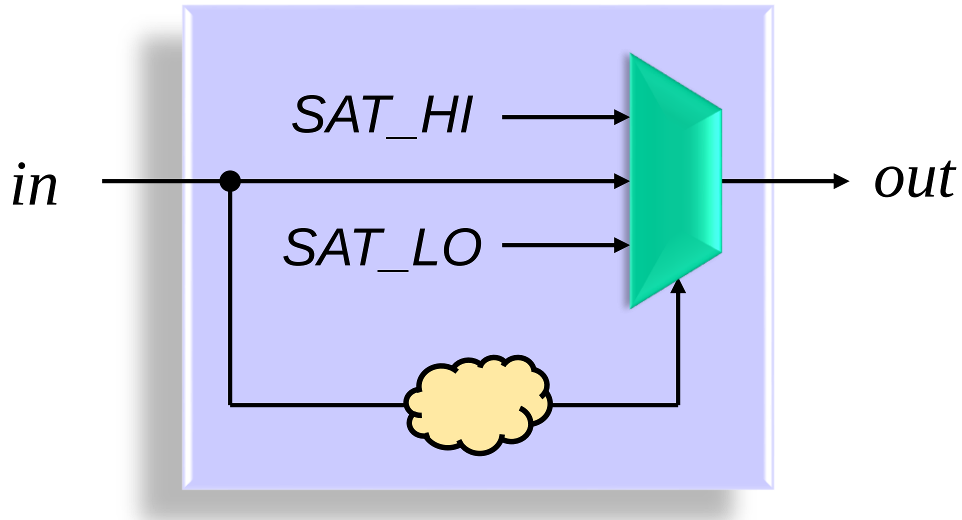
Saturation (or Clipping)

- Matlab code that produced previous example waveforms
- Copy, paste, and try it!

```
% Example saturated/clipped waveform
% 2009/02/03  Written

SatHi    =  1.0;
SatLo    = -1.0;
stepsize = 0.01;
a        = 0 : stepsize : 5;
index    = (a + stepsize) * (1/stepsize);  % matlab indexes start at 1 :-(
index    = round(index);                   % clear out VERY small offsets

b = 2.9 * sin(a*3) ./ (a+1);               % constants chosen to look nice

% plot original waveform
figure(1);clf;
plot(a,b);
grid on;
axis([0 5 -2 +2]);
title('Original waveform');

%print -dtiff 1.tiff     % 75 KB
print -dpng  1.png       % 16 KB, both look equally sharp

% saturate/clip original waveform
for l = index,
  if     b(l) > SatHi
    c(l) = SatHi;
  elseif b(l) < SatLo
    c(l) = SatLo;
  else
    c(l) = b(l);
  end
end

% plot saturated waveform
figure(2);clf;
plot(a,c);
grid on;
axis([0 5 -2 +2]);
title('Waveform saturated/clipped at \pm 1');

%print -dtiff 2.tiff     % 75 KB
print -dpng  2.png       % 16 KB, both look equally sharp
```

Saturation (Clipping)

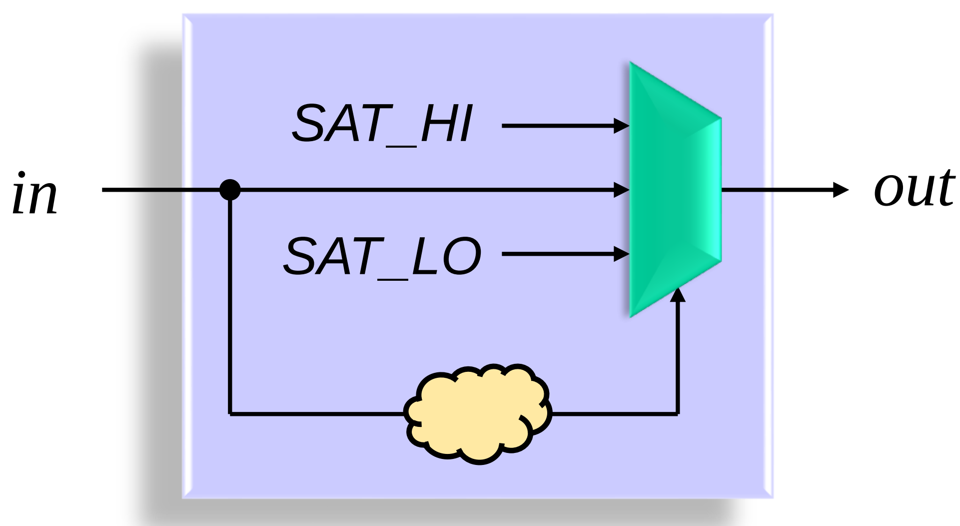
- A saturator checks for 3 possibilities:
  - *in* > SAT_HI or *in* ≥ SAT_HI
  - *in* < SAT_LO or *in* ≤ SAT_LO
  - else pass through
- Think of a saturator as a three-input mux



Saturation (Clipping)

Example: 4-bit input, ready for truncation to a 3-bit output after saturation

- A saturator checks for 3 possibilities:
  - *in* > SAT_HI or *in* ≥ SAT_HI
  - *in* < SAT_LO or *in* ≤ SAT_LO
  - else pass through
- Think of a saturator as a three-input mux



0111	SAT_HI = 011
0110	SAT_HI = 011
0101	SAT_HI = 011
0100	SAT_HI = 011
0**011**	*either sat or pass*
0010	in
0001	in
0000	in
1111	in
1110	in
1101	in
1**100**	*either sat or pass*
1011	SAT_LO = 100
1010	SAT_LO = 100
1001	SAT_LO = 100
1000	SAT_LO = 100

Saturation (Clipping)

Example: 4-bit input, ready for truncation to a 3-bit output after saturation

- If 0011 is pass through and 1100 is pass through, then the hardware can just look for when the MSB and MSB–1 bits are different.
- When the two bits are different, the MSB can not be simply dropped—the output must be saturated.
- Example verilog:

```
– if (in[MSB:MSB-1] == 2'b01)
    out = SAT_HI;
  else if (in[MSB:MSB-1] == 2'b10)
    out = SAT_LO;
  else
    out = in[MSB-1:0];
```

0111	SAT_HI = 011
0110	SAT_HI = 011
0101	SAT_HI = 011
0100	SAT_HI = 011
0**011**	*either sat or pass*
0010	in
0001	in
0000	in
1111	in
1110	in
1101	in
1**100**	*either sat or pass*
1011	SAT_LO = 100
1010	SAT_LO = 100
1001	SAT_LO = 100
1000	SAT_LO = 100

Multi-Bit Saturation (Clipping)

- The method to saturate more than one bit is similar
- To saturate *S* bits, look for when the *S*+1 MSB bits are not all the same value
- This make intuitive sense—*S* bits can not be removed unless the *S*+1 MSB bits are all identical
- Example verilog to saturate 2 MSB bits:
  -
    ```
    if (in[MSB:MSB-2] == 3'b000 || in[MSB:MSB-2] == 3'b111)
      out = in[MSB-2:0];          // pass through w/o 2 MSB bits
    else if (in[MSB] == 1'b0)     // positive
      out = SAT_HI;
    else                          // negative
      out = SAT_LO;
    ```