SATURATION & COMPRESSION
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.)
Saturation (or Clipping)

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

B. Baas, EEC 281
Saturation (or Clipping)

- Normally accompanied by a reduction in the word width
- Preserves precision of a fixed-width number representation
Saturation (Clipping)

- Basically check for 3 possibilities
  - \( in > \text{SAT}_{\text{HI}} \) or \( in \geq \text{SAT}_{\text{HI}} \)
  - \( in < \text{SAT}_{\text{LO}} \) or \( in \leq \text{SAT}_{\text{LO}} \)
  - else pass through

- Think of saturator as a three-input mux

\[
\begin{array}{c}
\text{in} \\
\text{SAT}_{\text{HI}} \\
\text{SAT}_{\text{LO}} \\
\text{mux} \\
\text{out}
\end{array}
\]

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

\[
\begin{align*}
0111 & \quad \text{SAT}_{\text{HI}} = 011 \\
0110 & \quad \text{SAT}_{\text{HI}} = 011 \\
0101 & \quad \text{SAT}_{\text{HI}} = 011 \\
0100 & \quad \text{SAT}_{\text{HI}} = 011 \\
0011 & \quad \text{either sat or pass} \\
0010 & \quad \text{in} \\
0001 & \quad \text{in} \\
0000 & \quad \text{in} \\
1111 & \quad \text{in} \\
1110 & \quad \text{in} \\
1101 & \quad \text{in} \\
1100 & \quad \text{either sat or pass} \\
1011 & \quad \text{SAT}_{\text{LO}} = 100 \\
1010 & \quad \text{SAT}_{\text{LO}} = 100 \\
1001 & \quad \text{SAT}_{\text{LO}} = 100 \\
1000 & \quad \text{SAT}_{\text{LO}} = 100
\end{align*}
\]
Saturation (Clipping)

- Look for when MSB and MSB–1 bits are different. When they are different, the MSB cannot be dropped and the output must be saturated.
  - if (in[MSB:MSB-1] == 2'b01) SAT_HI
  - if (in[MSB:MSB-1] == 2'b10) SAT_LO
  - else in[MSB-1:0]

- Similar approach to saturate more than one bit
  - To saturate S-1 bits, look for when the S MSB bits are not all the same value

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

```
0111  SAT_HI = 011
0110  SAT_HI = 011
0101  SAT_HI = 011
0100  SAT_HI = 011
0011  either sat or pass
0010  in
0001  in
0000  in
1111  in
1110  in
1101  in
1100  either sat or pass
1011  SAT_LO = 100
1010  SAT_LO = 100
1001  SAT_LO = 100
1000  SAT_LO = 100
```

B. Baas, EEC 281
Saturation Bias Effects

- Usually clip to: (+) 01111...111  +1023  +3
  (-) 10000...000  -1024  -4

- But this gives a non-zero mean

- This may cause problems
  - Very sensitive circuits; e.g., a signal path containing an accumulator
  - Worse for small-width words

- Sensitive circuits may require clipping to:
  (+) 01111...111  +1023  +3
  (-) 10000...001  -1023  -3

  - The SAT_LO comparison is now more complex: the saturation detection circuit in the critical path must now look at all bits in the input word.
**Compression: Gentle Saturation**

- In some cases, the harsh corners of a saturated waveform produce undesirable characteristics.
- A solution is to more gently saturate using a transfer function with a rounded shape that allows the signal to gently enter the saturation region.
- Commonly called “Compression”.
- Negative aspects:
  - The signal is distorted earlier (at a smaller magnitude).
  - The hardware is much more complex.
Compression

- This is the transfer function of a compression function using \( \sin() \) for the rounded sections.
- See the next slide for the matlab.
Compresssion

• This is the matlab code that generates the plots on the previous slide
• Copy, paste, and try it yourself!

```matlab
% compressor.m
%
% This module compresses the values [-64:63] (which are the same values representable by a 7-bit 2's complement number) into an output range of [-31:+31] which almost fills the range of a 6-bit 2's complement number.

% Outputs are compressed according to a piecewise function comprised of partial sin waveforms and saturated regions. They are not quantized or rounded in any other way.

% 2015/03/05 Cleaned up and documented better

PrintOn = 1;   % set to 1 to print tiff and png of figure(1)
XVals = [-64:63];   % range of a 7-bit 2's complement number
XOffset = (1 - XVals(1)); % offset needed for array index to begin at 1
Xsat = 48;   % +/- x value where full saturation begins
SatVal = 31;   % both neg and pos. max +/- range of 6-bit 2's compl

for l = XVals,
    if     l < -Xsat
        b(l+XOffset) = -SatVal;
    elseif l < 0
        b(l+XOffset) = SatVal * sin((l/Xsat)*(pi/2));
    elseif l < Xsat
        b(l+XOffset) = SatVal * sin((l/Xsat)*(pi/2));
    else
        b(l+XOffset) = +SatVal;
    end
end

figure(1);clf;
plot(XVals, b);
hold on;
grid on;
%plot([-24:24], [-24:24]*(pi/2), 'r');   % not slope=+1 line
xlabel('Input value');
ylabel('Output value');
title('Compressor function');
if (PrintOn) print -dtiff compressor.tiff; end
if (PrintOn) print -dpng compressor.png; end
```