wavefront SEMICONDUCTOR

We make the parts that set creative people free™

1KM AN3101-05 DSP

Application Note AN3101-05: Compressor and Expander for 1KM

by Shultz Wang

This application note provides the algorithm for a basic compression/expansion curve.

In log/log space, a straight line is described by the equation $y=Ax^B$. By taking the log of the equation, its behavior may be examined:

$$\log(y) = \log(A) + B \log(x).$$

It can be seen that log(A) controls the y-intercept of the line, and B controls the slope of the line.

This equation will now be used to construct the complete compander curve. The curve is made up of three segments: the upper compression segment $y=A_c x^{B_c}$, the middle passthrou segment y=x, and the lower expansion segment $y=A_e x^{B_e}$.

For compression, a decrease in slope is desired for higher ratios, so choose $B_c=1/R_c$. To solve for A_c, note that at the threshold the two lines meet, so

$$y(T_c) = T_c = A_c T_c^{(1/R_c)} \Rightarrow A_c = T_c^{(1-1/R_c)}.$$

Therefore

$$y = T_c^{(1-1/R_c)} x^{(1/R_c)}.$$

This is equal to the input times the compression gain G_c, therefore the gain is

$$G_c x = T_c^{(1-1/R_c)} x^{(1/R_c)},$$

$$G_c = (T_c/x)^{(1-1/R_c)}.$$

After a similar derivation, the expansion gain G_e can be derived to be

$$G_e = (T_e/x)^{(1-R_e)}.$$

The basic smoothing algorithm, applied to the calculated gain coefficients generates the attack and release times.

$$\text{Current} = \text{Current} + \tau\,(\text{Target} - \text{Current}).$$

The following table can serve as a guide for selecting the proper fractional value for good attack and release time constants (sampling period = 1/48kHz). Common attack times are 1ms to 100ms. Common release times are 0.5s to 3s.

Shift	Fractional	Hex	τ = -(Sample Pd)/ln(1-Fractional)
1 bit	1/2	$020000	0.030ms
2 bits	1/4	$010000	0.072ms
3 bits	1/8	$008000	0.156ms
4 bits	1/16	$004000	0.323ms
5 bits	1/32	$002000	0.656ms
6 bits	1/64	$001000	1.323ms
7 bits	1/128	$000800	2.656ms
8 bits	1/256	$000400	5.323ms
9 bits	1/512	$000200	10.656ms
10 bits	1/1024	$000100	21.323ms
11 bits	1/2048	$000080	42.656ms
12 bits	1/4096	$000040	85.323ms
13 bits	1/8192	$000020	0.170s
14 bits	1/16384	$000010	0.341s
15 bits	1/32768	$000008	0.682s
16 bits	1/65536	$000004	1.365s
17 bits	1/131072	$000002	2.731s
18 bits	1/262144	$000001	5.461s

Alesis Semiconductor
12555 Jefferson Blvd., Suite 285
Los Angeles, CA 90066
Phone (310) 301-0780 Fax (310) 306-1551 www.alesis-semi.com

1K code:

First calculate or select the following values for each channel (the x subscript in the names means it refers to both compression (subscript c) and expansion (subscript e)).

- T_x (Threshold)
- G_x (Gain) Address (non-rotating)
- R_x (Ratio)
- τ_{xa} (Attack fractional)
- τ_{xr} (Release fractional)
- G (Final gain)

The compression algorithm is

```
 ; Threshold detection
CM     $080000      $41#   ; Read Input channel # into A, scaled to +1 → -1
SKIP   !N           $1     ; Skip multiply by -1 if data positive
CM     $380000      $41#   ; Read -Input into A, scaled to +1 → -1
X1AC   $(-Tc)              ; Store |Input| into B, subtract Tc from |Input|
SKIP   N            $9     ; Skip Gc attack calculations if |Input| < Tc
 ; Attack conditional
LOGB                       ; LOG16(|Input|)
DAC    $f00         $(LOG16(Tc)) ; LOG16(Tc)-LOG16(|Input|)
CAD    $(1-1/Rc)    $0     ; (1-1/Rc)(LOG16(Tc)-LOG16(|Input|))
X1AC   $0                  ; A->B
EXPB                       ; EXP16((1-1/Rc)(LOG16(Tc)-LOG16(|Input|)))
CMA    $3c0000      $GcA   ; (-1)Gc+EXP16((1-1/Rc)(LOG16(Tc)-LOG16(|Input|)))
CAM    $(τca)       $GcA   ; Gc+τca(EXP16((1-1/Rc)(LOG16(Tc)-LOG16(|Input|)))-Gc)
SXCA   $0           $GcA   ; Write new attack Gc, store Gc in B.
SKIP                $3     ; Skip Gc release calculations since |Input| ≥ Tc
 ; Release conditional
1MC    $3c0000      $GcA   ; (Gc-1)
CAM    $(-τcr)      $GcA   ; Gc-τcr(Gc-1) = Gc+τcr(1-Gc)
SXCA   $0           $GcA   ; Write new release Gc, store Gc in B.
 ; Gc in A and B
```

For expansion, replace all compression constants with expansion constants, replace $(1-1/R_c)$ with $(1-R_e)$, replace the G_c address with the G_e address, and replace the second skip instruction's condition with !N.

```
SKIP   !N           $8     ; Skip Ge attack calculations if |Input| > Te
```

Multiply the input by the gains and write to output.

```
CM     $(G)         $GcA   ; Read Gc, multiplied by final gain
AMC    $0           $GeA   ; Gc*Ge*G
AMC    $0           $41#   ; Input=Input*Gc*Ge*G
SCA    $0           $41#   ; Write companded input to chan #
```

Note that gain reduction metering may be done by reading the gain value and subtracting it from unity. To output the metering to a channel, use the following code.

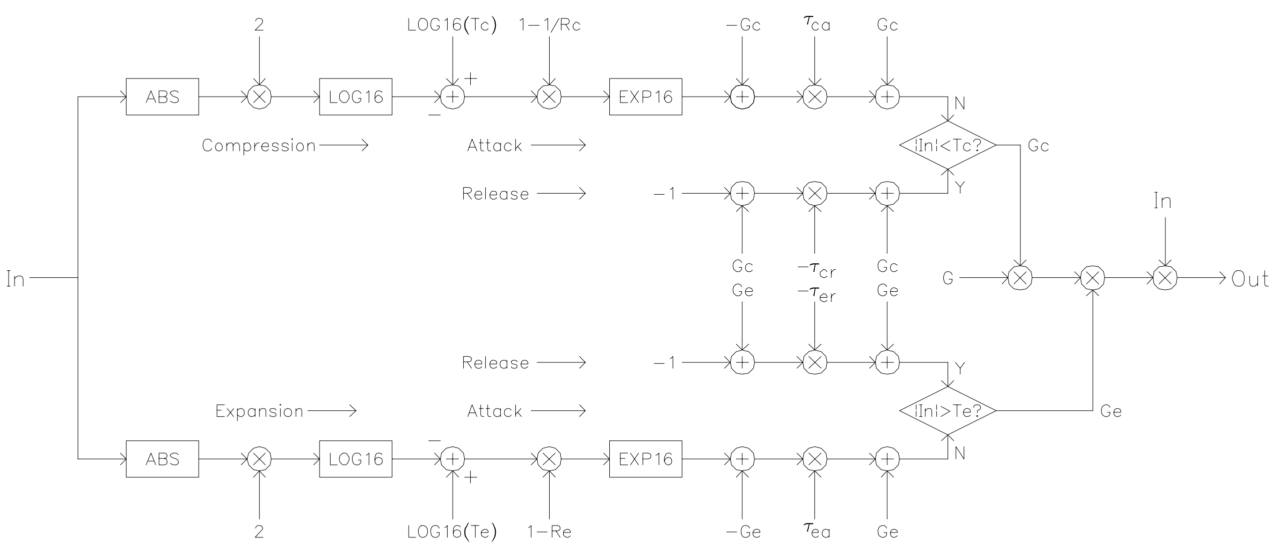
```
CM     $3c0000      $GxA   ; Read -Gx
1AC    $040000             ; 1-Gx
SCA    $0           $4##   ; Write gain reduction meter value to address ##
```


Alesis Semiconductor
12555 Jefferson Blvd., Suite 285
Los Angeles, CA 90066
Phone (310) 301-0780 Fax (310) 306-1551 www.alesis-semi.com

Note that since compression and expansion each need a register to store the current gain, to have both on every channel will require 16 registers.

Note that in this implementation only the threshold parameter requires two instruction alterations, all others require only one.

Compression/Expansion Algorithm Block Diagram



Sample compander:

; Compressor: Threshold = -6dB; Ratio = 4

; Attack τ = 0.323ms; Release τ = 0.682s

```
CM     $080000     $410   ; Read Input channel 0 into A, scaled to +1 → -1
SKIP   !N          $1     ; Skip multiply by -1 if data positive
CM     $380000     $410   ; Read -Input into A, scaled to +1 → -1
X1AC   $f800000           ; Store |Input| into B, subtract Tc from |Input|
SKIP   N           $9     ; Skip Gc attack calculations if |Input| < Tc
LOGB                      ; LOG16(|Input|)
DAC    $f00        $3f0000     ; LOG16(Tc)-LOG16(|Input|)
CAD    $030000     $0     ; (1-1/Rc)(LOG16(Tc)-LOG16(|Input|))
X1AC   $0                 ; A->B
EXPB                      ; EXP16((1-1/Rc)(LOG16(Tc)-LOG16(|Input|)))
CMA    $3c0000     $40f   ; (-1)Gc+EXP16((1-1/Rc)(LOG16(Tc)-LOG16(|Input|)))
CAM    $004000     $40f   ; Gc+τea(EXP16((1-1/Rc)(LOG16(Tc)-LOG16(|Input|)))-Gc)
SCA    $0          $40f   ; Write new attack Gc
SKIP               $3     ; Skip Gc release calculations since |Input| ≥ Tc
1MC    $3c0000     $40f   ; (Gc-1)
CAM    $3ffff8     $40f   ; Gc-τcr(Gc-1) = Gc+τcr(1-Gc)
SCA    $040000     $40f   ; Write new release Gc.
```


Alesis Semiconductor
12555 Jefferson Blvd., Suite 285
Los Angeles, CA 90066
Phone (310) 301-0780 Fax (310) 306-1551 www.alesis-semi.com

```
 ; Expander: Threshold = -72dB; Ratio = 4
 ; Attack τ = 0.323ms; Release τ = 0.682s
CM     $080000     $410  ; Read Input channel 0 into A, scaled to +1 → -1
SKIP   !N          $1    ; Skip multiply by -1 if data positive
CM     $380000     $410  ; Read -Input into A, scaled to +1 → -1
X1AC   $ffff000          ; Store |Input| into B, subtract Te from |Input|
SKIP   !N          $9    ; Skip Ge attack calculations if |Input| > Te
LOGB                     ; LOG16(|Input|)
DAC    $f00        $340000     ; LOG16(Te)-LOG16(|Input|)
CAD    $340000     $0    ; (1-Re)(LOG16(Te)-LOG16(|Input|))
X1AC   $0                ; A->B
EXPB                     ; EXP16((1-Re)(LOG16(Te)-LOG16(|Input|)))
CMA    $3c0000     $40e  ; (-1)Ge+EXP16((1-Re)(LOG16(Te)-LOG16(|Input|)))
CAM    $004000     $40e  ; Ge+τea(EXP16((1-Re)(LOG16(Te)-LOG16(|Input|)))-Ge)
SCA    $0          $40e  ; Write new attack Ge
SKIP               $3    ; Skip Ge release calculations since |Input| ≤ Te
1MC    $3c0000     $40e  ; (Ge-1)
CAM    $3ffff8     $40e  ; Ge-τer(Ge-1) = Ge+τer(1-Ge)
SCA    $040000     $40e  ; Write new release Ge.
 ; Multiply input by gains; Final gain = 1
CM     $040000     $40f  ; Read Gc, multiplied by final gain
AMC    $0          $40e  ; Ge*Gc*G
AMC    $0          $410  ; Input = Input*Ge*Gc*G
SCA    $0          $410  ; Write chan 0
```


Alesis Semiconductor
12555 Jefferson Blvd., Suite 285
Los Angeles, CA 90066
Phone (310) 301-0780   Fax (310) 306-1551   www.alesis-semi.com