

**ESPRIT PROJECT
EP29644**

Video Decoder Platform

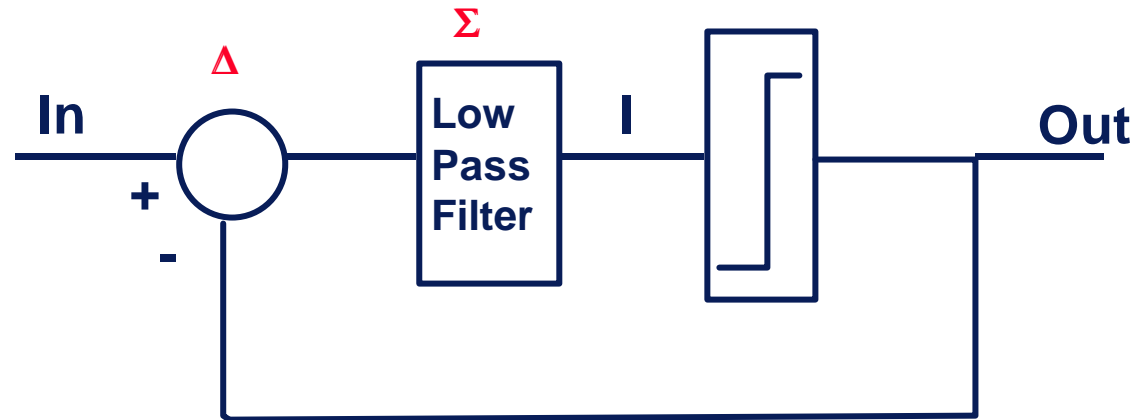
**Sigma Delta Couse
Part I**

March 2000

COURSE SCOPE

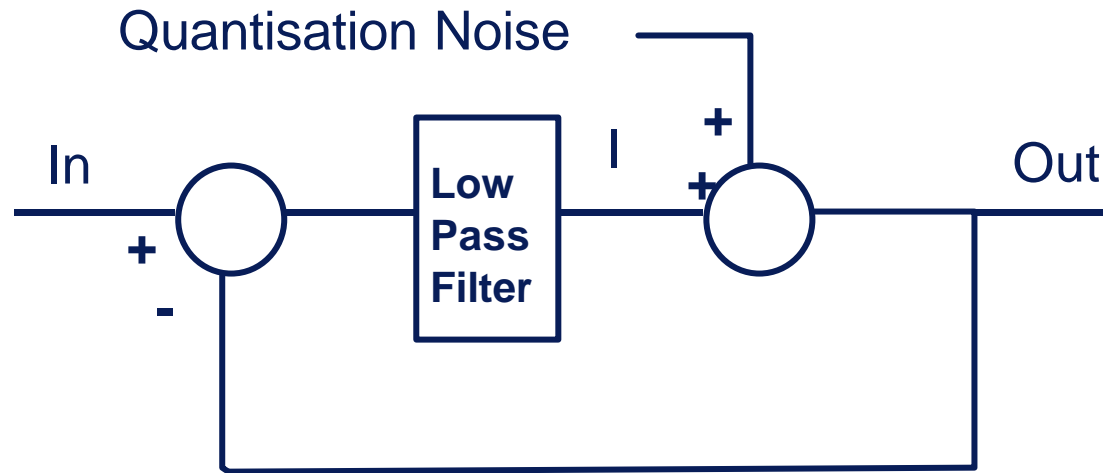
- Basic concepts of $\Sigma\Delta$ modulators, leaning towards ADCs
 - a modulators course, not a circuits course
- Filters for ADCs
- DACs
- Mostly consists of exercises in 'C'

Basic $\Sigma\Delta$ Structure

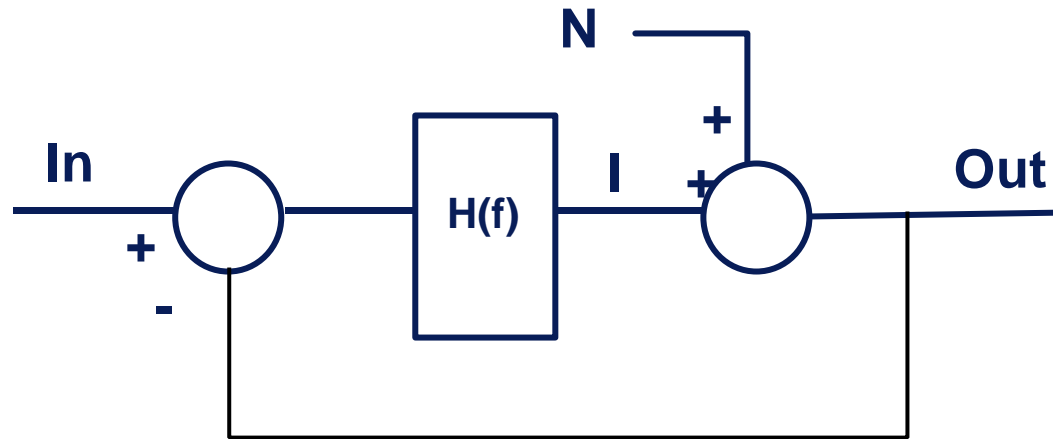


- Input is high resolution, unknown or many bits
- Output (and feedback) is quantised, often just 1 bit
- Output is preferred for some reason - linearity or simplicity. 1 bit feedback can be both.
- If the loop is stable, **output matches input, within passband**

'Linear' Model



- Think of it as a control system with negative feedback
- Quantisation noise is error between input and output of quantiser
 - its the 'designed-in error'
- The low pass filter need not be ideal
 - but should have high gain in the pass band



$$I = H(f) \cdot [In - Out]$$

$$Out = I + N$$

$$\Rightarrow Out = N + H(f) [In - Out]$$

$$\Rightarrow Out [1 + H(f)] = N + H(f) In$$

$$\Rightarrow Out = \frac{In \cdot H(f)}{1 + H(f)} + \frac{N}{1 + H(f)}$$

Input Transfer Function

$$\frac{H(f)}{1 + H(f)}$$

Noise Transfer Function

$$\frac{1}{1 + H(f)}$$

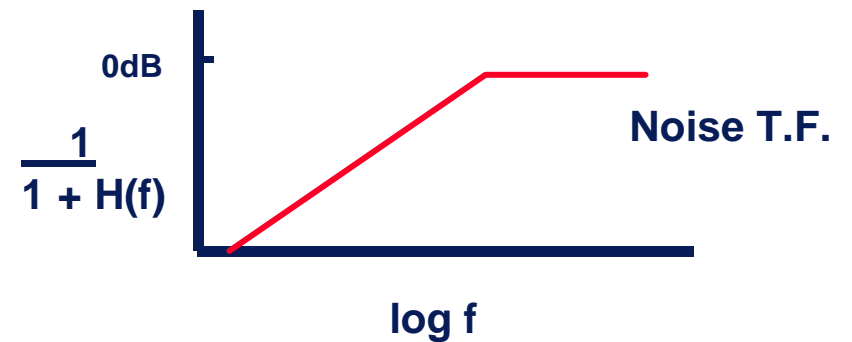
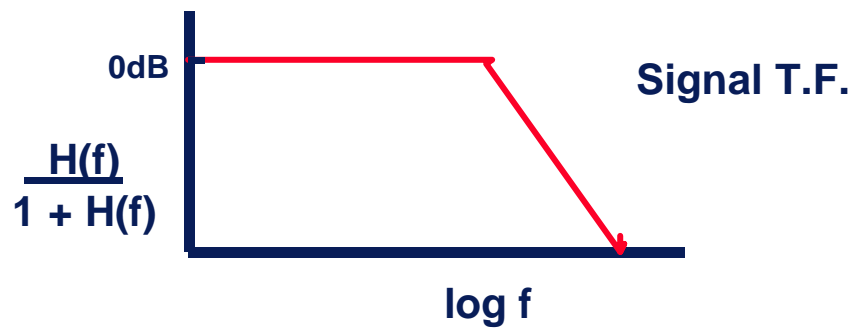
In the simplest cases, the loop filter response $H(f)$ is low pass, with very high gain at low frequency. Typical filter examples are a simple integrator or a cascade of integrators. Note that noise is 'shaped' by $1/[1 + H(f)]$ function.

Input Transfer Function

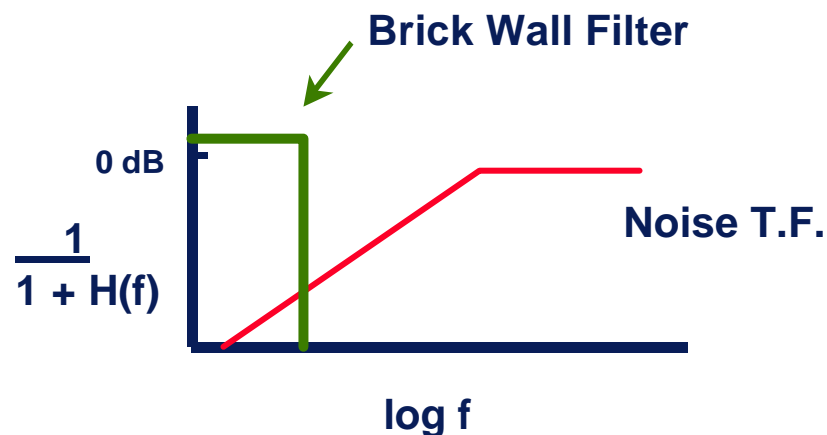
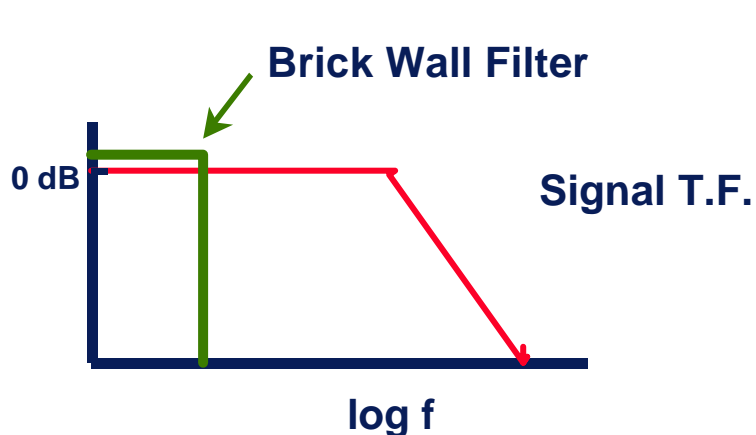
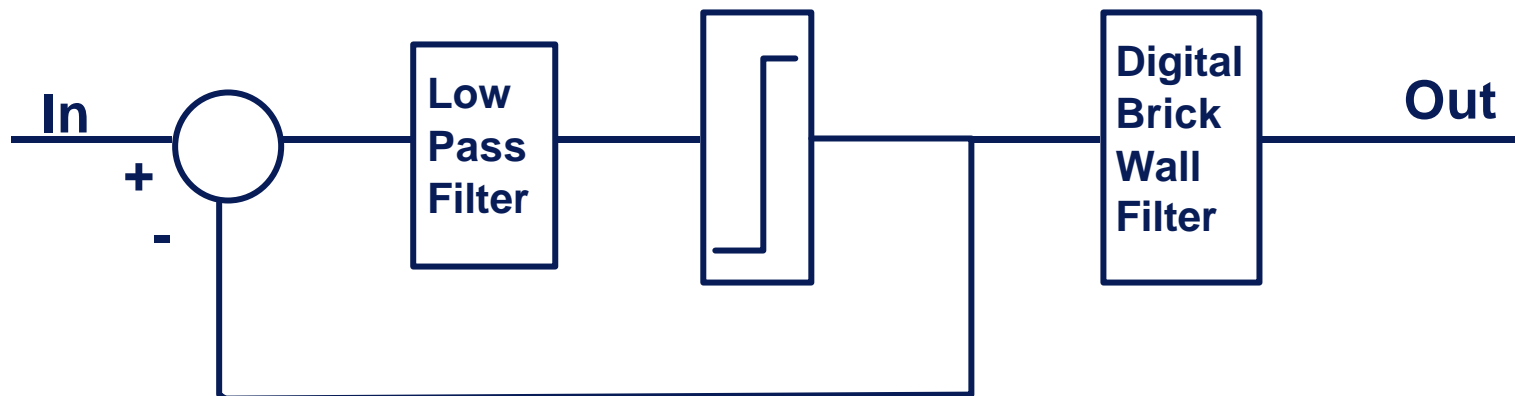
$$H(f) / [1 + H(f)]$$

Noise Transfer Function

$$1 / [1 + H(f)]$$

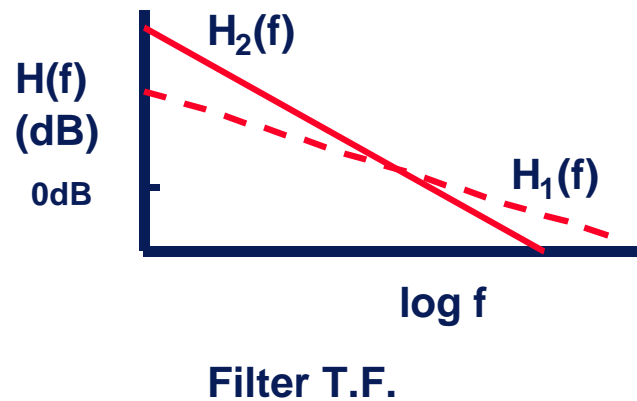


Data Conversion

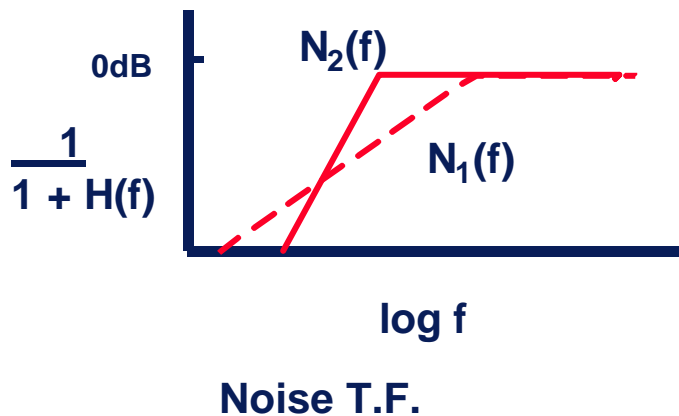


Applying a brick wall filter to the modulator output removes the quantisation noise, but retains the input signal. $\Sigma\Delta$ modulators are 'oversampled' in the sense that the clock rate is far higher than the signal bandwidth. Nyquist requires a sample clock rate of at least twice signal bandwidth. In a $\Sigma\Delta$, the ratio of clock rate to twice signal bandwidth (called the oversampling rate) is typically 32 - 256. A higher oversampling rate increases SNR.

Effect of Filter Order



One integrator gives $H_1(f)$ with slope of -20dB/decade . Two integrators in cascade gives $H_2(f)$ with a slope of -40dB/decade .

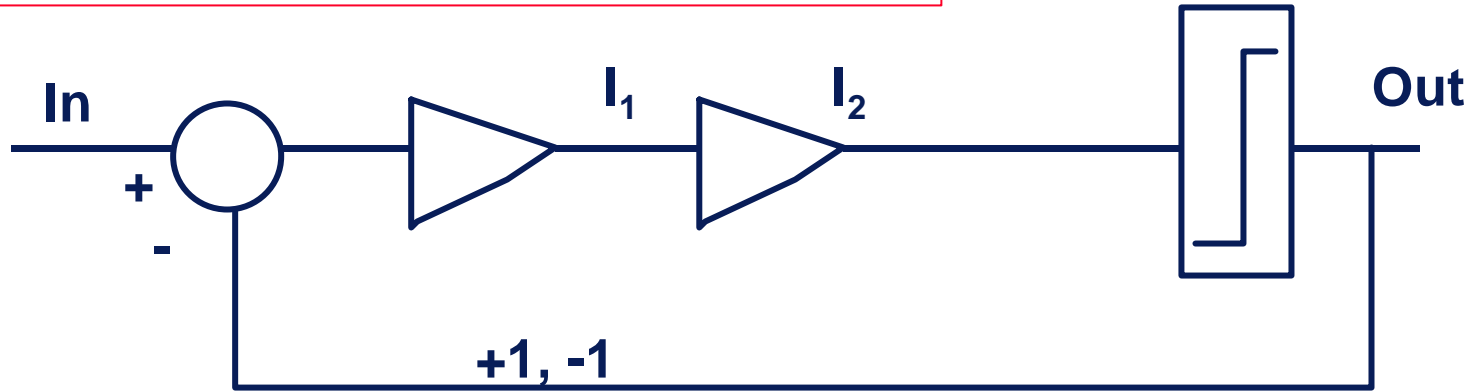


Corresponding noise transfer functions are $N_1(f)$ with 20dB/decade noise slope and $N_2(f)$ with 40dB/decade noise slope. $N_2(f)$ has much less low frequency noise. The cross-over is typically around $\text{clock}/16$.

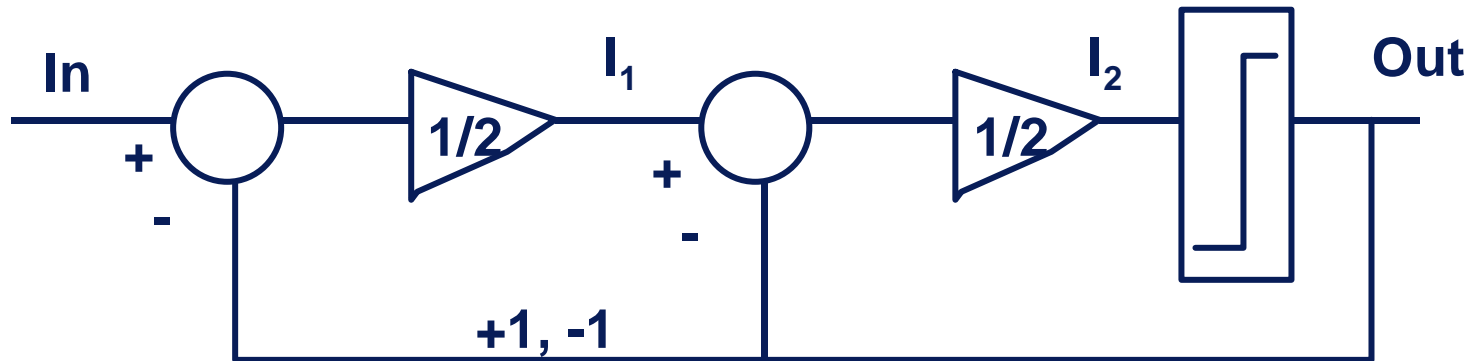
Higher Order $\Sigma\Delta$ Modulators

- First order $\Sigma\Delta$ modulators have similar net performance to other known converters such as delta modulators, integrating ADCs, voltage to frequency converters etc.
- First order $\Sigma\Delta$ modulators, coupled with first order filters, give a resolution of $1/N$ where N is the number of filter coefficients (or the over-sampling ratio). Thus for 16 bit accuracy an oversampling ratio of 65536 is required.
- Higher $\Sigma\Delta$ modulators greatly reduce the required oversampling ratio. A 2nd order modulator needs an oversampling ratio of about 256 for 16 bit accuracy.
 $\Sigma\Delta$ modulators became really interesting with the introduction of the first 2nd order modulator, by J.C. Candy in the early 1980s.

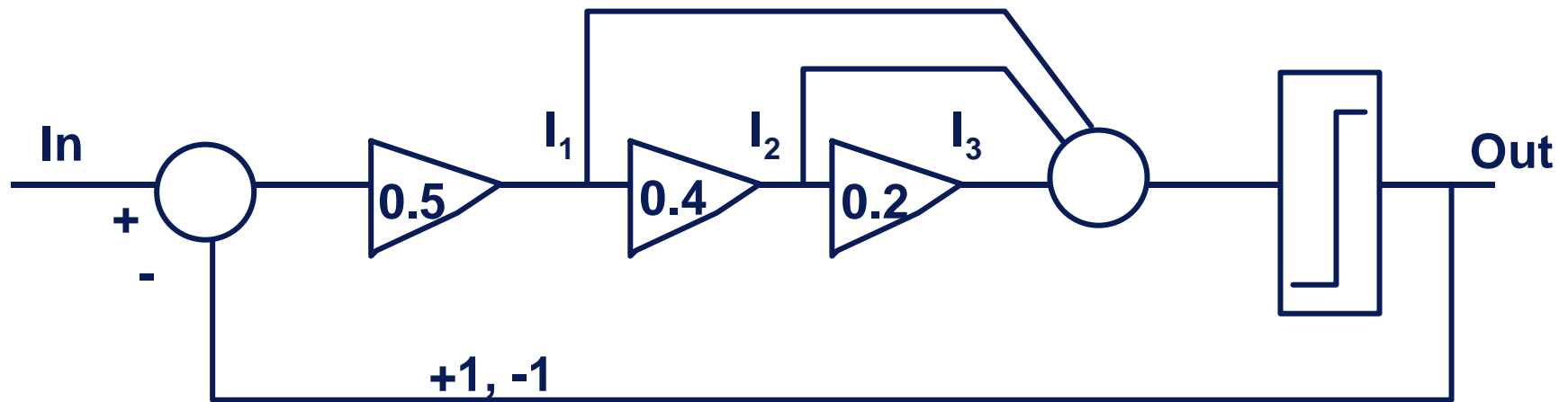
This 2nd order modulator will go unstable.



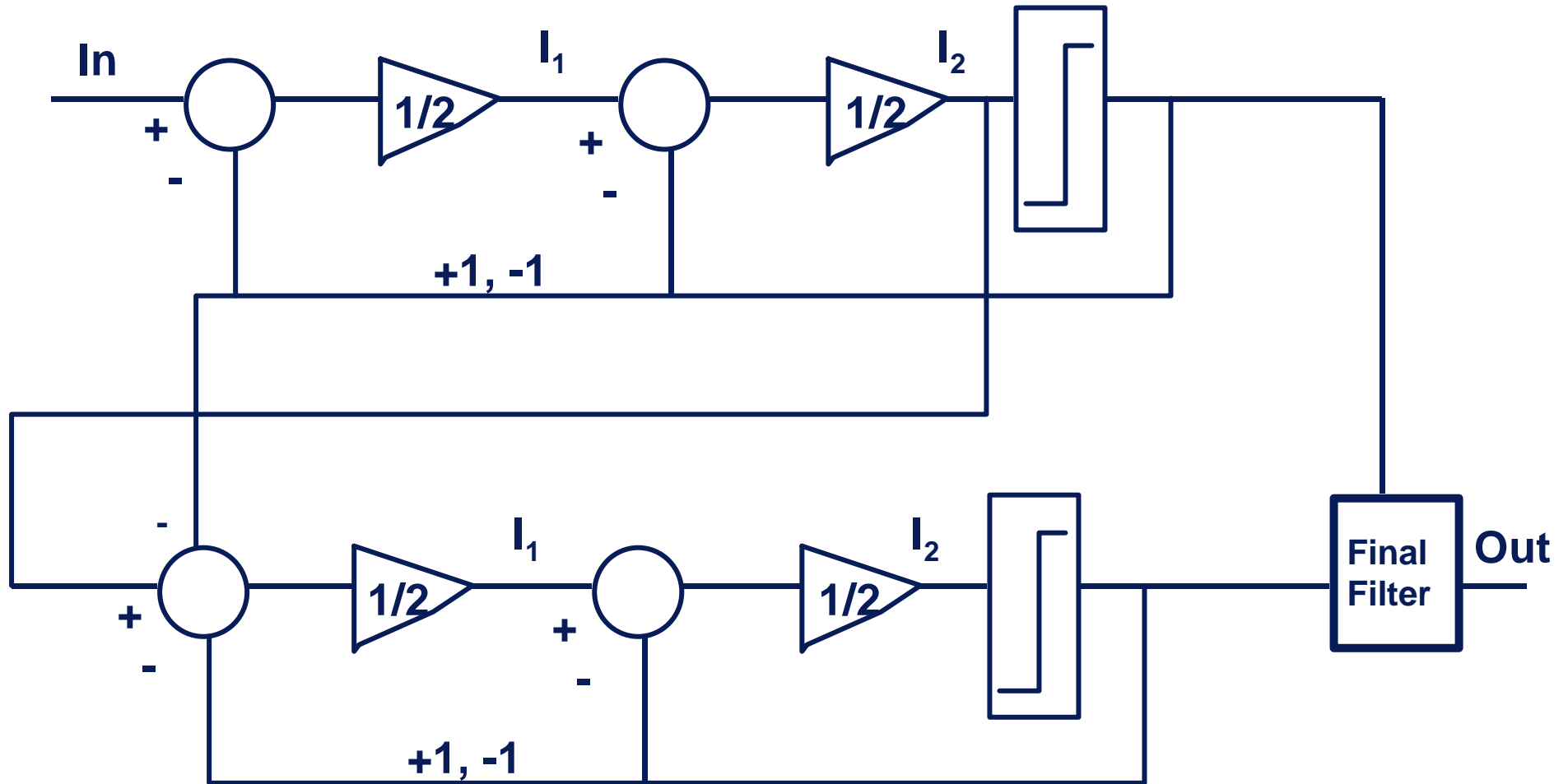
This 2nd order modulator, due to Candy, is stable.



This is a 3rd order modulator. It is not guaranteed stable.

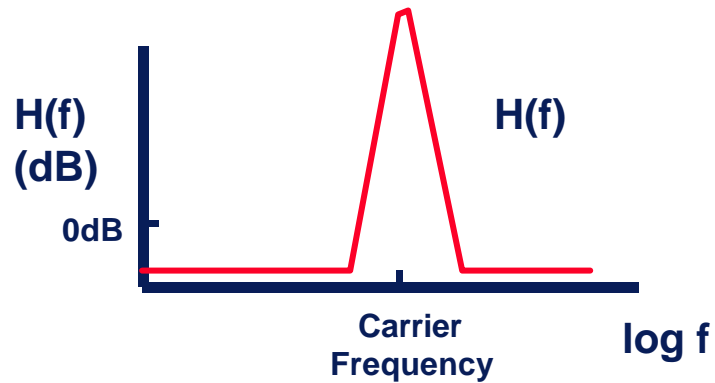


MASH

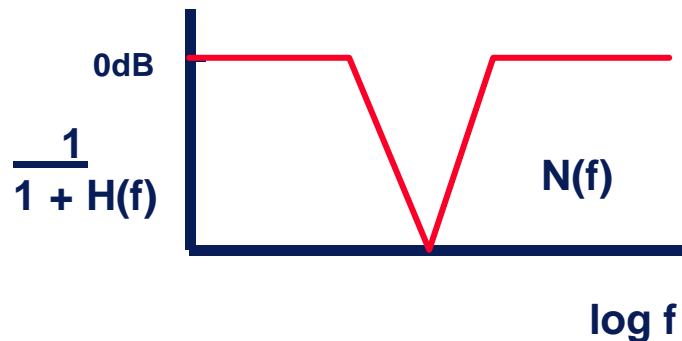


This 'MASH' modulator has a filter order of 4, since it is a cascade of two 2nd order modulators. It is stable since each 2nd order loop is stable. However, this system has 2 feedback bits and so will be susceptible to analog mismatch problems.

Bandpass $\Sigma\Delta$ for Communications

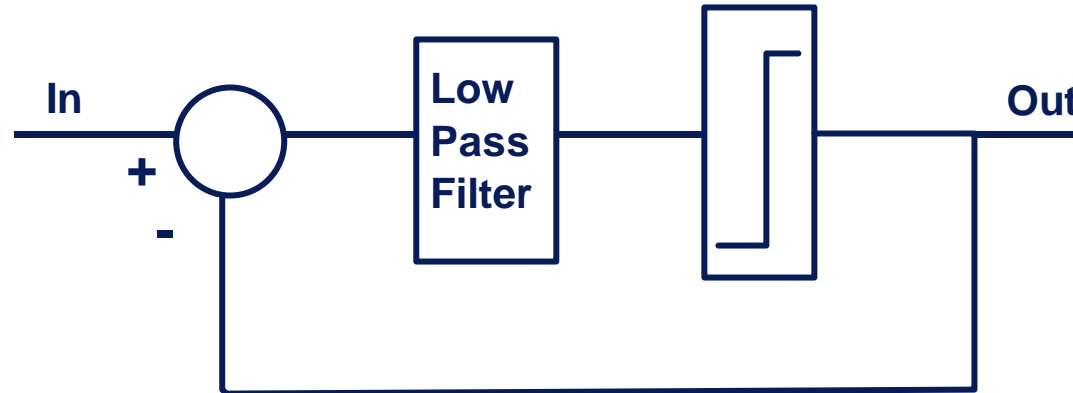


Placing the high gain region of the filter response at intermediate frequencies allows bandpass conversion (at the chosen intermediate frequency). Usually this frequency is $f_s/4$ or some other simple submultiple of f_s .



Corresponding noise transfer function has very low gain at the bandpass frequency. So signals around this frequency can be easily detected.

Modulator Loop Dynamics



- The modulator loop stability is very hard to analyse.
- The comparator/quantiser is inherently non-linear, so it has no Z-domain equivalent.
- First and second order loops are known to be stable, if correct integrator gains and ratio of input to feedback signal size is maintained.
- Low frequency idle channel tones arise in modulator loops of order 1 or 2
 - digital loops are worse
 - higher order loops are best
- Higher order loops are at best conditionally stable, with single bit feedback
- More quantiser levels make the modulator like a linear system. More than 2 quantiser levels requires very accurate matching. This will become very important.

Summary

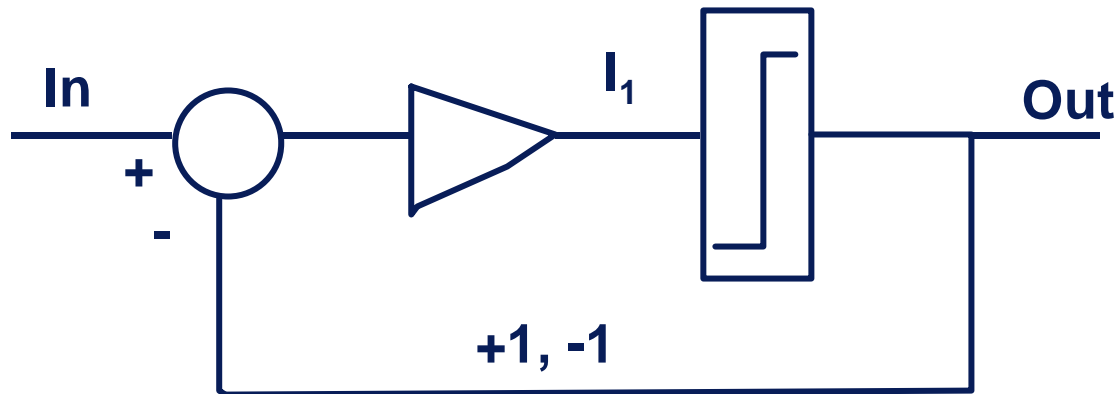
- Purpose is to drive a low resolution output so that it matches a high resolution input, within some specified, restricted, pass band
 - typically single bit output
 - low resolution signal preferred for accuracy or convenience
 - oversampling ratio is high - often 256
- $\Delta\Sigma$ modulators consist of a negative feedback loop with an internal filter that has very high gain in the pass band and very low gain elsewhere
-
- Big advantages if the loop filter is second order or higher

SIGMA DELTA SIMULATION EXERCISES

C. Lyden

Analog Devices May 2000

EXERCISE 1: First Order $\Sigma\Delta$



This system is described by the equations:

$$I_1^n = I_1^{n-1} + In - Out^{n-1}$$

$$Out^n = I_1^n > 0 ? 1 : -1$$

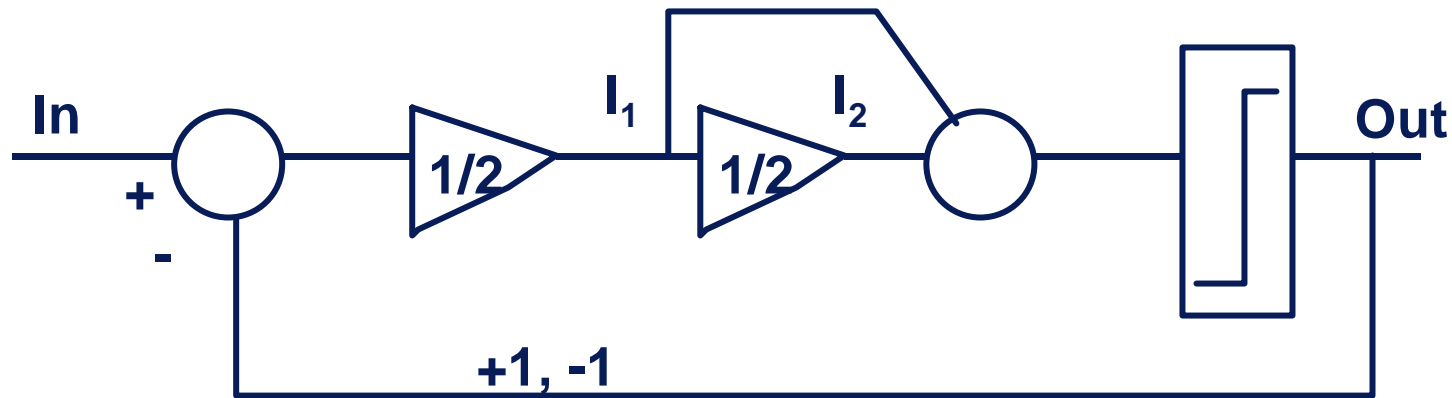
Set the initial value of I_1 to 0 and the initial value of Out to -1.

1. Simulate 100 clock cycles. Show that for dc inputs of +0.50103, 0.0101, -0.50301, the output is pulse density modulated.
2. Simulate for 10000 clock cycles with inputs of 0.5, 0, -0.5. What is the output?
3. Set the equivalent clock rate to 10 MHz. Apply a 20 kHz input signal, with amplitude of 0.1. Run for 32768 (32 k) clock cycles. FFT the output. Identify the input tone and the noise. Show that the 'slope' of the output is 20dB/decade (use a log scale for frequency). Why do sigma delta people claim 30 dB/decade (i.e. 1.5 bits per doubling) noise reduction?
4. Average the output over 100 clock cycles. Show that a sinewave results. FFT this, and compare to the FFT in 3.

EXERCISE 1 (Continued)

5. The 'quantisation noise' in this system is the difference between the output and the input of the comparator. Plot the quantisation noise in the time and frequency domains. Do you see a correlation between the spectrum of the quantisation noise and of the modulator output?
6. Change the feedback values (from +1, -1) to +1.25, -1.1. Run for 32 k cycles and FFT the output. Compare the input tone frequency and harmonics with that of 3 above. Note that these errors in the feedback have not introduced any signal distortion.
7. Change the negative feedback value to -0.2. Run 32 k clock cycles. What has gone wrong?
8. Reset the feedback values to +1, -1. Add a 50 kHz tone, of amplitude 0.1, to the negative feedback value. Run for 32 k clock cycles and FFT the output. Look for the error introduced in the result.

EXERCISE 2: Second Order $\Sigma\Delta$

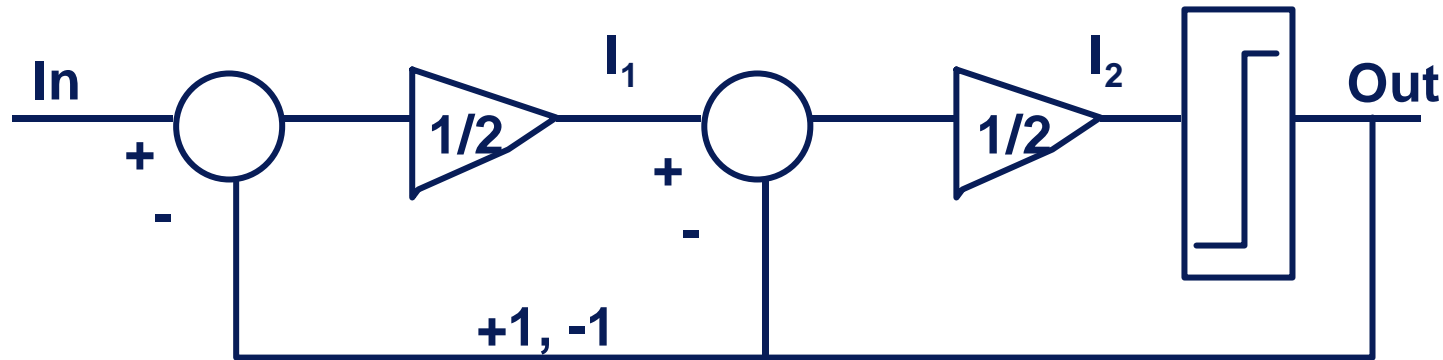


This system is described by the equations:

$$\begin{aligned} I_1^n &= I_1^{n-1} + (I_n - \text{Out}) / 2 \\ I_2^n &= I_2^{n-1} + I_1^{n-1} / 2 \\ \text{Out} &= I_1 + I_2 > 0 ? 1 : -1 \end{aligned}$$

1. Set the equivalent clock rate to 10 MHz. Apply a 20kHz input signal, with amplitude of 0.1. Run for 32768 (32k) clock cycles. FFT the output. Identify the input tone and the noise. Show that the slope of the output is 40dB/decade. How does the low frequency noise (say 0 to 10kHz) compare to that for the first order modulator? How does the signal power compare?
2. Estimate SNR in the first 22kHz for this and the first order modulator.
3. FFT the quantisation error introduced by the comparator. Compare it with the modulator output FFT.
4. Plot histograms of the quantiser input and of quantisation error. Do they look gaussian?

EXERCISE 2 (Continued)



This modulator, due to Candy, is described by the equations:

$$\begin{aligned} I_1^n &= I_1^{n-1} + (I_n - \text{Out}) / 2 \\ I_2^n &= I_2^{n-1} + (I_1^{n-1} - \text{Out}) / 2 \\ \text{Out} &= I_2 > 0 ? 1 : -1 \end{aligned}$$

4. Set the equivalent clock rate to 10 MHz. Apply a 20 kHz input signal, with amplitude of 0.1. Run for 32768 (32k) clock cycles. FFT the output. Identify the input tone and the noise. Show that the slope of the output is 40dB/decade.
5. FFT the quantisation error introduced by the comparator. Compare it with the modulator output FFT.
6. Spot the difference in performance between this and the other 2nd order modulator

EXERCISE 2 (Continued Again)

7. In switched capacitor $\Sigma\Delta$ analog to digital converters, the integrators will usually have a transfer function of

$$\frac{Z^{-1}}{1 - Z^{-1}}$$

Show that the transfer function (input to output) of the Candy modulator implemented using such integrators is

$$Z^{-2} / 4$$

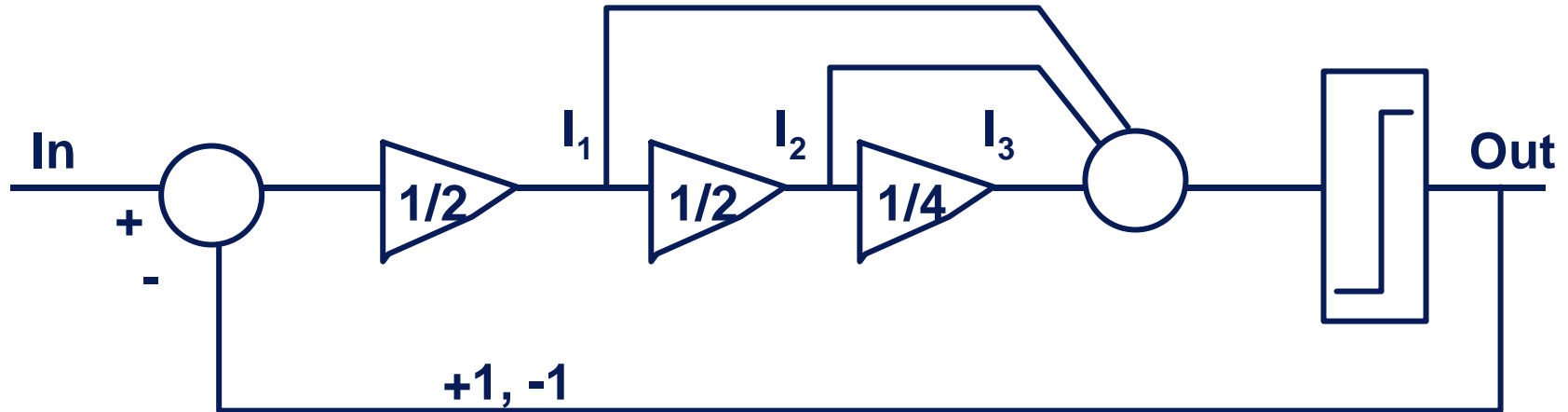
8. What is the transfer function of the previous modulator?

9. What is the effect (in the output bitstream) of changing the gain of the second integrator to 1 in the Candy modulator?

What is the effect of changing the gain of the first integrator?

Do the same for the previous, 'feedforward' modulator.

EXERCISE 3: 3rd Order $\Sigma\Delta$



This system is described by the equations:

$$I_1^n = I_1^{n-1} + (In - Out) / 2$$

$$I_2^n = I_2^{n-1} + I_1^{n-1} / 2$$

$$I_3^n = I_3^{n-1} + I_2^{n-1} / 4$$

$$Out = I_1 + I_2 + I_3 > 0 ? 1 : -1$$

1. Find the largest 20 kHz sinewave for which this loop is stable.
2. Change the integrator gains to 1/2, 1/2.5, 1/5, respectively and redo 1.
3. FFT the output. Identify the input tone and quantisation noise. Verify that the noise slope is 60dB/decade.
4. Increase the feedback resolution from 1bit to 3bits. Redo 1. FFT the output. See if the quantisation noise power is reduced.

EXERCISE 3: Continued

5. Introduce an error of 0.05 in the 3rd feedback level. Does this error show in the output FFT?
6. Change the gain of the 3rd integrator to 1/20. What is the effect on the output bitstream?
7. Repeat parts 5,6, and 7 from exercise 1.
8. Simulate a 4th order modulator with the same input as exercise 2 part 4. Find the largest gain in the 4th integrator which appears to allow the loop to be stable for one million clock cycles. Increase the input signal frequency to $F_{\text{clock}}/4$. FFT the output bitstream.
9. Re-instate the low frequency input. Modify your loop filter so that there are two poles at $F_{\text{clock}}/64$. You need to connect the output of the 4th integrator back to the input of the 3rd integrator. Can you see a corresponding zero in the noise of the simulated output?
10. Move your input signal frequency to the pole frequency. What is the effect on input signal gain of this pole?

EXERCISE 4: SINC Filters

1. Apply a moving average (sinc) filter to the output of the 1st order modulator. Average over 256 clock cycles. Simulate 64 k clock cycles. Compare the modulator output and filtered output, in the frequency domain. Look for zeros in the filtered output.
2. Apply sinc² and sinc³ filters, in turn, to the first order modulator output. How much improvement in the ratio of signal power to in-band noise (i.e. SNR) comes from each filter?
3. Apply sinc² and sinc³ filters to the 2nd order modulator output. Which filter gives the better signal to noise ratio?
4. Plot the impulse and frequency response of the sinc² and sinc³ filters. [The frequency response is the FFT of the impulse response.]
5. Model a sinc³ filter using the Hogenauer architecture. The filter should decimate by, and average over, 256 clock cycles. Limit the word widths to 24bits. FFT the filter output, when the input is provided by a 2nd order modulator.
6. Reduce the word widths to 12bits. What is the effect on SNR? How narrow can the words be before you see a problem?
7. Plot the signal transfer function of the modulator plus sinc³ filter at 1 kHz, 4 kHz, 16 kHz, 32 kHz, 48 kHz. Compare to 4 above.

EXERCISE 5: $\Sigma\Delta$ DAC Filters

1. Generate 1 kHz, 9.5 kHz and 10.5 kHz sinewaves sampled at 40kHz. Upsample it to 10.24 MHz. Count the number of aliases. Use 15 bit words.
2. Applying a sinc filter to the 10 MHz data. The filter should average over a 6.25 μ S period. FFT the output.
3. Repeat the last step, with a cascade of 3 sinc filters, each with a 6.25 μ S period. Count the number of aliases with less than 60 dB attenuation. What is the signal attenuation at 1 kHz, 10 kHz, 20 kHz, and 40 kHz?
4. Implement the upsampling sinc³ filter using the Hogenauer structure. Use a 24 bit word width.
5. When you go home, upsample the original signal by 4. Apply an FIR filter to remove the first aliases. Then apply your Hogenauer filter to upsample by a further 64. Design an FIR anti-alias filter that compensates for the attenuation due to the sinc filters.
6. Simulate the 'error feedback' second order modulator. Check for second order noise shaping. Compare its response to that of the corresponding Candy modulator by:
 - a) initialising all registers to 0
 - b) running for 100 clock cycles with 0 input
 - c) setting the input to 0.5 for one cycle and
 - d) running for a further 100 cycles with 0 input.

Sample ADICE Behavioural Model

1. Four files comprising a sample behavioural model of a second order sigma delta are available:

- a) **sigma2.am** code for the model, in c
- b) **sigma2.ckt** circuit level 'test bed' for the model
- c) **sigma2.use** use file (script file) to run the simulation
- d) **wfft.use** use file to FFT the results

2. The files are on Unix, in **/design/dsun525/clyden/sd_course**

3. These files were written by Dave Patrick, and sit in Cork.

4. It's recommended that people compile the model when they get it:

```
Unix% xamc sigma2.am
```

5. To run the simulation:

```
Unix% xadice5 sigma2.use
```

The simulation runs and produces a Bode plot.

**ESPRIT PROJECT
EP29644**

Video Decoder Platform

**Sigma Delta Couse
Part II**

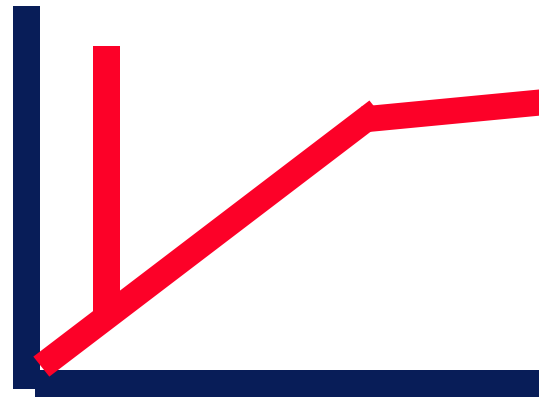
March 2000

Overview

- $\Sigma\Delta$ ADCs are noise-shaping oversampling converters
 - input signals are at low frequency
 - quantisation noise at high frequency to be removed
- Need low pass filter
- Need to remove noise before decimation
(i.e. before sample rate reduction)
- Filtering in two or more stages

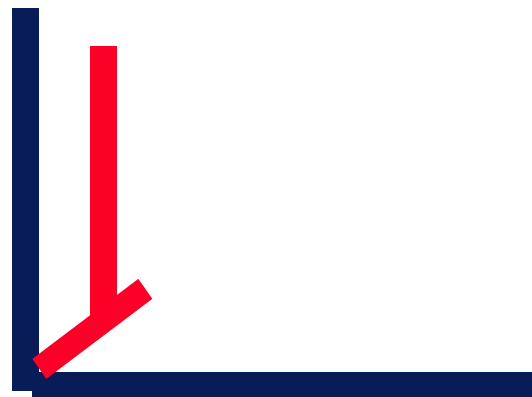
Filter And Decimate

Bitstream
(dB)



log f

Output Words
(dB)



log f

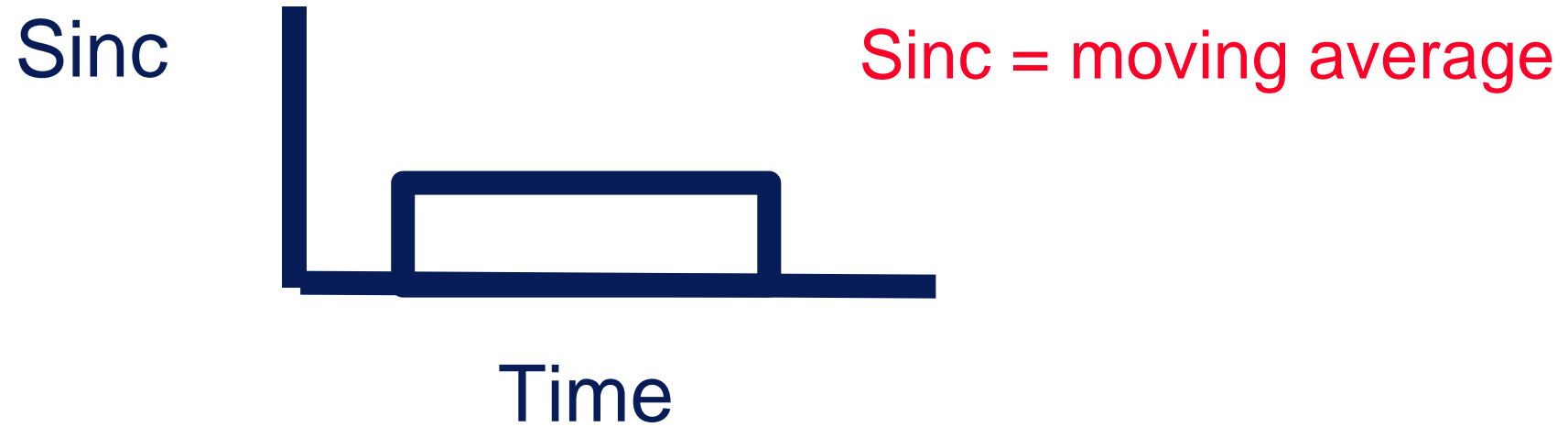
SINC Filters - First Stage

- A Sinc^P filter is normally used

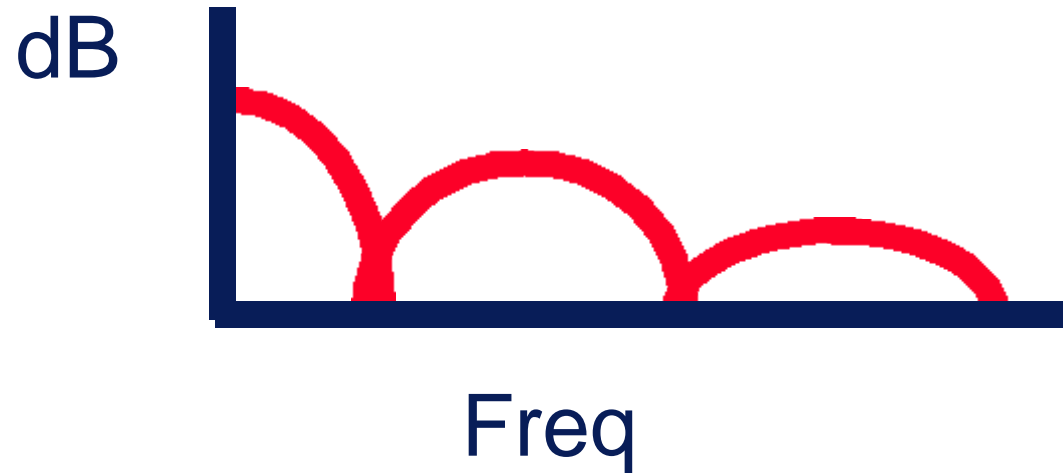
$$[(1 - z^{-N})/(1 - z^{-1})]^P$$

- Close to ideal for $\Sigma\Delta$ bitstreams
- Has zeroes at the frequencies that get aliased to dc
- Very simple (small) implementations possible
- Sinc filter of order 1 more than modulator, typically

SINC Filter Impulse Response



SINC Filter Frequency Response



- Profile is $\text{Sin}(x) / x$ - so called 'Sinc'
- Zeroes occur at $1/(\text{averaging period})$

SINC Filter Order

- A Sinc^P filter frequency response sits under an envelope of about $-P \times 20\text{dB/decade}$
- Quantisation noise from $\Sigma\Delta$ modulator of order P sits under an envelope of $P \times 20\text{dB/decade}$
- So if Sinc filter and modulator have same order, the high frequency noise is 'flattened'
- Increase the Sinc order by 1 and the high frequency noise is attenuated

Hogenauer Implementation

- Sinc transfer function

$$(1 - z^{-N})/(1 - z^{-1})$$

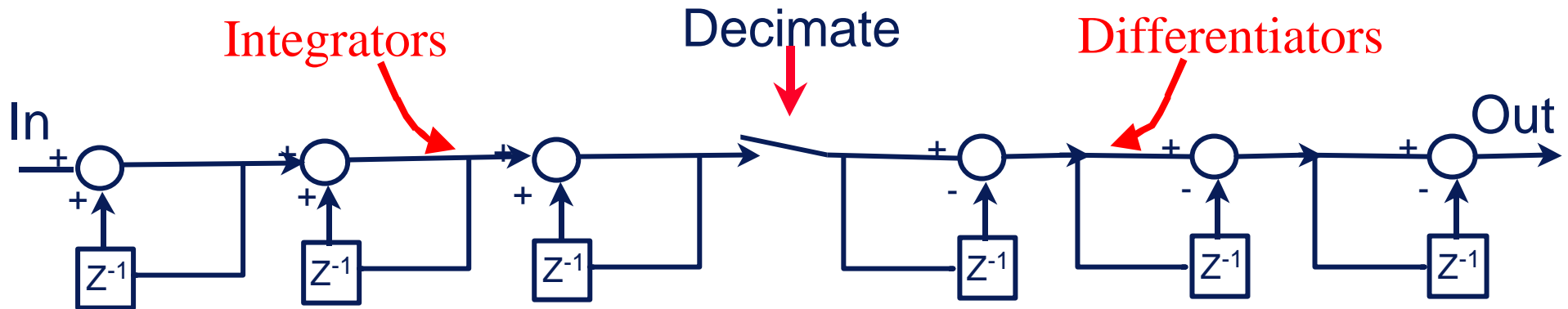
May be rewritten as

$$[1/(1 - z^{-1})] * [(1 - z^{-N})]$$

i.e. integrate followed by differentiate

- The integrator runs at the modulator clock rate
- The differentiator can be preceded by a decimate-by-N switch and run at the much slower decimated rate

Hogenauer SINC³



- Very small area
 - no coefficient storage
 - no multipliers
- Since differentiators work at decimated rate, their $(1 - z^{-1})$ is equivalent to $(1 - z^{-N})$ at the higher rate
- What about integrator overflow?

Second Stage Filter

- Sinc filters usually not used for decimating to Nyquist rate
 - roll off not steep enough
 - some inband signal attenuation
- Typically decimate to four times Nyquist rate and then use 'normal' FIR filter as second stage
- Second stage must:
 - remove some quantisation noise
 - compensate for in-band attenuation of sinc

Second Stage Design

- Passband response to be inverse of Sinc filter's
- Typical inband attenuation up to about 3dB
- Stopband attenuation related to overall ADC spec.
- Coefficients for equi-ripple filter can be generated by well know **Remez exchange** algorithm

Summary

- Two stage filtering normal
- First stage is often Sinc
 - Suited to decimation to about four times Nyquist
 - Very efficient implementation
- Second stage is more standard FIR

**ESPRIT PROJECT
EP29644**

Video Decoder Platform

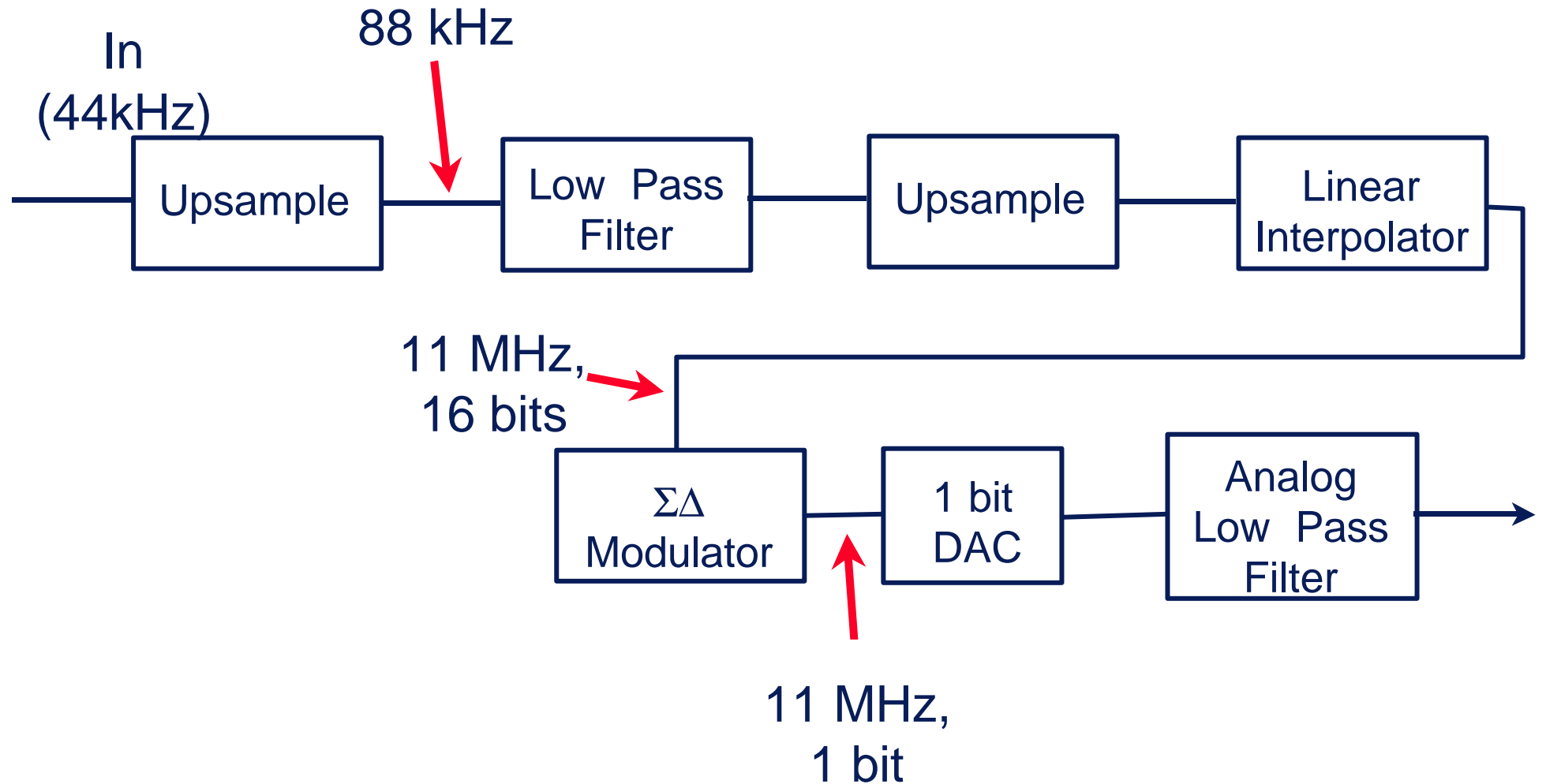
**Sigma Delta Couse
Part III**

March 2000

Overview

- Basic Sigma-Delta DAC Structure
- Modulator
- Interpolating Filter

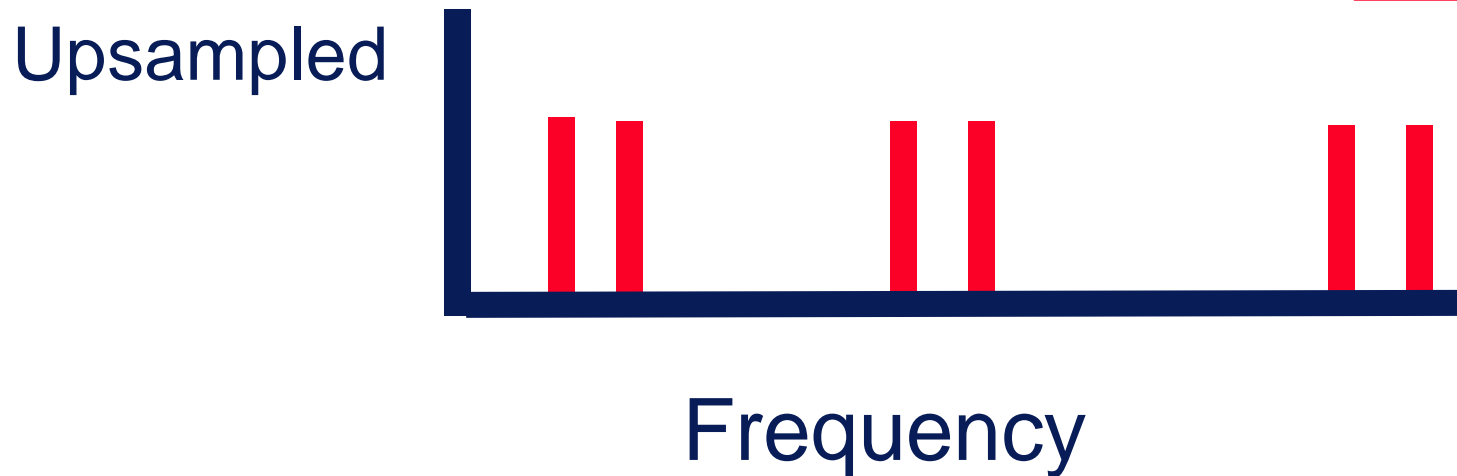
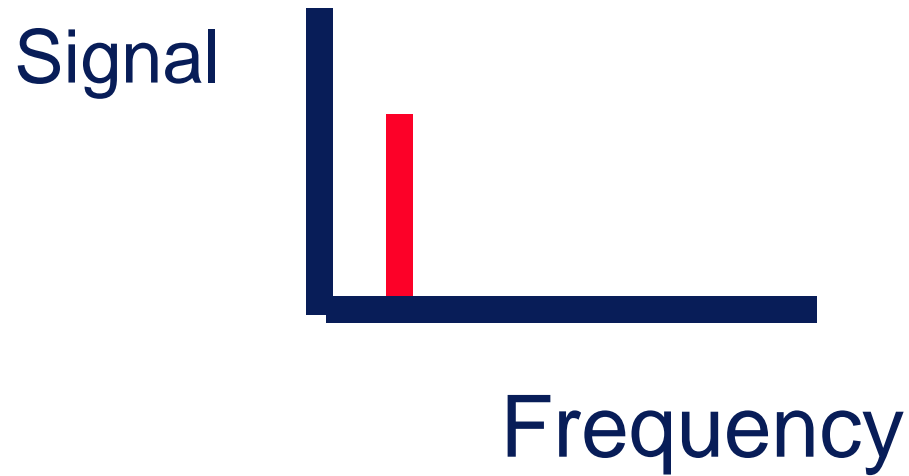
$\Sigma\Delta$ DAC Structure (Audio)



Comments

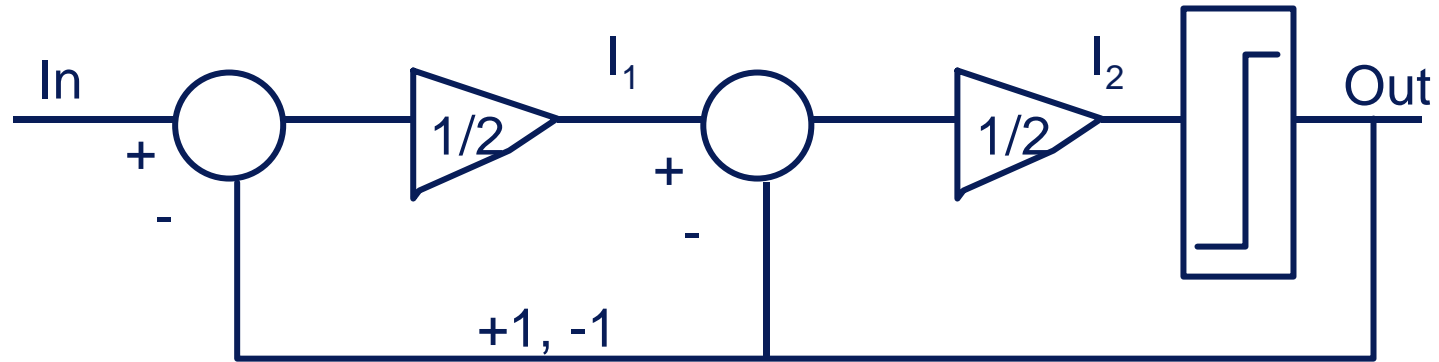
- Purpose of all the digital is make the DAC and analog low pass (reconstruction) filter easy
- Big problem is that upsampling creates aliases which must be removed
- Modulator and linear interpolator are quite easy and small

Upsampling Produces Aliases



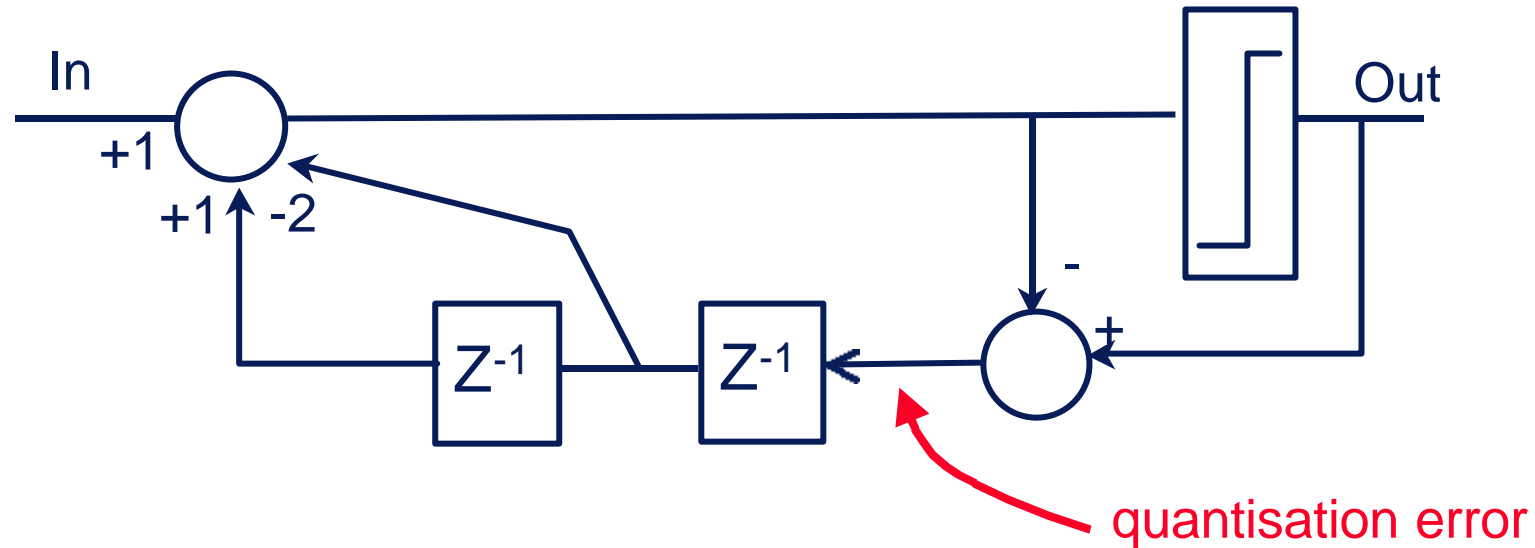
Aliases at multiples
of $F_{\text{signal}} \pm F_{\text{in}}$

$\Sigma\Delta$ Modulator



- Digital $\Sigma\Delta$ modulator could be implemented with Candy loop as above
- But, error feedback structure is more efficient in area and power

Error Feedback Modulator

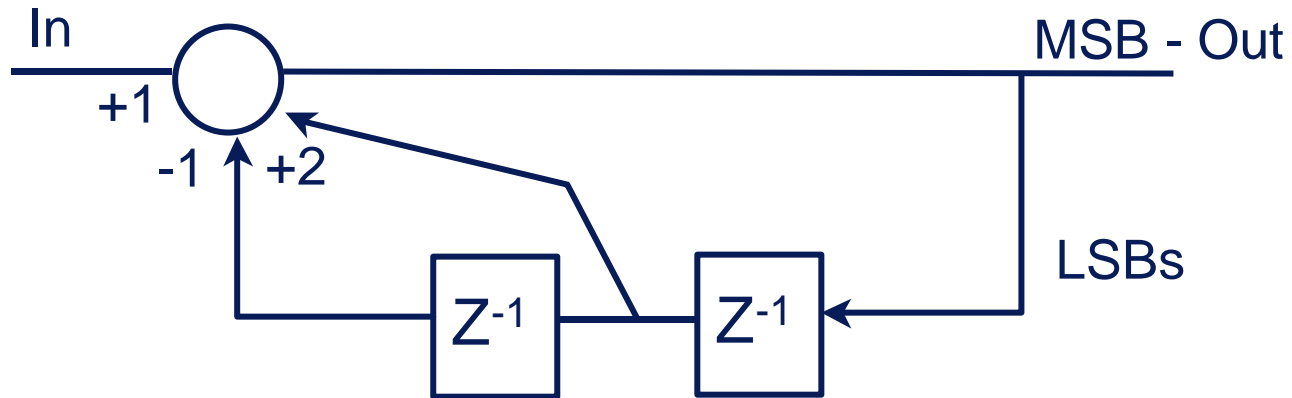


- Also gives second order shaping of error introduced at the comparator

- $Out^N = In^N + Error^N - 2 * Error^{N-1} + Error^{N-2}$

i.e. $Out = In + (\text{quantisation error differentiated twice})$

Error Feedback Modulator (2)

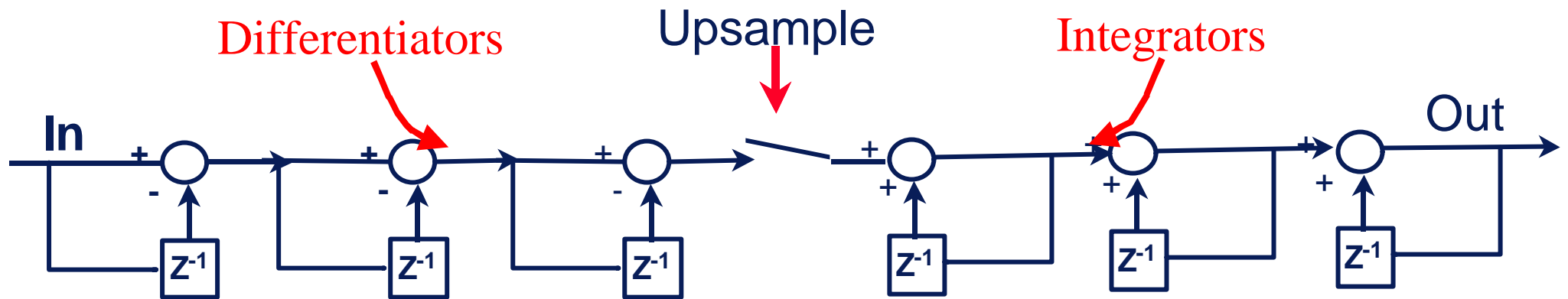


- Very efficient in area
- Any digital $\Sigma\Delta$ modulator is prone to tones
- a technique called dithering can help

Linear Interpolator

- A Sinc^P will smooth (interpolate) those aliases
- Place the zeroes of the Sinc at the frequencies that have largest aliases
- However an extra low pass filter is required to attenuate the lowest frequency alias

Interpolator



- Hogenauer Sinc³ interpolator shown
- Differentiate at lower clock rate, upsample
- Integrate at the higher clock rate
- Zeroes are centered at the center frequencies of the aliases

Digital Low-Pass Filter

- Removing first alias is difficult
- For audio CD, 16 kHz signals are sampled at 44 kHz on the CD
- Upsampling produces an image at 28 kHz
 - so filter must attenuate 28 kHz signals but,
 - pass 16 kHz signals
- That large digital filter is not addressed here

Summary

- Oversampling allows designer to use very simple DAC and reconstruction filter
- Modulator itself is quite simple
- Linear interpolation filter can be done with Sinc
- First stage filter is often large

**ESPRIT PROJECT
EP29644**

Video Decoder Platform

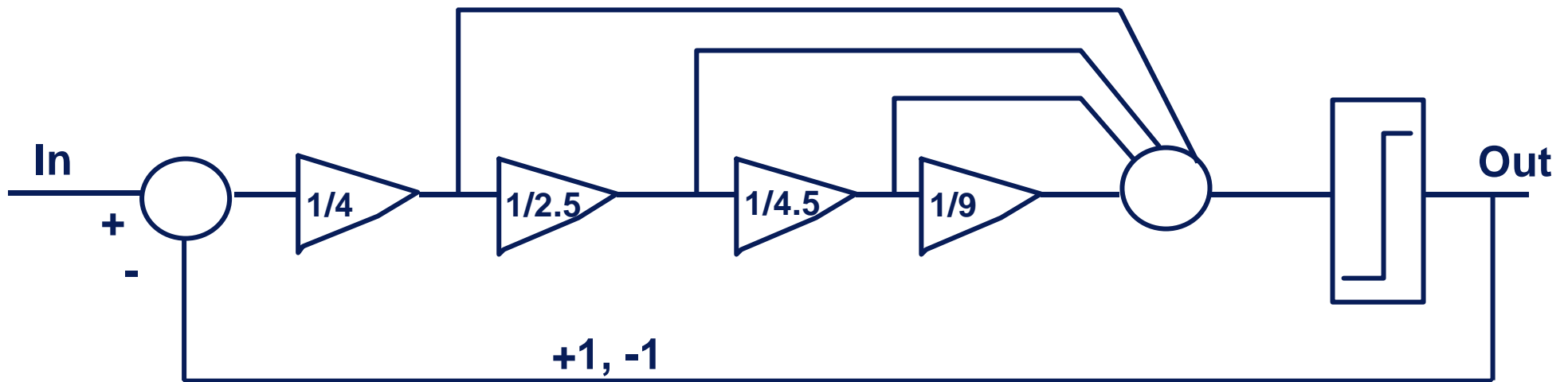
**Sigma Delta Couse
Part IV**

March 2000

Bit Shuffling Is Easy

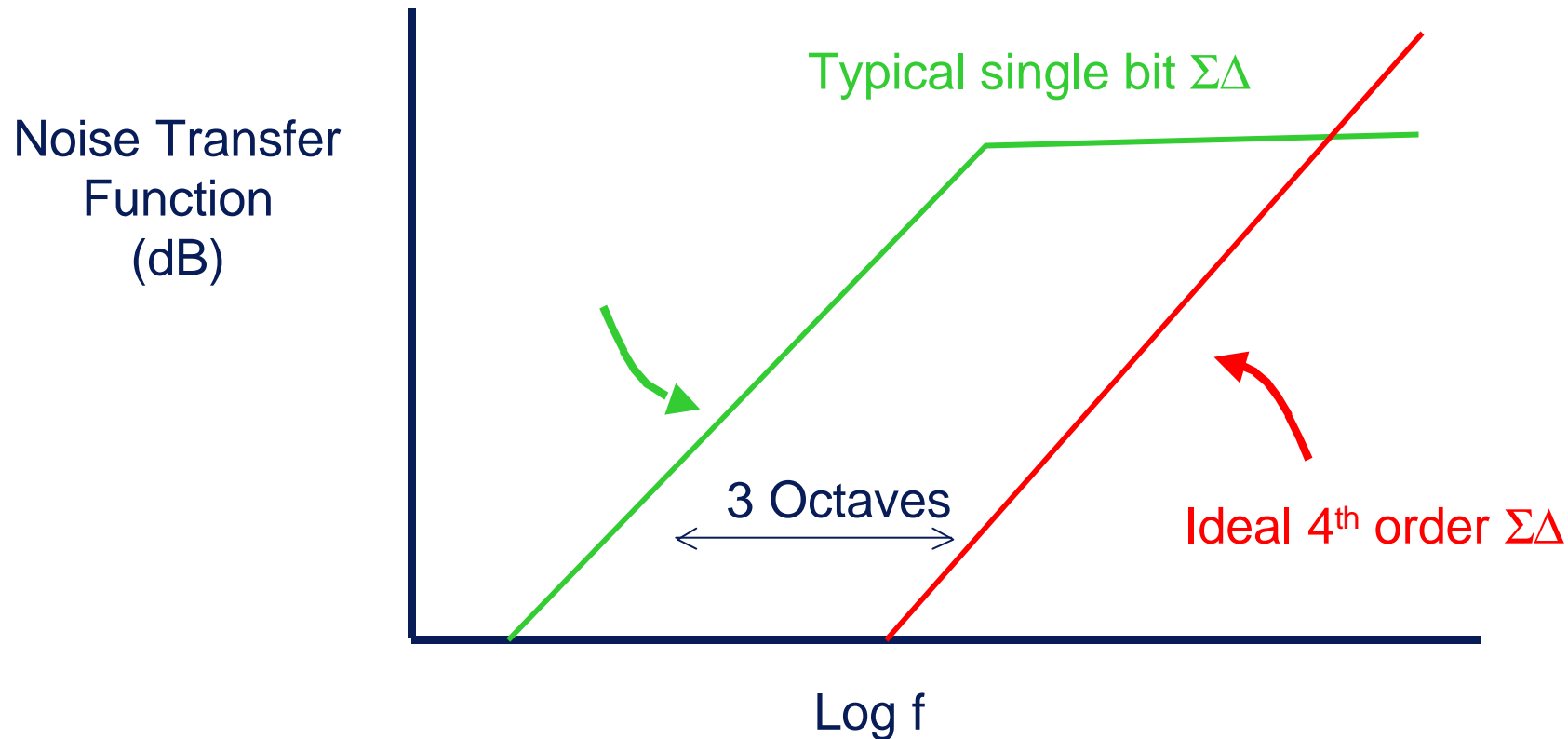
- Why multibit sigma delta
- Source of the error
- Rotating queue - a simple solution
- A more general structure
- Extension to larger DACs
- Conclusion

4th Order Modulator



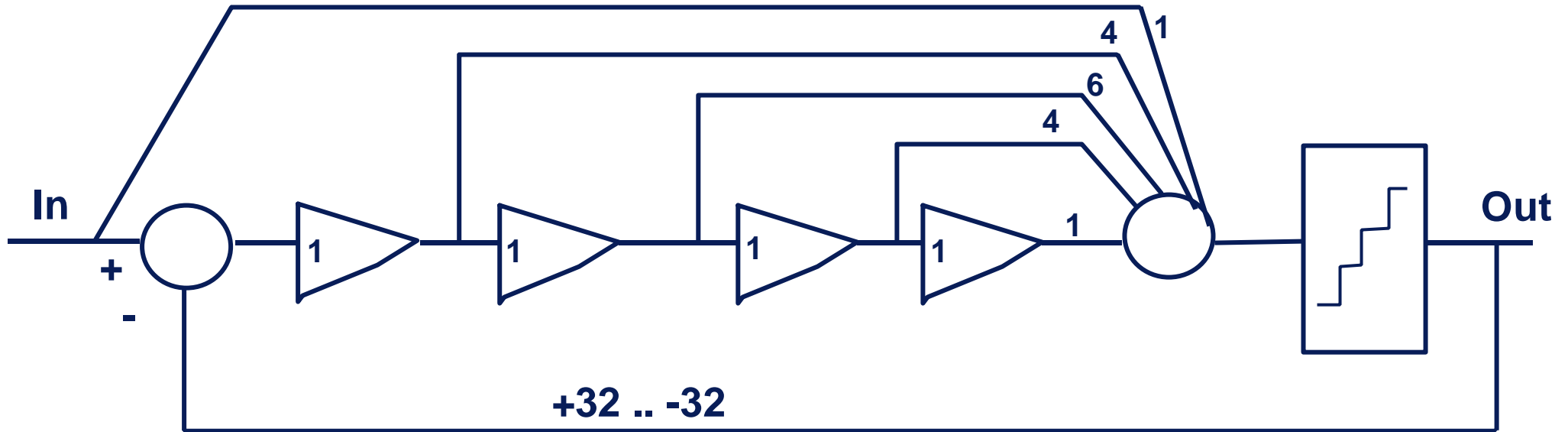
- Above 4th order single bit modulator is conditionally stable
- It is one of the better single bit 4th order modulators
- Unstable if quantiser is over-ranged
- Inputs limited to 1/4 scale through 3/4 scale
- Loop response 'dampened' to get some stability

Higher Order Noise Shaping



- To avoid overloading the quantiser, single bit, higher order $\Sigma\Delta$ s do not achieve minimum in-band noise
- Stability is tricky. Theory is difficult and not comprehensive

Multibit 4th Order Modulator

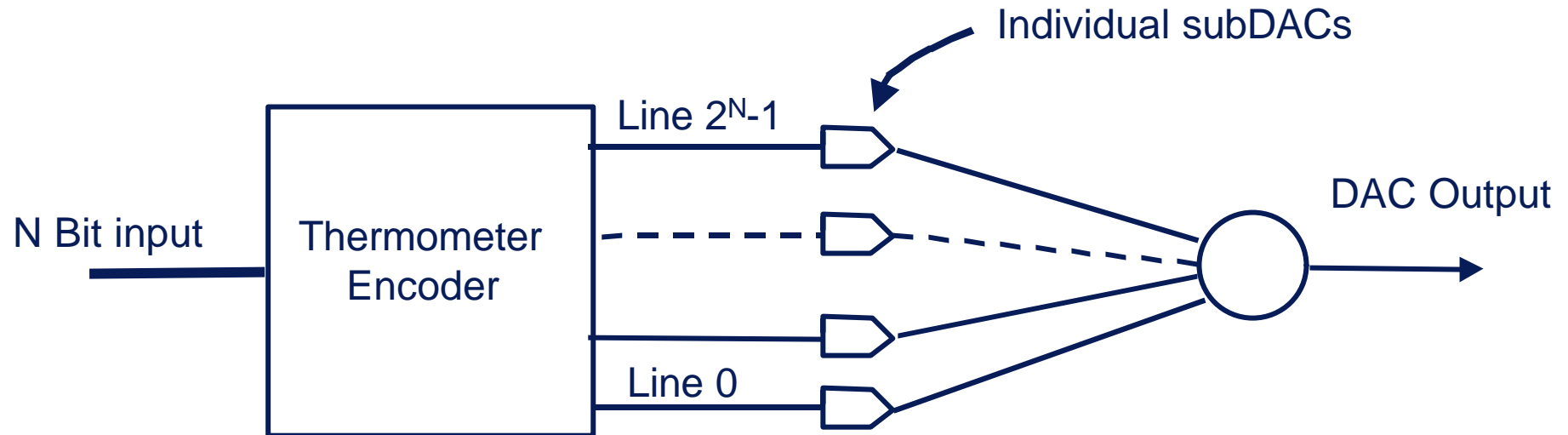


- Ideal 4th order differentiated quantisation noise
- Unity signal transfer function
- Stable, if input signal well inside feedback range (spare + / - 16 levels required)

Advantages and Disadvantages

- Direct improvement in SQNR from reduced quantisation noise
 - 1 bit / bit of extra resolution within the loop
- Improved stability allows much more aggressive loops
 - 96 dB SQNR at OSR = 64 from 1 bit 4th order modulator
 - 96 dB SQNR at OSR = 16 from 5 bit 4th order modulator
- DAC non-linearity is now an extra source of error
- Classical error source is passive component mismatch

Error Source (1)



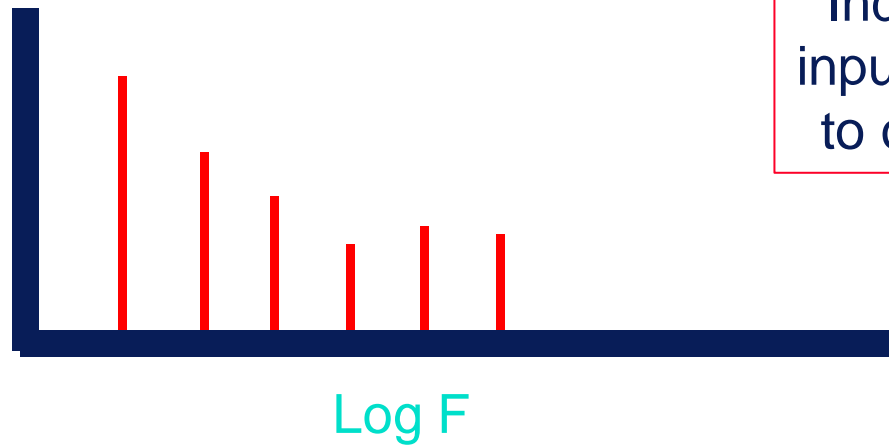
- Each DAC consists of individual analog subcomponents
- Mismatch between these subDACs is 'the' error source
- This mismatch, typically as little as 0.1%, arises from manufacturing variation

Error Source (2)

DAC Input
Signal
(dB)

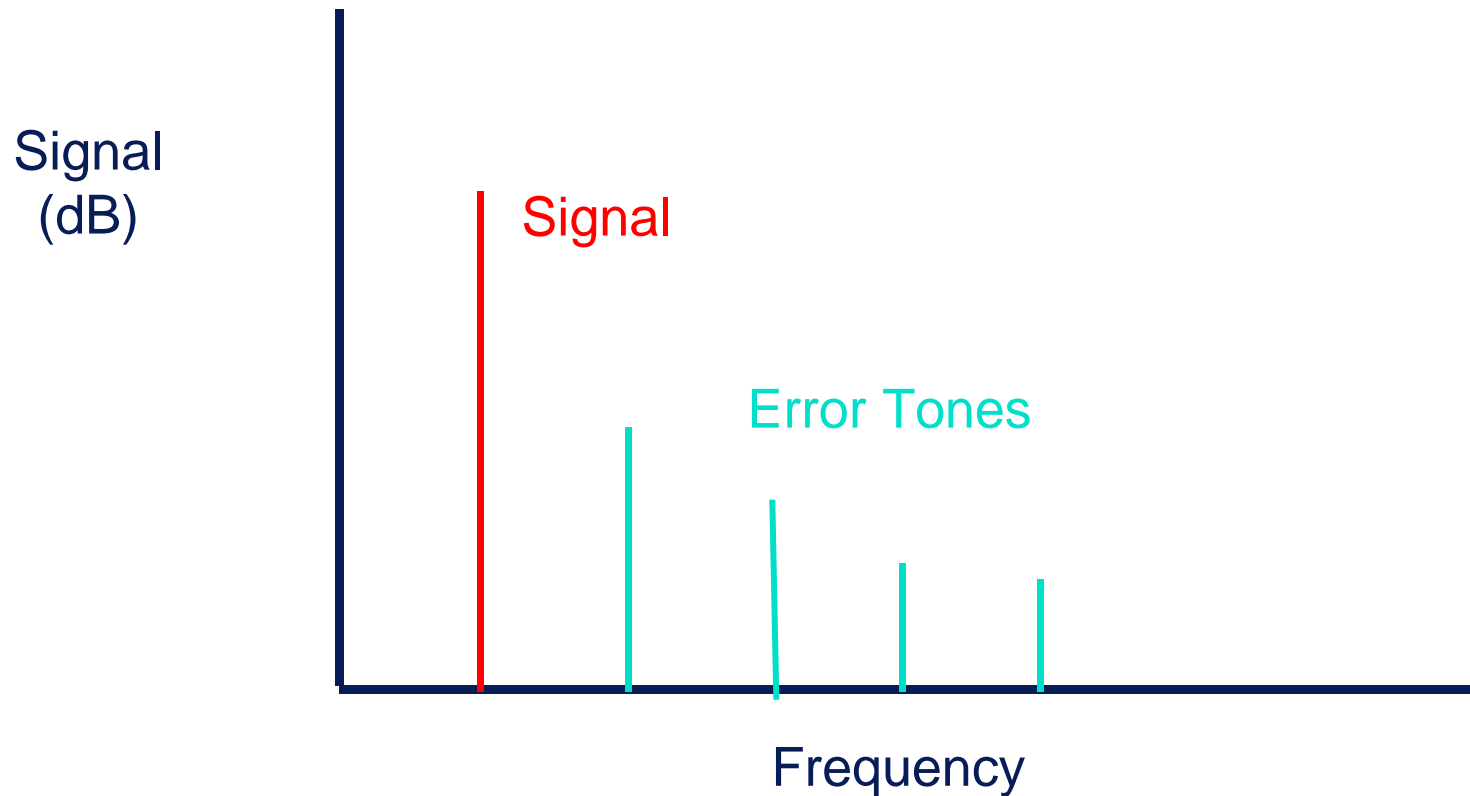


Typical subDAC
input signal
(dB)



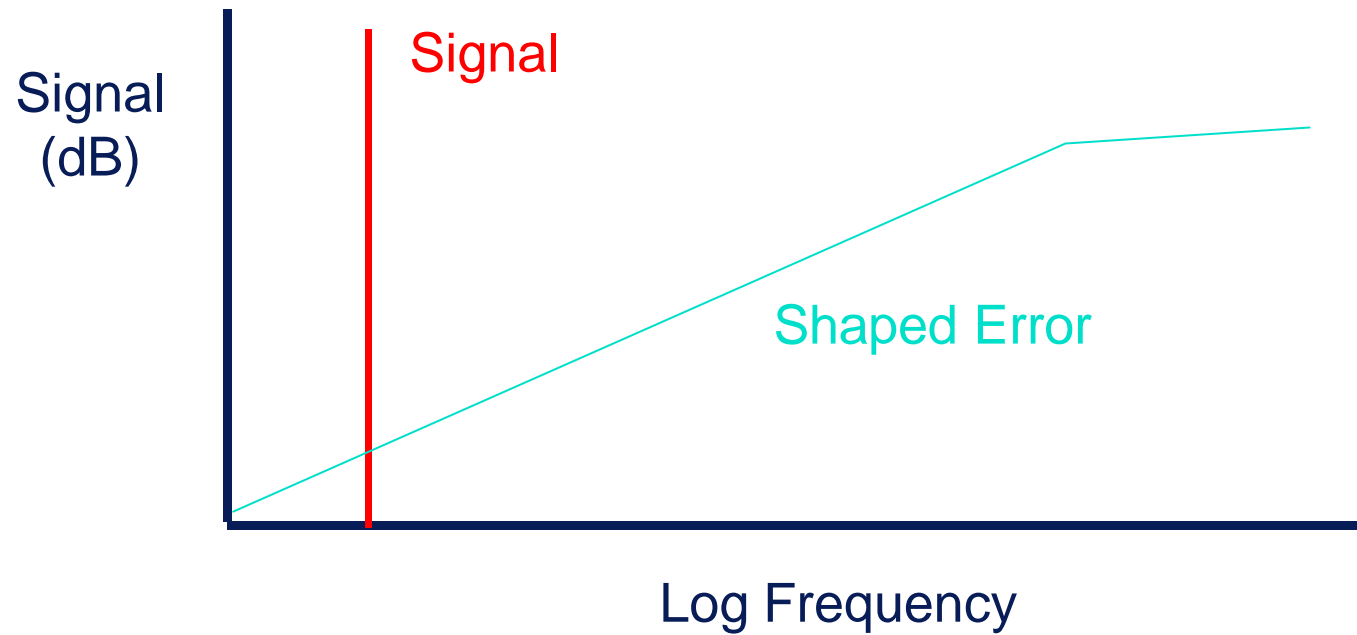
Individual SubDAC
inputs correlate poorly
to overall DAC input

Effect Of DAC Mismatch



- Sum of the individual segment inputs add up to DAC input signal
- Sum of segment outputs, in presence of individual segment gain errors, adds up to signal *plus error tones*

Ideal $\Sigma\Delta$ Treatment Of Mismatch



- In an oversampled system
- Ideally, the errors would be shaped out of band
- Little in-band error
- Possible mismatch induced DC offset

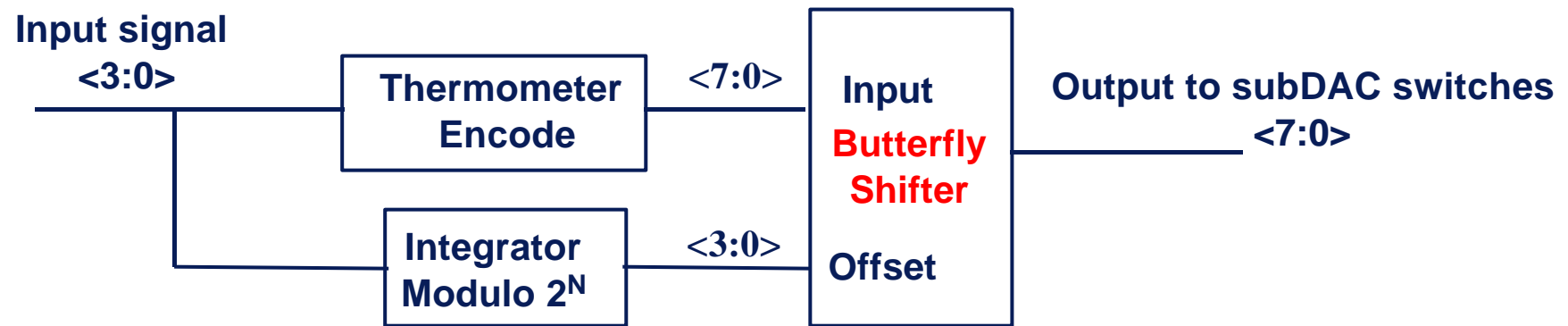
How to Bit-Shuffle (1)

- In a redundantly decoded DAC:
- Push mismatch out to high frequencies (or out of band) by ensuring that:
- The signals switching the individual DAC elements :
 - a) have in-band components that, as much as possible, follow the overall DAC input signal
 - b) push the differences out-of-band
 - c) ensure nominal DAC output is always correct

How to Bit-Shuffle (2)

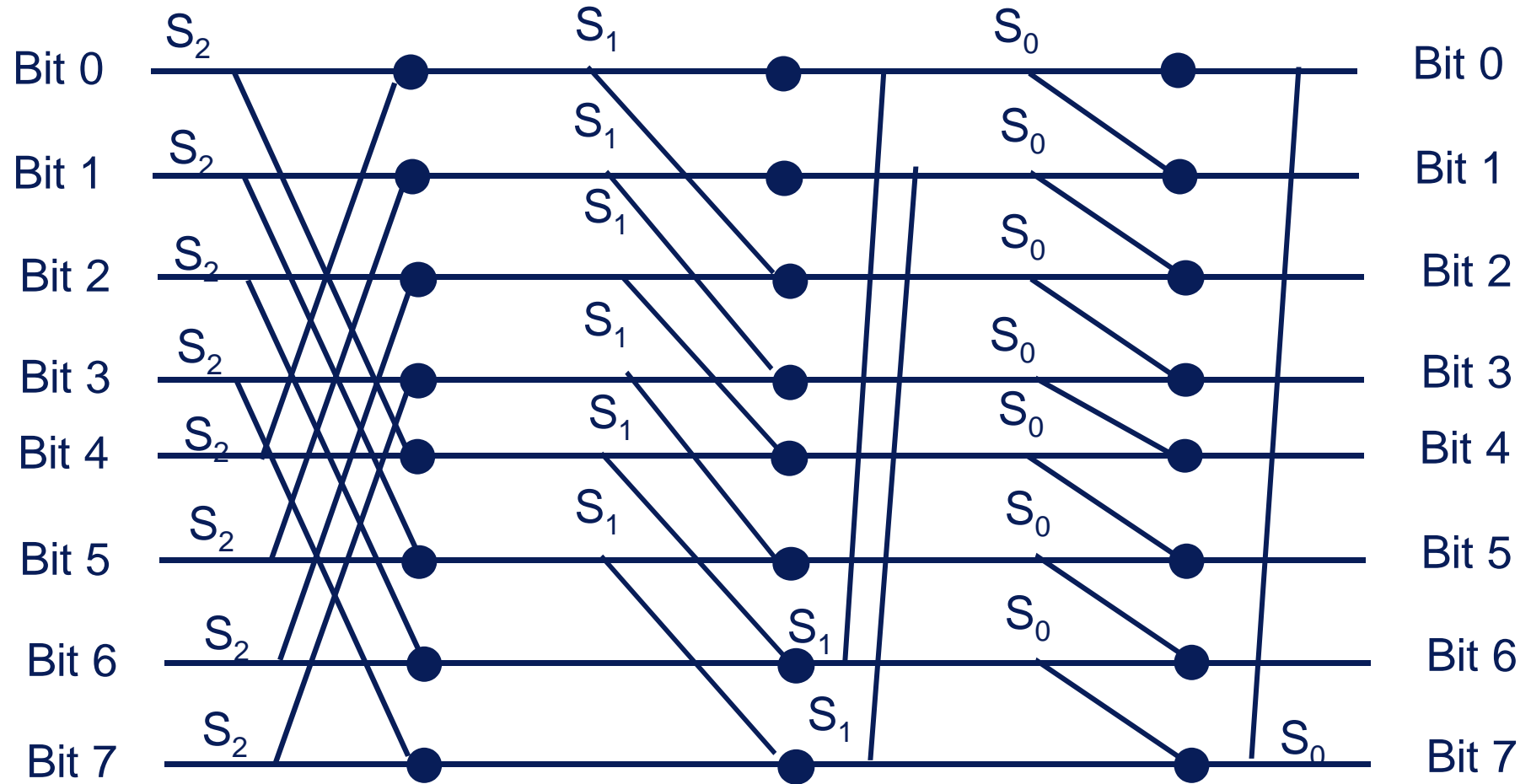
- Make segment control signals track DAC input, in-band
- Simpler to make segment control signals track each other, on average
- Simplest to use the segments in strict rotation
 - all segments used the same number of times to a granularity of one

Implementation of Segment Queue



Integrator output is the queue pointer

Butterfly Diagram



- Pass transistor logic DPL gets max delay down to 0.44 nS
- good enough for 108 MHz AD7185 sample rate

Note

- DNL should disappear
- INL is converted into high frequency noise
 - less raw INL leads to less noise
- Up to 1.5 bits per octave OSR in-band error reduction
- Digital only, no change so far to analog circuits
- DC inputs will lead to tones
 - Bob Adams data directed scrambler has lower tones
 - Could use PRBS to spread tones

Bit Shuffling

- *Won't* improve non-linear switching transients
- *Won't* reduce memory effects or glitches
 - $\Sigma\Delta$ people use RTZ DACs
- *Won't* improve superposition errors
- *Won't* improve total SNDR
 - it only improves in-band SNDR

Some References

"Multibit S-D A/D Converter Incorporating A Novel Class of Dynamic Element Matching Techniques", B. Leung, S. Sutarja, IEEE Trans. On Circuits and Systems II: vol. 39, no. 1, pp. 35-51, january 1992.

"Circuit and method for cancelling nonlinearity error associated with component value mismatches in a data converter", H.S. Jackson, US Patent 5221926, 1993

"Linearity Enhancement of Multibit DS A/D and D/A Converters using Data Weighted Averaging", R. Baird, T. Fiez, IEEE Trans. On Circuits and Systems II: vol. 42, no. 12, pp. 753-762, december 1995.

"Noise shaped multibit D/A converter employing unit elements ", R. Schreier, B. Zhang, Electron. Lett, vol. 31, 1995 pp 1712-1713
1995

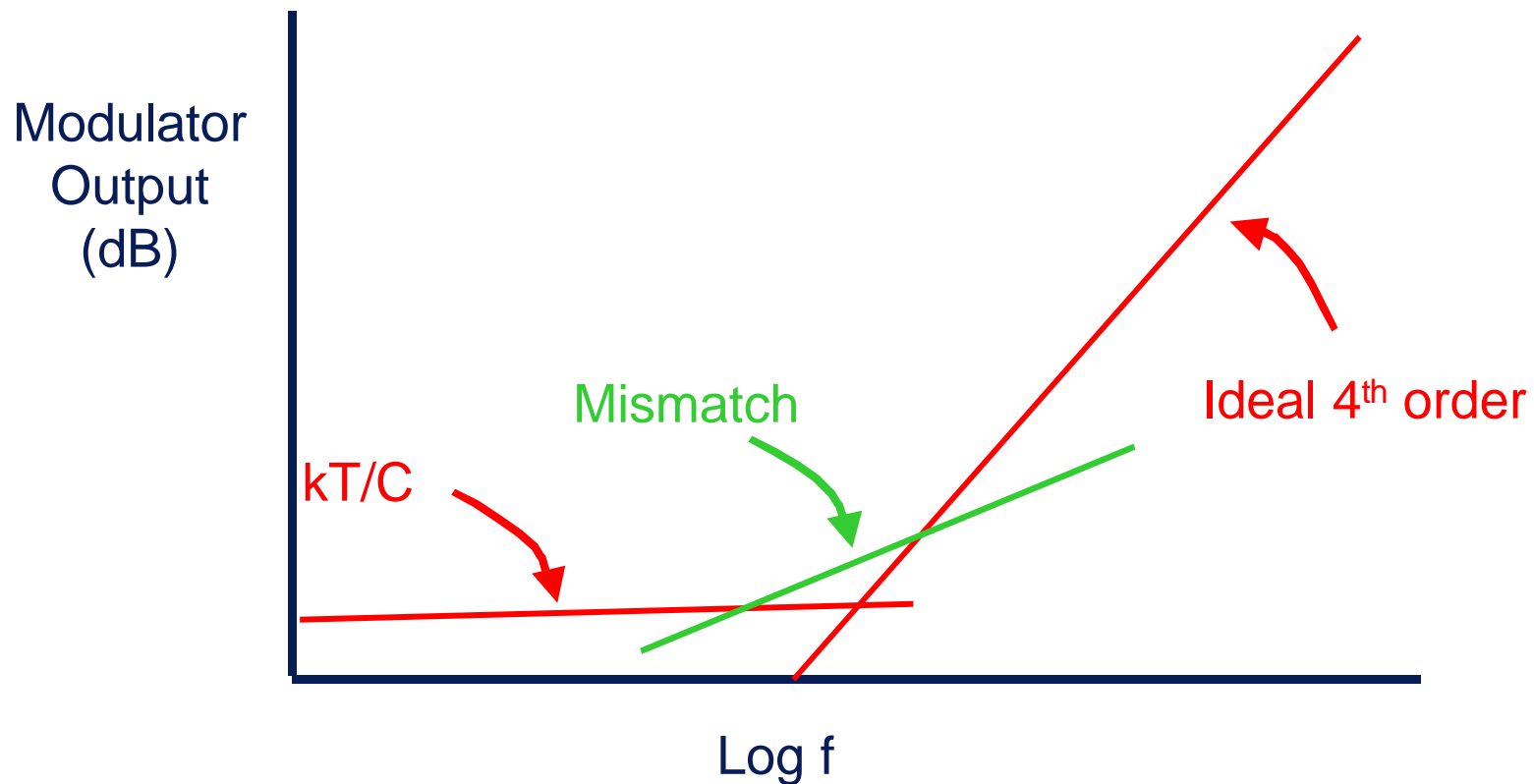
"A High Resolution Multibit Sigma-Delta Modulator with Individual Level Averaging ", F. Chen, B. Leung, IEEE Journal of Solid-State Circuits, vol. 30, no. 4, pp. 453-459, april 1995.

"Dynamic Element Matching Techniques with Arbitrary Noise Shaping Function", R. Henderson, O. Nys, Proceedings of the ISCAS 96, pp. 293-296.

"Delta-Sigma Data Converters", S. Norsworthy, R. Schreier, G. Temes, IEEE Press 1997.

"113-dB SNR Oversampling DAC With Segmented Noise-Shaped Scrambling"
R. Adams ;Q. Nguyen, K. Sweetland, IEEE Journal of Solid-State Circuits v 33 n 12 Dec 1998 IEEE pp 1871-1878

$\Sigma\Delta$ Converter with Bit Shuffling



- At high-ish resolution, shaped mismatch error may be the dominant error source
- If necessary, we can ensure that quantisation is not the limit

A General Strategy

- Filter each segment control line, to emphasise pass band
- At each conversion, prioritise segment selection so as to minimise divergence in filter outputs
- As long as loop is stable:
 - higher order filter gives higher order shuffler
 - bandpass filter gives bandpass shuffler

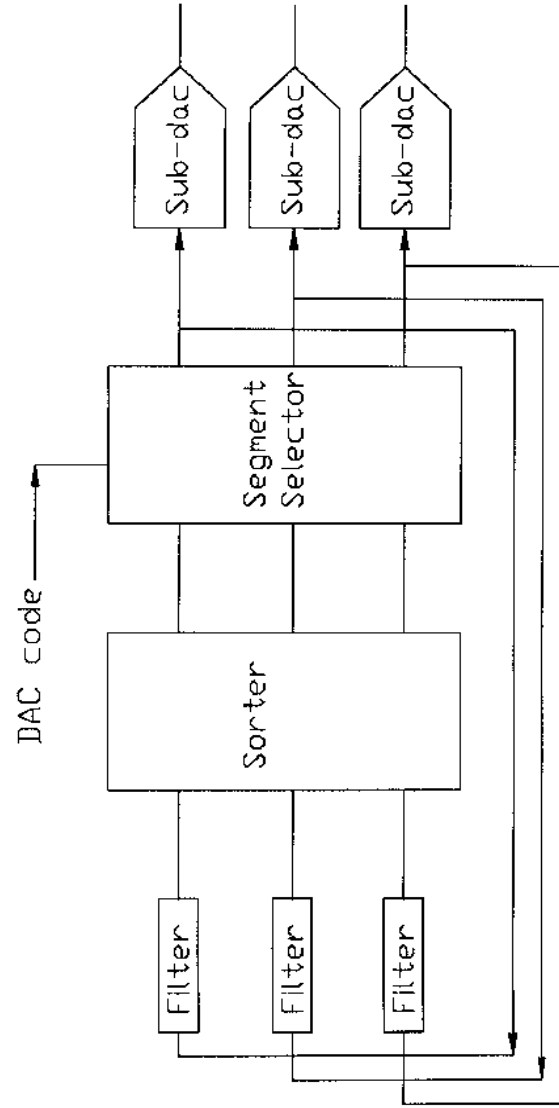


FIG. 4

Generalised Shuffler Structure



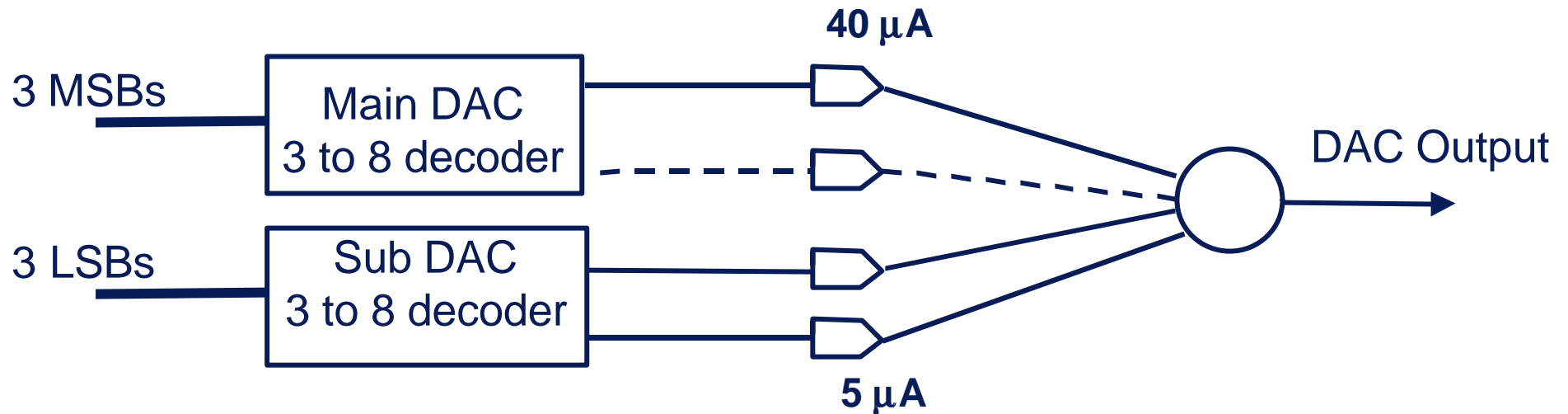
FIG. 6

Simulated Error from 2nd Order Shuffler

DAC-SubDAC Bit Shuffling

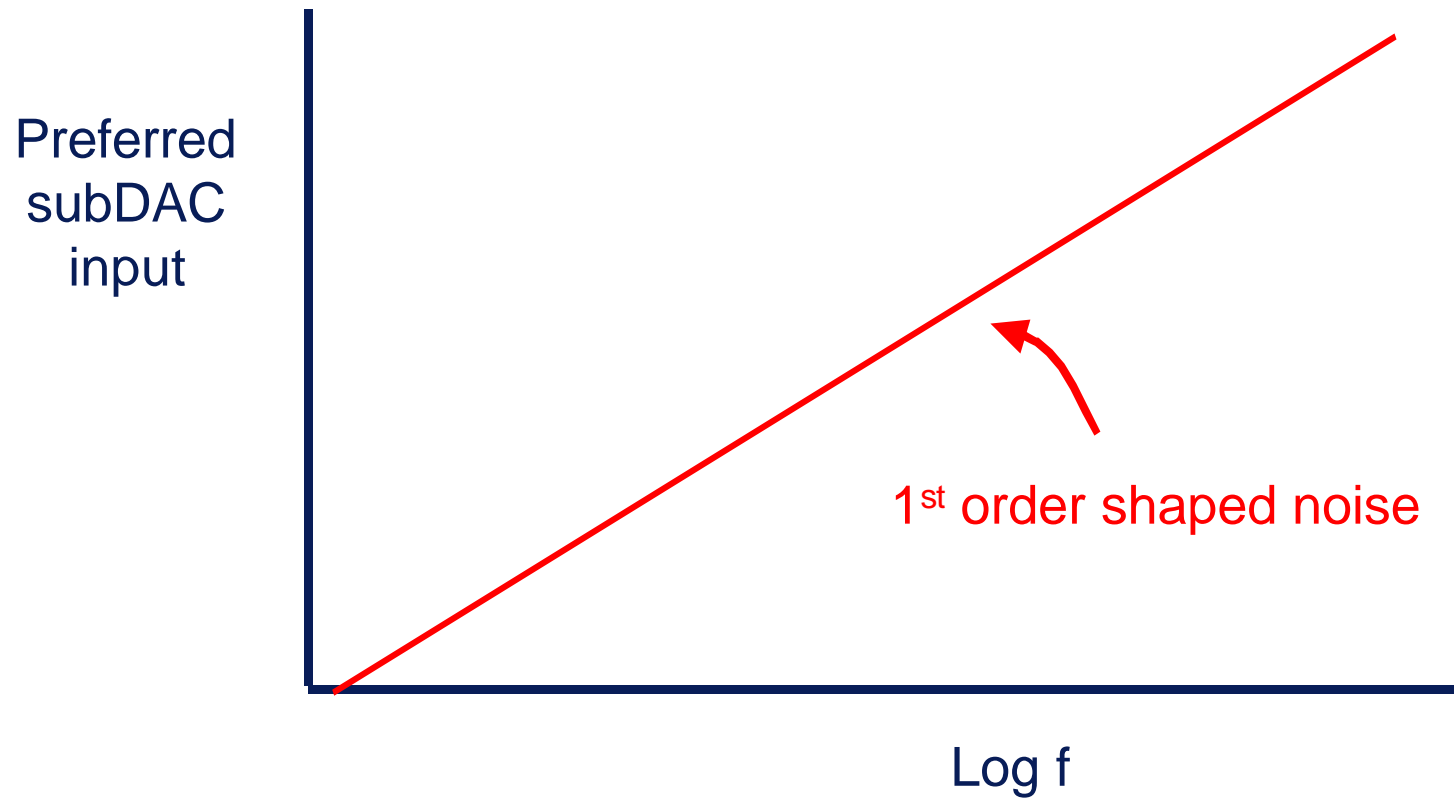
- Quantisation noise can be reduced by increasing number of bits in the DAC
- Using a fully segmented structure is awkward with more than 4 bits in the DAC
- Bob Adams first to publish a bit shuffler for a DAC plus subDAC

6-Bit DAC + SubDAC



- Main DAC is segmented
 - subDAC is also segmented
- Main DAC and subDAC are individually bit shuffled
- Gain mismatch between main DAC and subDAC remains as an error source

SubDAC Bit Shuffling



- If input to subDAC is first order shaped, then its gain error 'doesn't matter'
- Bit shuffler strategy is to first order shape the subDAC input

SubDAC Strategy

Nominal
subDAC
Input



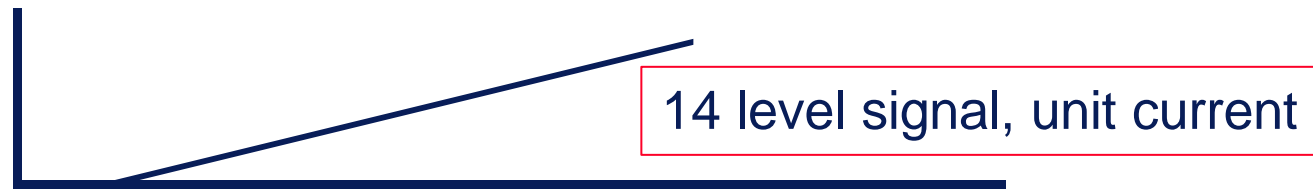
Log f

1st order
 $\Sigma\Delta$
approximation



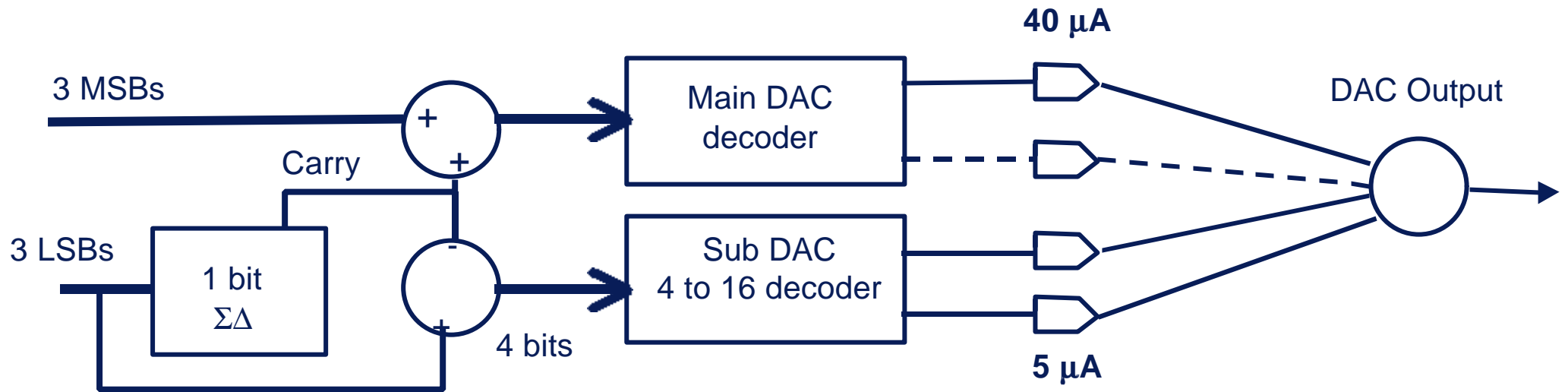
Log f

Difference
(actual subDAC
input)



Log f

6-Bit DAC + SubDAC



- SubDAC size is doubled
- Single bit approximation to subDAC input added to main DAC
- First order shaped, bipolar residue is subDAC input
- Main and subDAC are individually bit shuffled

SubDAC Bit Shuffler

- Technique can be extended to multiple levels of subDAC
- Its possible to shuffle a binary weighted DAC (R2R) but DAC size is doubled
- Mike Keane's AD74322 in development is a fully shuffled 10 bit DAC for audio (10 bits to minimise out-of-band noise).

By The Way ...

- Why don't SAR ADCs use this technology?
- Shuffle once per conversion
- Provide a simple Sinc filter for user
 - let the user decide if they want very good DNL and INL at lower frequencies

Bit Shufflers

- Make segment control signals approximate input signal, in-band
- Or, minimise non-signal, in-band, segment control signals
- DC component can usually be removed by cal.
- Simple digital blocks - easy to extend or find alternatives for speed/power efficiency

Conclusion

- First order shuffling of mismatch is easy
- Removes DNL
- Converts INL to high frequency noise
- Extension to bandpass is well published
- Extension to subDAC and even binary weighted DAC is understood
- Lot's of in-house expertise