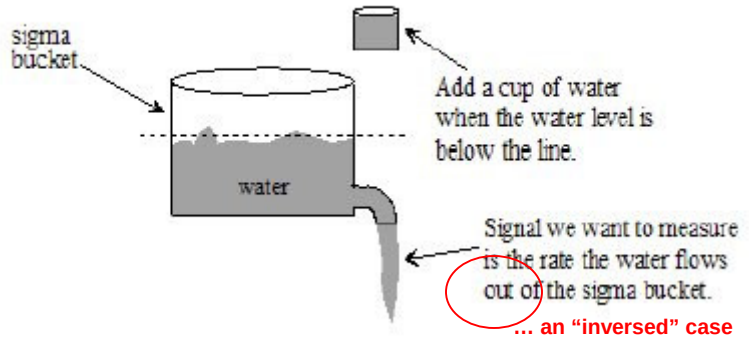
Using a bucket-water analogy to illustrate **DSM (Delta-Sigma modulation)**

A *robust* sensing scheme using *simple* signal processing (averaging) for measuring an analog quantity.



Averaging can be thought of as reducing the noise in the signal (the variations in water level because of sloshing). Averaging **how often** we add the cup of water gives a *digital representation* of the signal we are trying to measure.

The size of the cup that adds water is important.

- Using too small of a cup results in the water draining out of the bucket. (We can't add the water fast enough).
- Using a small cup for adding water increases the resolution.

As long as the water level is at a "constant value" the actual level is unimportant (offset doesn't matter).
If the sigma bucket is "leaky" and the water it holds leaks out the quality of the sense will be affected.

What limits the **resolution** of this scheme? 1) A leaky bucket, and 2) filling the cup imperfectly.

Example (… a "normal" case): Assume that the **rate** of water **flowing into** the sigma bucket, is 1 cup every 40 seconds. (0.25 cups per 10 seconds). We **remove** a cup of water **from** the bucket every time the water level is > 5 cups
Say that the height of the water in the bucket is checked every 10 seconds. We can write (assuming we want to keep, water height at 5 cups our reference line):

Time (secs)	Water level in sigma bucket (cups).	Remove cup? (Water level >5)	Average # cups
0	5	No, don't remove	0
10	5.25	Yes	1
20	4.5	No	0.5
30	4.75	No	0.33
40	5	No	0.25
50	5.25	Yes	0.4
60	4.5	No	0.33
70	4.75	No	0.29

$(5.25-1=)4.25+0.25=4.5$

$(1+0)/2=0.5$

$(1+0+0)/3=0.33$

$1/4=0.25$

$2/5=0.4$

. . .

Continuing, we can write:

Time (secs)	Water level in sigma bucket (cups).	Remove cup? (Water level >5)	Average # cups
80	5	No	0.25
90	5.25	Yes	0.33
100	4.5	No	0.3
110	4.75	No	0.27
120	5	No	0.25
130	5.25	Yes	0.31
140	4.5	No	0.29
150	4.75	No	0.26
160	5	No	0.25

Note how, as we increase the number of samples, the average bounces around 0.25 cups/10 seconds. The longer we average the closer the output converges on 0.25 .

- **The "input" signal is the product of the output number (average) and the feedback signal size (cup size) or here 0.25*10 .**

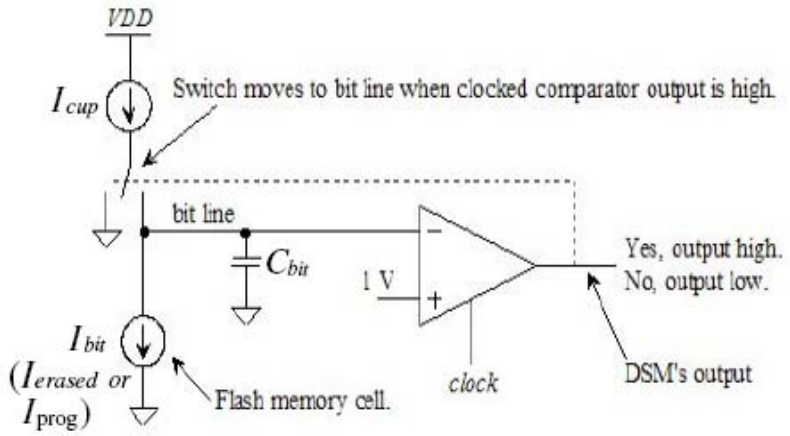
Note that if we make a wrong decision it doesn't really matter.

- If we do not determine the level of water correctly it really doesn't matter! The error will average out over time.
- A **counter** is used for averaging (count the number of times we remove water from the bucket).

Note that the longer we sense the better the sense …

Sensing a Flash Memory using **DSM**

State of the flash cell *erased* or *programmed* can be determined precisely by DSM.



C_{bit} = sigma bucket

I_{bit} (I_{erased} or I_{prog}) = the signal we are trying to measure, rate current flows out of the bucket.

I_{bit} can be determined very precisely by looking at the number of times the output of the DSM sensing circuit goes high.

It may be Greek to you, but sigma delta converters are not really hard to understand.

James Bryant -- **7/17/2006** http://www.analog.com/analog_root/static/raq/raq_sigmaDeltaConverters.html

Q. *Can you please explain, simply, as to a Bear of Little Brain[1], how sigma-delta converters work?*

A. By **over-sampling, noise shaping and digital filtering.**

Athens is a beautiful city, with the ambiance of many millennia of history. I was walking round the Acropolis with Spiros, one of our Greek distributors, when he asked me how sigma-delta (Σ-Δ) converters work. "Sigma and delta are letters of our Greek alphabet," he exclaimed, "but every article I have seen about their operation is double dutch[2] to me. They all start with several pages of partial differential equations and then go downhill from there."

If a voltage is measured many times, the average of the measurements will be more accurate than most individual measurements. This is **"over-sampling."** (Dither[3] may be necessary to randomize the errors in the individual measurements.)

There is a definite theoretical minimum limit to the possible noise of an analog-to-digital converter (ADC). When an ADC samples a signal at a frequency of f_s the digital output contains the signal and this "quantization noise" is usually spread *evenly* from dc to $f_s/2$. By sampling at a higher rate of Kf_s, the noise is spread over the wider band from dc to $Kf_s/2$. If we then remove all the noise above $f_s/2$ with a digital filter the signal-to-noise ratio (SNR) of the digital output is improved — effectively *improving* the ADC resolution.

Normally the SNR increases with the square root of K, so very high sampling rates are necessary for useful increases in SNR. But a Σ-Δ modulator does not produce uniformly distributed quantization noise. Although the *total* noise is unaltered in a Σ-Δ system, most of it is at high frequencies (HF). This is known as **noise shaping** and permits much lower values of K.

If the digital output from the Σ-Δ modulator is **filtered** to remove HF, leaving the frequencies from dc to $f_s/2$ (where the wanted signals are) then the SNR and resolution of the digital output are improved.

A Σ-Δ ADC simply consists of a Σ-Δ modulator and a digital low-pass filter, both of which are easily made with modern high-density digital technology. The *principle* of Σ-Δ ADCs has been known for more than 40 years, but the ability to build one on a *chip* is relatively recent.

1. *"When you are a **Bear of Very Little Brain**[1] and you think of Things, you find sometimes that a Thing which seemed very Thingish inside you is quite different when it gets out into the open and has other people looking at it." — AA Milne, "The House at Pooh Corner"*
2. *Double dutch means **gobbledygook**[2]*
3. *Dither — the addition of noise or some other AC signal in order to randomize errors.*

---

[1] "...milyen nehéz dolog egy CSEKÉLYÉRTELMŰ MEDVÉNEK, mikor mindezeket el akarja gondolni."
[2] nagyképű HALANDZSA

Getting Inspiration from Electrical Engineering … to Develop Interesting New Mathematics
Daubechies, http://www.nsf.gov/pubs/2002/nsf0120/nsf0120_15.htm

Applied mathematicians can use their craft to help scientists and engineers *solve numerical, simulational, or modeling problems*, and they have often done so, in many fields, with great success, developing new applied mathematics tools as they progress.
Talking to scientists and engineers should also lead to forms of applied mathematics where *the science or engineering problems pose conceptual challenges* that force us to (hopefully) develop whole new fields of mathematics (ultimately). **It is much harder to foretell what these fields will look like - the power and importance of Fourier Analysis would have been hard to predict before Fourier.** Our only hope to not miss these chances is to work hard and earnestly with scientists and engineers to get steeped in their problems, and not just be "consultants" for them. But it can be very hard to do this.

A/D conversion in EE

Digital signal processing has *revolutionized* the storage and transmission of audio signals, images and video, in consumer electronics as well as in more scientific settings (such as medical imaging). **The main advantage of digital signal processing is its robustness.** Although all operations have to be implemented with necessarily not quite ideal hardware, the *a priori* knowledge that all ideal outcomes must lie in a very restricted set of well separated numbers makes it possible to recover the ideal outcomes by rounding off appropriately. When bursty errors can compromise this scenario (as is the case in many communications channels, as well as for storage in memory), making the "perfect" data unrecoverable, knowledge of the type of expected contamination can be used to protect the data, prior to transmission or storage, by encoding them with error correcting codes. This is again done entirely in the digital domain. All these advantages have contributed to the present widespread use of digital signal processing. Many signals, however, are inherently "analog" rather than digital in nature; *audio* signals, for instance, correspond to functions, modeling rapid pressure oscillations, which depend upon a "continuous" variable, and the range of the signal typically also fills an interval.
For this reason, the first step in any digital processing of such signals must consist of a conversion of the Analog signal to the Digital world, usually abbreviated as **A/D** conversion. Note that at the other end of the chain, after the signal has been processed, stored, retrieved, transmitted, ..., all in digital form, it needs to be reconverted to an analog signal that can be understood by a human hearing system; we thus need a **D/A** conversion there.

The digitization of an audio signal rests on two pillars: sampling and quantization

Moving from "analog time" to "discrete time" can be done without any problems or serious loss of information. At this state, each of these samples is still a *real* number.
The transition to a discrete representation for each sample is called quantization. The simplest way to "quantize" the samples would seem to replace each by a **truncated binary** expansion. If we can "spend" k bits per sample, then a natural solution is to just select the first k bits in the binary expansion of the sample value. *Quantized representations of this type are used for the digital representations of audio signals, but they are, in fact, not the solution of choice for the A/D conversion step. (Instead, they are used after the A/D conversion, once one is firmly in the digital world.)* The main **reason** is that it is **very hard (and therefore very costly) to build analog devices** that can divide the amplitude range into 2^{k+1} **precisely** equal bins. It turns out that it is much *easier (=cheaper)* to increase the *oversampling* rate, and to spend fewer bits on each approximate representation. **Sigma-Delta** quantization schemes are a very popular way to do exactly this: they oversample significantly, and then spend very few bits per sample, but nevertheless achieve a very close approximation for the overall function when **coarsely quantized "samples"** are used instead of the true samples. In the most extreme case, every sample is replaced by just *one* bit. At the heart of this method lies an algorithm that **recursively rounds** (quantizes) each sample value to one of the few quantization levels in a way that is suited to achieve small *global* reconstruction error rather than small individual sample error. Surprisingly, very little math work has been done of these systems.

Sigma-delta modulation has its roots in the **60's** when Inose and Yasuda developed the first unity bit encoding by negative feedback. Similarly, the classical error diffusion algorithm of Floyd and Steinberg was invented in the **70's**. In the late **80's**, interest in the *information theory* community was led by Gray who discovered some of the best known theoretical results. It was, however, only in the late **90's** that an *approximation theoretical framework* was given to the problem by Daubechies and DeVore. Since then, there has been a rapid development in the *theoretical analysis* of sigma-delta modulation with emerging connections to other *mathematical* fields.

We constantly had to go back to the drawing board as we understood better what the engineers meant. The process of keeping in touch with engineers at all stages was essential.