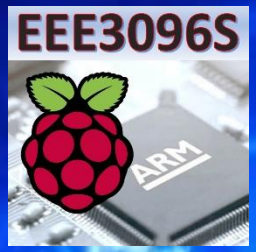


ADC & Sampling

Embedded Systems II

L20

Dr Simon Winberg

Electrical Engineering
University of Cape Town

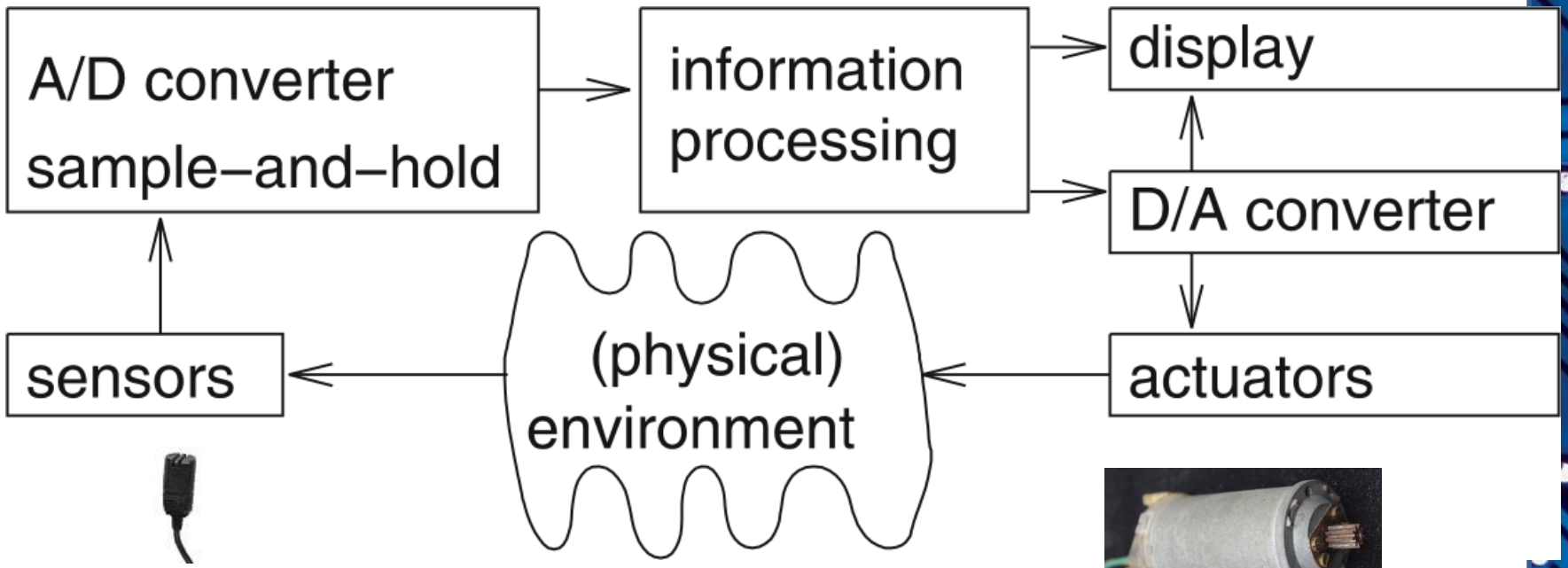
Outline of Lecture

- Hardware in a loop
- Signals
- Sample-and-hold circuits (also called ‘S/H’)
- Sampling (recap)
- Shannon and Nyquist & relevance to ADCs
- ADC – basic conceptual model
- ADCs Flavours

Slides adapted from course textbook “Embedded System Design Embedded Systems Foundations of Cyber-Physical Systems, and the Internet of Things” by Peter Marwedel

Embedded System Hardware

- Embedded system hardware is frequently used in a loop of sampling and actuating (i.e. ***"hardware in a loop"***):



☞ cyber-physical systems

Many examples of such loops

- Heating
- Lights
- Engine control
- Power supply
- …
- Robots

Examples of common sensors used in the loop

- Rain sensors for wiper control
  ("Sensors multiply like rabbits" [ITT automotive])
- Pressure sensors
- Proximity sensors
- Engine control sensors
- Hall effect sensors

Signals

Sensors generate *signals*

Definition: a **signal** s is a mapping

from the time domain D_T to a value domain D_V:

$$s : D_T \rightarrow D_V$$

D_T : continuous or discrete time domain

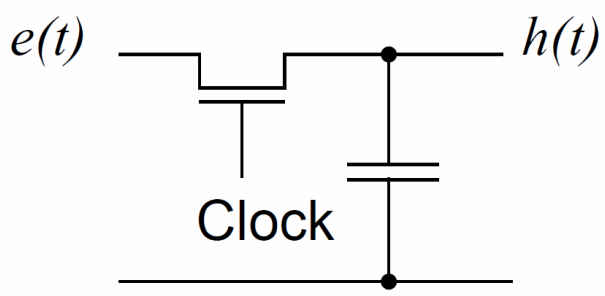
D_V : continuous or discrete value domain.

Sample-and-hold circuits (also called ‘S/H’)

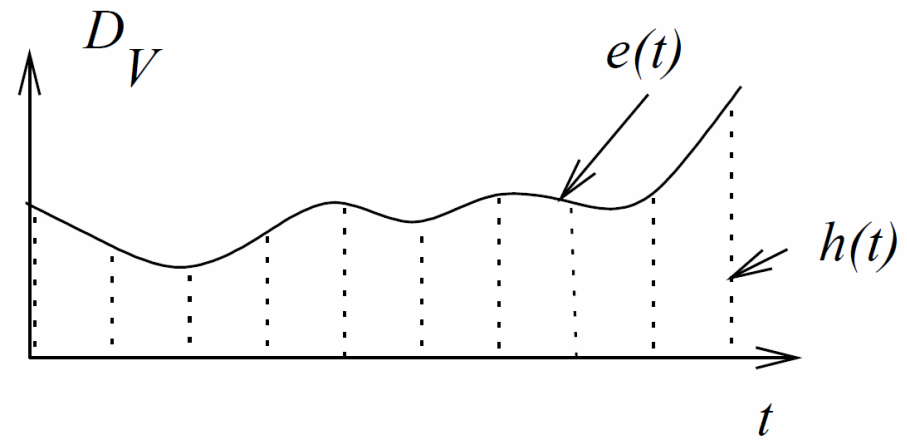
Clocked transistor + capacitor;

Capacitor stores sequence values

A good S/H is an essential part of a an ADC because you ideally want to represent the voltage at a certain instant.



Simple S/H implementation



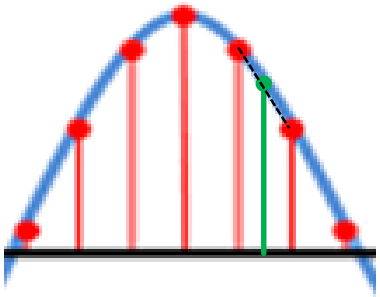
$e(t)$ is a mapping $\mathbb{R} \rightarrow \mathbb{R}$

$h(t)$ is a **sequence** of values or a mapping $\mathbb{Z} \rightarrow \mathbb{R}$

Basically what we are saying here is that you’ve got a certain countable numbering of instances that map to a value.

Where Z is of course integers (negative, zero, and positive)

Simon's Guide To The Essence of Sampling (recap)



EEE3096S

SAMPLING

Embedded Systems II

Sampling

Q: Do we lose information due to sampling? …

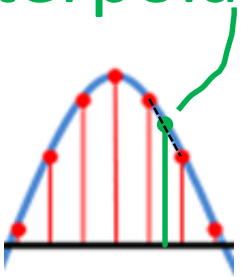
Well this question might seem to most like asking "Is the pope catholic?"

A: In case you are wondering, the answer I'm looking for is: Yes.

But we may not need all the information that is lost, what we sample may be sufficient. For example, high logic may be +5V and low logic may be 0V with a +/- 1V tolerance. So if we got a 5.1V that extra 0.1V is not carrying any useful information.

Sampling & Interpolation

- Sampling and quantization are important:
  - This translates the signal from the analog world to the digital world
- Sampling happens at particular instances*
- When the goal is to determine the value of the signal at some particular point in time, it may be necessary to use interpolation between actual samples.



** For all intense and purpose we can say 'instance' although the physics implies it is actually an averaged value over a tiny period starting at a particular moment.*

Shannon's theorem

- Shannon's Theorem gives an upper bound to the capacity of a link, in bits per second (bps), as a function of the available bandwidth and the signal-to-noise ratio of the link.
- The Theorem can be stated as:

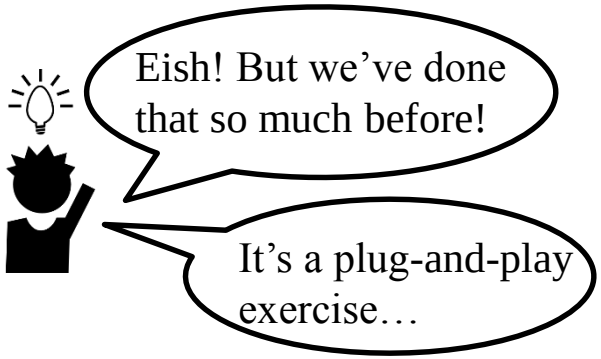
$$C = B * \log_2(1+ S/N)$$

where C is the achievable channel capacity,
B is the bandwidth of the line,
S is the average signal power and
N is the average noise power.

Example…

Example (Shannon's theorem)

- For a typical telephone line with a signal-to-noise ratio of 30dB and an audio bandwidth of 3kHz, what do we get as the maximum data rate? …



Do this quickly on paper or using your calculator… how many kbps is needed

Answer…

Using $C = B * \log_2(1+ S/N)$

S/N bB = $10\log_{10}(S/N)$ → $S/N = 10\text{^}3$

$C = 3000 * \log_2(1001)$

= 29901 bps

→ little less than **30 kbps**.

where C is the achievable channel capacity,
B is the bandwidth of the line,
S is the average signal power and
N is the average noise power.

Hint S/N is essentially given already but may need to de-log it

Based on example from http://www.inf.fu-berlin.de/lehre/WS01/19548-U/shannon.html

Limitations of Shannon's Theorem

- The theorem gives max channel capacity for an 'archetypal' communication link
- It is a bound on the maximum amount of error-free digital data (i.e. information) that can be transmitted in a specified bandwidth given a certain amount of noise interference

In reality there may be many factors to make it impossible to achieve this 'Shannon optimal' e.g. imperfect sensors, timer drift, etc.

BUT technically you could also get 'super Shannon' rates using compression etc.

Good Nyquist

Bad Nyquist

The Embedded Engineers Best Friend & Worst Nemesis

Nyquist's Theorem

- Nyquist's Theorem:
  - "Any signal can be represented by discrete samples if the sampling frequency is at least twice the bandwidth of the signal."
  - i.e. $f_{sampling} > 2 \times f_{signal}$

    where $f_{sampling}$ is sampling rate and f_{signal} is the maximum frequency of the signal you want to sample.

Further reading: http://whatis.techtarget.com/definition/Nyquist-Theorem , https://en.wikipedia.org/wiki/Nyquist%E2%80%93Shannon_sampling_theorem

Nyquist's Theorem

- A *bandlimited* analog signal that has been sampled can be perfectly reconstructed from an infinite sequence of samples if the sampling rate f_s exceeds $2f_{max}$ samples per second, where f_{max} is the highest frequency in the original signal.

  - If the analog signal does contain frequency components larger than $(1/2)f_s$, then there will be an aliasing error.
  - **Aliasing:** when the digital signal appears to have a different frequency than the original analog signal.

Difference between: Shannon and Nyquist's theorems

- **Nyquist's Theorem** deals with the concept of sampling rates
  - This is extremely important for embedded engineers, since the sampling needs to be twice as fast a the speed of the signal changes that you want to detect.
  - The general rule of thumb is that your sampling rate must be at least 2x the frequency of the signal you are sampling.
- **Shannon's Theorem** has to do with throughput
  - No cable or signal is perfectly clean. There is a lot of interference from other sources that cause noise on the line. What Shannon's theorem does is predicts the speed of the data that you can push through the cable before noise becomes too much of a factor.
    (Looking at this aspect may be useful in the ECE design project)

Summary: Nyquist is all about sampling. Shannon is all about noise. They are related as speed of data impacts both theorems.

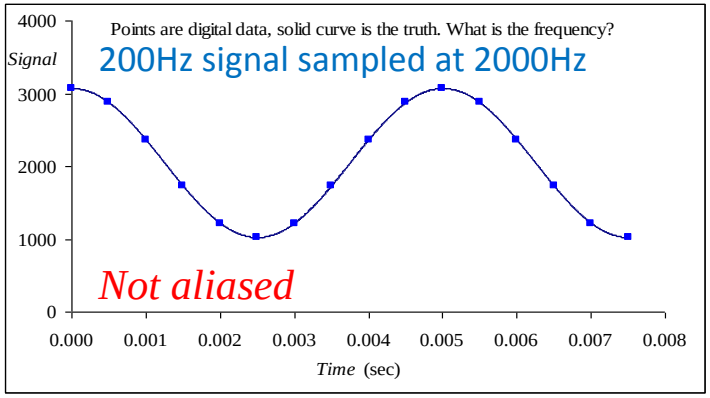
Shannon & Nyquist Relevant to ADCs

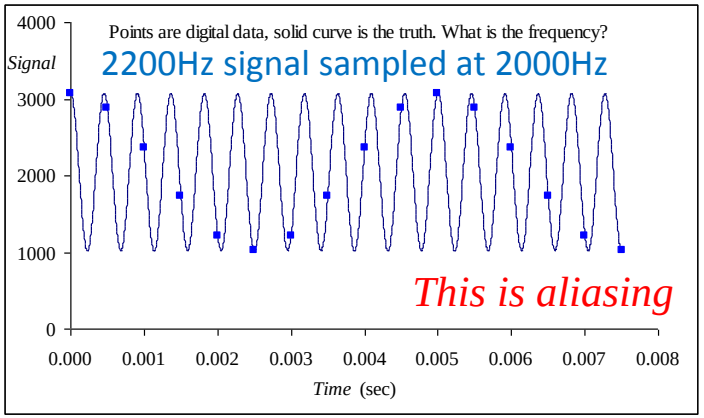
- These formulae are useful tools for planning what ADCs you need and what your system will be capable of in terms of sampling and utilizing channels
  - Shannon will tell you the most you can get out of your channel (excluding 'super Shannon')
  - Nyquist will tell you what frequencies you can expect to get out for a certain ADC
    (assuming the ADC is linear and pretty perfect, we will see later how ADCs may be imperfect which may mean you might not even be able to reliably sample $F_s/2$ frequencies)

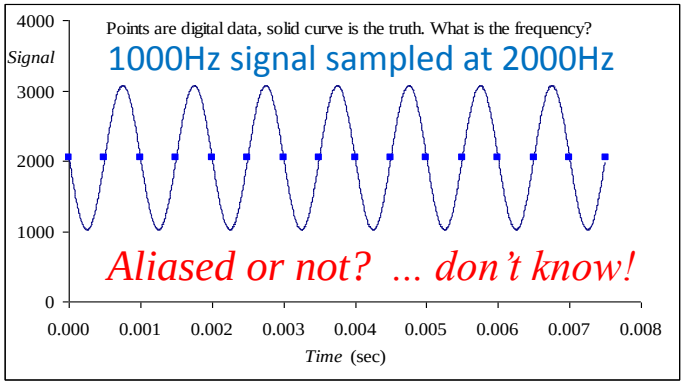
Sampling and aliasing

- Aliasing is an effect that causes different signals to become indistinguishable (or aliases of one another) when sampled.
- It also refers to the distortion or artefact that results when the signal reconstructed from samples is different from the original continuous signal.

Some examples…

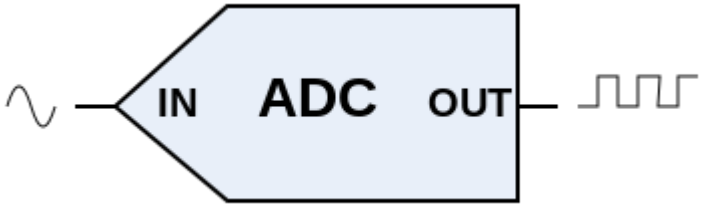






https://en.wikipedia.org/wiki/Aliasing

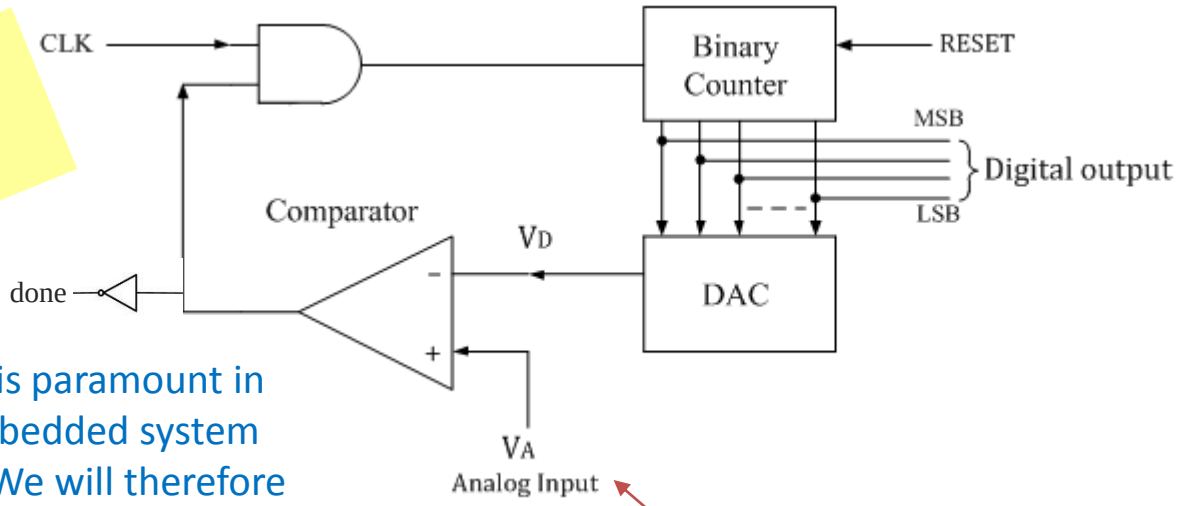
ADCs



Analogue to Digital Convertors (ADCs)

Conceptual ADC Operation

I won't waste time on this slide – we've seen similar to this already!



The ADC is paramount in many embedded system designs. We will therefore delve into some detail of ADC issues.

Sampled analogue signal

Notes about this conceptual representation:

- *While this conceptual ADC could be implemented is not such a practical architecture! It is very inefficient and only really useful to illustrate the operation of an ADC*
- *In practice, sample-and-hold (SAH) circuit prevents the ADC input, a time-varying analogue voltage x(t), from changing during the conversion time (see earlier slides on SAH).*
- *The counter stops counting when its output, which is converted into a voltage by the DAC (digital-to-analogue converter), equals x(t), indicated by comparator. The counter output is then the ADC output code corresponding to x(t).*

Conceptual ADC Operation

- Example DAC output voltages from the changing binary counter values (i.e. the DAC's transfer function) shown in the table below:

TABLE 7-1
Digital-to-Analog Converter Transfer Function

ADC Output Code (Q2, Q1, Q0)	V_{DAC} (volts)
000	–0.75
001	–0.50
010	–0.25
011	0.00
100	0.25
101	0.50
110	0.75
111	1.00

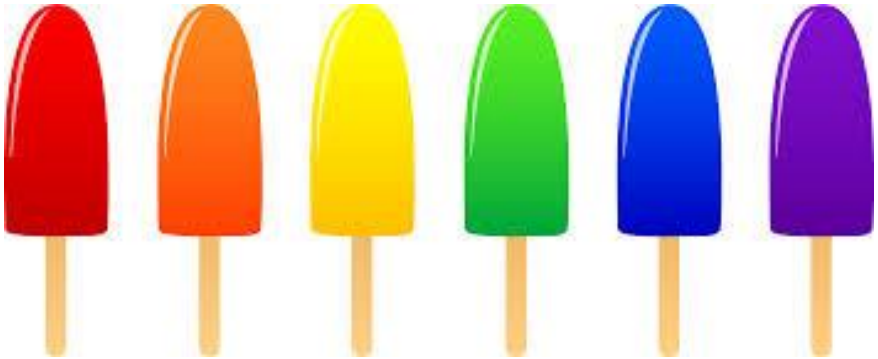
Basically you need to remember:

digital Output Codes corresponds to a input voltage and there will be metrics that described how these relate…

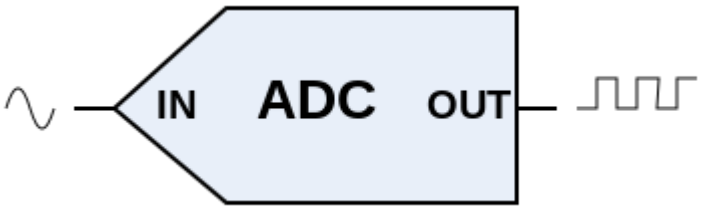
Notes about this conceptual representation:

In this conceptual design, the least significant bit (LSB), Q0, is the step size of 0.25 volts. This can also be quite an artificial assumption for a real system where the step size may fluctuate due to the physical properties of the system and environment.

ADCs Flavours



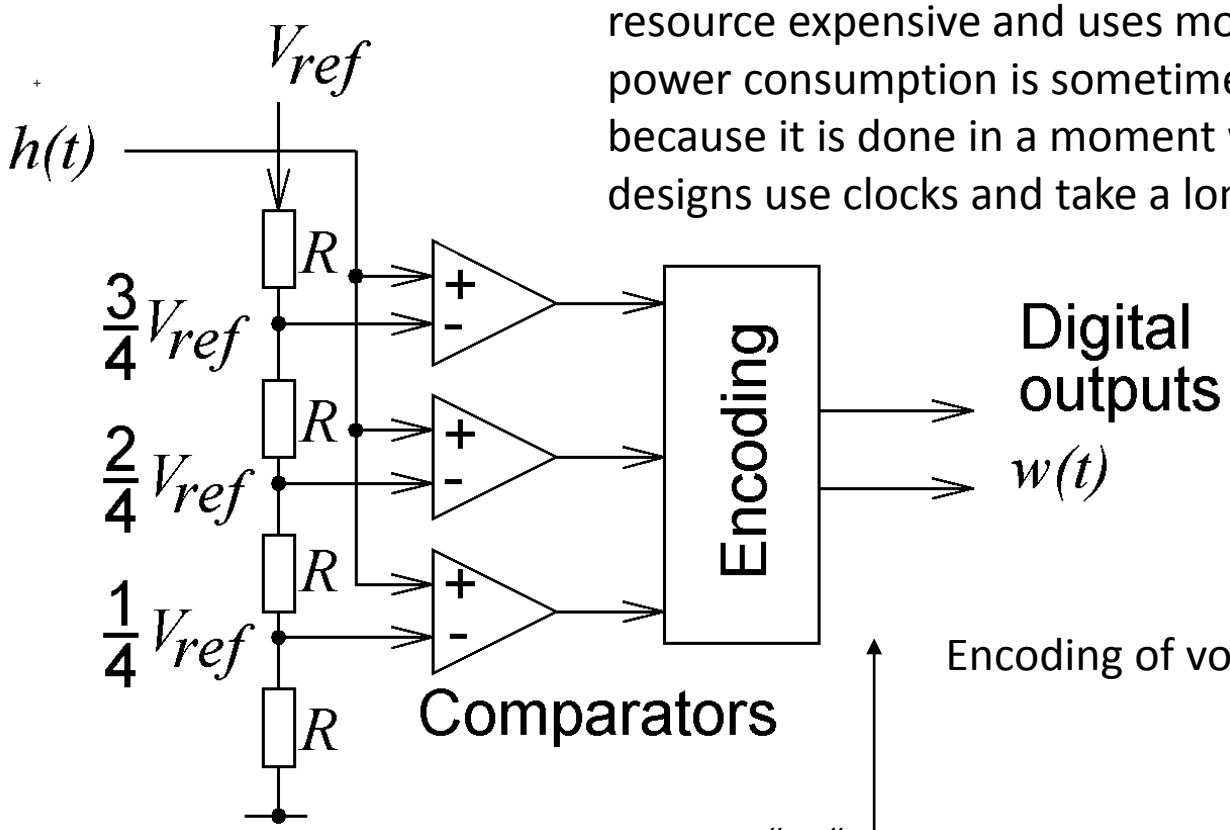
What flavour is your ADC?

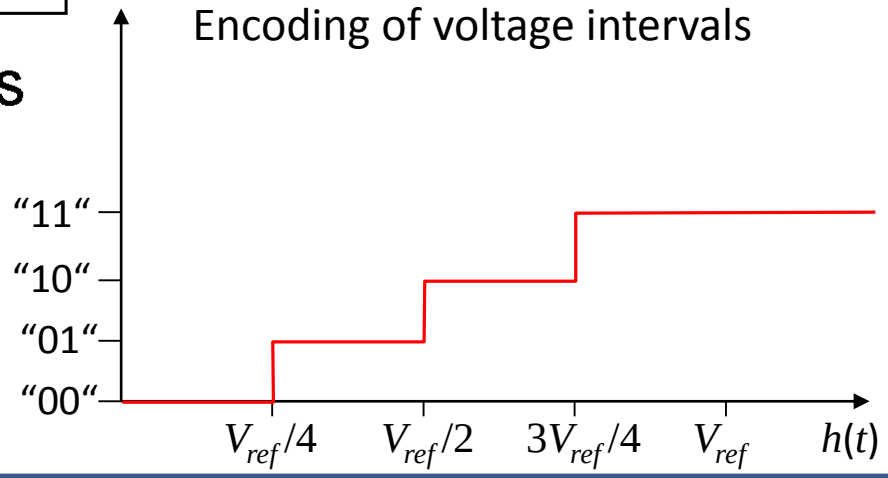


Analogue to Digital Convertors (ADCs)

Flash ADC converter

The Flash ADC is essentially the fastest but most resource expensive and uses most power. But the power consumption is sometimes not trivial because it is done in a moment whereas the other designs use clocks and take a longer time.





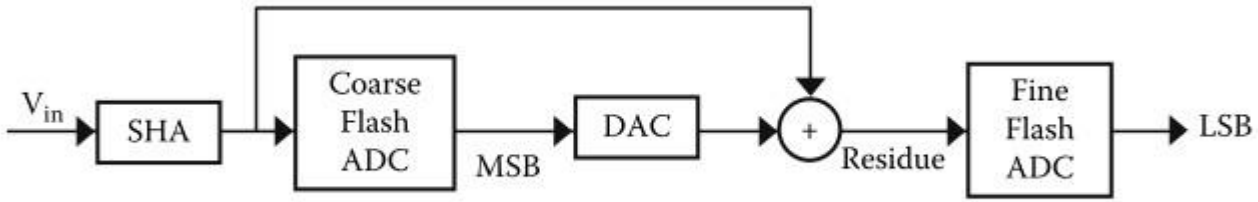
No decoding of $h(t) > V_{ref}$

Resolution and speed of Flash A/D-converter

- **Parallel comparison with reference voltage**
- Speed: $O(1)$
- Hardware complexity: $O(n)$
- **Applications**: e.g. in video processing

Multi-stage FLASH (NB: not pipelined)

- A multistage flash ADC uses a number of smaller flash ADCs to decide on a sample, but this isn't quite as fast as the pure flash



Example of 2-stage residual FLASH

Speed is depended on:

Course Flash + DAC + Residual + Fine Flash

which is about 1/3 the speed for one single big flash

But on the positive side:

single flash = 2^n comparators e.g. 8-bit → 256 comparators

s-stage flash = 2 * 2^(n/2) → e.g. 8-bit → 2 * 2^4 = 2*16 = 32 comparators

So for 8x the resources you get about 3x the speed

Pipelined FLASH

Generally this is the best bang for your buck… but

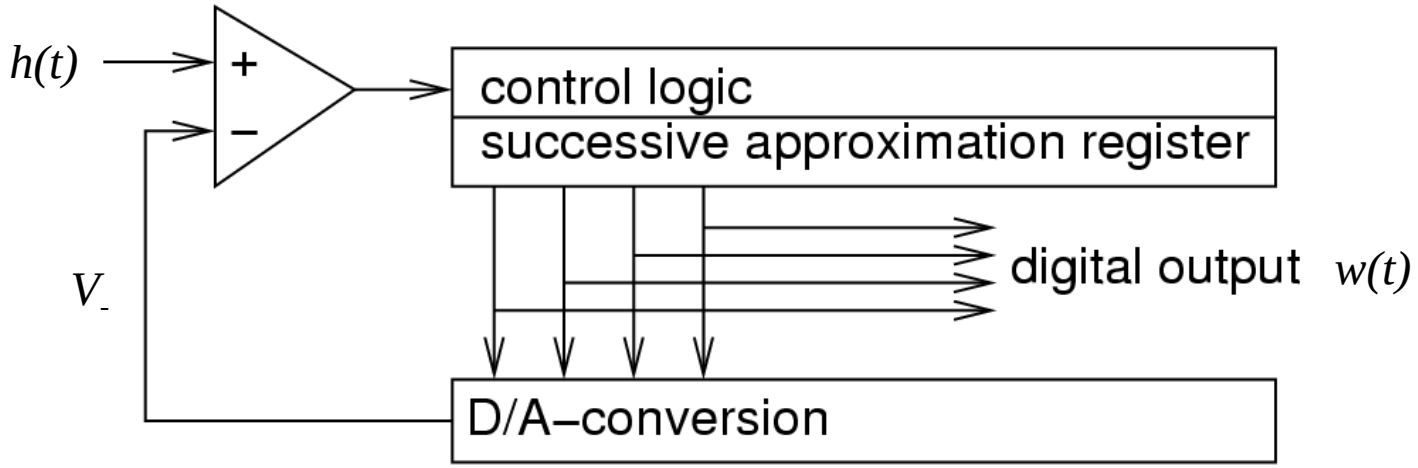
I'm the FLASH ADC
I might be… a little heavy,
But I'm FAST!

We're the Pipeline FLASH
We're almost as FAST
But we're a little cheeky!

But to understand the Pipeline Flash ADC
you need to first understand the SA-ADC
(i.e. the Successive Approximation ADC) which is up next…

The Pipeline Flash is essentially a combination of the two

Successive Approximation ADC (SA-ADC)



Key idea: binary search…

Set MSB='1' (all other ='0')

if too large:

reset MSB, set MSB-1='1'

if too large:

reset MSB-1, set MSB-2='1'

etc..

else if too small:

set MSB-1='1' …

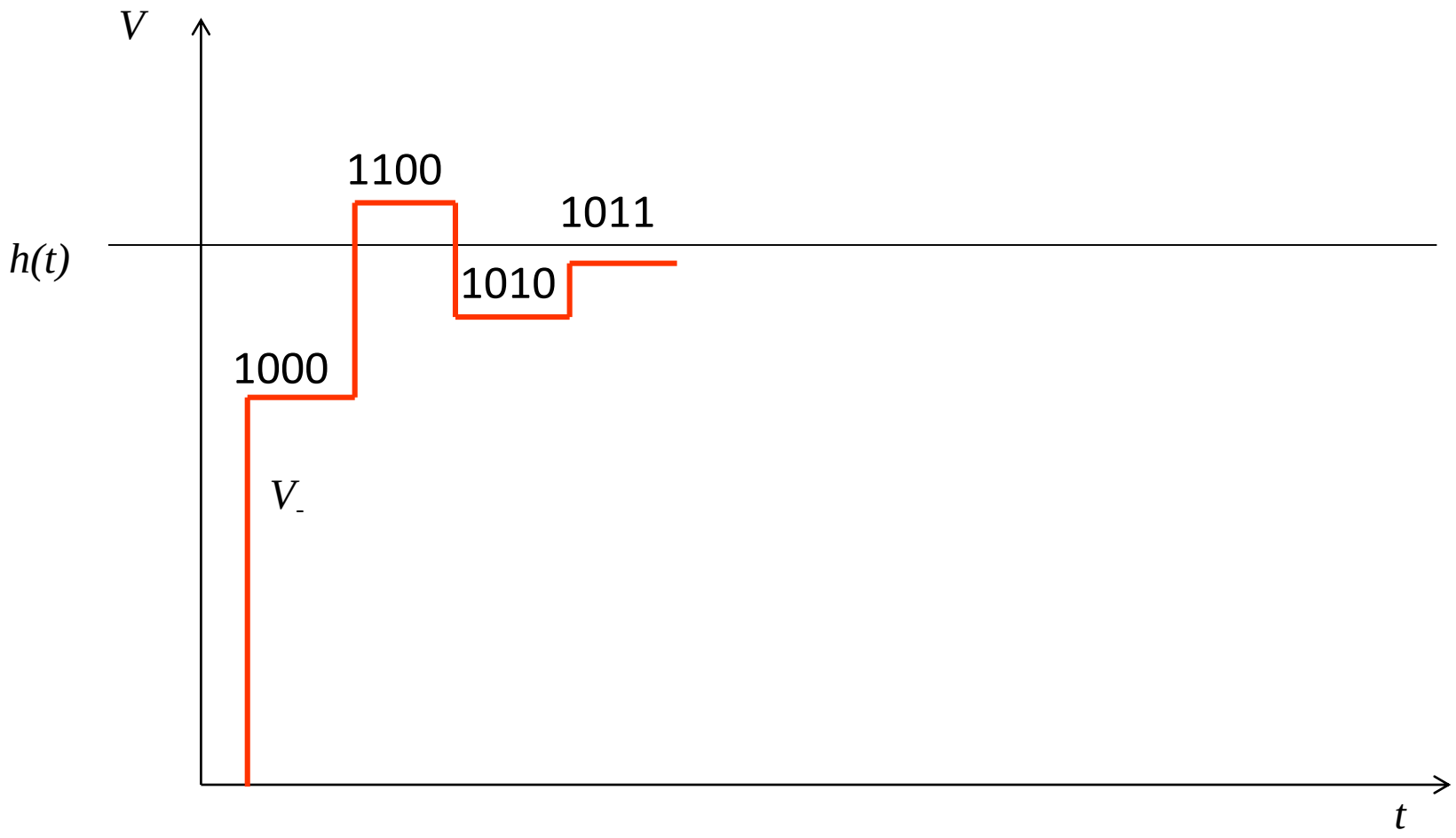
Speed: $O(\log_2(n))$

Hardware complexity: $O(\log_2(n))$

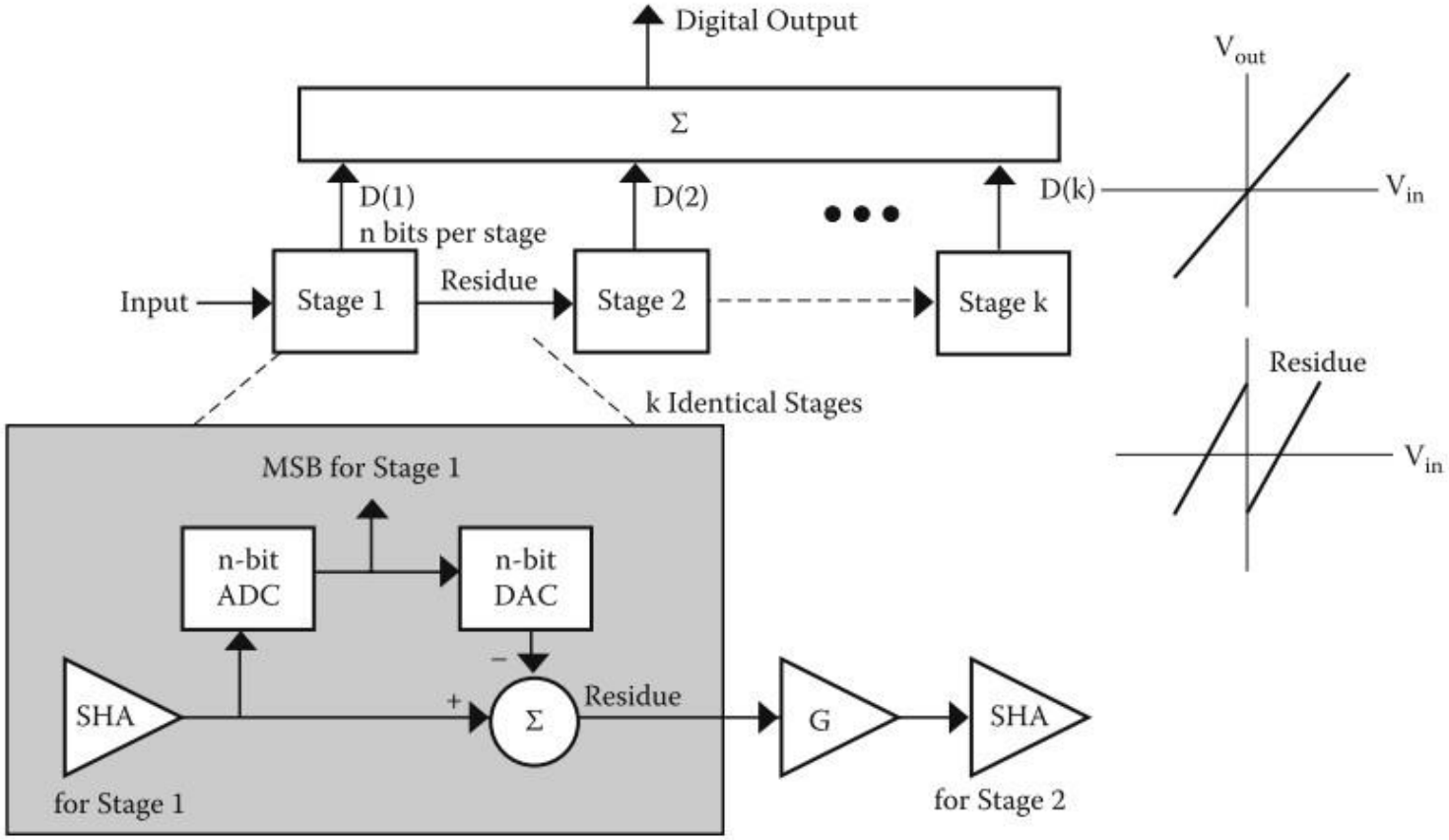
with n= # of distinguished voltage levels;

slow, but high precision possible.

Successive Approximation ADC (SA-ADC)



Pipelined FLASH



The conceptual operation isn't too tricky. Stage 1 runs first and does a coarse estimate. Then stage 2 runs and does a fine estimate … but simultaneously stage 1 start a course estimate on a new sample. The key aspect here is the SHA at each stage, essentially a smaller and smaller residual moves down the pipeline until the final LSB is decided. I think it's a really cleaver hybrid digital+analogue design!!

Quick Understanding Check

Q1. Will a SA-ADC probably use more comparators than a standard (i.e. ‘fat’) Flash?

Q2. Will a pipeline Flash probably be quicker than a SA-ADC? (explain your answer)

The Next Episode…

Lecture L21

ADCs cont. and DAC