SES Lab 2: Digital Signals

John Thompson
Electronics and Electrical Engineering Discipline

Week 5

With Grateful Thanks to **Prof Alan Murray!**

Week 5: Digital Signals

This week we will move on to look at Digital Signals:

- We will begin by discussing several basic types of digital signals and waveforms
- In Week 3, you designed an audio amplifier… now consider how we convert an audio signal into digital samples
- Also consider digitisation issues and artefacts

Digital Signals

- A microprocessor, such as the STM32 chip, understands digital signals and codes, based on voltages:
  - A low voltage (0V) can represent LOW or FALSE
  - A high voltage (e.g. 3.3V) can represent HIGH or TRUE

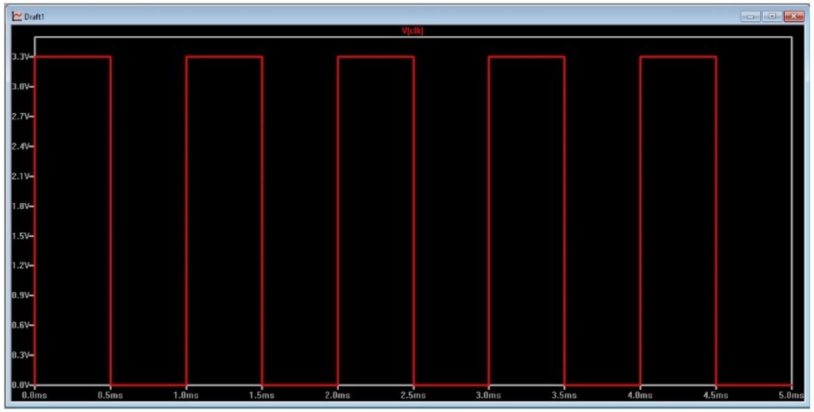
- The STM32 chip has many digital inputs/outputs that enable information to be communicated to/from the device

- Why are digital representations so widely used?
  - Results can be reproduced reliably on many devices
  - Designing digital processors is relatively easy and logical
  - Digital representations can processed very rapidly, allowing high speed operation

Clock Signals

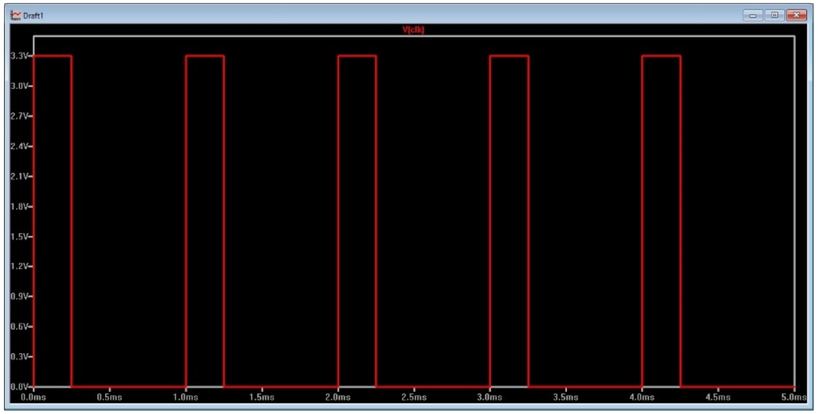
- In digital circuits, **square waves** are frequently encountered as clock signals or for conveying information
- Commonly clock signals have a 50% on/off duty cycle
- Usually the rising/falling edge of the clock signal will trigger an operation, such as:
  - Carrying out an instruction in a microprocessor
  - Updating a counter in the processor
  - Collecting a signal sample in an analogue-to-digital converter
  - Communicating data to/from a microprocessor

Example of a 1kHz, 3.3V Square Wave clock signal in LT-Spice

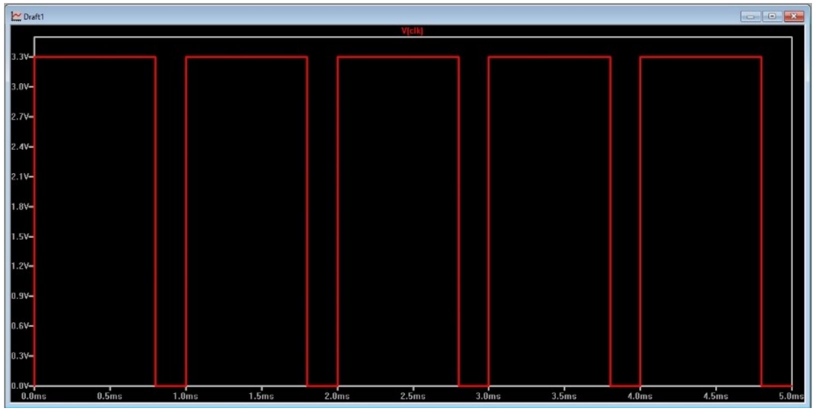


Pulse Width Modulation

- Digital signals can be used to control many different physical devices
- Sometimes it is useful to vary the ON/OFF duty cycle to vary the voltage received by a device such as robotic motor
- The longer the ON time, the more electrical energy that is delivered
- This is often called **pulse width modulation (PWM)** and most micro-processors have PWM digital lines

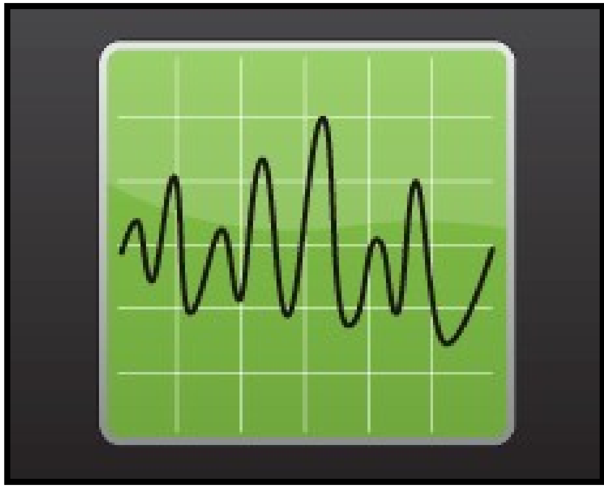


25% ON Duty Cycle Example

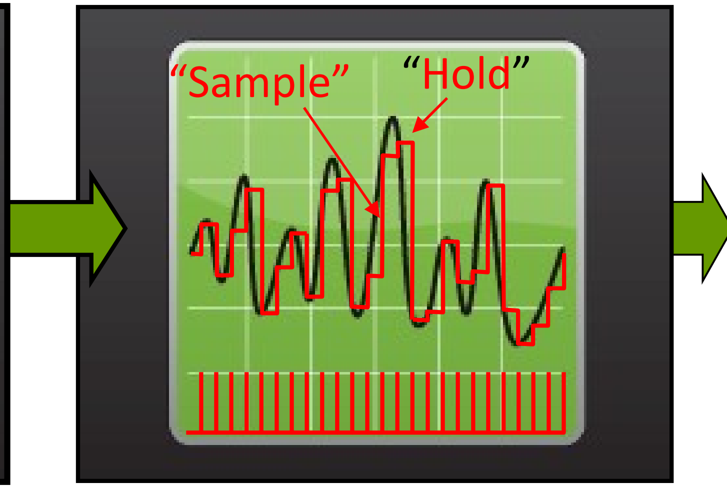


80% ON Duty Cycle Example

Analogue and Digital Signals - Overview

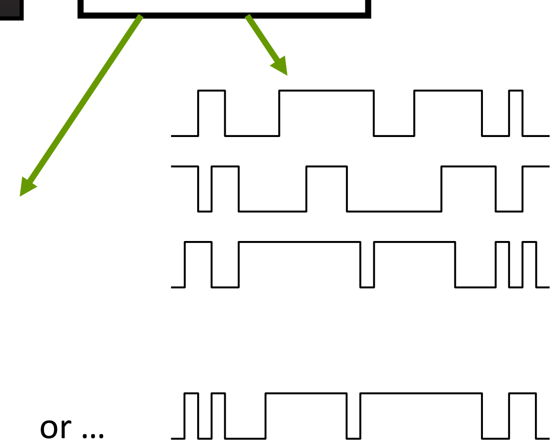


Analogue Signal



Sampled Analogue Signal

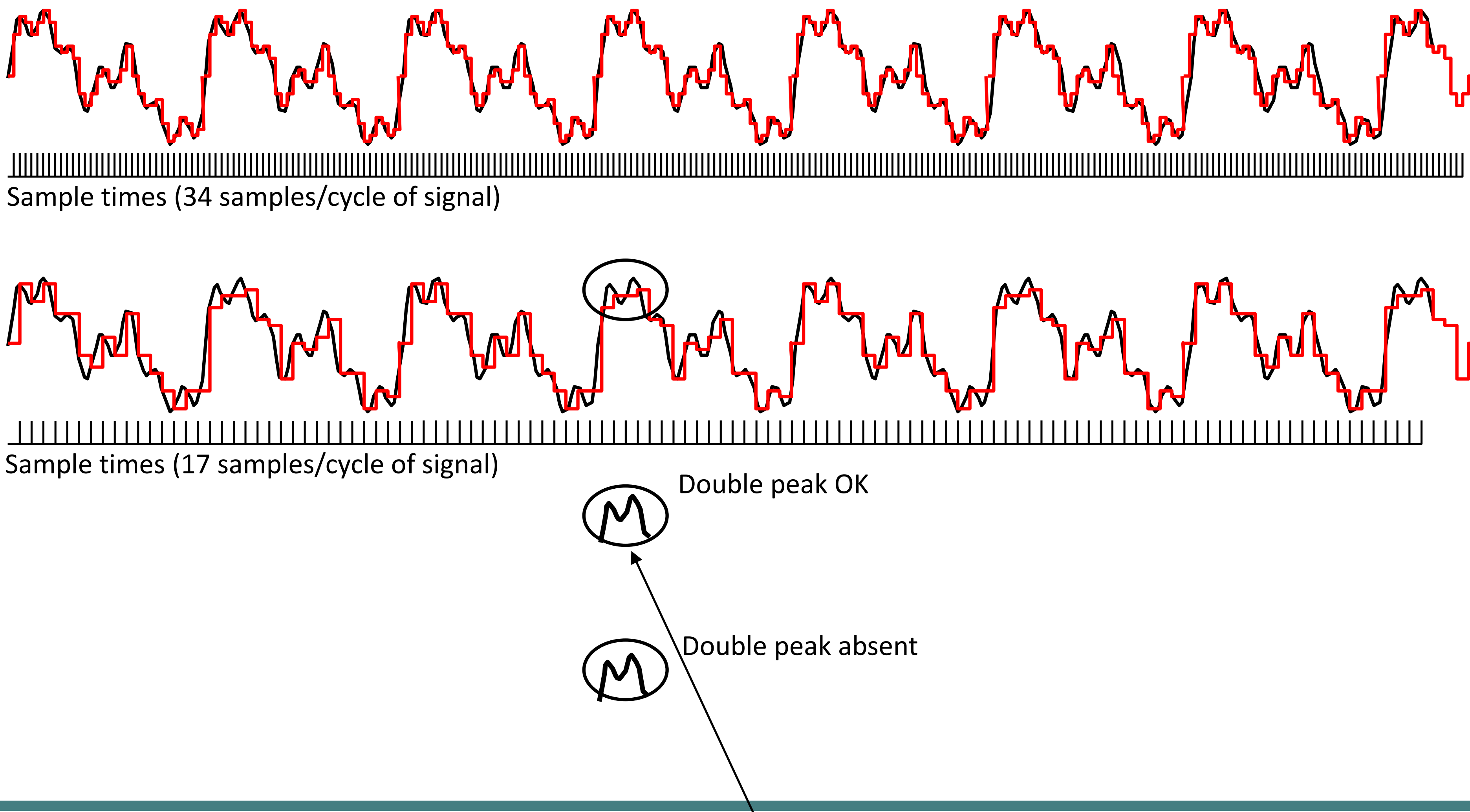
Analogue
To Digital
Converter
(ADC)



or …

Multi-bit digital output

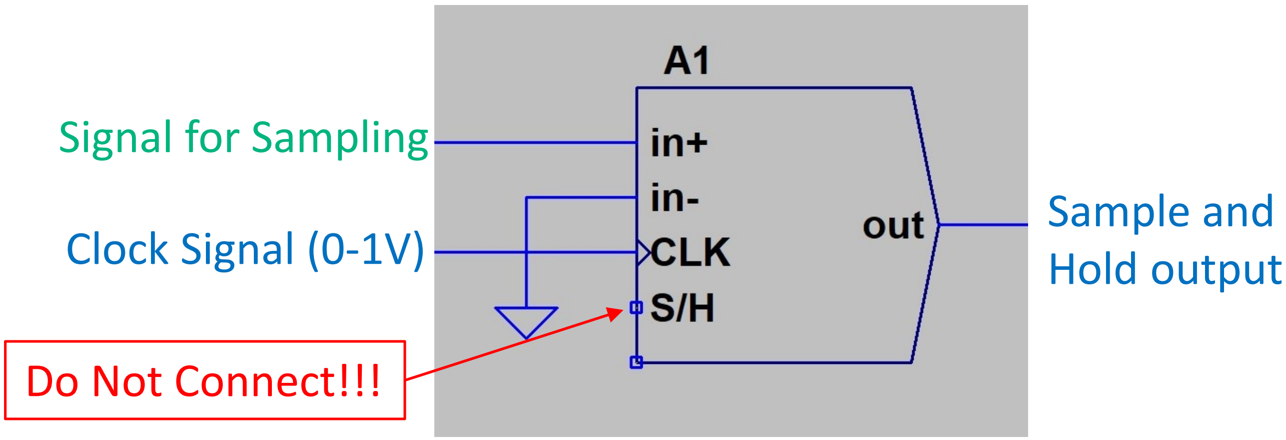
Sampling a musical signal …



More frequent samples = higher quality

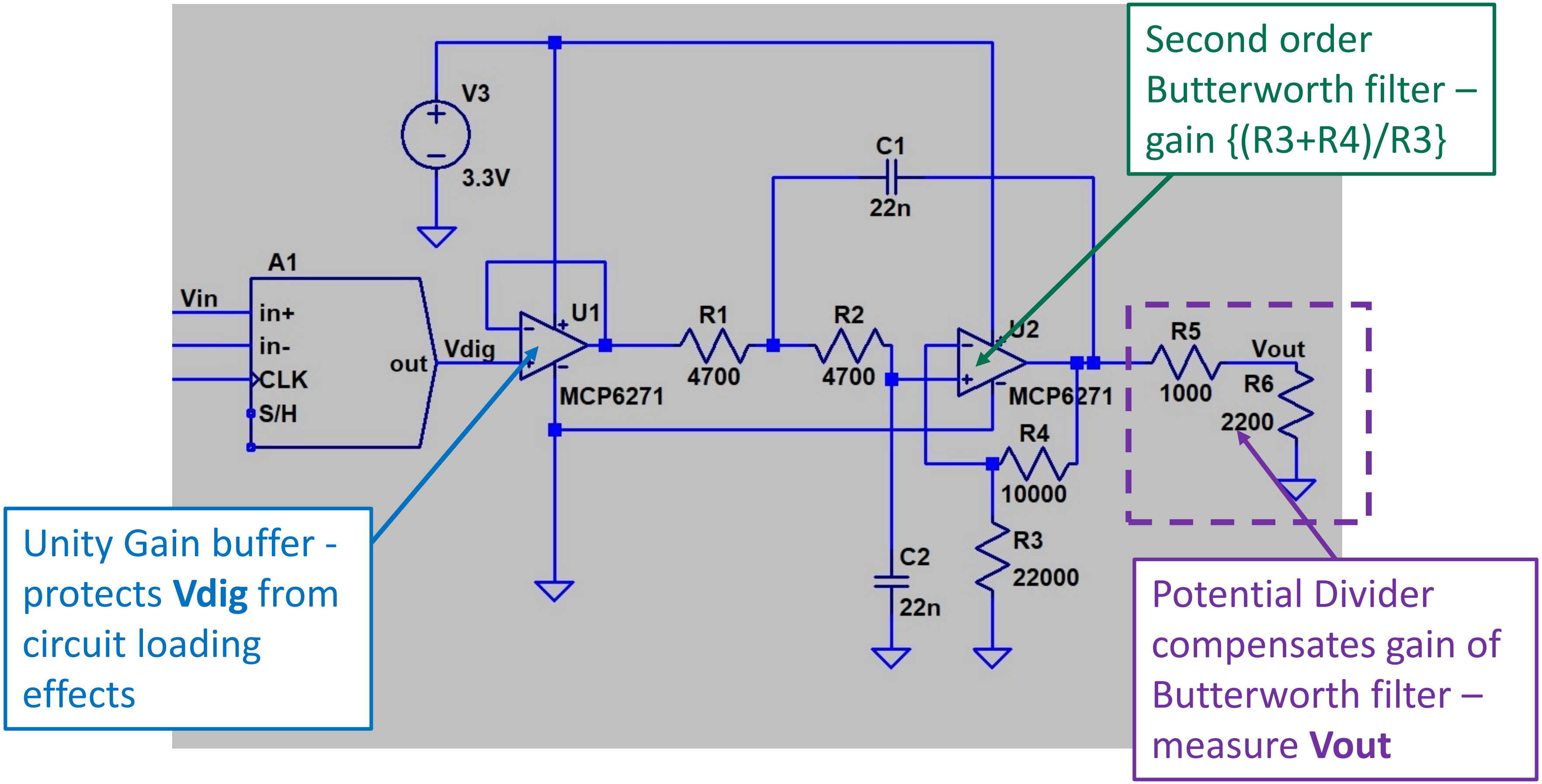
LT-Spice Sample and Hold

- LT-Spice contains a component “sample” that allows you to mimic a **sample-and-hold device**
- This is an important part of an Analogue-to-Digital Converter
- Connect up this device as shown below to operate
- A sample is recorded every time the clock goes high – a 0V/1V square wave should be sufficient to drive the device



Reconstruction Filter for 8 kHz Sampler

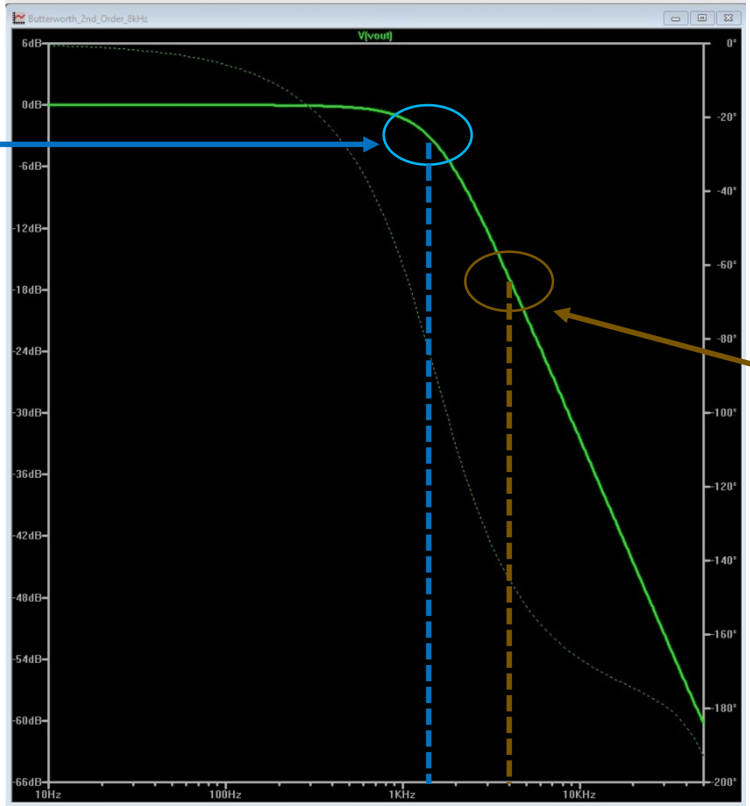
- **Second order** Butterworth low pass filter smooths the square pulses of the sample-and-hold, to reconstruct the original signal



Reconstruction Filter Freq Response

- **Frequency response** for second order Butterworth filter
- Sample at 8 kHz, so the maximum frequency (Nyquist) is 4 kHz

3dB Cutoff freq is about 1.45 kHz

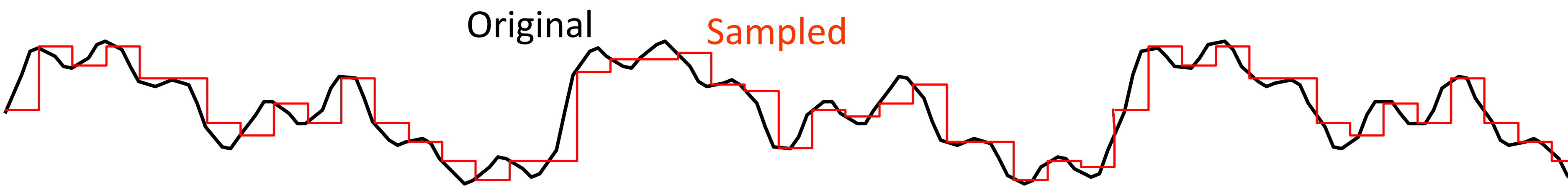


Gain at 4 kHz (Nyquist) is -17 dB

Reconstruction filter removes high frequency comp-onents > 4 kHz to allow the original signal to be seen

Now Work Through the Questions in **Worksheet 1**

Sampled – now digitise



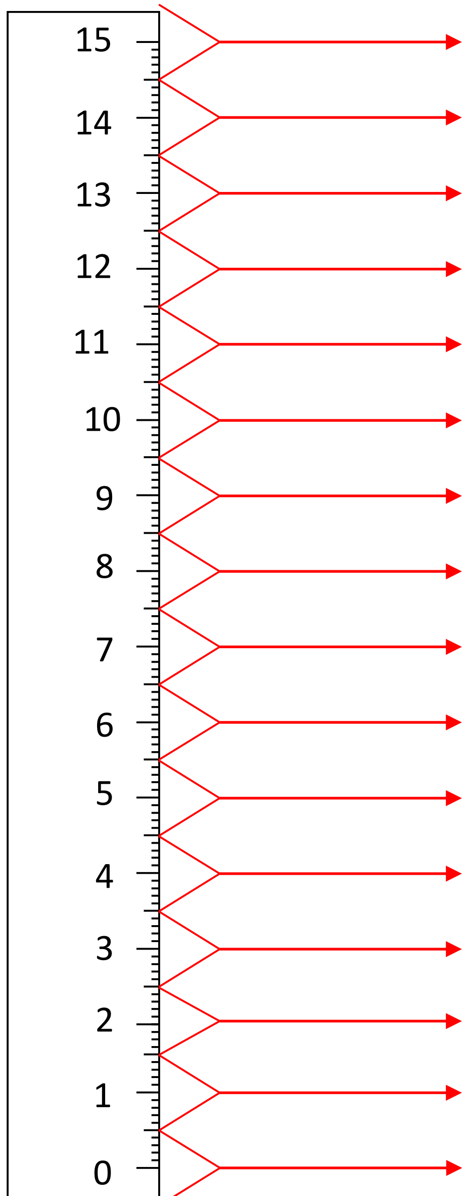
Sampled and digitised

101111001001000010010010010010001001001001000100 ...
100110100010010010001001011010010010000100100100 ...
001001001000100101110010010001010010010000100100100 ...

10011001010010010000010010010001001011100010010010 0 ...

Analogue-Digital

Reproducer Values		4-Bit Binary Patterns
15	=	1111
14	=	1110
13	=	1101
12	=	$\boxed{1100}$
11	=	1011
10	=	1010
9	=	1001
8	=	1000
7	=	0111
6	=	$\boxed{0110}$
5	=	0101
4	=	0100
3	=	0011
2	=	0010
1	=	0001
0	=	0000



	Decision Boundaries
1111	= 14.5→15.5
1110	= 13.5→14.5
1101	= 12.5→13.5
1100	= 11.5→12.5
1011	= 10.5→11.5
1010	= 9.5→10.5
1001	= 8.5→9.5
1000	= 7.5→8.5
0111	= 6.5→7.5
0110	= 5.5→6.5
0101	= 4.5→5.5
0100	= 3.5→4.5
0011	= 2.5→3.5
0010	= 1.5→2.5
0001	= 0.5→1.5
0000	=-0.5→0.5

Granular Region
Samples between -0.5 and 15.5

Overload Region:
Samples below -0.5 or above 15.5

Quantisation in MATLAB

- MATLAB allows us to implement the effect of analogue-to-digital conversion using the function **quantiz( )**
- To use this function we need to specify three things:
  - The data samples that we plan to quantise, stored in a vector
  - The reproducer values that the quantiser uses to represent each value
  - The decision boundaries between adjacent reproducer values
  - This function outputs the chosen reproducer values for our data
- Slide 11 shows that the reproducer values and decision boundaries have a regular structure
  - Use the MATLAB **colon operator** *a:b:c* to create these values
  - The value *a* is the first value, *b* is the step value and *c* is the final value
  - Example: to create the reproducer values in Slide 11 we can use the code: 0:1:15
  - **Overload Region:** Two values needed to tell MATLAB what to use below minimum boundary (-0.5) and above the maximum one (15.5)
  - Here we just repeat the minimum/maximum reproducer values

Now Work Through the Questions in **Worksheet 2**