Analogue to digital Conversion 2

- **Uniform quantisation** $\xrightarrow{\text{quantisation}}$ (dividing signal into 2^N equal intervals) (Amplitude Area)

- **N-bit Uniform Quantisation** (step 3)

- divides voltage into 2^N equal intervals with quantization step

Ranges on Axis

$$\boxed{\Delta = \frac{(V_{max} - V_{min})}{2^N}}$$ quantization step

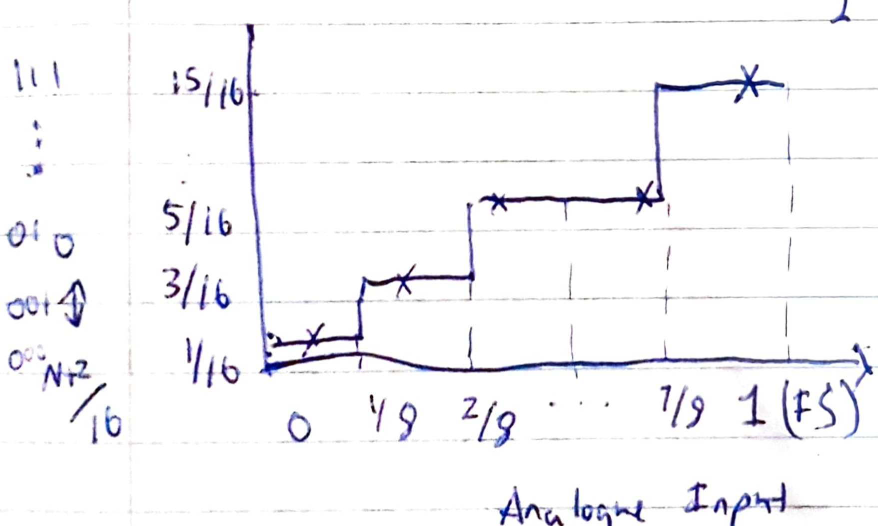
- Voltage interval $[k\Delta, (k+1)\Delta]$, $k = 0, 1, \dots, 2^N - 1$

$$\boxed{\text{- Discrete voltage} = \frac{(k + (k+1))\Delta}{2}}$$

- encode input voltage using N-bits
- Max quantisation error amplitude = $\boxed{\Delta/2}$

3-bit Uniform Quantiser

$2^3 = 8$



Since $N = 3$

$\approx 2^{-4} = \frac{1}{16} > 5\% \approx \frac{1}{20} = 5\%$

$2^{-(N+1)}$

Relative max amplitude ~~error~~ error

- **Quantisation Error**
  - Almost unavoidable in a ADC conversion
  - * - max error amplitude $\boxed{\Delta / 2}$

- * Relative maximum amplitude error $\boxed{= 2^{-(N+1)}}$

- * Quantisation noise power = $\boxed{\Delta^2 / 12}$

- **Flash and Sigma-Delta quantisers**
- **Flash Quantiser**
  - voltage ladder of $\boxed{2^N}$ Simultanious intervals
  - use 2^N comparaters to find the quantisation interval.
  - V_{in} is a single step
  - fast simple (could require a huge number of comparators) (diagram 4 for more info on diagram)

• Successive-Approximation Quantiser

(example) • 3-bit Uniform quantisation

• $V_{max} = 1$ V, $V_{min} = 0$ V • $\Delta = 1/8$

• Analogue input = 0.425V

currently 000

1. control unit starts with a bin code $100 \equiv 4 \Rightarrow 100$
DAC outputs $0.5^V (= 4\times\Delta) > 0.425^V$ $= 4$
first bit = 0

$4 \times \Delta = 4 \times \frac{1}{8} = 0.5$ first bit = 0

2. 010 = 2

$= 2 \times \frac{1}{8} = 0.25 < 0.425$ ∴ second bit = 1

3. $\underbrace{01}_{3}1$ $= 3 \times \frac{1}{8} = 0.375 < 0.425$ ∴ third bit = 1

∴ control unit outputs the quantisation result of 011 if $N=3$ $2^3 = 8$ ← spaces needed

• Steps view

1. get all relevant information
2. Start from 0 0 0 0 and add a 1 to every bit ~~xx~~ to find the value of that step.

e.g. 2^3 2^2 2^1 2^0 ■ = Bin code found
■ = active / current bin step

(bin code) 4 = 1 0 0 0

12 = 1 1 0 0

10 = 1 0 1 0

Then in every loop if (bin code × Δ) < Analogue input
bit = 1
else
bit = 0

3. end

output quantisation result
(xxxx)