SE3910 – REAL TIME SYSTEMS

NyQuist Criteria, Sampling Rates, and Programming Language Optimization

(OTW Known as Potpourri)

ROADMAP

- Today
  - Finish up Physical Media
  - More about coding
- Monday
  - Embedded Code Quality and MISRA
- Wednesday
  - Real Time Software Qualities
- Friday
  - Structured Design and Data Flow Diagrams

OBJECTIVES

- Calculate the maximum data rate of a channel under both noiseless and noisy signal conditions
- Explain the Nyquist theorem related to sampling
- Calculate the minimum sampling rate necessary to transmit a signal using the Nyquist Theorem
- Explain the relationship between the number of bits and quality when sampling a signal
- Critique the Java language for usage in Real Time Systems
- Optimize source code using well known optimization techniques, such as
  - Repeated calculations
  - Constant folding
  - Loop invariance removal
  - Induction variance
  - Loop unrolling
  - Loop jamming

WHAT IS THE MAXIMUM RATE OF A CHANNEL?

Maximum Data Rate of a Channel

Removed high-frequency

Nyquist (1924) studied the problem of data transmission for a finite bandwidth noiseless channel. Nyquist states:

1. If a signal has been run through a low-pass filter of bandwidth H, the filtered signal can be completely reconstructed by making $2H$ samples.

   2 H samples

   The important corollary to Nyquist's rule is that sampling more often is pointless because the higher frequencies have been filtered out.

2. If the encoding signal method consists of V states:

   maximum data rate $= 2H \log_2 V$ bps

What's the maximum data rate over phone lines? Going back to our telephone example, Nyquist's theorem tells us that a one-bit signal encoding can produce no better than: $2 \times 3000 \times \log_2 2 = 6000$bps.

But there is a catch. In practice, we don't come close to approaching this limit, because Nyquist's rule applies only to *noiseless* channels.

- Henry Nyquist
- Nyquist Theorem

# of Levels

- $maximum\ data\ rate = 2H\log_2 V\ \frac{bits}{sec}$

Noiseless scenario

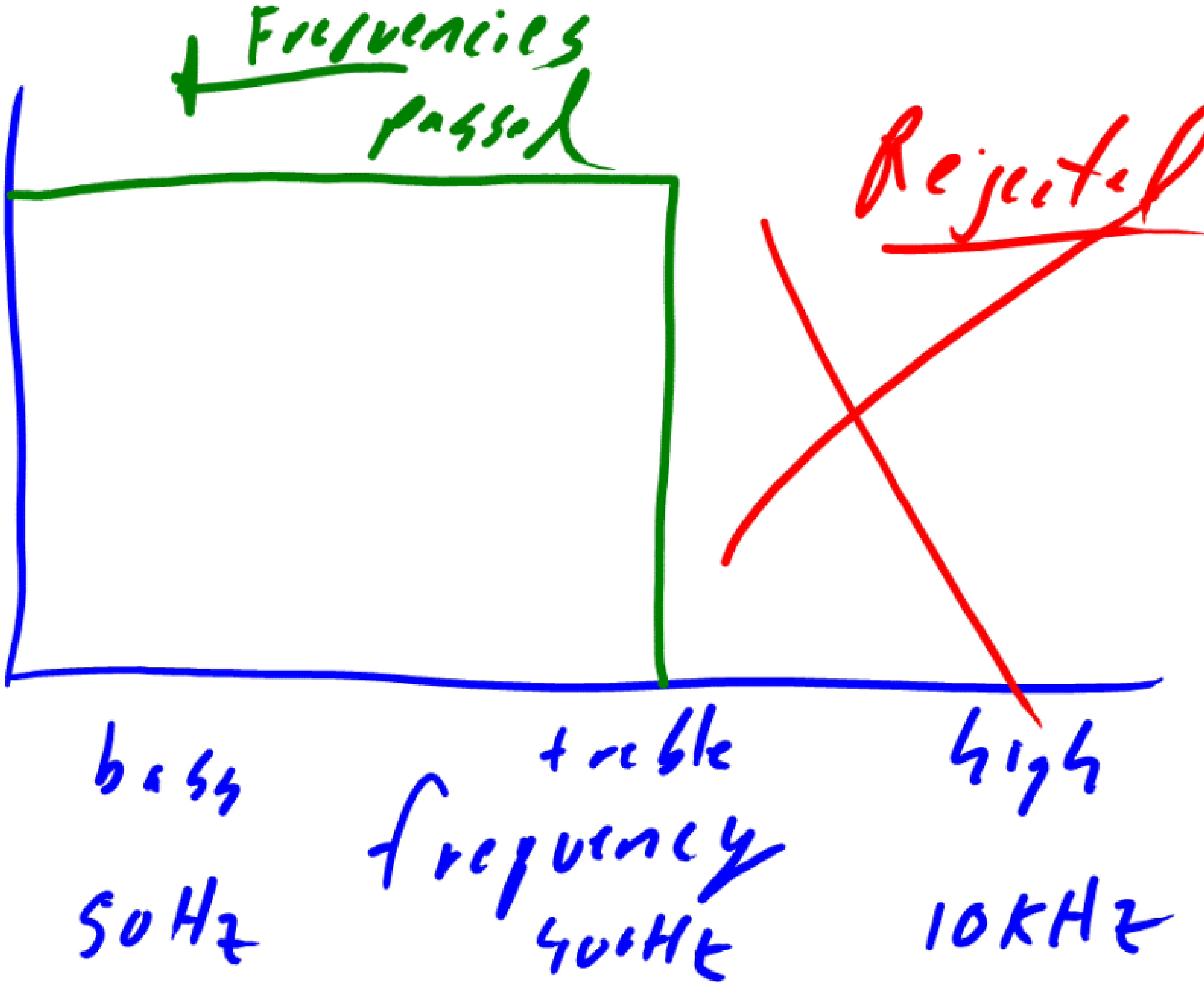
2H log

1000

SE3910 REAL TIME SYSTEMS

$2H \log_2 8 \Rightarrow 6000$

$2H \log_2 2 \Rightarrow 2000$



Frequencies passed
Rejected
bass
treble
high
frequency
50Hz
4000Hz
10KHz

WHAT IS THE MAXIMUM RATE OF A CHANNEL?

- Henry Nyquist
- Nyquist Theorem

Maximum Data Rate of a Channel

Nyquist (1924) studied the problem of data transmission for a fine bandwidth noiseless channel. Nyquist states:

1. If a signal has been run through a low-pass filter of bandwidth H, the filtered signal can be completely reconstructed by making $2H$ samples.

   The important corollary to Nyquist's rule is that sampling more often is pointless because the higher frequencies have been filtered out.

2. If the encoding signal method consists of V states:

   maximum data rate $= 2H \log_2 V$ bps

What's the maximum data rate over phone lines? Going back to our telephone example, Nyquist's theorem tells us that a one-bit signal encoding can produce no better than: $2 \times 3000 \times \log_2 2 = 6000$bps.

But there is a catch. In practice, we don't come close to approaching this limit, because Nyquist's rule applies only to *noiseless* channels.

1M

- $maximum\ data\ rate = 2H\log_2 V\ \frac{bits}{sec}$

2H => 1000

SE3910 REAL TIME SYSTEMS

1024 levels => 10

2 * 1000 * 10 => 20000

20 => 40000

SIGNAL TO NOISE RATIO

- Noise types → Electrical
  - Thermal noise – Noise from the agitation of electrons in a conductor
  - Intermodulation Noise – Noise which results from different frequencies sharing the same medium
  - Crosstalk Noise – Noise from coupling between wires
  - Impulse noise – Sharp transient noise, such as lightening

$$S/N_{(db)} = 10 \log_{10} \frac{signal\ power}{noise\ power}$$

In decibels

WHAT IS THE MAXIMUM RATE OF A CHANNEL WITH NOISE?

H	SN (db)	Maximum Number of Bits per secon
3000	0	3000
3000	10	10378.29486
3000	20	19974.63445
3000	30	29901.67878
3000	40	39863.56993
3000	50	49828.9647
3000	60	59794.71004
3000	70	69760.49043
3000	80	79726.[illegible]7432

Max data rate w/ any # of levels

- Claude Shannon (1948)

30 Db => 1000

- $maximum\ number\ of\ \frac{bits}{second} = H\log_2\left(1+\frac{S}{N}\right)$

Note: S/N Usually given in DB. To use, must convert using formula 10^DB/10

H => 1000

$1000 * \log(1+\frac{S}{N})$

$1000 * \log_2(1001) \Rightarrow \approx 10{,}000$

WHAT IS THE DIFFERENCE?

Shannon's Theorem

Shannon's theorem gives the maximum data rate for channels having noise (e.g., all real channels). Shannon's theorem states that the maximum data rate of a noisy channel of bandwidth H, signal-to-noise ratio of S/N is given by:

max data rate = $H \log_2(1 + S/N)$

Note: the signal to noise ratio S/N used in Shannon's theorem refers to the ratio of signal power to noise power, not the ratio expressed in dbs (decibels). Unlike Nyquist's limit, Shannon's limit is valid regardless of the encoding method.

Let's consider a phone line again. A typical value for the S/N ratio for phone lines is 30db.

$S/N = 10^{\frac{S/N_{(db)}}{10}} = 10^3 = 1000.$

Thus, the maximum data rate $= 3000 \times \log_2(1 + 1000) \approx 30,000$ bps.

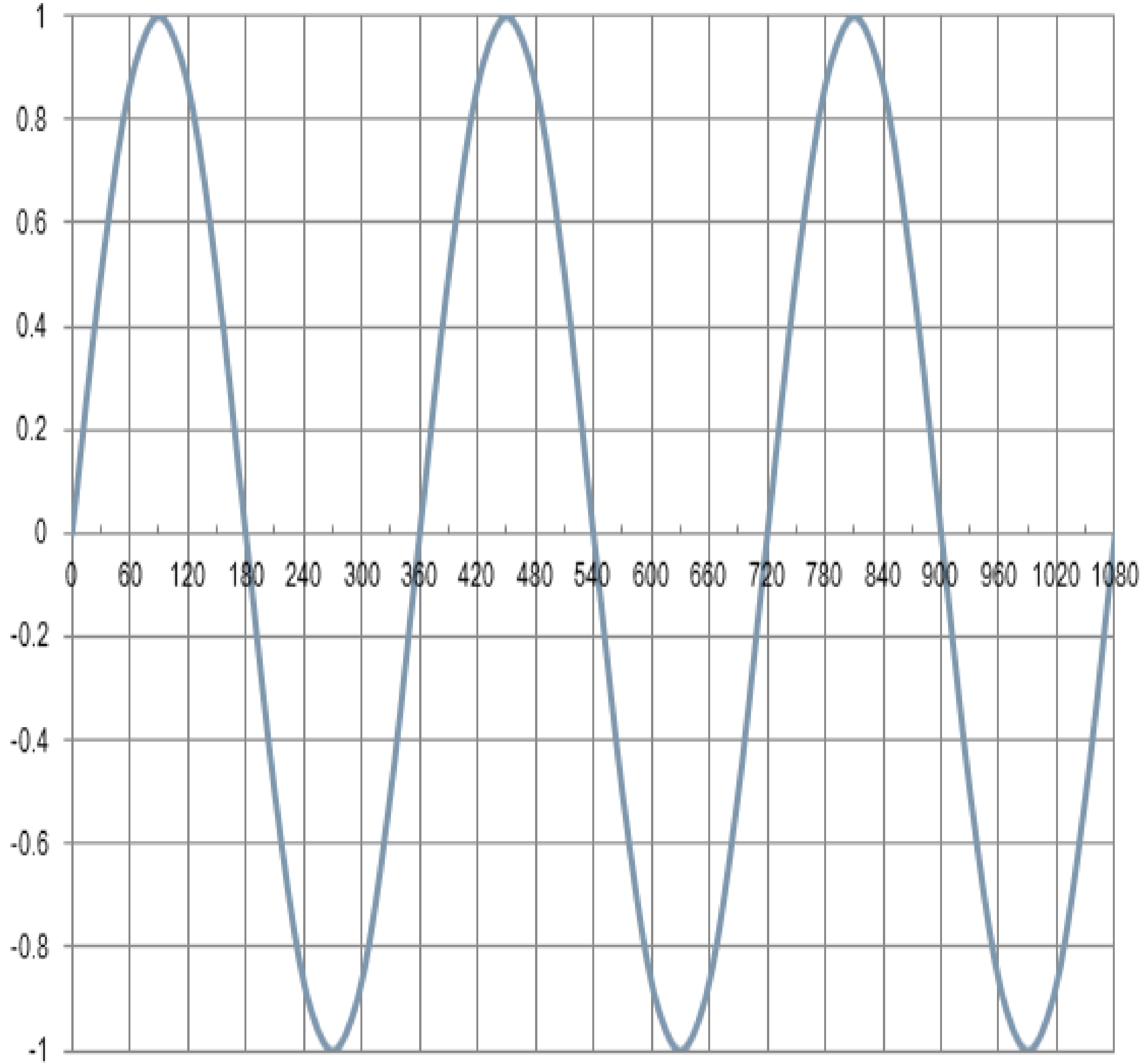
But wait — don't modems deliver data at 38.4 and 56 kbps? Many modem companies advertise that their modem deliver higher data rates, are they lying? Not necessarily. Read the fine print. Most likely, the modem uses data compression, and the high data rate is achieved only with text data.

WHAT'S THE DIFFERENCE (SOME MORE)

Let's summarize what Nyquist and Shannon say:

- Nyquist: sampling a received signal more frequently than $2H$ (where H is the bandwidth of the channel) is pointless.
- Nyquist: maximum data rate = $2Hlog_2V$ bps, where H is the bandwidth of the channel, and V is the number of distinct encodings for each baud. This result is a theoretical upper bound on the data rate *in the absence of noise*.
- Shannon: maximum data rate = $Hlog_2(1 + S/N)$, where S/N is the ratio of signal power to noise power. Note that Shannon's result is independent of the number of distinct signal encodings. Nyquist's theorem implies that we can alway increase the data rate by increasing the number of distinct encodings; Shannon's limit says that is not so for a channel with noise.

SAMPLING THEOREM



SAMPLING THEOREM

- Nyquist Criterion
- => fs > 2B

48000 KHz 4 bits/sample

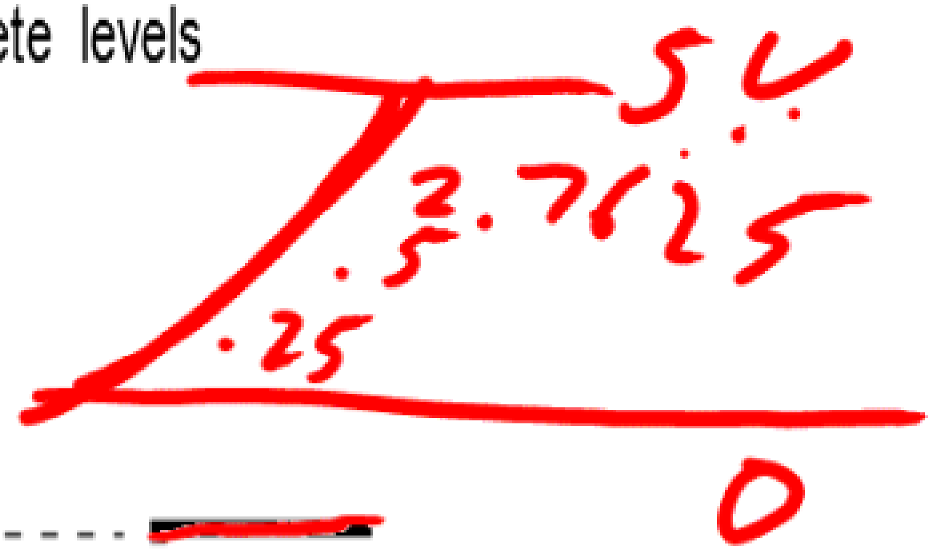
- Nyquist Rate => 2B
- Audio Sampling:

Sample rate	Quality level	Frequency range
11,025 Hz	Poor AM radio (low-end multimedia)	0-5,512 Hz
22,050 Hz	Near FM radio (high-end multimedia)	0-11,025 Hz
32,000 Hz	Better than FM radio (standard broadcast rate)	0-16,000 Hz
44,100 Hz	CD	0-22,050 Hz
48,000 Hz	Standard DVD	0-24,000 Hz
96,000 Hz	High-end DVD	0-48,000 Hz

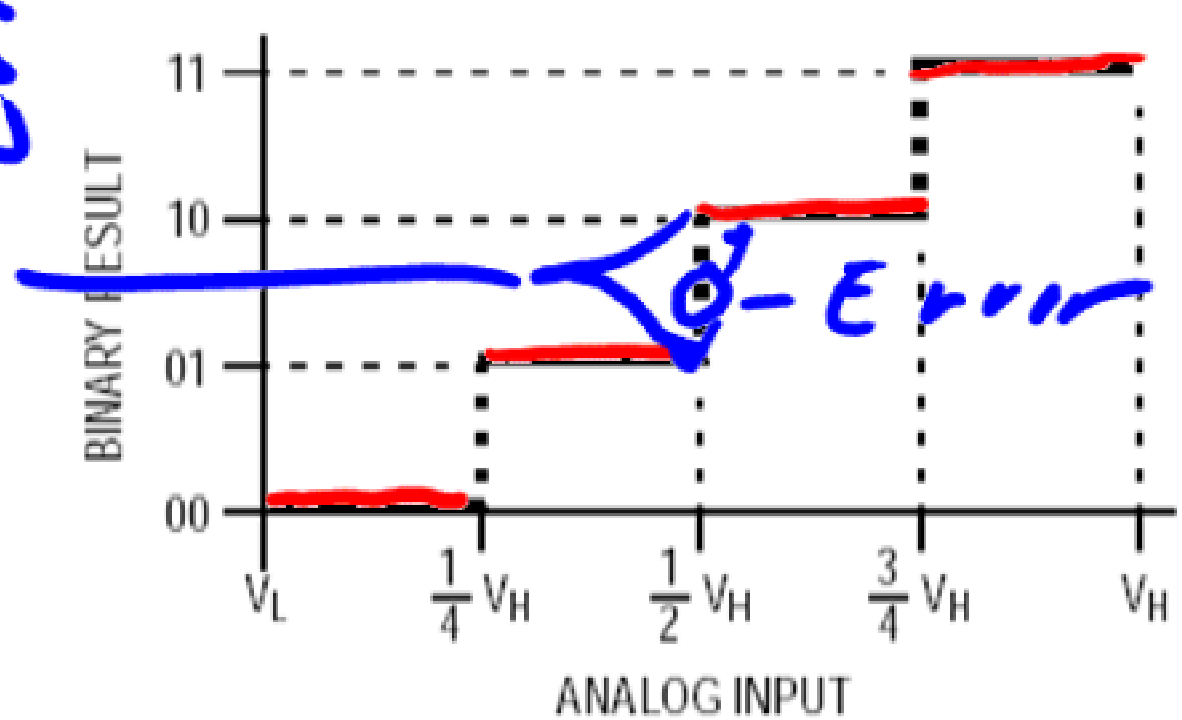
11 025 64bits/Sample

DISCRETIZATION?

- Consider an analog signal that will vary between two values – say 0 and V_H volts
- Discretization refers to the "levels" the ADC is able to resolve the analog signal to:
  - a 2-bit converter can resolve 4 different discrete levels



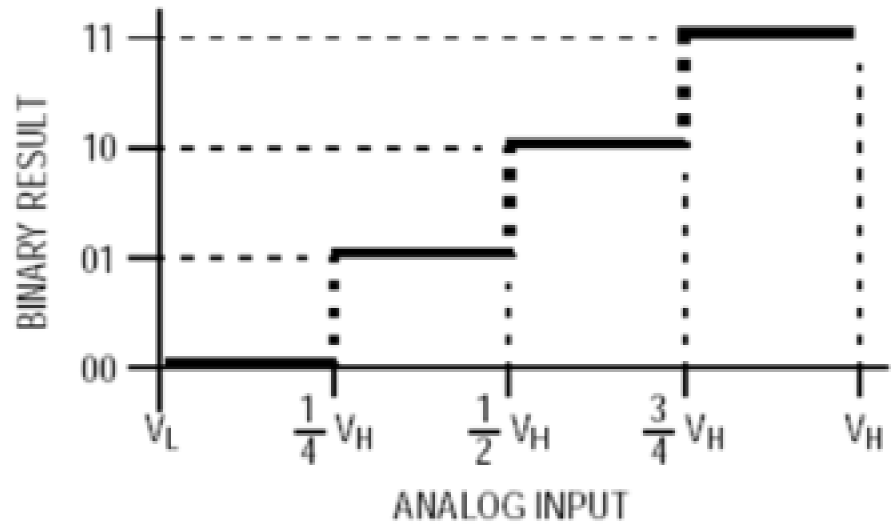
Gaussian Error



QUANTIZATION ERROR

Truncate signal ⇒ 3.25 ⇓ 3.00

Rounding ⇒ 3.00



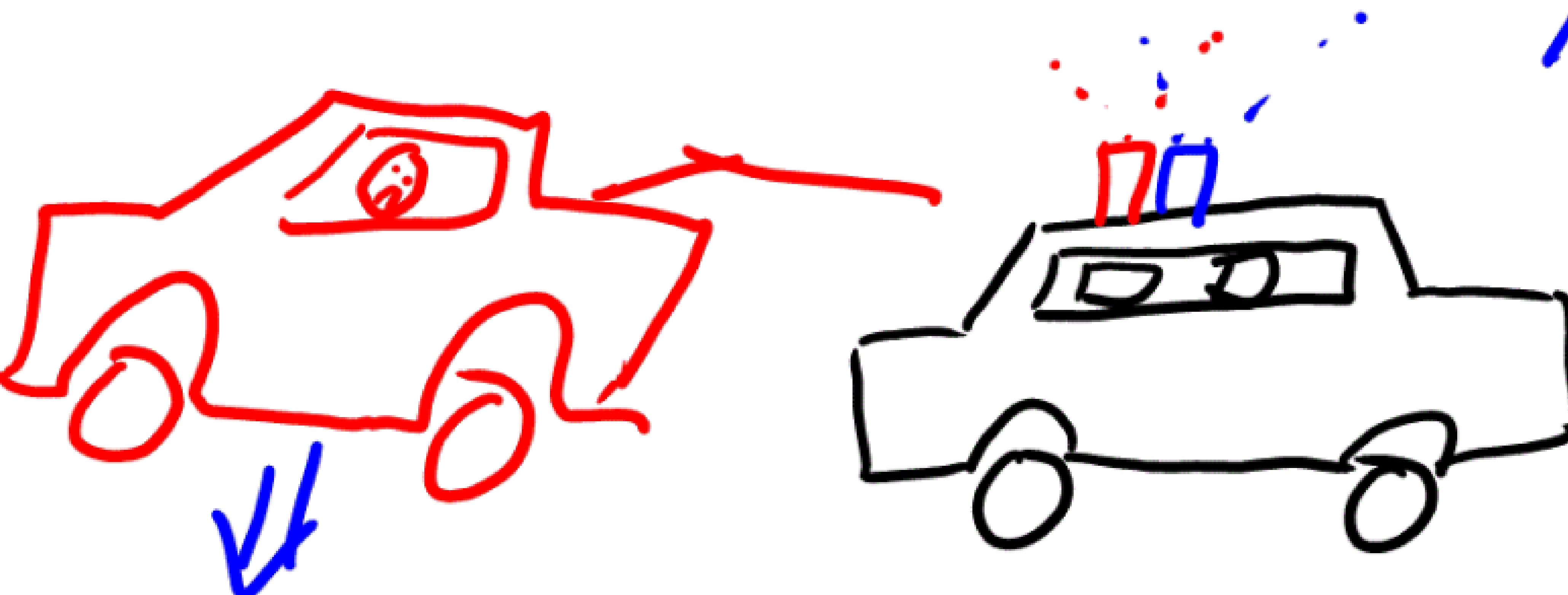
3.75 ⇒ 4.0

3.75 ⇓ 3.00

- Difference between the actual analog value and quantized digital value due is called **quantization error**.
  - Due either to rounding or truncation.

Speed Limit is 55

Radar only accurate to nearest 10 mph



$54 \Rightarrow 60$

NOISE AND THE NUMBER OF BITS

$$\text{SQNR} = 20\log_{10}(2^Q) \approx 6.02 \cdot Q \text{ dB}$$

How much is the S/N ratio?

$$6.02 \times 16$$

$$\Rightarrow 96.32 \text{ dB}$$

Java Real Time

⇒ what's wrong w/it?

⇒ Garbage Collection
is non-deterministic

⇒ Resource Access
JVM

WHAT'S WRONG WITH JAVA FOR REAL TIME SYSTEMS

- Unpredictibility -
- Garbage Collection -

Scheduling is unpredictable

- Run time binding
  - Adds delay to the execution of the program
- Java Virtual Machine
  - We're sitting on top of another OS, divorced from the actual system.

OPTIMIZATION OF CODE

- How can we optimize code?
  - Repeated calculations
  - Constant folding
  - Loop invariance removal
  - Induction variance
  - Loop unrolling
  - Loop jamming

REPEATED CALCULATIONS

- X = 6 + a * b;
- Y = a * b + z;

Same

- How could we simplify this?

T = a*b;
X = 6+t;
Y = t+z;

CONSTANT FOLDING

- X = 2.0 * x * 3.125;

Never changes

Do once @ compile time.

$X = 6.250 * X;$

LOOP INVARIANCE REMOVAL

- X = 100;
- While (x > 0)
- {
  - x = x – y + z;
- }

Ever change?

```
X = 100
T = -y+z;
while(x>0)
{
    x = x+T;
}
```

INDUCTION VARIANCE

- For (I = 1; I <= 10; i++)
- {
  - A[i+1] = 0;
- }

LOOP UNROLLING

- Duplicate instructions executed in a loop to reduce the number of operations and hence the loop overhead incurred

```
For (I = 1; i<=6;i++)
{
      A[i] = a[i]*8;
}
```

LOOP JAMMING

- Combine similar loops together to improve performance

```
For (I = 1; i<=100;i++)
{
      X[i] = y[i]*8;
}
For (I = 1; I <= 100; i++)
{
      Z[i] = x[i] * y[i];
}
```