Audio

u23 2013

andy, florob, gordin, ike, meise, tobix, zakx

Chaos Computer Club Cologne e.V.
http://koeln.ccc.de

2013-11-18

A/D-Wandlung

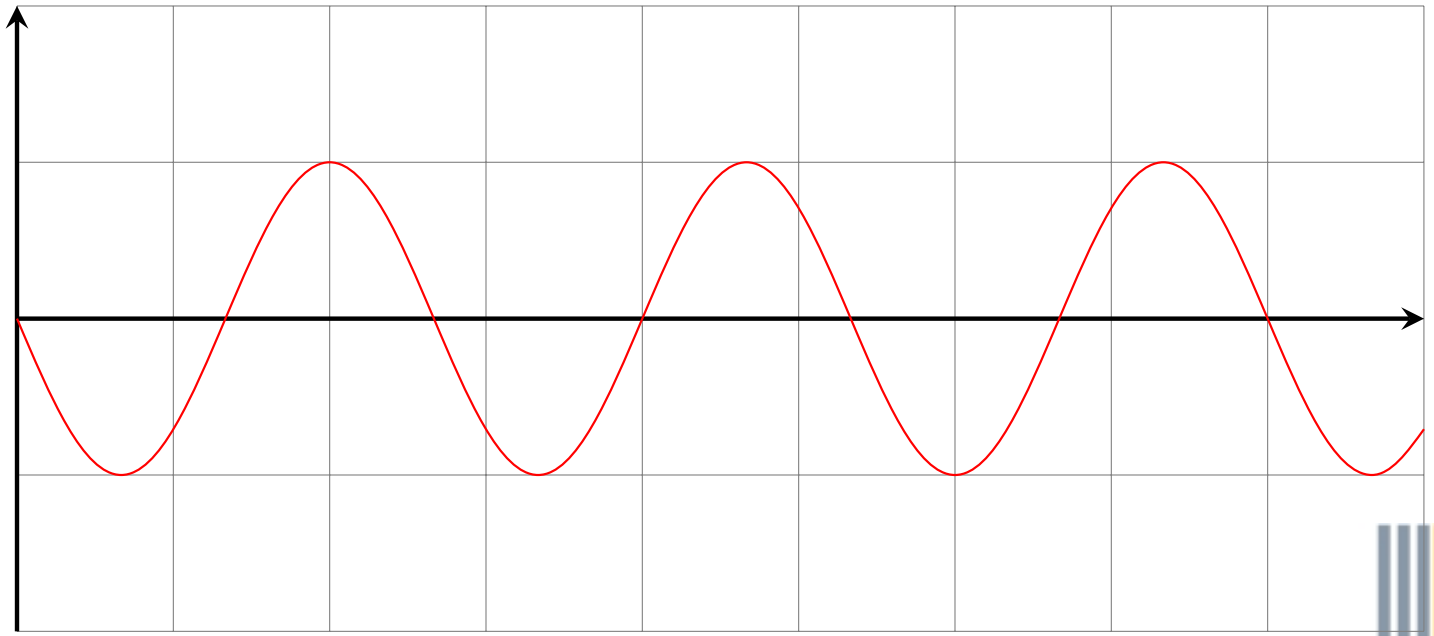
- Periodisches einlesen von Werten (*Abtastung*)
- Sampling-Frequenz f_s, z. B. 44 100 kHz
- Signale bis $f_g = f_s/2$, sonst Aliasing
- Tiefpass zum beschränken der Eingangsfrequenzen
- Diskretisieren der Abtastwerte (*Sample*)
- $2^b - 1$ Stufen
- z. B. 16-bit $\Rightarrow$ 65535 Abtastniveaus

Aliasing

Nyquist-Shannon-Theorem

$f_s > 2 \cdot f_g$, für perfekte Rekonstruktion

- $f_{\text{rot}} = 3\,\text{Hz}$
- $f_s = 4\,\text{Hz}$
- $f_{\text{blau}} = 1\,\text{Hz}$

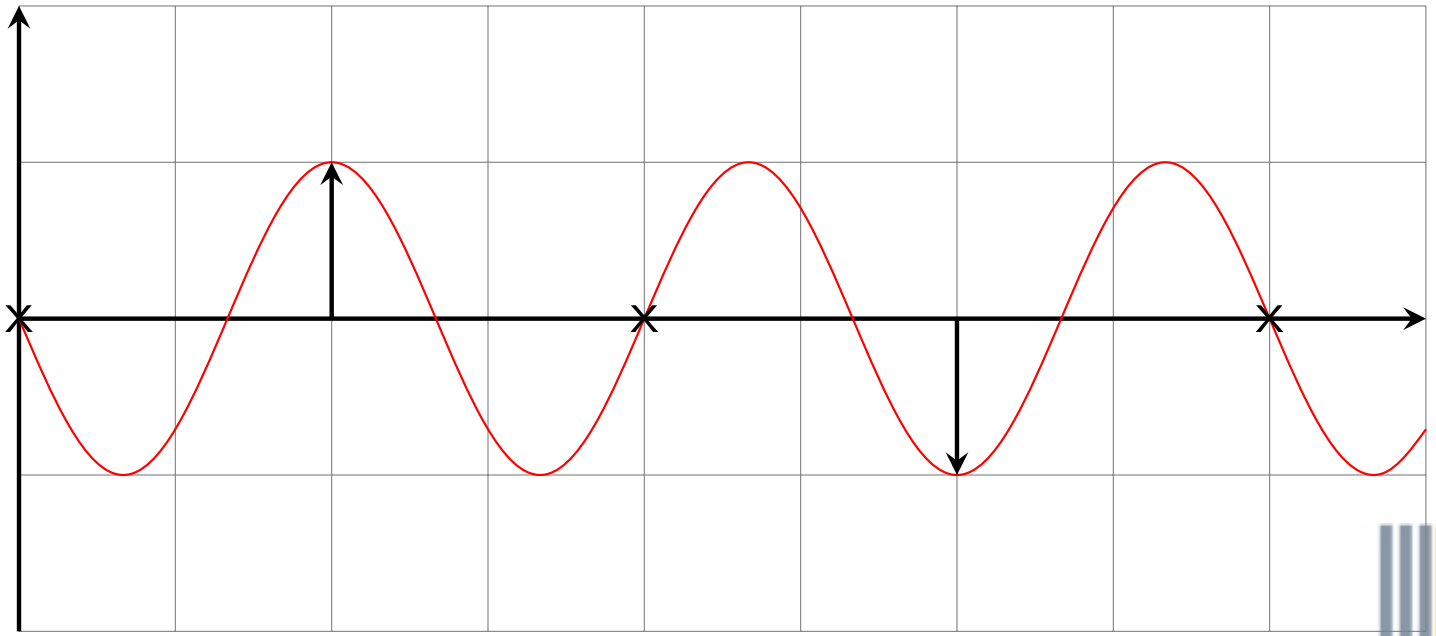


Aliasing

Nyquist-Shannon-Theorem

$f_s > 2 \cdot f_g$, für perfekte Rekonstruktion

- $f_{\text{rot}} = 3\,\text{Hz}$
- $f_s = 4\,\text{Hz}$
- $f_{\text{blau}} = 1\,\text{Hz}$

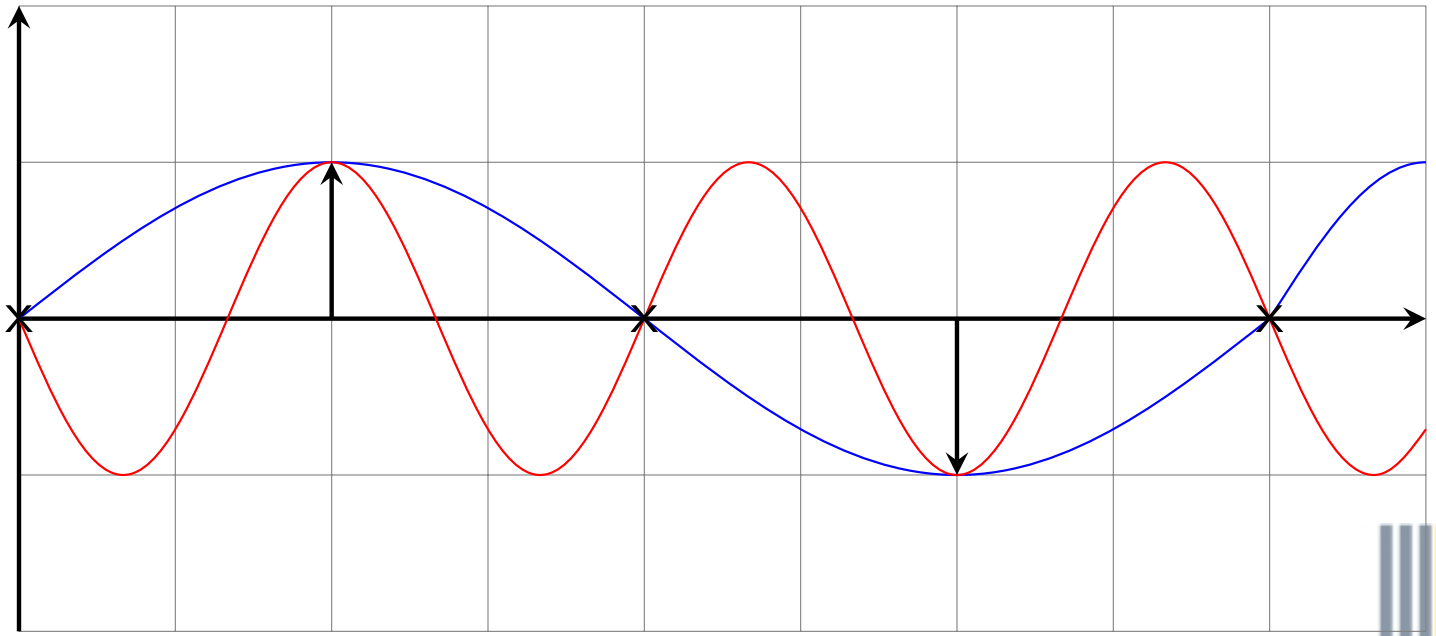


Aliasing

Nyquist-Shannon-Theorem

$f_s > 2 \cdot f_g$, für perfekte Rekonstruktion

- $f_{\text{rot}} = 3\,\text{Hz}$
- $f_s = 4\,\text{Hz}$
- $f_{\text{blau}} = 1\,\text{Hz}$



D/A-Wandlung

- Halte jedes Sample für $\frac{1}{f_s}$ s
- Kette von Rechtecken
- Tiefpassfilterung $\Rightarrow$ Glättung des Signals

Audio in der Library

- 2 Puffer werden abwechselnd bei Bedarf gefüllt
- Ausgabe erfolgt im Hintergrund über DMA
- Puffer beinhalten 128 Stereo Sample
- Beispiel: `examples/08_synth/`, `examples/11_guitar/`

API

```
1 void InitializeAudio(uint32_t freq);
2
3 typedef void AudioCallbackFunction(void *ctxt, ↩
      int16_t buffer[256]);
4 void PlayAudioWithCallback(AudioCallbackFunction ↩
      *callback, void *ctxt);
5 void StopAudio();
```

API: Beispiel

```
1  static void cb(void *ctxt, int16_t buffer[256]) {
2    static int v = 1;
3    for (int i = 0; i < 128; i++) {
4      buffer[2*i] = buffer[2*i+1] = v;
5      v *= -1;
6    }
7  }
8
9  int main(void) {
10   // ...
11   InitializeAudio(AudioFreq_11k);
12   PlayAudioWithCallback(cb, NULL);
13   // ...
14 }
```