

Entwicklung und Implementierung eines digitalen Funktionsgenerators in VHDL

Markus Hartlage

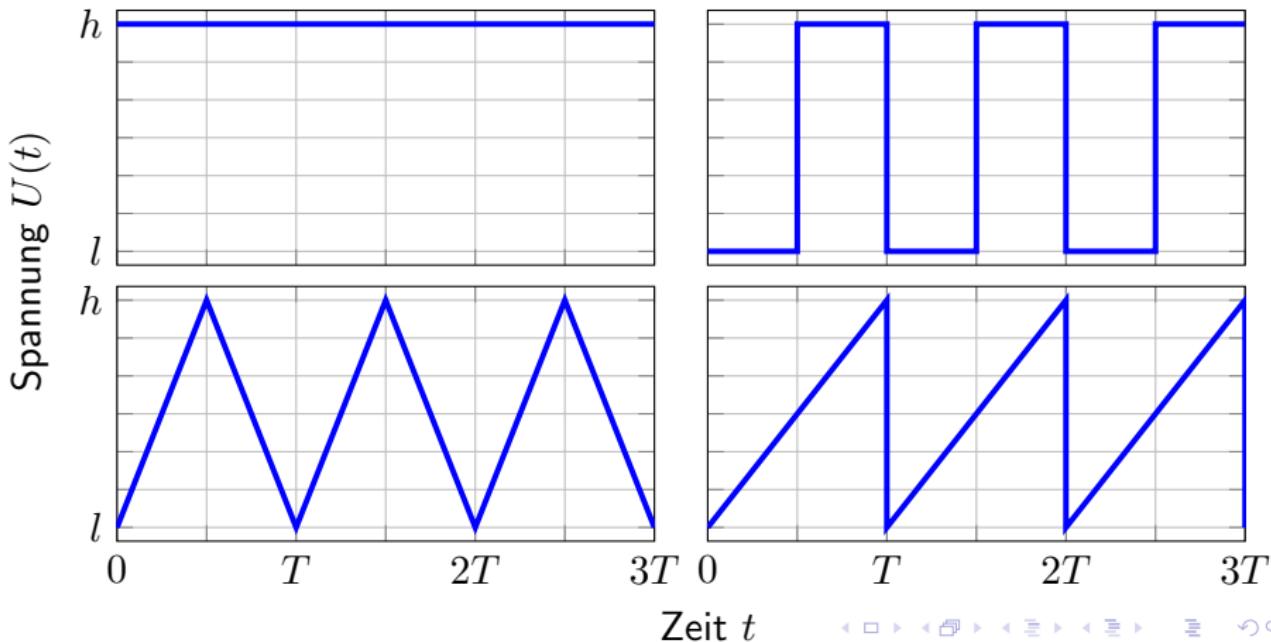
FH-Bielefeld

April 6, 2022



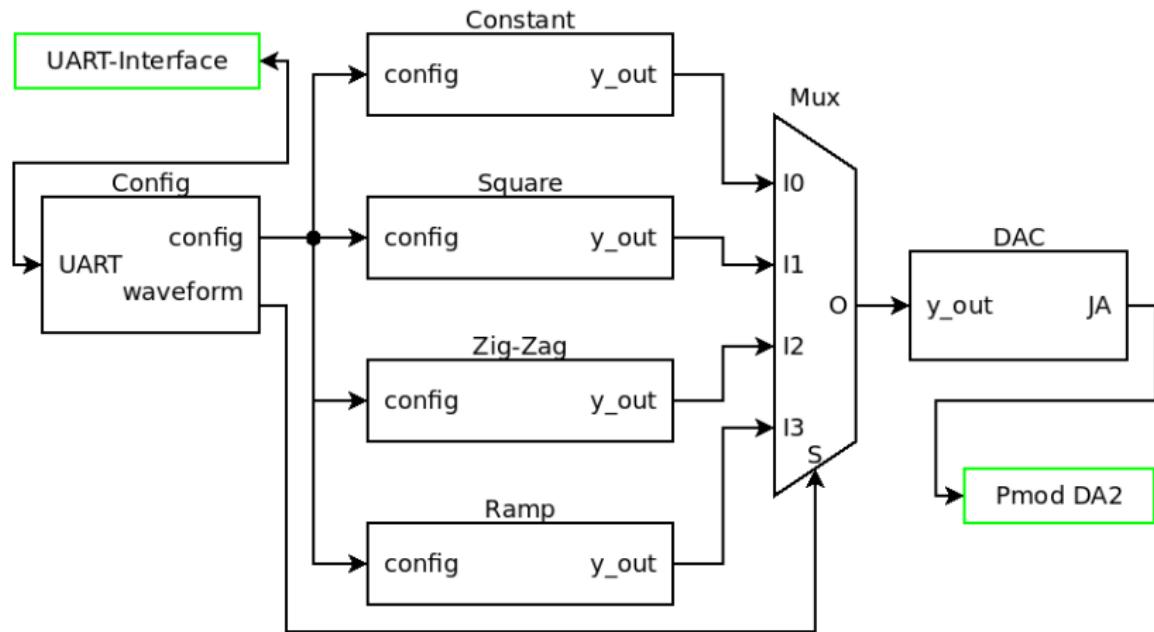
Konzept - Anforderungen

- ▶ Ausgabe vier verschiedener Funktionsverläufe
 - ▶ Konstante, Rechteck, Zick-Zack, Rampe
- ▶ Konfiguration per UART-Schnittstelle
 - ▶ high- und low-Wert, Zykluszeit, dutycycle, Flankensteigung

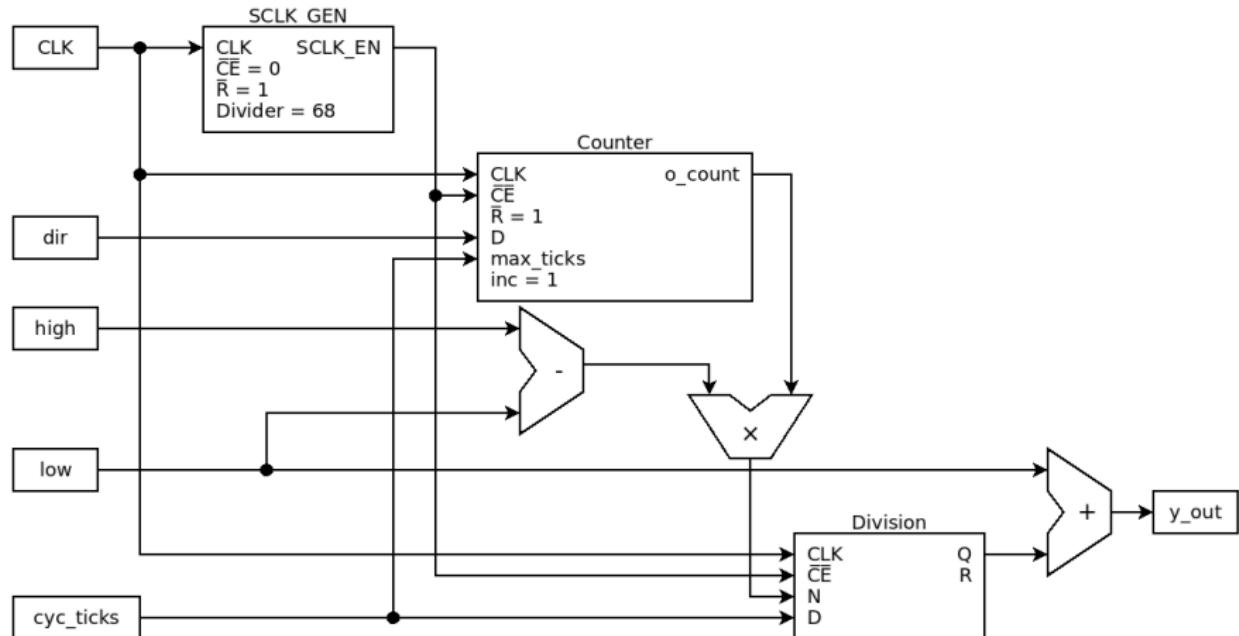


Konzept - Aufbau

- ▶ Aufbau aus Konfigurations-, Funktions- und DAC Komponente
- ▶ zusätzliche Hardware: Uart-Interface und digital-analog Konverter



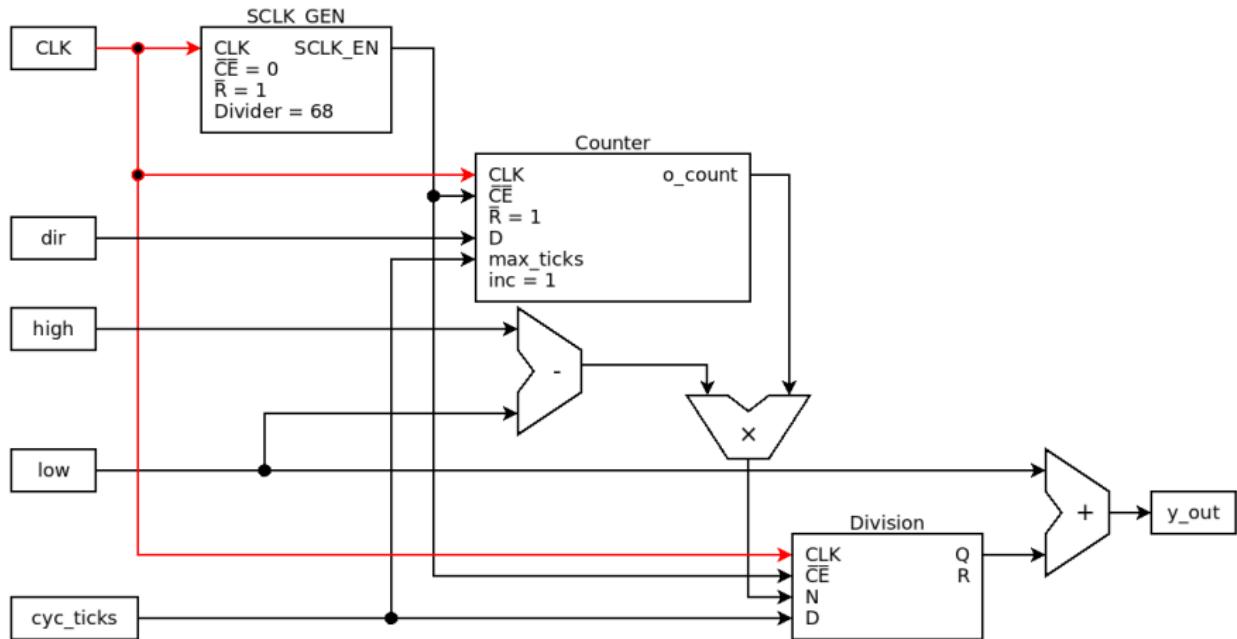
Komponenten - Rampe



- ▶ Abbildung des 24 Bit Zählstands auf 12 Bit Ausgang:

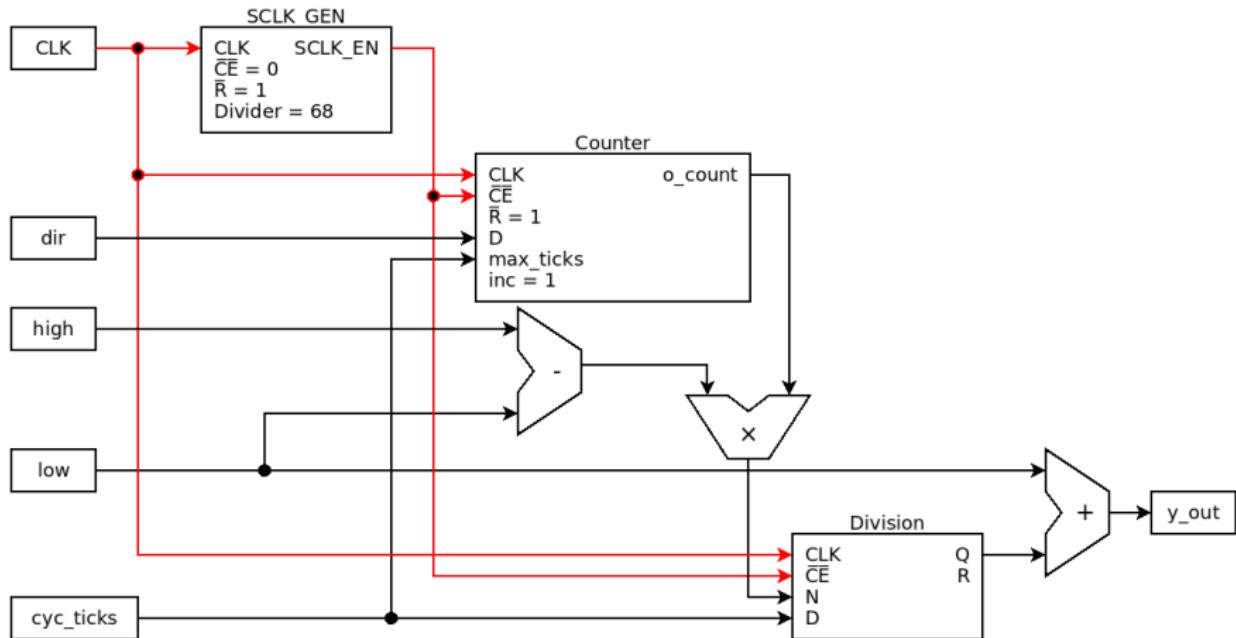
$$y_{out} = (high - low) \cdot o_count \div cyc_ticks$$

Komponenten - Rampe



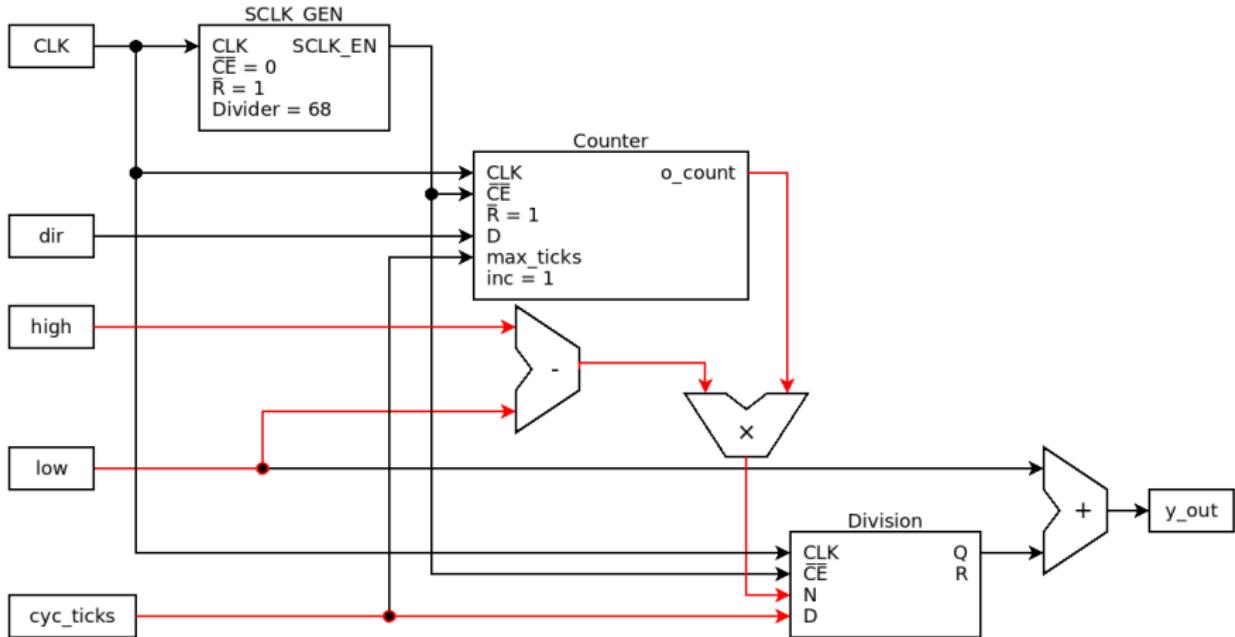
- ▶ Abzählen von 68 Taktzyklen

Komponenten - Rampe



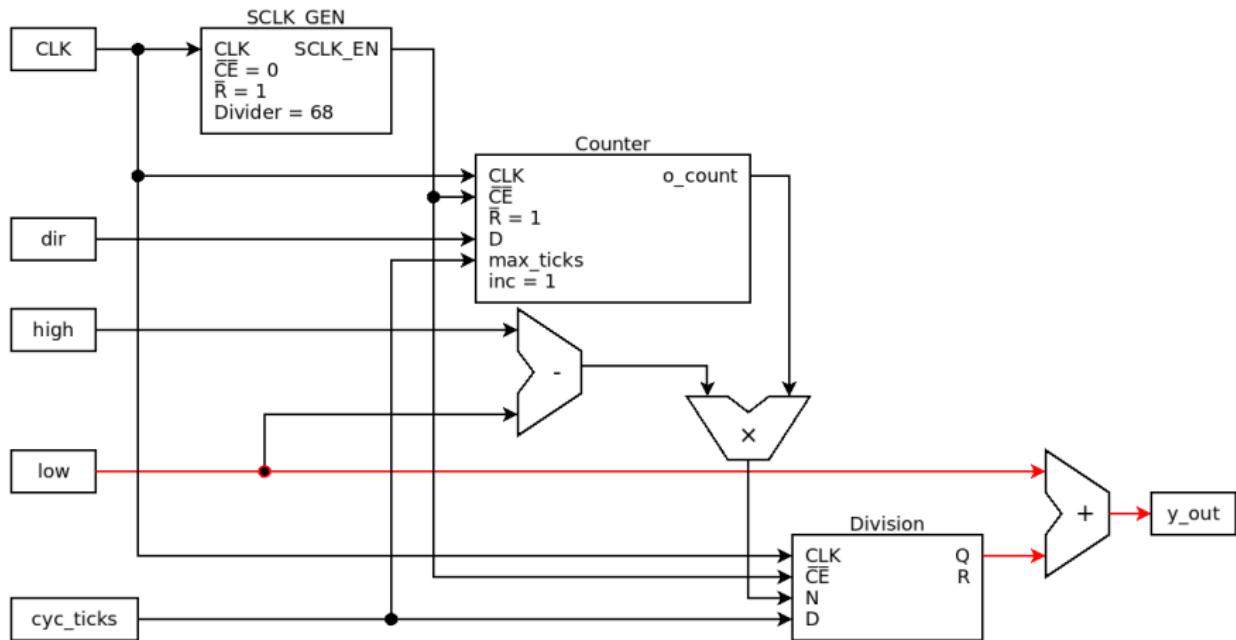
► Aktivieren von *Counter* und *Division*

Komponenten - Rampe



- neuer Zählstand wird mit Amplitude multipliziert und *Division* beginnt zu teilen

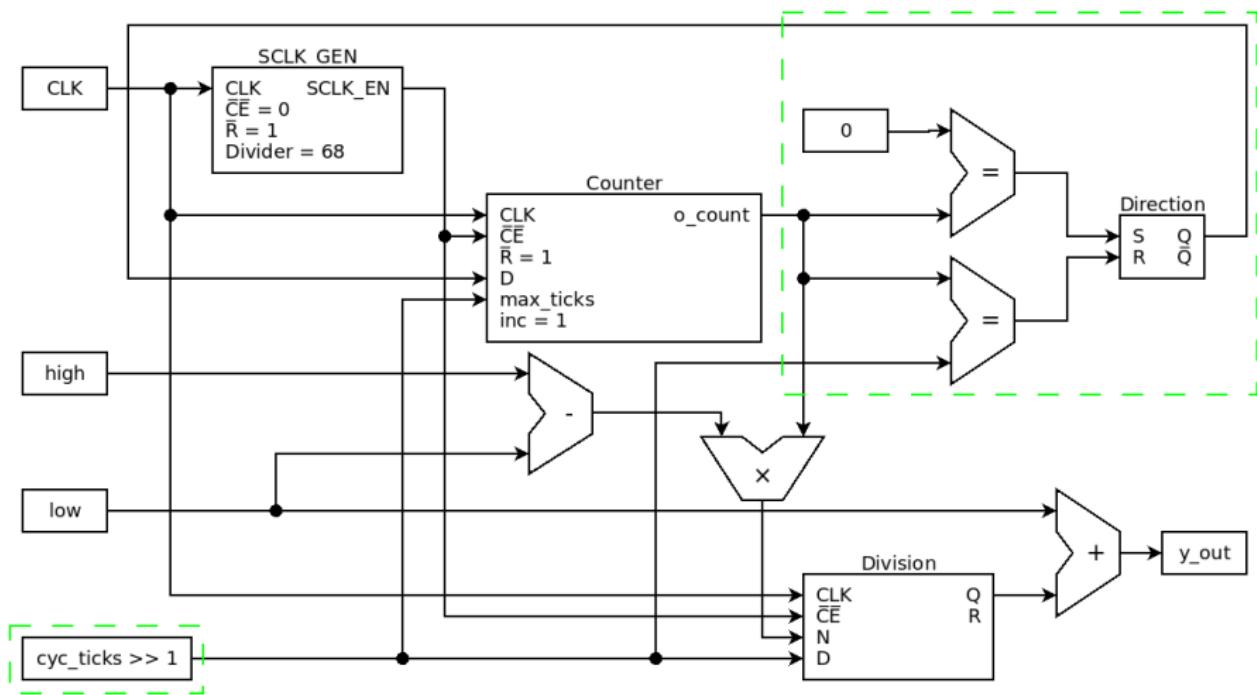
Komponenten - Rampe



- ▶ untere zwölf Bits von Q plus low ergeben y_out

Komponenten - Zick-Zack

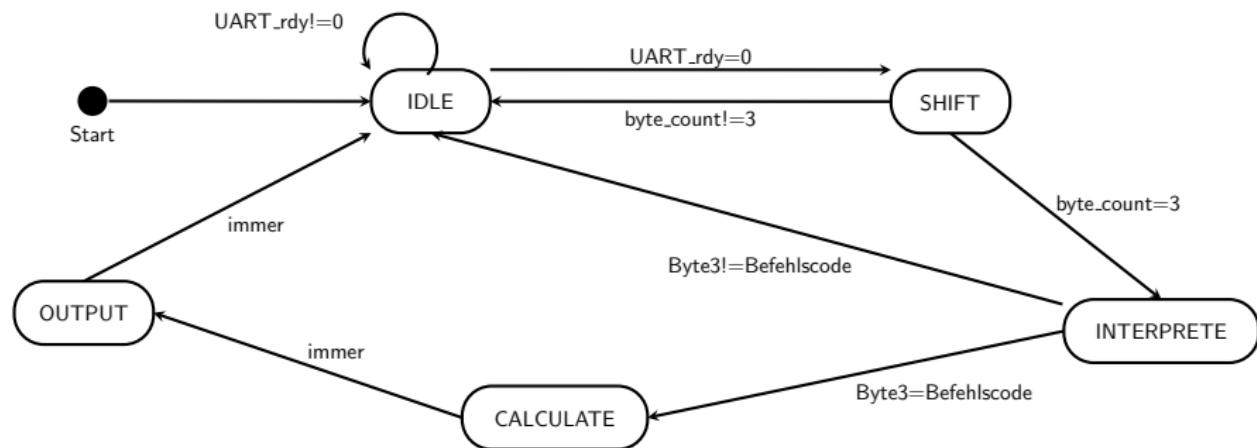
- Zählrichtung wird mit Komparatoren geregelt
- halbe Zykluszeit (*cyc_ticks* um ein Bit nach rechts geschoben)



Komponenten - Konfiguration

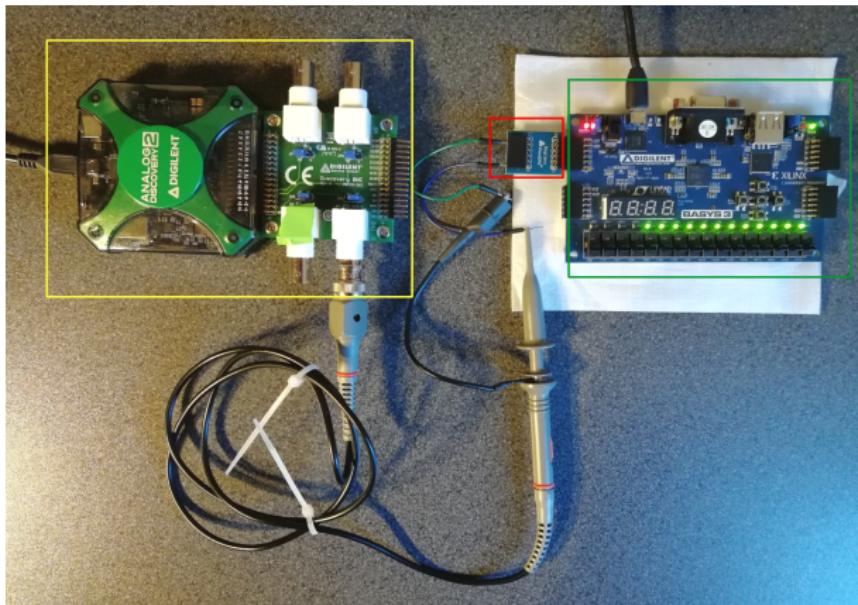
Aufbau als state machine:

- ▶ Byteweises Einlesen der UART Rx Signale
- ▶ Interpretation von vier Bytes als Befehl (Befehlscode + Argumente)
- ▶ Berechnung und Speicherung der neuen Konfiguration



Funktionstest - Versuchsaufbau

- ▶ Implementierung auf Basys 3 Board
- ▶ Messung mit Analog Discovery 2



Funktionstest - Versuchsdurchführung

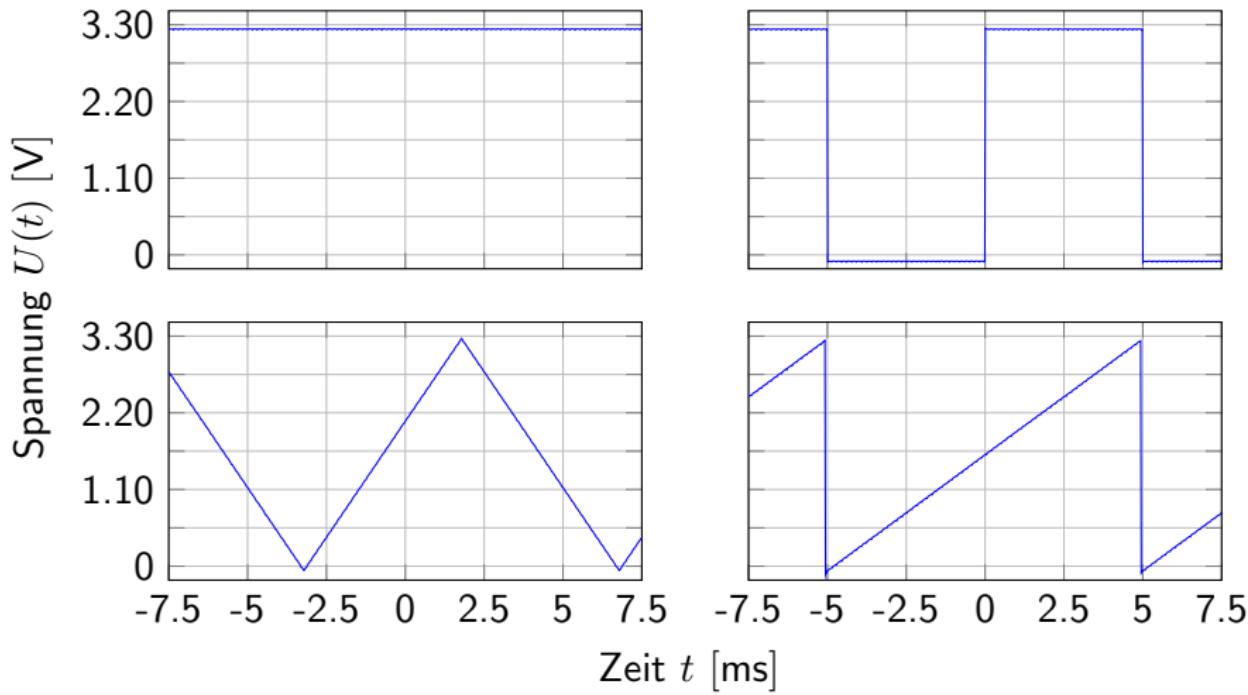
- ▶ Ausführen und Kontrollieren von jedem Konfigurationsbefehl
- ▶ Jede Funktion einmal bei 100 Hz mit $U_{low} = 0\text{ V}$ und $U_{high} = 3,3\text{ V}$ messen
- ▶ Zick-Zack-Funktion zur Kontrolle des Frequenzbereichs mit folgenden Frequenzen f :

f in Hz	0,08	0,1	1	10	100	10^3	10^4	10^5	$3,5 \cdot 10^5$	10^6
-----------	------	-----	---	----	-----	--------	--------	--------	------------------	--------

- ▶ $0,08\text{ Hz} < f_{min}$ und $1\text{ MHz} > f_{max}$

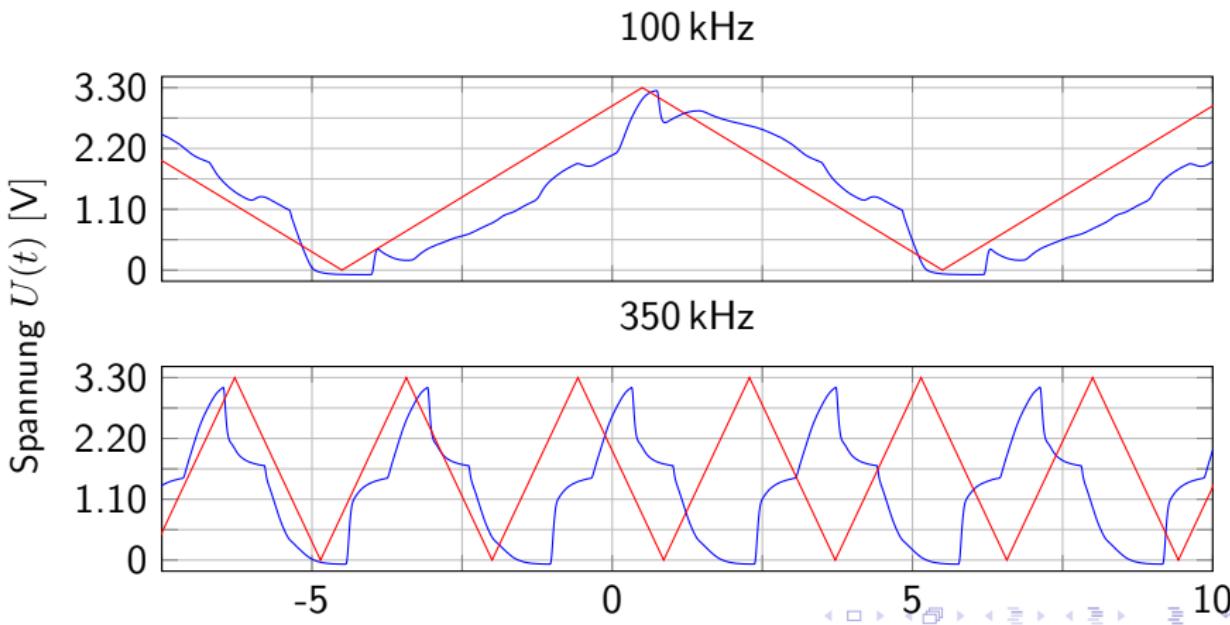
Funktionstest - Funktionsverläufe bei 100 Hz

- Frequenz wird korrekt ausgegeben
- 0 V wird unterschritten und 3,3V nicht erreicht



Funktions-test - Kontrolle des Frequenzbereichs

- bei $f < f_{min}$ kommt es zu Aliasing
- bei $f > f_{max}$ wird konstant 0 V ausgegeben
- bei $f > 10 \text{ kHz}$: Spannung und Frequenz weichen stark von idealer Kurve ab
 - reduzierte Auflösung, Trägheit des DACs



Fazit

- ▶ UART-Schnittstelle funktioniert
- ▶ Generierung von Funktionen
 - ▶ zuverlässig von 0,1 Hz bis 10 kHz
 - ▶ darüber hinaus sehr unsaubere Funktionsverläufe
 - ▶ sinkende Auflösung
 - ▶ evtl. zu langsame Änderung des DAC-Ausgangswerts
 - ▶ U_{high} und U_{low} scheinen verschoben zu sein
 - ▶ evtl. ungenaue Referenzspannung, da Referenzspannung der Versorgungsspannung entspricht

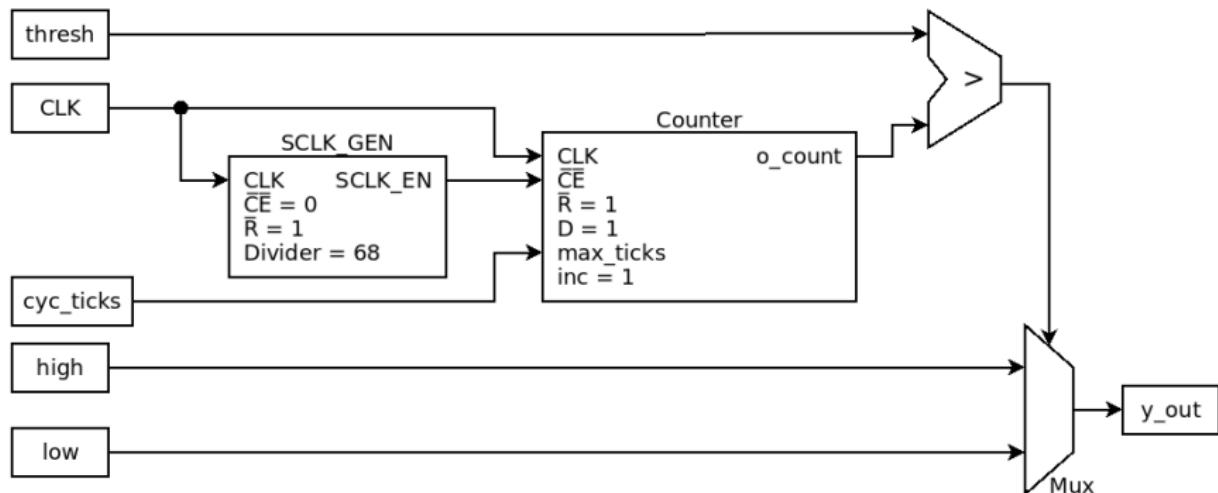
Referenzen

Source Code, Anleitung und Python Interface:

https://github.com/markushart/studienarbeit_function_generator.git

Komponenten - Rechteck

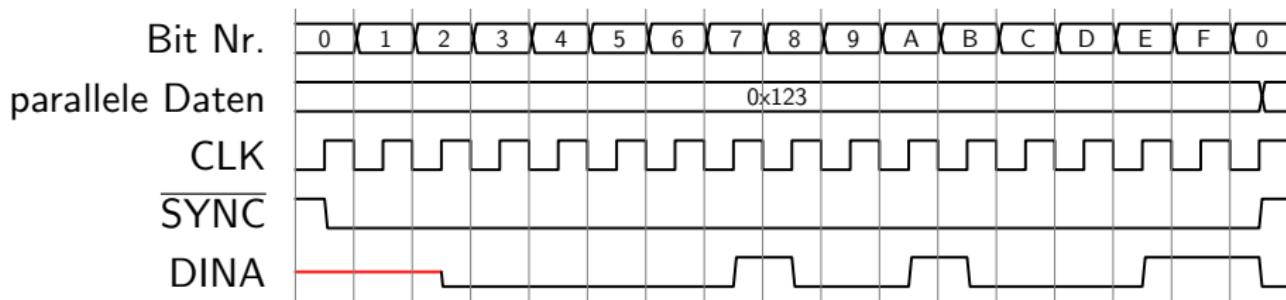
- nach 68 steigenden Flanken *Counter* aktivieren
- Ausgang auf *low* wenn *o_count* > *thresh*, sonst *high*



Komponenten - Digital-Analog-Konverter

- Implementierung des PmodDA2-Protokolls

- SYNC: Chip-Enable Signal für den DAC
- DINA, DINB: serielle DAC-Daten, aktuell nur Kanal A genutzt
- CLK: DAC-Clock läuft mit 25 MHz



Zick-Zack bei $f < f_{min}$ und $f > f_{max}$

- ▶ Frequenzbereich:
 - ▶ $f_{min} = 0,0877 \text{ Hz}$
 - ▶ $f_{max} = 367,5 \text{ kHz}$ für die Zick-Zack Funktion
 - ▶ $f_{max} = 735 \text{ kHz}$ für Rampe und Rechteck
- ▶ bei $f < f_{min}$ ist Ausgangsfrequenz 20,5 Hz
 - ▶ interner Zählstand für `cyc_ticks` nicht mehr darstellbar
- ▶ bei $f > f_{max}$ wird konstant 0V ausgegeben
 - ▶ $cyc_ticks/2 = 1$, `direction` wird nicht mehr getoggelt