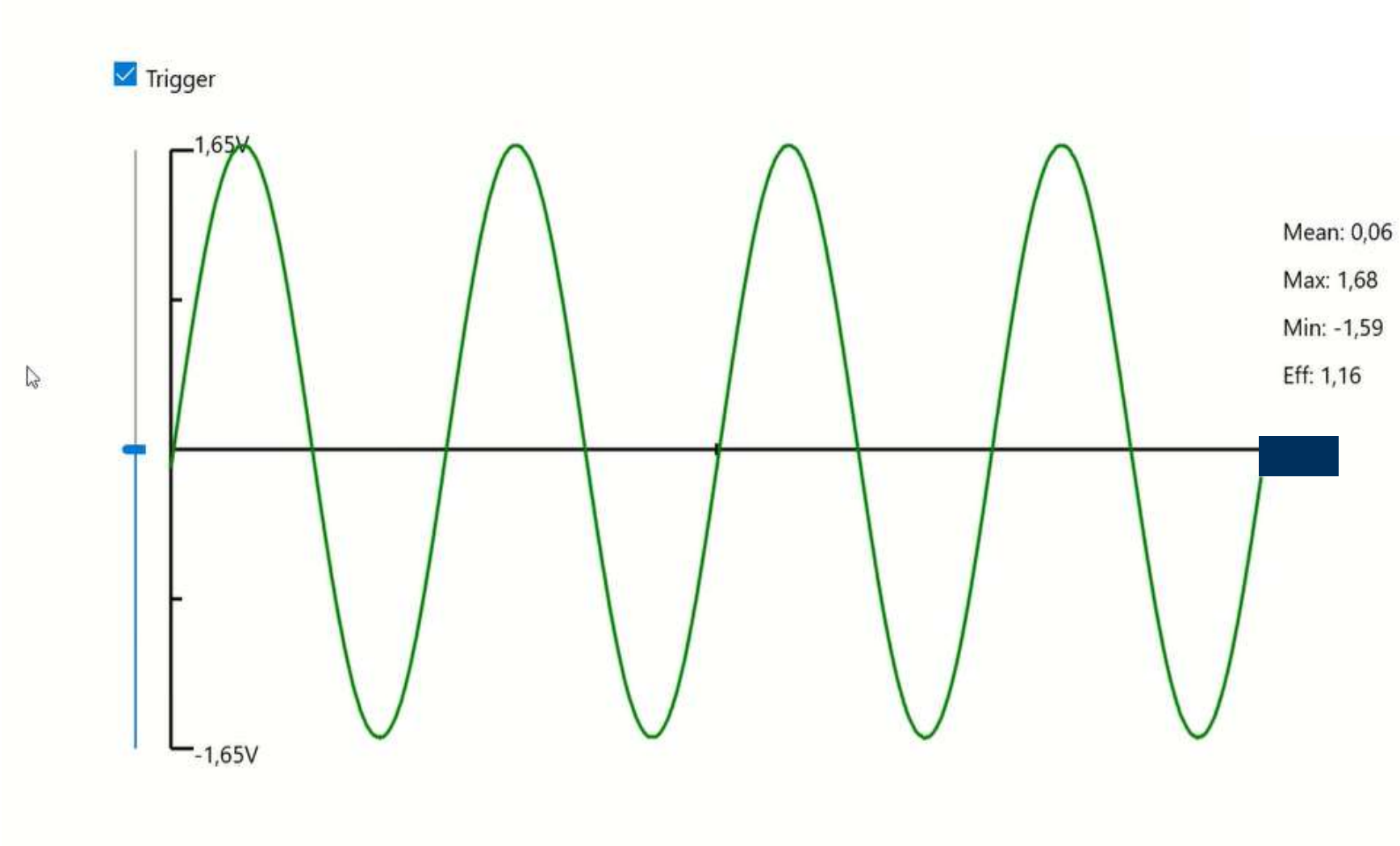
Übung 4 -- Oszilloskop Mikrocomputertechnik (MCT)

Prof. Dr. Wolfram Acker

Fachhochschule Kiel
Hochschule für Angewandte Wissenschaften

Ozilloskop



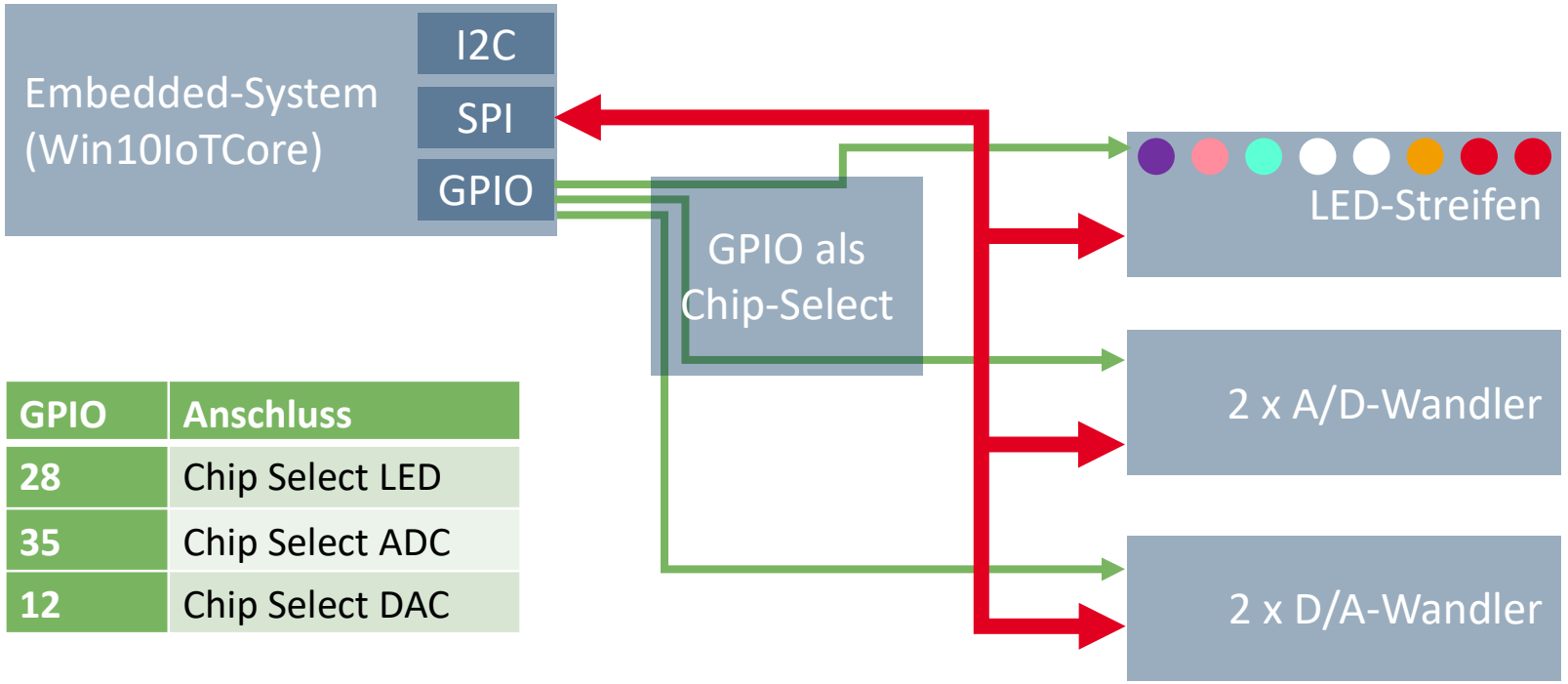
Bisherige Übungen

- Übung 1 (Taschenrechner)
  - Ereignisorientierte Programmierung
  - Einstieg in Grafische Benutzeroberflächen mit XAML
- Übung 2 (LED)
  - SPI
  - GPIO
  - Daten in einem Array verwalten
- Übung 3 (Blinker)
  - Timer
  - Trennung von Kern und Oberfläche
  - Grafik
- Übung 4 (Oszilloskop)
  - Ansteuerung von SPI und GPIO
  - Handhabung von Arrays und Schleifen
  - Ansteuerung von Hardware über Objekte

Schritte

1. Erstellen der grafischen Oberfläche in XAML
2. Ansprechen des A/D-Wandlers über SPI (und GPIO)
3. Auslesen einzelner Werte
4. Zusammenbau des Wertes aus den Bytes
5. Ermittlung des Zusammenhangs zwischen den binären Werten und den realen Spannungswerten
6. Grafische Darstellung
7. Berechnung von Signalkenngrößen und Darstellung
8. Implementierung eines „Triggers“

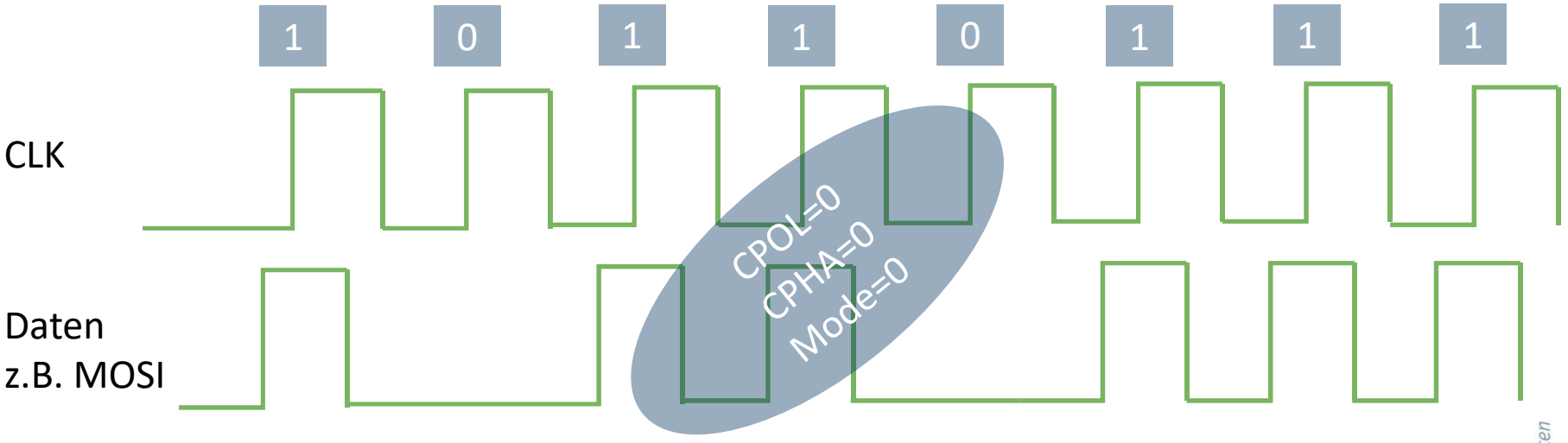
Die Hardware, der IoT-PC (nur SPI)



GPIO	Anschluss
28	Chip Select LED
35	Chip Select ADC
12	Chip Select DAC

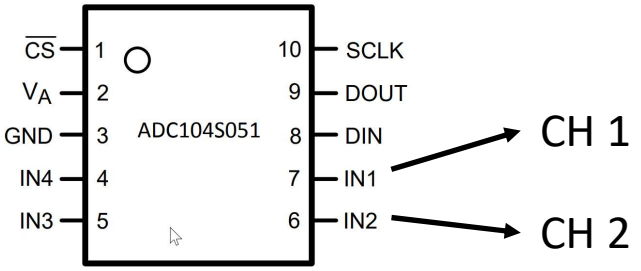
- Über SPI verbunden:
  - A/D-Wandler mit zwei Kanälen (Dieser Laborversuch)
  - D/A-Wandler mit zwei Kanälen
  - LED-Streifen

Timing am Empfänger

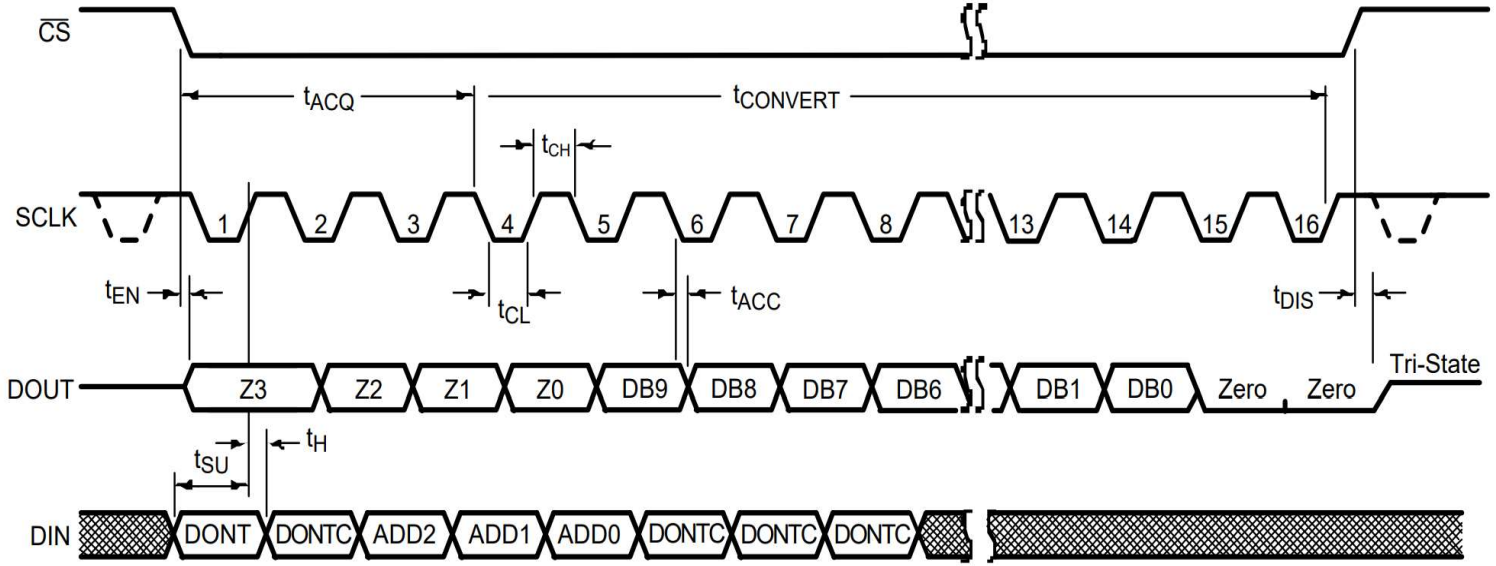


- Der Takt ist 0 wenn keine Daten übertragen werden (CPOL=0)
- In diesem Beispiel wird das jeweilige Bit bei der ersten Flanke (CPHA=0), d.h. an der steigenden Flanke übernommen
- Wie viele Kombinationen sind aus CPHA und CPOL sind möglich?

Ansteuerung des ADC



- ADC10S051 von Texas Instruments



[Quelle Bilder: Texas Instruments http://www.ti.com/lit/ds/symlink/adc104s051.pdf]

SPI Modes

- Welchen Mode brauchen wir hier?

```
namespace Windows.Devices.Spi
{
    ...public enum SpiMode
    {
        //
        // Zusammenfassung:
        //     CPOL = 0, CPHA = 0.
        Mode0 = 0,
        //
        // Zusammenfassung:
        //     CPOL = 0, CPHA = 1.
        Mode1 = 1,
        //
        // Zusammenfassung:
        //     CPOL = 1, CPHA = 0.
        Mode2 = 2,
        //
        // Zusammenfassung:
        //     CPOL = 1, CPHA = 1.
        Mode3 = 3
    }
}
```

General Purpose Input Outputs (GPIO)

- Ermöglichen das direkte Ansteuern von Hardware
- Einzelne Bits, die
  - als Ausgänge direkt aus dem C#-Code high oder Low geschaltet werden können
  - Als Eingänge direkt aus dem C#-Code heraus gelesen werden können
- Die GPIO werden über Objekte des Typs GPIO-Controller erzeugt

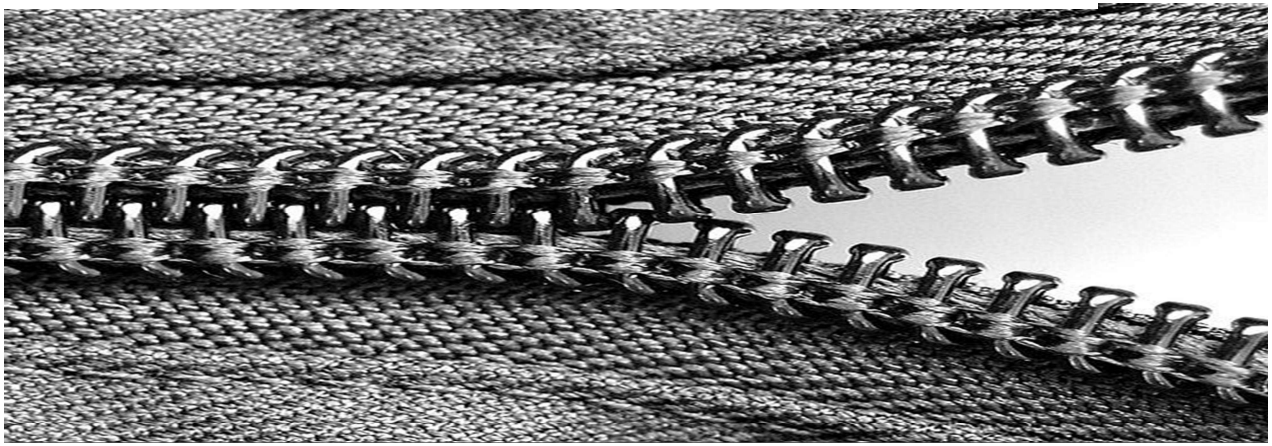
liefert das Standard-Controller-Objekt

```
void initGPIO()
{
    GpioController gpio = GpioController.GetDefault();
    PinChipSelectLED = gpio.OpenPin(28);
    PinChipSelectADC = gpio.OpenPin(35);
    PinChipSelectDAC = gpio.OpenPin(12);

    PinChipSelectLED.SetDriveMode(GpioPinDriveMode.Output);
    PinChipSelectADC.SetDriveMode(GpioPinDriveMode.Output);
    PinChipSelectDAC.SetDriveMode(GpioPinDriveMode.Output);

    PinChipSelectLED.Write(GpioPinValue.High);
    PinChipSelectADC.Write(GpioPinValue.High);
    PinChipSelectDAC.Write(GpioPinValue.High);
}
```

```
GpioPin PinChipSelectDAC;
GpioPin PinChipSelectLED;
GpioPin PinChipSelectADC;
```

Senden und Empfangen

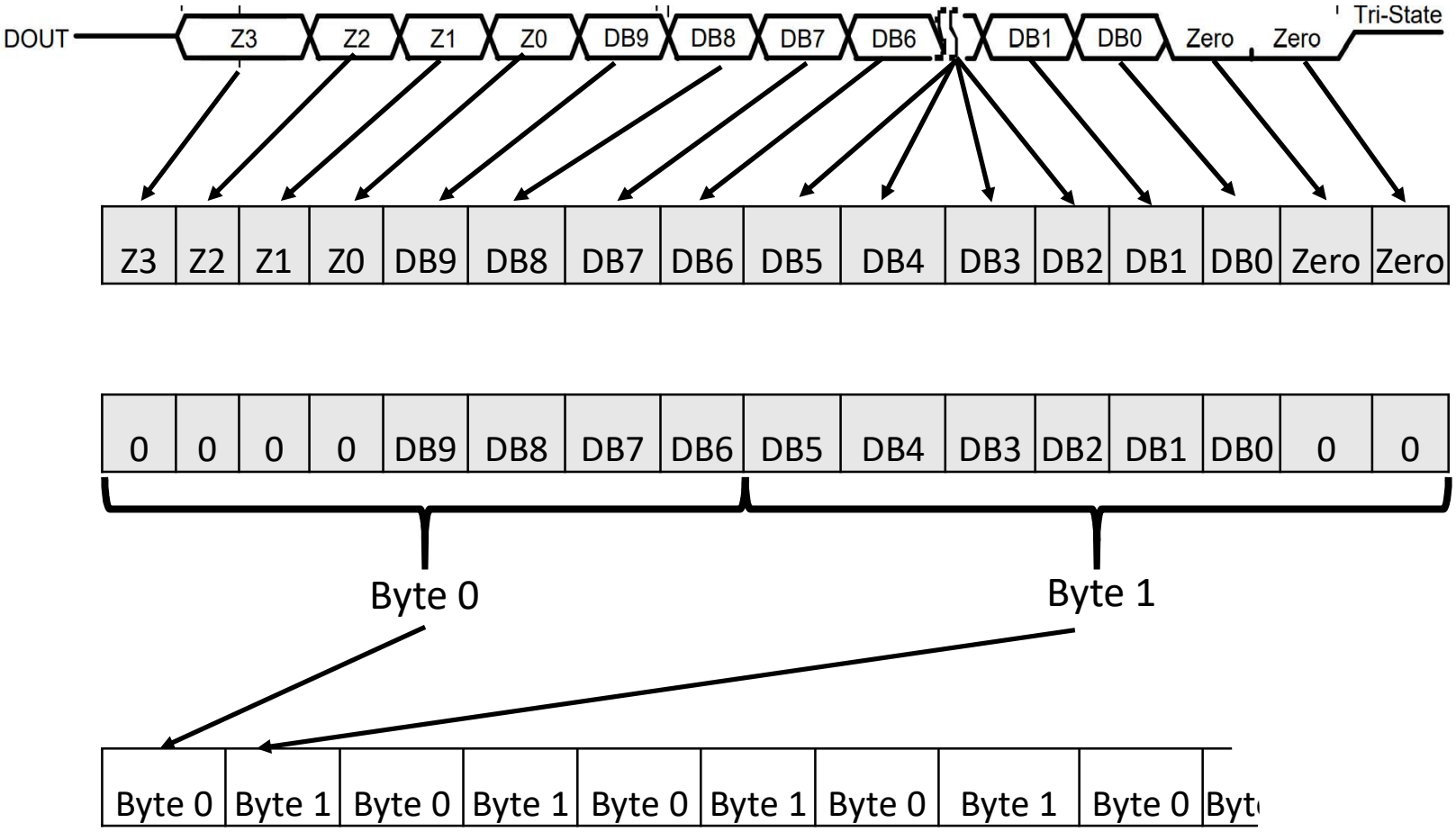
```
PinChipSelectADC.Write(GpioPinValue.Low);
ADC.TransferFullDuplex(SendBuf, RcvBuf);
PinChipSelectADC.Write(GpioPinValue.High);
```



	Byte 0								Byte 1							
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Gesendet	DONTC	DONTC	ADD2	ADD1	ADD0	DONTC	DONTC	DONTC	DONTC	DONTC	DONTC	DONTC	DONTC	DONTC	DONTC	DONTC
Empfangen	Z3	Z2	Z1	Z0	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Zero	Zero

[Quelle Bild: Rabensteiner Wikipedia]

Was kommt vom A/D-Wandler zurück?



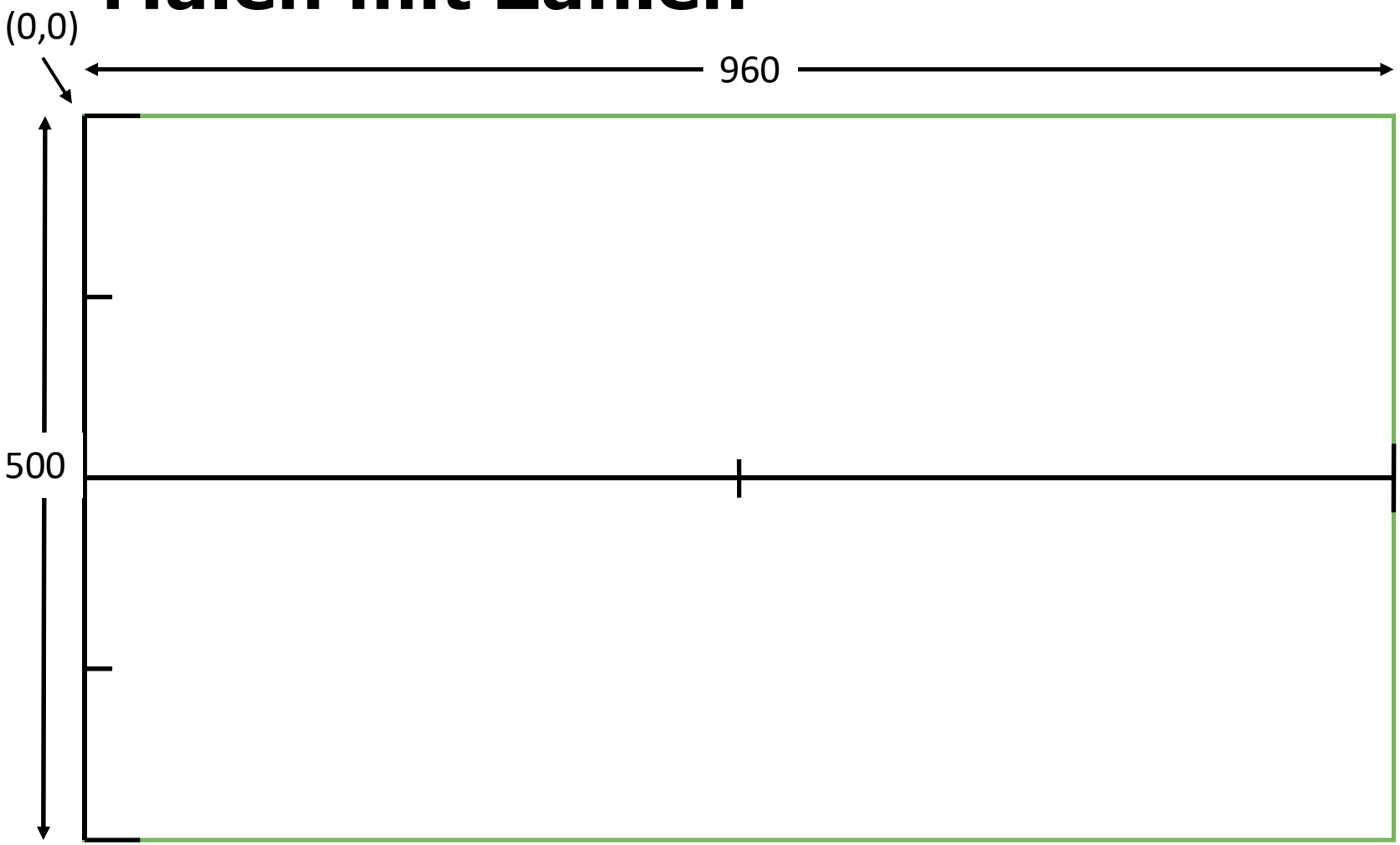
Objektorientierte Struktur

```
public delegate void neueDatenTyp(double [] buf)
```

```
class MainPage
{
Oszilloskop meinOszilloskop;
Public MainPage()
{
meinOszilloskop.neueDaten+=
neueDatenHandler;
}
void neueDatenHandler (double
[] daten)
{

}

}
```

```
class Oszilloskop
{
        GpioPin CS_ADC;
        public event neueDatenTyp neueDaten;
        DispatcherTimer timer;
        SPIDevice SPI_ADC;
        void GPIOInit()
        {
        }
        void SPI_Init()
        {
        }
        void TimerTick()
        {
        }
```

Malen mit Zahlen



Viel Erfolg !!!