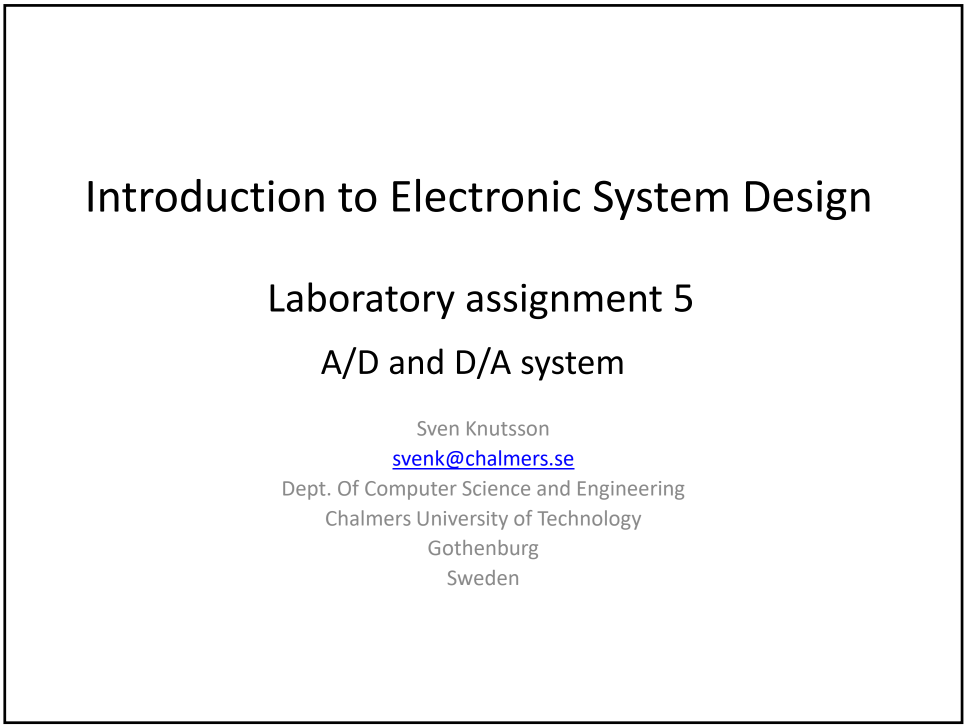
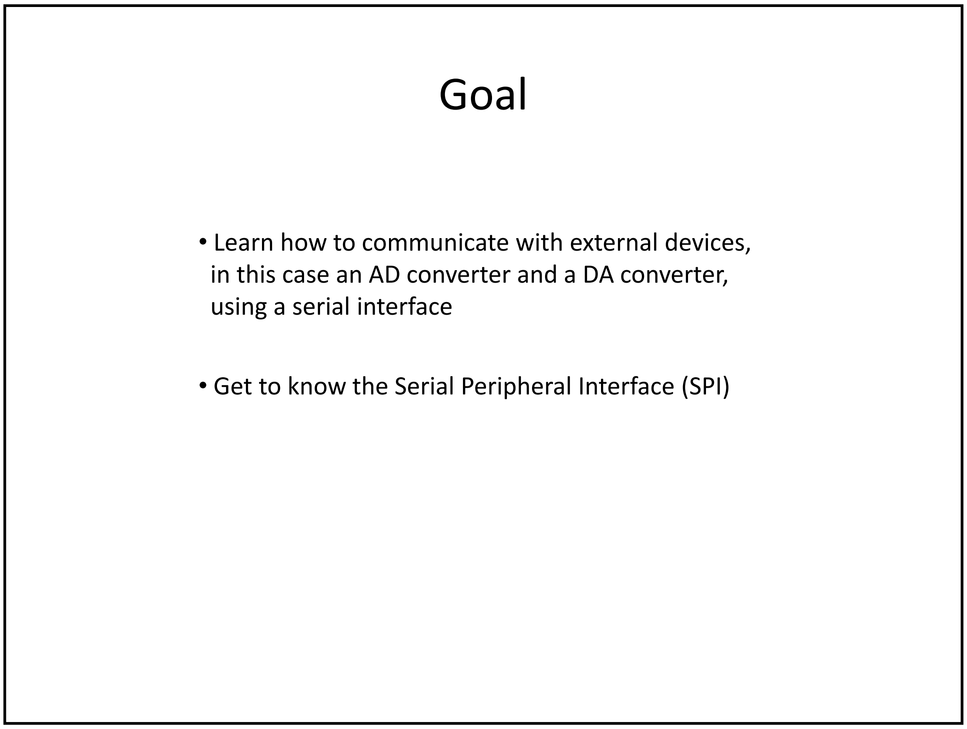


Introduction to Electronic System Design

Laboratory assignment 5

A/D and D/A system

Sven Knutsson
svenk@chalmers.se
Dept. Of Computer Science and Engineering
Chalmers University of Technology
Gothenburg
Sweden



Goal

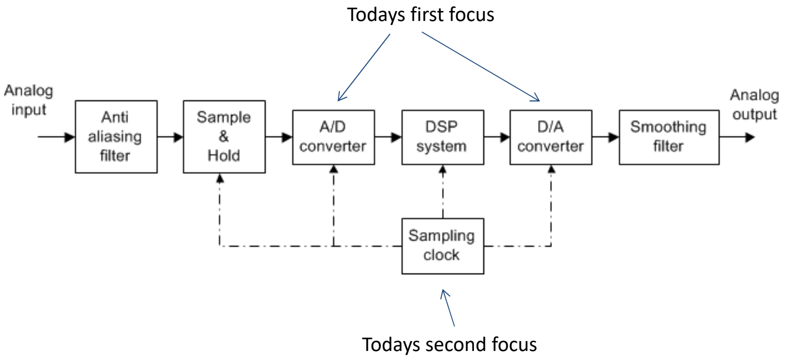
- Learn how to communicate with external devices, in this case an AD converter and a DA converter, using a serial interface
- Get to know the Serial Peripheral Interface (SPI)

Assignments

- Module to communicate with the ADC using SPI
- Module to communicate with the DAC using SPI
- Use the two converters together in a sampled system

Introduction

In lab assignment 2 we introduced a DSP system



Todays first focus

Analog input

Anti aliasing filter

Sample & Hold

A/D converter

DSP system

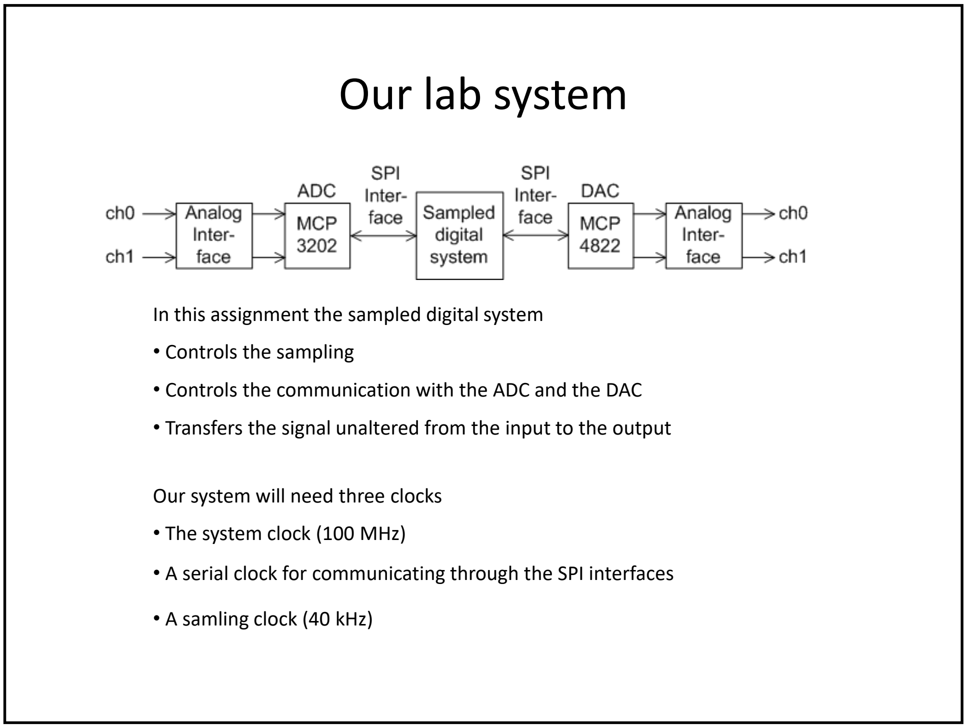
D/A converter

Smoothing filter

Analog output

Sampling clock

Todays second focus



Our lab system

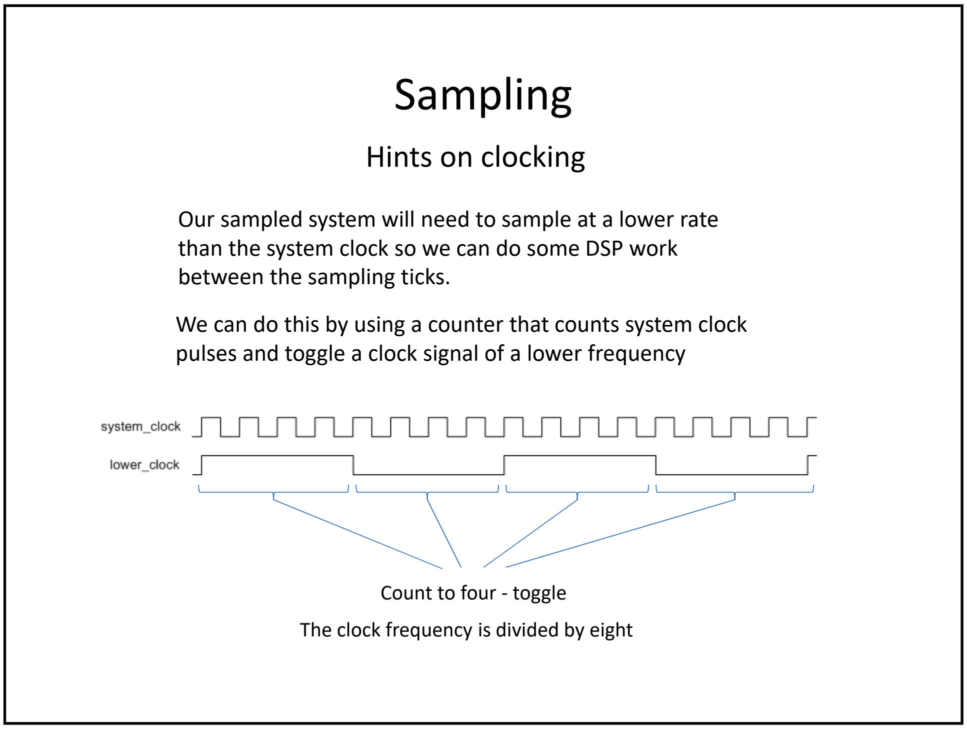
ch0
ch1
Analog Inter-face
ADC
MCP 3202
SPI Inter-face
Sampled digital system
SPI Inter-face
DAC
MCP 4822
Analog Inter-face
ch0
ch1

In this assignment the sampled digital system

- Controls the sampling
- Controls the communication with the ADC and the DAC
- Transfers the signal unaltered from the input to the output

Our system will need three clocks

- The system clock (100 MHz)
- A serial clock for communicating through the SPI interfaces
- A samling clock (40 kHz)



Sampling

Hints on clocking

Our sampled system will need to sample at a lower rate than the system clock so we can do some DSP work between the sampling ticks.

We can do this by using a counter that counts system clock pulses and toggle a clock signal of a lower frequency

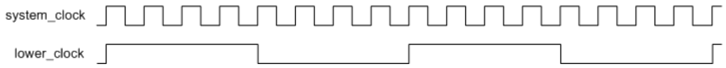
system_clock
lower_clock

Count to four - toggle

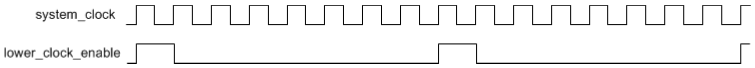
The clock frequency is divided by eight

Sampling

Hints on clocking cont.



This solution is not recommended since there are not that many clock nets in an FPGA and multiple clocks can screw up syncronisation



Use clock enable instead!

For the converters to work the serial clock for the AD and DA converters will have to be symmetrical though so you still need a symmetrical SPI clock too

Sampling

Hints on clocking cont.

How to we trigger on our clocks?

```
IF (rising_edge(system_clock) THEN
   .. .. ..
   .. .. ..
END IF;
```

```
~~IF (rising_edge(lower_clock) THEN~~
~~   .. .. ..~~
~~   .. .. ..~~
~~END IF;~~
```

```
IF (rising_edge(system_clock) THEN
   IF (lower_clock_enable='1') THEN.. .. ..
      .. .. ..
   END IF;
END IF;
```

!

There is a paper, Hints on clocking, on the home page

Let´s give some general information on A/D and D/A converters

Analog to digital and digital to analog converters

A/D converter	D/A converter
ADC	DAC
a2d	d2a

There are many names with the same meaning

General A/D- and D/A-quantities

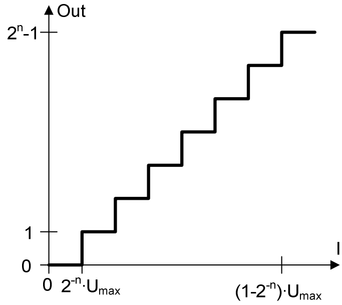
n bits gives N=2^n different values

Resolution $\Delta = \frac{U_{max}}{N} = \frac{U_{max}}{2^n}$ [Volts]

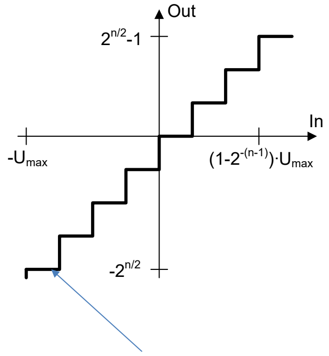
or $\frac{1}{N} = \frac{1}{2^n}$

Unipolar and bipolar converters

Unipolar

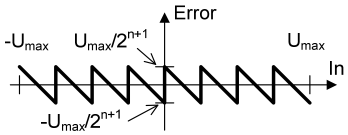


Bipolar



One more negative step
than the positive steps

Conversion error, SQNR



Rounding error

Unipolar

$$\mathrm{SQNR}_{\mathrm{unipolar}} = 20 \cdot {}^{10}\log\frac{\text{maximal level}}{\text{maximal error level}} = 20 \cdot {}^{10}\log\frac{U_{max}}{\frac{\Delta}{2}} = 20 \cdot {}^{10}\log\frac{U_{max}}{\frac{U_{max}}{2 \cdot N}} =$$

$$= 20 \cdot {}^{10}\log\frac{U_{max}}{\frac{U_{max}}{2 \cdot 2^n}} = 20 \cdot {}^{10}\log(2^{n+1}) = 20 \cdot (n+1) \cdot {}^{10}\log(2) = 6.02 \cdot (N+1) \approx 6 \cdot (N+1)$$

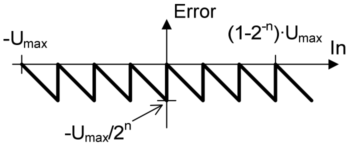
Conversion error, SQNR

Bipolar

$$\mathrm{SNQR}_{\mathrm{bipolar}} = 20 \cdot {}^{10}\log\frac{\text{maximal amplitude}}{\text{maximal error level}} = 20 \cdot {}^{10}\log\frac{\frac{U_{max}}{2}}{\frac{\Delta}{2}} = 20 \cdot {}^{10}\log\frac{\frac{U_{max}}{2}}{\frac{U_{max}}{2 \cdot N}} =$$

$$= 20 \cdot {}^{10}\log\frac{\frac{U_{max}}{2}}{\frac{U_{max}}{2 \cdot 2^n}} = 20 \cdot {}^{10}\log(2^n) = 20 \cdot n \cdot {}^{10}\log(2) = 6.02 \cdot n \approx 6 \cdot n$$

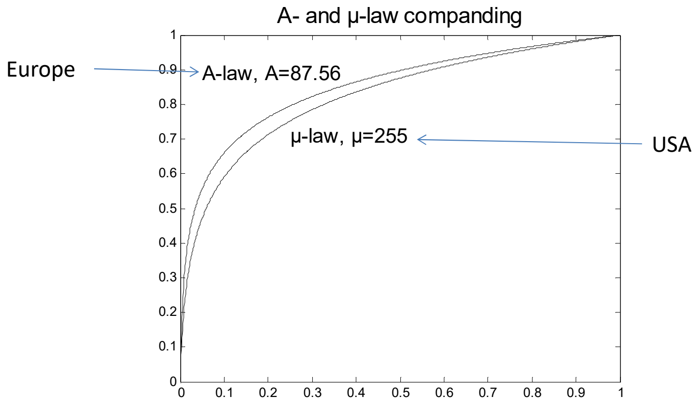
Number of bits	*8*	*12*	*16*	*20*	*24*
SQNR [dB]	*48*	*72*	*96*	*120*	*144*



Truncation error

Demonstration

Logarithmic converter compander



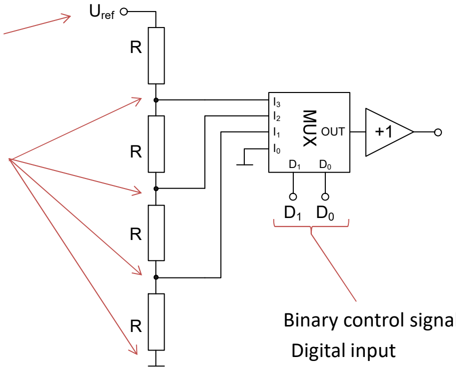
Compander=compressor/expander

Used in communication devices like mobile phones

Let´s start with D/A converters since they are a bit simplier than A/D converters and can be used as building blocks for A/D converters

D/A converters

Resistance ladder converter



Voltage span

Voltage levels

Analog output

Binary control signal

Digital input

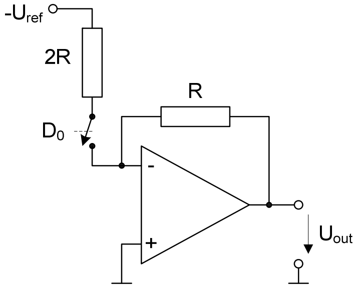
2^n resistors

Large MUX

Can only be used for a limited number of bits

Current summing D/A

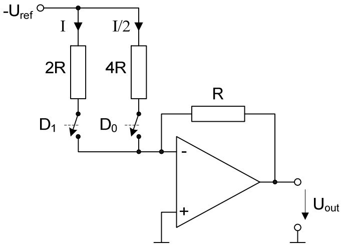
One bit



$$U_{out} = \frac{U_{max}}{2^1} = \frac{U_{max}}{2}$$

Current summing D/A

Two bits



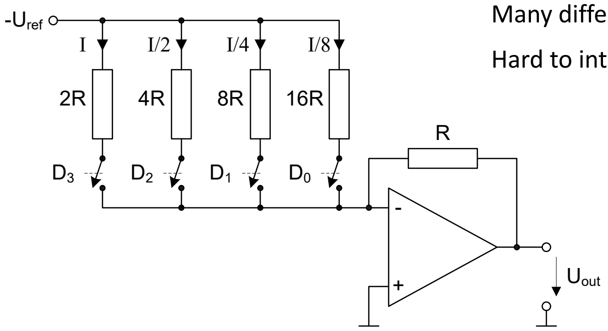
$$U_{out,bit\ 0} = \frac{U_{max}}{2^2} = \frac{U_{max}}{4}$$

$$U_{out,bit\ 1} = \frac{U_{max}}{2^1} = \frac{U_{max}}{2}$$

$$U_{out} = D_1 \cdot \frac{U_{max}}{2} + D_0 \cdot \frac{U_{max}}{4}$$

Current summing D/A cont.

Four bits



Many different resistor values

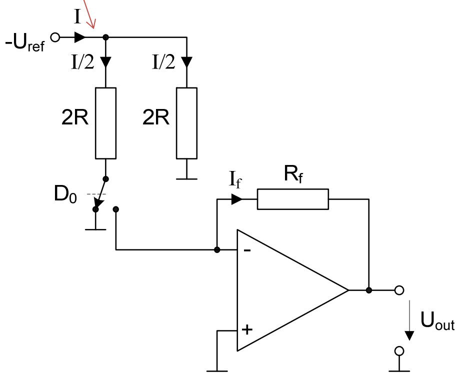
Hard to integrate with precision

$$U_{out} = D_3 \cdot \frac{U_{max}}{2^1} + D_2 \cdot \frac{U_{max}}{2^2} + D_1 \cdot \frac{U_{max}}{2^3} + D_0 \cdot \frac{U_{max}}{2^4}$$

R-2R ladder D/A

One bit

Constant current



$$I = \frac{U_{ref}}{2\cdot R} + \frac{U_{ref}}{2\cdot R} = \frac{U_{ref}}{R}$$

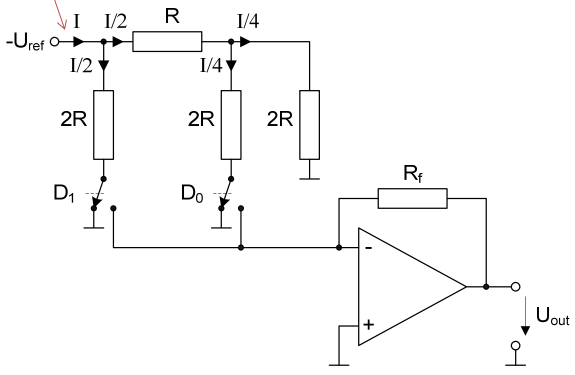
$$I_f = \frac{I}{2}$$

$$\Delta = I_f \cdot R_f = \frac{U_{max}}{2^1} = \frac{U_{max}}{2}$$

R-2R ladder D/A cont.

Two bits

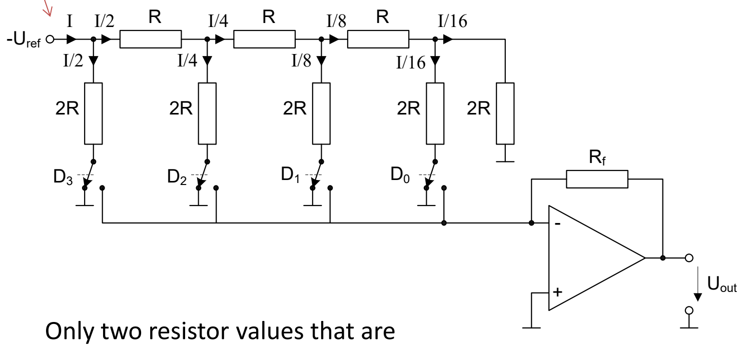
Constant current



R-2R ladder D/A cont.

Constant current

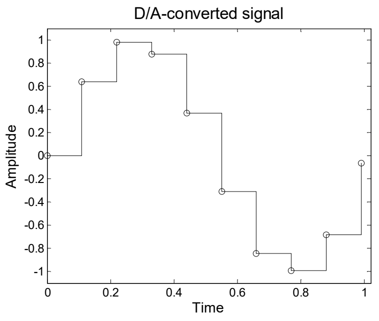
Four bits



Only two resistor values that are quite close in value

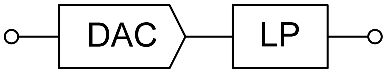
Easier to integrate with precision

Smoothing filter

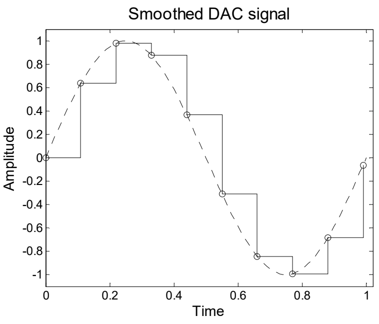


The analog output from the D/A looks like a staircase.

It should be smoothed by a lowpass filter.

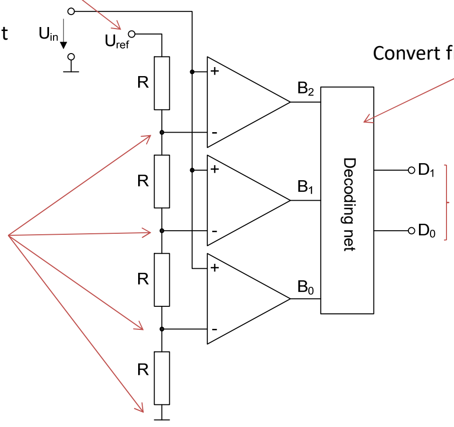


Demonstration



A/D converters

Flash converter



Voltage span

Analog input

Convert from bargraph to binary

Digital output

Reference levels

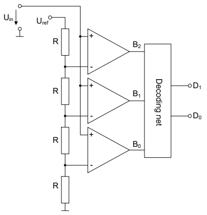
2^n resistors

2^n-1 comparators

Can only be used for a limited number of bits

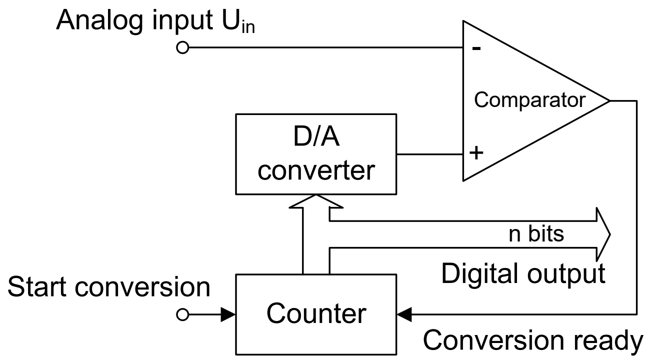
A/D converters

Flash converter cont.



Interval	*Bits from comparators*	*Binary word*
$0-\frac{U_{ref}}{4}$	*000*	*00*
$\frac{U_{ref}}{4}-\frac{U_{ref}}{2}$	*001*	*01*
$\frac{U_{ref}}{2}-\frac{3\cdot U_{ref}}{4}$	*011*	*10*
$\frac{3\cdot U_{ref}}{4}-U_{ref}$	*111*	*11*

Up-counting ADC


Analog input U_{in}

Comparator

D/A converter

n bits

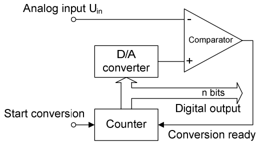
Digital output

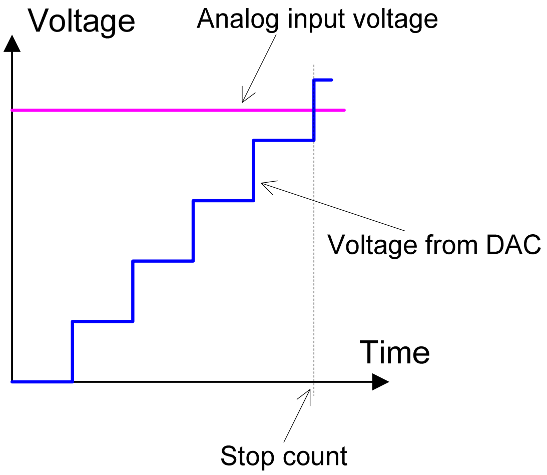
Start conversion

Counter

Conversion ready

Up-counting ADC cont.





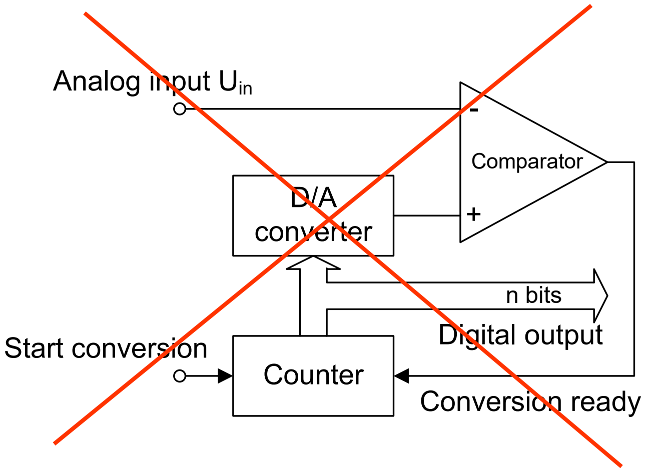
Up-counting ADC cont.

Limitations

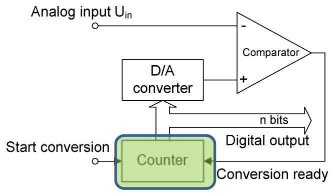
- Large variation in conversion time
  The conversion time depends on the level of the input voltage

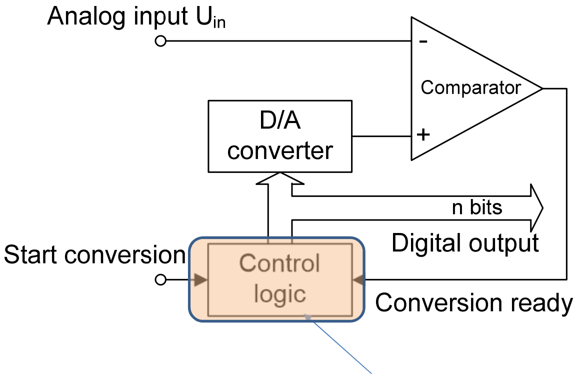
- Unpredictable conversion time

Up-counting ADC cont



ADC using successive approximation

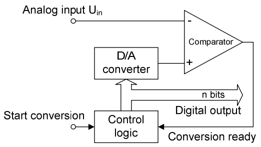




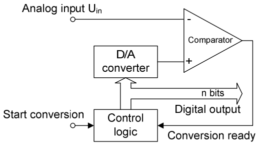
The counter has been replaced

ADC cont.

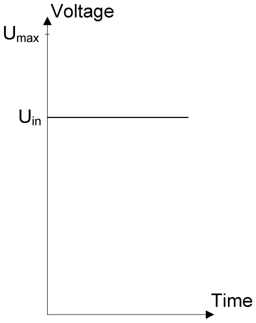
We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result



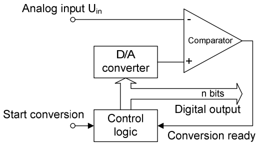
ADC cont.



We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result

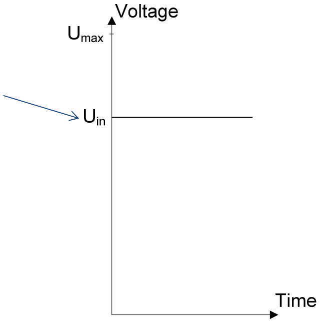


ADC cont.

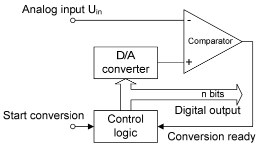


We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result

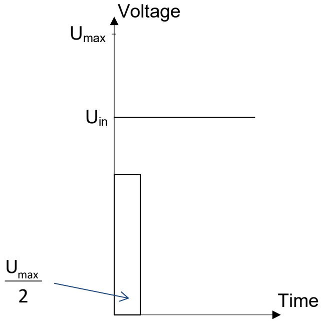
Analog input value



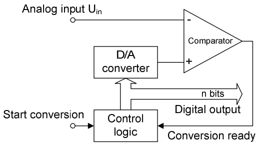
ADC cont.



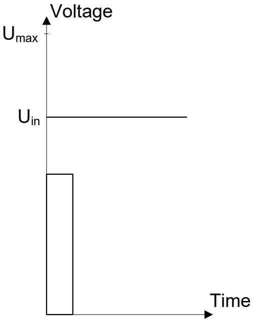
We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result



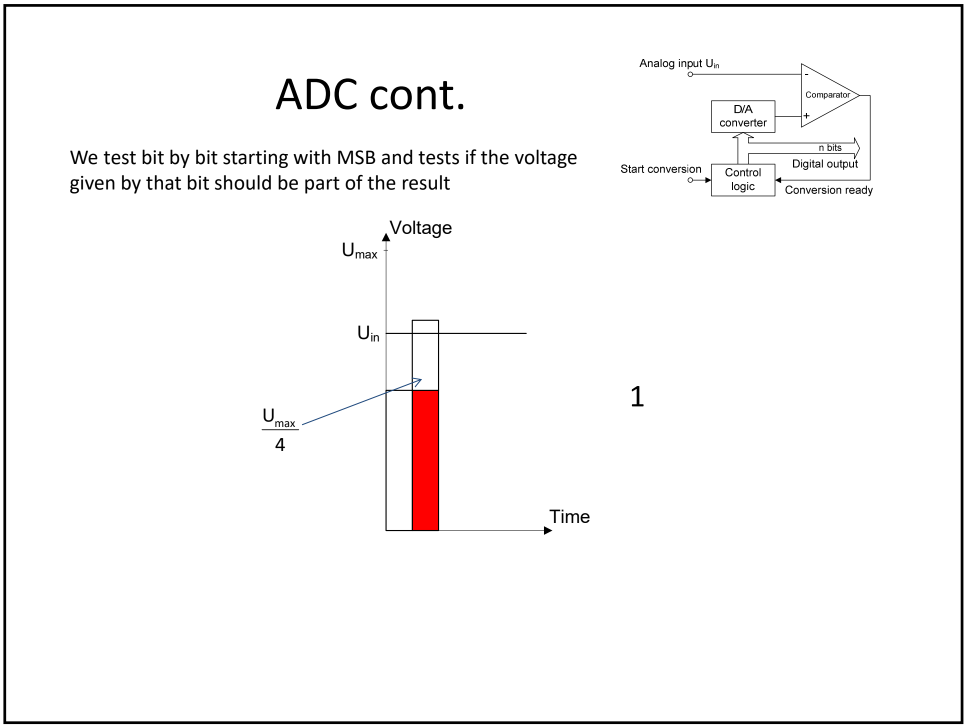
ADC cont.



We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result



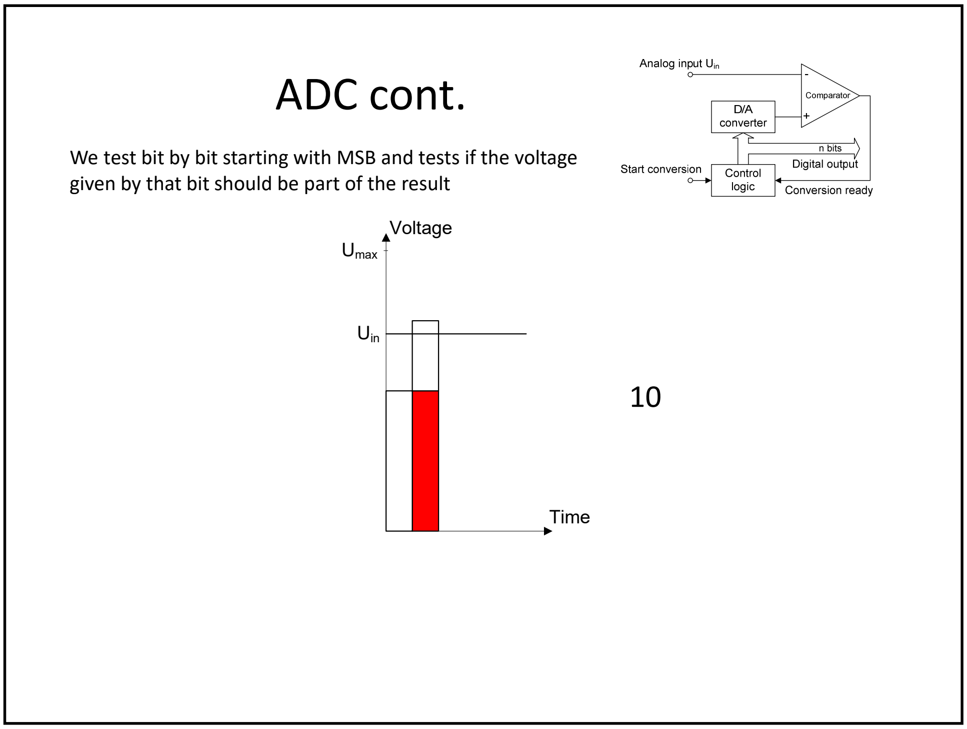
1



ADC cont.

We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result

1

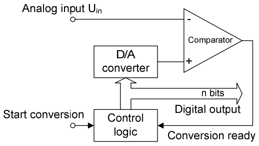


ADC cont.

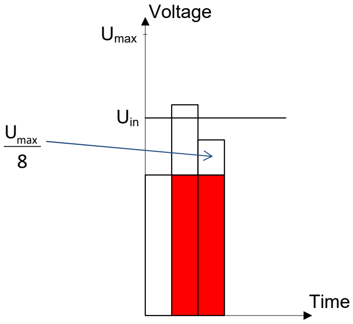
We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result

10

ADC cont.

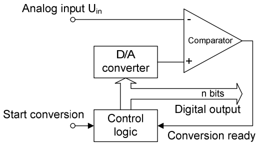


We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result

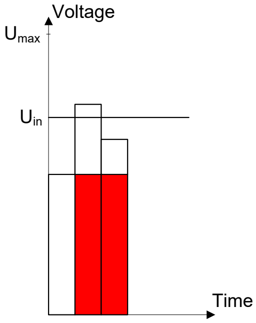


10

ADC cont.



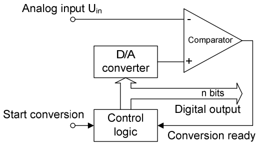
We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result

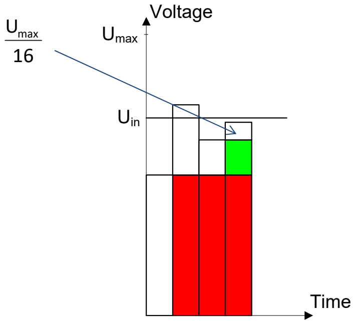


101

ADC cont.

We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result

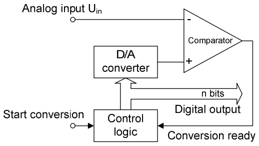


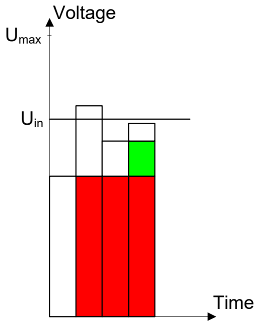


101

ADC cont.

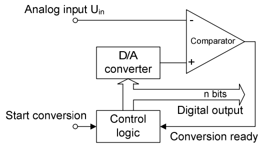
We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result



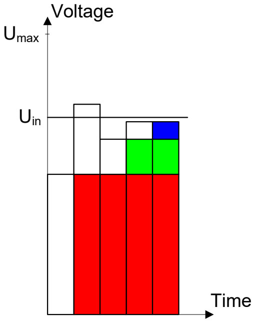


1011

ADC cont.



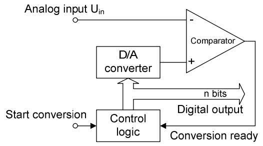
We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result



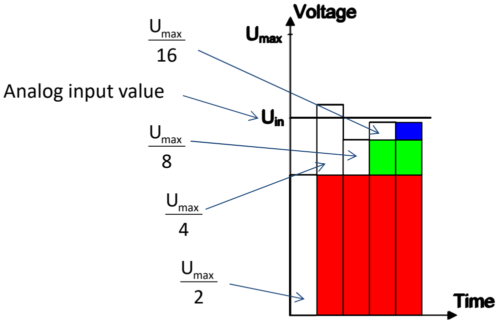
1011

The conversion will take as many clock cycles as there are bits in the binary word

ADC cont.



We test bit by bit starting with MSB and tests if the voltage given by that bit should be part of the result



1011

The conversion will take as many clock cycles as there are bits in the binary word

Sample & Hold

For an ADC the analog value should preferably be kept constant during the conversion.

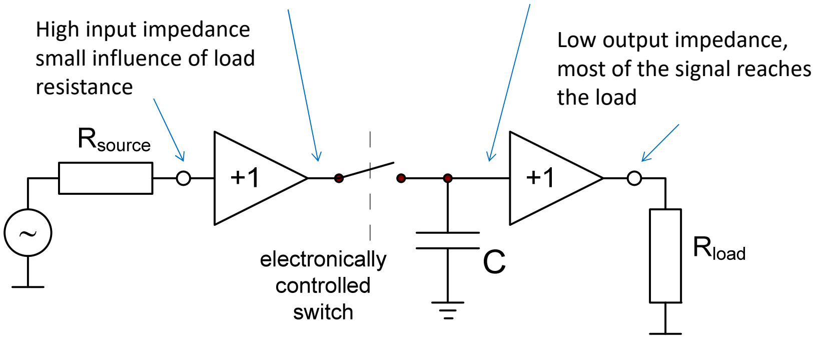
We use a sample and hold device, aka track and hold.

Low impedance, short time constant τ, fast charging of capacitor

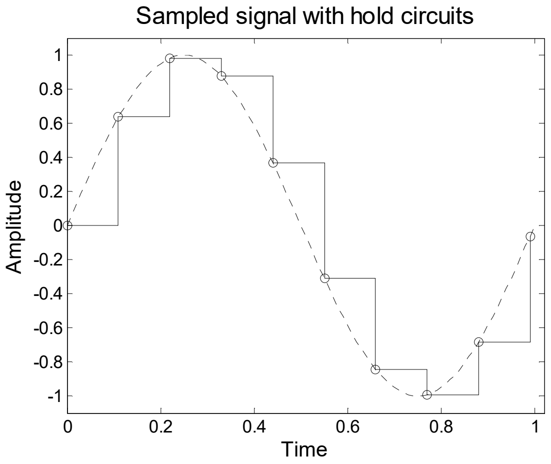
High impedance, large time constant, slow discharge of capacitor

High input impedance small influence of load resistance

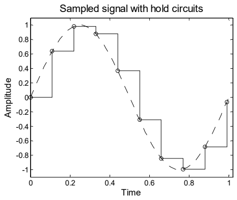
Low output impedance, most of the signal reaches the load

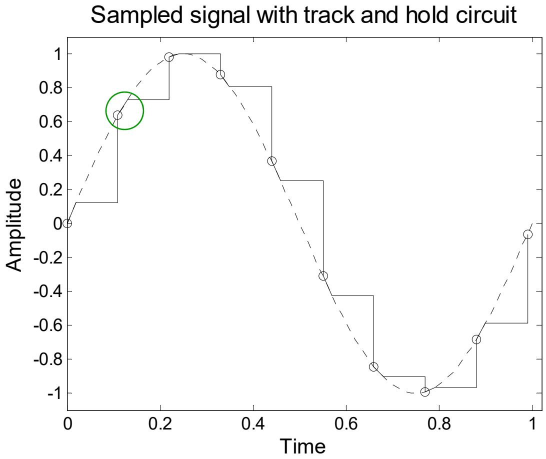


Sample & Hold cont.

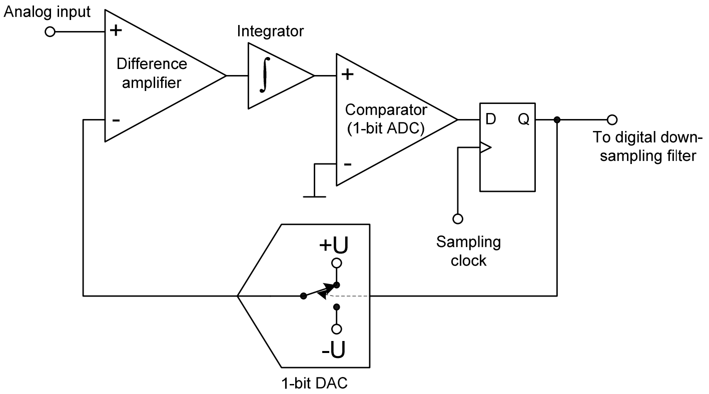


Sample & Hold cont.

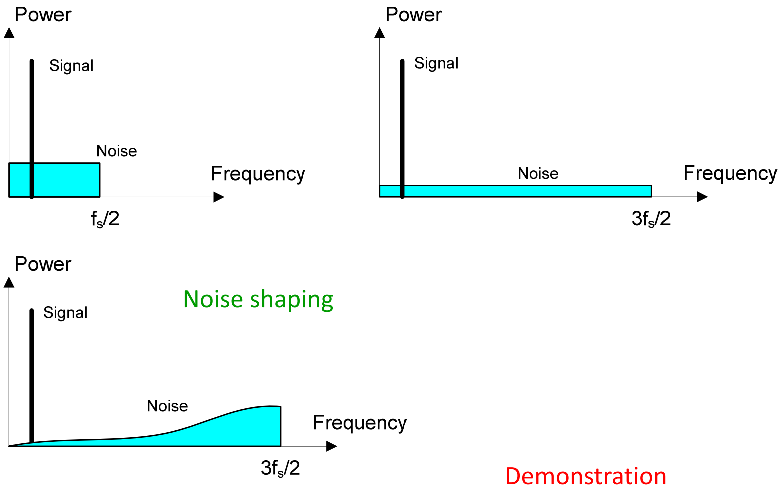




Sigma-delta modulator



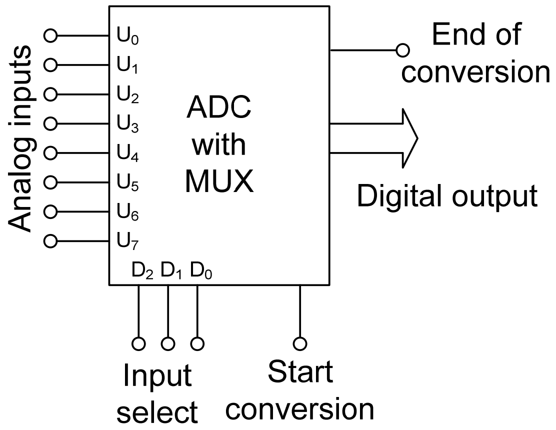
Sigma-delta modulators and noise



Noise shaping

Demonstration

ADC with multiplexed inputs

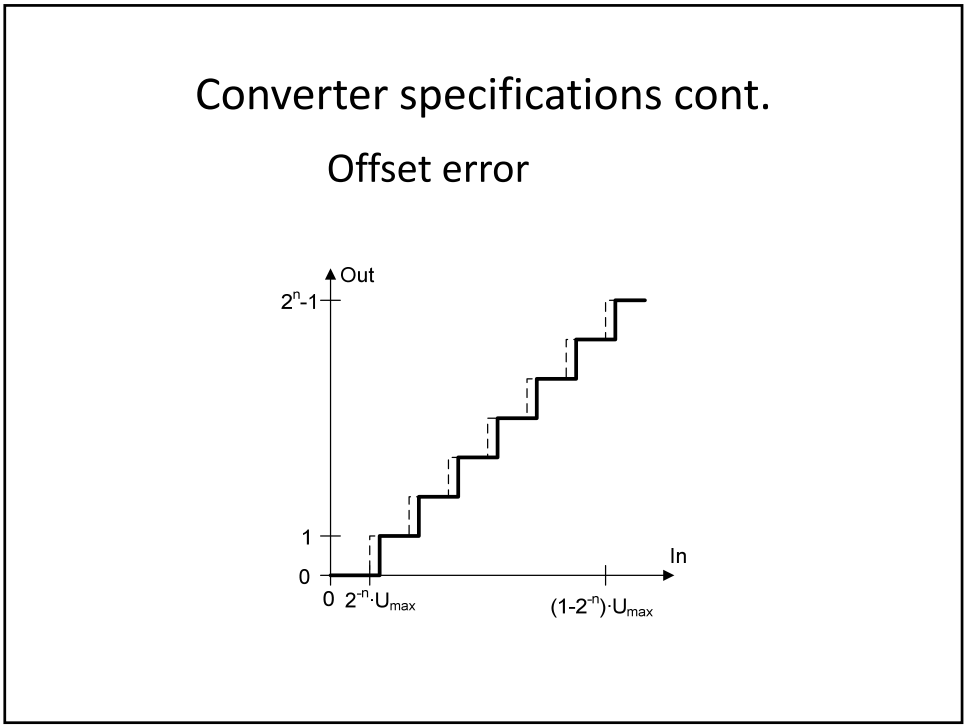


ADC with multiplexed inputs cont.

- Maximal sampling frequency with single channel $f_{s,\max}$
- Maximal sampling frequency with N channels $\dfrac{f_{s,\max}}{N}$

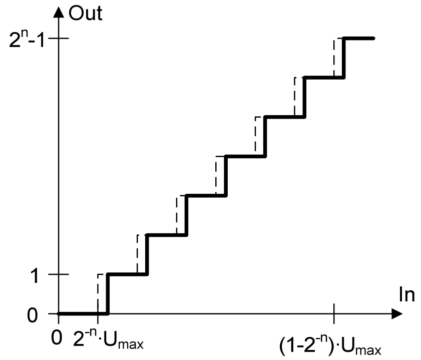
Converter specifications

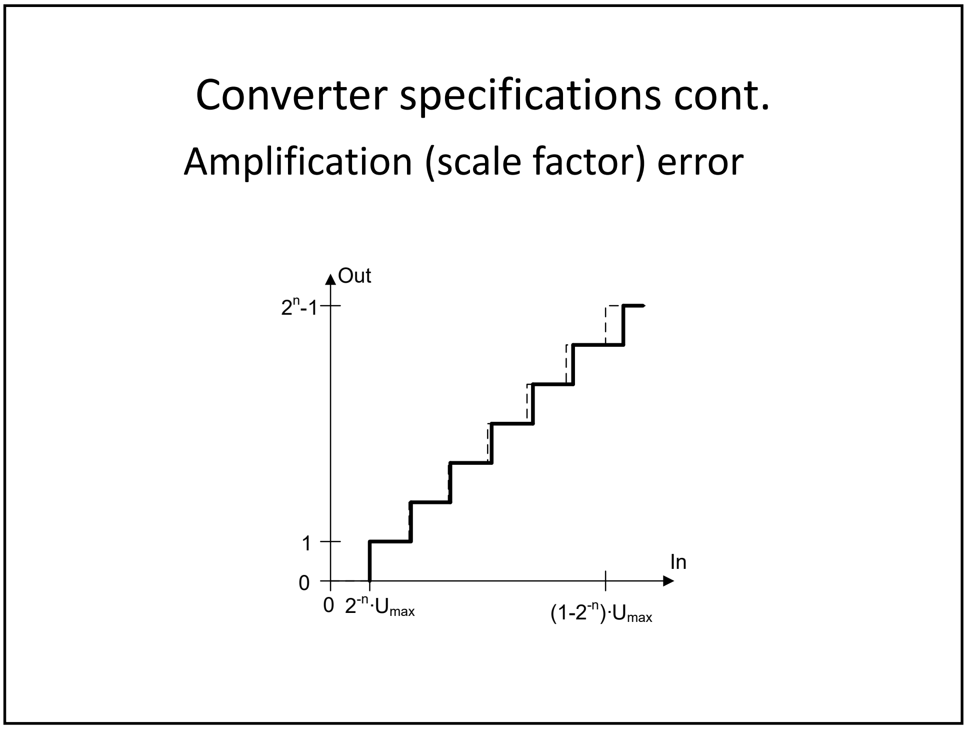
- Voltage span
- Resolution
- Accuracy
- Conversion time (ADC)
- Settling time (DAC)
- Offset error
- Amplification (scale factor) error
- Linearity error



Converter specifications cont.

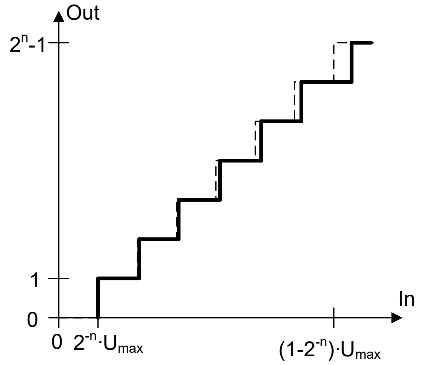
Offset error





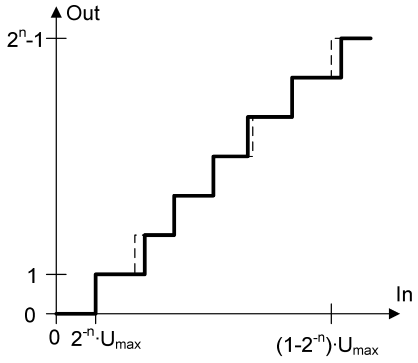
Converter specifications cont.

Amplification (scale factor) error

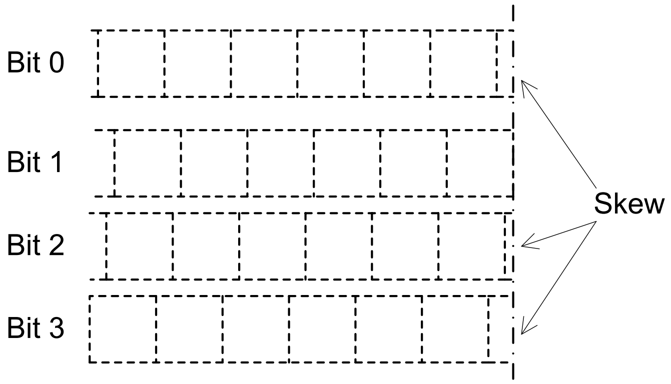


Converter specifications cont.

Linearity error



Skew in parallel communication



The bits might be falsely read.

We tend to go for serial interfaces.

This will also decrease the number of wires

SPI

The SPI (Serial Peripheral Interface) is a synchronous serial comminication interface

It is a master slave protocol with one master and one or more slaves

It uses four different signals

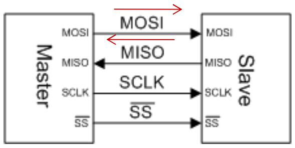
- DOUT, (MOSI = Master Out Slave In). Data from the master to the slave
- DIN, (MISO = Master In Slave Out). Data from the slave to the master
- SCLK, serial clock for synchronisation of the transfer
- SS, slave select to enable the slave

If we have more than one slave we will need several slave select signals to separate them and only activate one of the slaves at each time

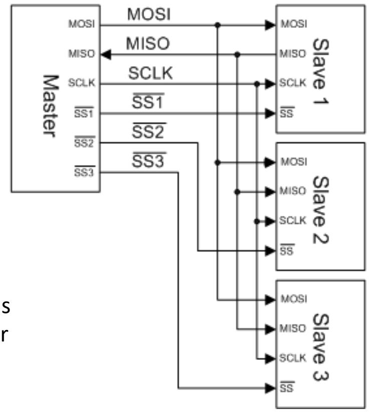
The transfer rate differs between different devices but could be up to 10 Mbit/s

SPI

One slave



Three slaves



At the same rate as data is shifted from the master to the slave, data is shifted from the slave to the master

The two are united into one shift register

DAC using SPI

Let´s start with our DAC chip since it is somewhat simpler than the ADC chip

We will only give some basic information. You will have to read the details in the data sheets for the devices

Find the data sheets by browsing the internet

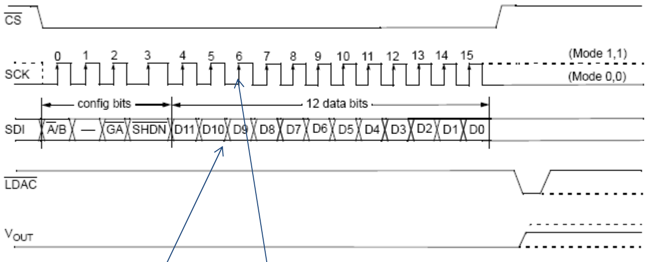
Our DAC chip is called MCP4822 and comes from the vendor Microchip, well known for their PIC processors

It has two channels and uses 12 bit data words

In this case the SPI interface is somewhat simplified since we are only sending configuration bits and data to the chip (MOSI).
We are not receiving any data from the DAC and the DIN (MISO) pin is removed

DAC using SPI

Let´s look at the transfer protocol



The device triggers on the positive edge of the SCLK signal

This means that the bits we want to send will have to be placed on the DOUT (here SDI) line before that so they are stable when the clock edge comes.

In the time diagram it looks like the data needs to be placed on DOUT at the negative edge of the clock but this is not necessary

It can actually be placed there on the positive flank before that since the trigging at that time has already occured if we do so

ADC using SPI

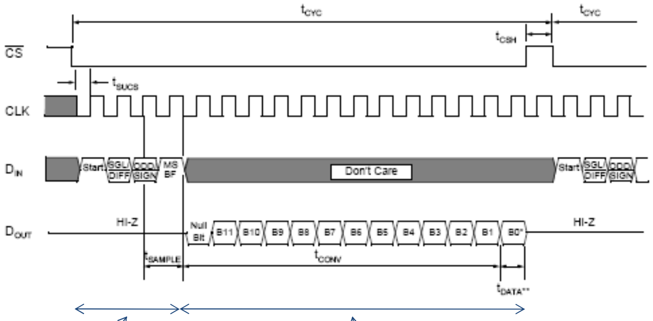
Our ADC chip is also from Microchip and is called MCP3202

It has two channels and uses 12 bit data words

The SPI interface here is somewhat more complicated than the interface for the DAC since in this case we have to send configuration bits to the chip and get AD converted data back from the device

ADC using SPI

Let´s look at the transfer protocol



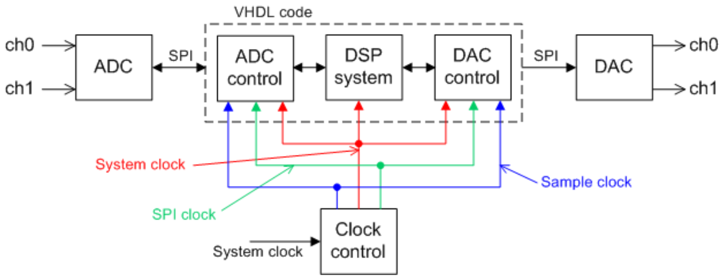
Configuration to the ADC

Data from the ADC

Bidirectional communication

Sampled system

Let´s create a complete sampled system



Observe that the system clock and the sample clock are not the same. Since we need to do some calculations between the sampling times the system clock will have to have a higher frequency.

There is also need for a serial clock for the SPI communication

Sampled system cont.

We need a number of clocks for the system

- The system clock (100 MHz)
- The sample clock (40 kHz)
- The SPI clock

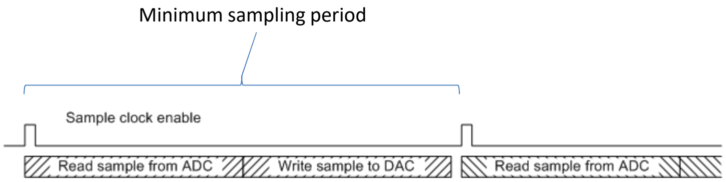
The frequency of the SPI clock will have to be high enough for us to read one sample from the ADC and write one sample to the DAC before the next pulse comes from the sample clock.

We shall soon see that this is not necessarily true since receiving one sample from the ADC and sending one sample to the DAC between each sample pulse can take place in parallel

Try to use clock enable signals instead of divided clock signals for the two lower clock frequencies (f_s and f_{SPI})

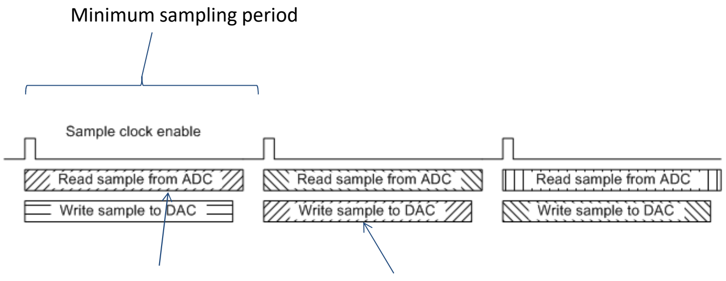
The clock signal sent to the physical A/D and D/A converts needs t be symmetrical though

Sampled system cont.



Can we improve the situation and use a higher sampling rate?

Sampled system cont.



The value that we read from the ADC here is written to the DAC here

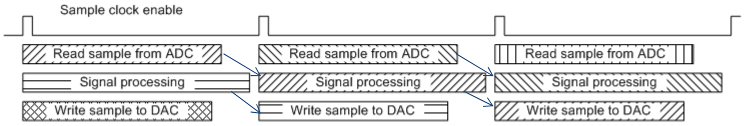
What´s the snag? We have one sampling periods delay.

But! Since we do ADC and DAC in parallel during a sample period the sampling period can be shortened and the sampling frequency can be increased.

The maximum sampling frequency is given by the conversion that takes the longest time, ADC or DAC

Sampled system cont.

A general system



Read sample n	Read sample n+1	Read sample n+2
Process sample n-1	Process sample n	Process sample n+1
Write sample n-2	Write sample n-1	Write sample n

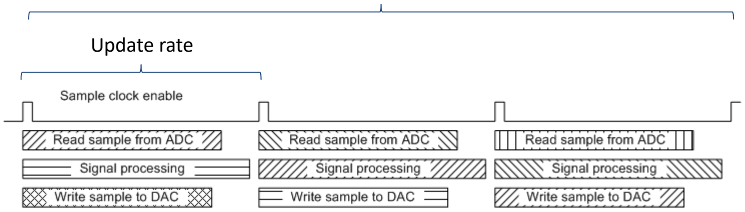
The system is pipelined

Sampled system cont.

A general system cont.

Time for a sample to pass the system

Update rate



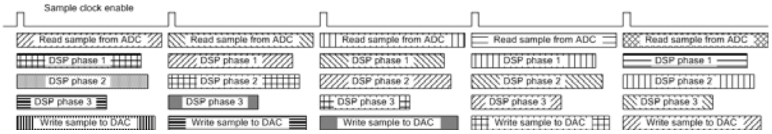
The update rate, how often we can read new samples, is called the **through put** of the system

The time for a sample to pass the system is called the **latency**, the delay of the system

Sampled system cont.

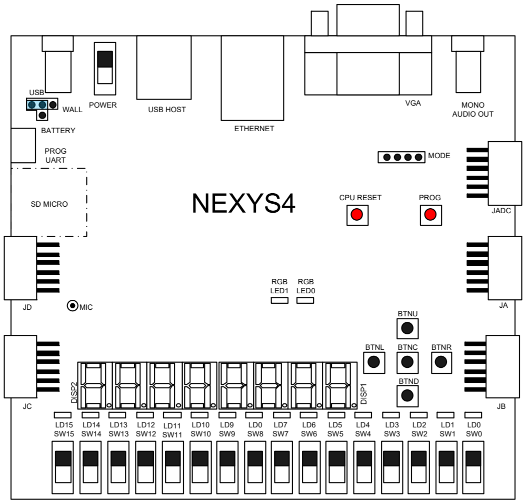
A general system cont.

In more advanced systems the DSP part could be split into sections that use more than one sampling period



Our lab system again

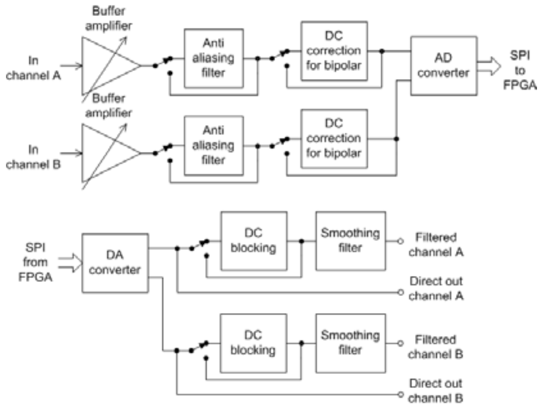
Xilinx Nexys4
Trainer Board



Our lab system again

AD/DA extention card

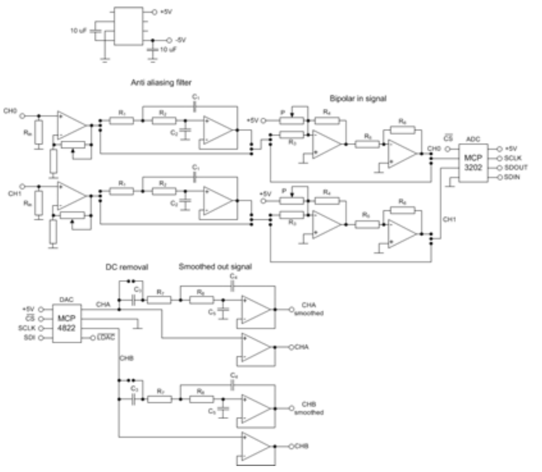
Block schematic



Our lab system again

AD/DA extention card

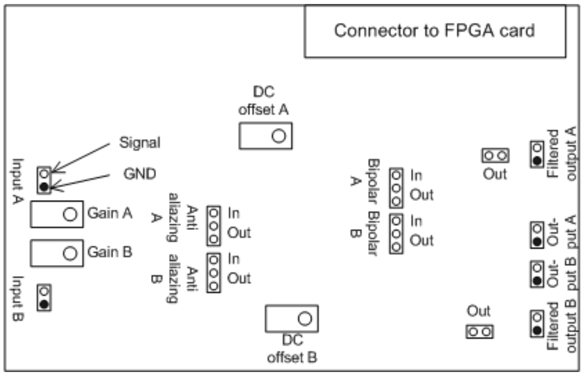
Schematic



Our lab system again

AD/DA extention card

Layout

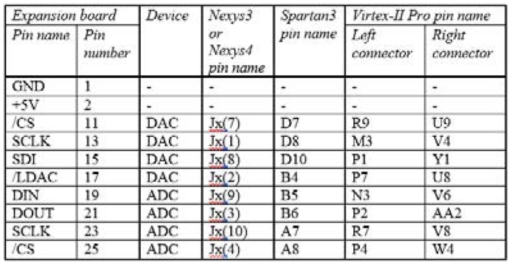


Our lab system again

AD/DA extention card

Pin connections

The AD/DA card can be used for a number of FPGA boards with devices from different FPGA families.



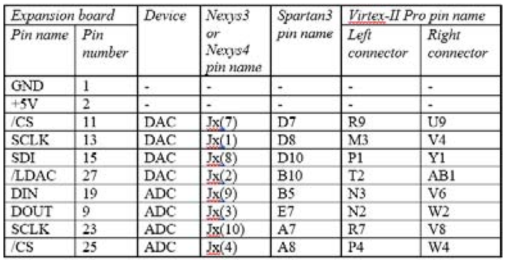
Expansion board		*Device*	*Nexys3 or Nexys4 pin name*	*Spartan3 pin name*	*Virtex-II Pro pin name*	
Pin name	*Pin number*				*Left connector*	*Right connector*
GND	1	-	-	-	-	-
+5V	2	-	-	-	-	-
/CS	11	DAC	Jx(7)	D7	R9	U9
SCLK	13	DAC	Jx(1)	D8	M3	V4
SDI	15	DAC	Jx(8)	D10	P1	Y1
/LDAC	17	DAC	Jx(2)	B4	P7	U8
DIN	19	ADC	Jx(9)	B5	N3	V6
DOUT	21	ADC	Jx(3)	B6	P2	AA2
SCLK	23	ADC	Jx(10)	A7	R7	V8
/CS	25	ADC	Jx(4)	A8	P4	W4

It can be used for boards FPGAs from the families Virtex-II, Spartan3, Spartan6 and Artix-7. The Spartan 6 device is placed on a Nexys3 board while the Artix-7 device is placed on a Nexys4 board.

Our lab system again

AD/DA extention card

Alternate pin connections

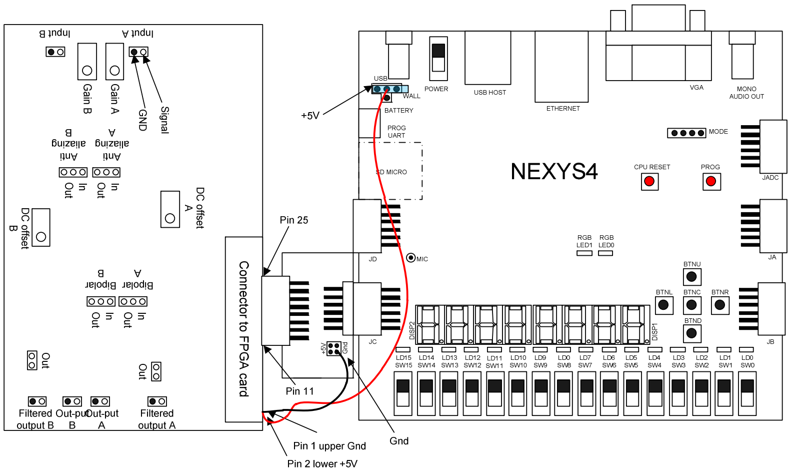


Expansion board		Device	Nexys3 or Nexys4 pin name	Spartan3 pin name	Virtex-II Pro pin name	
Pin name	Pin number				Left connector	Right connector
GND	1	-	-	-	-	-
+5V	2	-	-	-	-	-
/CS	11	DAC	Jx(7)	D7	R9	U9
SCLK	13	DAC	Jx(1)	D8	M3	V4
SDI	15	DAC	Jx(8)	D10	P1	Y1
/LDAC	27	DAC	Jx(2)	B10	T2	AB1
DIN	19	ADC	Jx(9)	B5	N3	V6
DOUT	9	ADC	Jx(3)	E7	N2	W2
SCLK	23	ADC	Jx(10)	A7	R7	V8
/CS	25	ADC	Jx(4)	A8	P4	W4

Since the AD/DA card is made for a older Spartan3 FPGA board you will need an adapter between the Nexys3/4 board and the AD/DA bord. There is a paper on the homepage, Connections between the Nexys4 board and the AD/DA board, giving information on how this is done

Our lab system again

You also have it here


Input B
Input A
Gain B
Gain A
GND
Signal
Anti alaizing B
Anti alaizing A
In
Out
In
Out
DC offset A
DC offset B
Bipolar B
Bipolar A
In
Out
In
Out
Out
Out
Filtered output B
Out-put B
Out-put A
Filtered output A
Connector to FPGA card
Pin 25
Pin 11
Pin 1 upper Gnd
Pin 2 lower +5V
+5V
Gnd
+5V
Gnd
USB
WALL
BATTERY
POWER
USB HOST
ETHERNET
VGA
MONO AUDIO OUT
PROG UART
SD MICRO
MODE
NEXYS4
CPU RESET
PROG
JADC
JA
JB
JC
JD
MIC
RGB LED1
RGB LED0
BTNU
BTNL
BTNC
BTNR
BTND
DISP2
DISP1
LD15 SW15
LD14 SW14
LD13 SW13
LD12 SW12
LD11 SW11
LD10 SW10
LD9 SW9
LD0 SW8
LD7 SW7
LD6 SW6
LD5 SW5
LD4 SW4
LD3 SW3
LD2 SW2
LD1 SW1
LD0 SW0