TECHNISCHE UNIVERSITÄT DRESDEN

Digital Signal Processing with FPGAs

Emil Matus
TU Dresden
emil.matus@tu-dresden.de

Terascale Detector Workshop
Dresden, March'19

Outline

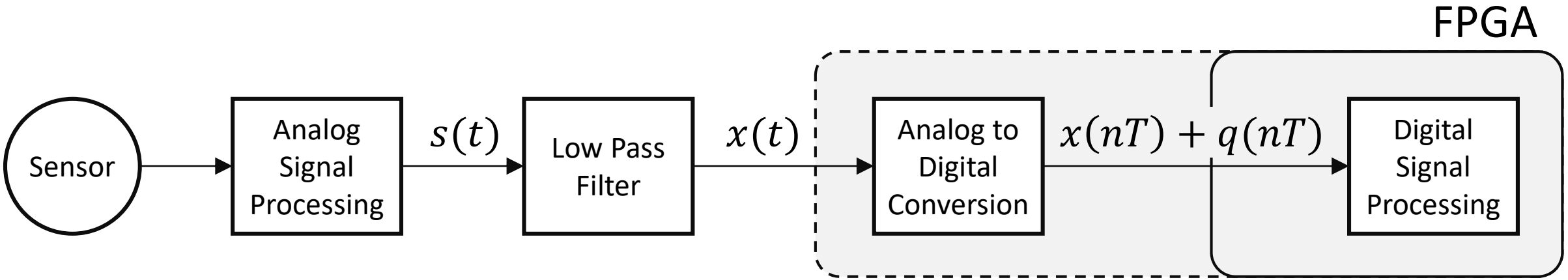
- Digital filter:
  - System theory
  - Filter design
  - Implementation
  - FPGA HW design

TECHNISCHE UNIVERSITÄT DRESDEN

System Theory

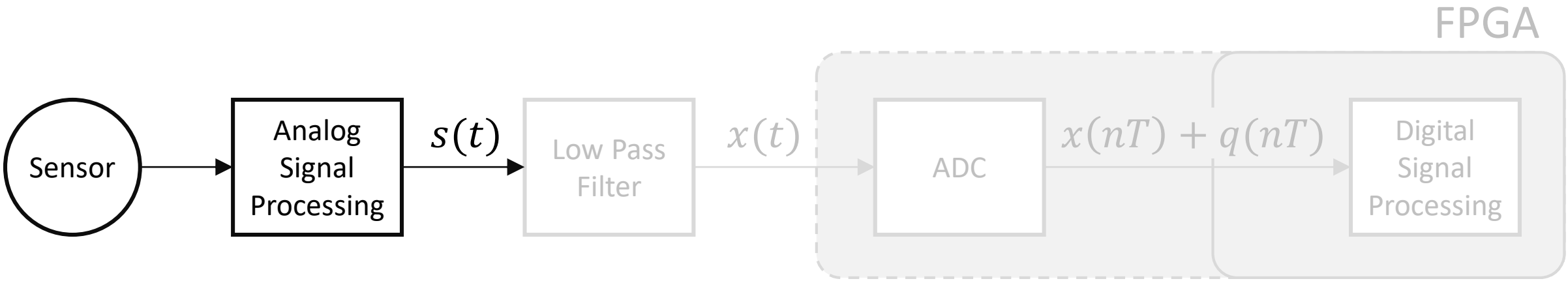
DSP System using FPGA

- DSP - Digital Signal Processing: use ‘digits’ to represent signals and systems

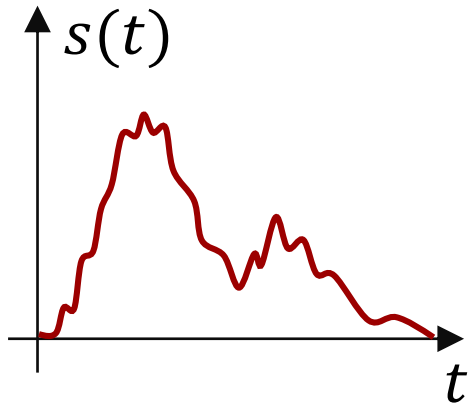


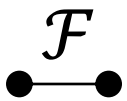
- $s(t)$ Analyzed time-continuous signal
- $x(t)$ Band-limited time-continuous signal
- $x(nT)$ Time-discrete signal with sampling period T
- $q(nT)$ Quantisation noise

Time-Continuous Signal

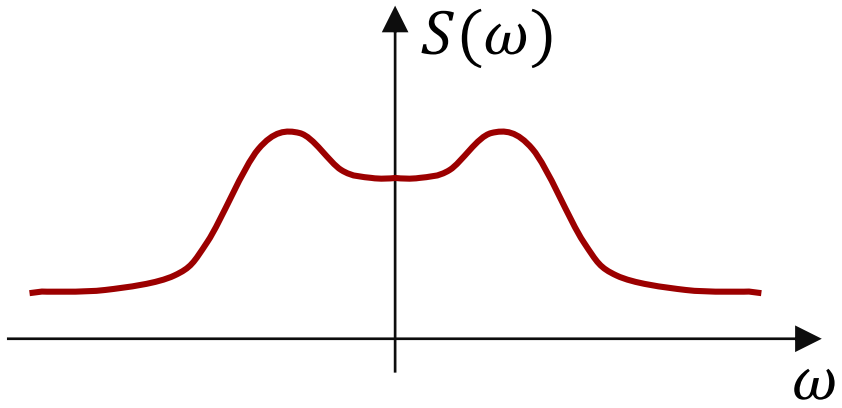


Time-continuous (noisy) signal

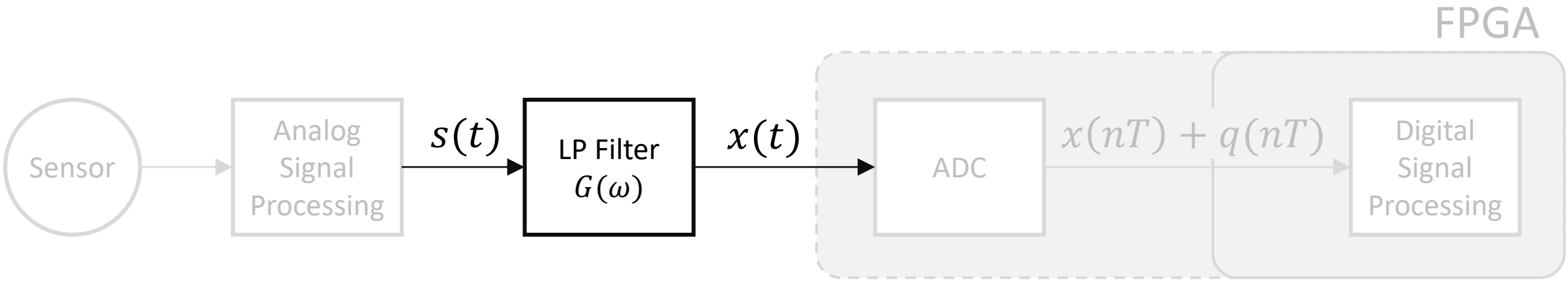


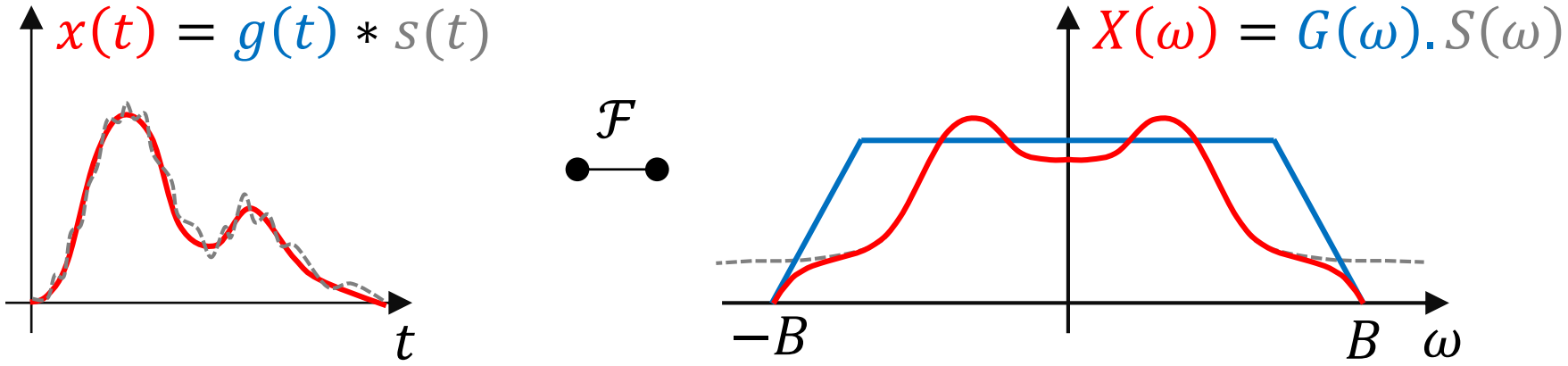


Frequency-continuous spectrum

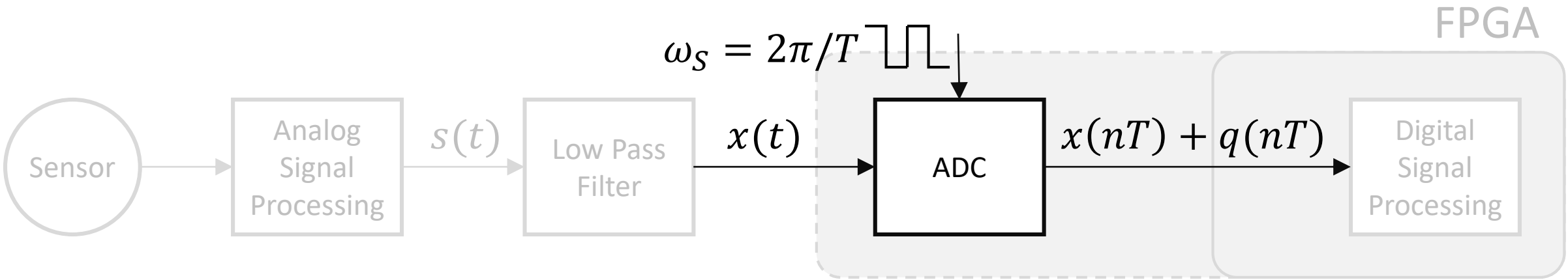


Antialiasing Low-Pass Filter

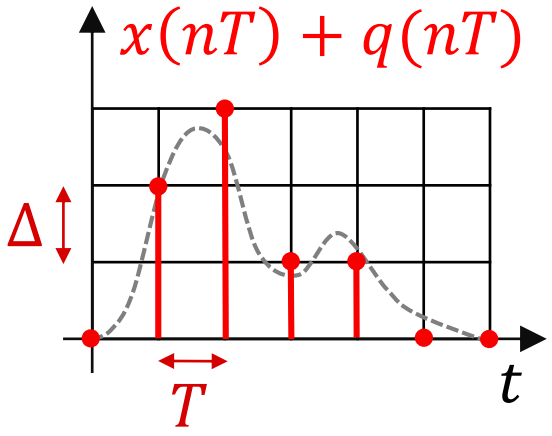




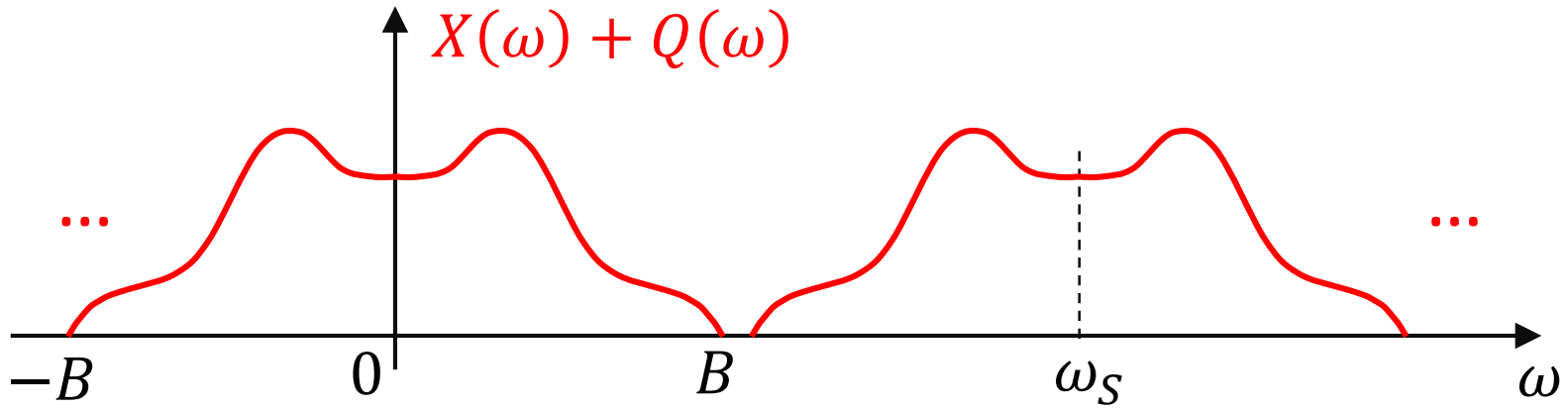
Analog-Digital Conversion: Sampling + Quantisation



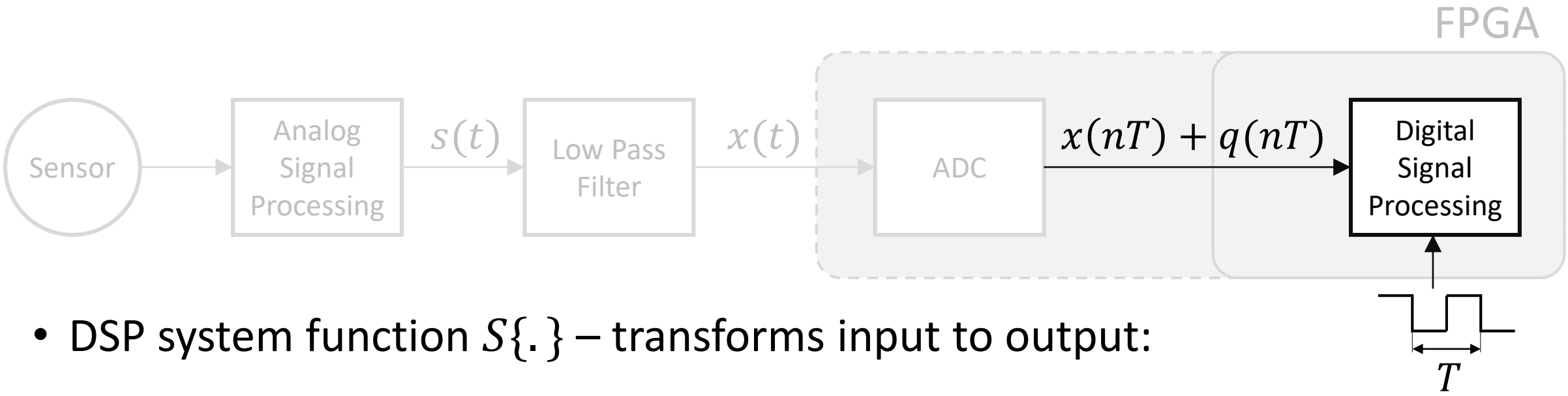
Sampling theorem: $\omega_s \geq 2B$



$\mathcal{F}$

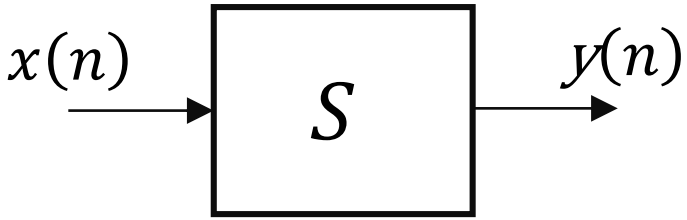


DSP: System Function



- DSP system function $S\{.\}$ – transforms input to output:

$$y(nT) = S\{x(nT)\} \quad \equiv \quad y(n) = S\{x(n)\}$$

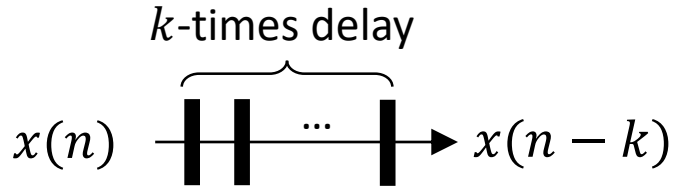


Linear Time-Invariant (LTI) System

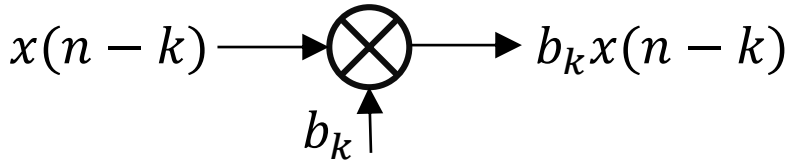
- LTI System - N^{th} order linear difference equation, $b_k, a_k \in \mathbb{R}$:

$$y(n) = \sum_{k=0}^{N-1} b_k x(n-k) - \sum_{k=1}^{N-1} a_k y(n-k)$$

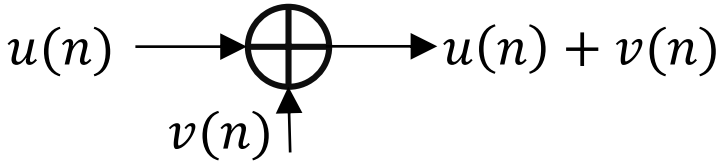
- Time-delay:



- Multiplication by constant:



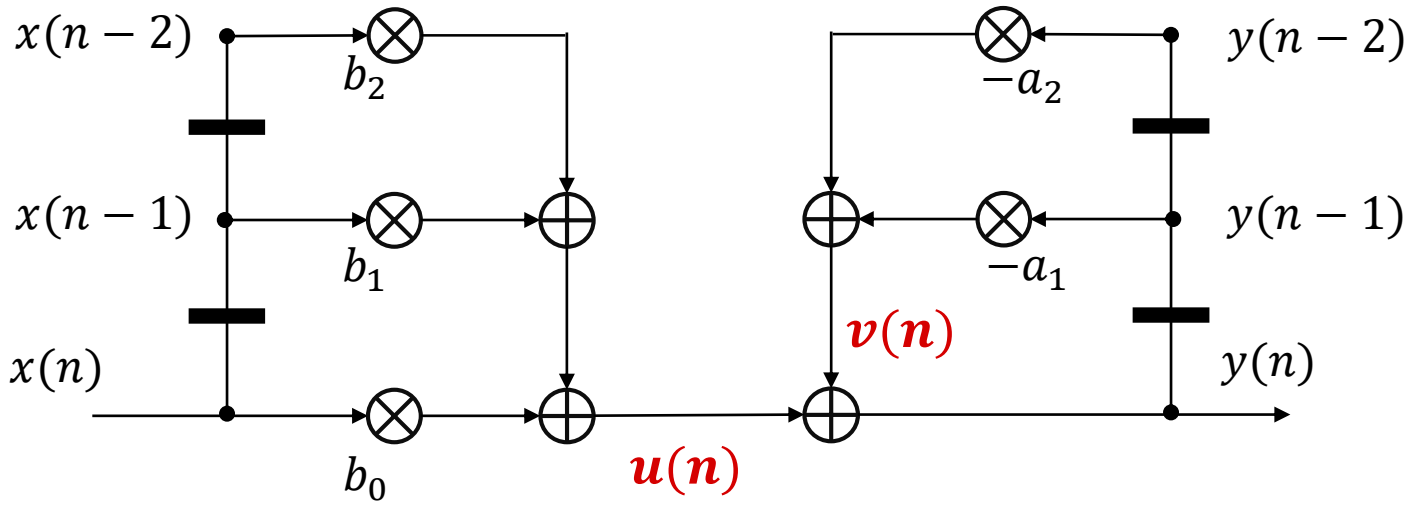
- Addition:



Signal Flow Graph

- Example - 2nd order filter with $N = 2$:

$$y(n) = \underbrace{b_0 x(n) + b_1 x(n-1) + b_2 x(n-2)}_{\boldsymbol{u(n)}} \underbrace{- a_1 y(n-1) - a_2 y(n-2)}_{\boldsymbol{v(n)}}$$



LTI System: Transfer Function

- Apply Z-transform to N^{th} order difference equation with $b_k, a_k \in \mathbb{R}$:

$$x(n-k) \quad \bullet\!\!-\!\!\bullet \quad z^{-k}X(z)$$

$$b_k x(n-k) \quad \bullet\!\!-\!\!\bullet \quad b_k z^{-k}X(z)$$

$$u(n)+v(n) \quad \bullet\!\!-\!\!\bullet \quad U(z)+V(z)$$

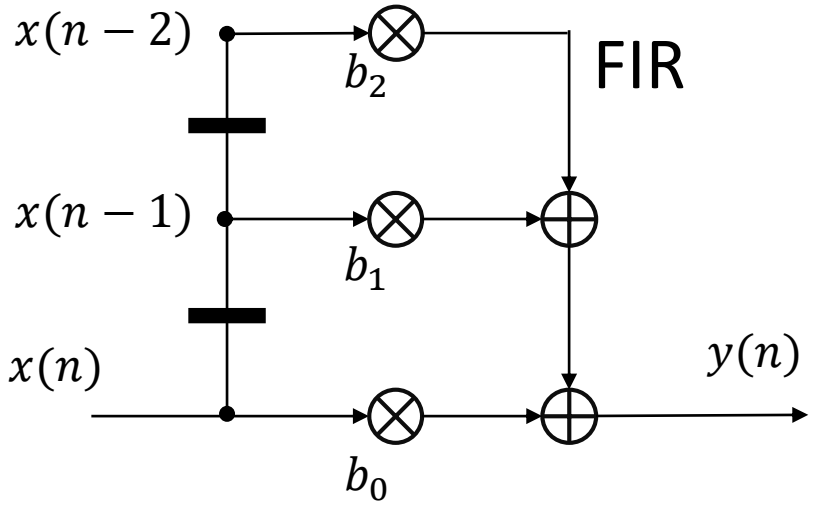
- Transfer function:

$$H(z)=\frac{Y(z)}{X(z)}=\underbrace{\frac{b_0+b_1z^{-1}+\cdots+b_{N-1}z^{-(N-1)}}{a_0+a_1z^{-1}+\cdots+a_{N-1}z^{-(N-1)}}}_{\text{Filter coefficients}\ \boldsymbol{b_k, a_k}}=\frac{b_0}{a_0}\underbrace{\frac{\prod_{k=0}^{N-1}(z-n_k)}{\prod_{k=0}^{N-1}(z-p_k)}}_{\text{Zeros/poles}\ \boldsymbol{n_k, p_k}}$$

FIR vs. IIR Filter

- IIR – Infinite Impulse Response: $a_k \neq 0$, for some $k = 1,2, \dots N-1$

$$H(z) = \frac{b_0 + b_1 z^{-1} + \cdots + b_{N-1} z^{-(N-1)}}{a_0 + a_1 z^{-1} + \cdots + a_{N-1} z^{-(N-1)}} = \sum_{k=0}^{\infty} h(n) z^{-k} \quad \rightarrow \text{Impulse response } \boldsymbol{h(n)}$$



- FIR - Finite Impulse Response:
  $a_0 = 1, a_k = 0, k = 1,2, \dots N-1$:

$$H(z) = \frac{b_0 + b_1 z^{-1} + \cdots + b_{N-1} z^{-(N-1)}}{1} = \sum_{k=0}^{N-1} b_k z^{-k} \quad \rightarrow \quad b_k = h(k)$$

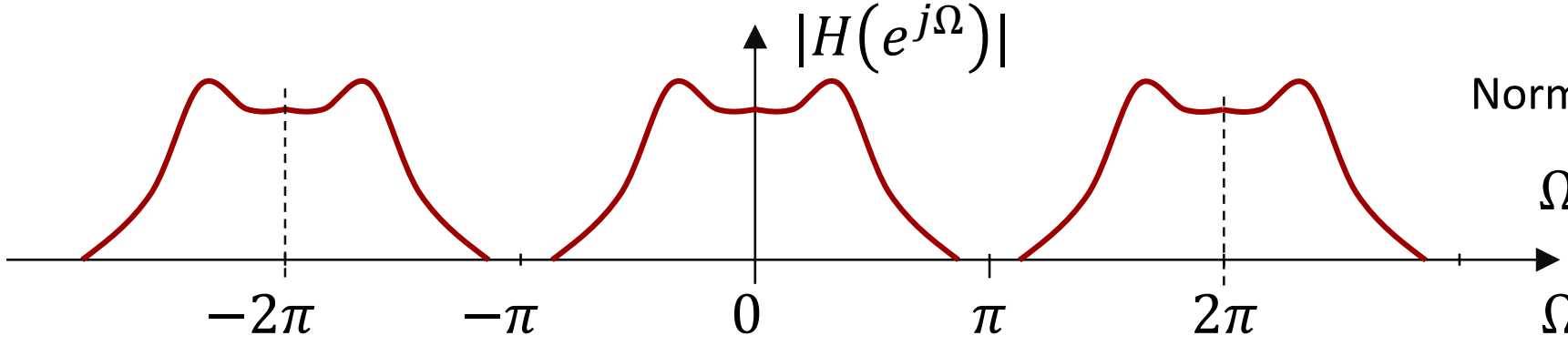
Frequency Response of LTI System

- Transform system function $H(z)$ by substituting $\boldsymbol{z = e^{j\Omega}}$, $\Omega = \omega T$:

$$H\left(e^{j\Omega}\right) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 e^{-j\Omega} + b_2 e^{-j2\Omega} + \cdots + b_{N-1} e^{-j(N-1)\Omega}}{a_0 + a_1 e^{-j\Omega} + a_2 e^{-j2\Omega} + \cdots + a_{N-1} e^{-j(N-1)\Omega}}$$

- Amplitude spectrum:

$$\left|H\left(e^{j\Omega}\right)\right| = \left|\frac{b_0 + b_1 e^{-j\Omega} + \cdots + b_{N-1} e^{-j(N-1)\Omega}}{a_0 + a_1 e^{-j\Omega} + \cdots + a_{N-1} e^{-j(N-1)\Omega}}\right|$$



Normalized sampling frequency:

$$\Omega_S = \omega_S T = \frac{2\pi}{T} T = 2\pi$$

Tools for LTI System Analysis

- Based on knowledge of system function b_k, a_k

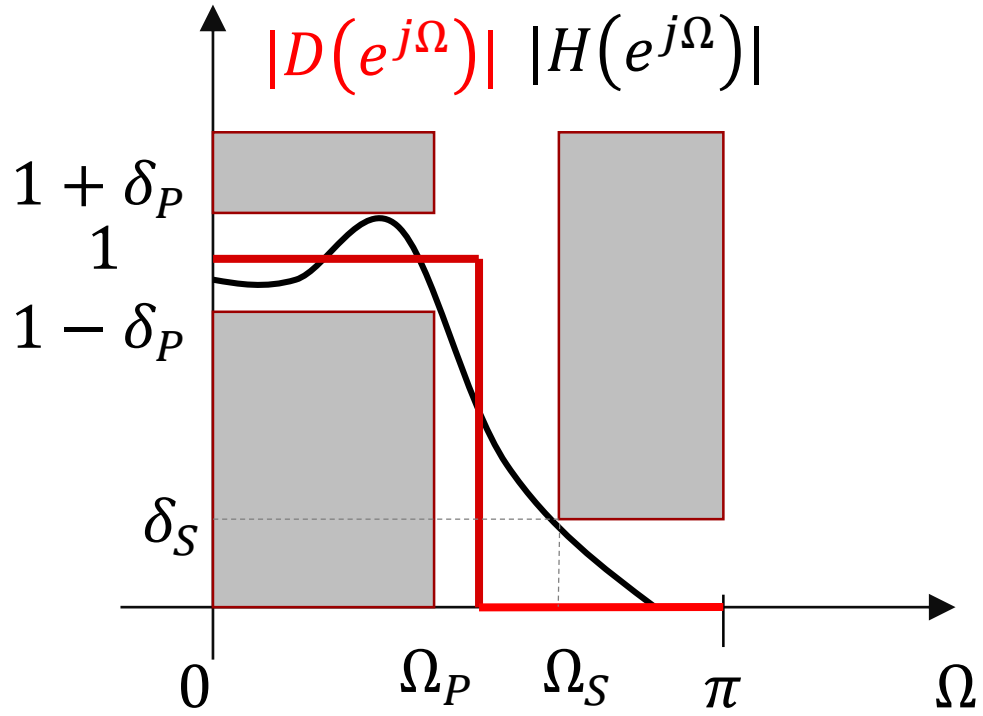
- MATLAB:
  - y=filter(**b, a**, x) - filter signal x
  - [H,W]=freqz(**b, a**) - spectral response
  - zplane(**b, a**) - zeros, poles locations
  - …

TECHNISCHE UNIVERSITÄT DRESDEN

Digital Filter Design

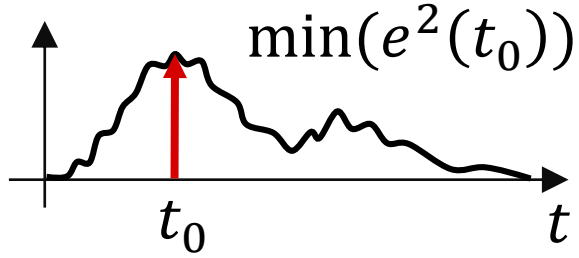
Filter Design 1

- Determine $\boldsymbol{N} \in \mathbb{N}$ and $\boldsymbol{b_k}, \boldsymbol{a_k} \in \mathbb{R}$ according to specific criteria :
  - Frequency response specification: $\min\limits_{b_k, a_k} \max\limits_{\Omega \in S} |H(\Omega) - D(\Omega)|$

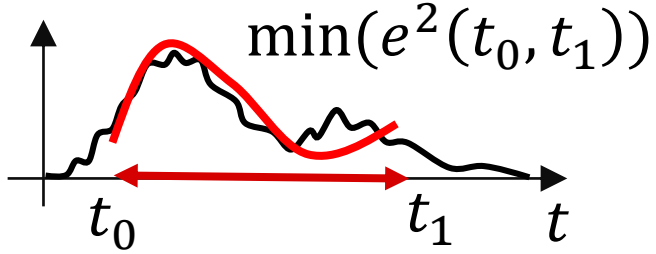


Filter Design 2

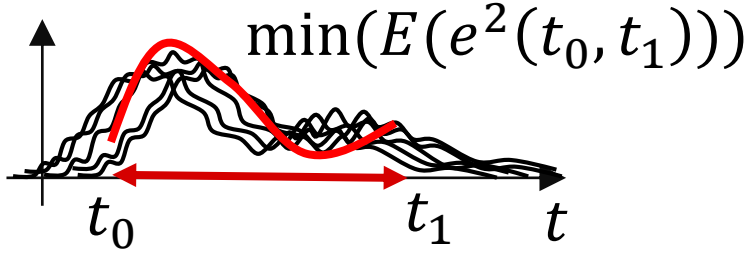
- Determine $\boldsymbol{N} \in \mathbb{N}$ and $\boldsymbol{b_k}, \boldsymbol{a_k} \in \mathbb{R}$ according to specific criteria :
  - Maximize SNR for known signal: Matched filter



  - Least square error with respect to desired signal



  - Minimum (statistical) mean square error (Wiener filter)



Filter Design 3

- Determine $\boldsymbol{N} \in \mathbb{N}$ and $\boldsymbol{b_k}, \boldsymbol{a_k} \in \mathbb{R}$ according to specific criteria :
  - Model analog function e.g.

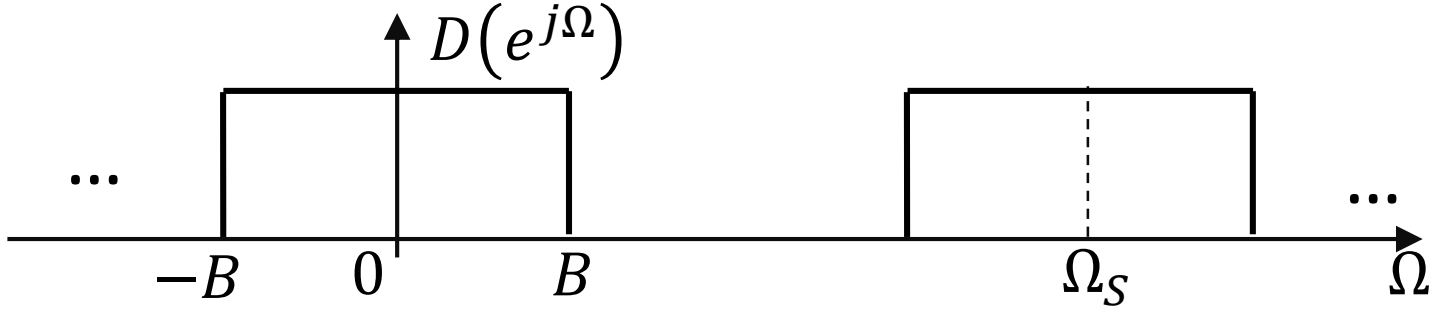
$$y(t) = \int_{-\infty}^{t} x(\tau) d\tau$$

$$y(t) = \frac{dx(t)}{dt}$$

  - …

FIR Filter Design: Fourier Series and Windowing

- Desired frequency response $D(e^{j\Omega})$ e.g. perfect low pass



- Continuous periodic function approximated using Fourier series:

$$D(e^{j\Omega}) = \sum_{n=-\infty}^{\infty} d(n)e^{-j\Omega n}$$

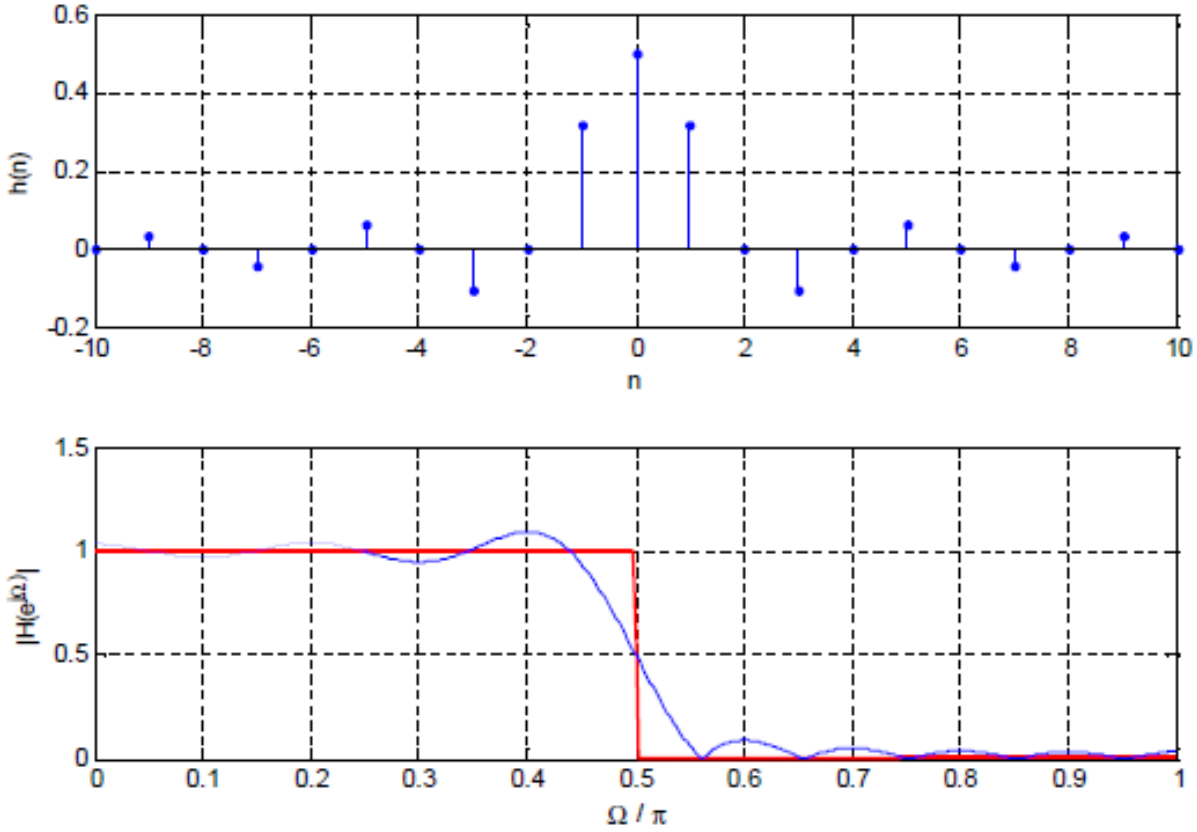
$$d(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} D(e^{j\Omega})e^{j\Omega n}d\Omega$$

FIR Filter Design: Fourier Series and Windowing

- Windowing:

$$h(n) = d(n).w(n) \quad \bullet\!\!-\!\!\bullet \quad H(e^{j\Omega}) = D(e^{j\Omega}) * W(e^{j\Omega})$$

- Example:
  - $f_D = 1kHz$
  - $f_S = 4kHz$
  - $w(n)$ – Recktangle N=2
- MATLAB:
  - `w=window(…);`
  - `b=fir1(N,W_D,'window')`



IIR Filter Design: Analog prototype filter

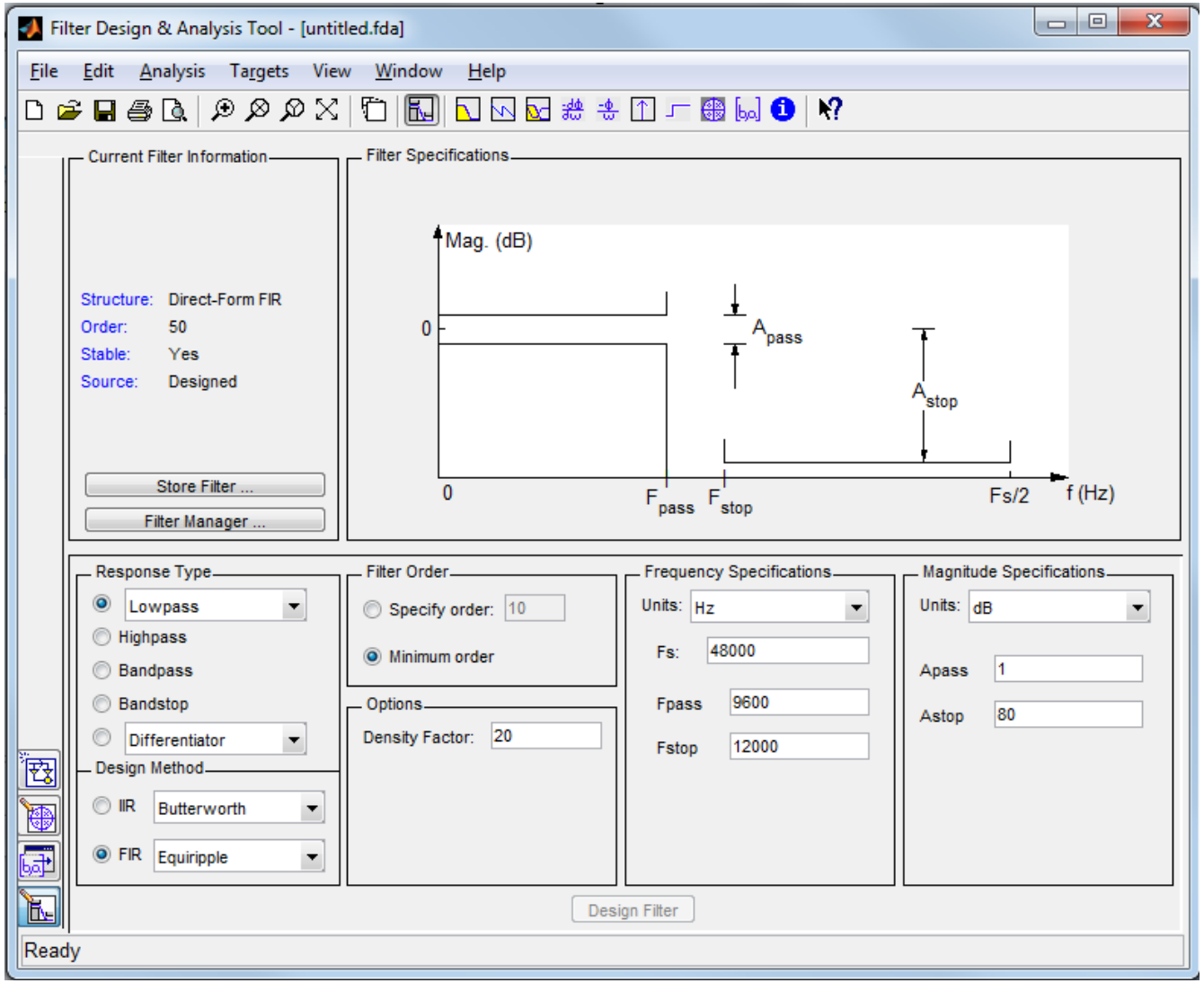
- Exploit theory of analog filter desing and transform analog filter to digital
- Filter types:
  - Butterworth: buttord() → butter()
  - Tschebyscheff I.: cheb1ord() → cheby1()
  - Tschebyscheff II.: cheb2ord() → cheby1()
  - Cauer (elliptic): ellipord() → ellip()

- Analog to digital filter transformation:
  - Impulse invariance method
  - Bilinear transformation
  - …

Stability and causality issues !!!

Filter Design Tools

- MATLAB **fdatool** – filter design and analysis tool IDE
- MATLAB signal processing toolbox:
  - fir1()
  - fir2()
  - firpm()
  - …



TECHNISCHE UNIVERSITÄT DRESDEN

Ditital Filter Implementation

Representation of Signal and Coefficient Values

- Floating point numbers:
  - Mantisa a
  - Exponent b
  - Number value:

$$d = a \text{ x } 2^b$$

- Fixed point number number:
  - Integer part i bits
  - Fraction part f bits

$$W = i + f \quad [bits]$$

  - Number value:

$$d = 2^i . 2^f$$

- Floating vs. fixed point: Higher accuracy, resolution, power and resource consumption

Signed fix-point numbers *W=4*

Binary				Interpreted value	
d_3	d_2	d_1	d_0	$d_3d_2d_1d_0.$	$d_3d_2.d_1d_0$
0	0	0	0	0	0.00
0	0	0	1	1	0.25
0	0	1	0	2	0.50
				…	…
0	1	1	1	7	1.75
1	0	0	0	-8	-2.00
1	0	0	1	-7	-1.75
1	0	1	0	-6	-1.50
				…	…
1	1	1	0	-2	-0.50
1	1	1	1	-1	-0.25

Binary Arithmetic

- **2's complement** of W-bit binary number d

- Simple addition, subtraction and multiplication for signed/unsigned numbers
- Example $\mathrm{W} = 4$ bit subtraction $5 - 2 = 3$:
  - Approach 1: $(0101)_2 - (0010)_2 = (0011)_2$
  - Approach 2: $5 - 2 = 5 + (-2)$
  - 2's complement (-2): $2^4 - |-2| = 14 = (1110)_2$

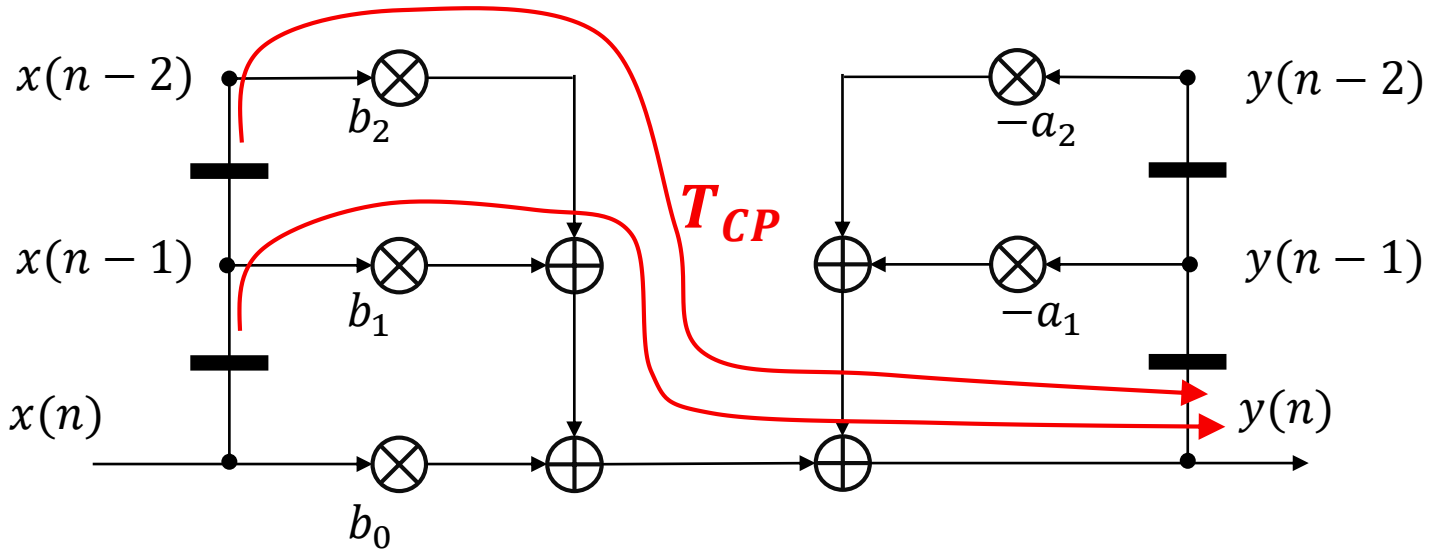
$$(0101)_2 + (1110)_2 = (\cancel{1}0011)_2$$

Signal Flow Graph & Critical Path

- Critical path (CP) – maximum propagation delay of combinatorial logic T_{CP}
- Consequence for maximum achievable clock frequency of digital circuit:

$$f_{clk} \leq \frac{1}{T_{CP}}$$

- Example 2nd order system *N=2*:



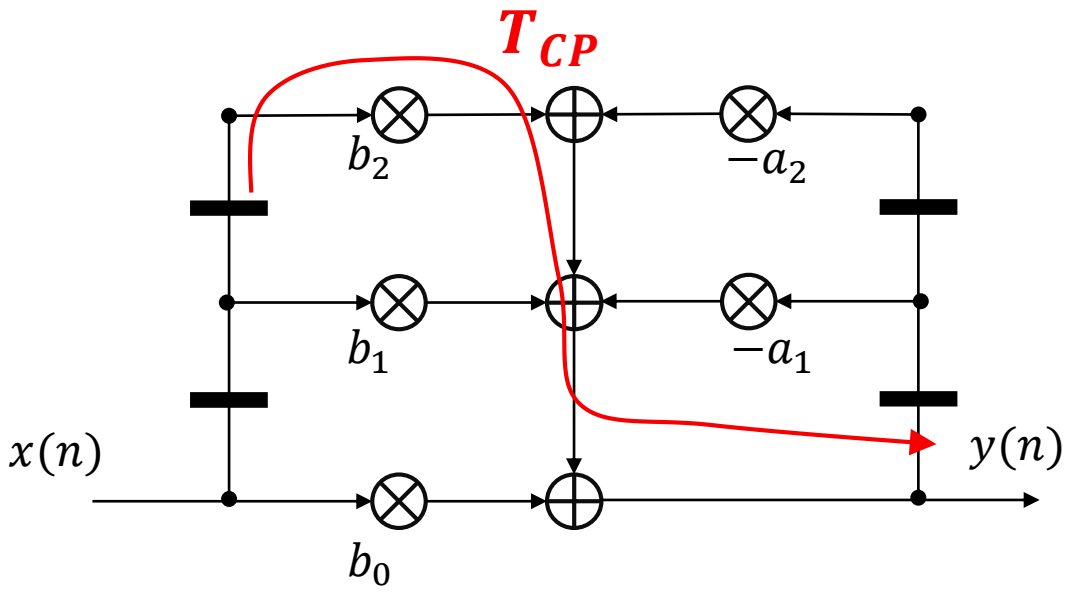
$$T_{CP} = T_{MUL} + (N+1)T_{ADD}$$

$T_{CP} \sim NT_{ADD}$!!!

Proportional to system order N

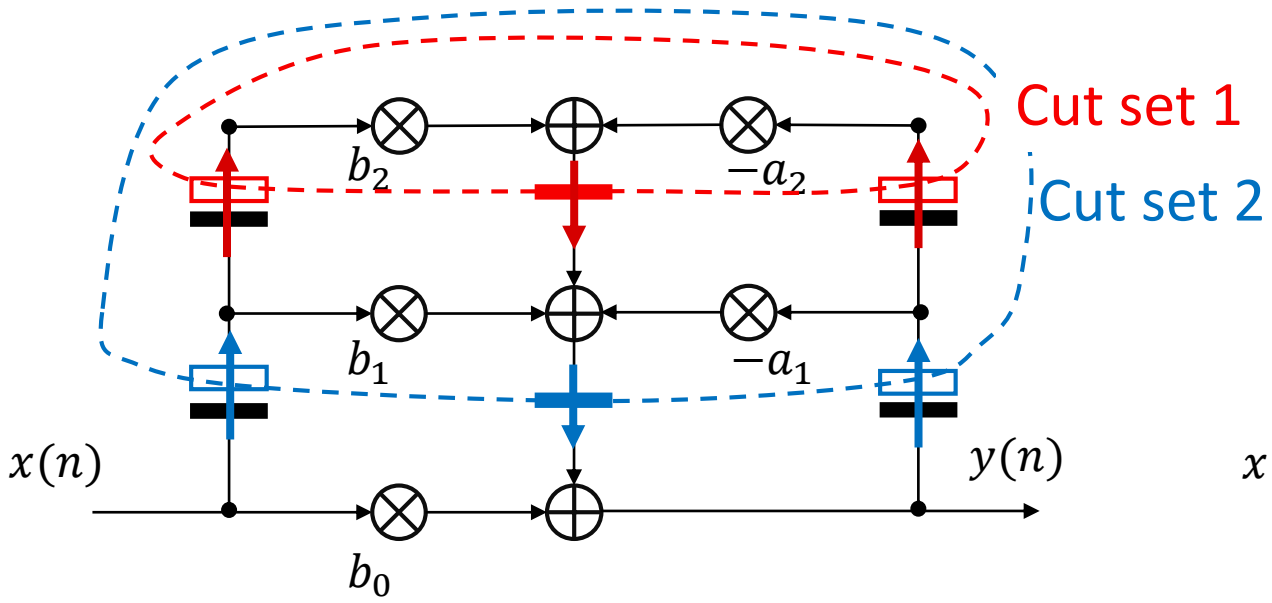
Pipelining Signal Flow Graph: Cut Sets

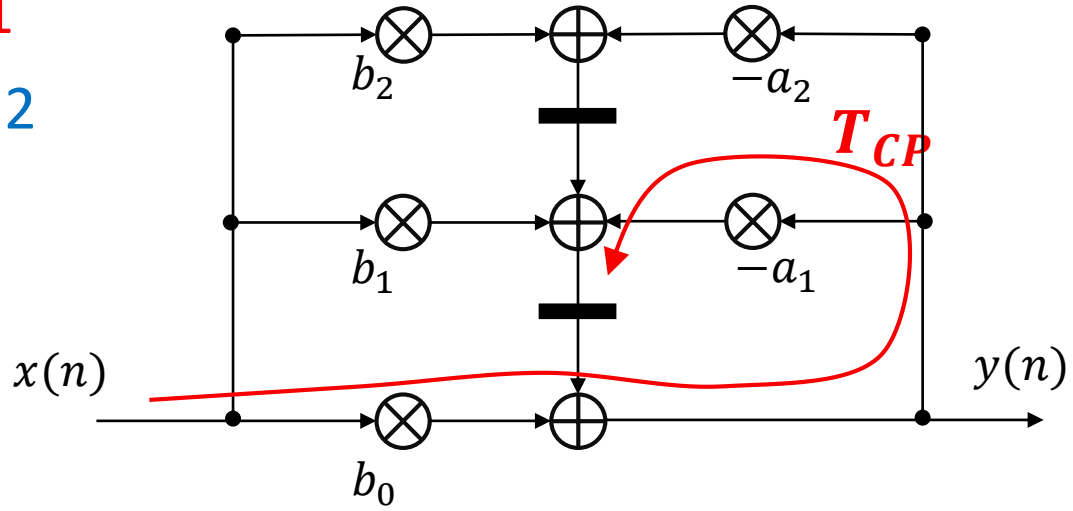
- **Pipelining –** shorten CP by inserting and/or relocating registers in signal flow graph
- **Cut set method -** pipelining approach preserving system function:
  - Cut graph edges using closed loop
  - Add **delay** $\boldsymbol{z^{-1}}$ and **speedup** $\boldsymbol{z^{+1}}$ ($z^{-1}.z^{+1} = 1$) elements at cut positions according to edge orientation



Pipelining Signal Flow Graph: Cut Sets

- **Pipelining –** shorten CP by inserting and/or relocating registers in signal flow graph
- **Cut set method -** pipelining approach preserving system function:
  - Cut graph edges using closed loop
  - Add **delay** $\boldsymbol{z^{-1}}$ and **speedup** $\boldsymbol{z^{+1}}$ ($z^{-1}.z^{+1} = 1$) elements at cut positions according to edge orientation





$$T_{CP} = 2(T_{MUL} + T_{ADD})$$ Not a function of order *N*

TECHNISCHE UNIVERSITÄT DRESDEN

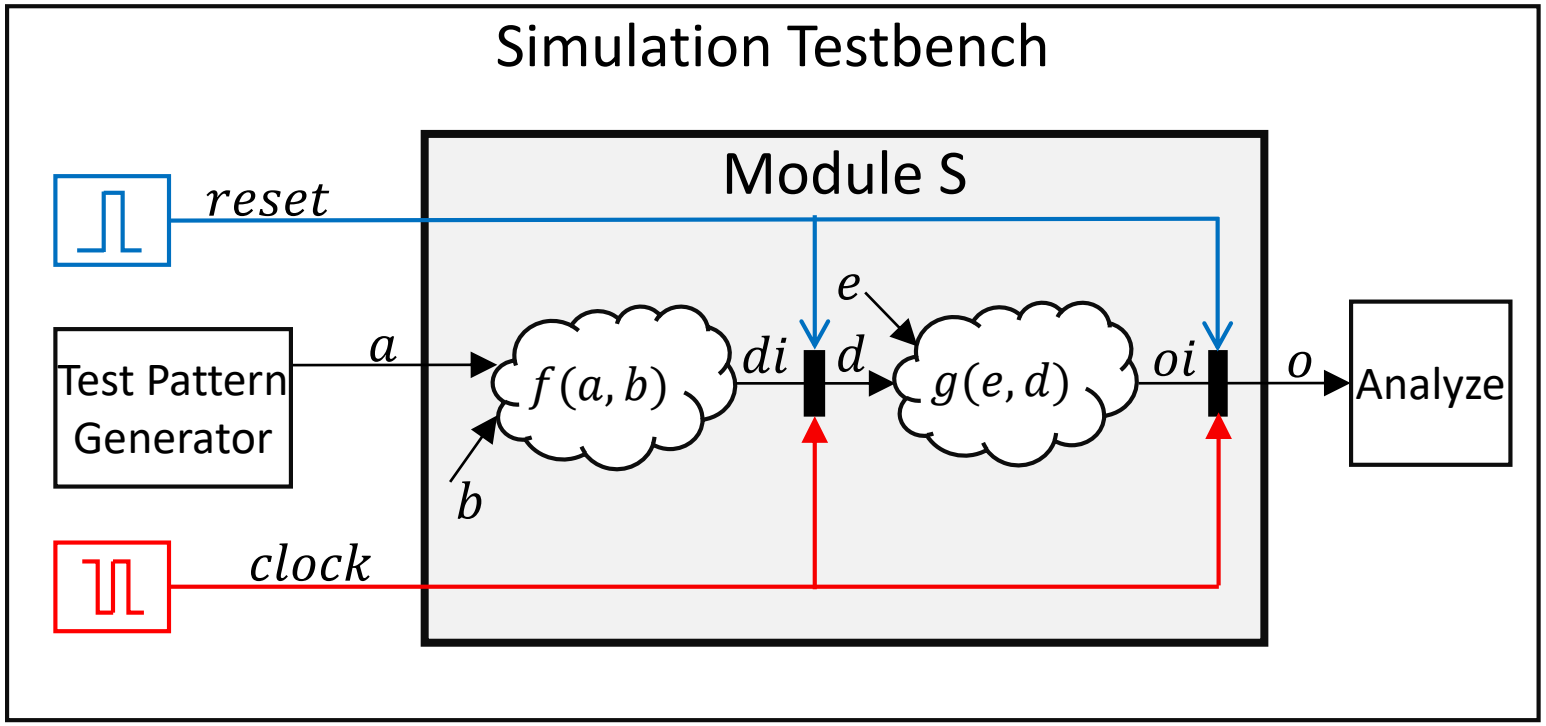
Digital Filter HW Design

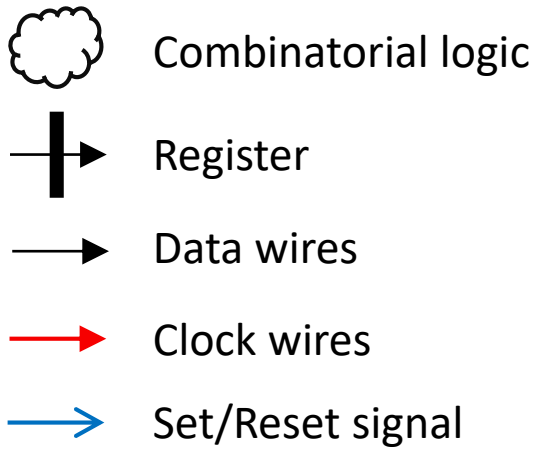
FPGA Introduction

- Field Programmable Gate Array – inherently parallel processing architecture
- Software defined devices: (re-)programmed by firmware to achieve desired function
- 1980: Invented
- 1985: Xilinx first commercial FPGA XC2064 – Logic + RAM
- 2019: FPGA scalable computing platform
  - 500MHz
  - **Logic blocks + large RAM**
  - **DSP blocks**
  - CPUs
  - High speed I/O + Interfaces (memory, **ADC,DAC,** USB, Ethernet, PCIe)
  - **PLLs** + DLLs
  - GPU + VPU

RTL Design

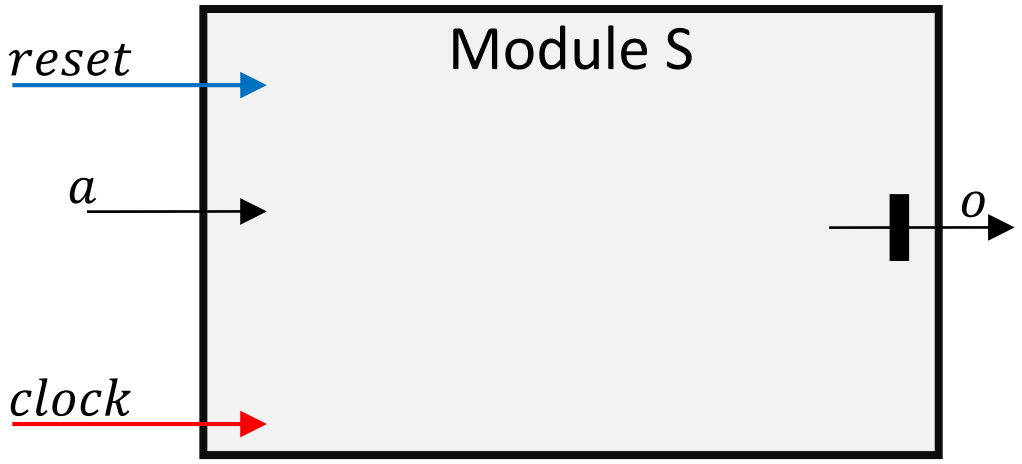
- Register-Transfer-Level design





RTL Design using Verilog

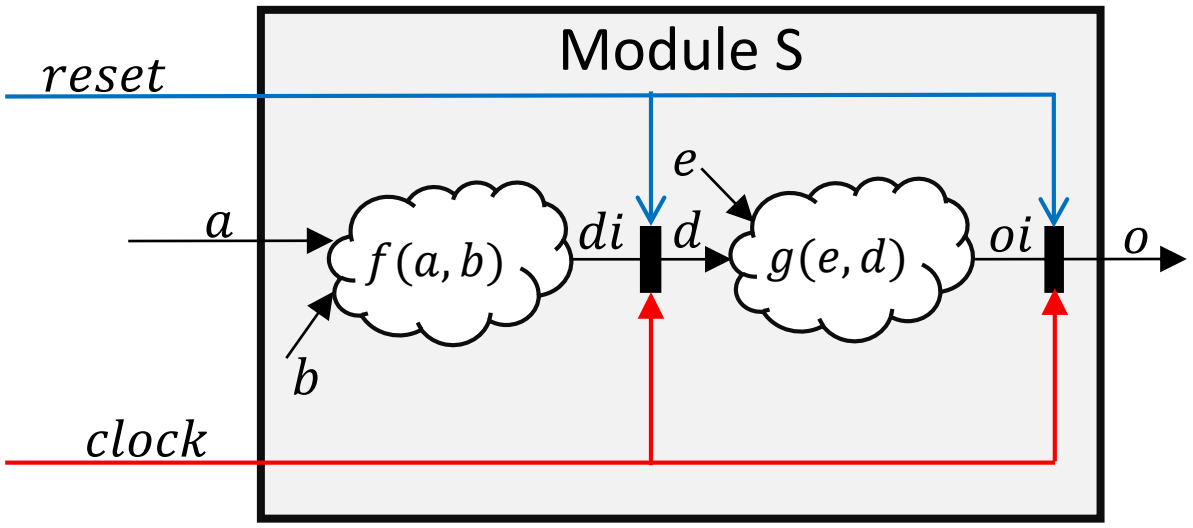
- Define module interface:




```
module S (
        input clock,
        input reset,
        input signed [15:0] a,
        output reg signed [15:0] o
);

…
endmodule
```

RTL Design using Verilog

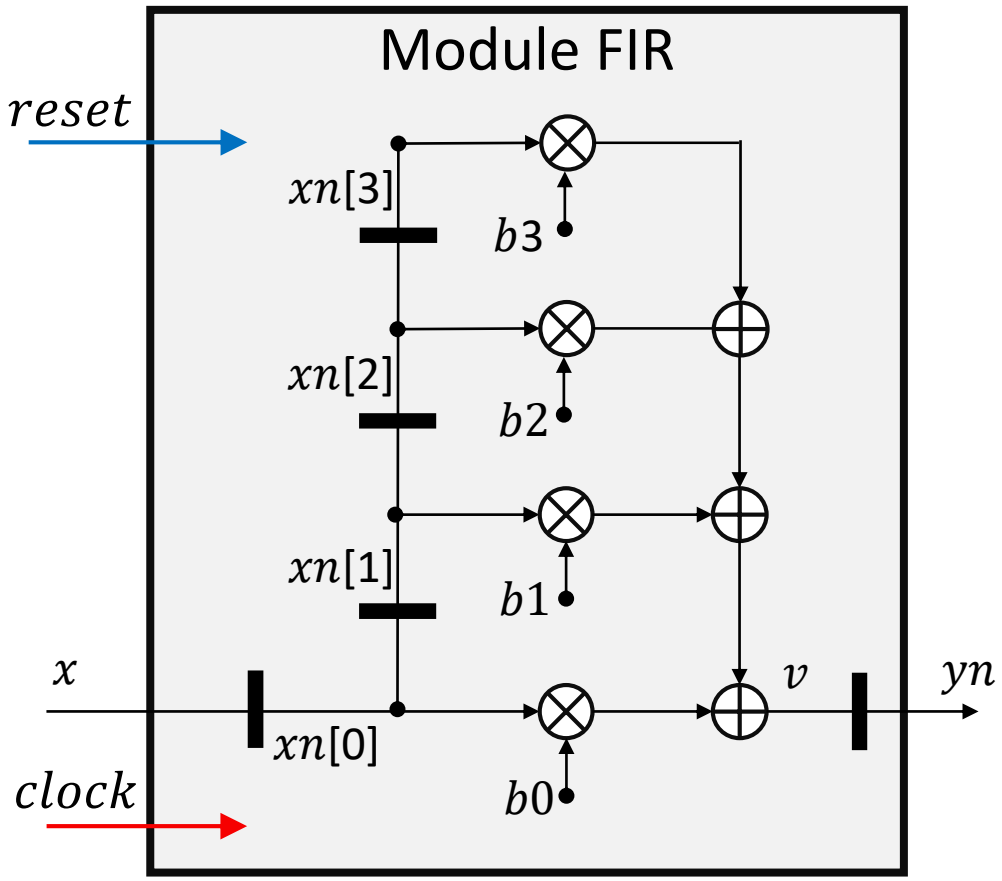
- Define: Registers, wires, combinatorial functions,




```
…
reg signed [15:0] b,d,e;
wire signed [15:0] di,oi;

assign di=f(a,b);
assign oi=g(e,d);

always @(posedge clock)
begin
        d<=di;
        o<=oi;
end

…
```

FIR Filter Design using HDL Verilog

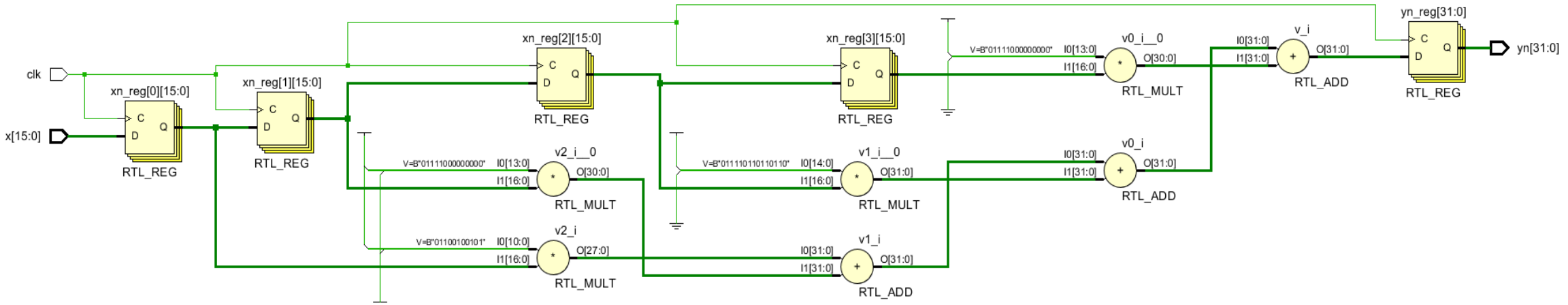
- Define module interface:




```
module FIR (input clock, input reset,
            input signed [15:0] x,
            output reg signed [15:0] yn
);
reg  signed [15:0] xn [2:0];
wire signed [31:0] v;

wire signed [15:0] b0 = 16'h0325;
wire signed [15:0] b1 = 16'h1e00;
wire signed [15:0] b2 = 16'h3db6;
wire signed [15:0] b2 = 16'h0352;

assign yi=b0*xn[0]+b1*xn[1]+b2*xn[2]+b3*xn[3];

always @(posedge clock)
begin
        xn[0]<=x;     xn[1]<=xn[0];
        xn[2]<=xn[1]; xn[3]<=xn[2];
        y<=v;
end
endmodule
```

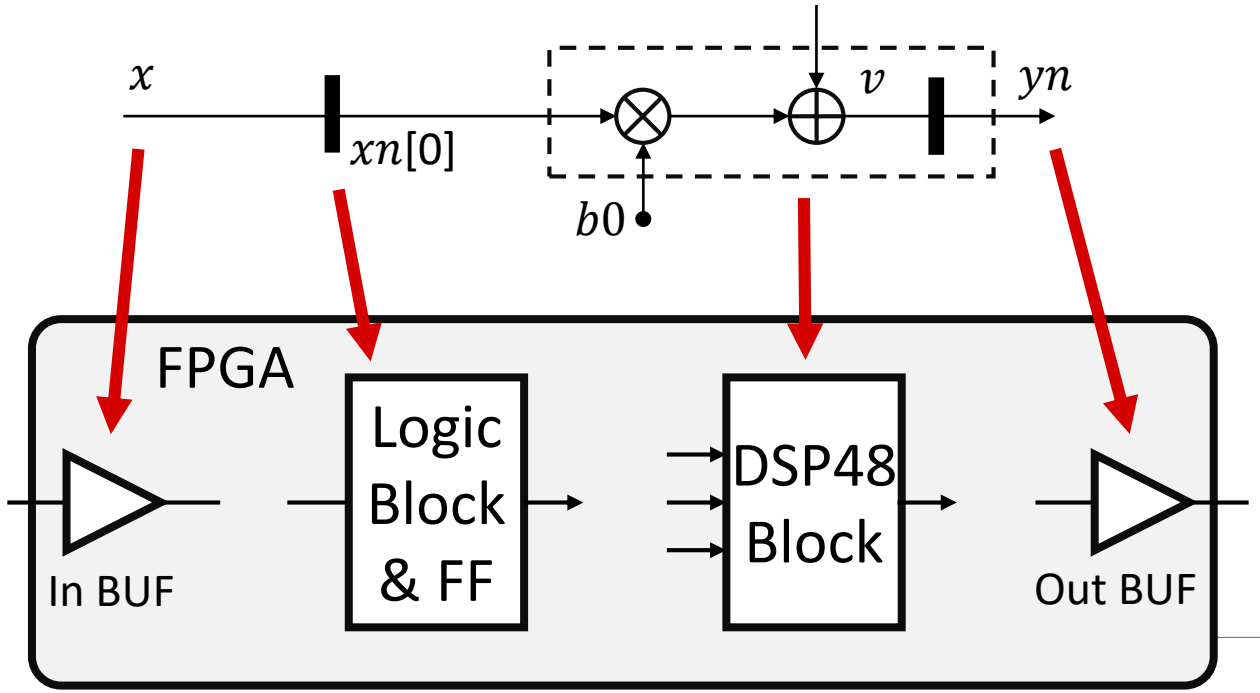
HDL Synthesis 1

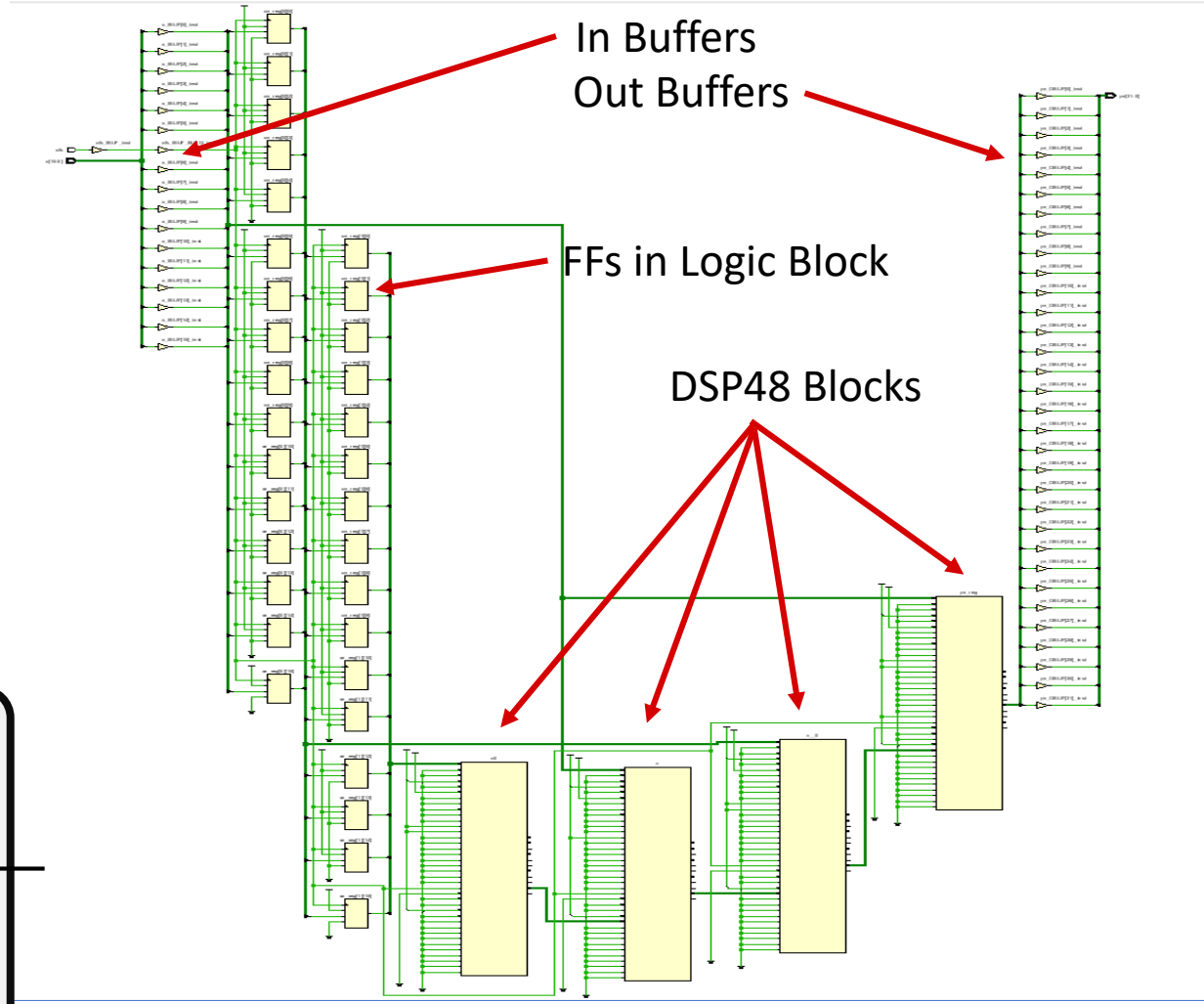
- Example: 4-tap direct form FIR filter
  - Device: Xilinx Artix-7
  - Synthesis: Xilinx Vivado
- RTL elaboration of Verilog source code to produce platform independent schematics



HDL Synthesis 2

- Example: 4-tap direct form FIR filter
- **Synthesis:** mapping logic functions to FPGA specific resources by meeting design constraints





FPGA Design Methodology

- Register-transfer-level (RTL): Hardware Description Language (HDL): Verilog, VHDL
- IP-core integration based design: pre-defined/customizable reusable units of logic
  - Vivado design suite: Xilinx logiCORE IP FIR Compiler
- High-level synthesis (HLS): C-language synthesis tool
- Model-based: High-level abstraction using Simulink/LabVIEW… blocks

Thank You!

Linear Time-Invariant (LTI) System

- LTI System representation by N^{th} order linear difference equation:

$$y(n) = \sum_{k=0}^{N-1} b_k x(n-k) - \sum_{k=1}^{N-1} a_k y(n-k)$$

$$y(n) = b_0 x(n) + \sum_{k=1}^{N-1} [b_k x(n-k) - a_k y(n-k)]$$

Output

Input

Internal state

DSP Design

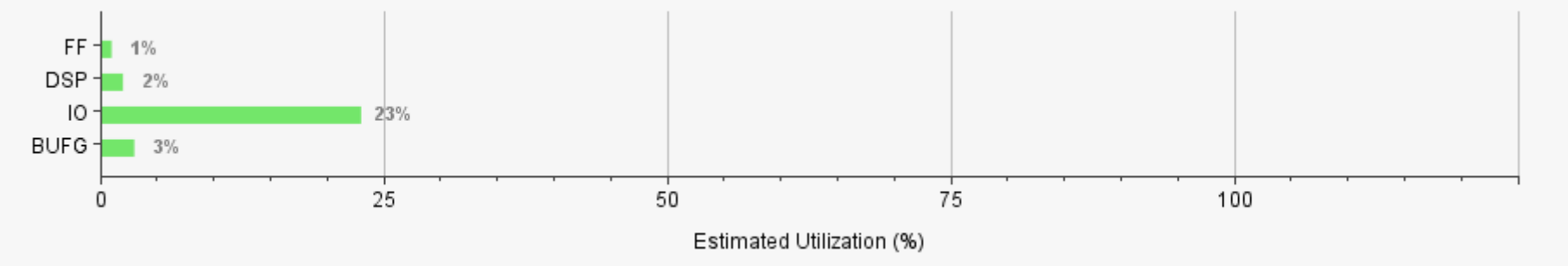
- Digital vs. Analog: more flexibility, lower power consumption, higher reliability, higher accuracy and much better scalability

Synthesis: 4-tap direct form FIR w/ DSP block

Product family:	Artix-7
Project part:	xc7a100tlcsg324-2L

Name	Slice Registers (126800)	DSPs (240)	Bonded IOB (210)	BUFGCTRL (32)
direct_fir_filter_4tap	32	4	49	1

Resource	Estimation	Available	Utilization %
FF	32	126800	0.03
DSP	4	240	1.67
IO	49	210	23.33
BUFG	1	32	3.13



Synthesis: 4-tap direct form FIR w/o DSP block

Name	Slice LUTs (63400)	Slice Registers (126800)	Bonded IOB (210)	BUFGCTRL (32)
direct_fir_filter_without_dsp_block	445	96	49	1

Resource	Estimation	Available	Utilization %
LUT	445	63400	0.70
FF	96	126800	0.08
IO	49	210	23.33
BUFG	1	32	3.13