RTR532 2018 Spring
Lecture 05

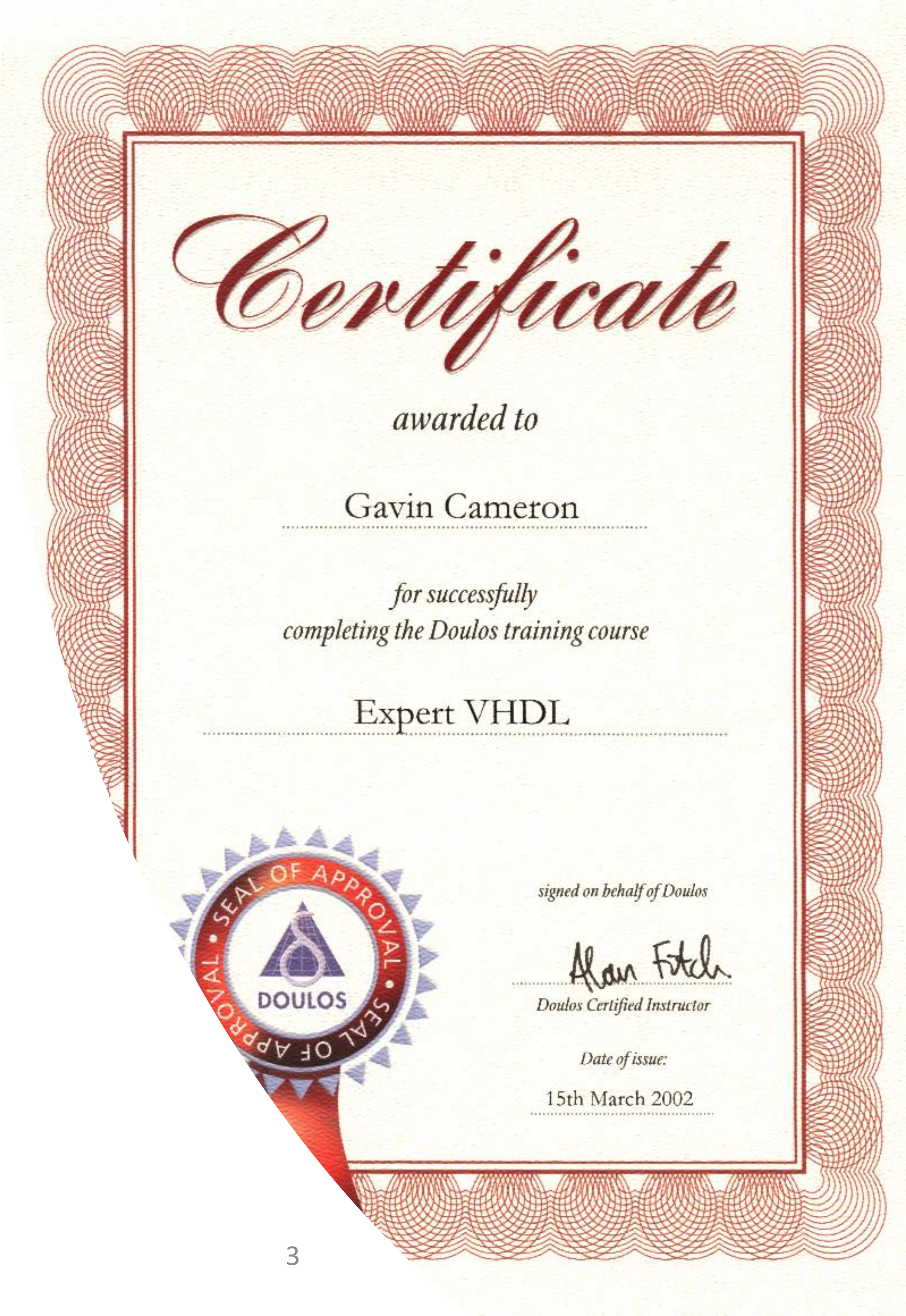
- **Digital filters with VHDL II (continued)**

Administrative stuff

- PS01 – PS04 results ETA May 6th
- Final project templates ETA May 6th
- Final project templates to be filled in by May 10th

Meanwhile

- Kriss & Gatis attended course «Expert VHDL Design» by Doulos
- https://www.doulos.com/content/training/vhdl_expert.php



Certificate

awarded to

Gavin Cameron

for successfully completing the Doulos training course

Expert VHDL

SEAL OF APPROVAL • SEAL OF APPROVAL •

DOULOS

signed on behalf of Doulos

Alan Fitch

Doulos Certified Instructor

Date of issue:

15th March 2002

Filter defined by transfer function

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_N z^{-N}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + A_M z^{-M}}$$

- Question remains - how to implement it in hardware ?

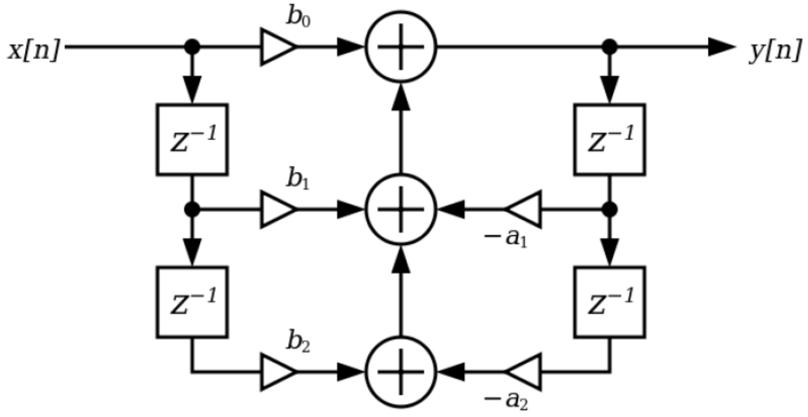
How do you start designing a filter?

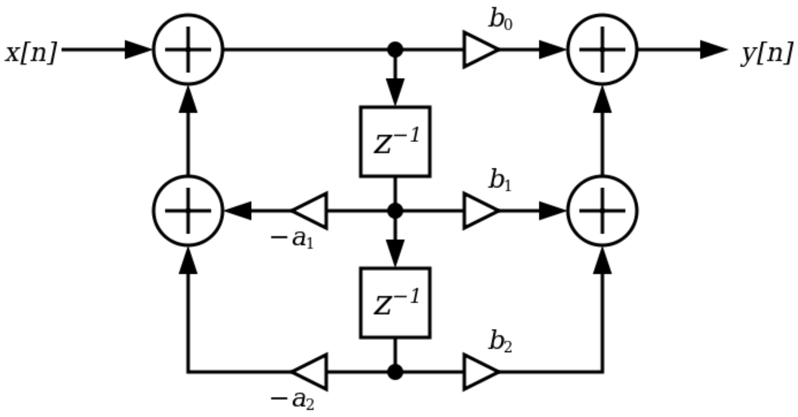
- Choose type – bandpass, low pass, high pass
- Cutoff frequency, phase response linearity
- Frequencies – normalized
- Coefficients – using filter design tools (Matlab – filter design)

Digital filter forms – comparison (figures from wikipedia)

Direct Form I

Direct Form II





Digital filter forms - comparison

	Direct Form I	Direct Form II
Multiplication operations	5	5
Accumulation operations	3	4
Delay taps	4	2

FPGA Available resources

	Low end FPGA Spartan6 LX9	Middle range FPGA KU3P	High end FPGA VU13P
Flip Flops	11’440	325’000	3’456’000
Multiplier and accumulator-s	16	1’368	12’288 (called DPS slices)
Transceiver bandwidth	0	32.75 * 16 = 524 Gb/s	32.75 * 128 = 4192 Gb/s
Price ($)	20	1500	15’000++

Case study – Spartan6 LX9

- Delay tap flip flops = bit width, e.g. 18

	Direct Form I	Direct Form II
Multiplications	5	5
Accumulations	3	4
Delay taps	4 (4*18=72 flip flops)	2 (2*18=36 flip flops)

	Low end FPGA Spartan6 LX9
Flip Flops	11’440
Multiplier and accumulators	**16**
Transceiver bandwidth	0
Price ($)	20

- We can fit in Spartan-6 LX9:
  - 2nd order IIR filter 5/16 multipliers, 2*2*18 = 72 flip flops
  - 3rd order IIR filter 7/16 multipliers, 3*2*18 = 108 flip flops
  - ..up to 7th order IIR filter - 15/16 multipliers, 7*2*18 = 252 flip flops
  - 16th order FIR filter
- Problem – limited DSP slices (multiplier elements)

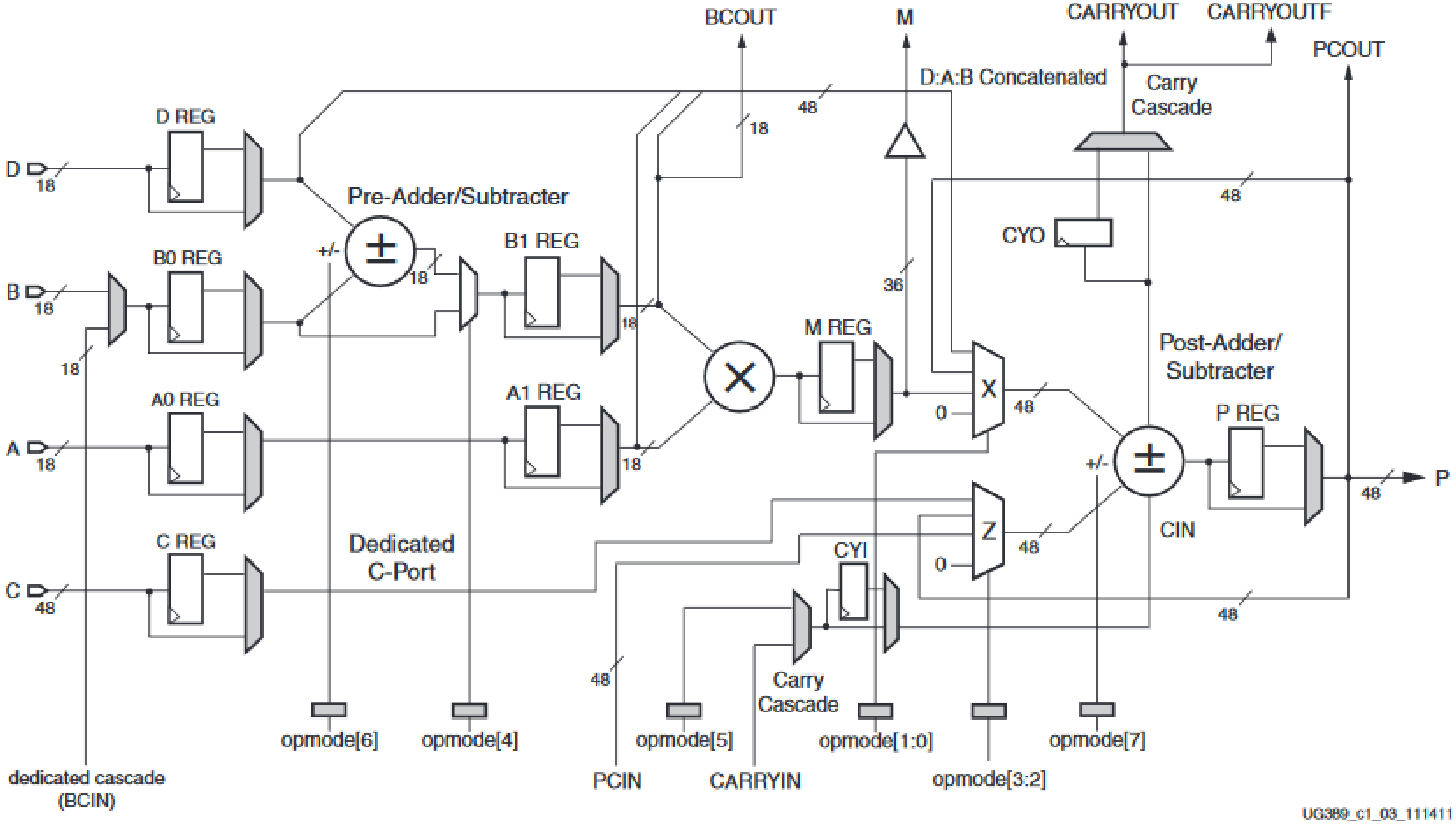
How to find data about FPGA Resources?

Several ways:

1. You know the IC – search by part number, check datasheet
2. You design the board – choose FPGA suitable for application

Multipliers are also called DSP slices, DSP blocks, etc... Pay attention!

FPGA Multiplier (Spartan-6)



Filter design approaches

- Standard / simple:
  - Pipelined filter architecture
- Advanced filter design architectures:
  - Multi-Cycle filters
  - Shared multiplier

Pipelined filter architecture

- One clock – sample clock

- Draw pipelined filter architecture for 2nd order FIR filter.

Multi-cycle filters

- Two clocks:
  - Sample clock (slower)
  - Multiplication and accumulation clock (faster)

- Draw pipelined filter architecture for 2nd order FIR filter (one multiplier).

Shared multipliers

- Hardware resource used in multiple filters
- Filters’ outputs phased

- Draw shared multiplier filter architecture for two 2nd order FIR filters (one multiplier)

Thoughts on filter type (FIR / IIR) and order

- Required cutoff slope depends on filter order
- High order IIR filters are sensitive to quantization errors (fixed point rounding, truncation)
- Possible solution – cascade of «simple filter»

Bi-Quad (2nd order) filters

- 2nd order FIR part
- 2nd order IIR part
- Cascade as required

$$H(z) = \prod_{i=1}^{N} \left(\frac{b_{i0} + b_{i1} Z^{-1*} + b}{1 + a_{i0} Z^{-1*} + a_{i1} Z^{-2}} \right)$$

VHDL BiQuad IIR implementation example

Direct Form I – BiQuad

- Low pass, fl = 500Hz, Fs = 48000Hz, PBR = .08 dB, SBR = .03 dB
- Input 18b signed
- Coefficients – 32b (form 2 bit integer , 30 bits fraction)
- Output – 18b signed
- Coefficients:
  - B0=0.0010232
  - B1=0.0020464
  - B2=0.0010232
  - A1=-1.9075016
  - A2=0.9115945
- Based on: https://eewiki.net/display/LOGIC/IIR+Filter+Design+in+VHDL+Targeted+for+18-Bit%2C+48+KHz+Audio+Signal+Use – make sure to check it out later, download the code.

Reading more

- https://se.mathworks.com/help/signal/ug/iir-filter-design.html
- https://eewiki.net/display/LOGIC/IIR+Filter+Design+in+VHDL+Targeted+for+18-Bit%2C+48+KHz+Audio+Signal+Use
- https://www.xilinx.com/support/documentation/white_papers/wp330.pdf