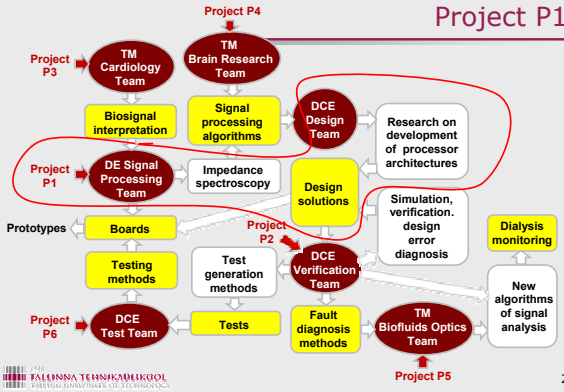
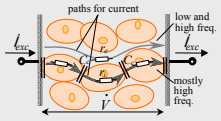
Reconfigurable High Speed Data Pre-Processing Unit for Bioimpedance Measurements

Peeter Ellervee[1]
Vadim Pesonen[1], Maksim Gorev[1], Paul Annus[2], Mart Min[2],
[1] Department of Computer Engineering, TUT
[2] Department of Electronics, TUT

Project P1



Bioimpedance



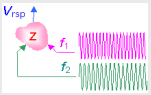
a) Essentials of the Electrical Bioimpedance EBI

Z = R + jX

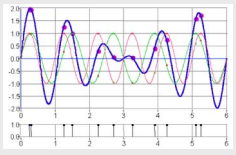
b) The 3-element equivalent of the static EBI

- ✓ Where is bioimpedance useful
  - devices that measure cardiac output and circulating blood volume
  - designing systems to monitor patients during surgery or in intensive care
  - investigating the biomechanics of injury and wound healing

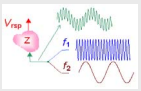
Two typical cases of measurement



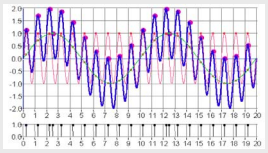
multisite



a) Impedance is measured from two points at a slightly differing frequency

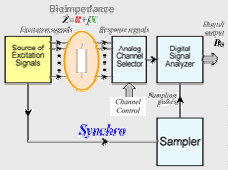


multifrequency



b) The same impedance is measured at (two) essentially different frequencies

EBI measurement using synchronous sampling



The direct current component DC can be determined as
DC = (Re⁺ + Re⁻) / 2
or
DC = (Im⁺ + Im⁻) / 2

The real Re and imaginary Im parts of the phasor $\dot{Z}$ are determined as
Re = (Re⁺ – Re⁻) / 2
and
Im = (Im⁺ – Im⁻) / 2

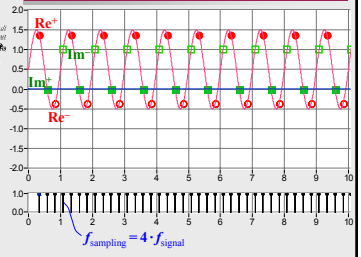


Fig. 2. Synchronous sampling of a single sine wave response. Real part samples Re⁺ are designated as filled red dots ● and Re⁻ as unfilled red ones ○, imaginary part samples Im⁺ as filled green squares ■, and Im⁻ as unfilled green squares □

P. Annus, et al, BEC 2006

Motivation

- ✓ Existing implementation

The FPGA:
- selects channels,
- generates sampling pulses,
- controls gain,
- generates excitation,
- reads samples from digitizers (ADCs),
- performs math functions, and
- provides communication with the outside world.

- ✓ *What can be improved?*
  - ✓ Channel sampling order
  - ✓ On-chip data processing
  - ✓ *Excitation signal generator*
  - ✓ *Analog part (ADC)*
  - ✓ ...

P. Annus, et al, BEC 2006

Digitizing and digital processing of response signal

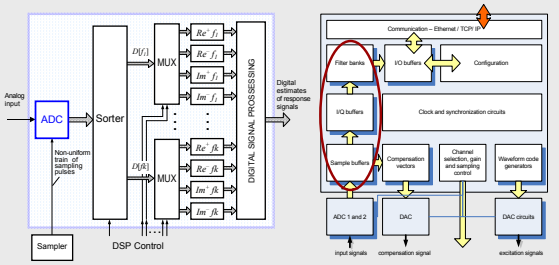
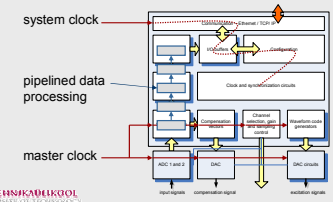


Fig.7. a) sampling and processing of the signal, b) FPGA based DSP unit, block diagram with I/O connections and peripheral components

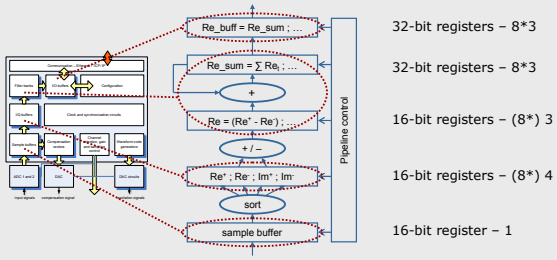
P. Annus, et al, BEC 2006

Architecture defined by data acquisition

- ✓ High sampling frequency (80 or even 160 MHz)
  - External master clock (stability is very important!)
  - Pipelined data processing
- ✓ PC interface
  - Asynchronous communication inside FPGA



Architecture defined by data acquisition



32-bit registers – 8*3

32-bit registers – 8*3

16-bit registers – (8*) 3

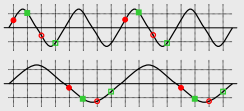
16-bit registers – (8*) 4

16-bit register – 1

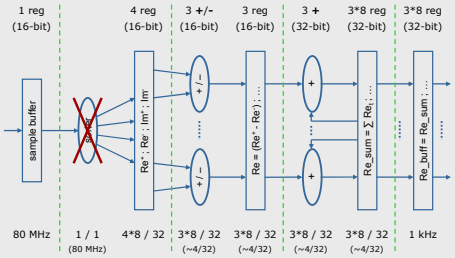
Optimizing the dataflow

- ✓ 8 channels & 4 samples per channel
  - Channel after channel – 4 input (buffer) registers
    - 4 samples in row are processed immediately
    - 4 clock-steps to make 3 calculations
    - 3 registers to store intermediate results

  + no need to sort input data
  – difficulties at some excitation frequencies



Channel after channel

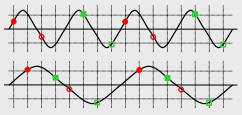


8 16-bit registers (instead of 57)

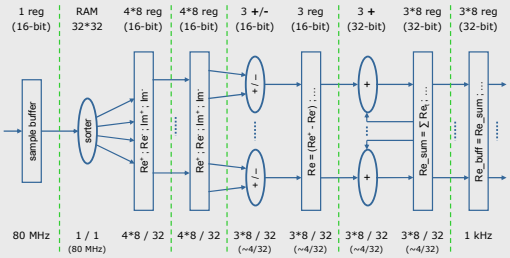
Optimizing the dataflow

- ✓ 8 channels & 4 samples per channel
  - random order – <u>All</u> samples must be stored
    - Extra pipeline stage needed
    - But the rest is the same…

  + any sampling order can be used
  – sorter and extra buffers needed
  – fast analogue channel selectors needed

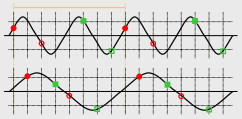


Random sampling order

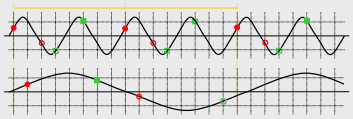


68 16-bit registers (instead of 57)

Is it good enough?



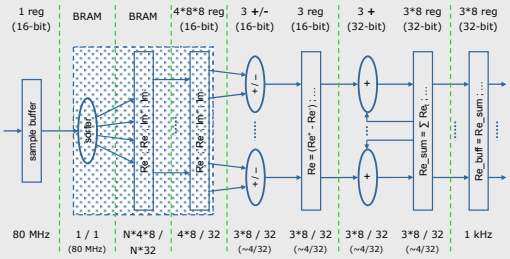
- ✓ 8 channels & 4 samples
  - Some frequencies may still cause the aliasing effect
  - Random access is more stressful for analog mux



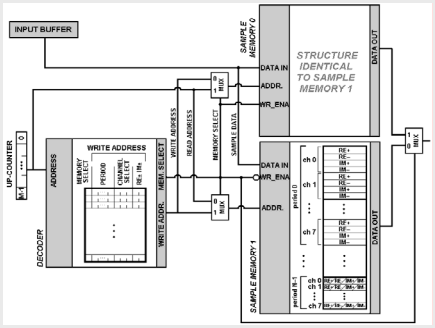
- ✓ Wider window?
- ✓ How wide?
- ✓ Resources?
- ✓ Sampling order?

Note: Two channels are assumed for simplicity. Sample order is illustrative.

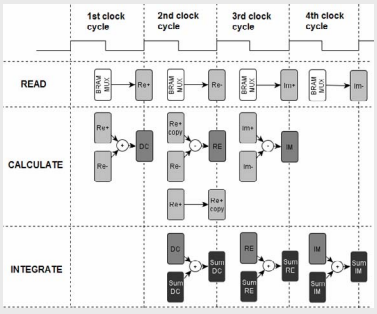
Reprogrammable sampling order



Collecting & sorting



Pipelined data-preprocessing



Results

- ✓ Spartan3 xc3s1000-4-ft256
  - Channel after channel, optimized dataflow
    - 3 16-bit & 3 32-bit add-subs / 8 16-bit & 48 32-bit reg-s
    - 1499 slices (19%); speed – 105 MHz ☺
  - Random sampling order, optimized dataflow
    - 3 16-bit & 3 32-bit add-subs / 68 16-bit & 48 32-bit reg-s
    - 2124 slices (27%); speed – 98 MHz 😐
  - Random sample order, 8-cycle window
    - 3 16-bit & 32 32-bit add-subs / 29 16-bit & 48 32-bit reg-s
    - 3 BRAMs, 1204 slices (15%); speed – 116 MHz ☺
  - Random sample order, 32-cycle window
    - 3 BRAMs, 1206 slices (15%); speed – 116 MHz ☺
  - Random sample order, 64-cycle window
    - 6 BRAMs, 1207 slices (15%); speed – 116 MHz ☺

Future work

- ✓ Utilization of the reprogrammability
  - How to find useful sampling orders?
  - How wide widows are needed?
- ✓ Tighter control of the synthesis process
  - accumulators are multi-cycle operations, etc.
- ✓ Higher sampling speed?
  - Virtex5 xc5vlx110t-1-ff1136: ~220 MHz
  - Two input ADC-s?

- ✓ Thank you for your attention