OC Workshop:
Architectures and Tools

DFG SPP 1183 – 10th Colloquium, February 22 in Hannover

Stephan Sigg
TU Braunschweig
Institute of Operating Systems
and Computer Networks
www.ibr.cs.tu-bs.de/dus

Emergent radio: Project focus

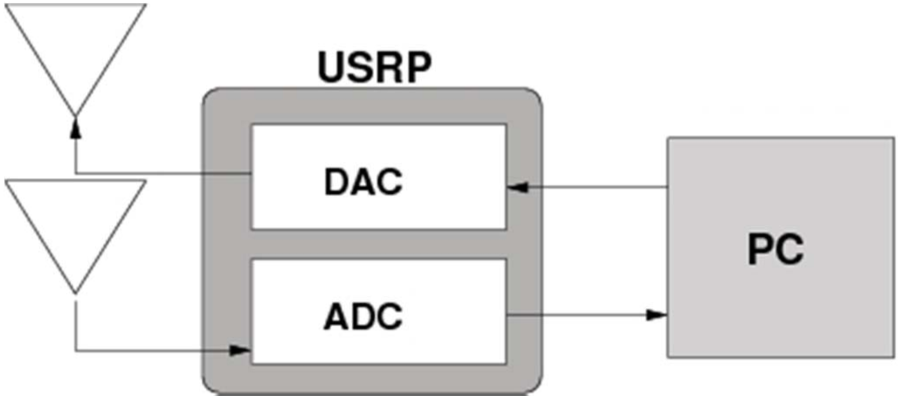
Project focus

- Adaptive, autonomous, self-improving distributed adaptive beamforming in wireless networks
- Adapt parameters to environmental changes
  - E.g. noise, synchronisation speed
- Develop environment-adaptive optimisation scheme

USRP software radios

The Universal Software Radio Peripheral (USRP)

- Communication interface controlled by standard PC
- USB connected



- Hardware:
  - Altera Cyclone FPGA
  - Four A/D converter (12 Bits/sample), sample rate 64 MegaSamples/sec
  - Daughterboads for various transmit/receive frequencies

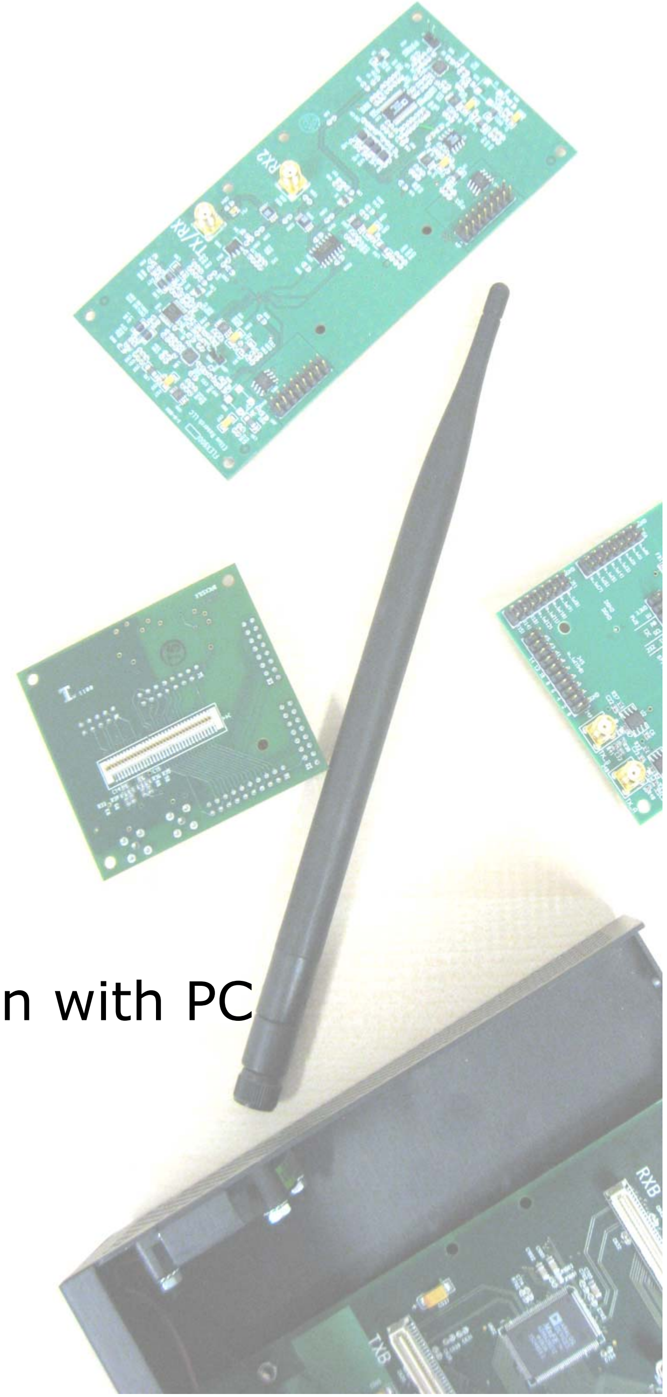
USRP software radios

Opportunities:

- Various frequencies
- Modulation/coding arbitrary
- Computational power of standard PC
- MIMO-capable

Limitations:

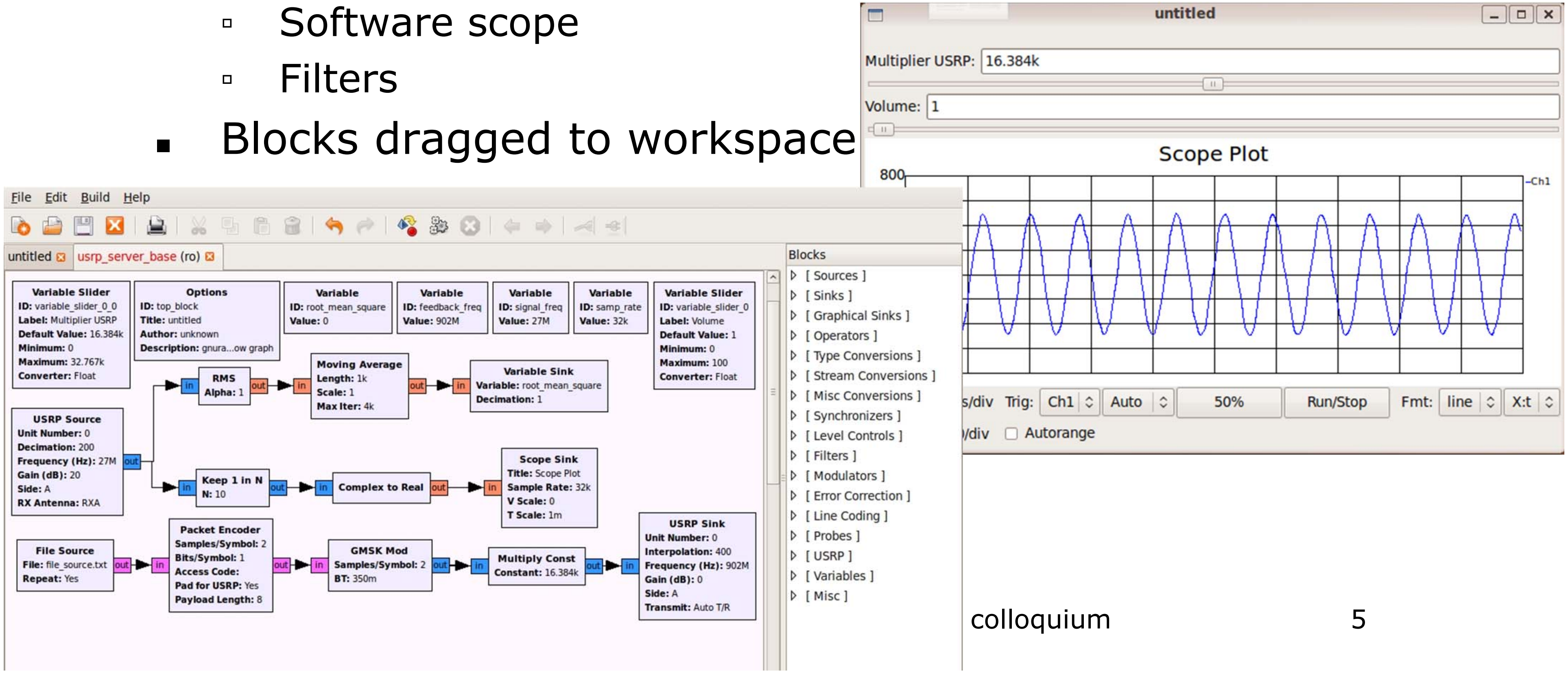
- Slow processing due to communication with PC
  - FPGA-programming possible



GNU Radio Companion

Workbench to create signal processing flow graphs

- Various preconfigured blocks available
  - Signal sources/sinks
  - Modulation schemes
  - Software scope
  - Filters
- Blocks dragged to workspace



untitled

Multiplier USRP: 16.384k

Volume: 1

Scope Plot

Trig: Ch1 Auto 50% Run/Stop Fmt: line X:t

☐ Autorange

untitled ☒ usrp_server_base (ro) ☒

Variable Slider
ID: variable_slider_0_0
Label: Multiplier USRP
Default Value: 16.384k
Minimum: 0
Maximum: 32.767k
Converter: Float

Options
ID: top_block
Title: untitled
Author: unknown
Description: gnura...ow graph

Variable
ID: root_mean_square
Value: 0

Variable
ID: feedback_freq
Value: 902M

Variable
ID: signal_freq
Value: 27M

Variable
ID: samp_rate
Value: 32k

Variable Slider
ID: variable_slider_0
Label: Volume
Default Value: 1
Minimum: 0
Maximum: 100
Converter: Float

RMS
Alpha: 1

Moving Average
Length: 1k
Scale: 1
Max Iter: 4k

Variable Sink
Variable: root_mean_square
Decimation: 1

USRP Source
Unit Number: 0
Decimation: 200
Frequency (Hz): 27M
Gain (dB): 20
Side: A
RX Antenna: RXA

Keep 1 in N
N: 10

Complex to Real

Scope Sink
Title: Scope Plot
Sample Rate: 32k
V Scale: 0
T Scale: 1m

File Source
File: file_source.txt
Repeat: Yes

Packet Encoder
Samples/Symbol: 2
Bits/Symbol: 1
Access Code:
Pad for USRP: Yes
Payload Length: 8

GMSK Mod
Samples/Symbol: 2
BT: 350m

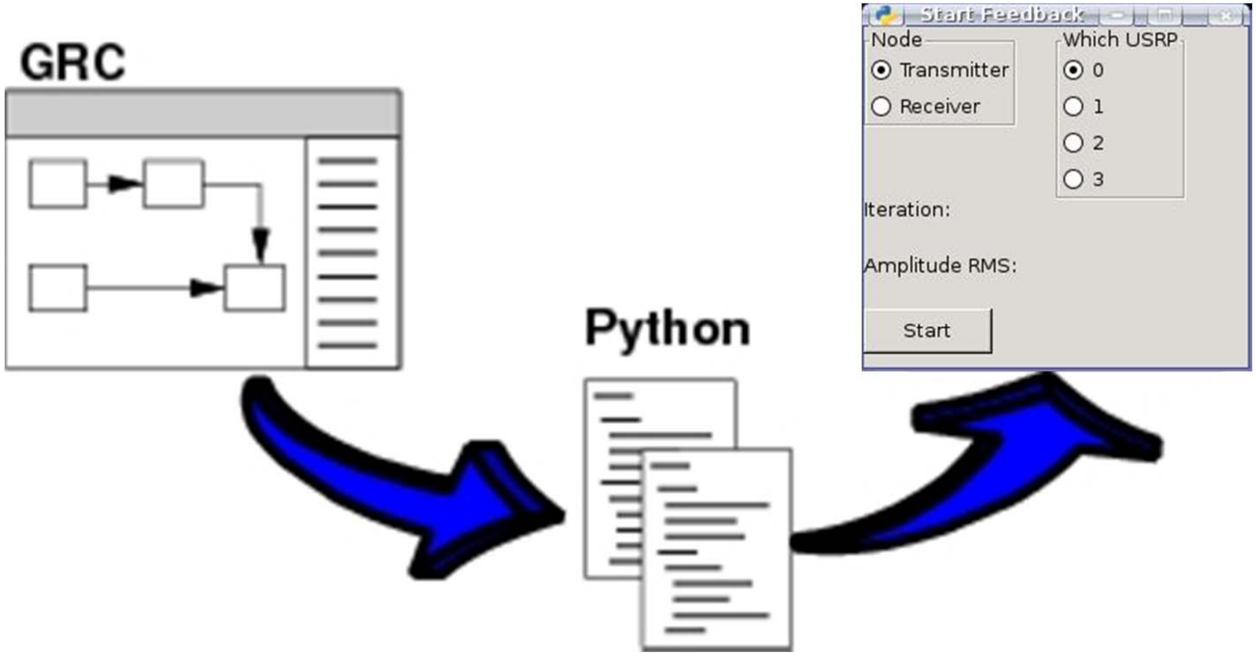
Multiply Const
Constant: 16.384k

USRP Sink
Unit Number: 0
Interpolation: 400
Frequency (Hz): 902M
Gain (dB): 0
Side: A
Transmit: Auto T/R

Python

Python code from the GRC

- The GRC creates python code from signal-processing flow-graphs
- Further program logic added in Python



Node
● Transmitter
○ Receiver

Which USRP
● 0
○ 1
○ 2
○ 3

Iteration:

Amplitude RMS:

Matlab

Control USRP devices via Matlab

- Control of USRP software radios via Matlab possible
  - Developed by the Institut für Nachrichtentechnik at the Universität Karlsruhe
  - Build models in Simulink that interface with the USRP
  - Direct use of signal processing capabilities of Matlab

Software radios for OC

Direct application to OC-Applications:

- OC-communication protocols
  - Easy implementation and case study with novel protocols
  - Protocol that fosters self-organisation

Indirect application to OC-Applications:

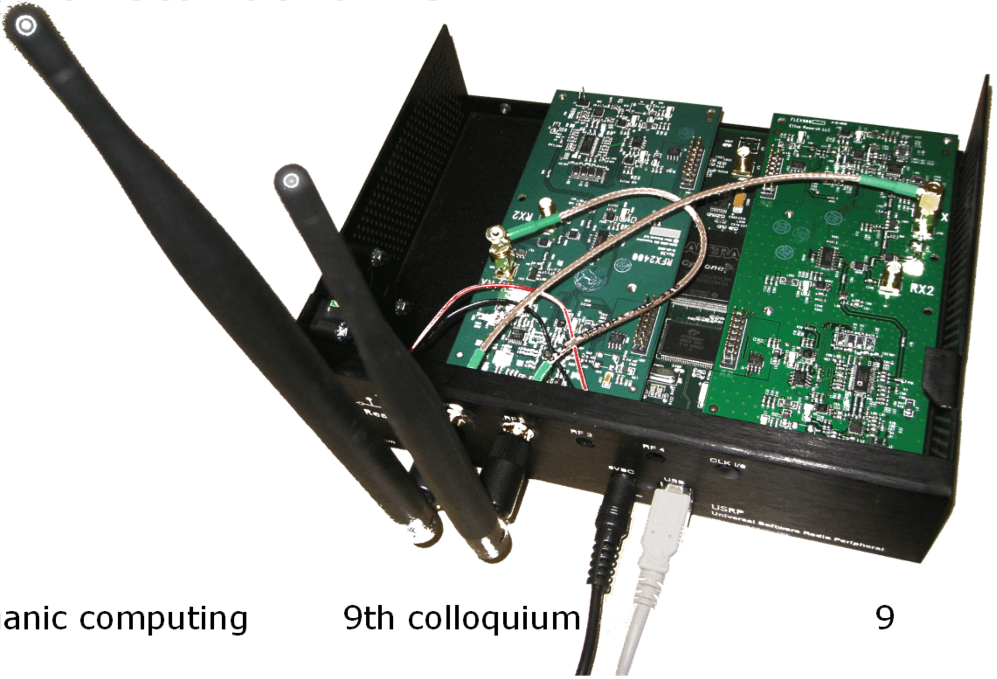
- Equip standard PC with communication interface
  - Test of distributed, autonomous operation of nodes
  - Distributed OC-components developed and executed on standard PC
  - Communication under real-life conditions

Use cases in other OC-SPP projects?

Conclusion

Benefits of utilising USRP-software radios for OC apps:

- Easy development and deployment
- Quick prototyping of application
- Communication under real-life conditions
- Computational power of standard PC



Questions?

Thank you for your attention.