TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik Institut für Technische Informatik, Professur für VLSI-Entwurfssysteme, Diagnostik und Architektur

Vorstellung

CRUSH:
Cognitive Radio Universal Software Hardware

Sebastian Schmid

Dresden, 10.07.13

Inhalt

Vorstellung des Papers:
CRUSH: Cognitive Radio Universal Software Hardware

Eichinger, G. / Chowdhury, K. / Leeser, M. (2012)
22nd International Conference on
Field Programmable Logic and Applications (FPL), S. 26-32, 2012

1 Grundlagen

2 CRUSH-Plattform

3 Implementierung

4 Ergebnisse

1 Grundlagen

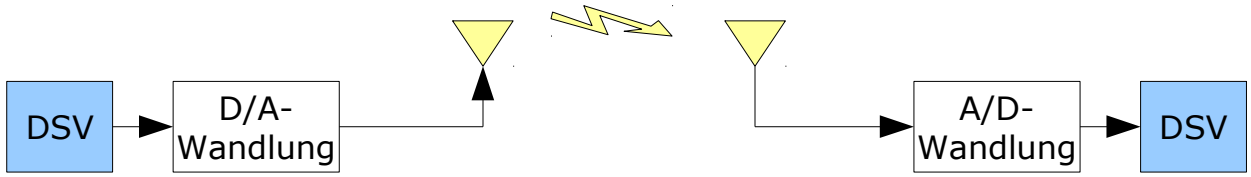
1.1 Software Defined Radio

Definition:

- "Radio in which some or all of the physical layer functions are software defined" ([Sd08], S. 1)
  → "software defined" meint hier die eigentliche Signalverarbeitung

Funktionsweise:

- Idealfall: direkte A/D- bzw. D/A-Wandlung an Antenne, keine analogen Bauelemente nötig
- Praxis: Ansprüche an Wandler-ICs zu hoch (Bsp. LTE: 2.6 GHz Band → 5.2 GSPS)
  → häufig analoge Mischung auf niedrigere Frequenzen



1 Grundlagen

1.1 Software Defined Radio (Fortsetzung)

Vorteile:

- hohe Flexibilität (Änderung elementarer Eigenschaften: Modulationsverfahren, Frequenzband, …) (vgl. [Sd08], S. 1)
  → nachträgliche Anpassung an neue Standards, Fehlerbehebung
  → interessant für Forschung

- Kosteneinsparungen möglich

Nachteile:

- aufwendige Digitalverarbeitung
  → leistungsfähige Geräte nötig
  → Energieverbrauch größer als bei herkömmlichen Funksystemen

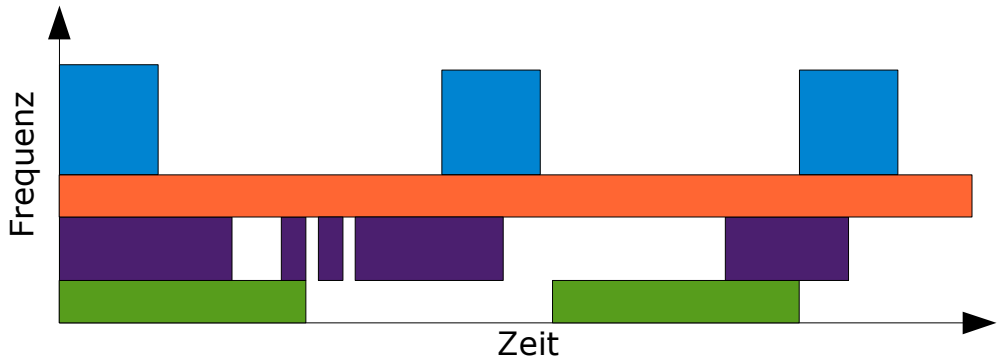
1 Grundlagen

1.2 Cognitive Radio

"Cognitive radio is radio in which communication systems are **aware of their** [...] **environment**, such as location and utilization on RF frequency spectrum at that location. They **can make decisions about their radio operating behaviour** by mapping that information against objectives."
([Sd08], S. 5)

Schwerpunkt:

- flexible Wahl der Übertragungsparameter unter Berücksichtigung anderer aktiver Funksysteme (PU, Primary User)
  → optimaler Nutzung des verfügbaren Frequenzbands (vgl. [Sd08], S. 6)
  → nötig: Spectrum Sensing

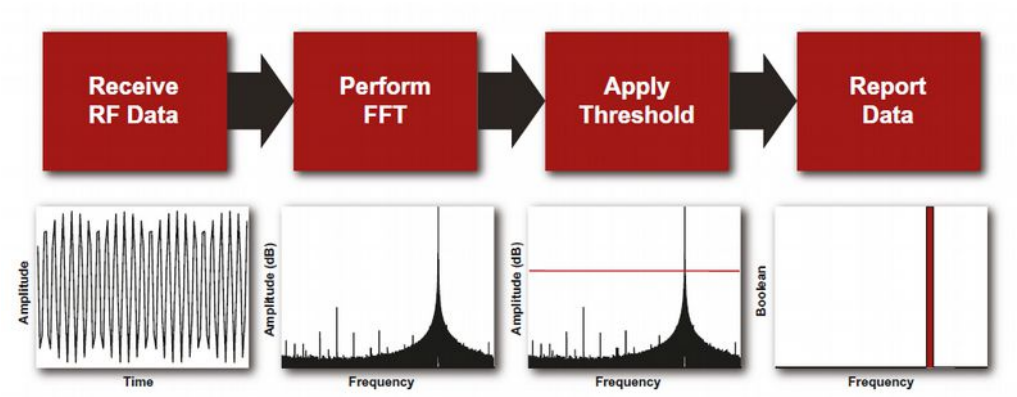


1 Grundlagen

1.2 Cognitive Radio (Fortsetzung)

Funktionsprinzip:

- Durchführen einer n-Punkte FFT
- Amplitude der resultierenden n komplexen Ergebnisse mit Schwellwert vergleichen: Amplitude größer → Frequenz durch PU belegt



[Ei12b], S. 28

Problem:

- wird ein PU wieder aktiv, darf dieser nur möglichst kurz gestört werden → „algorithms on the µs scale“, vgl. [Ei12a], S. 42

2 CRUSH-Plattform

2.1 Universal Software Radio Peripheral

- entwickelt von Ettus Research (seit 2010: Tochter von National Instruments)
- sehr verbreitet (insbesondere im universitären Umfeld)
- austauschbarer Hochfrequenzteil, verfügbar für verschiedene Frequenzbereiche

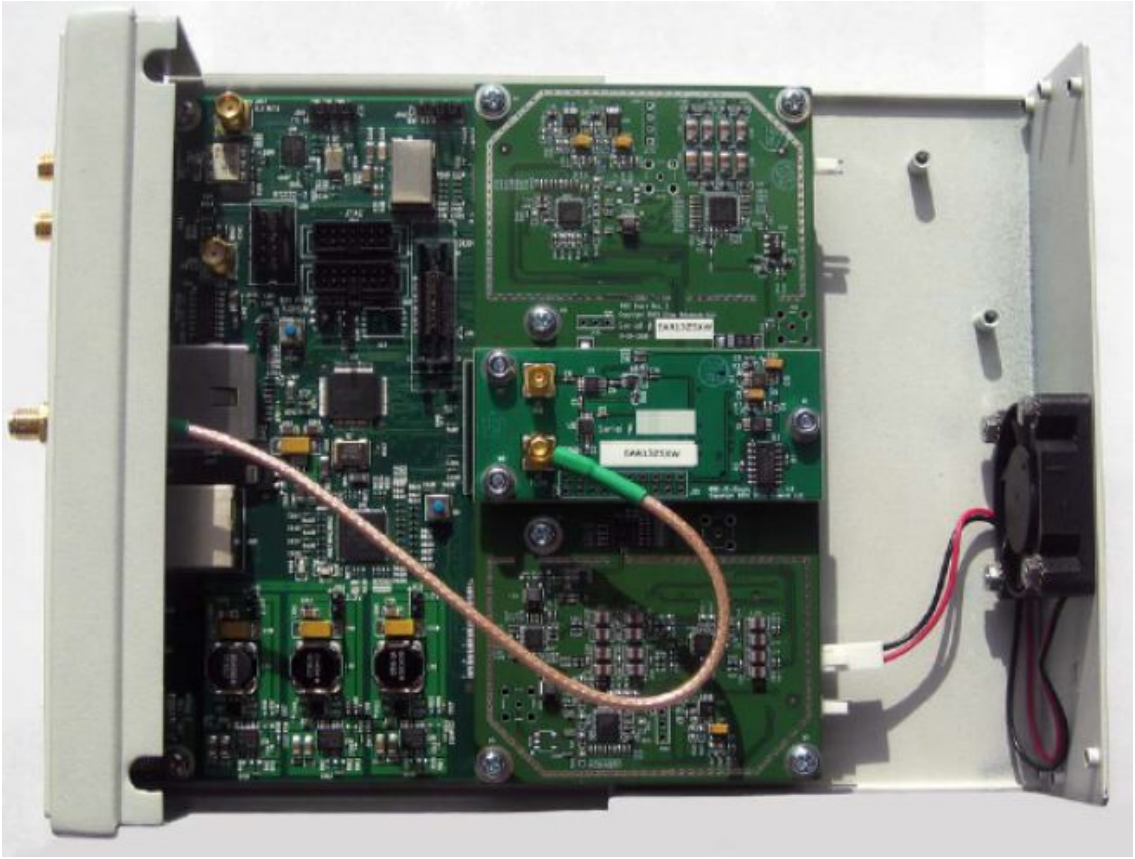
USRP N210:

vgl. [Et12]

- 14 Bit 100 MSPS A/D- und 16 Bit 400 MSPS D/A-Wandler
- Xilinx Spartan 3A-DSP 3400
- Gigabit-Ethernet-Schnittstelle zur Kommunikation mit dem Host
  → 1000 Mbit/s / (2x 16 Bit) = 31,25 MHz
  → Praxis: 25 MHz Bandbreite

2 CRUSH-Plattform

2.1 USRP (Fortsetzung)



[Ei12a], S. 6

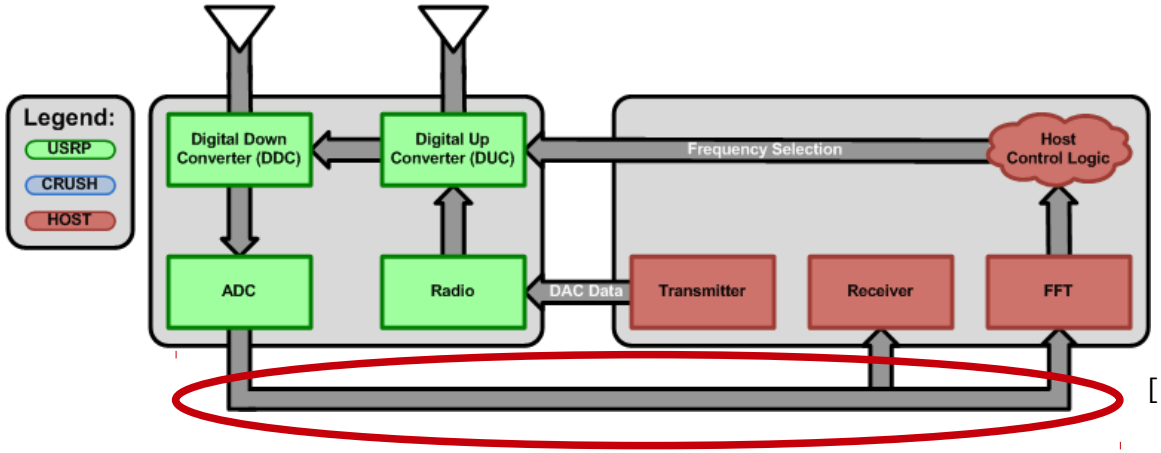
2 CRUSH-Plattform

2.2 Cognitive Radio mit USRP

Zielstellung:

- Nutzung eines USRP zur Untersuchung von Cognitive Radio-Systemen

→ Problem: Übertragung der Empfangsdaten über GBit-Ethernet führt zu unzulässig großen Verzögerungen



[Ei12a], S. 41

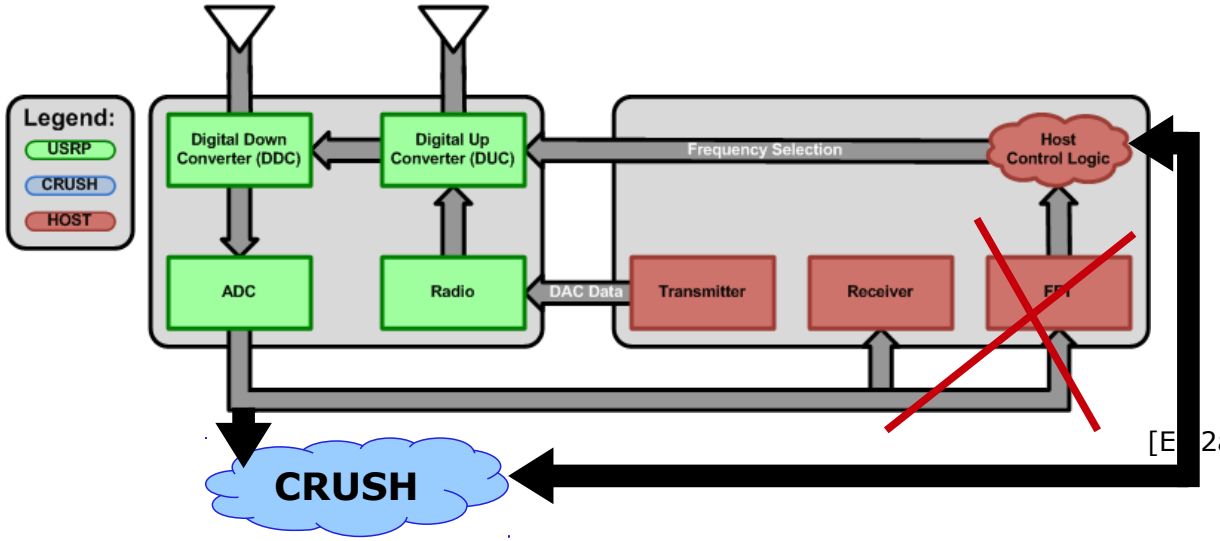
2 CRUSH-Plattform

2.2 Cognitive Radio mit USRP (Fortsetzung)

Lösung:

- Signalverarbeitung näher zum Empfänger bringen

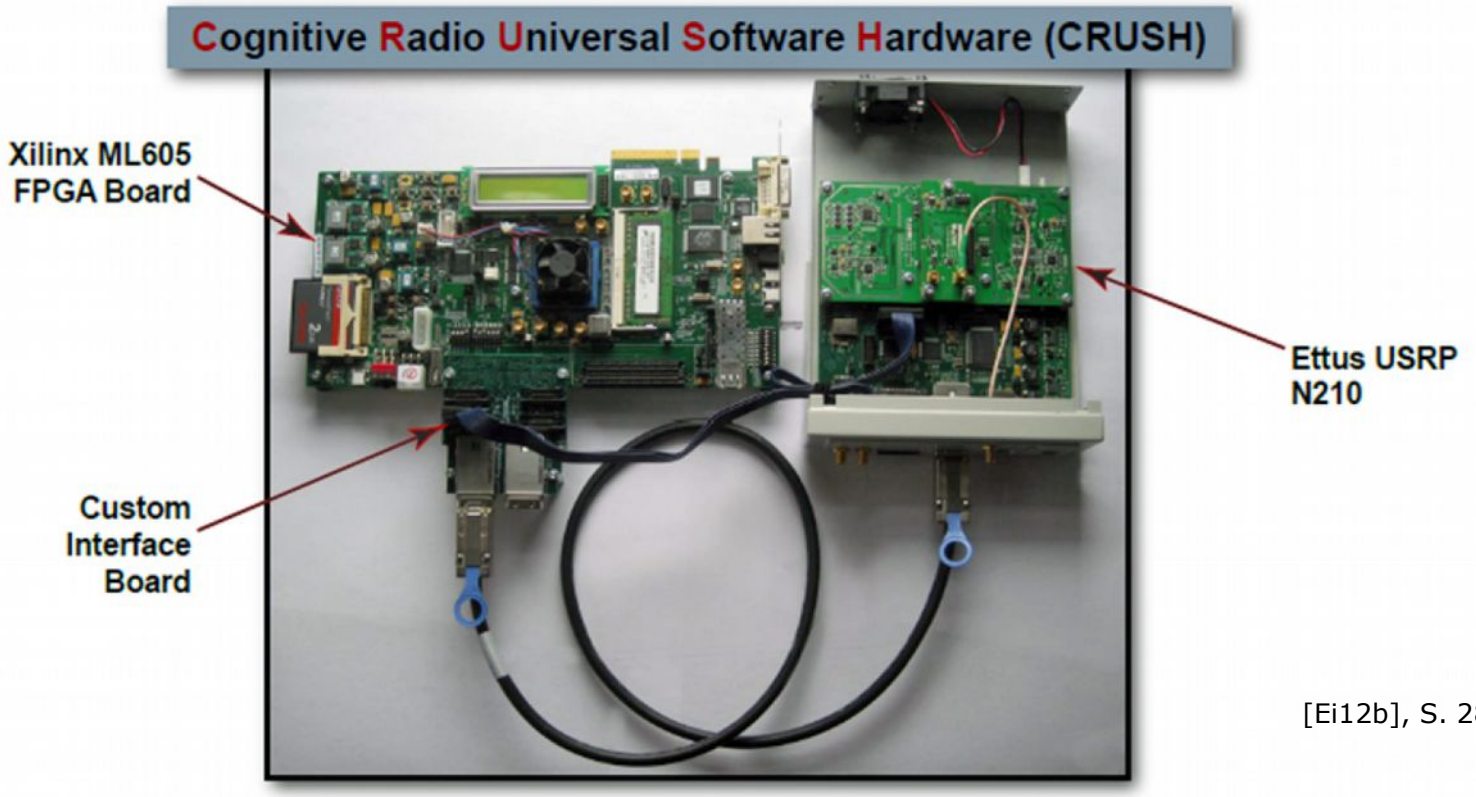
  → Spectrum Sensing in Hardware ausführen
  → nur aufbereitete Informationen (Frequenzbelegung) an Host übermitteln



[E 2a], S. 41

2 CRUSH-Plattform

2.3 Hardware-Übersicht



[Ei12b], S. 28

2 CRUSH-Plattform

2.4 FPGA

Warum zusätzliches FPGA?

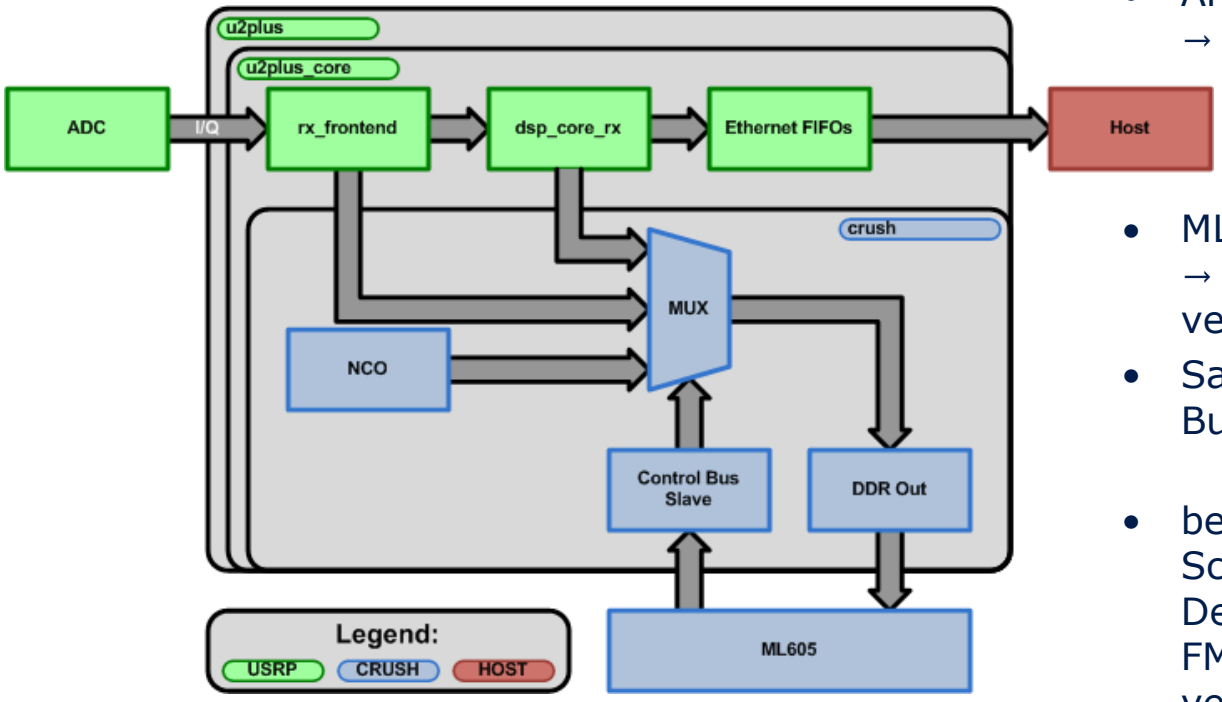
- Spartan 3A wenig leistungsfähig (kleiner Block-RAM, wenig DSP-Blöcke), bereits mit verschiedenen Funktionen des USRP belastet
- Kompatibilität zu anderen SDR-Plattformen ermöglichen (ältere USRP-Generationen, z.B. USRP2)
- größere Änderung an HDL des USRP vermeiden (Einarbeitungszeit, Timing-Probleme)

	Xilinx Spartan 3A-DSP3400 (USRP N210)	Xilinx Virtex-6 LX240T (ML605 Development Board)
Anzahl Slices	23.872	56.880
Block RAM [Kib]	2.268	14.976
DSP Blocks	126	768

vgl. [Xi10], [Xi12]

3 Implementierung

3.1 USRP HDL

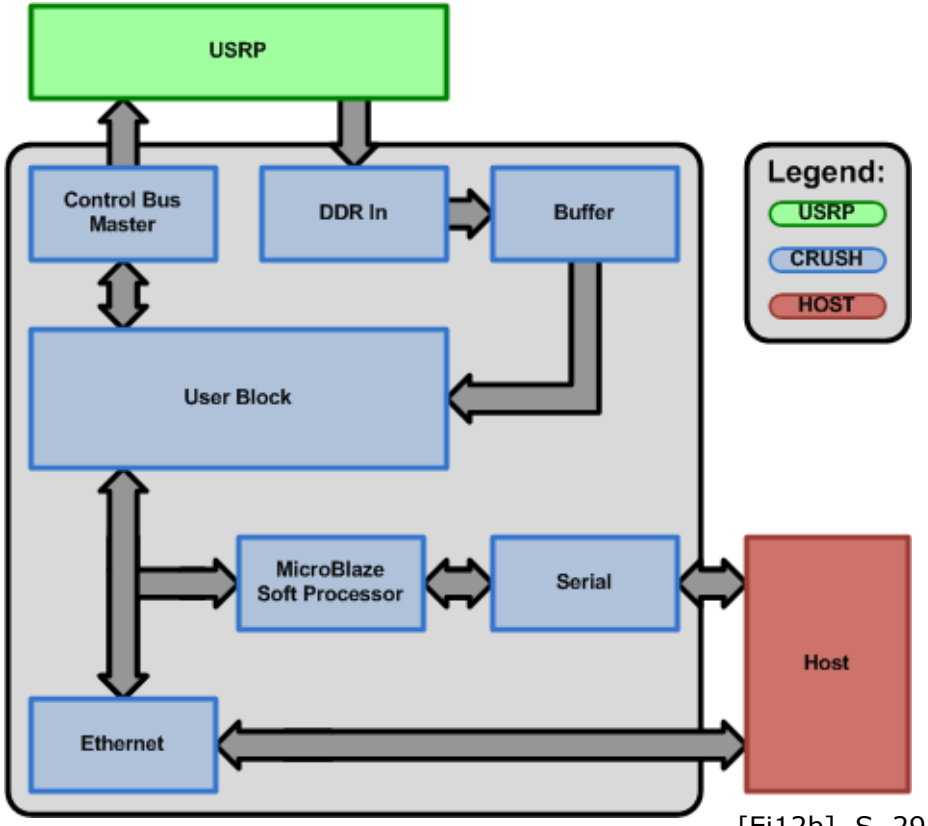


- Änderungen am USRP HDL → ermöglichen Kommunikation mit dem ML605 Board
- ML605 gibt Modus vor → MUX: Abgriff der Samples an verschiedenen Stellen möglich
- Samples werden auf DDR I/Q-Bus gegeben
- beide Busse über USRP MICTOR-Schnittstelle (ursprünglich für Debugging, 34 Bit breit) mit FMC-Schnittstelle des ML605 verbunden

[Ei12b], S. 29

3 Implementierung

3.2 ML605 HDL



[Ei12b], S. 29

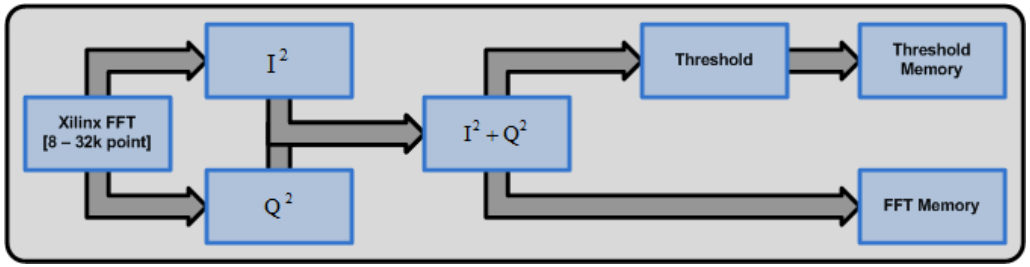
- Kommunikation sowohl über Ethernet als auch serielle Schnittstelle möglich
- für User Block stehen noch ca. 97% der Ressourcen zur Verfügung (vgl. [Eich12b], S.29)

3 Implementierung

3.2 ML605 HDL (Fortsetzung)

User Block für Spectrum Sensing:

- Xilinx Streaming FFT (8 bis 4.096 Punkte)
- Vergleich mit dem (vom Host) quadriert vorgegebenen Schwellwert



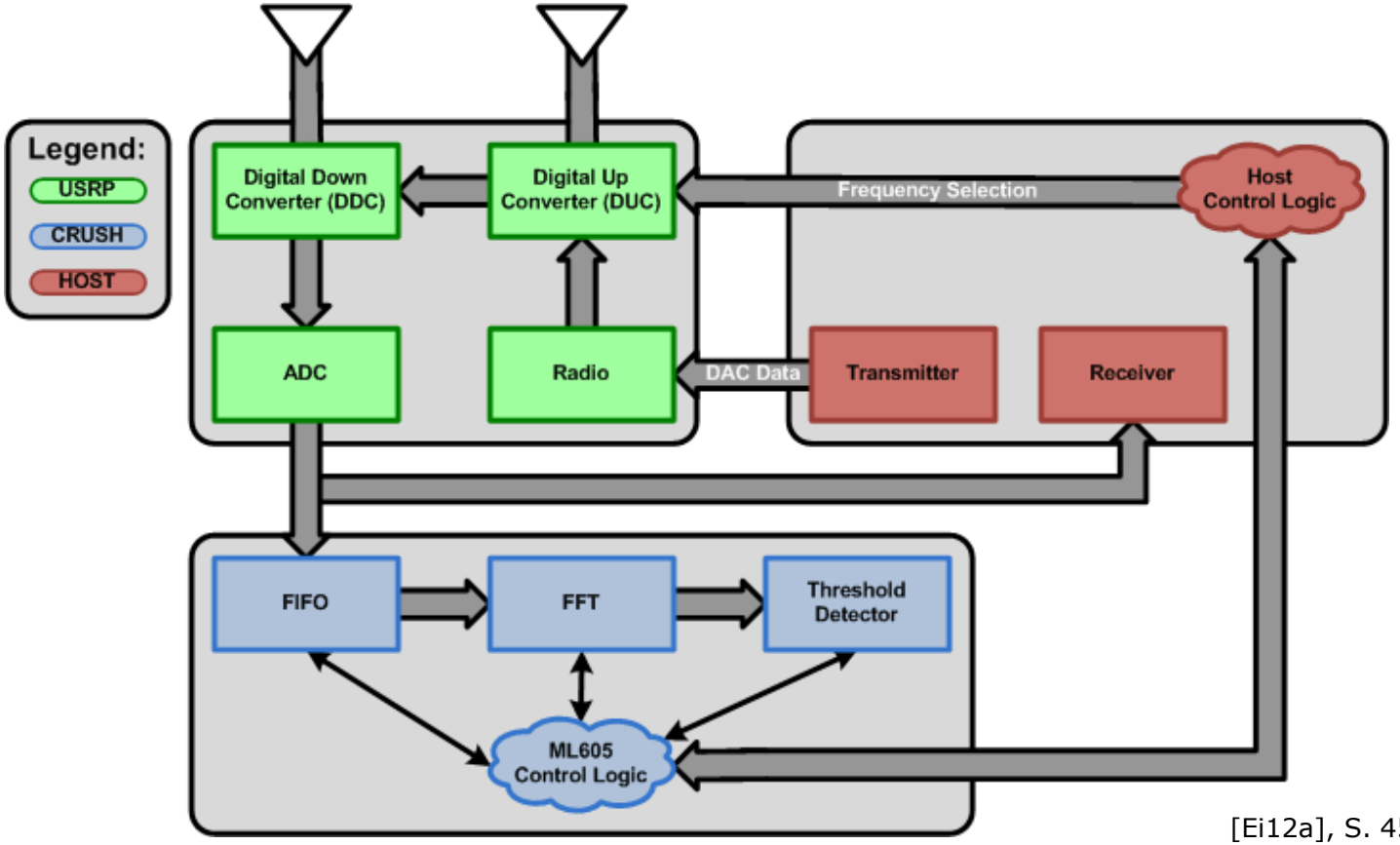
[Ei12b], S. 30

Optimierung der Verzögerung bei Kommunikation mit Host:

- Nutzung von UDP zur Vermeidung von Overhead
- Paketgröße wird passend zur Anzahl der FFT-Punkte gewählt
- Softcore berechnet Prüfsummen aller möglichen Pseudo-Header und speichert diese im Block-RAM → schneller Zugriff

3 Implementierung

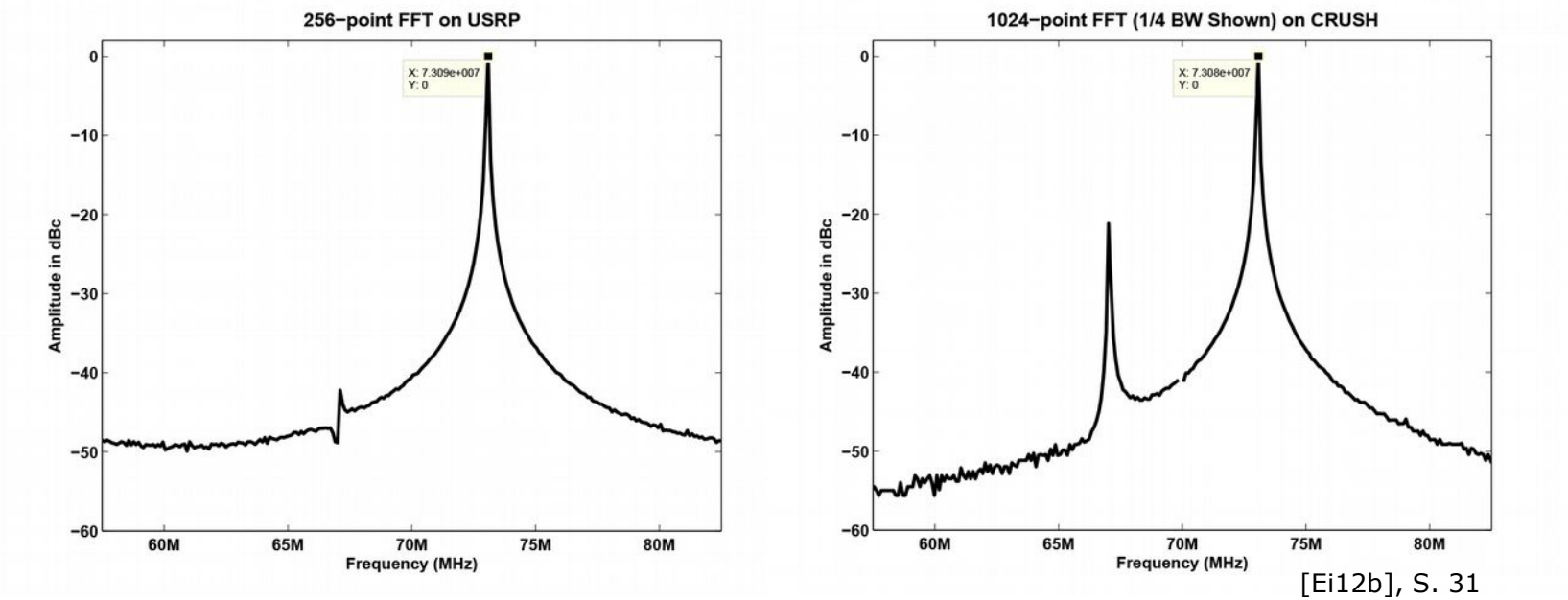
3.3 Übersicht



[Ei12a], S. 45

4 Ergebnisse

4.1 Funktionalität

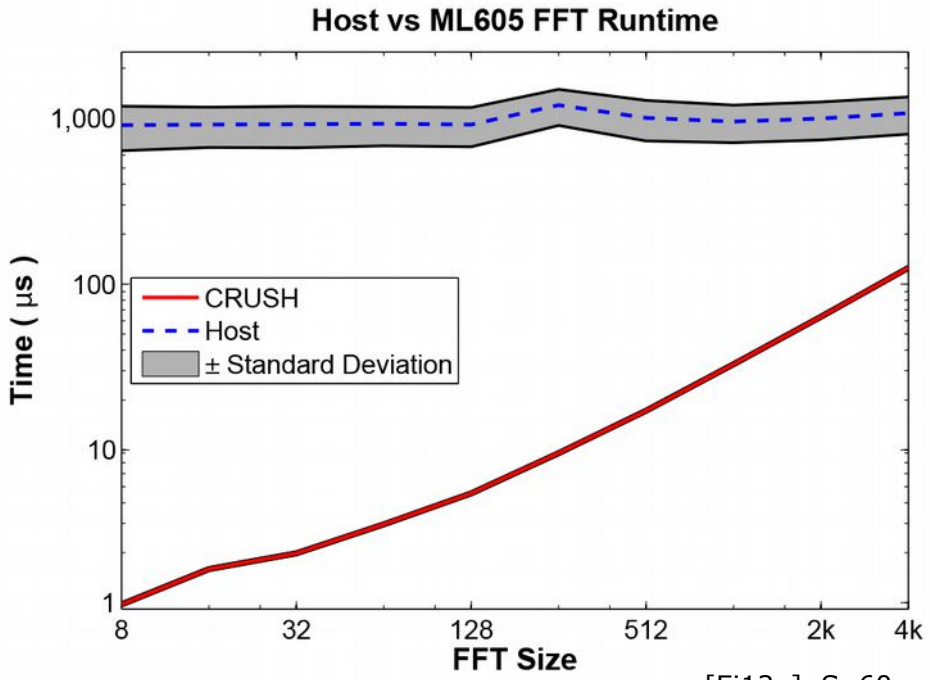


[Ei12b], S. 31

- Test des FFT-Blocks: Signalgenerator (73 MHz) am Eingang angeschlossen
- Spiegelbild-Artefakt bei 67 MHz (Mittenfrequenz des USRP lag bei 70 MHz)
  → stärker als bei USRP-Variante, da Signal weniger stark gefiltert

4 Ergebnisse

4.2 FFT-Performance



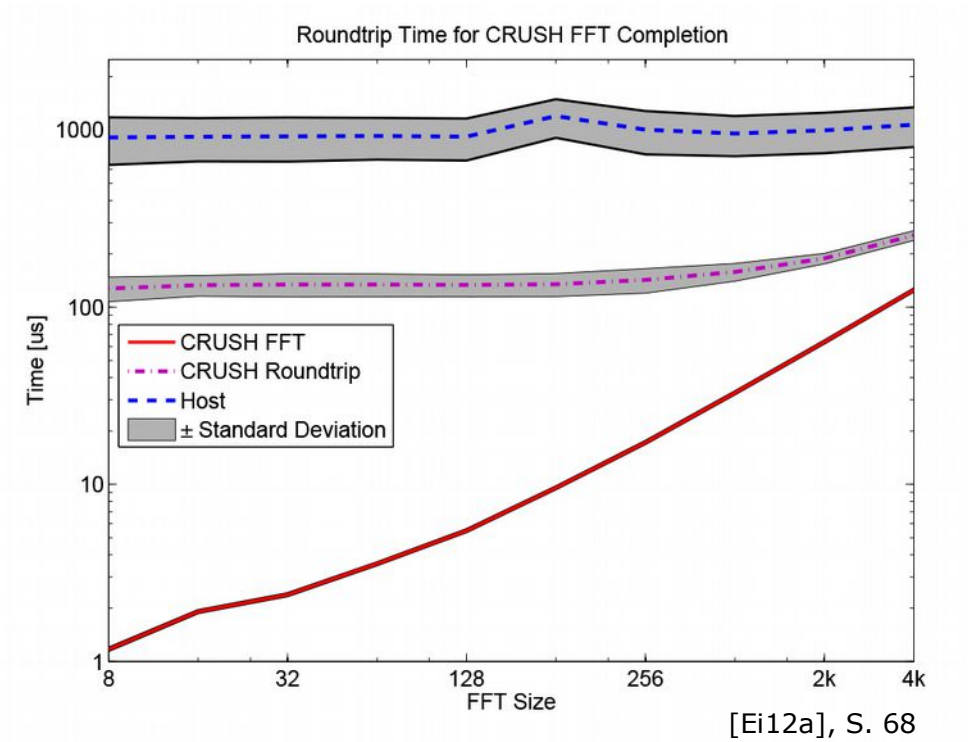
[Ei12a], S. 60

FFT Größe	ML605 [µs]	Host [µs]	Speed-up
8	1,17	907,72	774
16	1,91	915,89	479
32	2,38	920,07	386
64	3,56	925,47	259
128	5,47	916,54	167
256	9,56	1198,19	125
512	17,23	1003,83	58
1024	32,84	955,30	29
2048	63,55	995,26	15
4096	125,24	1071,79	8

vgl. [Ei12a], S. 62

4 Ergebnisse

4.3 Roundtrip-Performance



[Ei12a], S. 68

- Messung bis Host Informationen über belegte Frequenzen erhält
- Host-Variante: Verzögerung durch Netzwerkkommunikation groß, Anteil der FFT kaum zu erkennen
- CRUSH: Verzögerung durch Kommunikation deutlich geringer, aber immer noch Hauptanteil
  → trotzdem ca. eine Größenordnung zwischen beiden Varianten

4 Ergebnisse

4.4 Zusammenfassung

- CRUSH-Plattform hat sich als funktionsfähig herausgestellt, die Reaktion auf veränderte Kanalbedingungen konnte ca. um den Faktor 10 beschleunigt werden

  → dabei wurde die Flexibilität bei der Anpassung der Parameter (Anzahl der FFT Punkte, Schwellwert) des Spectrum Sensings beibehalten

- Ressourcen des Virtex 6 werden durch CRUSH kaum ausgeschöpft, Implementierung anderer zeitkritischer Algorithmen möglich

- Erweiterungen denkbar, CRUSH kann mit bis zu 3 USRPs kommunizieren

5 Quellen

[Ei12a] Eichinger, G. (2012):
"CRUSH: Cognitive Radio Universal Software Hardware",
Master-Thesis, Northeastern University Boston, 2012

[Ei12b] Eichinger, G. / Chowdhury, K. / Leeser, M. (2012):
"CRUSH: Cognitive Radio Universal Software Hardware",
22nd International Conference on Field Programmable
Logic and Applications (FPL), S. 26-32, 2012

[Et12] Ettus Research (2012):
"USRP Networked Series Spec. Sheet"

[Sd08] SDR Forum (2008):
"What is Software Defined Radio"

[Xi10] Xilinx (2010):
"Spartan-3A DSP FPGA Family Data Sheet"

[Xi12] Xilinx (2012):
"Virtex-6 Family Overview"

»Wissen schafft Brücken.«