Vortrag zum Diplom

Konzeption und Realisierung von Test- und Debugtechniken zur Prototypevaluation der grobgranular-rekonfigurierbaren ARRIVE Architektur

Martin Zimmerling

mz793134@inf.tu-dresden.de

Institut für Technische Informatik

Aufgabe

- Implementationsmöglichkeiten aufzeigen zur Inbetriebnahme, Test und Debugging anhand des VHDL-Modells der ARRIVE-Architektur
- Prototyp-Realisierung auf Basis eines FPGAs
- Visualisierung mittels einer Test- und Debug-Umgebung
- Evaluation bzgl. benötigter HW-Ressourcen und Taktfrequenz
- Vergleich zu Standardzellen-ASIC-Synthese

Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

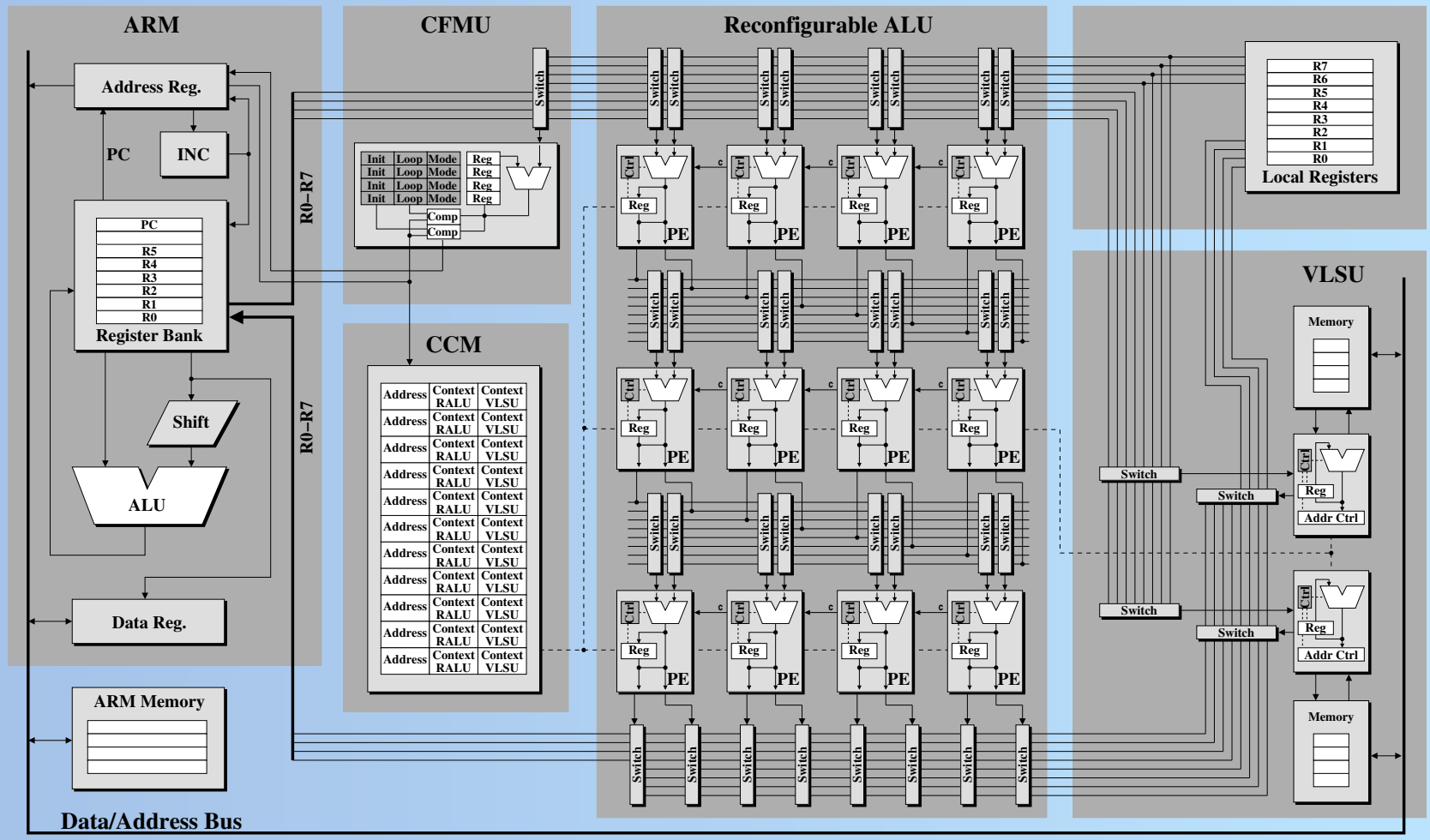
Gliederung

- Überblick über das ARRIVE-Modell
- allgemeine Debug-Konzepte
- Anforderungen an eine Debug-Schnittstelle
- Konzept der Debug-Unit
  - Aufbau
  - Befehlssatz
- Debugger Oberfläche
- Änderung des ARRIVE-Modells (CFMU)
- Ausblick

Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

Überblick ARRIVE

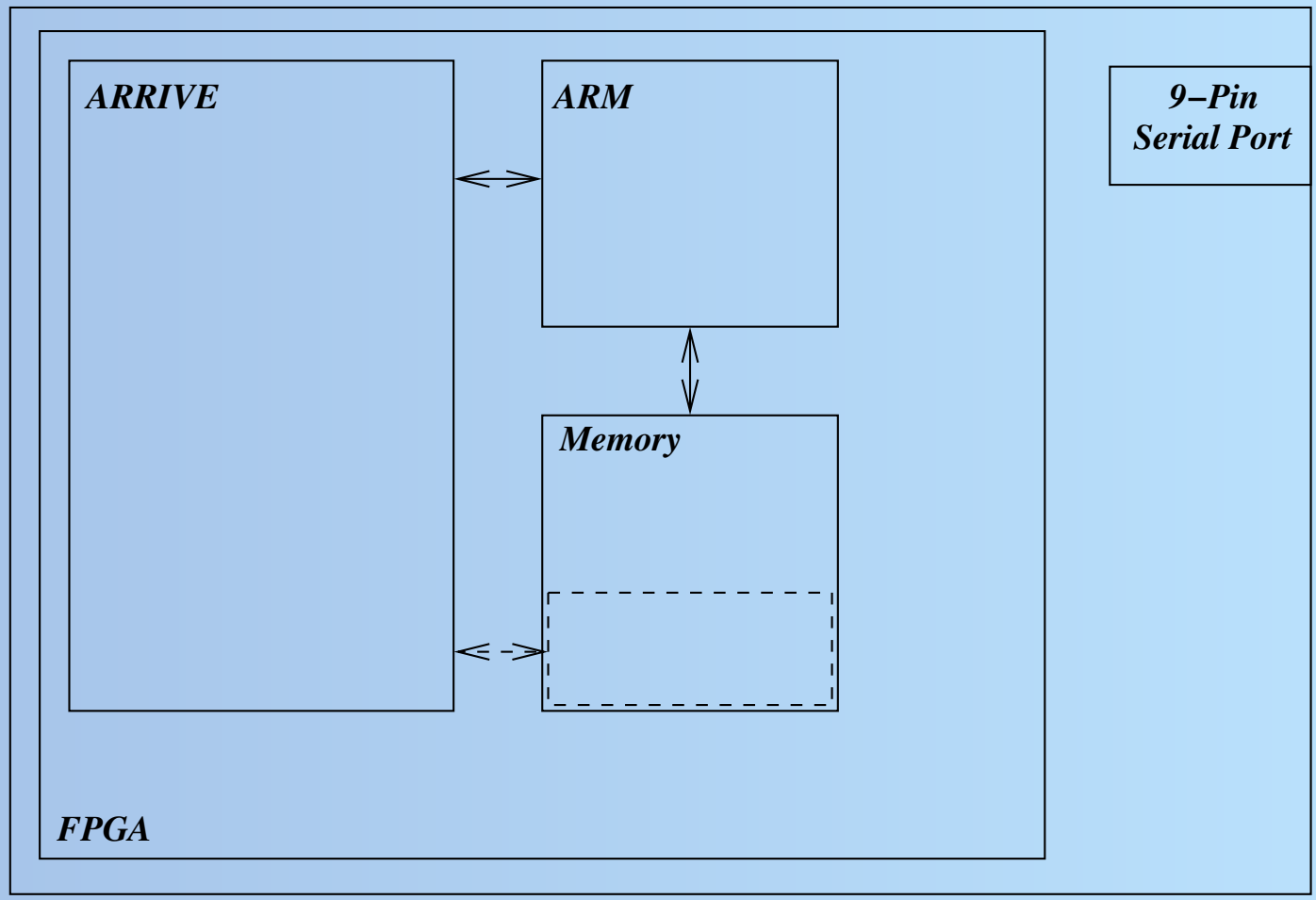
ARRIVE-Architektur:



Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

Überblick FPGA-Board 1

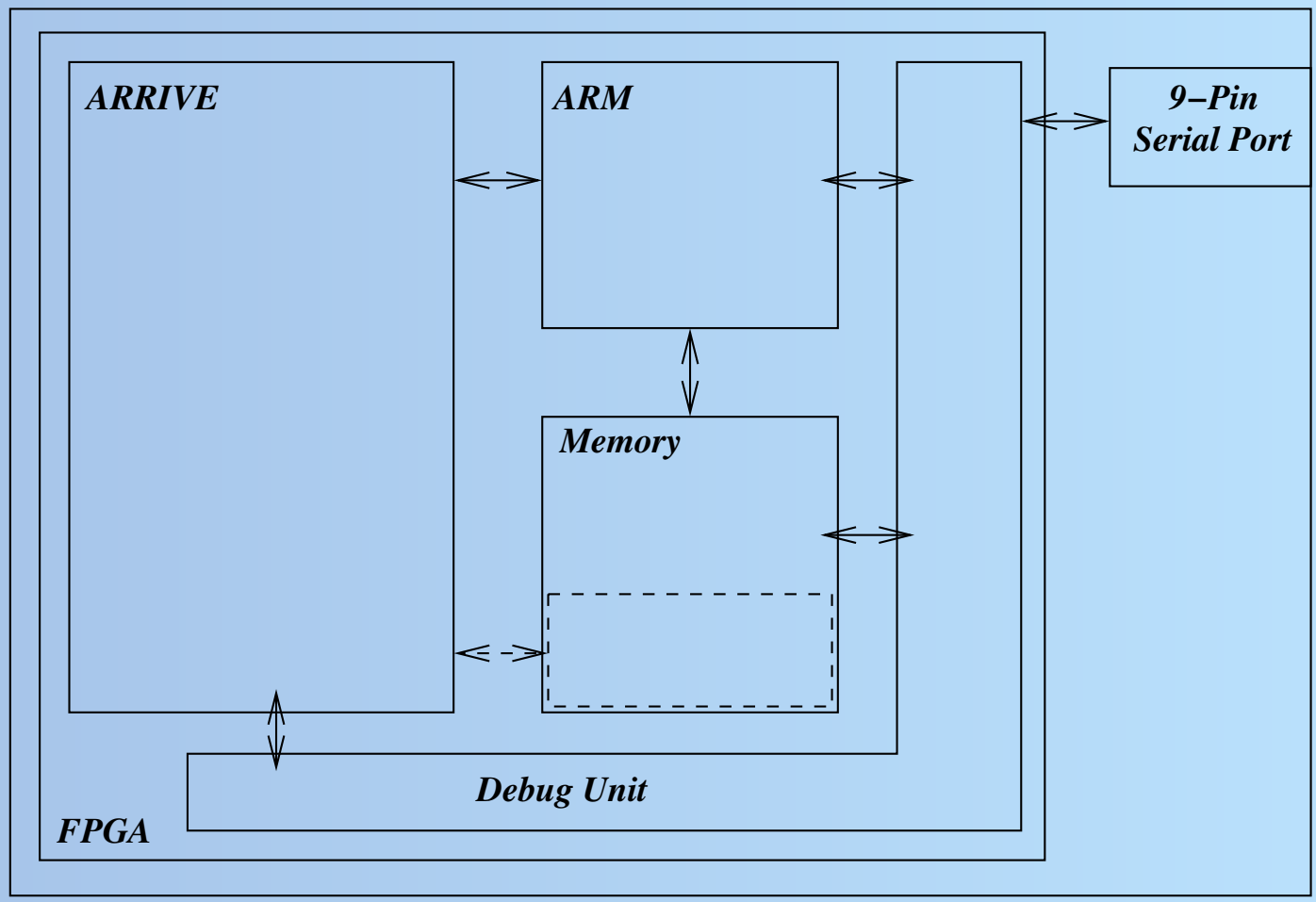
Ausgangsmodell:



Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

Überblick FPGA-Board 2

Zielmodell:



Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

allgemeine Debug-Konzepte

- Anhalten, Ausführen des Programmflusses
- Lesen / Ändern des internen Zustandes
- non-intrusive debugging (SoC)
- Debugregister
- HW-Breakpoints
- Single-Stepping
- Tracing, Trace-Cache (-)
- interne Zähler (-)

Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

Anforderungen I

Minimale Anforderungen:

- Lesen / Schreiben von Registern
- Lesen / Schreiben von Konfigurationstabellen
- Lesen / Schreiben von Speicher
- „unsichtbares“ Pipelining
- Konfigurierbarkeit wie VHDL-Modell

Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

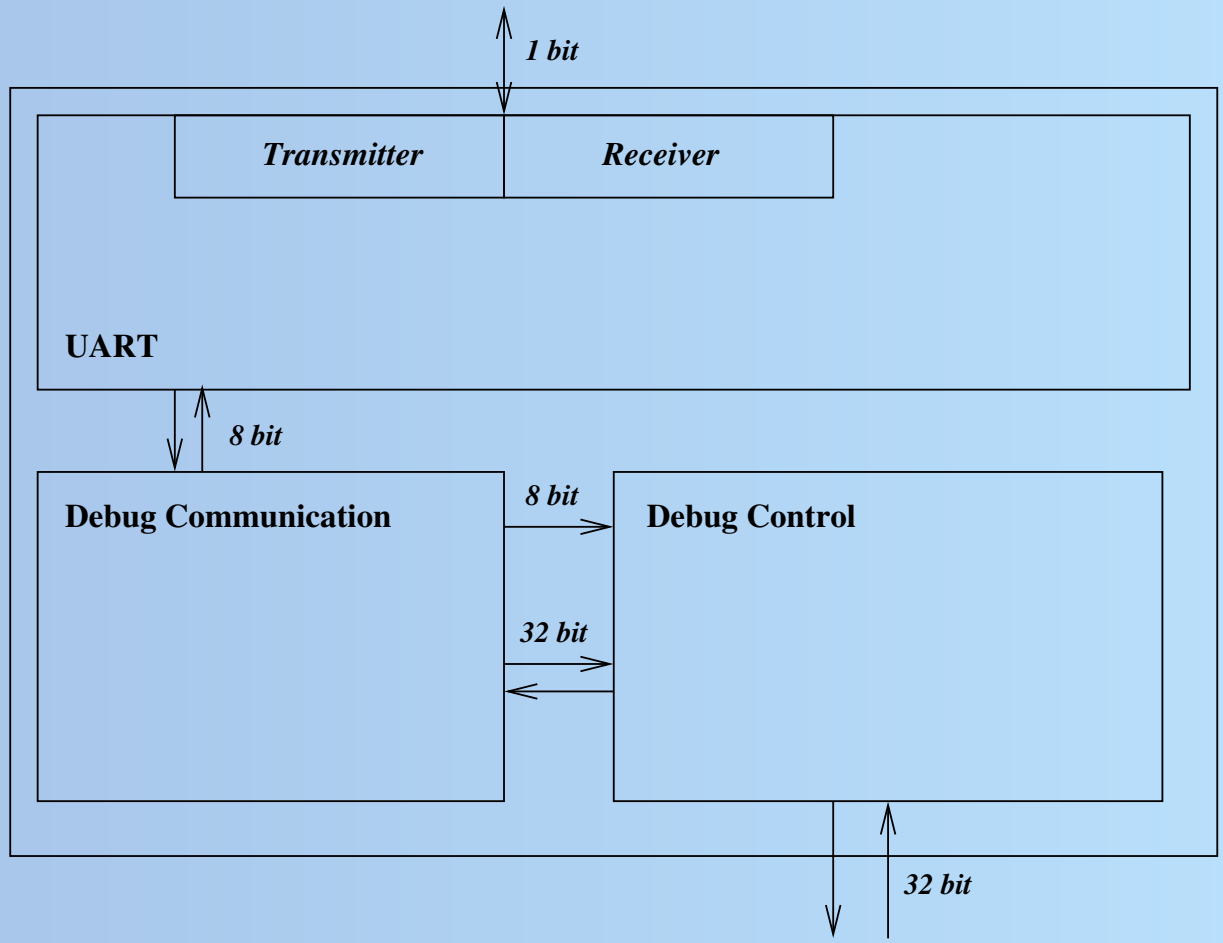
Anforderungen II

Zusätzliche Anforderungen:

- möglichst schnelle und sichere Datenübertragung
- geringer Einfluss auf Geschwindigkeit und HW-Aufwand
- modularer Aufbau, Ermöglichung der Nutzung einer schnelleren Schnittstelle
- Vielseitigkeit (Nutzung durch ARRIVE-Architektur selbst, Nutzung fuer Taktmessungen)

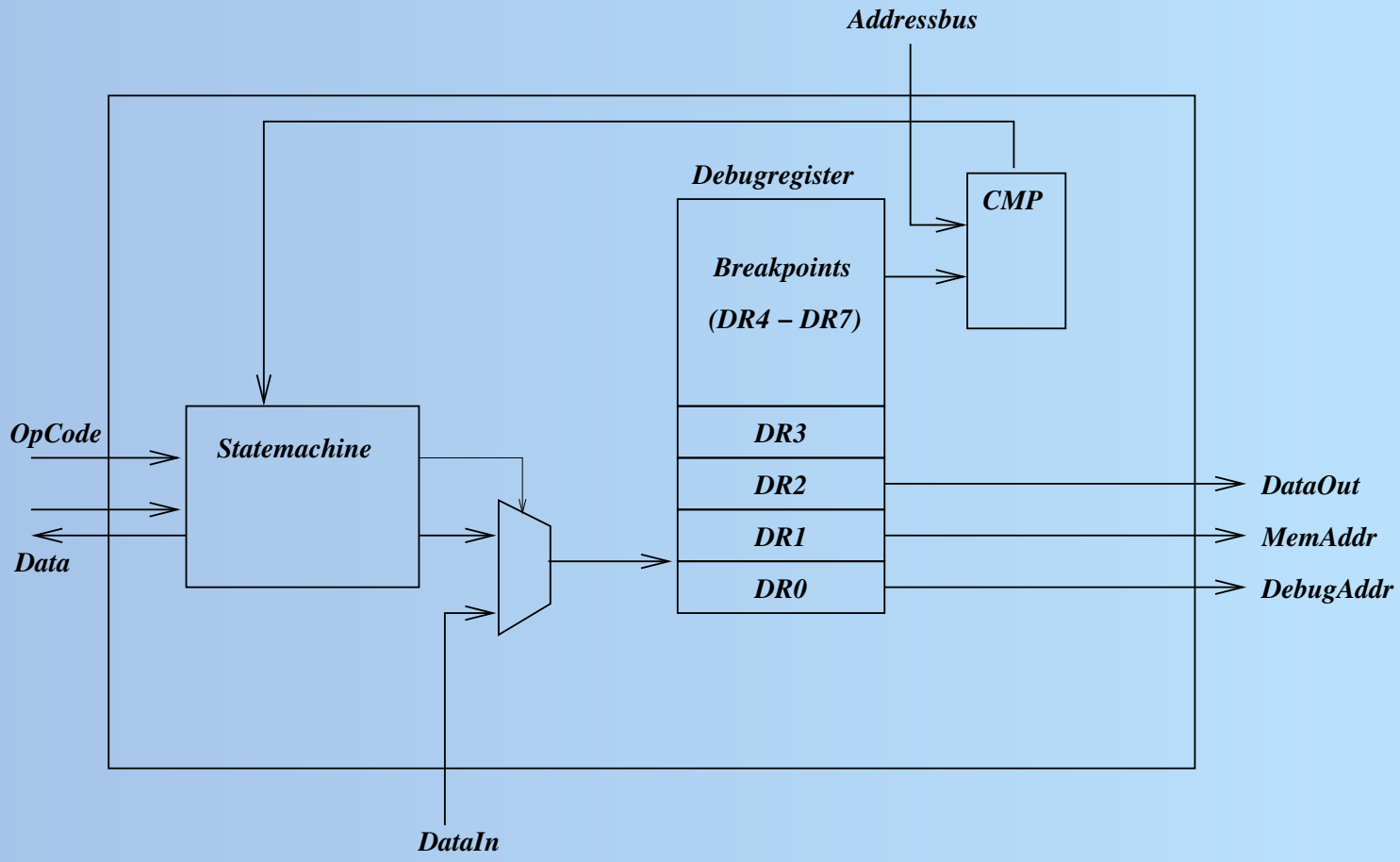
Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

DebugUnit:



Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

Debug Control (intern):



Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

Debugger Befehlssatz

- 8 Bit OpCode
- Steuerinformationen für DebugCommunication im OpCode enthalten

Ex	RW	M	I		RegSel	Befehl
1	0	0	0	1	Reg	SetDebugRegister (+32 bit)
1	1	0	0	1	Reg	GetDebugRegister (-32 bit)
0	0	1	I	0	xxx	WrMem[Inc]
0	1	1	I	0	xxx	RdMem[Inc]
0	0	0	I	0	xxx	WrReg[Inc]
0	1	0	I	0	xxx	RdReg[Inc]
0	0	0	0	1	000	STOP
0	0	0	0	1	001	STEP
0	0	0	0	1	111	RUN

Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

Debug-Adressraum

- 32-bit Adressraum (4G 32-bit Worte)
- Beispiel ARM/CFMU (untere 7 Bit):

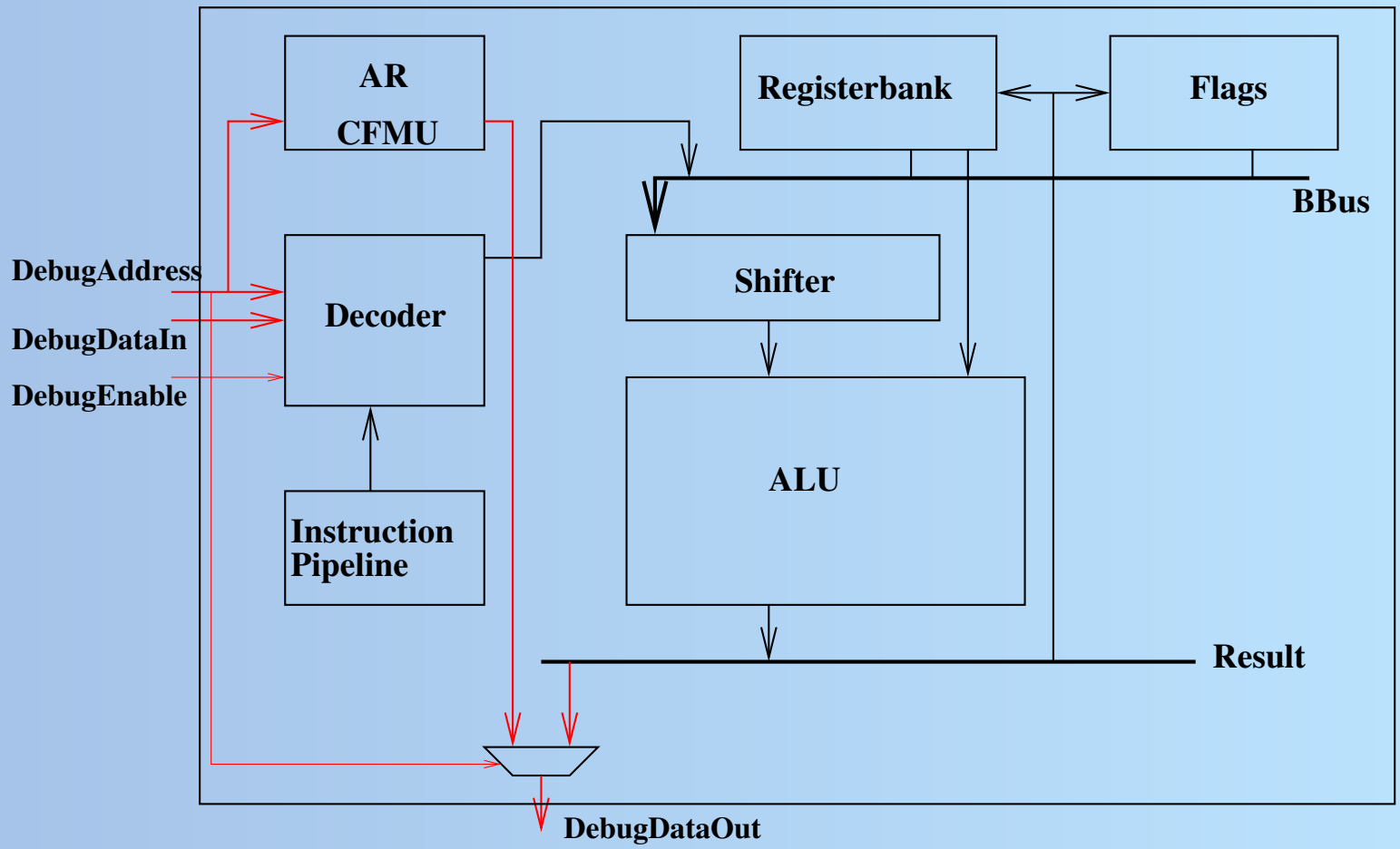
Bit	1	2	4	Beschreibung
	0	Status/Reg	RegNr	ARM
	1	A/C/S	SP	CFMU

A/C/S -> AddressStart, -End, Counter, Status

- Einbindung ARRIVE-Kern in Adressraum
  - Lokale Register
  - Rekonfigurationstabellen (CCM, VLSU, RALU)
  - Prozessorelemente (Ausgänge, Register)
- Konfigurierbarkeit des VHDL-Modells beachten

Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

Anbindung des ARM an den Debug-Adressraum



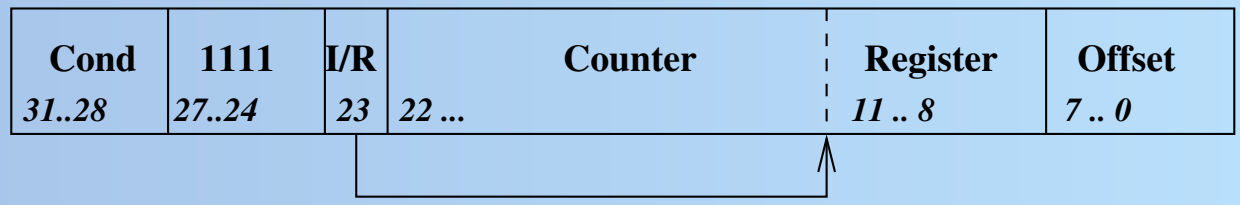
Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

User Interface

- Textoberfläche
- Konfigurierbarkeit entsprechend dem VHDL-Modell
- Interaktion durch Nutzer und Hardware (Breakpoints)
- Abbildung der Debugger-Befehle
- Makrobefehle (Programm laden, Speicherausgabe in Datei)

Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

- CFMU in Adressregister eingebunden
- Stack statt Tabelle, Befehl im Befehlsstrom
  - Vorteil: weniger HW-Aufwand
  - Nachteil: ein zusätzlicher Takt
  - Schleife mind. 2 Befehle lang
- Befehlsformat (SWI):



- Limitierung
  - Schleifenlänge max. 256 Befehle (1024 Byte)
  - Schleifenzähler max. 15 Bit, mind. 1 Durchlauf

Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de

Ausblick

- Implementierung fertigstellen
- Programmierung Debugger-Userinterface
- Abschätzung HW-Aufwand, Timing
- Vergleich zu ASIC Entwurf
- einfache Fehlererkennung

Martin Zimmerling
Institut für Technische Informatik
mz793134@inf.tu-dresden.de