Spektral gesteuerter Schweißregler
WCTR & SOS

- Aufgaben & Funktion
- Veränderter Lösungsansatz
- Selektion der Hardware
- Modifizierte Zielarchitektur
- Quinto-Interaktion
- Kommunikation
- Anschlüsse und Datenaustausch Quinto
- Anschlüsse Champ, dRec
- Software Aufgaben & Methoden
- Software Schwerpunkte
- Betriebssystem

Dr. G. Heinz, GFaI e.V.
Rudower Chausee 30
12489 Berlin
Tel. +49 (30) 6392 -1652
Fax. -1602
www.gfai.de/~heinz
heinz@gfai.de

Vortrag zur 3. Sitzung des PbA,
Projekt Spektral geregelte
Pulsschweißmaschinen (SPS),
17.7.2009 9.00 Uhr,
Felix-Hausdorff-Straße 2,
17489 Greifswald

www.gfai.de/~heinz/techdocs

Aufgaben & Funktion

Ziele

- Messung von Zeitfunktionen mit 4…6 spektral differenten Photodioden im Spektralbereich 0,3…3 µm
- Messung von U und I
- Aufnahme des Datenverkehrs zwischen Bedienteil (Prozeßregler) und Stromquelle der Cloos Quinto 403
- Prozeßregelung
  - Ausgabe geänderter Steuerparamter an Stromquelle
  - Abschaltsignal über RS485

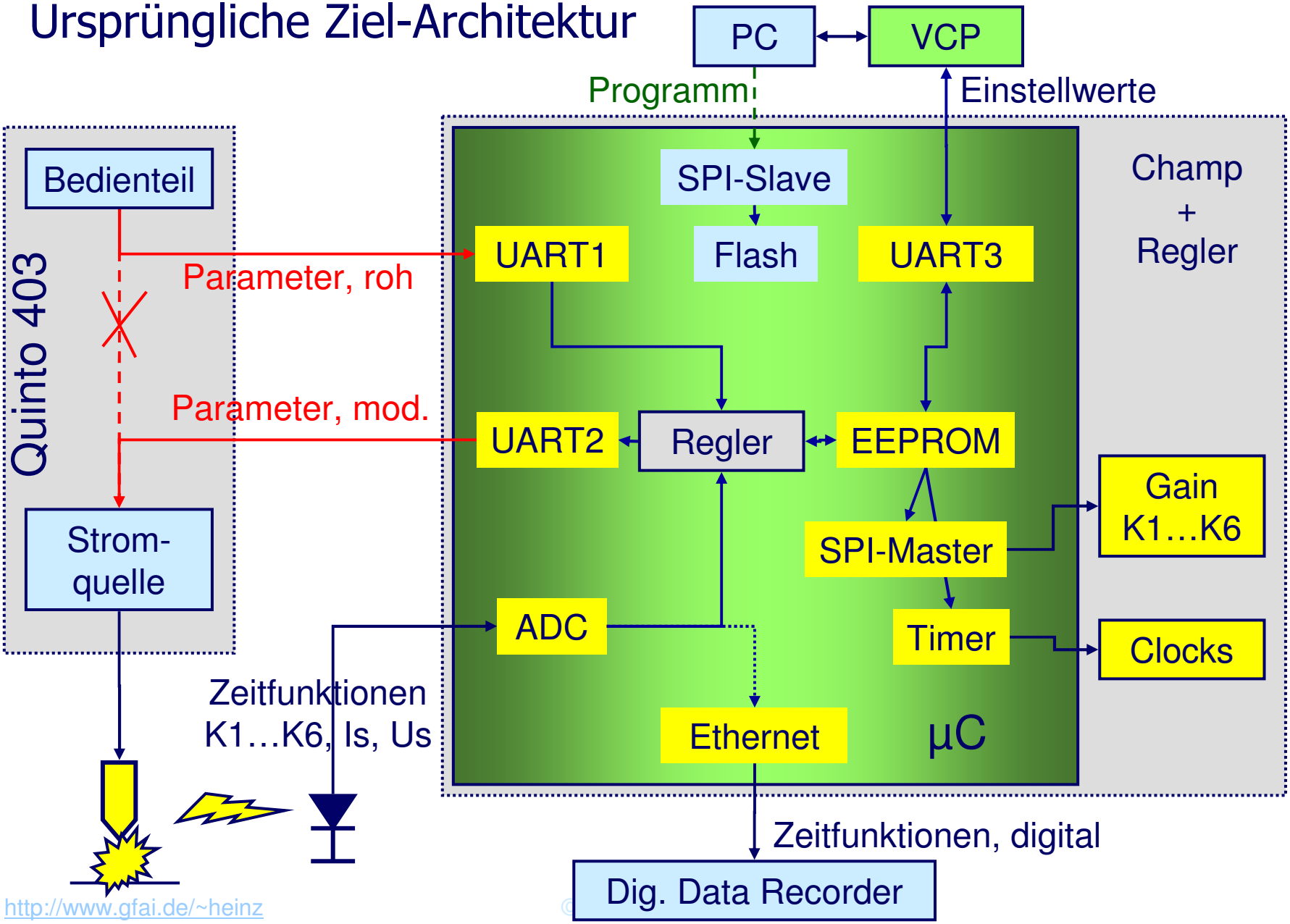
Randbedingungen

- IR-Signale zeigen sich undifferenziert -> Auflösung der ADC >8 Bit
- Prozessorleistung des Champ ist nicht üppig (max. 16 Mips*)
- UART-Schnittstellen der Quinto nur mit zwei Duplex-RS485: 4x UART
- Zahl nutzbarer Prozessoren (>5 UARTs) ist stark eingeschränkt

-> Nutzung eines Champ-Konzepts als Prozeßregler nicht befriedigend.

*Mips: Mega instructions per second

Ursprüngliche Ziel-Architektur

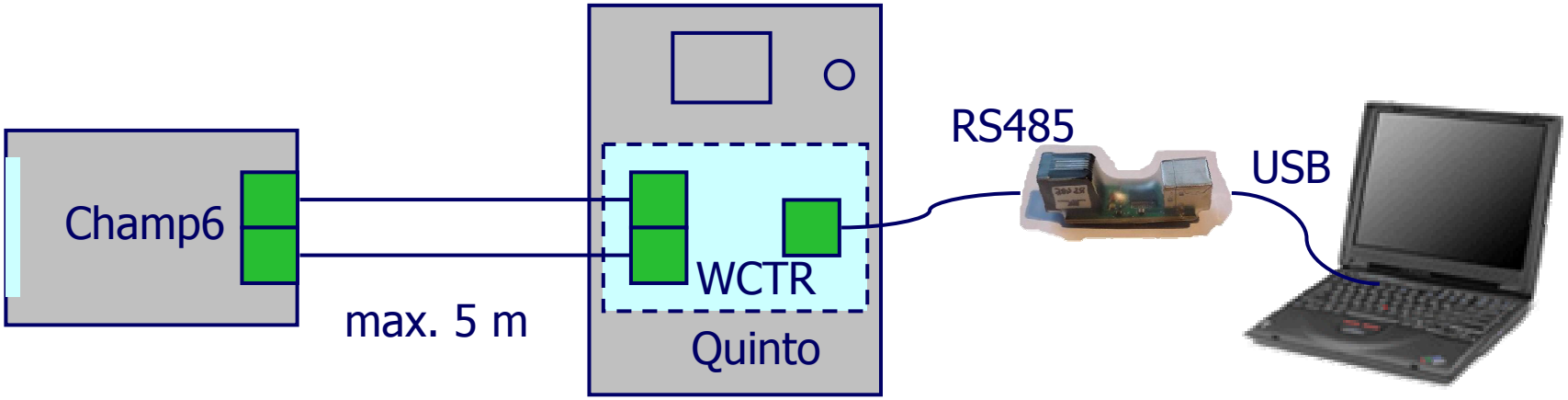


Lösungsansatz

- Räumliche Zuordnung von Spektralregler und Schweißmaschine (Kabellänge < 1 m) um UARTs direkt kommunizieren lassen zu können
- Nutzung der differentiellen Verbindungen des Champ zur Datenübertragung (bis 5 m)

Komponententeilung

- Isolierte USB-Programmierschnittstelle RS485 (USB2SERIAL) zum PC
- Photo-Vorverstärker Champ + ChampOS als Frontend
- Regler WCTR mit Hutschienengehäuse in der Schweißmaschine



Selektion der Hardware

- Anforderungen
  - >3 UARTs (2x SM, 1x PC, 1x Champ)
  - >8 ADC mit 1MHz (6x Photodioden + U + I)
  - >200 Befehle pro Sample (20µs) -> mehr als 10 Mips
- Prozessoren 8 Bit: (kürzeste Entwicklungszeiten)
  - ATmega 640/1280/2560 (je 4 UARTs, ADC real 7 Bit, <16 Mips)
  - ATxmega… (5…8 UARTs, 12…16 ADC a 10 Bit, <32 Mips)
- FPGA: 'Soft-Core', VHDL-Synthese, VHDL-Prozessor-Core
  - Evaluierung Actel Fusion, Actel Igloo-nano Designkits
  - Evaluierung eines VHDL-Core (PIC 8 Bit, 35 Befehle, <50 Mips)
    - extremer Zeitaufwand für Entwicklung
    - höchste Unflexibilität gegen Softwaremodifikationen
    - für Forschungsbelange zu unflexibel -> Abbruch
- Prozessoren 32 Bit: (hohe Leistung)
  - ARM7 CortexM3: STM32F10x (3…5 UARTs, 8x 12Bit ADC 1MS/s, 2x DAC, 72x4=288 MBps, 100 Mdips*, 100 Mwips**)

*dips: Drhystone instructions per second, includes integer operations

© heinz@g

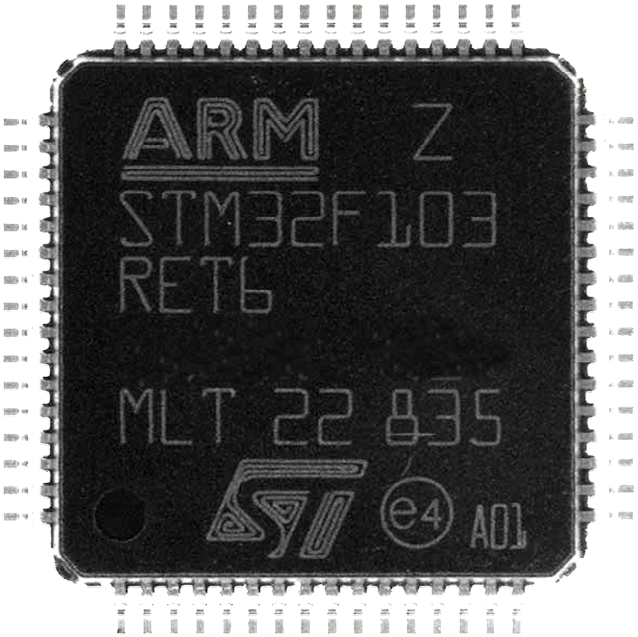
**wips: Whetstone instructions per second, includes floating point operations

Table 3. STM32F105xx and STM32F107xx family versus STM32F103xx family

STM32 device	Low-density STM32F103xx devices		Medium-density STM32F103xx devices			High-density STM32F103xx devices			STM32F105xx			STM32F107xx	
Flash size (KB)	16	32	32	64	128	256	384	512	64	128	256	128	256
RAM size (KB)	6	10	10	20	20	48	64	64	20	32	64	48	64
144 pins						5 × USARTs 4 × 16-bit timers, 2 × basic timers, 3 × SPIs, 2 × I^2Ss, 2 × I2Cs, USB, CAN, 2 × PWM timers 3 × ADCs, 1 × DAC, 1 × SDIO, FSMC (100- and 144-pin packages[1])							
100 pins				3 × USARTs 3 × 16-bit timers 2 × SPIs, 2 × I^2Cs, USB, CAN, 1 × PWM timer 2 × ADCs					5 × USARTs, 4 × 16-bit timers, 2 × basic timers, 3 × SPIs, 2 × I^2Ss, 2 × I2Cs, USB OTG FS, 2 × CANs, 1 × PWM timer, 2 × ADCs, 1 × DAC			5 × USARTs, 4 × 16-bit timers, 2 × basic timers, 2 × SPIs[2], 1 × I^2S[2], 1 × I2C[2], USB OTG FS, 2 × CANs, 1 × PWM timer, 2 × ADCs, 1 × DAC, Ethernet	
64 pins	2 × USARTs 2 × 16-bit timers 1 × SPI, 1 × I^2C, USB, CAN, 1 × PWM timer 2 × ADCs		2 × USARTs 2 × 16-bit timers 1 × SPI, 1 × I^2C, USB, CAN, 1 × PWM timer 2 × ADCs										
48 pins													
36 pins													

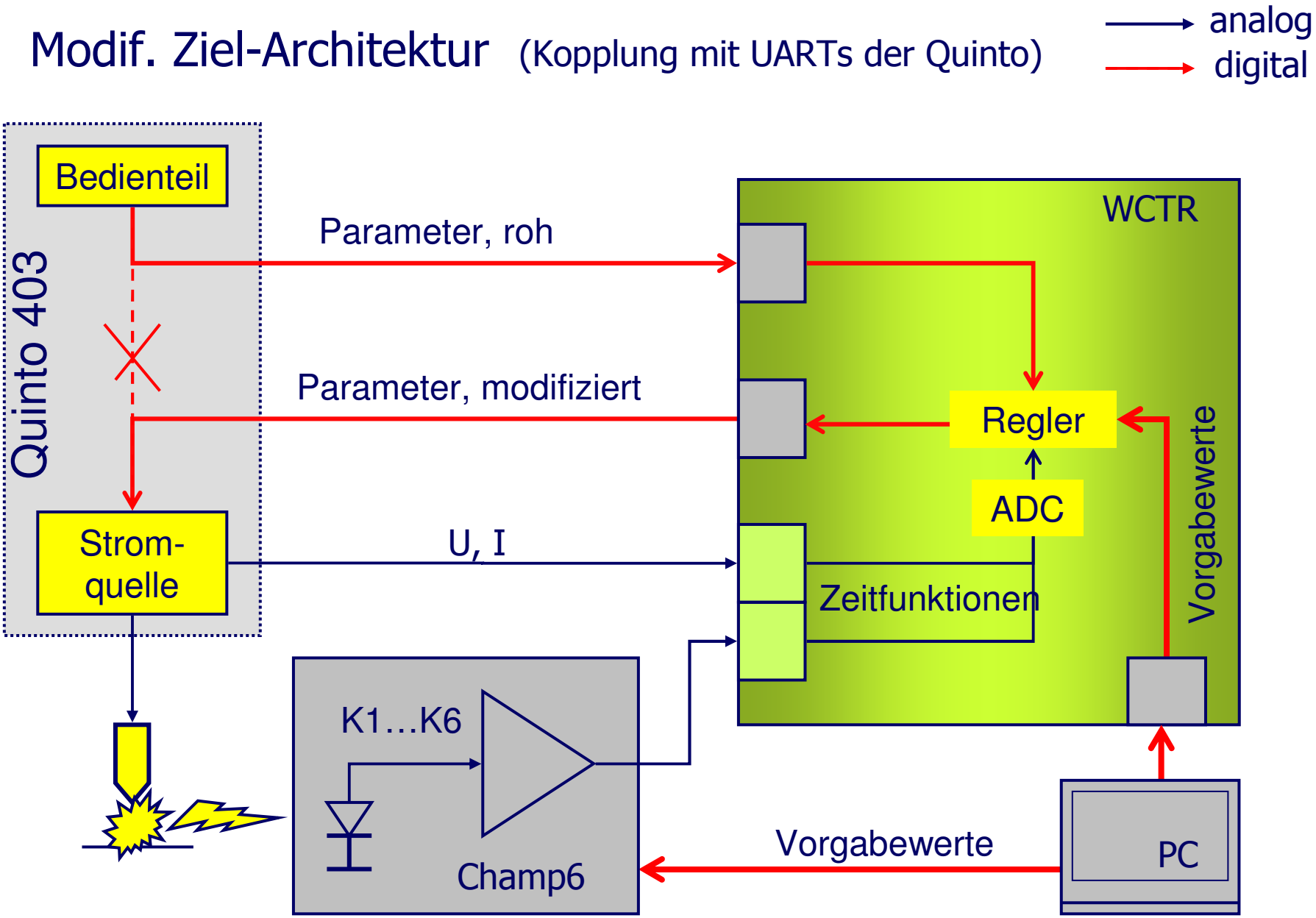
1. Ports F and G are not available in devices delivered in 100-pin packages.
2. In STM32F107xx devices, SPI2, I2S2 and I2C2 are not available when the Ethernet is being used.

ARM CortexM3: STM 32 F 103 RET6



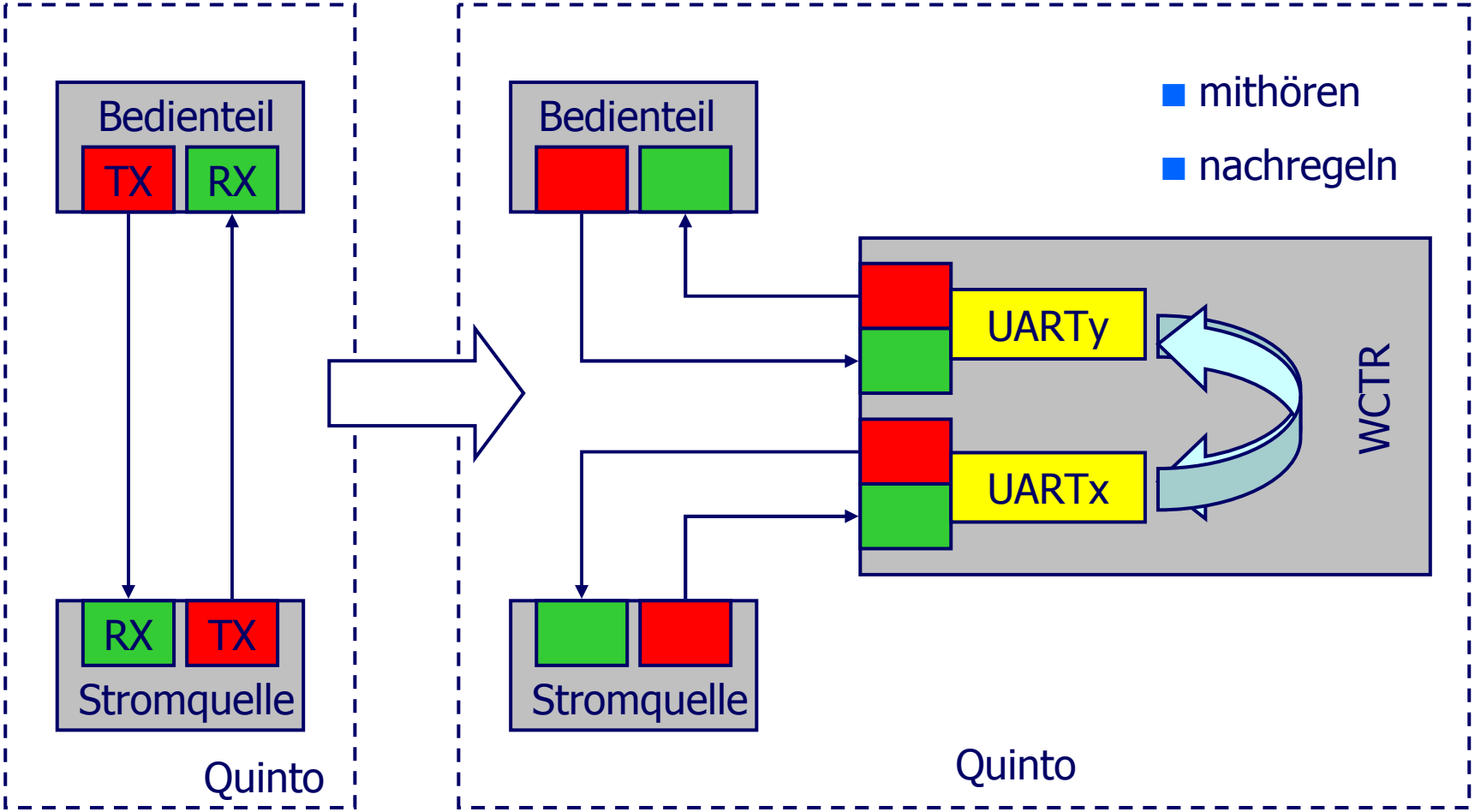
- Core: 32-bit ARM7 Cortex-M3 CPU
- 72 MHz maximum frequency
- 1.25 Mdips/MHz (Dhrystone 2.1)
- Single-cycle multiplication and division
- 512 Kbytes flash memory
- 64 Kbytes SRAM
- 2 × 12-bit A/D converters a 16 chl/ 1MSps -> 8 chl in
- 2 × 12-bit D/A converters >1MSps (1µs) -> analog out
- 13 interfaces -> 4 UARTs used
  (5 UARTs, 3 SPI, 2 I²C, SSI, SDIO, Irda, Ethernet, USB-OTG)
- Serial wire debug (SWD) & JTAG interface
- 48 pin LQFP … 64 pin LQFP … 100 pin BGA
- Problem: derzeit nur ein C-Compiler (Codesourcery G11),
  Open-Source Tools sind unbrauchbar, Profi-Tools funktionieren nicht:
  unvorhergesehener Zeitverlust etwa 6 Monate

Modif. Ziel-Architektur (Kopplung mit UARTs der Quinto)

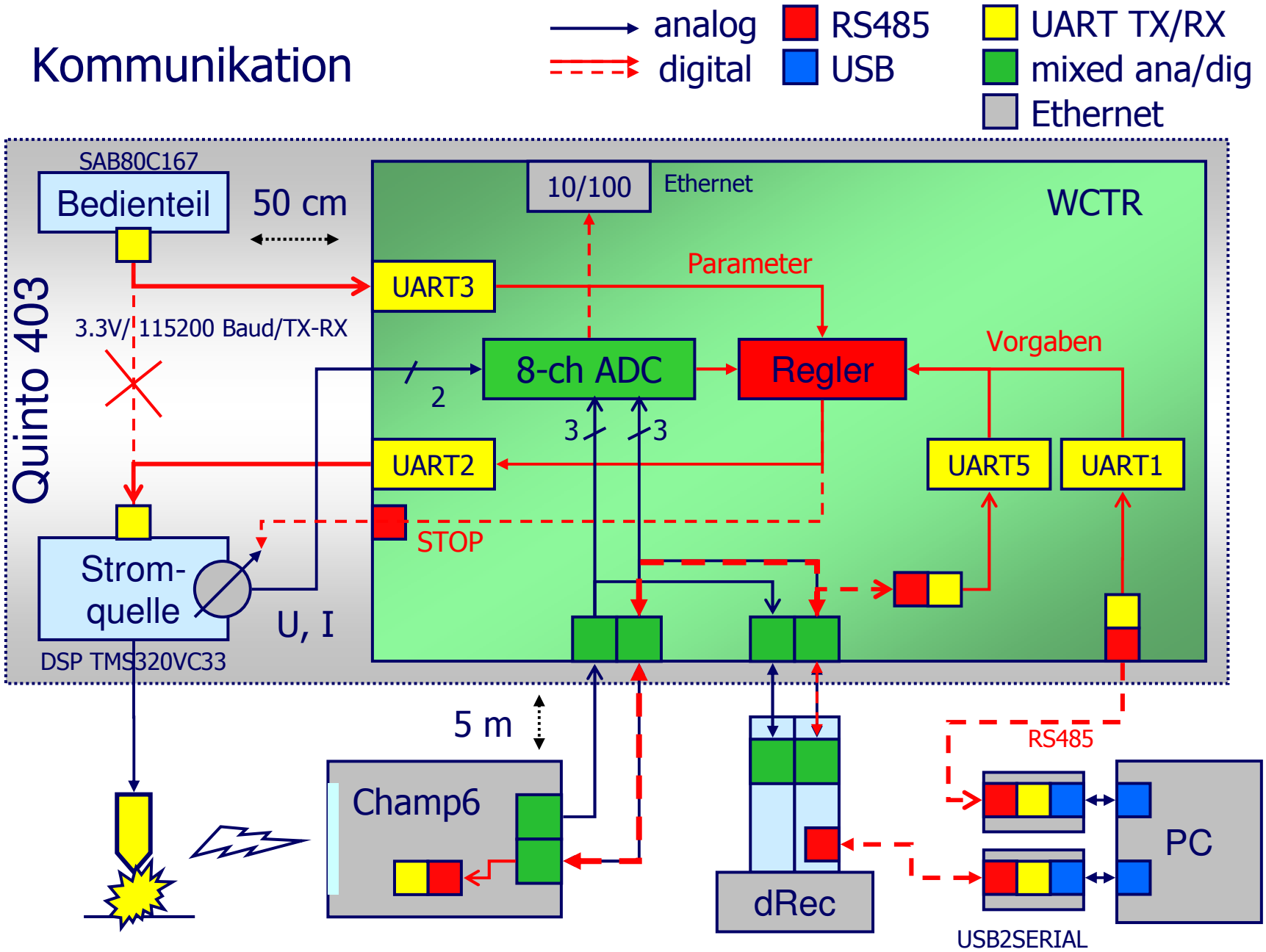


Quinto-Interaktionsansatz

- UART-FIFO (3.3V/ 115200 Baud/TX-RX mit XON/XOFF)



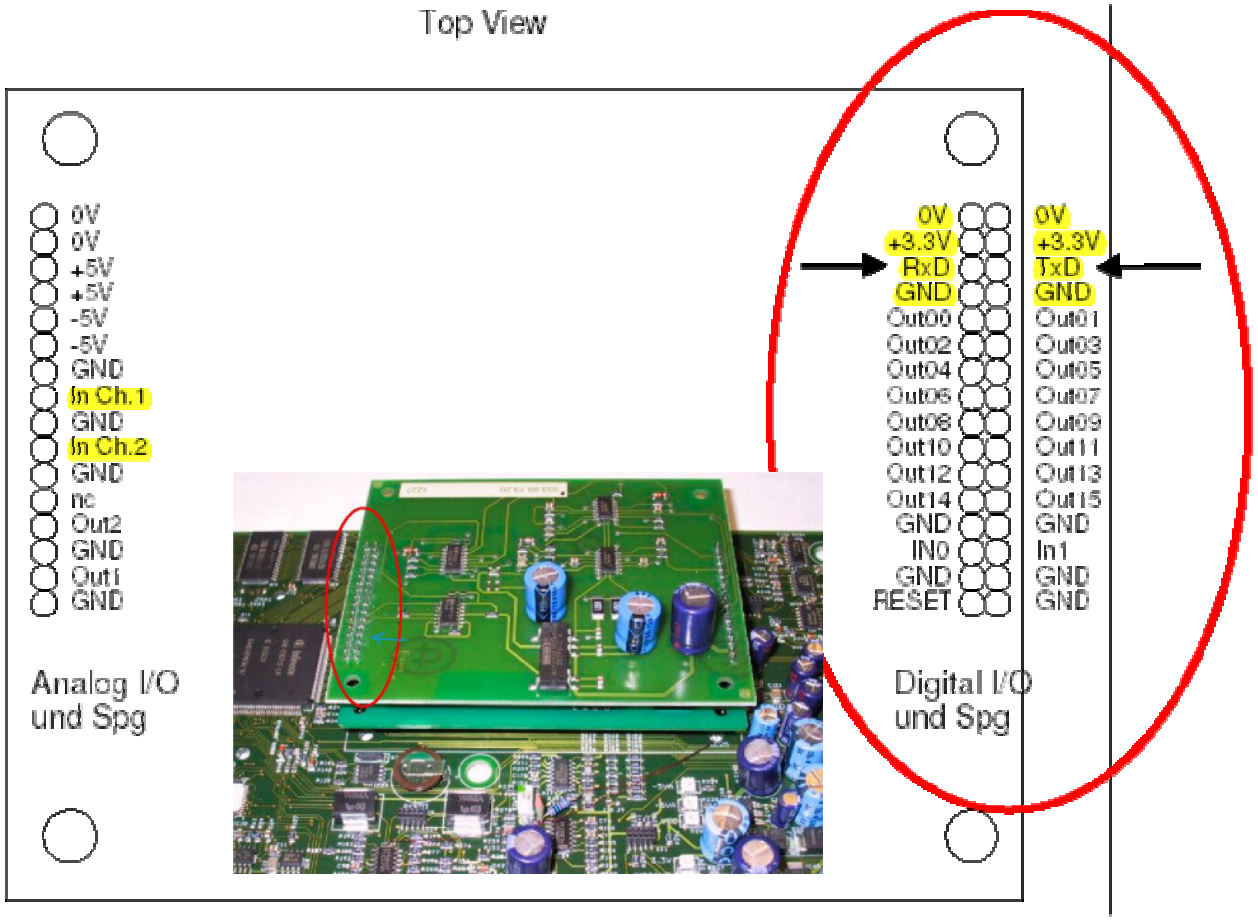
Kommunikation



© heinz@gfai.de

Anschluß Quinto

- Auftrennung der TxD/RxD-Schnittstelle zwischen Bedienteil und DSP
- Wenn WCTR nicht angeschlossen, sind Jumper zu setzen



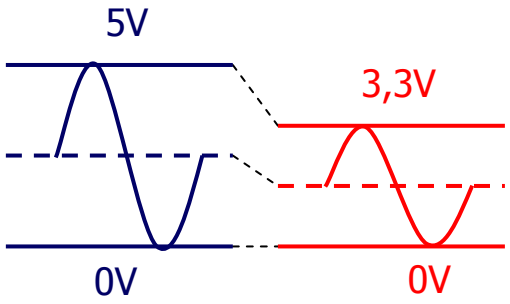
Datenaustausch mit Quinto ("Prozeßregler")

- Basis: Quinto II Prozessregler,
  System-Dokumentation Version 2.10 / 16.04.2002

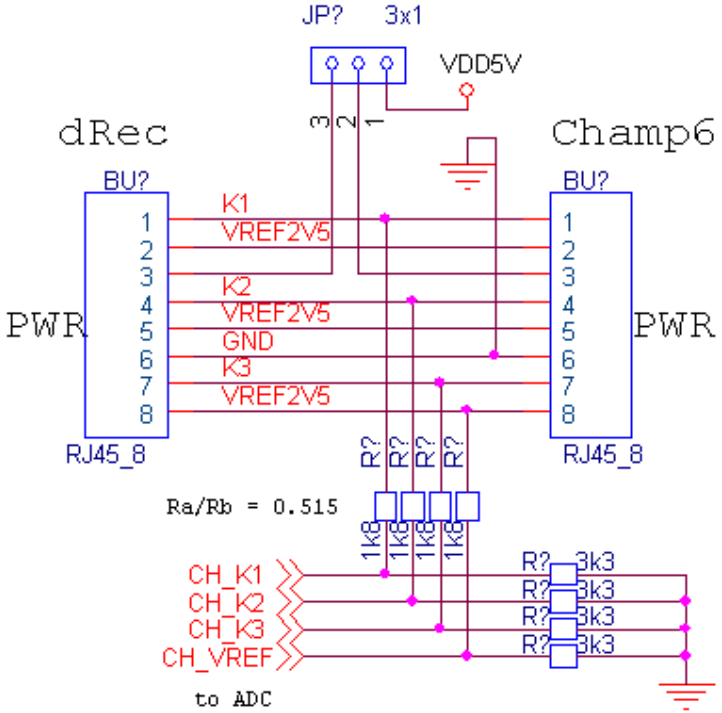
- Init über Bedienteil
- WCTR lauscht mit auf *rp*-Befehl, dieser legt den Regler für die Zündphase ("*rz*") oder die Prozessphase ("*rp*") fest
- Änderung von Prozessparametern im laufenden Prozeß über *p[<Index><Wert>...]* - Prozessparameter ändern
- Der Befehl *p* ändert einen oder mehrere Prozessparameter aus dem *rp*-Befehl. Damit kann bei laufendem Prozess in die Reglerparameter eingegriffen werden.

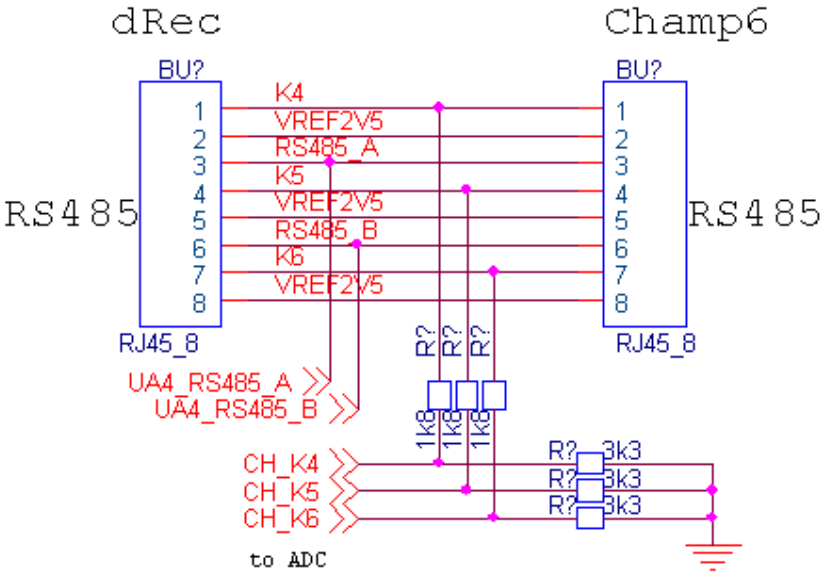
  Beispiel:
  Nachdem die Prozessphase des obigen Beispiels für E-Hand läuft, kann z.B. mit "*pb290d60|r|n*" der Parameter *SwtUDrive* auf 29 V und *ParKpDrive* auf 60 A/V umgestellt werden.

Champ-dRec Interface (Prinzip)



- Unsymmetrische Umsetzung der Analog-Pegel von 5V auf 3.3V (STM32F103-Input)





Title
CHAMP dRec ADC-Interface

© heinz@gfai.de

Anschluß Champ & dRec

Champ6

- WCTR sitzt zwischen CHAMP und dRec
- Anschlußformat des Champ bleibt unverändert bestehen
  - K1…K6 analog
  - 5V-Stromzufuhr
  - RS485
- Mini-OS "champOS" bleibt für Gain & Timing der Photoverstärker
- Kommunikation über WinXP über isol. RS485-USB mit Hyperterminal 9600-N-8-1-N
  - Atmel-Assembler/C unter Studio4/WINAVR
  - AVR-ISP-MKII über USBprog

UARTs

Data Frame Aufbau

- Startbit = 0
- Bit 0
- Bit 1
- Bit 2
- Bit 3
- Bit 4
- Bit 5
- Bit 6
- Bit 7
- Stopbit = 1
  (Pegel bleibt auf 1 stehen)

LSB zuerst!

Zeit t

Flags TXE und TC siehe Fig.241, S. 660,
STM32F10x Reference Manual, RM0008

- Zu Beginn jedes Frames wird Flag TXE = 1, Interrupt wird gegeben bei USART_CR1 / TXEIE (TXE interrupt enable)
- Frameende wird mit Flag TC = 1 signalisiert bei USART_CR1 / TCIE (TC interrupt enable)

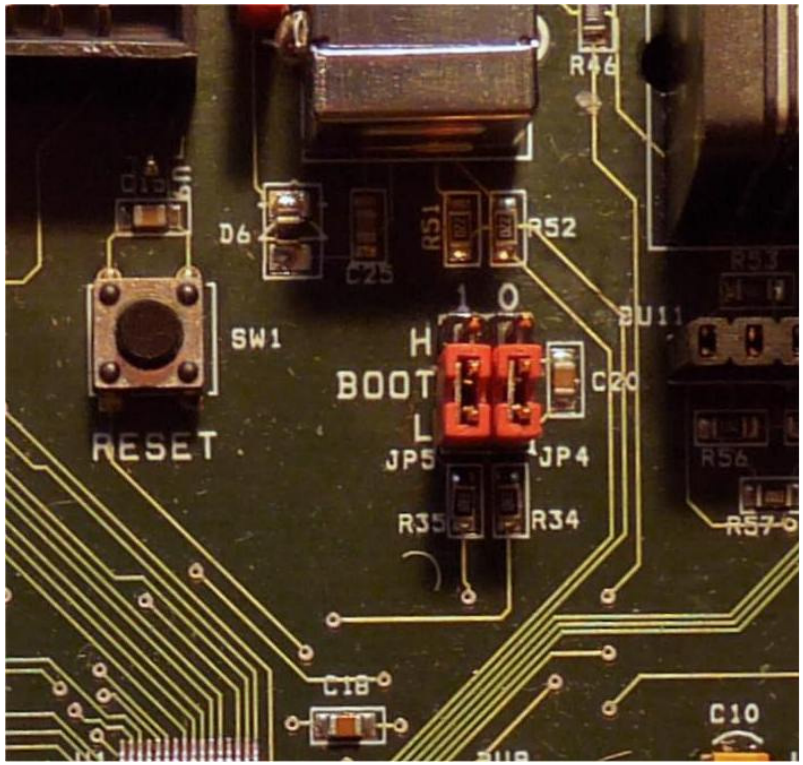
RS485:

READ = TXE & TC

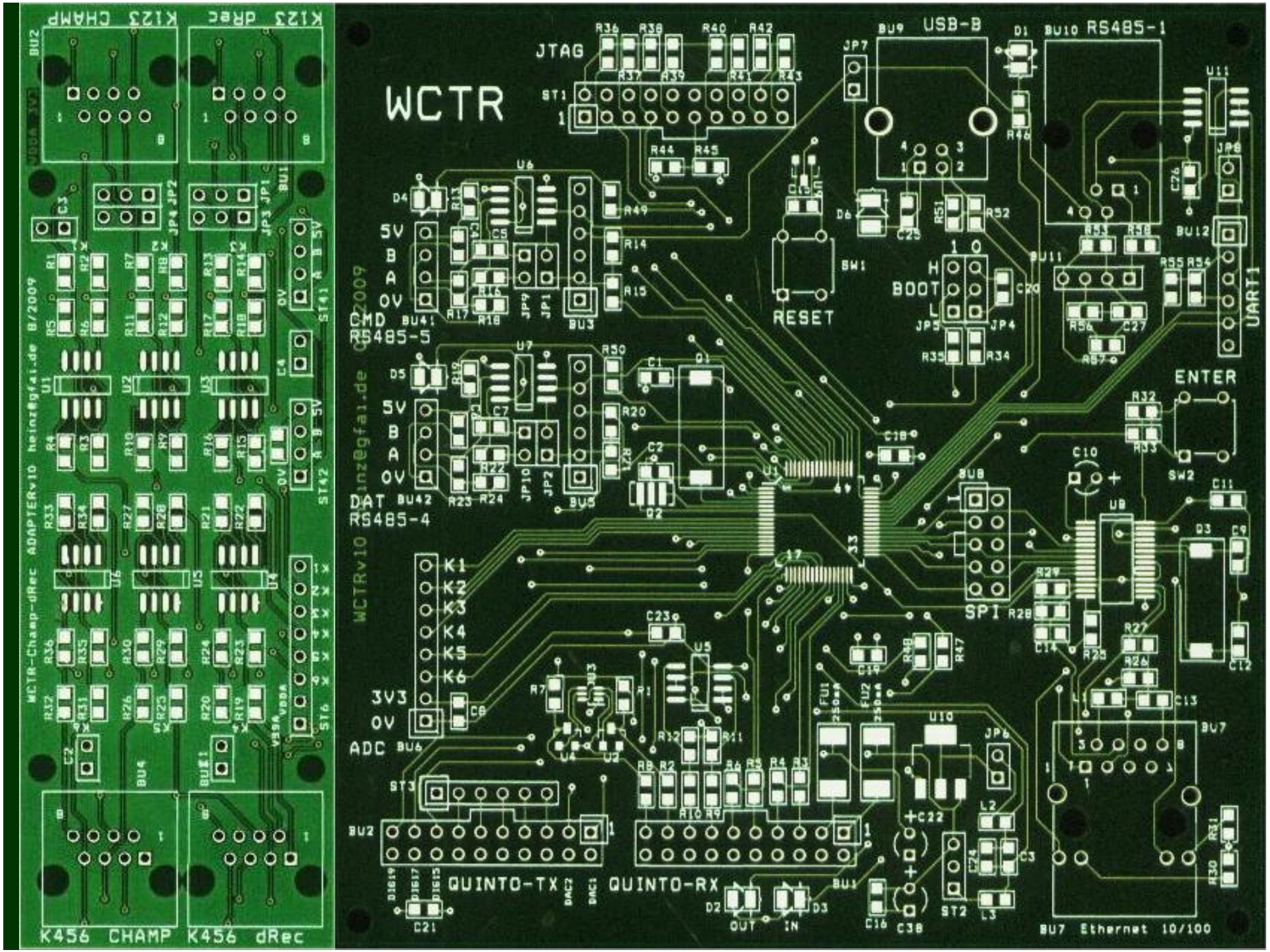
© heinz@gfai.de

Boot-Modes

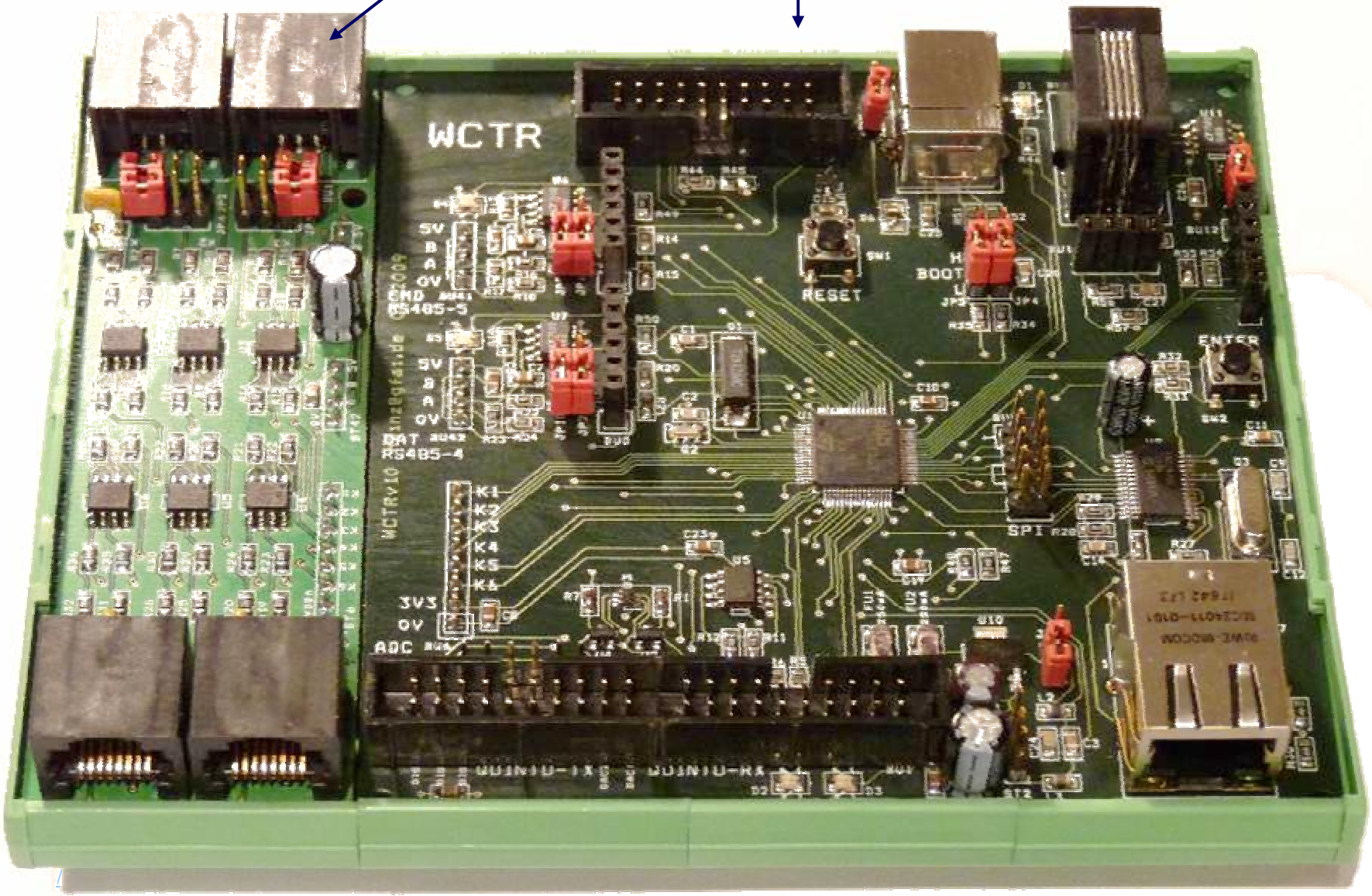
- Debug
  (with SRAM)
- Code download
  (UART1-flashloader)
- Work
  (Start from flash)



- Code-Download (Flashload) nicht über JTAG möglich,
  nach RESET geht nichts mehr
- Flash wird zerstört bei SRAM Debug über JTAG/SWD (Ursache unbekannt)

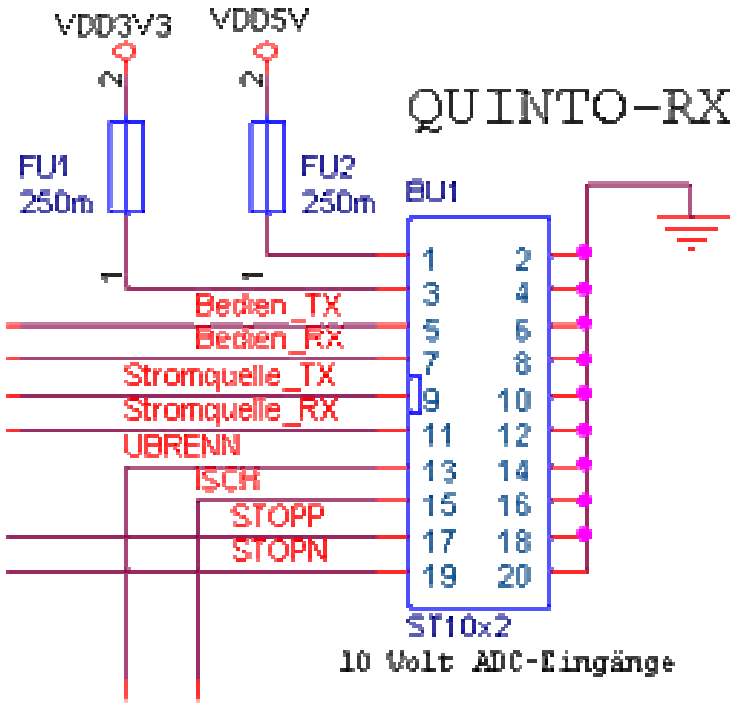

WCTR
JTAG
USB-B
RS485-1
RESET
BOOT
ENTER
UART1
CMD
RS485-5
DAT
RS485-4
K1
K2
K3
K4
K5
K6
3V3
0V
ADC
SPI
QUINTO-TX
QUINTO-RX
OUT
IN
BU7 Ethernet 10/100
WCTRv10 heinz@gfai.de 8/2009
K123 dRec
K123 CHAMP
K456 CHAMP
K456 dRec

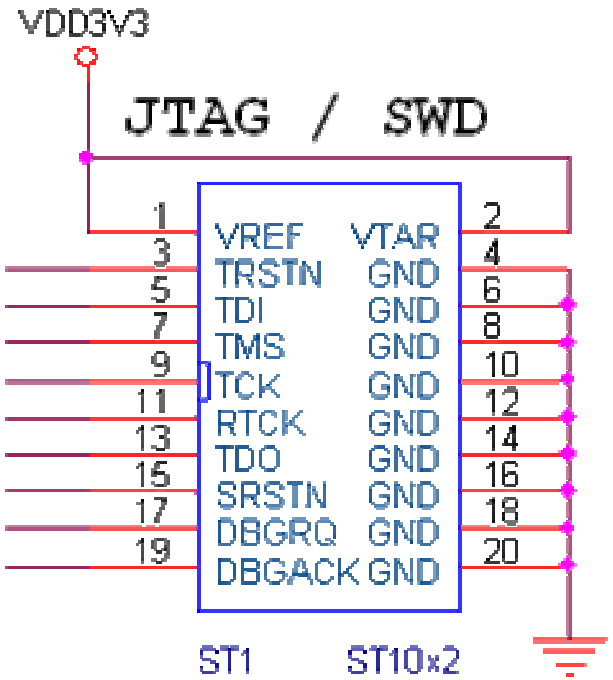
Versuchsaufbau Adapterv10 + WCTRv10

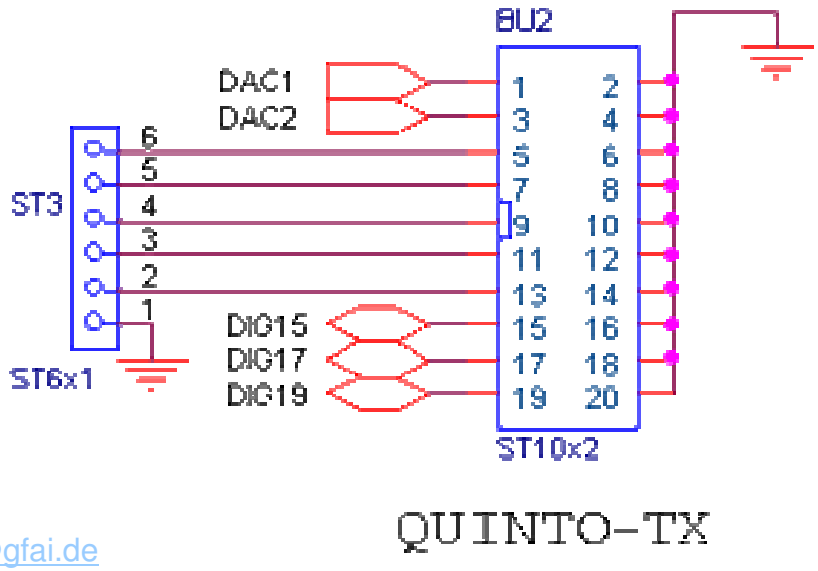


Buchsen und Steckerbelegungen

- Siehe auch Schaltpläne







© heinz@gfai.de

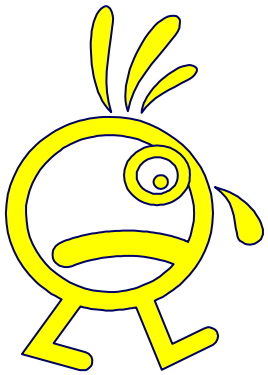
Schwerpunkte

Simulation/Emulation/Testumgebung:

- 4-Kanal Plasmasimulator (PCB AT13 mit ATtiny13, ASM)
- Emulation Bedienteil (Olimex Testboard ATmega8, ASM)
- Emulation Stromquelle (Olimex Testboard ATmega8, ASM)
- Inbetriebnahme GCC, openOCD, JTAG-Wiggler, Eclipse IDE
- Loggen des Datenverkehrs Quinto

WCTR-Module:

- ADC, DAC
- UART + RS_IO, UART-Spiegel
- Kommunikation PC
- Kommunikation CHAMP
- Kommunikation Bedienteil
- Kommunikation Stromquelle
- Control-Algorithmus
- Regler-Algorithmus



Software - Spectral welding OS "SOS"

WCTR

- Mini-OS "SOS" in C
- Kommunikation mit Bedienteil + Stromquelle über Format des Cloos "Quinto II Prozeßregler" über UART mit 3.3 V, 115200 8+1 TX/RX oder XON (11h) / XOFF (13h)
- Kommunikation mit WinXP über isol. RS485-USB mit Hyperterminal 9600-N-8-1-N
- CortexM3-Compiler: Raisonance Ride7
- JTAG-debug: Raisonance Rlink
- Code-Upload auch ST-flash bootload oder MCU-ISP
- Spezielle C-Librarys von ST nutzbar:
  - CEC (HDMI 1.3)
  - DSP (Speex Audio Coder/Decoder)
  - USB/OTG

Mini-Betriebsystem "SOS" (Spectral welding OS)

- Ohne Filesystem
- Es werden analog zum Champ interaktiv mit Kürzeln ladbare Programmteile entworfen
- Parts:
  - Reset
  - Init (Parameter, Offsetkompensation)
  - Listen (to Quinto)
  - List Parameter
  - Set Parameter
  - Work (Prozess nachregeln)

- Kommunikation mit Hyperterminal
- Hardware gestattet, daß beide über gemeinsame Schnittstelle kommunizieren

Arbeitsetappen

1. Kommunikation Bedienteil/Stromquelle aufzeichnen und an Hyperterminal weiterleiten
   - UART1: PC/Hyperterminal
   - UART2: Stromquelle
   - UART3: Bedienteil
2. ADC-Werte vom Champ aufnehmen und bearbeiten
3. Verarbeitungsalgorithmus implementieren
4. Modifizierte Einstellungen an Stromquelle weiterreichen

Referenzen

- **WCTRv10 + Adapterv10**
  - Schaltpläne
  - Bestückungslisten
  - Layout
- **Kommunikationsschnittstelle** Quinto403
- **STM32F103RET6**
  - Datenblatt
  - Simulationsdaten
  - CM3-Tools
  - Softwarebibliothek

© heinz@gfai.de

Vielen Dank für die Aufmerksamkeit!

Dr. G. Heinz
GFaI
Rudower Chausee 30
12489 Berlin
Tel. (030) 6392 -1652

© heinz@gfai.de

© Foto: Jan Gottsmann