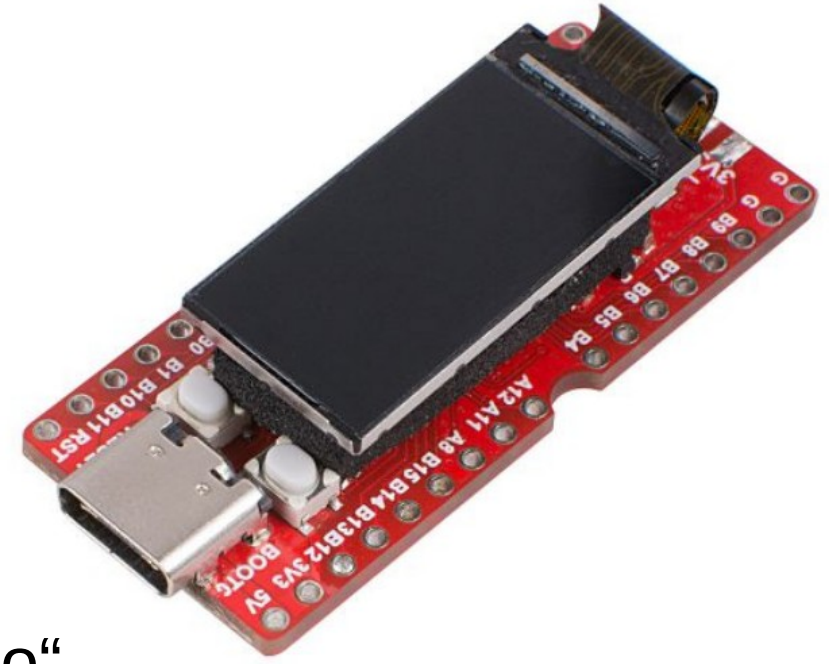
Forth auf RISC-V-Mikrocontrollern

- RISC-V
- Chips und Boards
- Mecrisp Quintus
- Projekt Feuerstein
- Demo „Blinky“
- Demo „Grafik auf Longan Nano“



RISC-V Einführung

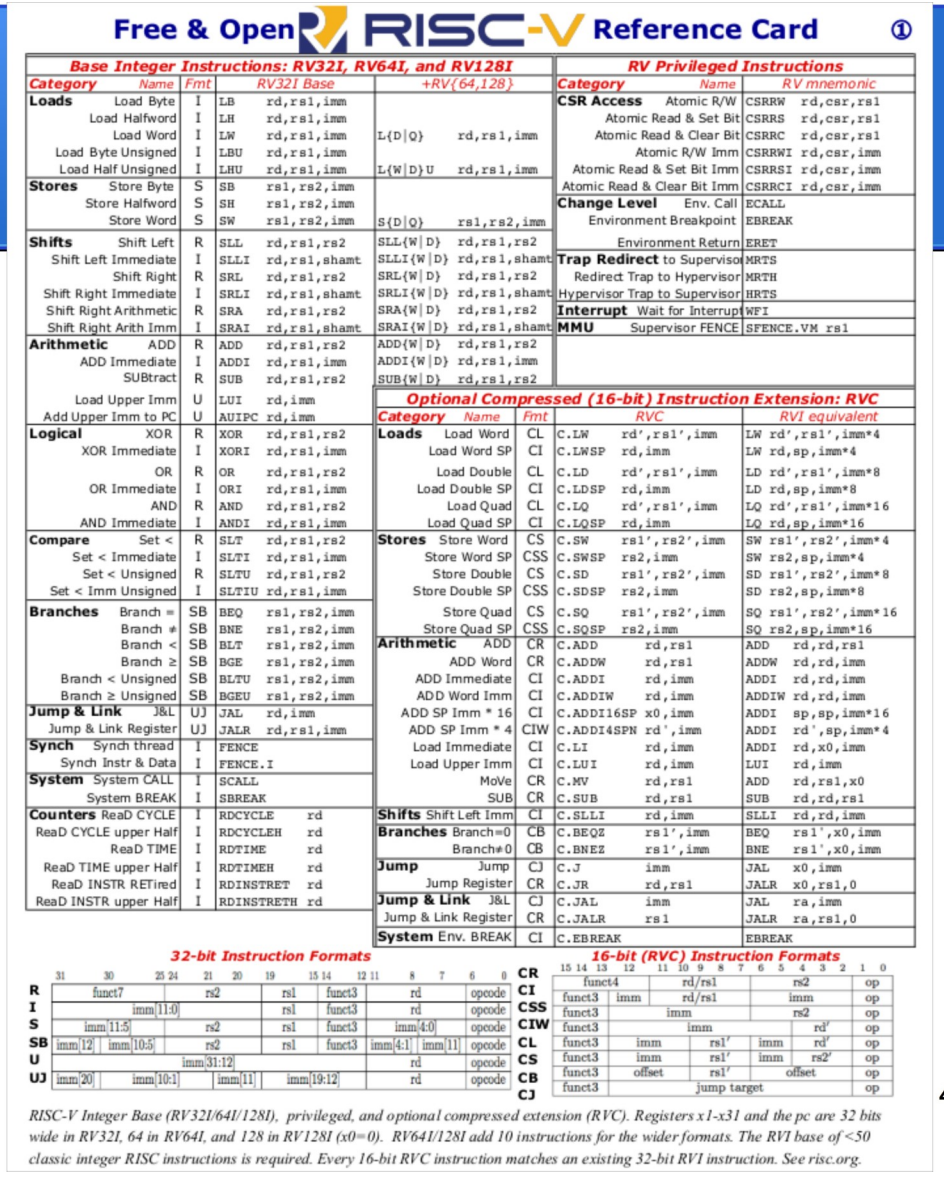
- Was es ist:
  - Eine freie und offene Befehlssatz-Architektur (ISA)
- Was es nicht automatisch ist:
  - Open Source Hardware
  - Sicher (Spectre, Meltdown)
- Ziele
  - Abdeckung vom kleinsten zum größten Prozessor
  - Stabilität
    - Basis-Befehlssatz ändert sich nicht
    - Befehlssatz kann nicht abgekündigt werden
  - Unterstützung extensiver Spezialisierungen
    - Western Digital (Permanent-Speicher-Controller)
    - Nvidia (Grafik-Chips)
  - Effizient für alle Mikroarchitekturen
  - Gut zu implementieren (ASIC, FPGA, Full-Custom-Chips)

Die RISC-V Foundation

- Gründung 2015
- Ziel: Aufbau einer offenen, kollaborativen Gemeinschaft von Software- und Hardware-Innovatoren mit der RISC-V ISA als Grundlage
- 2018 Ankündigung Zusammenarbeit mit der Linux Foundation
- 2020 „Flucht“ aus den USA in die Schweiz

RISC-V ISA
(Instruction Set Architecture)

- Modular, erweiterbar
- Beispiel: RV32IMAC
  - I -- Integer (Basis, immer vorhanden)
  - M -- Multiply and Divide
  - A -- Atomic
  - C – Compressed (16-Bit)
- F, D, V (Float, Double, Vector)
- G ist Kürzel für IMAFD
- Auch 64-Bit, 128-Bit



Free & Open RISC-V Reference Card ①

Base Integer Instructions: RV32I, RV64I, and RV128I

Category	Name	Fmt	RV32I Base	+RV{64,128}
Loads	Load Byte	I	LB rd,rs1,imm	
	Load Halfword	I	LH rd,rs1,imm	
	Load Word	I	LW rd,rs1,imm	L{D|Q} rd,rs1,imm
	Load Byte Unsigned	I	LBU rd,rs1,imm	
	Load Half Unsigned	I	LHU rd,rs1,imm	L{W|D}U rd,rs1,imm
Stores	Store Byte	S	SB rs1,rs2,imm	
	Store Halfword	S	SH rs1,rs2,imm	
	Store Word	S	SW rs1,rs2,imm	S{D|Q} rs1,rs2,imm
Shifts	Shift Left	R	SLL rd,rs1,rs2	SLL{W|D} rd,rs1,rs2
	Shift Left Immediate	I	SLLI rd,rs1,shamt	SLLI{W|D} rd,rs1,shamt
	Shift Right	R	SRL rd,rs1,rs2	SRL{W|D} rd,rs1,rs2
	Shift Right Immediate	I	SRLI rd,rs1,shamt	SRLI{W|D} rd,rs1,shamt
	Shift Right Arithmetic	R	SRA rd,rs1,rs2	SRA{W|D} rd,rs1,rs2
	Shift Right Arith Imm	I	SRAI rd,rs1,shamt	SRAI{W|D} rd,rs1,shamt
Arithmetic	ADD	R	ADD rd,rs1,rs2	ADD{W|D} rd,rs1,rs2
	ADD Immediate	I	ADDI rd,rs1,imm	ADDI{W|D} rd,rs1,imm
	SUBtract	R	SUB rd,rs1,rs2	SUB{W|D} rd,rs1,rs2
	Load Upper Imm	U	LUI rd,imm	
	Add Upper Imm to PC	U	AUIPC rd,imm	
Logical	XOR	R	XOR rd,rs1,rs2	
	XOR Immediate	I	XORI rd,rs1,imm	
	OR	R	OR rd,rs1,rs2	
	OR Immediate	I	ORI rd,rs1,imm	
	AND	R	AND rd,rs1,rs2	
	AND Immediate	I	ANDI rd,rs1,imm	
Compare	Set <	R	SLT rd,rs1,rs2	
	Set < Immediate	I	SLTI rd,rs1,imm	
	Set < Unsigned	R	SLTU rd,rs1,rs2	
	Set < Imm Unsigned	I	SLTIU rd,rs1,imm	
Branches	Branch =	SB	BEQ rs1,rs2,imm	
	Branch ≠	SB	BNE rs1,rs2,imm	
	Branch <	SB	BLT rs1,rs2,imm	
	Branch ≥	SB	BGE rs1,rs2,imm	
	Branch < Unsigned	SB	BLTU rs1,rs2,imm	
	Branch ≥ Unsigned	SB	BGEU rs1,rs2,imm	
Jump & Link	J&L	UJ	JAL rd,imm	
	Jump & Link Register	UJ	JALR rd,rs1,imm	
Synch	Synch thread	I	FENCE	
	Synch Instr & Data	I	FENCE.I	
System	System CALL	I	SCALL	
	System BREAK	I	SBREAK	
Counters	ReaD CYCLE	I	RDCYCLE rd	
	ReaD CYCLE upper Half	I	RDCYCLEH rd	
	ReaD TIME	I	RDTIME rd	
	ReaD TIME upper Half	I	RDTIMEH rd	
	ReaD INSTR RETired	I	RDINSTRET rd	
	ReaD INSTR upper Half	I	RDINSTRETH rd	

RV Privileged Instructions

Category	Name	RV mnemonic
CSR Access	Atomic R/W	CSRRW rd,csr,rs1
	Atomic Read & Set Bit	CSRRS rd,csr,rs1
	Atomic Read & Clear Bit	CSRRC rd,csr,rs1
	Atomic R/W Imm	CSRRWI rd,csr,imm
	Atomic Read & Set Bit Imm	CSRRSI rd,csr,imm
	Atomic Read & Clear Bit Imm	CSRRCI rd,csr,imm
Change Level	Env. Call	ECALL
	Environment Breakpoint	EBREAK
	Environment Return	ERET
Trap Redirect	to Supervisor	MRTS
	Redirect Trap to Hypervisor	MRTH
	Hypervisor Trap to Supervisor	HRTS
Interrupt	Wait for Interrupt	WFI
MMU	Supervisor FENCE	SFENCE.VM rs1

Optional Compressed (16-bit) Instruction Extension: RVC

Category	Name	Fmt	RVC	RVI equivalent
Loads	Load Word	CL	C.LW rd',rs1',imm	LW rd',rs1',imm*4
	Load Word SP	CI	C.LWSP rd,imm	LW rd,sp,imm*4
	Load Double	CL	C.LD rd',rs1',imm	LD rd',rs1',imm*8
	Load Double SP	CI	C.LDSP rd,imm	LD rd,sp,imm*8
	Load Quad	CL	C.LQ rd',rs1',imm	LQ rd',rs1',imm*16
	Load Quad SP	CI	C.LQSP rd,imm	LQ rd,sp,imm*16
Stores	Store Word	CS	C.SW rs1',rs2',imm	SW rs1',rs2',imm*4
	Store Word SP	CSS	C.SWSP rs2,imm	SW rs2,sp,imm*4
	Store Double	CS	C.SD rs1',rs2',imm	SD rs1',rs2',imm*8
	Store Double SP	CSS	C.SDSP rs2,imm	SD rs2,sp,imm*8
	Store Quad	CS	C.SQ rs1',rs2',imm	SQ rs1',rs2',imm*16
	Store Quad SP	CSS	C.SQSP rs2,imm	SQ rs2,sp,imm*16
Arithmetic	ADD	CR	C.ADD rd,rs1	ADD rd,rd,rs1
	ADD Word	CR	C.ADDW rd,rs1	ADDW rd,rd,imm
	ADD Immediate	CI	C.ADDI rd,imm	ADDI rd,rd,imm
	ADD Word Imm	CI	C.ADDIW rd,imm	ADDIW rd,rd,imm
	ADD SP Imm * 16	CI	C.ADDI16SP x0,imm	ADDI sp,sp,imm*16
	ADD SP Imm * 4	CIW	C.ADDI4SPN rd',imm	ADDI rd',sp,imm*4
	Load Immediate	CI	C.LI rd,imm	ADDI rd,x0,imm
	Load Upper Imm	CI	C.LUI rd,imm	LUI rd,imm
	MoVe	CR	C.MV rd,rs1	ADD rd,rs1,x0
	SUB	CR	C.SUB rd,rs1	SUB rd,rd,rs1
Shifts	Shift Left Imm	CI	C.SLLI rd,imm	SLLI rd,rd,imm
Branches	Branch=0	CB	C.BEQZ rs1',imm	BEQ rs1',x0,imm
	Branch≠0	CB	C.BNEZ rs1',imm	BNE rs1',x0,imm
Jump	Jump	CJ	C.J imm	JAL x0,imm
	Jump Register	CR	C.JR rd,rs1	JALR x0,rs1,0
Jump & Link	J&L	CJ	C.JAL imm	JAL ra,imm
	Jump & Link Register	CR	C.JALR rs1	JALR ra,rs1,0
System	Env. BREAK	CI	C.EBREAK	EBREAK

32-bit Instruction Formats

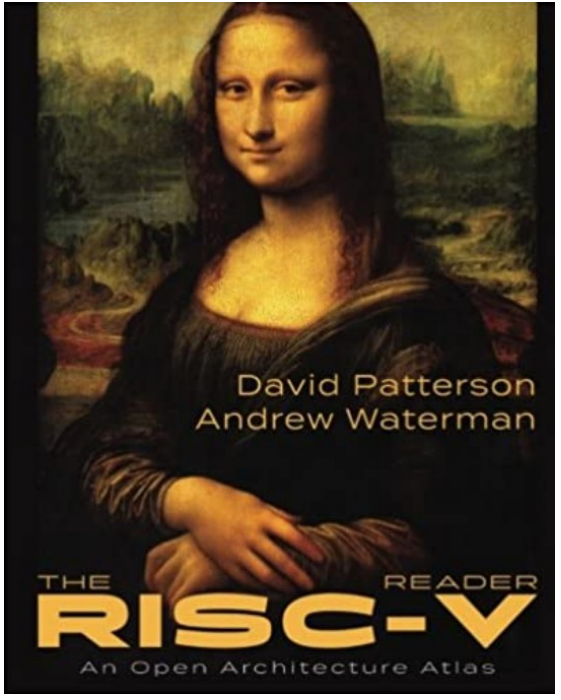
16-bit (RVC) Instruction Formats

RISC-V Integer Base (RV32I/64I/128I), privileged, and optional compressed extension (RVC). Registers x1-x31 and the pc are 32 bits wide in RV32I, 64 in RV64I, and 128 in RV128I (x0=0). RV64I/128I add 10 instructions for the wider formats. The RVI base of <50 classic integer RISC instructions is required. Every 16-bit RVC instruction matches an existing 32-bit RVI instruction. See risc.org.

RISC-V Greencard
Seite 1 von 2

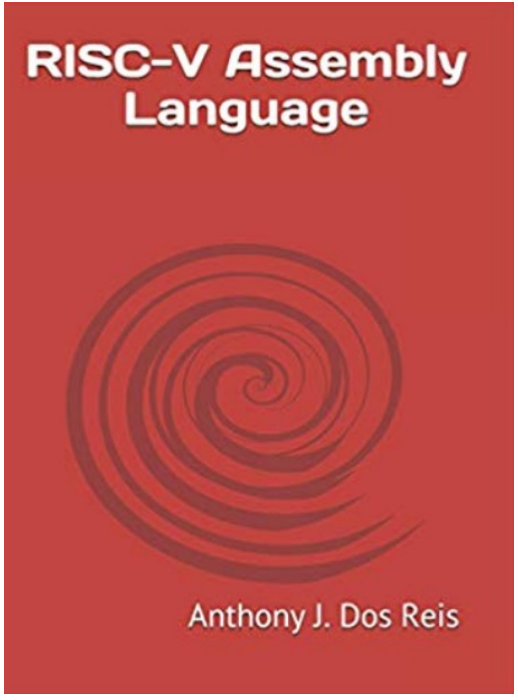
Buchtipp

- The RISC-V Reader: An Open Architecture Atlas
- - David Pattersen
  - Andrew Waterman
- 200 Seiten
- November 2017
- Amazon, 18,-- EUR



Buchtipp

- RISC-V Assembly Language
- Anthony J. Dos Reis
- 155 Seiten
- August 2019
- Amazon, 19,-- EUR



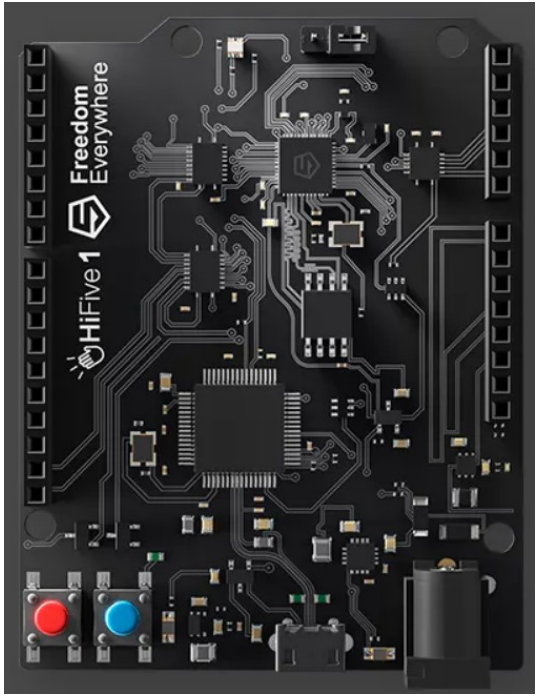
Chip: SiFive Freedom Everywhere 310 (FE310)

- Erster RISC-V Mikrocontroller auf dem Markt
- Kein Analog-Digital-Wandler
- hat kein Flash auf dem Chip

- Spezifikation
  - RV32IMAC, 320 MHz
  - 16 kB Instruction Cache
  - 16 kB Data SRAM
  - 3x PWM
  - SPI, UART, I²C
  - QSPI Flash Interface

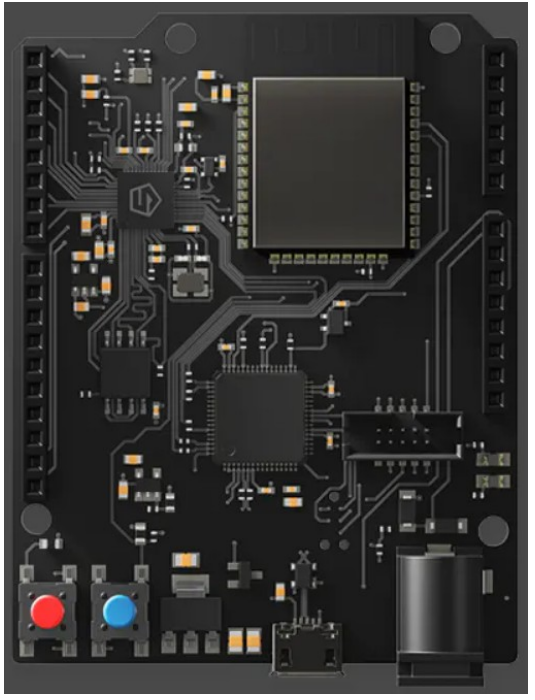
Board: SiFive HiFive1

- Erstes Board für Hobbyisten (2016)
- 80,-- EUR
- keine Analogeingänge
- Sperrig für Forth-Programmierung
- Im Forth e.V. - Verleih



Board: SiFive HiFive1RevB

- Wie HiFive1,
  aber mit Funkmodul



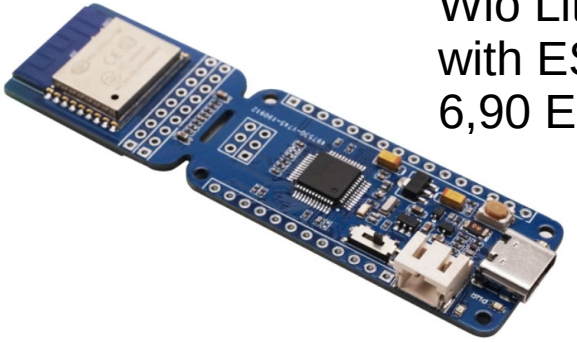
Chip: GigaDevice GD32VF103

- Erster „brauchbarer“ Mikrocontroller mit RISC-V ISA
- GigaDevice hatte schon mit dem GD32F103 einen „gepimpten“ Clone des STM32F103 mit ARM Cortex M3 (Bluepill) im Programm
- Nun beim GD32VF103 den ARM-Kern durch RISC-V ersetzt

- Spezifikation
  - RV32IMAC, 108 MHz
  - 128 kB Flash, Zero Waitstates!
  - 32 kB RAM
  - 7 Timer, 1 RTC, 2 Watchdogs
  - 3 USART, 2 I²C, 3 SPI
  - 2 CAN, 1 USB OTG FS
  - 2 ADC (je 10 Ch.), 2 DAC

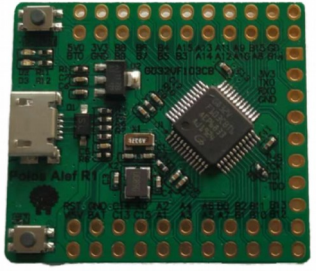
Boards mit GD32VF103

Sipeed
Longan Nano
4,60 EUR



Sipeed
Wio Lite RISC-V
with ESP8266
6,90 EUR

SeeedStudio
GD32 RISC-V Dev Board
6,50 EUR



Analoglamb
Polos GD32VF103
Alef Board
2,99 EUR

Mecrisp Quintus -- Forth für RISC-V

- Erste Version erschien März 2018
- Aktuell V0.27a vom 29.03.2020
- Wird aktiv entwickelt
- Status: experimentell, bereits sehr stabil
- Highlights:
  - unterstützt „Compressed“ (16-Bit-Befehle)
  - Assembler, Disassembler
  - Fixkomma-Arithmetik
  - bereits einfache Optimierung (Konstantenfaltung)
  - kompiliert ins Flash oder ins RAM

Projekt „Feuerstein“

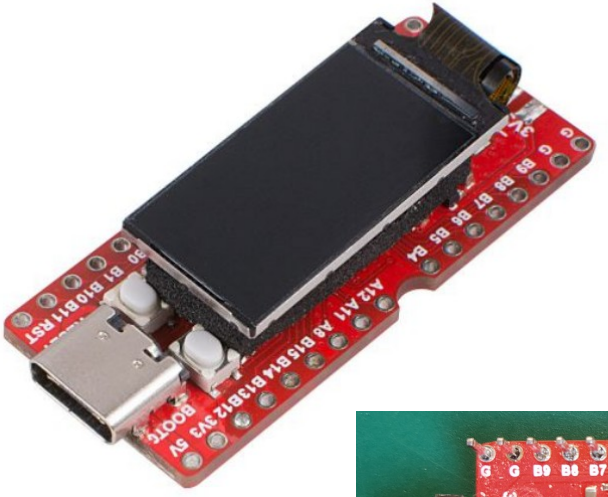
- Das Problem:
  - Ein Forth-Novize: „Wow, Forth auf Mikrocontrollern soll ja cool sein. Das probiere ich mal aus.“ :-)
    Doch dann:
    - Wieso muss ich an unzähligen Stellen suchen?
      Ich will doch erst einmal nur ein Lämpchen blinken lassen. :-(
    - Wieso bekomme ich das Forth nicht auf den Chip. :-(
    - Wieso steigt da jetzt Rauch auf? :-(
    - Wieso funktioniert das Terminal nicht? :-(
    - Wieso läuft das Beispielprogramm nicht? :-(
    - Was meinen die mit „Der Quelltext ist die Doku“? Oder:
      Wieso ist die Doku total veraltet? :-(
    - Wieso kann ich keinen fragen? :-(
  - Kannste knicken, ich bin raus. :-(

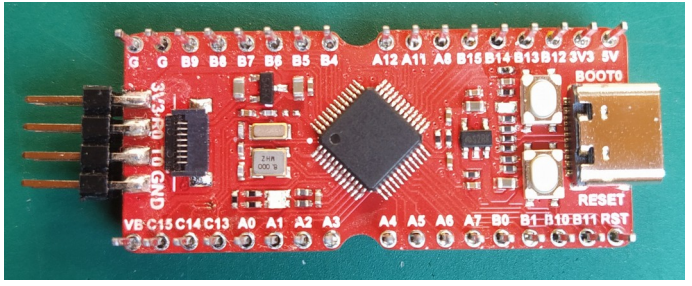
Projekt „Feuerstein“

- Angestrebte Lösung:
  - Es gibt eine Webseite
  - Dort finde ich alles, was ich brauche:
    - Hardware
    - Software
    - Dokumentation
  - Ich kann ein Rundum-Sorglos-Paket bestellen:
    - Teilbestückte Platinen, Bauteile zum Komplettieren
    - Aktuelle Dokumentation in Papier- und elektronischer Form
    - Full-Support: Mache ich die Platine kaputt, bekomme ich eine neue. Ich habe einen Ansprechpartner.
  - Ich kann auch alles selbst machen.
    Alles ist Open-Source und Open-Hardware

Hardware: Sipeed Longan Nano

- GD32VF103CBT6
  128 kB Flash, 32 kB RAM, 48 Pins
- RGB-LED
- 160x80 Pixel TFT (IPS)
- SD-Kartenhalter
- USB-C-Buchse

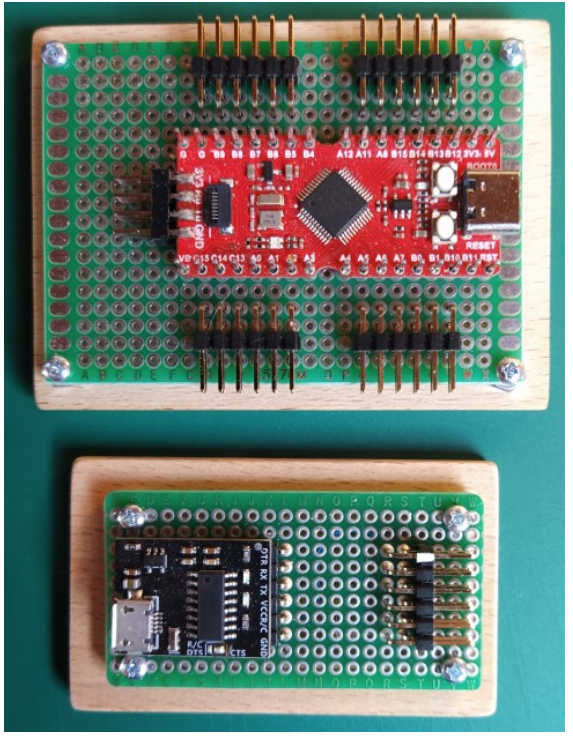




Hardware: Eigenentwicklung

- Hauptplatine
  - RISC-V GD32VF103
  - Spannungsregler
  - Quarze 8 MHz und 32,768 kHz
  - 8 MB SPI-Flash
  - LEDs, Taster
  - Pufferzelle für Echtzeituhr (Batterie oder Supercap)
  - USB-Buchse, Pmod Steckerleisten (Digilent)
  - Schaltkreis zum Vermessen von Bauteilen
- USB-Seriell-Wandler (Pmod)
  - galvanische Trennung
  - Chip CP2102N
  - USB-Kabel (Micro-USB oder USB-C)
- Ersatzteile zum Experimentieren

Prototyp



Software

- Auf dem Chip
  - Mecrisp Quintus
  - Forth-Beispieldateien (SPI-Flash)
  - Hilfesystem (SPI-Flash)
  - Interaktives Lernsystem
  - History-Buffer, Auto-Vervollständigung
  - VIS-System (Vocs, Items, Sticky words)
  - Hardwaretreiber
    - USB, USB-OTG, CAN
    - SPI, I2C, UART, FIFO
    - Interrupts, Timer, Echtzeituhr
    - SD-Karte, SPI-Flash
    - ADC, DAC
- Auf dem PC
  - e4thcom (Linux)
    (Manfreds Terminalprogramm)
  - picocom (Linux)
  - minicom (Linux)
  - Teraterm (Windows)
  - Lieblingseditor

Dokumentation

- Schnelleinstieg (erste Schritte)
  - Platine fertigstellen
  - Platine in Betrieb nehmen (Blinkenlights per Autostart)
  - Einrichten einer Entwicklungsumgebung
  - Blinkprogramm starten und verändern
- Mecrisp Quintus
  - Tutorial
  - Benutzerhandbuch
  - Referenzhandbuch
- Hardware Benutzerhandbuch
  - Belegung der Anschlüsse
  - Schaltplan
  - Beschreibung der Bausteine
  - Funktionsweise
- Datenblätter / Benutzerhandbücher der verwendeten Bausteine
  - GD32VF103
  - SPI-Flash
  - CP2102N

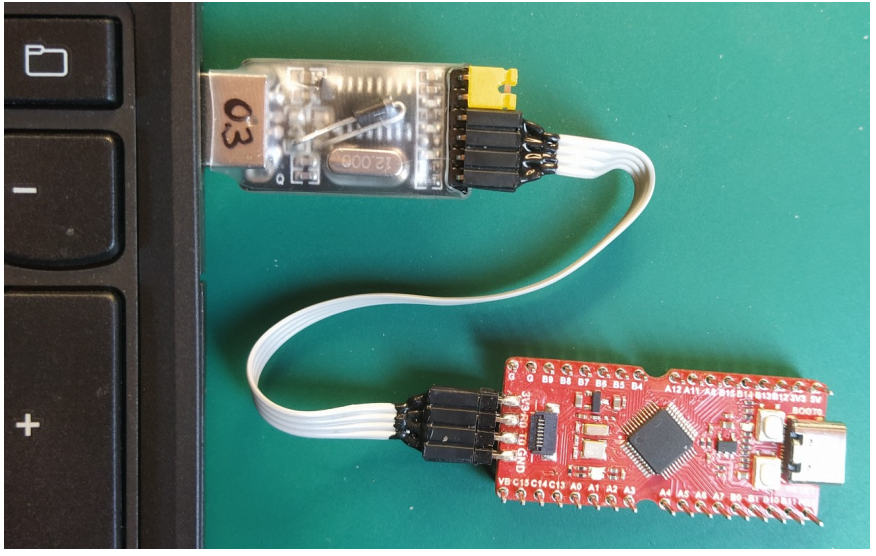
Habt ihr Fragen?

Demo „Blinky“

wolfgang@E560: ~/Tagung2020/RISC-V-Vortrag

Datei Bearbeiten Ansicht Suchen Terminal Hilfe

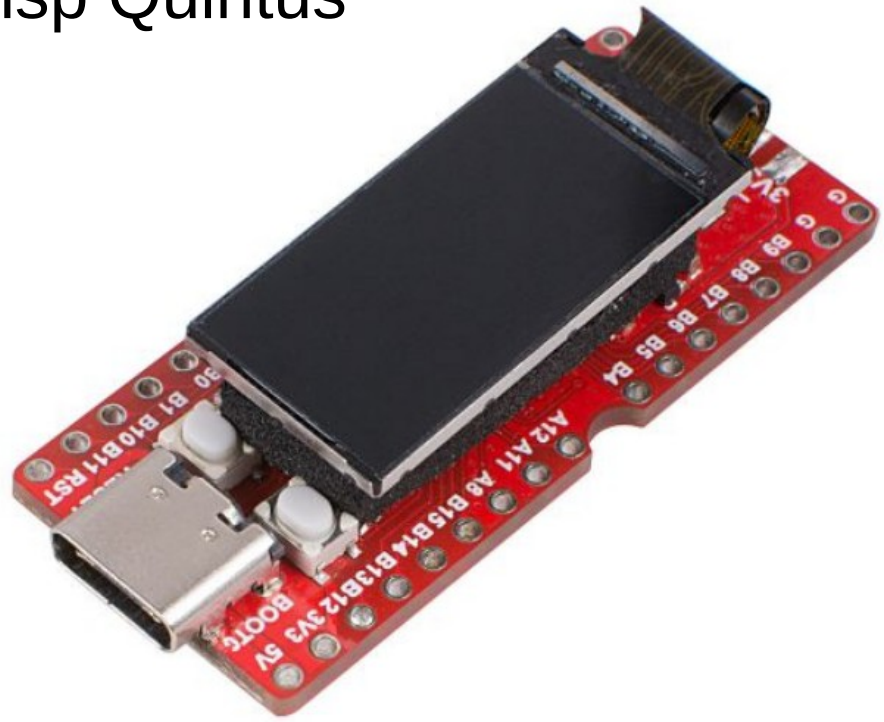
```
Mecrisp-Quintus 0.27 for RISC-V 32 IMC on GD32VF103CB by Matthias Koch

Have a nice day !
list
--- Mecrisp-Quintus 0.27 --- hflash! flash! flashpageerase eraseflash eraseflashfrom sp@ sp! rp@ rp! dup drop ?dup swap nip over
 tuck rot -rot pick depth rdepth >r r> r@ rdrop rpick 0= 0<> 0< >= <= < > u>= u<= u< u> <> = min max umin umax + - slt sltu 1- 1
+ 2- 2+ cell+ negate not shr shl 2* cells 2/ abs even base binary decimal hex eint? eint dint wfi mcause mepc cycles64 cycles un
handled fault irq-fault irq-collection hook-emit hook-key hook-emit? hook-key? hook-pause nop emit key emit? key? pause serial-e
mit serial-key serial-emit? serial-key? reset accept tib >in current-source setsource source query compare cr bl space spaces [c
har] char ( \ ." c" s" (.") (c") (s") count ctype type skipstring risc-v hex. .rs words h.s .s u.s token parse does> <builds cre
ate registerliteral, literal, call, inline, skipdefinition string, [’] ' postpone exit recurse state ] [ : ; execute immediate i
nline compileonly 0-foldable 1-foldable 2-foldable 3-foldable 4-foldable 5-foldable 6-foldable 7-foldable constant 2constant smu
dge setflags aligned here flashvar-here addrinflash? addrinram? align h, , allot forgetram compiletoram? compiletoram compiletof
lash (create) variable 2variable nvariable buffer: dictionarystart dictionarynext hook-find find (find) k j i leave (do) unloop
+loop loop do ?do case ?of of endof endcase uj-encoding? sb-encoding? cj-encoding? then else if ahead repeat while until again b
egin and bic or xor clz ror rol arshift rshift lshift not invert true false evaluate interpret hook-quit quit digit number .digi
t hold hold< sign #> f#S f# #S # <# f. f.n ud. d. u. . * um* m* / mod u/mod /mod 2dup 2drop 2swap 2nip 2over 2tuck 2rot 2-rot 2>
r 2r> 2r@ 2rdrop d2/ d2* dshr dshl dabs dnegate d- d+ s>d ud* udm* */ */mod u*/ u*/mod um/mod m/mod ud/mod d/mod d/ f* f/ 2! 2@
du< du> d< d> d0< d0= d<> d= move fill @ ! +! h@ h@signed h! h+! c@ c@signed c! c+! bis! bic! xor! bit@ hbis! hbic! hxor! hbit@
cbis! cbic! cxor! cbit@ --- Flash Dictionary --- list disasm-$ disasm-string name. u.4 u.8 u.ns const. addr. .decimal register.
inst funct3 funct7 .rs1 .rs2 .rd imm_i imm_s imm_sb imm_u imm_uj disasm-destination disasm-load disasm-immediate disasm-auipc di
sasm-store disasm-register disasm-lui disasm-branch disasm-jalr disasm-jal rvc-inst rvc-func3 rvc-reg. imm_css imm_csl imm_cu im
m_c imm_clwsw imm_cj imm_cb imm_addi4spn imm_addi16sp imm disasm-compressed disasm memstamp disasm-step seec see +inf -inf floor de
g0to360 deg-90to90 numbertable 0to1sqrt sqrt deg2rad rad2deg pi/2 pi/4 sin-coef half-q1-sin-rad cos-coef half-q1-cos-rad q1-sin-
rad q1toq4-sin atan-coef base-ivl-atan atan-table 0to1-atan sin cos tan atan asin acos log2-1to2 log2 log10of2 log10 lnof2 ln ex
p-coef exp-1to1 pow2 1overlnof2 exp ln10overln2 pow10 PORTA_Base PORTB_Base PORTC_Base PORTD_Base PORTE_Base PORTA_CRL PORTA_CRH
 PORTA_IDR PORTA_ODR PORTA_BSRR PORTA_BRR PORTB_CRL PORTB_CRH PORTB_IDR PORTB_ODR PORTB_BSRR PORTB_BRR PORTC_CRL PORTC_CRH PORTC
_IDR PORTC_ODR PORTC_BSRR PORTC_BRR PORTD_CRL PORTD_CRH PORTD_IDR PORTD_ODR PORTD_BSRR PORTD_BRR PORTE_CRL PORTE_CRH PORTE_IDR P
ORTE_ODR PORTE_BSRR PORTE_BRR Flamingo init cornerstone eraseflash  ok.
```



Habt ihr Fragen?

Demo „Grafik auf Longan Nano“
(Martin Bitter)

- Grafiktreiberpaket unter Mecrisp Quintus
  - Grundlagen
  - Programmiertechnik
  - Demo
- Turtlegrafik
  - Einführung
  - Demo



Links

- https://riscv.org
- http://www.riscvbook.com
- https://www.sifive.com
- https://www.seeedstudio.com
- https://www.sipeed.com
- https://www.analoglamb.com
- http://mecrisp.sourceforge.net