TA Hacke Dein bewaffnen Sie!

What can possibly go wrong?

_john
mCaCiCl: jDoDhCn3@Dt2u3x4c2oFdNeO.RoDrEg
(nur jeden 2. buchstaben lesen...)

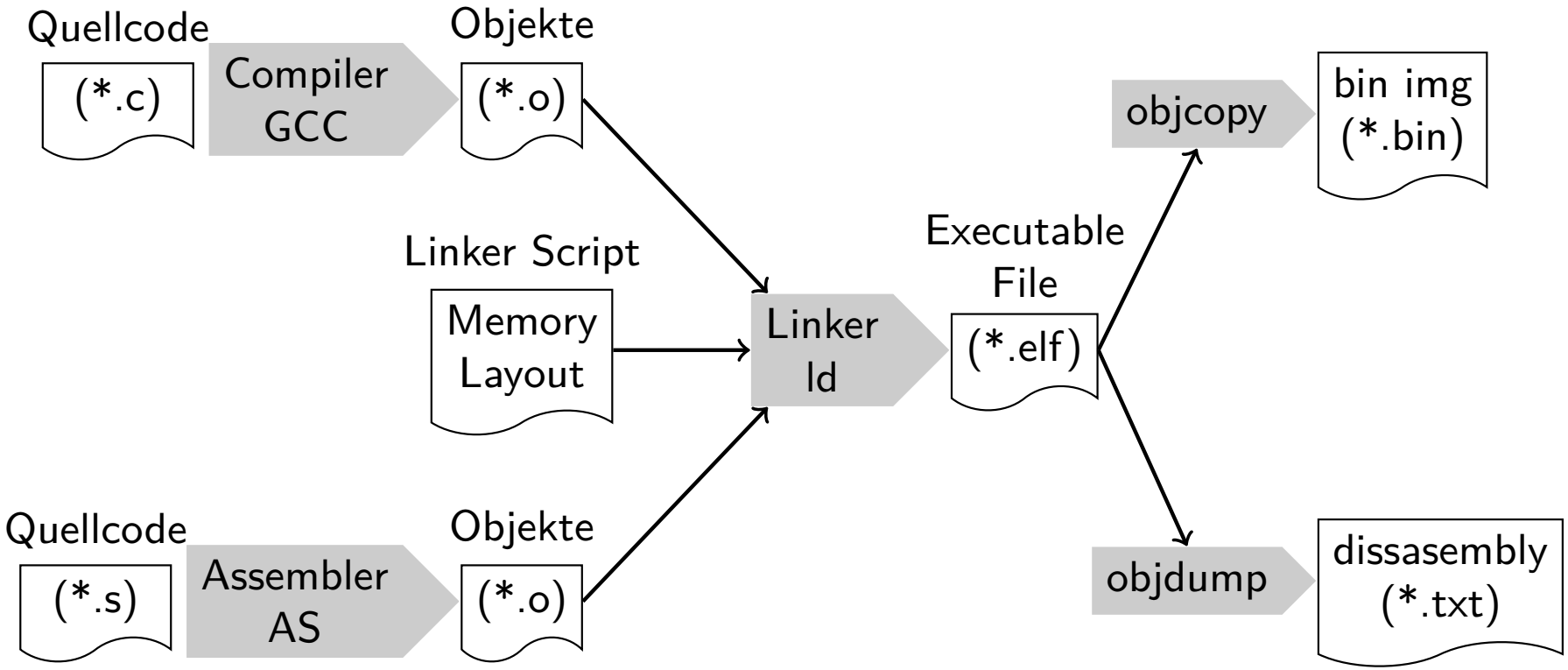
Juli 2012

Einleitung

gdb

Kerndetails Rinde M{3,4}

Programmerzeugung in C (GNU Toolchain)



Komplexe Sache

Der Teufel steckt im Detail.

Komplexe Sache

Der Teufel steckt in jedem Detail!

Problem:

- Was kann man tun?

klassiker: printf FTW?

- tip: debugf
- oder einfaches puts aufm uart (man puts)

makro:

```
#ifdef DEBUG
char debuf[32];
#define debugf(format, args...) do{sprintf(debuf,\
    "line%d:" format "\r\n", __LINE, ##args);\
    USART_puts(debuf); }while(0)
#else
#define debugf while(0){ };
#endif // ifdef DEBUG
```

more debugf

```
static void debug_dump_i2c_v(void)
{
    debugf("cr1:%x    cr2:%x",
           (((I2C_TypeDef*)I2C3_BASE)->CR1),
           (((I2C_TypeDef*)I2C3_BASE)->CR2));
    debugf("oar1:%x    oar2:%x",
           (((I2C_TypeDef*)I2C3_BASE)->OAR1),
           (((I2C_TypeDef*)I2C3_BASE)->OAR2));
    debugf("dr:%x    ccr:%x",
           (((I2C_TypeDef*)I2C3_BASE)->DR),
           (((I2C_TypeDef*)I2C3_BASE)->CCR));
    debugf("sr1:%x    sr2:%x",
           (((I2C_TypeDef*)I2C3_BASE)->SR1),
           (((I2C_TypeDef*)I2C3_BASE)->SR2));
    debugf("trise:%x    test:%x",
```

no UART

-blinkende leds....

arm-none-eabi-gdb

die toolchain liefert uns einen debugger!
arm-none-eabi-gdb foobar.elf

Einschub: Debugsymbole

Symbolnamen sind für Menschen praktischer.
Behalten wir sie also! -g / -ggdb
Optimierung ist für Menschen unpraktischer.
vll ausmachen!

remote target

stlink gdb-server kontrolliert Ausführung
Beispielsession 1:
remote extendet-target, kill, run

gdb-befehle 2

[l]ist	//Codeansicht
[br]eak	// Unterbrechungspunkt
[s]tep/[n]ext , [cont]inue	//Weiter ausführen
bt	//Stackdump
p	//Datenansicht
help foothema	//Hilfe!

Beispielsession 2

flash it!

Programmieren geht über den gdbserver:
`kill, load , run`

Exceptions

Typen von Ausnahmen:

- 1-15: system exceptions
- 16-255: external interrupt

siehe: [3, pp 115ff.]
leider nicht :[1, Kapitel 3.9]

Exception Vector Table

Bei Ausnahme:

- sieh in der Exception Vector Table nach
- spring an die eingetragene Adresse

Exception Vector Table

siehe pentstm32f4/core/startup_stm32f4xx.s oder [2, Kapitel 9]

operation Modes

Cortex-M{3,4} unterstüzt 2 Modi und 2 Privilegienstufen

	Privleged	User
running exception	HandlerMode	XXXXXX
running main prog	ThreadMode	ThreadMode

Faulthandler

Demo: Debugging mit HardFault Handler

- Fault Handler liefert Fehlerhafte Adresse
- objdump -dsl liefert lesbares Disassembly
- Auch linker ändert code!


ARM Limited: *Cortex-M4 Technical Reference Manual*.
ARM Limited, june 2010.
URL: http://infocenter.arm.com/help/topic/com.arm.doc.ddi0439c/DDI0439C_cortex_m4_r0p1_trm.pdf.


STMicroelectronics: *RM0090 Reference Manual STM32F405xx, STM32F407xx, STM32F415xx and STM32F417xx advanced ARM-based 32-bit MCUs*.
STMicroelectronics, September 2011.
URL: http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/REFERENCE_MANUAL/DM00031020.pdf.


Yiu, J.: *The Definitive Guide to the ARM Cortex-M3*.
Newnes, 2007.