Stefan, Pfeiffer
CST Institut für Informatik

Start the 

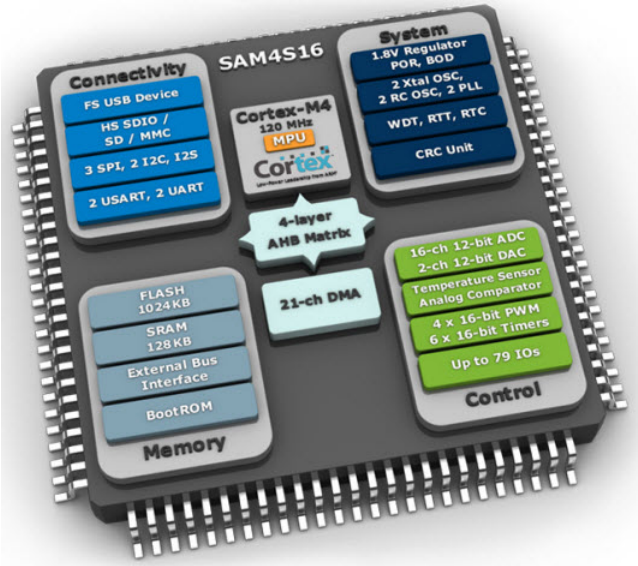
Embedded Systems / Microcontroller

Microprocessor vs. Microcontroller

- Microcontroller are microprocessor „plus“
- e.g. GPIO, SPI, I2C, I2S, USART, USB, MCI, Ethernet, CAN, DCMI, FSMC

Different usage model

- No keyboard, no mouse , no screen
- That means no local in- or outputdevice
- Forward STDIO over UART, USB or radiotechnologies or even very simple HID like buttons and LED

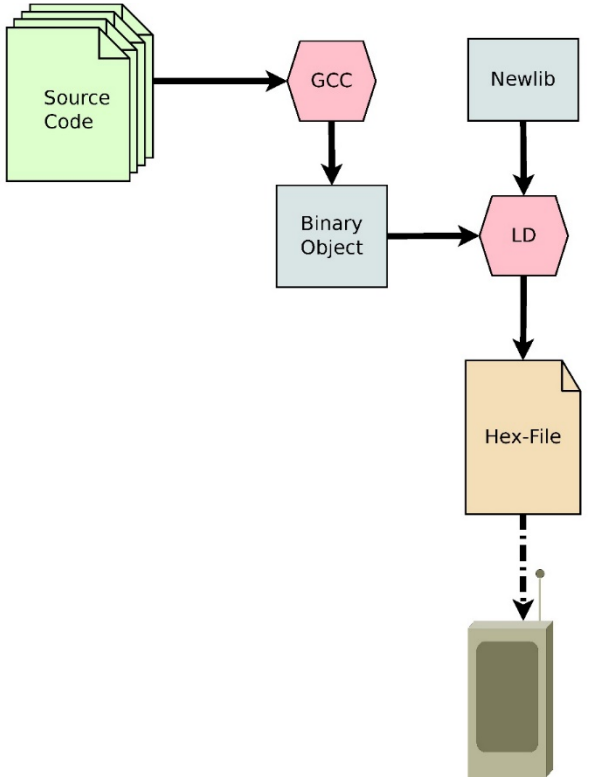


Embedded Development

Need other System(PC) for Development

- Crosscompiler
- Binary „Flashen“
- Static linking

- API: The microcontroller reference manual



OR use:

RIOT

http://riot-os.org/api/

„Hello World !“

RIOT Sourcecode:

git clone https://github.com/emmanuelsearch/RIOT.git

RIOT compile:

cd RIOT/examples

make

RIOT start:

make term

NEWLIB

http://sourceware.org/newlib/docs.html

- Standard ANSI C Bibliothek
- Supports functions for:
  - In- and Output

    `puts() printf() fprintf()`
  - String manipulation functions

    `strlen() strcat() strcmp()`
  - Math operations

    `sqrt() pow() sin()`
  - Memory management functions

    `malloc() free()`

Memory

- Linker collects all all binary objects and builds one big binary
- Seperation to different segments

```
SECTIONS
{       .text:
        {       // code und const -> flash memory
        }
        .data:
        {       // initialized variables -> RAM
        }
        .bss:
        {       // uninitialized data -> RAM
        }
}
```

Memory

- Which memory is used ?

```
void main(void)
{
        unsigned int foo = 0; -> Stack -> Sram
        char* fuu = malloc(128); -> SRam
}
```

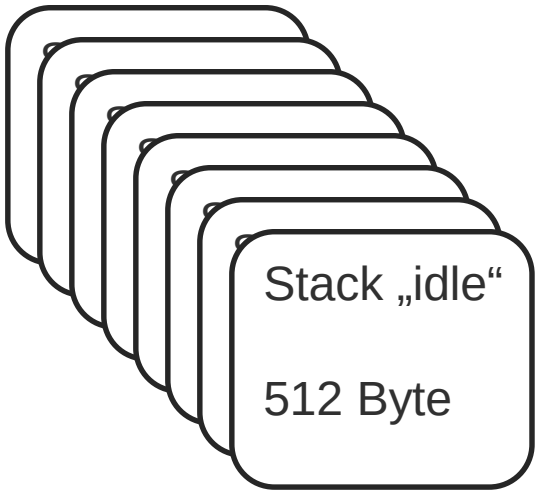
- How much memory is available ?e.g. STM32f4 (arm cortex m4)

FLASH 1 MByte

.text

Sram 128 kByte

.data
.bss
.stacks
.heap



Multithreading IPC

Threads

- Priority based(31(idle) … 1(highest))
- Tickless Scheduler -> Thread run until:
    1. abandon control -> `thread_yield()`
    2. an interrupt occurs

Threading API:

```
thread_create(…)
thread_sleep(…)
thread_wakeup(…)
thread_yield(…)
```

IPC API:

```
msg_send(…)
msg_receive(…)
msg_send_recieve(…)
msg_reply(…)
```

Links

RIOT Homepage

http://riot-os.org

RIOT on Github

https://github.com/emmanuelsearch/RIOT.git

RIOT Developers mailing list

devel@riot-os.org

RIOT API Documentation

http://riot-os.org/api

RIOT Development Image

http://download.riot-os.org/RIOT-dev.vdi.zip

Newlib Documentation

http://sourceware.org/newlib/docs.html