Hinweise – AVR-Architektur

Dr.-Ing. Volkmar Sieh

Department Informatik 4
Verteilte Systeme und Betriebssysteme
Friedrich-Alexander-Universität Erlangen-Nürnberg

WS 2019/2020

AVR – Dokumentation

Dokumentation: „Atmel AVR ATMEGA32“

- „Atmel AVR 8-bit Instruction Set“
  `http://www.atmel.com/images/Atmel-0856-AVR-Instruction-Set-Manual.pdf`
  Beschreibung der Assembler-Befehle sowie ihre Kodierung
- „Atmel 8-bit AVR Microcontroller with 32KBytes In-System Programmable Flash – ATmega32/ATmega32L“
  `http://www.atmel.com/images/doc2503.pdf`
  Beschreibung der Hardware: CPU, Flash, RAM, I/O

AVR – Tool-Chain

gcc-avr:	`avr-gcc`, `avr-g++`, ...
binutils-avr:	`avr-as`, `avr-ld`, `avr-ar`, `avr-objdump`, `avr-objcopy`, ...
avr-libc:	`libc.a`, `libm.a`, includes, man-pages

AVR-Architektur – Überblick

- CPU (16-Bit Befehle; 8-Bit Daten)
- 32KByte In-System/Self-programmable Flash, 2KByte RAM, 1KByte EEPROM
- 3 Timer
- Real Time Counter
- Interrupt-Controller
- 4 8-bit-Ports:
  - A: General-Purpose oder 8-Kanal/10-Bit-AD-Wandler
  - B: General-Purpose oder Comperator/SPI (Serial Peripheral Interface)
  - C: General-Purpose oder TWI (Two-Wire Serial Interface)
  - D: General-Purpose oder USART (Universal Sync/Async Receiver/Transmitter)

16 MHz Taktfrequenz, 5V Spannung, $<$ 5mA Stromaufnahme

AVR-Architektur – CPU

Register:

- Programm-Zähler
- Stack-Pointer
- Statusregister:
  - C: Carry-Flag
  - Z: Zero-Flag
  - N: Negative-Flag (= Sign-Flag)
  - V: Overflow-Flag
  - S: (= N xor V)
  - H: Half-Carry-Flag
  - T: Transfer-Bit (für Bit-Load/Bit-Set)
  - I: Interrupt-Enable
- ...

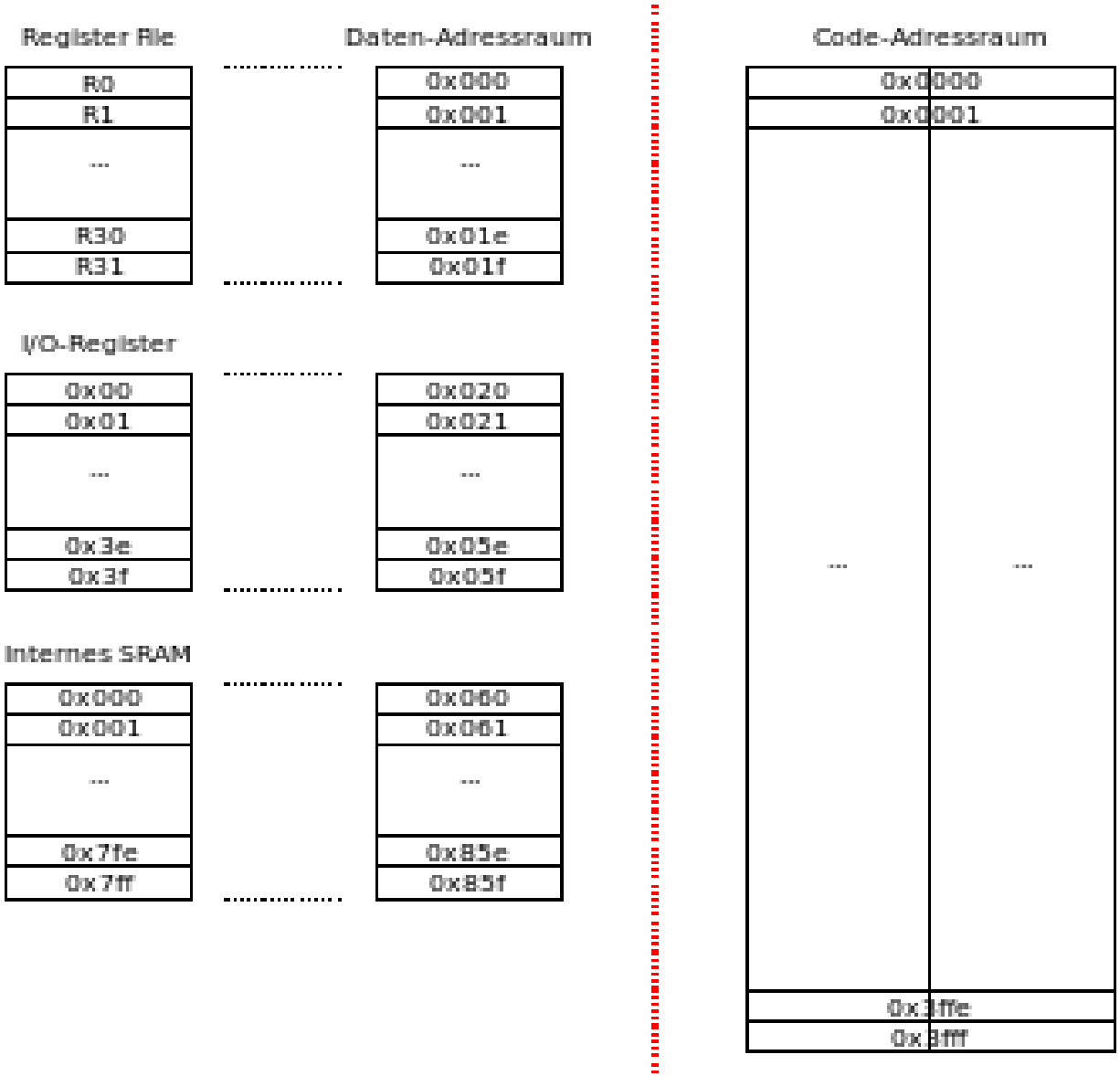
AVR-Architektur – CPU

- ...
- 26 8-Bit-Register:
  - R0
  - ...
  - R25
- 3 16-Bit-Register:
  - X (=R27:R26)
  - Y (=R29:R28)
  - Z (=R31:R30)

Register wie Speicherzellen ansprechbar:

7 0	Addr.	
R0	0x00	
R1	0x01	
R2	0x02	
...		
R25	0x19	
R26	0x1a	X (Low Byte)
R27	0x1b	X (High Byte)
R28	0x1c	Y (Low Byte)
R29	0x1d	Y (High Byte)
R30	0x1e	Z (Low Byte)
R31	0x1f	Z (High Byte)

AVR-Architektur – Speicher



Aufbau Opcodes

- Befehle meistens 16-Bit lang

Beispiele:

```
nop     0000 0000 0000 0000
ret     1001 0101 0000 1000
sleep   1001 0101 1000 1000
```

Aufbau Opcodes

- Befehle meistens 16-Bit lang
- Parameter im Opcode enthalten

Beispiele:

```
add Rd, Rr   0000 11rd dddd rrrr
adiw Rd, K   1001 0110 KKdd KKKK
pop Rd       1001 000d dddd 1111
```

Aufbau Opcodes

- Befehle meistens 16-Bit lang
- Ausnahme: Befehle mit „langen” Adressen

Beispiele:

```
call k      1001 010k kkkk 111k  kkkk kkkk kkkk kkkk
jmp k       1001 010k kkkk 110k  kkkk kkkk kkkk kkkk
lds Rd, k   1001 000d dddd 0000  kkkk kkkk kkkk kkkk
sts k, Rd   1001 001d dddd 0000  kkkk kkkk kkkk kkkk
```