RUHR-UNIVERSITÄT BOCHUM

RUB

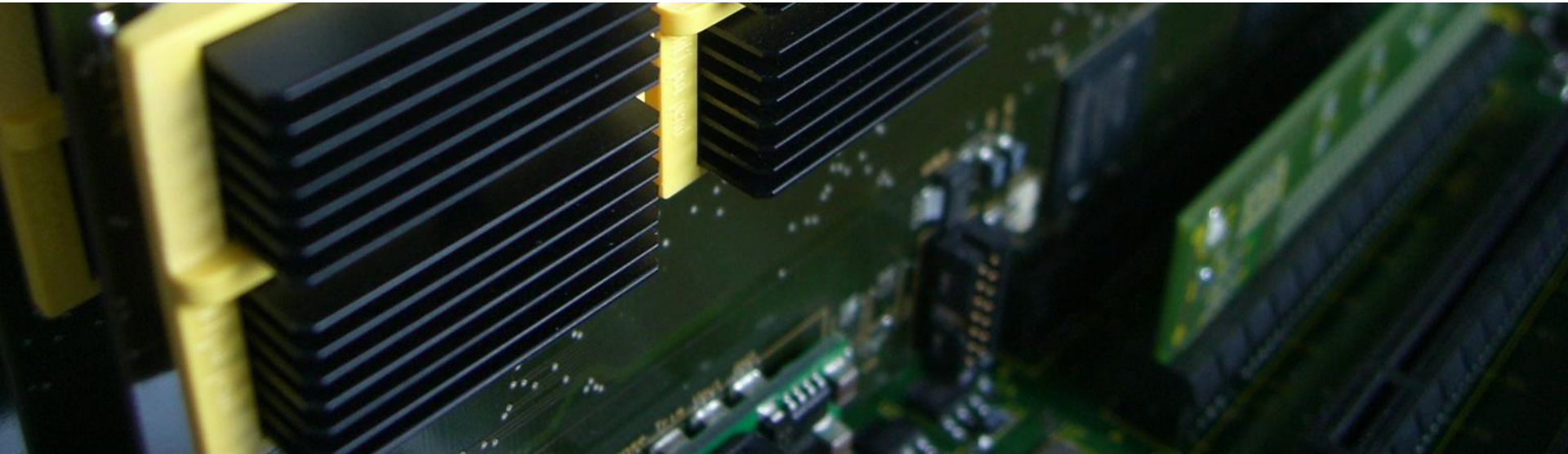
Eingebettete Prozessoren

SS 2014

Übung 7: Speicherzugriff

Dipl.-Ing. Thomas Pöppelmann
Arbeitsgruppe Sichere Hardware
Horst Görtz Institut für IT-Sicherheit

05.06.2014



Agenda

1. Organisatorisches
2. Speicherzugriff

1. Organisatorisches

- Am 12.06 keine Übung, am 13.06 keine Vorlesung
- Keine neue Hausaufgabe am 12.06 (Pfingstferien)

- Abgabefrist für Übung 7 ist in zwei Wochen (19.06.2014, 12:15 Uhr)

3. Speicherzugriff

Hinweis

- Als Ergänzung zur Präsenzübung befindet sich im Blackboard ein interaktives Tutorial aus dem Sommersemerster 2013 (Interaktive_Praesenzuebung.asm)

- Empfehlenswert ist außerdem das folgende Tutorial (neben Skript und Datenblatt):
  - http://www.mikrocontroller.net/articles/AVR-Tutorial:_Speicher

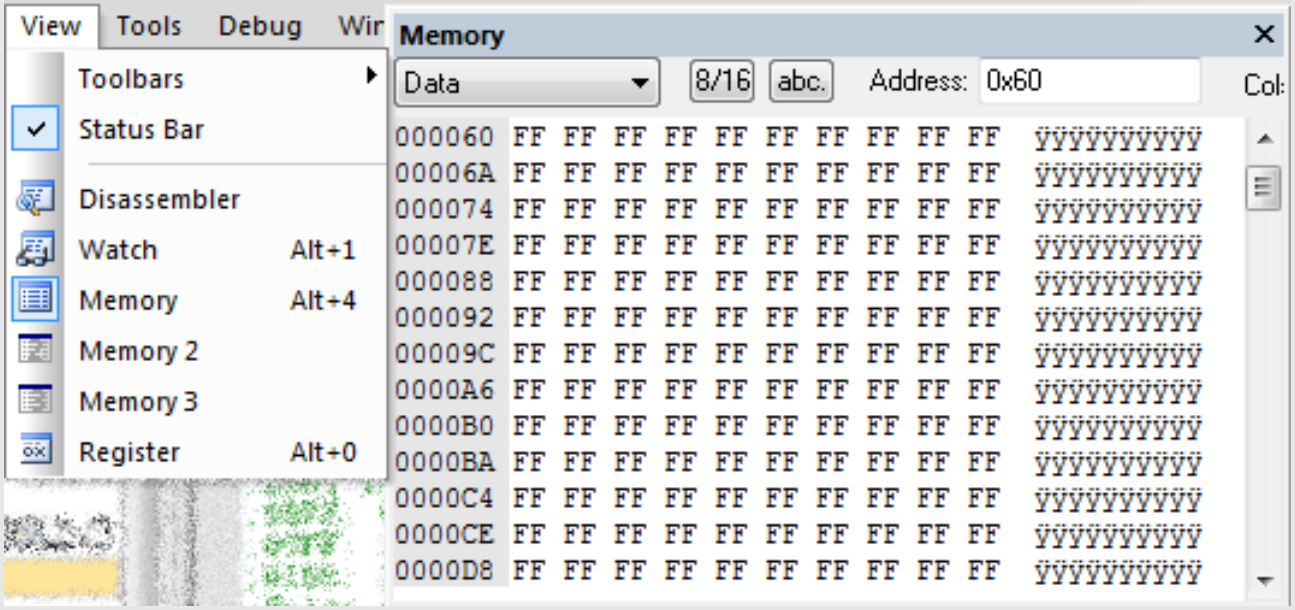
3. Speicherzugriff: SRAM vs. Flash

- SRAM
  - Flüchtig, byte-adressierbar
  - Lese-/Schreibzugriffe in 2 Zyklen
  - Adressmarken (.DSEG) realisieren Variablen im SRAM
  - Kann nicht initialisiert werden

- Flash
  - Nicht-flüchtig, 16-bit adressierbar
  - Lesezugriff in 3 Takten, Schreiben sehr aufwändig
  - Laden mit LPM/SPM
  - Little-endian (niederwertigstes Byte an niedrigster Adresse)

3. Speicherzugriff: Debugging

- AVR Studio Memory Ansicht
  - Beobachten und Manipulieren der verschiedenen Speicherbereiche (RAM, FLASH, EEPROM) während einer Debugging-Session



3. Speicherzugriff: SRAM schreiben

```
.include "m8def.inc"
.DSEG
Counter:   .BYTE  5 ;Anzahl
.CSEG
LDI r16, 0

;Direktzugriff
STS Counter,r16   ;Store direct To data Space

;Indirekter Zugriff
LDI ZL, LOW(Counter)
LDI ZH, HIGH(Counter)
INC r16
ST Z, r16              ;Indirektes Speichern
```

Debugger

3. Speicherzugriff: Flash auslesen

```
.include "m8def.inc"
.CSEG
to_main1: rjmp main1
flash_data:
.DB 0x11,0x22

main1:
   LDI    ZL, LOW(flash_data*2) ; Alternative: flash_data<<1
   LDI    ZH, HIGH(flash_data*2)
   LPM ; lese 0x11 in r0

   LDI   ZL, LOW(flash_data*2+1) ; Setze niederwertigstes Bit
   LDI   ZH, HIGH(flash_data*2+1) ; Alternative: flash_data | 0x01
   LPM ; lese 0x22 in r0
```

Debugger

3. Speicherzugriff: Pointerarithmetik

- Manipulieren der Pointeradressen aufwändig.

1. Indirekter Zugriff mit Adressverschiebung
   - **LDD R16 , Y+4** ; Lade Inhalt an Y+4 in R16
2. Indirekter Zugriff mit Prädekrement/Postinkrement
   - **LD R16 , Z+** ; Lade Inhalt aus Z in R16 und inkrementiere Z
3. ADIW – Add Immediate to Word
   - **ADIW ZH:ZL,63** ; Erhöhe Z-pointer(r31:r30) um 63