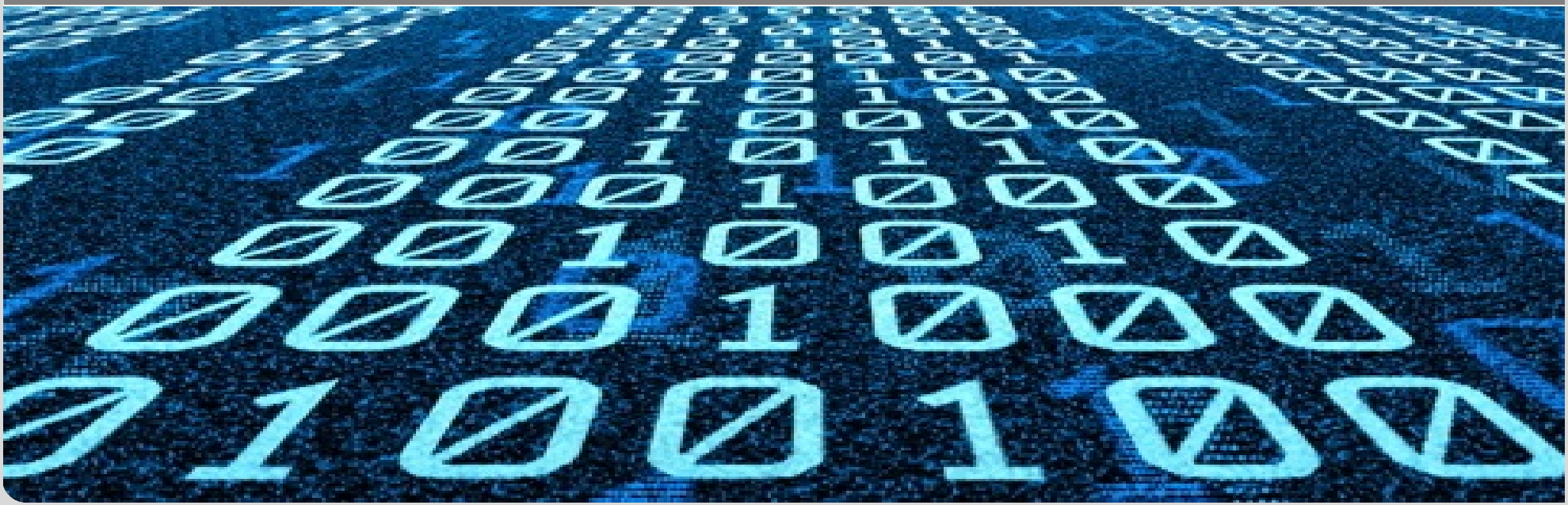
Karlsruher Institut für Technologie

3. Tutorium Mikroprogrammmierung und Assemblerprogrammierung

Rechnerorganisation, Tutorium #13

Patrick Röper | 19. November 2019

FAKULTÄT FÜR INFORMATIK



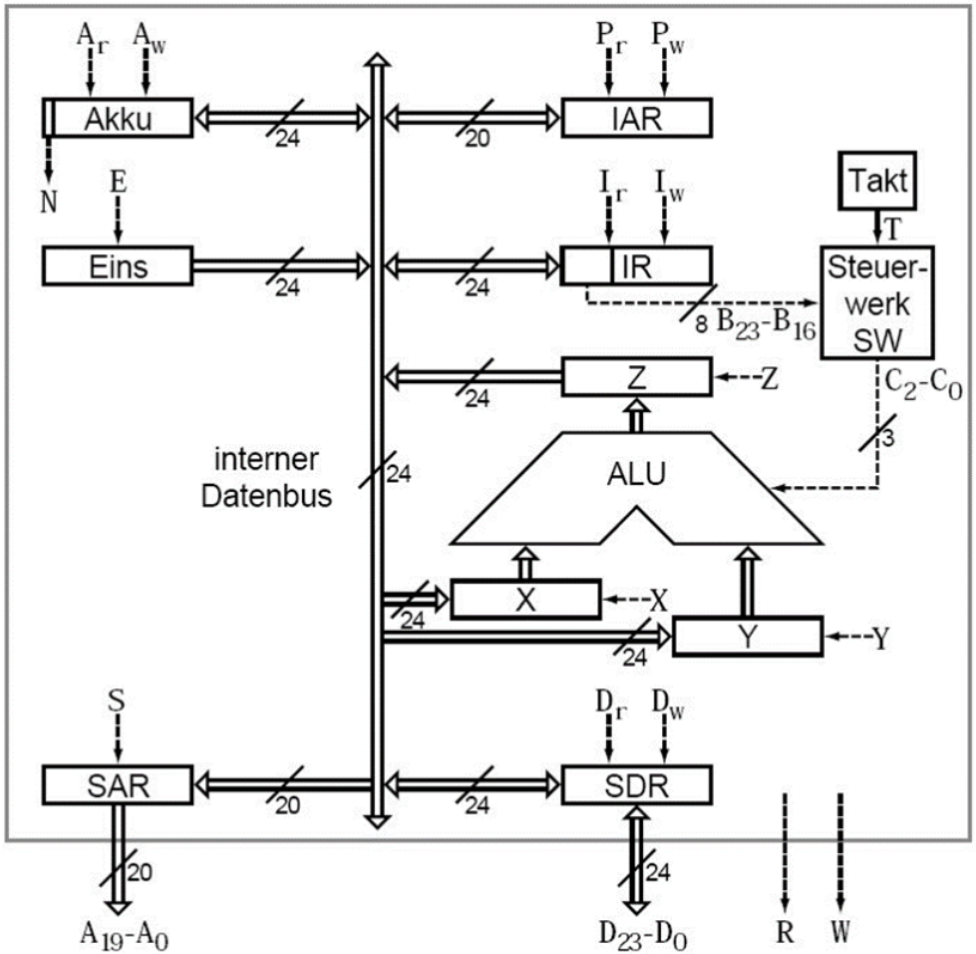
Roadmap

Tutorium 2

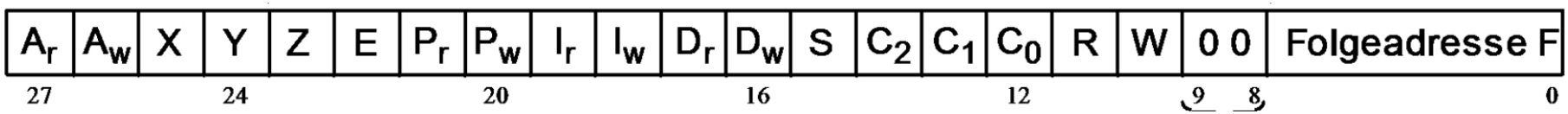
Zweierkomplement

```
Ausfuehrungsphase:
 7. Takt Akku -> X
 8. Takt ALU auf NOT (Einskomplement)
 9. Takt Z -> X
10. Takt Eins -> y
11. Takt ALU auf ADD
11. Takt Z -> Akku
```

Tutorium 2

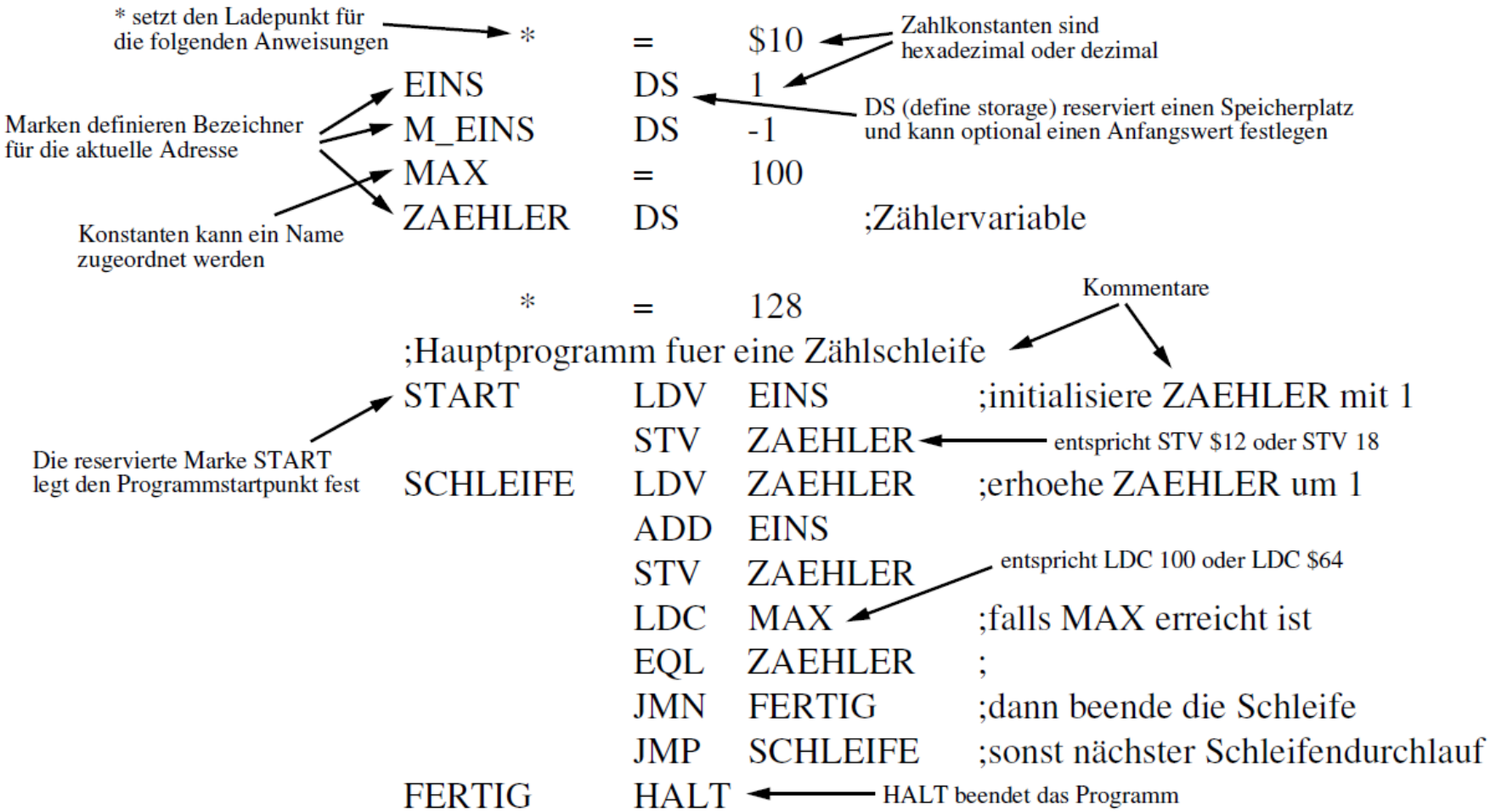


Mikrobefehlsformat



Roadmap

Assembler




```
                *       =       $10
EINS            DS      1
M_EINS          DS      -1
MAX             =       100
ZAEHLER         DS              ;Zählervariable

                *       =       128
;Hauptprogramm fuer eine Zählschleife
START           LDV     EINS            ;initialisiere ZAEHLER mit 1
                STV     ZAEHLER
SCHLEIFE        LDV     ZAEHLER         ;erhoehe ZAEHLER um 1
                ADD     EINS
                STV     ZAEHLER
                LDC     MAX             ;falls MAX erreicht ist
                EQL     ZAEHLER         ;
                JMN     FERTIG          ;dann beende die Schleife
                JMP     SCHLEIFE        ;sonst nächster Schleifendurchlauf
FERTIG          HALT
```

Roadmap

Aufgabe 1

Aufgabe

Zur Vereinfachung von Unterprogrammaufrufen wird die MIMA um die Befehle **JMS (jump subroutine)** und **JIND (jump indirect)** erweitert.

Der Befehl **JIND ziel** bewirkt, dass zu derjenigen Adresse verzweigt wird, die in der Speicherzelle ziel gespeichert ist. Dabei werden die vier höchstwertigen Bit des 24 Bit langen Speicherwortes ziel ignoriert, um eine 20 Bit lange Adresse zu erhalten.

Der Befehl **JMS ziel** bewirkt, dass die Adresse des nachfolgenden Befehls (Rücksprungadresse), an der Speicheradresse ziel abgelegt und anschliessend zum Befehl mit der Adresse ziel + 1 verzweigt wird.

Aufgabe 1

Aufgabe

OpCode	Mnemonik	Beschreibung
D	JIND a	<a> $\Rightarrow$ IAR
C	JMS a	IAR+1 $\Rightarrow$ <a>, a+1 $\Rightarrow$ IAR

Geben Sie die Mikroprogramme für die Ausführungsphase (execute phase) der Befehle JMS und JIND an (jeweils ab dem 7. Takt).

Lösung 1

Lösung 1 JIND

```
JIND:
7. Takt: IR -> SAR; R = 1
8. Takt: R = 1
9. Takt: R = 1
10. Takt: SDR -> IAR
```

Lösung 1

Lösung 1 JIND

```
JIND:
7. Takt: IR -> SAR; R = 1
8. Takt: R = 1
9. Takt: R = 1
10. Takt: SDR -> IAR
```

Lösung 1 JMS

```
JMS:
7. Takt: IAR -> SDR;
8. Takt: IR -> SAR; IR -> X; W = 1
9. Takt: EINS -> Y; W = 1
10. Takt: ALU auf ADD; W = 1
11. Takt: Z -> IAR
```

Aufgabe 2

Aufgabe

Schreiben Sie ein C-Programm, das den Wochentag zu einem beliebigen (TAG, MONAT) im Jahr 2004 berechnet. Dazu werden die Wochentage wie folgt definiert:

0 = Sonntag, 1 = Montag, 2 = Dienstag, 3 = Mittwoch, 4 = Donnerstag, 5 = Freitag, 6 = Samstag

eines Monats

Sie können davon ausgehen, dass die ersten Wochentage in einem Array '**ersterTag** gespeichert sind. Das Programm soll bei Ausführung den Wochentag auf den Standardoutputstream (std::out) printen.

Lösung 2

Lösung

```
main() {
      int ersterTag[]={-1,4,0,1,4,6,2,4,0,3,5,1,3};
      int TAG = 3;
      int MONAT = 6;
      int wochentag = TAG + ersterTag[MONAT] - 1;
      wochentag %= 7;
      printf("Wochentag = %i\n", wochentag);
}
```

Aufgabe 3

Aufgabe

Schreiben Sie ein MIMA-Programm, das den Wochentag zu einem beliebigen (TAG, MONAT) im Jahr 2004 berechnet. Dazu werden die Wochentage wie folgt deiniert:

0 = Sonntag, 1 = Montag, 2 = Dienstag, 3 = Mittwoch, 4 = Donnerstag, 5 = Freitag, 6 = Samstag

Das Programm soll an der Speicheradresse 0x00100 beginnen. Der TAG steht in der Speicherzelle 0x00020 zur Verfügung und der MONAT steht an Adresse 0x00021 bereit. Nach Ablauf des Programms soll sich die Wochentagszahl an Adresse 0x00030 befinden.

Aufgabe 3

C-Code Aufgabe

Sie können sich bei der berechnung an folgendem C-Code orientieren.

```
main() {
    int ersterTag[]={-1,4,0,1,4,6,2,4,0,3,5,1,3};
    int TAG = 3;
    int MONAT = 6;
    int wochentag = TAG + ersterTag[MONAT] - 1;
    wochentag %= 7;
    printf("Wochentag = %i\n", wochentag);
}
```

Die jeweiligen Wochentage des ersten Tages im Monat stehen hier im Array ersterTag zur Verfuegung. Addiert man zu diesem Wochentag den (TAG-1) und rechnet modulo 7, so erhaelt man den gewuenschten Wochentag wie oben definiert. Beachten Sie hierbei, dass in C die Arrayelemente mit 0 beginnend nummeriert werden, während der Kalender immer bei 1 anfängt zu zählen.

Lösung 2

```
; Definition der ersten Tage im Monat:
0x00001  000004  ARRAY:  DS  4  ; Januar (Do)
0x00002  000000          DS  0  ; Februar (So)
0x00003  000001          DS  1  ; März (Mo)
0x00004  000004          DS  4  ; April (Do)
0x00005  000006          DS  6  ; Mai (Sa)
0x00006  000002          DS  2  ; Juni (Di)
0x00007  000004          DS  4  ; Juli (Do)
0x00008  000000          DS  0  ; August (So)
0x00009  000003          DS  3  ; September (Mi)
0x0000A  000005          DS  5  ; Oktober (Fr)
0x0000B  000001          DS  1  ; November (Mo)
0x0000C  000003          DS  3  ; Dezember (Mi)
```

Lösung 2

```
; Eingabe:
0x00020 | 000003 |  TAG:     DS    3          ; Tag
0x00021 | 000006 |  MONAT:   DS    6          ; Monat
; Hilfsvariablen
0x00022 | FFFFF9 |  MINUS7:  DS    0xFFFFF9   ; -7
0x00023 | FFFFFF |  MINUS1:  DS    0xFFFFFF   ; -1
0x00024 | 000007 |  PLUS7:   DS    7          ; 7
; Ausgabe
0x00030 | 000000 |  ERG:     DS    0          ; Ergebniszelle
```

Lösung 2

```
; Hauptprogramm
0x00100 | 100021 | START:     LDV    MONAT     ; Akku=Monat
0x00101 | 300103 |            ADD    BEFEHL    ; OpCode für LDV dazu addieren
0x00102 | 200103 |            STV    BEFEHL    ; Befehl überschreiben
0x00103 | 100000 | BEFEHL:    LDV    0         ; Akku=ersterTagimMonat[Monat]
0x00104 | 300020 |            ADD    TAG       ; Akku+=Tag
0x00105 | 300023 |            ADD    MINUS1    ; Akku-=1;
0x00106 | 300022 | SCHLEIFE:  ADD    MINUS7    ; | Be-
0x00107 | 900109 |            JMN    RAUS      ; | rechnung
0x00108 | 800106 |            JMP    SCHLEIFE  ; | von
0x00109 | 300024 | RAUS:      ADD    PLUS7     ; | Akku%=7
0x0010A | 200030 |            STV    ERG       ; Wochentag in 0x30 schreiben
0x0010B | F00000 | ENDE:      HALT             ;
```

Was ihr jetzt kennen und können solltet…

- MIMA: Mikroprogrammmierung und Assemblerprogrammierung