

6. Übung

- Wdh. Unterprogrammaufrufe
- Unterbrechungen und Ausnahmen
 - Fallstudie MIPS

Einfacher Unterprogrammaufruf

```
main:      jal subroutine

subroutine: # keine weiteren
           # Unterprogrammaufrufe
           # für lokale Variablen
           # nur $t0 bis $t9
           jr $ra
```

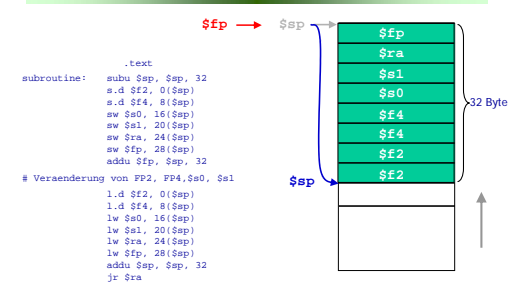
Beispiel für einen Unterprogrammaufruf

```
# Hauptprogramm      .text
                   subu $sp, $sp, 32
                   s.d $f2, 0($sp)
                   s.d $f4, 8($sp)
                   sw $s0, 16($sp)
                   sw $s1, 20($sp)
                   sw $ra, 24($sp)
                   addu $fp, $sp, 32
                   jal subroutine
                   lw $ra, 0($sp)
                   lw $fp, 4($sp)
                   addu $sp, $sp, 8
                   jr $ra

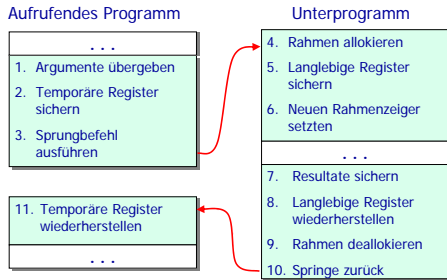
subroutine:        subu $sp, $sp, 32
                   s.d $f2, 0($sp)
                   s.d $f4, 8($sp)
                   sw $s0, 16($sp)
                   sw $s1, 20($sp)
                   sw $ra, 24($sp)
                   addu $fp, $sp, 32
                   l.d $f2, 0($sp)
                   l.d $f4, 8($sp)
                   lw $s0, 16($sp)
                   lw $s1, 20($sp)
                   lw $ra, 24($sp)
                   lw $fp, 28($sp)
                   addu $sp, $sp, 32
                   jr $ra
```

Unterprogrammaufrufe mit jal-Befehl!

Beispiel für einen Unterprogrammaufruf



Unterprogrammaufruf



Behandlung von Ausnahmesituationen

- Während des Betriebs eines Mikroprozessorsystems können Ausnahmesituationen (Exceptions) auftreten
- Eine Ausnahmesituation erfordert eine vorübergehende Unterbrechung oder gar den Abbruch des laufenden Programms
- Ursachen:
 - Fehler im System bei der Ausführung des Anwenderprogramms oder Fehler der Hardware
 - Wunsch externer Systemkomponenten, die Aufmerksamkeit des Prozessors zu erhalten
- Die Ausnahmebehandlung erfolgt durch eine Ausnahmeroutine (Interrupt Service Routine), deren Aktivierung durch eine Hardware-Komponente (Unterbrechungs-System, Interrupt System) im Steuerwerk unterstützt wird.
- Die Ausnahmeroutine hat Ähnlichkeit mit dem Aufbau eines Unterprogramms. Es gibt jedoch wesentliche Unterschiede:

Ausnahmeroutine/Unterprogramm

- Aktivierung:
 - call subroutine bei Unterprogramm
 - Hardware-Aktivierung durch Externes Signal bei Ausnahmeroutine
- Beendigung:
 - ret-Befehl bei Unterprogrammen (return from subroutine)
 - reti-Befehl bei Ausnahmebehandlung (return from interrupt)
- Einsprungsadresse ins Unterprogramm direkt im Programm, bei Ausnahmebehandlung über Interrupt-Tabelle
- Unterprogrammaufruf sichert meist nur den PC auf den Stack, Ausnahmebehandlungs-Aufruf meist auch das PSW
- Unterprogrammaufrufe werden immer durchgeführt, die meisten Ausnahmebehandlungen werden nur aktiviert, falls das Interrupt-Enable-Bit im Steuerregister (PSW) gesetzt ist

Unterbrechungen und Ausnahmen

- Unterbrechung (Interrupt):
 - zum Programmablauf asynchrone Ereignisse.
 - Sie werden durch die Hardware erzeugt.
 - Prozessorexterne Ursachen
 - Beispiel: Ein- und Ausgabegeräte, z.B. die Tastatur
- Ausnahme (Exception, Trap):
 - zum Programmablauf synchrone Ereignisse.
 - Sie treten an fest vorgegebenen Stellen im Programm ein.
 - Prozessorinterne Ursachen
 - Beispiel: arithmetische Fehler, ungültiger Befehl, ...

Prozessorexterne Ursachen

- RESET: Rücksetzen des Mikrorechnersystems, z. B. ausgelöst durch Taste, Schwankungen der Betriebsspannung, Watch-Dog, ...
- HALT: Anhalten des Prozessors, z. B. zur Vermeidung von Zugriffskonflikten auf dem Systembus bei DMA-Zugriffen, Paritätsfehler
- ERROR: Aufruf einer Fehlerbehandlungsroutine, z. B. bei Bus-Fehlern oder Zugriffe auf geschützte Datenbereiche

Prozessorexterne Ursachen

- (Hardware)-Interrupt: Unterbrechungsanforderung von einem peripheren Gerät, z. B. um verfügbare Daten eines Eingabegerätes anzukündigen
- 2 Arten: maskierbar/nicht maskierbar (NMI)
 - Nicht maskierbare Interrupts (NMI): werden nach Abschluss des gerade ausgeführten Befehls unbedingt durchgeführt (wichtige Ausnahmesituationen, z. B. Zusammenbruch der Betriebsspannung)
 - Maskierbare Interrupts (IRQ): werden nur dann ausgeführt, wenn im Steuerregister des Prozessors das Interrupt-Enable Flag (IE-Bit) gesetzt ist.

Prozessorinterne Ursachen

- Prozessorinterne Ursachen: Bei Prozessoren, die auf dem Chip interruptfähige Komponenten besitzen (z. B. serielle oder parallele Schnittstellen, Zeitgeber, ..). Treten synchron zum Programmablauf auf.
 - Software Interrupts: durch SWI- (software interrupt) oder INT-Befehl im Programm ausgelöste Unterbrechungen
 - Traps: Ausnahmesituationen durch Prozessorinterne Ereignisse, z. B. Overflow, Divide by Zero, Stack Overflow, ungültiger OpCode, Seitenfehler (Page Trap), Einzelschritt-Unterbrechung (Trace)

Interrupt-Vektortabelle

- Interrupt-Vektortabelle (Ausnahme-Vektortabelle):
 - Festwertspeicher an spezieller Speicheradresse (oft in einer der untersten Speicherseiten)
 - Enthält die Startadressen der Behandlungsroutinen
 - Interrupt-Quelle liefert bei Interrupt eine Interrupt-Vektor-Nummer (IVN), welche den Eintrag in der Interruptvektor-Tabelle charakterisiert
 - Basis-Adressregister enthält die Basisadresse der Tabelle

Beispiel einer Vektortabelle

Index	Ausnahmesituation	
0	divide by 0	Division durch 0
1	overflow	Zahlenbereichüberschreitung
2	array bound check	Indexbereichüberschreitung
3	invalid opcode	illegaler Befehlscode
4	SVC (supervisor call)	Betriebssystem-Aufruf
5	privilege violation	unerlaubter Aufruf einer privilegierten Operation
6	trace	Einzelschritt-Modus
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350</	

Ablauf einer Interrupt Service Routine

- Startadresse der Interrupt-Service-Routine ermitteln und in den PC laden
- Interrupt Service Routine ausführen:
 - meist werden zuerst die benutzte Register auf den Stack gesichert
 - Interrupt Enable Bit wieder setzen
 - Am Ende der Interrupt Service Routine: IRET-Befehl
- PSW und PC werden wiederhergestellt und mit dem unterbrochenen Programm wird fortgefahren.

MIPS - Ausnahmen

Code	Beschreibung
0	externe Unterbrechung
4	fehlerhafte Adresse beim Laden oder Befehl-Holen
5	fehlerhafte Adresse beim Speichern
6	Busfehler beim Befehl-Holen
7	Busfehler beim Laden oder Speichern von Daten
8	Systemaufruf
9	Programmunterbrechung (breakpoint) zur Fehlersuche
10	reservierter Befehl
12	arithmetischer Überlauf bei Ganzzahlberechnungen
14	ungültiges Fließkomma-Ergebnis
15	Division durch Null
16	Überlauf bei Fließkomma-Berechnung
17	Unterlauf bei Fließkomma-Berechnung

Software

Beispiele für Ausnahmen

Externe Unterbrechung:

- Drücken einer Taste
- Maus wurde bewegt
- Timer ist abgelaufen
- Peripheriegerät (z. B. Drucker) ist fertig
- serielle Schnittstelle hat ein Wort empfangen

Fehlerhafte Adresse:

```
.data
strng: .asciiz "arglwuz = "
result: .space 23

.text
.globl main
main: ...
lw $a0, result
```

Beispiele für Ausnahmen

Busfehler:

- Time-Out
- Paritätsfehler
- ungültige Port- oder Speicher-Adressen

Programmunterbrechung (breakpoint):

Ausgabe von Register- oder Speicherinhalten oder des Prozessorstatus zur Fehlersuche

Systemaufruf (syscall)

Software-Interrupt zur Ausführung von Unterprogrammen des Betriebssystems (vergleichbar mit INT n beim 8086)

Beispiele für Ausnahmen

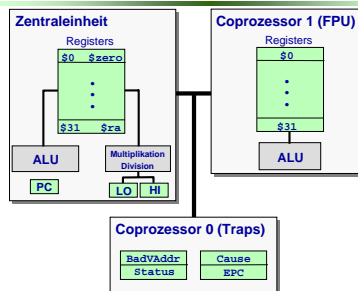
Reservierter Befehl:

Es wurde versucht, einen Befehl auszuführen, dessen Bits 31...26 nicht als Opcode definiert sind

Arithmetischer Überlauf bei Ganzzahlberechnungen:

Ergebnis einer ADD, ADDI oder SUB Instruktion lässt sich nicht als 32-Bit Zweierkomplement darstellen

Aufbau des MIPS-Prozessors



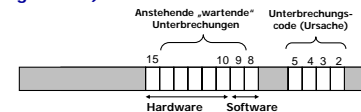
Register des Coprozessors 0

Coprozessor 0 enthält vier Register zur Behandlung von Ausnahmen und Unterbrechungen:

BadVAddr (Reg.-Nr. 8):

Falls die Unterbrechung durch einen Speicherzugriff verursacht wurde, ist hier die Speicheradresse enthalten

Cause (Reg.-Nr. 13):

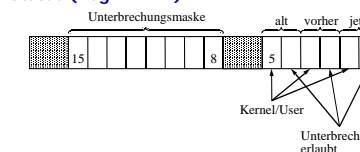


Register des Coprozessors 0

EPC (Reg.-Nr. 14):

Speicheradresse, in der sich der Befehl befindet, der zur Unterbrechung geführt hat

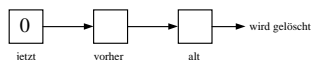
Status (Reg.-Nr. 12):



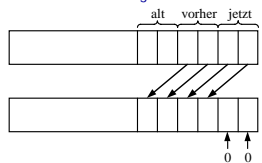
Die Unterbrechungsmaske bestimmt, welche Unterbrechungen zugelassen sind

Register des Coprozessors 0

Speichern der Kernel/User- und Interrupt-Enable-Bits:

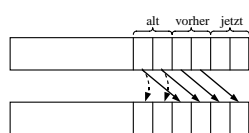


Beim Auftreten einer Unterbrechung:



Register des Coprozessors 0

Rückprung aus einer Unterbrechung (rfe Instruktion):



Ausnahmebehandlung (Interrupt-Handler)

Bei Ausnahme oder Unterbrechung erfolgt Sprung nach Adresse 8000 0080₁₆

```
.ktext 0x80000080
```

Benutzung des Stapels zur Sicherung von Daten nicht erlaubt, daher Sicherung von Registern im Kerneldatensegment

```
sw $a0, a0_save
sw $a1, a1_save
sw $ra, ra_save
```

Register \$at sichern

```
.set noat
sw $at, at_save
.set at
```

Ausnahmebehandlung (Interrupt-Handler)

Laden des Cause- und EPC-Registers

```
mfc0 $k0, $13
mfc0 $k1, $14
```

Unterbrechungen ignorieren

```
bgt $k0, 0x44, fertig
```

Spezifische Maßnahmen

```
... (Benutzung von $a0 und $a1)
jal print_exc
```

Abschluss der Ausnahmebehandlung

```
fertig: lw $a0, a0_save
lw $a1, a1_save
lw $ra, ra_save
.set noat
lw $at, at_save
.set at
```

Statusregister wiederherstellen

```
rfe
```

Fehlerhafte Instruktion soll nicht nochmals ausgeführt werden

```
addiu $k1, $k1, 4
```

Standard-Traphandler

```
s1: .word 0 # Speicher zum Sichern von Registern;
s2: .word 0 #
.ktext 0x80000080 # Code gehört zum Kernel Textsegment
.set noat # keine Nutzung von $at durch Assembler
move $k1, $at # Retten von $at in $k1 (für Kernel)
.set at # Nutzung durch von $at wieder möglich
sw $v0, s1 # Rette $v0 in feste Speicherzelle
sw $a0, s2 # Rette $a0 in feste Speicherzelle
# Nutzung von $at, $v0 und $a0
mfc0 $k0, $13 # Lade Unterbrechungsursache
... # Verzweige abhängig von $k0
... # Lösche Ursachenregister
lw $v0, s1 # Rückschreiben von $v0
lw $a0, s2 # Rückschreiben von $a0
.set noat
move $at, $k1 # Restore $at
.set at
rfe # Restaurieren von Statusregistern
mfc0 $k0, $14 # Hole alten Wert des PC
addiu $k0, $k0, 4 # Erhöhe um 4
jr $k0 # Führe unterbrochenes Programm fort
```