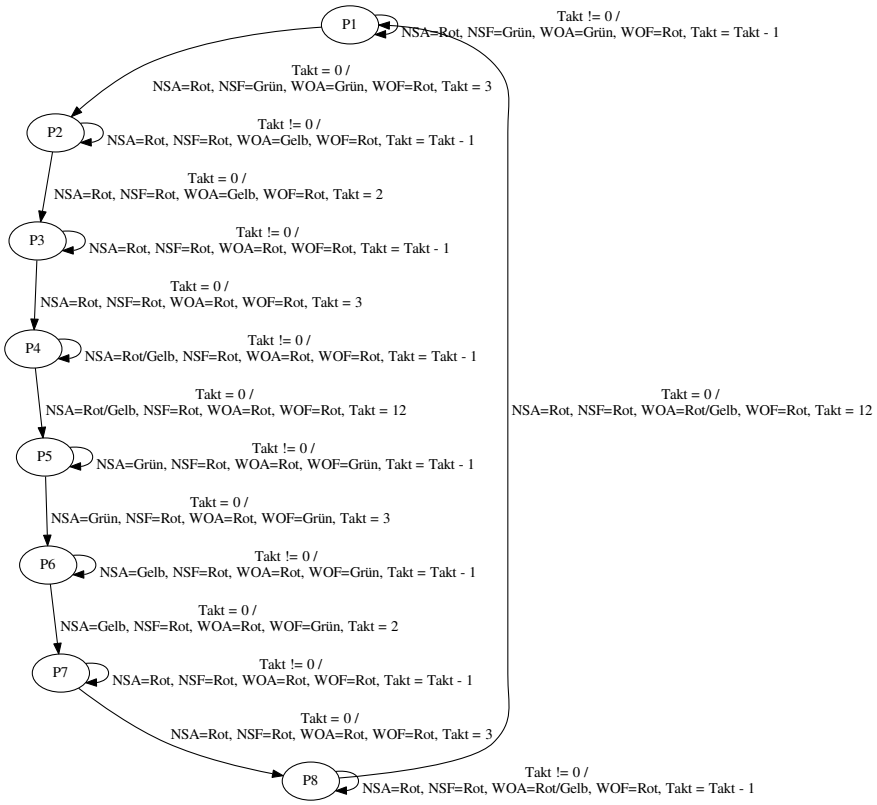
Grundlagen der Rechnerarchitektur

Tim Luchterhand, Paul Nykiel (Abgabegruppe 117)

12. Februar 2019

1 Automat



2 Implementierung

Hinweis: Die Ausgabe wurde bereits in Aufgabe 2 implementiert. Es gibt kein zusätzliches File für Aufgabe 3.

```
# Strings
.data
redS: .asciiz "\n\tRot: "
greenS: .asciiz "\n\tGruen: "
yellowS: .asciiz "\n\tGelb: "
nsa: .asciiz "\nNord-Sued Auto: "
nsf: .asciiz "\nNord-Sued Fussgaenger: "
woa: .asciiz "\nWest-Ost Auto: "
wof: .asciiz "\nWest-Ost Fussgaenger: "
red: .word 1
yellow: .word 2
red_yellow: .word 3
green: .word 4

.text

main:
    # 1. Phase: NSA=Rot, NSF=Gruen, WOA=Gruen, WOF=Rot, 12 Taktzyklen;
    lw $a0, red
    lw $a1, green
    lw $a2, green
    lw $a3, red
    jal set_ampel
    jal print
    li $a0, 12
    jal delay_ticks
    # 2. Phase: NSA=Rot, NSF=Rot, WOA=Gelb, WOF=Rot, 3 Taktzyklen;
    lw $a0, red
    lw $a1, red
    lw $a2, yellow
    lw $a3, red
    jal set_ampel
    jal print
    li $a0, 3
    jal delay_ticks
    # 3. Phase: NSA=Rot, NSF=Rot, WOA=Rot, WOF=Rot, 2 Taktzyklen;
    lw $a0, red
    lw $a1, red
    lw $a2, red
    lw $a3, red
    jal set_ampel
    li $a0, 2
    jal delay_ticks
    # 4. Phase: NSA=Rot/Gelb, NSF=Rot, WOA=Rot, WOF=Rot, 3 Taktzyklen;
    lw $a0, red_yellow
    lw $a1, red
    lw $a2, red
    lw $a3, red
```

```
49      jal set_ampel
50      jal print
51      li $a0, 3
52      jal delay_ticks
53      # 5. Phase: NSA=Gruen, NSF=Rot, WOA=Rot, WOF=Gruen, 12 Taktzyklen;
54      lw $a0, green
55      lw $a1, red
56      lw $a2, red
57      lw $a3, green
58      jal set_ampel
59      jal print
60      li $a0, 12
61      jal delay_ticks
62      # 6. Phase: NSA=Gelb, NSF=Rot, WOA=Rot, WOF=Rot, 3 Taktzyklen;
63      lw $a0, yellow
64      lw $a1, red
65      lw $a2, red
66      lw $a3, red
67      jal set_ampel
68      jal print
69      li $a0, 3
70      jal delay_ticks
71      # 7. Phase: NSA=Rot, NSF=Rot, WOA=Rot, WOF=Rot, 2 Taktzyklen;
72      lw $a0, red
73      lw $a1, red
74      lw $a2, red
75      lw $a3, red
76      jal set_ampel
77      jal print
78      li $a0, 2
79      jal delay_ticks
80      # 8. Phase: NSA=Rot, NSF=Rot, WOA=Rot/Gelb WOF=Rot, 3 Taktzyklen;
81      lw $a0, red
82      lw $a1, red
83      lw $a2, red_yellow
84      lw $a3, red
85      jal set_ampel
86      jal print
87      li $a0, 3
88      jal delay_ticks
89
90      j main
91
92      addi    $v0,$zero,10    #exit program
93      syscall
94
95  # Argumente: $a0: Anzahl der Ticks
96  delay_ticks:
97      addiu $sp, $sp, -4
98      sw $ra, 0($sp)
99      li $v0, 32 # sleep syscall
100     li $t0, 100
101     mul $a0, $a0, $t0 # delay = 1000 * ticks (one second per tick)
102     syscall
```

```
103     lw $ra, 0($sp)
104     addiu $sp, $sp, 4
105     jr $ra
106
107 # Schreibt den Zustand in s0 bis s3
108 # Argumente: $a0: NSA, $a1: NSF, $a2: WOA, $a3: WOF;
109 # Codierung: Rot -> 1, Gelb -> 2, Rot/Gelb -> 3, Gruen -> 4
110 set_ampel:
111     addiu $sp, $sp, -4
112     sw $ra, 0($sp)
113
114     move $s0, $a0
115     move $s1, $a1
116     move $s2, $a2
117     move $s3, $a3
118
119     lw $ra, 0($sp)
120     addiu $sp, $sp, 4
121     jr $ra
122
123 # Gibt den Zustand einer Ampel aus
124 # Argumente: a0 Name der Ampel, a1: Zustand der Ampel
125 print_ampel:
126     addiu $sp, $sp, -4
127     sw $ra, 0($sp)
128
129     # Lampenzustand aus Bits berechnen
130     andi $t0, $a1, 1
131     andi $t1, $a1, 2
132     andi $t2, $a1, 4
133
134     ble $t1, 0, l1
135     li $t1, 1
136
137 l1:
138     ble $t2, 0, l2
139     li $t2, 1
140
141 l2:
142
143     li $v0,4 #print(name)
144     syscall
145
146     la $a0, redS
147     addi $v0,$zero,4 #print(redS)
148     syscall
149     move $a0, $t0
150     addi $v0,$zero,1 #print(red)
151     syscall
152
153     la $a0, yellowS
154     addi $v0,$zero,4 #print(yellowS)
155     syscall
156     move $a0, $t1
```

```
157     addi $v0,$zero,1 #print(yellow)
158     syscall
159
160     la $a0, greenS
161     addi $v0,$zero,4 #print(greenS)
162     syscall
163     move $a0, $t2
164     addi $v0,$zero,1 #print(green)
165     syscall
166
167     lw $ra, 0($sp)
168     addiu $sp, $sp, 4
169     jr $ra
170
171
172 # Gibt den Zustand der Ampelanlage aus
173 print:
174     addiu $sp, $sp, -4
175     sw $ra, 0($sp)
176
177     # NSA Ampel
178     la $a0, nsa
179     move $a1, $s0
180     jal print_ampel
181     # NSF Ampel
182     la $a0, nsf
183     move $a1, $s1
184     jal print_ampel
185     # WOA Ampel
186     la $a0, woa
187     move $a1, $s2
188     jal print_ampel
189     # WOF Ampel
190     la $a0, wof
191     move $a1, $s3
192     jal print_ampel
193
194     lw $ra, 0($sp)
195     addiu $sp, $sp, 4
196     jr $ra
```

117_Aufgabe2.asm