Benutzer Dokumentation

Inhaltsverzeichnis

1) Das Mikroprogramm *strtolower_imm*

Das Mikroprogramm *strtolower_imm* hat den Opcode 02, die Register RA und RB sind unerheblich, da die Adresse, an welcher das Wort im Hauptspeicher liegt, durch einen Immediate angegeben wird. Nach der Ausführung sind alle Großbuchstaben ab der Adresse des Immediates bis zum Vorkommen der ersten 0 (0x0000) in Kleinbuchstaben umgewandelt worden.

Befehl	Opcode	RA	*RB*	Beispiel	Konstantenfeld	Ergebnis
strtolower imm	02	egal	egal	0205	3D8E	Alle Großbuchstaben im Hauptspeicher ab der Adresse 3D8E bis zum vorkommen einer 0 (0x0000) wurden in Kleinbuchstaben umgewandelt

2) Das Mikroprogramm *strtolower_[RB]*

Das Mikroprogramm *strtolower_[RB]* hat den Opcode 01 und die Adresse, an welcher das Wort im Hauptspeicher liegt, muss im Register RB liegen. Nach der Ausführung sind alle Großbuchstaben ab der Adresse im Register RB bis zum Vorkommen der ersten 0 (0x0000) in Kleinbuchstaben umgewandelt worden.

Befehl	Opcode	RA	*RB*	Beispiel	Konstantenfeld	Ergebnis
strtolower RB	02	egal	Rx	0205	egal	Alle Großbuchstaben im Hauptspeicher ab der Adresse in RB bis zum vorkommen einer 0 (0x0000) wurden in Kleinbuchstaben umgewandelt

3) Das Mikroprogramm *strtolower* als Folge seperater Mikrobefehle

Das Programm *strtolower* als Folge seperater Mikrobefehle würde sich am besten als eine Unterroutine in einem größeren Programm benutzen lassen. Die Adresse des Wortes im Hauptspeicher wird dazu in das Register RA kopiert und danach die Routine *strtolower* aufgerufen. Als Anmerkung: die beiden Seperaten Mikrobefehle sind bei weiten schneller als die Imlementation als Folge einzelner Anweisungen, was auch aus den verlaufsprotokollen hervorgeht. Es werden hierbei etwa zweieinhalb mal soviele takte benötigt.

Die Routine *strtolower* wurde wie folgt in einzelne Befehle aufgespalten:

```
0001:  move_[RA]_RB          ;Buchstabe an Adresse RA nach RB kopieren
0002:  cmp_imm_RA 0x0000     ;wenn 0, dann ist das Wort zu Ende
0003:  jmpz_imm              ;es wird zum Ende gesprungen
0004:  cmp_imm_RA 0x0041     ;wenn kleiner 41, dann kein Großbuchstabe
0005:  jplt_imm              ;Sprung um Adresse zu erhöhen
0006:  cmp_imm_RA 0x005A     ;wenn grösser 5A, dann kein Großbuchstabe
0007:  jpgt_imm              ;Sprung um Adresse zu erhöhen
0008:  add_imm_RB 0x0001     ;Adresse in RA um 1 erhöhen
0009:  mov_RA_[RB]           ;Umgewandelter Buchstabe wird zurück in den
                             ;Speicher geschrieben
0010:  add_imm_RB 0x0020     ;RA wird um 1 erhöht um den nächsten Buchstaben
                             ;adressieren zu können
0011:  jmp_imm 0x0000        ;auch dieser Buchstabe wird verarbeitet
```

Nach der Ausführung sind alle Großbuchstaben ab der Adresse im Register RA bis zum Vorkommen der ersten 0 (0x0000) in Kleinbuchstaben umgewandelt worden.

Befehl	Opcode	RA	*RB*	Beispiel	Konstantenfeld	Ergebnis
move_[RA]_RB	04	egal	Rx	0405	egal	Alle Großbuchstaben im Hauptspeicher ab der Adresse in RB bis zum vorkommen einer 0 (0x0000) wurden in Kleinbuchstaben umgewandelt
cmp_imm_RA	08	Rx	egal	0805	3D8E	RA wird mit 3D8E verglichen und die entsprechenden Flags werden gesetzt
jmpz_imm	06	egal	egal	0605	3D8E	Sollte ein vorhergehender Vergleich das Zero Flag gesetzt haben, so wird an Adresse 3D8E Gesprungen
jplt_imm	07	egal	egal	0705	3D8E	Sollte ein vorhergehender Vergleich das Carry Flag nicht gesetzt haben, so wird an Adresse 3D8E Gesprungen
jpgt_imm	01	egal	egal	0105	3D8E	Sollte ein vorhergehender Vergleich das Carry Flag und das Zero Flag nicht gesetzt haben, so wird an Adresse 3D8E Gesprungen
jmp_imm	02	egal	egal	0205	3D8E	Es wird an Adresse 3D8E Gesprungen
mov_RA_[RB]	05	Rx	Rx	0505	egal	Der Inhalt von Register R0 wird an die Adresse in RB im Hauptspeicher geschrieben
add_imm_RB	03	egal	Rx	0305	3D8E	Register RB wird um 3D8E erhöht