HSB
Hochschule Bremen
City University of Applied Sciences

Dokumentation BSIS

Befehle und Anmerkungen des *BSIS*

Modul Computerarchitektur

Prüfung Labor 3

Dozent Dr. Alexander Förster

Autoren Benjamin Bissendorf (5131381)
Simon Pfennig (5128655)

Datum 15.06.2023

Inhaltsverzeichnis

Einführung

Allgemein

In Verwendung unserer Namen entstand das Ben-Simon-Instruction-Set (BSIS) im Labor 3 des Moduls COMARCH. Dieses basiert auf den Vorgaben von Dr. Alexander Förster.

Erwartetes Befehlsformat

Ein Programm besteht aus einem sequentiell eingelesenen Strom an Bytes. Jeder Befehl des Programms wird durch ein Byte repräsentiert, dessen Wert dem OpCode des entsprechenden Befehls entspricht. Hat ein Befehl einen Operanden, wird angenommen, dass der Wert des Operanten sich im nächsten Byte im Speicher befindet.

Zero-Register

Der Befehl “BNZ” ist abhängig vom Zustand des Zero-Registers. In diesem Register wird gespeichert, ob das Ergebnis des letzten ausgeführten Befehl den Wert 0 hatte. Das Register bleibt solange unverändert, bis es durch einen Befehl überschrieben wird. Der folgenden Befehlstabelle ist zu entnehmen, ob ein Befehl den Wert des Registers verändern kann, ob er es unverändert lässt.

Befehlssatz

Mnm	OpCode	Operand	Länge	ZR*	Anmerkung
NOP	0x01	-	1 Byte	Nein	No Operation
HALT	0x1F	-	1 Byte	Nein	Stoppt die weitere Ausführung
INA	0x02	-	1 Byte	Ja	Lädt den Input in *A*
INB	0x0D	-	1 Byte	Ja	Lädt den Input in *B*
OUTA	0x03	-	1 Byte	Ja	Schreibt den Inhalt von *A* auf *OUTPUT*
OUTB	0x0E	-	1 Byte	Ja	Schreibt den Inhalt von *B* auf *OUTPUT*
MBA	0x0B	-	1 Byte	Ja	Verschiebt Inhalt von *A* nach *B*
MAB	0x10	-	1 Byte	Ja	Verschiebt Inhalt von *B* nach *A*
SET	0x07	#VAL	2 Byte	Ja	Lädt Konstante #VAL in *A*
SW	0x18	#ADDR	2 Byte	Nein	Schreibt *OUTPUT* in den RAM an #ADDR
LW	0x1E	#ADDR	2 Byte	Ja	Liest RAM an #ADDR in A ein
INCA	0x0A	-	1 Byte	Ja	Inkrementiert *A*
INCB	0x0F	-	1 Byte	Ja	Inkrementiert *B*
ADD	0x0C	-	1 Byte	Ja	*A* = *A* + *B*
SUB	0x13	-	1 Byte	Ja	*A* = *A* - *B*
AND	0x12	-	1 Byte	Ja	*A* = *A* & *B* (Bitweises UND)
OR	0x11	-	1 Byte	Ja	*A* = *A* | *B* (Bitweises ODER)
JMP	0x04	#ADDR	2 Byte	Nein	Springt zur absoluten Adresse #ADDR
BNZ	0x14	#ADDR	2 Byte	Nein	Springt zur absoluten Adresse #ADDR, wenn das Zero-Register nicht den Wert 0 hat

*Wenn Ja, wird das Zero-Register durch das Ergebnis des Befehls neu gesetzt

Microcode

Die folgende Tabelle beschreibt den Mikroprogrammspeicher:

Name	ADDR	ADDR HEX	Beschreibung	EN_A	EN_B	EN_MBR	EN_PC	EN_OUT	EN_ZERO	SIZ	BMUX	ADDR_MUX	EN_RAM	ALU_OP	Next	Next Name	JMPC_MUX	Code	Code Hex	Takte
FETCH	00000	00	Lädt nächsten Befehl vom MBR	0	0	1	0	0	0	0	00	0	0	000	11111	-	1	00001111110000010000	0FC10	1
NOP	00001	01	Inkrementiert PC und nimmt nächstes Mikroprogramm von	0	0	0	1	0	0	0	10	0	0	011	00000	FETCH	0	00000000000110001010	0018A	2
INA	00010	02	Lade Input in Register A	1	0	0	0	0	1	0	11	0	0	001	00001	NOP	0	00100000100011000011	208C3	3
OUTA	00011	03	Lade Register A in Output	0	0	0	0	1	1	0	00	0	0	000	00001	NOP	0	00100000100000000100	20804	3
JMP	00100	04	Inkrementiert PC	0	0	0	1	0	0	0	10	0	0	011	00101	JMP_2	0	00000010100110001010	0298A	4
JMP_2	00101	05	Speichere Daten von External Memory in MBR	0	0	1	0	0	0	0	00	0	0	000	00110	JMP_3	0	00000011000000010000	03010	3
JMP_3	00110	06	Schreibe MBR in PC	0	0	0	1	0	0	0	01	0	0	001	00000	FETCH	0	00000000000010001001	00089	2
SET	00111	07	Inkrementiert PC	0	0	0	1	0	1	0	10	0	0	011	01000	SET_2	0	00100100000110001010	2418A	5
SET_2	01000	08	Speichere Daten von PC in Register MBR	0	0	1	0	0	1	0	00	0	0	000	01001	SET_3	0	00100100100000010000	24810	4
SET_3	01001	09	Schreibe Daten von MBR in A	1	0	0	0	0	1	0	01	0	0	001	00001	NOP	0	00100000100011000001	208C1	3
INCA	01010	0A	Inkrementiert A	1	0	0	0	0	1	0	00	0	0	010	00001	NOP	0	00100000100101000000	20940	3
MBA	01011	0B	Verschiebt Inhalt von A nach B	0	1	0	0	0	1	0	00	0	0	000	00001	NOP	0	00100000100000100000	20820	3
ADD	01100	0C	Addiert A und B und speichert Ergebnis in A	1	0	0	0	0	1	0	00	0	0	100	00001	NOP	0	00100000101001000000	20A40	3
INB	01101	0D	Lade Input in Register B	0	1	0	0	0	1	0	11	0	0	001	00001	NOP	0	00100000100010100011	208A3	3
OUTB	01110	0E	Lade Register B in Output	0	0	0	0	1	1	0	00	0	0	001	00001	NOP	0	00100000100010000100	20884	3
INCB	01111	0F	Inkrementiert B	0	1	0	0	0	1	0	00	0	0	011	00001	NOP	0	00100000100110100000	209A0	3
MAB	10000	10	Verschiebt Inhalt von B nach A	1	0	0	0	0	1	0	00	0	0	001	00001	NOP	0	00100000100011000000	208C0	3
OR	10001	11	Bitweise ODER zwischen A und B speichert in A	1	0	0	0	0	1	0	00	0	0	111	00001	NOP	0	00100000101111000000	20BC0	3
AND	10010	12	Bitweise UND zwischen A und B speichert in A	1	0	0	0	0	1	0	00	0	0	110	00001	NOP	0	00100000101101000000	20B40	3
SUB	10011	13	Subtrahiert A von B und speichert Ergebnis in A	1	0	0	0	0	1	0	00	0	0	101	00001	NOP	0	00100000101011000000	20AC0	3
BNZ	10100	14	Inkrementiere PC	0	0	0	1	0	0	0	10	0	0	011	10101	BNZ_2	0	00001010100110001010	0A98A	5
BNZ_2	10101	15	Lade von RAM an der Stelle PC in MBR	0	0	1	0	0	0	0	00	0	0	000	10110	BNZ_3	0	00001011000000010000	0B010	4
BNZ_3	10110	16	Inkrementiere PC	0	0	0	1	0	0	0	10	0	0	011	10111	BNZ_4	0	00001011100110001010	0B98A	3
BNZ_4	10111	17	Wenn Zero-Bit = 1 lade MBR in PC	0	0	0	1	0	0	1	01	0	0	001	00000	FETCH	0	00010000000010001001	10089	2
SW	11000	18	Inkrementiere PC	0	0	0	1	0	0	0	10	0	0	011	11001	SW_2	0	00001100100110001010	0C98A	5
SW_2	11001	19	Lade von RAM an der Stelle PC in MBR	0	0	1	0	0	0	0	00	0	0	000	11010	SW_3	0	00001101000000010000	0D010	4
SW_3	11010	1A	Schreibe Output an Stelle MBR im RAM	0	0	0	0	0	0	0	00	1	1	000	00001	NOP	0	11000000100000000000	C0800	3
LW	11011	1B	Inkrementiere PC, für Quelladdresse	0	0	0	1	0	1	0	10	0	0	011	11100	LW_2	0	00101110000110001010	2E18A	6
LW_2	11100	1C	Lade von RAM an der Stelle PC in MBR (Somit auf Address	0	0	1	0	0	1	0	00	0	0	000	11101	LW_3	0	00101110100000010000	2E810	5
LW_3	11101	1D	Lese RAM auf MBR	0	0	1	0	0	1	0	00	1	0	000	11110	LW_4	0	10101111000000010000	AF010	4
LW_4	11110	1E	Schreibe MBR in A	1	0	0	0	0	1	0	01	0	0	001	00001	NOP	0	00100000100011000001	208C1	3
HALT	11111	1F	Ende das Programm, indem es immer wieder auf HALT ge	0	0	0	0	0	0	0	00	0	0	000	11111	HALT	0	00001111100000000000	0F800	Unendlich

Beispielprogramme

BNZ Test

Ein Programm, das von 3 auf 0 runterzählt und anschließend wieder von Vorne anfängt.

```
SET #1        ; Speichert Konstante #1 in A, zum dekrementieren
MBA           ; Verschiebt Wert zu
SET #3        ; Schreibe #3 in A
OUTA          ; Gib A auf OUTPUT aus
SUB           ; A = A-B, also A-1
BNZ #5        ; Springe zu #5 (OUTA), wenn A noch nicht 0
OUTA          ; (A ist 0), Gibt ein letztes Mal A auf OUTPUT aus
JMP #0        ; Springt zu #0 und fängt von Vorne an
```

Programm (12B): `07 01 0B 07 03 03 13 14 05 03 04 00`

Fibonacci

Das Programm initialisiert A und B mit dem Wert 1. Anschließend geht es in einer Schleife durch und addiert A und B, und tauscht ihre Werte miteinander über den RAM, wonach es wieder mit dem Addieren anfängt.

```
SET #1        ; Speichert Konstante #1 in A
MBA           ; Kopiert A zu B
ADD           ; Addiert A und B in A
OUTB          ; Gibt B auf OUTPUT aus
SW #18        ; Speichert den OUTPUT an #18
MBA           ; Kopiert A auf B
LW #18        ; Lädt Wert an #18 in A
JMP #3        ; Springt zum ADD
```

Programm (12B): `07 01 0B 0C 0E 18 12 0B 1B 12 04 03`