Matthias Buchhorn buchhorn@fhtw-berlin.de

Kurze Übersicht über die Befehle des Z80

Abkürzungen:

r,r'	- Einfachregister	A, B, C, D, E, H, L
dd	- Doppelregister	BC, DE, HL, SP
qq	- Doppelregister	AF, BC, DE, HL
pp	- Doppelregister	BC, DE, SP
n	- 8-Bit-Konstante	
nn	- 16-Bit-Konstante, Adresse	
d	- Verschiebung bei Adressierung über Indexregister, im Bereich -128 ≤ d ≤ 127	
b	- Bit, das in den Einzelbitbefehlen behandelt werden soll 0 ≤ b ≤ 7	
m,M	- Inhalt der 8-Bit-Speicherstelle, die durch HL adressiert wird (L enthält Bits 0-7 ; H Bits 8-15)	
p	- einer der Werte 00h, 08h, 10h, 18h, 20h, 28h, 30h, 38h	
CY	- Carry-Flag	
T	- Anzahl der Taktzyklen des Befehls	

Bei Befehlen, die Flag-Bedingungen für Programmsprünge auswerten, sind 2 Taktzyklen angegeben. Die erste Zahl bezieht sich auf den Fall, daß der Sprung ausgeführt, die zweite dafür, daß kein Sprung ausgeführt wird und am nächsten im Speicher stehenden Befehl weitergearbeitet wird. Die Beeinflussung der Flags durch die Befehle wird hinten beschrieben.

8-Bit-Ladebefehle:

Die Ladebefehle transportieren 8-Bit-Daten intern zwischen Registern oder zwischen Registern und dem Speicher. Dabei steht im Operandenfeld als erstes der Zielspeicherplatz und danach (durch Komma getrennt) der Quellenspeicherplatz. Der Inhalt des Quellenspeicherplatzes wird bei diesen Befehlen nicht verändert.

Mnemonik	T	Wirkungsweise des Befehls	SZHPNC V
LD r,r'	4	der Inhalt des Registers r' wird in das Register r geladen	------
LD r,n	7	die 8-Bit-Konstante n wird im Register r gespeichert	------
LD r,m	7	der Inhalt des durch HL adressierten Speicherplatzes wird in das Register r geladen	------
LD r,(IX+d)	19	der Inhalt des durch IX (oder IY) plus Verschiebung d adressierten Speicherplatzes wird in das Register r geladen	------
LD r,(IY+d)	19		
LD m,r	7	der Inhalt des Registers r wird in den durch HL adressierten Speicherplatz geladen	------
LD (IX+d),r	19	der Inhalt des Registers r wird in den durch IX (oder IY) plus Verschiebung d adressierten Speicherplatz geladen	------
LD (IY+d),r	19		
LD m,n	10	die Konstante n wird in den durch HL adressierten Speicherplatz geladen	------
LD (IX+d),n	19	die Konstante n wird in den durch IX (oder IY) plus Verschiebung d adressierten Speicherplatz geladen	------
LD (IY+d),n	19		
LD A,(BC)	7	der Inhalt des durch BC (oder DE) adressierten Speicherplatzes wird in den Akkumulator (A-Register) geladen	------
LD A,(DE)	7		
LD A,(nn)	13	der Inhalt des Speicherplatzes nn wird in den Akkumulator (A-Register) geladen	------
LD (BC),A	7	der Inhalt des Akkumulators (A-Register) wird in den durch BC (oder DE) adressierten Speicherplatz geladen	------
LD (DE),A	7		
LD (nn),A	13	der Inhalt des Akkumulators (A-Register) wird an die Stelle des Speicherplatzes nn geladen	------
LD A,I	9	der Inhalt des Interruptregisters I wird in den Akkumulator (A-Register) geladen	**0F0-
LD A,R	9	der Inhalt des Refreshregisters R wird in den Akkumulator (A-Register) geladen	**0F0-
LD I,A	9	der Inhalt des Akkumulators (A-Register) wird in das Interruptregister I geladen	------
LD R,A	9	der Inhalt des Akkumulators (A-Register) wird in das Refreshregister R geladen	------

16-Bit-Ladebefehle:

Die Ladebefehle transportieren 16-Bit-Daten intern zwischen Registern oder zwischen Registern und dem Speicher. Dabei steht im Operandenfeld als erstes der Zielspeicherplatz und danach (durch Komma getrennt) der Quellenspeicherplatz. Der Inhalt des Quellenspeicherplatzes wird bei diesen Befehlen nicht verändert. Spezielle 16-Bit-befehle sind die PUSH- und POP-Befehle. Mit ihnen werden 16-Bit-Daten aus Doppelregistern in den Kellerspeicher (Stack) gebracht bzw. zurück in die Doppelregister geholt.
Alle 16-Bit-Daten werden grundsätzlich in der Intel-Order (niederwertiges Byte zuerst) gespeichert.

Mnemonik	T	Wirkungsweise des Befehls	SZHPNC V
LD dd,nn	10	die Konstante nn wird in das Doppelregister geladen	------
LD IX,nn	14	die Konstante nn wird in das Indexregister IX (oder IY) geladen	------
LD IY,nn	14		
LD HL,(nn)	16	der Inhalt der Speicherplätze nn und nn+1 wird in das Doppelregister HL geladen (nn→L, nn+1→H)	------
LD pp,(nn)	20	der Inhalt der Speicherplätze nn und nn+1 wird in das Doppelregister pp geladen (nn→niederwertig, nn+1→höherw.)	------
LD IX,(nn)	20	der Inhalt der Speicherplätze nn und nn+1 wird in das Doppelregister IX (oder IY) geladen (nn→X (oder Y),	------
LD IY,(nn)	20	nn+1→I)	
LD (nn),HL	16	der Inhalt des Doppelregisters HL wird an die Adressen nn und nn+1 geladen (L→nn, H→nn+1)	------
LD (nn),pp	20	der Inhalt des Doppelregisters pp wird an die Adressen nn und nn+1 geladen (niederwertig→nn, höherwertig→nn+1)	------
LD (nn),IX	20	der Inhalt des Doppelregisters IX (oder IY) wird an die Adressen nn und nn+1 geladen	------
LD (nn),IY	20	(X (oder Y)→nn, I→nn+1)	
LD SP,HL	6	der Inhalt des Doppelregisters HL wird im Stackpointer SP gespeichert	------
LD SP,IX	10	der Inhalt des Doppelregisters IX (oder IY) wird im Stackpointer SP gespeichert	------
LD SP,IY	10		
PUSH qq	11	der Inhalt des Doppelregisters qq wird im Stack gespeichert DEC SP; LD (SP),H; DEC SP; LD (SP),L	------
PUSH IX	15	der Inhalt des Doppelregisters IX (oder IY) wird im Stack gespeichert DEC SP; LD (SP),I; DEC SP; LD (SP),X	------
PUSH IY	15		
POP qq	10	der letzte Wert im Stack wird in das Doppelregister qq geladen LD L,(SP); INC SP; LD H,(SP); INC SP	------
POP IX	14	der letzte Wert im Stack wird in das Doppelregister IX (oder IY) geladen LD X,(SP); INC SP; LD I,(SP); INC SP	------
POP IY	14		

Matthias Buchhorn buchhorn@fhtw-berlin.de

8-Bit-Arithmetik und Logikbefehle

Diese Befehle arbeiten mit Daten, die sich im Akkumulator (Register A als ersten Operanden) und mit Daten in anderen Registern oder auf Speicherplätzen (als zweiten Operanden) befinden. Das Ergebnis dieser Operantionen wird im Akkumulator (A-Register) abgelegt.

Mnemonik	**T**	**Wirkungsweise des Befehls**	**SZHPNC V**
ADD r	4	der Inhalt des Registers r wird zum Akkumulatorinhalt addiert	***V0*
ADD m	7	der Inhalt des durch das Register HL adressierten Speicherplatzes m wird zum Akkumulatorinhalt addiert	***V0*
ADD n	7	die Konstante n wird zum Akkumulatorinhalt addiert	***V0*
ADD (IX+d)	19	der Inhalt des durch das Register IX (oder IY) plus Verschiebung adressierten Speicherplatzes wird zum	***V0*
ADD (IY+d)	19	Akkumulatorinhalt addiert	
ADC r	4	der Inhalt des Registers r plus Carry-Flag wird zum Akkumulatorinhalt addiert	***V0*
ADC m	7	der Inhalt des durch HL adressierten Speicherplatzes m plus Carry-Flag wird zum Akkumulatorinhalt addiert	***V0*
ADC n	7	die Konstante n plus Carry-Flag wird zum Akkumulatorinhalt addiert	***V0*
ADC (IX+d)	19	der Inhalt des durch das Register IX (oder IY) plus Verschiebung plus Carry-Flag adressierten Speicherplatzes	***V0*
ADC (IY+d)	19	wird zum Akkumulatorinhalt addiert	
SUB r	4	der Inhalt des Registers r wird vom Akkumulatorinhalt subtrahiert	***V1*
SUB m	7	der Inhalt des durch das Register HL adressierten Speicherplatzes m wird vom Akkumulatorinhalt subtrahiert	***V1*
SUB n	7	die Konstante n wird vom Akkumulatorinhalt subtrahiert	***V1*
SUB (IX+d)	19	der Inhalt des durch das Register IX (oder IY) plus Verschiebung adressierten Speicherplatzes wird vom	***V1*
SUB (IY+d)	19	Akkumulatorinhalt subtrahiert	
SBC r	4	der Inhalt des Registers r plus Carry-Flag wird vom Akkumulatorinhalt subtrahiert	***V1*
SBC m	7	der Inhalt des durch HL adressierten Speicherplatzes m plus Carry-Flag wird vom Akkumulatorinhalt subtrahiert	***V1*
SBC n	7	die Konstante n plus Carry-Flag wird vom Akkumulatorinhalt subtrahiert	***V1*
SBC (IX+d)	19	der Inhalt des durch das Register IX (oder IY) plus Verschiebung plus Carry-Flag adressierten Speicherplatzes	***V1*
SBC (IY+d)	19	wird vom Akkumulatorinhalt subtrahiert	
AND r	4	logische UND-Verknüpfung mit dem Inhalt eines Registers, Speicherbytes oder Konstanten und dem	**1P00
AND m	7	Akkumulatorinhalt	**1P00
AND n	7	es wird bitweise konjunktiv verknüpft (nur dann 1, wenn beide Bits 1 sind)	**1P00
AND (IX+d)	19		**1P00
AND (IY+d)	19		**1P00
OR r	4	logische ODER-Verknüpfung mit dem Inhalt eines Registers, Speicherbytes oder Konstanten und dem	**0P00
OR m	7	Akkumulatorinhalt	**0P00
OR n	7	es wird bitweise disjunktiv verknüpft (nur dann 0, wenn beide Bits 0 sind)	**0P00
OR (IX+d)	19		**0P00
OR (IY+d)	19		**0P00
XOR r	4	Exclusiv-ODER-Verknüpfung mit dem Inhalt eines Registers, Speicherbytes oder Konstanten und dem	**0P00
XOR m	7	Akkumulatorinhalt	**0P00
XOR n	7	es wird bitweise exklusiv verknüpft (nur dann 1, wenn ein Bit 0 und das andere 1 ist)	**0P00
XOR (IX+d)	19		**0P00
XOR (IY+d)	19		**0P00
CP r	4	Vergleich vom Inhalt eines Registers, Speicherbytes oder Konstanten mit dem Akkumulatorinhalt	***V1*
CP m	7	Zero-Flag: 1 → beide Inhalte sind gleich; 0 → Inhalte sind unterschiedlich	***V1*
CP n	7	Carry-Flag: 1 → Akkumulatorinhalt ist kleiner zweitem Operanden; 0 → Akkumulator ist größer oder gleich	***V1*
CP (IX+d)	19	zweiten Operanden	***V1*
CP (IY+d)	19		***V1*
INC r	4	der Inhalt des Registers r wird um eins erhöht	***V0-
INC m	11	der Inhalt des durch HL adressierten Speicherplatzes m wird um eins erhöht	***V0-
INC (IX+d)	23	der Inhalt des durch das Register IX (oder IY) plus Verschiebung adressierten Speicherplatzes wird um eins	***V0-
INC (IY+d)	23	erhöht	
DEC r	4	der Inhalt des Registers r wird um eins vermindert	***V1-
DEC m	11	der Inhalt des durch HL adressierten Speicherplatzes m wird um eins vermindert	***V1-
DEC (IX+d)	23	der Inhalt des durch das Register IX (oder IY) plus Verschiebung adressierten Speicherplatzes wird um eins	***V1-
DEC (IY+d)	23	vermindert	
DAA	4	korrigiert nach Addition / Subtraktion zweier gepackter BCD-Zahlen den Akkumulatorinhalt so, daß im Akkumulator wieder die gepackte BCD-Darstellung erreicht wird	***P-*
CPL	4	bitweises Negieren (Komplementieren) des Akkumulatorinhalts (1er-Komplement)	--1-1-
NEG	8	subtrahieren des Akkumulatorinhalts von Null (2er-Komplement, bitweises Negieren aller Bits, dann um 1 erhöhen)	***V1*
CCF	4	Komplementieren des Carry-Flags	--x-0*
SCF	4	Setzen des Carry-Flags auf 1	--0-01

Matthias Buchhorn buchhorn@fhtw-berlin.de

16-Bit-Arithmetikbefehle

Diese Befehle arbeiten ähnlich den 8-Bit-Arithmetikbefehlen, jedoch mit Doppelregistern. Als “Akkumulator” wird eines der Doppelregister HL, IX oder IY benutzt.

Mnemonik	**T**	**Wirkungsweise des Befehls**	**SZHPNC V**
ADD HL,dd	11	der Inhalt des Registerspaares dd wird zum Inhalt des Registerpaares HL addiert	--x-0*
ADD IX,IX	15	der Inhalt des Registerspaares IX (oder IY) wird mit sich selbst addiert (Verdopplung)	--x-0*
ADD IY,IY	15		
ADD IX,pp	15	der Inhalt von pp wird zum Inhalt des Registerpaares IX (oder IY) addiert	--x-0*
ADD IY,pp	15		
ADC HL,dd	15	der Inhalt des Registerspaares dd plus Carry-Flag wird zum Inhalt des Registerpaares HL addiert	**xV0*
SBC HL,dd	15	der Inhalt des Registerspaares dd plus Carry-Flag wird vom Inhalt des Registerpaares HL subtrahiert	**xV1*
INC dd	6	der Inhalt des Doppelregisters dd wird um eins erhöht	------
INC IX	10	der Inhalt des Doppelregisters IX (oder IY) wird um eins erhöht	------
INC IY	10		
DEC dd	6	der Inhalt des Doppelregisters dd wird um eins vermindert	------
DEC IX	10	der Inhalt des Doppelregisters IX (oder IY) wird um eins vermindert	------
DEC IY	10		

Registeraustauschbefehle

Diese Befehle dienen dem schnellen Austausch von Doppelregisterinhalten und erschliessen dem Programmierer die Hintergrundregister.

Mnemonik	**T**	**Wirkungsweise des Befehls**	**SZHPNC V**
EX DE,HL	4	Austausch des Inhalts der Doppelregister DE und HL	------
EX AF,AF’	4	Austausch des Inhalts der Doppelregister AF und AF’	------
EXX	4	Austausch des Inhalts der Doppelregister BC ⇔ BC’ DE ⇔ DE’ HL ⇔ HL’	------
EX (SP),HL	19	Austausch des Inhalts des Doppelregisters mit dem letzten Wert im Stack (SP+1) ⇔ H (SP) ⇔ L	------
EX (SP),IX EX (SP),IY	23 23	Austausch des Inhalts des Doppelregisters mit dem letzten Wert im Stack (SP+1) ⇔ I (SP) ⇔ X (oder Y)	------

Programmverzweigungsbefehle

Es muß zwischen unbedingten und bedingten Sprüngen unterschieden werden. Weiterhin sind absolute und relative Sprünge mögich. Die relativen Sprünge sind nur in einer Umgebung von -126 bis +129 Byte möglich. Bei bedingten Sprüngen sind die Flag-Bedngungen als Operanden anzugeben und es werden die entsprechenden Flag-Bits getestet. In Abhängigkeit von diesem Test wird der Sprung ausgeführt oder ignoriert.

Mnemonik	**T**	**Wirkungsweise des Befehls**	**SZHPNC V**
JP nn	10	unbedingter Sprung nach Adresse nn	------
JP NZ,nn	10	Sprung nach Adresse nn, wenn das Zero-Flag gleich 0 ist	------
JP Z,nn	10	Sprung nach Adresse nn, wenn das Zero-Flag gleich 1 ist	------
JP NC,nn	10	Sprung nach Adresse nn, wenn das Carry-Flag gleich 0 ist	------
JP C,nn	10	Sprung nach Adresse nn, wenn das Carry-Flag gleich 1 ist	------
JP PO,nn	10	Sprung nach Adresse nn, wenn das P/V-Flag gleich 0 ist	------
JP PE,nn	10	Sprung nach Adresse nn, wenn das P/V-Flag gleich 1 ist	------
JP P,nn	10	Sprung nach Adresse nn, wenn das Sign-Flag gleich 0 ist	------
JP M,nn	10	Sprung nach Adresse nn, wenn das Sign-Flag gleich 1 ist	------
JR nn	12	relativer unbedingter Sprung nach Adresse nn	------
JR NZ,nn	12/7	relativer Sprung nach Adresse nn, wenn das Zero-Flag gleich 0 ist	------
JR Z,nn	12/7	relativer Sprung nach Adresse nn, wenn das Zero-Flag gleich 1 ist	------
JR NC,nn	12/7	relativer Sprung nach Adresse nn, wenn das Carry-Flag gleich 0 ist	------
JR C,nn	12/7	relativer Sprung nach Adresse nn, wenn das Carry-Flag gleich 1 ist	------
JP m	4	unbedingter Sprung zu der Adresse, die im Doppelregister HL steht	------
JP (IX) JP (IY)	8 8	unbedingter Sprung zu der Adresse, die im Doppelregister IX (oder IY) steht	------
DJNZ nn	13/8	der Inhalt des Registers B wird um eine vermindert, relative bedingter Sprung zur Adresse nn, wenn B <> 0	------

Matthias Buchhorn buchhorn@fhtw-berlin.de

Unterprogrammbefehle

Es ist wie bei den Sprungbefehlen zwischen bedingten und unbedingten Befehlen zu unterscheiden. Der Unterprogrammaufruf erfolgt mit der Speicherung der dem CALL-Befehl folgenden Adresse (Rückkehradresse) auf dem Stack. Wird das Unterprogramm mit dem RET-Befehl abgeschlossen, so wird das Programm ab der Rückkehradresse weiter abgearbeitet, in dem die Adresse vom Stack zurückgeladen wird.

Mnemonik	T	Wirkungsweise des Befehls	SZHPNC V
CALL nn	17	unbedingter Unterprogrammaufruf zur Adresse nn	------
CALL NZ,nn	17/10	Unterprogrammaufruf zur Adresse nn, wenn das Zero-Flag gleich 0 ist	------
CALL Z,nn	17/10	Unterprogrammaufruf zur Adresse nn, wenn das Zero-Flag gleich 1 ist	------
CALL NC,nn	17/10	Unterprogrammaufruf zur Adresse nn, wenn das Carry-Flag gleich 0 ist	------
CALL C,nn	17/10	Unterprogrammaufruf zur Adresse nn, wenn das Carry-Flag gleich 1 ist	------
CALL PO,nn	17/10	Unterprogrammaufruf zur Adresse nn, wenn das P/V-Flag gleich 0 ist	------
CALL PE,nn	17/10	Unterprogrammaufruf zur Adresse nn, wenn das P/V-Flag gleich 1 ist	------
CALL P,nn	17/10	Unterprogrammaufruf zur Adresse nn, wenn das Sign-Flag gleich 0 ist	------
CALL M,nn	17/10	Unterprogrammaufruf zur Adresse nn, wenn das Sign-Flag gleich 1 ist	------
RST p	11	der RST-Befehl ist ein spezieller Unterprogrammaufruf, es sind 8 folgende Restart-Adressen zugelassen: p = 00h,08h,10h,18h,20h,28h,30h,38h der höherwertige Adressteil ist stets Null, ansonsten entspricht der RST-Befehl einem unbedingten Unterprogrammaufruf	------
RET	10	unbedingter Rücksprung aus einem Unterprogramms die Ausführung erfolgt, in dem die Rückkehradresse wie bei einem POP-Befehl aus dem Stack geholt wird und von dieser Adresse weiterbearbeitet wird	------
RET NZ	11/5	bedingter Rücksprung, wenn das Zero-Flag gleich 0 ist	------
RET Z	11/5	bedingter Rücksprung, wenn das Zero-Flag gleich 1 ist	------
RET NC	11/5	bedingter Rücksprung, wenn das Carry-Flag gleich 0 ist	------
RET C	11/5	bedingter Rücksprung, wenn das Carry-Flag gleich 1 ist	------
RET PO	11/5	bedingter Rücksprung, wenn das P/V-Flag gleich 0 ist	------
RET PE	11/5	bedingter Rücksprung, wenn das P/V-Flag gleich 1 ist	------
RET P	11/5	bedingter Rücksprung, wenn das Sign-Flag gleich 0 ist	------
RET M	11/5	bedingter Rücksprung, wenn das Sign-Flag gleich 1 ist	------
RETI	14	Rücksprung aus Interruptbehandlungsroutine (ISR) eines maskierbaren Interrupts	------
RETN	14	Rücksprung aus Interruptbehandlungsroutine (ISR) eines nicht maskierbaren Interrupts	------

Rotations- und Verschiebebefehle

Durch diese Befehle wird die Möglichkeit gegeben, im Akkumulator (A-Register), in einem anderen Register oder in einem Speicherplatz Daten einfach zyklisch (bitweise) zu verschieben. Das aus dem Byte herausgeschobene Bit wird im Carry-Flag abgelegt.

Mnemonik	T	Wirkungsweise des Befehls	SZHPNC V
RLCA	4	Linksrotation des Akkumulatorinhalts um eine Bitposition nach links, Bit 7 wird zum Inhalt von Bit 0	--0-0*
RRCA	4	Rechtsrotation des Akkumulatorinhalts um eine Bitposition nach rechts, Bit 0 wird zum Inhalt von Bit 7	--0-0*
RLA	4	Linksrotation des Akkumulatorinhalts um eine Bitposition nach links durch das Carry-Flag, Bit 7 wird zum Inhalt des Carry-Flags und dessen alter Inhalt zum Bit 0	--0-0*
RRA	4	Rechtsrotation des Akkumulatorinhalts um eine Bitposition nach rechts durch das Carry-Flag, Bit 0 wird zum Inhalt des Carry-Flags und dessen alter Inhalt zum Bit 7	--0-0*
RLC r	8	Linksrotation des Registers oder Speicherbytes um eine Bitposition nach links, Bit 7 wird zum Inhalt von Bit 0	**0P0*
RLC m	15	(analog RLCA)	**0P0*
RLC (IX+d)	23		**0P0*
RLC (IY+d)	23		**0P0*
RRC r	8	Rechtsrotation des Registers oder Speicherbytes um eine Bitposition nach rechts, Bit 0 wird zum Inhalt von Bit 7	**0P0*
RRC m	15		**0P0*
RRC (IX+d)	23	(analog RRCA)	**0P0*
RRC (IY+d)	23		**0P0*
RL r	8	Linksrotation des Registers oder Speicherbytes um eine Bitposition nach links durch das Carry-Flag, Bit 7 wird zum Inhalt des Carry-Flags und dessen alter Inhalt zum Bit 0	**0P0*
RL m	15		**0P0*
RL (IX+d)	23	(analog RLA)	**0P0*
RL (IY+d)	23		**0P0*
RR r	8	Rechtsrotation des Registers oder Speicherbytes um eine Bitposition nach rechts durch das Carry-Flag, Bit 0 wird zum Inhalt des Carry-Flags und dessen alter Inhalt zum Bit 7	**0P0*
RR m	15		**0P0*
RR (IX+d)	23	(analog RRA)	**0P0*
RR (IY+d)	23		**0P0*
SLA r	8	Linksverschiebung eines Registers oder Speicherbytes um ein Bit durch das Carry-Flag. Das Bit 0 wird 0.	**0P0*
SLA m	15		**0P0*
SKA (IX+d)	23		**0P0*
SLA (IY+d)	23		**0P0*
SRA r	8	Rechtsverschiebung eines Registers oder Speicherbytes um ein Bit durch das Carry-Flag. Der Inhalt von Bit 7 bleibt erhalten.	**0P0*
SRA m	15		**0P0*
SRA (IX+d)	23		**0P0*
SRA (IY+d)	23		**0P0*
RLD	18	zyklische Verschiebung nach links zwischen dem Akkumulator und dem Inhalt des durch HL adressiertem Speicherplatzes (m). Die unteren 4 Bit von m werden in die oberen 4 Bitstellen übertragen und diese ihrerseits in die unteren 4 Bits des Akkumulators. Die unteren 4 Bits des Akkumulators werden in die unteren 4 Bitstellen von m transportiert. Die oberen 4 Bits des Akkumulators bleiben unverändert.	**0P0-
RRD	18	zyklische Verschiebung nach rechts zwischen dem Akkumulator und dem Inhalt des durch HL adressiertem Speicherplatzes (m). Die unteren 4 Bit von m werden in die unteren 4 Bitstellen des Akkumulators übertragen und diese ihrerseits in die oberen 4 Bits von m. Die oberen 4 Bits von m werden in die unteren transportiert. Die oberen 4 Bits des Akkumulators bleiben unverändert.	**0P0-

Matthias Buchhorn buchhorn@fhtw-berlin.de

Einzelbitbefehle

Diese Befehle erlauben es, einzelne Bits in Registern oder auf Speicherplätzen zu testen, zu setzen oder zu löschen.

Mnemonik	T	Wirkungsweise des Befehls	SZHPNC V
BIT b,r	8	die durch b gekennzeichnete Bitposition wird in dem Register r getestet. Das Komplement steht im Zero-Flag.	X*1x0-
BIT b,m	12	die durch b gekennzeichnete Bitposition wird in der Speicherstelle m getestet. Das Komplement steht im Zero-Flag.	X*1x0-
BIT b,(IX+d)	20	die durch b gekennzeichnete Bitposition wird in der Speicherstelle getestet. Das Komplement steht im Zero-Flag.	X*1x0-
BIT b,(IY+d)	20		
SET b,r	8	die durch b gekennzeichnete Bitposition wird in dem Register r auf 1 gesetzt.	------
SET b,m	12	die durch b gekennzeichnete Bitposition wird in der Speicherstelle m auf 1 gesetzt.	------
SET b,(IX+d)	20	die durch b gekennzeichnete Bitposition wird in der Speicherstelle auf 1 gesetzt.	------
SET b,(IY+d)	20		
RES b,r	8	die durch b gekennzeichnete Bitposition wird in dem Register r auf 0 zurückgesetzt.	------
RES b,m	12	die durch b gekennzeichnete Bitposition wird in der Speicherstelle m auf 0 zurückgesetzt.	------
RES b,(IX+d)	20	die durch b gekennzeichnete Bitposition wird in der Speicherstelle auf 0 zurückgesetzt.	------
RES b,(IY+d)	20		

CPU-Steuerbefehle

Diese Befehle diesen der Steuerung des Interruptsystems der CPU.

Mnemonik	T	Wirkungsweise des Befehls	SZHPNC V
NOP	4	die CPU führt keine Operation aus, es werden aber Refreshzyklen erzeugt	------
HALT	4	die CPU führt so lang eine Folge von NOP-Befehlen aus, bis ein Interrupt oder RESET an der CPU aktiv wird.	------
DI	4	der maskierbare Interrupt wird durch Rücksetzen der Interruptfreigabe-Flipflops IFF1 und IFF2 gesperrt. Nichtmaskierbare Interrupts werden anerkannt.	------
EI	4	der maskierbare Interrupt wird durch Setzen der Interruptfreigabe-Flipflops IFF1 und IFF2 freigegeben. Während der Ausführung dieses Befehls akzeptiert die CPU keine Interruptanforderungen.	------
IM 0	8	CPU in Interruptmodus 0 bringen	------
IM 1	8	CPU in Interruptmodus 1 bringen	------
IM 2	8	CPU in Interruptmodus 2 bringen	------

Blocktransfer- und -suchbefehle

Mit einem einzigen Befehl können beliebig große Datenmengen im Speicher kopiert werden bzw. es kann in einem Speicherbereich nach einem Datenbyte gesucht werden. Die Suche wird beendet, wenn das Byte gefunden oder das Ende des Speicherbereichs erreicht wurde.

Mnemonik	T	Wirkungsweise des Befehls	SZHPNC V
LDI	16	Kopiert ein Datenbyte von der Speicherstelle, die durch HL adressiert wird, an die Speicherstelle, die durch DE adressiert wird. Die Register DE und HL werden um eins erhöht, das Register BC um eins vermindert. BC = 0 → PV = 0 BC <> 0 → PV = 1	--0*0-
LDIR	21 16	kopiert mehrere Datenbytes durch Ausführung des Befehls LDI Wiederholung des Befehls, bis BC = 0 ist	--000-
LDD	16	Kopiert ein Datenbyte von der Speicherstelle, die durch HL adressiert wird, an die Speicherstelle, die durch DE adressiert wird. Die Register DE, HL und BC werden um eins vermindert. BC = 0 → PV = 0 BC <> 0 → PV = 1	--0*0-
LDDR	21 16	kopiert mehrere Datenbytes durch Ausführung des Befehls LDD Wiederholung des Befehls, bis BC = 0 ist	--000-
CPI	16	Vergleich des Inhalts des durch HL adressierten Speicherplatzes mit dem Inhalt des Akkumulators (A-Register) A = (HL) → Z = 1 A <> (HL) → Z = 0 anschließend wird das Register HL um eins erhöht und das Register BC um eins vermindert BC = 0 → PV = 0 BC <> 0 → PV = 1	****1-
CPIR	21 16	vergleicht mehrere Datenbytes durch Ausführung des Befehls CPI Wiederholung des Befehls, bis BC = 0 oder A = (HL) ist	****1-
CPD	16	Vergleich des Inhalts des durch HL adressierten Speicherplatzes mit dem Inhalt des Akkumulators (A-Register) A = (HL) → Z = 1 A <> (HL) → Z = 0 anschließend wird das Register HL und BC um eins vermindert BC = 0 → PV = 0 BC <> 0 → PV = 1	****1-
CPDR	21 16	vergleicht mehrere Datenbytes durch Ausführung des Befehls CPD Wiederholung des Befehls, bis BC = 0 oder A = (HL) ist	****1-

Matthias Buchhorn buchhorn@fhtw-berlin.de

Ein- und Ausgabebefehle

Mit diesen Befehlen können Datenbytes zwischen Registern oder Speicheradressen und externen Bausteinen ausgetauscht werden. Der externe Baustein wird dabei über eine sogenannte Portadresse (8-Bit-Wert) angesprochen. Diese Portadresse wird je nach Befehl entweder direkt angegeben (als Konstante) oder muß im Register C zur Verfügung stehen. Ähnlich den Blocktransferbefehlen existieren auch hier Befehle für die Datenein- und -ausgabe ganzer Speicherbereiche.
Wird für die Adressierung das Register C benutzt, liegt der Inhalt des Registers B an den höherwertigen 8 Bits des Adressbusses an.

Mnemonik	**T**	**Wirkungsweise des Befehls**	**SZHPNCV**
IN A,(n)	11	die Eingabekanaladresse wird mit der Konstante n eingestellt; Zielregister ist der Akkumulator (n) → A	------
IN r,(C)	12	die Eingabekanaladresse wird indirekt mit dem Register C eingestellt; Zielregister ist r (C) → r	**0P0-
INI	16	die Eingabekanaladresse wird mit der Konstante n eingestellt; Zielregister ist der durch HL adressierte Speicherplatz B wird um eins vermindert und HL um eins erhöht (C) → (HL) B-1 → B B=0? → Z=1 sonst Z=0 HL+1 → HL	x*xx1-
INIR	21 16	wiederholte Ausführung des Befehls INI, bis das Register B gleich Null ist	x1xx1-
IND	16	die Eingabekanaladresse wird mit der Konstante n eingestellt; Zielregister ist der durch HL adressierte Speicherplatz B und HL wird um eins vermindert (C) → (HL) B-1 → B B=0? → Z=1 sonst Z=0 HL-1 → HL	x*xx1-
INDR	21 16	wiederholte Ausführung des Befehls IND, bis das Register B gleich Null ist	x1xx1-
OUT (n),A	11	die Ausgabekanaladresse wird mit der Konstante n eingestellt; Quellregister ist der Akkumulator A → (n)	------
OUT (C),r	12	die Ausgabekanaladresse wird indirekt mit dem Register C eingestellt; Quellregister ist r r → (C)	------
OUTI	16	die Ausgabekanaladresse wird mit der Konstante n eingestellt; Quellregister ist der durch HL adressierte Speicherplatz; B wird um eins vermindert und HL um eins erhöht (HL) → (C) B-1 → B B=0? → Z=1 sonst Z=0 HL+1 → HL	x*xx1-
OTIR	21 16	wiederholte Ausführung des Befehls OUTI, bis das Register B gleich Null ist	x1xx1-
OUTD	16	die Ausgabekanaladresse wird mit der Konstante n eingestellt; Quellregister ist der durch HL adressierte Speicherplatz; B und HL wird um eins vermindert (HL) → (C) B-1 → B B=0? → Z=1 sonst Z=0 HL-1 → HL	x*xx1-
OTDR	21 16	wiederholte Ausführung des Befehls OUTD, bis das Register B gleich Null ist	x1xx1-

Matthias Buchhorn buchhorn@fhtw-berlin.de

Übersicht über die Befehlscodefolgen der Befehle des Z80

8-Bit-Ladebefehle

	A	**B**	**C**	**D**	**E**	**H**	**L**	**(HL)**	**(BC)**	**(DE)**	**(nn)**	**n**
LD A,.	*7F*	78	79	7A	7B	7C	7D	7E	0A	1A	3A....	3E..
LD B,.	47	*40*	41	42	43	44	45	46				06..
LD C,.	4F	48	*49*	4A	4B	4C	4D	4E				0E..
LD D,.	57	50	51	*52*	53	54	55	56				16..
LD E,.	5F	58	59	5A	*5B*	5C	5D	5E				1E..
LD H,.	67	60	61	62	63	*64*	65	66				26..
LD L,.	6F	68	69	6A	6B	6C	*6D*	6E				2E..
LD (HL),	77	70	71	72	73	74	75					36..
LD (BC),	02											
LD (DE),	12											
LD (nn),	32....											

	A	**B**	**C**	**D**	**E**	**H**	**L**
LD .,(IX+d)	DD7E..	DD46..	DD4e..	DD56..	DD5E..	DD66..	DD6E..
LD .,(IY+d)	FD7E..	FD46..	FD4e..	FD56..	FD5E..	FD66..	FD6E..
LD (IX+d),.	DD77..	DD70..	DD71..	DD72..	DD73..	DD74..	DD75..
LD (IY+d),.	FD77..	FD70..	FD71..	FD72..	FD73..	FD74..	FD75..

LD (IX+d),n DD36.... **LD (IY+d),n** FD36....

		S	**Z**	**H**	**P/V**	**N**	**C**
LD A,I	ED57	*	*	0	F	0	-
LD A,R	ED5F	*	*	0	F	0	-
LD I,A	ED47	-	-	-	-	-	-
LD R,A	ED4F	-	-	-	-	-	-

16-Bit-Ladebefehle

	BC	**DE**	**HL**	**SP**	**IX**	**IY**
LD ..,nn	01....	11....	21....	31....	DD21....	FD21....
LD ..,(nn)	ED4B....	ED5B....	2A....	ED7B....	DD2A....	FD2A....
LD (nn),..	ED43....	ED53....	22....	ED73....	DD22....	FD22....
LD SP,..			F9		DDF9	FDF9

Stackbefehle

	BC	**DE**	**HL**	**AF**	**IX**	**IY**
PUSH ..	C5	D5	E5	F5	DDE5	FDE5
POP ..	C1	D1	E1	F1	DDE1	FDE1

Registeraustauschbefehle

EX (SP),HL	E3	**EX DE,HL**	EB	
EX (SP),IX	DDE3	**EX AF,AF'**	08	*Austausch mit Hintergrundregister*
EX (SP),IY	FDE3	**EXX**	D9	*(BC-BC' DE-DE' HL-HL')*

Blocktransfer- und -suchbefehle

		S	**Z**	**H**	**P/V**	**N**	**C**	
LDI	EDA0	-	-	0	*	0	-	*LD (DE),(HL); INC HL; INC DE; DEC BC*
LDIR	EDB0	-	-	0	0	0	-	*wie LDI, wiederholen bis BC=0*
LDD	EDA8	-	-	0	*	0	-	*LD (DE),(HL); DEC HL; DEC DE; DEC BC*
LDDR	EDB8	-	-	0	0	0	-	*wie LDD, wiederholen bis BC=0*
CPI	EDA1	*	*	*	*	1	-	*CP A,(HL); INC HL; DEC BC*
CPIR	EDB1	*	*	*	*	1	-	*wie CPI, wiederholen bis BC=0 oder A=(HL)*
CPD	EDA9	*	*	*	*	1	-	*CP A,(HL); DEC HL; DEC BC*
CPDR	EDB9	*	*	*	*	1	-	*wie CPD, wiederholen bis BC=0 oder A=(HL)*

8-Bit-Arithmetikbefehle

	A	**B**	**C**	**D**	**E**	**H**	**L**	**(HL)**	**n**	**(IX+d)**	**(IY+d)**	**S**	**Z**	**H**	**P/V**	**N**	**C**
ADD .	87	80	81	82	83	84	85	86	C6..	DD86..	FD86..	*	*	*	V	0	*
ADC .	8F	88	89	8A	8B	8C	8D	8E	Ce..	DD8E..	FD8E..	*	*	*	V	0	*
SUB .	97	90	91	92	93	94	95	96	D6..	DD96..	FD96..	*	*	*	V	1	*
SBC .	9F	98	99	9A	9B	9C	9D	9E	DE..	DD9E..	FD9E..	*	*	*	V	1	*
AND .	A7	A0	A1	A2	A3	A4	A5	A6	E6..	DDA6..	FDA6..	*	*	1	P	0	0
XOR .	AF	A8	A9	AA	AB	AC	AD	AE	EE..	DDAE..	FDAE..	*	*	1	P	0	0
OR .	B7	B0	B1	B2	B3	B4	B5	B6	F6..	DDB6..	FDB6..	*	*	1	P	0	0
CP .	BF	B8	B9	BA	BB	BC	BD	BE	FE..	DDBE..	FDBE..	*	*	*	V	1	*
INC .	3C	04	0C	14	1C	24	2C	34		DD34..	FD34..	*	*	*	V	0	-
DEC .	3D	05	0D	15	1D	25	2D	35		DD35..	FD35..	*	*	*	V	1	-

		S	**Z**	**H**	**P/V**	**N**	**C**	
DAA	27	*	*	*	P	-	*	BCD-Korrektur im Akku
CPL	2F	-	-	1	-	1	-	Komplementiere Akku (1er-Komplement)
SCF	37	-	-	0	-	0	1	Setze Carry-Flag
CCF	3F	-	-	x	-	0	*	Komplementiere Carry-Flag
NEG	ED44	*	*	*	V	1	*	Komplementiere Akku (2er-Komplement)

Matthias Buchhorn buchhorn@fhtw-berlin.de

16-Bit-Arithmetikbefehle

	BC	DE	HL	SP	IX	IY	S	Z	H	P/V	N	C
ADD HL,.	09	19	29	39			-	-	x	-	0	*
ADC HL,.	ED4A	ED5A	ED6A	ED7A			*	*	x	V	0	*
SBC HL,.	ED42	ED52	ED62	ED72			*	*	x	V	1	*
ADD IX,.	DD09	DD19		DD39	DD29		-	-	x	-	0	*
ADD IY,.	FD09	FD19		FD39		FD29	_	-	x	-	0	*
INC .	03	13	23	33	DD23	FD23	-	-	-	-	-	-
DEC .	0B	1B	2B	3B	DD2B	FD2B	-	-	-	-	-	-

Sprung- & Unterprogrammbefehle

	Z	NZ	C	NC	PE	PO	M	P
JP	CA....	C2....	DA....	D2....	EA....	E2....	FA....	F2....
CALL	CC....	C4....	DC....	D4....	EC....	E4....	FC....	F4....
RET	C8	C0	D8	D0	E8	E0	F8	F0
JR	28..	20..	38..	30..				

	unbedingt	(HL)	(IX)	(IY)	RST	00	08	10	18	20	28	30	38
JP	C3....	E9	DDE9	FDE9		C7	CF	D7	DF	E7	EF	F7	FF
CALL	CD....												
JR	18..												
RET	C9												

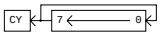
DJNZ	10..	*DEC B; JR NZ,xx*
RETI	ED4D	*zurück vom Interrupt*
RETN	ED45	*zurück vom nicht maskierbaren Interrupt*

Rotations- und Verschiebebefehle

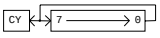
		S	Z	H	P/V	N	C	
RLCA	07	-	-	0	-	0	*	Rotiere Akku links
RRCA	0F	-	-	0	-	0	*	Rotiere Akku rechts
RLA	17	-	-	0	-	0	*	Rotiere Akku links durch Carry
RRA	1F	-	-	0	-	0	*	Rotiere Akku rechts durch Carry
RLD	ED6F	*	*	0	P	0	-	Rotiere Ziffern links zw. Akku und (HL)
RRD	ED67	*	*	0	P	0	-	Rotiere Ziffern rechts zw. Akku und (HL)

	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)
RLC	CB07	CB00	CB01	CB02	CB03	CB04	CB05	CB06	DDCB..06	FDCB..06
RRC	CB0F	CB08	CB09	CB0A	CB0B	CB0C	CB0D	CB0E	DDCB..0E	FDCB..0E
RL	CB17	CB10	CB11	CB12	CB13	CB14	CB15	CB16	DDCB..16	FDCB..16
RR	CB1F	CB18	CB19	CB1A	CB1B	CB1C	CB1D	CB1E	DDCB..1E	FDCB..1E
SLA	CB27	CB20	CB21	CB22	CB23	CB24	CB25	CB26	DDCB..26	FDCB..26
SRA	CB2F	CB28	CB29	CB2A	CB2B	CB2C	CB2D	CB2E	DDCB..2E	FDCB..2E
SLS	CB37	CB30	CB31	CB32	CB33	CB34	CB35	CB36	DDCB..36	FDCB..36
SRL	CB3F	CB38	CB39	CB3A	CB3B	CB3C	CB3D	CB3E	DDCB..3E	FDCB..3E

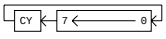
RLC



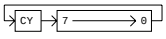
RRC



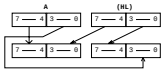
RL



RR

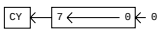


RLD

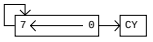


RRD

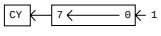
SLA



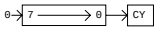
SRA



SLS



SRL



CPU-Steuerbefehle

NOP	00	Leerbefehl
HLT	76	führt keine Operationen mehr aus, bis RESET oder Interrupt
DI	F3	Interrupts sperren
EI	FB	Interrupts erlauben
IM 0	ED46	Interrupt-Modus 0
IM 1	ED56	Interrupt-Modus 1
IM 2	ED5E	Interrupt-Modus 2

Matthias Buchhorn buchhorn@fhtw-berlin.de

Einzelbitbefehle

	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)
BIT 0,.	CB47	CB40	CB41	CB42	CB43	CB44	CB45	CB46	DDCB..46	FDCB..46
BIT 1,.	CB4F	CB48	CB49	CB4A	CB4B	CB4C	CB4D	CB4E	DDCB..4E	FDCB..4E
BIT 2,.	CB57	CB50	CB51	CB52	CB53	CB54	CB55	CB56	DDCB..56	FDCB..56
BIT 3,.	CB5F	CB58	CB59	CB5A	CB5B	CB5C	CB5D	CB5E	DDCB..5E	FDCB..5E
BIT 4,.	CB67	CB60	CB61	CB62	CB63	CB64	CB65	CB66	DDCB..66	FDCB..66
BIT 5,.	CB6F	CB68	CB69	CB6A	CB6B	CB6C	CB6D	CB6E	DDCB..6E	FDCB..6E
BIT 6,.	CB77	CB70	CB71	CB72	CB73	CB74	CB75	CB76	DDCB..76	FDCB..76
BIT 7,.	CB7F	CB78	CB79	CB7A	CB7B	CB7C	CB7D	CB7E	DDCB..7E	FDCB..7E
RES 0,.	CB87	CB80	CB81	CB82	CB83	CB84	CB85	CB86	DDCB..86	FDCB..86
RES 1,.	CB8F	CB88	CB89	CB8A	CB8B	CB8C	CB8D	CB8E	DDCB..8E	FDCB..8E
RES 2,.	CB97	CB90	CB91	CB92	CB93	CB94	CB95	CB96	DDCB..96	FDCB..96
RES 3,.	CB9F	CB98	CB99	CB9A	CB9B	CB9C	CB9D	CB9E	DDCB..9E	FDCB..9E
RES 4,.	CBA7	CBA0	CBA1	CBA2	CBA3	CBA4	CBA5	CBA6	DDCB..A6	FDCB..A6
RES 5,.	CBAF	CBA8	CBA9	CBAA	CBAB	CBAC	CBAD	CBAE	DDCB..AE	FDCB..AE
RES 6,.	CBB7	CBB0	CBB1	CBB2	CBB3	CBB4	CBB5	CBB6	DDCB..B6	FDCB..B6
RES 7,.	CBBF	CBB8	CBB9	CBBA	CBBB	CBBC	CBBD	CBBE	DDCB..BE	FDCB..BE
SET 0,.	CBC7	CBC0	CBC1	CBC2	CBC3	CBC4	CBC5	CBC6	DDCB..C6	FDCB..C6
SET 1,.	CBCF	CBC8	CBC9	CBCA	CBCB	CBCC	CBCD	CBCE	DDCB..CE	FDCB..CE
SET 2,.	CBD7	CBD0	CBD1	CBD2	CBD3	CBD4	CBD5	CBD6	DDCB..D6	FDCB..D6
SET 3,.	CBDF	CBD8	CBD9	CBDA	CBDB	CBDC	CBDD	CBDE	DDCB..DE	FDCB..DE
SET 4,.	CBE7	CBE0	CBE1	CBE2	CBE3	CBE4	CBE5	CBE6	DDCB..E6	FDCB..E6
SET 5,.	CBEF	CBE8	CBE9	CBEA	CBEB	CBEC	CBED	CBEE	DDCB..EE	FDCB..EE
SET 6,.	CBF7	CBF0	CBF1	CBF2	CBF3	CBF4	CBF5	CBF6	DDCB..F6	FDCB..F6
SET 7,.	CBFF	CBF8	CBF9	CBFA	CBFB	CBFC	CBFD	CBFE	DDCB..FE	FDCB..FE

Flagbeeinflussung:

	S	Z	H	P/V	N	C	
BIT	x	*	1	x	0	-	*Komplement des Bits in Z*
RES	-	-	-	-	-	-	*0 in Bit*
SET	-	-	-	-	-	-	*1 in Bit*

Ein- & Ausgabebefehle

	A	B	C	D	E	H	L	(HL)	S	Z	H	P/V	N	C
IN .,(C)	ED78	ED40	ED48	ED50	ED58	ED60	ED68	ED70	*	*	*	P	0	-
OUT (C),.	ED79	ED41	ED49	ED51	ED59	ED61	ED69	ED71	-	-	-	-	-	-

		S	Z	H	P/V	N	C	
IN A,(n)	DB..	-	-	-	-	-	-	
OUT (n),A	D3..	-	-	-	-	-	-	
INI	EDA2	x	*	x	x	1	-	*IN (HL),(C); INC HL; DEC B*
INIR	EDB2	x	1	x	x	1	-	*wie INI, wiederholen bis B=0*
IND	EDAA	x	*	x	x	1	-	*IN (HL),(C); DEC HL; DEC B*
INDR	EDBA	x	1	x	x	1	-	*wie IND, wiederholen bis B=0*
OUTI	EDA3	x	*	x	x	1	-	*OUT (C),(HL); INC HL; DEC B*
OTIR	EDB3	x	1	x	x	1	-	*wie OUTI, wiederholen bis B=0*
OUTD	EDAB	x	*	x	x	1	-	*OUT (C),(HL); DEC HL; DEC B*
OTDR	EDBB	x	1	x	x	1	-	*wie OUTD, wiederholen bis B=0*

Flag-Register

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	S	Z	-	H	-	P/V	N	C

		gesetzt	**nicht gesetzt**	**wird gesetzt bei**
C	Carry-Flag	**C**	**NC**	Übertrag von Bit 7
N	Add-/Subtract-Flag			Subtraktion
P/V	Parity-/Overflow-Flag	**PE**	**PO**	gerader Parität
H	Half-Carry-Flag			Übertrag von Bit 3
Z	Zero-Flag	**Z**	**NZ**	Ergebnis gleich 0
S	Sign-Flag	**M**	**P**	negatives Ergebnis (MSB gesetzt)
-	nicht verwendet			

Beeinflussung:

- **-** unverändert
- **1** gesetzt
- **0** zurückgesetzt
- ***** entsprechend dem Ergebnis der Operation gesetzt (1 wenn erfüllt, 0 nicht erfüllt)
- **x** unbestimmt
- **V** Overflow-Funktion
- **P** Parity-Funktion
- **F** Inhalt des Interrupt-Flipflops IFF2

Matthias Buchhorn buchhorn@fhtw-berlin.de

Z80 PIO

- 2 TTL-kompatible Kanäle zu je 8 Bit, bidirektional
- 8 Ausgänge für direkte Ansteuerung von darlington-Transistoren
- Steuerleitungen für Quittungsbetrieb, Prioritäten und Interrupt
- 4 Betriebsarten

Befehle

– Laden des Interruptvektors, entsprechend Interrupt-Modus des Z80

D7	*D6*	*D5*	*D4*	*D3*	*D2*	*D1*	*D0*	Datenbits
V7	**V6**	**V5**	**V4**	**V3**	**V2**	**V1**	0	Steuerwort

– Festlegung des Interruptsteuerworts

D7	*D6*	*D5*	*D4*	*D3*	*D2*	*D1*	*D0*	Datenbits
0	**0**	**x**	**x**	**1**	**1**	**1**	**1**	Betriebsart 0 (Ausgabe)
0	**1**	**x**	**x**	**1**	**1**	**1**	**1**	Betriebsart 1 (Eingabe)
1	**0**	**x**	**x**	**1**	**1**	**1**	**1**	Betriebsart 2 (bidirektional, nur Kanal A)
1	**1**	**x**	**x**	**1**	**1**	**1**	**1**	Betriebsart 3 (Bit-Steuerung)

x = beliebiger Wert

Modus 0: Strobe-Impuls (ARDY/BRDY auf high) durch PIO, bis Quittung (ASTB/BSTB) durch externe Hardware
Modus 1: Bereitschaftsanzeige (ARDY/BRDY) von PIO, Strobe-Impuls durch externe Hardware (ASTB/BSTB)
Modus 2: Modus nur für Kanal A, Kanal B muß in Modus 3 ohne Interruptbetrieb geschaltet werden
Transfer PIO → ext.Hardware: siehe Modus 0 mit Signalen auf ARDY und ASTB
Transfer ext.Hardware → PIO: siehe Modus 1 mit Signalen auf BRDY und BSTB
Modus 3: die Signale ASTB und ARDY bzw. BSTB und BRDY werden von diesem Kanal nicht benutzt

– Festlegung des Interruptsteuerworts

D7	*D6*	*D5*	*D4*	*D3*	*D2*	*D1*	*D0*	Datenbits
x								0 = Interruptanmeldung nicht erlaubt 1 = Interruptanmeldung möglich
	x							0 = Interruptanmeldung, wenn alle geforderten Bits gesetzt (UND) 1 = Interruptanmeldung, wenn min. 1 gefordertes Bit gesetzt (ODER)
		x						0 = gefordertes Bit gilt als gesetzt, wenn es 0 (low) ist 1 = gefordertes Bit gilt als gesetzt, wenn es 1 (high) ist
			x					0 = es folgt kein weiteres Byte für dieses Steuerwort 1 = es folgt ein weiteres Byte als Interruptmaske
				0	**1**	**1**	**1**	Indikator für dieses Steuerwort

wenn eingestellt, dann wird diese Interruptmaske gesendet, die darüber entscheidet, welche Bits für eine Interruptanmeldung beachtet werden

D7	*D6*	*D5*	*D4*	*D3*	*D2*	*D1*	*D0*	Datenbits
x	**x**	**x**	**x**	**x**	**x**	**x**	**x**	x=0: Bit wird ignoriert, x=1: Bit wird für Interruptanmeldung gefordert

– Festlegung, ob eine Interruptanforderung durch die PIO an den Z80 geschehen darf

D7	*D6*	*D5*	*D4*	*D3*	*D2*	*D1*	*D0*	Datenbits
0	x	x	x	**0**	**0**	**1**	**1**	kein Interruptbetrieb (Polling)
1	x	x	x	**0**	**0**	**1**	**1**	Interruptbetrieb

– Festlegung, welche periphere Kanal-Datenbusleitung als Eingang oder Ausgang bei Betriebsart 3 (Bit-Steuerung) verwendet werden soll

D7	*D6*	*D5*	*D4*	*D3*	*D2*	*D1*	*D0*	Datenbits
0/1	**0/1**	**0/1**	**0/1**	**0/1**	**0/1**	**0/1**	**0/1**	0=Ausgang, 1=Eingang

Der Abschluß einer Interruptroutine (ISR) wird von der PIO, deren Interruptanmeldung vom Z80 akzeptiert wurde, ebenfalls mitempfangen und zur Rückstellung der Interruptprioritätskette benutzt

Signale am IC

B/¬A Kanalauswahl: low = A, high = B
C/¬D Auswahl Steuer-/Datenwort, low = Datenwort, high = Steuerwort bei Schreiboperation
ASTB Kanal A-Strobe (Eingang, low aktiv), wird von Peripherie gesetzt
Modus 0: Quittung der Hardware, daß das Datenbyte übertragen wurde (PIO→Hardware)
Modus 1: Anzeige, das ein gültiges Datenbyte am Eingang anliegt (Hardware→PIO)
Modus 2: Quittung der Hardware, daß das Datenbyte übertragen wurde (PIO→Hardware)
Modus 3: nicht benutzt
ARDY Quittung Kanal A (Ausgang, high aktiv), wird von PIO gesetzt
Modus 0: Anzeige, das ein gültiges Datenbyte am Ausgang anliegt (PIO→Hardware)
Modus 1: Quittung der PIO, daß das Datenbyte gelesen wurde (Hardware→PIO)
Modus 2: Anzeige, das ein gültiges Datenbyte am Ausgang anliegt (PIO→Hardware)
Modus 3: nicht benutzt
BSTB Kanal B-Strobe (Eingang, low aktiv), wird von Peripherie gesetzt
Modus 0: Quittung der Hardware, daß das Datenbyte übertragen wurde (PIO→Hardware)
Modus 1: Anzeige, das ein gültiges Datenbyte am Eingang anliegt (Hardware→PIO)
Modus 2: Quittung, daß das Datenbyte von Kanal A übertragen wurde (Hardware→PIO)
Modus 3: nicht benutzt
BRDY Quittung Kanal B (Ausgang, high aktiv), wird von PIO gesetzt
Modus 0: Anzeige, das ein gültiges Datenbyte am Ausgang anliegt (PIO→Hardware)
Modus 1: Quittung der PIO, daß das Datenbyte gelesen wurde (Hardware→PIO)
Modus 2: Anzeige, das ein gültiges Datenbyte am Eingang an Kanal A anliegt (Hardware→PIO)
Modus 3: nicht benutzt

Matthias Buchhorn buchhorn@fhtw-berlin.de

Z80 CTC

- 4 voneinander unabhängige, programmierbare, wahlweise 8-Bit-Zähler- bzw. 16-Bit-Zeitgeberkanäle
- Vorteiler für jeden Zeitgeberkanal von 16 oder 256
- Ein- und Ausgänge TTL-kompatibel
- Interruptauslösung bei programmierbaren Zähler- oder Zeitgeberwerten
- Vektorinterrupt und Prioritätssteuerung
- Ausgänge ZC/T0 können Darlington-Transistoren treiben

Programierung

– Laden des Interruptvektors, entsprechend Interrupt-Modus des Z80

D7	*D6*	*D5*	*D4*	*D3*	*D2*	*D1*	*D0*	Datenbits
V7	**V6**	**V5**	**V4**	**V3**	x	x	0	Steuerwort

x = beliebiger Wert

– Kanalsteuerwort

D7 D6 D5 D4 D3 D2 D1 D0	Datenbits
x	0 = Interruptanmeldung nicht erlaubt 1 = Interruptanmeldung möglich
x	0 = Betriebsart 16-Bit-Zeitgeber mit Vorteiler 16 oder 256 1 = Betriebsart 8-Bit-Zähler
x	in Betriebsart Zeitgeber, sonst ignoriert: 0 = Vorteiler des Systemtakts durch 16 1 = Vorteiler des Systemtakts durch 256
x	0 = Start durch negative Triggerflanke 1 = Start durch positive Triggerflanke
x	in Betriebsart Zeitgeber, sonst ignoriert: abhängig von Bit 2: Start der Zeitmessung frühestens mit nächstem M1-Zyklus Bit 2=0: 0 = Start sofort 1 = Start nach Laden der Zeitkonstante Bit 2=1: 0 = Start nach Triggerflanke 1 = Start nach Laden der Zeitkonstante und Triggerflanke
x	Zeitkonstante laden 0 = es folgt keine Zeitkonstante 1 = es folgt eine Zeitkonstante für den Rückwärtszähler
x	Rücksetzen 0 = Kanal zählt weiter 1 = Kanal unterbricht, bis eine Zeitkonstante geladen wird, Ausgang und Interruptbetrieb ist solange inaktiv
1	Indikator für dieses Steuerwort

Signale am IC

C/TRG0		Eingang	Takt/Trigger 0, high oder low aktiv, externer Takteingang für Kanal 0
C/TRG1		Eingang	Takt/Trigger 1, analog Kanal 1
C/TRG2		Eingang	Takt/Trigger 2, analog Kanal 2
C/TRG3		Eingang	Takt/Trigger 3, analog Kanal 3
ZC/TC0	0	Ausgang	Nulldurchgang/Zeitgebermeldung, high aktiv für Kanal 0
ZC/TC0	1	Ausgang	Nulldurchgang/Zeitgebermeldung, high aktiv für Kanal 1
ZC/TC0	2	Ausgang	Nulldurchgang/Zeitgebermeldung, high aktiv für Kanal 2 Kanal 3 hat keinen Ausgang und kann nur für den Interruptbetrieb benutzt werden
KS 0		2x	Kanalauswahl 0 bis 3
KS 1		Eingang	