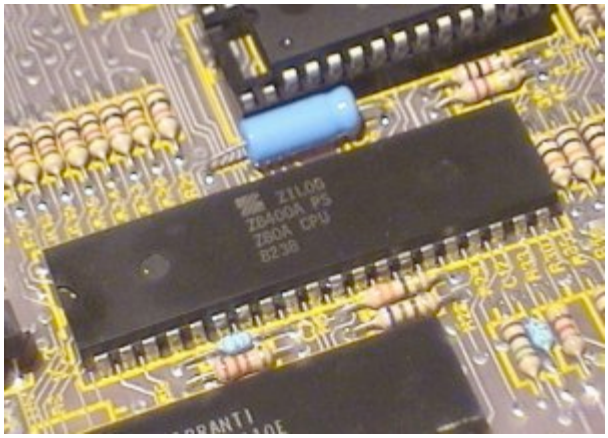
HOMECOMPUTER MUSEUM

- Info
- Sammeln
- Dokumentation
  - Anleitungen
  - Schnittstellen
  - Prozessoren
  - Videochips
  - Soundchips
- Computer
- Videospiele

E-Mail Gästebuch Forum
Englisch Impressum Suche

Zilog Z80

Beschreibung

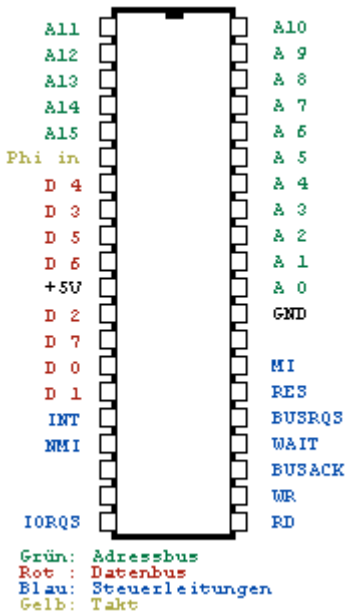


Der Z-80 zählt zu den erfolgreichsten Prozessoren aller Zeiten. Der Hersteller Zilog nahm sich Intels 8080 zum Vorbild und verbesserte ihn sowohl von der Integrierbarkeit in Rechnersysteme als auch vom Maschinenbefehlssatz her. Dadurch entstand ein sehr leistungsfähiger Prozessor, der allerdings wegen seiner ausgesprochen komplexen Befehle auch mit deutlich höherem Takt als sein spartanischerer Hauptkonkurrent 6502 betrieben werden musste, um die gleiche Geschwindigkeit zu entwickeln. Die typische Taktfrequenz lag bei 3,5 - 4 MHz.

Der wichtigste Meilenstein bei der Verbreitung des Z-80 war wohl das Betriebssystem CP/M von Digital Research. Ähnlich wie heute Windows 95/98 nur auf Intel-CPUs und Kompatiblen läuft, war CP/M lange Zeit nur für den Z-80 zu bekommen. Dadurch entstand ein Quasi-Monopol für den Z-80 bei Bürocomputern der späten 70er und frühen 80er Jahre.

Später kamen auch etliche Homecomputer dazu, angefangen vom Sinclair ZX-80, ZX-81 und Spectrum, den Amstrad-Computern bis hin zu den Rechnern des in Europa wenig erfolgreichen MSX-Standards.

Pinbelegung



Der Z-80 freut durch seinen einfachen Aufbau jeden Entwickler, der einen Computer drumherum bauen soll. Der Adressbus ist voll aufgelegt, es wird nur eine Versorgungsspannung benötigt. Beim Steuerbus zeigen sich einige Unterschiede zum großen Konkurrenten 6502: Die Signalisierung von Lese- oder Schreibzugriffen ist durch getrennte Leitungen ausgeführt. Der größte Unterschied besteht jedoch in der Art des Zugriffes auf Peripheriechips:

Während deren Register beim 6502 als normale Speicherstellen angesprochen werden, kann der Z80 über eine eigene Leitung (IORQST) signalisieren, daß er nun auf ein I/O-Register zugreifen möchte. Der Vorteil liegt in der einfacheren Handhabung für Boardentwickler und in der Tatsache, daß der adressierbare Speicher nicht durch I/O-Bereiche zergliedert wird, wie das bei 6502-Rechnern üblich ist.

Mit der Leitung M1 teilt der Z80 mit, daß er nun einen neuen Maschinenbefehl zu lesen beabsichtigt.

Register

A	Akkumulator	Dient bei fast allen Rechenoperationen als Ergebnisspeicher
B,C,D,E,H,L	Register	verschiedene Aufgaben. B wird bei Schleifenbefehlen als Zähler verwendet.

BC,DE,HL	16-Bit-Register	entstehen durch Zusammenfassung von je zwei der Register B-L. BC wird oft als Zählregister verwendet, HL zur indirekten Speicheradressierung
IX,IY	16-Bit-Indexregister	zur indirekten Speicheradressierung, z.B. ADD A,(IX+5): Addiert zu A den Inhalt der Speicherstelle, auf die IX+5 zeigt und hinterlässt das Ergebnis in A.
SP	Stapelzeiger	Zeiger auf die Adresse des obersten Stapelelements. 16-Bit-Register, d.h. der Stapel kann den gesamten Speicher adressieren.
AF',BC',DE',HL'	Sekundäre Register	zweiter Registersatz. AF' heißt Akku + Flags.
I	spez. Aufgaben	(Hardware)
R	spez. Aufgaben	(Hardware)

Zeichenerklärung

Um nicht alle Befehle in allen Adressierungsarten anführen zu müssen (das wären über 600!), wurden folgende Zusammenfassungen verwendet:

x Steht nach einem Befehl "x", so kann er in den folgenden Adressierungsarten verwendet werden:
A, B, C, D, E, H, L: Die normalen Register
n: fester Wert

r Soweit sinnvoll, sind folgende Adressierungsarten möglich:
BC, DE, HL: die normalen Register, paarweise zu 16-Bit-Registern zusammengefasst
(HL): Der Inhalt des HL-Registers wird als Adresse interpretiert und auf diese zugegriffen.
IX, IY: die Indexregister
(IX+d), (IY+d): Zum Inhalt des Indexregisters wird ein fester Wert addiert, das Ergebnis als Adresse interpretiert und auf diese zugegriffen.
nn: fester Wert
(nn): feste Adresse
SP: Stapelzeiger
(SP): Speicherstelle, auf die der Stapelzeiger verweist

n n bestimmt eine Bitnummer, ist also ein Festwert zwischen 0 und 7.

Großbuchstaben stehen für Register.

Hinter jedem Befehl der Befehlsliste stehen die Buchstaben s, z, p und c (oder auch nicht). Diese Buchstaben stehen für die Prozessorflags:

s Sign = Vorzeichen der letzten Operation

z Zero = Nullflag; Wird gesetzt, wenn Ergebnis = 0.

p Parity = Paritätsflag

c Carry = Übertrag; Wird gesetzt, wenn das Ergebnis außerhalb des verfügbaren Bereiches liegt, z.B. $80+$94=$14 mit c=1

Die in der Befehlsliste am Zeilenende stehenden Flags werden durch den entsprechenden Befehl verändert.

1. Arithmetikbefehle

Addition / Subtraktion

ADC A,x	A=A+x+c	Addition mit Übertrag	s	z	p	c
ADC HL,r	HL=HL+r+c	16-Bit-Addition mit Übertrag	s	z	p	c
ADD A,x	A=A+x	Addition ohne Übertrag	s	z	p	c
ADD HL,r	HL=HL+r	16-Bit-Addition ohne Übertrag	s	z	p	c
ADD IX,r	IX=IX+r	16-Bit-Addition im Indexregister IX	s	z	p	c
ADD IY,r	IY=IY+r	16-Bit-Addition im Indexregister IY	s	z	p	c
SBC A,x	A=A-x-c+1	Subtraktion mit Übertrag	s	z	p	c
SBC HL,r	HL=HL-r-c+1	16-Bit-Subtraktion mit Übertrag	s	z	p	c

SUB x	A=A-x	Subtraktion ohne Übertrag	s	z	p	c
INC x	x=x+1	Registerinhalt inkrementieren	s	z	p	c
INC r	r=r+1	16-Bit-Registerinhalt inkrementieren				
DEC x	x=x-1	Registerinhalt dekrementieren	s	z	p	c
DEC r	r=r-1	16-Bit-Registerinhalt dekrementieren				
Binäre Verknüpfungen						
AND x	A = A AND x	UND-Verknüpfung	s	z	p	c=0
CPL	A = A XOR $FF	Komplementbildung	s	z	p	c
OR x	A = A OR x	ODER-Verknüpfung	s	z	p	c=0
XOR x	A = A XOR x	Exklusiv-ODER-Verknüpfung	s	z	p	c=0
Rotier- und Schiebebefehle						
RL x	rotate left x	Bitweise Rotation nach links	s	z	p	c
RLA	rotate left A	Bitweise Rotation nach links				c
RLC x	RL without c	Bitweise Rotation nach links ohne Übertrag	s	z	p	c
RLCA	RLA without c	Bitweise Rotation nach links ohne Übertrag				c
RLD	rotate left decimal (A/HL)	Nibble-Rotation nach links	s	z	p	
RR x	rotate right x	Bitweise Rotation nach rechts	s	z	p	c
RRA	rotate right A	Bitweise Rotation nach rechts				c
RRC x	RR without c	Bitweise Rotation nach rechts ohne Übertrag	s	z	p	c
RRCA	RRA without c	Bitweise Rotation nach rechts ohne Übertrag				c
RRD	rotate right decimal (A/HL)	Nibble-Rotation nach rechts	s	z	p	
SLA x	shift left arith. x	Arithmetisches Linksschieben	s	z	p	c
SRA x	shift right arith. x	Arithmetisches Rechtsschieben	s	z	p	c
SRL x	shift right log. x	Logisches Rechtsschieben	s	z	p	c
Vergleichsbefehle						
CP x	A=x -> z=1 ; A>x -> s=1 ; A<=x -> s=0 ; A<>x -> z=0	Vergleich A mit x, Ergebnis wird über die Flags angezeigt	s	z	p	c
CPD		Vergleich (HL) mit A, dann HL=HL-1, BC=BC-1; wenn BC=0, dann p=0	s	z	p	c
CPDR		wie CPD, jedoch wiederholt bis z=1 oder BC=0	s	z	p	c
CPI		wei CPD, allerdings HL=HL+1	s	z	p	c
CPIR		wei CPDR, allerdings HL=HL+1	s	z	p	c
sonstige Arithmetikbefehle						
BIT n,x		Invertiert Bit n von x und schreibt es ins z-Flag	s	z	p	
CCF	complement carry	c-Flag invertieren				c=-c
SCF	set carry	c-Flag setzen				c=1

SET n,x	set bit n of x	Bit setzen				
RES n,x	reset bit n of x	Bit rücksetzen				
NEG	A = -A	A negieren			p	c
DAA		falsche BCD-Ziffern korrigieren	s	z	p	c

2. Registerbefehle

LD-Befehle

LD x1,x2	x1 = x2	Register kopieren				
LD A,I	A = I	I-Register in Akku kopieren	s	z	p	
LD A,R	A = R	R-Register in Akku kopieren	s	z	p	
LD r,nn	r = nn	16-Bit-Regiter mit festem Wert laden				
LD r,(nn)	r = (nn)	16-Bit-Regiter mit Inhalt einer Speicherstelle laden				
LD I,A	I = A	Akku ins I-Register kopieren				
LD R,A	R = A	Akku ins R-Register kopieren				
LD SP,r	SP = r	Stapelzeiger setzen				

sonstige Registerbefehle

EX AF,AF'	AF <=> AF'	AF und AF' vertauschen				
EX DE,HL	DE <=> HL	DE und HL vertauschen				
EX (SP),HL	(SP) <=> HL	16-Bit-Wert vom Stapel mit Inhalt von HL vertauschen				
EX (SP),IX	(SP) <=> IX	16-Bit-Wert vom Stapel mit Inhalt von IX vertauschen				
EX (SP),IY	(SP) <=> IY	16-Bit-Wert vom Stapel mit Inhalt von IY vertauschen				
EXX	BC <=> BC', DE <=> DE', HL <=> HL'	Register mit Schattenregistern vertauschen				
PUSH r		Inhalt eines 16-Bit-Registers auf Stapel ablegen				
POP r		Inhalt eines 16-Bit-Registers vom Stapel holen				
LDD	(DE)=(HL), HL=HL-1, DE=DE-1, BC=BC-1; wenn BC=0, dann p=0	Dient in Schleife zum Kopieren eines Speicherbereichs			p	
LDDR	(DE)=(HL), HL=HL-1, DE=DE-1, BC=BC-1;	Kopieren eines Speicherbereichs			p=0	

	wiederholen bis BC=0				
LDI	(DE)=(HL), HL=HL+1, DE=DE+1, BC=BC-1; wenn BC=0, dann p=0	Dient in Schleife zum Kopieren eines Speicherbereichs			p
LDIR	(DE)=(HL), HL=HL+1, DE=DE+1, BC=BC-1; wiederholen bis BC=0	Kopieren eines Speicherbereichs			p=0

3. Sprungbefehle

Unbedingte Sprünge

JP nn	PC=nn	Sprung nach nn
JP r	PC=r	Sprung nach Adresse, die in r abgelegt ist
JR nn	PC=PC+nn	relativer Sprung
CALL nn	(SP)=PC; PC=nn	Unterprogrammaufruf
RST nn	PC=nn	Reset; nn=$00,$08,$10,...,$38
RET	PC=(SP)	Rückkehr aus Unterprogramm
RETI	PC=(SP)	Rückkehr aus Interrupt
RETN	PC=(SP)	Rückkehr aus nichtmaskierbarem Interrupt

Bedingte Sprünge

JP bed, nn	Sprung nach nn, wenn Bedingung zutrifft. Bedingungen sind: C (c=1), M (s=1), NC (c=0), NZ (z=0), P (s=0), PE (p=1), PO (p=0), Z (z=1)

Bedingte Sprünge

JR bed, nn	relativer Sprung nach nn, wenn Bedingung zutrifft.
CALL bed, nn	Unterprogrammaufruf, wenn Bedingung zutrifft.
RET bed	Rückkehr aus Unterprogramm, wenn Bedingung zutrifft.
DJNZ nn	B = B-1; relativer Sprung nach nn, wenn B = 0.

4. Ein-/Ausgabebefehle

Eingabe

IN x,(C)	Port, dessen Nr. in Register C steht einlesen und nach x schreiben	s	z	p

IN A,(nn)		Port nn (8 Bit) einlesen und nach A schreiben			
IND		Port C einlesen und nach (HL) schreiben. HL=HL-1; B=B-1		z	
INDR		wie IND, wiederholen bis B=0		z=1	
INI		Port C einlesen und nach (HL) schreiben. HL=HL+1; B=B-1		z	
INIR		wie INI, wiederholen bis B=0		z=1	
Ausgabe					
OUT (C),x		Inhalt von x in den Port, dessen Nr. in Register C steht schreiben			
OUT (nn),A		Inhalt des Akkus in Port nn (8 Bit) schreiben			
OUTD		Inhalt von (HL) nach Port C schreiben. HL=HL-1; B=B-1	s	z	p
OTDR		wie OUTD, wiederholen bis B=0	s	z=1	p
OUTI		Inhalt von (HL) nach Port C schreiben. HL=HL+1; B=B-1	s	z	p
OTIR		wie OUTI, wiederholen bis B=0	s	z=1	p

5. Interruptbefehle

DI	disable irq	Interrupts abschalten
EI	enable irq	Interrupts einschalten
HALT		warten auf Interrupt
IM nn	interrupt mode	Interrupt-Modus nn wählen: nn=0: Befehl auf Datenbus nn=1: Sprung nach $38 nn=2: Sprung nach Adresse auf Datenbus, High-Byte in I
NOP	no operation	Nullbefehl