Polyadisches Zahlensystem

B: Basis des Zahlensystems

$0 \leq a_i < B \qquad a_i \in N_0 \qquad B \in (N > 1)$

Ganze Zahlen:

$$\mathbf{Z = \sum_{i=0}^{n-1} a_i \cdot B^i}$$

$$Z = a_0 \cdot B^0 + a_1 \cdot B^1 + a_2 \cdot B^2 + \ldots\ldots\ldots + a_{n-1} \cdot B^{n-1}$$

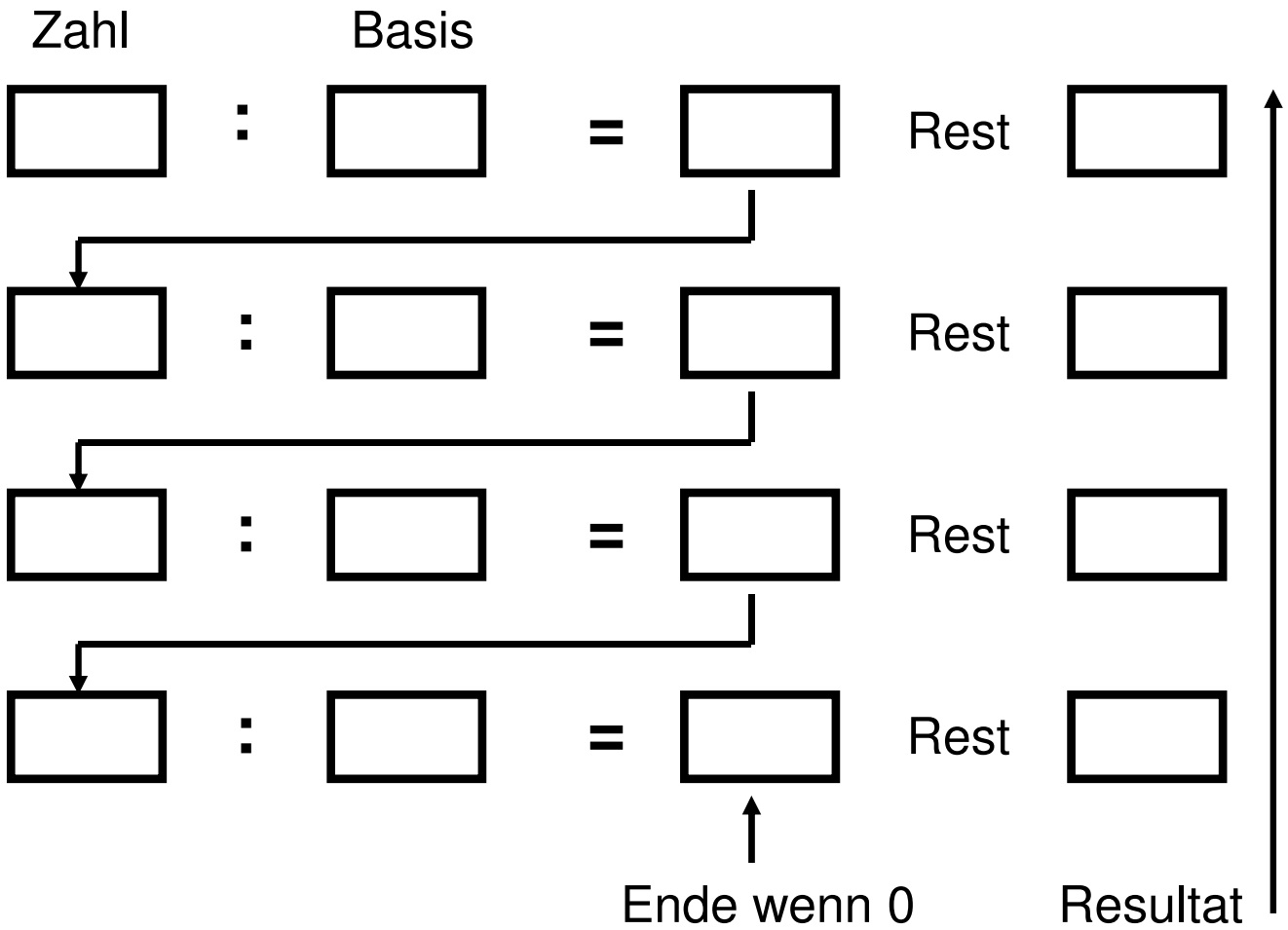
Rationale Zahlen:

$$\mathbf{Z = \sum_{i=-m}^{n-1} a_i \cdot B^i}$$

$$Z = a_{-m} \cdot B^{-m} \mathbf{+ \ldots. +} a_0 \cdot B^0 + a_1 \cdot B^1 + a_2 \cdot B^2 + \ldots\ldots\ldots + a_{n-1} \cdot B^{n-1}$$

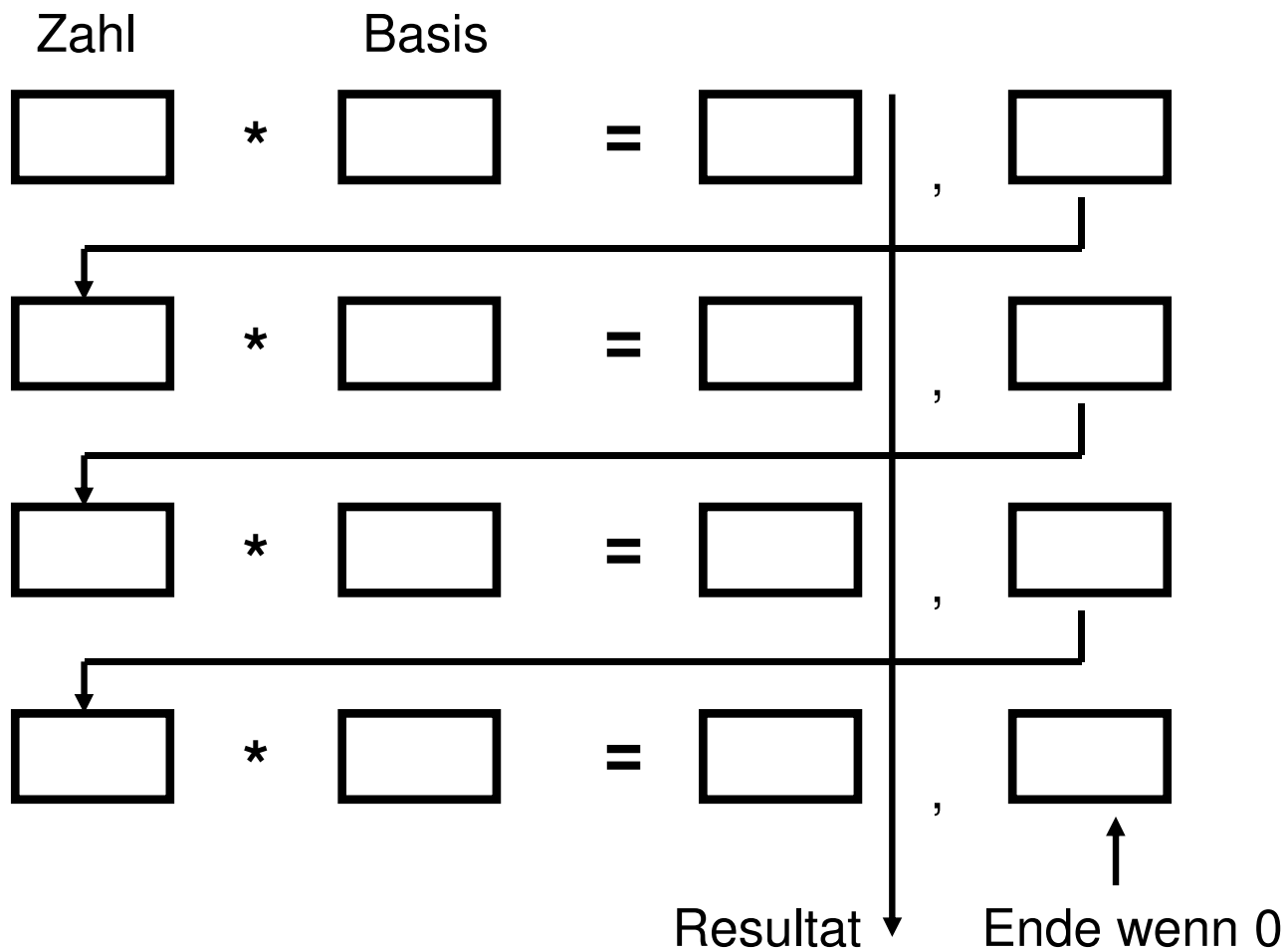
Wandlung einer Dezimalzahl in ein anderes System

Verfahren für ganzzahligen Anteil



Wandlung einer Dezimalzahl in ein anderes System

Verfahren für gebrochenen Anteil



Wichtige Dualzahlen

Oktal-Ziffern (2->8)

000	=	0
001	=	1
010	=	2
011	=	3
100	=	4
101	=	5
110	=	6
111	=	7

Hex-Ziffern (2->16)

0000	=	0
0001	=	1
0010	=	2
0011	=	3
0100	=	4
0101	=	5
0110	=	6
0111	=	7
1000	=	8
1001	=	9
1010	=	A
1011	=	B
1100	=	C
1101	=	D
1110	=	E
1111	=	F

Tabelle der Zweierpotenzen

n	2^n	2^{-n}
0	1	1
1	2	0,5
2	4	0,25
3	8	0,125
4	16	0,0625
5	32	0,03125
6	64	0,015625
7	128	0,0078125
8	256	0,00390625
9	512	0,001953125
10	1.024 = 1K	0,0009765625
11	2.048 = 2K	
12	4.096 = 4K	
13	8.192 = 8K	
14	16.384 = 16K	
15	32.768 = 32K	
16	65.536 = 64K	
17	131.072 = 128K	
18	262.144 = 256K	
19	524.288 = 512K	
20	1.048.576 = 1M	

Gängige Abkürzungen:

10^3 = 1k (Kilo)	2^{10} = 1Ki (Kibi)
10^6 = 1M (Mega)	2^{20} = 1Mi (Mebi)
10^9 = 1G (Giga)	2^{30} = 1Gi (Gibi)
10^{12} = 1T (Tera)	2^{40} = 1Ti (Tebi)
10^{15} = 1P (Peta)	2^{50} = 1Pi (Pebi)
10^{18} = 1E (Exa)	2^{60} = 1Ei (Exbi)
10^{21} = 1Z (Zetta)	2^{70} = 1Zi (Zebi)
10^{24} = 1Y (Yotta)	2^{80} = 1Yi (Yobi)
10^{-3} = 1m (Milli)	2^{-10}
10^{-6} = 1µ (Micro)	2^{-20}
10^{-9} = 1n (Nano)	2^{-30}
10^{-12} = 1p (Pico)	2^{-40}
10^{-15} = 1f (Femto)	2^{-50}
10^{-18} = 1a (Atto)	2^{-60}
10^{-21} = 1z (Zepto)	2^{-70}
10^{-24} = 1y (Yokto)	2^{-80}

Zahlensystem-Darstellung in verschiedenen Sprachen

Sprache	*Dual*	*Oktal*	*Hexadezimal*
Zahlen	$1011\ 0111_2$	267_8	$B7_{16}$
Assembler	1011_0111B	267Q	0B7H
VHDL Variante 1	2#1011_0111#	8#267#	16#B7#
VHDL Variante 2	B"1011_0111"		X"B7"
C		0267	0xB7

Darstellung ganzer Zahlen (ohne Vorzeichen)

3 Bit

000	=	0
001	=	1
010	=	2
011	=	3
100	=	4
101	=	5
110	=	6
111	=	7

8 Bit

0000 0000	=	0
0000 0001	=	1
0000 0010	=	2
0000 0011	=	3
0000 0100	=	4
0000 0101	=	5
0000 0110	=	6
0000 0111	=	7

Darstellung ganzer Zahlen (mit Vorzeichen)

4 Bit

1000	=	-0
1001	=	-1
1010	=	-2
1011	=	-3
1100	=	-4
1101	=	-5
1110	=	-6
1111	=	-7

8 Bit

1000 0000	=	-0
1000 0001	=	-1
1000 0010	=	-2
1000 0011	=	-3
1000 0100	=	-4
1000 0101	=	-5
1000 0110	=	-6
1000 0111	=	-7

Nachteile der Vorzeichendarstellung

1. Die Darstellung negativer Zahlen verändert sich bei Bereichserweiterungen:

- 5 als 4-Bit Zahl =	1101
- 5 als 1-Byte Zahl =	1000 0101
- 5 als 2-Byte Zahl =	1000 0000 0000 0101

2. Die Addition einer positiven und einer negativen Zahl funktioniert anders als üblich:

- 5	1000 0101	
+ 12	+ 0000 1100	
---	---	
7	+ 0000 0111	???

3. Null hat zwei verschiedene Darstellungen:

1000 0000	=	- 0
0000 0000	=	+0

Ziel: Subtraktion durch Addition des Komplements

Komplementbildung:

- Erweiterung einer n-stellige Zahl ohne VZ zu einer n+1-stelligen Zahl mit VZ
- Ergänzung zu B^{n+1}

Weg für Addition und Subtraktion:

- Positive Zahlen um VZ-Stelle mit Null erweitern
- Negative Zahlen komplementieren
- Normale Addition durchführen
- Überlauf auf Stelle n+2 ignorieren
- Negative Zahlen ggf. komplementieren

Überlauferkennung

Überlauf falls, VZ der beiden zu verknüpfenden Zahlen gleich, aber unterschiedlich vom Ergebnis.

Überlauf:

0…	1…
0…	1…
-----	----
1…	0…

Ok:

0…	1…	0…	1…
1…	0…	0…	1…
-----	----	----	----
….	….	0…	1…

Komplementarithmetik (modifizierter Weg)

Weg für Addition und Subtraktion:

- **Positive Zahlen um VZ-Stelle mit Null erweitern**
- **Negative Zahlen ziffernweise zur höchsten Ziffer ergänzen (komplementieren)**
- **Normale Addition durchführen**
- **Falls Überlauf auf Stelle n+2, 1 addieren**
- **Negative Zahlen ggf. ziffernweise zur höchsten Zahl ergänzen (komplementieren)**

Einerkomplement

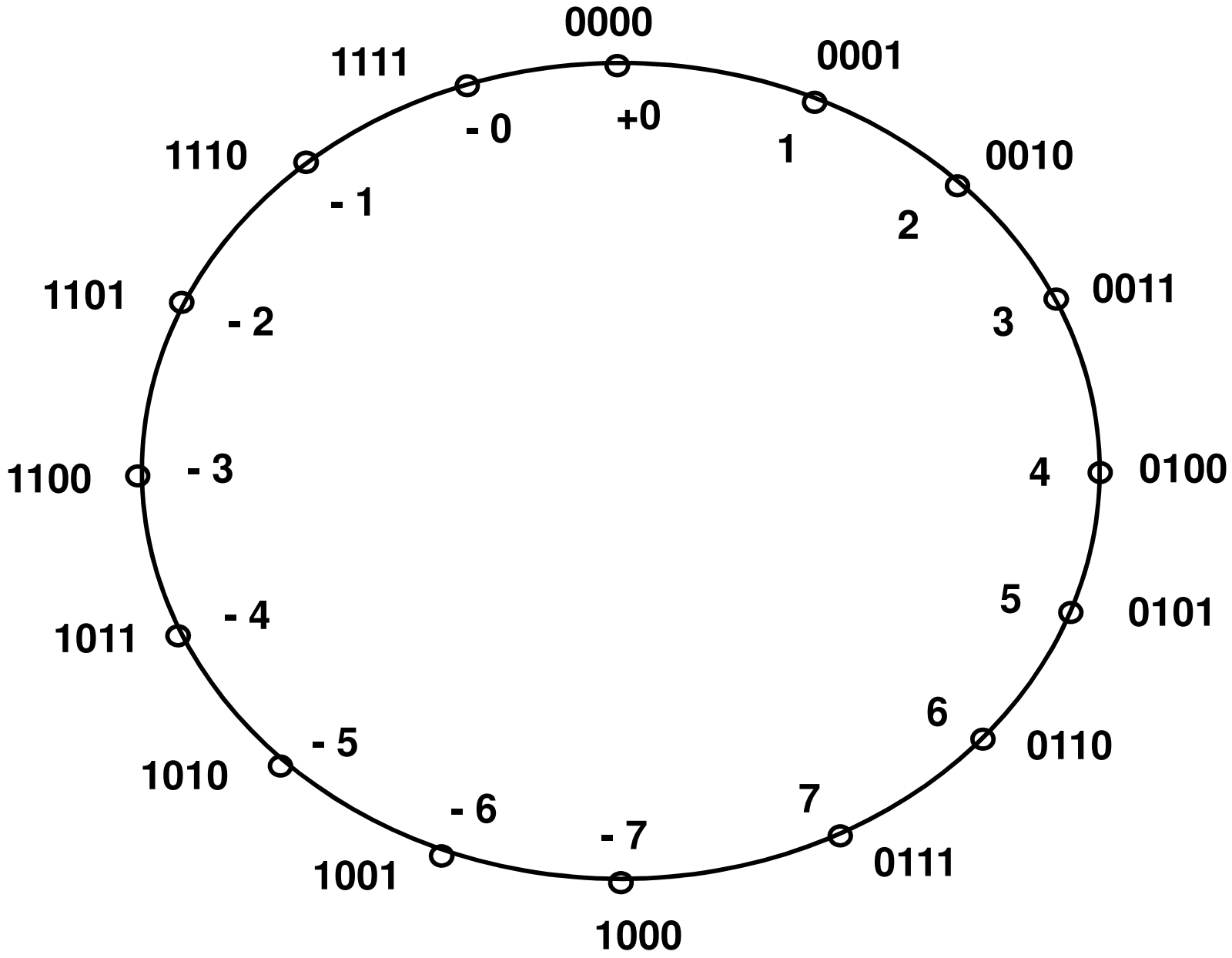
Einerkomplementdarstellung

In der Einerkomplementdarstellung negiert man
Zahlen durch bitweises Komplementieren.

0000	=	0	1000	=	- 7
0001	=	1	1001	=	- 6
0010	=	2	1010	=	- 5
0011	=	3	1011	=	- 4
0100	=	4	1100	=	- 3
0101	=	5	1101	=	- 2
0110	=	6	1110	=	- 1
0111	=	7	1111	=	- 0

Das erste Bit gibt das Vorzeichen an.
Es gibt zwei Darstellungen für Null.

Einerkomplement am Zahlenkreis



Komplementarithmetik (2. modifizierter Weg)

Weg für Addition und Subtraktion:

- **Positive Zahlen um VZ-Stelle mit Null erweitern**
- **Negative Zahlen ziffernweise zur höchsten Ziffer ergänzen (komplementieren)**
- **1 addieren**
- **Normale Addition durchführen**
- **Überlauf auf Stelle n+2 ignorieren**
- **Negative Zahlen ggf. ziffernweise zur höchsten Zahl ergänzen (komplementieren) und 1 addieren**

Zweierkomplement

Zweierkomplementdarstellung

Einerkomplementdarstellung:
Zahlen-Negierung durch
bitweises Komplementieren

0000	=	0	1000	=	- 7
0001	=	1	1001	=	- 6
0010	=	2	1010	=	- 5
0011	=	3	1011	=	- 4
0100	=	4	1100	=	- 3
0101	=	5	1101	=	- 2
0110	=	6	1110	=	- 1
0111	=	7	1111	=	- 0

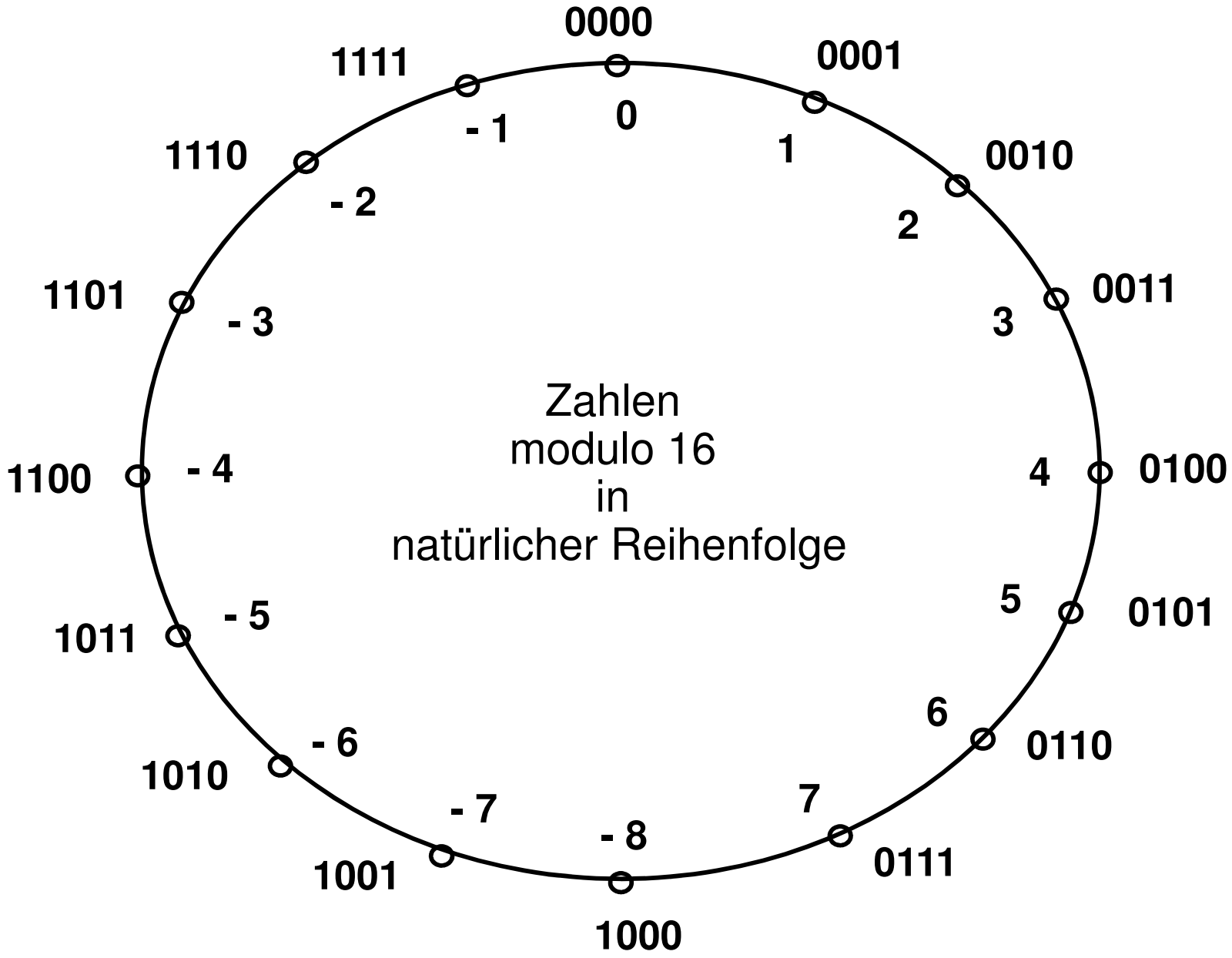
Erstes Bit: Vorzeichen
Zwei Darstellungen für Null.

Zweierkomplementdarstellung:
Durchlauf zunächst der positiven
Zahlen, dann der negativen Zahlen
in umgekehrter Reihenfolge

0000	=	0	1000	=	- 8
0001	=	1	1001	=	- 7
0010	=	2	1010	=	- 6
0011	=	3	1011	=	- 5
0100	=	4	1100	=	- 4
0101	=	5	1101	=	- 3
0110	=	6	1110	=	- 2
0111	=	7	1111	=	- 1

Auch bei Zweierkomplementdarstellung
Erste Ziffer: Vorzeichen

Zweierkomplement am Zahlenkreis



Binärcodierte Dezimalziffern

4 Bits entsprechen genau einer Hexadezimalziffer. Verzichtet man auf die Nutzung der Bitfolgen 1010 .. 1111, dann kann man jeweils 4 Bits nutzen um eine Dezimalziffer zu codieren:

0000 = 0	**0100 = 4**	**1000 = 8**	**1100 = C**
0001 = 1	**0101 = 5**	**1001 = 9**	**1101 = D**
0010 = 2	**0110 = 6**	**1010 = A**	**1110 = E**
0011 = 3	**0111 = 7**	**1011 = B**	**1111 = F**

BCD - Code (**B**inary **C**oded **D**ecimal)

BCD

An der Dezimaldarstellung orientierte Abspeicherung von Zahlen.
Einzelne Ziffern (0 bis 9) werden binär abgespeichert.
n Dezimalstellen erfordern n Speicherworte

Beispiel:
Angenommen sei ein 16bit-System, Abspeichern der Zahl 255_{10}

	Dual	**BCD**
Adresse n:	0000 0000 1111 1111	0000 0000 0000 0101
Adresse n+1	-	0000 0000 0000 0101
Adresse n+2	-	0000 0000 0000 0010

Sehr aufwendig und speicherintensiv.

Mögliche Abhilfe: **Packed BCD**
Mehrere BCD Ziffern werden in einem Speicherwort zusammengefasst.

0000 0010 0101 0101

BCD-Arithmetik

Beispiel: 5 + 7

```
  0101 (5)
+0111 (7)
------
  1100 (?) ist keine gültige BCD Ziffer
+0110 (6) Korrekturwert falls Ergebnis > 9)
------
10010 (1,2) 2 mit Übertrag in die nächste Stelle.
```

Normale Addition und Subtraktion, aber:
Korrektur des Ergebnisses falls es keine BCD Ziffer ist.
Abarbeitung von rechts nach links !!

Wird oft von der Prozessorhardware unterstützt
Addition einer Stelle mit 2 Befehlen:
1. Normale Addition, 2. BCD Korrektur

Vorteil: Beliebige Genauigkeit, unter Software Kontrolle.
Nachteil: Langsam, alles wird über Schleifen abgewickelt

Gleitkommazahlen

Analog zu der Schreibweise $815 = 0.815 * 10^3$

werden binäre Gleitkommazahlen gebildet.

Gleitkommazahlen (engl.: floating point numbers)

bestehen aus:

- dem Vorzeichen V
- dem Exponenten E
- der Mantisse M

V, E und M repräsentieren dann die Zahl

$$(-1)^V * M * 2^E$$

Normierte Gleitkommazahlen

Durch Verschieben des Kommas und gleichzeitiger Anpassung des Exponenten kann man erreichen, dass die erste Stelle der Mantisse 1 ist.

Diese Darstellung heißt **normierte Gleitkommadarstellung**.

$$0.01010111 * 2^{14}$$

$$= \quad 0.1010111 * 2^{13}$$

$$= \quad 1.010111 * 2^{12}$$

$$= \quad 10.10111 * 2^{11}$$

normiert

Vorteil normierter Gleitkommazahlen: Optimale Nutzung der Mantissenbits, da keine überflüssigen Nullen gespeichert werden müssen.

Die führende 1 braucht auch nicht gespeichert zu werden.

Darstellung von Gleitkommazahlen nach IEEE 754

Einfache Genauigkeit (Single Precision): 32 bit

V	Exponent e	Mantisse m
1 bit	8 bit	23 bit

Doppelte Genauigkeit (Double Precision): 64 bit

V	Exponent e	Mantisse m
1 bit	11 bit	52 bit

V: Vorzeichen. 0 = positiv; 1 = negativ. Betragsdarstellung der Mantisse.
m: Mantisse, normiert. Nur fraktioneller Teil, ohne die 1 vor dem Komma.
=> mit 23 bit Mantisse 24 bit Genauigkeit
e: Exponent, dargestellt als e = E+127 bzw. e = E+1023 -> Immer positiv

Darstellung von Gleitkommazahlen nach IEEE 754

Darstellbare Zahlenbereiche

	Einfache Genauigkeit	*Doppelte Genauigkeit*
Bereich für E:	-126 .. 127	-1022 .. 1023
Kleinste positive Zahl	$2^{-126} \approx 1{,}2 * 10^{-38}$	$2^{-1022} \approx 2{,}2 * 10^{-308}$
Bits in Mantisse	24	53
Größte positive Zahl	$(2 - 2^{-23}) * 2^{127} \approx 3{,}4 * 10^{38}$	$(2 - 2^{-52}) * 2^{1023} \approx 1{,}8 * 10^{308}$
Größter relativer Fehler	2^{-24}	2^{-53}
Genauigkeit	$\approx$ **7 Dezimalstellen**	$\approx$**16 Dezimalstellen**

Darstellung von Gleitkommazahlen nach IEEE 754

Warum Exponent nur -126 bis 127 (254 Werte). Restliche 2 ?

Kleinster Exponent 00 und größter Exponent 0FFH werden für Sonderfälle benutzt.

Vorzeichen	*Exponent*	*Mantisse*	*Bedeutung*
0/1	0	0	+/- Null
0/1	0	1 bis 111....11	Nicht normierte Zahlen
0/1	FFH	0	+/- Unendlich
-	FFH	1 bis 111.....11	NaN (Not a Number)

e = 0, M = 0: Plus oder Minus Null
e = 0, M ≠ 0: Nicht normiert: 0,M: Zahlenbereich um die 0 herum
e = 0FFH, M = 0: Plus oder Minus (V) Unendlich
e = 0FFH, M ≠ 0: NAN, Not A Number, Undefiniert, z.B. ∞/ ∞, ∞ – ∞

Beispielaufgabe zu IEEE 754

Stellen Sie die Dualzahl -1 0011 10011,000 0110 0111
als einfach genaue Gleitkommazahl gemäß IEEE754 dar. *(5 Punkte)*

Es gilt das folgende Schema:

v e1 e2 … e8 m1 m2 … m23

v: Vorzeichen (0:positiv, 1:negativ)
e: Exponent e = e1 e2 … e8

m: Mantisse m = m1 m2 … m23
aus normierter dualer Gleitkommadarstellung 1. mk...mk+n *2E
m: nur fraktioneller Anteil der Mantisse der dualen Gleitkommadarstellung

e = E+127

Zeichen des ASCII Codes

	High	*000*	*001*	*010*	*011*	*100*	*101*	*110*	*111*
Low	**Hex**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0000	**0**	NUL	DLE		0	@	P	`	p
0001	**1**	SOH	DC	!	1	A	Q	a	q
0010	**2**	STX	DC	"	2	B	R	b	r
0011	**3**	ETX	DC	#	3	C	S	c	s
0100	**4**	EOT	DC	$	4	D	T	d	t
0101	**5**	ENQ	NAK	%	5	E	U	e	u
0110	**6**	ACK	SYN	&	6	F	V	f	v
0111	**7**	BEL	ETB	'	7	G	W	g	w
1000	**8**	BS	CAN	(	8	H	X	h	x
1001	**9**	HT	EM	)	9	I	Y	i	y
1010	**A**	LF	SUB	*	:	J	Z	j	z
1011	**B**	VT	ESC	+	;	K	[	k	{
1100	**C**	FF	FS	,	<	L	\	l	|
1101	**D**	CR	GS	-	=	M	]	m	}
1110	**E**	SO	RS	.	>	N	^	n	~
1111	**F**	SI	US	/	?	O	_	o	DEL

Zeichen des IBM ASCII Codes

Ctrl	Dec	Hex	Char	Code
^@	0	00		NUL
^A	1	01	☺	SOH
^B	2	02	☻	STX
^C	3	03	♥	ETX
^D	4	04	♦	EOT
^E	5	05	♣	ENQ
^F	6	06	♠	ACK
^G	7	07	•	BEL
^H	8	08	◘	BS
^I	9	09	○	HT
^J	10	0A	◙	LF
^K	11	0B	♂	VT
^L	12	0C	♀	FF
^M	13	0D	♪	CR
^N	14	0E	♫	SO
^O	15	0F	☼	SI
^P	16	10	►	DLE
^Q	17	11	◄	DC1
^R	18	12	↕	DC2
^S	19	13	‼	DC3
^T	20	14	¶	DC4
^U	21	15	§	NAK
^V	22	16	▬	SYN
^W	23	17	↨	ETB
^X	24	18	↑	CAN
^Y	25	19	↓	EM
^Z	26	1A	→	SUB
^[	27	1B	←	ESC
^\	28	1C	∟	FS
^]	29	1D	↔	GS
^^	30	1E	▲	RS
^_	31	1F	▼	US

Dec	Hex	Char
32	20	
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	32	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?

Dec	Hex	Char
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_

Dec	Hex	Char
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	⌂

Dec	Hex	Char
128	80	Ç
129	81	ü
130	82	é
131	83	â
132	84	ä
133	85	à
134	86	å
135	87	ç
136	88	ê
137	89	ë
138	8A	è
139	8B	ï
140	8C	î
141	8D	ì
142	8E	Ä
143	8F	Å
144	90	É
145	91	æ
146	92	Æ
147	93	ô
148	94	ö
149	95	ò
150	96	û
151	97	ù
152	98	ÿ
153	99	Ö
154	9A	Ü
155	9B	¢
156	9C	£
157	9D	¥
158	9E	₧
159	9F	ƒ

Dec	Hex	Char
160	A0	á
161	A1	í
162	A2	ó
163	A3	ú
164	A4	ñ
165	A5	Ñ
166	A6	ª
167	A7	º
168	A8	¿
169	A9	⌐
170	AA	¬
171	AB	½
172	AC	¼
173	AD	¡
174	AE	«
175	AF	»
176	B0	░
177	B1	▒
178	B2	▓
179	B3	│
180	B4	┤
181	B5	╡
182	B6	╢
183	B7	╖
184	B8	╕
185	B9	╣
186	BA	║
187	BB	╗
188	BC	╝
189	BD	╜
190	BE	╛
191	BF	┐

Dec	Hex	Char
192	C0	└
193	C1	┴
194	C2	┬
195	C3	├
196	C4	─
197	C5	┼
198	C6	╞
199	C7	╟
200	C8	╚
201	C9	╔
202	CA	╩
203	CB	╦
204	CC	╠
205	CD	═
206	CE	╬
207	CF	╧
208	D0	╨
209	D1	╤
210	D2	╥
211	D3	╙
212	D4	╘
213	D5	╒
214	D6	╓
215	D7	╫
216	D8	╪
217	D9	┘
218	DA	┌
219	DB	▒
220	DC	▒
221	DD	▌
222	DE	▐
223	DF	▀

Dec	Hex	Char
224	E0	α
225	E1	ß
226	E2	Γ
227	E3	π
228	E4	Σ
229	E5	σ
230	E6	µ
232	E7	τ
232	E8	Φ
233	E9	Θ
234	EA	Ω
235	EB	δ
236	EC	∞
237	ED	φ
238	EE	ε
239	EF	∩
240	F0	≡
241	F1	±
242	F2	≥
243	F3	≤
244	F4	⌠
245	F5	⌡
246	F6	÷
247	F7	≈
248	F8	°
249	F9	∙
250	FA	·
251	FB	√
252	FC	ⁿ
253	FD	²
254	FE	■
255	FF	

Zeichen des ANSI Codes

Tabelle A.1 ANSI-Zeichensatz

0	■	24	■	48	0	72	H	96	`	120	x	144	■	168	¨
1	■	25	■	49	1	73	I	97	a	121	y	145	‘	169	©
2	■	26	■	50	2	74	J	98	b	122	z	146	’	170	ª
3	■	27	■	51	3	75	K	99	c	123	{	147	■	171	«
4	■	28	■	52	4	76	L	100	d	124	|	148	■	172	¬
5	■	29	■	53	5	77	M	101	e	125	}	149	■	173	-
6	■	30	■	54	6	78	N	102	f	126	~	150	■	174	®
7	■	31	■	55	7	79	O	103	g	127	■	151	■	175	¯
8	**	32		56	8	80	P	104	h	128	■	152	■	176	°
9	**	33	!	57	9	81	Q	105	i	129	■	153	■	177	±
10	**	34	”	58	:	82	R	106	j	130	■	154	■	178	²
11	■	35	#	59	;	83	S	107	k	131	■	155	■	179	³
12	■	36	$	60	<	84	T	108	l	132	■	156	■	180	´
13	**	37	%	61	=	85	U	109	m	133	■	157	■	181	µ
14	■	38	&	62	>	86	V	110	n	134	■	158	■	182	¶
15	■	39	'	63	?	87	W	111	o	135	■	159	■	183	·
16	■	40	(	64	@	88	X	112	p	136	■	160		184	¸
17	■	41	)	65	A	89	Y	113	q	137	■	161	¡	185	¹
18	■	42	*	66	B	90	Z	114	r	138	■	162	¢	186	º
19	■	43	+	67	C	91	[	115	s	139	■	163	£	187	»
20	■	44	,	68	D	92	\	116	t	140	■	164	¤	188	¼
21	■	45	-	69	E	93	]	117	u	141	■	165	¥	189	½
22	■	46	.	70	F	94	^	118	v	142	■	166	¦	190	¾
23	■	47	/	71	G	95	_	119	w	143	■	167	§	191	¿

Tabelle A.1 ANSI-Zeichensatz *(Fortsetzung)*

192	À	208	Ð	224	à	240	ð
193	Á	209	Ñ	225	á	241	ñ
194	Â	210	Ò	226	â	242	ò
195	Ã	211	Ó	227	ã	243	ó
196	Ä	212	Ô	228	ä	244	ô
197	Å	213	Õ	229	å	245	õ
198	Æ	214	Ö	230	æ	246	ö
199	Ç	215	×	231	ç	247	÷
200	È	216	Ø	232	è	248	ø
201	É	217	Ù	233	é	249	ù
202	Ê	218	Ú	234	ê	250	ú
203	Ë	219	Û	235	ë	251	û
204	Ì	220	Ü	236	ì	252	ü
205	Í	221	Ý	237	í	253	ý
206	Î	222	Þ	238	î	254	þ
207	Ï	223	ß	239	ï	255	ÿ

ISO 8859

- 8-Bit Codefamilie
- 15 verschiedene Teilnormen
- Code 000-127 bei allen Teilnormen gleich (entspricht ASCII-Code)
- Besonders wichtig: ISO-8859-1 (Latin-1, ANSI, EBCDIC)
- ISO 8859-x

-1	*Latin-1*, Westeuropäisch
-2	*Latin-2*, Mitteleuropäisch
-3	*Latin-3*, Südeuropäisch
-4	*Latin-4*, Baltisch
-5	Kyrillisch
-6	Arabisch
-7	Griechisch
-8	Hebräisch
-9	*Latin-5*, Türkisch
-10	*Latin-6*, Nordisch
-11	Thai
-13	*Latin-7*, Baltisch
-14	*Latin-8*, Keltisch
-15	*Latin-9*, Westeuropäisch
-16	*Latin-10*, Südosteuropäisch

Unicode

- Universelle Symboltabelle
- Jedes Zeichen bekommt einen eindeutigen binären Code mit 5+16=21 Bit
- Der Binärcode ist auf jeder Hardware, unter jedem Betriebssystem und in jeder Programmiersprache gleich.
- Es gibt 17 Codebereiche (Planes) mit je 65536 möglichen Zeichen
- Wichtig: Basic Multilingual Plane (BMP)
- Die ersten 256 Zeichen der BMP sind identisch mit ISO 8859-1
- Zur Codierung des Unicode: Universal Transformation Format (UTF)
- UTF-8, UTF-16, UTF-32

Einige Zeichen des Unicodes

Latin Extended-A

	010	011	012	013	014	015	016	017
0	Ā 0100	Đ 0110	Ġ 0120	İ 0130	ŀ 0140	Ő 0150	Š 0160	Ű 0170
1	ā 0101	đ 0111	ġ 0121	ı 0131	Ł 0141	ő 0151	š 0161	ű 0171
2	Ă 0102	Ē 0112	Ģ 0122	IJ 0132	ł 0142	Œ 0152	Ţ 0162	Ų 0172
3	ă 0103	ē 0113	ģ 0123	ij 0133	Ń 0143	œ 0153	ţ 0163	ų 0173
4	Ą 0104	Ĕ 0114	Ĥ 0124	Ĵ 0134	ń 0144	Ŕ 0154	Ť 0164	Ŵ 0174
5	ą 0105	ĕ 0115	ĥ 0125	ĵ 0135	Ņ 0145	ŕ 0155	ť 0165	ŵ 0175
6	Ć 0106	Ė 0116	Ħ 0126	Ķ 0136	ņ 0146	Ŗ 0156	Ŧ 0166	Ŷ 0176
7	ć 0107	ė 0117	ħ 0127	ķ 0137	Ň 0147	ŗ 0157	ŧ 0167	ŷ 0177
8	Ĉ 0108	Ę 0118	Ĩ 0128	ĸ 0138	ň 0148	Ř 0158	Ũ 0168	Ÿ 0178
9	ĉ 0109	ę 0119	ĩ 0129	Ĺ 0139	ʼn 0149	ř 0159	ũ 0169	Ź 0179

Cherokee

	13A	13B	13C	13D	13E	13F
0	D 13A0	Ꭵ 13B0	G 13C0	Ꮔ 13D0	Ꭼ 13E0	Ᏼ 13F0
1	R 13A1	Ꭻ 13B1	Ꮕ 13C1	Ꮥ 13D1	Ꮵ 13E1	Ᏽ 13F1
2	T 13A2	Ꭾ 13B2	h 13C2	R 13D2	P 13E2	Ꮶ 13F2
3	Ꮃ 13A3	W 13B3	Z 13C3	Ꮮ 13D3	Ꮷ 13E3	Ꮾ 13F3
4	Ꮄ 13A4	Ꮸ 13B4	Ꮤ 13C4	Ꮻ 13D4	V 13E4	B 13F4
5	i 13A5	Ꮹ 13B5	Ꮅ 13C5	Ꮪ 13D5	Ꮿ 13E5	
6	Ꮪ 13A6	Ꮶ 13B6	Ꭲ 13C6	Ꮼ 13D6	K 13E6	
7	Ꮗ 13A7	M 13B7	Ꮷ 13C7	Ꮍ 13D7	Ꮗ 13E7	
8	Ꮈ 13A8	Ꮻ 13B8	Ꮸ 13C8	Ꮍ 13D8	Ꮾ 13E8	
9	Y 13A9	Ꮹ 13B9	Ꮉ 13C9	V 13D9	Ꮆ 13E9	

Bopomofo Extended

	31A	31B
0	ㆠ 31A0	ㇿ 31B0
1	ㆡ 31A1	ㇱ 31B1
2	ㆢ 31A2	ㇲ 31B2
3	ㆣ 31A3	ㇳ 31B3
4	ㆤ 31A4	ㇴ 31B4
5	ㆥ 31A5	ㇵ 31B5
6	ㆦ 31A6	ㇶ 31B6
7	ㆧ 31A7	ㇷ 31B7
8	ㆨ 31A8	
9	ㆩ 31A9	