Some Routines From Microsoft Basic

Jim Butterfield, Toronto

KIM	SYM	AIM	OSI	Description
2000	C003	B00A	A000	Action addresses for primary keywords
203A	C03D	B044	A038	Action addresses for functions
2068	C06B	B072	A066	Hierarchy and action addresses for operators
2086	C089	B090	A084	Table of Basic keywords
2169	C16E	B175	A164	Basic messages, mostly error messages
2274	C1AB	B1AC	A1A1	Search the stack for FOR or GOSUB activity
22A2	C1D9	B1DA	A1CF	Open up space in memory
22E5	C21C	B21D	A212	Test: stack too deep?
22F2	C229	B22A	A21F	Check available memory
231F	C256	B257	A24C	Send canned error message, then:
2348	C27E	B27F	A274	Warm start; wait for Basic command
236A	C2A0	B29D	A295	Handle new Basic line input
23F1	C32C	B329	A32E	Rebuild chaining of Basic lines
2420	C359	B356	A34B	Receive line from keyboard
2466	C39F	B3AE	A3A6	Crunch keywords into Basic tokens
24F2	C427	B436	A432	Search Basic for given line number
2521	C456	B465	A461	Perform NEW
253C	C472	B481	A68C	Perform CLEAR
256B	C49F	B4AE	A4A7	Reset Basic execution to start
2579	C4AC	B4BC	A4B5	Perform LIST
2608	C535	B55C	A556	Perform FOR
26AA	C5DA	B601	A5FF	Execute Basic statement
26CB	C60A	B631	A61A	Perform RESTORE
26DA	C619	B640	A62C	Check stop key
26E8	C622	B65C	A638	Perform STOP or END
2711	C64B	B685	A661	Perform CONT
272B	C665		A67B	Perform NULL
273C	C676	B69F	FFF7	Perform SAVE
278C	C6B7		FFF4	Perform LOAD
		B6AB		Special AIM input routines
27CA	C707	B6EC	A691	Perform RUN
27D5	C712	B6F7	A69C	Perform GOSUB
27F2	C72F	B714	A6B9	Perform GOTO
281F	C75C	B741	A6E6	Perform RETURN, then:
2845	C782	B767	A70C	Perform DATA: skip statement
2853	C790	B775	A71A	Scan for next Basic statement
2857	C793	B778	A71D	Scan for next Basic line
2875	C7B2	B797	A73C	Perform IF, and perhaps:
2888	C7C5	B7AA	A74F	Perform REM: skip line
2898	C7D5	B7BA	A75F	Perform ON
28B8	C7F5	B7DA	A77F	Input fixed-point number
28F2	C82F	B814	A7B9	Perform LET
		B89D		Enable printer
297B	C8B8	B8A9	A829	Perform PRINT
2A13	C94F	B94A	A8C3	Print string from memory
2A35	C971	B967	A8E0	Print single format character
2A59	C991	B988	A904	Handle bad input data
2A7E		B9AD		Perform GET
2A8D	C9B0	B9BC	A923	Perform INPUT
2AB0	C9DC	B9E7	A946	Prompt and receive input
2AB9	C9E5	B9F0	A94F	Perform READ
2BA2	CAB4	BADC	AA1C	Canned Input error messages
2BC6	CAD8	BB00	AA40	Perform NEXT
2C34	CB43	BB59	AAAD	Check type mismatch
2C48	CB57	BB7F	AAC1	Evaluate expression
2D82	CC9F	BCB9	ABF5	Evaluate expression within parentheses
2D88	CCA5	BCBF	ABFB	Check parenthesis, comma
2D99	CCB6	BCD0	AC0C	Syntax error exit
2D9E	CCBB	BCD5	AC11	Setup for functions
2DA5	CCC2	BCDC	AC18	Variable name setup
2DC5	CCE6	BD00	AC27	Set up function references
2E04	CD25	BD3F	AC66	Perform OR, AND
2E34	CD55	BD6F	AC96	Perform comparisons
2E9F	CE11	BDDA	AD01	Perform DIM
2EA9	CE5F	BDE4	AD0B	Search for variable
2F3D	CEF3	BE78	AD8B	Create new variable
2FA3	CF57	BEDC	ADE6	Setup array pointer
2FB4	CF68	BEED	ADF7	Evaluate integer expression
2FD4	CF8B	BF10	AE17	Find or make array
3181	D138	C0BD	AFAD	Perform FRE, and:
3195	D14C	C0D1	AFC1	Convert fixed-to-floating
31A2	D159	C0DE	AFCE	Perform POS
31A8	D15F	C0E4	AFD4	Check not Direct
31B2	D16C	C0F1	AFDE	Perform DEF
31E0	D19A	C11F	B00B	Check FNx syntax
31F3	D1AD	C132	B01E	Evaluate FNx
3266	D21E	C1A3	B08C	Perform STR$
3276	D22E	C1B3	B09C	Do string vector
3288	D240	C1C5	B0AE	Scan, set up string
32EF	D2A9	C232	B115	Build descriptor
3321	D2DB	C264	B147	Garbage collection
3434	D3F2	C37B	B24D	Concatenate
3471	D42F	C3B8	B28A	Store string
349A	D458	C3E1	B2B3	Discard unwanted string
34D2	D490	C419	B2EB	Clean descriptor stack
34E3	D4A1	C42A	B2FC	Perform CHR$
34F7	D4B5	C43E	B310	Perform LEFT$
3523	D4E1	C46A	B33C	Perform RIGHT$

Routines were identified by examining specific machines. There may well be other versions of Basic on these machines; the user is urged to exercise caution.

OSI is from a C2-4 machine. KIM is a cassette tape version. SYM and AIM are the ROM versions.

The addresses given identify the start of the area in which the described routine lies. This may not be the proper program entry point or calling address.

352E	D4EC	C475	B347	Perform MID$
3556	D516	C49F	B36F	Pull string data
3573	D531	C4BA	B38C	Perform LEN
3579	D537	C4C0	B392	Switch string to numeric
3582	D540	C4C9	B39B	Perform ASC
3592	D550	C4D9	B3AB	Get byte parameter
35A4	D562	C4EB	B3BD	Perform VAL
35E3	D5A1	C52A	B3FC	Get two parameters for POKE or WAIT
35EF	D5AD	C536	B408	Convert floating-to-fixed
3605	D5C3	C54C	B41E	Perform PEEK
3610	D5DA	C563	B429	Perform POKE
3619	D5E3	C56C	B432	Perform WAIT
3635	D5FF	C588	B44E	Add 0.5
363C	D606	C58F	B455	Perform subtraction
364E	D618	C5A6	B467	Perform addition
3765	D6FD	C686	B537	Complement accum#1
379C	D734	C6BD	B564	Overflow exit
37A1	D739	C6C2	B569	Multiply-a-byte
3802	D772	C6FB	B59C	Constants
3830	D7A0	C729	B5BD	Perform LOG
386E	D7DE	C76A	B5FB	Perform multiplication
3904	D842	C7CB	B64D	Unpack memory into accum#2
392F	D86D	C7F6	B673	Test & adjust accumulators
394C	D88A	C813	B690	Handle overflow and underflow
395A	D898	C821	B69E	Multiply by 10
3971	D8AF	C838	B6B5	10 in floating binary
3976	D8B4	C83D	B6B9	Divide by 10
3987	D8C5	C846	B6CA	Perform divide-by
398C	D8CA	C851	B6CF	Perform divide-into
3A1A	D958	C8E1	B74B	Unpack memory into accum#1
3A3F	D97D	C906	B76B	Pack accum#1 into memory
3A74	D9B2	C93B	B79B	Move accum#2 to #1
3A84	D9C2	C94B	B7AB	Move accum#1 to #2
3A93	D9D1	C95A	B7BA	Round accum#1
3AA3	D9E1	C96A	B7CA	Get accum#1 sign
3AB1	D9EF	C978	B7D8	Perform SGN
3AD0	DA0E	C997	B7F5	Perform ABS
3AD3	DA11	C99A	B7F8	Compare accum#1 to memory
3B13	DA51	C9DA	B831	Floating-to-fixed
3B44	DA82	CA0B	B862	Perform INT
3B6B	DAA9	CA32	B887	Convert string to floating-point
3C0A	DB3B	CABD	B912	Get new ASCII digit
3C3F	DB70	CAF2	B947	Constants
3C4E	DB7F	CB01	B953	Print IN, then:
3C55	DB86	CB0C	B95A	Print Basic line #
3C69	DB9A	CB1C	B96E	Convert floating-point to ASCII
3D99	DCCA	CC4C	BA96	Constants
3DC2	DCF3	CC75	BAAC	Perform SQR
3DCC	DCFD	CC7F	BAB6	Perform power function
3E05	DD36	CCB8	BAEF	Perform negation
3E10	DD41	CCC3	BAFA	Constants
3E3E	DD6F	CCF1	BB1B	Perform EXP
3E91	DDC2	CD44	BB6E	Series evaluation
3EDB	DE0C	CD8E	BBB8	RND constants
3EE3	DE14	CD96	BBC0	Perform RND
3F1F		CDD2	BBFC	Perform COS
3F26		CDD9	BC03	Perform SIN
3F6F		CE22	BC4C	Perform TAN
3F9B		CE86	BC78	Constants
3FD3			BC99	Perform ATN
4003			BCC9	Constants
4041	DE50	CE86	BCEE	CHRGET sub for zero page

Remaining routines are Basic startup.

Basic Memory Map (Page O)

Compiled by Jim Butterfield

OSI is C2-4P. There may be differences in particular implementations of Basic.

KIM	SYM	AIM	OSI	Description
0000		0000		New-line jump
0003	000A	0003		USR jump
0006	0008	B006		Vector to 'fixed-to-floating' subroutine
0008	0006	B008		Vector to 'floating-to-fixed' subroutine
000A	000D	0006	005B	Search character
000B	000E	0007	005C	Scan-between-quotes flag
000C	000F	0008	005D	Input buffer pointer; # of subscripts
000D	0010	0009	005E	Defaault DIM flag
000E	0011	000A	005F	Type: FF=string, 00=numeric
000F	0012	000B		Type: 80=integer, 00=floating point
0010	0013	.000C	0060	Flag: DATA scan; LIST quote; memory
0011	0014	000D	0061	Subscript flag; FNX flag
0012	0015	000E	0062	0=INPUT; $40=GET; $98=READ
0013	0016	000F	0063	Comparison Evaluation flag
0014	0017	0010	0064	Input flag (suppress output)
0016	0019	0011	000E	Position on print line
0017	001A	0012	000F	Maximum print line width
0018	001B	0013	0010	Input column limit
0019	001C	0014	0011	Integer value (for GOTO etc)
001B	001E	0016	0013	Start of input buffer
0062	0065	005D	005A	End of input buffer
0063	0066	005E	0065	Pointers for descriptor stack
0066	0069	0061	0068	Descriptor stack(temp strings)
006E	0071	0069	0070	End of Descriptor stack
006F	0072	0072	0071	Utility pointeer area
0073	0076	006E	0075	Product area for multiplication
0078	007B	0073	0079	Pointer: Start-of-Basic
007A	007D	0075	007B	Pointer: Start-of-Variables
007C	007E	0077	007D	Pointer: Start-of-Arrays
007E	0081	0079	007F	Pointer: End-of-Arrays
0080	0083	007B	0081	Pointer: String-storage(moving down)
0082	0085	007D	0083	Utility string pointer
0084	0087	007F	0085	Pointer: Limit-of-memory
0086	0089	0081	0087	Current Basic line number
0088	008B	0083	0089	Previous Basic line number
008A	008D	0085	008B	Pointer: Basic statement for CONT
008C	008F	0087	008D	Current DATA line number
008E	0091	0089	008F	Current DATA address
0090	0093	008B	0091	Input vector
0092	0095	008D	0093	Current variable name
0094	0097	008F	0095	Current variable address
0096	0099	0091	0097	Variable pointer for FOR/NEXT
0098	009B	0093	0099	Start of work area, pointers, etc
00A1	00A4	009C	00A1	Jump vector for functions
00A4	00A7	009F	00A4	Misc numeric work area
00AE	00B1	00A9	00AC	Accum#1: Exponent
00AF	00B2	00AA	00AD	Accum#1: Mantissa
00B3	00B6	00AE	00B0	Accum#1: Sign
00B4	00B7	00AF	00B1	Series evaluation constant pointer
00B5	00B8	00B0	00B2	Accum#1 hi-ordeer (overflow)
00B6	00B9	00B1	00B3	Accum#2: Exponent, etc.
00BC	00BF	00B7	00B8	Sign comparison, Acc#1 vs #2
00BD	00C0	00B8	00B9	Accum#1 lo-order (rounding)
00BE	00C1	00B9	00BA	Series pointer
	00C3	00BB		Error jump
	00C6			SAVE jump
	00C9			LOAD jump
00C0	00CC	00BF	00BC	CHRGET subroutine; get Basic char
00C6	00D2	00C5	00C2	Sub entry: get prev character
00C7	00D3	00C6	00C3	Basic pointer (within subrtn)
00D8	00E4	00D7	00D4	Random number seed.