TI STRICKLY PRIVATE

GROUND SQUIRREL
PERSONAL COMPUTER

SOFTWARE SPECIFICATION

(DWG NO-00000000)

SECTION 1

SCOPE

This document defines the features, functionality and documentation requirements for the software of the GROUND SQUIRREL Personal Computer. Three separate software modules are specified: (1) User Interface, (2) BASIC Interpreter and (3) Device Service Routines (DSR's).

1.1 PRODUCT FEATURES

The GROUND SQUIRREL Personal Computer is a low cost, entry level, BASIC Language computer targeted for the 'first-time-buyer' market. Design of the hardware and software are oriented toward the 'computer literacy' learning aid needs of the novice computer programmer. Features will be included to provide higher level capabilities for all consumers that progress beyond the computer literacy phase. Packaged software (cassette tapes and solid state software) will be supported by the GROUND SQUIRREL console, however, TI-99/4A Command modules will not be supported. The console will support the ALC I/O bus system to provide peripheral expansion capabilities. An expansion port will also be provided to support RAM/ROM expansion and solid state software modules.

1.2 PRODUCT CONFIGURATION

The GROUND SQUIRREL Console will consist of a 48 station keyboard, audio cassette interface (no motor control), ALC I/O port, system expansion port and RF video output for a black and white television. No sound, color, user defined characters or video sprites will be supported. For overall TI product line integration, the GROUND SQUIRREL Console BASIC will be a funtional subset of the Console BASIC language in the TI-99/4A Computer.

1.2.1 SOFTWARE MODULES. Software for the GROUND SQUIRREL Computer will be specified in terms of 3 modules: User Interface, BASIC Interpreter and Device Service Routines.

1.2.1.1 User Interface. The User Interface module includes all software segments that provide screen displays, prompting, menus, program editing/execution controls and error handling. Within the functional limitations of the GROUND SQUIRREL architecture, the User Interface shall be compatible with that of the TI-99/4A Computer system.

1.2.1.2 BASIC Interpreter. The BASIC Interpreter module includes all software segments that enable the GROUND SQUIRREL to be user programmed in BASIC language. This interpreter shall implement a maximized subset of the CONSOLE BASIC that is incorporated into the TI-99/4A Computer.

1.2.1.3 Device Service Routines. The Device Service Routines (DSRs) include all software elements that implement system features and facilitate usage of peripherals. This includes (at a minimum):

* B/W Video Display -- Standard black/white television providing 24 row by 32 column format.

* Keyboard -- 48 key, QWERTY format keyboard.

* ALC I/O Port -- A 8 wire (6 signal, 2 power/ground) bus that provides nibble-wide data transfer with a synchronized 'Hand-shake' method.

* Expansion Port -- Complete system bus structure will be provided at the Expansion Port to allow for RAM expansion (maximum of 32 kbytes), solid state software ROM and ALC I/O peripheral expansion DSR ROMs (enhanced peripheral capability).

1.2.2 DOCUMENTATION. System level documentation will include software segmentations, theories-of-operation, flow diagrams and memory utilization maps. Documentation shall be sufficient to allow project personnel to understand all functional capabilities, options and interactions of all modules.

SECTION 2

APPLICABLE DOCUMENTS

2.1 HARDWARE

* GROUND SQUIRREL Product Specification
* TMS-9995 Microprocessor Data Manual
* ALC I/O Peripheral Bus Specification

2.2 SOFTWARE

* American National Standard for Minimal BASIC Language
* TI-99/4A BASIC Interpreter Specification

SECTION 3

SYSTEM HARDWARE

The GROUND SQUIRREL Console (minimum system) consists of a black/white TV interface, ALC I/O port, 2306 bytes of RAM (2 kbytes system and and 258 bytes 'on-chip' of the TMS-9995 processor), 12 kbytes of system ROM, audio cassette interface and a system expansion port. System memory can be expanded by 48 kbytes with 2 kbyte multiples of RAM and 4 kbyte multiples of ROM. Future ALC I/O compatible devices may be augmented by DSRs in the expansion port ROM. Solid state software (application programs) will be supported by the expansion port ROM/RAM.

3.1 MEMORY

All memory devices are 450 ns parts, 8 bits wide with no parity. The system memory and CRU maps are shown in Figure 3-1.

3.1.1 SYSTEN ROM. The 12 kbyte system ROM consists of one 8 kbyte ROM from address >0000 to >1FFF and a 4 kbyte ROM from address >2000 to >2FFF.

3.1.2 SYSTEM RAM. System RAM consists of one 2 kbyte RAM chip from address >E800 to >EFFF.

3.1.3 PROCESSOR RAM. The 9995 processor has a 252 byte segment of 'on-chip' RAM from address >F000 to >F0FB and 6 additional bytes from address >FFFA to >FFFF for on-chip register/vector storage.

3.1.4 EXPANSION ROM. Expansion ROM is provided in multiples of 4 kbytes starting at address >3000 and continuing to the nearest 4 kbyte boundary preceeding the expansion RAM. Identification of the existance of a ROM will be determined (at power-up) by system software interrogating the first 'word' of each 4 kbyte boundary, beginning at >3000. The value of the first word is the checksum of the 4 kbyte ROM. The second word is the address of the initialization routine for that 4 kbyte ROM. IF the second byte is >0000, no initialization program will be executed.

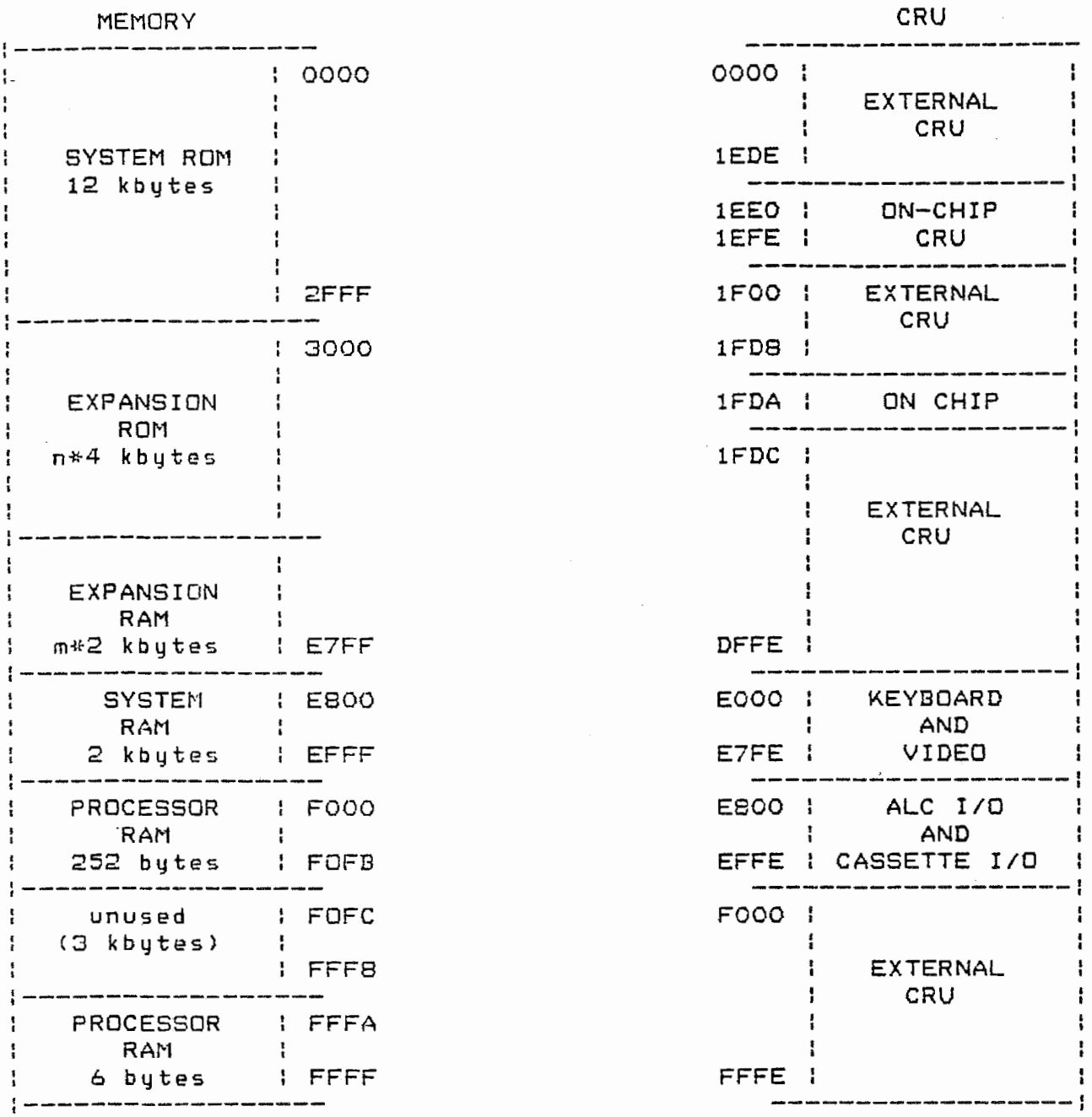


Figure 3-1 MEMORY AND CRU ADDRESS MAPS

3.1.5 EXPANSION RAM. Expansion RAM will always be positioned to be contigous to the system RAM and thus, will occupy the address space from >E7FF down toward the upper ROM boundary (or address >6800 whichever is larger), in 2 kbyte multiples. A maximum of 32 kbytes of RAM can be added.

3.1.6 CRU MAP. System level CRU I/O devices are mapped into two addresses (refer to Figure 3-1): KEYBOARD CRU at >E000 to E7FE and PERI I/O at >E800 to >EFFE. Definitions of CRU bits for each address are shown in Table 3-1.

Table 3-1 CRU BIT DEFINITIONS

CRUBASE	INPUT	OUTPUT
>E000	keyboard input COLUMN 0	keyboard output ROW 0
>E002	" " " 1	" " " 1
>E004	" " " 2	" " " 2
>E006	" " " 3	" " " 3
>E008	" " " 4	" " " 4
>E00A	" " " 5	keyboard output ROW 5
>E00C	" " " 6	not used
>E00E	keyboard input COLUMN 7	video enable (VIDENA)
>E800	ALC data line D0	ALC data line D0
>E802	" " " D1	" " " D1
>E804	" " " D2	" " " D2
>E806	ALC data line D3	ALC data line D3
>E808	ALC handshake HSK	ALC handshake HSK
>E80A	ALC bus available BAV	ALC bus available BAV
>E80C	not used	ALC inhibit INH
>E80E	cassette in CASIN	cassette out CASOT

3.2 KEYBOARD

The keyboard is a non-interrupt driven input device. It consists of 6 rows (OUTPUT) and 8 columns (INPUT) lines to give a total of 48 key stations. Keycode definitions are shown in Figure 3-2.

3.2.1 CRU BASE. The CRU BASE address for the keyboard is >EF00 to >EF0E. The CRU defined lines are bidirectional and the CRU instructions of SBZ, SBO, TB, LDCR and STCR are usable without changing the CRU BASE address register (R12).

3.2.2 SCAN METHOD. All rows (scan lines) and columns are normally "high" (logic one). To scan the keyboard, each scan line is individually pulled low and the column lines are read by CRU instructions. If a column line is "low" then a key has been detected at the intersection of the row being scanned and the column that is low. Multiple key closures will be detected by this method, however, only the SHIFT, CNTL and FNCT keys are allowed to be pressed at the same time as another key. Any other multiple key closure condition is invalid and only the first key detected will be accepted. Software must provide key debounce logic to eliminate false key entries. Software must, also, convert the row/column number to the appropriate key codes, adjusted for SHIFT and FCNT keys. For the console with no expansion module, the CNTL key will be ignored. However, a unique expansion module may utilize the CNTL key but it will also provide an alternate keycode definition table to replace the one in the console ROM.

OUTPUT LINES (rows)	MODE	SCAN LINES (columns) 0	1	2	3	4	5
0	function	REDO	BACK		QUIT		ENTER
	shift	*	(	)	+	-	ENTER
	normal	8	9	0	=	/	ENTER
1	function	AID					SHIFT
	shift	&	?	'	"	:	SHIFT
	normal	7	I	O	P	;	SHIFT
2	function	PROC'D					FCTN
	shift	^					FCTN
	normal	6	Y	U	K	L	FCTN
3	function	BEGIN					
	shift	%	]			<	>
	normal	5	T	H	J	,	.
4	function	CLEAR					
	shift	$	[	{	}		
	normal	4	R	F	G	N	M
5	function	ERASE	↑	→			
	shift	#	↑	→	`		
	normal	3	E	D	C	V	B
6	function	INS		←	↓	CTRL	SPACE
	shift	@	~	←	↓	CTRL	SPACE
	normal	2	W	S	X	CTRL	SPACE
7	function	DEL				SHIFT	BREAK
	shift	!			\	SHIFT	BREAK
	normal	1	Q	A	Z	SHIFT	BREAK

Figure 3-2 KEYBOARD DECODES

3.3 VIDEO DISPLAY

The video display consists of 24 lines of 32 characters. To display information the processor places the correct character codes into the display RAM buffer. The processor has no responsibility for converting the display RAM buffer to character dot patterns for the actual RF video generation. All this is the responsibility of the of the CRT controller chip. However, the CRT controller utilizes DMA techniques for video generation and, as such, will periodically take control of the system address and data buses. This activity can only be inhibited by disabling the CRT controller (which blanks the screen) by setting VIDENA low at CRU address >E00E. Note, the DMA action minimizes the ability of the processor to execute software timing loops with any repeatability.

3.3.1 VIDEO SCREEN RAM. The 24 rows of 32 characters are stored in system RAM (video RAM buffer) starting at address >EC00 and continuing to >EEFF (768 bytes). The next byte (>EF00) is the screen attritubute control byte. Figure 3-3 defines the character set to be implemented.

3.3.2 CHARACTER PATTERN ROM. Dot patterns for each of the 96 displayable characters are stored in system ROM (CHARPAT) starting at address >~~EC00~~ 1000 and extending to >1EFF for a total of 768 bytes (96 times 8). Figure 3-4 defines the character patterns.

3.3.3 SCREEN ATTRIBUTE CONTROL. The byte immediately following the video display RAM (>1F00) is used to control the screen attributes relative to border and screen/character color. The bit definitions are as follows (B0 is the LSB):

B7-B3	B2	B1	B0
not used	screen color 1 = black	border color 1 = white	reserved for future use

3.3.4 BLANK-END-OF-LINE CONTROL. Any character code greater than >70 will be used as a Blank-End-Of-Line (BEOL) character which will stop the controller from refreshing the remainder of the line and relenquish control of the address and data buses. The BEOL feature should be used on all non-full display lines to provide more compute time for the CPU. Also, writing a >7X into the first character of a line will blank the line on the screen, however the video RAM is not cleared. This facilitates turning messages "on" and "off" on the screen. Video may be fully disabled by setting the VIDENA CRU bit to a low.

3.4 AUDIO CASSETTE INTERFACE

The audio cassette is useful only for program load/store functions. Since no CPU control of the recorder motor is provided, the use of the cassette to input or store data files would be inefficient because the user has to start the recorder prior to running the program. However, the ability to PRINT/INPUT to/from the cassette should not be ignored.

3.4.1 CRU BASE. The CRU address of the cassette I/O is at >E80E and is a bidirectional port.

3.4.2 DATA ENCODE/DECODE. Data is encoded as binary '1' and '0' on the cassette tape using a pulse width modulation technique. Since the cassette media for the GROUND SQUIRREL must be compatible with the TI-99/4A system, the software for the encoding should be as specified by the TI-99/4A BASIC Interpreter specification.

CODE (hex)	CHAR	CODE (hex)	CHAR	CODE (hex)	CHAR
00	⋮	26	&	4C	L
01	⋯	27	'	4D	M
02	⊙	28	(	4E	N
03	⊗	29	)	4F	O
04	≷	2A	*	50	P
05	X	2B	+	51	Q
06	/	2C	,	52	R
07	\	2D	-	53	S
08	⌐	2E	.	54	T
09	⌐	2F	/	55	U
0A	⌐	30	0	56	V
0B	⌐	31	1	57	W
0C	⌐	32	2	58	X
0D	⌐	33	3	59	Y
0E	⌐	34	4	5A	Z
0F	⌐	35	5	5B	[
10	⌐	36	6	5C	\
11	⌐	37	7	5D	]
12	⌐	38	8	5E	^
13	⌐	39	9	5F	
14	⌐	3A	:	60	▯
15	⌐	3B	;	61	▪
16	⌐	3C	<	62	▪
17	⌐	3D	=	63	▪
18	⌐	3E	>	64	▪
19	⌐	3F	?	65	▪
1A	⌐	40	@	66	▪
1B	{	41	A	67	▪
1C	¦	42	B	68	▪
1D	}	43	C	69	▪
1E	~	44	D	6A	▪
1F	`	45	E	6B	▪
20	space	46	F	6C	▪
21	!	47	G	6D	▪
22	"	48	H	6E	▪
23	#	49	I	6F	▪
24	$	4A	J	70--FF	BEOL
25	%	4B	K	(blank-end-of-line)	

Figure 3-3 DISPLAY CHARACTER SET

CODE	PATTERN	CHAR	CODE	PATTERN	CHAR
00	0000, 1000, 1000, 1000	⋮	38	0038, 4444, 3844, 4438	8
01	0000, 0000, 5400, 0000	⋯	39	0038, 4444, 3C04, 0830	9
02	3C42, 8199, 9981, 423C	⊙	3A	0000, 3030, 0030, 3000	:
03	3C42, A599, 99A5, 423C	⊗	3B	0000, 3030, 0030, 1020	;
04	0040, 2010, 7840, 2010	≈	3C	0008, 1020, 4020, 1008	<
05	8142, 2418, 1824, 4281	X	3D	0000, 007C, 007C, 0000	=
06	8040, 2010, 0804, 0201	/	3E	0020, 1008, 0408, 1020	>
07	0102, 0408, 1020, 4080	\	3F	0038, 4404, 0810, 0010	?
08	0000, 0000, 1F10, 1010	┌	40	0038, 445C, 545C, 4038	@
09	1010, 1010, 1F00, 0000	└	41	0010, 2844, 447C, 4444	A
0A	1010, 1010, F000, 0000	┐	42	0078, 2424, 3824, 2478	B
0B	0000, 0000, F010, 1010	┘	43	0038, 4440, 4040, 4438	C
0C	0000, 0000, FF00, 0000	─	44	0078, 2424, 2424, 2478	D
0D	1010, 1010, 1010, 1010	│	45	007C, 4040, 7840, 407C	E
0E	1010, 1010, FF00, 0000	┴	46	007C, 4040, 7840, 4040	F
0F	0000, 0000, FF10, 1010	┬	47	003C, 4040, 5C44, 4438	G
10	1010, 1010, F010, 1010	┤	48	0044, 4444, 7C44, 4444	H
11	1010, 1010, 1F10, 1010	├	49	0038, 1010, 1010, 1038	I
12	1010, 1010, FF10, 1010	┼	4A	0004, 0404, 0404, 4438	J
13	2854, 4444, 4428, 2810	♡	4B	0044, 4850, 6050, 4844	K
14	1028, 2844, 4444, 5428	△	4C	0040, 4040, 4040, 407C	L
15	0000, 7886, 4186, 7800	▷	4D	0044, 6C54, 5444, 4444	M
16	0000, 1E61, 8261, 1E00	◁	4E	0044, 6464, 544C, 4C44	N
17	0000, 1020, 7C20, 1000	←	4F	007C, 4444, 4444, 447C	O
18	0000, 1008, 7C08, 1000	→	50	0078, 4444, 7840, 4040	P
19	0000, 1038, 5410, 1000	↑	51	0038, 4444, 4454, 4834	Q
1A	0000, 1010, 5438, 1000	↓	52	0078, 4444, 7850, 4844	R
1B	0018, 2020, 4020, 2018	{	53	0038, 4440, 3804, 4438	S
1C	0010, 1010, 0010, 1010	¦	54	007C, 1010, 1010, 1010	T
1D	0030, 0808, 0408, 0830	}	55	0044, 4444, 4444, 4438	U
1E	0000, 2054, 0800, 0000	~	56	0044, 4444, 2828, 1010	V
1F	0000, 2010, 0800, 0000	`	57	0044, 4444, 5454, 5428	W
20	0000, 0000, 0000, 0000	space	58	0044, 4428, 1028, 4444	X
21	0010, 1010, 1010, 0010	!	59	0044, 4428, 1010, 1010	Y
22	0028, 2828, 0000, 0000	"	5A	007C, 0408, 1020, 407C	Z
23	0028, 287C, 287C, 2828	#	5B	0038, 2020, 2020, 2038	[
24	0038, 5450, 3814, 5438	$	5C	0000, 4020, 1008, 0400	\
25	0060, 6408, 1020, 4C0C	%	5D	0038, 0808, 0808, 0838	]
26	0020, 5050, 2054, 4834	&	5E	0000, 1028, 4400, 0000	^
27	0008, 0810, 0000, 0000	'	5F	0000, 0000, 0000, 007C	_

Figure 3-4 CHARACTER PATTERNS

CODE	PATTERN	CHAR	CODE	PATTERN	CHAR
28	0008,1020,2020,1008	(	60	FF81,8181,8181,81FF	
29	0020,1008,0808,1020	)	61	FFFF,FFFF,FFFF,FFFF	
2A	0000,2810,7C10,2800	*	62	F0F0,F0F0,0000,0000	
2B	0000,1010,7C10,1000	+	63	0F0F,0F0F,FFFF,FFFF	
2C	0000,0000,0030,1020	,	64	0F0F,0F0F,0000,0000	
2D	0000,0000,7C00,0000	-	65	F0F0,F0F0,FFFF,FFFF	
2E	0000,0000,0000,3030	.	66	0000,0000,0F0F,0F0F	
2F	0000,0408,1020,4000	/	67	FFFF,FFFF,F0F0,F0F0	
30	0038,4444,4444,4438	0	68	0000,0000,F0F0,F0F0	
31	0010,3010,1010,1038	1	69	FFFF,FFFF,0F0F,0F0F	
32	0038,4404,0810,207C	2	6A	F0F0,F0F0,0F0F,0F0F	
33	0038,4404,1804,4438	3	6B	0F0F,0F0F,F0F0,F0F0	
34	0008,1828,487C,0808	4	6C	F0F0,F0F0,F0F0,F0F0	
35	007C,4078,0404,4438	5	6D	0F0F,0F0F,0F0F,0F0F	
36	0018,2040,7844,4438	6	6E	FFFF,FFFF,0000,0000	
37	007C,0408,1020,2020	7	6F	0000,0000,FFFF,FFFF	

Figure 3-4 CHARACTER PATTERNS cont'd

3.5 ALC I/O PORT

The ALC I/O port is used by the GROUND SQUIRREL console as a printer port, to access a low cost printer system that is compatible with the ALC I/O system. System LIST and PRINT functions will default to the screen BUT can be routed to the ALC I/O port by use of the file handling features OPEN, CLOSE and the alternate forms of PRINT #n or LIST #n for output instructions. DSRs required to effectively utilize future ALC I/O devices, may be contained in the appropriate expansion modules.

3.5.1 CRU BASE. CRU base address for the ALC I/O will at >E800 to >E80C. The first 6 CRU bits are bi-directional. The 7th bit (>E80C) is the message inhibit bit (INH) and is an output only. The INH serves two functions:

* as long as the bit is low, normal ALC I/O functions are allowed. If the bit is a one, all data lines, handshake and bus available functions are disabled.

* INH is toggled high-to-low as a 'message ignore' indication. After a frame is initiated by some device on the line, other than the GROUND SQUIRREL, software will read the Device ID byte sent by the requesting

device. If the ID does not match the GROUND SQUIRREL, toggling the INH bit will inhibit the ALC I/O port from responding to any more nibbles (i.e. handshakes) until the next message frame, as signified by the falling edge of BAV.

3.5.2 ALC I/O EXPANSION DSRs. DSRs required to implement 'user friendly' algorithms to facilitate ALC I/O peripherals ,other than a printer, may be provided in ROMs in an expansion module.

3.5.3 SOLID STATE SOFTWARE. Preprogrammed application packages or enhancement features will be supported by ROM/RAM in expansion modules.

3.5.4 SELF-TEST SOFTWARE. Unique self-test software will be accessed through the expansion port module. Initation of self-test algorithms will be by an interrupt line at the expansion port or by an unique keyboard sequence.

SECTION 4

SYSTEM SOFTWARE

The GROUND SQUIRREL Computer System will utilize an interpretively operated, high level language software environment.

4.1 USER INTERFACE

The User Interface provided by the screens, menus and operational modes will be compatible with the features defined for the 99/4A system.

4.1.1 POWER-UP/RESET. At Power-up or Reset, the video display will be off (VIDENA=0). System software will set the Video Control byte (>1F00) to >02 and then interrogate the expansion module to identify the resource configuration relative to RAM, ROM, DSRs and solid state software packages that are available. A screen image will be developed that is similar to the power-up screen of the 99/4A, within the architectural limitations of the GROUND SQUIRREL system.

4.1.2 PROGRAM SELECTION MENU. After power-up the screen is displayed and when the user touches any key, a menu will be presented that identifies all programs available for execution. The first entry will be TI-BASIC and then all programs available in the expansion module. Selection of the programs will be in a manner functionally equivalent to the 99/4A methodology.

4.1.3 MODES OF OPERATION. The software shall provide modes of operation that are functionally equivalent to the following TI-99/4A modes:

(1) EDIT -- for entering and editing programs
(2) BASIC Command Execution -- execution of BASIC instructions
(3) BASIC Program Execution -- execution of BASIC program steps

4.1.4 PROMPTING AND ERROR HANDLING. Software will provide user prompting and error messages that are functionally equivalent to that provided in the 99/4A, for features that are included in the

GROUND SQUIRREL system. Some unique messages will be required for handling the occurance of commands/instructions that are not supported.

4.2 BASIC INTERPRETER

The BASIC interpreter software will incorporate a maximized subset of the BASIC system provided by the 99/4A. To insure compatibility, the commands included in this interpreter will be implemented as defined by the TI-99/4A Interpreter specification and will function, to the maximum extent possible, in the same manner as for the 99/4A. Variations in functionality will be as a result of unique GROUND SQUIRREL market demands and hardware limitations.

4.2.1 FEATURES TO BE INCLUDED. The subset of TI-99/4A Console BASIC to be included are defined in Table 4-1. All features that are negoitable are marked with an "*".

4.2.2 ADDITIONAL FUNCTIONS. TI-99/4A Console BASIC does not support assembly language capabilities (only with Extended BASIC). However, the GROUND SQUIRREL has a market defined need for supporting these function to fulfill the requirement as a "computer literacy" learning aid. Therefore, this BASIC interpreter will support the following Extended BASIC functions:

PEEK	INIT
LOAD	LINK

4.2.3 VARIABLES. This interpreter shall support both numeric and string variables with names at least 8 characters in length. Actual length will be defined by resource limitations for name storage.

4.2.4 EXPRESSIONS AND OPERATORS. The GROUND SQUIRREL BASIC interpreter will support numeric, string and relational operators as defined for the 99/4A.

Table 4-1 BASIC INTERPRETER FEATURES

INSTRUCTION TYPE	FEATURES	
COMMANDS	NEW	BYE
	* BREAK	* UNBREAK
	* TRACE	* UNTRACE
	EDIT	RUN
	OLD	SAVE
	DELETE	LIST
	* NUMBER	* RESEQUENCE
PROGRAM STATEMENTS	LET	REM
	END	STOP
	GOTO	ON-GOTO
	GOSUB	ON-GOSUB
	RETURN	IF-THEN-ELSE
	FOR-TO-NEXT	DIM
	OPTION BASE	
I/O STATEMENTS	READ	DATA
	RESTORE	INPUT
	DISPLAY	PRINT
	OPEN	CLOSE
BUILT-IN FUNCTIONS	ABS	ATN
	COS	EXP
	INT	LOG
	RANDOMIZE	RND
	SGN	SIN
	SQR	TAN
	ASC	CHR$
	LEN	POS
	SEG$	STR$
	VAL$	
BUILT-IN SUBROUTINES	CLEAR	SCREEN
	HCHAR	VCHAR
	GCHAR	KEY

4.2.5 UNSUPPORTERD TI-99/4A FEATURES. When reading cassette tapes that contain 99/4A Console BASIC programs, the features that are not supported by the GROUND SQUIRREL must be processed. Some of the unsupported features will produce fatal execution errors, if ignored, but others will cause no effect on program execution (i.e. CALL COLOR, CALL SOUND, etc....). All unsupported features that cause fatal run errors must be flagged with an ERROR message at program read time. Non-fatal unsupported features may be ignored. Table 4-2 identifies the categories of unsupported features.

Table 4-2 UNSUPPORTED CONSOLE BASIC FEATURES

COMMANDS TO BE IGNORED	COMMANDS TO BE FLAGGED
COLOR SOUND	JOYST CHAR
EOF	

4.3 DEVICE SERVICE ROUTINES

Device operations (INPUT, LIST/PRINT, OPEN, CLOSE, etc....) will be implemented using the File I/O methodology utilized in the 99/4A, within the limitations of the GROUND SQUIRREL architecture. At power-up, system software will interrogate the hardware configurations, including the expansion port, to identify all DSRs that are available and build a file I/O access block in processor RAM. In the event of a conflict between a console based DSR and an equivalent function DSR in the expansion module, the expansion module DSR will take priority. This will allow maximum flexibility for future expansion of the GROUND SQUIRREL system.

4.3.1 VIDEO DISPLAY. The Video DSR will control access to the video display RAM by writing character codes into the 768 bytes starting at >ECOO. This DSR should be callable by programs in the expansion module to facilitate video control for application packages or other DSRs. Software will be responsible for positioning the CURSOR which will be a filled rectangle and will "blink" at a 2 Hz rate in the appropriate modes. Display scrolling will be performed by the software in such a manner as to shift the entire screen up one line. To maximize CPU processing time, a BEOL character (>70 or greater) will be

inserted as the last character of each line of less than 32 characters. Video on/off is controlled by writing a 1/0 to CRU bit >E00E (VIDENA). Border and screen colors are controlled by placing a 1/0 (white/black) in the 2nd and 3rd bits, respectively, of the byte at >EF00.

4.3.2 KEYBOARD. The keyboard DSR will control all access to the keyboard for identification of key inputs. All system level and expansion module software will use this DSR for accessing the keyboard. This facilitates the ability to "redefine" the keyboard DSR with an expansion module DSR via the I/O access block. Keyboard scan and decode methodology should be patterned after the TI-99/4A Console software.

4.3.3 AUDIO CASSETTE. The audio cassette DSR must provide BASIC program storage on tape that is totally compatible with the 99/4A system. Thus, the specification is the same as for the 99/4A BASIC interpreter. To insure adequate CPU time for cassette I/O operations, the CRT must be disabled by writing a 0 to CRU base >E00E.

4.3.4 ALC I/O PORT. This DSR will provide access to the ALC I/O port for two types of operations:

* Printer output via the LIST #n and PRINT #n commands of BASIC. The equivalent LIST "TP" and PRINT "TP" commands will default to the ALC I/O printer connected to the bus.

* Primitive input/output commands will be provided to accomodate other ALC I/O bus peripherals that will be available. These primitive commands will be PRINT #n and INPUT #n and will be accessed through the OPEN/CLOSE features of the File I/O system.

4.3.5 I/O EXPANSION MODULE. The system software will, at power-up, search for and store away the existance of any DSRs in the expansion module. DSRs in the expansion module take prescedence over equivalent DSRs in the console.

4.4 EXPANSION MODULE SOFTWARE

System software will include features to allow identification, recognition (for menus) and execution of solid state software programs or extended BASIC enhancement software in the expansion module. That is, the BASIC interpreter software must be capable of identifing extended or modified BASIC token tables contained in the expansion module. To optimally serve the market need for a flexible "computer literacy" learning aid, the system software and command interpreter must be able to recognize commands that are not common to BASIC but have interpretation tables in the expansion module. This will allow development of a rudimentary FORTRAN, PASCAL, LOGO, etc.... learning aid. Although the system will not function exactly as a PASCAL or FORTRAN based machine, the capability will exist to learn the fundamental constructs and syntax of these languages.

4.5 SELF-TEST SOFTWARE

Software will provide features and capabilities to implement a system level self-test.

4.5.1 ROM CRC CHECK. The first two bytes of every 4 kbyte ROM boundary contain the CRC for that 4 kbyte segment. However, because of the system vectoring, the first 8 kbyte ROM will have the CRC for the entire 8 kbytes stored at location >0014 and >0015.

4.5.2 RAM BIT TEST. A RAM bit test will be provided to determine the amount of RAM in the system and the functionality of all RAM bits. The data pattern to be used will be 'AA' and '55'.

4.5.3 CASSETTE INTERFACE TEST. The cassette I/O interface will be tested by a 'bit echo' technique, whereby external test equipment will inject a signal into the CASIN (recorder output) line and monitor the CASOUT (recorder input) line for a duplicate of the original signal. Software will read the input data and echo the bits back to the output line.

4.5.4 ALC I/O PORT PRINTER TEST. The software will transmit a 'SYSTEM READY' message to the printer port.

4.5.5 VIDEO DISPLAY TEST. The test for the video will be to continously repeat the entire 96 character set with a blank line between each repetition. The characters will be written at a rate of not less than 200 characters/second and scrolling will be in effect after each 24 lines.

4.5.6 EXPANSION PORT BASED TEST PROGRAMS. The expansion port/module will contain a unique test program, located at an address specified by the 2nd and 3rd bytes of the ROM. These programs will be developed as part of the test systems effort and are not part of the system software. However, the software must be capable of recognizing the test programs and initiating the tests.

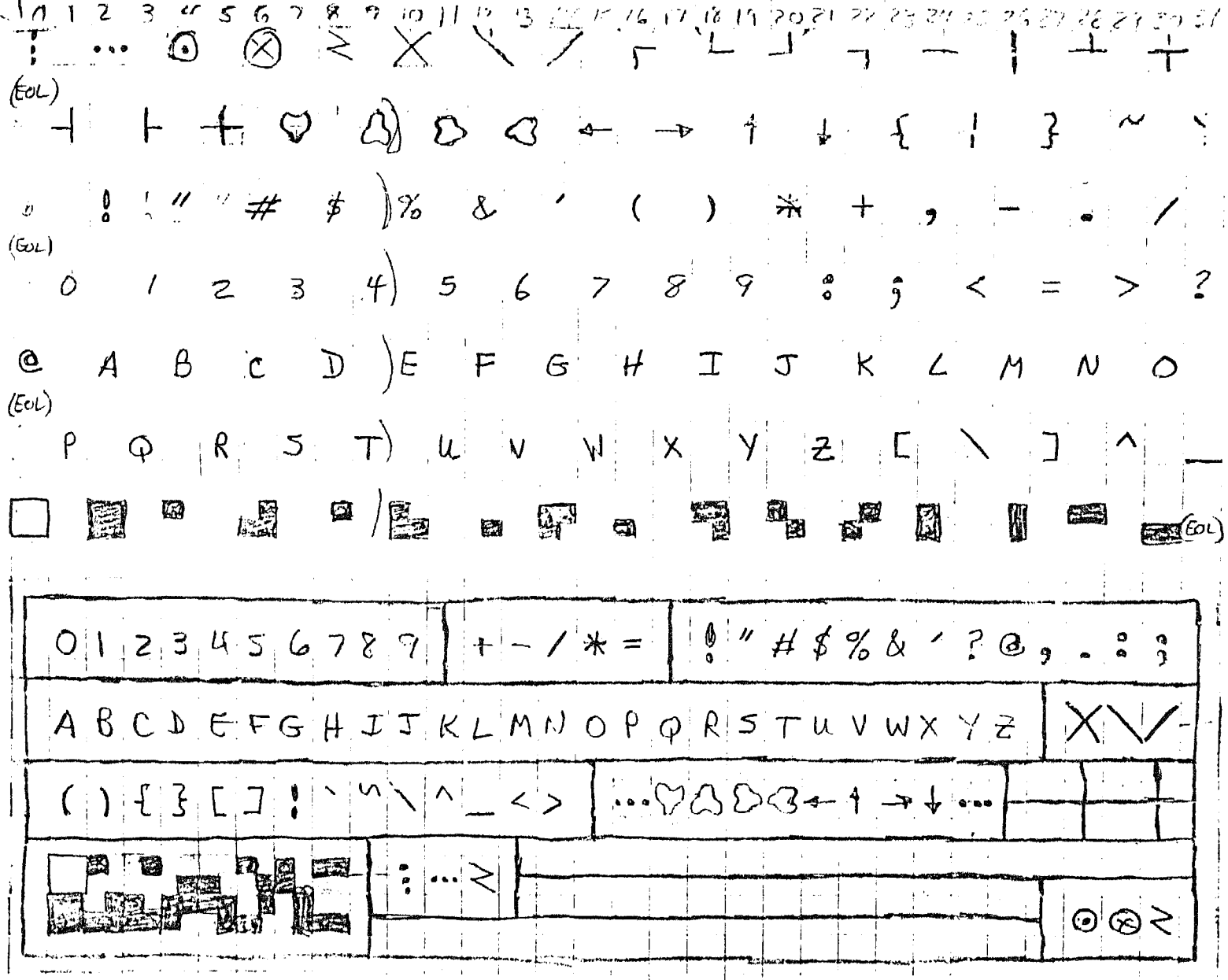
ROM ADDR	CHAR CODE (HEX)	CHAR	VIDEO BITS B0	B1	B2	B3	B4	B5	B6	B7
0000	00	0	00	08	00	08	00	08	00	08
0008	01	1	00	00	00	00	55	00	00	00
0010	02	2	3C	42	81	99	99	81	42	3C
0018	03	3	3C	42	A5	99	99	A5	42	3C
0020	04	4	20	10	08	7C	20	10	08	04
0028	05	5	81	42	24	18	18	24	42	81
0030	06	6	80	40	20	10	08	04	02	01
0038	07	7	01	02	04	08	10	20	40	80
40	08	8	00	00	00	00	0F	08	08	08
48	09	9	08	08	08	08	CF	00	00	00
50	0A	10	08	08	08	08	F8	00	00	00
58	0B	11	00	00	00	00	F8	08	08	08
60	0C	12	00	00	00	00	FF	00	00	00
68	0D	13	08	08	08	08	0F	08	08	08
70	0E	14	08	08	08	08	FF	00	00	00
78	0F	15	00	00	00	00	FF	08	08	08
80	10	16	08	08	08	08	F8	08	08	08
88	11	17	08	08	08	08	0F	08	08	08
90	12	18	08	08	08	08	FF	08	08	08
98	13	19	24	5A	42	42	42	24	24	18
00A0	14	20	18	24	24	42	42	42	5A	24
00A8	15	21	00	78	86	41	41	86	78	00
B0	16	22	00	1E	61	82	82	61	1E	00
B8	17	23	00	00	10	20	7C	02	10	00
C0	18	24	00	00	10	08	7C	08	10	00
C8	19	25	00	00	10	32	5C	10	10	00
D0	1A	26	00	00	10	10	54	38	10	00
D8	1B	27	00	08	10	10	20	10	10	08
E0	1C	28	00	10	10	10	00	10	10	10
E8	1D	29	00	20	10	10	00	10	10	20
F0	1E	30	00	20	54	08	00	00	00	00
00F8	1F	31	00	18	18	10	08	00	00	00

Ram ADD	CHAR CODE (HEX)		CHAR	VIDEO BYTES B0	B1	B2	B3	B4	B5	B6	B7
0100	20	32	space	00	00	00	00	00	00	00	00
0108	21	33	!	00	04	04	04	04	04	00	04
0110	22	34	"	00	0A	0A	0A	00	00	00	00
0118	23	35	#	00	0A	0A	1F	0A	1F	0A	0A
0120	24	36	$	00	04	0F	14	0E	05	1E	04
0128	25	37	%	00	18	19	02	04	08	13	03
130	26	38	&	00	08	14	14	08	15	12	0D
138	27	39	'	00	0C	0C	08	10	00	00	00
140	28	40	(	00	02	04	08	08	08	04	02
148	29	41	)	00	08	04	02	02	02	04	08
150	2A	42	*	00	04	15	0E	1F	0E	15	04
158	2B	43	+	00	00	04	04	1F	04	04	00
160	2C	44	,	00	00	00	00	0C	0C	08	10
168	2D	45	-	00	00	00	00	1F	00	00	00
170	2E	46	.	00	00	00	00	00	00	0C	0C
178	2F	47	/	00	00	01	02	04	08	10	00
180	30	48	0	00	0E	11	13	15	19	11	0E
188	31	49	1	00	04	0C	04	04	04	04	0E
190	32	50	2	00	0E	11	01	0E	10	10	1F
198	33	51	3	00	0E	11	01	06	01	11	0E
1A0	34	52	4	00	02	06	0A	12	1F	02	02
1A8	35	53	5	00	1F	10	1E	01	01	11	0E
1B0	36	54	6	00	06	08	10	1E	11	11	0E
1B8	37	55	7	00	1F	01	02	04	08	10	10
1C0	38	56	8	00	0E	11	11	0E	11	11	0E
1C8	39	57	9	00	0E	11	11	0F	01	02	0C
1D0	3A	58	:	00	00	0C	0C	00	0C	0C	00
1D8	3B	59	;	00	0C	0C	00	0C	0C	08	10
1E0	3C	60	<	00	02	04	08	10	08	04	02
1E8	3D	61	=	00	00	00	1F	00	1F	00	00
1F0	3E	62	>	00	08	04	02	01	02	04	08
1F8	3F	63	?	00	0E	11	01	02	04	00	04

96
15
112

ROM ADDR	CHAR CODE (HEX)	CHAR		VIDEO BITS B0	B1	B2	B3	B4	B5	B6	B7
200	40	64	@	00	0E	11	01	0D	15	15	0E
208	41	65	A	00	04	0A	11	11	1F	11	11
210	42	66	B	00	1E	09	09	0E	09	09	1E
218	43	67	C	00	0E	11	10	10	10	11	0E
220	44	68	D	00	1E	0A	0A	0A	0A	0A	1E
228	45	69	E	00	1F	10	10	1C	10	10	1F
230	46	70	F	00	1F	10	10	1C	10	10	10
238	47	71	G	00	0F	10	10	13	11	11	0F
240	48	72	H	00	11	11	11	1F	11	11	11
248	49	73	I	00	0E	04	04	04	04	04	0E
250	4A	74	J	00	01	01	01	01	01	11	0E
258	4B	75	K	00	11	12	14	18	14	12	11
260	4C	76	L	00	10	10	10	10	10	10	1F
268	4D	77	M	00	11	1B	15	15	11	11	11
270	4E	78	N	00	11	19	15	13	11	11	11
278	4F	79	O	00	0E	11	11	11	11	11	0E
280	50	80	P	00	1E	11	11	1E	10	10	10
288	51	81	Q	00	0E	11	11	11	15	12	0D
290	52	82	R	00	1E	11	11	1E	14	12	11
298	53	83	S	00	0E	11	10	0E	01	11	0E
2A0	54	84	T	00	1F	04	04	04	04	04	04
2A8	55	85	U	00	11	11	11	11	11	11	0E
2B0	56	86	V	00	11	11	11	0A	0A	04	04
2B8	57	87	W	00	11	11	11	11	15	1B	11
2C0	58	88	X	00	11	11	0A	04	0A	11	11
2C8	59	89	Y	00	11	11	0A	04	04	04	04
2D0	5A	90	Z	00	1F	01	02	04	08	10	1F
2D8	5B	91	[	00	0E	08	08	08	08	08	0E
2E0	5C	92	\	00	00	10	08	04	02	01	00
2E8	5D	93	]	00	0E	02	02	02	02	02	0E
2F0	5E	94	^	00	04	0A	11	00	00	00	00
2F8	5F	95	_	00	00	00	00	00	00	00	1F

ROM ADDR	CHAR CODE (HEX)	CHAR		VIDEO BITS B0	B1	B2	B3	B4	B5	B6	B7
300	60	96	□	FF	81	81	81	81	81	81	FF
308	61	97	▓	FF	FF	FF	FF	FF	FF	FF	FF
310	62	98	▪	F0	F0	F0	F0	00	00	00	00
318	63	99	▦	0F	0F	0F	0F	FF	FF	FF	FF
320	64	100	▪	0F	0F	0F	0F	00	00	00	00
328	65	101	▬	F0	F0	F0	F0	FF	FF	FF	FF
330	66	102	▪	00	00	00	00	0F	0F	0F	0F
338	67	103	▀	FF	FF	FF	FF	F0	F0	F0	F0
340	68	104	▪	00	00	00	00	F0	F0	F0	F0
348	69	105	▤	FF	FF	FF	FF	0F	0F	0F	0F
350	6A	106	▪	F0	F0	F0	F0	0F	0F	0F	0F
358	6B	107	▪	0F	0F	0F	0F	F0	F0	F0	F0
360	6C	108	▪	F0	F0	F0	F0	F0	F0	F0	F0
368	6D	109	▪	0F	0F	0F	0F	0F	0F	0F	0F
370	6E	110	▬	FF	FF	FF	FF	00	00	00	00
378	6F	111	▬	00	00	00	00	FF	FF	FF	FF
380	70										
388	71										
390	72										
398	73										
3A0	74										
3A8	75			NO VIDEO							
3B0	76										
3B8	77										
3C0	78										
3C8	79										
3D0	7A										
3D8	7B										
3E0	7C										
3E8	7D										
3F0	7E										
3F8	7F										


SLM
ROW
(EOL)
(EOL)
(EOL)
(EOL)
0 1 2 3 4 5 6 7 8 9
+ - / * =
! " # $ % & ' ? @ , . : ;
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
( ) { } [ ] ! ` " \ ^ _ < >
: ... ≥