The information contained herein is subject to change without notice.

Reproduction of this manual either in part or in entirety is forbidden.

Note that the manufacturer assumes no responsibility for any injury or loss incurred while using this manual.
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM VARIABLES</td>
<td>73</td>
</tr>
<tr>
<td>YEAR</td>
<td>73</td>
</tr>
<tr>
<td>MONTH</td>
<td>74</td>
</tr>
<tr>
<td>DATE</td>
<td>75</td>
</tr>
<tr>
<td>HOUR</td>
<td>76</td>
</tr>
<tr>
<td>MINUTE</td>
<td>77</td>
</tr>
<tr>
<td>SECOND</td>
<td>78</td>
</tr>
<tr>
<td>IDNUMBER$</td>
<td>79</td>
</tr>
<tr>
<td>APPENDIX A LIMITATIONS OF KEYBOARD CONTROL LANGUAGE</td>
<td>80</td>
</tr>
<tr>
<td>APPENDIX B ERROR MESSAGES</td>
<td>81</td>
</tr>
<tr>
<td>APPENDIX C RS-232 PIN CONFIGURATIONS</td>
<td>84</td>
</tr>
<tr>
<td>OPERATION GUIDE</td>
<td>87</td>
</tr>
</tbody>
</table>
Document Conventions

This manual uses the following typographic conventions

<table>
<thead>
<tr>
<th>Examples of convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; &gt;</td>
<td>Angle brackets, enclose mnemonic representations of ASCII control characters.</td>
</tr>
<tr>
<td>[ ]</td>
<td>Square brackets, the data within square brackets is optional.</td>
</tr>
<tr>
<td>Ctrl</td>
<td>Bold letters represent a key on the keyboard.</td>
</tr>
<tr>
<td>Ctrl-C</td>
<td>Two keys with a dash, means to press them simultaneously.</td>
</tr>
<tr>
<td><em>Italic</em></td>
<td>Italic letters represent explanations given in the context.</td>
</tr>
<tr>
<td><strong>Bold Italic</strong></td>
<td>Note. Important information.</td>
</tr>
</tbody>
</table>
Quick Start

1. Please verify if the pin configuration of keyboard COM1 port (keyboard cable) is compatible with your printer COM port. If they are not compatible, please refer to section Specification and Appendix A for more information.
2. Turn off printer power. Connect keyboard COM1 port to printer serial port.
3. Press F1 to setup the communication parameters of keyboard. Please refer to section Keyboard Utilities for more information.
4. Press ALT-EXIT to reset keyboard. It is ready for downloading the program.
5. Key in the program as below:
   ```bas
   CLS
   PRINT “HELLO !”
   A=INKEY()
   END
   
   NOTE: TSKL only accepts capital letters.
   ```
6. Save the file with filename of “DEMO.BAS”.
7. Select Utilities | Download File to download program file to keyboard.
   
   Note: When download programs (*.BAS), please select BAS file extension in the Download file dialog box.
8. A sharp and short beep sound means the file has been downloaded to keyboard memory.
9. Press FORM key and to select the ↑ and ↓ to execute downloaded program.
10. “HELLO !” will be shown on the LCD display. Press any key to end the program.
11. Press ALT-EXIT to reset printer.
Features of SMART KEYBOARD

- 68 keys large keypad layout
- Big LCD screen (20 characters × 2 lines)
- Additional RS-232 port for other input device
- Maximum 50 files can be stored in memory (SRAM and FLASH)
- Upload or download files through both serial port
- Real Time Clock (Y2K compatible)
- Built-in Euro logo (ASCII 176, 177)
- Floating point calculation
- Password locking security
- Auto execution function

Specifications

- Keyboard Unit

<table>
<thead>
<tr>
<th>Size</th>
<th>261mm(L)×142.2mm(W)×31mm(H)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>440g</td>
</tr>
<tr>
<td>LCD</td>
<td>20 characters × 2 lines</td>
</tr>
<tr>
<td>Max Current</td>
<td>5V, 250mA</td>
</tr>
<tr>
<td>External Power</td>
<td>5V, 150mA</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>40°F<del>104°F (5°C</del>40°C)</td>
</tr>
<tr>
<td>Storage Temperature</td>
<td>-4°F<del>140°F (-20°C</del>60°C)</td>
</tr>
</tbody>
</table>

- Memory

<table>
<thead>
<tr>
<th>FLASH</th>
<th>512 KB for system, 1 MB for application</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM</td>
<td>128 KB for system, 128KB for application</td>
</tr>
</tbody>
</table>

Communication Interface

The available communication parameters for both 2 serial ports are listed as below:
Baud rate: 2400, 4800, 9600, 19200 bps
Parity check: none, even or odd
Smart Keyboard Programming Manual

Data bits: 7 or 8
Stop bit(s): 1 or 2

COM1 Port

The serial interface COM1 is a 9-pin, male D-style subminiature connector with cable, the pin assignments are shown as below:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Power input 5 volts, 250mA</td>
</tr>
<tr>
<td>2</td>
<td>RxD</td>
</tr>
<tr>
<td>3</td>
<td>TxD</td>
</tr>
<tr>
<td>4</td>
<td>DTR</td>
</tr>
<tr>
<td>5</td>
<td>Ground</td>
</tr>
<tr>
<td>6</td>
<td>DSR</td>
</tr>
<tr>
<td>7</td>
<td>RTS</td>
</tr>
<tr>
<td>8</td>
<td>CTS</td>
</tr>
<tr>
<td>9</td>
<td>Connect with Pin1 internally</td>
</tr>
</tbody>
</table>

COM2 Port

The COM2 is a 9-pin, female D-style subminiature connector. The pin assignments are shown as below:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Power input 5 volts, 250mA</td>
</tr>
<tr>
<td>2</td>
<td>TxD</td>
</tr>
<tr>
<td>3</td>
<td>RxD</td>
</tr>
<tr>
<td>4</td>
<td>DSR</td>
</tr>
<tr>
<td>5</td>
<td>Ground</td>
</tr>
<tr>
<td>6</td>
<td>DTR</td>
</tr>
<tr>
<td>7</td>
<td>CTS</td>
</tr>
<tr>
<td>8</td>
<td>RTS</td>
</tr>
<tr>
<td>9</td>
<td>Connect with Pin1 internally</td>
</tr>
</tbody>
</table>
Safety Regulation

FCC Class A
CE Class A

Check-List

Verify the contents of the container according to the list below, if any parts are missing, please contact your local representative.

- SMART KEYBOARD keyboard unit
- SMART KEYBOARD User’s Manual
- 25 pin to 9 pin RS-232 converter
- 2 screws for SI thread
- Codepage stickers

Options

- External power set
  - AC adapter
    - Input: 110V AC or 220V AC
    - Output: 5V DC
  - RS-232 cable with power adapter jack
Keyboard Setup

Please turn off printer power prior connecting keyboard COM1 port to printer. COM2 port of keyboard is used for download files from PC or can be connected to other input devices such as bar code scanner.

Keyboard Operation

● Power-on Utilities

■ Auto Execution
SMART KEYBOARD will execute the program automatically when turning on power without pressing any key, If one of the downloaded programs is named “AUTO.BAS” Please presses ALT-F1 first to disable the auto execution functions prior keyboard configuration setup, on-line editing, initialization and extend files manager utilities if the auto execution file is downloaded in keyboard.

■ Keyboard Initialization
Press the ALT-CLEAR while turning on printer power will initialize the keyboard. All files stored in the SRAM will be deleted. Code page and country codes are set to defaults value of 437 and 001 respectively. Both of serial port communications parameters are set to 9600 bps, none parity check, 8 data bits and 1 stop bit.

<table>
<thead>
<tr>
<th>Items</th>
<th>Default Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>COM1 Port</td>
<td>9600,N,8,1,Xon/Xoff</td>
</tr>
<tr>
<td>COM2 Port</td>
<td>9600,N,8,1,DSR/DTR</td>
</tr>
<tr>
<td>Country Code</td>
<td>001</td>
</tr>
<tr>
<td>Code Page</td>
<td>437</td>
</tr>
<tr>
<td>Key Sound</td>
<td>On</td>
</tr>
<tr>
<td>ID Number*</td>
<td>T.S.C.</td>
</tr>
<tr>
<td>Password*</td>
<td>None</td>
</tr>
</tbody>
</table>

Note: The items with asterisk (*) can’t be initialized.
Keyboard Configuration Setup

Keyboard configurations can be changed by pressing **ALT-F1** keys. The left and right arrow keys are used to select different parameters. Press **ENTER** key to enable the selected parameter. Press **EXIT** to return to upper level of menu.

- **COM1, COM2 SETUP:**
  With COM1, COM2 SETUP, users can select different communication parameter as well as handshaking to download or upload data.

- **DATE / TIME SETUP:**
  The built-in Real Time Clock is compatible with year 2000. The leap year timing is automatic. The available setting of year is from 1950 to 2049.

- **INFORMATION:**
  The information lists the available memory (SRAM) in system and the extended Flash memory status. If the extended memory is installed, it shows 8M else “None” is shown on the display.

- **UTILITY SETUP:**
  The code page, country code and key sound on/off is set in UTILITY SETUP. The available code pages and country code are listed as below:

  - Code pages
437: United States
850: Multilingual
852: Slavic
860: Portuguese
863: Canadian/French
865: Nordic

- Country codes
  001: USA
  002: Canadian French
  003: Spanish (Latin America)
  031: Dutch
  032: Belgian
  033: French (France)
  034: Spanish (Spain)
  036: Hungarian
  038: Yugoslavian
  039: Italian
  041: Switzerland
  042: Slovak
  044: United Kingdom
  045: Danish
  046: Swedish
  047: Norwegian
  048: Polish
  049: Germany
  055: Brazil
  061: English (International)
  351: Portuguese
  358: Finnish

There is a beep sound after each key strokes. It can be disabled by set the key sound off.

- PASSWORD SETUP:

SMART KEYBOARD supports password security. With password locked on, the download, uploads programs, on-line editing, deleting files, firmware upgrade and extend files manager will be disabled. The maximum characters of password are up to 14 characters, which are not case sensitive.

Note: If the password is lost, please contact the Customer Service Department of your distributor or reseller.
SMART KEYBOARD provides another password namely ID Number, which can be verified in the program. The allowable maximum number of characters of ID number is 49. The ID number is not case sensitive. The default ID number is T.S.C.

**Keyboard BIOS Update**

Press **ALT-SHIFT** keys is used to enter the update BIOS utility. Users can upgrade the BIOS by copying the firmware through specified serial port to keyboard in DOS environment. If the check sum of BIOS is not correct after upgraded, the “UPDATE FAIL” will show on LCD screen. Please turn off power and enter upgrade mode again. If the upgrade process is successful, SMART KEYBOARD will boot automatically.
On-Line Editing Utility

On-line editing utility can be accessed by pressing ALT-F2 keys. This feature is useful for modifying programs in the field. The available maximum editing characters are 79 characters x 250 lines. The upward and downward arrow keys are used to scroll the files displayed on LCD screen. Enter key is used to select program for editing. ALT-CLEAR keys are used to delete the files stored in keyboard memory. F1 key is used to upload the files through the specified serial port of keyboard to the connected device if the cursor is stopped at selected file. Select “New File” and press Enter to editing a new file. The available filename is up to 8 characters and 3 characters for extension. Press F2 to save the file when completed editing.
Extend Files Manager

Extend Files Manager can be accessed by press ALT-D keys. Files stored in flash memory are read only and file list, deleting, and download utility must be operated through Extended Files Manager. The download and delete files process will be different than files stored in standard memory. The operations of Extended Files Manager is described as below:

- **LIST FILES:**
  All the files stored in Flash memory will be listed. The BASIC files stored in Flash memory will also be listed in BASIC program file list by press Form key.

- **ERASE FILES:**
  This utility delete all the files stored in Flash memory. The “KILL” command can’t delete files stored in Flash memory.

- **DOWNLOAD FILES:**
  Enter this utility before download files to Flash memory. Download files either by DOS command or by SMART KEYBOARD Windows software. Press EXIT key to exit download utility.

  **Note:**
  1. **All files will be deleted when invoked DOWNLOAD FILES utility. Copy all the files again although only one file is appended.**
  2. **With password locked on, the utilities above will be disabled.**

Execute downloaded BASIC program

Press FORM key to list all the BASIC files stored in smart keyboard. The upper and downward arrow keys are used to scroll files displayed in LCD screen. SMART
KEYBOARD also supports hot key feature to find the program. For example: If the program is named DEMO.BAS, press **FORM** and then press **D**. It scrolls the files to the first one that begins with D automatically. Execute the selected program by press **ENTER** key.
Using SMART KEYBOARD

By the end of this section, you will be able to
1. Edit a TSKL program file on PC.
2. Download the program file to keyboard.
3. Execute the program.

Edit A Program

To edit a program file, you need a plain text editor, for example: DOS Editor, or Windows
NotePad

The following example is editing by general text editor.
1. Open a new text file.
2. Add the header at the first line of the file as listed below:
   
   DOWNLOAD “MYPROG.BAS”

   Where “MYPROG.BAS” is the file name to be stored in keyboard memory.

3. Edit the program contents.
4. Insert a line of command at the end of program as below:
   
   EOP

5. Save this file.
Download A Program

Users can download the program by DOS command.
Connect PC and smart keyboard by board COM2 port.
Connect smart keyboard COM1 RS-232 cable to printer.

- Download the program by DOS command:
  Enter the commands as below:
  C:\> MODE COM2 96,N,8,1
  C:\> COPY MYPROG.BAS /B COM2
  Where MYPROG.BAS is the name of program file, COM2 is the serial port of
  PC.

A sharp and short beep sound means the file has been downloaded to keyboard memory.

Execute A Program

Reset smart keyboard by pressing ALT-EXIT.
Press FORM to list program files.
Press ↑ and ↓ to select a program to execute.
If you want to interrupt the program, press ALT-EXIT.
Variables, Constants, Functions, Operators, and Expressions

The information in this chapter will help you to learn about variables, constants, functions, operators, and expressions in TSKL. Variables and constants are manipulated by operators to form expressions.

Variables

Variables are placeholders used to store values; they have names and data types. The data type of a variable determines how the bits representing those values are stored in the computer's memory. When you declare a variable, you can also supply a data type for it. All variables have a data type that determines what kind of data they can store.

The variable name in TSKL can vary from one to ten characters. The first character must be a letter or an underscore with subsequent characters being letters, numbers, or underscore. There are two categories of variables in TSKL: numeric data type and string data type.

TSKL supplies several numeric data types: long integer, float and double float.

Then maximum numbers of variable available in one program is listed as following:
- 200 long integer and float data type variables
- 100 double float data type variables
- 50 string data type variables

The range of data types in the system are listed as below:

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Identifier</th>
<th>Byte</th>
<th>Range</th>
<th>Significant digits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Long integer</td>
<td>%</td>
<td>4</td>
<td>-2147483648 to 2147483647</td>
<td>10</td>
</tr>
<tr>
<td>Float</td>
<td>N/A</td>
<td>4</td>
<td>-99999999 to 99999999</td>
<td>7</td>
</tr>
<tr>
<td>Double float</td>
<td>#</td>
<td>8</td>
<td>-9999999999999999999999 to</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>999999999999999999999999999999</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>$</td>
<td>256</td>
<td>254 characters</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Long Integer Data Type Variables

The “%” identifier is used to declare a long integer variable by placing the “%” at the end of variable name. For example, A% and SUM% are integer variables.

Float Data Type Variables

The default data type in TSKL is float data type. If no identifier is placed at the end of variable name. The variable will be treated as float data type in the system. For example, A and B are float variables. The precision of float data type is to 6 digits.
Double Float Data Type Variables
The “#” identifier is used to declare a double float variable by placing the “#” at the end of variable name. For example, A# and SUM# are floating point variables. The precision of double float data type is to 15 digits.

String Variables
The “$” identifier is used to declare a string variable by placing the “$” at the end of variable name. For example, A$ and TITLE$ are string variables. Each string variable can store 254 byte of data.

System Variables
System variables are the data maintained by smart keyboard. For example, Real Time Clock.
All the system variables are listed below:
YEAR
MONTH
DATE
HOUR
MINUTE
SECOND
IDNUMBER$
Functions
Functions are built-in procedures or subroutines used to evaluate, make calculations on, or transform data.

Functions used in TSKL can be grouped into numeric functions, string functions. For more information, please refer to TSKL Functions.

Numeric Functions
Numeric functions include integral functions and floating point functions. For example, INT(), ASC().

String Functions
String functions include string conversion, string manipulation. For example, RIGHT$( ), STR$( ).

Operators
The operators used in TSKL can be grouped into numeric operators, string operators and relational operators.

Arithmetic Operators
Arithmetic operators: ‘+’, ‘-’, ‘*’, ‘/’.

String Operators
String operators: ‘+’.

Relational Operators
Expressions

Operators, constants, and variables are the constituents of expressions. An expression in TSKL is any valid combination of these pieces. There are three kinds of expressions in TSKL.

Arithmetic Expressions

Arithmetic expressions can be integral expression or floating point expression, depends on the calculation value. In integral expression, floating point operands will be converted to integer, and vice versa.

String Expressions

There is only one operator in string expression, ‘+’, that is, add a string to another string.

Relational Expressions

The relational expressions are used to determine the relationship of one quantity to another. The result is true if the value is non-zero, otherwise, it is false.
Commands and Statements

Commands instruct smart keyboard to work accordingly. Sometimes, commands followed by one or several parameters, For example, INPUT A$. For more information refer to the command in this programming manual.

A statement is composed of one command or several commands, For example, IF…THEN…ELSE.
TSKL Commands

<ESC>!R

Description
This command is used to reset the keyboard. Keyboard will search for AUTO.BAS auto execution program first after reset.

Syntax
<ESC>!R

Remarks
<ESC> is ASCII 27 escape character.

Example
N/A
<ESC>!W

Description
This command is used to upgrade the firmware. Keyboard will enter BIOS upgrade mode after receive this command and disable to execution of AUTO.BAS program.

Syntax
<ESC>!W

Remarks
N/A

Example
N/A
SOUND

Description
Turn the speaker on at the specified frequency of interval.

Syntax
SOUND frequency, interval

where
Frequency: 0~15
Interval: 0~65535

Remarks
N/A

Example
DOWNLOAD “SOUND.BAS”
FOR F=0 TO 15
    FOR I=0 TO 65535
        SOUND F,I
        FOR J=1 TO 10
            NEXT J
    NEXT I
NEXT F
EOP
CLS

Description
Clears the LCD display.

Syntax
CLS

Remarks
CLS clears the LCD and places the cursor in the upper left corner. (at position 0,0)

Example
DOWNLOAD "CLS.BAS"
PRINT "THIS WORD WILL BE CLEARED"
  FOR I = 1 TO 5000
  NEXT I
CLS
  FOR I = 1 TO 2000
  NEXT I
PRINT "PLEASE PRESS ANY KEY "
K= INKEY()
EOP
CLEAR

Description
Remove all the declared variables, arrays, and opened files in the program from memory.

Syntax
CLEAR

Remarks
N/A

Example
CLEAR
CURSOR

Description
Select cursor appearance.

Syntax
CURSOR mode

Remarks
The available modes of cursor are listed below:
0: Hides the cursor.
1: Block with blinked cursor.
2: Fixed Underline cursor.
3: Block blinked and underline fixed cursor. (default)

Example
CURSOR 1
LOCATE

Description
Position cursor in LCD display.

Syntax
LOCATE x, y

Remarks
LOCATE moves the cursor to the given position in LCD display. If the coordinates are invalid, the command will be ignored.
The available value of x parameter is between 0 to 79.
The available value of y parameter is 0 and 1. Where 0 and 1 is the first line and the second line of LCD display respectively.

Example
DOWNLOAD "LOCATE.BAS"
PRINT "ENTER POSITION: (0<NUM<79)"
INPUT "",A$
B=VAL(A$)
LOCATE B,1
Cur_pos = POS()
PRINT "CURSOR POSITION "+STR$(Cur_pos)
END
PRINT...[USING]

Description
PRINT command is used to output the numeric and strings to LCD display. TSKL also supports formatted output to LCD display by PRINT USING command. The available maximum numbers of specified format for PRINT USING command is 15.

Syntax
PRINT list of expression[:]

Specify the numeric data output format:
PRINT USING "###.##",A

Specify the string data output format:
PRINT USING "\   ",A$

Remarks
A blank line is displayed on LCD if there is no expressions after the PRINT command. The PRINT USING command sends 0D 0A (carriage return and line feed) at the end of the expression if no ";" (semicolon) is at the end of the expression. The "," (comma) is used to separate the values by 8 spaces. There are no spaces between the two values if the ";" (semicolon) is used between the two expressions. If ";" (semicolon) is the last character of PRINT statement, the cursor will stop in this line. The "#" (pound sign) and "\" (back slash) is used to specify the output format of numeric data and string data respectively. If the actual value is greater that the specified format, a "%" (percent) sign will be paced at the end of value.

Example
DOWNLOAD "P_USING.BAS"
B=123.45
PRINT 12+3;ASC("A")
K=INKEY()
PRINT USING "####.##",B
END
EOP
INPUT

Description
Input numeric or string from keyboard keypad or COM2 port and assign them to variables.

Syntax
INPUT prompt; variables

Remarks
Prompt must be a string constant. Variable list contains the variable names to be assigned.
The input value will pass to the variable if ENTER key is pressed. The other input devices such as bar code scanner can be connected to the COM2 port of keyboard to get data instead of key in data by keyboard.

Example
DOWNLOAD "INPUT.BAS"
   PRINT "ENTER YOUR NAME:"
   INPUT "",ITEM$
   PRINT "YOUR NAME IS  "+ITEM$
   K=INKEY()
EOP
OUT

Description
Output data stream from keyboard serial port.

Syntax
OUT port; list of expressions[:]

Remarks
Port specifies the serial port to send data. It can be 0 (keyboard COM1) or 1 (keyboard COM2). List of expressions consists of string or numeric expression separated by semicolons. The OUT command sends 0D0A (carriage return and line feed) at the end of the expression except that the semicolon is used. ("," (comma) also sends 0D0A to serial port)

Example
DOWNLOAD "OUT.BAS"
C$=CHR$(34)
   PRINT "ENTER YOUR NAME:";
   INPUT "",A$
OUT 1;"YOUR NAME IS ";A$
OUT 0;"TEXT10,10,"+ C$+"4"+ C$+",0,1,1,"+ C$+"YOUR NAME IS "+A$+ C$
OUT 0;"PRINT 1,1"
   PRINT "PRESS ANYKEY TO LEAVE"
K=INKEY()
EOP
FOUT

Description
Output specified file from keyboard serial port.

Syntax
FOUT Port, FileHandle, FileSize

Remarks
Port specifies the serial port to send data. It can be 0 (keyboard COM1) or 1 (keyboard COM2). The number of FileHandle is between 0-14. FileSize is expressed in bytes.

Example
DOWNLOAD "DATA1.DAT",137,
   SIZE 4,2.5
   GAP 0,0
   SPEED 3
   DENSITY 10
   CLS
   TEXT 10,10,"4",0,1,1,"FOUT TEST"
   BARCODE 10,50,"39",100,1,0,2,5,"123456"
   PRINT 1,1

DOWNLOAD "FOUT.BAS"
   OPEN "DATA1.DAT" FOR INPUT AS #1
   FOUT 1,1,137
   SEEK #1,0
   FOUT 0,1,137
   CLOSE #1

EOP
OUT USING

Description
Output formatted data stream from keyboard serial port.

Syntax
OUT port USING "format",list of expressions[;]

Remarks
Port specifies keyboard serial port to send data. It can be 0 (keyboard COM1) or (keyboard COM2) port.
List of expressions consists of string or numeric expression separated by semicolons. The OUT USING command sends 0D0A (carriage return and line feed) at the end of the expression except that the semicolon (";") is used. Comma (",") also sends 0Dh 0Ah to serial port

Example
DOWNLOAD "O_USING.BAS"
A=123.45
B$="TSC-PRINTER"
OUT 1 USING "#####.##",A
OUT 1 USING "\",B$
EOP
DOWNLOAD

Description
The “DOWNLOAD” keyword is the identifier to save files into keyboard memory. There are two types of file that can be downloaded into keyboard memory: program file and data file.

Syntax
Program file:
DOWNLOAD “FILENAME.BAS”
   File contents…
   EOP

Data file:
DOWNLOAD “FILENAME”, FILESIZE,<DATA FILE>

Remarks
The maximum character of filename is up to 8 characters and extension is up to 3 characters.
The extension of program file must be BAS.
Data file can be any format of file. 0D 0A is used as separator of each data for text data file.
The FILESIZE parameter is calculated by bytes.

Example
Program file:
DOWNLOAD “DEMO.BAS”
   CLS
   PRINT “This is a test”
   EOP

Data file:
DOWNLOAD “DEMO.DAT”,10,0123456789
EOP

Description
End of program. The keyword must be placed at the end of program file is the
DOWNLOAD keyword is used in the program.

Syntax
EOP

Remarks
N/A

Example
DOWNLOAD “EOP.BAS”
CLS
PRINT “This is a test”
EOP
OPEN

Description
To establish file handles for file access. 15 files can be accessed in one program.

Syntax
OPEN "filename" FOR mode AS #FileHandle.

Remarks
Filename is the name of the file.
Mode parameter specifies the file operation mode. It can be:
   INPUT: Position to the beginning of the file and this file is “read only”. If the file
does not exist, The “File not found” error is displayed on LCD display.
   OUTPUT: Position to the beginning of the file, and this file is “write only”. If the
   file does not exist, a new file is created.
   APPEND: Append characters to the end of file.
FileHandle is a constant number or expression result, between 0 and 14.
Remember to close the file handle when the file is no longer used.

Example
DOWNLOAD "DATA1.DAT",135,
   SIZE 4,2.5
   GAP 0,0
   SPEED 3
   DENSITY 10
   CLS
   TEXT 10,10,"4",0,1,1,,OPEN TEST"
   BARCODE 10,50,"39",100,1,0,2,5,"123456"
   PRINT 1,1
DOWNLOAD "OPEN.BAS"
   OPEN "DATA1.DAT" FOR INPUT AS #1
   FOUT 1,1,135
   SEEK #1,0
   FOUT 0,1,135
   CLOSE #1
   EOP
CLOSE

Description
To clear file handles.

Syntax
CLOSE #FileHandle

Remarks
FileHandle must be constant number which the file is opened.

Example
DOWNLOAD "DATA1.DAT",136,
  SIZE 4,2.5
  GAP 0,0
  SPEED 3
  DENSITY 10
  CLS
  TEXT 10,10,"4",0,1,1,"CLOSE TEST"
  BARCODE 10,50,"39",100,1,0,2,5,"123456"
  PRINT 1,1

DOWNLOAD "CLOSE.BAS"
  OPEN "DATA1.DAT" FOR INPUT AS #1
  FOUT 1,1,136
  SEEK #1,0
  FOUT 0,1,136
  CLOSE #1
  EOP
KILL

Description
To delete file(s) in keyboard memory.

Syntax
KILL "filename"
KILL "*.*"
KILL "*"

Remarks
Filename can be any file in the memory.
Wild card (*) supports all files only.
The file must be closed before deleting.

Example
KILL "DEMO.BAS"
KILL "*.*"
KILL "*"
KILL "*"
SEEK

Description
Reposition a file pointer in specified file buffer.

Syntax
SEEK #FileHandle, offset.

Remarks
Offset is the number from the beginning of file to the new position.

Example
DOWNLOAD "DATA1.DAT",135,
   SIZE 4,2.5
   GAP 0,0
   SPEED 3
   DENSITY 10
   CLS
   TEXT 10,10,"4",0,1,1,"SEEK TEST"
   BARCODE 10,50,"39",100,1,0,2,5,"123456"
   PRINT 1,1

DOWNLOAD "SEEK.BAS"
   OPEN "DATA1.DAT" FOR INPUT AS #1
   SEEK #1,14
   FOUT 0,1,135
   FOUT 1,1,135
   CLOSE #1
   EOP

(Notice)
1. Printer or PC will receive the below commands, because it begins from
   15th byte to the end of file.
2. The 14 bytes of SEEK means SIZE 4,2.5 10 bytes and ODh OAh x2

GAP 0,0
SPEED 3
DENSITY 10
CLS
TEXT 10,10,"4",0,1,1,"SEEK TEST"
BARCODE 10,50,"39",100,1,0,2,5,"123456"
PRINT 1,1
READ

Description
Read data from data file and assign them to variables.

Syntax
READ #FileHandle; list of variables.

Remarks
FileHandle specifies the file to read data from.
The variables store the data read from the FileHandle.

Example
DOWNLOAD "READ.BAS"
OPEN "DATA" FOR INPUT AS #1
READ #1;A$;A
CLOSE #1
PRINT A$
K=INKEY()
PRINT A
END
WRITE

Description
Write data to an opened file.

Syntax
WRITE #FileHandle; list of expressions.

Remarks
FileHandle specifies the file to write data to.
The variables are used to write data to opened data file.

Example
DOWNLOAD "WRITE.BAS"
OPEN "DATA" FOR OUTPUT AS #1
WRITE #1;"THIS IS (WRITE) TEST"
WRITE #1;"12345678"
CLOSE #1
PRINT "WRITE OK!"
END
GOSUB…RETURN

Description
To branch to, return from a subroutine.

Syntax
GOSUB label
Statement block1
Label:
Statement block2
RETURN

Remarks
Label is a tag to mark a specified position in the program. The available maximum label name is 20 characters. A return statement will cause the program return to the statement following the GOSUB statement. The total numbers of GOSUB…RETURN statement can’t exceed than 40 in one program.

Example
DOWNLOAD "GOSUB.BAS"
   PRINT "MAIN ROUTINE (START)"
       K=INKEY()
       GOSUB SUB1
   PRINT "MAIN ROUTINE (END)"
END
SUB1:
   PRINT "SUBROUTINE"
   K=INKEY()
   RETURN
EOP

The execution result should look like as following.
OK
MAIN ROUTINE
SUBROUTINE
MAIN ROUTINE
Loop Statements

Loop statements allow a program to execute one or more lines of code repetitively. The loop statements that TSKL supports include:

- **DO…LOOP**
- **FOR…NEXT**
- **WHILE…WEND**
DO...[EXITDO]...LOOP

Description
Use a DO loop to execute a block of statements by an indefinite number of times. There are several variations of DO...LOOP statement, but each evaluates a numeric condition to determine whether to continue execution.

Syntax
DO {WHILE | LOOP}
    Statements
    {EXITDO}
LOOP

Or

DO
    Statements
    {EXITDO}
LOOP {WHILE | LOOP}

Or
DO
    Statements
    {EXITDO}
LOOP

Remarks
The total numbers of nested DO...LOOP statement in one program can not exceed than 40 levels.
The maximum numbers of nested IF...THEN...ELSE, FOR...NEXT, WHILE...WEND, and DO...LOOP statements in one program is up to 40 levels.

Example
DOWNLOAD "DOLOOP.BAS"
A=0
    DO WHILE A<10
A=A+1
PRINT A
K=INKEY()
LOOP
PRINT "EXIT"
END
EOP
FOR...[EXITFOR]...NEXT

Description
To execute a series of instructions by a specified number of times in a loop.

Syntax
FOR variable= I TO J [STEP K]
    statements
NEXT variable

Remarks
I, J, K are numeric expressions
I: The initial value of the counter
J: The final value of the counter
K: The increment of the counter. If K parameter is ignored, the default increment is 1.
The maximum numbers of IF...THEN...ELSE, FOR...NEXT, WHILE...WEND, and
DO...LOOP statements available in one program is up to 40.

Example
The following sample program prints out the sum of numbers between 1 and 10:
DOWNLOAD "FOR.BAS"
SUM=0
FOR I=1 TO 10
    SUM=SUM+I
    PRINT SUM
    K=INKEY()
NEXT I
END
EOP
WHILE...WEND

Description
To execute a series of statements in a loop until the given condition is false.

Syntax
WHILE expression
  statements
WEND

Remarks
If expression is true the program will be executed until the WEND statement is encountered, then return to the WHILE statement to check again, as encountering a false condition, program will branch to the statement following the WEND.

The total numbers of nested WHILE…WEND statement in one program can not exceed than 40 levels.
The maximum numbers of nested IF…THEN…ELSE, FOR…NEXT, WHILE…WEND, and DO…LOOP in one program is up to 40 levels.

Example
DOWNLOAD "WHILE.BAS"
A=1
WHILE A<10
  PRINT "A= "+STR$(A)
  A=A+1
  K=INKEY()
WEND
EOP
GOTO

Description
Branches from program to a specified block of statements.

Syntax
GOTO label

Remarks
Label is a tag to mark a specified position in the program.
The maximum characters of the label name are limited to 20 characters.
The total numbers of GOTO statement in a program can not exceed than 200.

Example
DOWNLOAD "IF.BAS"
BEGIN:
    PRINT "LOOK HERE"
    K=INKEY()
    CLS
    PRINT "REPEAT?(y/n)"
    v_CHAR = INKEY()
    IF v_CHAR=89 THEN
        CLS
        GOTO BEGIN
    ELSE IF v_CHAR=121 THEN
        CLS
        GOTO BEGIN
    ENDIF
    PRINT "Press key to quit!"
    SOUND 5,10000
END
EOP
Program-Control Statements

The program-control statements are the essence of any computer language because they govern the flow of program execution. Program-control statements may be separated into two categories:

- IF...THEN
- IF...THEN...ELSE
IF...THEN...ELSE

Description
Use an IF...THEN block to execute one or more statements conditionally. You can use either a single-line syntax or multiple-line “block” syntax:

Syntax
IF condition THEN statement

Notice that the single-line form of IF...THEN does not use an ENDIF statement.

Or

IF condition THEN
    Statements
ENDIF

Or

IF condition THEN
    Statements
ELSE
    Statements
ENDIF

Or

IF condition1 THEN
    Statement block 1
ELSEIF condition2 THEN
    Statement block 2
ELSE
    Statement block n
ENDIF
Remarks
If the result of the expression is nonzero, the statement following THEN will be executed. If the result of the expression is zero, and the statement following the ELSE present, it will be executed. Otherwise the next line of statement is executed.
If there are block of statements in IF…THEN …ELSE, ENDIF must be used at the end of the IF…THEN…ELSE statement.

Limitations:
The total numbers of nested IF…THEN…ELSE statement in a program can not exceed than 40 levels.
The total numbers of IF…THEN…ELSE, FOR…NEXT, WHILE…WEND, and DO…LOOP in a program can not exceed than 40 levels.

Example
DOWNLOAD "IF.BAS"
BEGIN:
  PRINT "LOOK HERE"
  K=INKEY()
  CLS
  PRINT "REPEAT?(y/n)"
  v_CHAR = INKEY()
  IF v_CHAR=89 THEN
    CLS
    GOTO BEGIN
  ELSE IF v_CHAR=121 THEN
    CLS
    GOTO BEGIN
  ENDIF
  PRINT "Press key to quit!"
  SOUND 5,10000
  END
EOP
Description
Terminates the program execution.

Syntax
END

Remarks
END statement may be placed anywhere in a program to terminate the execution. With END statement, all variables will be released from memory and closed file handles.

Example
DOWNLOAD "IF.BAS"
BEGIN:
   PRINT "LOOK HERE"
   K=INKEY()
   CLS
   PRINT "REPEAT?(y/n)"
   v_CHAR = INKEY()
   IF v_CHAR=89 THEN
      CLS
      GOTO BEGIN
   ELSE IF v_CHAR=121 THEN
      CLS
      GOTO BEGIN
   ENDIF
   PRINT "Press key to quit!"
   SOUND 5,10000
END
EOP
DIM

Description
An array is a collection of variables of the same type that is referenced by a common name. DIM statement is used to declare the array variables of integer, float and double float data types. The lowest address corresponds to the first element, and the highest address to the last element. A specific element in an array is accessed by an index.

Syntax
DIM variable (subscripts)[, variable (subscripts), …]

Remarks
The base of an array index always begins from 0. For example, DIM A(10), there are totally 11 elements of variable A (0 to 10).
Do not duplicate declare array in the program without executing the FREE statement.
The DIM statement sets all the elements of specified arrays to an initial value of numbers to zero and strings to null string.
The total numbers of array elements (no matter what kind of data type) cannot exceed than 200 elements.
The available maximum array variable name is 10 characters.
The maximum dimensions of an array variables is 3 dimension.

Example
DOWNLOAD "DIM.BAS"
DIM A(8),B(8)
FOR I=0 TO 8
A(I)=I+1
FOR J=0 TO 8
B(J)=J+1
OUT 1; A(I);"**";B(J);"=";A(I)*B(J)
NEXT J
NEXT I
PRINT “PRESS ANY KEY!”
END
FREE A
FREE B
EOP
FREE

Description
Eliminates array variables from memory.

Syntax
FREE dimension variable

Remarks
Arrays can be re-dimensioned after they are freed, or the memory space previously allocated to the array may be used for other purposes. If an attempt is made to re-dimension an array without first freeing it, an error occurs.

Example
DOWNLOAD "DIM.BAS"
DIM A(8),B(8)
FOR I=0 TO 8
A(I)=I+1
FOR J=0 TO 8
B(J)=J+1
OUT 1; A(I);"*";B(J);"=";A(I)*B(J)
NEXT J
NEXT I
END
FREE A
FREE B
EOP
REM

Description
 Inserts explanatory remarks in a program.

Syntax
 REM comments

Remarks
 REM statements are not executed.

Example
 REM *** this is an example ***
 DOWNLOAD “REMB.BAS”
 PRINT “YOU WILL NOT SEE REM”
 REM *** REM ***
 END
 EOP
TSKL Functions

POS()

Description
Gets the current position of the cursor on LCD display.

Syntax
POS()

Remarks
This function returns the value between 0~79.

Example
DOWNLOAD "POS.BAS"
   PRINT "ENTER POSITION: (0<NUM<79)"
   INPUT "", A$
   B = VAL(A$)
   LOCATE B, 1
   Cur_pos = POS()
   PRINT "CURSOR POSITION " + STR$(Cur_pos)
END
EOP
INKEY()

Description
To return the ASCII code of the character read from keyboard. The returned ASCII code will not shown on LCD display.

<table>
<thead>
<tr>
<th>ASCII</th>
<th>ASCII</th>
<th>ASCII</th>
<th>ASCII</th>
<th>ASCII</th>
<th>ASCII</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>65</td>
<td>a</td>
<td>97</td>
<td>1</td>
<td>49</td>
</tr>
<tr>
<td>B</td>
<td>66</td>
<td>b</td>
<td>98</td>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>C</td>
<td>67</td>
<td>c</td>
<td>99</td>
<td>3</td>
<td>51</td>
</tr>
<tr>
<td>D</td>
<td>68</td>
<td>d</td>
<td>100</td>
<td>4</td>
<td>52</td>
</tr>
<tr>
<td>E</td>
<td>69</td>
<td>e</td>
<td>101</td>
<td>5</td>
<td>53</td>
</tr>
<tr>
<td>F</td>
<td>70</td>
<td>f</td>
<td>102</td>
<td>6</td>
<td>54</td>
</tr>
<tr>
<td>G</td>
<td>71</td>
<td>g</td>
<td>103</td>
<td>7</td>
<td>55</td>
</tr>
<tr>
<td>H</td>
<td>72</td>
<td>h</td>
<td>104</td>
<td>8</td>
<td>56</td>
</tr>
<tr>
<td>I</td>
<td>73</td>
<td>l</td>
<td>105</td>
<td>9</td>
<td>57</td>
</tr>
<tr>
<td>J</td>
<td>74</td>
<td>j</td>
<td>106</td>
<td>0</td>
<td>48</td>
</tr>
<tr>
<td>K</td>
<td>75</td>
<td>k</td>
<td>107</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>76</td>
<td>l</td>
<td>108</td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>77</td>
<td>m</td>
<td>109</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>78</td>
<td>n</td>
<td>110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td>79</td>
<td>o</td>
<td>111</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>80</td>
<td>p</td>
<td>112</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q</td>
<td>81</td>
<td>q</td>
<td>113</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>82</td>
<td>r</td>
<td>114</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S</td>
<td>83</td>
<td>s</td>
<td>115</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>84</td>
<td>t</td>
<td>116</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U</td>
<td>85</td>
<td>u</td>
<td>117</td>
<td></td>
<td></td>
</tr>
<tr>
<td>V</td>
<td>86</td>
<td>v</td>
<td>118</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>87</td>
<td>w</td>
<td>119</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>88</td>
<td>x</td>
<td>120</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td>89</td>
<td>y</td>
<td>121</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Z</td>
<td>90</td>
<td>z</td>
<td>122</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Syntax
INKEY( )
Remarks
All tasks are pending until a key is pressed.

Example

DOWNLOAD "IF.BAS"
BEGIN:
    PRINT "LOOK HERE"
    K=INKEY()
    CLS
    PRINT "REPEAT?(y/n)"
v_CHAR = INKEY()
IF v_CHAR=89 THEN
    CLS
    GOTO BEGIN
ELSE IF v_CHAR=121 THEN
    CLS
    GOTO BEGIN
ENDIF
PRINT "Press key to quit!"
    SOUND 5,10000
END
EOP
INP$()

Description
To return one byte that received from serial port.

Syntax
INP$(expression)

Remarks
The result of the expression must be numeric. (0 or 1)
0: COM1 (Serial port with cable on keyboard)
1: COM2 (Serial port mount on keyboard)
The returned value is a string

Example
DOWNLOAD "INP.BAS"
REM *****Total Milage*****
    PRINT "******Distance******"
    Milage1$=""
    OUT 0;"~!@"
    Milage2$=""
    DO
        Milage2$=INP$(0)
        IF ASC(Milage2$)=13 THEN EXITDO
        Milage1$=Milage1$+Milage2$
    LOOP
PRINT Milage1$
END
EOP
EOF()

Description
To return nonzero when the end of a file has been reached, or to return 0 if the end of file (EOF) has not been found.

Syntax
EOF(file number)

Remarks
If file pointer points to the end of the file, EOF returns non-zero.

Example
DOWNLOAD "DATA",10,1234567890
DOWNLOAD "EOF.BAS"
OPEN "DATA" FOR INPUT AS #1
C=0
Repeat:
IF EOF(1)>0 THEN GOTO End_of_file
READ #1;A
C=C+1
GOTO Repeat
End_of_file:
PRINT "PRESS ANY KEY!"
END
EOP
LOF()

Description
   To return the size of file.

Syntax
   LOF(FileHandle)

Remarks
   An integer is returned to indicate the size of file

Example
   DOWNLOAD "LOF.BAS"
   OPEN "DATA" FOR APPEND AS #1
   WRITE #1;"ABC"
   File_length = LOF(1)
   PRINT "File_length= "+STR$(File_length)
   CLOSE #1
   END
   EOP
ABS()

Description
To return the absolute value of the expression.

Syntax
ABS(expression)

Remarks
The result of the expression must be numeric.

Example
DOWNLOAD "ABS.BAS"
  X=ABS(5-12)
  PRINT "Absolute value: "+X
  K=INKEY()
  X=5-12
  PRINT "Value: "+X
END
EOP
ASC()

Description
To return the value of the ASCII code for the first character of the expression.

Syntax
ASC(expression)

Remarks
The result of the expression must be a string.

Example
DOWNLOAD "ASC.BAS"
PRINT "*** INPUT YOUR WORD ***"
INPUT "Only one world: ",A$
X=ASC(A$)
PRINT "ASCII code value "+STR$(X)+" = "+A$
END
EOP
LEN()

Description
To return the length of the string.

Syntax
LEN(string expression)

Remarks
The maximum returned string length is 255.

Example
DOWNLOAD "LEN.BAS"
BEGIN:
    PRINT "HOW MANY DIGITS?"
    INPUT "ABC-DEF ",A$
    X=LEN("ABC-DEF")
    IF A$=STR$(X) THEN
        PRINT "CORRECT! ANSWER IS "+X
    ELSE
        PRINT "WRONG! ANSWER IS "+STR$(X)
    ENDIF
END
EOP
FRE()

Description
Return the size of free memory.

Syntax
FRE( )

Remarks
It calculates and returns an integer to indicate free memory size of keyboard in K Bytes.

Example
DOWNLOAD "FRE.BAS"
A=FRE( )
PRINT "REMAIN SPACE: "+STR$(A)+" K bytes"
END
EOP
INT()

Description
To truncate an expression to an integral number.

Syntax
INT(expression)

Remarks
The result of the expression must be numeric.

Example
DOWNLOAD "INT.BAS"
    PRINT "INTEGER OF 5/2 = ?"
    K=INKEY()
    A=INT(5/2)
    PRINT "ANSWER IS: "+STR$(A)
END
EOP
VAL()

**Description**
To return the numerical value of the string expression.

**Syntax**
VAL(expression)

**Remarks**
The result of the expression must be a string.

**Example**
DOWNLOAD "VAL.BAS"
   PRINT "A=123, B=321, A+B=?"
   K=INKEY()
      A=VAL("123")
      B=VAL("321")
      C=A+B
      PRINT "A+B= "+STR$(C)
END
EOP
RND()

**Description**
RND returns a number between 0 and 1

**Syntax**
A=RND()

**Remarks**
N/A

**Example**
DOWNLOAD "RND.BAS"
A=RND()
PRINT STR$(A)
K=INKEY()
EOP
CHR$()

Description
To return the ASCII character of the numerical expression.

Syntax
CHR$(expression)

Remarks
The result of the expression must be numeric.

Example
DOWNLOAD "CHR.BAS"
PRINT "*** INPUT YOUR VALUE ***"
   INPUT " ",A
   X$=CHR$(A)
   PRINT "ASCII character "+X$+" = "+STR$(A)
END
EOP
FREAD$(())

Description
   To read a number of bytes from specified file.

Syntax
   FREAD$(FileHandle, count)

Remarks
   Count is the number of data bytes. Maximum count size is 255.

Example
   DOWNLOAD "DATA",10,1234567890
   DOWNLOAD "FREAD.BAS"
   OPEN "DATA" FOR INPUT AS #1
   SEEK #1,5
   A$=FREAD$(1,5)
   CLOSE #1
   PRINT A$
   END
   EOP
LEFT$(())

Description
To return a number of the leftmost characters of the string expression.

Syntax
LEFT$(expression, count)

Remarks
The result of the expression must be a string.
Count is the number of the leftmost characters.

Example
DOWNLOAD "LEFT.BAS"
A$=CHR$(34)
PRINT "LEFT("+A$+"123456"+A$+",3) =?"
B$=LEFT$("123456",3)
K=INKEY()
PRINT B$
K=INKEY()
EOP
RIGHT$(())

Description
To return a number of the rightmost characters of the string expression.

Syntax
RIGHT$(expression, count)

Remarks
The result of the expression must be a string.
Count is the number of the rightmost characters.

Example
DOWNLOAD "RIGHT.BAS"
A$=CHR$(34)
PRINT "RIGHT$("+A$+"123456"+A$+",3) =?"
B$=RIGHT$("123456",3)
K=INKEY()
PRINT B$
K=INKEY()
EOP
MID$(())

Description
To return a number of characters begin from the starting parameter to the specified length.

Syntax
MID$(String, Starting, Length)

Remarks
String: Can be a string or expression
Start: Character position in string at which the part to be taken begins. If start is greater than the number of characters in string, Mid returns a zero-length string ("").
Length: The numbers of characters to be returned.

Example
DOWNLOAD "MID.BAS"
A$=CHR$(34)
PRINT "MID$("+A$+"123456"+A$+"",3,3) =?"
B$=MID$("123456",3,3)
K=INKEY()
PRINT B$
K=INKEY()
EOP
STR$(())

Description
Returns a string representation of a number.

Syntax
STR$(expression)

Remarks
The result of the expression must be a numeric.

*Note: STR$(()) function can not be included in other functions.*

*Example: VAL(STR$(“123”))*

Example
DOWNLOAD "STRING.BAS"
   PRINT "1+2+3+4+5=?"
   K=INKEY()
   SUM=1+2+3+4+5
   SUM$=STR$(SUM)
   PRINT "SUM= "+SUM$
   K=INKEY()
EOP
SPC$()

Description
To insert a number of spaces in a string.

Syntax
SPC$(expression)

Remarks
The result of the expression must be numeric.

Example
DOWNLOAD "SPACE.BAS"
    PRINT "PRICE";SPC$(5);"123.5"
    K=INKEY()
    EOP
System Variables

YEAR

Description
Set and returns the year of Real Time Clock. The available numbers to set YEAR variable begins from 00 to 99. The returned data is a 4 digits number. Ex: 1998, 2001.

Syntax
YEAR=nn
Where
nn=00~99

A=YEAR

Remarks
The Real Time Clock is Year 2000 compatible.

Example
DOWNLOAD "YEAR.BAS"
YEAR$=STR$(YEAR)
PRINT "YEAR "+YEAR$
K=INKEY()
YEAR$=MID$(YEAR$,3,2)
PRINT "YEAR "+YEAR$
K=INKEY()
EOP
MONTH

Description
Set and returns the month of Real Time Clock. The available numbers begins from 01 to 12.

Syntax
MONTH=nn
Where
nn=01~12

A=MONTH

Remarks
N/A

Example
DOWNLOAD "MONTH.BAS"
MONTH$=STR$(MONTH)
PRINT "MONTH "+ MONTH$
K=INKEY()
EOP
DATE

Description
Set and returns the date of Real Time Clock. The available numbers begins from 00 to 99.

Syntax
DATE=nn
Where
nn=01~31

A=DATE

Remarks
The leap year timing is automatic

Example
DOWNLOAD "DATE.BAS"
DATE$=STR$(DATE)
PRINT "DATE "+ DATE$
K=INKEY()
EOP
HOUR

Description
Set and returns the hour of Real Time Clock. The available numbers begins from 00 to 23.

Syntax
HOUR=nn
Where
nn=00~23

A=HOUR

Remarks
N/A

Example
DOWNLOAD "HOUR.BAS"
HOUR$=STR$(HOUR)
PRINT "HOUR "+ HOUR$
K=INKEY()
EOP
MINUTE

Description
Set and returns the minute of Real Time Clock. The available numbers begins from 00 to 59.

Syntax
MINUTE=nn
  Where
    nn=00~59

    A=MINUTE

Remarks
N/A

Example
DOWNLOAD "MINUTE.BAS"
  MINUTE$=STR$(MINUTE)
  PRINT "MINUTE "+ MINUTE$
  K=INKEY()
SECOND

Description
Set and returns the second of Real Time Clock. The available numbers begins from 00 to 59.

Syntax
SECOND=nn
Where
nn=00~59

A=SECOND

Remarks
N/A

Example
DOWNLOAD " SECOND.BAS"
SECON =STR$( SECOND)
PRINT " SECOND "+ SECOND $
K=INKEY()
EOP
IDNUMBER$

Description
IDNUMBER$ is read only at run time. The ID number can be set in the Setup Utility. With password locked on, users can’t change the ID number.

Syntax
A$=IDNUMBER$

Remarks
N/A

Example
DOWNLOAD "ID.BAS"
BEGIN:
   PRINT "ENTER THE ID NUMBER:"
   INPUT "",A$

   IF A$=IDNUMBER$ THEN
      PRINT "HELLO ~ WELCOME!"
      K=INKEY()
      END
   ELSE
      PRINT "WRONG ID -- TO NEXT!"
      K=INKEY()
      GOTO AGAIN
   ENDIF

AGAIN:
   PRINT "INPUT AGAIN? (Y/N)"
   v_CHAR = INKEY()
   IF v_CHAR=89 THEN
      CLS
      GOTO BEGIN
   ELSE IF v_CHAR=121 THEN
      CLS
      GOTO BEGIN
   ENDIF
   PRINT "Press key to quit!"
   SOUND 5,10000
   END
END
EOP
Appendix A Limitations of Keyboard Control Language

- The total numbers of nested IF…THEN…ELSE, FOR…NEXT, WHILE…WEND and DO…LOOP in a program can’t exceed than 40.layers
- The maximum numbers of nested GOSUB…RETURN can’t exceed than 40 layers.
- The number of labels can’t exceed than 200.
- 200 long integer and float data type variables
- 100 double float data type variables
- 50 string data type variables
- The maximum number of characters for variable name is 10 characters.
- The maximum number of characters for label name is 20 characters.
- The available memory for download is 128 KB
- The numbers of downloaded file can’t exceed than 50 files for SRAM, 150 files for flash memory (V1.08 and later versions).
- 15 files can be opened simultaneously in one program.
- Maximum numbers of dimensions for array variable is 3 dimensions.
- The total numbers of specified format of PRINT USING and OUT USING statement can’t exceed that 15.
Appendix B Error Messages

(01) Syntax error
   The statement syntax is not correct.

(02) IF without THEN
   If statement without THEN keyword.

(03) IF without ENDSUB
   If statement without ENDSUB keyword.

(04) ELSEIF without IF
   IF…THEN…ELSE…ELSEIF statement without ELSEIF keyword.

(05) ELSE without IF
   IF…THEN…ELSE statement without IF keyword.

(06) ENDSUB without IF
   IF…THEN…ELSE statement without IF keyword.

(07) FOR without NEXT
   FOR…NEXT statement without NEXT keyword.

(08) NEXT without FOR
   FOR…NEXT statement without FOR keyword.

(09) EXITFOR without FOR
   FOR…NEXT statement without FOR keyword.

(10) WHILE without WEND
    WHILE…WEND statement without WEND keyword.

(11) WEND without WHILE
    WHILE…WEND statement without WHILE keyword.

(12) DO without LOOP
    DO…LOOP statement without LOOP keyword.

(13) LOOP without DO
    DO…LOOP statement without DO keyword.

(14) RETURN without GOSUB
    GOSUB…RETURN without GOSUB keyword

(15) Mode error
    File operation access error.

(16) File number error
Illegal file handle number.

(17) File not found
The access file name does not exit in memory.

(18) Too many labels
The total numbers of label in the program is exceed than 200.

(19) Duplicate label
Declared twice of label name.

(20) Label not found
The specified label does not exit in program.

(21) Illegal function call
Function does not exit.

(22) Illegal variable
The variable name exits illegal letters.

(23) Variable not defined
The variable does not assigned value in the program.

(24) Array not defined
The accessed array does not exit.

(25) Out of string space
The string is longer than 254 bytes.

(26) Too many variables
The number of variables is larger than the definition.

(27) Too many formats
The numbers of specified format of PRINT USING command and OUT USING command is larger than 15.

(28) Type mismatch
The data types of the two variables or returned values are not the same.

(29) Port error
The port specified in OUT and FOUT command is other than 0 and 1.

(30) Stack overflow
The total numbers of GOSUB...RETURN, WHILE...WEND, DO...LOOP, FOR...NEXT, IF...THEN...ELSE statement is larger than the specified number.

(31) Division by zero
The dividend is zero.

(32) Unknown operator
The operator is not defined in keyboard

(33) **Expression too complex**
Arithmetic expression is too complex.

(35) **Duplicate array**
Duplicate declare array variable.

(36) **Out of memory**
There is no enough memory for download program or data file.

(37) **Too many files**
The total number of files stored in memory exceeds that 50.

(38) **Unmatched brace**
One of the right or left parentheses is missing.
# Appendix C RS-232 Pin Configurations

Host and Printer RS-232 PIN Definition:

A. RS-232 Serial Interface 9 Pin D Type connector define:

<table>
<thead>
<tr>
<th>Host</th>
<th>Device</th>
<th>Printer</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM PC</td>
<td>Keyboard</td>
<td>Keyboard</td>
</tr>
<tr>
<td>COM1</td>
<td>COM1 (Note 1)</td>
<td>COM1 (Note 2)</td>
</tr>
<tr>
<td>Pin 1</td>
<td>/CD ←</td>
<td>+ 5V ←</td>
</tr>
<tr>
<td>Pin 2</td>
<td>RXD ←</td>
<td>RXD ←</td>
</tr>
<tr>
<td>Pin 3</td>
<td>TXD →</td>
<td>TXD →</td>
</tr>
<tr>
<td>Pin 4</td>
<td>/DTR →</td>
<td>/DTR →</td>
</tr>
<tr>
<td>Pin 5</td>
<td>GND</td>
<td>GND</td>
</tr>
<tr>
<td>Pin 6</td>
<td>/DSR ←</td>
<td>/DSR ←</td>
</tr>
<tr>
<td>Pin 7</td>
<td>/RTS →</td>
<td>/RTS →</td>
</tr>
<tr>
<td>Pin 8</td>
<td>/CTS ←</td>
<td>/CTS ←</td>
</tr>
<tr>
<td>Pin 9</td>
<td>RI←</td>
<td>+ 5V ←</td>
</tr>
</tbody>
</table>

Note 1: Smart Keyboard Hardware SW1 & SW2 must ON.
Note 2: SMART KEYBOARD Hardware SW1 & SW2 must OFF.
Note 3: ← symbol emblem INPUT, → symbol emblem OUTPUT.
B. RS-232 Serial Interface 25 Pin D Type connector define:

<table>
<thead>
<tr>
<th>Printer</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS-232C</td>
<td>IBM PC PC-8801 ProTool+</td>
</tr>
<tr>
<td>Zebra</td>
<td>(Note 1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Printer</th>
<th>Host</th>
<th>(Note 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Protective Ground</td>
<td>SHELL GND</td>
<td>Frame GND</td>
</tr>
<tr>
<td>2</td>
<td>Transmitted Data</td>
<td>TXD → TXD</td>
<td>RXD ← TXD</td>
</tr>
<tr>
<td>3</td>
<td>Received Data</td>
<td>RXD ← RXD</td>
<td>TXD → RXD</td>
</tr>
<tr>
<td>4</td>
<td>Request to Send</td>
<td>/RTS → RTS</td>
<td>/CTS ← /RTS</td>
</tr>
<tr>
<td>5</td>
<td>Clear to Send</td>
<td>/CTS ← /CTS</td>
<td>/RTS ← /CTS</td>
</tr>
<tr>
<td>6</td>
<td>Data Set Ready</td>
<td>/DSR ← /DSR</td>
<td>/DTR → /DSR</td>
</tr>
<tr>
<td>7</td>
<td>Signal Ground</td>
<td>GND GND GND</td>
<td>GND GND</td>
</tr>
<tr>
<td>8</td>
<td>Received Line Signal Detector</td>
<td>/CD ← DCD</td>
<td>NC NC</td>
</tr>
<tr>
<td>9</td>
<td>(Reserved for Data Set Testing)</td>
<td>NC NC + 5V 0.25A ← 5V 1A →</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>(Reserved for Data Set Testing)</td>
<td>NC NC + 5V 0.25A ← NC</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Unassigned</td>
<td>NC NC NC NC</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Secondary Received Line</td>
<td>NC NC NC NC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Signal Detector</td>
<td>NC NC NC NC</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Secondary Clear to Send</td>
<td>NC NC NC NC</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Secondary Transmitted Data</td>
<td>NC NC + 5V 0.25A ← NC</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Transmission Signal Element Timing</td>
<td>NC NC NC NC</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Secondary Received Data</td>
<td>NC NC NC NC</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Receiver Signal Element Timing</td>
<td>NC RXC ← NC</td>
<td>NC</td>
</tr>
<tr>
<td>18</td>
<td>Unassigned</td>
<td>NC NC NC NC</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Secondary Request to Send</td>
<td>NC NC NC NC</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Data Terminal Ready</td>
<td>/DTR → /DTR → /DSR ← /DTR →</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Signal Quality Detector</td>
<td>NC NC NC NC</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>Ring Indicator</td>
<td>RI ← NC NC NC</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>Data Signal Rate Selector</td>
<td>NC NC + 5V 0.25A ← NC</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>Transmitter Signal Element Timing</td>
<td>NC TXC → NC</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Unassigned</td>
<td>NC NC NC NC</td>
<td></td>
</tr>
</tbody>
</table>

(Continued on next page)
<table>
<thead>
<tr>
<th>No.</th>
<th>Description</th>
<th>DATAMAX</th>
<th>Ring</th>
<th>TEC</th>
<th>SATO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Protective Ground</td>
<td>CHASSIS</td>
<td>Frame GND</td>
<td>Frame GND</td>
<td>Frame GND</td>
</tr>
<tr>
<td>2</td>
<td>Transmitted Data</td>
<td>TXD →</td>
<td>TXD →</td>
<td>RXD ←</td>
<td>TXD →</td>
</tr>
<tr>
<td>3</td>
<td>Received Data</td>
<td>RXD ←</td>
<td>RXD ←</td>
<td>TXD →</td>
<td>RXD ←</td>
</tr>
<tr>
<td>4</td>
<td>Request to Send</td>
<td>/RTS →</td>
<td>/RTS →</td>
<td>/CTS ←</td>
<td>/RTS →</td>
</tr>
<tr>
<td>5</td>
<td>Clear to Send</td>
<td>/CTS ←</td>
<td>/CTS ←</td>
<td>/RTS ←</td>
<td>/CTS ←</td>
</tr>
<tr>
<td>6</td>
<td>Data Set Ready</td>
<td>NC ←</td>
<td>/DSR ←</td>
<td>/DTR →</td>
<td>/DSR ←</td>
</tr>
<tr>
<td>7</td>
<td>Signal Ground</td>
<td>GND</td>
<td>GND</td>
<td>GND</td>
<td>GND</td>
</tr>
<tr>
<td>8</td>
<td>Received Line Signal Detector</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>9</td>
<td>(Reserved for Data Set Testing)</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>10</td>
<td>(Reserved for Data Set Testing)</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>11</td>
<td>(Unassigned)</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>12</td>
<td>(secondary Received Line Signal Detector)</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>13</td>
<td>Secondary Clear to Send</td>
<td>NC ←</td>
<td>To Be Left Unused</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>14</td>
<td>Secondary Transmitted Data</td>
<td>+ 5V 0.1A →</td>
<td>To Be Left Unused</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>15</td>
<td>Transmission Signal Element Timing</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>16</td>
<td>Secondary Received Data</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>17</td>
<td>Receiver Signal Element Timing</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>18</td>
<td>Unassigned</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>19</td>
<td>Secondary Request to Send</td>
<td>NC ←</td>
<td>To Be Left Unused</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>20</td>
<td>Data Terminal Ready</td>
<td>BUSY →</td>
<td>/DTR →</td>
<td>/DSR ←</td>
<td>/DTR →</td>
</tr>
<tr>
<td>21</td>
<td>Signal Quality Detector</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>22</td>
<td>Ring Indicator</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>23</td>
<td>Data Signal Rate Selector</td>
<td>NC ←</td>
<td>+ 5V</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>24</td>
<td>Transmitter Signal Element Timing</td>
<td>NC ←</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
</tr>
<tr>
<td>25</td>
<td>Unassigned</td>
<td>NC ←</td>
<td>To Be Left Unused</td>
<td>NC</td>
<td>NC</td>
</tr>
</tbody>
</table>
Operation Guide

Auto Execution Utility

| Power on | Execute the “AUTO.BAS” program automatically |

Power-on Utility

<table>
<thead>
<tr>
<th>Power on with</th>
<th>Disable the auto execution utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALT F1</td>
<td>Upgrade keyboard BIOS</td>
</tr>
<tr>
<td>ALT SHIFT</td>
<td>Keyboard initialization</td>
</tr>
<tr>
<td>ALT CLEAR</td>
<td>Extend Files Manager utility</td>
</tr>
<tr>
<td>ALT D or ALT E</td>
<td></td>
</tr>
</tbody>
</table>

Keyboard Utilities

<table>
<thead>
<tr>
<th>FORM</th>
<th>Select BASIC files to execute</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTER</td>
<td>Execute the selected item</td>
</tr>
<tr>
<td>EXIT</td>
<td>Exit sub-menu</td>
</tr>
<tr>
<td>← →</td>
<td>Select parameters</td>
</tr>
<tr>
<td>↑ ↓</td>
<td>Scroll files or cursor</td>
</tr>
<tr>
<td>ALT F1</td>
<td>Keyboard configuration setup</td>
</tr>
<tr>
<td>ALT F2</td>
<td>On-line editing utility</td>
</tr>
<tr>
<td>ALT F2 F1</td>
<td>Upload files</td>
</tr>
<tr>
<td>ALT F2 CLEAR</td>
<td>Delete files</td>
</tr>
<tr>
<td>ALT D</td>
<td>Extend files manager</td>
</tr>
<tr>
<td>ALT EXIT</td>
<td>Reset keyboard</td>
</tr>
</tbody>
</table>