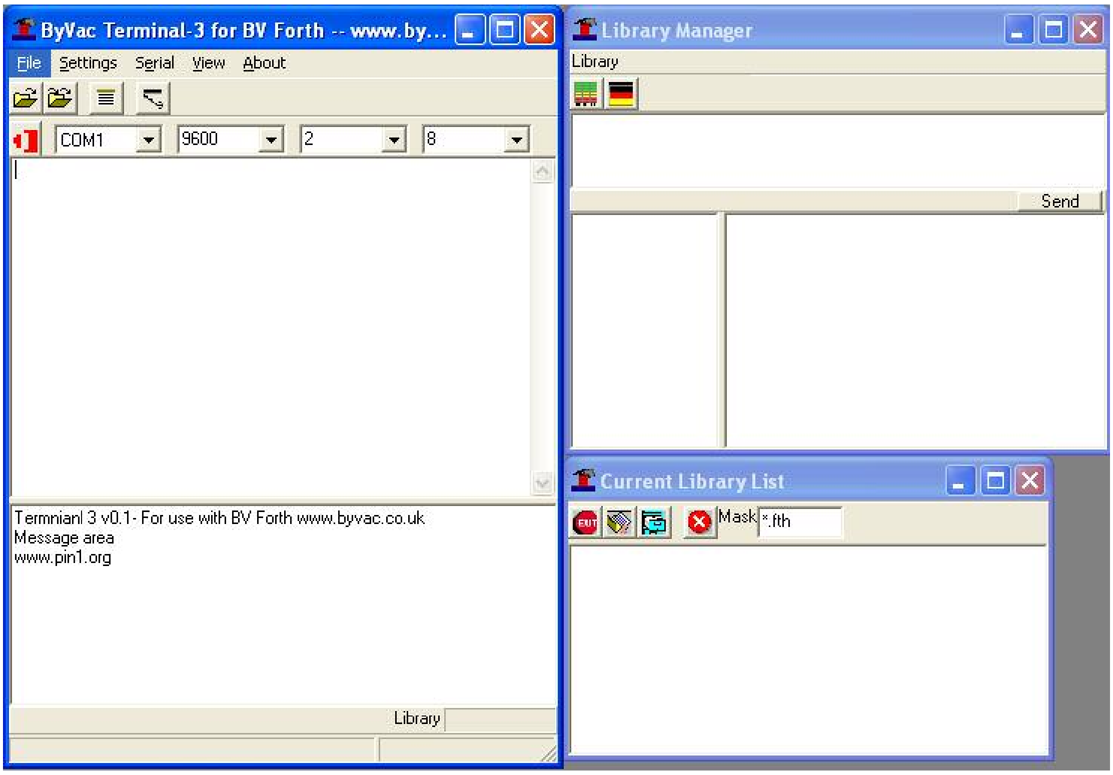
Contents



Termniani 3 v0.1- For use with BV Forth www.byvac.co.uk
Message area
www.pin1.org

ByVac Terminal-3

User Guide Oct 2007

1. Corrections to this section

Rev	Change
Oct 2007	Preliminary
Nov 2007	Version 3.17, SID now processed within file.
Feb 2008	Updated the way SID is handled
Feb 2008	New directives for internet, and library search order added, version 3.20

2. Introduction

The BV Terminal was originally designed as a simple communication device for the PC comm. Port, simpler than HyperTerminal and easier to use with all of the main port configurations available on the window.

Since then the ARM microcontroller has been introduced that uses Forth as its operating system. This version of BV Terminal (BVT) works with the BV Forth without making the Forth unusable in other terminals.

The software can be uses as a normal terminal for other uses, the biggest advantage is that it can be run from a CD-ROM. It is a single exe file that does not require installation and does not use the registry.

Most of this user guide is concerned with the file transfer protocol and features for working with Forth files. It may work with other Forth's, other than BV Forth but this has not been tried.

3. New Features

- Library loading and Maintenance
- Automatic loading of undefined words
- Syntax colouring
- Single word loading
- Message screen
- In console editing

4. Terminal

The BV Terminal 3 (BVT) will behave as a normal terminal emulator, all of the features are in addition to the main function which is a serial communications terminal.

5. File Transfer Protocol

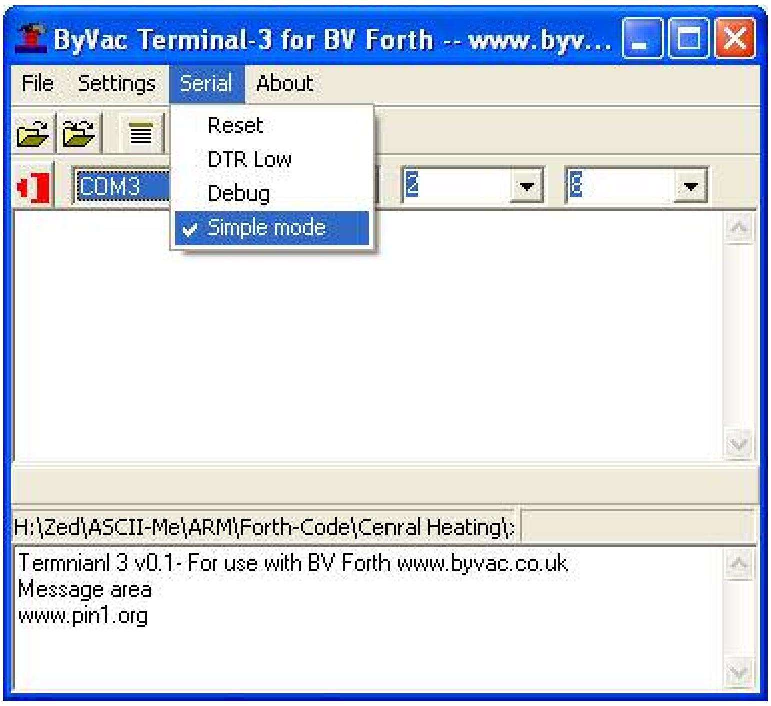
There are two available protocols a simple and a more complex but both have a very simple method of sending a file line by line with some simple handshaking. The simple protocol is compatible with BV Terminal 1 and 2. If the other windows are closed then this terminal should be used in preference to BVT-2 as some bugs have been fixed in this version.

5.1. Simple Transfer

To activate this make sure simple mode is selected as shown in Figure 1. In this mode there is no difference to previous versions of BV Terminal (BVT). When a file transfer is initiated BVT can optionally send a fixed string, this will normally be LOADF (can be changed using text settings) followed by a user selectable delay.

The purpose is to tell the target that some text is about to be sent. In the case of BV Forth, LOADF initiates the file loading protocol. After this is sent, BVT gets a line of text from the specified file and sends this to the target where upon it waits until a specific character is sent by the target. The character by default is ACSII 6 (ACK) but this can be changed, it can also be changed in BV Forth using the ACK word.

From the targets point of view, once it has received a line of text, it sets about processing it, it sends ACK only when it is ready to receive the next line. BVT will wait up to 600ms for the ACK character at which point it will time out.



H:\Zed\ASCII-Me\ARM\Forth-Code\Cenral Heating\:

```
Termnianl 3 v0.1- For use with BV Forth www.byvac.co.uk
Message area
www.pin1.org
```

Figure 1 Simple mode

This protocol has proved very reliable and easy to use. Most terminals support waiting for a character when a line is sent, so from the targets point of view this is very flexible.

5.2. Extended Protocol

This mode, with ‘Simple mode’ left unchecked is designed to work with BV Forth. The actual file transfer protocol is exactly the same as simple mode but with some additions.

The first addition is that when an error occurs in the target, such as trying to use an undefined word, instead of sending ACK, ASCII 6, it will send ASCII 5. BVT can make use of this information and search for the missing word. In addition to this it will handle ‘directives’ and output messages to indicate what is going on.

5.3. Output Window

At the bottom of the terminal is an output window to indicate what is being loaded. When a word is not found it is indicated that the library is being searched, if the word is found, the screen shows the library in which it was found.

In the serial menu there is a debug check item that will give a full output of what is being sent and received.

6. Library and Word Loading Features

When working with Forth files it is now possible to use directives, within the source code such as #include to load files prior to the main source. There is also a library directive but this can also be managed with the library manager.

6.1. SID (Scope ID)

Scope ID (SID) is a feature of BVF and the directives can optionally set this to be used as the default on an include or library. It can still be overridden if a SID is encountered within the file. The parameter (SID) is optional and if left out will be set to 0.

When used with an include, this will set the value of SID (nn SID !) before including the file. When used with a library, all of the words of that library will be compiled with that SID, except words beginning with <0> (e.g. <0>word). This is so that the library words can remain private to the library and words that are needed for other modules can be made public by preceding them with <0>.

When a word is defined with a leading <0>, the library manager removes this from the word but it will still be compiled using the SID of the library, so for example if a word has been defined as <0>p! … ; it will appear in the library manager as P! but will compile with the SID set as defined by SID=NN.

7. Directives

Directives Summary

Directive	Description
#Include "name" [sid=nnn]	Includes a named file when it is encountered and sets the SID as directed.
#Library "name" [sid=nnn]	Loads a library in the BVT memory for later use, the library words will be subsequently loaded using the specified SID.
#libdir "mask" [sid=nnn]	Loads a full directory into BVT ram using the mask given. If SID is specified all of the libraries loaded will be loaded with that SID number.
#reload	Clears all of the libraries from BVT and forces them to be loaded again.
#send "text" delay=nnn	Sends text to the terminal followed by a delay.
#URL-lib "text" [sid=nn]	As #Library but will load from an internet source.

#URL-inc "text" [sid=nn]	As #Include but will load from an internet source.
#CurrentSID sid=nn	By default the file loading has a SID of 0, this will change that.

A directive is a line that begins with // #, for example // #include. Using the comment characters will enable the code to be compatible with other terminal software. The correct format for a directive is:

//<space>#<directive><space/s>"<some text>"<sid=nnn>

It is important to observe the correct format as above. All words are translated to upper case so the directive and its contents are **not case sensitive**.

At the moment only full paths or current directory files can be specified, the use of ./ and ../ will not work.

7.1. #include

This works as in other software in that it will include a file specified at the point it is encountered, for example:

```
// Forth file example using include
//
: startUp
        6 6 +
;

// #include "c:\forth\library\tools.fth"
// #include "head.fth"
```

In the above example two files will be included as part of the code. The files can be specified with a path or if they are specified without a path, the current directory will be searched. The current directory is the one shown on the status bar of the terminal window.

Each time the above file is loaded both files will be included.

7.2. #URL-inc

This works in exactly the same way as #include but it allows a URL to be specified so that files can come from the internet, for example.

```
// #URL-inc "http://pin1.org/flb/i2c/i2c.flb"
```

The above will load i2c.flb from a directory at pin1.org. It is important that the exact file name is used, wild cards such as * cannot be used.

7.3. #Library

This directive probably makes the #include directive obsolete, but there may be circumstances where include is more appropriate so it has been left in. It is used in exactly the same way:

// #library "filename.fth"

// #library "filename.fth" sid=27

When the library directive is used the contents of the file are loaded into the BVT RAM, there can be any number of files specified up to the capacity of approximately 50,000 lines of code. The purpose of this is so that if a word is specified in the current file that can't be found, the library is searched and the word loaded. As an example:

```
// Main forth file that uses libraries
//
// #library "c:\forth\tools\soft1.fth"
// #library "c:\forth\tools\i2c-a.fth"
// #library "stepper.fth"
// #library "c:\forth\tools\pinsel.fth"
// #library "c:\forth\tools\AtoD.fth"
//
: outx@ (  -- v1 )
        3 p0@ 5 +
;
```

The above is a file called main.fth (it can be called anything). In this example a selection of files are kept in tools but the file stepper.fth is in the current directory and so the path does not need to be specified. The quotes " are important and must be there.

When this is selected and sent, all of the library files are loaded into BVT RAM, but not the target RAM. The actual output is shown below, I have tidied it up a bit and added line numbers to aid explanation.

```
1)  : outx@ (  -- v1 )
2)     3 p0@ 5 +
3) ? P0@

4): p0@
5)     1 swap lshift
6)     IO0PIN @
7) ? IO0PIN

8)  &e0028000 constant IO0PIN

9) : p0@
10)     1 swap lshift
11)     IO0PIN @
12)     and
13)     0= invert      // test for value
14) ;

15) : outx@ (  -- v1 )
16)     3 p0@ 5 +
17) ;

18) ;s
```

At 1) the word outx@ is sent to the target and this compiles okay as it is a new word to be defined. When P0@ is encountered 2), this has not been defined before so the target outputs and error. This is picked up by BVT which goes off to look in the library for the PO@ word. It finds this in pinsel.fth which has been loaded into ram previously by the library directives.

BVT then send the word PO@ to the target and the target begins to compile it, during compilation it comes across another word that has not been defined 6) IO0PIN, so BVT looks for that. This is in fact a constant which BVT sends to the target. It than sends 9) p0@ again which compiles correctly this time. Now this is compiled it can go back to the original word at 15) and compile it. (clever eh)

There are some points to note about this if you hadn't realised it already:

- Although 5 files were specified in the library statements only 3 words have been loaded into the target, PO@, IO0PIN and OUTX@
- This could have been done manually but it world be very difficult to get the correct words first time, it is very easy to miss nested dependencies.
- BV Terminal is able to carry on from the word that cause the error and it can nest this. It performs this using a stack which can be up to 50 deep.
- Words are only compiled into the target if they are needed, loading more library files does not effect the size of the target code.
- Only words are loaded into a library so in source code specified in a library that contains directives, the directives will be ignored. This means that all required library files must be specified in the current file. An #include directive can of course be used that contains only libraries.
- The libraries are indexed when loaded and searched backwards, this is in keeping with all Forth code in that the latest or later word will be used in preference to an earlier word.
- The library is maintained in ram until it is either cleared by using the clear library button or by using the #reload directive. The data and time is stored with each library file so if a library file is modified it is re-loaded automatically. The new version of the file is loaded after the old version so that any modified words in the newer version of the library will be picked up first and used (backward search). The old version is not removed from the library so eventually the RAM will become full if this is done time and again.
- SID: by default all library words will be loaded with a SID of 0, this will make them accessible to all words. This can be overridden by using the SID parameter.

Search Order

Added in version 3.20 is the order in which the libraries are searched for choosing the correct word.

The order is: 1) use a word from the same library file 2) use a word with the highest SID.

In the first instance if a word is being used from a library and has not been defined yet then its own file will be searched first. In the second instance if there are two words the same, the word form the library with the highest SID number will be used.

7.4. #URL-lib

This has exactly the same functionality as the #library function except that the files can come from an external source specified by a URL, for example

// #URL-Lib "http://pin1.org/flb/i2c/i2c.flb" sid=210

7.5. #LibDir [mask] [sid=nnn]

This will load all of the files in the current directory to the BVT ram. The mask by default is taken from the current file being loaded so for example if the current file is test.fth then the mask will be *.fth.

The mask can also be specified thus // #LibDir "*.lib"

Only one mask can be specified for each directive, but more than one directive can be used.

If a SID is specified then this will be used for all of the libraries loaded.

7.6. #Reload

This clears the current library and forces a reload of any libraries. This is exactly the same as using the library clear button.

If a library has been previously loaded and not updated (the time stamp is checked), it will be skipped. This enables a Forth file containing library directives to be loaded over and over again without filling the library space in the BVT RAM.

7.7. #Send "text" delay=nnn

Sends text to the terminal and follow this with a delay. This is used for commands that may take a relatively long time and may terminate the loading process, SAVE for example.

7.8. #CurrentSID

By default when a file is loaded the SID is set to 0, regardless of the state of SID within the BV Forth device. This directive is used to tell BVT that the current file should be loaded using the SID as specified in the directive.

The syntax is slightly different to the other directives as it does not have any text in surrounding quotes, for example

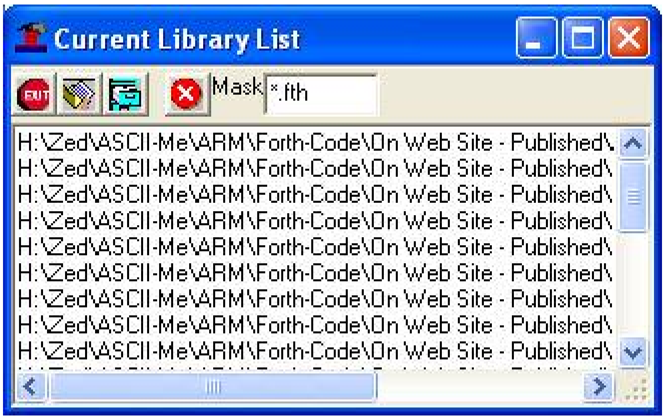
// #CurrentSID sid=11

The directive can be used anywhere in the file and from that point on the current file will be set to whatever is specified for SID.

8. Library Manager

All of the above can be accomplished with the library manager, this consists of two windows that can be individually opened and used. By default the windows open when BVT is first used.

8.1. Library List



```
*.fth
```

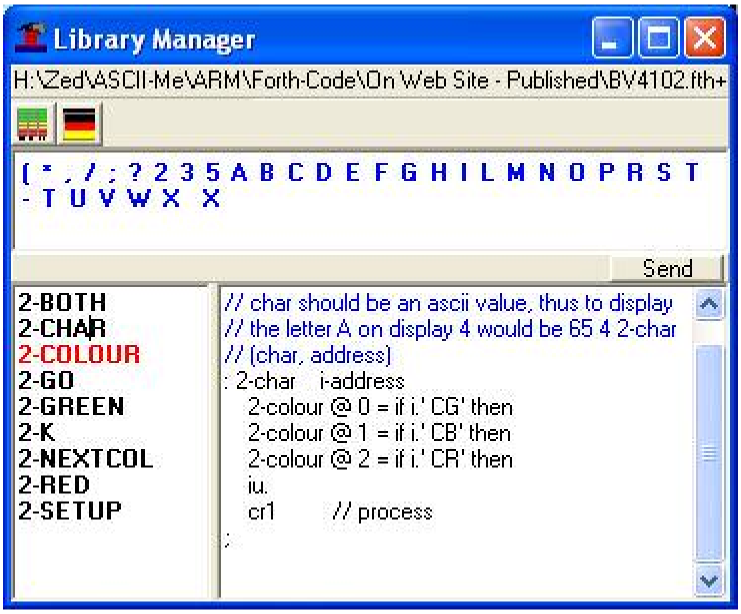
```
H:\Zed\ASCII-Me\ARM\Forth-Code\On Web Site - Published\
H:\Zed\ASCII-Me\ARM\Forth-Code\On Web Site - Published\
H:\Zed\ASCII-Me\ARM\Forth-Code\On Web Site - Published\
H:\Zed\ASCII-Me\ARM\Forth-Code\On Web Site - Published\
H:\Zed\ASCII-Me\ARM\Forth-Code\On Web Site - Published\
H:\Zed\ASCII-Me\ARM\Forth-Code\On Web Site - Published\
H:\Zed\ASCII-Me\ARM\Forth-Code\On Web Site - Published\
H:\Zed\ASCII-Me\ARM\Forth-Code\On Web Site - Published\
H:\Zed\ASCII-Me\ARM\Forth-Code\On Web Site - Published\
```

Figure 2 Library List Window

Libraries are loaded either the library list window or by using a directive within the Forth code. One button will load an individual file and the other button will load a whole directory. The figure shows that a whole directory has been loaded. The mask is used when loading whole directories.

8.2. Library manager

The library manager will list all of the current words that are loaded into the BVT RAM.



```
H:\Zed\ASCII-Me\ARM\Forth-Code\On Web Site - Published\BV4102.fth+
```

```
( * , / ; ? 2 3 5 A B C D E F G H I L M N O P R S T
- T U V W X  X
```

```
2-BOTH
2-CHAR
2-COLOUR
2-GO
2-GREEN
2-K
2-NEXTCOL
2-RED
2-SETUP
```

```
// char should be an ascii value, thus to display
// the letter A on display 4 would be 65 4 2-char
// (char, address)
: 2-char    i-address
   2-colour @ 0 = if i.' CG' then
   2-colour @ 1 = if i.' CB' then
   2-colour @ 2 = if i.' CR' then
   iu.
   cr1        // process
;
```

Figure 3 Library Manager

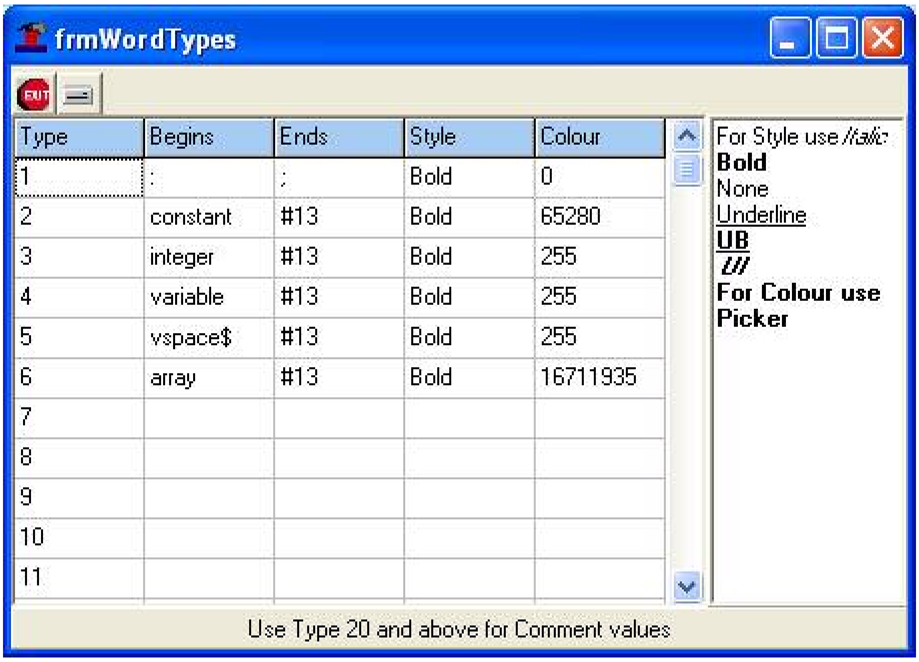
The second tool button from the left will re-index the library and present a list of characters in the top pane. The list is the first letter of a currently loaded word, this is the index into the loaded library. In the above example '2' has been selected and this presents all of the words that begin with 2 in the left hand pane. Colon definition words are coloured in black, constants are green and other words, variables etc. are in red. The example shown in the figure has had '2-char' selected by clicking on it. This will present that word in the right hand pane along with any of this type '\\' of comment that may be above the word.

Using this manager it is possible to see any word defined in the library. The actual library (Forth source code) where this word comes from is shown at the very top, above the tool bar. The SEND button will send the word displayed in the right hand pane to the target, providing all of the words within the word to be sent are in the library BVT will find them and present to the target.

The far left button is the syntax colouring which is dealt with in the next section.

9. Words and Comments

To make this work BVT needs to know something about Forth. As a minimum it needs to know the defining words and what constitutes a comment.



Type	Begins	Ends	Style	Colour
1	:	;	Bold	0
2	constant	#13	Bold	65280
3	integer	#13	Bold	255
4	variable	#13	Bold	255
5	vspace$	#13	Bold	255
6	array	#13	Bold	16711935
7				
8				
9				
10				
11				

For Style use *Italic:*
Bold
None
Underline
UB
///
For Colour use Picker

Use Type 20 and above for Comment values

Figure 4 Syntax colouring dialog

The above tables tells BVT what a defining word begins and ends with. BVT needs to know this so that it can rewind to the start of the word when an error is encountered by the target. The word types table is used to define the defining words. If a defining word finishes at the end of the line than #13 denotes this. A colour and text style can also be chosen for the particular type of word.

Under normal circumstances there should be no need to add to this table but has been included in case you decide to create your own defining words.

9.1. Comments

Comments are defined from line 20 of the dialog box onwards, the following comments are defined.

// anything after this for the whole line is ignored

\ anything after this for the whole line is ignored

(anything after this is ignored until) or the end of the line is reached

{ anything after this is ignored until } or the end of the line is reached

The last rule for the { type brackets is necessary otherwise BVT may confuse a local variable with a global variable.

The definition for the contents of this dialog are kept in 'BVwordTypes.txt'. This file must exist at BV start up otherwise an error occurs.

10. Limitations

In an attempt to keep the process as simple as possible there are some limitations. For example Forth will quite happily accept:

0 CONSTANT FALSE -1 CONSTANT TRUE

On one line but BVT will insert the whole line. The only negative consequence of this is that both constants will be defined even if only one is required.

11. End Marker

The end of file marker (;s) is not now required as BVT will stop sending when it reaches the end of the file. BVT will then send the ;ss. This may cause some 'messy' endings but it does not effect the loading. If an existing file sends an end marker, BVT will finish by timing out. The end marker (;ss) by default can be changed using the settings dialogue.

12. BV Forth

In order to use the special features of this software the Forth must be version 1.201 or above. This is because of the new protocol. If the file has an end marker in it (;s), BV Forth will report an error if a successful look up has been found in the library, this may well be a false error.

To explain, with word ;S will terminate the loading and report any errors that may have occurred during loading. As far as the target is concerned if there have been any word not found then this is an error even though BVT may have handled this. To get round this there is another word ;SS that is in version 1.201 and above that will terminate the loading process but not report any errors. The status screen of BVT will indicate if there have been any problems in finding words in the library.

There may be other as yet undiscovered limitations but these can be overcome by using a combination of include, library and code form the current file. If any are discovered please write to me and I will attempt to fix, jim.@byvac.com