2nd November 2018 Martin Head

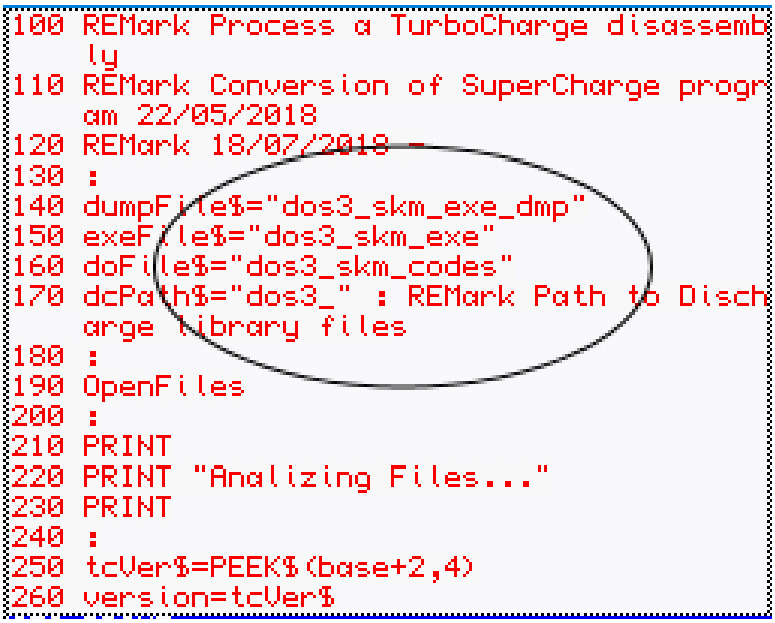
DisCharge worked example

This is a worked example of decompiling a TurboCharged executable file. The program we are going to use is Super Kit Merger by Emmanuel Verbeeck. The full program including the original BASIC program is included in **Super_Kit_Merger_V1_60.zip**.

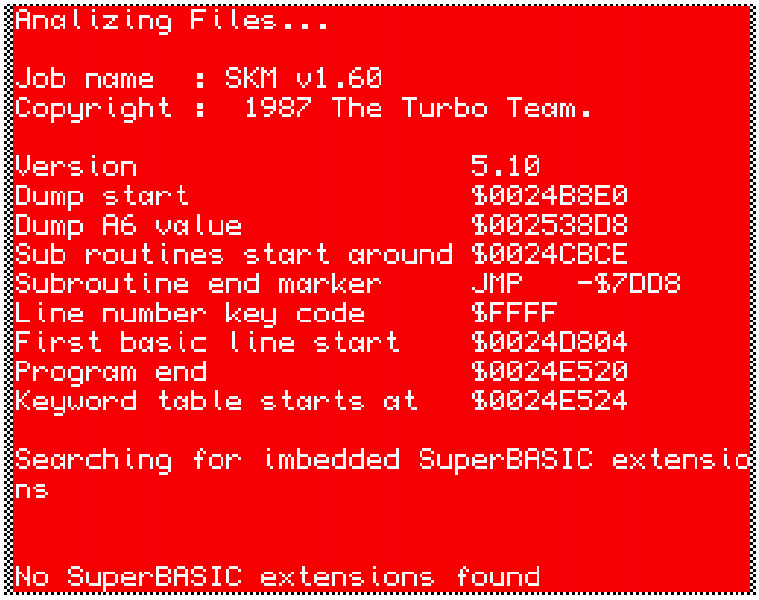
The first step is to Disassemble the compiled executable, **skm_exe**, in the Talent/Quanta Assembler Workbench. You could use another disassembler, but you may have to make changes to the DisCharge programs to cater for any differences in the generated disassembly. For the sake of this example I have supplied a disassembly of the program for you, **skm_exe_dmp**.

The next step is to process this disassembly to identify the various subroutines generated by the compiler when Super Kit Merger, was originally compiled with TurboCharge, So you will need to use the program **TurboProcessDump_bas**.

Load TurboProcessDump_bas into QPC2 and edit lines 140 to 170



Now **RUN** this program.



This will create **skm_exe_dmp_lib** and **skm_codes**. **skm_exe_dmp_lib** is a modified copy of **skm_exe_dmp** with the various routines separated and identified where possible. **skm_codes** is a list of these routines that is used by the main decompiler program.

In this example you should not need to look at these files, but if some of the routines were not identified by **TurboProcessDump_bas**, then you would need to use **skm_exe_dmp_lib** to try to identify them by hand.

Looking at the screen shot above, note that **Line number key code** is **$FFFF**. This means that the original BASIC program was compiled with the **Omit Line No** option. So there are no in-bedded line numbers in the compiled program, and Discharge will need to generate it's own line numbers. It also means that DisCharge will have no idea of where the original program lines start and end. So it will assume one program statement, per program line. So if the original program had three statements on one line, DisCharge will generate three program lines.

The absence of line numbers mean that you will need to use the **TurboDisCharge2_bas** to do the decompilation. **TurboDisCharge1_bas** is used for compiled programs that have in_bedded line numbers.

The next step is to load **TurboDischarge2_bas** into QPC2, and edit lines 170,190,and 1020

```
100 REMark TurboDisCharge
110 REMark
120 REMark Started 25/05/2018
122 REMark Version for line numbers ommited
124 REMark Main program split 24/09/2018
130 REMark 23/10/2018
140 :
150 CLS
160 REMark Filename is the executable TurboCharged task
170 filename$="dos3_skm_exe"
180 REMark Code array contains the subroutine offsets
190 codeArrayData$="dos3_skm_codes"
200 :
210 LoadFile filename$
220 GetVersion
230 :
240 PRINT#0,"Type GOTO 1000 from now on "
250 PRINT
260 :
270 STOP
1000 REMark ******************************************
1010 REMark Program is loaded. Start from here now.
1020 pauseLine=40000 : REMark Change to pause at at required line
1030 :
1040 CLS
1050 REMark Define arrays used in program
1060 DIM nameList(NameListLen(keywords)) : REMark array of offsets
1070 DIM arrayElements(30,1) : REMark 31 entries of 2
1080 DIM DATAarray(30)       : REMark DATA statement RESTORE point
```

Line 1020, sets a line number that the decompiler will pause at. After the decompilation listing stops at this point, Press any key to continue one instruction/line at a time.

Setting it at 40000 is higher than any valid line number, so the decompilation should not pause at all. Note, if you are decompiling a very large program with omitted line numbers, it is possible that the decompiler will generate listings with line numbers greater than 32768.

Now **RUN** the program. After the run, use **GO TO 1000** start the listing generation.

If you encounter any problems, or want to perform the listing generation again, then just **GO TO 1000** again.

```
Found 5 Array(s)

Scanning for Procedure and Function calls, code 99
No Proc/Fun code key in key code array
 for code 0. Aborting scan
Scanning for Procedure and Function calls, code 100
No Proc/Fun code key in key code array
 for code 0. Aborting scan
Start of code       00257B64
A6 value            0025FB5C
Line number prefix  FFFF
Variable Init start 00259A9C    00001F38
BASIC program start 00259ABE    00001F5A
BASIC program end   0025A7A2    00002C3E
Keyword table start 0025A7E6    00002C82
End of code         0025A8B0

Enter filename for output file
_bas & _log extensions will be added
ENTER alone for output to screen

Filename -
```

Just press **ENTER** at the filename prompt, and you should see the code listing scroll up the screen.

Enter **GO TO 1000** again, and this time enter a file name at the prompt. A BASIC program will be generated with a _bas extension.

Before trying to load this program into QPC2, First load it into a text editor and have a look for any obvious problems that might upset SBASIC.

116 END_WHEN/CONTINUE this should be 116 END_WHEN

2044 EXIT 3396
2054 EXIT 2214 : END IF : END IF
2210 GO TO 1758
2674 EXIT 3326 : END IF
2750 EXIT 3326 : END IF
3316 NEXT 1146
3322 GO TO 2644
3392 GO TO 1146

The **NEXT** and **EXIT**'s along with the **GO TO**'s suggest **REPeat** loops.

Add the following lines

1145 REPeat loop1146
1757 REPeat loop1758
2643 REPeat loop2644

And edit the other lines to

2044 EXIT loop1146
2054 EXIT loop1758 : END IF : END IF
2210 END REPeat loop1758
2674 EXIT loop2644 : END IF
2750 EXIT loop2644 : END IF
3316 NEXT loop1146
3322 END REPeat loop2644
3392 END REPeat loop1146

There are some integer **FOR** loops, where the **END FOR** variable end with a %, but the corresponding **FOR** variable does not have the % added .

Add a % onto the **FOR** variables in lines 1614, 1978, 2290, 2986, and 3326.

Now try loading the program into QPC2.

SMSQ/E complains about the syntax of line 2924

2924 PRINT#6,"30 LBYTES "" & var9294$ & "",adr"

This is confusion over single and double quotes, and should be

2924 PRINT#6,'30 LBYTES "' & var9294$ & '",adr'

If you look at the original program, the line was

2970 PRINT #6,'30 LBYTES "' & sortie$ & '",adr'

The program will now load into QPC2 without error.

Before trying to run the program. If you don't have Turbo Toolkit loaded, you will have to REMark out the WHEN_ERROR and RETRY_HERE commands.

There are also some CALL commands into the ROM that you might want to prevent happening.

When you RUN the program you will get an unknown procedure error on line 1938

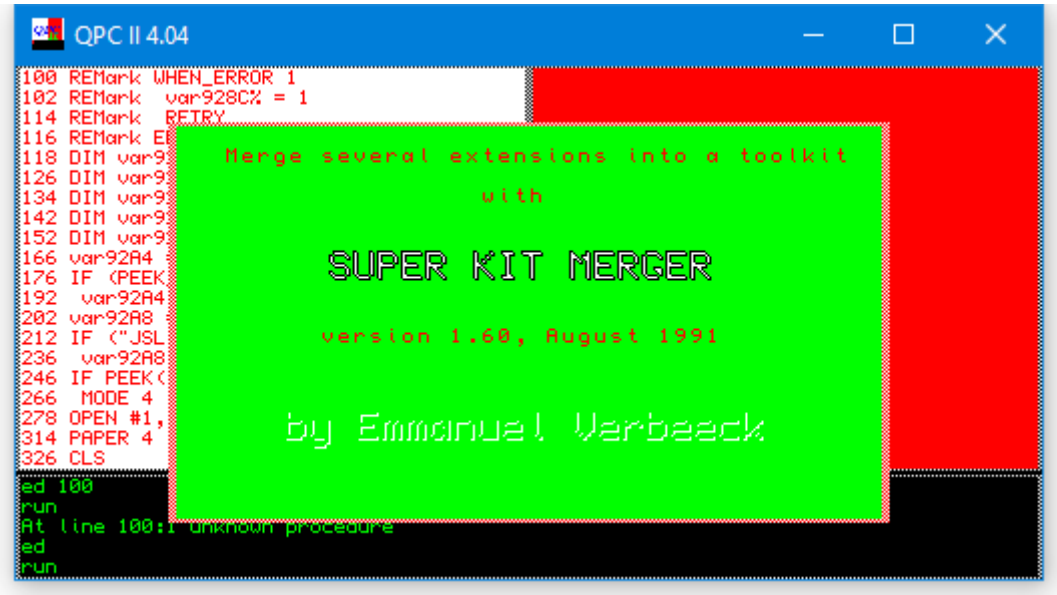
1926 INPUT#3,
1938 var92A0$(var92B4%)

this should be

1926 INPUT#3,var92A0$(var92B4%)

DisCharge has problems with INPUT to an array element.

RUN again



If you do a side by side comparison with the original executable running in Qemulator, you will notice enter the **Full names of input LRESPR files** they end up as two lines instead of one.

This is due to the decompiler not being able to keep track of the **PRINT** position of all possibly open channels. So you will usually have to add a few print separators when **PRINT**ing does not come out quite correct.

Add a semicolon (;) onto the ends of lines 1926, 2336, and 3014

There may be one or two other little things that require attention, But the sake of this example you have decompiled the executable back to a SuperBASIC program.