RomHacking 103

Script Insertion

Index

Introduction & Tools:

The document will guide you to get a proper script insertion (for a GB/GBC) from scratch, using a GBC game as an example. A good advice is reading http://www.romhacking.net/start/ first. However many of the methods, procedures and knowledge involved may be and are probably used in other game console/hardware.

For this document/tutorial, you will be required to have and use the following tools.

Notepad++, https://notepad-plus-plus.org/
Windhex, https://www.romhacking.net/utilities/291/
Game’s ROM: Kikansha Thomas - Sodor-tou no Nakamatachi (J) [C][!]
Should have been edited previously with the English font inserted.
BGB emulator, https://bgb.bircd.org/
Atlas, https://www.romhacking.net/utilities/224/
Any hex calculator

Preface

Curiosity leads to knowledge,
be curious. - Bunkai

Step 01, Getting ready for Atlas use

Make sure the insertion table is encoded in ASCII / UTF-8 or Atlas won't work.
You may want to use the same encoding for your script. But be careful not to lose your translation in the process.

For the insertion part, you need to make sure you did the cartographer dump with "atlas compatible = on". If you followed the romhacking 101 document, you probably have that covered.

Now, you add the syntax for Atlas to read the English table:

```
#VAR(Table, TABLE)
#ADDTBL("ThomasENGTable.tbl", Table)
#ACTIVETBL(Table)
```

Followed by the two pointer values for base and ending of the script text in the ROM's space.

Use the base point from the "romhacking 101" document and/or cartographer's configuration

```
#HDR($44000)
```

And, with it, calculate the offsets for the bank where you are inserting the script. For that, you'll be using the pointer reasoning in reverse. [Again, check the "romhacking 101" if you need more info].

Go to the beginning of the pointer table which, as you knew from before, is at address 49850.
The first pointer is $505a.

Since game boy is little endian, you need to invert those numbers to get the actual address, $5a50.

Subtract $4000 which is the value from the gameboy CPU
$5a50-$4000=$1a50

And then, add the starting position of the bank
$1a50+$48000=$49a50

[Another option is to directly add the value to the base point $5a50 + $44000 = $49a50]

A best approach would be using the same first offset of the text dumped, for the beginning offset.
$49852.

This will be the beginning of our text script. Go there in windhex, and, check it out!

If our assumption is correct, you can use any of the space between $49852 and the end of the bank, 4bfff, because it is a big fat empty space. That's why you'll use that bank to insert the English script. [If the assumption is not correct, just find an empty space in a different bank].

```
#JMP($49852, $4BFFF)
```

In summary, you should have something like this at the beginning of your script.txt:

```
//GAME NAME:
//BLOCK #000 NAME:        Thomas

//POINTER #0 @ $49852 - STRING #0 @ $49A50

#VAR(Table, TABLE)
#ADDTBL("ThomasENGTable.tbl", Table)
#ACTIVETBL(Table)
#HDR($44000)
#JMP($49852, $4BFFF)

//POINTER #1 @ $49852 - STRING #1 @ $49A52

#W16($49852)
```

If you wrote the “\n” or “\r” codes in the English table, delete them.
[Atlas doesn’t recognize them as control codes and they will make the execution fail]

```
FF00=<BRK>\n       → FF00=<BRK>
FF01=<NXT>\n       → FF01=<NXT>
/FF02=<END>\n\n    → /FF02=<END>
```

Step 02, Using Atlas

A reading of the Atlas.pdf is advised, but since this document try to be as easy as possible, the minimum requirements are written here.

Put all the files in the same folder:

- Atlas program,
- GBC ROM (thomas.gbc),
- English table file (ThomasENGTable.tbl), and
- script (script_00.txt),

Open the console command or command prompt and insert the Atlas command:

```
Atlas –d debug.log thomas.gbc script_00.txt
```

[You don’t really need the log file, but if you want to have a file to use as a reference when you are bug-fixing, it will be useful].

If everything went well, now your thomas.gbc is now your translated ROM.
[If something went wrong, read the log, it should tell you how to fix it].

Yep, is that easy. Go and enjoy your translated game!

Step 03, Creating a patch file

The most known options, today, have a pretty straightforward GUI:

Beat (BPS patcher),	https://www.romhacking.net/utilities/893/
Lunar IPS (IPS patcher),	http://www.romhacking.net/utilities/240/

Click the appropriate button and follow the instructions. When you finish, you will have a nice patch ready to share, or the patched ROM ready to be played. Enjoy!

[To patch your ROMs you can also use the RHDN online patcher https://www.romhacking.net/patch/].

Additional notes

The 103 document ends here, if you found an error and want to report it please send a private message to Bunkai in the https://www.romhacking.net/forum/

If you have a question about the process you may do them in the forum or in the discord, using the proper channels for that. But I won't answer any private question about your projects.

A translation project should include: script dump, font and sprites edit, and script insertion.
This document is part of a series to cover those 3, but any of the papers can be read as standalone.
If you want to check any of the others, go to:

RomHacking 101 - Script Dump, https://www.romhacking.net/documents/871/
RomHacking 102 - Font & Sprites, https://www.romhacking.net/documents/872/
RomHacking 103 - Script Insertion, (this document)

PS: A style revision and consistency revision should be made, but the intention is/was to have the 3 documents out first, so people can used them, and to include any bug fix in the review.

Credits

Guidance and explanations: Pluvius, Lusofonia.
Additional explanations: 343, Bootleg Porygon.
Document author: Bunkai

Special Thanks

TF1945 for providing the files with most of the boring stuff done.
To the RHDN discord server and the RHDN community in general for their wonderful support to this hobby with manuals, tools, explanations and so on.