EECS 222A
System-on-Chip Description and Modeling
Fall 2007

Assignment 6

Posted: November 16, 2007 (week 8)
Due: November 30, 2007 (week 10)

Tasks: Part 1: Creating Behaviors in C Code using Source Re-coder
Part 2: Pointer Elimination using Source Re-coder

Instructions: (by Pramod Chandraiah)

Part1 – Create behaviors using Source Re-Coder

In the last two assignments you introduced behaviors and replaced pointers manually. As you have experienced, this manual conversion is very time consuming and error-prone. Compared to the time taken by the automatic exploration using SCE, this time is very large. You might have also noticed that some of these manual re-coding tasks can be automated to speed-up the overall process of specification development.
So we introduce source re-coder. Source re-coder is an integration of various automatic code transformations tools and an interactive editor. With source re-coder you can not only manually type in code, but also invoke automatic tools to create to automatically re-code specification.
For this assignment, you will use the source re-coder to repeat the tasks that were performed manually in the previous assignments.

Initial Setup

The initial setup files are in the tar file `mp3_v3.tar.gz`. Untar this file using the command:

```
gtar xvzf mp3_v3.tar.gz
```

A directory by name `mp3_v3` will be created. Change into this directory

```
cd mp3_v3
```

The entire floating point MP3 source code is in single file `mp3decoder.sc` and the fix-point code is in `mp3_fixpt.sc`. There is a `Makefile` to compile and test the decoder for a set of MP3 streams. There are two directories reference/ and `testStream`/ which you don't have to worry about at this stage.
You need to set the path for the SpecC compiler. Source the setup shell script as below:

```
source /opt/cars/scc/bin/setup.csh
```

Now compile and test the decoder by running following commands

```
make clean
make all
make test
make all_fix
```

`make test_fix`
The setup should compile and simulate without errors. (ignore "`Can't step back`" and "`read length less than max`" messages)

Given MP3 Code

The MP3 code (`mp3decoder.sc, mp3_fixpt.sc`) given are the same sources you received for your previous assignments.

Source re-coder

Run the source re-coder by typing: `/opt/cars/cute/bin/cute &` from `mp3_v3` directory.
An editor comes up. There are 3 views in the editor: the *main view* which is initially blank, a *side view* which lists the directories and files and a *message panel* at the bottom with different tabs for each message type.
Open the file `mp3decoder.sc` from **`File→Open`** and selecting `mp3decoder.sc`. The code will appear in the *main view*. Now enable the line numbers from **`View→Line numbers`**. Scroll down and you will see the following 4 behaviors in the code: `Stimulus` [at line 5724], `Monitor` [at line 5692], `DUT` [at line 5640] and `MP3Decoder` [at line 2755]. Notice that bodies of behaviors are colored in blue and channels in light brown and all the global entities are uncolored.

Source re-coder has 2 tool bars. The first one consists of basic editing tools copy/paste and so on. The 2nd tool bar consists of tools to perform the code transformations. If you hover the mouse over these icons, you can see the names of each of these tools.
For this part of the assignment you only need to be aware of following tools:

- `Current Position` **(CP)**: 2nd button from the left (Icon: Bulb)
- `Build Design` **(BD)**: 3rd button from the left (Icon: Green recycle symbol)
- `Function 2 Behavior` **(F2B)**: 4th button from the right (Icon: Blue rectangle with rounded edges)
- `Statement 2 Behavior` **(S2B)**: 3rd button from the right (Icon: Lines and Blue rectangle)
- `ReScope Variable to Class Scope` **(RCS)**: (Icon:3 red vertical triangles)

Now on, these tools will be referred as **`CP, BD, F2B, S2B`** and **`RCS`**.
Before we start, we have to compile the code and build the design. Use can you the **`BD`** button to build the main design.
The tools are invoked by placing the cursor at the desired point of interest in the code and by clicking on the appropriate button in transformation tool bar. The source re-coder invokes the tool on the object at the current cursor position.
Since the design is sparser than the source code, not every cursor position points to an object (variable/behavior/function/statement) in the design. So before you invoke **`F2B/S2B/RCS`** tool, place the cursor at the point of interest and

invoke **CP.** This will display different possible objects corresponding to this cursor position and the result will appear in the *SpecOut* tab of the *Message Panel*. Usually, this list will contain the variable and the statement at that position. For example, if you put the cursor on line 2778 next to function call `decodeMP3( )` (specifically, to the left of letter 'd') and invoke **CP**, you will see 2 entries in the *message panel*. 1st entry for the function symbol `decodeMP3`, and 2nd for the whole function call statement. If you see your object of interest in this panel then you can invoke the necessary tool. If you don't see the object of interest, then the current cursor position is not valid and you have to re-position the cursor to a different position where the same variable is again used.

Now you will invoke the tools from the transformation tool bar to introduce some behaviors including those which you introduced manually in the previous 2 assignments. Note that you will re-build the design by through **BD** button after invoking any transformation that changes to the code.

Task set 1:

1. **Please make a note of the start time T0**.
2. Open the file `mp3decoder.sc` as explained above in the source re-coder. Build the design using **BD** button.
3. First, we will encapsulate one of the calls to the function `decodeMP3()` in a behavior. In the source recode navigate to the line number 2778 and place the cursor next to the function `decodeMP3( )`. Specifically, place the cursor to the left of letter 'd'. Now invoke the **F2B** tool and realize the transformation instantly. The cursor will be re-positioned following the changes. You will see a behavior `B_decodeMP3` at line number 2764. Navigate around to see the changes to the code.

4. Re-build the design by clicking on **BD**
5. Scroll down to line 2833 to the function call `do_layer3( )`, position the cursor appropriately and invoke **F2B** . New behavior `B_do_layer3` will appear starting at 2770.
6. Re-build the design by clicking on **BD**
7. Save the file by **File→SaveAs** and choose the filename `mp3decoder.sc`. Note that this is the only way to save changes; the other ways through `Save/SaveAll` buttons do not work.
8. Re-build the design by clicking on **BD**
9. **Note the time T1.**

10. Now we would like to wrap the statements from line 2792 to 2823 into behavior. Before that, we have to move the variables to the behavior scope. This can be done using RCS tool. If you analyze these statements, you will realize that following local variables must be rescoped: `stereo, stereo1, single, sideinfo, sfreq, ms_stereo, i_stereo, granules`.

Place the cursor to the left of these variable names in the code, either at their line of declaration, or at the lines they are used and click on **RCS** button. When not sure if the cursor is pointing to the right symbol, place the cursor and click on **CP** button. **RCS** must be invoked on each of the above listed variables one at a time. You do not need to use the **BD** button in-between successive **RCS** invocation as this is taken care by **BD**.

11. Re-build the design by clicking on **BD**
12. Save the file by **File→SaveAs** and choose the filename `mp3decoder.sc`.
13. Re-build the design by clicking on **BD**

14. To wrap the statements from 2792 to 2823, select the lines (by pressing and holding the left-mouse button or using shift key and scrolling down) from **2793** to **2812** and invoke **S2B**. New behavior `B_child_B_do_layer3` will be created at line 2781 with all the lines from 2792 to 2823 selected. Note that we are not highlighting all the lines till 2823. This is because, as we mentioned before, there do not exist exact one to one correspondence between the cursor position and the objects in the main design. This is an exception and applies only to these set of lines.

15. Re-build the design by clicking on **BD**
16. Save the file by **File→SaveAs** and choose the filename `mp3decoder.sc`.
17. Re-build the design by clicking on **BD**

18. Note the time T2.

19. Rescope the variables used between lines 2896 to 2921 (`hybridIn, scalefacs, gr_info, gr`)
20. Now wrap the statements from line 2898 to 2925 into a behavior (lines numbers slightly changed after re-scoping) using **S2B**. After this, the statements are wrapped and replaced with instance call `I_B_child_B_do_layer4` at line 2927.

21. **BD, File→SaveAs, BD**
22. **Note the time T3**.

Task set 2:

23. In addition to the 4 behaviors, we will introduce couple of more behaviors. **Note the time T4**
24. Wrap statements from line 2941 to 2968 into behavior. Highlight from 2941-2968 and call **S2B.** `I_Bchild_B_do_layer5.main()` will be introduced at line 2985.
25. **BD**, **SaveAs, BD**
26. Encapsulate function call `III_antialias` at line 2992 into behavior using **F2B.** Do not forget to re-scope `gr_info` before doing this.

27. **BD**, **SaveAs, BD**
28. Repeat the same for function III_hybrid( ) which is now at line 3050. Do not forget to re-code `ch, hybridOut`, before doing so. I_B_III_hybrid.main() will appear at line 3139.
29. **BD**, **SaveAs, BD**
30. Close the file.
31. **Note time T5**
32. Test the changes by running following commands from your terminal:

```
make clean
make all
make test
```

At the end, please report the times **T0, T1, T2, T3, T4** and **T5.**

Part2 - Pointer Replacement

Just like the way behaviors can be created with a click of a button using source re-coder, pointers can also be recoded.

Given Fix-point MP3 Code

This is the same fix-point MP3 implementation you used for the previous assignment.

Source recoder

Run the source re-coder by typing: `/opt/cars/cute/bin/cute &` from `mp3_v3` directory.
Note the 2 more tools in the transformation tool bar.

- `Points-to List` **`(PL)`**: 9th button from right (Icon: Pointer with a question mark)
- `Recode Pointer` **`(RP)`**: 8th button from right(Icon: Pointer with crossing line)

Now on these tools will be referred to as **`PL`** and **`RP`**

Open the file `mp3_fixpt.sc` from **`File→Open`** menu. The code will appear in the *main view*. Now enable the line numbers from **`View→Line numbers`**. Scroll down and notice the following 4 behaviors in the code: `Mad_Stimulus` [at line 13681], `Mad_Monitor` [at line 13533], `MP3Decoder` [at line 13717] and behavior `Main` [at line 13746]. We will replace couple of pointers in the behavior `Calc_sample` located at line 13177.
Build the design using **BD.**

Replace pointers (fe, fx, fo, Dptr) in behavior Calc_sample

1. **Note the time T0**

2. Build the design using **BD**
3. Place the cursor next to `fe` (say at line 13202 next to letter 'f') and click on **PL**. This will display the target variable the pointer points-to in the *message panel*. If the pointer points to only one variable (as in the case of fe) then you can recode it. If the **PL** does not display any target variable, then it means that source re-coder failed to analyze the code.
4. Invoke **RP** and observe the changes to the code.
5. Re-build the design through **BD.**
6. Repeat these steps 3 to 5 for `fx`, `fo`, and `Dptr`.
7. **BD**, **SaveAs, BD**
8. `make clean`
9. `make all_fix`
10. make `test_fix`
11. **Note the time T1**

Deliverables:

- 1-paragraph description about each of the tasks above
  (i.e. how far you got, what were the problems, how did you solve it)
- Please also report the times for part 1 (T0 to T5, in seconds/minutes), and for part 2 (T0 to T1, in seconds/minutes).

Due: Week 10 (Nov 30, 2007)

--
Rainer Doemer (ET 444C, x4-9007, doemer@uci.edu)