Modifications of GHC for

David Sabel

last update for version 1.1.: October 22, 2003

This document contains a list of the modifications of the GHC source code (version 5.04.3) for HasFuse.

Notice: Modified or added lines in the source code are marked with `[DS]`, so you can easy search for those lines.

Preliminary remark

Since version 1.1 we differ between two modes in which the compiler is running:

1. **HasFuse mode** (ON by default)

   Only safe (in our meaning) program transformations are performed in optimisation level 0 and 1. Level 2 is ONLY for testing purposes.

2. **GHC mode** (comes with `-fno-hasfuse`)

   The compiler should behave like a "normal" GHC, so it's not safe in our meaning. Note, that the libraries are built in HasFuse mode and so the compiler doesn't behave such efficient like a normal GHC.

ghc/compiler/main/DriverFlags.hs

- added a new flag `--hasfuse`, which shows some information about HasFuse.
- added a new flag `-fstrictness`, which turns on strictness analysis in HasFuse mode. (Strictness analysis is OFF by default in HasFuse mode and ON by default in GHC mode). The modes are controllable with the `-fno-hasfuse` mode.

- added the flag `-fno-hasfuse`, which turns off the HasFuse mode and switches to the "GHC mode", i.e. the compiler should behave like a "normal" GHC, but note that the libraries are built in HasFuse mode and aren't so efficient.

ghc/compiler/main/DriverState.hs

- changed the default value for the global variable `v_Strictness` to `False`.
- added the global variable `v_HasFuseMode` with default value `True`.

  `True` ⇔ compiler is in HasFuse mode and performs only optimisations which are safe in our sense (except if `-O2` option is used)

  `False` ⇔ compiler is in "GHC mode" and should behave like a normal GHC
- new function `getHasFuseMode` with returns the mode, by reading the global variable `v_HasFuseMode`
- Changes in `hsc_minusO_flags` for the HasFuse mode:
  - ignore interface pragmas
  - omit interface pragmas
  - don't do the foldr-build transformation
  - turn off cpr
- Changes in `hsc_minusO2_flags` for the HasFuse mode:
  - There's a real definition now, because we turn foldr-build on and cpr not off. The rest of the flags are same as `-O1` flags.
- Big changes in `buildCoreToDo` for `-O1` and `-O2` for the HasFuse mode:
  - Strictness analysis is now off by default at level 0 and 1, but it can be switched on by the flag `-fstrictness` (Worker Wrapper is then not on!).
  - Optimisation level 1:
    Do only Floating-In and run the Simplifier
  - Optimisation level 2:
    Do every optimisation as before, that aren't tested for safeness.
  - The unsafe transformations Floating-Out and Common Subexpression Elimination aren't performed in any optimisation level! The liberate case transformation is also never performed, because we have nearly no documentation about it.

ghc/driver/ghc-usage.txt

- I added information about HasFuse and the `--hasfuse` flag.
- I added information for use of optimisation level 2, which isn't proven as safe.

ghc/mk/version.mk

- I changed the Projectname to HasFuse

ghc/compiler/coreSyn/CoreUtils.lhs

- The changes in `exprIsCheap`, `exprIsTrivial`, `exprEtaExpandArity`, `exprIsValue` only appear if the compiler runs in HasFuse mode. Therefore the mode is explicitly imported.
- I've made some changes in the definition of `exprIsCheap`:
  - An expression is now cheap, if it's a variable, an unapplied primitive operator with positive arity, a literal, a constructor or a constructor application where the arguments are cheap.
  - I also allow things about types to be cheap, because FUNDIO has no types.
  - Primitive operators are treated like abstractions, so primitive operators with arity 0 aren't cheap
- I've made some changes in the definition of `exprIsTrivial`:
  - I don't allow to duplicate primitive operators or foreign-call Ids with arity 0, so I changed `exprIsTrivial` for the `Var` case.
- I replaced the definition of `exprEtaExpandArity` by `exprArity`, because the eta-expansion performed by the GHC is not safe in the sense of FUNDIO.
- I've made some changes in the definition of `exprIsValue`:
  - Partial applications are no longer treated as values in general. I only allow applications to constructors with to few arguments as values.
  - `exprOkForSpeculation` is no longer used in `exprIsValue`, it's replaced by `False`. It would be nice to turn off `exprOkForSpeculation` fully. I tried that with the definition `exprOkForSpeculation = False`, but then the compiler fails while building the libraries (`Data/Array/Base.hs`).

ghc/compiler/simplCore/OccurAnal.lhs

- Importing the mode (HasFuse/GHC) from CmdLineOpts.
- I've made some changes in `occurAnal` for the `Lam` case, so that `InsideLam` is now also `True` for one-shot-lambdas. This changes only appear in HasFuse mode.

ghc/compiler/simplCore/SimplUtils.lhs

- Importing the mode (HasFuse/GHC) from CmdLineOpts
- I changed `mkCase1` for the HasFuse mode: We don't check for `exprOkForSpeculation`, because we don't know if case elimination is correct in this case.

ghc/compiler/main/Main.hs

- We print a new message for the modified version (HasFuse), each time GHC is running in HasFuse mode.
- changed the verbosity information for HasFuse.
- added a warning if `-O2` is used in HasFuse mode (level 2 is only for testing)

ghc/compiler/simplCore/SimplMonad.lhs

- Importing the mode (HasFuse/GHC) from CmdLineOpts.
- turned off RULES completely for the HasFuse mode, by changing the definition of `activeRule`, so that the result is `Nothing`.

ghc/compiler/utils/Panic.lhs

- changed the panic-message in `showGhcException (Panic s)`, bugs should be reported to the HasFuse webpage.

ghc/compiler/main/SysTool.lhs

- added a new function `showHasFuseMess`, that's used for printing the HasFuse message, which comes with flag `--hasfuse`. Therefore I added the global var `v_Path_hasfusemess` with initialisation.

ghc/driver/hfmess.txt

- This file is new, it contains the message that comes with `--hasfuse`.

ghc/compiler/main/CmdLineOpts.lhs

- added a new optimisation option `opt_HasFuseMode` for the HasFuse mode:

  `opt_HasFuseMode` = `True` $\Leftrightarrow$ HasFuse mode
  `opt_HasFuseMode` = `False` $\Leftrightarrow$ GHC mode (flag `-fno-hasfuse` is set)