The Design and Implementation of Gnu Compiler Collection

Uday Khedker

(www.cse.iitb.ac.in/~uday)

Department of Computer Science and Engineering,

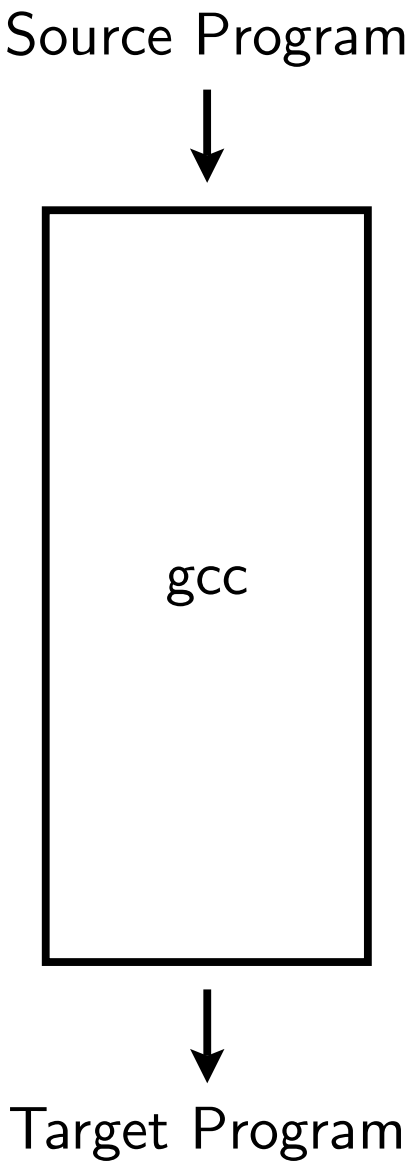
Indian Institute of Technology, Bombay

Aug 2008

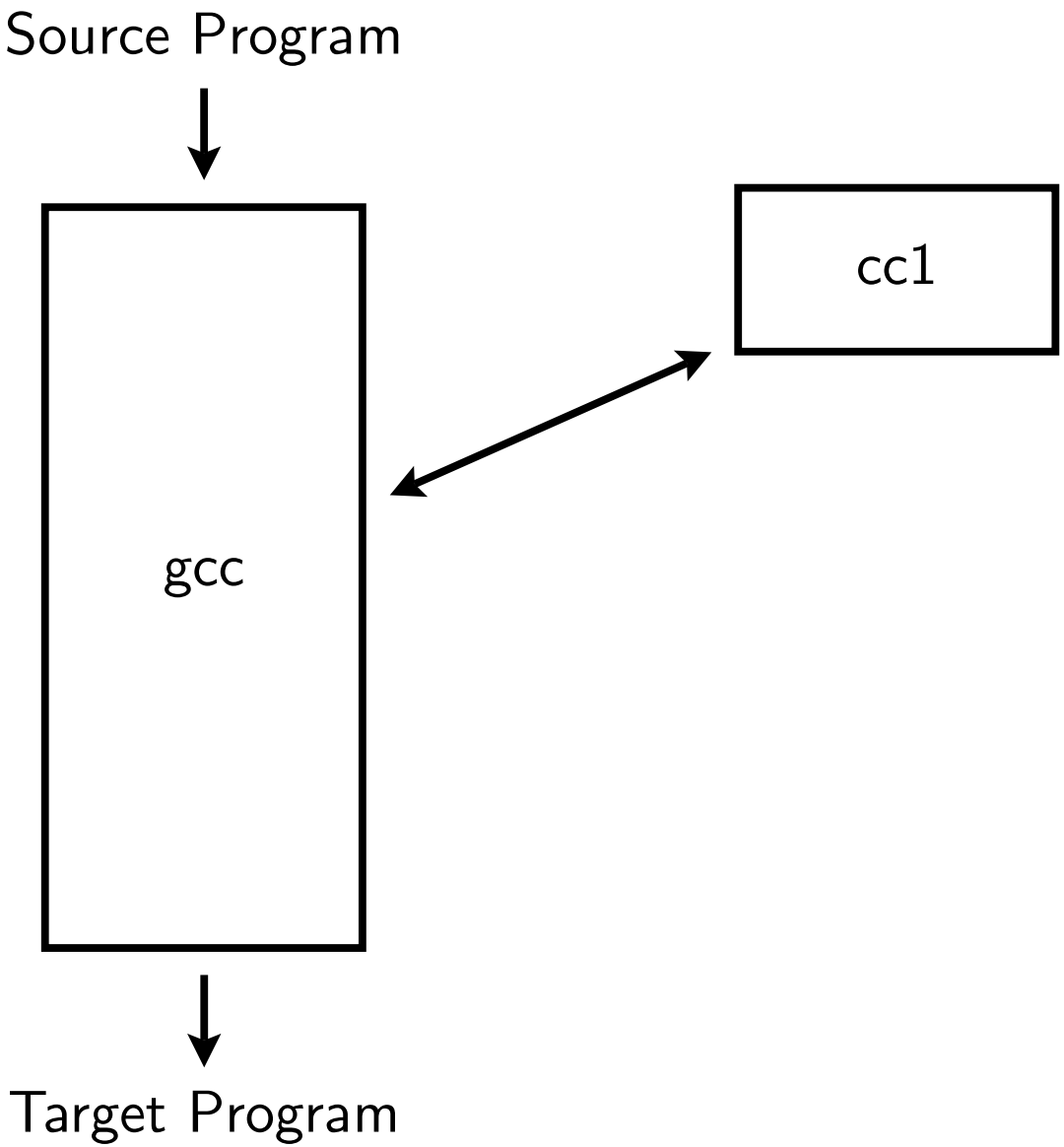
Part 1

Compilation Flow: Machine Independent Phases

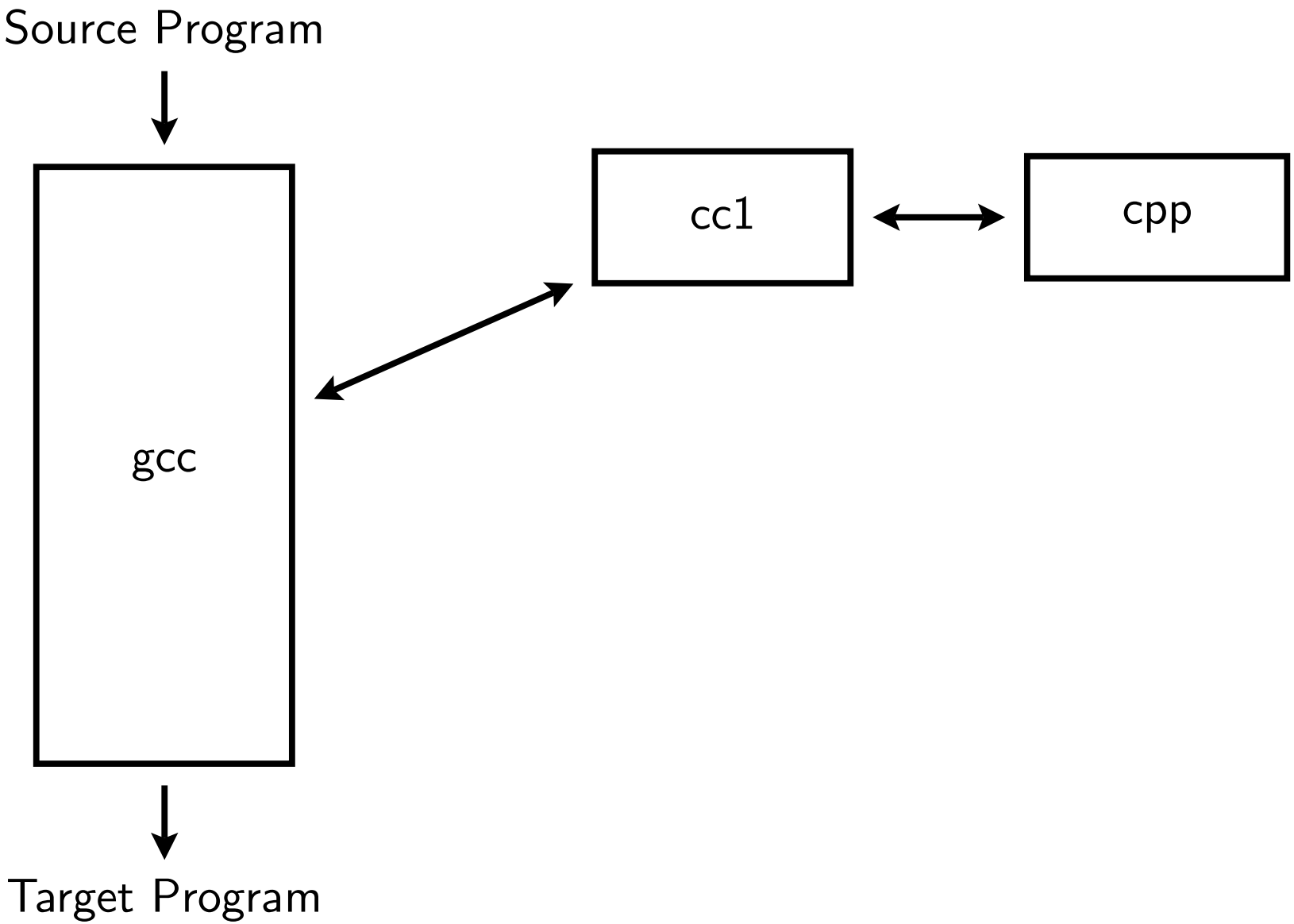
The GNU Tool Chain



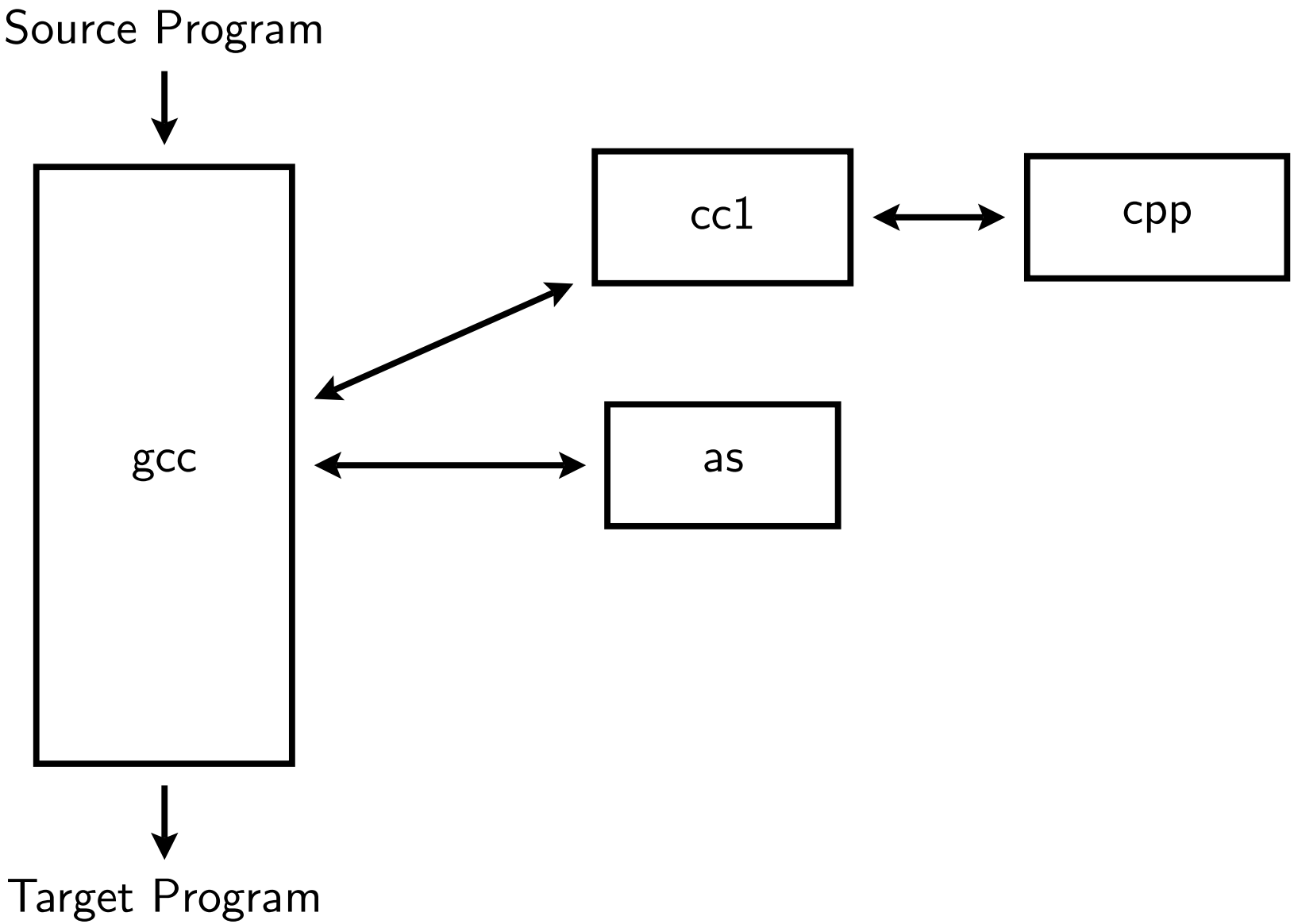
The GNU Tool Chain



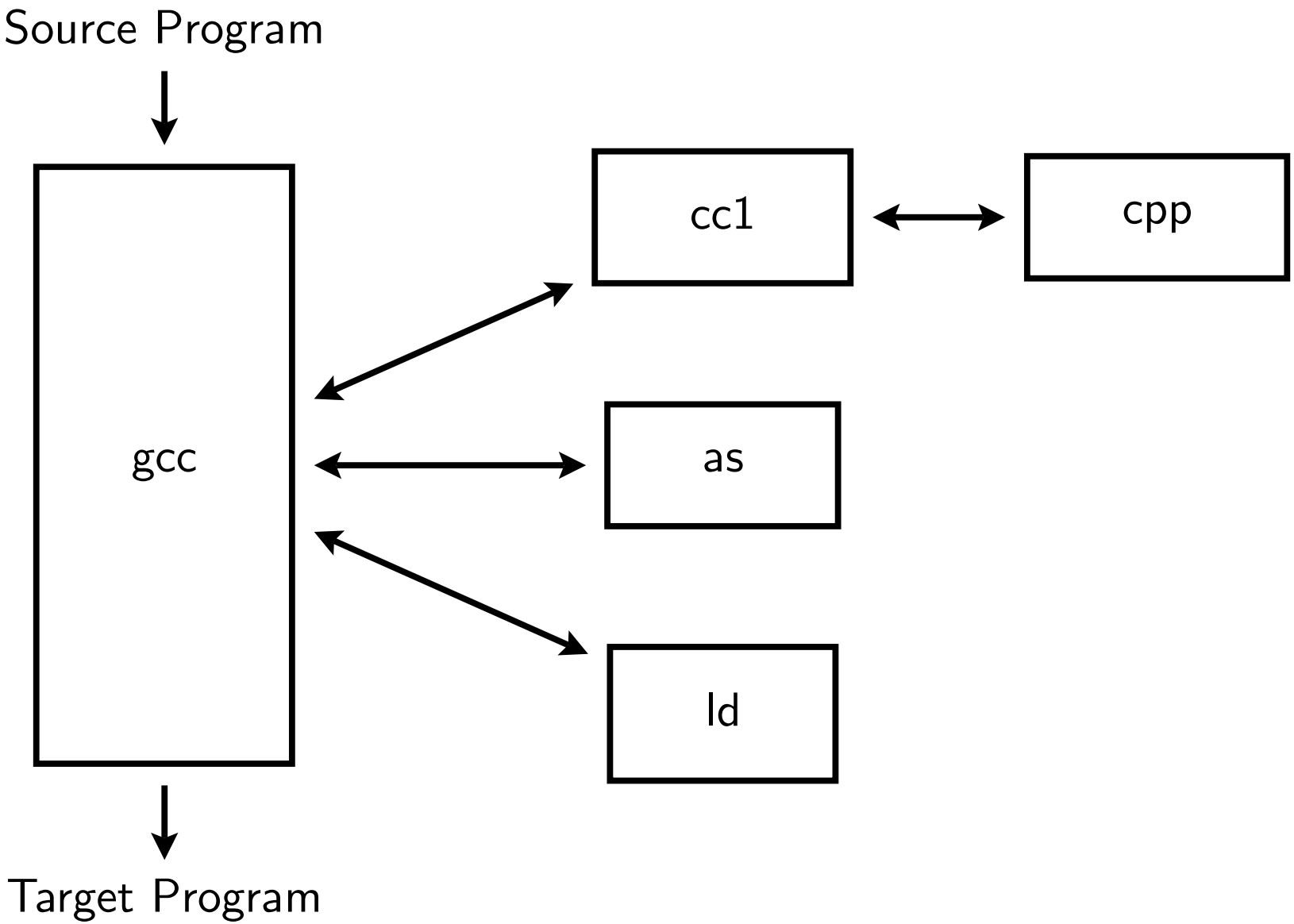
The GNU Tool Chain



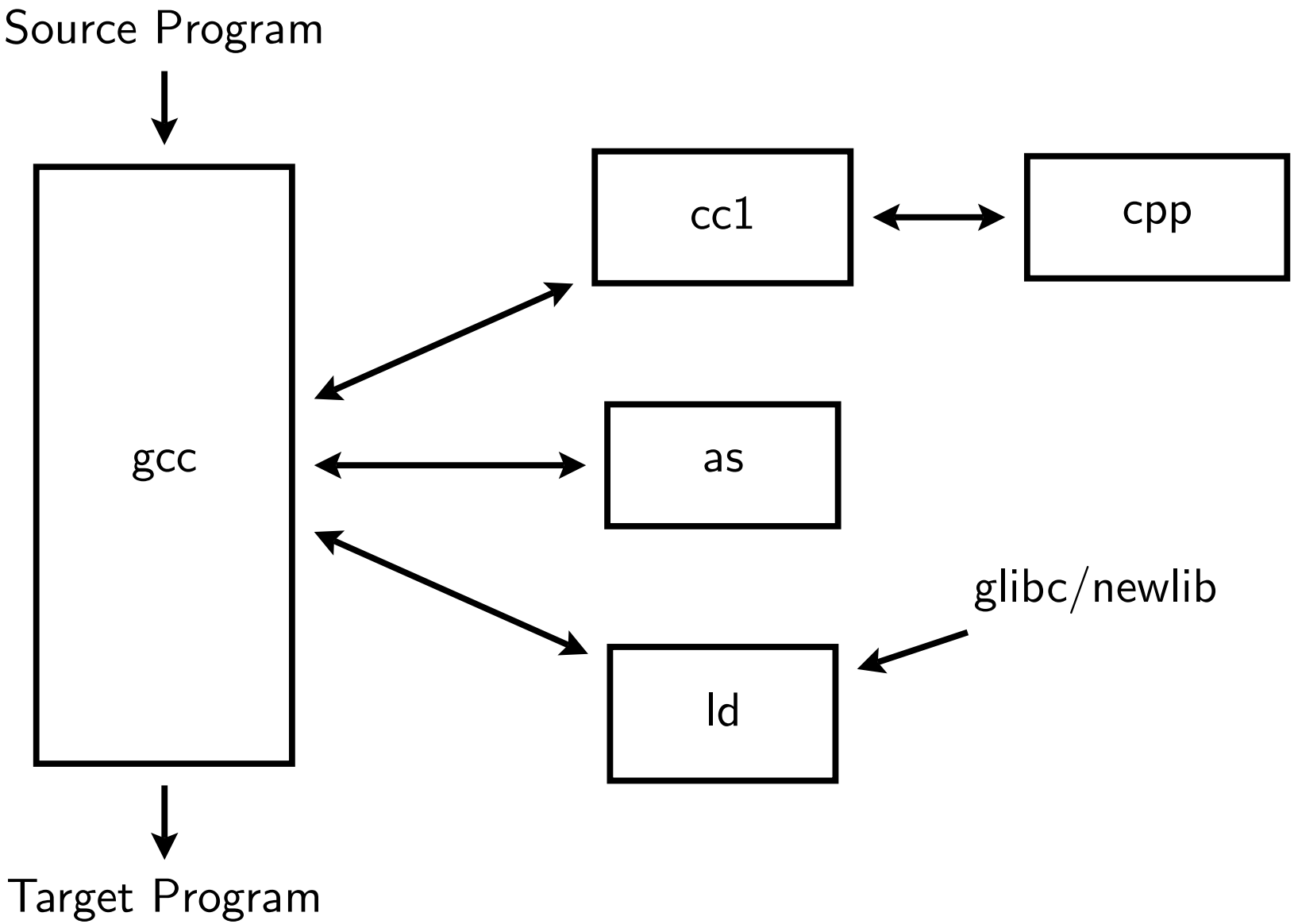
The GNU Tool Chain



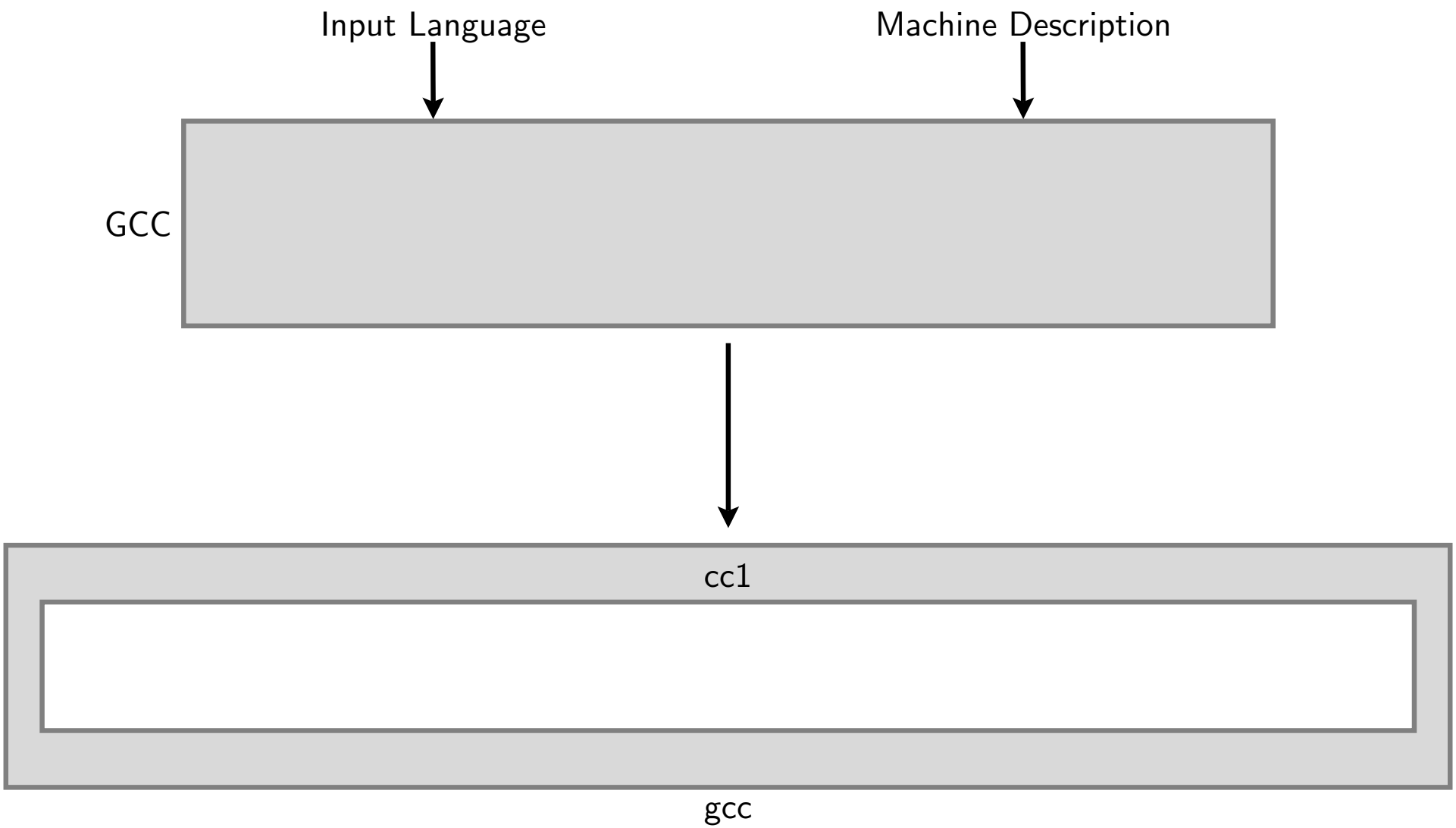
The GNU Tool Chain



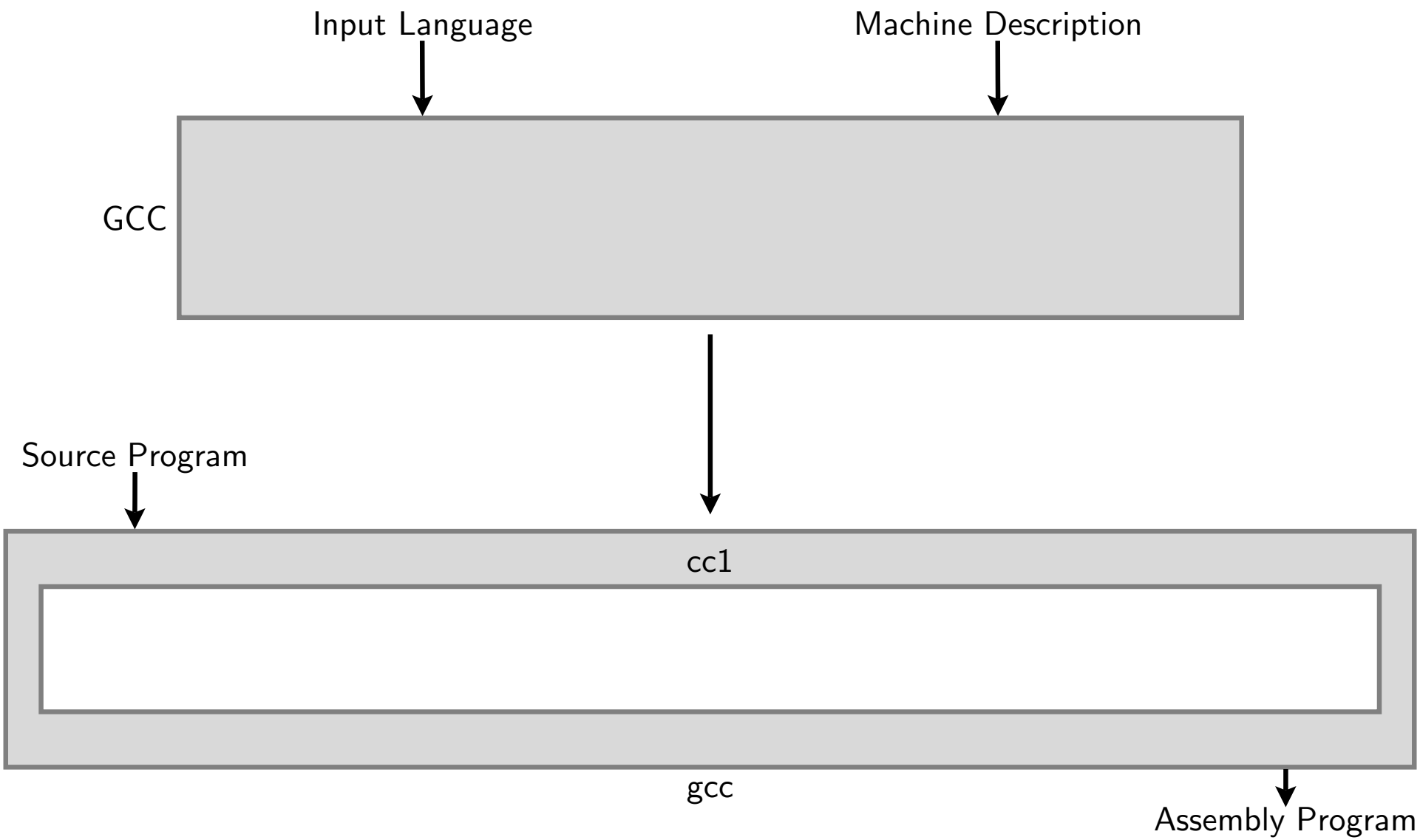
The GNU Tool Chain



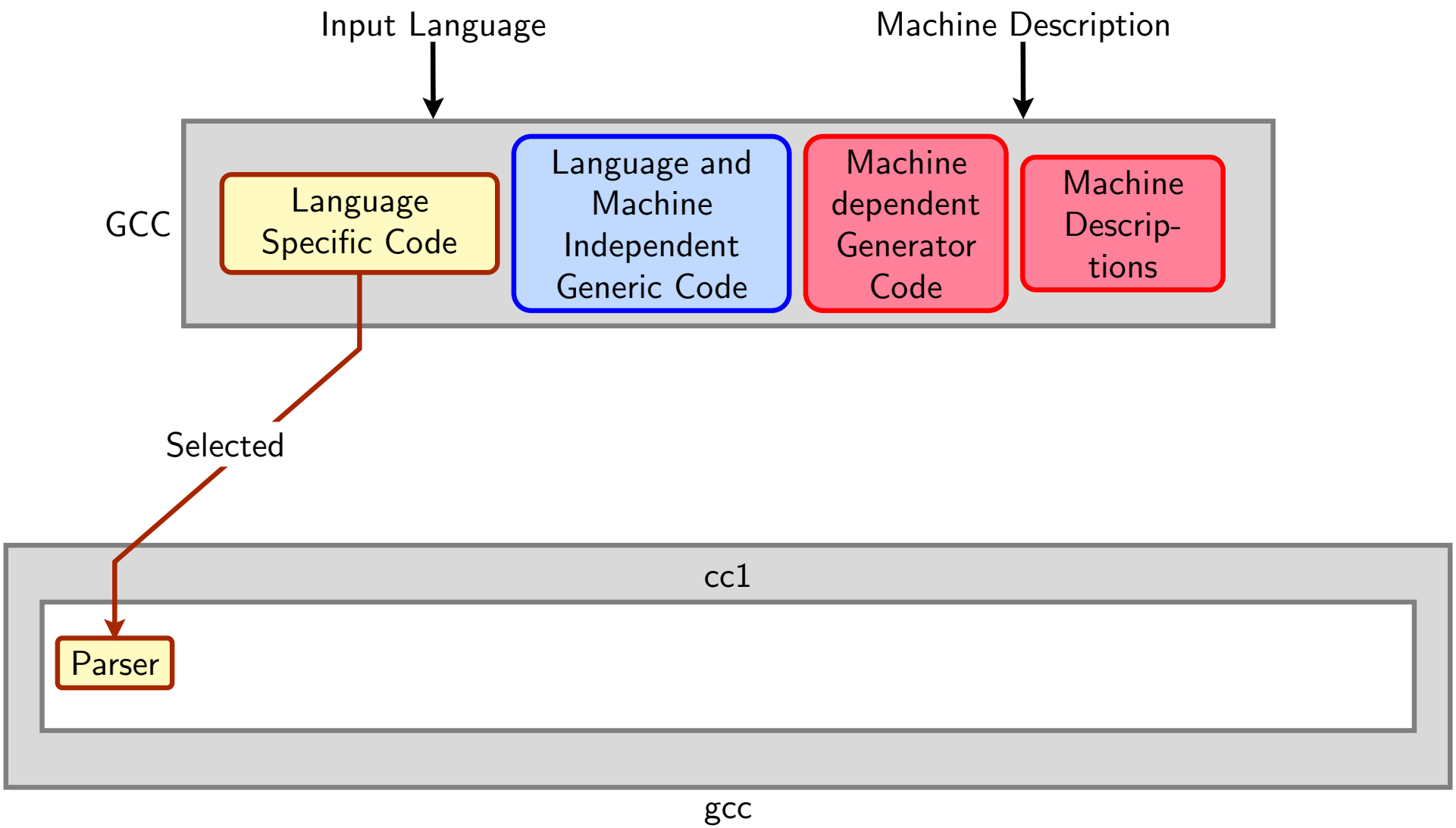
The GCC Framework



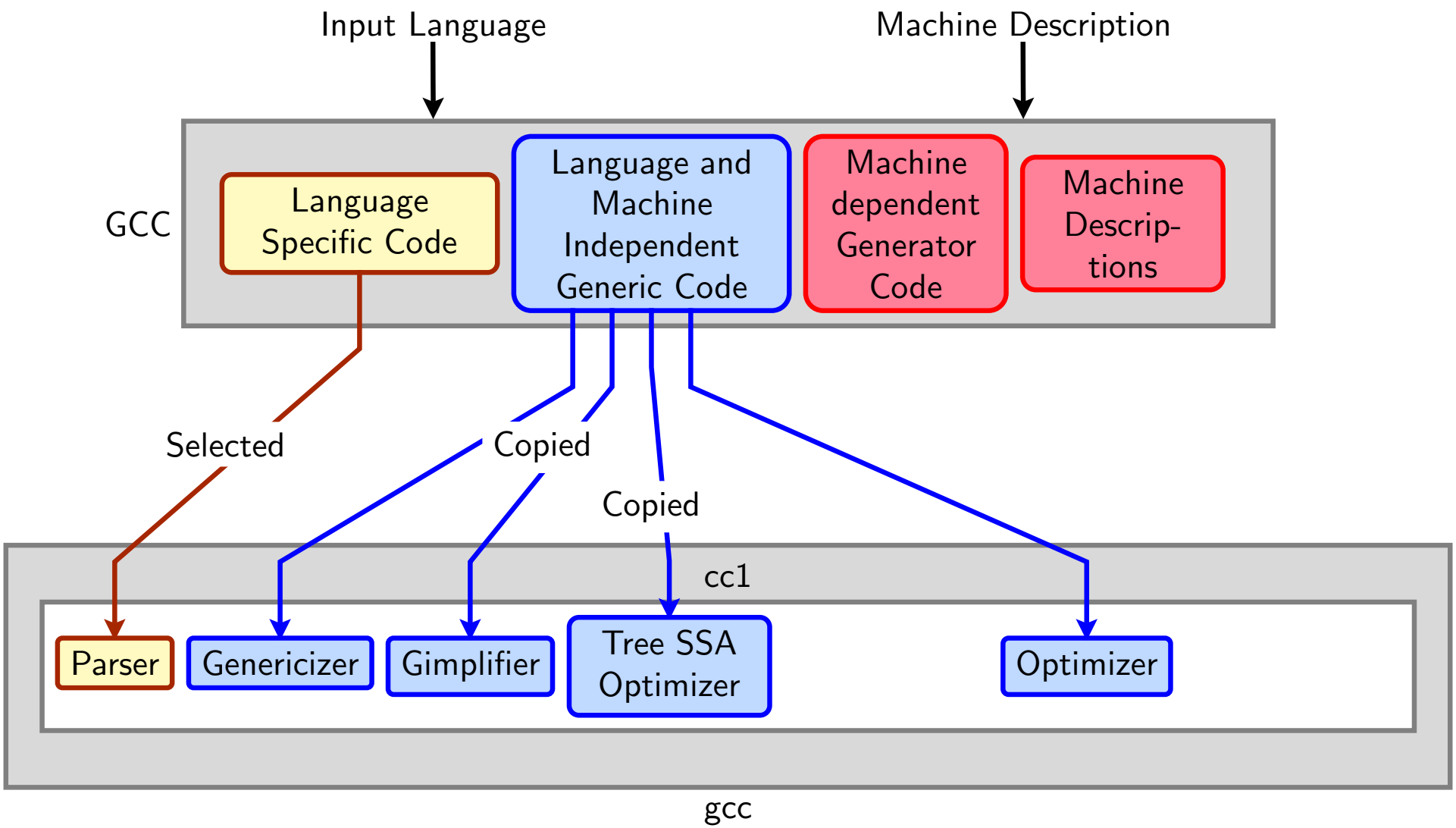
The GCC Framework



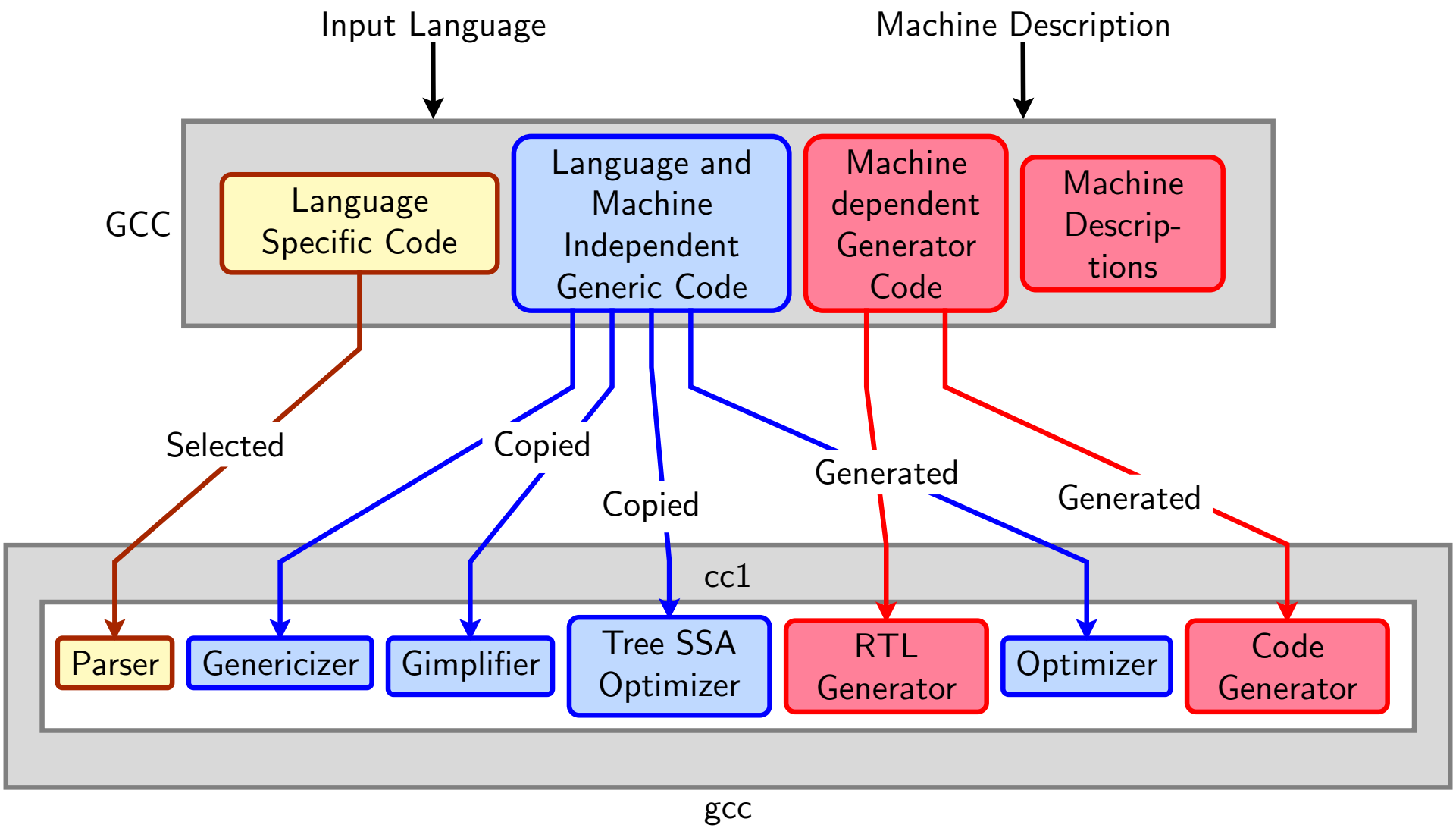
The GCC Framework



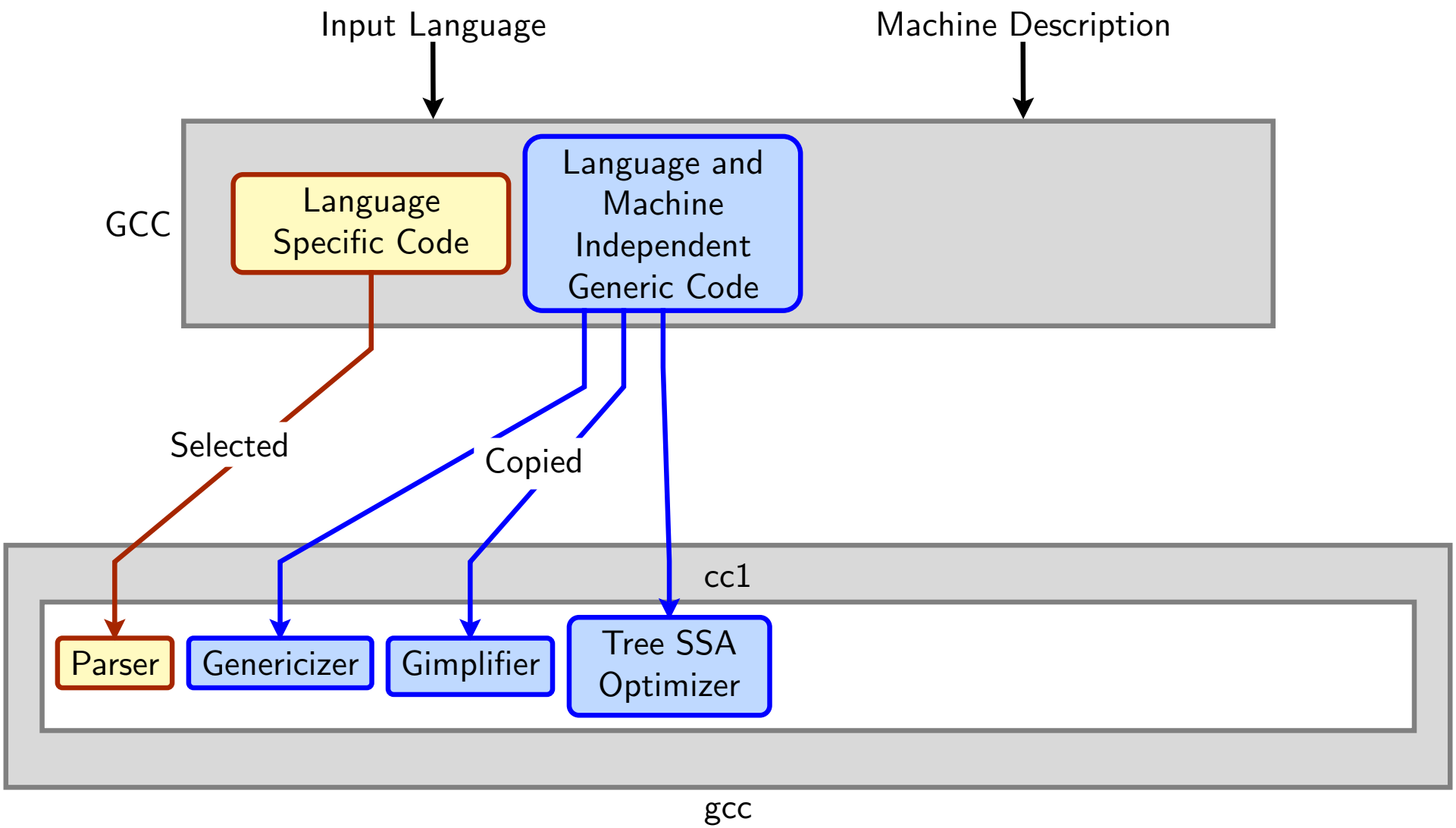
The GCC Framework



The GCC Framework



The GCC Framework



Invocation of cc1 from gcc (4.0.2)

```
main ()                                                    gcc.c
  do_spec ()                                               gcc.c
    do_spec_2 ()                                           gcc.c
      do_spec_1 ()                                         gcc.c
        execute ()                                         gcc.c
          pexecute ()                        ../libiberty/pex-unix.c
 /* TO: cc1 */
```

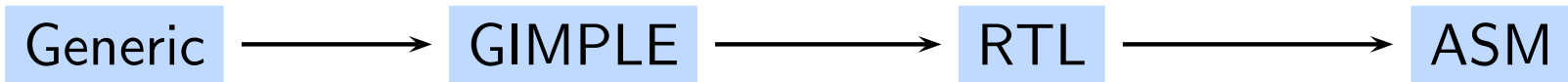
Tip

Static Inspection: Use cscope and/or ctags

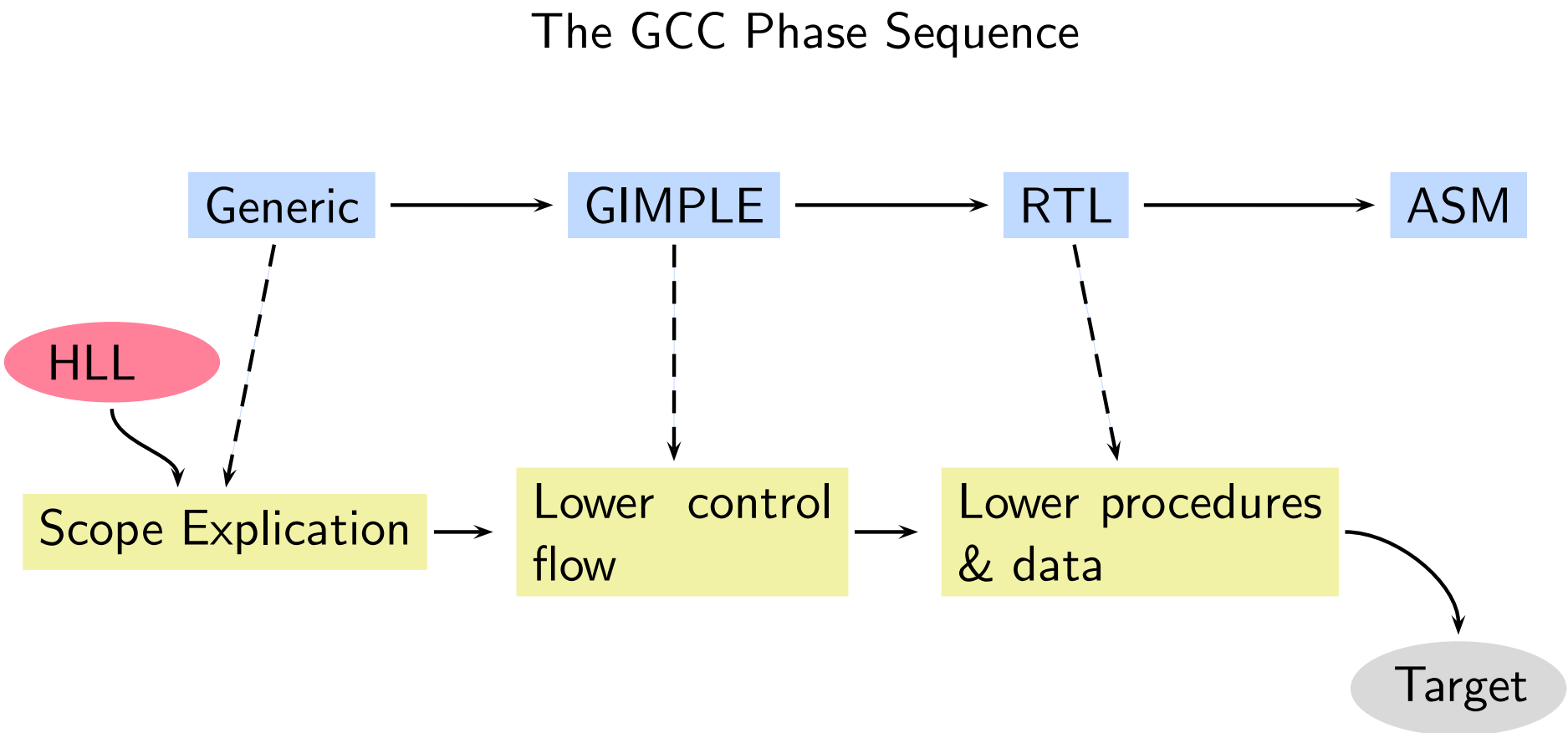
Dynamic Inspection: Set breakpoints in gdb on cc1.

The `cc1` Phase Sequence as IR Chain

The GCC Phase Sequence



The `cc1` Phase Sequence as IR Chain



Front End Processing Sequence for C in cc1 and GCC (4.0.2)

```
toplev_main ()                                          toplev.c
  general_init ()                                       toplev.c
    init_tree_optimization_passes ()                    tree-optimize.c
  decode_options ()                                     toplev.c
  do_compile ()                                         toplev.c
    compile_file()                                      toplev.c
      lang_hooks.parse_file ()                          toplev.c
        c_common_parse_file ()                          c-opts.c
          c_parse_file ()                               c-parse.y (c-parse.c)
            finish_function ()                          c-decl.c
              c_genericize ()                           c-gimplify.c
 /* TO: Gimplification */
```

GIMPLE Phase sequence in cc1 and GCC (4.0.2)

Creating GIMPLE representation in cc1 and GCC

```
finish_function ()                                   c-decl.c
  c_genericize()                                  c-gimplify.c
    gimplify_function_tree()                        gimplify.c
      gimplify_body()                               gimplify.c
        gimplify_stmt()                             gimplify.c
          gimplify_expr()                           gimplify.c
  c_expand_body()                                     c-decl.c
    tree_rest_of_compilation()                  tree-optimize.c
      execute_pass_list()                       tree-optimize.c
        execute_one_pass()                      tree-optimize.c
        /* calls pass entry point function pointer */
    targetm.asm_out.constructor()
```

The Tree passes list (4.0.2)

(Partial) Passes list (`tree-optimize.c`) ($\sim$ 63 passes)

```
pass_remove_useless_stmts     // Pass
pass_lower_cf                 // Pass
pass_all_optimizations        // Optimiser
   pass_build_ssa             // Optimiser
   pass_dce                   // Optimiser
   pass_loop                  // Optimiser
      pass_complete_unroll    // Optimiser
      pass_loop_done          // Optimiser
   pass_del_ssa               // Optimiser
pass_warn_function_return     // Optimiser
pass_expand                   // RTL Expander
pass_rest_of_compilation      // RTL passes
```


Part 2

Adding a Pass on Gimple IR

GCC Tree Passes: Code organisation

Tree Pass Organisation

- Data structure records pass info: name, function to execute etc. (`struct tree_opt_pass` in `tree-pass.h`)
- Instantiate a `struct tree_opt_pass` variable in each pass file.
- List the pass variables (in `init_tree_optimization_passes.c`).

Adding a Pass on Gimple IR

- Step 0. Write function `cs715_main()` in file `cs715.c`.
- Step 1. Create the following data structure in file `cs715.c`.

```
struct tree_opt_pass pass_cs715 =
{ "cs715",      /* name */
  NULL,         /* gate, for conditional entry to this pass */
  cs715_main,   /* execute, main entry point */
  NULL,         /* sub-passes, depending on the gate predicate */
  NULL,         /* next sub-passes, independ of the gate predicate */
  0,            /* static_pass_number , used for dump file name*/
  0,            /* tv_id */
  0,            /* properties_required, indicated by bit position */
  0,            /* properties_provided , indicated by bit position*/
  0,            /* properties_destroyed , indicated by bit position*/
  0,            /* todo_flags_start */
  0,            /* todo_flags_finish */
  0             /* letter for RTL dump */
};
```


Adding a Pass on Gimple IR

- Step 2. Add the following line to `tree-pass.h`

```
extern tree_opt_pass pass_cs715;
```


Adding a Pass on Gimple IR

- Step 2. Add the following line to `tree-pass.h`
  ```
  extern tree_opt_pass pass_cs715;
  ```
- Step 3. Include the following call at an appropriate place in the function `init_tree_optimization_passes()` in the file `tree-optimize.c`
  ```
  NEXT_PASS (pass_cs715);
  ```

Adding a Pass on Gimple IR

- Step 2. Add the following line to `tree-pass.h`
  `extern tree_opt_pass pass_cs715;`
- Step 3. Include the following call at an appropriate place in the function `init_tree_optimization_passes()` in the file `tree-optimize.c`
  `NEXT_PASS (pass_cs715);`
- Step 4. Add the file name in the Makefile
  - Either in `$SOURCE/gcc/Makefile.in`
    Reconfigure and remake
  - Or in `$BUILD/gcc/Makefile`
    Remake

Adding a Pass on Gimple IR

- Step 2. Add the following line to `tree-pass.h`
  `extern tree_opt_pass pass_cs715;`
- Step 3. Include the following call at an appropriate place in the function `init_tree_optimization_passes()` in the file `tree-optimize.c`
  `NEXT_PASS (pass_cs715);`
- Step 4. Add the file name in the Makefile
  - Either in `$SOURCE/gcc/Makefile.in`
    Reconfigure and remake
  - Or in `$BUILD/gcc/Makefile`
    Remake
- Step 5. Build the compiler

Adding a Pass on Gimple IR

- Step 2. Add the following line to `tree-pass.h`
  ```
  extern tree_opt_pass pass_cs715;
  ```
- Step 3. Include the following call at an appropriate place in the function `init_tree_optimization_passes()` in the file `tree-optimize.c`
  ```
  NEXT_PASS (pass_cs715);
  ```
- Step 4. Add the file name in the Makefile
  - Either in `$SOURCE/gcc/Makefile.in`
    Reconfigure and remake
  - Or in `$BUILD/gcc/Makefile`
    Remake
- Step 5. Build the compiler
- Step 6. Wonder what went wrong?

Traversing Control Flow Graph

```
for(n=0; n < number_of_nodes; n++)  {
      bb = VARRAY_BB(dfs_order_bb,n);
      for( bsi =bsi_start(bb);!bsi_end_p(bsi);bsi_next(&bsi)) {









      }
}
```


Traversing Control Flow Graph

```
for(n=0; n < number_of_nodes; n++)  {
      bb = VARRAY_BB(dfs_order_bb,n);
      for( bsi =bsi_start(bb);!bsi_end_p(bsi);bsi_next(&bsi)) {
            stmt = bsi_stmt(bsi);
            switch(TREE_CODE(stmt)) {









            }
      }
}
```


Traversing Control Flow Graph

```
for(n=0; n < number_of_nodes; n++)  {
      bb = VARRAY_BB(dfs_order_bb,n);
      for( bsi =bsi_start(bb);!bsi_end_p(bsi);bsi_next(&bsi)) {
            stmt = bsi_stmt(bsi);
            switch(TREE_CODE(stmt)) {
                  case MODIFY_EXPR:









            }
      }
}
```


Traversing Control Flow Graph

```
for(n=0; n < number_of_nodes; n++)  {
      bb = VARRAY_BB(dfs_order_bb,n);
      for( bsi =bsi_start(bb);!bsi_end_p(bsi);bsi_next(&bsi)) {
            stmt = bsi_stmt(bsi);
            switch(TREE_CODE(stmt)) {
                  case MODIFY_EXPR:
                        expr = TREE_OPERAND(stmt,1);
                        /* SET_BIT(GEN(current_pf_L,bb),expr); */
                        lval = TREE_OPERAND(stmt,0);
                        FOR expr_no in exprList(lval) {
                          /*  RESET_BIT(av_L[bb]->gen),expr_no); */
                          /*  SET_BIT(av_L[bb]->kill),expr_no);  */
                        }
                        break;
                  ...
            }
      }
}
```