Setting up the DSPCAD Lightweight Dataflow Environment: Setup Guide for LIDE Version 0.2*

Yang Jiao, Kishan Sudusinghe, and Shuvra S. Bhattacharyya
Department of Electrical and Computer Engineering, and
Institute for Advanced Computer Studies
University of Maryland at College Park, USA
{yjiao1, kishans, ssb}@umd.edu

April 5, 2015

This document provides instructions on setting up LIDE, the DSPCAD Lightweight Dataflow Environment. The setup instructions provided here complement the resources provided on the LIDE Project Website [1], and the LIDE overview report [2]. The overview report provides general background on LIDE, and is therefore useful as a preliminary orientation to the LIDE package. However, it should be noted that some features and command names have changed since the publication of the overview report; for the most up-to-date information, one should consult other documentation available from the LIDE Project Website [1].

1 Setup Instructions

This section provides information on installing LIDE. The following steps outline the installation process.

1. Download the LIDE package from [1], and unpack the archived download file `lide.tar.gz` (this will result in a singled directory called `lide`). Place this `lide` directory in the directory location where you want it to reside. This location is referred to in the remainder of this document as your *LIDE installation directory*. For example, if one has placed the downloaded, unarchived `lide` directory in `/users/me/import/applications`, then the LIDE installation directory is:

   ```
   /users/me/import/applications/lide
   ```

2. Go to the parent directory of your `lide_user` directory (or the directory in which you want `lide_user` to be created if it does not yet exist). The `lide_user` directory is a directory in which user-specific files related to LIDE are maintained. For further background about the `lide_user` directory, see the LIDE overview report [2].

3. Run the following commands (replacing the right hand side of the first assignment statement appropriately for your LIDE installation directory).

   ```
   UXLIDE="/users/me/import/applications/lide"
   source "$UXLIDE"/setup/runme
   ```

*Technical Report DSPCAD-TR-2015-01, Maryland DSPCAD Research Group, Department of Electrical and Computer Engineering and Institute for Advanced Computer Studies, University of Maryland at College Park. April 05, 2015.

4. If you want to run the two commands above through a script, you can use the following as the contents of your script (again with the appropriate directory path replacement).

```
#!/usr/bin/env bash

UXLIDE="/users/me/import/applications/lide"
source "$UXLIDE"/setup/runme
```

2 Startup Instructions

To use LIDE in a given `Bash` session, one must first start up LIDE within that session. Furthermore, since LIDE depends on the DICE package, DICE must be started up before starting up LIDE. For instructions on setting up and starting up DICE, see [3], and for further background on the DICE package, see [4].

Starting up LIDE involves loading necessary environment settings so that you can use all of the features in LIDE.

To start up LIDE, follow these steps:

1. Start a `Bash` shell.
2. Start up DICE (see [3]) within this `Bash` shell.
3. `cd` to your LIDE user directory (e.g., run `cd ~/lide_user`).
4. Run

```
source startup/lide_startup
```

IMPORTANT: The `lide_startup` file must be invoked from the `lide_user` directory — e.g., as opposed to running it as

```
source lide_startup
```

from `~/lide_user/startup` or running it from your home directory.

As a basic test of the startup process one can run the `lideversion` command, which takes no arguments, from the enclosing `Bash` session after LIDE has been started up. If LIDE has been properly set up and started up, the `lideversion` command should execute and produce a brief message on standard output that gives the version number and other basic background information for the corresponding installation of LIDE.

3 Startup Script

The Bash script shown in Listing 1 provides a way to start up (or "load") both DICE and LIDE conveniently just after starting a new Bash session. The comments in Listing 1 provide more details on how to use the script. Note that here one can use "." as a convenient shorthand for invoking the "source" command.

```
#!/usr/bin/env bash

#
# This script can be placed in your home directory ... e.g., as
# bashrc-local.
#
# Then you can load DICE and LIDE upon starting the shell by running:
#
# source bashrc-local
#
# NOTE: if your lide_user and dice_user directories are not contained
# directly in your home directory, then you need to modify the arguments
# to the two pushd commands below as needed to ensure that they find the
# locations of your lide_user and dice_user directories.
#

# LIDE depends on DICE, so start up DICE first
pushd ~/dice_user
source startup/dice_startup
popd

# Now start up LIDE
pushd ~/lide_user
source startup/lide_startup
popd
```

Listing 1: A Bash script for starting up DICE and LIDE.

4 A First Example

To retrieve a simple example of a LIDE project, run the following command from any directory.

```
lideget basicdemo
```

This example, stored in a directory called `basic_demo`, consists of three tests for an inner product actor. This actor computes inner products of vectors that arrive on its inputs, and produces the inner product results on its output. The tests are developed using test organizational conventions of the DICE package [4], and the test suite can be executed by running the DICE `dxtest` command. The three tests are stored in the subdirectories `test_demo1`, `test_demo2`, and `test_demo3`. Each test exercises the inner product actor with a different data set. Code for the inner product actor can be examined by following the guidelines in Section 5.

A simple LIDE-based dataflow graph construction is used to carry out the execution of the actor on specific data sets, and to collect the output results. Each data set (for a single execution of this dataflow graph) consists of three input files and results in one output file. The dataflow graph used for testing the inner product actor in this way is defined in the subdirectory `basic_demo/util`. This subdirectory contains a file called `lide_c_inner_product_driver.c`, which provides a simple of example that demonstrates: (1) how a LIDE-based dataflow graph can be constructed and executed; and (2) an actor-level testing approach, using "file source" and "file sink" actors, that uses a dataflow graph construction to test an individual actor (in this example, the dataflow graph is constructed to provide a test structure for the inner product actor). Code for the file source and file sink actors employed in this dataflow graph can also be examined by following the guidelines in Section 5.

To compile and run the `basic_demo` example, use the following steps.

```
cd basic_demo/util
./makeme
cd ..
dxtest
```

This will execute all three of the tests together using `dxtest` and report a summary of the results from this testing. See [4] for more information about the DICE `dxtest` command.

The following steps show an example of how one can execute just one of the tests by itself.

```
cd basic_demo/test_demo1
dxtest
```

5 Examples of LIDE Actors

To retrieve a copy of the actors used in this first example, along with several other relatively simple examples of dataflow actors, run the following command from any directory.

```
lideget basicgems
```

Here *gems*, a standard abbreviation in LIDE terminology, stands for "Graph EleMentS."

The command above will produce a directory called `basic` in your current working directory (CWD). You may then browse the actor definitions in the `basic` directory within your CWD without risk of accidental modifications to the installed versions of these gems. It may also be useful to consider these actors as potential samples, starting points or templates when designing your own actors.

6 Updating LIDE

To update your installation of LIDE to a new version (without resetting your `lide-user` directory), use the following steps.

1. Start up LIDE (i.e., your currently-installed version of LIDE).
2. Go (`cd`) to the directory that contains the new version of LIDE.
3. Make sure that the newly downloaded version has been unarchived (extracted) — i.e., it should appear as the *subdirectory* `lide` (not as the *file* `lide.tar` or `lide.tar.gz`).
4. Run the following command:

   ```
   lideupdate
   ```

5. This will install the new version of LIDE, and save your previous version of LIDE in `"$UXTMP"/lide-bak.tar.gz`.
6. Exit all open LIDE sessions and their enclosing Bash sessions.
7. Build the new version of LIDE unless instructions with the new release have been provided indicating that a build is not needed for this update. If in doubt, build the new version of LIDE. To build the new version of LIDE, run the following command:

   ```
   lidebuild
   ```

After working through these steps successfully, one can use the new version of LIDE by starting up LIDE in the usual way. For example, one can start by running the `lideversion` command (see Section 2) to check the version number or revision date of the installed version.

Acknowledgments

This work was sponsored in part by the Laboratory for Physical Science, Laboratory for Telecommunication Sciences, and US National Science Foundation.

We are grateful also to the following people who have made valuable contributions to LIDE, and earlier software components that have evolved into parts of LIDE: Bishnupriya Bhattacharya, Nitin Chandrachoodan, Chung-Ching Shen, Soujanya Kedilaya, and Robert Ricketts.

References

[1] "LIDE project website," `http://www.ece.umd.edu/DSPCAD/projects/lide/lide.htm`.

[2] C. Shen, L. Wang, I. Cho, S. Kim, S. Won, W. Plishker, and S. S. Bhattacharyya, "The DSPCAD lightweight dataflow environment: Introduction to LIDE version 0.1," Tech. Rep. UMIACS-TR-2011-17, Institute for Advanced Computer Studies, University of Maryland at College Park, 2011, http://hdl.handle.net/1903/12147.

[3] S. S. Bhattacharyya, W. Plishker, N. Sane, K. Sudusinghe, and G. Zaki, "Setting up the DSPCAD integrative command-line environment: Setup guide for DICE version 1.2," Tech. Rep. DSPCAD-TR-2014-01, Maryland DSPCAD Research Group, Department of Electrical and Computer Engineering, University of Maryland at College Park, May 2014.

[4] S. S. Bhattacharyya, W. Plishker, C. Shen, N. Sane, and G. Zaki, "The DSPCAD integrative command line environment: Introduction to DICE version 1.1," Tech. Rep. UMIACS-TR-2011-10, Institute for Advanced Computer Studies, University of Maryland at College Park, 2011, http://drum.lib.umd.edu/handle/1903/11422.