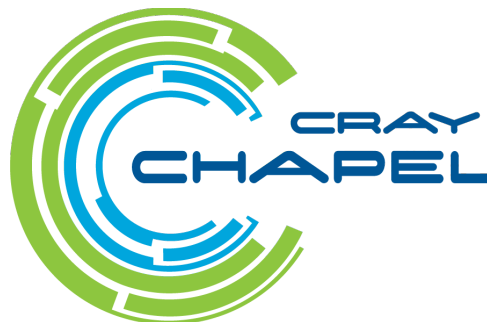




# Hands-On 0: Hello World





# Safe Harbor Statement

This presentation may contain forward-looking statements that are based on our current expectations. Forward looking statements may include statements about our financial guidance and expected operating results, our opportunities and future potential, our product development and new product introduction plans, our ability to expand and penetrate our addressable markets and other statements that are not historical facts. These statements are only predictions and actual results may materially vary from those projected. Please refer to Cray's documents filed with the SEC from time to time concerning factors that could affect the Company and these forward-looking statements.





# Steps to “Hello World”

- **Download and build Chapel**
  - or get a Cray account from us and use a pre-installed version
  - but most find it nice to get started on a familiar machine...
  - see <http://chapel.cray.com/tmp/SC16/> for pointers and resources
  - swan: /tmp/chapel-SC16/ has the example codes etc.
- **Create hello.chpl with your favorite editor:**
  - `writeln("Hello World");`
- **Compile it**
  - `chpl -o hello hello.chpl`
- **Run it**
  - `./hello`



# "Hello World" in Chapel: Two Versions

- Fast prototyping

```
writeln("Hello, world!");
```

- "Production-grade"

```
module Hello {

    proc main() {
        writeln("Hello, world!");
    }
}
```

# "Hello World" in Chapel: Two Versions

- Fast prototyping (configurable)

```
config const audience = "world";
writeln("Hello, ", audience, "!");
```

- “Production-grade” (configurable)

```
module Hello {
  config const audience = "world";

  proc main() {
    writeln("Hello, ", audience, "!");
  }
}
```

- To change ‘audience’ for a given run:

- ./hello --audience=SC16



# Check out other “Hello world” versions

- **Take a look at `$CHPL_HOME/examples/hello*.chpl`**
  - Six variations that show off different language features
  - Also rendered online at: <http://chapel.cray.com/docs/latest/examples/>
- **Do whatever else you can / want before lunch!**
- **After lunch:**
  - Task parallelism
  - Locality
  - Distributed Data Parallelism
  - More hands-on
  - etc.

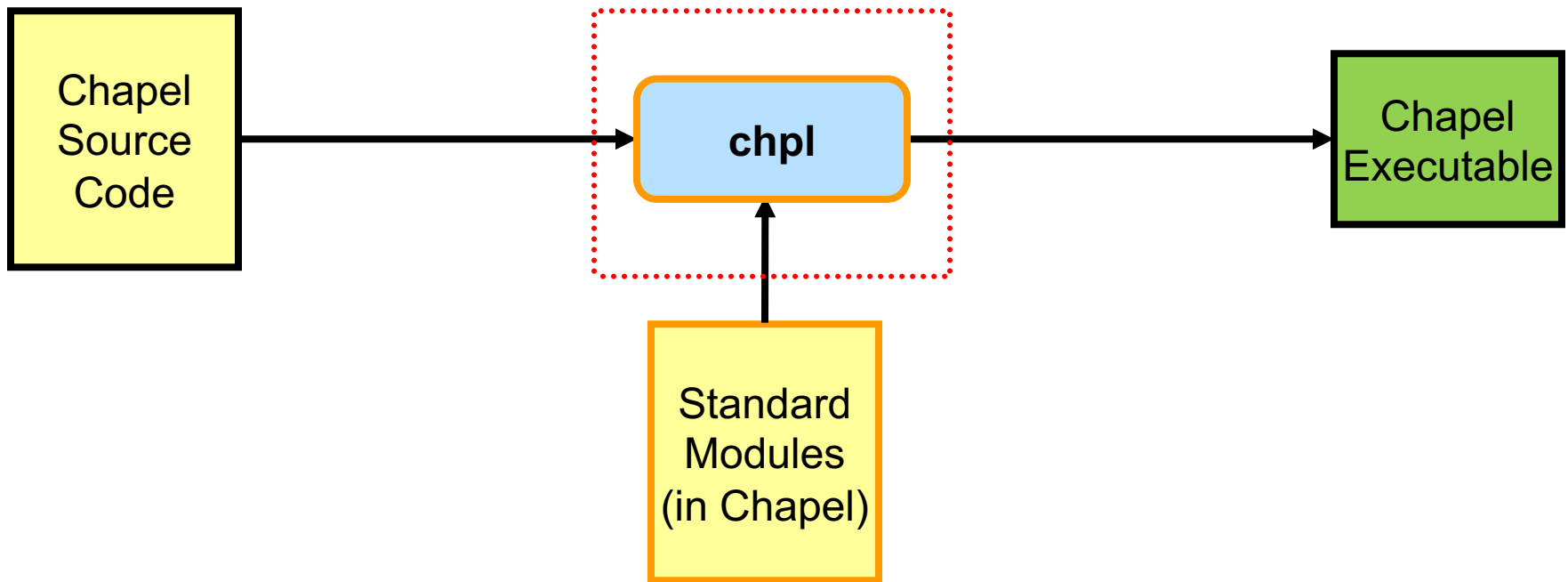




# Optional Compiler Approach Sidebar

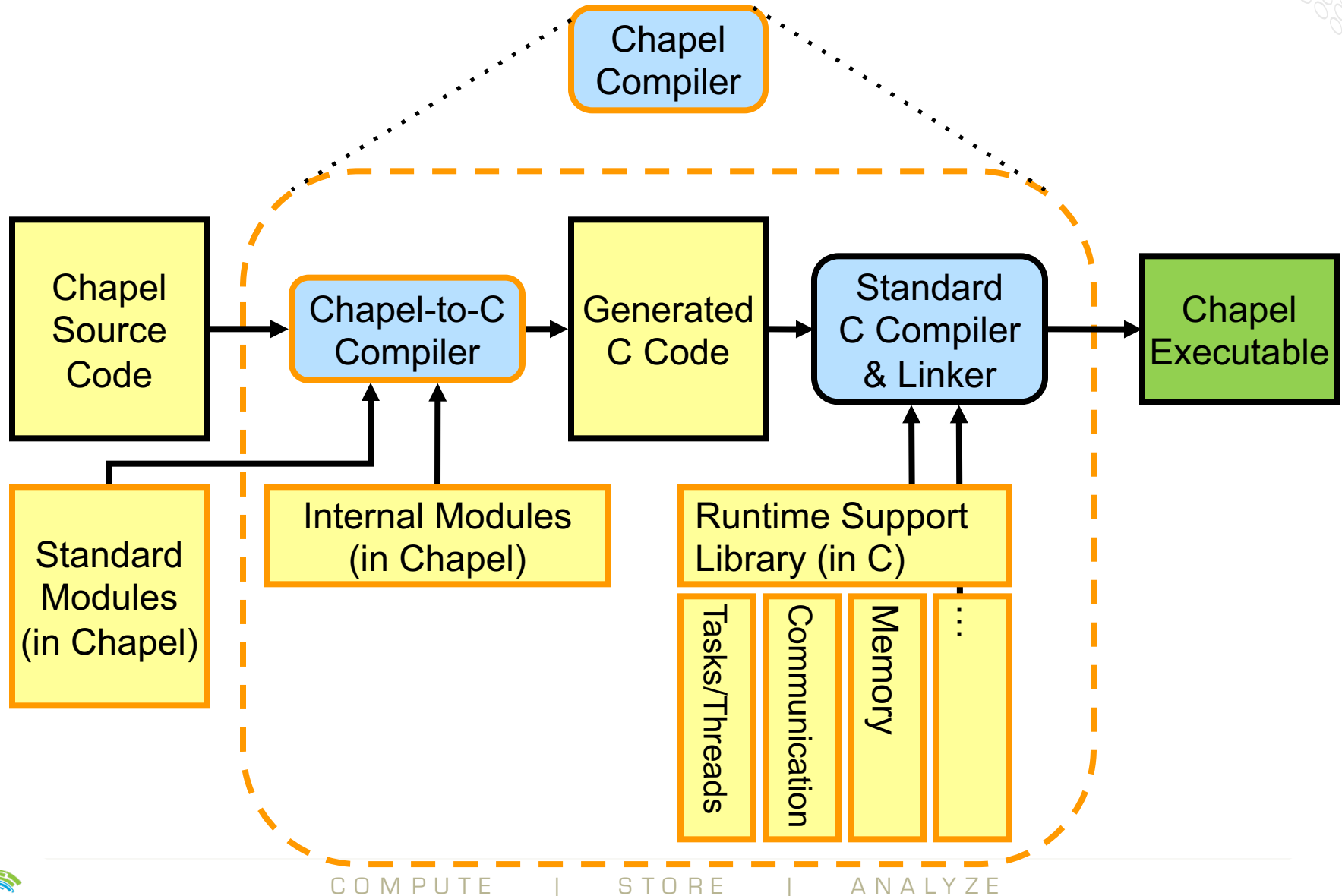


# Compiling Chapel





# Chapel Compiler Architecture





# LLVM back-end

- Chapel typically code-generates C
- Targeting LLVM is also an option
  - Not on by default due to build-time required
  - Most of our team is not ramped up on LLVM's IR either...
- To enable:
  - set `CHPL_LLVM=llvm` and re-'make'
  - compile with '`chpl --llvm`' (or set `CHPL_LLVM_CODEGEN`)
- Enabling LLVM also supports an interoperability capability

```
extern {  
    void myCFunc() { // myCFunc() can be called from Chapel  
        printf("Hello from C!");  
    }  
    #include "gsl.h" // as can functions defined in gsl.h...  
}
```





# Legal Disclaimer

*Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.*

*Cray Inc. may make changes to specifications and product descriptions at any time, without notice.*

*All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.*

*Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.*

*Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.*

*Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.*

*The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, and URIKA. The following are trademarks of Cray Inc.: ACE, APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.*

