Workshop on Essential Abstractions in GCC

An Overview of Compilation and GCC

GCC Resource Center
(www.cse.iitb.ac.in/grc)

Department of Computer Science and Engineering,
Indian Institute of Technology, Bombay

30 June 2012
Outline

- Introduction to Compilation
- An Overview of Compilation Phases
- An Overview of GCC
Part 1

Introduction to Compilation
Nothing is known except the problem
Binding

Overall strategy, algorithm, data structures etc.

No. of unbound objects

Conceptualisation

Time
Binding

- Functions, variables, their types etc.

- No. of unbound objects

- Conceptualisation

- Coding

- Time

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
**Binding**

No. of unbound objects

Machine instructions, registers etc.

Conceptualisation  Coding  Compiling  

--- Time ---
Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Binding

- Conceptualisation
- Coding
- Compiling
- Linking
- Loading

No. of unbound objects

Time

Actual addresses of code and data

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Binding

Values of variables

No. of unbound objects

Conceptualisation  Coding  Compiling  Linking  Loading  Execution

Time

Essential Abstractions in GCC
We will look at different binding times related to compiling.
Implementation Mechanisms

Source Program

Translator

Target Program

Machine
Implementation Mechanisms

Source Program
  ↓
  Translator
  ↓
Target Program
  ↓
  Machine

Input Data
Implementation Mechanisms

Source Program

Translator

Target Program

Machine

Input Data

Source Program

Interpreter

Machine

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Implementation Mechanisms as “Bridges”

• “Gap” between the “levels” of program specification and execution

Program Specification

Machine
Implementation Mechanisms as “Bridges”

- “Gap” between the “levels” of program specification and execution

```
<table>
<thead>
<tr>
<th>Program Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Translation</td>
</tr>
<tr>
<td>Machine</td>
</tr>
</tbody>
</table>
```
Implementation Mechanisms as “Bridges”

- “Gap” between the “levels” of program specification and execution

Program Specification

\[\text{Translation} \Downarrow \quad \text{Interpretation} \Uparrow\]

Machine
Implementation Mechanisms as “Bridges”

- “Gap” between the “levels” of program specification and execution

Program Specification

<table>
<thead>
<tr>
<th>Translation</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>State: Variables</td>
<td></td>
</tr>
<tr>
<td>Operations: Expressions, Control Flow</td>
<td></td>
</tr>
</tbody>
</table>

Machine

<table>
<thead>
<tr>
<th>State: Memory, Registers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operations: Machine Instructions</td>
</tr>
</tbody>
</table>
Input C statement

```c
a = b<10?b:c;
```

Spim Assembly Equivalent

```assembly
lw  $t0, 4($fp) ; t0 <- b  # Is b smaller
slti $t0, $t0, 10 ; t0 <- t0 < 10  # than 10?
not $t0, $t0  ; t0 <- !t0
bgtz $t0, L0:  ; if t0>0 goto L0
lw  $t0, 4($fp) ; t0 <- b  # YES
b   L1:  ; goto L1
L0: lw $t0, 8($fp) ;L0: t0 <- c  # NO
L1: sw 0($fp), $t0 ;L1: a <- t0
```
High and Low Level Abstractions

Input C statement
\[ a = b < 10 ? b : c; \]

Spim Assembly Equivalent

```
lw $t0, 4($fp) ; t0 <- b  # Is b smaller
slti $t0, $t0, 10 ; t0 <- t0 < 10  # than 10?
not $t0, $t0 ; t0 <- !t0
bgtz $t0, L0: ; if t0>0 goto L0
lw $t0, 4($fp) ; t0 <- b  # YES
b L1: ; goto L1
L0: lw $t0, 8($fp) ;L0: t0 <- c  # NO
L1: sw 0($fp), $t0 ;L1: a <- t0
```
High and Low Level Abstractions

Input C statement

\[ a = b<10?b:c; \]

Spim Assembly Equivalent

\[
\begin{align*}
\text{lwan} & \ $t0, 4($fp) ; & t0 & \leftarrow b & \# \text{Is b smaller} \\
\text{slti} & \ $t0, $t0, 10 ; & t0 & \leftarrow t0 < 10 & \# \text{than 10?} \\
\text{not} & \ $t0, $t0 ; & t0 & \leftarrow \neg t0 \\
\text{bgtz} & \ $t0, L0: ; & \text{if } t0 > 0 \text{ goto L0} \\
\text{lwan} & \ $t0, 4($fp) ; & t0 & \leftarrow b & \# \text{YES} \\
\text{b} & \ L1: ; & \text{goto L1} \\
\text{L0:} & \text{lwan} & \ $t0, 8($fp) ;L0: & t0 & \leftarrow c & \# \text{NO} \\
\text{L1:} & \text{sw} & \ 0($fp), $t0 ;L1: & a & \leftarrow t0
\end{align*}
\]
High and Low Level Abstractions

Input C statement

```c
a = b<10?b:c;
```

Spim Assembly Equivalent

```assembly
lw   $t0, 4($fp) ;  t0 <- b  # Is b smaller
slti $t0, $t0, 10 ; t0 <- t0 < 10  # than 10?
nor  $t0, $t0  ;  t0 <- !t0
bgtz $t0, L0: ; if t0>0 goto L0
lw  $t0, 4($fp) ;  t0 <- b  # YES
     b   L1: ;  goto L1
L0: lw $t0, 8($fp) ;L0: t0 <- c  # NO
L1: sw 0($fp), $t0 ;L1: a <- t0
```

NOT Condition

Conditional jump

Fall through

True Part

False Part

NOT Condition

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Implementation Mechanisms

- Translation  =  Analysis + Synthesis
- Interpretation  =  Analysis + Execution
Implementation Mechanisms

- Translation = Analysis + Synthesis
- Interpretation = Analysis + Execution

- Translation Instructions $\Rightarrow$ Equivalent Instructions
Implementation Mechanisms

- Translation = Analysis + Synthesis
Interpretation = Analysis + Execution

- Translation Instructions $\Rightarrow$ Equivalent Instructions
Interpretation Instructions $\Rightarrow$ Actions Implied by Instructions
Language Implementation Models

- Analysis
- Synthesis
- Compilation
- Execution
- Interpretation
Language Processor Models

- Front End
- Optimizer
- Back End
- Virtual Machine

- C, C++
- Java, C#
Part 2

An Overview of Compilation Phases
The Structure of a Simple Compiler

- **Parser**
- **Scanner**
- **Semantic Analyser**
- **Symtab Handler**

Source Program
The Structure of a Simple Compiler

- **Parser**
- **AST**
- **Instruction Selector**
- **Insn**
- **Assembly Emitter**

Dependencies:
- **Scanner**
- **Semantic Analyser**
- **Syntab Handler**
- **Register Allocator**

Input:
- **Source Program**

Output:
- **Assembly Program**
The Structure of a Simple Compiler

Front End
Parser
  \[\rightarrow\] AST
  \[\rightarrow\] Instruction Selector
  \[\rightarrow\] Register Allocator
  \[\rightarrow\] Assembly Emitter
  \[\rightarrow\] Assembly Program

Back End
Parser
  \[\rightarrow\] AST
  \[\rightarrow\] Instruction Selector
  \[\rightarrow\] Register Allocator
  \[\rightarrow\] Assembly Emitter
  \[\rightarrow\] Assembly Program

Scanner
Semantic Analyser
Symtab Handler

Source Program
Translation Sequence in Our Compiler: Parsing

Input

```c
a=b<10?b:c;
```
Translation Sequence in Our Compiler: Parsing

Input: 
```
a = b < 10 ? b : c;
```

Parse Tree:
```
AsgnStmtn
  /
 /  
Lhs = E ;
   /
  / 
name E ? E : E
  /
 name E < E name name
    /
     name num
```

Issues:
- Grammar rules, terminals, non-terminals
- Order of application of grammar rules
  eg. is it (a = b < 10?) followed by (b : c)?
- Values of terminal symbols
  eg. string “10” vs. integer number 10.
Translation Sequence in Our Compiler: Semantic Analysis

Input

a=b<10?b:c;

Parse Tree

AsgnStmtnt

Lhs = E ;

name E ? E : E

E < E name name

name num
Translation Sequence in Our Compiler: Semantic Analysis

\[ a = b < 10 ? b : c; \]

**Input**

**AsgnStmnt**

\[
\text{Lhs} = \text{E} ;
\]

\[
\begin{align*}
\text{name} & \quad \text{E} \quad ? \quad \text{E} \quad : \quad \text{E} \\
\text{E} & \quad < \quad \text{E} \quad \text{name} \quad \text{name}
\end{align*}
\]

**Parse Tree**

**Abstract Syntax Tree (with attributes)**

**Issues:**

- **Symbol tables**
  Have variables been declared? What are their types?
  What is their scope?

- **Type consistency of operators and operands**
  The result of computing \( b < 10 ? \) is bool and not int
Translation Sequence in Our Compiler: IR Generation

```plaintext
a = b < 10 ? b : c;
```

**Parse Tree**

**Abstract Syntax Tree (with attributes)**

```
Lhs := E ;
E ? E : E
E < E name name
name num

name (a, int)
?: (int)
name (b, int)
name (c, int)
name (b, int)
num (10, int)
```

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Translation Sequence in Our Compiler: IR Generation

\[ a = b < 10 ? b : c; \]

**Input**

**Tree List**

\[ T_0 \]

\[ \begin{array}{c}
T_0 = b \\
\text{IfGoto} \\
\text{Not} \\
\text{L0:} \\
T_0 = b \\
\text{Goto} \\
\text{L1:} \\
\text{L0:} \quad = \\
T_1 = c \\
\text{L1:} \quad = \\
a = T_1 \\
\end{array} \]

**Parse Tree**

**Abstract Syntax Tree (with attributes)**

Issues:

- Convert to maximal trees which can be implemented without altering control flow
  Simplifies instruction selection and scheduling, register allocation etc.

- Linearise control flow by flattening nested control constructs
Translation Sequence in Our Compiler: Instruction Selection

\[ a = b < 10 \text{?} b : c; \]

Input:

Tree List:

- \( T_0 \): 
  \( = \)
  \( b \rightarrow 10 \)
  \( \text{IfGoto} \)
  \( \text{Not} \)
  \( \text{L0:} \)

- \( T_0 \): 
  \( = \)
  \( b \rightarrow \)
  \( \text{Goto} \)
  \( \text{L1:} \)

- \( T_1 \): 
  \( = \)
  \( c \rightarrow \)
  \( \text{L0:} \)

- \( L1: \)
  \( = \)
  \( a \rightarrow T_1 \)

AsgnStmt

\[ \text{Lhs} \rightarrow E \]

\[ \text{name} \rightarrow E \]

\[ \text{E} \rightarrow \text{E} < \text{E} \]

\[ \text{name} \rightarrow \text{name} \]

\[ \text{name} \rightarrow \text{num} \]

\[ \text{name} \rightarrow \text{name} \]

\[ \text{name} \rightarrow \text{num} \]

\[ \text{name} \rightarrow \text{name} \]

\[ \text{name} \rightarrow \text{name} \]

\[ \text{name} \rightarrow \text{name} \]

\[ \text{name} \rightarrow \text{name} \]

\[ \text{parse tree} \]

\[ \text{Abstract Syntax Tree (with attributes)} \]

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Translation Sequence in Our Compiler: Instruction Selection

a = b < 10 ? b : c;

Input

Tree List

Parse Tree

Abstract Syntax Tree (with attributes)

Issues:

- Cover trees with as few machine instructions as possible
- Use temporaries and local registers
Translation Sequence in Our Compiler: Emitting Instructions

a=b<10?b:c;

Input

Parse Tree

Abstract Syntax Tree (with attributes)

Instruction List

T₀ ← b
T₀ ← T₀ < 10
T₀ ← ! T₀
if T₀ > 0 goto L0:
T₁ ← b
goto L1:
L0: T₁ ← c
L1: a ← T₁

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Translation Sequence in Our Compiler: Emitting Instructions

Input:

```
a = b < 10 ? b : c;
```

Tree List:

```
\[ T_0 = \begin{array}{c}
  \text{IfGoto} \\
  \text{Not} \\
  \text{L0:} \\
\end{array} \]
```

```
\[ T_0 = b < 10 \]
```

```
\[ \text{Goto} \]
```

```
\[ \text{L1:} \]
```

```
\[ T_1 = c \]
```

```
\[ \text{L1:} \]
```

```
\[ a = T_1 \]
```

Issues:

- Offsets of variables in the stack frame
- Actual register numbers and assembly mnemonics
- Code to construct and discard activation records

Abstract Syntax Tree (with attributes):

```
\[ \begin{array}{c}
  \text{name} \\
  \text{(a, int)} \\
  \text{?:} \\
  \text{(int)} \\
  \text{<} \\
  \text{name} \\
  \text{(b, int)} \\
  \text{name} \\
  \text{(c, int)} \\
  \text{name} \\
  \text{(b, int)} \\
  \text{num} \\
  \text{(10, int)}
\end{array} \]
```

Instruction List:

```
\[ T_0 \leftarrow b \]
\[ T_0 \leftarrow T_0 < 10 \]
\[ T_0 \leftarrow \neg T_0 \]
if $T_0 > 0$ goto L0:
\[ T_1 \leftarrow b \]
goto L1:
\[ \text{L0:} \]
\[ T_1 \leftarrow c \]
\[ \text{L1:} \]
\[ a \leftarrow T_1 \]
```

Assembly Code:

```
lw $t0, 4($fp)
slti $t0, $t0, 10
not $t0, $t0
bgtz $t0, L0:
lw $t0, 4($fp)
b L1:
lw $t0, 8($fp)
sw 0($fp), $t0
```

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Part 3

Compilation Models
Compilation Models

Aho Ullman Model

Davidson Fraser Model
Compilation Models

Aho Ullman Model

Front End

Input Source Program

AST

Davidson Fraser Model
Compilation Models

Aho Ullman Model

Input Source Program

Front End

AST

Optimizer

Target Indep. IR

Davidson Fraser Model

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Compilation Models

Aho Ullman Model

Input Source Program

- Front End
- AST
- Optimizer
- Target Indep. IR
- Code Generator
- Target Program

Davidson Fraser Model

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Compilation Models

Aho Ullman Model

Front End

AST

Optimizer

Target Indep. IR

Code Generator

Target Program

Davidson Fraser Model

Input Source Program

Front End

AST

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Compilation Models

Aho Ullman Model

Front End

AST

Optimizer

Target Indep. IR

Code Generator

Target Program

Davidson Fraser Model

Input Source Program

Front End

AST

Expander

Register Transfers

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Compilation Models

Aho Ullman Model

Front End → AST → Optimizer → Target Indep. IR → Code Generator → Target Program

Davidson Fraser Model

Input Source Program → Front End → AST → Expander → Register Transfers → Optimizer → Register Transfers

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Compilation Models

**Aho Ullman Model**
- Front End
- AST
- Optimizer
- Target Indep. IR
- Code Generator

**Davidson Fraser Model**
- Input Source Program
- Front End
- AST
- Expander
- Register Transfers
- Optimizer
- Register Transfers
- Recognizer
- Target Program

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Aho Ullman: Instruction selection

- over optimized IR using
- cost based tree tiling matching

Davidson Fraser: Instruction selection

- over AST using
- simple full tree matching based algorithms that generate
- naive code which is
  - target dependent, and is
  - optimized subsequently
Typical Front Ends

Parser
Typical Front Ends

Source Program → Tokens → Scanner → Parser
Typical Front Ends

- Source Program
- Scanner
- Tokens
- Parse Tree
- AST
- Semantic Analyzer
- AST or Linear IR + Symbol Table

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Typical Front Ends

Source Program → Scanner → Tokens → Parse Tree → AST or Linear IR + Symbol Table → Semantic Analyzer

Parser

AST

Symtab Handler

Error Handler

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Typical Back Ends in Aho Ullman Model

- Compile time evaluations
- Eliminating redundant computations
Typical Back Ends in Aho Ullman Model

- Compile time evaluations
- Eliminating redundant computations
- Instruction Selection
- Local Reg Allocation
- Choice of Order of Evaluation

m/c Ind. → m/c Ind. → m/c Ind. → Code Generator → m/c Dep.

IR          IR          IR

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Typical Back Ends in Aho Ullman Model

- Compile time evaluations
- Eliminating redundant computations

- Instruction Selection
- Local Reg Allocation
- Choice of Order of Evaluation

Assembly Code
Typical Back Ends in Aho Ullman Model

- Compile time evaluations
- Eliminating redundant computations
- Instruction Selection
- Local Reg Allocation
- Choice of Order of Evaluation
### Retargetability in Aho Ullman and Davidson Fraser Models

<table>
<thead>
<tr>
<th></th>
<th>Aho Ullman Model</th>
<th>Davidson Fraser Model</th>
</tr>
</thead>
</table>
| **Instruction Selection** | • Machine independent IR is expressed in the form of trees  
                          • Machine instructions are described in the form of trees  
                          • Trees in the IR are “covered” using the instruction trees |                                |
| **Optimization**     |                                                                                 |                                |
# Retargetability in Aho Ullman and Davidson Fraser Models

<table>
<thead>
<tr>
<th></th>
<th>Aho Ullman Model</th>
<th>Davisdon Fraser Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instruction Selection</td>
<td>• Machine independent IR is expressed in the form of trees</td>
<td>• Machine instructions are described in the form of trees</td>
</tr>
<tr>
<td></td>
<td>• Machine instructions are described in the form of trees</td>
<td>• Trees in the IR are “covered” using the instruction trees</td>
</tr>
<tr>
<td>Optimization</td>
<td>Cost based tree pattern matching</td>
<td></td>
</tr>
</tbody>
</table>
# Retargetability in Aho Ullman and Davidson Fraser Models

<table>
<thead>
<tr>
<th></th>
<th>Aho Ullman Model</th>
<th>Davidson Fraser Model</th>
</tr>
</thead>
</table>
| **Instruction Selection** | • Machine independent IR is expressed in the form of trees  
• Machine instructions are described in the form of trees  
• Trees in the IR are “covered” using the instruction trees | Cost based tree pattern matching                                                              |
| **Optimization**     |                                                                                                           | Structural tree pattern matching                                                        |
### Retargetability in Aho Ullman and Davidson Fraser Models

<table>
<thead>
<tr>
<th></th>
<th>Aho Ullman Model</th>
<th>Davisdon Fraser Model</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Instruction Selection</strong></td>
<td>• Machine independent IR is expressed in the form of trees</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Machine instructions are described in the form of trees</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Trees in the IR are “covered” using the instruction trees</td>
<td></td>
</tr>
<tr>
<td><strong>Optimization</strong></td>
<td>Cost based tree pattern matching</td>
<td>Structural tree pattern matching</td>
</tr>
<tr>
<td></td>
<td>Machine independent</td>
<td></td>
</tr>
</tbody>
</table>

**Note:**
- Essential Abstractions in GCC
- GCC Resource Center, IIT Bombay
## Retargetability in Aho Ullman and Davidson Fraser Models

<table>
<thead>
<tr>
<th>Instruction Selection</th>
<th>Aho Ullman Model</th>
<th>Davidson Fraser Model</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Machine independent IR is expressed in the form of trees</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Machine instructions are described in the form of trees</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Trees in the IR are “covered” using the instruction trees</td>
<td></td>
</tr>
<tr>
<td>Cost based tree pattern matching</td>
<td>Structural tree pattern matching</td>
<td></td>
</tr>
<tr>
<td>Optimization</td>
<td>Machine independent</td>
<td>Machine dependent</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Retargetability in Aho Ullman and Davidson Fraser Models

<table>
<thead>
<tr>
<th>Instruction Selection</th>
<th>Aho Ullman Model</th>
<th>Davidson Fraser Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Machine independent IR is expressed in the form of trees</td>
<td>• Machine instructions are described in the form of trees</td>
<td>Cost based tree pattern matching</td>
</tr>
<tr>
<td>• Machine instructions are described in the form of trees</td>
<td>• Trees in the IR are “covered” using the instruction trees</td>
<td>Optimization</td>
</tr>
<tr>
<td>• Trees in the IR are “covered” using the instruction trees</td>
<td></td>
<td>Machine independent</td>
</tr>
</tbody>
</table>

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Part 4

\textit{GCC} \equiv \textit{The Great Compiler Challenge}
What is GCC?

• For the GCC developer community: The GNU Compiler Collection

• For other compiler writers: The Great Compiler Challenge 😊
The GNU Tool Chain for C

Source Program

\[ \text{gcc} \]

Target Program
The GNU Tool Chain for C

Source Program

\[ \text{gcc} \]

Target Program

\[ \text{cc1} \]
The GNU Tool Chain for C

Source Program

→

gcc

→

Target Program

→

cc1

←

cpp
The GNU Tool Chain for C

Source Program

gcc

Target Program

cc1

cpp

as
The GNU Tool Chain for C

Source Program

```
gcc
```

Target Program

```
cc1
cpp
as
ld
```
The GNU Tool Chain for C

Source Program

→

gcc

→

cc1 ← cpp

→

as

→

ld

→

glibc/newlib

Target Program
The GNU Tool Chain for C

Source Program

 gcc

 →

 Target Program

 cc1 → cpp

 as

 →

 ld

 →

 glibc/newlib
Why is Understanding GCC Difficult?

Some of the obvious reasons:

- **Comprehensiveness**
  
  GCC is a production quality framework in terms of completeness and practical usefulness

- **Open development model**
  
  Could lead to heterogeneity. Design flaws may be difficult to correct

- **Rapid versioning**
  
  GCC maintenance is a race against time. Disruptive corrections are difficult
Open Source and Free Software Development Model

The Cathedral and the Bazaar [Eric S Raymond, 1997]
Open Source and Free Software Development Model

The Cathedral and the Bazaar [Eric S Raymond, 1997]

- **Cathedral: Total Centralized Control**

  *Design, implement, test, release*
Open Source and Free Software Development Model

The Cathedral and the Bazaar [Eric S Raymond, 1997]

- Cathedral: Total Centralized Control
  
  Design, implement, test, release

- Bazaar: Total Decentralization
  
  Release early, release often, make users partners in software development
Open Source and Free Software Development Model

The Cathedral and the Bazaar [Eric S Raymond, 1997]

- **Cathedral**: Total Centralized Control
  
  *Design, implement, test, release*

- **Bazaar**: Total Decentralization
  
  *Release early, release often, make users partners in software development*

  “Given enough eyeballs, all bugs are shallow”
Open Source and Free Software Development Model

The Cathedral and the Bazaar [Eric S Raymond, 1997]

- **Cathedral: Total Centralized Control**
  
  Design, implement, test, release

- **Bazaar: Total Decentralization**
  
  Release early, release often, make users partners in software development

“Given enough eyeballs, all bugs are shallow”

Code errors, logical errors, and architectural errors
Open Source and Free Software Development Model

The Cathedral and the Bazaar [Eric S Raymond, 1997]

- **Cathedral**: Total Centralized Control
  
  Design, implement, test, release

- **Bazaar**: Total Decentralization
  
  Release early, release often, make users partners in software development

  “Given enough eyeballs, all bugs are shallow”

  Code errors, logical errors, and architectural errors

  **A combination of the two seems more sensible**
GCC follows a combination of the Cathedral and the Bazaar approaches

- GCC Steering Committee: Free Software Foundation has given charge
  - Major policy decisions
  - Handling Administrative and Political issues

- Release Managers:
  - Coordination of releases

- Maintainers:
  - Usually area/branch/module specific
  - Responsible for design and implementation
  - Take help of reviewers to evaluate submitted changes
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
Comprehensiveness of GCC: Wide Applicability

- Input languages supported:
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- Processors supported in standard releases:
  - Common processors:
  - Lesser-known target processors:
  - Additional processors independently supported:
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  - C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha,
  - **Lesser-known target processors:**
    - Additional processors independently supported:
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  - C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada
- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha, ARM,
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- Input languages supported:
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada
- Processors supported in standard releases:
  - Common processors:
    Alpha, ARM, Atmel AVR,
  - Lesser-known target processors:
  - Additional processors independently supported:
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin,

  - **Lesser-known target processors:**

  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha, ARM, Atmel AVR, Blackfin, HC12,
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300,
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86),
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  - C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64,
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  
  - **Common processors:**
    
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64,

  - **Lesser-known target processors:**

  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  - C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000,
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS,
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC,

  - **Lesser-known target processors:**

  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- Input languages supported:
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- Processors supported in standard releases:
  - Common processors:
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11,
  - Lesser-known target processors:
  - Additional processors independently supported:
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC,
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C,

  - **Lesser-known target processors:**

  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU,
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries,
  - **Lesser-known target processors:**

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

• Input languages supported:
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

• Processors supported in standard releases:
  ▶ Common processors:
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH,
  ▶ Lesser-known target processors:

  ▶ Additional processors independently supported:
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC,
  - **Lesser-known target processors:**

  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
    A29K,

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
    A29K, ARC,

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
    A29K, ARC, ETRAX CRIS,

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

• Input languages supported:
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

• Processors supported in standard releases:
  ▶ Common processors:
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64,
    Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU,
    System/390/zSeries, SuperH, SPARC, VAX
  ▶ Lesser-known target processors:
    A29K, ARC, ETRAX CRIS, D30V,

▶ Additional processors independently supported:
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  - C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada
- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
    - A29K, ARC, ETRAX CRIS, D30V, DSP16xx,

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
    A29K, ARC, ETRAX CRIS, D30V, DSP16xx, FR-30,

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
    A29K, ARC, ETRAX CRIS, D30V, DSP16xx, FR-30, FR-V,

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
    A29K, ARC, ETRAX CRIS, D30V, DSP16xx, FR-30, FR-V, Intel i960,

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

• **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

• **Processors supported in standard releases:**
  ▶ **Common processors:**
  Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  ▶ **Lesser-known target processors:**
  A29K, ARC, ETRAX CRIS, D30V, DSP16xx, FR-30, FR-V, Intel i960, IP2000,

▶ **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  
  - **Common processors:**
    
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  
  - **Lesser-known target processors:**
    
    A29K, ARC, ETRAX CRIS, D30V, DSP16xx, FR-30, FR-V, Intel i960, IP2000, M32R,

  
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  - C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  
  - **Common processors:**
    
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  
  - **Lesser-known target processors:**
    
    A29K, ARC, ETRAX CRIS, D30V, DSP16xx, FR-30, FR-V, Intel i960, IP2000, M32R, 68HC11, MCORE, MMIX,

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
    A29K, ARC, ETRAX CRIS, D30V, DSP16xx, FR-30, FR-V, Intel i960, IP2000, M32R, 68HC11, MCORE, MMIX, MN10200,

  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
    A29K, ARC, ETRAX CRIS, D30V, DSP16xx, FR-30, FR-V, Intel i960, IP2000, M32R, 68HC11, MCORE, MMIX, MN10200, MN10300,

- **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  - C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
    - A29K, ARC, ETRAX CRIS, D30V, DSP16xx, FR-30, FR-V, Intel i960, IP2000, M32R, 68HC11, MCORE, MMIX, MN10200, MN10300, Motorola 88000,
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
    A29K, ARC, ETRAX CRIS, D30V, DSP16xx, FR-30, FR-V, Intel i960, IP2000, M32R, 68HC11, MCORE, MMIX, MN10200, MN10300, Motorola 88000, NS32K,

  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  
  - **Common processors:**
    
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX

  - **Lesser-known target processors:**
    

  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  - C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
    - D10V,
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
    D10V, LatticeMico32, MeP,
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
    D10V, LatticeMico32, MeP,
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
    D10V, LatticeMico32, MeP, Motorola 6809,
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
    D10V, LatticeMico32, MeP, Motorola 6809, MicroBlaze,
Comprehensiveness of GCC: Wide Applicability

• Input languages supported:
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

• Processors supported in standard releases:
  ▶ Common processors:
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  ▶ Lesser-known target processors:
  ▶ Additional processors independently supported:
    D10V, LatticeMico32, MeP, Motorola 6809, MicroBlaze, MSP430,
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  
  ▶ **Common processors:**
  Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX

  ▶ **Lesser-known target processors:**

  ▶ **Additional processors independently supported:**
  D10V, LatticeMico32, MeP, Motorola 6809, MicroBlaze, MSP430, Nios II and Nios,
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
    D10V, LatticeMico32, MeP, Motorola 6809, MicroBlaze, MSP430, Nios II and Nios, PDP-10,
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  - C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada
- **Processors supported in standard releases:**
  - **Common processors:**
    - Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
    - D10V, LatticeMico32, MeP, Motorola 6809, MicroBlaze, MSP430, Nios II and Nios, PDP-10, TIGCC (m68k variant),
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
    D10V, LatticeMico32, MeP, Motorola 6809, MicroBlaze, MSP430, Nios II and Nios, PDP-10, TIGCC (m68k variant), Z8000,
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**

  - **Common processors:**
    
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX

  - **Lesser-known target processors:**
    

  - **Additional processors independently supported:**
    
    D10V, LatticeMico32, MeP, Motorola 6809, MicroBlaze, MSP430, Nios II and Nios, PDP-10, TIGCC (m68k variant), Z8000, PIC24/dsPIC,
Comprehensiveness of GCC: Wide Applicability

- **Input languages supported:**
  C, C++, Objective-C, Objective-C++, Java, Fortran, and Ada

- **Processors supported in standard releases:**
  - **Common processors:**
    Alpha, ARM, Atmel AVR, Blackfin, HC12, H8/300, IA-32 (x86), x86-64, IA-64, Motorola 68000, MIPS, PA-RISC, PDP-11, PowerPC, R8C/M16C/M32C, SPU, System/390/zSeries, SuperH, SPARC, VAX
  - **Lesser-known target processors:**
  - **Additional processors independently supported:**
    D10V, LatticeMico32, MeP, Motorola 6809, MicroBlaze, MSP430, Nios II and Nios, PDP-10, TIGCC (m68k variant), Z8000, PIC24/dsPIC, NEC SX architecture
Comprehensiveness of GCC: Size

- Overall size

<table>
<thead>
<tr>
<th>Subdirectories</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcc-4.4.2</td>
<td>3794</td>
</tr>
<tr>
<td></td>
<td>62301</td>
</tr>
<tr>
<td>gcc-4.5.0</td>
<td>4056</td>
</tr>
<tr>
<td></td>
<td>65639</td>
</tr>
<tr>
<td>gcc-4.6.0</td>
<td>4383</td>
</tr>
<tr>
<td></td>
<td>71096</td>
</tr>
</tbody>
</table>

- Core size (src/gcc)

<table>
<thead>
<tr>
<th>Subdirectories</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcc-4.4.2</td>
<td>257</td>
</tr>
<tr>
<td></td>
<td>30163</td>
</tr>
<tr>
<td>gcc-4.5.0</td>
<td>283</td>
</tr>
<tr>
<td></td>
<td>32723</td>
</tr>
<tr>
<td>gcc-4.6.0</td>
<td>336</td>
</tr>
<tr>
<td></td>
<td>36503</td>
</tr>
</tbody>
</table>

- Machine Descriptions (src/gcc/config)

<table>
<thead>
<tr>
<th>Subdirectories</th>
<th>.c files</th>
<th>.h files</th>
<th>.md files</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcc-4.4.2</td>
<td>36</td>
<td>241</td>
<td>426</td>
</tr>
<tr>
<td>gcc-4.5.0</td>
<td>42</td>
<td>275</td>
<td>478</td>
</tr>
<tr>
<td>gcc-4.6.0</td>
<td>42</td>
<td>275</td>
<td>466</td>
</tr>
</tbody>
</table>
### ohcount: Line Count of gcc-4.4.2

<table>
<thead>
<tr>
<th>Language</th>
<th>Files</th>
<th>Code</th>
<th>Comment</th>
<th>Comment %</th>
<th>Blank</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>15638</td>
<td>1840245</td>
<td>394682</td>
<td>17.7%</td>
<td>366815</td>
<td>2601742</td>
</tr>
<tr>
<td>cpp</td>
<td>19622</td>
<td>872775</td>
<td>190744</td>
<td>17.9%</td>
<td>189007</td>
<td>1252526</td>
</tr>
<tr>
<td>java</td>
<td>6342</td>
<td>681656</td>
<td>643045</td>
<td>48.5%</td>
<td>169465</td>
<td>1494166</td>
</tr>
<tr>
<td>ada</td>
<td>4206</td>
<td>638557</td>
<td>294881</td>
<td>31.6%</td>
<td>218000</td>
<td>1151438</td>
</tr>
<tr>
<td>autoconf</td>
<td>76</td>
<td>445046</td>
<td>393</td>
<td>0.1%</td>
<td>58831</td>
<td>504270</td>
</tr>
<tr>
<td>make</td>
<td>82</td>
<td>110064</td>
<td>3268</td>
<td>2.9%</td>
<td>13270</td>
<td>126602</td>
</tr>
<tr>
<td>html</td>
<td>480</td>
<td>103080</td>
<td>5658</td>
<td>5.2%</td>
<td>21438</td>
<td>130176</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2164</td>
<td>73366</td>
<td>1570</td>
<td>2.1%</td>
<td>9454</td>
<td>84390</td>
</tr>
<tr>
<td>assembler</td>
<td>183</td>
<td>42460</td>
<td>9607</td>
<td>18.5%</td>
<td>7084</td>
<td>59151</td>
</tr>
<tr>
<td>shell</td>
<td>137</td>
<td>39347</td>
<td>8832</td>
<td>18.3%</td>
<td>5485</td>
<td>53664</td>
</tr>
<tr>
<td>fortranfree</td>
<td>690</td>
<td>11852</td>
<td>2582</td>
<td>17.9%</td>
<td>1414</td>
<td>15848</td>
</tr>
<tr>
<td>objective_c</td>
<td>395</td>
<td>10562</td>
<td>1768</td>
<td>14.3%</td>
<td>2951</td>
<td>15281</td>
</tr>
<tr>
<td>automake</td>
<td>61</td>
<td>6014</td>
<td>853</td>
<td>12.4%</td>
<td>956</td>
<td>7823</td>
</tr>
<tr>
<td>perl</td>
<td>24</td>
<td>4111</td>
<td>1138</td>
<td>21.7%</td>
<td>732</td>
<td>5981</td>
</tr>
<tr>
<td>scheme</td>
<td>1</td>
<td>2775</td>
<td>153</td>
<td>5.2%</td>
<td>328</td>
<td>3256</td>
</tr>
<tr>
<td>ocaml</td>
<td>5</td>
<td>2482</td>
<td>538</td>
<td>17.8%</td>
<td>328</td>
<td>3348</td>
</tr>
<tr>
<td>python</td>
<td>6</td>
<td>1135</td>
<td>211</td>
<td>15.7%</td>
<td>220</td>
<td>1566</td>
</tr>
<tr>
<td>awk</td>
<td>9</td>
<td>1127</td>
<td>324</td>
<td>22.3%</td>
<td>193</td>
<td>1644</td>
</tr>
<tr>
<td>pascal</td>
<td>4</td>
<td>1044</td>
<td>141</td>
<td>11.9%</td>
<td>218</td>
<td>1403</td>
</tr>
<tr>
<td>csharp</td>
<td>9</td>
<td>879</td>
<td>506</td>
<td>36.5%</td>
<td>230</td>
<td>1615</td>
</tr>
<tr>
<td>dcl</td>
<td>2</td>
<td>497</td>
<td>99</td>
<td>16.6%</td>
<td>30</td>
<td>626</td>
</tr>
<tr>
<td>tcl</td>
<td>1</td>
<td>392</td>
<td>113</td>
<td>22.4%</td>
<td>72</td>
<td>577</td>
</tr>
<tr>
<td>haskell</td>
<td>48</td>
<td>149</td>
<td>0</td>
<td>0.0%</td>
<td>16</td>
<td>165</td>
</tr>
<tr>
<td>emacsclisp</td>
<td>1</td>
<td>59</td>
<td>21</td>
<td>26.2%</td>
<td>4</td>
<td>84</td>
</tr>
<tr>
<td>matlab</td>
<td>2</td>
<td>57</td>
<td>0</td>
<td>0.0%</td>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td>Total</td>
<td>50312</td>
<td>4938881</td>
<td>1567750</td>
<td>24.1%</td>
<td>1071986</td>
<td>7578617</td>
</tr>
<tr>
<td>Language</td>
<td>Files</td>
<td>Code</td>
<td>Comment</td>
<td>Comment %</td>
<td>Blank</td>
<td>Total</td>
</tr>
<tr>
<td>------------------</td>
<td>-------</td>
<td>--------</td>
<td>---------</td>
<td>-----------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>c</td>
<td>15638</td>
<td>1840245</td>
<td>394682</td>
<td>17.7%</td>
<td>366815</td>
<td>2601742</td>
</tr>
<tr>
<td>cpp</td>
<td>19622</td>
<td>872775</td>
<td>190744</td>
<td>17.9%</td>
<td>189007</td>
<td>1252526</td>
</tr>
<tr>
<td>java</td>
<td>6342</td>
<td>681656</td>
<td>643045</td>
<td>48.5%</td>
<td>169465</td>
<td>1494166</td>
</tr>
<tr>
<td>ada</td>
<td>4206</td>
<td>638557</td>
<td>294881</td>
<td>31.6%</td>
<td>218000</td>
<td>1151438</td>
</tr>
<tr>
<td>autoconf</td>
<td>76</td>
<td>445046</td>
<td>393</td>
<td>0.1%</td>
<td>58831</td>
<td>504270</td>
</tr>
<tr>
<td>make</td>
<td>82</td>
<td>110064</td>
<td>3268</td>
<td>2.9%</td>
<td>13270</td>
<td>126602</td>
</tr>
<tr>
<td>html</td>
<td>480</td>
<td>103080</td>
<td>5658</td>
<td>5.2%</td>
<td>21438</td>
<td>130176</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2164</td>
<td>73366</td>
<td>1570</td>
<td>2.1%</td>
<td>9454</td>
<td>84390</td>
</tr>
<tr>
<td>assembler</td>
<td>183</td>
<td>42460</td>
<td>9607</td>
<td>18.5%</td>
<td>7084</td>
<td>59151</td>
</tr>
<tr>
<td>shell</td>
<td>137</td>
<td>39347</td>
<td>8832</td>
<td>18.3%</td>
<td>5485</td>
<td>53664</td>
</tr>
<tr>
<td>fortranfree</td>
<td>690</td>
<td>11852</td>
<td>2582</td>
<td>17.9%</td>
<td>1414</td>
<td>15848</td>
</tr>
<tr>
<td>objective_c</td>
<td>395</td>
<td>10562</td>
<td>1768</td>
<td>14.3%</td>
<td>2951</td>
<td>15281</td>
</tr>
<tr>
<td>automake</td>
<td>61</td>
<td>6014</td>
<td>853</td>
<td>12.4%</td>
<td>956</td>
<td>7823</td>
</tr>
<tr>
<td>perl</td>
<td>24</td>
<td>4111</td>
<td>1138</td>
<td>21.7%</td>
<td>732</td>
<td>5981</td>
</tr>
<tr>
<td>scheme</td>
<td>1</td>
<td>2775</td>
<td>153</td>
<td>5.2%</td>
<td>328</td>
<td>3256</td>
</tr>
<tr>
<td>ocaml</td>
<td>5</td>
<td>2482</td>
<td>538</td>
<td>17.8%</td>
<td>328</td>
<td>3348</td>
</tr>
<tr>
<td>python</td>
<td>6</td>
<td>1135</td>
<td>211</td>
<td>15.7%</td>
<td>220</td>
<td>1566</td>
</tr>
<tr>
<td>awk</td>
<td>9</td>
<td>1127</td>
<td>324</td>
<td>22.3%</td>
<td>193</td>
<td>1644</td>
</tr>
<tr>
<td>pascal</td>
<td>4</td>
<td>1044</td>
<td>141</td>
<td>11.9%</td>
<td>218</td>
<td>1403</td>
</tr>
<tr>
<td>csharp</td>
<td>9</td>
<td>879</td>
<td>506</td>
<td>36.5%</td>
<td>230</td>
<td>1615</td>
</tr>
<tr>
<td>dcl</td>
<td>2</td>
<td>497</td>
<td>99</td>
<td>16.6%</td>
<td>30</td>
<td>626</td>
</tr>
<tr>
<td>tcl</td>
<td>1</td>
<td>392</td>
<td>113</td>
<td>22.4%</td>
<td>72</td>
<td>577</td>
</tr>
<tr>
<td>haskell</td>
<td>48</td>
<td>149</td>
<td>0</td>
<td>0.0%</td>
<td>16</td>
<td>165</td>
</tr>
<tr>
<td>emacsclisp</td>
<td>1</td>
<td>59</td>
<td>21</td>
<td>26.2%</td>
<td>4</td>
<td>84</td>
</tr>
<tr>
<td>matlab</td>
<td>2</td>
<td>57</td>
<td>0</td>
<td>0.0%</td>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>50312</strong></td>
<td><strong>4938881</strong></td>
<td><strong>1567750</strong></td>
<td><strong>24.1%</strong></td>
<td><strong>1071986</strong></td>
<td><strong>7578617</strong></td>
</tr>
</tbody>
</table>
### ohcount: Line Count of gcc-4.5.0

<table>
<thead>
<tr>
<th>Language</th>
<th>Files</th>
<th>Code</th>
<th>Comment</th>
<th>Comment %</th>
<th>Blank</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>16985</td>
<td>1967826</td>
<td>413941</td>
<td>17.4%</td>
<td>39183</td>
<td>2773650</td>
</tr>
<tr>
<td>cpp</td>
<td>20813</td>
<td>912618</td>
<td>210084</td>
<td>18.7%</td>
<td>199605</td>
<td>1322307</td>
</tr>
<tr>
<td>java</td>
<td>6342</td>
<td>681810</td>
<td>643127</td>
<td>48.5%</td>
<td>169483</td>
<td>1494420</td>
</tr>
<tr>
<td>ada</td>
<td>4412</td>
<td>647372</td>
<td>302226</td>
<td>31.8%</td>
<td>222481</td>
<td>1172079</td>
</tr>
<tr>
<td>autoconf</td>
<td>79</td>
<td>358996</td>
<td>422</td>
<td>0.1%</td>
<td>55631</td>
<td>415049</td>
</tr>
<tr>
<td>html</td>
<td>487</td>
<td>144535</td>
<td>5667</td>
<td>3.8%</td>
<td>31773</td>
<td>181975</td>
</tr>
<tr>
<td>make</td>
<td>93</td>
<td>114490</td>
<td>3438</td>
<td>2.9%</td>
<td>14434</td>
<td>132362</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2535</td>
<td>85905</td>
<td>1817</td>
<td>2.1%</td>
<td>11394</td>
<td>99116</td>
</tr>
<tr>
<td>assembler</td>
<td>197</td>
<td>45098</td>
<td>10082</td>
<td>18.3%</td>
<td>7528</td>
<td>62708</td>
</tr>
<tr>
<td>shell</td>
<td>136</td>
<td>39789</td>
<td>8984</td>
<td>18.4%</td>
<td>5511</td>
<td>54284</td>
</tr>
<tr>
<td>scheme</td>
<td>7</td>
<td>13725</td>
<td>1192</td>
<td>8.0%</td>
<td>1524</td>
<td>16441</td>
</tr>
<tr>
<td>fortranfree</td>
<td>760</td>
<td>12955</td>
<td>2889</td>
<td>18.2%</td>
<td>1546</td>
<td>17390</td>
</tr>
<tr>
<td>objective_c</td>
<td>396</td>
<td>10782</td>
<td>1835</td>
<td>14.5%</td>
<td>2959</td>
<td>15576</td>
</tr>
<tr>
<td>automake</td>
<td>64</td>
<td>6388</td>
<td>914</td>
<td>12.5%</td>
<td>994</td>
<td>8296</td>
</tr>
<tr>
<td>perl</td>
<td>25</td>
<td>4144</td>
<td>1139</td>
<td>21.6%</td>
<td>739</td>
<td>6022</td>
</tr>
<tr>
<td>xslt</td>
<td>20</td>
<td>2805</td>
<td>436</td>
<td>13.5%</td>
<td>563</td>
<td>3804</td>
</tr>
<tr>
<td>ocaml</td>
<td>5</td>
<td>2515</td>
<td>540</td>
<td>17.7%</td>
<td>328</td>
<td>3383</td>
</tr>
<tr>
<td>python</td>
<td>10</td>
<td>1686</td>
<td>322</td>
<td>16.0%</td>
<td>383</td>
<td>2391</td>
</tr>
<tr>
<td>awk</td>
<td>10</td>
<td>1352</td>
<td>372</td>
<td>21.6%</td>
<td>218</td>
<td>1942</td>
</tr>
<tr>
<td>pascal</td>
<td>4</td>
<td>1044</td>
<td>141</td>
<td>11.9%</td>
<td>218</td>
<td>1403</td>
</tr>
<tr>
<td>csharp</td>
<td>9</td>
<td>879</td>
<td>506</td>
<td>36.5%</td>
<td>230</td>
<td>1615</td>
</tr>
<tr>
<td>dcl</td>
<td>2</td>
<td>402</td>
<td>84</td>
<td>17.3%</td>
<td>13</td>
<td>499</td>
</tr>
<tr>
<td>tcl</td>
<td>1</td>
<td>392</td>
<td>113</td>
<td>22.4%</td>
<td>72</td>
<td>577</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>emacs-lisp</td>
<td>1</td>
<td>59</td>
<td>21</td>
<td>26.2%</td>
<td>4</td>
<td>84</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Language</td>
<td>Files</td>
<td>Code</td>
<td>Comment</td>
<td>Comment %</td>
<td>Blank</td>
<td>Total</td>
</tr>
<tr>
<td>--------------</td>
<td>-------</td>
<td>--------</td>
<td>---------</td>
<td>-----------</td>
<td>-------</td>
<td>--------</td>
</tr>
<tr>
<td>c</td>
<td>16985</td>
<td>1967826</td>
<td>413941</td>
<td>17.4%</td>
<td>39183</td>
<td>2773650</td>
</tr>
<tr>
<td>cpp</td>
<td>20813</td>
<td>912618</td>
<td>210084</td>
<td>18.7%</td>
<td>199605</td>
<td>1322307</td>
</tr>
<tr>
<td>java</td>
<td>6342</td>
<td>681810</td>
<td>643127</td>
<td>48.5%</td>
<td>169483</td>
<td>1494420</td>
</tr>
<tr>
<td>ada</td>
<td>4412</td>
<td>647372</td>
<td>302226</td>
<td>31.8%</td>
<td>222481</td>
<td>1172079</td>
</tr>
<tr>
<td>autoconf</td>
<td>79</td>
<td>358996</td>
<td>422</td>
<td>0.1%</td>
<td>55631</td>
<td>415049</td>
</tr>
<tr>
<td>html</td>
<td>487</td>
<td>144535</td>
<td>5667</td>
<td>3.8%</td>
<td>31773</td>
<td>181975</td>
</tr>
<tr>
<td>make</td>
<td>93</td>
<td>114490</td>
<td>3438</td>
<td>2.9%</td>
<td>14434</td>
<td>132362</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2535</td>
<td>85905</td>
<td>1817</td>
<td>2.1%</td>
<td>11394</td>
<td>99116</td>
</tr>
<tr>
<td>assembler</td>
<td>197</td>
<td>45098</td>
<td>10082</td>
<td>18.3%</td>
<td>7528</td>
<td>62708</td>
</tr>
<tr>
<td>shell</td>
<td>136</td>
<td>39789</td>
<td>8984</td>
<td>18.4%</td>
<td>5511</td>
<td>54284</td>
</tr>
<tr>
<td>scheme</td>
<td>7</td>
<td>13725</td>
<td>1192</td>
<td>8.0%</td>
<td>1524</td>
<td>16441</td>
</tr>
<tr>
<td>fortranfree</td>
<td>760</td>
<td>12955</td>
<td>2889</td>
<td>18.2%</td>
<td>1546</td>
<td>17390</td>
</tr>
<tr>
<td>objective_c</td>
<td>396</td>
<td>10782</td>
<td>1835</td>
<td>14.5%</td>
<td>2959</td>
<td>15576</td>
</tr>
<tr>
<td>automake</td>
<td>64</td>
<td>6388</td>
<td>914</td>
<td>12.5%</td>
<td>994</td>
<td>8296</td>
</tr>
<tr>
<td>perl</td>
<td>25</td>
<td>4144</td>
<td>1139</td>
<td>21.6%</td>
<td>739</td>
<td>6022</td>
</tr>
<tr>
<td>xslt</td>
<td>20</td>
<td>2805</td>
<td>436</td>
<td>13.5%</td>
<td>563</td>
<td>3804</td>
</tr>
<tr>
<td>ocaml</td>
<td>5</td>
<td>2515</td>
<td>540</td>
<td>17.7%</td>
<td>328</td>
<td>3383</td>
</tr>
<tr>
<td>python</td>
<td>10</td>
<td>1686</td>
<td>322</td>
<td>16.0%</td>
<td>383</td>
<td>2391</td>
</tr>
<tr>
<td>awk</td>
<td>10</td>
<td>1352</td>
<td>372</td>
<td>21.6%</td>
<td>218</td>
<td>1942</td>
</tr>
<tr>
<td>pascal</td>
<td>4</td>
<td>1044</td>
<td>141</td>
<td>11.9%</td>
<td>218</td>
<td>1403</td>
</tr>
<tr>
<td>csharp</td>
<td>9</td>
<td>879</td>
<td>506</td>
<td>36.5%</td>
<td>230</td>
<td>1615</td>
</tr>
<tr>
<td>dcl</td>
<td>2</td>
<td>402</td>
<td>84</td>
<td>17.3%</td>
<td>13</td>
<td>499</td>
</tr>
<tr>
<td>tcl</td>
<td>1</td>
<td>392</td>
<td>113</td>
<td>22.4%</td>
<td>72</td>
<td>577</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>emacsclisp</td>
<td>1</td>
<td>59</td>
<td>21</td>
<td>26.2%</td>
<td>4</td>
<td>84</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>
### ohcount: Line Count of gcc-4.6.0

<table>
<thead>
<tr>
<th>Language</th>
<th>Files</th>
<th>Code</th>
<th>Comment</th>
<th>Comment %</th>
<th>Blank</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>18463</td>
<td>2100237</td>
<td>444333</td>
<td>17.5%</td>
<td>418292</td>
<td>2962862</td>
</tr>
<tr>
<td>cpp</td>
<td>22002</td>
<td>985076</td>
<td>229541</td>
<td>18.9%</td>
<td>214781</td>
<td>1429398</td>
</tr>
<tr>
<td>java</td>
<td>6342</td>
<td>681938</td>
<td>645505</td>
<td>48.6%</td>
<td>169046</td>
<td>1496489</td>
</tr>
<tr>
<td>ada</td>
<td>4605</td>
<td>680043</td>
<td>315956</td>
<td>31.7%</td>
<td>234467</td>
<td>1230466</td>
</tr>
<tr>
<td>autoconf</td>
<td>91</td>
<td>405461</td>
<td>509</td>
<td>0.1%</td>
<td>62914</td>
<td>468884</td>
</tr>
<tr>
<td>html</td>
<td>457</td>
<td>168355</td>
<td>5669</td>
<td>3.3%</td>
<td>38146</td>
<td>212170</td>
</tr>
<tr>
<td>make</td>
<td>98</td>
<td>121545</td>
<td>3659</td>
<td>2.9%</td>
<td>15618</td>
<td>140822</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2936</td>
<td>99413</td>
<td>1927</td>
<td>1.9%</td>
<td>13659</td>
<td>114999</td>
</tr>
<tr>
<td>shell</td>
<td>148</td>
<td>48032</td>
<td>10451</td>
<td>17.9%</td>
<td>6586</td>
<td>65069</td>
</tr>
<tr>
<td>assembler</td>
<td>208</td>
<td>46727</td>
<td>10227</td>
<td>18.0%</td>
<td>7853</td>
<td>64807</td>
</tr>
<tr>
<td>xml</td>
<td>75</td>
<td>36036</td>
<td>282</td>
<td>0.8%</td>
<td>3827</td>
<td>40145</td>
</tr>
<tr>
<td>objective_c</td>
<td>866</td>
<td>28014</td>
<td>5000</td>
<td>15.1%</td>
<td>8115</td>
<td>41129</td>
</tr>
<tr>
<td>fortranfree</td>
<td>821</td>
<td>13857</td>
<td>3147</td>
<td>18.5%</td>
<td>1695</td>
<td>18699</td>
</tr>
<tr>
<td>tex</td>
<td>2</td>
<td>11060</td>
<td>5776</td>
<td>34.3%</td>
<td>1433</td>
<td>18269</td>
</tr>
<tr>
<td>scheme</td>
<td>6</td>
<td>11023</td>
<td>1010</td>
<td>8.4%</td>
<td>1205</td>
<td>13238</td>
</tr>
<tr>
<td>automake</td>
<td>67</td>
<td>9440</td>
<td>1038</td>
<td>9.9%</td>
<td>1456</td>
<td>11934</td>
</tr>
<tr>
<td>perl</td>
<td>28</td>
<td>4445</td>
<td>1316</td>
<td>22.8%</td>
<td>837</td>
<td>6598</td>
</tr>
<tr>
<td>ocaml</td>
<td>6</td>
<td>2814</td>
<td>576</td>
<td>17.0%</td>
<td>378</td>
<td>3768</td>
</tr>
<tr>
<td>xslt</td>
<td>20</td>
<td>2805</td>
<td>436</td>
<td>13.5%</td>
<td>563</td>
<td>3804</td>
</tr>
<tr>
<td>awk</td>
<td>11</td>
<td>1740</td>
<td>396</td>
<td>18.5%</td>
<td>257</td>
<td>2393</td>
</tr>
<tr>
<td>python</td>
<td>10</td>
<td>1725</td>
<td>322</td>
<td>15.7%</td>
<td>383</td>
<td>2430</td>
</tr>
<tr>
<td>css</td>
<td>24</td>
<td>1589</td>
<td>143</td>
<td>8.3%</td>
<td>332</td>
<td>2064</td>
</tr>
<tr>
<td>pascal</td>
<td>4</td>
<td>1044</td>
<td>141</td>
<td>11.9%</td>
<td>218</td>
<td>1403</td>
</tr>
<tr>
<td>csharp</td>
<td>9</td>
<td>879</td>
<td>506</td>
<td>36.5%</td>
<td>230</td>
<td>1615</td>
</tr>
<tr>
<td>dcl</td>
<td>2</td>
<td>402</td>
<td>84</td>
<td>17.3%</td>
<td>13</td>
<td>499</td>
</tr>
<tr>
<td>tcl</td>
<td>1</td>
<td>392</td>
<td>113</td>
<td>22.4%</td>
<td>72</td>
<td>577</td>
</tr>
<tr>
<td>javascript</td>
<td>4</td>
<td>341</td>
<td>87</td>
<td>20.3%</td>
<td>35</td>
<td>463</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>bat</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Total</td>
<td>57359</td>
<td>5464598</td>
<td>1688150</td>
<td>23.6%</td>
<td>1202428</td>
<td>8355176</td>
</tr>
<tr>
<td>Language</td>
<td>Files</td>
<td>Code</td>
<td>Comment</td>
<td>Comment %</td>
<td>Blank</td>
<td>Total</td>
</tr>
<tr>
<td>---------------</td>
<td>-------</td>
<td>--------</td>
<td>---------</td>
<td>-----------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>c</td>
<td>18463</td>
<td>2100237</td>
<td>444333</td>
<td>17.5%</td>
<td>418292</td>
<td>2962862</td>
</tr>
<tr>
<td>cpp</td>
<td>22002</td>
<td>985076</td>
<td>229541</td>
<td>18.9%</td>
<td>214781</td>
<td>1429398</td>
</tr>
<tr>
<td>java</td>
<td>6342</td>
<td>681938</td>
<td>645505</td>
<td>48.6%</td>
<td>169046</td>
<td>1496489</td>
</tr>
<tr>
<td>ada</td>
<td>4605</td>
<td>680043</td>
<td>315956</td>
<td>31.7%</td>
<td>234467</td>
<td>1230466</td>
</tr>
<tr>
<td>autoconf</td>
<td>91</td>
<td>405461</td>
<td>509</td>
<td>0.1%</td>
<td>62914</td>
<td>468884</td>
</tr>
<tr>
<td>html</td>
<td>457</td>
<td>168355</td>
<td>5669</td>
<td>3.3%</td>
<td>38146</td>
<td>212170</td>
</tr>
<tr>
<td>make</td>
<td>98</td>
<td>121545</td>
<td>3659</td>
<td>2.9%</td>
<td>15618</td>
<td>140822</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2936</td>
<td>99413</td>
<td>1927</td>
<td>1.9%</td>
<td>13659</td>
<td>114999</td>
</tr>
<tr>
<td>shell</td>
<td>148</td>
<td>48032</td>
<td>10451</td>
<td>17.9%</td>
<td>6586</td>
<td>65069</td>
</tr>
<tr>
<td>assembler</td>
<td>208</td>
<td>46727</td>
<td>10227</td>
<td>18.0%</td>
<td>7853</td>
<td>64807</td>
</tr>
<tr>
<td>xml</td>
<td>75</td>
<td>36036</td>
<td>282</td>
<td>0.8%</td>
<td>3827</td>
<td>40145</td>
</tr>
<tr>
<td>objective_c</td>
<td>866</td>
<td>28014</td>
<td>5000</td>
<td>15.1%</td>
<td>8115</td>
<td>41129</td>
</tr>
<tr>
<td>fortranfree</td>
<td>821</td>
<td>13857</td>
<td>3147</td>
<td>18.5%</td>
<td>1695</td>
<td>18699</td>
</tr>
<tr>
<td>tex</td>
<td>2</td>
<td>11060</td>
<td>5776</td>
<td>34.3%</td>
<td>1433</td>
<td>18269</td>
</tr>
<tr>
<td>scheme</td>
<td>6</td>
<td>11023</td>
<td>1010</td>
<td>8.4%</td>
<td>1205</td>
<td>13238</td>
</tr>
<tr>
<td>automake</td>
<td>67</td>
<td>9440</td>
<td>1038</td>
<td>9.9%</td>
<td>1456</td>
<td>11934</td>
</tr>
<tr>
<td>perl</td>
<td>28</td>
<td>4445</td>
<td>1316</td>
<td>22.8%</td>
<td>837</td>
<td>6598</td>
</tr>
<tr>
<td>ocaml</td>
<td>6</td>
<td>2814</td>
<td>576</td>
<td>17.0%</td>
<td>378</td>
<td>3768</td>
</tr>
<tr>
<td>xslt</td>
<td>20</td>
<td>2805</td>
<td>436</td>
<td>13.5%</td>
<td>563</td>
<td>3804</td>
</tr>
<tr>
<td>awk</td>
<td>11</td>
<td>1740</td>
<td>396</td>
<td>18.5%</td>
<td>257</td>
<td>2393</td>
</tr>
<tr>
<td>python</td>
<td>10</td>
<td>1725</td>
<td>322</td>
<td>15.7%</td>
<td>383</td>
<td>2430</td>
</tr>
<tr>
<td>css</td>
<td>24</td>
<td>1589</td>
<td>143</td>
<td>8.3%</td>
<td>332</td>
<td>2064</td>
</tr>
<tr>
<td>pascal</td>
<td>4</td>
<td>1044</td>
<td>141</td>
<td>11.9%</td>
<td>218</td>
<td>1403</td>
</tr>
<tr>
<td>csharp</td>
<td>9</td>
<td>879</td>
<td>506</td>
<td>36.5%</td>
<td>230</td>
<td>1615</td>
</tr>
<tr>
<td>dcl</td>
<td>2</td>
<td>402</td>
<td>84</td>
<td>17.3%</td>
<td>13</td>
<td>499</td>
</tr>
<tr>
<td>tcl</td>
<td>1</td>
<td>392</td>
<td>113</td>
<td>22.4%</td>
<td>72</td>
<td>577</td>
</tr>
<tr>
<td>javascript</td>
<td>4</td>
<td>341</td>
<td>87</td>
<td>20.3%</td>
<td>35</td>
<td>463</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>bat</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Total</td>
<td>57359</td>
<td>5464598</td>
<td>1688150</td>
<td>23.6%</td>
<td>1202428</td>
<td>8355176</td>
</tr>
</tbody>
</table>
## Line Count of gcc-4.6.2

### Language Files Code Comment Comment % Blank Total

<table>
<thead>
<tr>
<th>Language</th>
<th>Files</th>
<th>Code</th>
<th>Comment</th>
<th>Comment %</th>
<th>Blank</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>18624</td>
<td>2106311</td>
<td>445288</td>
<td>17.5%</td>
<td>419325</td>
<td>2970924</td>
</tr>
<tr>
<td>cpp</td>
<td>22206</td>
<td>989098</td>
<td>230376</td>
<td>18.9%</td>
<td>215739</td>
<td>1435213</td>
</tr>
<tr>
<td>java</td>
<td>6342</td>
<td>681938</td>
<td>645505</td>
<td>48.6%</td>
<td>169046</td>
<td>1496489</td>
</tr>
<tr>
<td>ada</td>
<td>4616</td>
<td>680251</td>
<td>316021</td>
<td>31.7%</td>
<td>234551</td>
<td>1230823</td>
</tr>
<tr>
<td>autoconf</td>
<td>91</td>
<td>405517</td>
<td>509</td>
<td>0.1%</td>
<td>62919</td>
<td>468945</td>
</tr>
<tr>
<td>html</td>
<td>457</td>
<td>168378</td>
<td>5669</td>
<td>3.3%</td>
<td>38146</td>
<td>212193</td>
</tr>
<tr>
<td>make</td>
<td>98</td>
<td>121136</td>
<td>3658</td>
<td>2.9%</td>
<td>15555</td>
<td>140349</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2989</td>
<td>100688</td>
<td>1950</td>
<td>1.9%</td>
<td>13894</td>
<td>116532</td>
</tr>
<tr>
<td>shell</td>
<td>148</td>
<td>48032</td>
<td>10451</td>
<td>17.9%</td>
<td>6586</td>
<td>65069</td>
</tr>
<tr>
<td>assembler</td>
<td>208</td>
<td>46750</td>
<td>10227</td>
<td>17.9%</td>
<td>7854</td>
<td>64831</td>
</tr>
<tr>
<td>xml</td>
<td>75</td>
<td>36178</td>
<td>282</td>
<td>0.8%</td>
<td>3827</td>
<td>40287</td>
</tr>
<tr>
<td>objective_c</td>
<td>869</td>
<td>28049</td>
<td>5023</td>
<td>15.2%</td>
<td>8124</td>
<td>41196</td>
</tr>
<tr>
<td>fortranfree</td>
<td>831</td>
<td>13996</td>
<td>3204</td>
<td>18.6%</td>
<td>1728</td>
<td>18928</td>
</tr>
<tr>
<td>tex</td>
<td>2</td>
<td>11060</td>
<td>5776</td>
<td>34.3%</td>
<td>1433</td>
<td>18269</td>
</tr>
<tr>
<td>scheme</td>
<td>6</td>
<td>11023</td>
<td>1010</td>
<td>8.4%</td>
<td>1205</td>
<td>13238</td>
</tr>
<tr>
<td>automake</td>
<td>67</td>
<td>9442</td>
<td>1039</td>
<td>9.9%</td>
<td>1457</td>
<td>11938</td>
</tr>
<tr>
<td>perl</td>
<td>28</td>
<td>4445</td>
<td>1316</td>
<td>22.8%</td>
<td>837</td>
<td>6598</td>
</tr>
<tr>
<td>ocaml</td>
<td>6</td>
<td>2814</td>
<td>576</td>
<td>17.0%</td>
<td>378</td>
<td>3768</td>
</tr>
<tr>
<td>xslt</td>
<td>20</td>
<td>2805</td>
<td>436</td>
<td>13.5%</td>
<td>563</td>
<td>3804</td>
</tr>
<tr>
<td>awk</td>
<td>11</td>
<td>1740</td>
<td>396</td>
<td>18.5%</td>
<td>257</td>
<td>2393</td>
</tr>
<tr>
<td>python</td>
<td>10</td>
<td>1725</td>
<td>322</td>
<td>15.7%</td>
<td>383</td>
<td>2430</td>
</tr>
<tr>
<td>css</td>
<td>24</td>
<td>1589</td>
<td>143</td>
<td>8.3%</td>
<td>332</td>
<td>2064</td>
</tr>
<tr>
<td>pascal</td>
<td>4</td>
<td>1044</td>
<td>141</td>
<td>11.9%</td>
<td>218</td>
<td>1403</td>
</tr>
<tr>
<td>csharp</td>
<td>9</td>
<td>879</td>
<td>506</td>
<td>36.5%</td>
<td>230</td>
<td>1615</td>
</tr>
<tr>
<td>dcl</td>
<td>2</td>
<td>402</td>
<td>84</td>
<td>17.3%</td>
<td>13</td>
<td>499</td>
</tr>
<tr>
<td>tcl</td>
<td>1</td>
<td>392</td>
<td>113</td>
<td>22.4%</td>
<td>72</td>
<td>577</td>
</tr>
<tr>
<td>javascript</td>
<td>4</td>
<td>341</td>
<td>87</td>
<td>20.3%</td>
<td>35</td>
<td>463</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>bat</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Total</td>
<td>57801</td>
<td>5476188</td>
<td>1690108</td>
<td>23.6%</td>
<td>1204724</td>
<td>8371020</td>
</tr>
</tbody>
</table>

---

**Essential Abstractions in GCC**

**GCC Resource Center, IIT Bombay**

---

**ohcount**
<table>
<thead>
<tr>
<th>Language</th>
<th>Files</th>
<th>Code</th>
<th>Comment</th>
<th>Comment %</th>
<th>Blank</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>18624</td>
<td>2106311</td>
<td>445288</td>
<td>17.5%</td>
<td>419325</td>
<td>2970924</td>
</tr>
<tr>
<td>cpp</td>
<td>22206</td>
<td>989098</td>
<td>230376</td>
<td>18.9%</td>
<td>215739</td>
<td>1435213</td>
</tr>
<tr>
<td>java</td>
<td>6342</td>
<td>681938</td>
<td>645505</td>
<td>48.6%</td>
<td>169046</td>
<td>1496489</td>
</tr>
<tr>
<td>ada</td>
<td>4616</td>
<td>680251</td>
<td>316021</td>
<td>31.7%</td>
<td>234551</td>
<td>1230823</td>
</tr>
<tr>
<td>autoconf</td>
<td>91</td>
<td>405517</td>
<td>509</td>
<td>0.1%</td>
<td>62919</td>
<td>468945</td>
</tr>
<tr>
<td>html</td>
<td>457</td>
<td>168378</td>
<td>5669</td>
<td>3.3%</td>
<td>38146</td>
<td>212193</td>
</tr>
<tr>
<td>make</td>
<td>98</td>
<td>121136</td>
<td>3658</td>
<td>2.9%</td>
<td>15555</td>
<td>140349</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2989</td>
<td>100688</td>
<td>1950</td>
<td>1.9%</td>
<td>13894</td>
<td>116532</td>
</tr>
<tr>
<td>shell</td>
<td>148</td>
<td>48032</td>
<td>10451</td>
<td>17.9%</td>
<td>6586</td>
<td>65069</td>
</tr>
<tr>
<td>assembler</td>
<td>208</td>
<td>46750</td>
<td>10227</td>
<td>17.9%</td>
<td>7854</td>
<td>64831</td>
</tr>
<tr>
<td>xml</td>
<td>75</td>
<td>36178</td>
<td>282</td>
<td>0.8%</td>
<td>3827</td>
<td>40287</td>
</tr>
<tr>
<td>objective_c</td>
<td>869</td>
<td>28049</td>
<td>5023</td>
<td>15.2%</td>
<td>8124</td>
<td>41196</td>
</tr>
<tr>
<td>fortranfree</td>
<td>831</td>
<td>13996</td>
<td>3204</td>
<td>18.6%</td>
<td>1728</td>
<td>18928</td>
</tr>
<tr>
<td>tex</td>
<td>2</td>
<td>11060</td>
<td>5776</td>
<td>34.3%</td>
<td>1433</td>
<td>18269</td>
</tr>
<tr>
<td>scheme</td>
<td>6</td>
<td>11023</td>
<td>1010</td>
<td>8.4%</td>
<td>1205</td>
<td>13238</td>
</tr>
<tr>
<td>automake</td>
<td>67</td>
<td>9442</td>
<td>1039</td>
<td>9.9%</td>
<td>1457</td>
<td>11938</td>
</tr>
<tr>
<td>perl</td>
<td>28</td>
<td>4445</td>
<td>1316</td>
<td>22.8%</td>
<td>837</td>
<td>6598</td>
</tr>
<tr>
<td>ocaml</td>
<td>6</td>
<td>2814</td>
<td>576</td>
<td>17.0%</td>
<td>378</td>
<td>3768</td>
</tr>
<tr>
<td>xslt</td>
<td>20</td>
<td>2805</td>
<td>436</td>
<td>13.5%</td>
<td>563</td>
<td>3804</td>
</tr>
<tr>
<td>awk</td>
<td>11</td>
<td>1740</td>
<td>396</td>
<td>18.5%</td>
<td>257</td>
<td>2393</td>
</tr>
<tr>
<td>python</td>
<td>10</td>
<td>1725</td>
<td>322</td>
<td>15.7%</td>
<td>383</td>
<td>2430</td>
</tr>
<tr>
<td>css</td>
<td>24</td>
<td>1589</td>
<td>143</td>
<td>8.3%</td>
<td>332</td>
<td>2064</td>
</tr>
<tr>
<td>pascal</td>
<td>4</td>
<td>1044</td>
<td>141</td>
<td>11.9%</td>
<td>218</td>
<td>1403</td>
</tr>
<tr>
<td>csharp</td>
<td>9</td>
<td>879</td>
<td>506</td>
<td>36.5%</td>
<td>230</td>
<td>1615</td>
</tr>
<tr>
<td>dcl</td>
<td>2</td>
<td>402</td>
<td>84</td>
<td>17.3%</td>
<td>13</td>
<td>499</td>
</tr>
<tr>
<td>tcl</td>
<td>1</td>
<td>392</td>
<td>113</td>
<td>22.4%</td>
<td>72</td>
<td>577</td>
</tr>
<tr>
<td>javascript</td>
<td>4</td>
<td>341</td>
<td>87</td>
<td>20.3%</td>
<td>35</td>
<td>463</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>bat</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Total</td>
<td>57801</td>
<td>5476188</td>
<td>1690108</td>
<td>23.6%</td>
<td>1204724</td>
<td>8371020</td>
</tr>
</tbody>
</table>
### ohcount: Line Count of gcc-4.4.2/gcc

<table>
<thead>
<tr>
<th>Language</th>
<th>Files</th>
<th>Code</th>
<th>Comment</th>
<th>Comment %</th>
<th>Blank</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>13296</td>
<td>1254253</td>
<td>282582</td>
<td>18.4%</td>
<td>283766</td>
<td>1820601</td>
</tr>
<tr>
<td>ada</td>
<td>4196</td>
<td>636876</td>
<td>294321</td>
<td>31.6%</td>
<td>217401</td>
<td>1148598</td>
</tr>
<tr>
<td>cpp</td>
<td>7418</td>
<td>184186</td>
<td>52163</td>
<td>22.1%</td>
<td>54048</td>
<td>290397</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2086</td>
<td>67988</td>
<td>1521</td>
<td>2.2%</td>
<td>9079</td>
<td>78588</td>
</tr>
<tr>
<td>assembler</td>
<td>132</td>
<td>31092</td>
<td>7243</td>
<td>18.9%</td>
<td>4770</td>
<td>43105</td>
</tr>
<tr>
<td>autoconf</td>
<td>3</td>
<td>26996</td>
<td>10</td>
<td>0.0%</td>
<td>3383</td>
<td>30389</td>
</tr>
<tr>
<td>fortranfree</td>
<td>652</td>
<td>10898</td>
<td>2376</td>
<td>17.9%</td>
<td>1314</td>
<td>14588</td>
</tr>
<tr>
<td>objective_c</td>
<td>391</td>
<td>10155</td>
<td>1654</td>
<td>14.0%</td>
<td>2830</td>
<td>14639</td>
</tr>
<tr>
<td>make</td>
<td>3</td>
<td>5340</td>
<td>1027</td>
<td>16.1%</td>
<td>814</td>
<td>7181</td>
</tr>
<tr>
<td>scheme</td>
<td>1</td>
<td>2775</td>
<td>153</td>
<td>5.2%</td>
<td>328</td>
<td>3256</td>
</tr>
<tr>
<td>ocaml</td>
<td>5</td>
<td>2482</td>
<td>538</td>
<td>17.8%</td>
<td>328</td>
<td>3348</td>
</tr>
<tr>
<td>shell</td>
<td>16</td>
<td>2256</td>
<td>712</td>
<td>24.0%</td>
<td>374</td>
<td>3342</td>
</tr>
<tr>
<td>awk</td>
<td>7</td>
<td>1022</td>
<td>251</td>
<td>19.7%</td>
<td>187</td>
<td>1460</td>
</tr>
<tr>
<td>perl</td>
<td>1</td>
<td>772</td>
<td>205</td>
<td>21.0%</td>
<td>137</td>
<td>1114</td>
</tr>
<tr>
<td>haskell</td>
<td>48</td>
<td>149</td>
<td>0</td>
<td>0.0%</td>
<td>16</td>
<td>165</td>
</tr>
<tr>
<td>matlab</td>
<td>2</td>
<td>57</td>
<td>0</td>
<td>0.0%</td>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>28258</td>
<td>2242738</td>
<td>647591</td>
<td>22.4%</td>
<td>579484</td>
<td>3469813</td>
</tr>
</tbody>
</table>

**Essential Abstractions in GCC**

GCC Resource Center, IIT Bombay
### Essential Abstractions in GCC

GCC Resource Center, IIT Bombay

<table>
<thead>
<tr>
<th>Language</th>
<th>Files</th>
<th>Code</th>
<th>Comment</th>
<th>Comment %</th>
<th>Blank</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>13296</td>
<td>1254253</td>
<td>282582</td>
<td>18.4%</td>
<td>283766</td>
<td>1820601</td>
</tr>
<tr>
<td>ada</td>
<td>4196</td>
<td>636876</td>
<td>294321</td>
<td>31.6%</td>
<td>217401</td>
<td>1148598</td>
</tr>
<tr>
<td>cpp</td>
<td>7418</td>
<td>184186</td>
<td>52163</td>
<td>22.1%</td>
<td>54048</td>
<td>290397</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2086</td>
<td>67988</td>
<td>1521</td>
<td>2.2%</td>
<td>9079</td>
<td>78588</td>
</tr>
<tr>
<td>assembler</td>
<td>132</td>
<td>31092</td>
<td>7243</td>
<td>18.9%</td>
<td>4770</td>
<td>43105</td>
</tr>
<tr>
<td>autoconf</td>
<td>3</td>
<td>26996</td>
<td>10</td>
<td>0.0%</td>
<td>3383</td>
<td>30389</td>
</tr>
<tr>
<td>fortranfree</td>
<td>652</td>
<td>10898</td>
<td>2376</td>
<td>17.9%</td>
<td>1314</td>
<td>14588</td>
</tr>
<tr>
<td>objective_c</td>
<td>391</td>
<td>10155</td>
<td>1654</td>
<td>14.0%</td>
<td>2830</td>
<td>14639</td>
</tr>
<tr>
<td>make</td>
<td>3</td>
<td>5340</td>
<td>1027</td>
<td>16.1%</td>
<td>814</td>
<td>7181</td>
</tr>
<tr>
<td>scheme</td>
<td>1</td>
<td>2775</td>
<td>153</td>
<td>5.2%</td>
<td>328</td>
<td>3256</td>
</tr>
<tr>
<td>ocaml</td>
<td>5</td>
<td>2482</td>
<td>538</td>
<td>17.8%</td>
<td>328</td>
<td>3348</td>
</tr>
<tr>
<td>shell</td>
<td>16</td>
<td>2256</td>
<td>712</td>
<td>24.0%</td>
<td>374</td>
<td>3342</td>
</tr>
<tr>
<td>awk</td>
<td>7</td>
<td>1022</td>
<td>251</td>
<td>19.7%</td>
<td>187</td>
<td>1460</td>
</tr>
<tr>
<td>perl</td>
<td>1</td>
<td>772</td>
<td>205</td>
<td>21.0%</td>
<td>137</td>
<td>1114</td>
</tr>
<tr>
<td>haskell</td>
<td>48</td>
<td>149</td>
<td>0</td>
<td>0.0%</td>
<td>16</td>
<td>165</td>
</tr>
<tr>
<td>matlab</td>
<td>2</td>
<td>57</td>
<td>0</td>
<td>0.0%</td>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>28258</td>
<td>2242738</td>
<td>647591</td>
<td>22.4%</td>
<td>579484</td>
<td>3469813</td>
</tr>
</tbody>
</table>
30 June 2012

Overview: GCC ≡ The Great Compiler Challenge

**ohcount: Line Count of gcc-4.5.0/gcc**

<table>
<thead>
<tr>
<th>Language</th>
<th>Files</th>
<th>Code</th>
<th>Comment</th>
<th>Comment %</th>
<th>Blank</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>14565</td>
<td>1368937</td>
<td>300284</td>
<td>18.0%</td>
<td>305671</td>
<td>1974892</td>
</tr>
<tr>
<td>ada</td>
<td>4402</td>
<td>645691</td>
<td>301666</td>
<td>31.8%</td>
<td>221882</td>
<td>1169239</td>
</tr>
<tr>
<td>cpp</td>
<td>7984</td>
<td>197798</td>
<td>54719</td>
<td>21.7%</td>
<td>57312</td>
<td>309829</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2453</td>
<td>80403</td>
<td>1768</td>
<td>2.2%</td>
<td>11008</td>
<td>93179</td>
</tr>
<tr>
<td>assembler</td>
<td>136</td>
<td>31802</td>
<td>7431</td>
<td>18.9%</td>
<td>4864</td>
<td>44097</td>
</tr>
<tr>
<td>autoconf</td>
<td>3</td>
<td>27317</td>
<td>10</td>
<td>0.0%</td>
<td>3876</td>
<td>31203</td>
</tr>
<tr>
<td>scheme</td>
<td>7</td>
<td>13725</td>
<td>1192</td>
<td>8.0%</td>
<td>1524</td>
<td>16441</td>
</tr>
<tr>
<td>fortranfree</td>
<td>722</td>
<td>12001</td>
<td>2683</td>
<td>18.3%</td>
<td>1446</td>
<td>16130</td>
</tr>
<tr>
<td>objective_c</td>
<td>392</td>
<td>10375</td>
<td>1721</td>
<td>14.2%</td>
<td>2838</td>
<td>14934</td>
</tr>
<tr>
<td>make</td>
<td>3</td>
<td>5886</td>
<td>1039</td>
<td>15.0%</td>
<td>854</td>
<td>7779</td>
</tr>
<tr>
<td>ocaml</td>
<td>5</td>
<td>2515</td>
<td>540</td>
<td>17.7%</td>
<td>328</td>
<td>3383</td>
</tr>
<tr>
<td>shell</td>
<td>14</td>
<td>2101</td>
<td>642</td>
<td>23.4%</td>
<td>347</td>
<td>3090</td>
</tr>
<tr>
<td>awk</td>
<td>8</td>
<td>1247</td>
<td>299</td>
<td>19.3%</td>
<td>212</td>
<td>1758</td>
</tr>
<tr>
<td>perl</td>
<td>2</td>
<td>805</td>
<td>206</td>
<td>20.4%</td>
<td>144</td>
<td>1155</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>30747</td>
<td>2406202</td>
<td>677035</td>
<td>22.0%</td>
<td>613025</td>
<td>3696262</td>
</tr>
</tbody>
</table>

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
<table>
<thead>
<tr>
<th>Language</th>
<th>Files</th>
<th>Code</th>
<th>Comment</th>
<th>Comment %</th>
<th>Blank</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>14565</td>
<td>1368937</td>
<td>300284</td>
<td>18.0%</td>
<td>305671</td>
<td>1974892</td>
</tr>
<tr>
<td>ada</td>
<td>4402</td>
<td>645691</td>
<td>301666</td>
<td>31.8%</td>
<td>221882</td>
<td>1169239</td>
</tr>
<tr>
<td>cpp</td>
<td>7984</td>
<td>197798</td>
<td>54719</td>
<td>21.7%</td>
<td>57312</td>
<td>309829</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2453</td>
<td>80403</td>
<td>1768</td>
<td>2.2%</td>
<td>11008</td>
<td>93179</td>
</tr>
<tr>
<td>assembler</td>
<td>136</td>
<td>31802</td>
<td>7431</td>
<td>18.9%</td>
<td>4864</td>
<td>44097</td>
</tr>
<tr>
<td>autoconf</td>
<td>3</td>
<td>27317</td>
<td>10</td>
<td>0.0%</td>
<td>3876</td>
<td>31203</td>
</tr>
<tr>
<td>scheme</td>
<td>7</td>
<td>13725</td>
<td>1192</td>
<td>8.0%</td>
<td>1524</td>
<td>16441</td>
</tr>
<tr>
<td>fortranfree</td>
<td>722</td>
<td>12001</td>
<td>2683</td>
<td>18.3%</td>
<td>1446</td>
<td>16130</td>
</tr>
<tr>
<td>objective_c</td>
<td>392</td>
<td>10375</td>
<td>1721</td>
<td>14.2%</td>
<td>2838</td>
<td>14934</td>
</tr>
<tr>
<td>make</td>
<td>3</td>
<td>5886</td>
<td>1039</td>
<td>15.0%</td>
<td>854</td>
<td>7779</td>
</tr>
<tr>
<td>ocaml</td>
<td>5</td>
<td>2515</td>
<td>540</td>
<td>17.7%</td>
<td>328</td>
<td>3383</td>
</tr>
<tr>
<td>shell</td>
<td>14</td>
<td>2101</td>
<td>642</td>
<td>23.4%</td>
<td>347</td>
<td>3090</td>
</tr>
<tr>
<td>awk</td>
<td>8</td>
<td>1247</td>
<td>299</td>
<td>19.3%</td>
<td>212</td>
<td>1758</td>
</tr>
<tr>
<td>perl</td>
<td>2</td>
<td>805</td>
<td>206</td>
<td>20.4%</td>
<td>144</td>
<td>1155</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Total</td>
<td>30747</td>
<td>2406202</td>
<td>677035</td>
<td>22.0%</td>
<td>613025</td>
<td>3696262</td>
</tr>
</tbody>
</table>

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
## ohcount: Line Count of gcc-4.6.0/gcc

<table>
<thead>
<tr>
<th>Language</th>
<th>Files</th>
<th>Code</th>
<th>Comment</th>
<th>Comment %</th>
<th>Blank</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>15787</td>
<td>1462494</td>
<td>321820</td>
<td>18.0%</td>
<td>324179</td>
<td>2108493</td>
</tr>
<tr>
<td>ada</td>
<td>4595</td>
<td>678362</td>
<td>315396</td>
<td>31.7%</td>
<td>233868</td>
<td>1227626</td>
</tr>
<tr>
<td>cpp</td>
<td>8666</td>
<td>252213</td>
<td>61026</td>
<td>19.5%</td>
<td>67144</td>
<td>380383</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2850</td>
<td>93549</td>
<td>1878</td>
<td>2.0%</td>
<td>13260</td>
<td>108687</td>
</tr>
<tr>
<td>assembler</td>
<td>137</td>
<td>31548</td>
<td>7446</td>
<td>19.1%</td>
<td>4857</td>
<td>43851</td>
</tr>
<tr>
<td>autoconf</td>
<td>3</td>
<td>28775</td>
<td>12</td>
<td>0.0%</td>
<td>4020</td>
<td>32807</td>
</tr>
<tr>
<td>objective_c</td>
<td>861</td>
<td>27465</td>
<td>4822</td>
<td>14.9%</td>
<td>7967</td>
<td>40254</td>
</tr>
<tr>
<td>fortranfree</td>
<td>783</td>
<td>12903</td>
<td>2936</td>
<td>18.5%</td>
<td>1595</td>
<td>17434</td>
</tr>
<tr>
<td>scheme</td>
<td>6</td>
<td>11023</td>
<td>1010</td>
<td>8.4%</td>
<td>1205</td>
<td>13238</td>
</tr>
<tr>
<td>make</td>
<td>4</td>
<td>6078</td>
<td>1070</td>
<td>15.0%</td>
<td>893</td>
<td>8041</td>
</tr>
<tr>
<td>tex</td>
<td>1</td>
<td>5441</td>
<td>2835</td>
<td>34.3%</td>
<td>702</td>
<td>8978</td>
</tr>
<tr>
<td>ocaml</td>
<td>6</td>
<td>2814</td>
<td>576</td>
<td>17.0%</td>
<td>378</td>
<td>3768</td>
</tr>
<tr>
<td>shell</td>
<td>16</td>
<td>1980</td>
<td>597</td>
<td>23.2%</td>
<td>338</td>
<td>2915</td>
</tr>
<tr>
<td>awk</td>
<td>9</td>
<td>1635</td>
<td>323</td>
<td>16.5%</td>
<td>251</td>
<td>2209</td>
</tr>
<tr>
<td>perl</td>
<td>3</td>
<td>866</td>
<td>225</td>
<td>20.6%</td>
<td>158</td>
<td>1249</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>33777</td>
<td>2617304</td>
<td>721972</td>
<td>21.6%</td>
<td>660832</td>
<td>4000108</td>
</tr>
<tr>
<td>Language</td>
<td>Files</td>
<td>Code</td>
<td>Comment</td>
<td>Comment %</td>
<td>Blank</td>
<td>Total</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------</td>
<td>---------</td>
<td>---------</td>
<td>-----------</td>
<td>--------</td>
<td>----------</td>
</tr>
<tr>
<td>c</td>
<td>15787</td>
<td>1462494</td>
<td>321820</td>
<td>18.0%</td>
<td>324179</td>
<td>2108493</td>
</tr>
<tr>
<td>ada</td>
<td>4595</td>
<td>678362</td>
<td>315396</td>
<td>31.7%</td>
<td>233868</td>
<td>1227626</td>
</tr>
<tr>
<td>cpp</td>
<td>8666</td>
<td>252213</td>
<td>61026</td>
<td>19.5%</td>
<td>67144</td>
<td>380383</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2850</td>
<td>93549</td>
<td>1878</td>
<td>2.0%</td>
<td>13260</td>
<td>108687</td>
</tr>
<tr>
<td>assembler</td>
<td>137</td>
<td>31548</td>
<td>7446</td>
<td>19.1%</td>
<td>4857</td>
<td>43851</td>
</tr>
<tr>
<td>autoconf</td>
<td>3</td>
<td>28775</td>
<td>12</td>
<td>0.0%</td>
<td>4020</td>
<td>32807</td>
</tr>
<tr>
<td>objective_c</td>
<td>861</td>
<td>27465</td>
<td>4822</td>
<td>14.9%</td>
<td>7967</td>
<td>40254</td>
</tr>
<tr>
<td>fortranfree</td>
<td>783</td>
<td>12903</td>
<td>2936</td>
<td>18.5%</td>
<td>1595</td>
<td>17434</td>
</tr>
<tr>
<td>scheme</td>
<td>6</td>
<td>11023</td>
<td>1010</td>
<td>8.4%</td>
<td>1205</td>
<td>13238</td>
</tr>
<tr>
<td>make</td>
<td>4</td>
<td>6078</td>
<td>1070</td>
<td>15.0%</td>
<td>893</td>
<td>8041</td>
</tr>
<tr>
<td>tex</td>
<td>1</td>
<td>5441</td>
<td>2835</td>
<td>34.3%</td>
<td>702</td>
<td>8978</td>
</tr>
<tr>
<td>ocaml</td>
<td>6</td>
<td>2814</td>
<td>576</td>
<td>17.0%</td>
<td>378</td>
<td>3768</td>
</tr>
<tr>
<td>shell</td>
<td>16</td>
<td>1980</td>
<td>597</td>
<td>23.2%</td>
<td>338</td>
<td>2915</td>
</tr>
<tr>
<td>awk</td>
<td>9</td>
<td>1635</td>
<td>323</td>
<td>16.5%</td>
<td>251</td>
<td>2209</td>
</tr>
<tr>
<td>perl</td>
<td>3</td>
<td>866</td>
<td>225</td>
<td>20.6%</td>
<td>158</td>
<td>1249</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Total</td>
<td>33777</td>
<td>2617304</td>
<td>721972</td>
<td>21.6%</td>
<td>660832</td>
<td>4000108</td>
</tr>
</tbody>
</table>
**ohcount: Line Count of gcc-4.6.2/gcc**

<table>
<thead>
<tr>
<th>Language</th>
<th>Files</th>
<th>Code</th>
<th>Comment</th>
<th>Comment %</th>
<th>Blank</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>15943</td>
<td>1468358</td>
<td>322727</td>
<td>18.0%</td>
<td>325166</td>
<td>2116251</td>
</tr>
<tr>
<td>ada</td>
<td>4606</td>
<td>678570</td>
<td>315461</td>
<td>31.7%</td>
<td>233952</td>
<td>1227983</td>
</tr>
<tr>
<td>cpp</td>
<td>8845</td>
<td>255337</td>
<td>61446</td>
<td>19.4%</td>
<td>67828</td>
<td>384611</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2900</td>
<td>94766</td>
<td>1901</td>
<td>2.0%</td>
<td>13490</td>
<td>110157</td>
</tr>
<tr>
<td>assembler</td>
<td>137</td>
<td>31551</td>
<td>7446</td>
<td>19.1%</td>
<td>4856</td>
<td>43853</td>
</tr>
<tr>
<td>autoconf</td>
<td>3</td>
<td>28791</td>
<td>12</td>
<td>0.0%</td>
<td>4020</td>
<td>32823</td>
</tr>
<tr>
<td>objective_c</td>
<td>864</td>
<td>27500</td>
<td>4845</td>
<td>15.0%</td>
<td>7976</td>
<td>40321</td>
</tr>
<tr>
<td>fortranfree</td>
<td>793</td>
<td>13042</td>
<td>2993</td>
<td>18.7%</td>
<td>1628</td>
<td>17663</td>
</tr>
<tr>
<td>scheme</td>
<td>6</td>
<td>11023</td>
<td>1010</td>
<td>8.4%</td>
<td>1205</td>
<td>13238</td>
</tr>
<tr>
<td>make</td>
<td>4</td>
<td>6087</td>
<td>1070</td>
<td>15.0%</td>
<td>893</td>
<td>8050</td>
</tr>
<tr>
<td>tex</td>
<td>1</td>
<td>5441</td>
<td>2835</td>
<td>34.3%</td>
<td>702</td>
<td>8978</td>
</tr>
<tr>
<td>ocaml</td>
<td>6</td>
<td>2814</td>
<td>576</td>
<td>17.0%</td>
<td>378</td>
<td>3768</td>
</tr>
<tr>
<td>shell</td>
<td>16</td>
<td>1980</td>
<td>597</td>
<td>23.2%</td>
<td>338</td>
<td>2915</td>
</tr>
<tr>
<td>awk</td>
<td>9</td>
<td>1635</td>
<td>323</td>
<td>16.5%</td>
<td>251</td>
<td>2209</td>
</tr>
<tr>
<td>perl</td>
<td>3</td>
<td>866</td>
<td>225</td>
<td>20.6%</td>
<td>158</td>
<td>1249</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>34186</td>
<td>2627919</td>
<td>723467</td>
<td>21.6%</td>
<td>662858</td>
<td>4014244</td>
</tr>
<tr>
<td>Language</td>
<td>Files</td>
<td>Code</td>
<td>Comment</td>
<td>Comment %</td>
<td>Blank</td>
<td>Total</td>
</tr>
<tr>
<td>------------------</td>
<td>-------</td>
<td>--------</td>
<td>---------</td>
<td>-----------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>c</td>
<td>15943</td>
<td>1468358</td>
<td>322727</td>
<td>18.0%</td>
<td>325166</td>
<td>2116251</td>
</tr>
<tr>
<td>ada</td>
<td>4606</td>
<td>678570</td>
<td>315461</td>
<td>31.7%</td>
<td>233952</td>
<td>1227983</td>
</tr>
<tr>
<td>cpp</td>
<td>8845</td>
<td>255337</td>
<td>61446</td>
<td>19.4%</td>
<td>67828</td>
<td>384611</td>
</tr>
<tr>
<td>fortranfixed</td>
<td>2900</td>
<td>94766</td>
<td>1901</td>
<td>2.0%</td>
<td>13490</td>
<td>110157</td>
</tr>
<tr>
<td>assembler</td>
<td>137</td>
<td>31551</td>
<td>7446</td>
<td>19.1%</td>
<td>4856</td>
<td>43853</td>
</tr>
<tr>
<td>autoconf</td>
<td>3</td>
<td>28791</td>
<td>12</td>
<td>0.0%</td>
<td>4020</td>
<td>32823</td>
</tr>
<tr>
<td>objective_c</td>
<td>864</td>
<td>27500</td>
<td>4845</td>
<td>15.0%</td>
<td>7976</td>
<td>40321</td>
</tr>
<tr>
<td>fortranfree</td>
<td>793</td>
<td>13042</td>
<td>2993</td>
<td>18.7%</td>
<td>1628</td>
<td>17663</td>
</tr>
<tr>
<td>scheme</td>
<td>6</td>
<td>11023</td>
<td>1010</td>
<td>8.4%</td>
<td>1205</td>
<td>13238</td>
</tr>
<tr>
<td>make</td>
<td>4</td>
<td>6087</td>
<td>1070</td>
<td>15.0%</td>
<td>893</td>
<td>8050</td>
</tr>
<tr>
<td>tex</td>
<td>1</td>
<td>5441</td>
<td>2835</td>
<td>34.3%</td>
<td>702</td>
<td>8978</td>
</tr>
<tr>
<td>ocaml</td>
<td>6</td>
<td>2814</td>
<td>576</td>
<td>17.0%</td>
<td>378</td>
<td>3768</td>
</tr>
<tr>
<td>shell</td>
<td>16</td>
<td>1980</td>
<td>597</td>
<td>23.2%</td>
<td>338</td>
<td>2915</td>
</tr>
<tr>
<td>awk</td>
<td>9</td>
<td>1635</td>
<td>323</td>
<td>16.5%</td>
<td>251</td>
<td>2209</td>
</tr>
<tr>
<td>perl</td>
<td>3</td>
<td>866</td>
<td>225</td>
<td>20.6%</td>
<td>158</td>
<td>1249</td>
</tr>
<tr>
<td>haskell</td>
<td>49</td>
<td>153</td>
<td>0</td>
<td>0.0%</td>
<td>17</td>
<td>170</td>
</tr>
<tr>
<td>matlab</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0.0%</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Total</td>
<td>34186</td>
<td>2627919</td>
<td>723467</td>
<td>21.6%</td>
<td>662858</td>
<td>4014244</td>
</tr>
</tbody>
</table>

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Why is Understanding GCC Difficult?

Deeper technical reasons

• GCC is not a compiler but a *compiler generation framework*
  Two distinct gaps that need to be bridged
  ▶ Input-output of the generation framework
    The target specification and the generated compiler
  ▶ Input-output of the generated compiler
    A source program and the generated assembly program

• GCC generated compiler uses a derivative of the Davidson-Fraser model of compilation
  ▶ Early instruction selection
  ▶ Machine dependent intermediate representation
  ▶ Simplistic instruction selection and retargetability mechanism
The Architecture of GCC

Compiler Generation Framework

Language Specific Code

Language and Machine Independent Generic Code

Machine Dependent Generator Code

Machine Descriptions
The Architecture of GCC

Compiler Generation Framework

- Language Specific Code
- Language and Machine Independent Generic Code
- Machine Dependent Generator Code
- Machine Descriptions

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
The Architecture of GCC

Input Language → Compiler Generation Framework → Target Name

- Language Specific Code
- Language and Machine Independent Generic Code
- Machine Dependent Generator Code
- Machine Descriptions

Selected → Copied → Copied → Generated

- Parser
- Gimplifier
- Tree SSA Optimizer
- Expander
- Optimizer
- Recognizer

Source Program → Generated Compiler (cc1) → Assembly Program
The Architecture of GCC

- **Input Language**
  - Language Specific Code
  - Language and Machine Independent Generic Code
  - Machine Dependent Generator Code
  - Machine Descriptions

- **Target Name**
  - Compiler Generation Framework
  - Parser
  - Gimplifier
  - Tree SSA Optimizer
  - Expander
  - Optimizer
  - Recognizer

- **Development Time**
  - Build Time
  - Use Time

- **Generated Compiler (cc1)**
  - Source Program
  - Assembly Program

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
An Example of The Generation Related Gap

- Predicate function for invoking the loop distribution pass

```c
static bool
gate_tree_loop_distribution (void)
{
    return flag_tree_loop_distribution != 0;
}
```
An Example of The Generation Related Gap

- Predicate function for invoking the loop distribution pass

```c
static bool
gate_tree_loop_distribution (void)
{
    return flag_tree_loop_distribution != 0;
}
```

- There is no declaration of or assignment to variable `flag_tree_loop_distribution` in the entire source!
An Example of The Generation Related Gap

- Predicate function for invoking the loop distribution pass

```c
static bool
gate_tree_loop_distribution (void)
{
    return flag_tree_loop_distribution != 0;
}
```

- There is no declaration of or assignment to variable `flag_tree_loop_distribution` in the entire source!

- It is described in `common.opt` as follows

```
ftree-loop-distribution
Common Report Var(flag_tree_loop_distribution) Optimization
Enable loop distribution on trees
```
An Example of The Generation Related Gap

- Predicate function for invoking the loop distribution pass

```c
static bool gate_tree_loop_distribution (void)
{
    return flag_tree_loop_distribution != 0;
}
```

- There is no declaration of or assignment to variable `flag_tree_loop_distribution` in the entire source!

- It is described in `common.opt` as follows

```
ftree-loop-distribution
Common Report Var(flag_tree_loop_distribution) Optimization
Enable loop distribution on trees
```

- The required C statements are generated during the build
Another Example of The Generation Related Gap

- Locating the main function in gcc-4.6.2/gcc using cscope -R
Another Example of The Generation Related Gap

- Locating the main function in gcc-4.6.2/gcc using cscope -R

7359 occurrences!
Another Example of The Generation Related Gap

- Locating the main function in gcc-4.6.2/gcc using cscope -R
  7359 occurrences!

- Number of main functions in the entire tarball
Another Example of The Generation Related Gap

- Locating the main function in gcc-4.6.2/gcc using cscope -R
  7359 occurrences!

- Number of main functions in the entire tarball
  11777!
Another Example of The Generation Related Gap

- Locating the main function in gcc-4.6.2/gcc using cscope -R
  7359 occurrences!

- Number of main functions in the entire tarball
  11777!

- What if we do not search recursively?
Another Example of The Generation Related Gap
Locating the main function in the directory gcc-4.6.2/gcc using cscape
Another Example of The Generation Related Gap
Locating the main function in the directory gcc-4.6.2/gcc using cscoop

<table>
<thead>
<tr>
<th>File</th>
<th>Line</th>
<th>Main Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>collect2.c</td>
<td>1076</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>fp-test.c</td>
<td>85</td>
<td>main (void )</td>
</tr>
<tr>
<td>gcc.c</td>
<td>6092</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>gcov-dump.c</td>
<td>76</td>
<td>main (int argc ATTRIBUTE_UNUSED, char **argv)</td>
</tr>
<tr>
<td>gcov-iov.c</td>
<td>29</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>gcov.c</td>
<td>360</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genattr.c</td>
<td>164</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genattrtab.c</td>
<td>4820</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genautomata.c</td>
<td>9459</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genchecksum.c</td>
<td>97</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>gencodes.c</td>
<td>51</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genconditions.c</td>
<td>209</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genconfig.c</td>
<td>261</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genconstants.c</td>
<td>79</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genemit.c</td>
<td>830</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genenums.c</td>
<td>48</td>
<td>main (int argc, char **argv)</td>
</tr>
</tbody>
</table>
### Another Example of The Generation Related Gap

Locating the main function in the directory `gcc-4.6.2/gcc` using cscope

<table>
<thead>
<tr>
<th>File</th>
<th>Line No.</th>
<th>Function Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>genextract.c</td>
<td>402</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genflags.c</td>
<td>251</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>gengenrtl.c</td>
<td>282</td>
<td>main (void)</td>
</tr>
<tr>
<td>gengtype.c</td>
<td>4825</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genhooks.c</td>
<td>335</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genmddeps.c</td>
<td>43</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genmodes.c</td>
<td>1376</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genopinit.c</td>
<td>473</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genoutput.c</td>
<td>999</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genpeep.c</td>
<td>353</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genpreds.c</td>
<td>1388</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>genrecog.c</td>
<td>2691</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>lto-wrapper.c</td>
<td>628</td>
<td>main (int argc, char *argv[])</td>
</tr>
<tr>
<td>main.c</td>
<td>34</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>mips-tdump.c</td>
<td>1393</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>mips-tfile.c</td>
<td>655</td>
<td>main (void)</td>
</tr>
<tr>
<td>mips-tfile.c</td>
<td>4693</td>
<td>main (int argc, char **argv)</td>
</tr>
<tr>
<td>tlink.c</td>
<td>64</td>
<td>const char *main;</td>
</tr>
</tbody>
</table>
GCC Retargetability Mechanism

Input Language

Compiler Generation Framework

Language Specific Code

Language and Machine Independent Generic Code

Machine Dependent Generator Code

Machine Descriptions

Target Name

Selected

Copied

Copied

Generated

Generated

Generated Compiler

Parser

Gimplifier

Tree SSA Optimizer

Expander

Optimizer

Recognizer

Development Time

Build Time

Use Time

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
GCC Retargetability Mechanism

Input Language

Compiler Generation Framework

Language Specific Code

Language and Machine Independent Generic Code

Machine Dependent Generator Code

Machine Descriptions

Target Name

Development Time

Build Time

Use Time

Selected

Copied

Copied

Generated

Generated

Gimple → IR-RTL

IR-RTL → ASM

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
GCC Retargetability Mechanism

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
GCC Retargetability Mechanism

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
GCC Retargetability Mechanism

Compiler Generation Framework

Input Language

Language Specific Code

Language and Machine Independent Generic Code

Machine Dependent Generator Code

Machine Descriptions

Target Name

Parser

Gimplifier

Tree SSA Optimizer

Expander

Optimizer

Recognizer

Generated Compiler

Gimple → PN

PN → IR-RTL

IR-RTL → ASM

Gimple → IR-RTL

IR-RTL → ASM

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
The generated compiler uses an adaptation of the Davison Fraser model:

- Generic expander and recognizer
- Machine specific information is isolated in data structures
- Generating a compiler involves generating these data structures
The GCC Challenge: Poor Retargetability Mechanism

Symptoms:

- Machine descriptions are large, verbose, repetitive, and contain large chunks of C code

Size in terms of line counts in gcc-4.6.2 (counted using wc -l)

<table>
<thead>
<tr>
<th>Files</th>
<th>i386</th>
<th>mips</th>
<th>arm</th>
</tr>
</thead>
<tbody>
<tr>
<td>*.md</td>
<td>38851</td>
<td>15534</td>
<td>30951</td>
</tr>
<tr>
<td>*.c</td>
<td>39780</td>
<td>16793</td>
<td>26165</td>
</tr>
<tr>
<td>*.h</td>
<td>17879</td>
<td>5667</td>
<td>18713</td>
</tr>
<tr>
<td>Total</td>
<td>96510</td>
<td>37996</td>
<td>75929</td>
</tr>
</tbody>
</table>
The GCC Challenge: Poor Retargetability Mechanism

Symptoms:

- Machine descriptions are large, verbose, repetitive, and contain large chunks of C code

Size in terms of line counts in gcc-4.6.2 (counted using `wc -l`)

<table>
<thead>
<tr>
<th>Files</th>
<th>i386</th>
<th>mips</th>
<th>arm</th>
</tr>
</thead>
<tbody>
<tr>
<td>*.md</td>
<td>38851</td>
<td>15534</td>
<td>30951</td>
</tr>
<tr>
<td>*.c</td>
<td>39780</td>
<td>16793</td>
<td>26165</td>
</tr>
<tr>
<td>*.h</td>
<td>17879</td>
<td>5667</td>
<td>18713</td>
</tr>
<tr>
<td>Total</td>
<td>96510</td>
<td>37996</td>
<td>75929</td>
</tr>
</tbody>
</table>

- Machine descriptions are difficult to construct, understand, debug, and enhance
# Meeting the GCC Challenge

<table>
<thead>
<tr>
<th>Goal of Understanding</th>
<th>Methodology</th>
<th>Needs Examining</th>
</tr>
</thead>
<tbody>
<tr>
<td>Translation sequence of programs</td>
<td>Gray box probing</td>
<td>No</td>
</tr>
<tr>
<td>Build process</td>
<td>Customizing the configuration and building</td>
<td>Yes</td>
</tr>
<tr>
<td>Retargetability issues and machine descriptions</td>
<td>Incremental construction of machine descriptions</td>
<td>No</td>
</tr>
<tr>
<td>IR data structures and access mechanisms</td>
<td>Adding passes to massage IRs</td>
<td>No</td>
</tr>
<tr>
<td>Retargetability mechanism</td>
<td></td>
<td>Yes</td>
</tr>
</tbody>
</table>

- **Makefiles**: No
- **Source**: No
- **MD**: No

---

**Essential Abstractions in GCC**

GCC Resource Center, IIT Bombay
Workshop Coverage

Compiler Specifications

Compiler Generator

Generated Compiler

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Workshop Coverage

<table>
<thead>
<tr>
<th>External View</th>
<th>Internal View</th>
</tr>
</thead>
</table>

- Compiler Specifications
  - Compiler Generator
    - Generated Compiler
Workshop Coverage

Compiler Specifications

Compiler Generator

Generated Compiler

External View

Gray box probing

Internal View
Workshop Coverage

<table>
<thead>
<tr>
<th>Compiler Specifications</th>
<th>External View</th>
<th>Internal View</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compiler Generator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Generated Compiler</td>
<td>Gray box probing Pass structure and IR</td>
<td></td>
</tr>
</tbody>
</table>
Workshop Coverage

External View

Compiler Specifications

Compiler Generator

Configuration and building

Internal View

Generated Compiler

Gray box probing

Pass structure and IR

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Overview: GCC ≡ The Great Compiler Challenge

Workshop Coverage

- Compiler Specifications
  - Compiler Generator
  - Generated Compiler
- External View
  - Configuration and building
  - Gray box probing
  - Pass structure and IR
- Internal View
  - Front end hooks

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Workshop Coverage

Compiler Specifications

- Compiler Generator

External View

- Configuration and building

Generated Compiler

Gray box probing

Pass structure and IR

Internal View

Pass structure

Front end hooks
Workshop Coverage

Compiler Specifications

Compiler Generator

Generated Compiler

External View

Configuration and building

Pass structure and IR

Gray box probing

Internal View

Front end hooks

Pass structure

Control flow

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
**Workshop Coverage**

- **External View**
  - Compiler Specifications
  - Configuration and building
  - Gray box probing
  - Pass structure and IR

- **Internal View**
  - Front end hooks
  - Pass structure
  - Control flow
  - Static and dynamic plugin mechanisms

---

**Essential Abstractions in GCC**

GCC Resource Center, IIT Bombay
Workshop Coverage

- Compiler Specifications
  - Compiler Generator
    - Generated Compiler
      - Gray box probing
      - Pass structure and IR
      - Data Flow Analysis
  - Configuration and building
    - Front end hooks
  - Pass structure
    - Control flow
    - Static and dynamic plugin mechanisms

Essential Abstractions in GCC
Workshop Coverage

- Compiler Specifications
  - Compiler Generator
    - Generated Compiler
  - Configuration and building
    - Gray box probing
      - Pass structure and IR
      - Data Flow Analysis
    - Front end hooks
      - Pass structure
      - Control flow
      - Static and dynamic plugin mechanisms

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Workshop Coverage

Compiler Specifications

Compiler Generator

Generated Compiler

External View

Machine descriptions

Configuration and building

Gray box probing
Pass structure and IR
Data Flow Analysis

Internal View

Front end hooks

Retargetability mechanism

Pass structure
Control flow
Static and dynamic plugin mechanisms

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Workshop Coverage

Compiler Specifications

- Compiler Generator

Generated Compiler

- Gray box probing
- Pass structure and IR
- Data Flow Analysis
- Parallelization, Vectorization

External View

- Machine descriptions

Internal View

- Front end hooks
- Retargetability mechanism

Pass structure
- Control flow
- Static and dynamic plugin mechanisms

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay