



# Birla Institute of Technology & Science, Pilani

Pilani Campus

INSTRUCTION DIVISION  
FIRST SEMESTER 2016-2017  
Course Handout Part II

In addition to part-I (General Handout for all courses appended to the time table) this portion gives further specific details regarding the course

**Course No. :** CS G551 / SS G551  
**Course Title :** Advanced Compilation Techniques

**Instructor-in-Charge:** Shan Sundar Balasubramaniam (email: [sundarb](mailto:sundarb))  
**Instructors:**

## 1. a. Scope and Objective:

The objective of the course is to provide the student a project-based hands-on experience with modern compilers and runtime environments. The scope of this course includes: specific techniques for code analysis and low level code optimizations; automatic memory management; virtual machine implementations and just-in-time compilation; and compilers for parallel programs.

## b. Learning Outcome

On completion of this course the student shall be able to  
*design and implement compiler or run-time modules for production quality compilers / environments*

## 2. Reading:

### (a) Text Book:

**T1. Steven S. Muchnick.** *Advanced Compiler Design & Implementation*. Morgan Kauffman (Elsevier India) 1997

### (b) References:

**R1. Aho, A., Monica Lam, Ravi Sethi, Jeffrey Ullman.** *Compilers – Principles, Techniques and Tools*. 2<sup>nd</sup> Edition. Pearson 2009.

**R2. Andrew Appel.** *Modern Compiler Implementation in C*. Revised Edition. Cambridge 2000.

**AR** Additional readings as assigned (and/or made available online) by the instructor





# Birla Institute of Technology & Science, Pilani

Pilani Campus

## 3. Course Plan:

### 3.a. Modules

Module	Theme	Prior Preparation
REV	Review of basics	-
OPT	Optimization – Register Allocation and (Low Level) Code Optimization Techniques	Basic Structure of a compiler. Code Generation. Von Neumann architecture.
GC	Dynamic Memory Management - Garbage Collection Techniques	Basic Data Structures – Linked Lists and Graph traversals. Runtime Memory Models and Environments.
PAR	Parallelization – Compilers for languages/machines supporting concurrency	Parallel Programming Basics; Synchronization / Communication Issues. Modern Computer Architectures.
RUN	Runtime Environments, Managed Execution - Virtual Machines, Just-in-Time Compilers	Basic Structure of a compiler. Code Generation. Von Neumann architecture

### 3. b. Lecture Schedule:

Lecture #	Module #	Topic	Learning Outcome(s) [The student will be able to: ]	Reading
L1	REV	Introduction and Motivation. Course Administration	-	Any under-graduate text on Compilers
L2-3	REV	Review of Compilation and Execution Models; Performance Model.	-	
L4	RUN	Virtual Machines and JIT Compilers	<ul style="list-style-type: none"> <li>State the relation between IL and VM.</li> <li>Estimate when JIT would be useful</li> </ul>	-





# Birla Institute of Technology & Science, Pilani

Pilani Campus

			<ul style="list-style-type: none"> <li>Identify modules of a JIT compiler</li> </ul>	
L5-6	REV	Review of Runtime Memory Layout Models	<ul style="list-style-type: none"> <li>Relate features of a language to runtime memory layout</li> <li>Design a runtime memory layout for a given language</li> </ul>	R1. Sec. 7.1-7.3
L7	RUN	Runtime Memory Layout for advanced language features	<ul style="list-style-type: none"> <li>Adapt a runtime memory layout for new advanced features</li> </ul>	AR
L8-9	RUN	Runtime Data Organization	<ul style="list-style-type: none"> <li>Identify how data of each type is organized in runtime memory</li> </ul>	R2 Ch.14
L10-12	RUN	Introduction to Garbage Collection – Basic Techniques: Reference Counting, Mark and Sweep;  Issues: Response Time, Throughput, and Fragmentation; Improvements: Compaction, and Copying.	<ul style="list-style-type: none"> <li>Explain how basic GC techniques work.</li> <li>State and argue typical issues with design and implementation of GC.</li> </ul>	AR
L13-15	RUN	Incremental and Concurrent Garbage Collectors – Design and Implementation Issues.	<ul style="list-style-type: none"> <li>Explain and illustrate issues with incremental and concurrent GCs.</li> <li>Design concurrent GCs.</li> </ul>	AR
L16-17	RUN	Generational Collectors – Design and Implementation Issues.	<ul style="list-style-type: none"> <li>Explain and illustrate issues with incremental and concurrent GCs.</li> <li>Design concurrent GCs.</li> </ul>	AR
L18	RUN	GC for modern environments – Case Study.	-	AR
L19-20	OPT	Overview of Code Optimization: Objectives, Issues, and approaches. Typical Early Optimizations and Procedure Optimizations.	<ul style="list-style-type: none"> <li>State typical objectives of code optimization.</li> <li>Identify typical early optimizations and procedure optimizations used in a given compiler</li> </ul>	T1. Ch. 11, 12, & 15
L21-22	OPT	Liveness Analysis and Register Allocation.	<ul style="list-style-type: none"> <li>Design and implement liveness analysis of a given piece of code.</li> <li>Design and implement a naïve register allocator.</li> </ul>	R2. Ch. 10 & 12
L23-24	OPT, RUN	Modern Register Allocation Techniques – Tradeoff in Compile-Time vs. Run-Time, Register Allocation in JIT compilers	<ul style="list-style-type: none"> <li>Compare and contrast register allocation techniques.</li> <li>Design a register allocator for a JIT compiler.</li> </ul>	AR





# Birla Institute of Technology & Science, Pilani

Pilani Campus

L25-26	OPT	Instruction Scheduling for Superscalar Architectures	<ul style="list-style-type: none"> <li>Design and implement a rudimentary instruction scheduler</li> </ul>	T1 Sec. 17.1 – 17.3
L27-28	OPT	Software Pipelining	<ul style="list-style-type: none"> <li>Explain and illustrate how software pipelining works</li> <li>Perform software pipelining on code</li> </ul>	T1 Sec. 17.4
L29	OPT	Code Scheduling and Energy Constraints	-	AR
L30-31	PAR	Compiling for Parallel and Distributed Systems – Models and Examples	<ul style="list-style-type: none"> <li>State and illustrate typical issues in compiling for parallel/distributed systems.</li> <li>State and illustrate how data parallel code can be generated</li> </ul>	AR
L32-33	OPT, PAR	Optimizing for Parallelism – Arrays and Loops	<ul style="list-style-type: none"> <li>Illustrate when and how a loop can be parallelized.</li> <li>Design a loop optimizer for typical loops of arrays.</li> </ul>	R1 Sec. 11.1 to 11.4
L34-35	OPT, PAR	Data Reuse and Optimizing for Locality	<ul style="list-style-type: none"> <li>State and illustrate optimizations applicable for improving locality.</li> </ul>	R1 Sec. 11.5 & 11.10 T1 Ch. 20
L36-37	PAR	Code Generation based on Parallel Patterns	<ul style="list-style-type: none"> <li>Identify parallel patterns from code and illustrate how they can be translated to parallel code.</li> </ul>	AR
L38-39	RUN	Managed Execution and JIT Compilers – Implementation Issues	<ul style="list-style-type: none"> <li>State typical issues in managed execution</li> <li>Design a runtime environment using a JIT compiler.</li> </ul>	AR
L40	-	Course Summary		

## 4. Project

A term project running through the entire semester will be used as an active, hands-on learning medium. The project will require understanding and working with a large compiler codebase, as well as presentations and teamwork. The project will involve design and implementation of compiler module(s) in a real-life / production-quality compiler. It is expected that the students interact frequently with the instructor while working on the project.





# Birla Institute of Technology & Science, Pilani

## Pilani Campus

### 4.a. Project Components

Student Teams of 2 or 3 will choose a project in consultation with the instructor. The project themes are to be aligned one of the four course modules (see 4.a). Each project will include the following components:

Component	Work Description
PC1	Code Base Analysis: Read, analyze, and present the code base of an existing compiler.
PC2	Problem Definition: Read literature, choose a problem, scope the work, and define the problem in terms of objectives, environment, deliverables and testing/validation.
PC3	Design: Identify issues in solving the problem, propose a detailed design with reference to the chosen code base, outline implementation strategies/techniques, and develop test cases.
PC4	Implementation: Implement your design in the chosen code base, and test/validate it.

### 4.d. Project Presentations

Component	Description	Mode / Deliverables
PC1	Code Analysis	Presentation
PC2	Problem Definition & Scoping	Problem Document & Presentation
PC3	Design	Design Document & Presentation
PC4	Implementation	Codebase, Demonstration, & Presentation.

**Total class meeting time for Presentations:  $(10 \times 75)/50 = 15$  sessions (outside of the 40 lectures).**

### 5.a. Evaluation Scheme:





# Birla Institute of Technology & Science, Pilani

## Pilani Campus

Component	Mode	Duration	Date&Time	Weight
Interaction	In-class	-	-	15%
PC1	Take home & In-class Presentation	2 weeks	20 <sup>th</sup> Aug.	7%
PC2	Take home, Problem Document, & Presentation	4 weeks (concurrent with PC1)	3 <sup>rd</sup> Sep.	8%
PC3	Take Home, Design Document, & Presentation	3 weeks	26 <sup>th</sup> Sep-28 <sup>th</sup> Sep.	10%
Test	Open Book	90 minutes	Scheduled centrally	10%
PC4	Take Home, Codebase & Demo	5 weeks	21 <sup>st</sup> to 26 <sup>th</sup> Nov.	30%
Comprehensive Exam (Lab)	Open Book	3 hours	Scheduled centrally	20%

### 5.b. Evaluation Policies:

(i) While projects may be pursued in teams of 2 or 3, evaluation for the course would be done for each individual wherever applicable. Therefore the teams are required to clearly identify and agree upon individual contributions. The instructor's estimate of the contribution will decide the marks obtained by each student.

(ii) Plagiarism or malpractice of any kind will lead to severe penalties including no grades for the course. Plagiarism includes seeking or obtaining answers/solutions from other students, from sources on the web or other equivalent forms. Wherever legitimate sources are permitted by the instructor and used by the students proper attribution/citation is a must

### 5. b. Make-up Policy:

- **No Make-up will be available for Assignments under any condition.**
- **Late submission of assignment will incur a penalty of 25% per 24 hours after the deadline.**





# Birla Institute of Technology & Science, Pilani

## Pilani Campus

- Prior Permission of the Instructor is usually required to get a make-up for the mid-term test.
- Prior Permission of (Associate) Dean, Instruction is usually required to get a make-up for the comprehensive exam.
- A make-up shall be granted only in genuine cases where - in the Instructor's / Dean's judgment - the student would be physically unable to appear for the quiz/test/exam. Instructor's / Dean's decision in this matter would be final.

#### 4.c. Fairness Policy:

- Student teams are expected to work on their own on assignments.
- All students are expected to contribute equally within a team. The instructor's assessment regarding the contributions of team members would be final.
- Any use of unfair means in quizzes, assignment, or test/exam will be reported to the Unfair means committee and will be subject to the severest penalty applicable:
  - Unfair means would include copying from other students or from the Web or from other sources of information including electronic devices.
  - All parties involved would be treated equally responsible: *allowing others to copy one's work is enabling unfair means and is equally un-acceptable.*

#### 5. Consultation / Office Hours:

- **Tuesdays 2.30pm to 3.30pm and Wednesdays 4.30pm to 5.30pm** in Room 6120-L, CSIS Dept., NAB.
- **Or by appointment via email**

**6. Contents and Notices:** All lecture slides will be posted on the course website on Nalanda. Notices concerning this course will be displayed online (on Nalanda) only. If there is a need, email would be used on short notice (12 hours) – only BITS Pilani mail would be used.

**Instructor-In- Charge, CS G551.**

