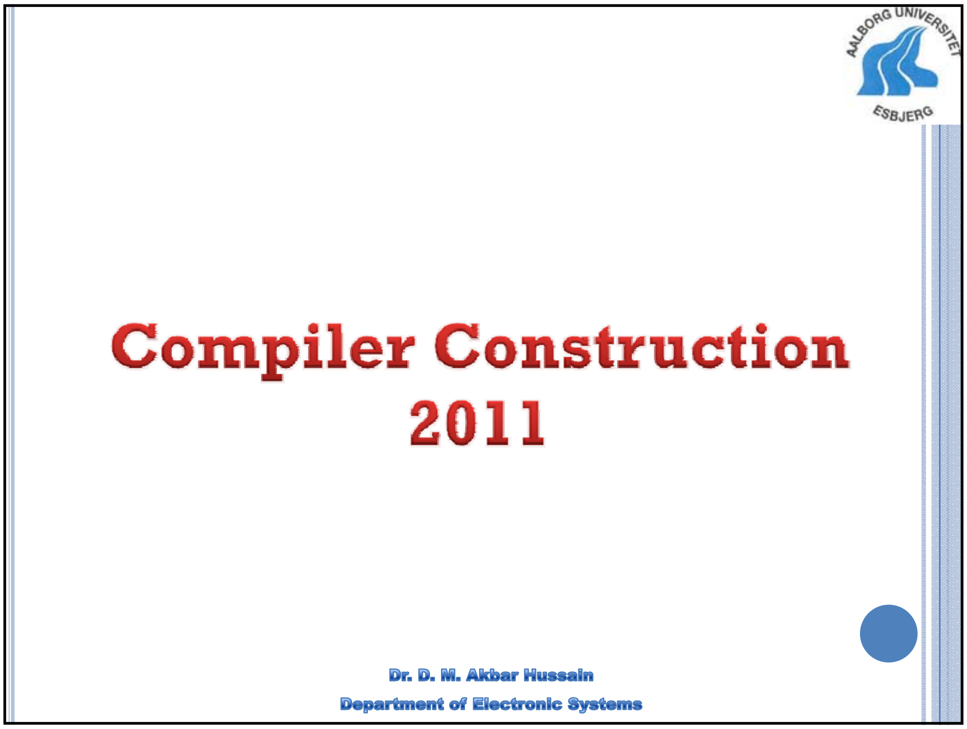
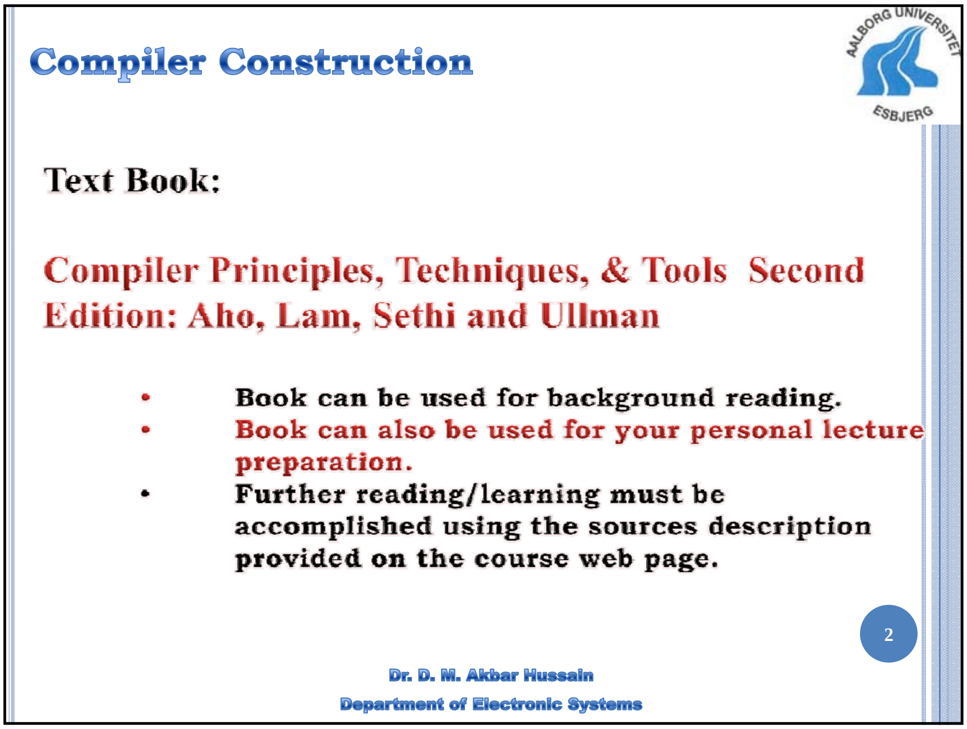


Compiler Construction 2011

Dr. D. M. Akbar Hussain

Department of Electronic Systems



Compiler Construction

Text Book:

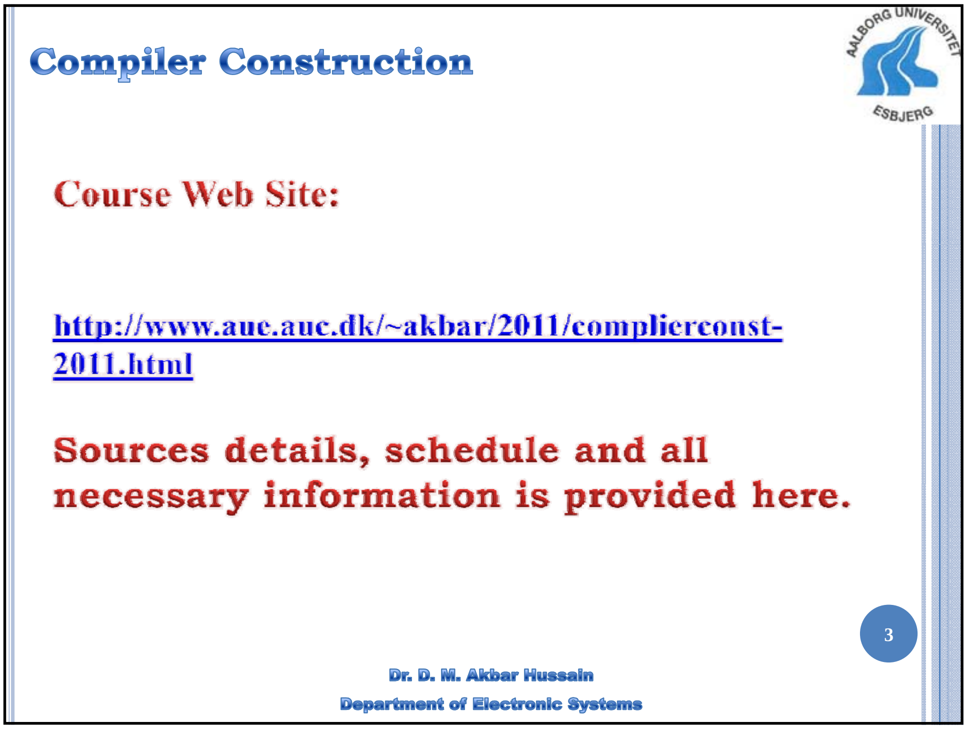
Compiler Principles, Techniques, & Tools Second Edition: Aho, Lam, Sethi and Ullman

- Book can be used for background reading.
- Book can also be used for your personal lecture preparation.
- Further reading/learning must be accomplished using the sources description provided on the course web page.

2

Dr. D. M. Akbar Hussain

Department of Electronic Systems



Compiler Construction

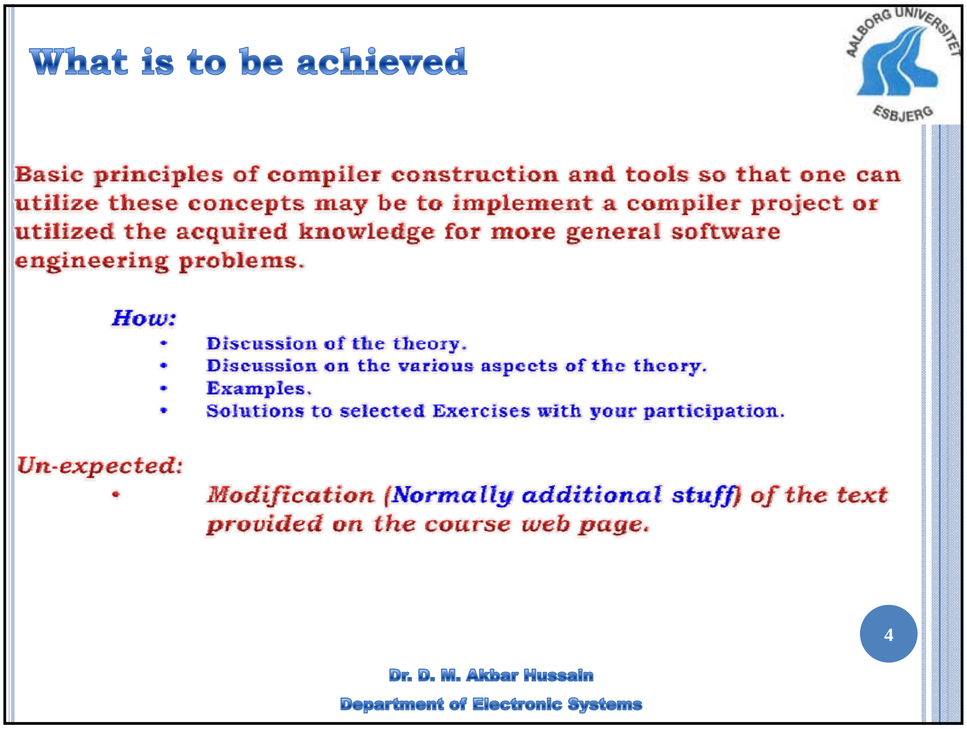
Course Web Site:

http://www.aue.auc.dk/~akbar/2011/complierconst-2011.html

Sources details, schedule and all necessary information is provided here.

3

Dr. D. M. Akbar Hussain
Department of Electronic Systems



What is to be achieved

Basic principles of compiler construction and tools so that one can utilize these concepts may be to implement a compiler project or utilized the acquired knowledge for more general software engineering problems.

How:

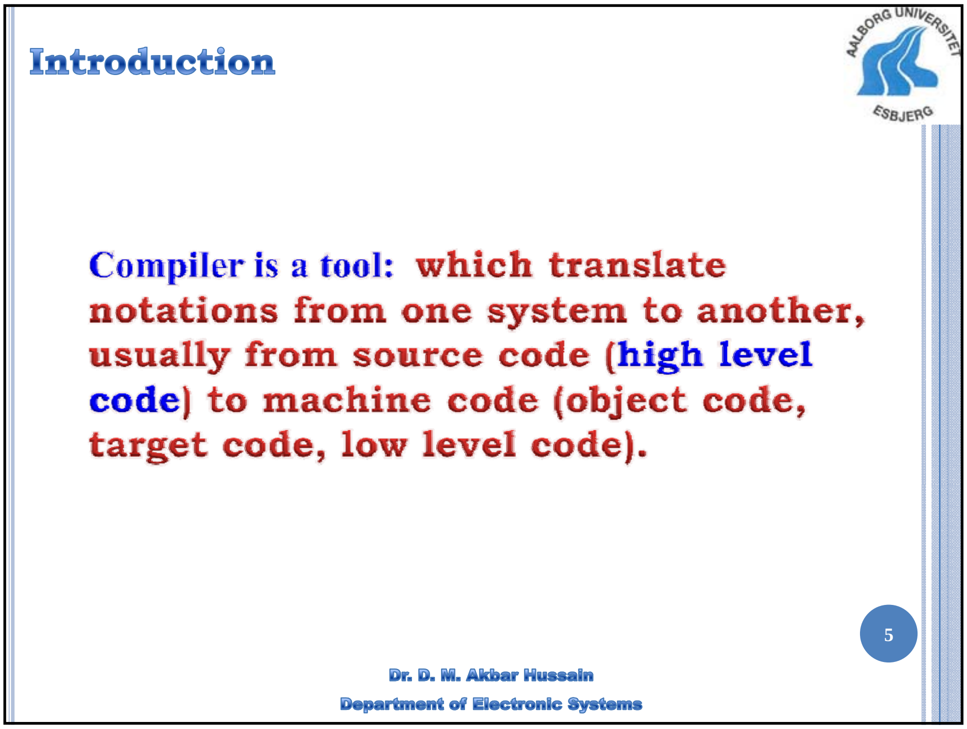
- Discussion of the theory.
- Discussion on the various aspects of the theory.
- Examples.
- Solutions to selected Exercises with your participation.

Un-expected:

- *Modification (Normally additional stuff) of the text provided on the course web page.*

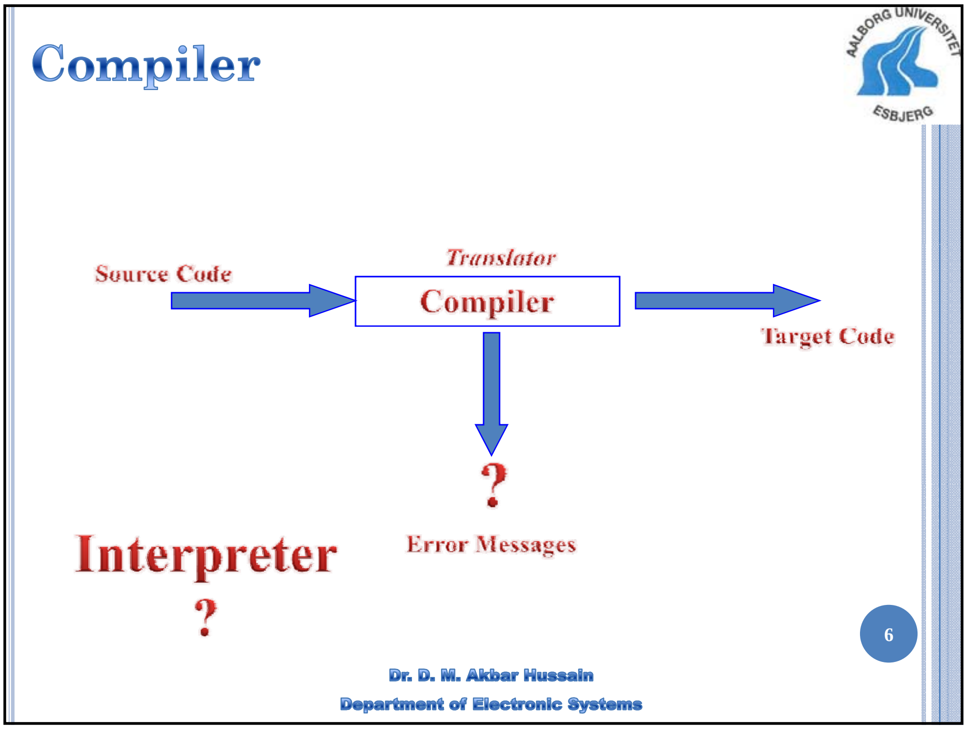
4

Dr. D. M. Akbar Hussain
Department of Electronic Systems



Introduction

Compiler is a tool: which translate notations from one system to another, usually from source code (high level code) to machine code (object code, target code, low level code).



Compiler

Source Code

Translator

Compiler

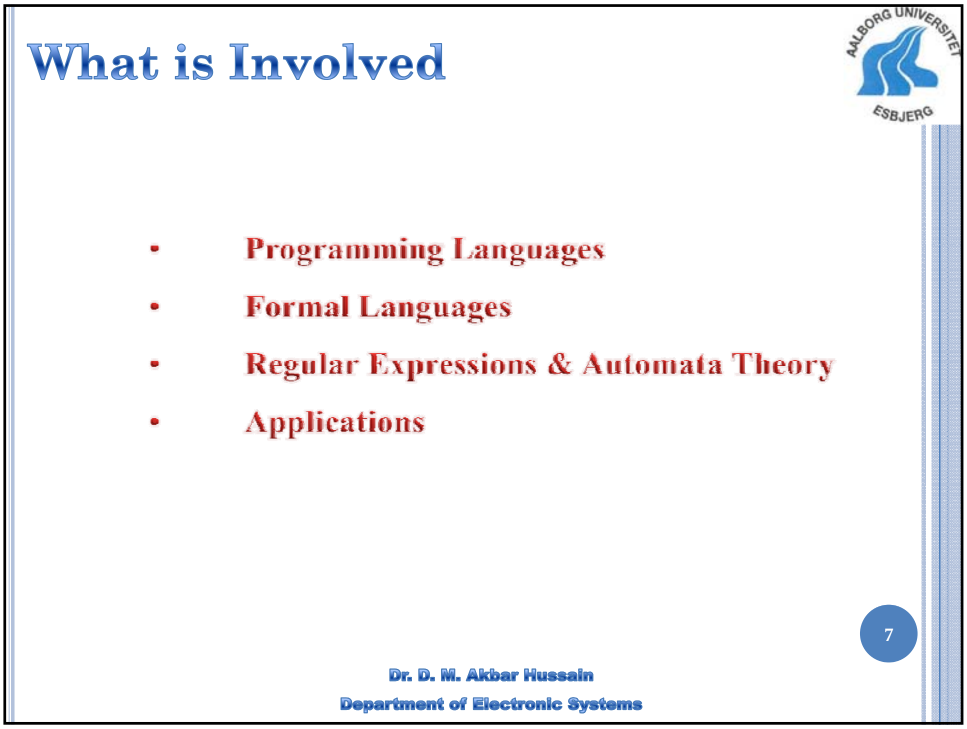
Target Code

?

Error Messages

Interpreter

?

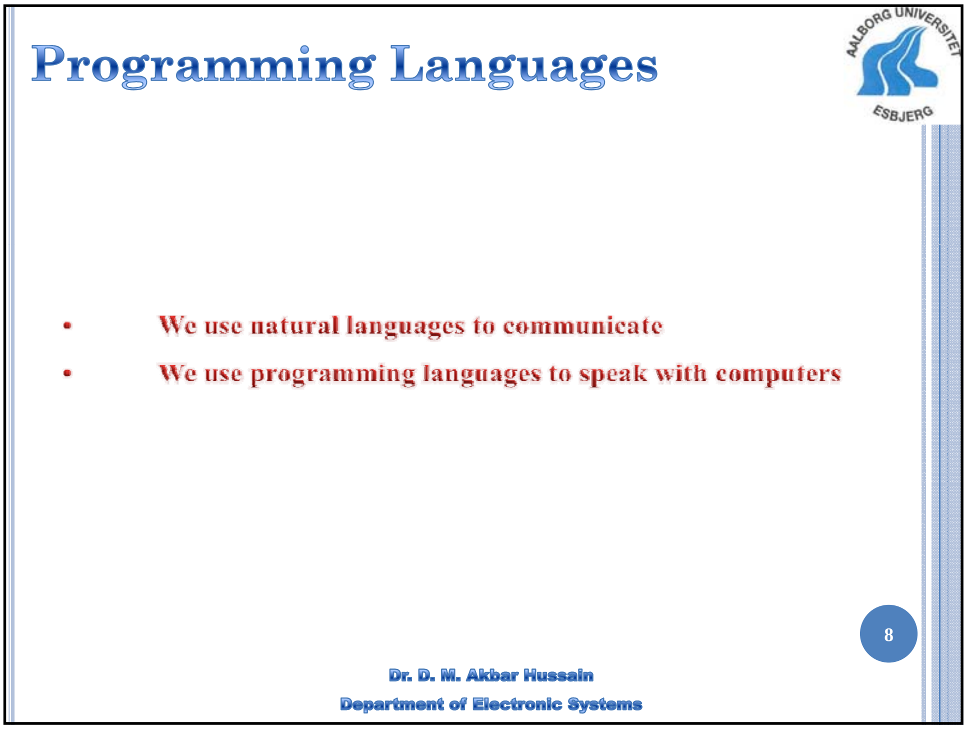


What is Involved

- Programming Languages
- Formal Languages
- Regular Expressions & Automata Theory
- Applications

7

Dr. D. M. Akbar Hussain
Department of Electronic Systems

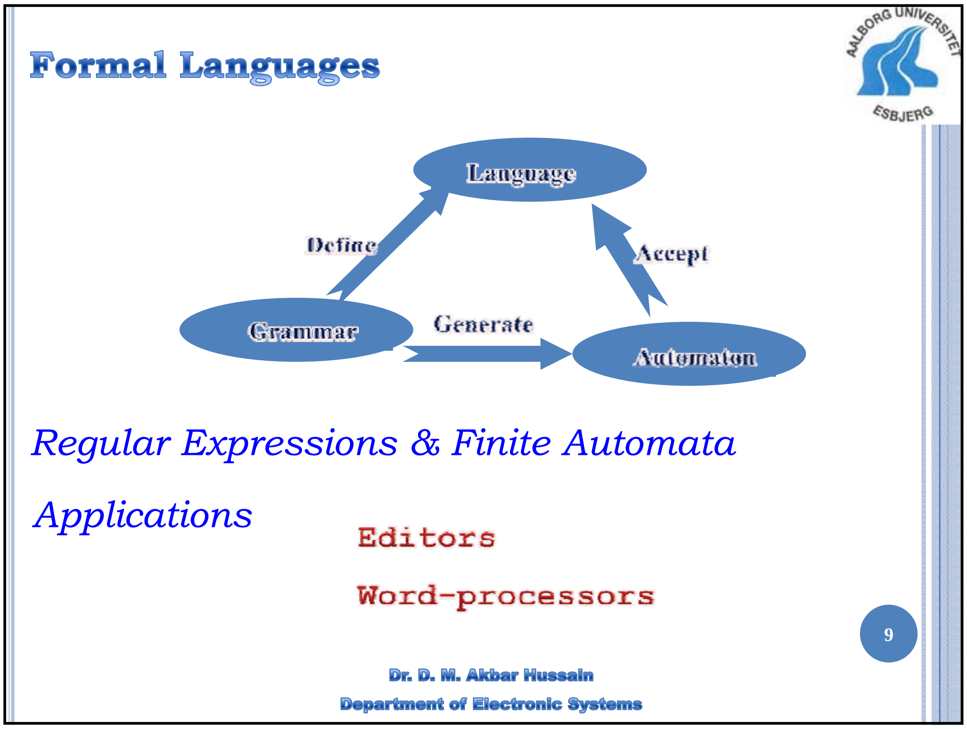


Programming Languages

- We use natural languages to communicate
- We use programming languages to speak with computers

8

Dr. D. M. Akbar Hussain
Department of Electronic Systems



Formal Languages

Language

Define

Accept

Grammar

Generate

Automaton

Regular Expressions & Finite Automata

Applications

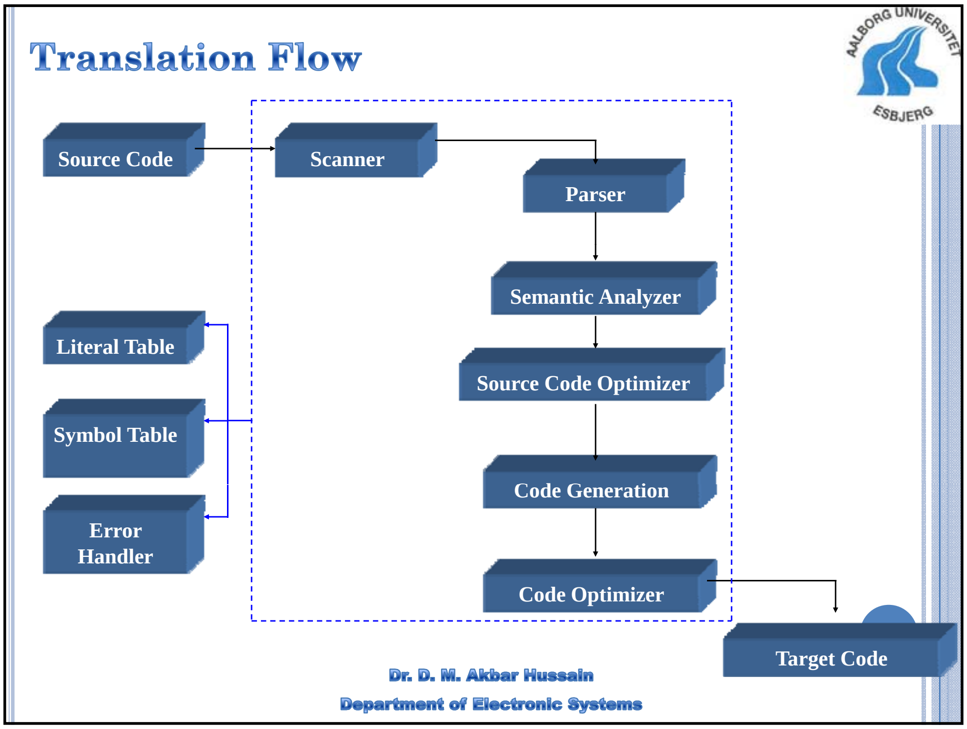
Editors

Word-processors

9

Dr. D. M. Akbar Hussain

Department of Electronic Systems



Translation Flow

Source Code

Scanner

Parser

Semantic Analyzer

Source Code Optimizer

Code Generation

Code Optimizer

Target Code

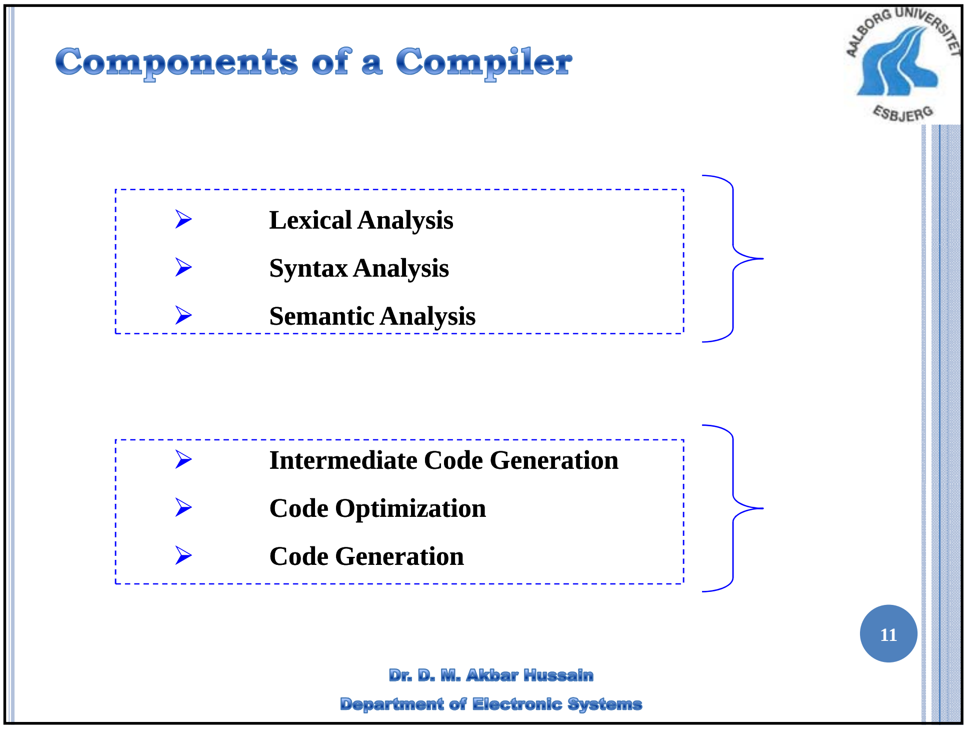
Literal Table

Symbol Table

Error Handler

Dr. D. M. Akbar Hussain

Department of Electronic Systems



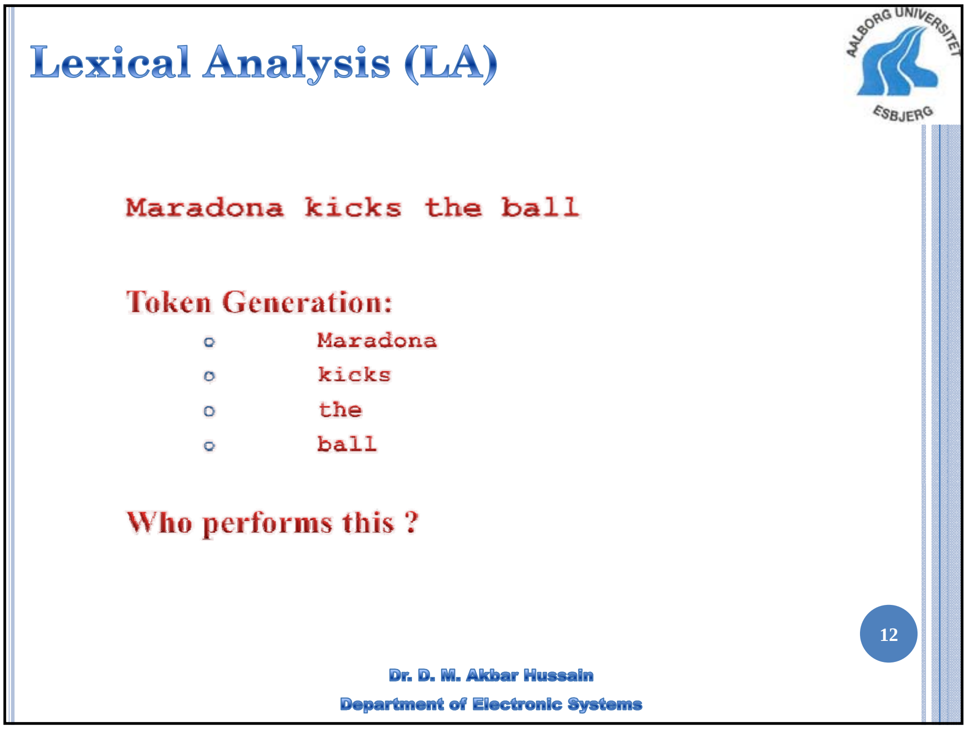
Components of a Compiler

- Lexical Analysis
- Syntax Analysis
- Semantic Analysis

- Intermediate Code Generation
- Code Optimization
- Code Generation

11

Dr. D. M. Akbar Hussain
Department of Electronic Systems



Lexical Analysis (LA)

Maradona kicks the ball

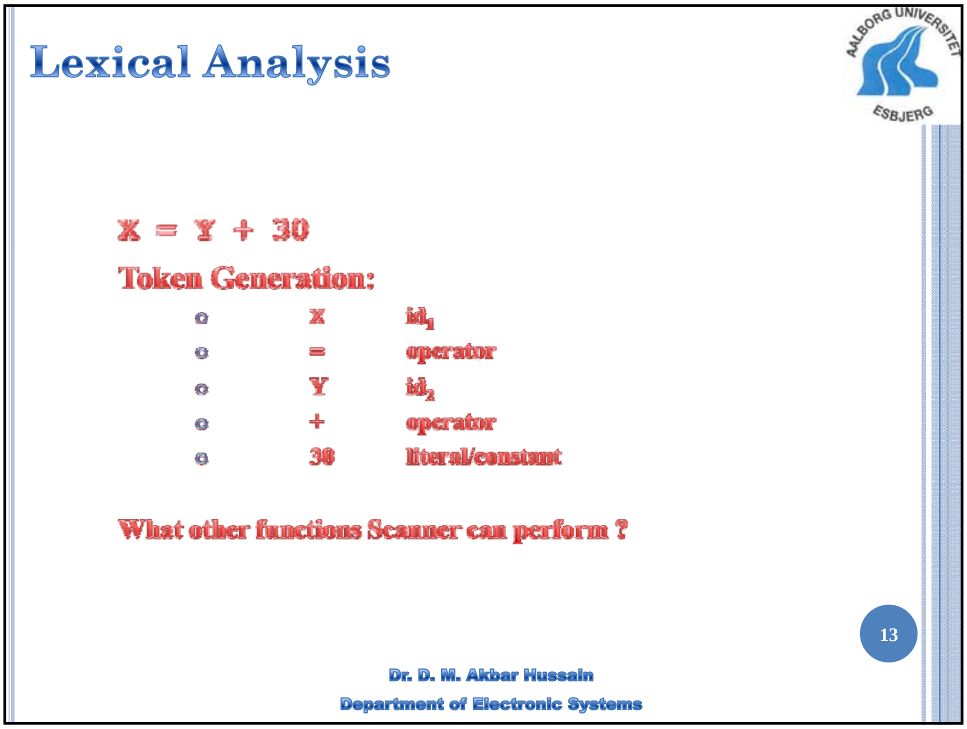
Token Generation:

- Maradona
- kicks
- the
- ball

Who performs this ?

12

Dr. D. M. Akbar Hussain
Department of Electronic Systems



Lexical Analysis

X = Y + 30

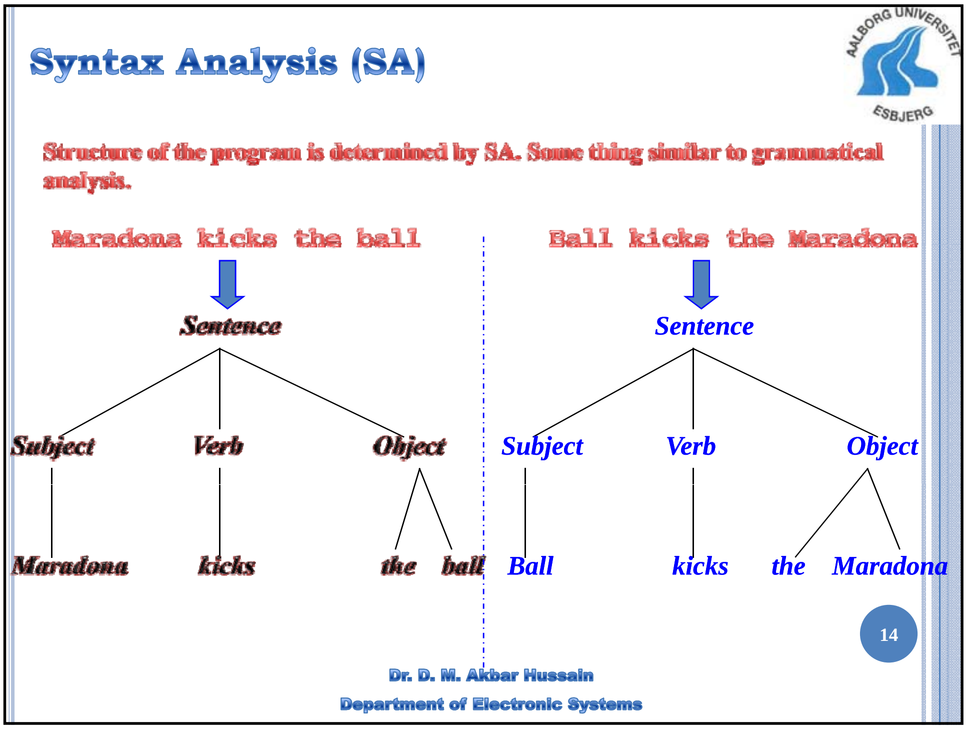
Token Generation:

o	X	id_1
o	=	operator
o	Y	id_2
o	+	operator
o	30	literal/constant

What other functions Scanner can perform ?

13

Dr. D. M. Akbar Hussain
Department of Electronic Systems



Syntax Analysis (SA)

Structure of the program is determined by SA. Some thing similar to grammatical analysis.

Maradona kicks the ball

Sentence

Subject Verb Object

Maradona kicks the ball

Ball kicks the Maradona

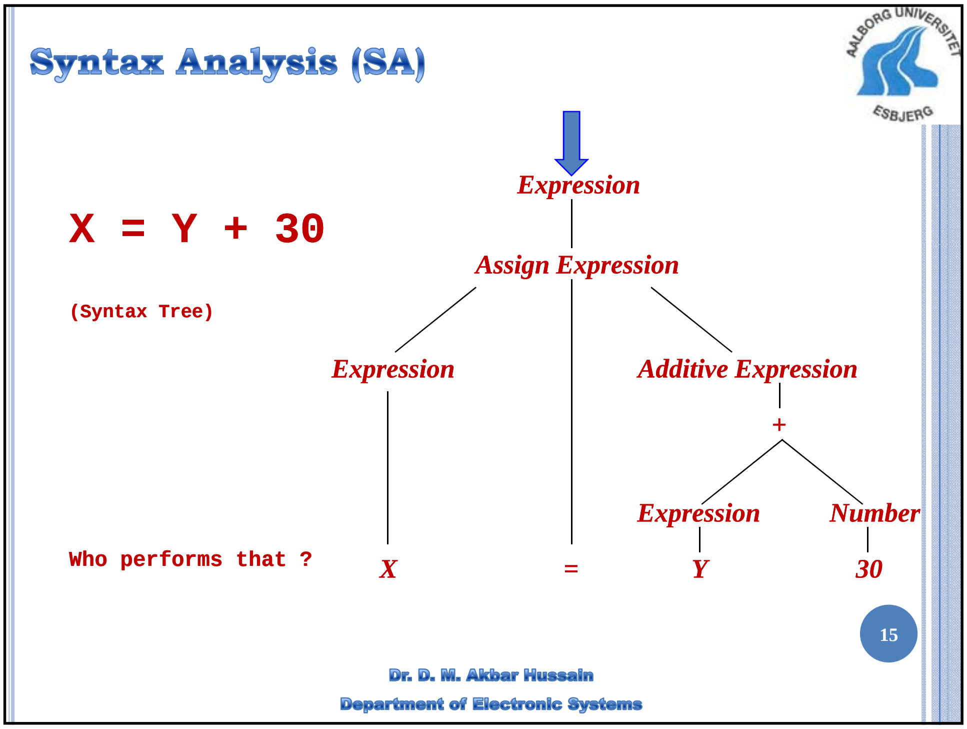
Sentence

Subject Verb Object

Ball kicks the Maradona

14

Dr. D. M. Akbar Hussain
Department of Electronic Systems



Syntax Analysis (SA)

X = Y + 30

(Syntax Tree)

Who performs that ?

Expression

Assign Expression

Expression

Additive Expression

+

Expression

Number

X

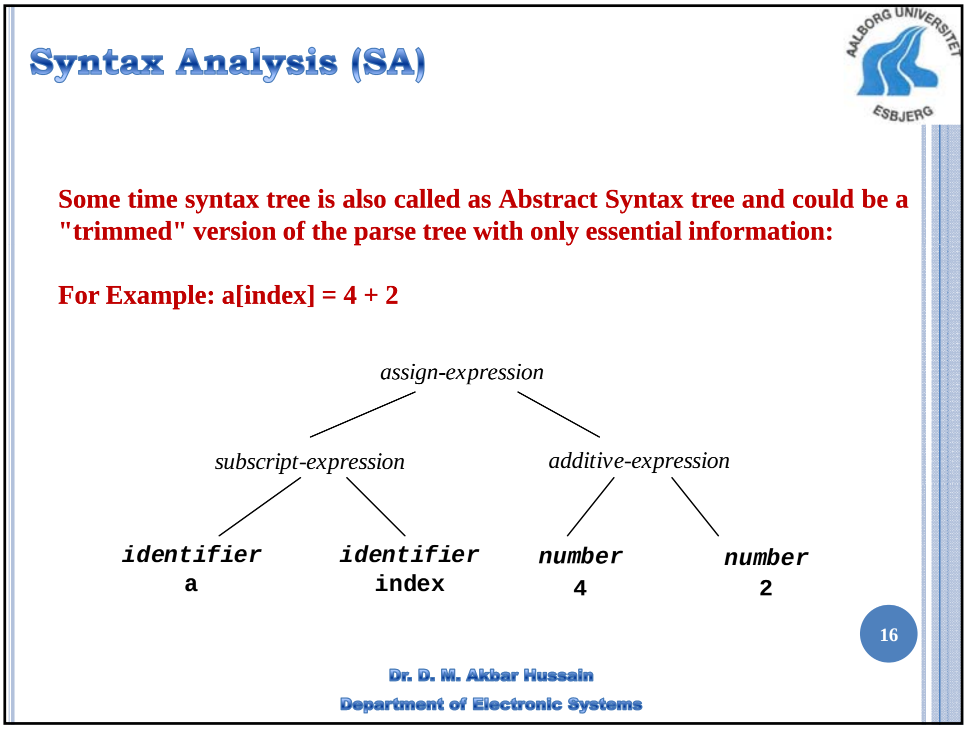
=

Y

30

15

Dr. D. M. Akbar Hussain
Department of Electronic Systems



Syntax Analysis (SA)

Some time syntax tree is also called as Abstract Syntax tree and could be a "trimmed" version of the parse tree with only essential information:

For Example: a[index] = 4 + 2

assign-expression

subscript-expression

additive-expression

identifier
a

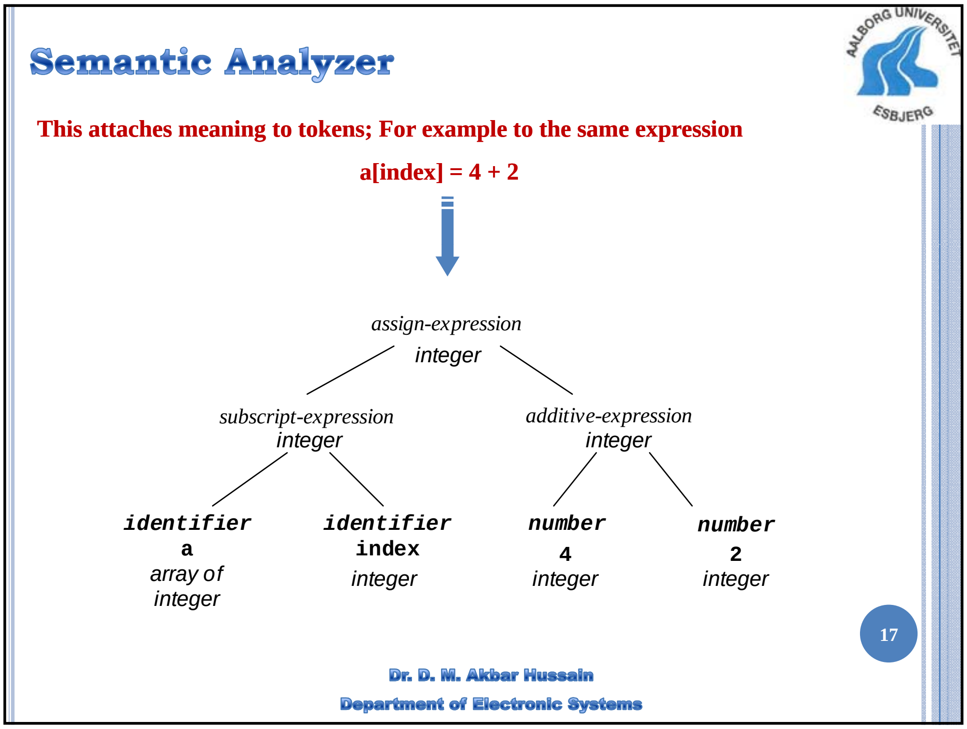
identifier
index

number
4

number
2

16

Dr. D. M. Akbar Hussain
Department of Electronic Systems



Semantic Analyzer

This attaches meaning to tokens; For example to the same expression

a[index] = 4 + 2

assign-expression
integer

subscript-expression
integer

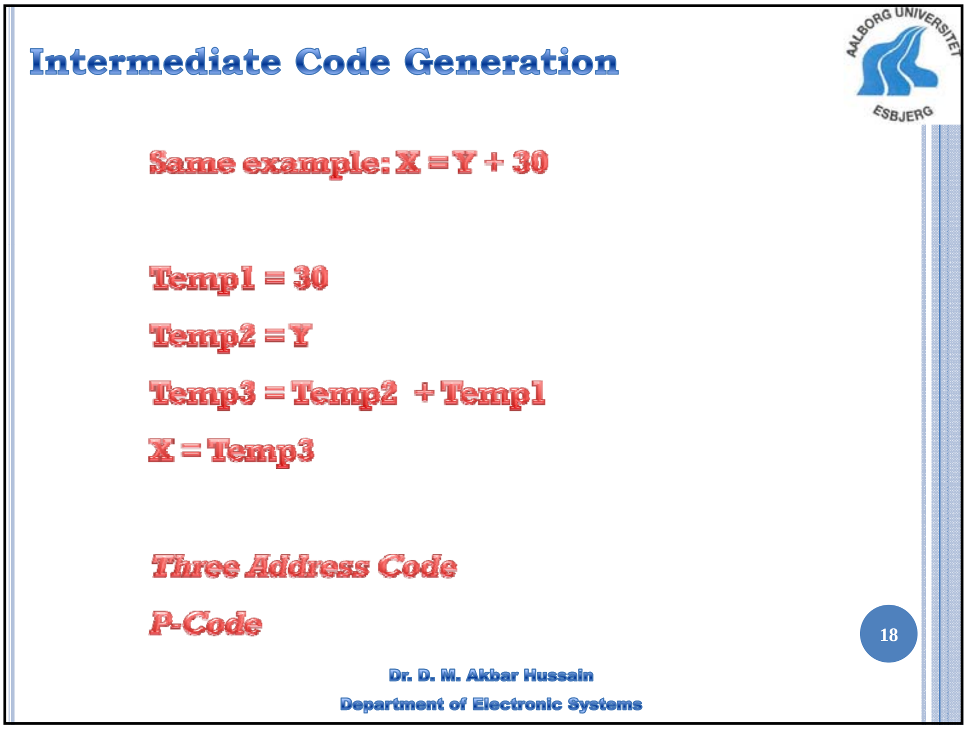
additive-expression
integer

`identifier`
`a`
array of integer

`identifier`
`index`
integer

`number`
`4`
integer

`number`
`2`
integer



Intermediate Code Generation

Same example: X = Y + 30

Temp1 = 30

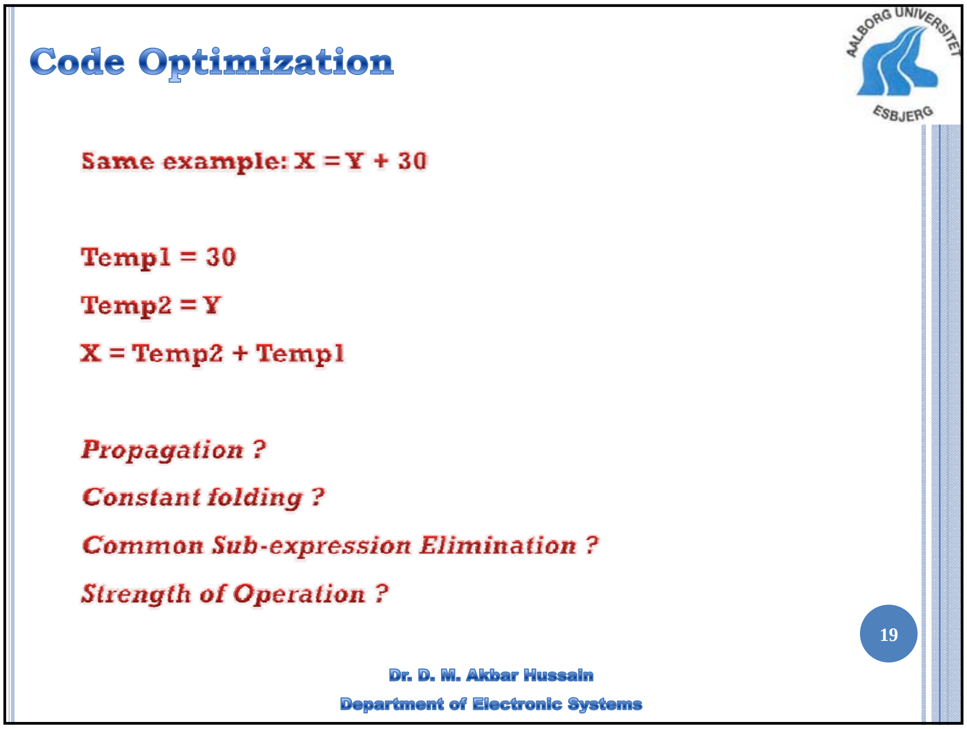
Temp2 = Y

Temp3 = Temp2 + Temp1

X = Temp3

Three Address Code

P-Code



Code Optimization

Same example: X = Y + 30

Temp1 = 30

Temp2 = Y

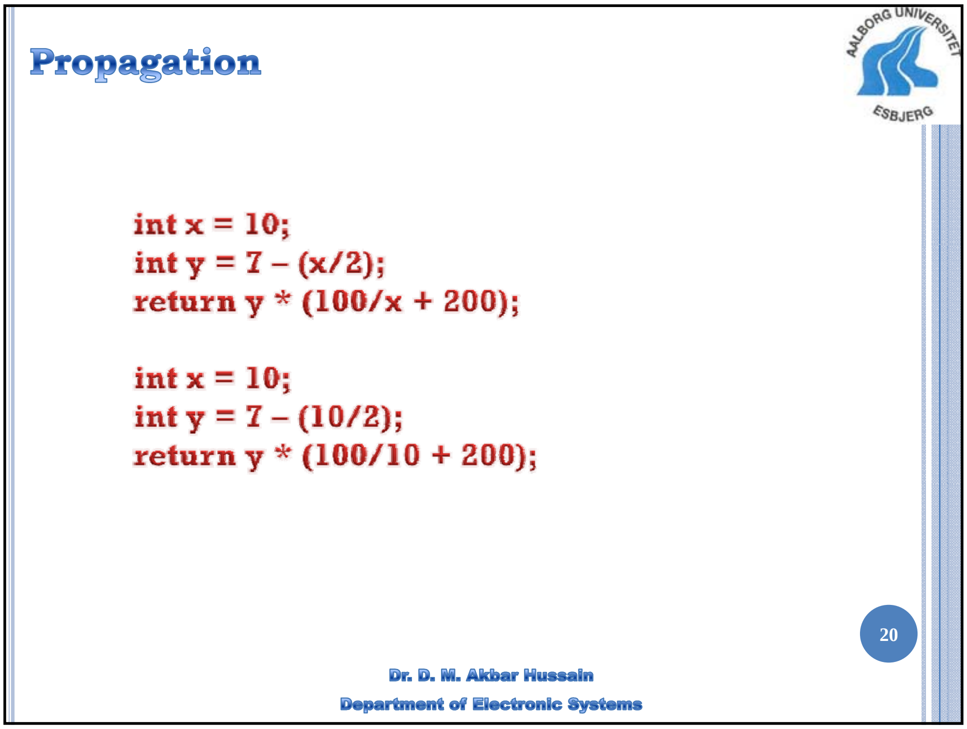
X = Temp2 + Temp1

Propagation ?

Constant folding ?

Common Sub-expression Elimination ?

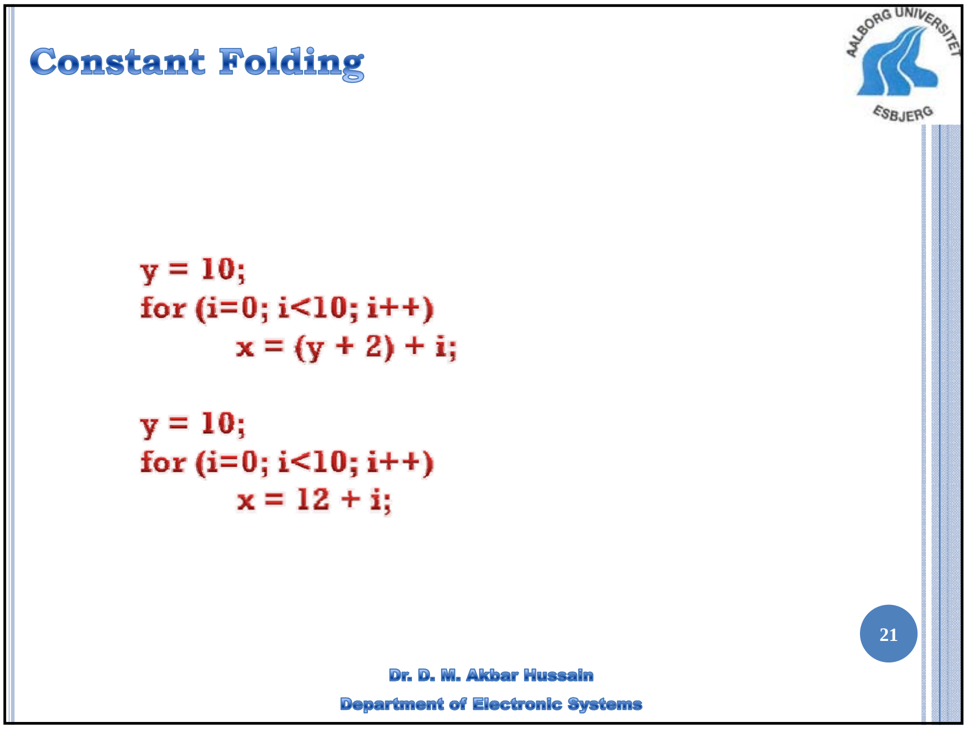
Strength of Operation ?



Propagation

```
int x = 10;
int y = 7 – (x/2);
return y * (100/x + 200);

int x = 10;
int y = 7 – (10/2);
return y * (100/10 + 200);
```



Constant Folding

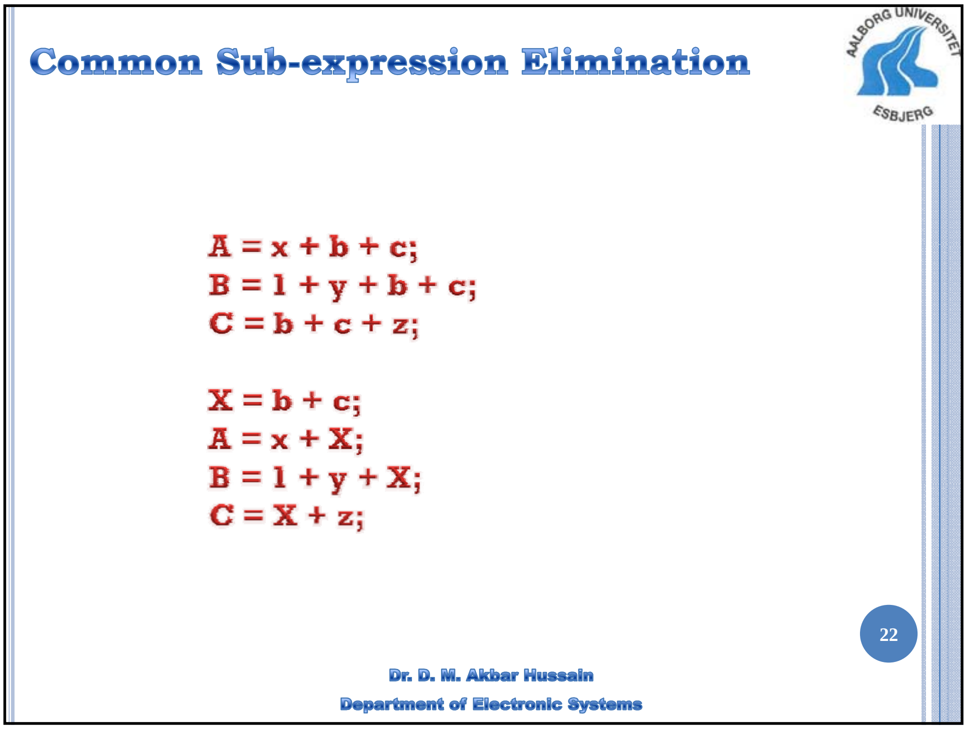
```
y = 10;
for (i=0; i<10; i++)
        x = (y + 2) + i;

y = 10;
for (i=0; i<10; i++)
        x = 12 + i;
```

21

Dr. D. M. Akbar Hussain

Department of Electronic Systems



Common Sub-expression Elimination

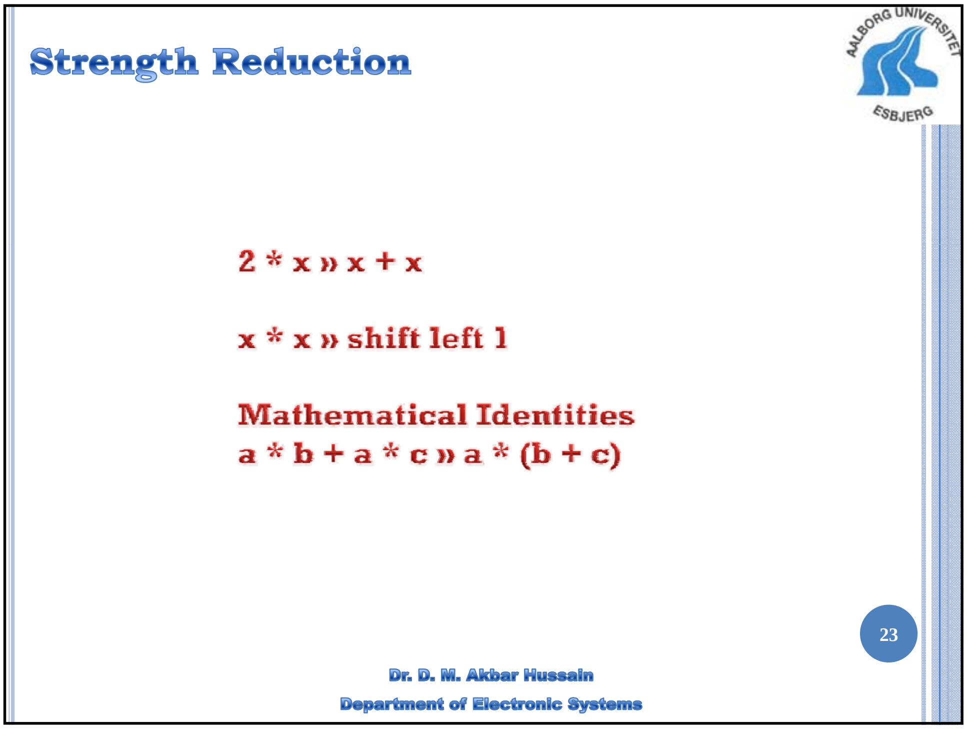
```
A = x + b + c;
B = 1 + y + b + c;
C = b + c + z;

X = b + c;
A = x + X;
B = 1 + y + X;
C = X + z;
```

22

Dr. D. M. Akbar Hussain

Department of Electronic Systems



Strength Reduction

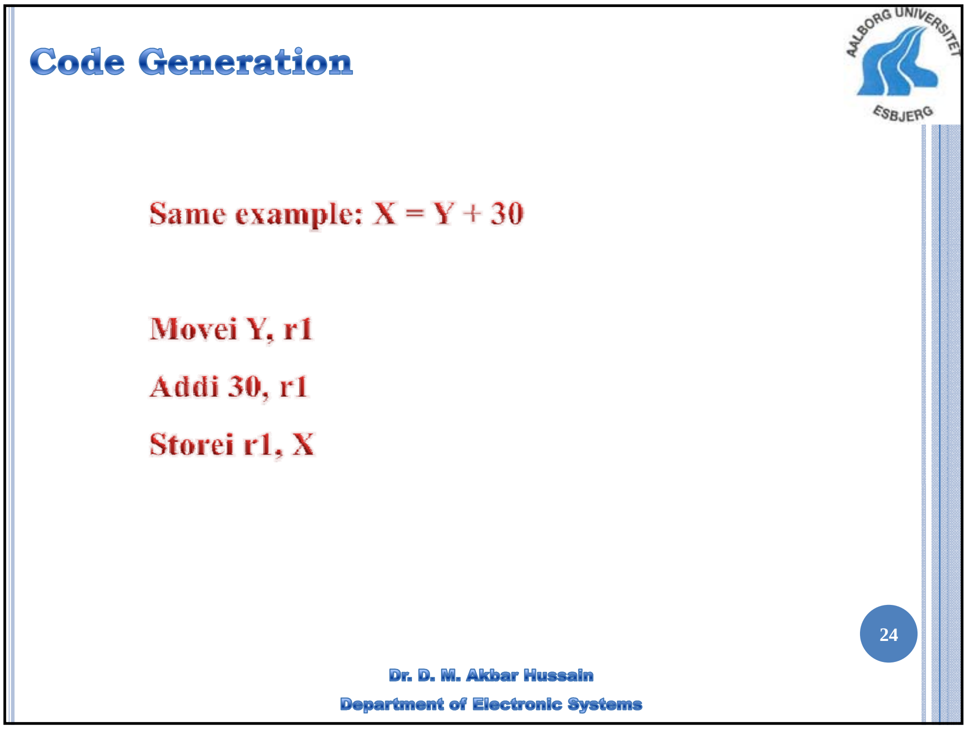
2 * x » x + x

x * x » shift left 1

Mathematical Identities
a * b + a * c » a * (b + c)

23

Dr. D. M. Akbar Hussain
Department of Electronic Systems



Code Generation

Same example: X = Y + 30

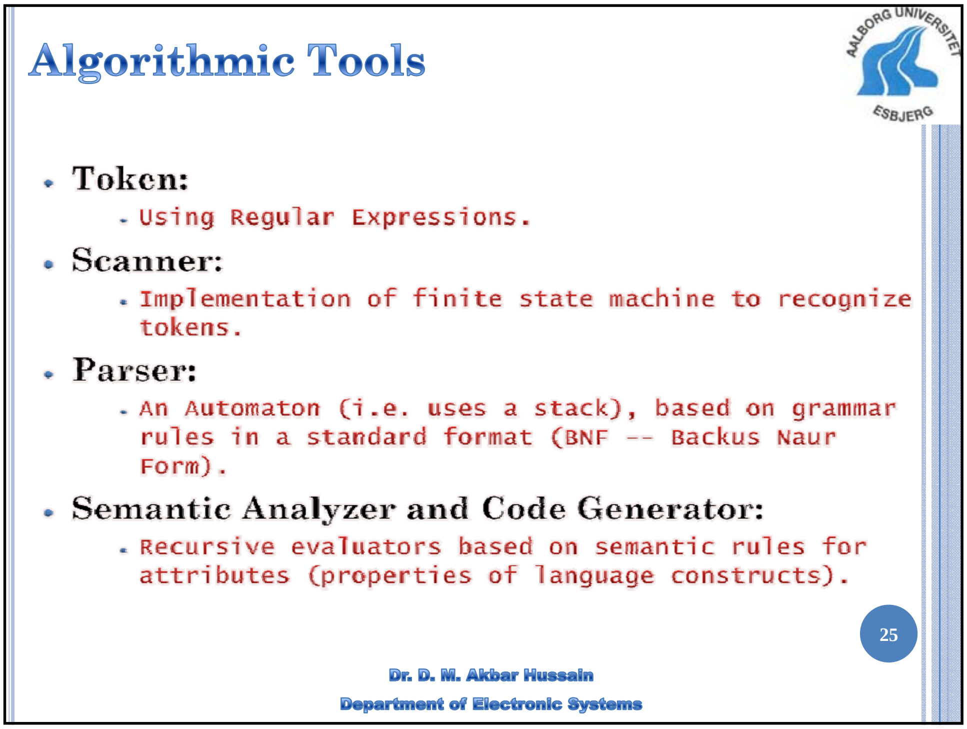
Movei Y, r1

Addi 30, r1

Storei r1, X

24

Dr. D. M. Akbar Hussain
Department of Electronic Systems



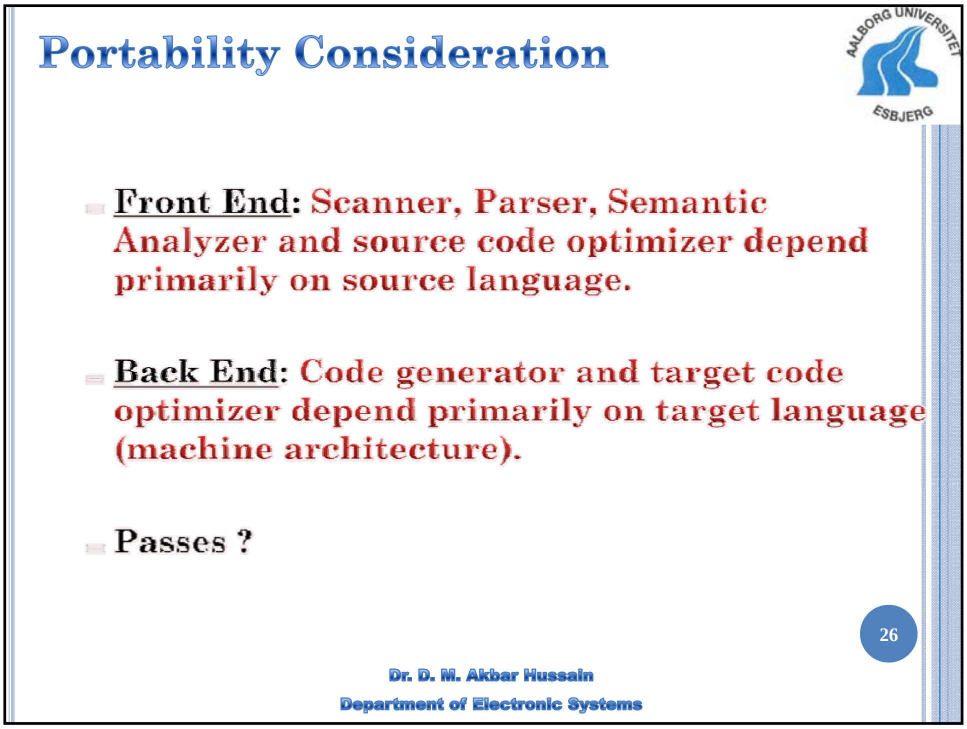
Algorithmic Tools

- Token:
  - Using Regular Expressions.
- Scanner:
  - Implementation of finite state machine to recognize tokens.
- Parser:
  - An Automaton (i.e. uses a stack), based on grammar rules in a standard format (BNF -- Backus Naur Form).
- Semantic Analyzer and Code Generator:
  - Recursive evaluators based on semantic rules for attributes (properties of language constructs).

25

Dr. D. M. Akbar Hussain

Department of Electronic Systems



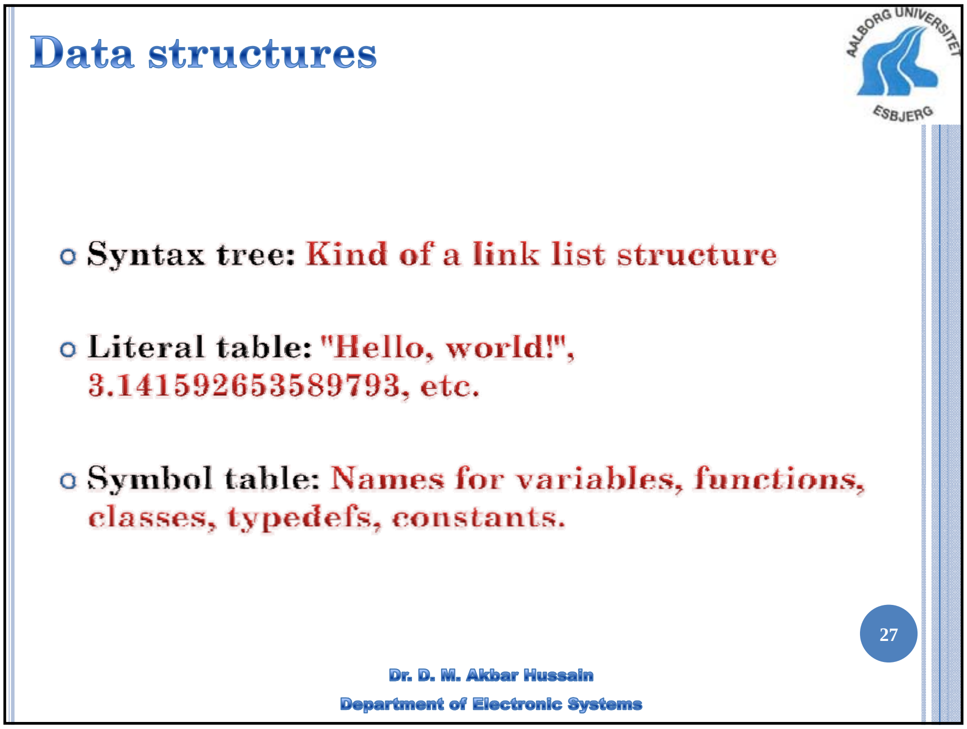
Portability Consideration

- Front End: Scanner, Parser, Semantic Analyzer and source code optimizer depend primarily on source language.
- Back End: Code generator and target code optimizer depend primarily on target language (machine architecture).
- Passes ?

26

Dr. D. M. Akbar Hussain

Department of Electronic Systems

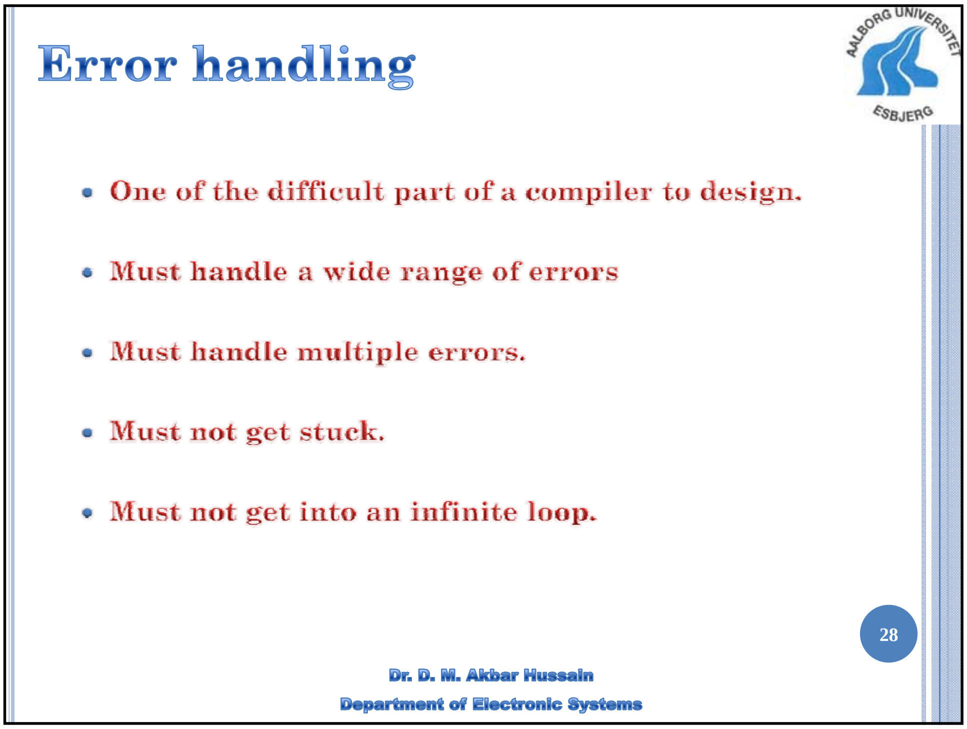


Data structures

- Syntax tree: Kind of a link list structure
- Literal table: "Hello, world!", 3.141592653589793, etc.
- Symbol table: Names for variables, functions, classes, typedefs, constants.

27

Dr. D. M. Akbar Hussain
Department of Electronic Systems

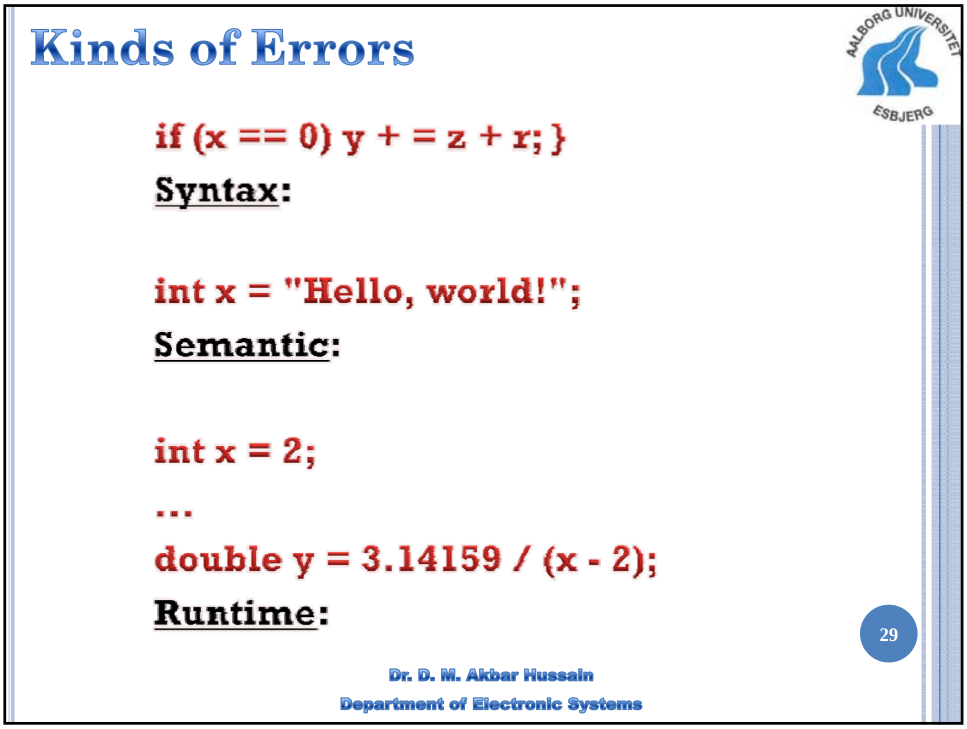


Error handling

- One of the difficult part of a compiler to design.
- Must handle a wide range of errors
- Must handle multiple errors.
- Must not get stuck.
- Must not get into an infinite loop.

28

Dr. D. M. Akbar Hussain
Department of Electronic Systems



Kinds of Errors

```
if (x == 0) y + = z + r; }
```

Syntax:

```
int x = "Hello, world!";
```

Semantic:

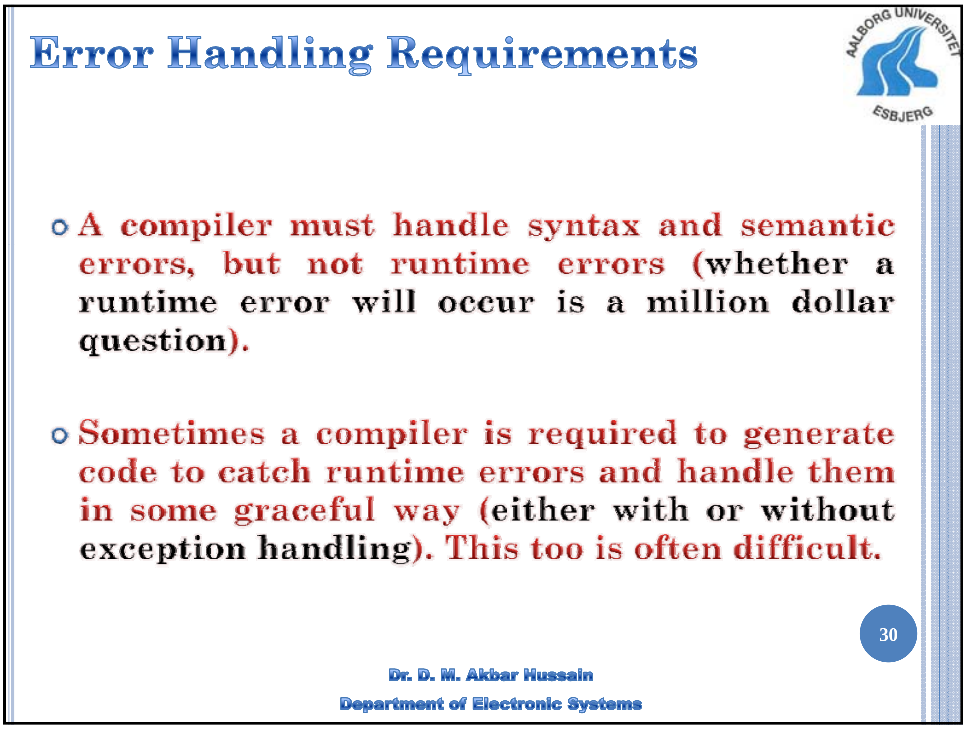
```
int x = 2;
...
double y = 3.14159 / (x - 2);
```

Runtime:

29

Dr. D. M. Akbar Hussain

Department of Electronic Systems



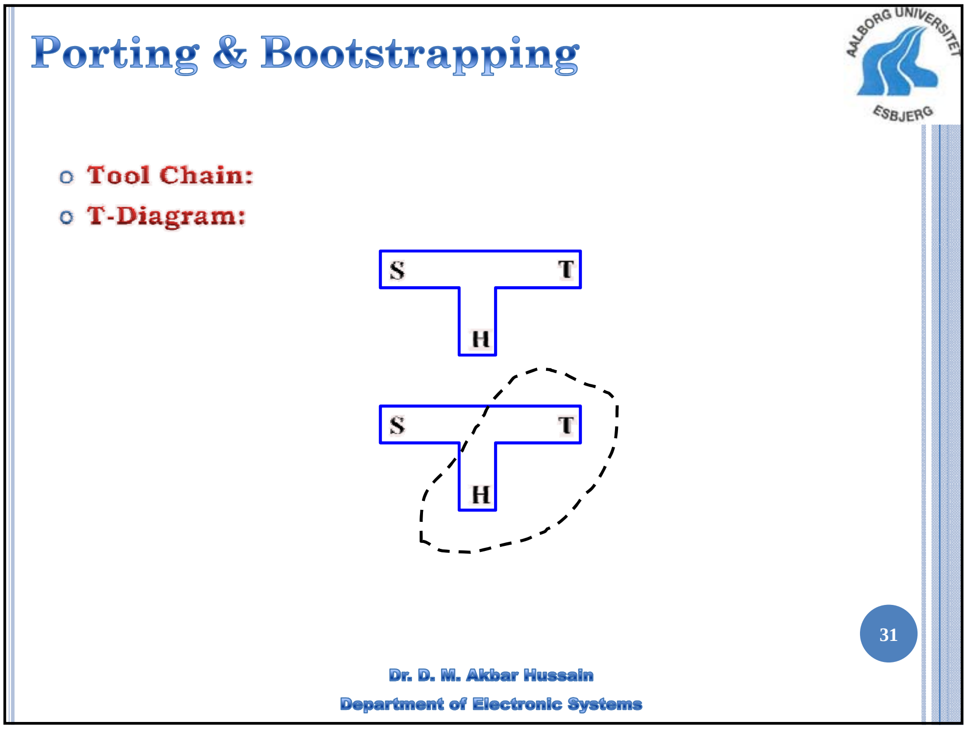
Error Handling Requirements

- A compiler must handle syntax and semantic errors, but not runtime errors (whether a runtime error will occur is a million dollar question).
- Sometimes a compiler is required to generate code to catch runtime errors and handle them in some graceful way (either with or without exception handling). This too is often difficult.

30

Dr. D. M. Akbar Hussain

Department of Electronic Systems

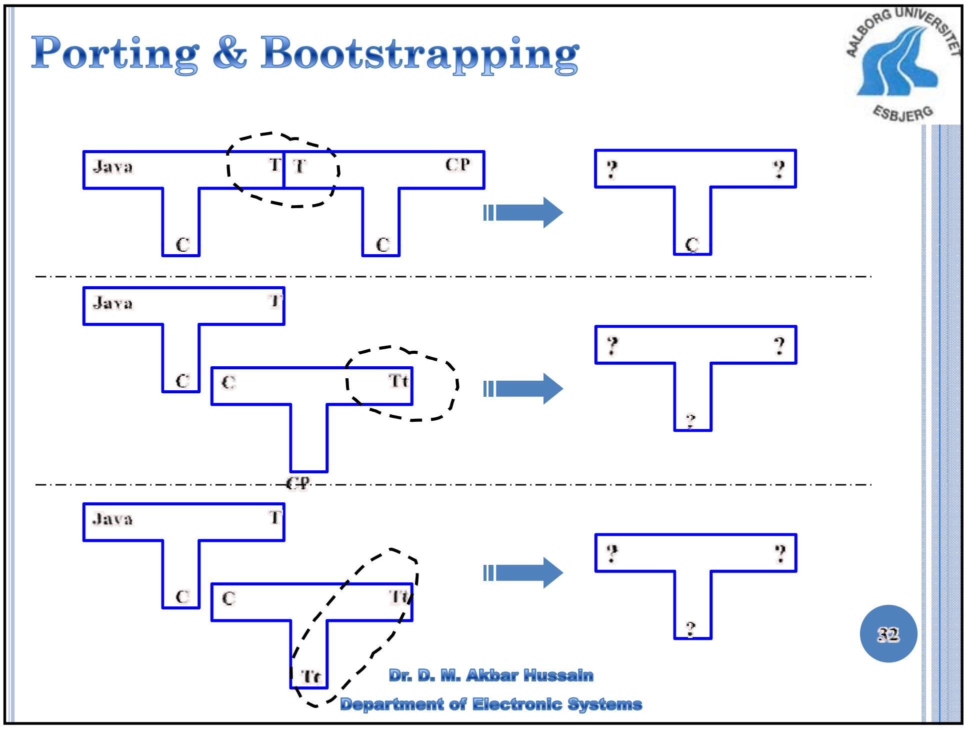


Porting & Bootstrapping

- Tool Chain:
- T-Diagram:

31

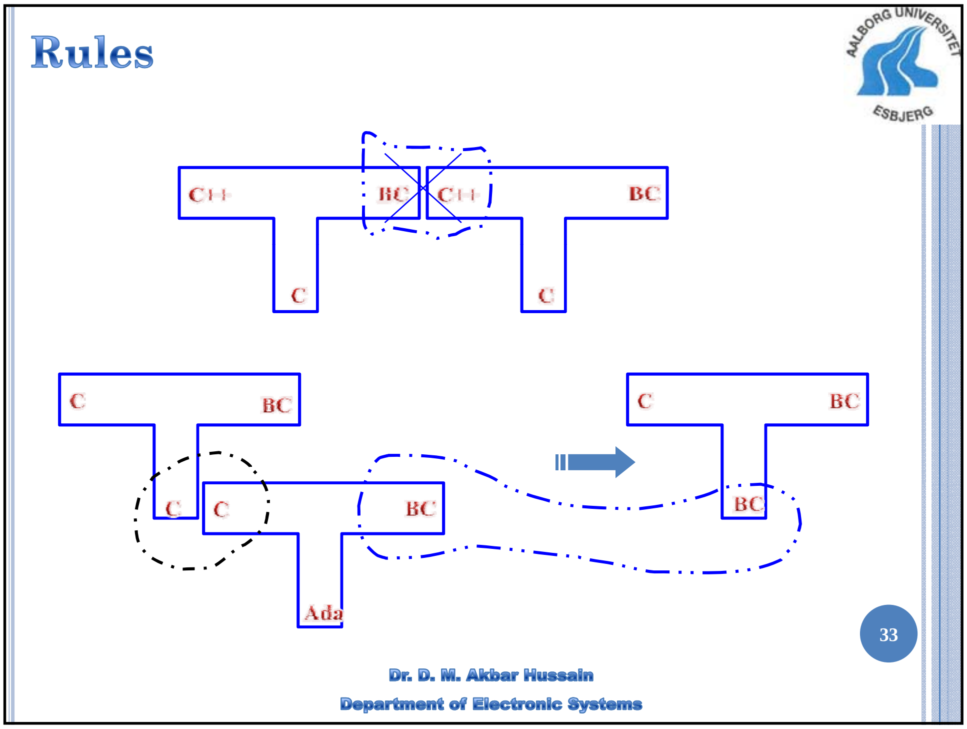
Dr. D. M. Akbar Hussain
Department of Electronic Systems



Porting & Bootstrapping

32

Dr. D. M. Akbar Hussain
Department of Electronic Systems

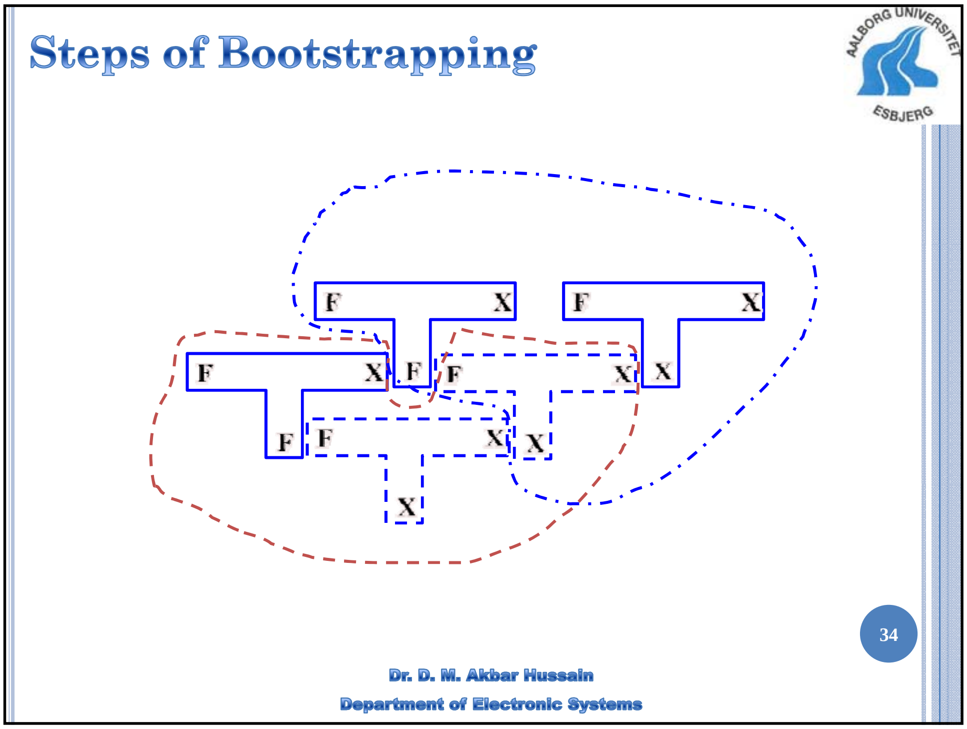


Rules

33

Dr. D. M. Akbar Hussain

Department of Electronic Systems

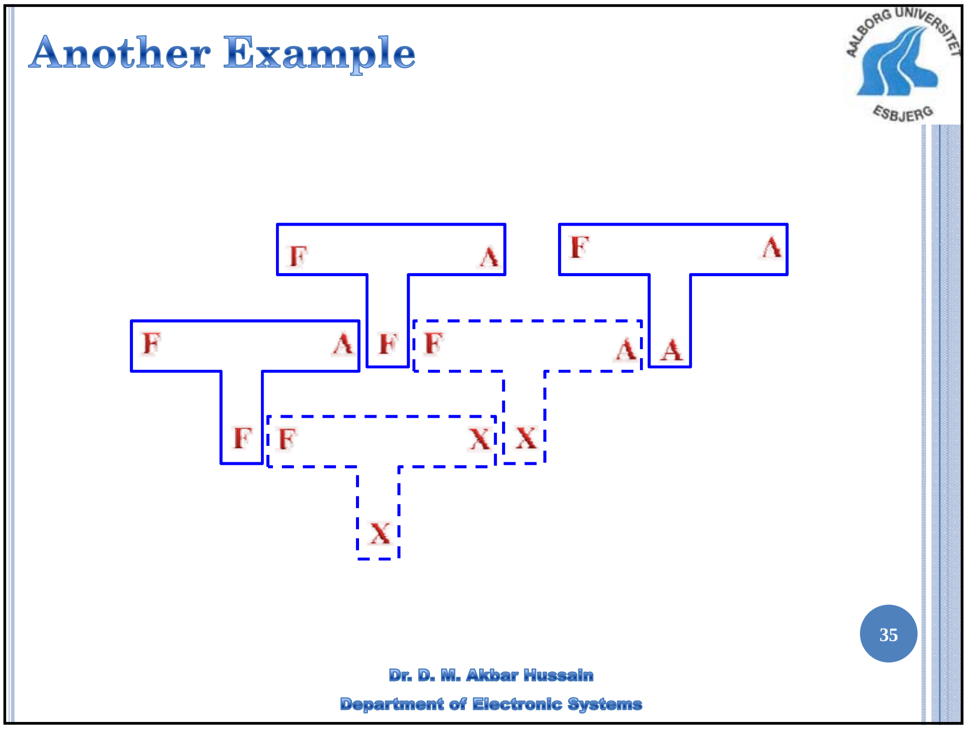


Steps of Bootstrapping

34

Dr. D. M. Akbar Hussain

Department of Electronic Systems



Another Example

35

Dr. D. M. Akbar Hussain

Department of Electronic Systems