1. Introduction

Alecsis 2.3 is the newest version of the integral, hybrid simulator developed in the Laboratory for Electronic Design Automation (LEDA) of the Faculty of Electronics, University of Niš. The name itself is an acronym of **A**nalogue and **L**ogic **E**lectronic **C**ircuit **SI**mulation **S**ystem. Alecsis 2.3 is an integrated simulator because it represents the coupling of the interpreter and the linker with the simulation engine, and the connection between the mechanisms for analogue and digital simulation. The following chapters will show that the circuit description syntax of the analogue and digital portion varies insignificantly across both types due to their inseparability.

All modern hardware description languages (HDLs) use one of the standard programming languages as a basis. Programming language C was chosen as the basis for Alecsis 1.1 due to its widespread acceptance. During the development of the next version (Alecsis 2.1) object-oriented programming capabilities were added through the use of C++ constructs. Version 2.3 was released as an improved, more effective version of Alescis 2.1. HDL used in Alecsis 2.3 is therefore a superset of C++ (the same way C++ is a superset of C.) The authors strive to emphasize the details which differ from the rules of C, or C++. If there exists ambiguity beyond these details any handbook on C, or C++ will help.

Alecsis is a package consisting of language interpreter/compiler and the simulation engine. In the text the package will be referred to as Alecsis 2.3, or just Alecsis. In order to differentiate between the abstract notion of a language and the functional programming package the language itself will be referred to as AleC++.

This Manual is split in eight chapters and seven appendices. The second and third chapter define the lexical and grammatical rules of AleC++. Object-oriented construct of the language are introduced in the fourth chapter. Basic simulation capabilities are presented in the fifth chapter, and their application in the in the analogue and digital domains in the sixth and seventh chapters, respectively. Finally, the application to the hybrid circuits is given in the eighth chapter. Application and installing of the whole package, as well as the content of the standardized libraries is given in the Appendix A. Appendix B offers the complete syntax of AleC++. Commands of the assembly language of the virtual processor used in the simulator are explained in Appendix C. Appendix D describes ALM (Alecsis Library Manager) which enables easy handling of the user-defined libraries. Appendix E develops the description of the graphical postprocessor Agnu 1.1, used in conjunction with a shareware graphical

display program Gnuplot. The graphical display of the simulation results using Agnu is possible during the simulation run, too.

The following terminology will be used in Appendix B and the parts of the text concerning the syntax rules of AleC++.

- Syntax categories will be defined using the cursive, e.g.

global_definition:

global_variable

function

- Listing of categories using ':' has the meaning "one of." If a category may, or may not be included, it will be marked using '<' and '>':

ptr_operator:

** <cv_qualifier>*

- If multiple options were to appear in single row the will be emphasized using the note "one of:"

decimal_digit: *one of*

0 1 2 3 4 5 6 7 8 9

- Bold symbols represent the reserved words:

cv_ qualifier :

const

volatile