

Abstracts

Application of Error-Correcting Codes in Computer Reliability Studies, M. Y. Hsiao and J. T. Tou, *IEEE Transactions on Reliability* R-18, No. 3 (August 1969). The use of error-correcting codes as one of the important techniques to increase computer system reliability is introduced. The different codes used in the central processing unit (CPU) are described. Since the CPU usually contains the data path, logic, and arithmetic units, the codes used in this area are error-detecting codes, such as parity check codes and residue codes. The codes used or suggested for the memory system are discussed, emphasis being placed on parity check codes, two-dimensional codes, Hamming codes and other recently developed codes. The various codes used in the input/output system are presented. The input/output area of the computer system is relatively unreliable as compared with CPU or memory; therefore, error-correcting codes used in this area usually are much more powerful than single parity check codes. These include codes for the magnetic tape, disk, and drum units. The error coding techniques are compared with other techniques for increasing computer system reliability. The future trend of using error-correcting codes in a computer system is also discussed.

A Case Study in Programming for Parallel-Processors, J. L. Rosenfeld, *Communications of the ACM* 12, No. 12, 645-655 (December 1969). An affirmative partial answer is provided to the question of whether it is possible to program parallel-processor computing systems to efficiently decrease execution time for useful problems. A program for the determination of the distribution of current in an electrical network was written for a parallel-processor computing system, and execution of this program was simulated. The data gathered from simulation runs demonstrate the efficient solution of this problem, typical of a large class of important problems. It is shown that, with proper programming, solution time when N_p processors are applied approaches $1/N_p$ times the solution time for a single processor, while improper programming can actually lead to an increase of solution time with the number of processors. Storage interference and other measures of performance are discussed. Stability of the method of solution was also investigated.

A Computer Program for the Synthesis of Decoupled Multivariable Feedback Systems, E. G. Gilbert and J. R. Pivnichny, *IEEE Transactions on Automatic Control* **14**, No. 6, 652 (December 1969). Recently Gilbert obtained general solution results on the decoupling of multivariable systems by state feedback. This paper presents a general-purpose computer program which carries out all of the calculations necessary for reducing these results to a useful synthesis procedure. The program is described in general terms and several examples of its application are given.

Controlling the Functional Testing of an Operating System, W. R. Elmen-dorf, *IEEE Transactions on Systems Science and Cybernetics* **SSC-5**, No. 4, 284-290 (October 1969). Functional testing of operating systems is in transition from a predominantly imprecise art to an increasingly precise science. The process that controls this testing is maturing correspondingly. The laissez-faire approach is giving way to a disciplined approach characterized by rigorous definition of the test plan, systematic control of the test effort, and objective quantitative measurement of the test coverage. This paper describes just such a disciplined test control process, which is composed of five steps: (1) the survey, which establishes the intended extent of testing; (2) the identification, which creates a list of functional variations eligible for testing; (3) the appraisal, which ranks and subsets the eligible variations so that test resources can be directed at those with the higher payoff; (4) the review, which calculates the test coverage of the test case library; and (5) the monitor, which verifies attainment of the planned test coverage. Throughout the test process, specification testing is distinguished from program testing.

A General Purpose Digital Traffic Simulator, A. M. Blum, *Simulation* **14**, No. 1, 9-25 (January 1970). The described vehicle traffic simulator is designed to facilitate analysis of traffic flow and to experiment with postulated traffic control systems. It offers users a large amount of flexibility in specifying network, intersection, vehicle, and control parameters. The logical flow of vehicles, as well as much of the control system, is preprogrammed. Vehicles varying in size may change lanes, turn, change velocity (including reaction and acceleration delays), and merge. Input volumes may be varied, turns may be eliminated, and vehicles may be routed through the network. The user supplies the geometrical characteristics and input information unique to his particular network in data cards for the program. Some applications to real traffic networks (including a validation procedure) and a simple control experiment are discussed. The model, programmed in a special version of GPSS II and in FAP for the IBM 7090/94 systems, can be used in the simulation of single intersections, arterial routes, grid networks, and, for some special cases, limited access roadways.

Line Control and Terminal Management in OS/360—A Proposed Control System, R. M. Winick, *Software Age* **4**, No. 1, 23-26 (January 1970). This article presents some of the problems and peculiarities of controlling the hardware in a telecommunications environment. Because of the wide diversity of terminals comprising a teleprocessing network and the innumerable ways that software facilities can be combined to produce a total real-time control system, a set of programming functions rather than an integrated software package is proposed. Emphasis placed upon the line control and terminal discipline required to manage the remote stations. The set of functions described will provide control for both start-stop and binary synchronous hardware.

A New In-plant Data Communication System, R. M. Duncan, *Telecommunications* 4, No. 2, 32-34 (February 1970). Recent advances in data communication technology have been applied to develop a system for effective monitoring and control of the work flow through a manufacturing plant. A system has been specifically designed for two-way communication with many reporting locations, fast response, and a capability to handle peak loads of relatively short messages in a relatively confined geographical area. The two-wire system allows ease of installation and relocation of terminal devices.

Space Applications of a Minimization Algorithm, J. P. Roth and M. Perlman, *IEEE Transactions on Aerospace and Electronic Systems* AES-5, No. 5, 703-711 (September 1969). A detailed account is given of the application of a minimization program to several design problems. Specifically these applications are concerned with the design of a curve function generator for a mass spectrometer for a proposed Mars probe and the design of autonomous shift registers with linear and nonlinear feedback, used for classification of binary sequences and counting tasks for spacecraft scientific data processing.

A Structural View of PL/I, D. Beech, *Computing Surveys* 2, No. 1, 33-64 (March 1970). This tutorial paper is presented at an intermediate level, which assumes that the reader already has either some elementary knowledge of PL/I or a more general familiarity with other high level languages. An attempt is made to break fresh ground by addressing a question concerning the design philosophy of PL/I, namely whether it is possible for the more experienced programmer to grasp the deeper structure and achieve complete mastery of a language with such broad scope. (It is not concerned with showing how easily parts of the language may be used for simple applications, although several introductory texts are cited.) The method of the exposition is first to dispose of certain preprocesses which may be performed, and then to organize the major part of the material under two heads, static aspects and dynamic aspects. The static aspects are those essentially related to inspection rather than execution of a program; the dynamic part of the discussion outlines the semantics in terms of the internal storage, external storage, and flow of control in a conceptual PL/I machine. Every statement and type of object in the language is given a place in this scheme.

Syntax-Directed Documentation for PL360, H. D. Mills, *Communications of the ACM* 13, No. 4, 216-222 (April 1970). The language PL360, together with its phrase structure grammar, is used as a concrete basis for illustrating an idea called syntax-directed documentation. This idea is: (1) to use the phrase structure of a program to define the structure of a formal documentation for that program; (2) to use the syntactic types and identifiers in the resulting structure to trigger the automatic formation of questions to the programmer, whose answers will become part of that documentation; and (3) to provide automatic storage and retrieval facilities so that other programmers who want to understand or modify the program can

access the resulting documentation, which is cross-indexed in various ways by syntactic types and objects. A small PL360 program, already found in the literature, is worked out as an example.

The Use of Quadratic Residue Research, C. E. Radke, *Communications of the ACM* **13**, No. 2, 103-4 (February 1970). A quadratic residue search method has previously been suggested to avoid the clustering usually encountered when hash address collisions occur and linear search methods are used. The search size, because of the property of quadratic residues, is limited to one half of the storage table. It is shown that for some classes of prime numbers the complement of the set of quadratic residues can easily be determined and hence the entire table of size p , where p is that prime number, can be searched.