INTERNATIONAL ATOMIC ENERGY AGENCY
UNITED NATIONS EDUCATIONAL, SCIENTIFIC AND CULTURAL ORGANIZATION
INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS
I.C.T.P., P.O. BOX 586, 34100 TRIESTE, ITALY, CABLE: CENTRATOM TRIESTE

UNITED NATIONS INDUSTRIAL DEVELOPMENT ORGANIZATION

INTERNATIONAL CENTRE FOR SCIENCE AND HIGH TECHNOLOGY
c/o INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS 34100 TRIESTE (ITALY) VIA GRIGNANO, 9 (ADRIATICO PALACE) P.O. BOX 586 TELEPHONE 040-224572 TELEFAX 040-224575 TELEX 460449 APH I

SMR/474 - 12

COLLEGE ON
"THE DESIGN OF REAL-TIME CONTROL SYSTEMS"
1 - 26 October

Additional material for lectures

by

P. BARTHOLDI
Observatory of Geneva
CH-12900 Sauverny
Switzerland

These are preliminary lecture notes, intended only for distribution to participants.

Bibliography

Paul Bartholdi
Observatory of Geneva
CH-1290 Sauverny
Switzerland

Preliminary notes - Trieste - october 1990

College on "Design of Real-Time Control Systems"

Contents

bartho@obs.unige.ch bartho@cgeuge54.bitnet 20579::ugobs::bartho
tel +41 22 755 26 11 fax +41 22 755 39 83

1 High Level Languages

- Allen, J.R.
  Anatomy of Lisp
  Mc Graw Hill 1978
- Brodie, L.
  Starting Forth
  Prentice-Hall 1981
- Brodie, L.
  Thinking Forth
  Prentice-Hall 1984
- Downes, V.A. and S.J. Goldsack
  Programming embedded systems with Ada
  Prentice-Hall 1982
- Friedman, D. and M. Felleisen
  The little LISPer
  Science Research Associates, Inc. 1986
- Ghezzi, C. and M. Jazayeri
  Programming language concepts
  John Wiley and sons 1982
- Griswold, R.E. and M.T. Griswold
  A SNOBOL 4 primer
  Prentice-Hall 1973
- Henderson, P.
  Functional programming, application and implementation
  Prentice-Hall 1980
- Kernighan, B.W. and D.M. Ritchie
  The C programming language
  Prentice-Hall 1978
- Kerridge, J.
  Occam programming : a practical approach
  Blackwell Scientific Publications 1987
- Metcalf, M.
  Effective Fortran 77
  Clarendon Press 1985
- Winston, P.H. and B.K.P. Horn
  LISP
  Addison-Wesley 1981
- Wirth, N.
  Programming in Modula-2
  Springer Verlag 1983

2 Operating Systems

- Brinch-Hansen, P.
  Operating system principles
  Prentice-Hall 1973
- Brinch-Hansen, P.
  The architecture of concurrent programs
  Prentice-Hall 1977
- Christian, K.
  The Unix Operating system
  John Wiley and sons, 1983
- Coffman, E.G. and P.J. Denning
  Operating system theory
  Prentice-Hall 1973
- Comer, D.
  Operating system design, the XINU approach
  Prentice-Hall 1984
- Comer, D.
  Operating system design, Internetworking with XINU
  Prentice-Hall 1988
- Gilton, H. and R. Morgan
  Introducing the Unix system
  Mc Graw Hill 1983
- Hoare, C.A.R.
  Communicating Sequential Processes
  Prentice-Hall 1985
- Kernighan, E.G. et al.
  Unix time-sharing system
  The Bell system technical journal, 57 1897-1991 (1978)
- Kernighan, E.G. and J.R. Mashey
  The Unix programming environment
  Soft. Pract. and Exp. 9 1-5 (1979)
- Organick, E. I.
  Computer System Organization, The B5700/B6700 Series
  Academic Press, ACM Monograph Series 1973
- Tanembaum, A. S.
  Operating Systems: Design and Implementation (Minix)
  Prentice-Hall 1987

3 Structured programming

- Dahl, O. Dijkstra, E.W. and C.A.R. Hoare
  Structured programming
  Academic Press 1972
- Dijkstra, E.W.
  GOTO statements considered harmful
  CACM 11 147-148 (1968)
- Dijkstra, E.W.
  A discipline of programming
  Prentice-Hall 1976
- Hughes, Ch.E. et al.
  Advanced course in programming using Fortran
  John Wiley and sons 1978
- Kruse, R.L.
  Data structures and program design
  Prentice-Hall 1984
- Kernighan, B.W. and P.J. Plauger
  Software tools
  Addison-Wesley 1976
- Kernighan, B.W. and P.J. Plauger
  The elements of programming style
  Addison-Wesley 1978
- Meek, B. and P. Heath
  Guide to good programming practice
  Ellis Horwood Ltd 1980
- Pressman, R. S.
  Software Engineering, A Practitioner's Approach
  MacGraw-Hill International Editions 1987
- Wirth, N.
  Program development by stepwise refinement
  CACM 14, 221-227 (1971)
- Wirth, N.
  Systematic programming
  Prentice-Hall 1973
- Wirth, N.
  Algorithms + Data-Structure = Programs
  Prentice-Hall 1976
- Yourdon, E.
  Techniques of program structure and design
  Prentice-Hall 1975

4 Modular decomposition, abstraction and information hiding

- Parnas, D.L.
  On the criteria to be used in decomposing systems into modules
  CACM 15, 1053-1058 (1972)
- Parnas, D.L. et al.
  The modular structure of complex systems
  IEEE Trans. on Soft. Eng. SE11,3 259-266 (1985)

5 Algorithms

- Bentley, J.
  Programming Pearls
  Addison-Wesley Publishing Co 1986
- Bentley, J.
  More Programming Pearls, Confessions of a Coder
  Addison-Wesley Publishing Co 1988
- Dewar, R.B.K.
  Indirect threaded code
  CACM 18 330-331 (1975)
- Knuth, D.E.
  The art of computer programming
  vol 1 : Fundamental algorithms
  vol 2 : Seminumerical algorithms
  vol 3 : Sorting and searching
  Addison-Wesley
- Loeliger, R.G.
  Threaded interpretive languages
  Byte Books 1981
- Sedgewick, R.
  Algorithms
  Addison-Wesley 1983

6 Floating Point operations

- Cody, W.J.Jr and W. Waite
  Software manual for the elementary functions
  Prentice-Hall 1980
- Cody, W.J.Jr
  Analysis of the proposals for the floating point standard
  IEEE Computer march (1981)
- Tanenbaum, A.S.
  Structured computer organization
  Prentice-Hall 1976

- Titus, j.
  Design precaution ensure the benefits of using floating-point coprocessors
  EDN june 57-64 (1986)

- Waser, S. and M.J. Flynn
  Introduction to arithmetic for digital systems designers
  Holt, Rinehart and Winston 1982

7 Numerical methods

- Press, W. H. et al.
  Numerical recipes : Methods for numerical computation
  (with sources in Fortran 77 and turbo pascal)
  Cambridge University Press 1986

- Press, W. H. et al.
  Numerical recipe examples
  (with sources in Fortran 77 and turbo pascal)
  Cambridge University Press 1986

- Abramowitz, M. and I.A. Stegun
  Handbook of mathematical functions, with formulas, graphs and mathematical tables
  Dover 1965

- de Boor, C.
  A practical guide to splines
  Springer 1978

- Bracewell, R,
  The Fourier transform and its applications
  Mc Graw-Hill 1965

- Chan, T.F. and J.G. Lewis
  Computing standard deviations : accuracy
  CACM 22 526-531, (1979)

- Cloutier, M.J. and M.J. Friedman
  Precision averageing for real-time analysis
  CACM 26 525-529, (1983)

- Garbow, B.S. et al.
  Matrix eigensystem routines, EISPACK guide extensions
  Springer 1977

- Hamming, R.W.
  Digital filters
  Prentice-Hall 1977

- Lawson, C.L. and R.J. Hanson
  Solving least squares problems
  Prentice-Hall 1974

- Maron, M.J.
  Numerical analysis
  Macmillan 1982

- Piessens, R. et al.
  QUADPACK, a subroutine package for automatic integration
  Springer 1983

- Shampine, L.F. and M.K. Gordon
  Computer solution of ordinary differential equations, the initial value problem
  Freeman 1975

- Smith, B.T. et al.
  Matrix eigensystem routines, EISPACK guide
  Springer 1970

- West, D.H.D.
  Updating mean and variance estimates : an improved method
  CACM 22 532-535, (1979)

8 Unclassified

- Bentley, J.L.
  Writing efficient programs
  Prentice-Hall 1982

- Dreyfus, H.L. and S.F. Dreyfus
  Mind over Machine : The Power of Human Intuition and Expertise in the Era of the Computer
  Basil Blackwell Ltd. 1986

- Weizenbaum, J.
  Computer power and human reason, from judgment to calculation
  Freeman 1975

Debugging :

Goal : get as soon as possible a working exercice (prototype)

Proverbs: • To arrive early , spare your horse
• Go slowly , we are in a hurry (Swiss mountain climber

Rules : CHECK ALL I/O operations
CHECK ALL SYSTEM CALLS
CHECK ALL INPUT VARIABLES
TRACE execution with PRINTs
(it is easier to delete than add a line)
when you enter a bloc, before doing something special
USE wrong results as clues indices (dont throw them away because they are wrong, find what can produce this particular result)
APPLY binary search technic to find faulty statement(
START analysis from both ends of the program
HAVE all your manuals easily available on your desk, check once more that your syntax is correct, that your understanding of the semantic was correct ...
ASK someone else to review the program with you, You may find the error yourself while you explain your program.
LOOK for possible confusions ("=" instead of "==", "l" for "1" etc.)

□

What to do if your program is too slow?

- Improve the hardware (clock, cpu, memory) → cheapest!
- Improve the software
  1. get an idea of where you spend most of the time
     - use a profiler if available (see option -P in os9 cc)
     - modify your program to simulate one
       . Initialize to zero one counter per bloc, loop etc.
       . increment by one these counters when you enter the
       . blocs and loops
       . replace all "exit()" by "sexit()", a new routine that print all counters, with their names, then do a real exit.
     - write a true time profiler (= other process) that fetch the Program Counter of the first process, and make an histogram of it (PC). [how do you get the PC of an other process? find out using os9 manuals!].
  2. find the place(s), bloc(s) that are worth improving
  3. Change algorithm
     ex: from 1950 to 1990, the hardware (fastest machines) has improved by a factor of 10^6, the algorithm also by 10^6. The total net improvement is 10^{12}...
     ex. buble, shell sort → quick sort
     ex Fourier Transform → Fast Fourier Transform
     $t \div N^2$ $t \div N \log_2 N$

4. Code improvements :

- Extrat non changing statements from loops
- Test first most probable stopping condition
- unroll small loops
  replace procedure calls by inline code (demodularize...)
- replace multiple indexes per single (1D) ones.
- replace sequential indexing with incremental pointers
- avoid type conversions
- avoid errors/situations that cause exception handling
- know your hardware strength and weakness, and adapt yourself
- replace / with * ; ** with * , * with + .
- use table lookup for functions (very effective!)
- if your memory is accessed by page (virtual ek.) keep working set smal, avoid paging

A factor of 5-10 is not impossible (neither warranted!)

□

PS. Do not improve the "sleep" loop!