EE4390 Microprocessors

Lesson 8

Structured Programming

Overview - Structured Design

- If this worth my time - a parable
- The “divide-and-conquer” technique
- Requirements
- Partitioning - “The Black Box”
- Structure Chart
- Pseudo (Fake) Code
- Implementation Techniques
- Testing Techniques
- Documentation
- Unified Modeling Language (UML)

If this worth my time - a parable

- Long, long ago in a graduate program far, far away…
  - needed to learn C
  - couldn’t pass entrance quiz
  - took prereq course is Pascal
  - course was supposed to be Pascal & data structures
    - discussed structured design techniques!?
  - my view changed!

The “divide-and-conquer” technique

- Paper writing/Book writing
  - Solid outline allows “big picture” view
  - Write project a paragraph at a time
- Use same technique in SW/HW design
  - divide project into understandable, doable pieces
  - A.K.A.: top-down-design, bottom-up-implementation...

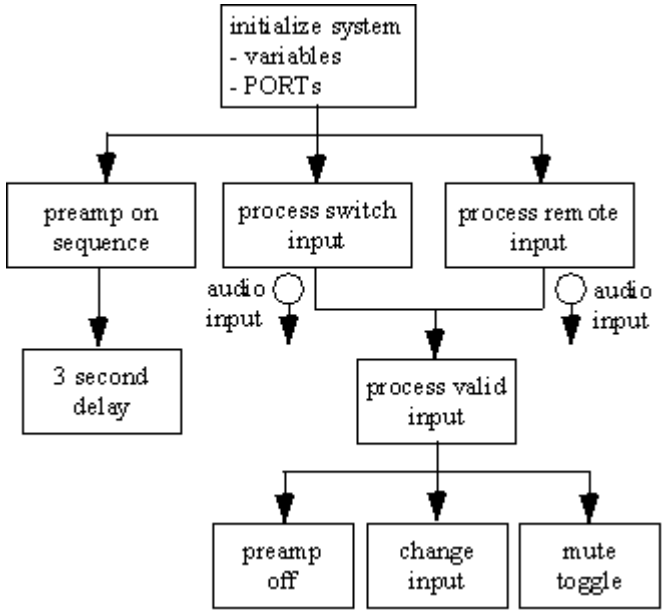
Requirements

- Overall goal of structured design is to provide tools to transform system requirements into a plan into implement a system
- Your responsibility to ensure you understand requirements
  - iterative process with customer

Partitioning - “The Black Box”

- Break a large, complex system into a hierarchical description of “black boxes”
  - “black box”: small definable pieces
    - know inputs, outputs, general details of function
  - define relationship between “black boxes”
    - use a graphical tools relationship
    - Structure Chart provides big picture

Structure Chart



Pseudo (Fake) Code

- Once hierarchy is defined begin working out details of black box.
- Develop functional relationship between the boxes' inputs and outputs
- Use pseudocode to defer details
  - not trying to avoid details
  - defer until higher level details worked out

Implementation Techniques

- Incremental Approach - get a little bit working at a time
- Top-down: implement top module (e.g. menu software)
  - lower level code simulated with stubs (empty modules)
- Bottom-up: implement module at lowest level.
  - Higher level code simulated with drivers
- Hybrid: use of mixture of both techniques and meet in the middle

Testing Techniques

- Compile time errors
- Run Time errors
- Everything is O.K. except project completed to incorrect requirements!!!
- Test Plan

Documentation

- External documentation - support information
  - Structure Chart
- Internal documentation
  - Comments
  - Self-documenting code - wise choice of variable, function names
  - Program Formatting - “pretty printing” - use blank spaces to help illustrate the control structure of the program

Unified Modeling Language (UML)

- Standardized set of graphical tools to model a complex system prior to implementation
  - fundamental property -- communication!
- Used to describe object-oriented design
- Activity Diagram -- UML-compliant flow chart

Unified Modeling Language (UML)