REAL-TIME OPERATING SYSTEM FOR SELECTED INTEL PROCESSORS

W. R. Pool
Ford Aerospace
Houston, Texas

This paper addresses development of a real-time operating system for selected Intel processors, in terms of four project phases.

The first phase develops the system development rationale. Included are reasons for not using vendor supplied operating systems. The second phase deals with the system design and performance goals. While many of these goals are dictated by problems with vendor supplied systems, other goals surfaced as a result of a design for a custom system able to span multiple projects. The third phase deals with system implementation. Discussed are system development and management problems and areas that required redesign or major code changes. The final phase deals with project results. First, the relative successes of the initial projects are developed. Then, a generic description of the actual project is provided. Finally, the ongoing support requirements and future plans are discussed.

Preceding Page Blank

Make/Buy: Requirements Make the Decision

- First, determine what system must do
  - -- Real-time - quick response
  - -- Multi-tasking - priority scheduling
  - -- High performance, especially for I/O
  - -- Disk file system compatible with ISIS-II
  - -- Block mode CRT support
- Selection: what will do the job ?
  - -- Does system exist that meets requirements ?
  - -- Can a system be modified at less expense/time ?
- Developing your own
  - -- Does talent exist ?
  - -- Is there time/money ?
  - -- Are development facilities available ?

System Design: A Real Experience

Design Goals

- Modification ease: add foreign devices
- Modular: use only necessary parts
- Standard device interfaces to applications
- Support any board/device configuration

System Design: A Real Experience (Cont'd)

Control Element Components

Control Program	Control Block
• Task Management	Task Control Block (TCB)
• Event Signaling	Event Control Block (ECB)
• I/O Control	File Control Block (FCB)
• Interrupt and Device Handling	Device Control Block (DCB)

System Design: A Real Experience (Cont'd)

Utility Functions

- Display support
- Disk file management
- Loader
- Operator interface
- Advisory facilities
- Debug monitor

Additional Services

- Timer services
- Dynamic memory allocation

Implementation: No Easy Victories

The Team

- Designer: control block contents, program functions
- Programmer: program design, development

The Plan

- Develop task management, interrupt service, and I/O initiation first
- Validate control block structure
- Correct any control block and design errors
- Rewrite as necessary
- Develop device support and utilities

Implementation: No Easy Victories (Cont'd)

The Problems

- Development systems slow and unreliable
- Processor poorly suited for multi-tasking and re-entrant code
- Early delivery resulted in system not fully checked out
- Tight bindings must be replaced with loose binding to separate OS from applications
- User documentation slow/non-existent
- Disk support control block formats must be changed to support different densities
- Dynamic memory support should have been included in original design
- Needed to fund support group as user group increased
- Configuration control problems due to different releases on projects and inadequate library facilities
- Should have had cross software and timesharing network at beginning

Results

Met All Requirements and Design Goals

- Currently used in three systems; being implemented in four others
- Being used in different hardware configurations
- Interrupt processing capabilities faster than other available systems
- Achieved true multi-tasking capabilities

Cost Effective

- Effort and budget savings via use on multiple projects
- No license fees

Results (Cont'd)

System Characteristics

- 6 months to produce first version; 1 year to produce debugged version in macro form
- 4K PROM for basic system with task management, I/O support, and loader
- 8K PROM adds display support, debug monitor, advisory, and full disk support
- 4-12K of RAM necessary for buffers and control blocks
- Normally will support CRT, line printer, and two flexible disk drives
- Systems usually embedded in larger equipment systems

Future Plans

- Recode for different processor
- Change user interface to loose binding type
- Augment dynamic memory support
- Convert to higher level language
- Provide support for users/projects
- Redesign disk support/control block
- Provide software library on host development system