ECE354: RTX Project Final Report

Ben Ridder - brridder
Casey Banner - cccbanne
David Janssen - dajjanss

July 23, 2011

Contents

Chapter 1

Software Design

1.1 Introduction

1.2 Global Information

1.3 Primitives

1.4 Processes

1.5 Software Interrupt Handlers

1.6 Hardware Interrupt Handlers

1.7 Hot Keys

A total of five hot key commands were implemented to assist in debugging and testing of the system and user processes. A hot key decoder function was used to interpret the inputted command from the UART interrupt process. The interrupt process checks the first character for an exclamation mark ('!') and calls the hot key function directly with the input string as a parameter. See Table 1.1 for a summary of the hot key strings and descriptions. All of the code related to the debug hot keys are wrapped in `#ifdef _DEBUG_HOTKEYS` and `#endif` to ensure that in normal operation of the system that this code will not interfere.

Table 1.1: Hot Keys and Descriptions

Hot Key Command	Description
!RQ	Print ready processes and priorities
!BMQ	Print processes blocked on memory
!BRQ	Print processes blocked on receiving messages
!FM	Print the current number of free memory blocks
!M	Print the last i messages sent where i is the debug message log size

Each of the debugging functions are kernel level calls as they require access to data structures in the kernel. This is done using the same soft interrupt method as the the rest of the kernel routines. The hot key command decoder uses a similar parsing method as the `process_wall_clock()` process to interrupt which kernel routine to run. The debugging processes related to the ready queues and blocked queues use the kernel print queue function as described in Listing 1.1. The specific queue debugging function simply passes in the relavent queue into the queue print function.

Listing 1.1: Kernel Print Queue Psuedo Code

```
int queue_debug_print(process_queue queue[]) {
    for each process in the queue:
        print process ID
        print process priority
}
```

The hot key, `!FM`, simply outputs the number of free blocks and the allocation bit field. The allocation bit field is described in Section **??**.

The last hot key, `!M`, is the most complicated of the five.

1.8 Initialization

1.9 Implementation

Chapter 2

Major Design Changes

Chapter 3

Measurements

Chapter 4

Leasons Learned