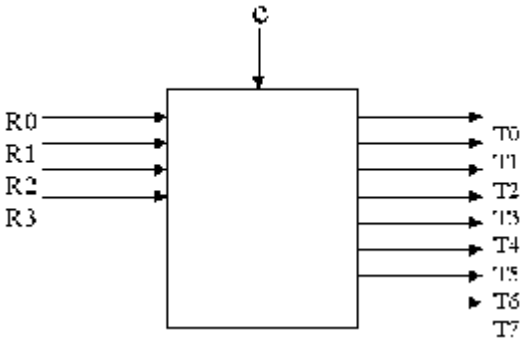
Name:_____

CSE466 Quiz 2, 11/13/00, Open Book, 50 Minutes

Embedded Software Architecture



Your company is designing a special microprocessor for network routing applications and wants to know what performance can be expected. The proposed architecture has four input-only UARTs (serial interfaces) R0-R3 and eight output-only UARTs T0-T7, as shown in the figure. The processor also has a one input-only UART, C, to be used for control purposes. Here are some technical details:

Software Requirements

1. The main function of the processor is to transfer data received on an input port to the appropriate output port as determined by a route table. For example if ROUTE[i] == j then any byte received on input port Ri should be transmitted on output port Tj. No buffering is required and flow control is managed in hardware as described below.

2. Any byte received on port C specifies a routing rule. The top four bits of the byte is the input port, and the bottom four bits is the corresponding output port. You can assume that two inputs are never routed to the same output at the same time.

Hardware Specifications (designed for easy programming…really)

1. Each of the 13 UARTs has a separate interrupt vector location. The interrupt enable control bits are RI0-RI3, TI0-TI7, and CI. The corresponding read-only interrupt request flags are RF0-RF3, TF0-TF7, and CF.

2. For each input UART, RFn is set when a byte is received, and is cleared whenever the software reads Rn (the corresponding receive register). Due to hardware flow control, you can assume that external transmitters will not send another byte until Rn is read (RFn is cleared). No need to send ACK bytes.

3. For each output UART, TFn is set only when the attached receiver is ready to receive another byte. TFn is cleared when the data buffer Tn is written by the software. An attempt to write to Tn when TFn is still clear will block until the receiver becomes ready (this is HW flow control). No need to receive ACK bytes.

Name:_____

Part 1

A. Using round-robin scheduling with no interrupts, write a C program for this system. For simplicity, your compiler is capable of bit-indexing, so TF1 is the same as TF[i] when i = 1. Likewise R1 is the same as R[i] if i = 1. This means you can use a for-loop if you want. Your processor does not have enough

B. Assuming a 32MHz oscillator on an 8051 type architecture, what is the best case throughput (bytes/sec) and worst case latency (sec) of your system? Draw one or two task timing diagrams to illustrate BOTH the maximum total throughput possible for the system as well as the worst case input-to-output latency. For the purposes of worst case latency, assume that the TFi flags are always set (receiver is always ready). Your calculations and timing diagrams should reflect reasonable estimated clock cycle timing of your software and should clearly show the relationships between external and internal events.

C. List at least two major weakness of this architecture (round-robin w/ no interrupts).

Name:_____

Work Page and Template for Timing Diagrams

Name:_____

Part 2

A. For this application, which of the below architectures would you pick as the best alternative to the above.

1. Round-robin scheduling as above, but with the use of interrupts (no OS).
2. A preemtive dynamic scheduling kernel like RTX-full that supports task priorities and interrupts.
3. A user-space Linux program with device drivers for accessing the I/O ports.

B. Explain how your selection would address the weaknesses identified in Part 1. Does your choice introduced any new weaknesses to the system?

Extra Credit: Write the application code in your chosen architecture.