CSE 379

Interrupts

A Gentle Introduction

Polling vs. Interrupts

- Polling
  - Repeated querying of device to see if data is available
  - Inefficient
- Interrupts
  - Allow program to do useful work
  - Interrupt the processor when data is ready
  - Control automatically transferred to interrupt handler routine
    - NO BL needed!
    - Happens in hardware!
  - Clear Interrupt
  - Handle interrupt
  - Return

What Generates Interrupts?

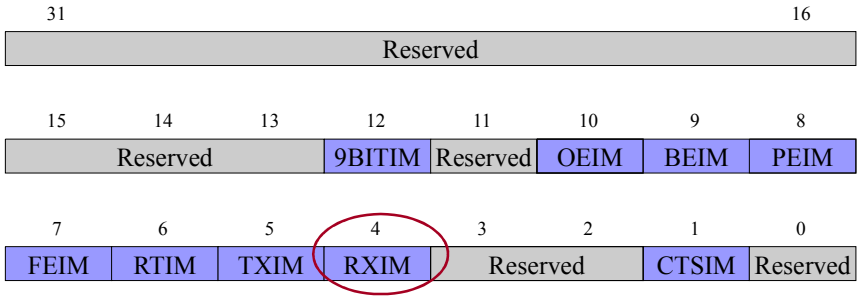
- The CSE 379 specific answer
  - The UART
    - When data is received, interrupt the processor
  - The Switch (SW1)
    - When pressed, interrupt the processor
  - The Timer
    - When the count reaches a certain value, interrupt the processor
    - Allows for creating timed, reoccurring events

How Are Interrupts Programmed?

- Setup each device so it is configured to interrupt the processor
- Setup the processor to enable interrupts for a device
  - If not enabled, a device may try to interrupt the processor, but that interrupt will be ignored
- Use new startup code
  - tm4c123gh6pm_startup_ccs.c
  - Available on Labs page of course website

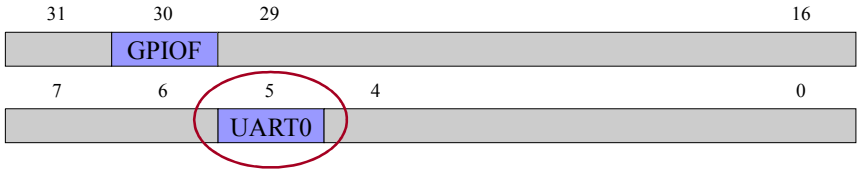
Configuring the UART for Interrupts

- Set the Receive Interrupt Mask (RXIM) bit in the UART Interrupt Mask Register (UARTIM)
- Details
  - UART0 Base Address
    - 0x4000C000
  - UARTIM Offset
    - 0x038
  - RXIM Bit Position
    - Bit 5



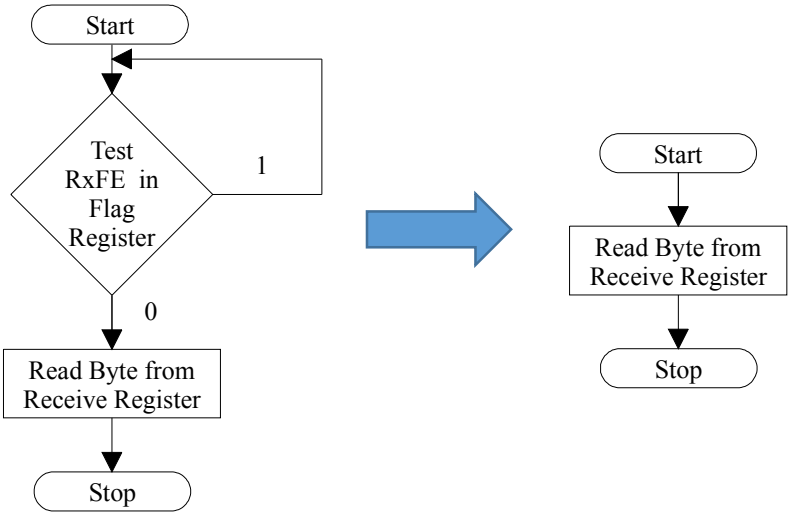
Configure Processor to Allow UART to Interrupt Processor

- Set the bit 5 bit in the Interrupt 0-31 Set Enable Register (EN0)
- Details
  - EN0 Base Address
    - 0xE000E000
  - EN0 Offset
    - 0x100
  - UART0 Bit Position
    - Bit 5



The Effects on read_character

- Polling no longer required
- Just read from the UART0 Data Register
- read_character is transformed



The Basic Procedure

- Your program runs normally
  - When an interrupt occurs, the processor is interrupted
  - Program is temporarily suspended
  - Control is transferred to interrupt handler (UART0_Handler)
    - Occurs automatically in hardware
    - NO branch and link to interrupt handler!
    - Processor mode is changed (to Handler Mode)
    - Interrupt is handled
      - ✓ Registers r4 through r11 must be specifically preserved across the Handler
      - ✓ Registers r0 through r3, r12, LR, & PC don't need preservation
        - The processor handles this automatically
    - Mode is returned to what it was when interrupt occurred
    - Control is transferred back to interrupted program

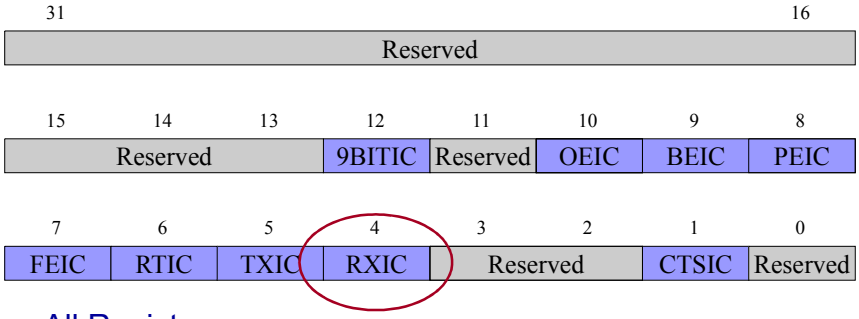
    - ✍ It is important to note that when the interrupt is handled, it must be transparent!
      - ✓ In other words, it must not appear that the program was interrupted.
        - Program state must be maintained
        - Any registers used must be restored to their original state

What Happens When an Interrupt Occurs?

- Processor transfers control to handler for specific interrupt
  - UART0_Handler

Interrupt Servicing in the Handler

- Clear Interrupt
  - Set the bit 4 (RXIC) in the UART Interrupt Clear Register (UARTICR)
    - Details
      - ✓ UART0 Base Address
        - 0x4000C000
      - ✓ UARTICR Offset
        - 0x044
      - ✓ UART0 Bit Position
        - Bit 4



- Handle Interrupt
  - Remember to Save & Restore All Registers
    - When returning from an interrupt service routine, the interrupted program must continue to execute properly!
- Return to Interrupted Instruction
  - BX lr
  - Do NOT use MOV pc, lr

Testing Your Code

- In your main routine (lab5)
  - Setup a loop, waiting for the keystroke to put you into your handler
  - Set a breakpoint at the handler
    - When a keystroke occurs, the breakpoint at the handler should be hit
  - After handling the interrupt, you'll be placed back into that loop

References

- Kris Schindler, *Introduction to Microprocessor Based Systems Using the ARM Microprocessor*, Second Edition, Pearson, 2013
- Muhammad Ali Mazidi, Shujen Chen, Sarmad Naimi, Sepehr Naimi, *Programming ARM Corect-M4 TM4C123G with C*, First Edition, MicroDigitalEd, 2014-2016
- Texas Instruments Incorporated, *Tiva™ TM4C123GH6PM Microcontroller Data Sheet*, June 12, 2014, Texas Instruments – Production Data, 2007-2014
- *Alice EduBase for Tiva and MSP432 Launchpad User's Guide*, Version 1.21, EVB+, February 4, 2017
- ARM Ltd., *Cortex–M4 Devices Generic User Guide*, DUI0553A, ARM, 2010
- http://www.arm.com