COMP 4550
Real-Time Systems
Timers

Autonomous Agents Lab, University of Manitoba
jacky@cs.umanitoba.ca
http://www.cs.umanitoba.ca/~jacky
http://aalab.cs.umanitoba.ca
Timers

- In many applications, we must keep track of the elapsed time
  - Enforce constraints on deadlines etc.
  - Double click vs bounce
- Timers are automatic counters
  - Increment a value in a register with a given frequency
- Register can be 8 (AVR), 16 (AVR), or 32 bits
- Frequency is determined by
  - Processor frequency
  - Pre-scalar divider: 1, 8, 32, 64, 128, 256, 1024
ATMega 128 Timers

- ATMega128 has four timer/counters
  - Timer 0: 8 bit
  - Timer 1: 16 bit
  - Timer 2: 8 bit
  - Timer 3: 16 bit
  - 16 bit timers have more complex functionality and longer count periods
- Timers usually have some additional functionality
  - Generate a PWM signal
  - Generate periodic interrupt
  - Auto-reload on match
Setting Up a Timer: Timer 0/2

- **Clock/Counter Source**
  - Internal at fixed frequency
  - External asynchronous

- **Frequency**
  - Prescalar

- **Operation Mode**
  - Normal mode
    - Count up
    - Enable interrupt on overflow 0xff-0x00
  - Clear Timer on Compare Match (CTC)
    - Count down until TCNTn matches OCRn
    - Then clear TCNTn
  - Fast PWM Mode
  - Phase Correct PWM Mode
Timers: Timer/Counter 0/2

- Data register (TCNT0 – 8 bit)
  - contains current “time”
- Always clear data register before you start timer
- Config register(s) TCCR0A
  - set frequency
  - enable/disable irq
  - start/stop timer (by setting freq to 0)
- Overflow interrupt TIMSK0, TIFR0
  - when register rolls over from 0xFF to 0x00
- Take overflow into consideration when calculating time
- Timer synchronization TSM
  - Start Timer 0 and 1 at the same time
Timers: Timer/Counter 1/3

- The 16 bit timers have more functionality
  - 3 separate PWM channels
  - Input capture module
- Accessing 16 bit registers
  - Takes two instructions to read or write a 16 bit value on an 8 bit microcontroller
  - Can corruption occur?
  - How to avoid corruption?
  - ATMega128 uses a hidden 8 bit tmp register
    - Shared across the timers
  - Therefore
    - Write: the high byte must be written before the low byte
    - Read: low byte must be read before the high byte
    - Done by the C compiler if you use 16 bit access
  - 16 bit read/write must be atomic if ISR also accesses 16 bit registers
Timers

- Use timers to determine
  - event ordering (when did what happen)
  - control when things happen (e.g., set PB3 to H)
- Limited number of timers in a uC
  - share timers
    - usually use one common heartbeat (e.g., 1ms) from one timer
  - operating system resource
    - RTOS must assign timers to tasks/threads
Measuring Time

● Determine amount of time between two events
  ○ Record time and calculate difference
  ○ Must track overflows
  ○ Must determine number of overflows

● Synchronization problems
  ○ Timer updates register during read
  ○ Overflows during read