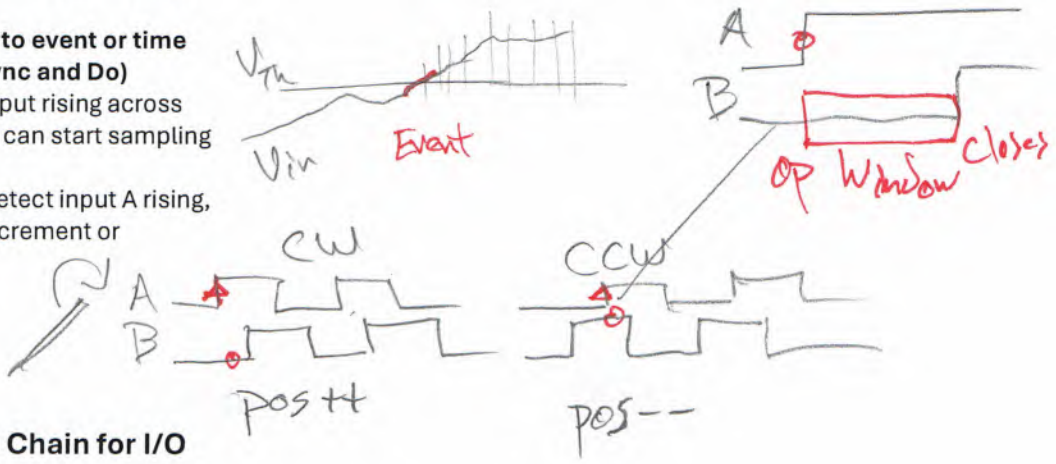
Lecture 03 Notes –
I/O, Timing and Synchronization

I. Overview

A. **Timing requirements for I/O activities are major driver for embedded system design decisions**

B. **May need to synchronize to event or time before doing the work (Sync and Do)**

1. Scope trigger: detect input rising across threshold voltage, then can start sampling data
2. Quadrature decoder: detect input A rising, then sample input B, increment or decrement count



II. Understanding Process Chain for I/O Activities

A. **Synchronize with something**

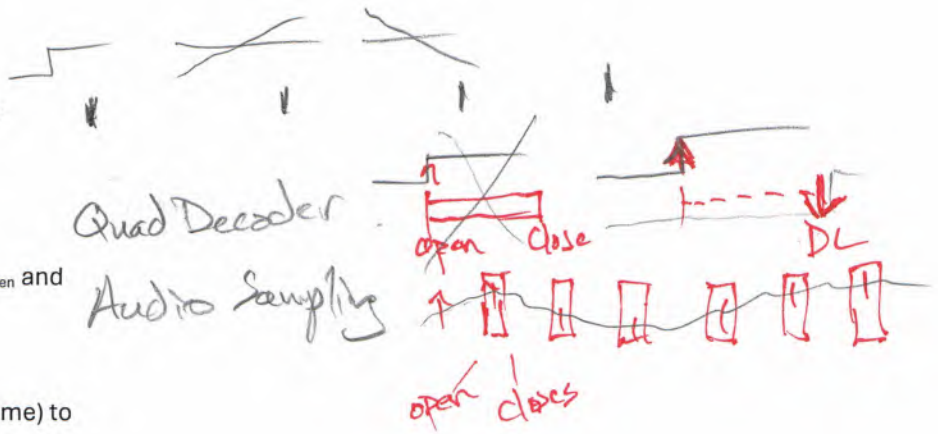
1. Types
   a. Event-Triggered: Detect event
   b. Time-Triggered: Await target time

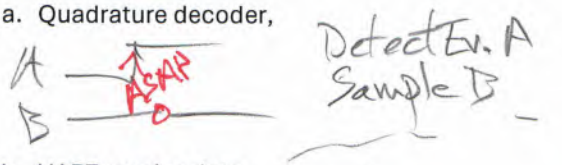
B. **Do processing in response**

1. Timing requirements:
   a. Simple deadline: within T_{DL} of event/time
   b. Window deadline: Between T_{DL_Open} and T_{DL_Close} of event/time

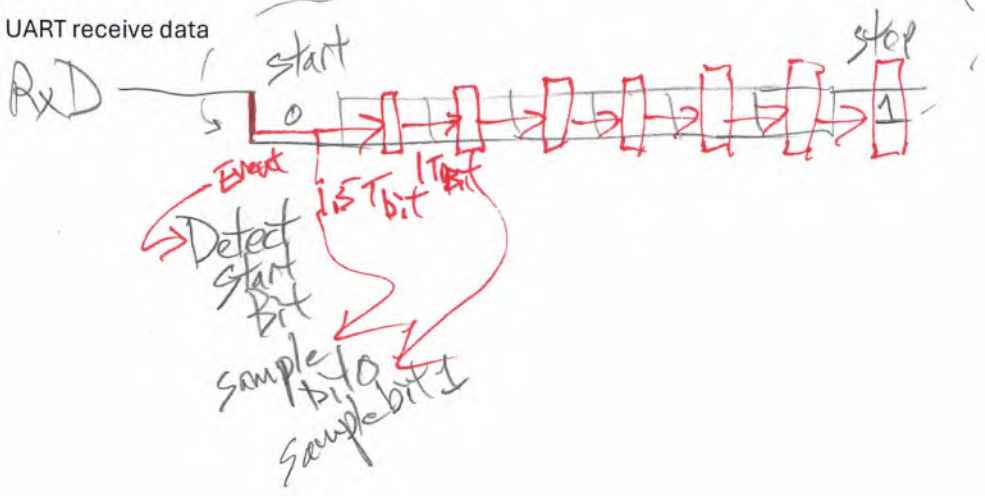
C. **Repeat?**

1. May have burst or sequence of I/O activities, so next will sync (event or time) to next part or do it immediately/ASAP
2. Examples inputs:
   a. Quadrature decoder,





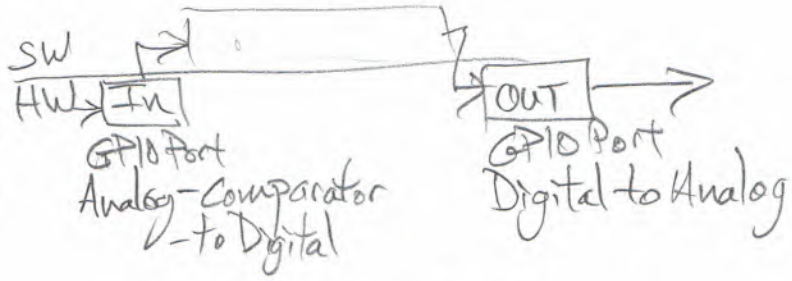
   b. UART receive data



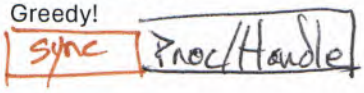
III. How to Synchronize?

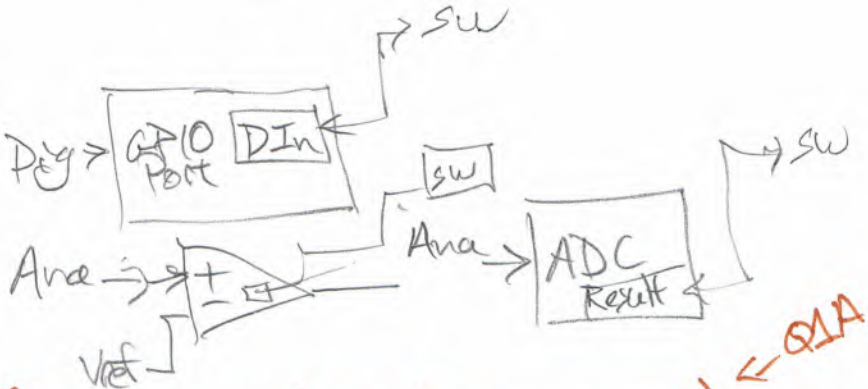
A. **All Hardware**
   1. Easy: Dedicated signals

B. **Some Software**
   1. HW/SW allocation and processing chain. SW polls hardware (input peripheral)
   2. Hard, since software timing is sloppy, gets even harder when sharing CPU
      a. Timing variation diagram (ramp), sync to stabilize/cut timing variation



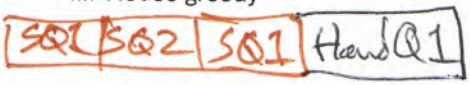
   3. Start simple: Not sharing CPU
      a. Detect with blocking SW loop polling (busy-waiting)
      b. Responsiveness
      c. Greedy!

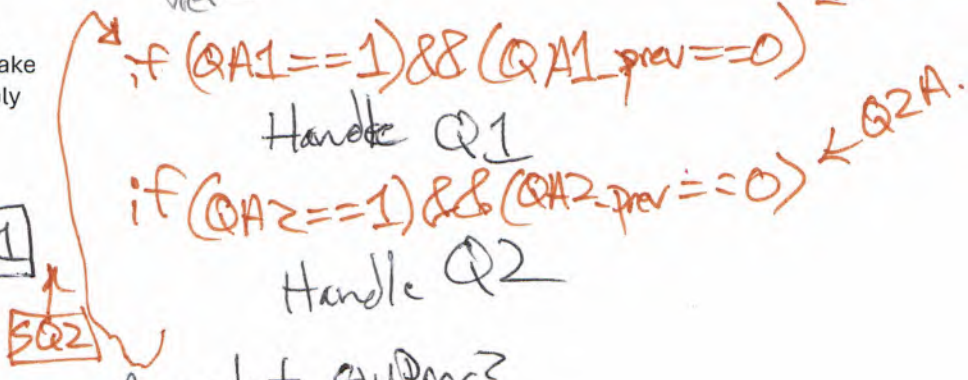




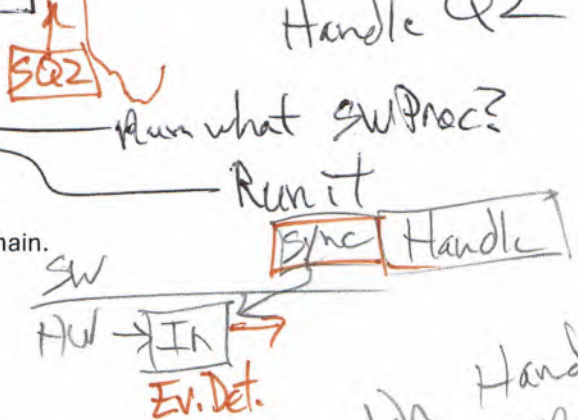
   4. Share CPU with software scheduling method
      a. Round-Robin Loop/Cyclic Exec.
         i. Detector doesn't block, but take turns with other code (possibly multiple detectors)
         ii. Responsiveness
         iii. Not so greedy

non-blocking



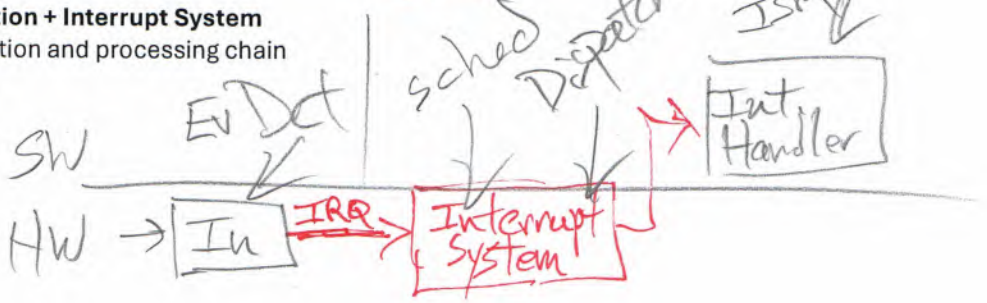


      b. Many other sharing options. Prioritization, preemption ...
         i. + Schedule, dispatch.



C. **HW Event Detection**
   1. Hardware peripheral detects event
   2. HW/SW allocation and processing chain. SW polls event detector

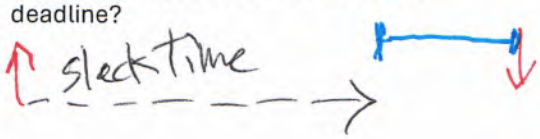
D. **HW Event Detection + Interrupt System**
   1. HW/SW allocation and processing chain
   2. Handler runs



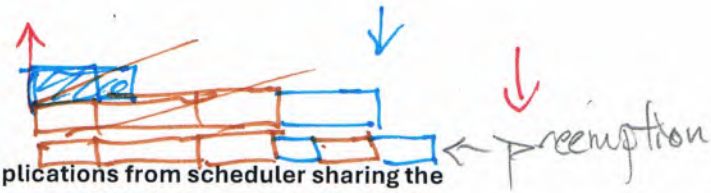
IV. Basic Timing Analysis

A. Approaches

1. Slack time
   a. How late can process start and meet deadline?

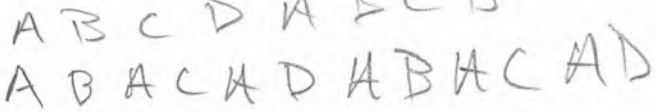


2. Response time
   a. When will this process finish, considering effects of other processes in system



B. Complications from scheduler sharing the CPU among SW processes scheduler

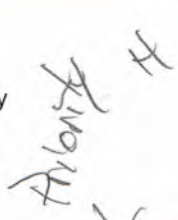
1. Basic: static fixed schedule



2. Dynamic scheduling – different orders possible
   a. Prioritize SW procs
      i. Static or dynamic?
      ii. Timing-based or other?

3. Preemption of SW proc
   a. By interrupt service routines
   b. By other SW processes

4. Results: timing delays
   a. Interference by same, higher-priority SW processes
   b. Blocking
      i. Non-preemptive scheduler
      ii. by lower-priority SW processes sharing resource with this process



HPrio Task - Interference Task

Blocking Low-Prio Task - Shared Resource Schedules (Not preemptive)