Lecture 05 (Interrupt and Timer)

1 Interrupt

1. Alternate to polling
2. Saves overhead of having to poll every time
3. Hardware based process
4. Processor reacts to interrupt from interrupt handler, whenever it comes
5. Usually have a single interrupt handler (H/W component), which signals the processor about all incoming interrupts
6. Interrupts can be delayed/priortized based on code execution path

1.1 Working of Interrupt Handler

1. $\exists$ Interrupt Service Routine (ISR) which reads the interrupt register and calls interrupt handler code
2. To prevent same interrupt firing multiple times, two more registers are defined
   - Interrupt Mask Register (IMR)
   - Interrupt Pending Register (IPR) - actually where the interrupt is read from
3. Order of interrupts wrt bit position helps with priority

2 Timer

1. One trivial timer - clock
2. Implemented using an oscillator (RLC ciruit)
3. Any other timer is implemented using a counter on the clock

2.1 Inaccuracy

1. Error in clock is measured in ppm (#seconds off in 10^6 seconds)
2. Clock accuracy decreases over time
3. 25 ppm is considered decent
4. Inaccuracy is used to identify the source of a signal (helps in identifying malicious senders)
5. Useful metric since inaccuracy is difficult to mimic

2.2 Timing

1. When reporting time taken, need to be careful of overflows
2. Can use an interrupt to catch overflows
3. Can count the number of times overflow occurs using the interrupt