EECS192 Lecture 12
Apr. 12, 2016

Notes:

1. Check off 4/8: practice course, 5 min
2. Mon. 4/25: (430-530 pm) round 2 (NATCAR rules)
    1. 13 makes first turn
    2. 15 half track in < 5 minutes
    3. 18 whole track in less than 1 minute
    4. > 18 For cars which are fast and/or well-stabilized
3. CalDay Sat. April 16 @ UCB, 0900 Courtyard (in class room if rain)
4. Lab share Tues 5-7 pm, all of April. Also two benches
5. Quiz 5 on 4/12 on steering control

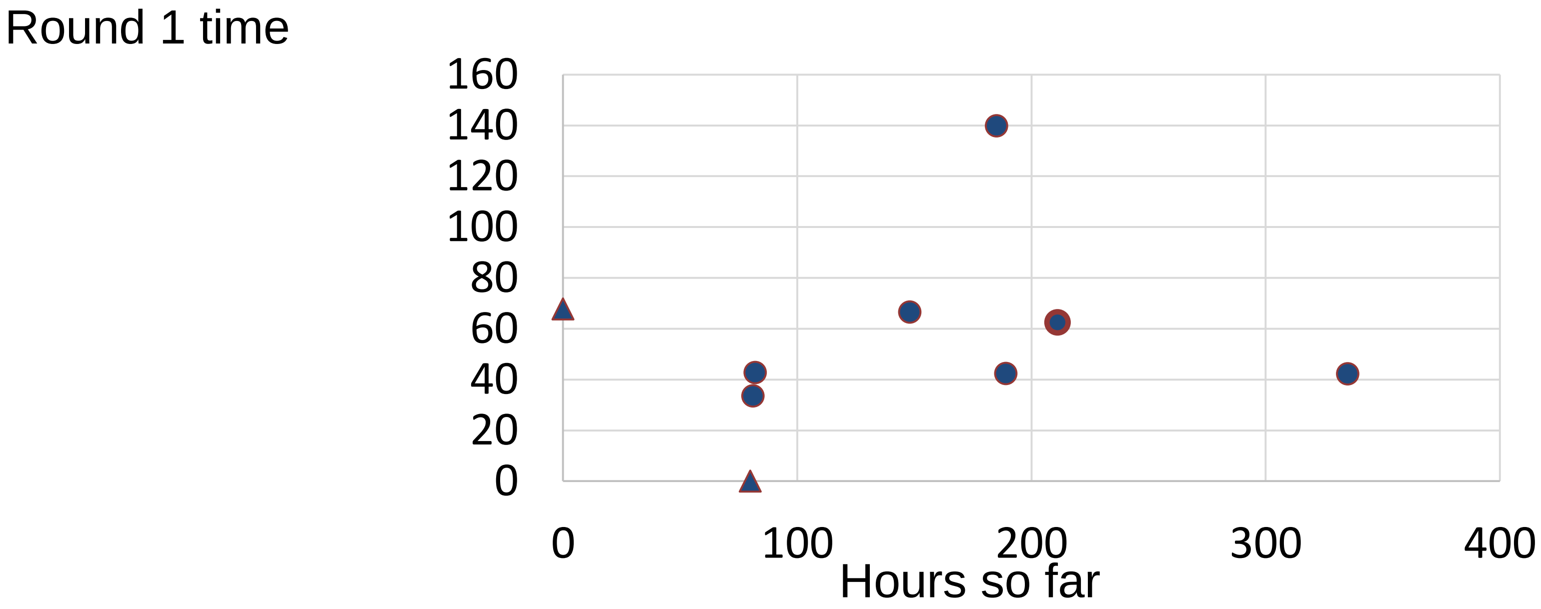
Topics

- Round 1 results
- Progress report notes: Hamamatsu
- Hardware Robustness
- C.O.P. Watchdog
- Digital Filtering
- Software Robustness
- Supervisor Systems

Round 1 Results

- **2015**: 48.14, 46.56, 67.93, DNF, 53.72, 85.51, 40.37, 40.24, 41.61, DNF, 62.88, 45.59

- **2016**: 33.61, 42.32, 42.48, 42.83, 55.14, 62.59, 66.72, 67.69, 140.01, DNF,



Progress report notes

- 81,82,82,148,185,189,211,335 hours

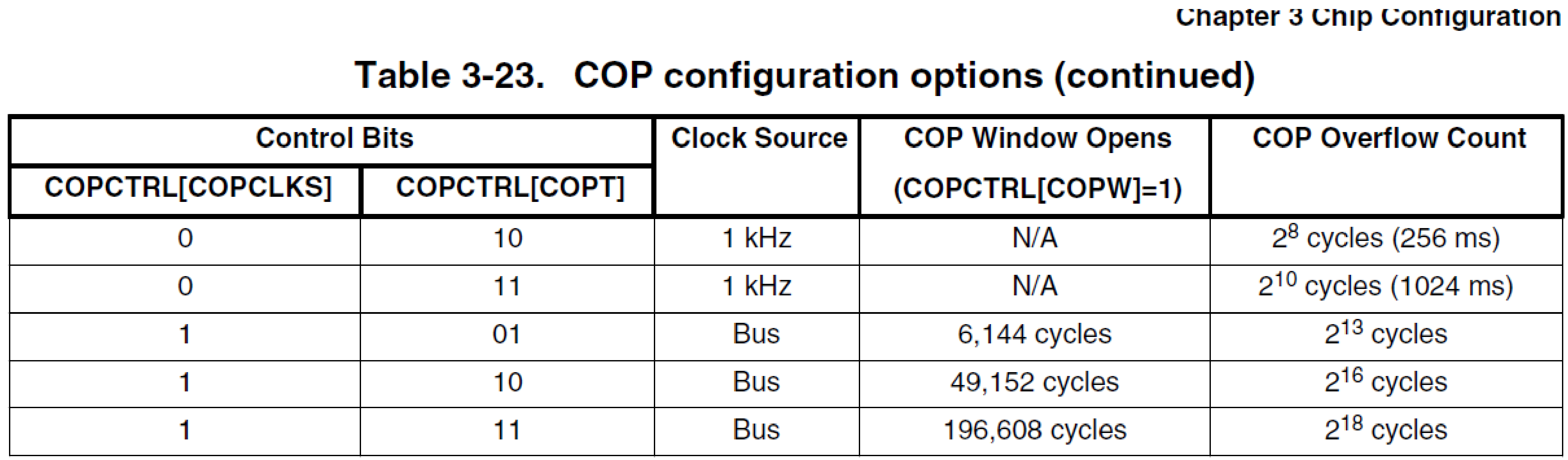
- Hamamatsu: 5 volts, be careful
- Motor snubber caps

Hardware Robustness

- Mechanical oscillations
- Lock washers
- Strain relief on connections
  - (stranded vs solid core wire)

C.O.P. Watchdog timer

- Despite extensive software and hardware testing, faults will still occur in real devices. Even momentary noise spikes on a power supply can lock up a processor occasionally. Such events will occur on the power grid several times a year. Watchdog timers provide a last line of defense to prevent system failure with minimal hardware cost.
- https://developer.mbed.org/cookbook/WatchDog-Timer



Chapter 3 Chip Configuration

Table 3-23. COP configuration options (continued)

Control Bits		Clock Source	COP Window Opens	COP Overflow Count
COPCTRL[COPCLKS]	COPCTRL[COPT]		(COPCTRL[COPW]=1)	
0	10	1 kHz	N/A	2^{8} cycles (256 ms)
0	11	1 kHz	N/A	2^{10} cycles (1024 ms)
1	01	Bus	6,144 cycles	2^{13} cycles
1	10	Bus	49,152 cycles	2^{16} cycles
1	11	Bus	196,608 cycles	2^{18} cycles

Need to change systemInit:

```
void SystemInit (void)
{ #if (DISABLE_WDOG)   /* Disable the WDOG module */
 /* SIM_COPC: COPT=0,COPCLKS=0,COPW=0 */
SIM->COPC = (uint32_t)0x00u;
#endif /* (DISABLE_WDOG) */
```

```
// Kick (feed, reload) our watchdog timer
 void wdt_kick()
{
 SIM->SRVCOP = (uint32_t)0x55u;
 SIM->SRVCOP = (uint32_t)0xAAu;
 }
```

Digital Filtering

- Moving average
  - $y1[n] = (y[n-2]+y[n-1]+y[n])/3$
- Median filter (outlier rejection)
- Notch filter (mechanical vibration)
  - $y[n] = (x[n-2]+2x[n-1]+x[n])/4$

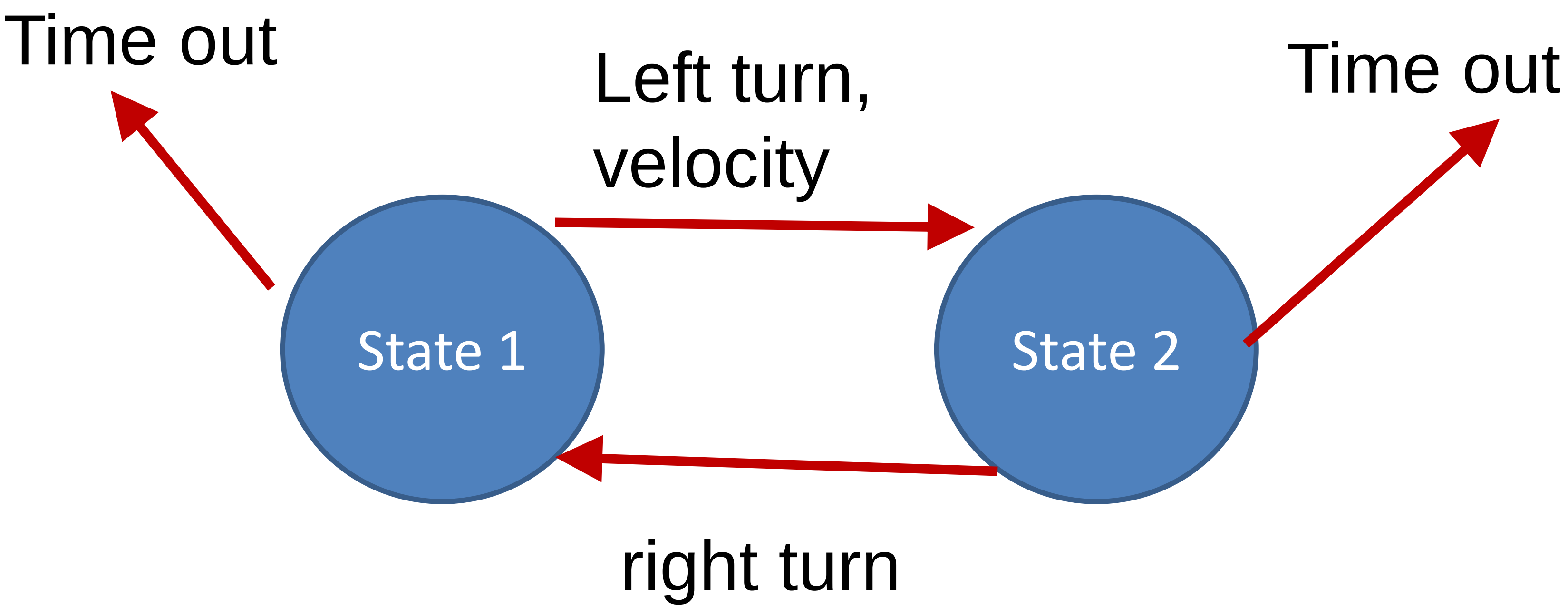
- Model based filtering (or Kalman filter)

(on board)

Software Robustness

- Checksums for bit rot
- Watch dog timer/computer operating properly COP
- Lost track detection
- Autocalibration at startup
  - (sanity check for steering angle vs line error)
  - AGC
- State Observer/estimator
- Discrete State observer

FSM Recognizer (generalized WDT)



Time out

Left turn,
velocity

Time out

State 1

State 2

right turn