Timestamp Temporal Logic (TTL) and Testing Methodology for Monitoring the Timing of Cyber-Physical Systems

Dr. Patricia Derler, Research Scientist

National Instruments

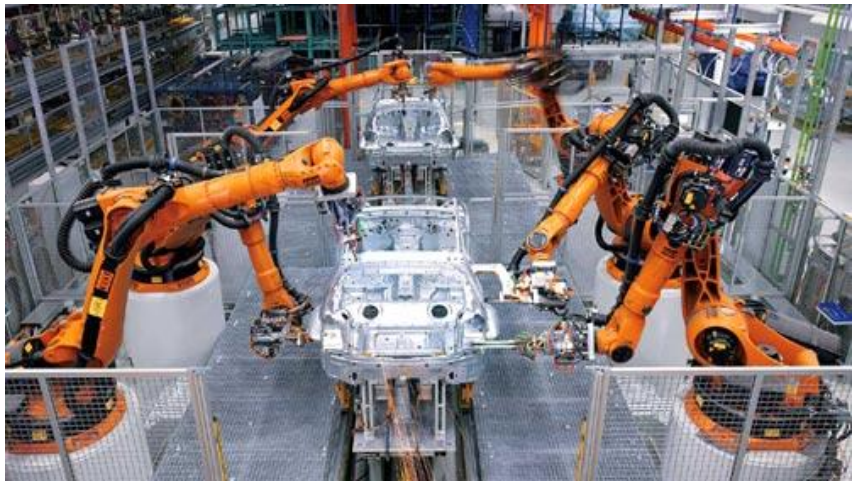
Joint work with:
Aviral Shrivastava, Mohammadreza Mehrabian, Mohammad Khayatian (ASU),
John Eidson (UCB), Patricia Derler (NI), Marc Weiss (Marc Weiss Consulting),
Edward Griffor, Ya-Shian Li Baboud, Dhananjay Anand (NIST), Hugo Andrade (Xilinx), Kevin Stanton (Intel)

CPS: Exciting times

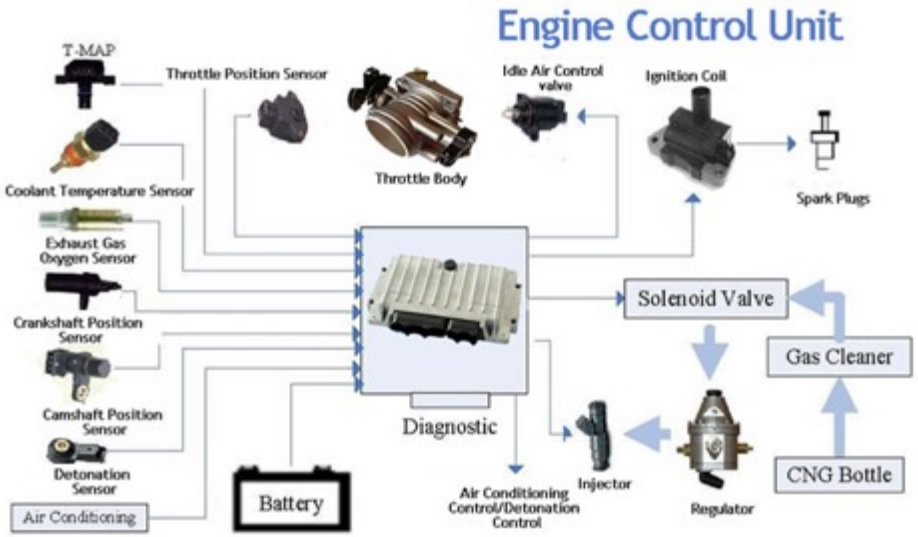
- Sensors → sensor networks
- Embedded systems → networked embedded systems
- Control has come to software
  - *Mechanical → Electrical → Computer*

The real question for our times is:
If I can sense anything, and I can control everything, what (good) can I do?

- Nexus of CPS, Machine Learning and Big Data
  - *Enable large scale and compelling applications*
  - *Autonomous cars, Smart cities, City-level traffic management, Smart grid*



Timing is fundamental to CPS



- Tight sense-compute-actuation loop
- Hard-real-time CPS
  - Correctness depends on functionality as well as correct **timing** [1].
  - Failure of timing can lead to catastrophe!
  - e.g. Autonomous cars
- Timing is important even in the non-real-time case
  - Timing predictability can lead to performance improvements
- Timing constraints come from
  - System stability requirements
  - System performance requirements
  - Legal requirements

[1] Shrivastava, Aviral, et al. "Time in cyber-physical systems." *Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2016 International Conference on*. IEEE, 2016.

Overview of this Talk

- Importance of Timing in CPS
  - CODES+ISSS 2016
- How do we specify the timing constraints of a distributed CPS?
  - EMSOFT-TECS 2017
- How can we monitor if a CPS is meeting its timing constraints?
  - DAC 2018
- A testbed for monitoring timing constraints
  - Reconfig 2015
- CPS Testbeds
  - Flying paster, synchronized cameras, phase synchronization over the internet, intersection for autonomous cars
- Timing Health Monitoring and Management System

How to specify timing constraints of CPS?

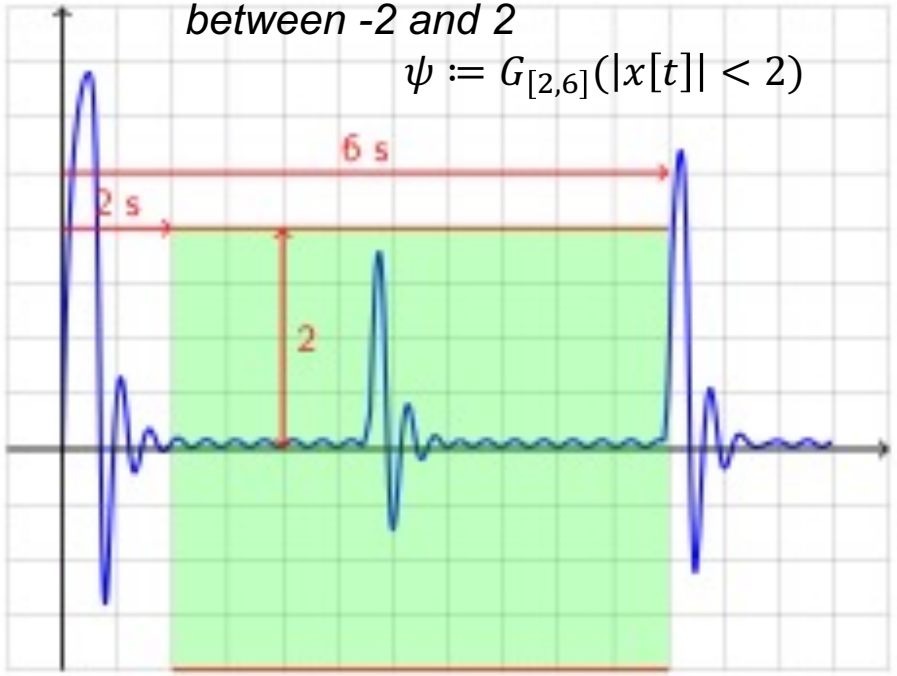
Instead of specifying timing constraints in natural language on paper

Formal, mathematical, unambiguous specification of timing constraints in Logic: specify patterns that timed behaviors of systems should (not) satisfy
LTL, MTL, STL, …

STL (Signal Temporal Logic): Predicates over real value, real-time

Example: Between 2s and 6s the signal is between -2 and 2

$$\psi := G_{[2,6]}(|x[t]| < 2)$$



Why a new logic?

- Difficult to express sequential timing constraints on events – they become nested constraints

  Example: Whenever x_1 rises above 0.5, x_2 should rise above 0.6 within 1 second and after that, x_3 should fall below 0.4 within 5 seconds

$$\psi = \square((x_1 > 0.5) \wedge (\neg(x_1 > 0.5) S \top)) \vee (\neg(x_1 > 0.5) \wedge ((x_1 > 0.5) U \top))) \Rightarrow (\Diamond_{[0,1]}((x_2 > 0.6) \wedge (\neg(x_2 > 0.6) S \top)) \vee (\neg(x_2 > 0.6) \wedge ((x_2 > 0.6) U \top)) \Rightarrow (\Diamond_{[0,5]}((\neg(x_3 > 0.4) \wedge ((x_3 > 0.4) S \top))) \vee (((x_3 > 0.4) \wedge (\neg(x_3 > 0.4) U \top))))))$$

- We want domain-specific support for CPS time constraints

$$\psi = \square((\uparrow(x_1 > 0.5)) \Rightarrow (\Diamond_{[0,1]}(\uparrow(x_2 > 0.6)) \Rightarrow (\Diamond_{[0,5]}(\downarrow(x_3 > 0.4)))))$$

$\square$: Globally, $\Diamond$: Eventually, $\uparrow$: Rise operator, $\downarrow$: fall operator

$$\uparrow\psi = (\psi \wedge (\neg\psi S \top)) \vee (\neg\psi \wedge (\psi U \top))$$

$$\downarrow\psi = (\neg\psi \wedge (\psi S \top)) \vee (\psi \wedge (\neg\psi U \top))$$

Timestamp Temporal Logic

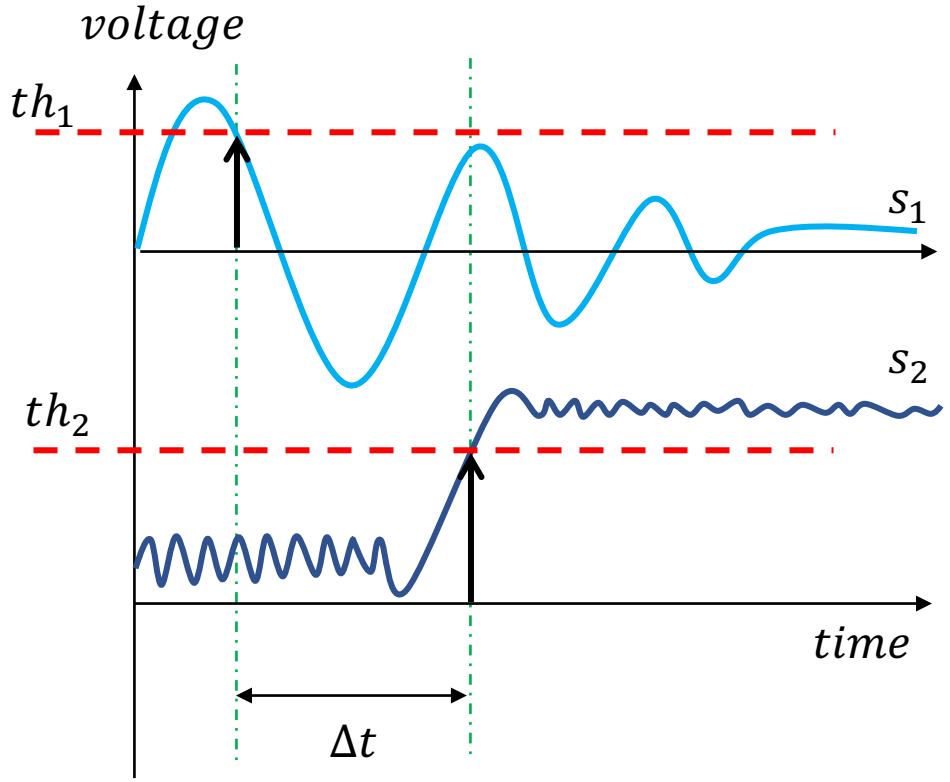
- *Latency*
  - $\mathcal{L}\ (\langle s_1, th_1, \searrow\rangle, \langle s_2, th_2, \nearrow\rangle, \varepsilon)\ \{<, >, =\}$ c
    - $\mathcal{L}\ (\langle s_1, th_1, \searrow\rangle, \langle s_2, th_2, \nearrow\rangle, \varepsilon) = \Delta t$
    - $\Delta t < c + \varepsilon; \Delta t > c - \varepsilon;$
    - $c - \varepsilon < \Delta t < c + \varepsilon$
- *Simultaneity*
  - $\mathcal{S}(\langle s_1, th_1, \nearrow\rangle, \langle s_2, th_2, \searrow\rangle, \langle s_3, th_3, \nearrow\rangle, \varepsilon)$
    - $max\{t_1, t_2, t_3\} - \min\{t_1, t_2, t_3\} < \varepsilon$
    - t is the timestamp of the event
- *Chronological*
  - $\mathcal{C}(\langle s_1, th_1, \nearrow\rangle, \langle s_2, th_2, \searrow\rangle, \langle s_3, th_3, \nearrow\rangle, \ldots, \varepsilon)$
    - *ε is the minimum latency between the events.*
- *Frequency*
  - $\mathcal{F}(\langle s_1, th_1, \nearrow\rangle, \varepsilon_1, \varepsilon_2)\ \{<, >, =\}\ c$
  - c = $\frac{1}{T_1 \pm \varepsilon_1}$ *(T_1 is the period of threshold crossing)*
    - $\mathcal{F}(\langle s_1, th_1, \searrow\rangle, \varepsilon_1) = \Delta$f
    - Δf $< c + \varepsilon_2; \Delta f > c - \varepsilon_2;$
    - $c - \varepsilon_2 < \Delta f < c + \varepsilon_2$
- *Phase*
  - $\mathcal{P}(\langle s_1, th_1, \nearrow\rangle, \langle s_2, th_2, \searrow\rangle, \varepsilon_1, \varepsilon_2)\ \ \{<, >, =\}$ c
  - $\frac{1}{T_1 \pm \varepsilon_1}$ *(T_1 is the period of threshold crossing of s_1)*
  - $\frac{1}{T_2 \pm \varepsilon_1}$ *(T_2 is the period of threshold crossing of s_2)*
    - $\mathcal{P}\ (\langle s_1, th_1, \searrow\rangle, \langle s_2, th_2, \nearrow\rangle, \varepsilon_1, \varepsilon_2) = \Delta$t
    - Δt $< c + \varepsilon_2; \Delta t > c - \varepsilon_2; c - \varepsilon_2 < \Delta t < c + \varepsilon_2$
- *Burst*
  - $\mathcal{B}(\langle s_1, th_1, \nearrow\rangle, N, d_k, m, \varepsilon)$ (N events in d_k duration then m time unit in silence)
    - $t_{N+1} > t_N + m + \varepsilon$

Latency Constraint

Latency constraint captures the time difference between the occurrence of two events ($\langle s_1, th_1, \searrow \rangle$ *and* $\langle s_2, th_2, \nearrow \rangle$).

$$\mathcal{L}(\langle s_1, th_1, \searrow \rangle, \langle s_2, th_2, \nearrow \rangle) \left\{ \begin{array}{c} < \\ > \\ = \end{array} \right\} \; c \pm \epsilon$$

s: *analog signal*
th: *threshold*
$\nearrow$: crossing from below
$\searrow$: *crossing from above*



Timestamp Temporal Logic

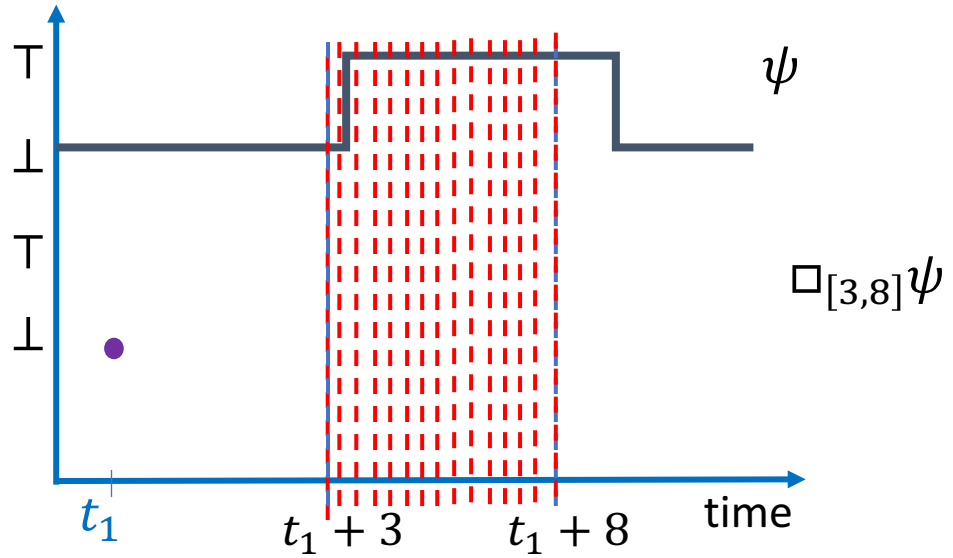
- *Latency*
  - $\mathcal{L}\ (\langle s_1, th_1, \searrow\rangle, \langle s_2, th_2, \nearrow\rangle, \varepsilon)\ \{<, >, =\}$ c
    - $\mathcal{L}\ (\langle s_1, th_1, \searrow\rangle, \langle s_2, th_2, \nearrow\rangle, \varepsilon) = \Delta t$
    - $\Delta t < c + \varepsilon; \Delta t > c - \varepsilon;$
    - $c - \varepsilon < \Delta t < c + \varepsilon$
- *Simultaneity*
  - $\mathcal{S}(\langle s_1, th_1, \nearrow\rangle, \langle s_2, th_2, \searrow\rangle, \langle s_3, th_3, \nearrow\rangle, \varepsilon)$
    - $max\{t_1, t_2, t_3\} - \min\{t_1, t_2, t_3\} < \varepsilon$
    - t is the timestamp of the event
- *Chronological*
  - $\mathcal{C}(\langle s_1, th_1, \nearrow\rangle, \langle s_2, th_2, \searrow\rangle, \langle s_3, th_3, \nearrow\rangle, \ldots, \varepsilon)$
    - *ε is the minimum latency between the events.*
- *Frequency*
  - $\mathcal{F}(\langle s_1, th_1, \nearrow\rangle, \varepsilon_1, \varepsilon_2)\ \{<, >, =\}\ c$
  - c = $\frac{1}{T_1 \pm \varepsilon_1}$ *(T_1 is the period of threshold crossing)*
    - $\mathcal{F}(\langle s_1, th_1, \searrow\rangle, \varepsilon_1) = \Delta$f
    - Δf $< c + \varepsilon_2; \Delta f > c - \varepsilon_2;$
    - $c - \varepsilon_2 < \Delta f < c + \varepsilon_2$
- *Phase*
  - $\mathcal{P}(\langle s_1, th_1, \nearrow\rangle, \langle s_2, th_2, \searrow\rangle, \varepsilon_1, \varepsilon_2)\ \ \{<, >, =\}$ c
  - $\frac{1}{T_1 \pm \varepsilon_1}$ *(T_1 is the period of threshold crossing of s_1)*
  - $\frac{1}{T_2 \pm \varepsilon_1}$ *(T_2 is the period of threshold crossing of s_2)*
    - $\mathcal{P}\ (\langle s_1, th_1, \searrow\rangle, \langle s_2, th_2, \nearrow\rangle, \varepsilon_1, \varepsilon_2) = \Delta$t
    - Δt $< c + \varepsilon_2; \Delta t > c - \varepsilon_2; c - \varepsilon_2 < \Delta t < c + \varepsilon_2$
- *Burst*
  - $\mathcal{B}(\langle s_1, th_1, \nearrow\rangle, N, d_k, m, \varepsilon)$ (N events in d_k duration then m time unit in silence)
    - $t_{N+1} > t_N + m + \varepsilon$

Overview of this Talk

- Importance of Timing in CPS
  - CODES+ISSS 2016
- How do we specify the timing constraints of a distributed CPS?
  - EMSOFT-TECS 2017
- How can we monitor if a CPS is meeting its timing constraints?
  - DAC 2018
- A testbed for monitoring timing constraints
  - Reconfig 2015
- CPS Testbeds
  - Flying paster, synchronized cameras, phase synchronization over the internet, intersection for autonomous cars
- Timing Health Monitoring and Management System

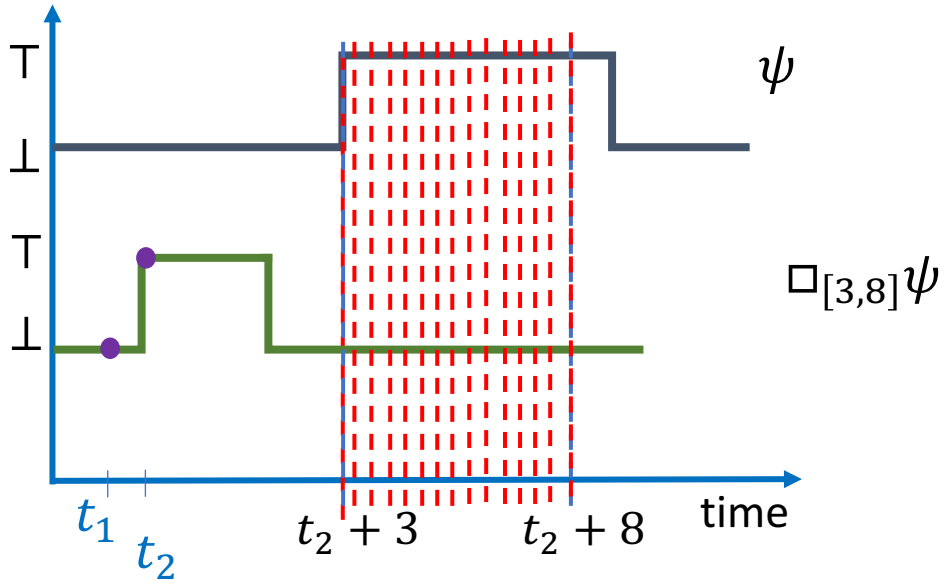
Monitoring Timing Constraints

- Consider a global timing constraint
  - $\square_{[3,8]}(s > 2V)$
  - Between 3 to 8 time unit after now, signal s should be greater than $2V$
- Traditional methods
  - Look at all time-steps in the interval to evaluate for every time-step.



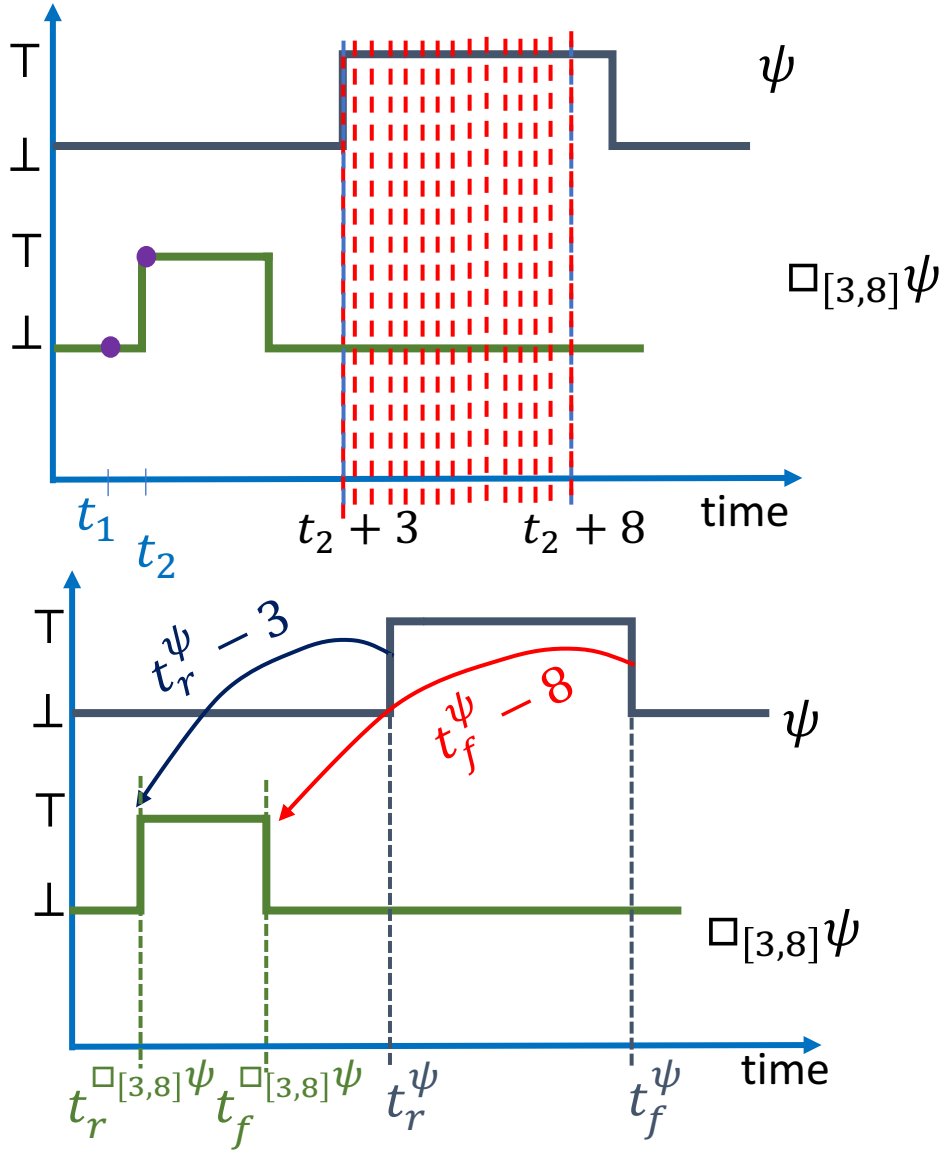
Monitoring Timing Constraints

- Globally
  - $\square_{[3,8]}(s > 2V)$
  - Between 3 to 8 time unit after now, signal s should be always greater than $2V$
- Traditional methods
  - Look at all time-step in the interval to evaluate for every time-step.



Timestamp-based Monitoring Approach

- **Traditional**
- No. of Operations:
  - $\big((8-3)\times f_s\big)\times T\times f_s$
  - T: test duration
  - f_s: sampling frequency
- Memory:
  - It needs the entire interval which is in future
    - ✓ $(8-3)\times f_s$
    - ✓ Per constraint
- **TMA – Our Approach**
- No. of operations:
  - 2 timestamps
  - Per the most recent event timestamps
- Memory:
  - It needs the most recent event timestamps
    - ✓ 2 timestamps
    - ✓ Per falling and rising edge



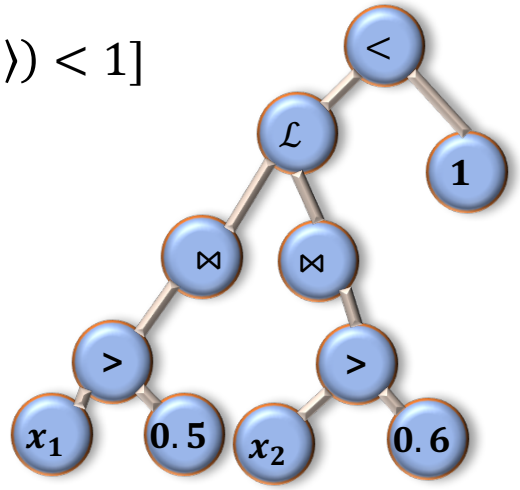
TMA – Online tool for monitoring timing constraints

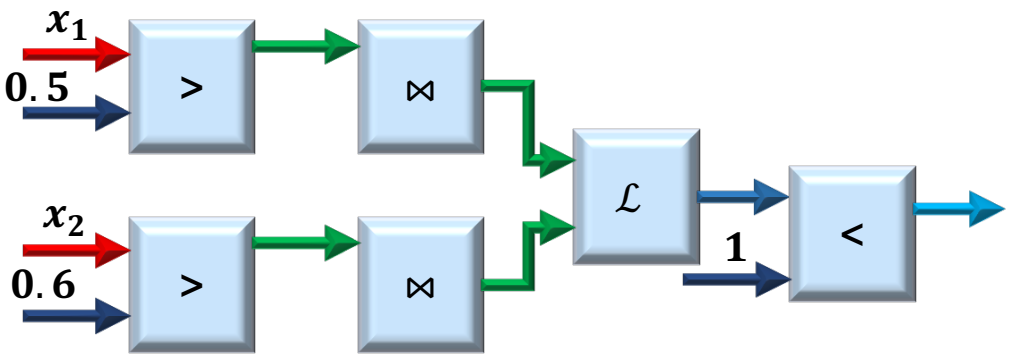
- TMA – Enables online Timing Monitoring
- Download and try http://aviral.lab.asu.edu/cps-software-download/
- Matlab based tool

- Inputs:
  - A trace of timestamped signals, timestamp-value pairs for signals
  - Timing constraint in TTL (and STL)
- Outputs:
  - When the timing constraints are met

- Coming soon: A FPGA-based tool for online monitoring of timing constraints of a real CPS

- Whenever signal x_1 rises above 0.5, signal x_2 should rise above 0.6 within 1 second:
- $\psi = [L(\langle x_1, 0.5, \nearrow\rangle, \langle x_2, 0.6, \nearrow\rangle) < 1]$



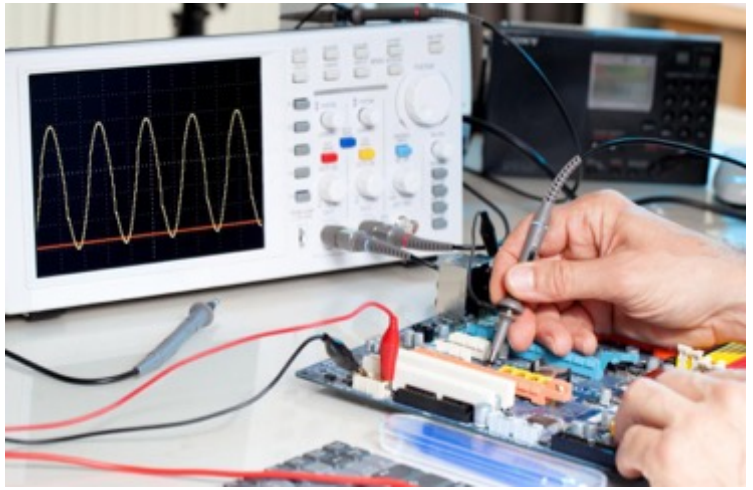


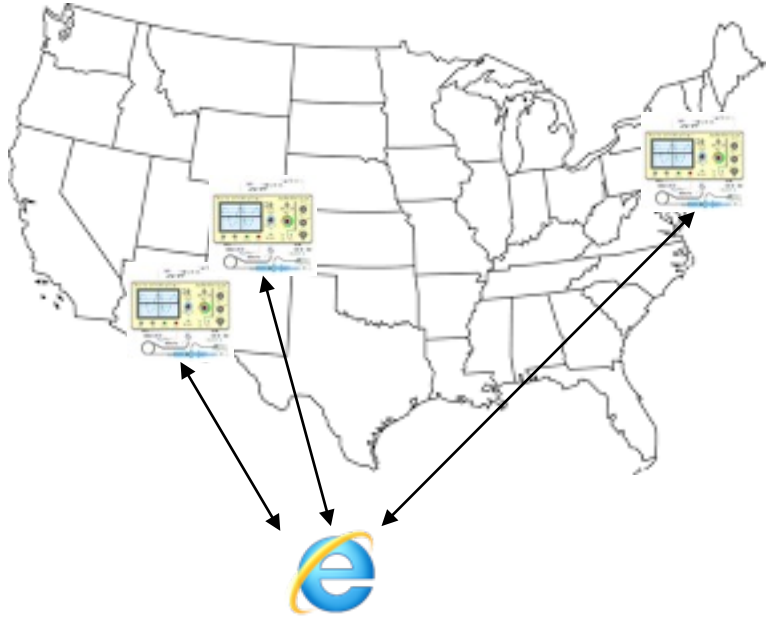
Overview of this Talk

- Importance of Timing in CPS
  - CODES+ISSS 2016
- How do we specify the timing constraints of a distributed CPS?
  - EMSOFT-TECS 2017
- How can we monitor if a CPS is meeting its timing constraints?
  - DAC 2018
- A testbed for monitoring timing constraints
  - Reconfig 2015, DAC 2017
- CPS Testbeds
  - Flying paster, synchronized cameras, phase synchronization over the internet, intersection for autonomous cars
- Timing Health Monitoring and Management System

Timing Monitoring

- Single systems monitoring
  - Monitor on oscilloscope or Data Acquisition (DAQ)
  - ADC resolution (e.g. 14-bit) and range (0-5V)
  - Sampling frequency (e.g. 100kHz)
  - Input impedance (e.g. 1 MΩ)
- Additionally in Distributed Monitoring [4]
  - Having the same notation of time (requires synchronization)
  - Synchronization accuracy (e.g. 100us)
  - Clock error (e.g. 5 ppm)





What can be monitored?

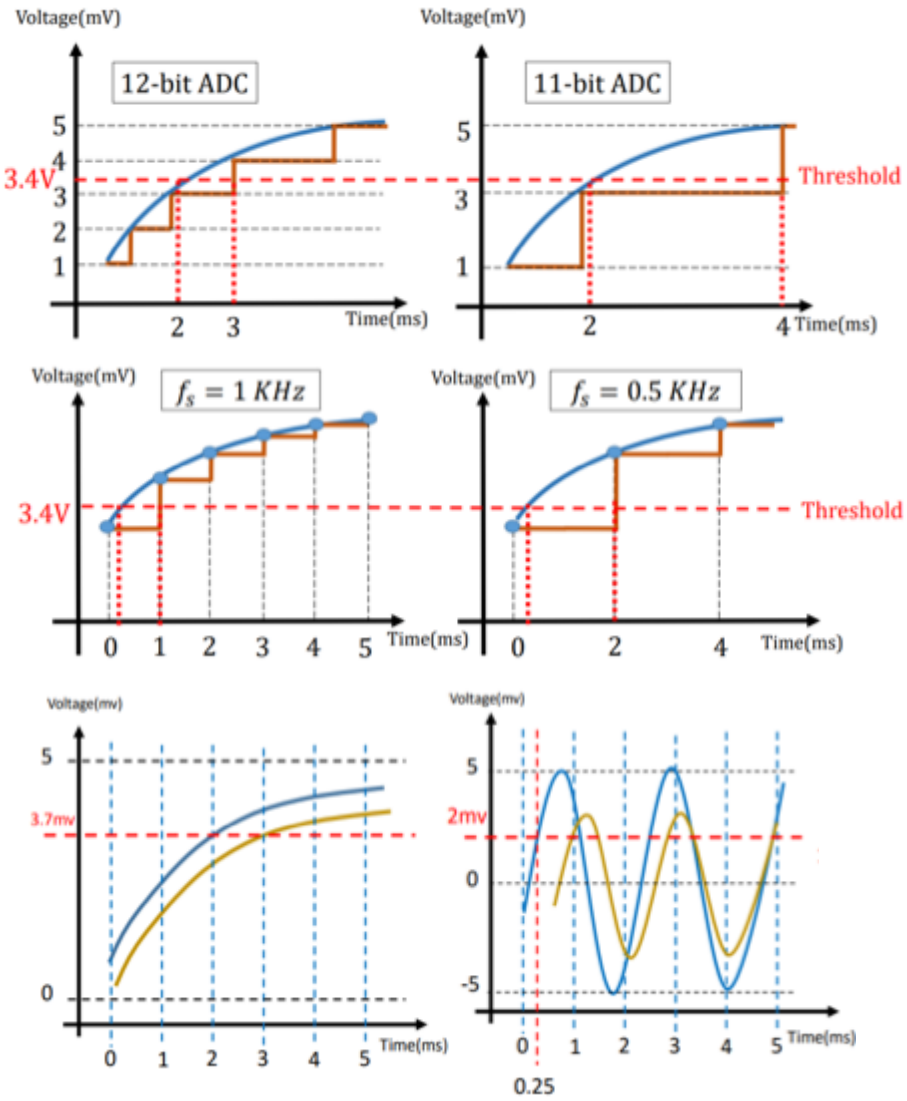
- Monitoring the exact occurrence time of an event needs special instruments.
- Parameters and especially epsilon of the timing constraints put limitations on the resolution and sampling rate of ADC converters.

  Quantization error < ε
- Any physical connection between SUT and monitoring testbed can change the shape of signals.

  Loading effect < ε
- Clocks are not perfect and even after synchronization, there is an error.

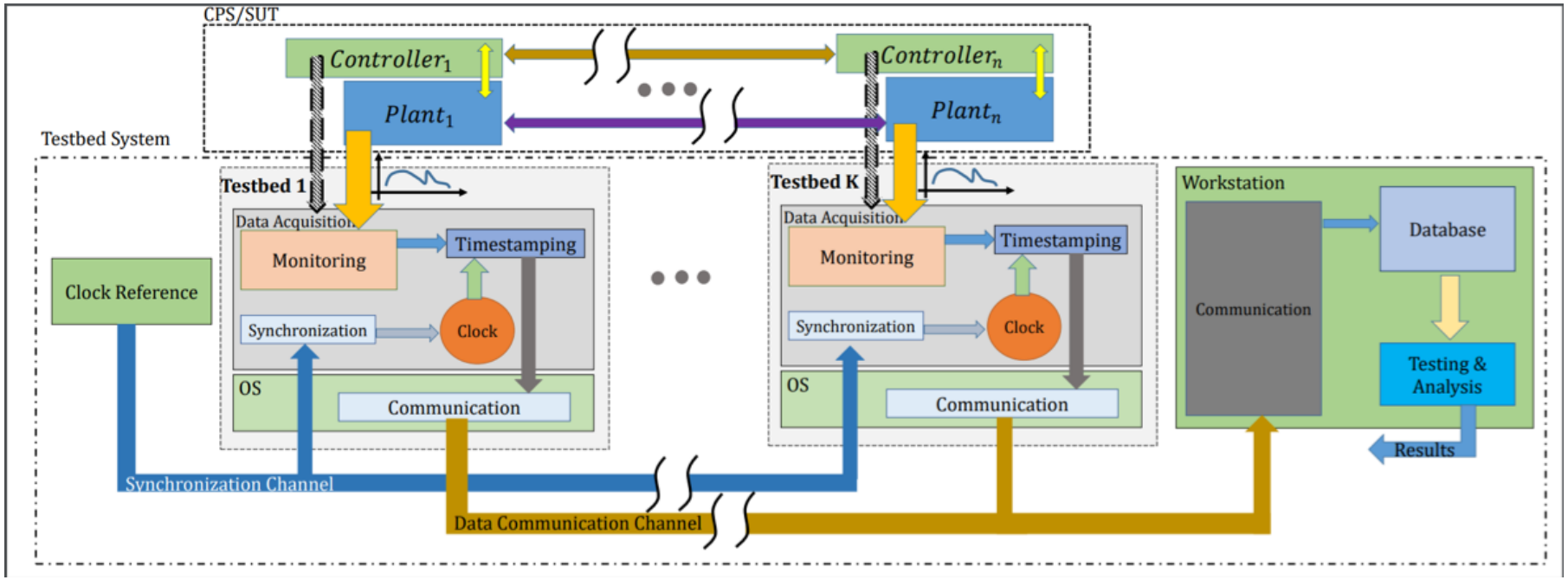
  Clock drift < ε

Quantization error + loading error + clock drift < **ε**



CPS Timing Testbed

- Distributed CPS, Distributed testbed
- Synchronize time among the distributed tested components
- Testbed captures timestamped signal at each node
- Runtime or offsite evaluation of timing constraints
- Can provide timing constraints remotely, testbed will adjust the sampling rate, ADC resolution, synchronization etc.



[4] Shrivastava, Aviral, et al. "A testbed to verify the timing behavior of cyber-physical systems." Design Automation Conference (DAC), 2017 54th ACM/EDAC/IEEE. IEEE, 2017.

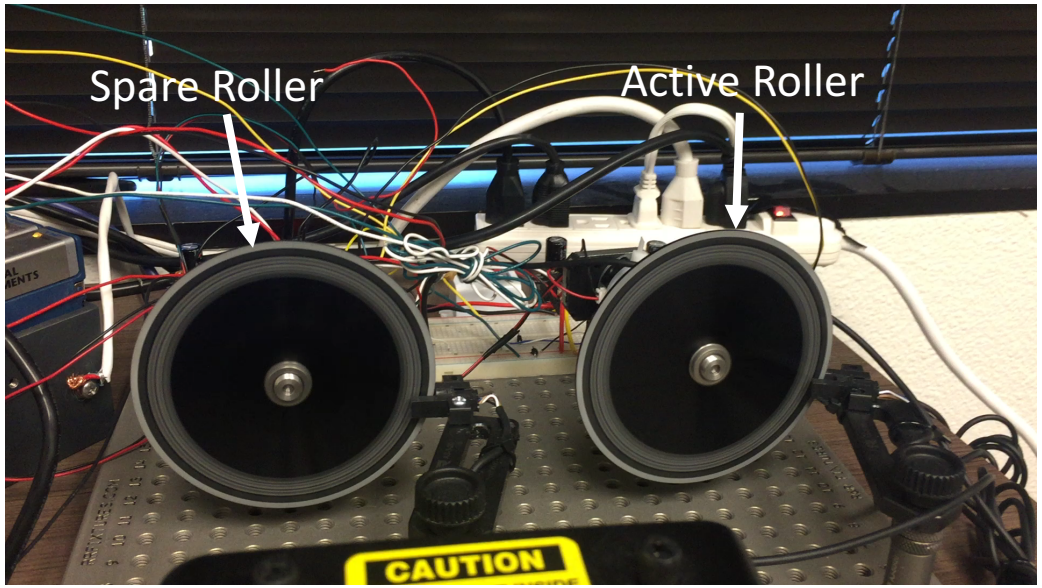
Overview of this Talk

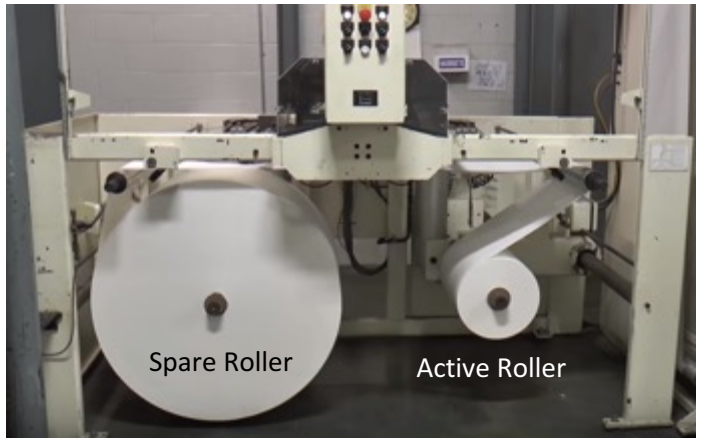
- Importance of Timing in CPS
  - CODES+ISSS 2016
- How do we specify the timing constraints of a distributed CPS?
  - EMSOFT-TECS 2017
- How can we monitor if a CPS is meeting its timing constraints?
  - DAC 2018
- A testbed for monitoring timing constraints
  - Reconfig 2015, DAC 2017
- CPS Testbeds
  - Flying paster, synchronized cameras, phase synchronization over the internet, intersection for autonomous cars
- Timing Health Monitoring and Management System

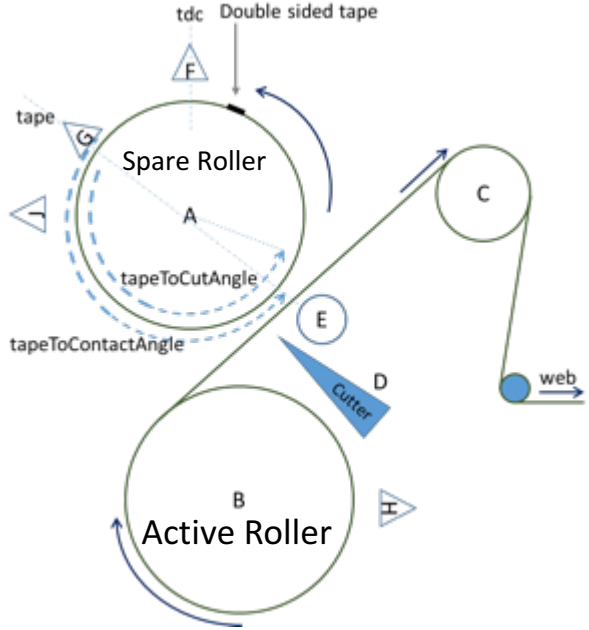
Flying Paster

A Flying Paster is a splicer for a web press that is used for continuous production. It works by "pasting" a spare roll onto the active roll so that the press does not have to stop.

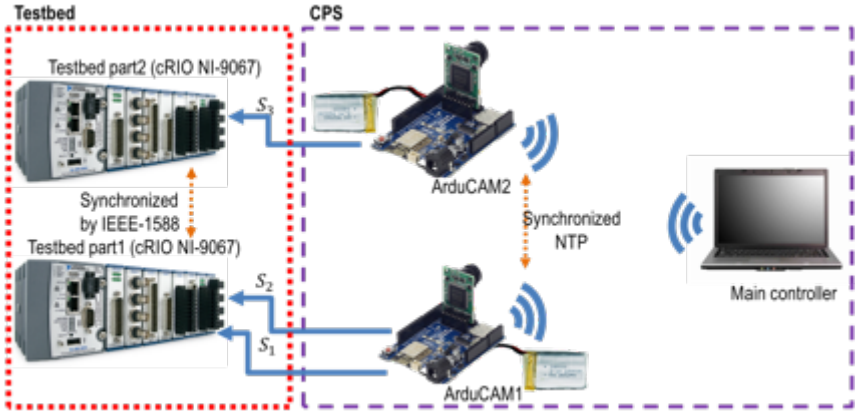
- Active roller rotates in a constant speed.
- Spare roller start to rotate as the Active roller speed after AOP signal.
- Match signal rises when two speeds are the same.
- $225°$ after Match, strobe flashes for contact command, $290°$ after Match, strobe flashes for cut command.
- We monitored it by expressing its timing constraints in TTL and implementing the automated time-testing methodology







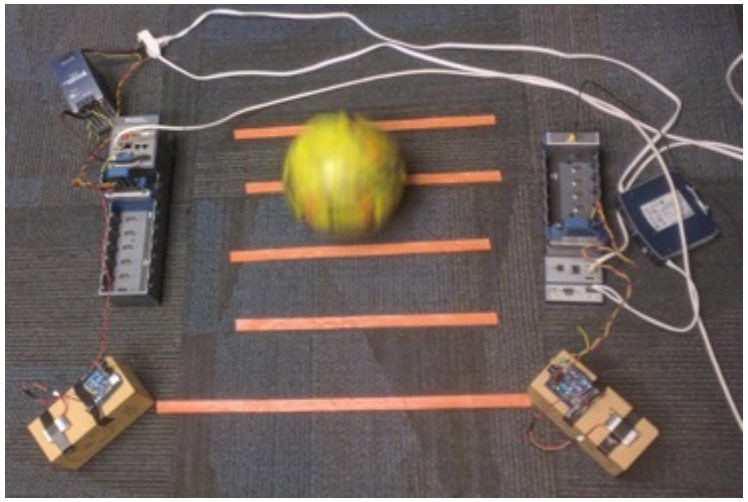
Synchronized Cameras



3D image reconstruction based on multiple 2D images taken from different angles of a scene.

- The capturing for two cameras should be at the same time with $0.01\ s$ as the tolerance.
- The latency between issuing the command and actual capturing should be less than $0.2\ s$.

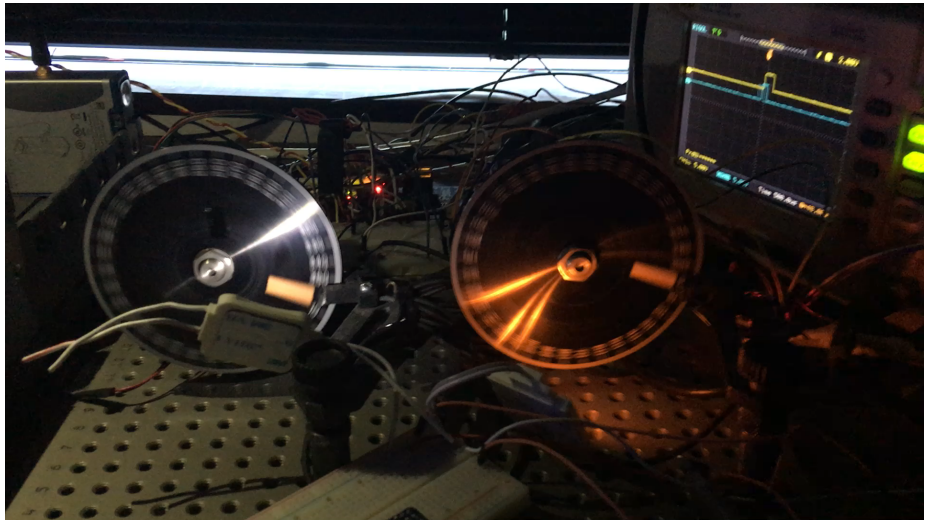
$$S(\langle s_2, 2.5, \nearrow\rangle, \langle s_3, 2.5, \nearrow\rangle, 0.01) \wedge L(\langle s_1, 2.5, \nearrow\rangle, \langle s_2, 2.5, \nearrow\rangle) < 0.2)$$

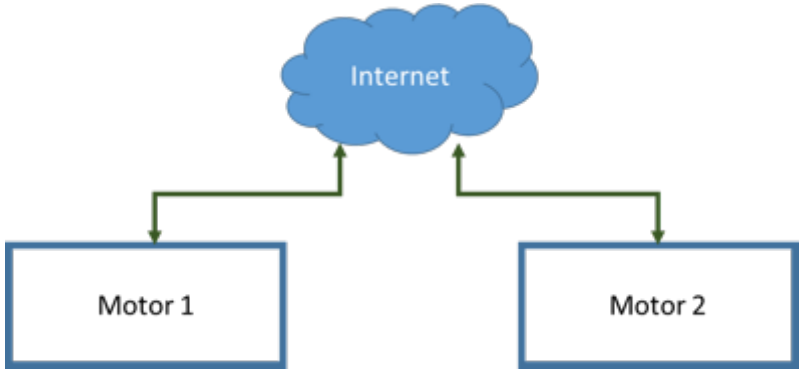


- cRIO $f_{clock} = 5ppm$
  - $t_{sync} = 100ns$ and $r_{sync} = 1s$
- Sampling rate $f_s = 20KHz$
- The maximum error $\epsilon_{ADC} = \frac{1}{f_s}$
- Therefore, the total error:
  - $\epsilon_{total} = \frac{1}{20KHz} + \frac{5\mu s}{1s} + 100ns \approx 55\mu s$
  - $55\mu s << 0.2$
  - The monitoring device is qualified to monitor this application

Synchronize motor phases over the internet

- The generated power in the distributed power generation systems resources should be matched in order to avoid short circuit.
  - In power grid system, a pair of generators connected to the same grid should generate a sinusoidal signal with frequency of 60 Hz.
  - The phase between two generator shouldn’t be greater than 10 degree.
    - We implemented an emulation of Power Grid system by two motors that were controlled by separated controller.
    - One of the motors is connected to cRIO-9067 and the other is connected to cRio_9035
    - The controllers are connected through a real network traffic of Internet.
    - They could be synchronized within 1.5 ms.



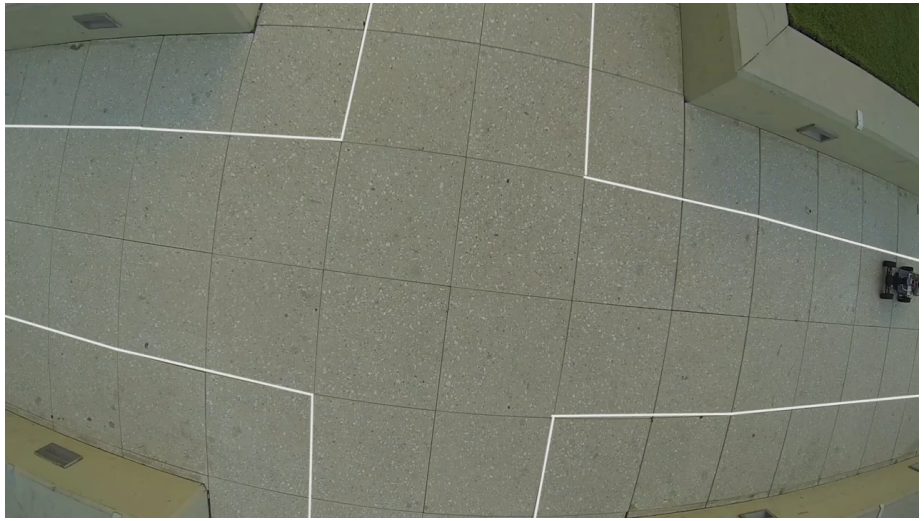


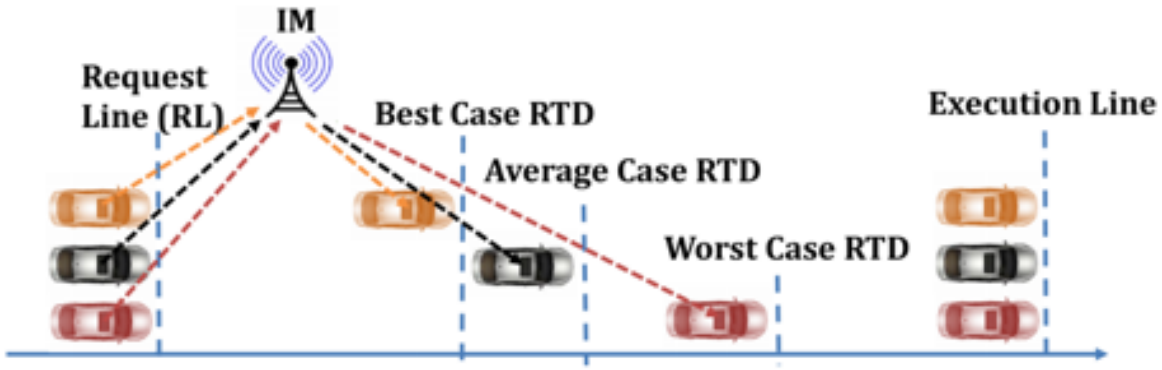
Autonomous vehicle Intersection management

- A testbed with Safety-Critical timing constraints
- Nondeterminism
  - Sensor error
  - Network delay
  - Computation delay

1/10 scale mode intersection of RC cars.

- Crossroads [5] – A time-sensitive autonomous intersection management technique





Nondeterministic Round Trip Delay (RTD) when a car sends a request to the Intersection Manager (IM)

[5] Andert, Edward, Mohammad Khayatian, and Aviral Shrivastava. "Crossroads: Time-Sensitive Autonomous Intersection Management Technique." 54th Annual Design Automation Conference. ACM, 2017.

Overview of this Talk

- Importance of Timing in CPS
  - CODES+ISSS 2016
- How do we specify the timing constraints of a distributed CPS?
  - EMSOFT-TECS 2017
- How can we monitor if a CPS is meeting its timing constraints?
  - DAC 2018
- A testbed for monitoring timing constraints
  - Reconfig 2015, DAC 2017
- CPS Testbeds
  - Flying paster, synchronized cameras, phase synchronization over the internet, intersection for autonomous cars
- Timing Health Monitoring and Management System

Timing Health Monitoring and Reasoning System

- Reasoning in TTL
  - To deduce the complex timing constraints from the simpler ones and reason about the complicated timing constraints feasibility.
- Automatic timing health monitoring
  - Utilizing TTL reasoning for performance degrading when the timing constraints are not met instead of terminating CPS operation.
- Timing Correct-by-construction
  - Utilizing TTL to defining the CPS temporal behavior in the design phase.

Summary

- Importance of Timing in CPS
  - CODES+ISSS 2016
- How do we specify the timing constraints of a distributed CPS?
  - EMSOFT-TECS 2017
- How can we monitor if a CPS is meeting its timing constraints?
  - DAC 2018
- A testbed for monitoring timing constraints
  - Reconfig 2015, DAC 2017
- CPS Testbeds
  - Flying paster, synchronized cameras, phase synchronization over the internet, intersection for autonomous cars
- Timing Health Monitoring and Management System

Thank You

- Questions?

Backup Material

How to specify timing constraints of CPS?

Instead of specifying timing constraints in natural language on paper

Formal, mathematical, unambiguous specification of timing constraints in Logic: specify patterns that timed behaviors of systems should (not) satisfy

- LTL (Linear Temporal Logic)
  - Deals with discrete sequence of states
  - Based on logic operators (¬, $\wedge$, $\vee$), and temporal operators, next (N), always (G), eventually (F), and until (U).
  - Boolean predicates, discrete time
  - Example constraint: G (r => F g)
  - Sense of time is "states"
  - *Not sufficient for CPS – CPS operate in continuous time.*

- MTL (Metric Temporal Logic)
  - Boolean predicates, real-time
  - Example constraint: G (r => $F_{[0,5]}$ g)
  - *Not sufficient for CPS – CPS may have real-valued logic*

STL – Signal Temporal Logic

Predicates over real value, real-time

Assume signals $x_1[t]$, $x_2[t]$, ..., $x_n[t]$, then atomic predicates are of the form:

$\mu = f(x_1[t],\ldots,x_n[t]) > 0$

Example: Between 2s and 6s the signal is between -2 and 2

$$\psi := G_{[2,6]}(|x[t]| < 2)$$

Express timing constraints in STL

- Intuitive to specify value-based (level-based) timing constraints

  Example: the value of x[t] is less than 2

  $\phi := G_{[2,6]}\ (|x[t]| < 2)$

- Hard to express event-based (edge-based) timing constraints

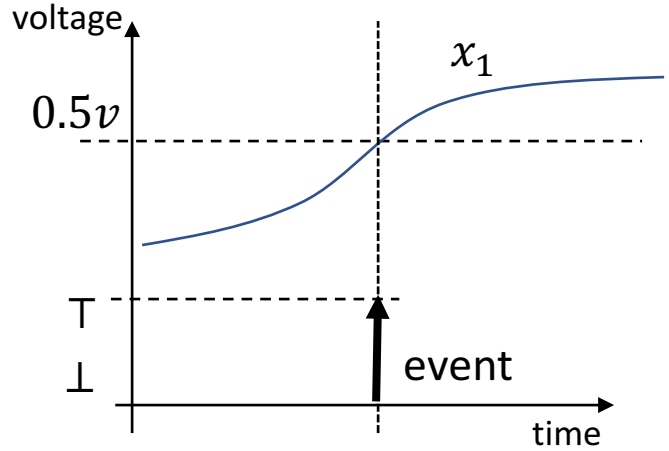
  Example: Whenever x_1 rises above 0.5, x_2 should rise above 0.6 within 1 sec.

  $\square\ (\uparrow(x_1>0.5) \Rightarrow (\Diamond_{[0,1]}(\uparrow(x_2>0.6))))$

  $\square$: Globally, $\Diamond$: Eventually, $\uparrow$: Rise operator

  Definition of the rise operator

  $\uparrow\psi = (\psi \wedge (\neg\psi S \top)) \vee (\neg\psi \wedge (\psi U \top))$



Express timing constraints in STL

- Harder to express sequential timing constraints on events – they become nested constraints

  Example: Whenever x_1 rises above 0.5, x_2 should rise above 0.6 within 1 second and after that, x_3 should fall below 0.4 within 5 seconds

  $\psi = \square((\uparrow(x_1>0.5)) \Rightarrow (\Diamond_{[0,1]}(\uparrow(x_2>0.6)) => (\Diamond_{[0,5]}(\downarrow(x_3>0.4)))))$

  $\square$: Globally, $\Diamond$: Eventually, $\uparrow$: Rise operator, $\downarrow$: fall operator

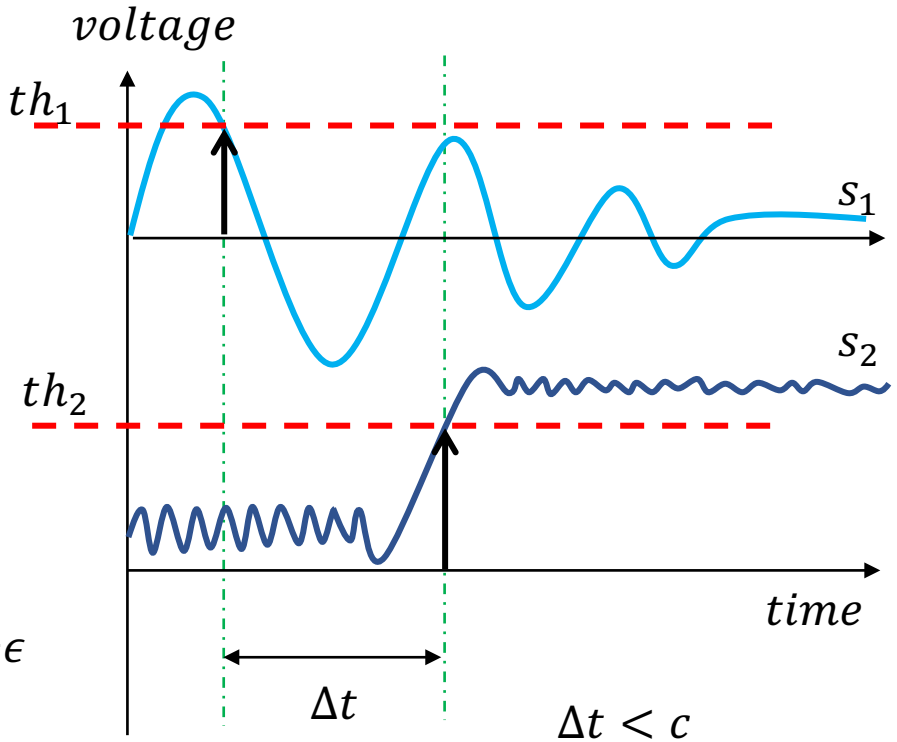
  $\uparrow\psi = (\psi \wedge(\neg\psi S^{\top})) \vee (\neg\psi \wedge(\psi U^{\top}))$

  $\downarrow\psi = (\neg\psi \wedge(\psi S^{\top})) \vee (\psi \wedge(\neg\psi U^{\top}))$

  $\psi = \square((x_1> 0.5) \wedge (\neg(x_1 > 0.5)S^{\top})) \vee (\neg(x_1 > 0.5) \wedge ((x_1 > 0.5)U^{\top}))) \Rightarrow$
  $(\Diamond_{[0,1]}((x_2 > 0.6) \wedge (\neg(x_2 > 0.6)S^{\top})) \vee (\neg(x_2 > 0.6) \wedge ((x_2 > 0.6)U^{\top})) \Rightarrow$
  $(\Diamond_{[0,5]}((\neg(x_3 > 0.4) \wedge ((x_3 > 0.4)S^{\top}))) \vee (((x_3 > 0.4) \wedge (\neg(x_3 > 0.4)U^{\top}))))))$

Minimum Latency Constraint

A Minimum Latency constraint expresses that the latency between the occurrence of two events ($\langle s_1, th_1, \searrow \rangle$ *and* $\langle s_2, th_2, \nearrow \rangle$)is less than some value (c).

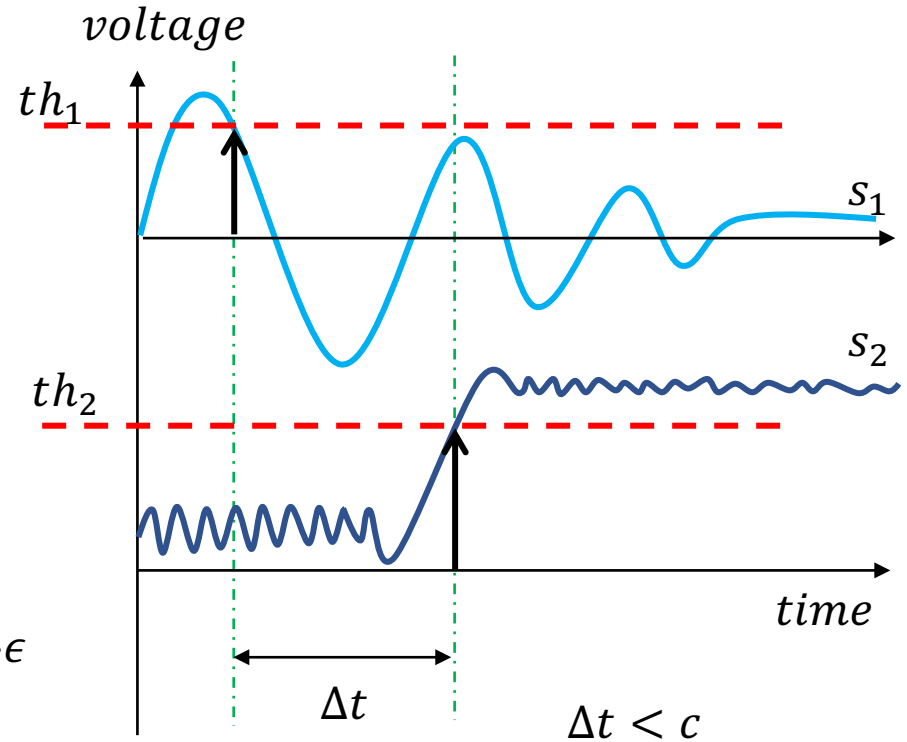


$\mathcal{L}(\langle s_1, th_1, \searrow \rangle, \langle s_2, th_2, \nearrow \rangle) < c+\epsilon$

s: *analog signal*
th: *threshold*
$\nearrow$: crossing from below
$\searrow$: *crossing from above*

Maximum Latency Constraint

A Maximum Latency constraint expresses that the latency between the occurrence of two events ($\langle s_1, th_1, \searrow \rangle$ *and* $\langle s_2, th_2, \nearrow \rangle$)is greater than some value (c).

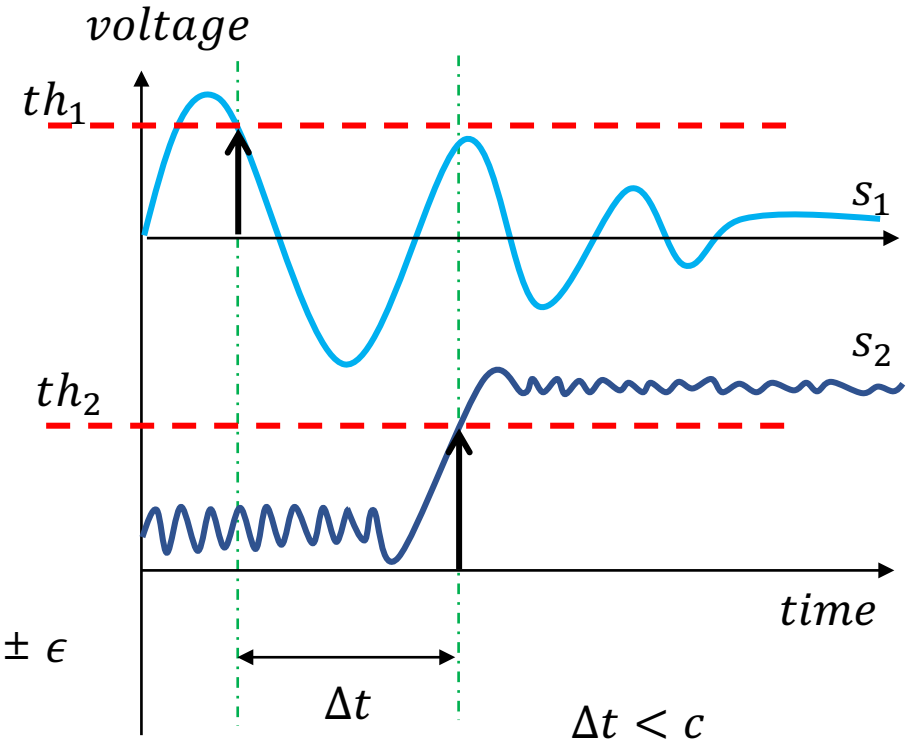


$\mathcal{L}(\langle s_1, th_1, \searrow \rangle, \langle s_2, th_2, \nearrow \rangle)$ < c-ϵ

s: *analog signal*
th: *threshold*
$\nearrow$: crossing from below
$\searrow$: *crossing from above*

Exact Latency Constraint

An Exact Latency constraint expresses that the latency between the occurrence of two events ($\langle s_1, th_1, \searrow \rangle$ *and* $\langle s_2, th_2, \nearrow \rangle$)is equal to some value (c) with some tolerance (ε).



$\mathcal{L}(\langle s_1, th_1, \searrow \rangle, \langle s_2, th_2, \nearrow \rangle) < c \pm \epsilon$

s: *analog signal*
th: *threshold*
↗: crossing from below
↘: *crossing from above*