CPS: Exciting times

- Sensors $\rightarrow$ sensor networks
- Embedded systems $\rightarrow$ networked embedded systems
- Control has come to software
  - *Mechanical* $\rightarrow$ *Electrical* $\rightarrow$ *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

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

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

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

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

\[
\psi = [\Box((x_1 > 0.5) \land (\neg (x_1 > 0.5))S\top)) \lor (\neg (x_1 > 0.5) \land ((x_1 > 0.5)U\top))) \Rightarrow
(\Diamond_{[0,1]}((x_2 > 0.6) \land (\neg (x_2 > 0.6))S\top)) \lor (\neg (x_2 > 0.6) \land ((x_2 > 0.6)U\top))) \Rightarrow
(\Diamond_{[0,5]}((\neg (x_3 > 0.4) \land ((x_3 > 0.4))S\top)) \lor (((x_3 > 0.4) \land (\neg (x_3 > 0.4))))])
\]

• We want domain-specific support for CPS time constraints

\[
\psi = \Box((\uparrow (x_1 > 0.5)) \Rightarrow (\Diamond_{[0,1]}(\uparrow (x_2 > 0.6)))
\]

\( \Box \): Globally, \( \Diamond \): Eventually, \( \uparrow \): Rise operator, \( \downarrow \): fall operator

\( \uparrow \psi = (\psi \land (\neg \psi S\top)) \lor (\neg \psi \land (\psi U\top)) \)

\( \downarrow \psi = (\neg \psi \land (\psi S\top)) \lor (\psi \land (\neg \psi U\top)) \)
Timestamp Temporal Logic

• Latency
  • $\mathcal{L} (\langle s_1, th_1, \uparrow \rangle, \langle s_2, th_2, \uparrow \rangle, \epsilon) \{<, >, =\} c$
  • $\mathcal{L} (\langle s_1, th_1, \downarrow \rangle, \langle s_2, th_2, \uparrow \rangle, \epsilon) = \Delta t$
  • $\Delta t < c + \epsilon; \Delta t > c - \epsilon$
  • $c - \epsilon < \Delta t < c + \epsilon$

• Simultaneity
  • $\mathcal{S} (\langle s_1, th_1, \uparrow \rangle, \langle s_2, th_2, \downarrow \rangle, \langle s_3, th_3, \uparrow \rangle, \epsilon)$
    • $\max\{t_1, t_2, t_3\} - \min\{t_1, t_2, t_3\} < \epsilon$
    • $t$ is the timestamp of the event

• Chronological
  • $\mathcal{C} (\langle s_1, th_1, \uparrow \rangle, \langle s_2, th_2, \downarrow \rangle, \langle s_3, th_3, \uparrow \rangle, ..., \epsilon)$
    • $\epsilon$ is the minimum latency between the events.

• Frequency
  • $\mathcal{F} (\langle s_1, th_1, \uparrow \rangle, \langle s_2, th_2, \downarrow \rangle, \epsilon_1, \epsilon_2) \{<, >, =\} c$
    • $c = \frac{1}{T_1 \pm \epsilon_1}$ ($T_1$ is the period of threshold crossing)
    • $\Delta f < c + \epsilon_2; \Delta f > c - \epsilon_2$
    • $c - \epsilon_2 < \Delta f < c + \epsilon_2$

• Phase
  • $\mathcal{P} (\langle s_1, th_1, \uparrow \rangle, \langle s_2, th_2, \downarrow \rangle, \epsilon_1, \epsilon_2) \{<, >, =\} c$
    • $\frac{1}{T_1 \pm \epsilon_1}$ ($T_1$ is the period of threshold crossing of $s_1$)
    • $\frac{1}{T_2 \pm \epsilon_1}$ ($T_2$ is the period of threshold crossing of $s_2$)
    • $\mathcal{P} (\langle s_1, th_1, \downarrow \rangle, \langle s_2, th_2, \uparrow \rangle, \epsilon_1, \epsilon_2) = \Delta t$
    • $\Delta t < c + \epsilon_2; \Delta t > c - \epsilon_2; c - \epsilon_2 < \Delta t < c + \epsilon_2$

• Burst
  • $\mathcal{B} (\langle s_1, th_1, \uparrow \rangle, N, d_k, m, \epsilon) \{N \text{ events in } d_k \text{ duration then } m \text{ time unit in silence}\}$
    • $t_{N+1} > t_N + m + \epsilon$
Latency Constraint

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

\[\mathcal{L}(s_1, th_1, \searrow, s_2, th_2, \nearrow) = \begin{cases} < & c \pm \epsilon \\ > & \end{cases}\]

- \(s\): analog signal
- \(th\): threshold
- \(\searrow\): crossing from below
- \(\nearrow\): crossing from above
Timestamp Temporal Logic

• **Latency**
  - $\mathcal{L}(⟨s_1, t h_1, \exists⟩, ⟨s_2, t h_2, \forall⟩, \varepsilon) \{<, >, =\} c$
  - $\mathcal{L}(⟨s_1, t h_1, \exists⟩, ⟨s_2, t h_2, \forall⟩, \varepsilon) = \Delta t$
  - $\Delta t < c + \varepsilon; \Delta t > c - \varepsilon;$
  - $c - \varepsilon < \Delta t < c + \varepsilon$

• **Simultaneity**
  - $\mathcal{S}(⟨s_1, t h_1, \forall⟩, ⟨s_2, t h_2, \exists⟩, ⟨s_3, t h_3, \forall⟩, \varepsilon)$
  - $\max\{t_1, t_2, t_3\} - \min\{t_1, t_2, t_3\} < \varepsilon$
  - $t$ is the timestamp of the event

• **Chronological**
  - $\mathcal{C}(⟨s_1, t h_1, \forall⟩, ⟨s_2, t h_2, \exists⟩, ⟨s_3, t h_3, \forall⟩, ..., \varepsilon)$
  - $\varepsilon$ is the minimum latency between the events.

• **Frequency**
  - $\mathcal{F}(⟨s_1, t h_1, \forall⟩, \varepsilon_1, \varepsilon_2) \{<, >, =\} c$
  - $c = \frac{1}{T_1 \pm \varepsilon_1}$ ($T_1$ is the period of threshold crossing)
  - $\Delta f < c + \varepsilon_2; \Delta f > c - \varepsilon_2;$
  - $c - \varepsilon_2 < \Delta f < c + \varepsilon_2$

• **Phase**
  - $\mathcal{P}(⟨s_1, t h_1, \forall⟩, ⟨s_2, t h_2, \exists⟩, \varepsilon_1, \varepsilon_2) \{<, >, =\} c$
  - $\frac{1}{T_1 \pm \varepsilon_1}$ ($T_1$ 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 $s_2$)
  - $\mathcal{P}(⟨s_1, t h_1, \forall⟩, ⟨s_2, t h_2, \exists⟩, \varepsilon_1, \varepsilon_2) = \Delta t$
  - $\Delta t < c + \varepsilon_2; \Delta t > c - \varepsilon_2; c - \varepsilon_2 < \Delta t < c + \varepsilon_2$

• **Burst**
  - $\mathcal{B}(⟨s_1, t h_1, \forall⟩, N, d_k, m, \varepsilon)$ ($N$ 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
  • $\Box_{[3,8]}(s > 2V)$
  • 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
  • $\Box_{[3,8]}(s > 2V)$
  • 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:**
    - \( (8 - 3) \times f_s \times T \times f_s \)
    - \( T \): 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
- 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:

$\psi = [L((x_1, 0.5, \triangleright), (x_2, 0.6, \triangleright)) < 1]$
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.
  \[ \text{Quantization error} < \epsilon \]

- Any physical connection between SUT and monitoring testbed can change the shape of signals.
  \[ \text{Loading effect} < \epsilon \]

- Clocks are not perfect and even after synchronization, there is an error.
  \[ \text{Clock drift} < \epsilon \]

\[ \text{Quantization error} + \text{loading error} + \text{clock drift} < \epsilon \]
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.

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
  • Reconf 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 $0.01 \, \text{s}$ as the tolerance.
- The latency between issuing the command and actual capturing should be less than $0.2 \, \text{s}$.

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

- cRIO $f_{\text{clock}} = 5\, \text{ppm}$
  - $t_{\text{sync}} = 100\, \text{ns}$ and $r_{\text{sync}} = 1\, \text{s}$
- Sampling rate $f_s = 20\, \text{KHz}$
- The maximum error $\epsilon_{\text{ADC}} = \frac{1}{f_s}$
- Therefore, the total error:
  - $\epsilon_{\text{total}} = \frac{1}{20\, \text{KHz}} + \frac{5\mu\text{s}}{1\, \text{s}} + 100\, \text{ns} \approx 55\, \mu\text{s}$
  - $55\, \mu\text{s} \ll 0.2$
  - 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
• Crossroads [5] – A time-sensitive autonomous intersection management technique

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

- LTL (Linear Temporal Logic)
  - Deals with discrete sequence of states
  - Based on logic operators (¬, ∧, ∨), 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*

Formal, mathematical, unambiguous specification of timing constraints in Logic: specify patterns that timed behaviors of systems should (not) satisfy
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:
$\mu = f(x_1[t],...,x_n[t]) > 0$

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.
  $\Box (\uparrow(x_1 > 0.5) \Rightarrow (\Diamond_{[0,1]} (\uparrow(x_2 > 0.6))))$
  $\Box$: Globally, $\Diamond$: Eventually, $\uparrow$: Rise operator
  Definition of the rise operator
  $\uparrow\psi = (\psi \land (\neg\psi S^T)) \lor (\neg\psi \land (\psi U^T))$
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

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

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

$$\uparrow \psi = (\psi \land (\neg \psi S^T)) \lor (\neg \psi \land (\psi U^T))$$

$$\downarrow \psi = (\neg \psi \land (\psi S^T)) \lor (\psi \land (\neg \psi U^T))$$

$$\psi = \Box (((x > 0.5) \land \neg (x > 0.5) S^T) \lor (\neg (x > 0.5) \land ((x > 0.5) U^T)))) \implies$$
$$((\Diamond_{[0,1]}((x_2 > 0.6) \land (\neg (x_2 > 0.6) S^T)) \lor (\neg (x_2 > 0.6) \land ((x_2 > 0.6) U^T)))) \implies$$
$$((\Diamond_{[0,5]}((\neg (x_3 > 0.4) \land ((x_3 > 0.4) S^T)) \lor (((x_3 > 0.4) \land (\neg (x_3 > 0.4) U^T))))))$$
Minimum Latency Constraint

A Minimum Latency constraint expresses that the latency between the occurrence of two events \((s_1, th_1, \Uparrow)\ and \(s_2, th_2, \Downarrow)\) is less than some value (c).

\[
\mathcal{L}(s_1, th_1, \Uparrow), (s_2, th_2, \Downarrow) < c + \epsilon
\]

\(\mathcal{L}\): analog signal
\(th\): threshold
\(\Uparrow\): crossing from below
\(\Downarrow\): crossing from above
A Maximum Latency constraint expresses that the latency between the occurrence of two events \( (s_1, th_1, \forall) \) and \( (s_2, th_2, \exists) \) is greater than some value \( c \).

\[
\mathcal{L}((s_1, th_1, \forall), (s_2, th_2, \exists)) < c - \epsilon
\]
An Exact Latency constraint expresses that the latency between the occurrence of two events \( (s_1, th_1, \forall) \text{ and } (s_2, th_2, \forall) \) is equal to some value (c) with some tolerance (\( \varepsilon \)).

\[
\mathcal{L}((s_1, th_1, \forall), (s_2, th_2, \forall)) < c \pm \varepsilon
\]