Quartz
Cyber-physical Applications
Coordination is Key

Figure 1: The scale of coordination in time and space
Shared Notion of Time

Clock synchronization systems:

- GPS
- Network Time Protocol
- Precision Time Protocol

Agnostic to application-specific requirements

Clock synchronization is not perfect -> uncertainty
Time-as-a-Service (TaaS)

The ability to provide an application-specific clock tracking time reference, such that the timing uncertainty does not exceed application specified requirements

Quality of Time: end-to-end uncertainty bounds corresponding to a timestamp, with respect to a clock reference

1) Safety constraints
2) Performance requirements
3) Assumptions/tolerances of the controller
Quartz

Features fully user-space implementation that

1) Supports multi-tenancy
2) Operates at geo-distributed (WAN) scale
3) Portable to an array of application domains and platforms
4) Provides API for distributed coordination based on abstracting timeline
DronePorter

Figure 2: *DronePorter*: Drone coordination

(1) Time Sensitive Entrepreneurial Solution: High grading
Timeline Abstraction

Abstracts away clock synchronization from applications

A timeline provides a shared virtual clock reference to all distributed components of an application

Provides functionalities

1) Allows application to specify which components coordinate with each other
2) Provides visibility into where each application component is deployed, and what its QoT requirements are with respect to the timeline reference

To allow orchestration of clock-synchronization protocols which ensures QoT requirements are met
# Quartz API

<table>
<thead>
<tr>
<th>Category</th>
<th>API Call</th>
<th>Return Type</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeline Association</td>
<td><code>timeline_bind</code> (node_name, accuracy, resolution)</td>
<td>timeline</td>
<td>Bind to a timeline</td>
</tr>
<tr>
<td></td>
<td><code>timeline_unbind</code> (timeline)</td>
<td>status</td>
<td>Unbind from a timeline</td>
</tr>
<tr>
<td></td>
<td><code>timeline_setaccuracy</code> (timeline, accuracy)</td>
<td>status</td>
<td>Set Binding accuracy</td>
</tr>
<tr>
<td></td>
<td><code>timeline_setresolution</code> (timeline, resolution)</td>
<td>status</td>
<td>Set Binding resolution</td>
</tr>
<tr>
<td>Time Management</td>
<td><code>timeline_gettime</code> (timeline)</td>
<td>timestamp+QoT</td>
<td>Get timeline reference time with uncertainty</td>
</tr>
<tr>
<td></td>
<td><code>timeline_translate</code> (timestamp, src_timeline, dst_timeline)</td>
<td>timestamp+QoT</td>
<td>Translate a timestamp on one timeline into another</td>
</tr>
<tr>
<td>Event Scheduling &amp; Timestamping</td>
<td><code>timeline_waituntil</code> (timeline, absolute_time)</td>
<td>timestamp+QoT</td>
<td>Absolute blocking wait</td>
</tr>
<tr>
<td></td>
<td><code>timeline_sleep</code> (timeline, interval)</td>
<td>timestamp+QoT</td>
<td>Relative blocking wait</td>
</tr>
<tr>
<td></td>
<td><code>timeline_set_schedparams</code> (timeline, period, start_offset)</td>
<td>status</td>
<td>Set period and start offset</td>
</tr>
<tr>
<td></td>
<td><code>timeline_waituntil_nextperiod</code> (timeline)</td>
<td>timestamp+QoT</td>
<td>Absolute blocking wait until next period</td>
</tr>
<tr>
<td></td>
<td><code>timeline_timer_create</code> (timeline, period, start_offset, count, callback)</td>
<td>timer</td>
<td>Register a periodic callback</td>
</tr>
<tr>
<td></td>
<td><code>timeline_timestamp_events</code> (timeline, event_type, event_config, enable, callback)</td>
<td>status</td>
<td>Configure events/external timestamping on a pin</td>
</tr>
<tr>
<td>Latency</td>
<td><code>timeline_reqlatency</code> (timeline, src_node, dst_node, num_measure, percentile)</td>
<td>duration</td>
<td>Get the latency between two nodes on a timeline</td>
</tr>
</tbody>
</table>
Quartz Code Example

Listing 1: Simple Periodic App using the Quartz API

```python
def main_func(timeline_uuid: str, app_name: str):
    # Initialize the TimelineBinding class as an app
    binding = TimelineBinding(app)
    # Bind to the timeline with 1ms accuracy and resolution
    ret = binding.timeline_bind(timeline_uuid, app_name, 1ms, 1ms)
    if ret != ReturnTypes.QOT_RETURN_TYPE_OK:
        print('Unable to bind to timeline, terminating ....')
        exit(1)
    # Set the Scheduling Period and Offset (1s and 0ns respectively)
    binding.timeline_set_schedparams(1000000000, 0)
    while running:
        # Wait until the next period
        binding.timeline_waituntil_nextperiod()
        # Do Something -> Read the time with the uncertainty
        tl_time = binding.timeline_gettime()
        print('Timeline time is %f' % tl_time["time_estimate"])
        print('Upper Uncertainty is %f' % tl_time["interval_above"])
        print('Lower Uncertainty is %f' % tl_time["interval_below"])
        # Unbind from the timeline
        binding.timeline_unbind()
```
Architecture and Implementation

Challenges to overcome:

● Scalability (architecture)
● Autonomy (architecture)
● Portability (implementation)
● Ease of development (api)
Hierarchical Architecture

**Node**: single computing node/device with an independent clock

**Cluster**: any administrator-defined set of networked nodes that can communicate each other

- nodes setup over LAN

**Global**: represents global set of clusters
Types of Timelines

**Local**: discoverable only on nodes inside cluster in which the timeline is created

**Global**: discoverable by any node in the global set of clusters.
Figure 4: Quartz Time-as-a-Service. Solid boxes indicate components, dashed boxes indicate interfaces.
Timeline Service

Figure 4: Quartz Time-as-a-Service. Solid boxes indicate components, dashed boxes indicate interfaces.
QoT Clock-Synchronization Service

Figure 4: Quartz Time-as-a-Service. Solid boxes indicate components, dashed boxes indicate interfaces.
Coordination Service

Figure 4: Quartz Time-as-a-Service. Solid boxes indicate components, dashed boxes indicate interfaces.
Global View

Cluster 1

Cluster Node 1
- App 1
- App 2
- App 3
- Timeline Service
- Clk-Sync Service

Cluster Node N
- App 3
- App 4
- Timeline Service
- Clk-Sync Service

Cluster N
- App 1
- App 2
- App 4
- Timeline Service
- Clk-Sync Service

Coordination Service

Global Discovery Service

Figure 5: Quartz Time-as-a-Service at global scope
Quartz Clocks

\[ tl_{\text{now}} = tl_{\text{last}} + tl_{\text{drift}} \times (core_{\text{now}} - core_{\text{last}}) \]  \hspace{1cm} (1)

\[ \epsilon = tl_{\text{bound}} + tl_{\text{skew}} \times (core_{\text{now}} - core_{\text{last}}) \]  \hspace{1cm} (2)
Hardware Timestamping

Network interfaces usually have their own clocks and provide ability to timestamp in hardware at physical layer to enable accurate timestamping and clock synchronization
How is it autonomous?

Figure 6: Adaptive NTP: Server Selection & Rate Adaptation
Timeline Clock Synchronization

Global

NTP to synchronize to Universal Coordinated Time (UTC)

Local (cluster scope)

PTP Precision Time Protocol

Huygens - state of the art protocol
Evaluation: Accuracy

Assess accuracy of the clock-synchronization protocols that Quartz supports NTP, PTP, and Huygens

Two embedded/edge-form-factor platforms:

   - Intel NUC and Beaglebone Black (BBB)

How well do they track UTC?
## Evaluation: Accuracy

### Table 2: NTP [15] Accuracy (μs)

<table>
<thead>
<tr>
<th>Platform</th>
<th>Timestamps</th>
<th>Cluster</th>
<th>Stratum</th>
<th>Max</th>
<th>Mean</th>
<th>Std. Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUC</td>
<td>HW</td>
<td>Intra</td>
<td>1</td>
<td>4267</td>
<td>380</td>
<td>633</td>
</tr>
<tr>
<td></td>
<td>HW</td>
<td>Intra</td>
<td>2</td>
<td>12607</td>
<td>2480</td>
<td>3351</td>
</tr>
<tr>
<td>BBB</td>
<td>SW</td>
<td>Intra</td>
<td>1</td>
<td>1638</td>
<td>542</td>
<td>245</td>
</tr>
<tr>
<td></td>
<td>SW</td>
<td>Intra</td>
<td>2</td>
<td>5855</td>
<td>2380</td>
<td>717</td>
</tr>
<tr>
<td></td>
<td>SW</td>
<td>Inter</td>
<td>1</td>
<td>2127</td>
<td>929</td>
<td>553</td>
</tr>
<tr>
<td></td>
<td>SW</td>
<td>Inter</td>
<td>2</td>
<td>6033</td>
<td>3582</td>
<td>1032</td>
</tr>
</tbody>
</table>

1μs = 1 microsecond = a millionth of a second
## Evaluation: Accuracy

### Table 3: PTP [16] Accuracy ($\mu$s)

<table>
<thead>
<tr>
<th>Platform</th>
<th>Timestamps</th>
<th>Rate (s)</th>
<th>Max</th>
<th>Mean</th>
<th>Std. Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUC</td>
<td>HW</td>
<td>1</td>
<td>183</td>
<td>31</td>
<td>113</td>
</tr>
<tr>
<td></td>
<td>HW</td>
<td>2</td>
<td>220</td>
<td>24</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>HW</td>
<td>4</td>
<td>13</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>BBB</td>
<td>HW</td>
<td>1</td>
<td>14</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>HW</td>
<td>2</td>
<td>39</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>HW</td>
<td>4</td>
<td>39</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>
# Evaluation: Accuracy

## Table 4: Huygens [26] Accuracy (μs)

<table>
<thead>
<tr>
<th>Platform</th>
<th>Timestamps</th>
<th>Rate (ms)</th>
<th>Max</th>
<th>Mean</th>
<th>Std. Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUC</td>
<td>HW</td>
<td>10</td>
<td>401 (1596)</td>
<td>294 (1099)</td>
<td>21 (501)</td>
</tr>
<tr>
<td></td>
<td>HW</td>
<td>100</td>
<td>405 (382)</td>
<td>104 (105)</td>
<td>64 (75)</td>
</tr>
<tr>
<td></td>
<td>SW</td>
<td>10</td>
<td>1835 (1205)</td>
<td>294 (252)</td>
<td>242 (163)</td>
</tr>
<tr>
<td></td>
<td>SW</td>
<td>100</td>
<td>1251 (965)</td>
<td>234 (328)</td>
<td>259 (243)</td>
</tr>
<tr>
<td>BBB</td>
<td>HW</td>
<td>100</td>
<td>13000000</td>
<td>2000000</td>
<td>3000000</td>
</tr>
<tr>
<td></td>
<td>SW</td>
<td>10</td>
<td>782</td>
<td>170</td>
<td>153</td>
</tr>
<tr>
<td></td>
<td>SW</td>
<td>100</td>
<td>4593</td>
<td>1091</td>
<td>340</td>
</tr>
</tbody>
</table>
Evaluation: Scalability

Assess ability to provide time-as-a-Service at geo-distributed scale

Continental - 15 VMS -> 3 states (VA, OHIO, OR)

Global - 20 VMS -> 5 continents (NA, EU, AUS, ASIA)
## Continental Scalability Results

<table>
<thead>
<tr>
<th>Specified QoT (Accuracy)</th>
<th>Worst Delivered QoT</th>
<th>Best Delivered QoT</th>
</tr>
</thead>
<tbody>
<tr>
<td>500μs</td>
<td>442μs</td>
<td>284μs</td>
</tr>
<tr>
<td>1ms</td>
<td>994μs</td>
<td>233μs</td>
</tr>
</tbody>
</table>
# Global Scalability Results

## Table 6: Geo-distributed Scalability: Microsoft Azure

<table>
<thead>
<tr>
<th>QoT Spec.</th>
<th>Region</th>
<th>Worst QoT</th>
<th>Best QoT</th>
<th>Average QoT</th>
<th>Fraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>500μs</td>
<td>east-us</td>
<td>506μs</td>
<td>200μs</td>
<td>327μs</td>
<td>0.98916</td>
</tr>
<tr>
<td></td>
<td>central-us</td>
<td>504μs</td>
<td>216μs</td>
<td>354μs</td>
<td>0.98844</td>
</tr>
<tr>
<td></td>
<td>west-europe</td>
<td>508μs</td>
<td>249μs</td>
<td>415μs</td>
<td>0.97398</td>
</tr>
<tr>
<td></td>
<td>east-australia</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>east-asia</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>1 ms</td>
<td>east-us</td>
<td>635μs</td>
<td>199μs</td>
<td>365μs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>central-us</td>
<td>568μs</td>
<td>140μs</td>
<td>293μs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>west-europe</td>
<td>640μs</td>
<td>307μs</td>
<td>476μs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>east-australia</td>
<td>1003μs</td>
<td>490μs</td>
<td>758μs</td>
<td>0.99076</td>
</tr>
<tr>
<td></td>
<td>east-asia</td>
<td>1006μs</td>
<td>459μs</td>
<td>645μs</td>
<td>0.97398</td>
</tr>
</tbody>
</table>
Key Contributions

1) Overcoming challenges and architectural decisions in exposing TaaS to maintain timelines and estimate QoT at geo-distributed scale
2) Introduces techniques to make clock-synchronization protocols that are adaptive to application QoT requirements
3) Introduces Quartz