Ease testing of clock based systems
Emmanuel Gaudin & Mihal Brumbulli
Clock based system principle

- An input comes from reading a sensor every tick. Usually from a few times to a few hundred times per second.
- The information is computed.
- An output is generated.
- Close to the implementation
Clock based models

- Logic system
  - Binary sensor (on/off)
  - Easier to verify
Clock based models

- Continuous system
- Complex sensor (temperature, gyroscope...)
- Control law with differential equations
Testing clock based models

- Generate a value for each tick
  - Represents a massive amount of information
  - The underlying logic is not easily readable
- Example of rail signaling system
  - The door closes after 10 seconds
  - If a sample 10 times per second this will generate 100 times the same information
  - The same for the expected output to be verified

<table>
<thead>
<tr>
<th>Tick</th>
<th>Input 1</th>
<th>Input 2</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>284</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>285</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>286</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>287</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>288</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>289</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>290</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>291</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>292</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Event driven systems

- Based on messages
  - Telecommunication systems
  - High abstraction level
    - Globally Asynchronous
    - Locally Synchronous
  - Close to the high level requirements
- Parallel processing
  - Generates complexity
  - Very hard to verify
Testing of event driven systems

- Based on messages
- Stays readable
- Sequence of events more important than absolute timing
- Interleaving of messages and various parameter values create a large number of possible scenarios
- Impossible to test all possible scenarios

```java
testcase simple()
{
    port.send(m1)
    alt
    {
        [port.receive(m2)
        {
            setverdict(pass);
        }
        [port.receive(m4)
        {
            setverdict(fail);
        }
        ]
    }
}
```
Event based testing applied to clock based system

• The logical value triggers an event
  • Example
    Sync 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
    Async - - - - - - - - - - - - - - - - - - - - - - - - -
  • It is only a matter of indicating when the event occurs
  • With event driven approach only the sequence of events matter
Event based testing applied to clock based system

- The continuous value eventually triggers an event
  - An event is generated when a threshold is reached.
  - The threshold might have a margin for triggering the event.
High level requirements

- Is often event based
- Model based
- Conformance testing is close to the requirements
- Conformance testing is most suited with an event based approach
- Links to the clock based implementation
Cyber physical systems

- Combines event driven and clock based approach
- Event driven testing can be used on both parts
- Event based requirements
- Event based testing is most suited
FMI

- Functional Mock-up Interface (FMI)
- Connects Functional Mock-up Units (FMU)
- Initially made to connect clock based models
- The clock is part of the interface
- With the same testing principles, it can be a link between event driven and clock based approach
Example

- Cruise control example
- Speed and throttle are physical measures
- Start or stop are logical events
- Increase or decrease speed are events
Example

- Requirements covered by test cases
Example

- Graphical view of the test cases
Conclusion

• Even driven testing is best suited
  • Readable
  • Closer to the requirements
  • Can test any type of system (sync, async, and CPS)