KRONO-SAFE

Safe design in real-time

Poster Presentation

Formal Verification of Real-Time applications based on the Logical-Execution Time paradigm

Fabien SIRON

CIFRE PhD at Krono-Safe and Université Côte d'Azur, Inria, Kairos

September 20, 2021

Context

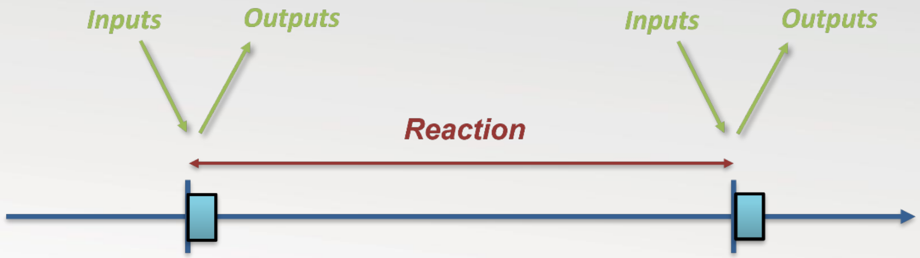
- **Our context:**
  - Safety-critical and real-time systems
  - Targets parallel platforms (multicore…)
- ***How to formally ensure that the temporal design satisfies its temporal requirements?***
  - Examples:
    - Latencies
    - Safety properties (e.g. temporal exclusion)

Multiform Logical Time

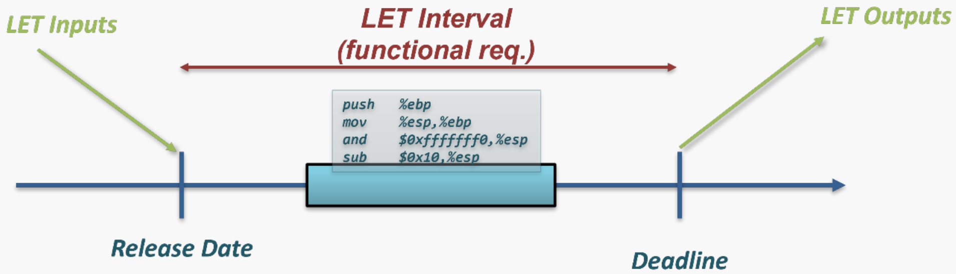
- **Give logical abstraction to real-time durations**
  - Such durations are not known during early design phases
  - They might be target dependent
- **Examples:**
  - A temporal constraint might be in a time in ms
  - … or in a time that depends of the speed of a wheel

Synchronous and LET models

- **Synchronous model:**
  - Temporally deterministic and concurrent
  - Very expressive and well suited for formal verification
  - Complex to compile (parallel platforms)



- **Logical Execution Time (LET):**
  - Temporally deterministic and concurrent
  - Easier to compile (due to logical durations)



The PsyC language

- Produced by Krono-Safe, dedicated to safety-critical real-time software integration
- Implement a variation of the LET paradigm that we call synchronous LET (sLET)
- Extension of the C language:
  – Declare logical clocks
  – Declare agents (tasks that are run concurently)
  – Express temporal behavior of agents through (s)LET intervals

Methodology

1. **Synchronous translation**
   – PsyC can be translated to Esterel
2. **Properties modeling**
   – Properties can be expressed using synchronous observers
3. **Properties verification**
   – At compile-time, using model-checking
   – At runtime, using runtime monitoring