Formal Verification of Implementability of Timing Requirements

Xiayong Hu[⋆†]
[⋆]Perimeter Financial Corp.
2 Queen Street East, Suite 1800
PO Box 21
Toronto, ON M5C 3G7
huxy@mcmaster.ca

Mark Lawford[†] and Alan Wassyng[†]
[†]Software Quality Research Laboratory
Department of Computing and Software
McMaster University, Hamilton, Canada L8S 4K1
lawford@mcmaster.ca wassyng@mcmaster.ca

Abstract

While there has been a large amount of work on validation of timing requirements, there has been relatively little work on the implementability of timing requirements. Wassyng et al. (2005) provided definitions of fundamental timing operators that explicitly considered tolerances on property durations and intersample jitter. In this work we refine the model and formalize the analysis of one of the operators in the PVS theorem prover. The first result is a full formal proof of necessary and sufficient conditions for when it is possible to implement a requirement that a boolean condition has been sustain for a duration d within a tolerance of $[d - \delta_L, d - \delta_R]$ with a discrete implementation with intersample times in the range $t_{n+1} - t_n \in [T_{min}, T_{max}]$.

1 Introduction

This work was motivated by our work on the Darlington Nuclear Generating Station Shutdown Systems software redesign project Wassyng and Lawford (2003). One of the most difficult and time consuming parts of the work involved the verification of timing properties.

2 Preliminaries

A common functional timing requirement is one that specifies that a condition must be sustained over a particular time duration. For example, to filter out the effect of a noisy signal we may specify that an event in which a sensor signal is above its setpoint should be sustained for 300 ms before it can cause a "trip". This means that the implementation must guarantee that if the sensor event is sustained for less than 300 ms, the trip must not occur. Similarly, if the sensor event is sustained for 300 ms or longer, the trip must be generated. Without tolerances on the time duration, these requirements would be impossible to meet.

We can introduce tolerances on the time duration in the above example. Assume that the sensor trip condition should be sustained for 300 ±50 ms as shown on the left of Fig. 1.

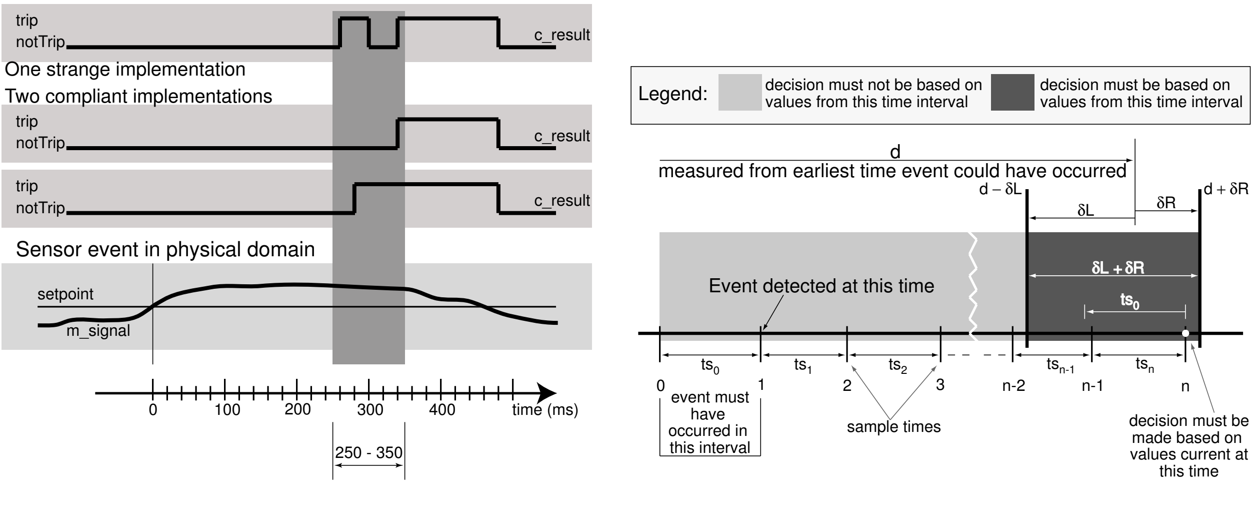
trip
notTrip
c_result
One strange implementation
Two compliant implementations
trip
notTrip
c_result
trip
notTrip
c_result
Sensor event in physical domain
setpoint
m_signal
0
100
200
300
400
time (ms)
250 - 350
Legend:
decision must not be based on values from this time interval
decision must be based on values from this time interval
d
measured from earliest time event could have occurred
d – δL
d + δR
δL
δR
δL + δR
ts0
Event detected at this time
ts0
ts1
ts2
tsn-1
tsn
0
1
2
3
n-2
n-1
n
event must have occurred in this interval
sample times
decision must be made based on values current at this time

Figure 1: Sample Intervals Required for Sustained Events

Let *Sample* : $\mathbb{N} \rightarrow \mathbb{R}^{\geq 0}$ be a sequence of sample times such that $\forall n : T_{min} \leq \mathit{Sample}(n+1) - \mathit{Sample}(n) \leq T_{max}$. We can assume $\mathit{Sample}(0) = 0$ and then for $n \geq 1$, $\mathit{Sample}(n) = \sum_{i=1}^{n} ts_i$ in Figure 1 and $\forall i : T_{min} \leq ts_i \leq T_{max}$.

3 Results

There are several different implementation scenarios under which we might have to recognize a sustained event. These are (a) Omniscient - we know the exact time that the condition becomes true but can only react at sample times, (b) We only know the value of the condition at sample instances but we know the exact timing of samples (i.e. we can look at a perfect real valued clock), (c) same as (b) but with access to an imperfect clock (e.g. finite precision, bounded drift) , and (d) Knowledge only of T_{min} and T_{max} and the number of samples since the condition became true. Below we formally state the condition for implementability in case (b).

$$\begin{aligned} \mathit{Feasible}(d) : \mathit{bool} \quad &= \quad \forall \mathit{Sample} : \forall n_0 : \exists n : \forall (t | t >= \mathit{Sample}(n_0) \wedge t <= \mathit{Sample}(n_0+1)) : \\ & \qquad \mathit{Sample}(n) - t >= d - \delta_L \wedge \mathit{Sample}(n) - t <= d + \delta_R \end{aligned}$$

The following theorem has been formalized and proven in PVS.

Theorem 1. *Assume that* $T_{min} < T_{max}$, $\delta_L, \delta_R > 0$, *and* $d > \max(\delta_R, T_{max} + \delta_L)$. *Let* $K_{min} = \lfloor (d - delta_L)/Tmax \rfloor$. *Then*

Case 1 *: If* $T_{max} \leq \frac{\delta_L + \delta_R}{2}$ *then* $\mathit{Feasible}(d)$

Case 2 *:* $\frac{\delta_L + \delta_R}{2} < T_{max} \leq \delta_L + \delta_R$ *then*

$$T_{min} \geq \frac{d - \delta_L}{K_{min} + 1} \wedge (K_{min} + 2) * T_{max} <= d + \delta_R \Leftrightarrow \mathit{Feasible}(d)$$

Case 3 *:* $T_{max} > \delta_L + \delta_R$ *then* $\neg \mathit{Feasible}(d)$

4 Related Work and Conclusions

Within the timed automata formalism, Wulf et al. (2005) provide a sufficient condition for implementability of the timed automata in terms of a global upper bound on the system latency. By contrast we provide a necessary and sufficient condition for the implementability of the basic real-time sustained condition requirement in terms of tolerance on the duration, the sample rate and the jitter. The later theory allows for per requirement tolerances to be specified and verified rather than requiring all parts of the implementation meet some global minimum response time. Case 2 is the interesting since it implies that sometimes it is possible to make a requirement implementable by sampling slower!

Acknowledgements

This work was partially funded by the Natural Sciences and Engineering Research Council of Canada. Mark Lawford is currently on sabbatical at the University of Limerick's Software Quality Research Lab and would like to thank Prof. David Parnas and the Science Foundation of Ireland for their financial support. Xiayong Hu would like to acknowledge the support of Perimeter Financial Corp.

References

A. Wassyng, M. Lawford, and X. Hu. Timing tolerances in safety-critical software. In J. Fitzgerald, I.J. Hayes, and A. Tarlecki, editors, *FM 2005: Formal Methods: International Symposium of Formal Methods Europe Proceedings*, volume 3582 of *LNCS*, pages 157 – 172, Newcastle, UK, July 2005. Springer-Verlag.

Alan Wassyng and Mark Lawford. Lessons learned from a successful implementation of formal methods in an industrial project. In K. Araki, S. Gnesi, and D. Mandrioli, editors, *FME 2003: International Symposium of Formal Methods Europe Proceedings*, volume 2805 of *LNCS*, pages 133–153, Pisa, Italy, August 2003. Springer-Verlag.

Martin De Wulf, Laurent Doyen, and Jean-Franois Raskin. Systematic implementation of real-time models. In J. Fitzgerald, I.J. Hayes, and A. Tarlecki, editors, *FM 2005: Formal Methods: International Symposium of Formal Methods Europe Proceedings*, volume 3582 of *LNCS*, pages 139 – 156, Newcastle, UK, July 2005. Springer-Verlag.