VASY reports a deadlock in the IEEE 1394 "Firewire" standard

February 13, 1998

Executive Summary

This document reports an error in the part of the state machine describing the asynchronous mode of the Link Layer protocol of the IEEE Standard 1394 "Firewire" high-speed serial bus. This error leads the protocol to a deadlock state. It was revealed by the use of formal methods (more specifically the ISO Formal Description Technique LOTOS and the software engineering toolbox CADP). A modification of the Link Layer state machine is proposed.

1. Introduction

Formal methods, especially model-checking verification techniques, are useful to detect logical errors in complex designs of computer systems, e.g., communication protocols, distributed systems, sequential circuits, etc.

In particular, formal methods can be used to find nontrivial errors in international standards. According to Prof. Edmund M. Clarke, the first time that formal methods were used to find nontrivial errors in an IEEE standard concerned the verification of the cache coherence protocol in the IEEE Futurebus+ Standard. This verification revealed several errors that were previously undetected.

This document reports on a recent experiment conducted with another IEEE Standard 1394 ("Firewire"). The Firewire is a high-speed serial bus for multimedia PCs. Technical and commercial information about the Firewire can be obtained from the 1394 Trade Association.

This experiment was conducted by the VASY action of Inria Rhône-Alpes and Dyade, the Bull-Inria Joint Venture for Advanced Research. It took place in the framework of the European project COST 247 (Verification and Validation Methods for Formal Descriptions) and was presented during the 2nd COST 247 International Workshop on Applied Formal Methods in System Design (June 1997, Zagreb, Croatia).

The three layers of the Firewire bus (Physical, Link, and Transaction layers) were formally specified in LOTOS (a specification language standardized by ISO [ISO-8807] and E-LOTOS, an extension of LOTOS under development. They were verified using CADP, a software engineering toolbox for protocols and distributed systems.

This experiment revealed an error in the part of the state machine describing the asynchronous mode of the Firewire Link layer protocol. The remainder of this document focuses on the error and suggests a correction of the IEEE standard. A detailed description of the experiment can be found in [Sighireanu-Mateescu-97].

2. Problem Report

When searching for deadlocks in the Firewire protocol, the Exhibitor tool of CADP revealed a sequence of 50 transitions leading to a deadlock.

This sequence was minimal, in the sense that no deadlock can occur in less than 50 transitions (this is important, since the probability to discover this deadlock using simulation and testing techniques is quite low).

Roughly speaking, the deadlock occurs after the following sequence of events:

- The Link Layer of some node receives a broadcast packet
- The Link Layer indicates the arrival of this packet to its Transaction Layer
- The Transaction Layer sends a Link Data Response to the Link Layer with the parameter NO_OPERATION

More precisely, the figure below represents the exact sequence of events found by the Exhibitor tool for the particular scenario (scenario 3 in [Sighireanu-Mateescu-97]) where only two nodes (Node 0 and Node 1) are connected to the Bus and the first one (Node 0) sends two consecutive broadcast packets. (The comments on each action are introduced by "--". More explanations follow after this figure.)

```
<initial state>
"i" (i)
"i" (i)
"LDREQ !0 !2 !C1 !D1"           -- Node 0 sends the first broadcast packet
"PAREQ !0 !FAIR"                -- Beginning of Bus arbitration and packet transmission
"PACON !0 !WON"
"PCIND !0"
"PDREQ !0 !START"
"PDIND !1 !START"
"PCIND !0"
"PDREQ !0 !DHEAD"
"PDIND !1 !DHEAD"
"PCIND !0"
"PDREQ !0 !SIG (DESTSIG (2))"
"PDIND !1 !SIG (DESTSIG (2))"
"PCIND !0"
"PDREQ !0 !SIG (HEADERSIG (C1, CHECK))"
"PDIND !1 !SIG (HEADERSIG (C1, CHECK))"
"PCIND !0"
"PDREQ !0 !SIG (DATASIG (D1, CHECK))"
"PDIND !1 !SIG (DATASIG (D1, CHECK))"
"PCIND !0"
"PDREQ !0 !END"
"PDIND !1 !END"                 -- End of packet transmission
"LDIND !1 !BROADREC (C1, D1)"   -- Node 1 indicates the packet arrival
"LDCON !0 !BROADSENT"           -- Node 0 confirms the sending
"PDIND !0 !SUBACTGAP"           -- The Bus Layer indicates the end of a packet
"PDIND !1 !SUBACTGAP"           -- transmission to all nodes
"ARBRESGAP"                     -- Start of a new arbitration period
"i" (i)
"i" (i)
"LDREQ !0 !2 !C1 !D1"           -- Node 0 sends the second broadcast packet
"PAREQ !0 !FAIR"                -- Beginning of Bus arbitration and packet transmission
"PACON !0 !WON"
"PCIND !0"
"PDREQ !0 !START"
"PDIND !1 !START"
"PCIND !0"
"PDREQ !0 !DHEAD"
"PDIND !1 !DHEAD"
"PCIND !0"
"PDREQ !0 !SIG (DESTSIG (2))"
"PDIND !1 !SIG (DESTSIG (2))"
"PCIND !0"
"PDREQ !0 !SIG (HEADERSIG (C1, CHECK))"
"PDIND !1 !SIG (HEADERSIG (C1, CHECK))"
"PCIND !0"
"PDREQ !0 !SIG (DATASIG (D1, CHECK))"
"PDIND !1 !SIG (DATASIG (D1, CHECK))"
"PCIND !0"
"PDREQ !0 !END"
"PDIND !1 !END"                 -- End of packet transmission
"LDCON !0 !BROADSENT"           -- Node 0 confirms the sending
"PDIND !0 !SUBACTGAP"           -- The Bus Layer indicates the end of a packet transmission to Node 1 and blocks
<deadlock>
```

This sequence corresponds to the following evolution of the system:

- The Transaction Layer of Node 0 communicates to its Link Layer a request for sending the first broadcast (first action "LDREQ !0 !2 !C1 !D1")
- The Link Layer of Node 0 sends the packet over the Bus after receiving the Bus access (the twelve actions between "PAREQ !0 !FAIR" and "PDIND !1 !END")
- The Link Layer of Node 1 indicates the arrival of a broadcast packet to its Transaction Layer

(actions "LDIND !1 !BROADREC (C1, D1)")

> In response to this indication, the Transaction Layer of Node 1 must send to its Link Layer a message with the parameter NO_OPERATION through the Link Data Response gate. However, the communication *cannot* take place because the Link Layer of Node 1 (as described in the Link Layer state machine given at page 166 of the IEEE-1394 Standard) *cannot* accept such a message. The Transaction Layer of Node 1 is blocked by this unsuccessful communication.

- The Link Layer of Node 0 confirms the successful transmission of the packet to its Transaction Layer (action "LDCON !0 !BROADSENT")
- The Bus Layer indicates the end of a packet transmission to all Link Layers (actions "PDIND !0 !SUBACTGAP" and "PDIND !1 !SUBACTGAP") and restarts a new arbitration period (action "ARBRESGAP")
- The Transaction Layer of Node 0 communicates to its Link Layer the request for sending the second broadcast (second action "LDREQ !0 !2 !C1 !D1")
- The Link Layer of Node 0 sends the packet over the Bus after receiving the Bus access (the twelve actions between "PAREQ !0 !FAIR" and "PDIND !1 !END")

> After reception of the packet, the Link Layer of the Node 1 must indicate to its Transaction Layer the arrival of the second broadcast packet. This is not possible because the Transaction Layer of Node 1 is blocked. By consequence, the Link Layer of the Node 1 is blocked.

- The Link Layer of Node 0 confirms the successful transmission of the packet to its Transaction Layer (action "LDCON !0 !BROADSENT")
- The Bus Layer must indicate the end of a packet transmission to all the Link Layers. It can do this for the first node (action "PDIND !0 !SUBACTGAP"), but it cannot do the same for the second node because the Link Layer of Node 1 is blocked. The Bus Layer is blocked, and the system deadlocks.

3. Proposed Change in the IEEE 1394 Standard

This deadlock is caused by the fact that, in the state machine diagram of the Link Layer (page 166 of IEEE Standard 1394), no Link Data Response can be accepted after a Data Indication following the reception of a broadcast packet.

Therefore, the Link Layer state machine given on page 166 of IEEE Standard 1394 should be modified as follows:

6.3.3 Details of link layer operation

The operation of the link layer packet transmitter and receiver is described by the state machine in figure 6-19.

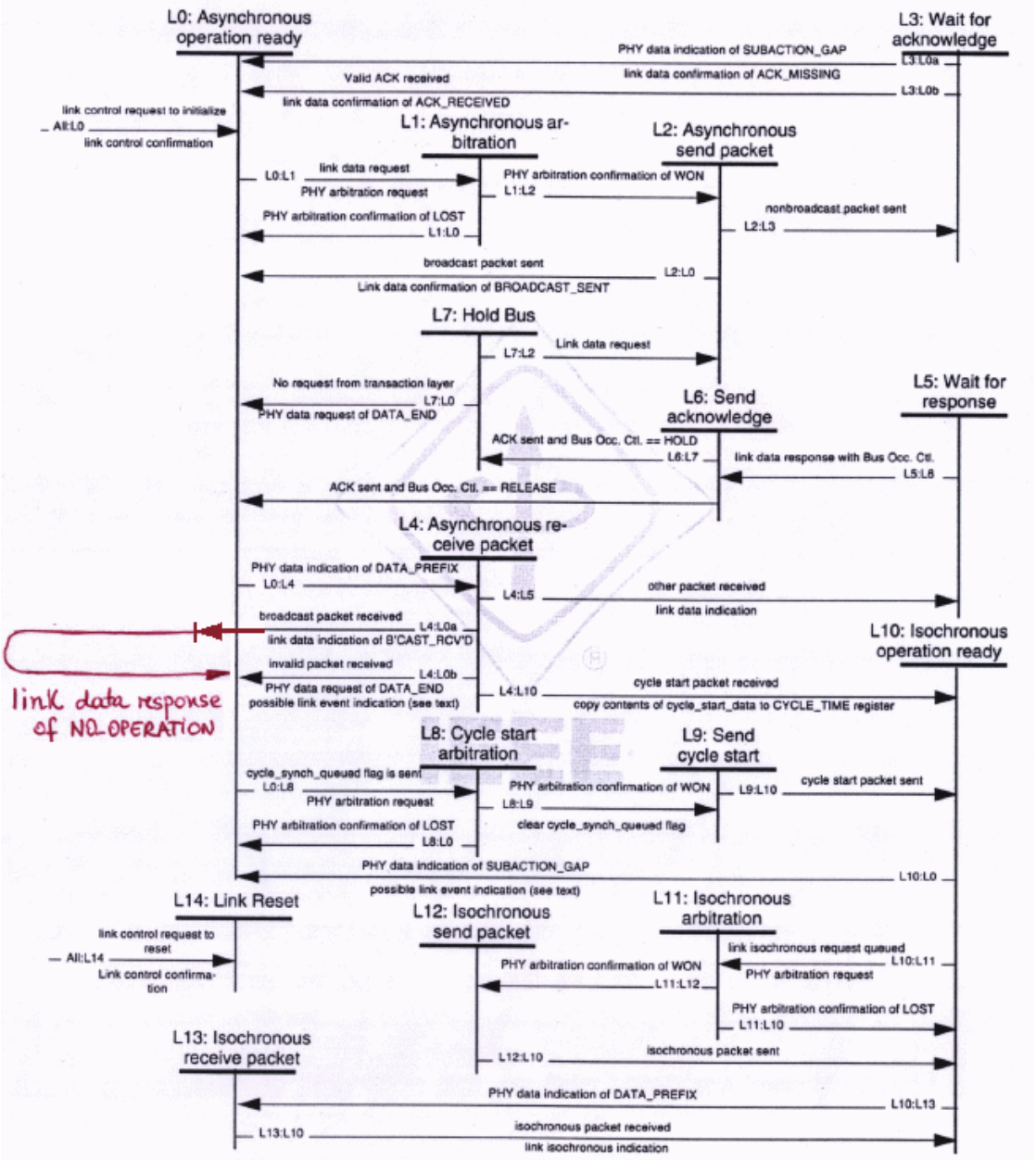

Figure 6-19—Link layer packet transmit/receive state machine

Precisely, the transition L4:L0a should be split in two successive transitions separated by an intermediate state:

- The first transition consists of the existing transition L4:L0a, whose target is modified: the transition no longer arrives in state L0, but in a new intermediate state.
- From this intermediate state, a second transition starts. It is labeled with the condition Link Data Response of NO_OPERATION and no action. The target of this transition is L0.

It is worth noticing that the proposed change is compatible with the informal explanations given on page 141 of IEEE Standard 1394, stating that:

> "The Transaction Layer shall communicate this response [i.e., Link Data Response] after receiving a Link Data Indication."

and, on page 142:

> "[Upon receipt of a Link Data Response with] NO_OPERATION, the Link Layer shall do

> nothing".

It is clear that the state machine given on page 166 is incorrect and should be amended as suggested above, because state machines have precedence over textual explanations. Indeed, page 10 of the Standard states that:

> "The description of operations in this standard are done in three ways: state machines, C++ code segments, and English language. If more than one description is present, then priority shall be given first to the state machines, then the C++ code segments, and finally to the English text (including the state machine notes).

4. Another Error Discovered in the IEEE 1394 Standard

There is another problem in the Link Layer protocol. This other problem was found independently by Bas Luttik [Luttik-97] and Lars Kuhne [Kuhne-Hooman-Roever-97] who studied the protocol in the framework of theorem-proving (using respectively muCRL and PVS).

The problem appears when the Link Layer receives a packet from the Physical Layer while it has a pending request from the Transaction Layer to send some other packet. The state machines of the IEEE 1394 Standard do not specify what should happen with the pending request of the Transaction Layer.

To solve the problem, Bas Luttik and Lars Kuhne suggested two different changes:

- The solution proposed by Bas Luttik is to add to the Link Layer a one place buffer. This buffer contains packets received from the Transaction Layer and not yet sent. The Link Layer may receive a request from the Transaction Layer if and only if its buffer is empty.

- The solution proposed by Lars Kuhne follows the indication of the explanatory text of the state machine given in the IEEE 1394 Standard at page 167. The pending request for arbitration is considered lost together with the request of the Transaction Layer.

In the experiment described in [Sighireanu-Mateescu-97], the solution proposed by Bas Luttik was used. However, the aforementioned deadlock problem is independent from Luttik's solution, and would occur as well with Kuhne's solution.

5. Concluding Remarks

Formal methods such as LOTOS and verification tools such as CADP provide a valuable help for specifying and analyzing the behaviour of complex systems. This is clear in the experiment described in [Sighireanu-Mateescu-97], which was carried out in only one man.month, without prior knowledge of the Firewire protocol.

Even if the semi-formal description techniques used in IEEE Standard 1394 (i.e., state machines) are more precise than informal English text, they are not precise enough and leave room for ambiguities and misinterpretations.

In particular, the semantics of these state machines is problematic for several reasons:

- Actions are possible in discrete states *and* on transitions. This does not correspond with the widely accepted formalisms for communicating state machines, such as Mealy machines, etc.

- The description of discrete states is not given in terms of state machines but in natural language. In this way, some actions are possible in discrete states but they are not visible in the state machine diagrams because they are hidden in the discrete states.

- The semantics of state machine interconnection is not defined. This is clearly a problem, because the whole Firewire protocol is based on parallel execution of state machines.

Acknowledgments

We are grateful to Dr Ron Koymans (Philips Research, Eindhoven, The Netherlands) for his suggestions and careful reading of this page. Also, we thank Viktor Gyuris (University of Illinois, Chicago) who pointed out an ambiguity in a previous version of this page.

References

ISO-8807	LOTOS. A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour. International Standard ISO/IEC 8807. September 1988
Kuhne-Hooman-Roever-97	Lars Kuhne and Jozef Hooman and Willem-Paul de Roever. Towards Mechanical Verification of Parts of the IEEE P1394 Serial Bus. In Proceedings of the 2nd COST 247 International Workshop on Applied Formal Methods in System Design (Zagreb, Croatia), June 1997. Contact: lku@informatik.uni-kiel.de, wsinjh@win.tue.nl, wpr@informatik.uni-kiel.de
Luttik-97	Bas Luttik. Description and Formal Specification of the Link Layer of P1394. In Proceedings of the 2nd COST 247 International Workshop on Applied Formal Methods in System Design (Zagreb, Croatia), June 1997. Available on-line at: http://www.cwi.nl/~luttik/
Sighireanu-Mateescu-97	Mihaela Sighireanu and Radu Mateescu. Validation of the Link Layer Protocol of the IEEE-1394 Serial Bus (``FireWire''): an Experiment with E-LOTOS. In Proceedings of the 2nd COST 247 International Workshop on Applied Formal Methods in System Design (Zagreb, Croatia), June 1997. The full version of this paper is available as INRIA Research Report RR-3172

Contact persons

Mihaela SIGHIREANU and Radu MATEESCU
INRIA Rhone-Alpes / Dyade
655, Avenue de l'Europe
38330 Montbonnot Saint Martin, FRANCE
Tel: +(33) 04 76 61 52 83
Fax: +(33) 04 76 61 52 52
E-mail: Mihaela.Sighireanu@inria.fr, Radu.Mateescu@inria.fr
Web: http://cadp.inria.fr

Version 1.16 - Last updated 2015/09/18 16:08:17

Back to the VASY Home Page