DISTRIBUTED INTERRUPTS MECHANISM VERIFICATION AND INVESTIGATIONBY MODELING ON SDL AND SYSTEMC

Session: SpaceWire networks and protocols

Short Paper

Liudmila Onishchenko, Artur Eganyan, Irina Lavrovskaya

Saint-Petersburg University of Aerospace Instrumentation. 67, B. Morskaya, Saint-Petersburg, Russia

E-mail: luda_o@rambler.ru, artfla@rambler.ru, i_lavrovskaya@mail.ru

ABSTRACT

Distributed Interrupt mechanism has been proposed for next SpaceWire standard release. Interrupt codes and Interrupt_Acknowledge codes are low-latency signaling codes, according to specification their distribution does not depend on data flow. That makes it useful for real-time distributed systems interconnections. In this paper we present: the distributed interrupts mechanism parameters and their estimation, verification by using model on the Specification and Description Language (SDL); investigation and analysis of this mechanism by using the SpaceWire Network Functional Model in SystemC and we also give some recommendations about using the distributed interrupts mechanism and choosing the timeout parameters for efficient recovery.

1. DISTRIBUTED INTERRUPT MECHANISM OVERVIEW

The detailed Distributed Interrupt description is in [1]. Interrupt-Code represents a system signal request. It is issued by a node link that will be considered as the source node for this interrupt (Interrupt Source). It is distributed over the network to all other nodes. An Interrupt-Code should be accepted for handling in some node of the SpaceWire network, which will be called the Interrupt Handler. The host of the node is supposed to implement some interrupt processing routine. One of 32 interrupt request signals (interrupt source identifiers) could be identified by the Interrupt-Code.

Interrupt_Acknowledge-Code represents a confirmation that the Interrupt-Code has reached some Interrupt Handler and has been accepted by it for processing. The Interrupt Handler node should send an Interrupt_Acknowledge-Code with the same five-bit interrupt source identifier as in the accepted Interrupt Code. The Interrupt-Code is broadcasted to find an Interrupt Handler node. To eliminate infinite cycling of the broadcasted control code specific mechanisms and rules for its handling in nodes and routers are provided:

Each link controller of a node and each router contain one 32-bit Interrupt Source Register (ISR). When the link interface receives from its host an interrupt request with a five-bit interrupt identifier it sets appropriate bit to '1' in the 32-bit ISR. Then it sends out the Interrupt-Code with the five-bit interrupt source identifier field. If the correspondent bit in the ISR is in '1'state already then the Interrupt-Code is not sent out. A subsequent Interrupt-Code with the same interrupt source identifier can be sent by the link only after receipt of an Interrupt Acknowledge with the correspondent interrupt source identifier. In a router, when a link interface receives an Interrupt-Code it checks the correspondent bit in the ISR. If the bit is '0' it sets the ISR bit to '1' the signal propagates to all the router output ports (except the port that have issued the signal). But if the correspondent bit in the 32-bit ISR is equal to '1' the Interrupt-Code will be ignored (to prevent repeated Interrupt-Code propagation in networks with circular connections). The router shall not retransmit the Interrupt-Code to its output ports.

The Interrupt-Code handling time T_H shall be more that maximum Interrupt-Code propagation time from Interrupt Source to the farthest node (may be not Interrupt Source) in the network by the longest way in the worst case (T_{IHmax}). The Time Tg between getting by Interrupt Source Interrupt_Acknowledge-Code and sending next Interrupt-Code shall be more then maximum Interrupt_Acknowledge-Code propagation time from Interrupt Handler to the farthest node (may be not Interrupt Handler) in the network by the longest way in the worst case (T_{IGPmax}). Because of symmetry Interrupt-Code and Interrupt_Acknowledge-Code mechanism propagation $T_{IHmax} = T_{IGPmax} = T_{IPmax}$.

In a SpaceWire network faults and errors may occur: link disconnect error or parity error can cause an Interrupt -Code/Interrupt_Acknowledge-Code loss; there may be spontaneous change of an ISR bit state as a result of intermittent faults in a node or in a router. To ensure tolerance against faults and spontaneous changes in ISR special timers are used. Each ISR in a node or in a router has a timer per ISR bit. A timer starts at the receipt of an Interrupt-Code with correspondent five-bit interrupt source identifier and resets at receipt of an Interrupt_Acknowledge-Code with the same interrupt source identifier. In case of timeout before the timer is reset, the ISR timeout event arises; the correspondent ISR bit should be reset to '0'.

2. DISTRIBUTED INTERRUPT PARAMETERS

To use Distributed Interrupt parameters we should define the following parameters: $T_{timeoutN}$ – a timeout value for ISR in Nodes; $T_{timeoutR}$ – a timeout value for ISR in Routers; Tg, T_H – see before. This parameters depends on the network topology, router architecture and link bit rates and they affect to the latency characteristics that important for real-time. We give some expressions for them below.

Let D – network diameter (depends upon the SpaceWire network interconnection topology); T_{bit} – one bit transfer time; T_{wtc}– Time-code transport through router delay (ignoring interference with previous characters/codes; depends upon implementation); $PLen$ – the length of the longest way between any two nodes in the network (depends only on the network topology). The T_{IPmax}, T_H and Tg values are: $T_{IPmax} = (PLen - 1)*(Twtc + 13Tbit + 31*14*Tbit) + (14Tbit)* PLen$; and $Tg = T_H = 1,5\ T_{IPmax}$

A maximum Interrupt-Code (and Interrupt_Acknowledge-Code) time delivery T_{Imax} is:

T_{Imax} = (D-1)*($Twtc$ + 13$Tbit$ + 31*14*$Tbit$) + (14$Tbit$)*D

In practice it is hardly ever to distinguish a maximum propagation time, but we should use it for timeout value computation. Timeout value $T_{timeoutR}$ should be less then timeout value $T_{timeoutN}$ on T_{IPmax}. It will provide, that in the node and in all routers timeouts $T_{timeoutR}$ will be over, when timeout $T_{timeoutN}$ is expired. Timeout value for router $T_{timeoutR}$ shall be: $T_{timeoutR}$ = 2(2T_{Imax} + 1,5 T_{IPmax}), then $T_{timeoutN}$ = $T_{timeoutR}$ + T_{IPmax}. A mean Interrupt-Code and Interrupt_Acknowledge-Code propagation time is much lower then maximum value: T_{Imean} = (D-1)*($Twtc$ + 13$Tbit$ + 2*14*$Tbit$) + (14$Tbit$)*D

If all links in the network have different bit rates, it should be taken into account in computation of timeout values and processing time, for example to use the value of the minimal bit rate in the network.

3. EXAMPLE

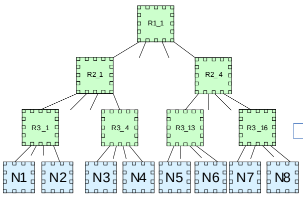


Fig. 1. Tree topology

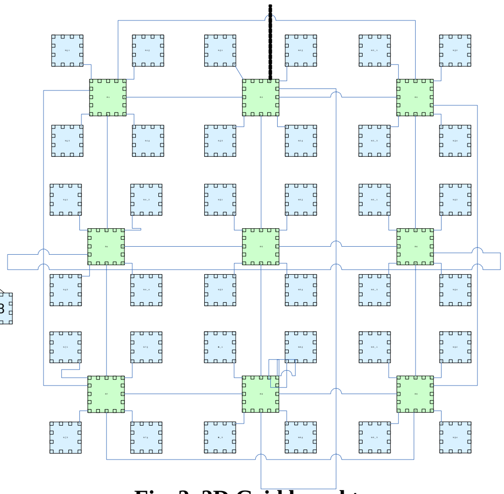


Fig. 2. 2D Grid based tor

For example we use topology shown in Fig.1. and Fig.2 At 400 Mb/s and 10Mb/s we got values (see Table 1.). First two rows correspond to Grid based tor network, last two-correspond tree network

Table 1

PLen	*D*	*Twtc*,us	*Tbit*,us	T_{IPmax},us	$Tg = T_H$,us	T_{Imax},us	$T_{timeoutR}$,us	$T_{timeoutN}$,us	T_{Imean},us
10	4	0.2	0.0025	12.2075	18.31125	4.0925	52.9925	65.2	1.0475
10	4	0.2	0.1	418.1	627.15	140.3	1815.5	2233.6	18.5
6	6	0.2	0.0025	6.7975	10.19625	6.7975	47.5825	54.38	1.7225
6	6	0.2	0.1	232.9	349.35	232.9	1630.3	1863.2	29.9

4. SDL TOOLSUITE

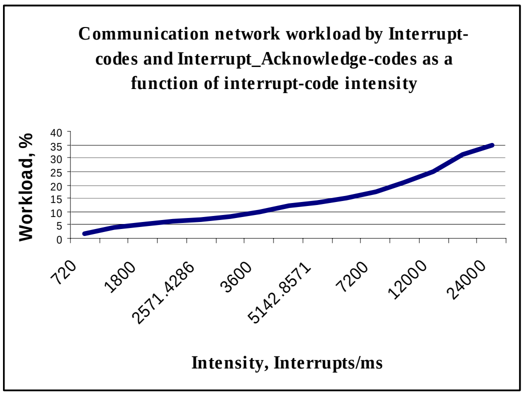
SDL ToolSuite gives an ability to implement specifications. A distributed interrupt system implementation on the SDL allows to have a reference implementation for it and check how Interrupt codes and InterruptAcknowledge codes are sent through the network. This model can be verified and it is very useful to check the correctness of the specification.

The model implemented on the SDL fully conforms to the distributed interrupts mechanism of the SpaceWire network. SDL model is intended for a demonstration of an interrupt mechanism functionality and for a verification of the SpaceWire specification. This model isn't just an example of a network functionality, but it is in a strict correspondence with the specification.

The SDL model of the distributed interrupts includes the description of general elements of the SpaceWire network. These elements are a node, a router and a link. Communication between node and router is implemented as two unidirectional channels, turned different directions. Using these elements it is possible to create networks of any difficulty and construction. In the presented model it is possible to observe how the Interrupt codes and InterruptAcknowledge codes go through the network. Also there is an opportunity to model some difficult in investigation situations with lost of data, errors in links, distribution recovery in case of errors etc.Using the SDL ToolSuite it is possible to investigate distributed interrupts mechanism, check the correctness of the realization by verification.

5. Investigation by SpaceWire Network Functional model

The SpaceWire Network Functional model (SpWNM) includes a description of basic SpaceWire network elements like node, routing switch and link, allows to assemble a SpaceWire interconnection system of required structure, implements wormhole routing, time flow and distributed interrupts mechanisms, generation and transmission of data packets. We use this tool for distributed interrupts mechanism investigations. In the sections 2 and 3 we calculate distributed interrupt parameters. We use them for investigation in SpWNM. The 2D Grid based tor is more complicated for distributed interrupts so it is more interesting. Some investigation results are shown below.



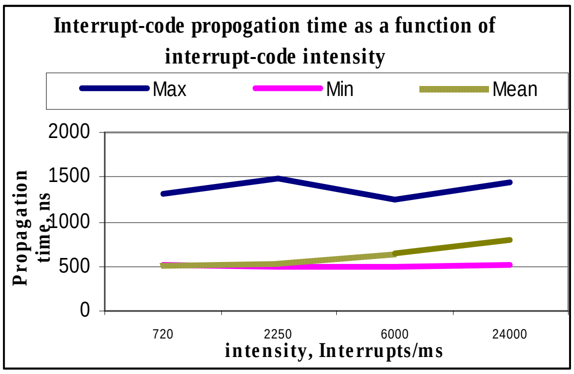


Fig.3. Communication network workload by Interrupt-codes and Interrupt_Acknowledge-codes as a function of interrupt-code intensity. Interrupt-code propogation time as a function of interrupt-code intensity

6. References

1. Yuriy Sheynin, Sergey Gorbatchev, Liudmila Onishchenko, “Real-Time Signalling in SpaceWire Networks”. International SpaceWire Conference, Dundee 2007. Conference Proceedings. ISBN: 978-0-9557196-0-8, 4pg.