

Retransmission Control Scheme for RDM Protocol for Lightning Control Networks

Min Gyung Kwak
Pangyo R&D Center
Hanwha Techwin, Korea
m.kwak@hanwha.com

Jinsuk Baek, Paul S. Fisher
Department of Computer Science
Winston-Salem State University, USA
{baekj, fisherp}@wssu.edu

Abstract— In lighting control networks, remote device management is supported by the Remote Device Management (RDM) protocol. However, the RDM protocol does not provide any mechanism for reliable packet transmission. For reliable transmission, a controller needs to integrate the status information of its lightning devices and perform error recovery for these devices using the data stored in its buffer. We propose a retransmission control scheme that uses two types of ACKs. The proposed scheme significantly reduces ACK implosion at the controller and ultimately provides scalable reliable device management for the lighting control network.

Keywords—Reliable transmission; Lighting control network; RDM; DMX-512A

I. INTRODUCTION

Digital MultipleX (DMX) is the most successful and widely deployed entertainment lighting control standard. DMX is simple but effective to use and implement. The best known version is DMX-512A [1] supporting 512 asynchronous control slots, which are available for transmission. DMX-512A is considered unreliable from a networking perspective, in that the data is sent but not acknowledged by the receiving devices. The only error correction inherent to DMX-512A is the constant refresh of data. This simplicity leads to a fast transmission at first. However, as the lightning market has matured and expanded, the limitations of DMX-512A have become apparent, especially in large or complex networks.

In order to resolve these complexities, Remote Device Management (RDM) [2] over DMX512-A Networks was released by ESTA (now PLASA) in 2006, with a revision in 2010. RDM has many desirable features that manage the devices in lighting control networks. RDM provides an open, bi-directional communication approach between a control console and connected lighting devices. It also allows a controller to automatically discover connected lighting devices, and then configure, monitor, and manage them remotely.

The RDM error recovery procedure appears straightforward at first, but in the real world, transmissions do not always perform correctly. Moreover, any mechanism for reliable transmission has not been considered in the RDM protocol. It means there is no recovery process for erroneous or damaged packets or retransmission process for lost packets. Of course, the lack of reliable transmission can make the DMX-512A network unavailable over time due to the existence of large quantities of erroneous packets. Therefore,

the RMD protocol must handle abnormal transmissions if they are to operate reliably. Developing an efficient and reliable transmission mechanism becomes one of the more challenging issues when building an RDM protocol.

In order to provide reliable transmission for the RDM, a sequence-based scheme [3] has been proposed. With the scheme, each device is required to send an ACK for every successfully received packet by attaching the Transaction Number (TN) of the packet. By checking the receiving order of the TNs, the controller identifies the lost packets and then retransmits those packets. However, it does not scale up well due to the ACK implosion occurring at the controller.

To overcome this limitation, we propose a new scheme combining an adaptive Selective ACK (S_ACK) and infrequent Cumulative ACK (C_ACK). Under our scheme, a device sends S_ACKs to the controller to request packet retransmission. At fixed, infrequent intervals, it also sends C_ACKs to indicate which packets can be safely discarded from the controller's sending buffer.

II. EXISTING SCHEMES

DMX-512A is designed based on a physical layer specification, called EIA-485-A, which is the most commonly used standard in industrial applications. The RDM uses the full bi-directional capability of EIA-485-A to interleave bi-directional data on a DMX-512-A link. By using the interleaved data, the controller can set DMX-512-A addresses remotely, monitor device status, such as whether a lamp is on, or the internal temperature or currently available battery power of a device. Since the EIA-485-A is a specification for a long-distance, multi-drop data bus, all of this can be done along with a single pair of wires in which a number of devices can be distributed [4].

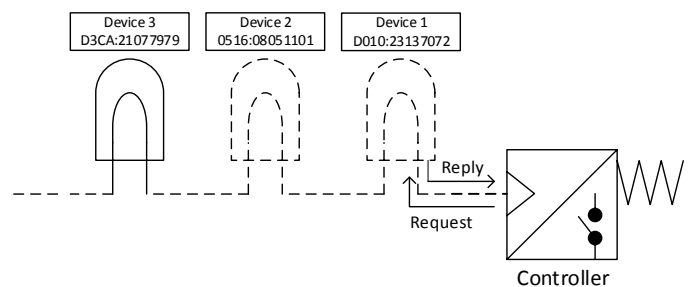


Figure 1. An example of daisy-chain network topology

In an RDM, general controller-to-device communication occurs in a request-response pattern. The controller sends the request together with an 8-bit TN to the devices by addressing it using the device's six-byte UID. The first two bytes of a UID are Manufacturer ID (MID) and the last four bytes are the Device ID (DID). UIDs are represented as hexadecimal numbers, with a colon separating the MID and the DID (e.g., 4F5A:AA00C216). When the request has been sent, the controller temporally stores the packet in its buffer and waits for the response from the device for a specific period of time (e.g., 2.1 ms). During this time, the controller relinquishes control of the DMX line and the device can transmit its response along with the TN. If the response arrives, the controller processes the response including 1) checking to see that the request-response TN is matched, 2) verifying the checksum field for data integrity, 3) removing the packet from its buffer if the delivery was successful, and 4) preparing the next request if there are any. This process is repeated until there is no additional requests to send. If the controller fails to receive a response within a given period of time, the controller will assume the response packet is lost. Although, a legacy RDM provides the minimum level of error detection by involving a checksum field, it does not attempt to recover from an error. Instead, the controller simply discards the damaged response. Also, the retransmission is not required for the lost packets. Figure 1 shows the linear, daisy-chain, network topology of the DMX network with data flow between a controller and its associated devices.

In sequence-based retransmission scheme [3], each device is required to send an S_ACK as a response for the request. This response explicitly identifies the successful reception of the corresponding request packet because each device is supposed to attach the same TN to the response packet as that of request packet. The controller performs retransmission by referencing the TN of the response packet. That is, the out-of-order TN is used as an indicator for unsuccessful packet delivery of the previously transmitted but not yet acknowledged request packets. The controller retransmits those packets to the device.

III. PROPOSED SCHEME

The S_ACK alone cannot support an efficient buffer management mechanism because it is not a safe indicator to discard multiple packets from the controller's buffer. For example, even though the controller receives an S_ACK (5), if the packet is out-of-order, the controller cannot safely discard up to packet number 5 from its buffer. Also, as the DMX link itself is fairly reliable, sending an S_ACK for an individual packet introduces feedback pollution on the DMX link. In order to address these issues, the proposed scheme requires the device to adaptively send a Cumulative ACK (C_ACK) or a new type of S_ACK based on the given packet reception status.

With the proposed scheme, the controller sets max_B that is the maximum number of packets that it is willing to maintain in its buffer. That is, max_B is a buffer size of the controller to keep already sent but not yet acknowledged packets as well as a current packets being sent. The controller explicitly informs the device of the size of max_B. Once the size of max_B is sent, the device will set its feedback sending

interval INT. Note that the size of the INT should be set to an integer number which is at most one less than max_B, because the sender cannot keep more than max_B packets including a currently sending packet. This can be set at the end of device discovery phase [5, 6]. The device sends a C_ACK or S_ACK to the controller no more frequently than every INT packets depending on the reception status of the INT packets.

$$INT \leq \max_B - 1 \quad (1)$$

By referencing the TN specified on C_ACK sent by the device, the controller can safely discard up to that packet from its buffer. Even though this optimistic property will require extended buffer space compared to legacy RDM protocol, the C_ACKs allow the controller to safely discard multiple packets from its buffer at once.

When the device receives an out-of-order packet in terms of TN, it observes the packet reception status within the INT packets. After that, it sends a single S_ACK to request retransmission of the lost packets. Note that our S_ACK has two fields. The first field contains the max_TN, which is the maximum TN the device has received up to that point, and the second field contains f_TN, which is the first out-of-order TN the device has experienced. The controller then discards up to max_TN packets and an f_TN packet from its buffer and then retransmits packets from max_TN+1 to f_TN-1. In other words, the proposed S_ACK explicitly specifies which packets can be discarded from the controller's buffer and which packets need to be retransmitted. In order to satisfy the condition, saying at most one feedback out of INT packets, the device needs to reschedule its next C_ACK sending timing by considering the number of lost packets it has experienced in the current C_ACK sending period.

$$S_ACK(\max_TN, f_TN), 1 \leq \max_TN \leq f_TN \quad (2)$$

Figure 2 shows the operation of the proposed scheme when the controller sets max_B to 10 and the device sets the INT to 5, respectively. The device sends a single C_ACK for every 5 packets when all of 5 packets have been successfully received. As a single C_ACK for all sent packets is received, the controller can discard those packets from its buffer and transmit new packets, which are packet 6 through 10. So, the device monitors the packet reception status of the next 5 packets. Suppose packet 7 and 8 are lost or incorrectly received and packet 9 is successfully received. The device sends a S_ACK (6, 9), which indicates it has successfully received up to packet 6, but retransmission is required for packet 7 and 8. Since the S_ACK is triggered for packet 9, it also indicates the packet 9 was successfully received. After sending S_ACK, the device now reschedules its next C_ACK sending timing to packet 12, which is the next 5 packets. When the packet 7 and 8 are retransmitted, C_ACK (12) will allow the controller to discard the next five packets including packet 7, 8, and 10 through 12. That, the device initially sets its feedback sending timing to packet P sets to INT and adjusts it as follows.

$$P = \begin{cases} P + INT & , \text{ if C_ACK} \\ P + (f_TN - \max_TN + 1), & \text{ if S_ACK} \end{cases} \quad (3)$$

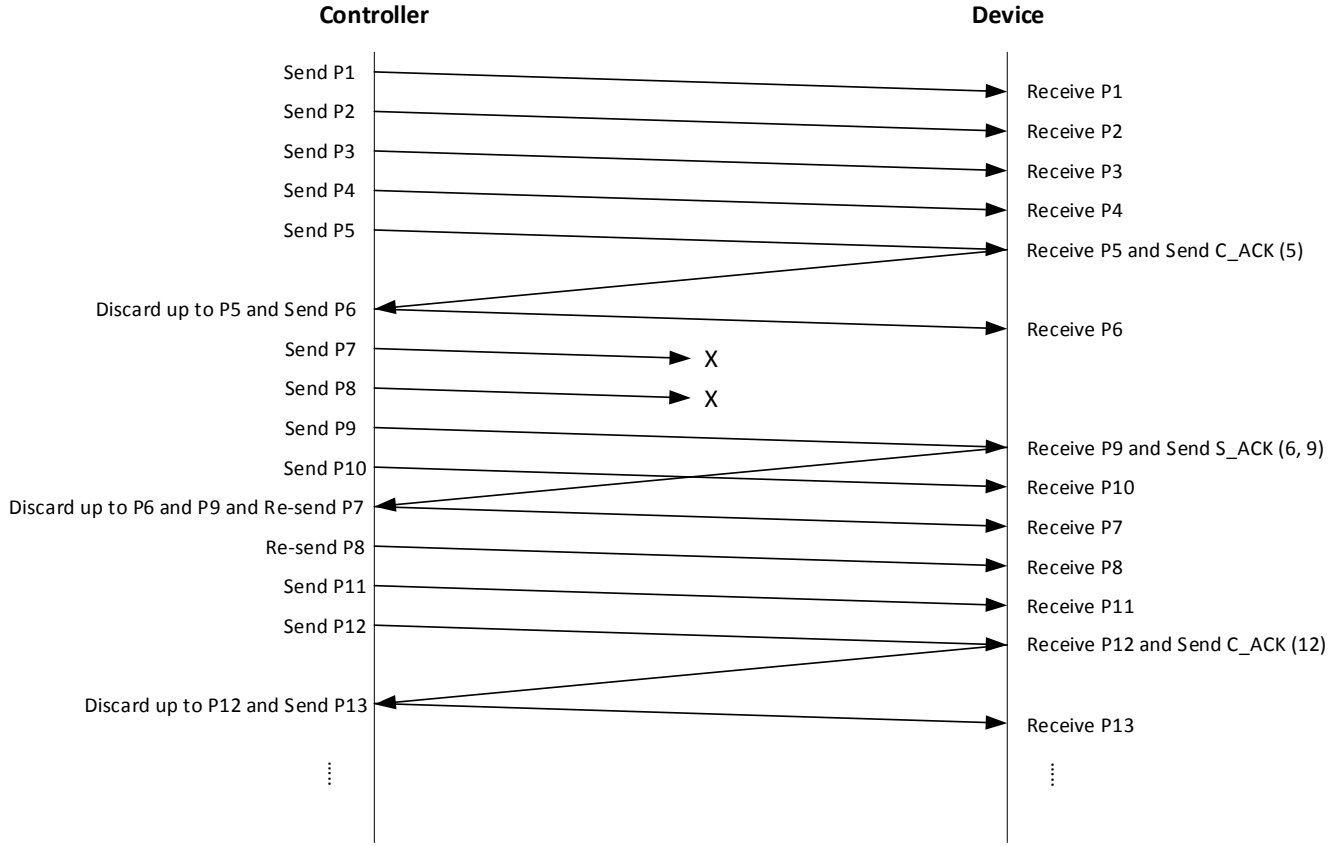


Figure 2. Operation of the proposed scheme with C_ACK and S_ACK

The proposed retransmission scheme has multiple advantages over existing schemes. First, reliable transmission is achieved with a significantly reduced number of feedback messages, which provides energy efficiency for both controller and its associated lightning devices. This advantage is more significant when we consider large scale DMX networks where response conflict among the multiple devices can be an important issue. Second, the controller does not need to maintain a logical ACK timer for each already sent packet because the receiver will send S_ACK if the packets are lost. Recall that the newly defined S_ACK functionally includes C_ACK. This alleviates computational overhead for the controller. Finally, the proposed scheme allows the controller to discard at least some packets whenever it receives any type of feedback from its device. This feature provides an efficient mechanism for buffer refreshment for the controller.

IV. PERFORMANCE DISCUSSIONS

In this section, we evaluate the performance of the proposed scheme. Our evaluation considers the fact that link errors remain, and are a major problem in DMX networks. Further, the packet losses can be correlated with previous transmission failure. Therefore, packet losses tend to occur in error bursts during the transmission process.

We consider each device having two states, namely state <1> meaning it has correctly received the last packet and state <0> meaning it has not correctly received that packet. Every

time a packet is sent to the device, the device will experience a transition that could either leave it in its current state or move it to another state. Since our concern is to current packet loss, we focus on the two transitions leading to state <0>, it includes <00> and <10>. These transitions follow Easton's model [7] defining r as a packet loss correlation factor and L as a packet loss probability. This model is given by

$$P_{00} = r + (1 - r)L, \quad P_{10} = (1 - r)L$$

where $r, L \leq 1$, and $L \ll r$. The steady-state probability P_0 of losing a given packet is given by

$$P_0 = P_0P_{00} + P_1P_{10} = P_0r + P_0(1 - r)L + (1 - P_0)(1 - r)L,$$

which simplifies into

$$P_0 = P_0r + (1 - r)L \quad \text{and} \quad P_0 = L \quad (4)$$

Hence, the L parameter represents the steady state probability for not receiving a correct packet. The r parameter affects the duration of error bursts. With $r = 0$ all packet losses are independent events. When r increases packet losses become more highly correlated. We show how that parameter can be estimated from the average duration of error bursts. The probability that an error burst will affect exactly k packets is given by

$$P(k \text{ lost packets per error burst}) = 1P_{01} + 2P_{00}P_{01} + 3P_{00}^2P_{01} + \dots \\ = \sum_{k=0}^{\infty} (k+1)P_{00}^kP_{01},$$

which is the mean of a geometric distribution. Hence the mean number of lost packets per error burst, EB_a , is given by

$$EB_a = \frac{1}{1 - P_{00}} = \frac{1}{P_{01}}$$

Most networks are fairly reliable and have $P_{00} \ll P_{01}$. In that case, $P_{00} \approx r$. The equation above can be rewritten as

$$EB_a \approx \frac{1}{1 - r} \quad (5)$$

Hence, $r = 0.8$ roughly corresponds to an average number of 5 lost packets per error burst.

Over a controller-to-device communication involves m packet transmissions, the proposed scheme will require each device to send at most $\lceil m/\text{INT} \rceil$ C_ACK packets when there is no packet loss. Otherwise, the number of packets sent to each device can be calculated in Equation (6).

$$\frac{mL}{EB_a} + \left\lceil m - \left\lceil \frac{(\text{INT} + EB_a)mL}{EB_a} \right\rceil \right\rceil / \text{INT} \quad (6)$$

The first term represents the number of S_ACKs while the second term represents the number of C_ACKs. Note that the existence of S_ACK packet will reschedule the C_ACK sending timing as defined in Equation (3). Under the same condition, it sends m S_ACK packets with the sequence-based scheme. For a numerical example, let us suppose the controller sends 100 packets ($m=100$) to a device setting its INT to 5 over the link having a loss probability ($L=0.1$) and an average length of the error burst ($EB_a=2$). With the proposed scheme the device will send 13 C_ACKs and 5 S_ACKs totaling 18 response packets while the controller sends 100 packets with the sequence-based schemes.

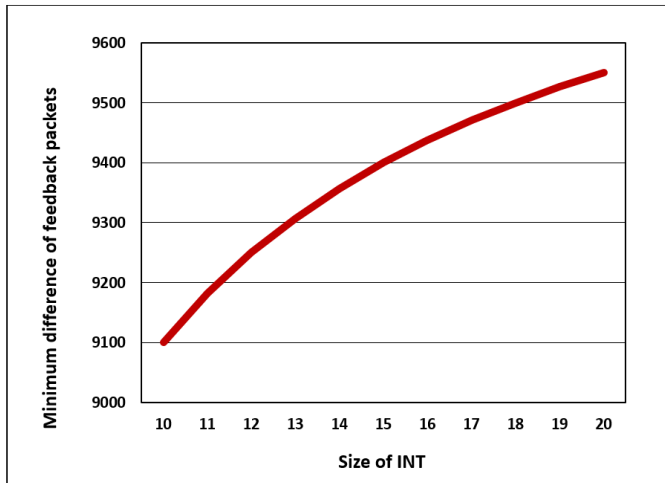


Figure 3. Minimum difference vs. different sizes of INT

Figure 3 shows the minimum difference in terms of the number of feedbacks sent by a device between the proposed scheme and sequence-based scheme with $m=10,000$, $L=0.1$, and $EB_a=5$ as a function of size of INT. With $\text{INT}=20$, the difference is 9550 feedbacks. Note that this difference is only for a device. If a DMX network covers N devices, the difference is N times larger. This significantly reduced level of

ACK implosion at the controller indicates the proposed scheme allows the implementation of a more scalable controller, since it will be able to serve more devices by handling the same number of feedback packets.

We need to mention that a lighting control network with the DMX-512A system is not usually based on a star network, with data cables radiating out to end points from an intermediate switching device such as router. Rather it appears more like an Ethernet environment where, the cabling runs from device to device in a daisy-chain fashion. This makes it difficult to implement an efficient random access protocol. The existence of simultaneously generated response packets by multiple devices in the same network domain increases the probability of data collision. This will introduce retransmission delay.

The proposed scheme also provides an efficient buffer management mechanism for the controller. That is, the buffer operates with its defined goal, saying the requested packets will always be available at the buffer for retransmission and the successfully received packets will be safely discarded from the buffer. This goal is achieved without requiring any extended buffer space.

V. CONCLUSION

We have proposed a new retransmission control scheme combining S_ACKs and infrequent C_ACKs to provide scalability and reliability for lightning control networks. S_ACKs are used to request packet retransmission and both types of ACK explicitly indicate which packets can be safely discarded from controller's buffer. The proposed scheme eventually reduces the retransmission delay, because (1) the requested packets are always available in the buffer and (2) the significantly reduced number of feedback packets sent by each device decreases the probability of a request conflict among the devices. It achieves this goal without requiring an extended buffer. More work is still needed to ascertain the optimal C_ACK sending interval for devices.

REFERENCES

- [1] ANSI E1.11 – 2008 (R2013), Entertainment Technology – SUITT DMX512-A, Asynchronous Serial Digital Data Transmission Standard for Controlling Lighting Equipment and Accessories.
- [2] ANSI E1.20 – 2010 Entertainment Technology RDM – Remote Device Management Over DMX512 Networks.
- [3] S-I. Choi, S. Lee, S-J. Koh, S-K. Lim, I. Kim, and T-G. Kang, "Reliable Transmission for Remote Device Management (RDM) Protocol for Lightning Control Networks," Lecture Notes in Electronic Engineering, Vol. 280, pp. 51–58, 2014.
- [4] J. Butler, "DMX versus Ethernet Infrastructures for Lightning Control Networks," <http://www.rdmprotocol.org> (last retrieved on November 2015).
- [5] M. G. Kwak, J. Baek, P. S. Fisher, "A Degree Constrained Spanning Tree-Based Device Discovery Scheme for Lightning Control Networks," *Proceedings of 5th International Conference on Internet of Things*, Oct. 2015.
- [6] S. Choi, S. Lee, S-J. Koh, S-K. Lim, I. Kim, and T-G. Kang, "Fast Device Discovery for Remote Device Management in Lightning Control Networks," *J. of Inf. Processing Systems*, 11(1):125–133, Mar. 2015.
- [7] M. C. Easton, "Model for database reference Strings Based on Behavior of Reference Clusters," *IBM Journal of Research and Development*, 22(2):197–202, March 1978.



Min Gyung Kwak received his B.S. degree in Computer Science from Winston-Salem State University (WSSU), Winston-Salem, NC in 2009, and his M.S. degree in Computer Science from Illinois Institute of Technology (IIT), Chicago, IL in 2012. He worked for Wake Forest Baptist Medical Center as an Intern. He was a member of the Network Protocol Research Group at the WSSU. He is currently working at the Hanhwa Techwin. His research interests include network

communication protocols, and network security protocols.



Jinsuk Baek (M'04) is Associate Professor of Computer Science at the Winston-Salem State University (WSSU), Winston-Salem, NC. He is the director of Network Protocols Group at the WSSU. He received his B.S. and M.S. degrees in Computer Science and Engineering from Hankuk University of Foreign Studies (HUFS), Korea, in 1996 and 1998, respectively and his Ph.D. in Computer Science from the University of Houston (UH) in 2004. Dr. Baek

was a post doctorate research associate of the Distributed Multimedia Research Group at the UH. He acted as a consulting expert on behalf of Apple Computer, Inc in connection with Rong and Gabello Law Firm which serves as legal counsel to Apple computer. He has served on an Editor of the KSII Transactions on Internet and Information Systems. He also served or currently serving as a reviewer and Technical Program Committee for many important Journals/Conferences/Symposiums/Workshop in Computer Communications Networks area. His research interests include wireless sensor networks, scalable reliable communication protocols, mobile computing, network security protocols, proxy caching systems, and formal verification of communication protocols. He is a member of the IEEE.



Paul S. Fisher is R. J. Reynolds Distinguished Professor of Computer Science at the Winston-Salem State University (WSSU), Winston-Salem, NC. He is the director of High Performance Computing Group at the WSSU. He received his B.A. and M.A. degrees in Mathematics from University of Utah and his Ph.D. in Computer Science from Arizona State University. He has written and managed more than 100 proposal efforts for corporations and DoD involving teams of 1

to 15 people. He worked as consultant to the U.S Army, U.S Navy, U.S Air Force and several companies over the years. In the 1990's he commercialized an SBIR funded effort and built Lightning Strike, a wavelet compression codec, then sold the company to return to academe. His current research interests include wired/wireless communication protocols, image processing and pattern recognition.