

Container over FC-AE-ASM: A Method for Mixed Data Transmission in Avionics Systems

Ding XU*, Tong WANG*, Qiao LI*

*School of Electronics & Information Engineering, Beihang University, No. 37 Xueyuan Road Beijing, China

xuding.bj@gmail.com, tongwang@buaa.edu.cn, liqbuaa@ee.buaa.edu.cn

Abstract— Existing Fibre channel (FC) network interface cards usually support either FC-AE-ASM or ADVB upper protocol. However, different upper protocols are used in different specific application environments. FC-AE-ASM and ADVB are respectively designed for transmitting commands and video/audio stream. Aiming to this, we present a light-weighted solution. A simplified container defined in ADVB is introduced into FC-AE-ASM and a modified ASM header is used to differentiate video and commands. Therefore, this container-based FC-AE-ASM protocol achieves a simultaneous and flexible transmission of commands along with video stream. Compared with FC-AE-ASM without container, simulation results show that proposed scheme nearly keep same frame error rate with the growing of video resolution, even though it also leads to a slight increase on the overhead rate at the same time.

Keywords— Avionics Networks, Fibre Channel, Light-weighted Protocol, Container, Mixed Transmission

I. INTRODUCTION

The high speed and low latency of Fibre Channel (FC) make it a candidate interconnection protocol in Avionics environment. Several subsets are provided in FC-4 layer to satisfy the needs of different application environments, such as Fibre Channel Anonymous Synchronous Messaging (FC-AE-ASM) and Avionics Digital Video Bus (ADVB). FC-AE-ASM is mainly used to transmit avionic commands; and ADVB defines a video interface system between video capture and display. ADVB was originally designed to point to point video transmission. However, ADVB is a simplification of Fibre Channel Audio Video (FC-AV) protocol aiming to Avionics environments. Its nucleus lies in the container system, which utilizes the simple mode of container in FC-AV [1]. Since the container in FC-AV could be used in switched networks, therefore, this container system could be introduced into FC-AE-ASM to control video stream transmission in both point to point and switched networks.

Several researches have been conducted on the implementation of FC interface card [2][3][4]. But they mainly focus on the design and implementation of FC-0 and FC-1 layer. There is little research on FC-4 layer protocol design. Yajun Xu proposed a light-weighted IP over FC protocol [5], but it is used to test the performance of FC and mention no transmission mechanism about video stream

transmission control. Moreover, existing FC network interface cards usually support either standard FC-AE-ASM or ADVB upper protocol. While in IMA avionics systems, nodes need to send commands and periodical data together with video stream. If ADVB and FC-AE-ASM could be integrated into one system, then high rate video and commands could be effectively transmitted in the same network under certain specific FC-4 protocol.

In this paper, a light-weighted container over FC-AE-ASM (CoASM) solution is proposed. We introduce the container from ADVB into FC-AE-ASM and keep command transmission remaining the same with FC-AE-ASM. Meanwhile, a video stream is transmitted as a set of containers. A container, namely a video frame in ADVB, represents an ASM message in FC-AE-ASM. It sends a sequence of FC frames with the same ASM header in front of them. Then this container will be reassembled in recipient as an ASM message. This container-based FC-AE-ASM protocol achieves a flexible mixed transmission of commands and video stream. By adopting CoASM, it greatly reduces frame error rate even though there is a slight increase on overhead rate when transmitting a video stream.

The rest of this paper is organized as follows. Section II gives a brief introduction on FC frame structure, FC-AE-ASM protocol and container system. Section III describes the CoASM scheme and section IV puts forward an implementation of CoASM proposal. Section V discusses the performance of CoASM compared to FC-AE-ASM without container. Finally, section VI ends this paper with some conclusions.

II. ANALYSIS OF FC UPPER PROTOCOL

A. FC Frame Structure

A FC frame is made up of a SOF ordered set, 0 or more extended header, a frame header, payload, CRC and an EOF ordered set, which is shown in Figure 1. SOF and EOF respectively represent the beginning and ending of a FC frame. Extended headers, if presented, are used to extend the functionality provided by the frame header. Moreover, a frame header consists of 24 bytes which filled routing control, class specific control, frame control, destination ID, source ID, etc. The payload is a variable field with the maximum size of 2112

bytes, filling the content data of a FC frame. Besides, CRC is a 4 bytes field used to verify the data integrity of a FC frame except for SOF and EOF fields. Additionally, there must be at least six primitive signals, such as Idles, between each EOF and the next SOF [6].

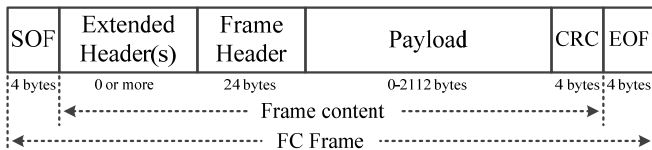


Figure 1. FC frame structure

B. FC-AE-ASM Protocol

FC-AE-ASM is a low-latency FC-4 layer communication protocol designed for avionic command, control, and signal processing usage [7]. FC-AE-ASM protocol defines some necessary functions offered in FC-2 layer in order to provide corresponding ASM service in FC-4 layer.

Every message in FC-AE-ASM comes from a single unidirectional exchange sequence. The protocol defines that a recipient may expect the message to arrive at a predetermined rate and does not know where the message is originating [7]. If an ASM message exceeds the payload limit of a single FC frame, multiple frames will be used for transmission, and these frames will be reassembled based on Message ID field in ASM header. Moreover, FC-AE-ASM mainly uses Class 3 service which is a connectionless service without any notification of non-delivery. In this way, frames could be routed in different link to arrive at destination and decrease processing overhead brought by notification, which contributes to improve the utilization ratio in FC networks.

FC-AE-ASM also defines the header format of message. The first four words of payload of each frame are reserved for the FC-AE-ASM header. In multi-frame ASM messages, all FC frames should contain a copy of this header [7]. The format of the FC-AE-ASM header is shown in Table 1.

TABLE 1. FC-AE-ASM HEADER FORMAT

bits word	31 - 24	23 - 16	15 - 08	07 - 00
0 (0-3)	Message ID			
1 (4-7)	Vendor Specific - Security			
2 (8-11)	Reserved			
3 (12-15)	L	Priority	Message Payload Length (Bytes)	

Bytes 0-3 are a 32-bit Message ID that uniquely identifies the message in system. Particularly, the ID value of hex '00 00 00 00' and 'FF FF FF FF' are reserved. Bytes 4-7 are reserved for security information and bytes 8-11 are reserved for further use. The L bit in byte 12 is valid only when the Message Payload Length is hex '00 00 00 00'. If L bit and payload length are both zero, then the message has a payload length of 16,777,216 bytes. While if L bit is 1 and payload is 0, the message payload length is 0 byte. Moreover, priority field could be divided into two parts: bits 30-25 of word 3 are filled

with priority and bit 24 is the preemption request flag. If it is 0, then this message could not be preempted and vice versa. This field is identical with the CS_CTL field in FC frame header and shall not be modified in a single FC sequence. Additionally, bytes 13-15 are used to fill the message payload length which specifies the total number of bytes of the entire ASM message. And this field should exclude the length of all ASM headers [7].

C. Container System

A Container consists of a set of FC frames to transmit video and audio. It is made up of a header and objects. Objects are defined as certain types of data with one or multiple FC frames. There are four types of objects. Object 0 refers to some ancillary data related to the video. Object 1, if used, represents audio data. Object 2 and Object 3 indicate video data. For interlaced video, Object 2 is used for odd field payload and Object 3 is used for even field payload [8].

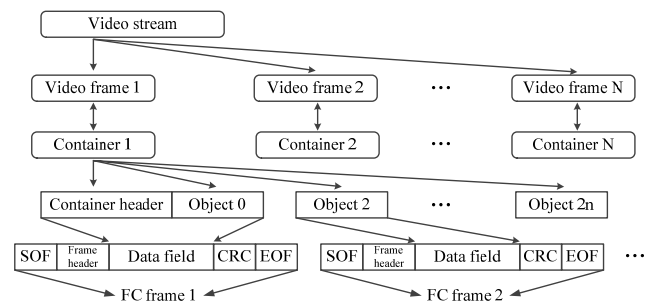


Figure 2. Relationship of FC frame sequence with progressive video

In most cases, a container, which is also a FC sequence in FC-2, corresponds to a single video frame. Also, there are many ways to control the timing of video transfer. Line Segmented and buffered mode is a typical way. At the beginning of a container, a 22 words container header and object 0 is encapsulated in a FC frame for transmission. A container header includes container count, video frame rate, offsets of objects within the container; and object 0 defines some more specific information such as number of rows/columns, format and color information of the transmitted video and so on. Then this FC frame is followed by a set of object 1, object 2 and 3 to transmit audio and video data. It is worth mentioning that, in most cases, a line or a column of video exactly maps into a single Object 2 or Object 3 in line segmented situation. However, if data size of a single line exceeds the payload limit of FC frame, half line data will be encapsulated as an object for transmission in general. Therefore, multiple FC frames will be used in a container to transmit a video frame. For example, the video resolution of an SXGA display is 1280 × 1024. The data size of each line with 24 bits/pixel is 3840 bytes which exceeds the limit of 2112 bytes. Then half a line with 1920 bytes data will be encapsulated in a frame. Figure 2 shows the relationship of FC frame sequence with a progressive video requiring only Object 2.

III. CONTAINER OVER FC-AE-ASM SCHEME

Considering the FC-AE-ASM design for command purpose and the effects of container on controlling video transmitting, a simplified container defined in ADVB is introduced into FC-AE-ASM. A modified ASM header is used to differentiate video and commands data. Command transmission keeps the same with FC-AE-ASM. A video stream is transmitted by containers as ASM messages and it could be reassembled according to container header of each video frame, namely ASM message. We neglect audio data and only consider progressive video with Object 2. This container over FC-AE-ASM (CoASM) scheme is not a replacement for conventional FC-AE-ASM but is a simplified commands and video mixed transmission proposal over Fibre Channel. We will elaborate the modified ASM header and procedures of command and video stream in the following parts of this section.

A. ASM Header Transformation

Since message ID is unique in the whole system, we use it as an identifier to differentiate commands and video stream. As for video stream, data will be transmitted as the form of containers. Table 2 shows the modified ASM header format of CoASM.

TABLE 2. COASM ASM HEADER FORMAT

bits word	31 - 24	23 - 16	15 - 08	07 - 00
0 (0-3)	Domain ID	Node ID	Message No.	type
1 (4-7)	Vendor Specific - Security			
2 (8-11)	Message deadline			Criticalness
3 (12-15)	L	Priority	Message Payload Length (Bytes)	

Bytes 0-3 are still a 32-bit message ID but it is divided into 4 parts. Domain ID and Node ID are used to manage the network distribution so that each node could send message with unique message ID. Also, Message No. is used to configure local message number. Particularly, bit 0 of byte 3 is type field used as an identifier to differentiate command and video. A message will be considered as a command data if type field is 0. Likewise, a message will be viewed as a video stream if type field is 1. In this way, considering message ID hex '00 00 00 00' and 'FF FF FF FF' are reserved, domain ID with hex '00' and 'FF' are reserved. Then a network could have at most 254 domains with at most 256 nodes in each domain; and each node could configure 32768 messages. Every node has some unique message ID used for transmission; and it is also convenient for FC-2 layer to identify whether a message is a command or a video stream.

Bytes 8-11, if used, are for priority scheduling purpose. The former three bytes are filled with message deadline and criticalness is encapsulated in the last byte. A criticalness-deadline first scheduling could be achieved through this field.

Bytes 13-15 represent message payload length. As for command data, the length stands for the size of command content; as for video stream, this length refers to the size of each video frame.

In addition, since we only take progressive video with Object 2 into account, thus we simplify the container header from 22 words to 14 words, omitting the words of Object 1 and Object 3 in container header. Moreover, Figure 3 gives the modified map structure of FC frame sequence with progressive video. Since video stream is transmitted as ASM messages, corresponding ASM header should be added in the beginning of every FC frame data field.

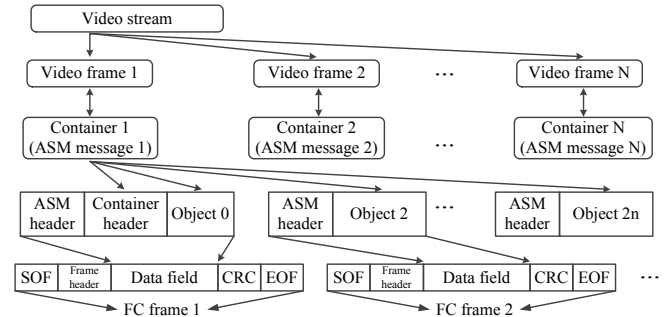


Figure 3. Modified map structure

B. CoASM Processing Procedures

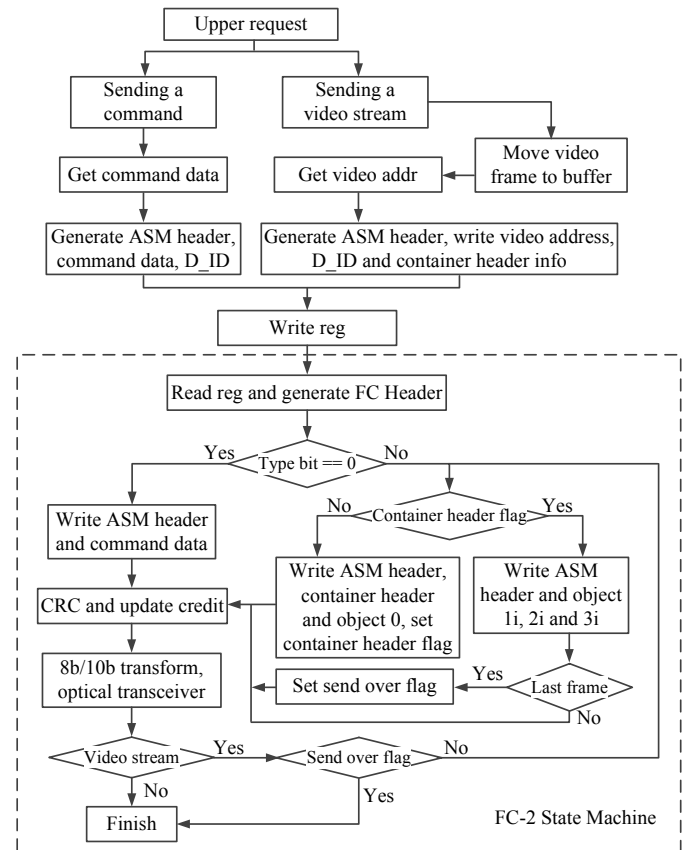


Figure 4. Sending flow chart

Figure 4 shows a flow chart of sending messages. When a command is to be transmitted, FC-4 layer firstly gets command data and some necessary FC header information, then generates corresponding ASM header with type field

filling 0. Then the information is written to a register recording commands information, including FC header, ASM header and command data. Accordingly, FC-2 state machine firstly generates FC header, followed by ASM header and command data, then encapsulates CRC, SOF and EOF. In the end, FC-2 modifies port credit and sends FC frame to FC-1 and FC-0 for 8b/10b transform and optical transmission.

When transmitting a video stream, video will be transmitted frame by frame as containers. FC-4 layer firstly move a video frame into a buffer and get its address; then FC header information, ASM header with type bit filling 1, container header and video address are generated and written into a register saving video information. FC-2 state machine then encapsulates a container header and Object 0 together with ASM header into a FC frame at the beginning. Then FC frames consisting of a FC header, an ASM header and an Object 2 with a line of video frame, will be sent frame by frame until the last FC frame of a container is sent over. Also, port credit will decrease very time a FC frame is sent over.

Particularly, since generally a video ASM message requires multiple FC frames, the FC-2 state machine will check the content of command register immediately after a FC frame of a container is sent over. If a new command is required at that time, FC-2 state machine will turn to transmit command and then continue to transmit the container.

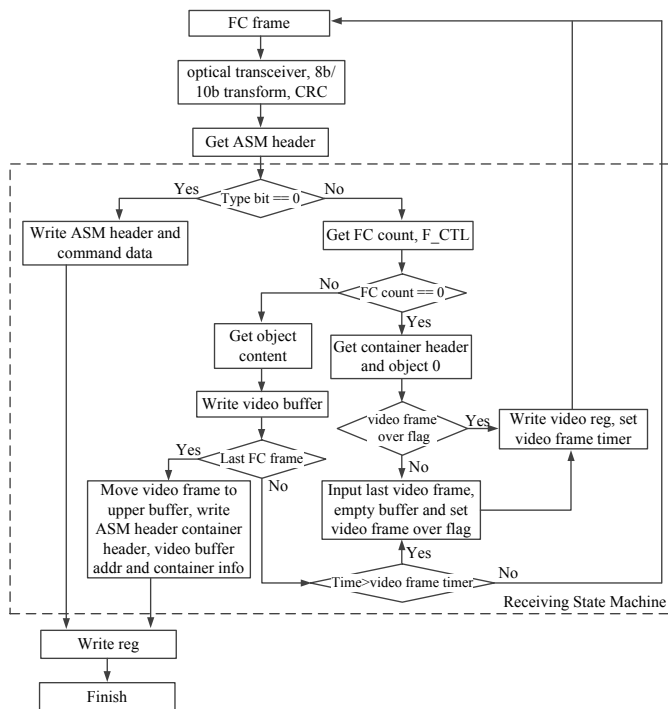


Figure 5. Receiving flow chart

Similar to sending messages, Figure 5 indicates the flow chart of receiving a modified ASM message. When transceiver receives a FC frame, FC-2 will get its ASM header at first. If type bit is 0, namely command message, FC-2 writes the ASM header and command into a register saving command information. Then FC-4 reassembles the message

according to message ID and message payload length in ASM header. Likewise, if type bit is 1, which represents a video stream message, FC-2 state machine will get Seq_ID and Seq_count. If seq_count is 0, then this frame is a container header frame. Then FC-2 state machine will save container header and object 0 into a register and set a timer, which is used to ensure the video frame timing. When next FC frame of the container arrives, video content filled as Object 2 will be removed into a buffer. At last, after receiving the last frame of a container, identified by the F_CTL field of FC header, FC-2 state machine will move the video frame to upper buffer and write ASM header, container header and video address to a register saving video information to inform FC-4. FC-4 will reassemble a video stream according to the Message ID and container count in container header.

IV. IMPLEMENTATION OF COASM SCHEME

Based on the analysis above, we propose an implementation of CoASM scheme, which is shown in Figure 6, to achieve a light-weighted mixed transmission of commands and video stream. In this system, three levels are divided: FC4 layer, FC2 layer and FC1/FC0 layer.

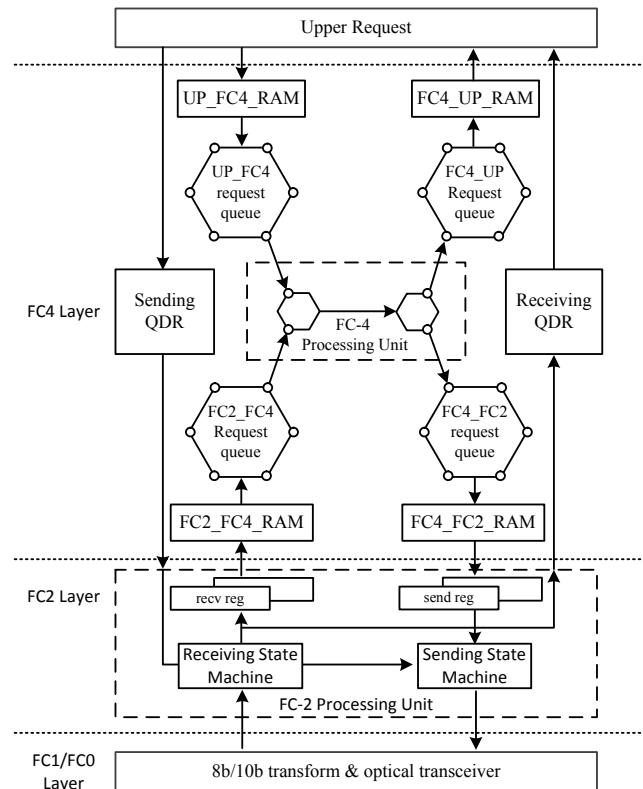


Figure 6. CoASM system implementation diagram

FC4 consists of four queues, four RAMs, one FC-4 processing unit and two QDRs which are used to saving contents of commands and video stream. The first word in these RAMs is a Flag area representing the update of data, followed by specific RAM content. Besides, those four queues, UP_FC4, FC4_FC2, FC4_UP and FC2_FC4, respectively

used to cache user message sending request, ASM message sending request, user message receiving request and ASM message receiving request.

The nucleus of FC4 lies in the FC4 processing unit. It inquiries flag areas of UP_FC4_RAM and FC2_FC4_RAM at first. If new requests have arrived, the data in the RAMs will be pushed into corresponding queues. Next it operates the head request of UP_FC4 queue. As for video stream transmission, processing unit will divide video into containers and produce multiple new requests pushed into FC4_FC2 queue for further processing, with each request including an ASM header and some ancillary information such as container header and Object 0. Then, FC4 processing unit analyzes the information of the head of FC2_FC4 queue to generate a user request to the tail of FC4_UP queue informing the user. Finally, FC4 processing unit will write the head of FC4_UP and FC4_FC2 queues into FC4_UP_RAM and FC4_FC2_RAM to complete current operating cycle.

Moreover, FC2 is made up of a sending state machine, a receiving state machine FC2, a dual-buffered sending register (we define them as reg_send_A and reg_send_B) and a dual-buffered receiving register (defined as reg_rcv_A and reg_rcv_B). To begin with, the content of FC4_FC2_RAM will be copied into reg_send_A based on the flag area of FC4_FC2_RAM. Then FC2 will get the type bit of ASM header to enter the right state of sending state machine. Corresponding FC frames will be sent based on the interface information of FC4_FC2_RAM and QDR. During this time, reg_send_B will copy the content of FC4_FC2_RAM. When the ASM message in reg_send_A is sent over, FC2 will automatically turn into reg_send_B for next message's transmission. The dual-buffer register reduces the access time of FC4_FC2_RAM and contributes to improve the processing efficiency of node through this mechanism.

Meanwhile, if a FC frame arrives, it will enter the receiving state machine. FC2 gets the type bit in ASM header to receive commands or video frames. FC header, ASM header and some ancillary information will be saved in reg_rcv_A and the content of ASM message will be recorded in QDR, which is also accessible for upper users. In this way, it reduces reading times of data to be received by users, and helps to facilitate receiving processing time. When a message receives completely, state machine returns to IDLE state to wait for the next FC frame and saving information into reg_rcv_B. At the same time, data in reg_rcv_A will move into FC2_FC4_RAM for storage.

Sending and receiving operations in FC2 processing unit have their own registers and QDRs, so that they are mutually independent and could be operated in the meantime.

V. SIMULATION EXPERIMENTS

The analysis and implementation of CoASM is described above in theory. A series of simulations are carried out to evaluate its performance as follows.

A. Overhead Rate

Overheads are some ancillary information which is used to make sure the effective transmission of data. To be common, we define overhead rate as the ratio of overhead data to the sum of data to be transmission and overhead data.

When transferring commands, CoASM and FC-AE-ASM have the same overheads, which consist of FC header, ASM header and CRC. However, when transmitting a video stream, as for FC-AE-ASM system without container, overheads are composed of FC header, ASM header and CRC. While for CoASM, overheads are made up of Container headers and Object 0 besides FC header, ASM header and CRC. Particularly, the size of Object 2 relate to the line size of video. Thus overheads of CoASM are relevant to video resolution.

In order to facilitate the calculation, we assume that a progressively scanned video stream is to be transmitted with video resolution $N \times N$. Each pixel is comprised of 24 bits, respectively representing three 8 bit sub-pixels R, G, and B. Figure 7 shows the overhead rate of CoASM and FC-AE-ASM without container regarding video resolutions.

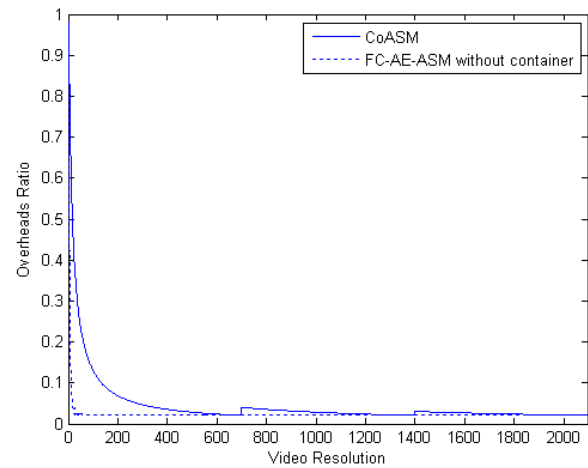


Figure 7. Overhead rate regarding video resolution

From the figure we could see that CoASM has two singular points which divide the curve into three sections. Because of the limit of payload size of FC frame, a line of video frame is separated into two FC frames when $698 \leq N \leq 1397$ and three FC frames when $1398 \leq N \leq 2096$. Also, with the growing of video resolution, the difference of overhead rate between CoASM and FC-AE-ASM without container decreases in each section. Moreover, when the payload of Object 2 in a container equals to the maximum size of FC frame, its overhead rate equals to the situation of FC-AE-ASM without container.

B. Frame Error Rate

When transmitting a video stream, container header, Object 0 and each Object 2 in CoASM consist of effective sequence and CRC information to control the assembly of video. Considering the bit error, any lines that have a CRC failure should be discarded. Replacing them with "black" is a simple strategy. Then video frames could still be deciphered based on container ID and each Object 2. Therefore, they still could be

recognized as received. While if there is bit error in those ancillary information, such as container header and Object 0, it may result the decipherment failure of a video frame.

However, in FC-AE-ASM without container situation, video frames are sent as ASM messages without any ancillary controlling information. An ASM message is separated into multiple FC frames due to the payload limit. In this way, if some FC frames are discarded because of CRC failure, it may directly result the decipherment error of a whole video frame in recipient.

Assume the bit error rate of FC is 10^{-12} [9] and bits are independent with each other. Figure 8 indicates the frame error rate of CoASM and FC-AE-ASM without container regarding video resolution. With the growing of video resolution, more FC frames are used to send a container, namely a video frame. This leads to a slow increase of frame error rate under CoASM situation. But the growing of video resolution leads to more data bits; and therefore results a rapid increase of frame error rate under FC-AE-ASM without container situation. From Figure 8 we could see that frame error rate of CoASM is substantially linearly proportional to the video resolution while frame error rate of FC-AE-ASM without container is roughly proportional to the square of video resolution. Thus CoASM could effectively reduce the frame error rate considering its container mechanism.

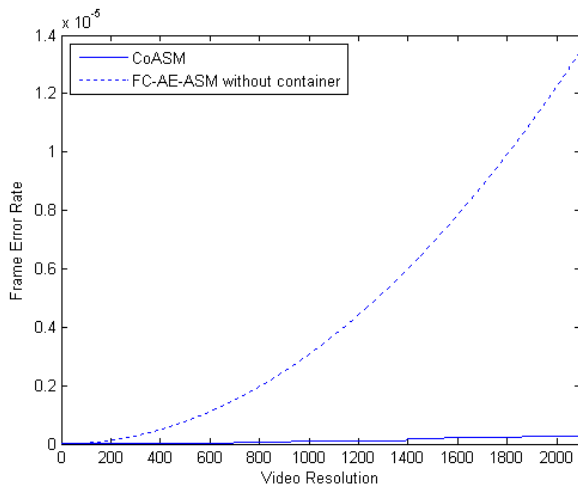


Figure 8. Frame error rate regarding video resolution

VI. CONCLUSIONS

In this paper, we presented a simplified container over FC-AE-ASM (CoASM) scheme to achieve command and video stream mixed transmission based on FC-AE-ASM protocol. A modified ASM header is given to identify the type of message. Then a sending and receiving flow chart are proposed, which define some constraints to FC-4 and FC-2. As for command transmission, it keeps the same with FC-AE-ASM; and as for video stream transmission, a simplified container system is introduced from ADVB into FC-AE-ASM used to transmit video ancillary information and reassemble video frames. Then we provide an implementation on this container over FC-AE-ASM and followed by some simulation calculations to analyse the performance of CoASM.

By adopting CoASM, we achieve a light-weighted method on mixed data transmission on one specific upper protocol. The simulation results indicate that although the overhead rate increased a little, however, it greatly reduces frame error rate with the growing of video resolution and facilitates the decipherment of video stream. In the future, more work can be done to apply CoASM to transmit video stream in switched FC networks.

REFERENCES

- [1] Tim Keller, Jon Alexander, "Extending ARINC 818: The Development of An ARINC 818 Switch Architecture," in Proc. DASC'10, 2010, p 5.A.4-1 - 5.A.4-9.
- [2] Zhonglou Meng, Xiaodong Tu, Jun Xie, "Design and Implementation of Fiber Channel Interface Card on Avionics Environment Networks," in Proc. ICCP'10, 2010, p. 399 - 401.
- [3] Yu Zhang, Dan Feng, Wei Tong, Jingning Liu, "Implementation of FC-1 and FC-2 Layer for Multi-Gigabit Fibre Channel Transport," in Proc. FGNC'07, 2007, p 203 - 207.
- [4] Bei Liu, Yuejian Fang, Zhonghai Wu, "The research and implementation of device login in FC2 of Fiber Channel," in Proc. ICNDC'10, p60-64
- [5] Yajun Xu, Xiaolin Zhang, Caijian Guo, Huagang Xiong, "One lightweight internet protocol over fibre channel" Journal of Beijing University of Aeronautics and Astronautics, vol. 32, pp. 1246-1249+1254, Oct. 2006.
- [6] Fibre Channel framing and signaling-2 (FC-FS-2), 2006
- [7] Fibre Channel Avionics Environment- Anonymous subscriber messaging (FC-AE-ASM), 2008
- [8] Jon Alexander, Tim Keller, "Using ARINC 818 Avionics Digital Video Bus (ADVB) for Military Displays," in Proc. SPIE'07, 2007
- [9] Fibre Channel electrical interface, Wikipedia [Online]. Available: http://en.wikipedia.org/wiki/Fibre_Channel_electrical_interface