Considerations for Isochronous Data Services over the Proximity-1 Space Link

Jay L. Gao*

Jet Propulsion Laboratory, Pasadena, CA 91109-8099

Future mission concepts for robotic and human explorations will involve a high level of real time control/monitoring operations such as tele-operation for spacecraft rendezvous and surface mobile platforms carrying life-support equipments. The timely dissemination of voice, command, and real-time telemetry for monitoring and coordination purposes is critical for mission success. It is envisioned that future missions will require a network infrastructure capable of supporting isochronous data services. The CCSDS Proximity-1 Space Link Protocol[1] could be used to provide isochronous service over the surface-to-Earth relay as well as "beyond-the-horizon" communications between distant Lunar or Mars surface elements. This paper will analyze the latency, jitter, and throughput performance of the Proximity-1 protocol for isochronous applications. In particular we will focus on constrained scenarios where the protocol operates in full-duplex mode, carrying isochronous traffic in one direction and error-controlled traffic in the other direction. We analyze the impact of the strict priority scheme in Proximity-1 on delay jitter and the impact of the isochronous traffic on the efficiency of the reliable data transfer in the other direction, and discuss methods for performance optimization. In general, jitter performance is driving by relative loading of isochronous traffic on the forward link compared to the acknowledgement traffic. Under light loading condition, the upper-bound of the delay jitter is the transmission duration of an acknowledgement frame on the forward link; for higher loading scenarios, the maximum jitter is scaled up by the inverse of the residual bandwidth, i.e., the spare capacity available in the forward link to carry isochronous traffic.

Nomenclature

ACK = acknowledgement for automatic repeat request
ARQ = automatic repeat request
CCSDS = Consultative Committee for Space Data System
COP-P = Command Operation Procedure – Proximity
EVA = Extravehicular Activity
FEC = Forward Error Correction
FIFO = First-In First-Out
Forward link = typically indicates out-going transmission from Earth to remote space assets; when using an orbiter for relay communications, the orbiter-to-remote spacecraft direction is the forward direction.
Return link = typically indicates in-coming transmission from remote space asset to Earth; when using an orbiter for relay communications, the remote spacecraft-to-orbiter direction is the return direction.
LSAM = Lunar Surface Access Module
Need_ACK = Boolean variable indicating need to update ARQ state to the sender
PLCW = Proximity Link Control Word - the data unit that carries acknowledgement/report of ARQ states
PLTU = Proximity Link Transmission Unit – carries data frame and the associated synchronization marker and parity bits
TDM = Time Division Multiplexing
T_{data_frame} = transmission time of a PLTU on the forward link (isochronous traffic)
λ_{data_frame} = data frame arrival rate of on the forward link (isochronous traffic)

* Technical Staff, Communications Networks Group, Telecommunications Architecture and Research Section, Jet Propulsion Laboratory, 4800 Oak Grove Drive, Mail Stop: 238-343, Email: Jay.L.Gao@jpl.nasa.gov.

ρ_{data} = loading of the isochronous traffic normalized to the capacity of the forward link
ρ_{Ack} = loading of the acknowledgement traffic normalized to the capacity of the forward link
R_{return} = data rate of the return link
$R_{forward}$ = data rate of the forward link
N_{PLTU_ret} = number of bits in a return link PLTU
N_{PLTU_for} = number of bits in a forward link PLTU
N_{PLCW} = number of bits in a PLCW
T_{Ack} = transmission time of a PLCW on the forward link
T_{Need_Ack} = periodicity of the Need_ACK state being set to "True"
N = the minimum window size of the Proximity-1's COP-P retransmission procedure that maximize efficiency of the ARQ when there are no ACK losses and gaps
n = size of ACK frame gaps
η_{ARQ} = throughput of the return link, normalized to the error-free throughput of the ARQ algorithm
$T_{round\text{-}trip}$ = round-trip time of the proximity-1 space link that incorporates propagation, transmission, coding, and process delays
$T_{def}(i)$ = the delay experienced by the i-th ACK frame, from the moment the latest Need_ACK state is set to *True*, as a result of blocking by another ACK frame; when i=1, the delay is due to blocking by a data frame
D_{max_jitter} = the maximum jitter experienced by a data frame as a result of channel access contention with the ACK frames

I. Introduction

THE focus of this study is to analyze the performance of using the CCSDS Proximity-1 Space Link Protocol[1] to provide isochronous services such as voice, video, and command for tele-operations of networked space assets. In particular, we examine the constrained scenario where isochronous traffic were intermixed with Proximity-1 acknowledgement frames when reliable data transfer occurs on the reverse direction of the link. For example pre-cursor lunar exploration may involve tele-operations of robotic assets on the surface of the moon from Earth; later phases of the campaign will involve astronauts monitoring and controlling of space assets, while reliable data were also being transported back to the astronaut and mission control through the network.

II. Potential Scenarios for Isochronous Services

As the utility of the CCSDS Proximity-1 Space Link protocol has proven itself in the success of the Mars Exploration Rover, the concept of relay communication will soon be extended to the arena of human exploration of the Moon and eventually to Mars. With the envisioned throughput for lunar exploration being very high, relay communications will again be the key to keeping the link budget at a reasonable level while meeting the system requirement. With the propagation delay between Earth and the Moon at about 1.5 seconds, tele-operation of lunar surface elements is a possibility, particularly for the early pre-cursor robotic missions. Also Proximity-1 protocol facilitates over-the-horizon tele-operation of lunar surface assets by an Extra Vehicular Activity (EVA) astronaut.

In tele-operation, isochronous services is highly desirable on the forward direction (from the controller to the remote asset) to ensure low jitter, low latency delivery of control directives; usually reliability is best provided through a combination of forward error correction (FEC) and sufficient link margin. In the return direction, the proximity protocol will most likely carry a mixture of low rate motion-imagery, real-time status information, and science telemetry that may require link layer reliability.

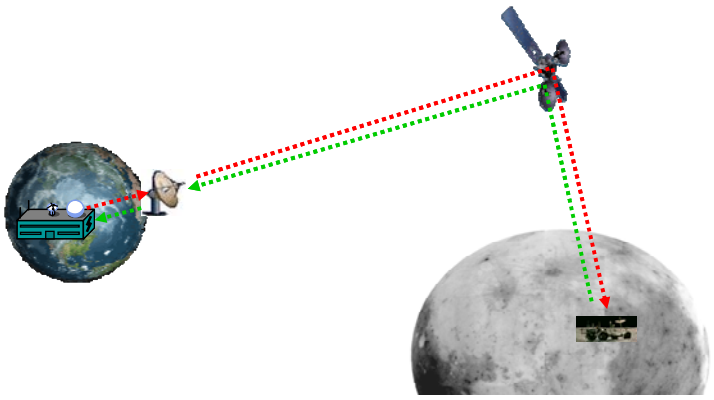

Figure 1: Tele-Operation of Lunar Asset

Figure 1 shows a potential pre-cursor mission scenario where a rover on the lunar surface is tele-operated from mission control via a relay orbiter. The relay orbiter improves the link capacity of the rover as well as providing capability of operating on the "far-side" of the moon. Here the Proximity-1 protocol will operate over the orbiter-rover link. An extension of the scenario, as depicted in Figure 2, may involve additional lunar elements; say EVA astronaut and Lunar Surface Access Module (LSAM), which are "cc-ed" on the return telemetry for situational awareness.

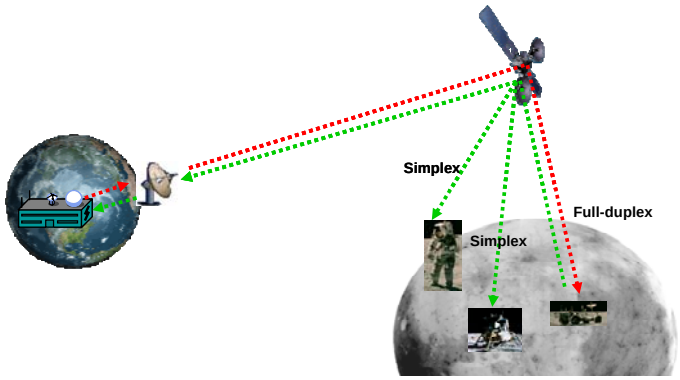

Figure 2: Tele-operation with EVA and LSAM monitoring progress

As shown in Figure 2, the CCSDS Proximity-1 Space Link protocol is simultaneously transmitting control directives to the rover as well as relaying return telemetry to the EVA astronaut and LSAM. The nominal operational scenario for Proximity-1 protocol is point-to-point communication. However, the framing structure allows for one-to-many transmission in a Time-Division Multiplexing (TDM) fashion with operational adaptation. Specifically the transmitting side (the orbiter), can mark each frame with the proper spacecraft ID. On the receiving end, the physical layer handshaking and synchronization is completed between the relay orbiter and the rover, while the EVA astronaut and LSAM remain in simplex mode and listen to the orbiter's data stream and pull out frames addressed to them. This requires no change to the protocol but it is critical that the operations be coordinated so that only one surface asset radiates energy on the return link; otherwise the interference will cause the handshaking process (hailing) to fail.

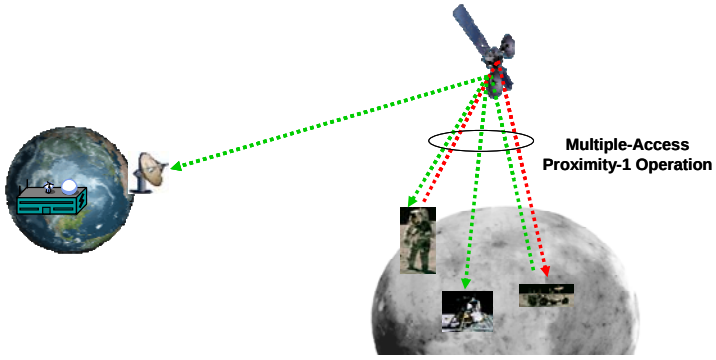

Figure 3: Over-the-horizon tele-operation over the lunar surface
(multiple access proximity-1 operation required)

Figure 3 shows another variation of the remote tele-operation scenario where the controller is a lunar surface EVA astronaut. The orbiter and the Proximity-1 protocol provide the over-the-horizon range extension so that the astronaut can operate the rover without line-of-sight communication. LSAM and mission control are both monitoring the return telemetry from the rover. In this case, the proximity-1 protocol is required to operate in full multiple-access mode, transmitting to and receiving from multiple assets concurrently. Since this scenario depends on future extensions/modifications to the current standard, we will not consider it in our study. It should be noted that the envisioned usage of proximity-1 to provide isochronous services is not limited to the tele-operation scenario. Voice traffic could also be benefited by very low latency and jitter link layer. The choice of these particular scenarios is motivated by their near term applicability to pre-cursor lunar exploration missions. However, most importantly, we examine this type of scenarios because the combination of isochronous forward link traffic and reliable return link traffic, i.e., using automatic repeat request (ARQ), presents a stressed case for the protocol. By analyzing this scenario, we understand the performance characteristics of proximity-1 protocol, how to make it work, and what modifications, if any, will be needed.

III. Prioritization Method in Proximity-1 Protocol

The CCSDS Proximity-1 Space Link protocol provides a point-to-point transfer of data frames in two modes; expedited and sequence-controlled mode. The sequence controlled mode provides stronger reliability via the (Command Operations Procedure - Proximity) COP-P mechanism, which essentially implements a Go-back-N ARQ. The expedited mode provides best effort reliability via FEC but is given higher priority in terms of access the channel. The highest priority however is given to internal protocol messages such as acknowledgements or directives. The three types of frames, directive/acknowledgement, expedited, and sequence-controlled obtain access to the physical channel via a strict priority scheduler, as depicted in Figure 4. The directive/acknowledgements have the highest priority, expedited has second priority, the sequence-controlled frames has the lowest priority to the channel. One should note two FIFO queues are maintained for user data frames; one for expedited frames and the other for sequence-controlled frames. For directives and acknowledgements, at most one will be buffered at a time. When the message is a directive, a response is usually needed before another directive will be generated so queuing is not needed. When the message is an ACK, only the ACK frame with the latest value will be sent.

Nominally, one would carry isochronous data through the expedited frames, since it has higher priority over the sequence-controlled frame and it is less likely to experience head-of-the-queue blocking, except when the ACK/directive traffic is heavy. Therefore, when protocol messaging is light, the expedited frames essentially see an open channel with minimal delay introduced by waiting for an on-going transmission, either data frame or ACK/directives to finish. For an expedited frame arriving to a busy channel, it could be delayed by the transmission time (T_{tx}) of a single frame before re-gain access to the channel. Let us for a moment assume that lunar exploration will require link capacity on the order of 6Mbps, the average requirement for supporting standard definition motion imagery, then the blocking time of the channel is at most 2.7 msec.[†] which should be adequate for any real-time operation where the time constant of the control loop may range from 125msec (Low Earth Orbit) to 3+ seconds, the round-trip delay between Earth and moon.

[†] The maximum data frame size, including synchronization marker and FEC parity, is 16440 bits.

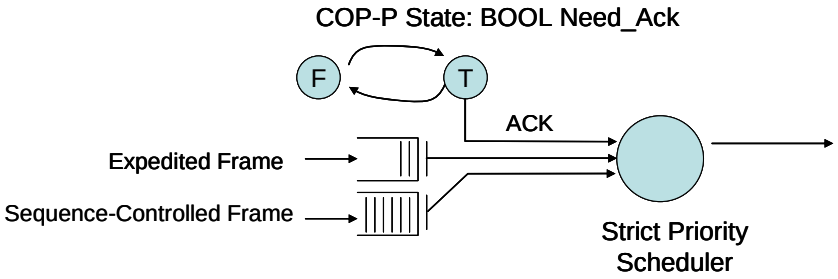

Figure 4: Strict Priority Scheduling in Proximity-1

Therefore, the stress case for isochronous service is when the directive/ACK traffic load becomes high. The directives are only issued for changing link configuration, which should not happen very often; the ACK messages, however, could take up significant bandwidth if there is on-going reliable frame transmission in the opposite direction. Figure 5 shows a scenario where the isochronous traffic is multiplexed together with the ACK frames. Depending how fast the sequence-controlled frames arrives, the rate at which acknowledgements were sent will change. Given the ACK frames have strict priority over the expedited frames; the latency of the expedited frame carrying isochronous traffic could have significant variation if the ACK traffic load is high.

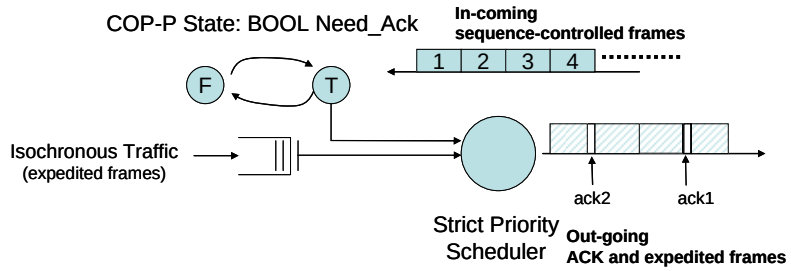

Figure 5: Isochronous traffic sharing the proximity-1 channel with acknowledgement frames

Intuitively, one expects the system to behave in the following manner. If the rate of the in-coming sequence-controlled frames is low, then the isochronous traffic will experience some but minimal channel blocking, especially when the ACK frames are much smaller than regular data frames. As the in-coming sequence control frame rate goes up, the ACK rate will increase to the point where some queuing becomes necessary. The absolute latency as well delay jitter will both increase for isochronous traffic. One could potentially mitigate the situation by changing the prioritization scheme to protect the isochronous traffic through alternative prioritization method such as round-robin or weighed-fair scheduling. However, reduced availability of the channel to the ACK traffic will, at some point, impact the efficiency of Go-back-N ARQ procedure, causing a reduction of the volume of return link telemetry.

In the coming sections, we will analyze the Proximity-1 protocol performance under such scenario where there is a mixture of expedited traffic in one direction and reliable traffic on the other, and identify circumstances under which mitigation strategies or even modifications to the specification is required.

IV. Performance of Isochronous Service using Current Proximity-1 Standard

The scenario shown in Figure 5 will be used for performances analysis. It captures the contention of the channel between an isochronous traffic stream and the Proximity-1's ARQ acknowledgment frame.

A. Contention Process between the Data and ACK frames

We begin by examining in more details the contention process between the data frames and the ACK frames. Let us refer to the out-going direct as the *forward* direction, and the in-coming direction as the *return* direction from now on. We distinguish two types of delay: queuing and blocking. *Queuing* delay is the time duration measured from the moment a data frame arrives at the queue, to the moment the data frame reached the *head of the queue*. *Blocking* delay is the time duration measured from the moment a data frame reached the head of the queue, to the

moment the frame actually gains access to the channel, i.e., the time instance when transmission begins. In this study, we will mostly focus our analysis on the blocking delay, but also discuss the queuing latency whenever appropriate.

Keeping the strict prioritization scheme in the current Proximity-1 standard, the loading of the ACK traffic on the forward link is determined by the transition rate between *True* and *False* for the "Need_ACK" variable. The Need_ACK state is set to *True* whenever a change occurs in the state of the ARQ algorithm, and it is reset to *False* whenever a PLCW, the frame that carries the ARQ acknowledgement, is sent over the channel. So the Need_ACK variable, in a nutshell, keeps track of whether the latest ARQ state has been conveyed to the other side. Nominally, the rate of ARQ state change is approximately the arrival rate of new data frame on the return link (each new data frame, if in sequence, will advance the ARQ window by 1). If the forward link is dedicated to the transmission ACK frames (as shown in Figure 6) and has sufficient capacity, then the departure rate of the ACK frames will equal to the arrival rate of data frames.

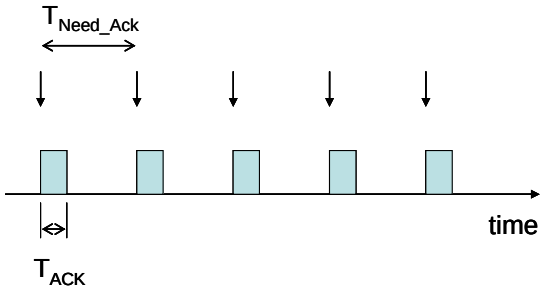

Figure 6: ACK Transmission without Contention

Let T_{Need_Ack} be the interval between each successive transition to the *True* state and let T_{ACK} be the transmission time of the ACK frame. Then we have

$$T_{Need\_Ack} = \frac{N_{pltu\_ret}}{R_{return}} \quad , \qquad T_{Ack} = \frac{N_{plcw}}{R_{forward}} \tag{1}$$

where N_{PLTU_ret}, N_{PLCW}, R_{return}, and $R_{forward}$ are the size of the data frame on the return link, size of the ACK frame, the data rate of the return link and the data rate of the forward link, respectively. The normalized loading, duty cycle, of the ACK traffic is:

$$\rho_{ack} = \frac{T_{Ack}}{T_{Need\_ACK}} = \frac{N_{plcw}}{N_{pltu\_ret}} \cdot \frac{R_{return}}{R_{forward}} \tag{2}$$

So whenever an isochronous traffic frame arrives into the system, it may encounter blocking. Obviously the larger the data frame size or forward link data rate, the lower the duty cycle on the channel, or ρ_{ack}, will be and the channel is more available to carry data traffic in the forward direction.

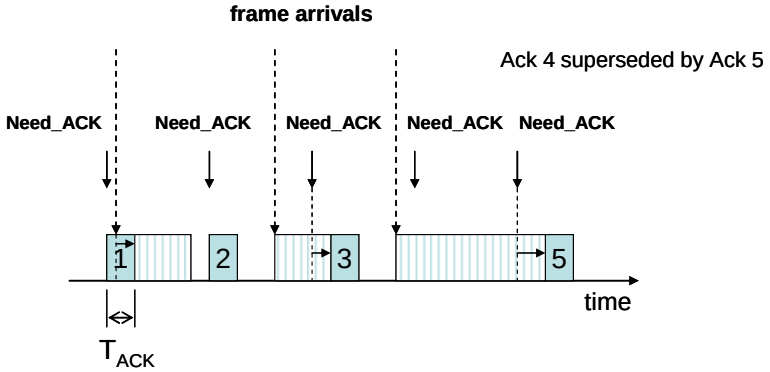

Figure 7: Contention situations between ACK and data frame

Figure 7 illustrates several different contention situations between the ACK frames and the data frames. The first data frame arrived to the head of the queue while an ACK frame is being transmitted, so it is blocked until the ACK frame cleared the channel, i.e., finished transmission on the channel. The second data frame arrives to an available channel, but its transmission time extends beyond the time instant when the Need_ACK #3 was triggered, so the ACK frame is blocked until the data frame clears the channel. The third data frame arrives before Need_ACK #4 is triggered but its transmission time extends beyond the time instant when Need_ACK #5 is set to true. Because the Proximity-1 specification requires that all acknowledgements should sent out only the most up-to-date information regarding the ARQ process, ACK #4 is replaced by ACK#5 will be send after the data frame clears the channel. The performance of the data frame transmission, therefore, depends on which type of contention situation is the most dominant, given the loading of the data traffic and the ACK traffic on the forward link. Let ρ_{data} denote the offered load of the forward link data traffic, then we have

$$\rho_{data} = T_{data\_frame} \cdot \lambda_{data\_frame} = \frac{N_{pltu\_for}}{R_{forward}} \cdot \lambda_{data\_frame} \tag{3}$$

where T_{data_frame}, λ_{data_frame} are the transmission time of the data frame and the offered rate of the data frame on the forward link. Note that $1/T_{data\_frame} < \lambda_{data\_frame}$. The stability of the forward link requires that

$$\rho_{data} < 1 - \rho_{ACK} = 1 - \frac{N_{plcw}}{N_{pltu\_ret}} \frac{R_{return}}{R_{forward}} \tag{4}$$

Notice that the ratio N_{plcw}/N_{pltu_ret} is the ratio of the ACK frame size to the return link data frame size; if the maximum payload is used, it is approximately 0.0068. The typical return link to forward link data rate ratio, for the Mars Exploration Rover and Odyssey orbiter, is about 128kbps to 8kbps or a factor of 16. This means that the normalized loading of the forward link data frame cannot exceed 89% of the forward link bandwidth for MER-Odyssey. However, as the loading of the data traffic approach this limit, the contention will create very significant delay and jitter to the system that one should be taken into account when designing the link. Also the ARQ process on the return link may experience significant throughput degradation if the protocol is not properly configured.

B. Performance under light loading conditions

Let us start by considering what will happen under light loading conditions, i.e., $\rho_{data} << 1 - \rho_{ack}$. Specifically, we define light loading to mean that the inter-arrival time between consecutive data frames on the forward link is sufficiently large such that blocking and latency experienced by consecutive data frames are not correlated. In other words, the contention process between one data frame and the ACK frames does not create residual effects on the next arriving data frame; the effect of one data frame on the ARQ process of the return link, does not propagate to the next data frame. This definition is rather qualitative but we will provide more quantitative definition in the analysis. Under light loading condition, we only need to examine the impact of one data frame on the system to understand the average behavior over time.

1. Jitter

In the worst case scenario, a data frame arrives to the head of the queue just as an ACK frame begins transmission. At the end of the ACK frame transmission, the channel will free up because $\rho_{ack} < 1$. So the data frame can gain access. Therefore, the maximum blocking delay jitter under light loading condition is just T_{ACK}, which is the maximum latency a data frame will experience

Even though the data frame arrival rate is low, the size of the data frame may have a significant impact on the ARQ process for the return link. Here we have to consider the impact of how the data frame may block and even cause gaps in ARQ ACK frames. We distinguish two cases based on the transmission time of the data frame.

2. Impact on ARQ: Short Data Frame on Forward Link

If the transmission time $T_{data\_frame} \leq T_{Need\_Ack}$, any data frame transmission cannot overlap two consecutive reset of the Need_ACK state. Therefore, no ACK frames will be superseded by later ACKs due to extended blocking by a data frame. One expects that the throughput of the ARQ process on the return link data frame will not be impacted significantly. The effect of the jitter of the ACK frames on the ARQ process can be neutralized by simply increasing the window size by 1, which will accommodate the maximum delay of an ACK frame without causing premature re-transmissions.

3. Impact on ARQ: Long Data Frame on Forward Link

However, if the data frame on the forward link is large such that $T_{data\_frame} > n * T_{Need\_Ack}$ where $n \geq 1$, then each data frame *could* potentially block n + 1 ACKs, and causing n ACKs to be skipped. Figure 7 shows one such case. In other words, the ACK for data frame #m (on the return link) will, after some delay, be followed by the ACK for data frame #(m+n-1) (on the return link). The ARQ algorithm will continue to operate correctly under such situation because the protocol recognized cumulative acknowledgement, i.e., by acknowledging frame #m, all frame prior to frame #m are implicitly acknowledged. However, there will be some loss of efficiency if the protocol is not configured to take this into account.

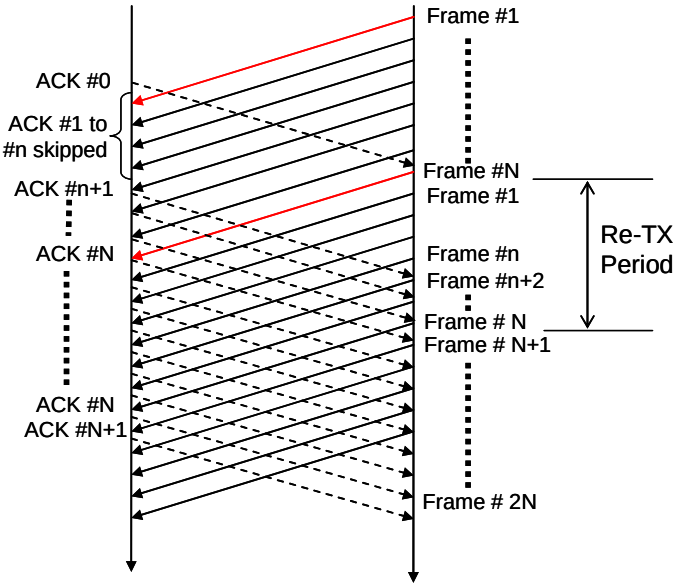

Figure 8: ARQ with ACK gap n < N

Under nominal conditions, the window size of the ARQ protocol is set to a specific value N, that ensures before frame #N completes transmission the acknowledgement of frame #1 should be received. N is typically selected, depending on the round-trip delay and other latency factors in the system, such that the RF channel is kept at its maximum utilization, with new frames being transmitted continuously without "dead air". However, N is usually selected under the assumption there is a continuous, gap-free reception of ACKs – a reasonable assumption if the forward link if very reliable and is dedicated to transmission of ACK. When there is contention between the ACK and data frame, the typical value of N may cause pre-mature re-transmission.

We define an re-tx period as illustrated in Figure 8 in which we assumed that $n \leq N-1$. The period begins when at the end of sending frame #N, the sender did not received the expected acknowledgement for frame #1, so it goes into a "progressive retransmission" process and start sending frame #1 again assuming that frame #1 to #N were lost. After missing n ACKs, ACK for frame #n+1 is finally received by the sender, at which time the sender has just finished re-transmitting frame #n and will jump from sending frame #n to #n+2, having received the ACK for frame #n+1. The cycle completes when the sender starts the transmission of frame #N+1. All together N-1 frames were re-transmitted unnecessarily, $T_{re\_tx\_period} = N_{pltu}/R_{return}$ (N-1).

One can further extend the analysis to situations where n > N-1. In that case when the ACKs finally resumed, it will acknowledge frame #N because that is the highest ever observed by the receiver, and the sender side will immediately jump right up to transmit frame # N+1. (See Figure 9) So there are total of n duplicate transmissions.

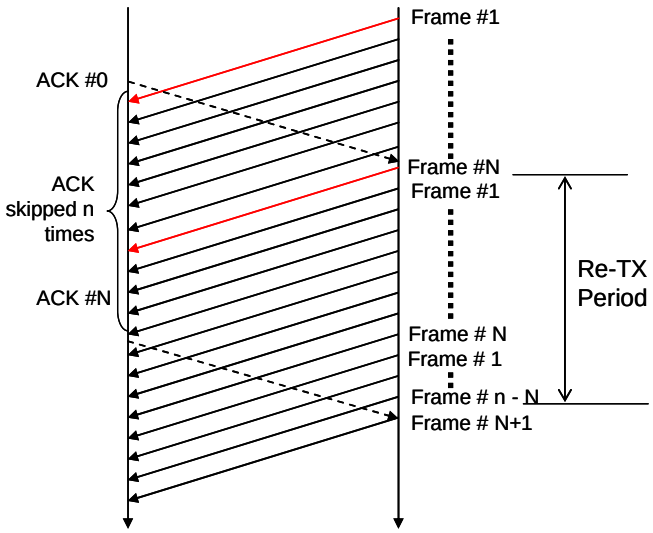

Figure 9: ARQ with ACK gap cycle (n ≥ N)

We have examined the impact of one ACK gap caused by a single data frame on the forward link because under light loading conditions where the inter-arrival time of the data frames are sufficiently large; in other words, $1/\lambda_{data\_frame} > \max(n,N-1) * T_{Need\_ACK} = \max(n,N-1) * N_{pltu\_ret}/R_{return}$. We know that n is the greatest integer less than the ratio of the transmission time of the forward link data frame and the Need_ACK period, which is given by $\lfloor T_{data\_frame}/T_{Need\_ACK} \rfloor$. Then the efficiency of the ARQ process, or η_{ARQ}, when $T_{data\_frame} > T_{Need\_ACK,}$ is given by:

$$\begin{aligned}\eta_{ARQ} &= 1-\lambda_{data\_frame}\cdot\max\left(n, N-1\right)\cdot\frac{N_{pltu\_ret}}{R_{return}} \\ &= 1-\lambda_{data\_frame}\cdot\max\left(\left\lfloor\frac{N_{pltu\_for}\cdot R_{return}}{N_{pltu\_ret}\cdot R_{forward}}\right\rfloor, N-1\right)\cdot\frac{N_{pltu\_ret}}{R_{return}}\end{aligned} \tag{5}$$

Typically N is chosen to be the least number of return frames that will fill the round-trip time, which is $\lceil T_{round\text{-}trip}/(N_{pltu\_ret}/R_{return}) \rceil$, The round trip time $T_{round\text{-}trip}$ includes the two-way propagation time and the transmission, coding, and processing delay of the data frame on the return link and ACK frame on the forward link. Substituting this into equation (5), we can compute the expected ARQ efficiency under light loading condition.

$$\eta_{ARQ} = 1-\lambda_{data\_frame}\cdot\max\left(\left\lfloor\frac{N_{pltu\_for}\cdot R_{return}}{N_{pltu\_ret}\cdot R_{forward}}\right\rfloor, \left\lceil\frac{T_{round-trip}\cdot R_{return}}{N_{pltu\_ret}}\right\rceil-1\right)\cdot\frac{N_{pltu\_ret}}{R_{return}} \tag{6}$$

4. General Recommendations under light loading conditions

The jitter experienced by forward link frame is T_{ack} under light loading condition, which is a function of the forward link data rate, which needs to be factored into the link design. Because contention is actually low, the delay jitter will not be improved by giving isochronous data frame higher access priority than the ACK frames; the jitter is simply caused by arrival during on-going transmission of an ACK.

As equation (5) indicates, the ARQ efficiency is driven by the round-trip time as well as the size transmission time of the forward link data frames. To minimize loss in ARQ efficiency, one can either: (a) increase the window size to N + n, where n is the maximum number of ACKs that could be skipped due to blocking by a forward link frame, to prevent pre-mature re-transmissions, or (b) reduce the forward link frame size so that $T_{data\_frame} < T_{Need\_ACK}$; this will eliminate the creation of ACK gaps.

In general, under light loading conditions there is no need to modify the current prioritization scheme in the Proximity-1 specification. However, if the required jitter cannot be met by increasing forward link bandwidth, one can modify the current standard to include a weighted fair scheduler so that data frames will receive guaranteed

bandwidth at the expense of creating ACK gaps. One can then mitigate the impact of the ACK gap by increasing the ARQ window size appropriately.

C. Performance under high loading conditions

High loading condition means that the offered load of the forward link data, ρ_{data} is significant compare to 1 - ρ_{ack}, which means that the contention between data and ACK frames become very intense and tend to correlated from one data frame to the next. The effect of this contention will create more channel blocking and higher jitter, which is detrimental to the isochronous traffic. To understand the latency jitter, the worst case blocking by the ACK frames will be analyzed.

The current Proximity-1 specification requires that only latest information should be sent in ARQ acknowledgement message. Therefore, when the ARQ state is updated, any buffered ACK frame *waiting* for transmission will be replaced by a new ACK frame. Because only the latest ACK frame is queued for transmission, long blockage due to ACK frame transmission does not occur normally. The only scenario in which a data frame could be blocked by multiple, consecutive ACK frames is when the ARQ state is updated *while* an ACK frame is being transmitted. Then at the end of the ACK frame transmission, the channel will not be released to the data frame but rather kept for sending the newly updated ACK. So this is the focus of our jitter analysis.

If we assume that $\rho_{ack} < 1$, the ARQ state will not change on top of an on-going ACK frame transmission unless the ACK frame being transmitted was previously delayed by a data frame. Two cases can be further distinguished and considered: (1) $\rho_{ack} < 1/2$ and (2) $1 > \rho_{ack} \geq 1/2$.

1. *Case 1: $\rho_{ack} < 1/2$*

Figure 10 illustrates a possible scenario. If an ACK frame is delayed sufficiently such that during its transmission time, the ARQ state changed again, then at the end of the ACK frame transmission, another ACK frame will follow immediately. Under high loading condition, we assume that the 2nd data frame could arrive potentially arrive and move immediately to the head of the queue within the window of 2 * T_{ack}. Then the 2nd data frame will not gain channel access until the end of the 2nd ACK frame transmission. Because $\rho_{ack} < 1/2$, (or equivalently, $T_{Need\_Ack} > 2 * T_{Ack}$) there is no danger of having the second ACK frame blocking the third ACK frame. This means that number of consecutive ACK frames transmissions is upper-limited to two, thus the maximum delay jitter due to channel contention is 2 * T_{ack}, doubling the bound under light loading condition.

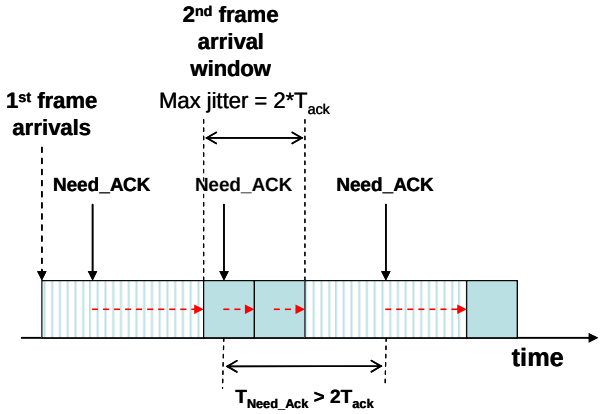

Figure 10: High Loading Condition with two consecutive ACK frames

When the loading of the data frame approach 1-ρ_{ack}, significant queuing delay, as well as contention (blocking delay) will dominate the latency and jitter performance and degrade the quality of service. However, the analysis of queueing latency requires knowledge of the arrival statistics of the traffic so we will not treat the queueing issue in depth but rather only focus on blocking latency here.

2. *Case 2: $1 > \rho_{ack} \geq 1/2$*

There can be a stream of k ACK frames depending on ρ_{ack} and how long an ACK frame delayed as shown in Channel blocked by K consecutive ACK frames.

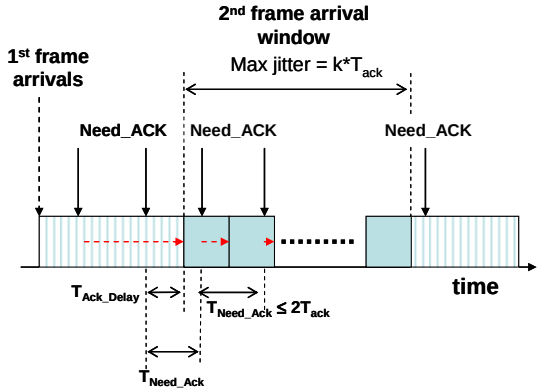

Figure 11: Channel blocked by K consecutive ACK frames

To compute the number of consecutive ACK frames that may occur, we define $T_{def}(i+1)$ as the delay experienced by the (i+1)th ACK frame, due to blocking by the i-th ACK frame. Note that by definition T_{def} is always less than T_{Need_Ack} because only the latest ACK frame is buffered for transmission. Figure 12 shows the timing relationship.

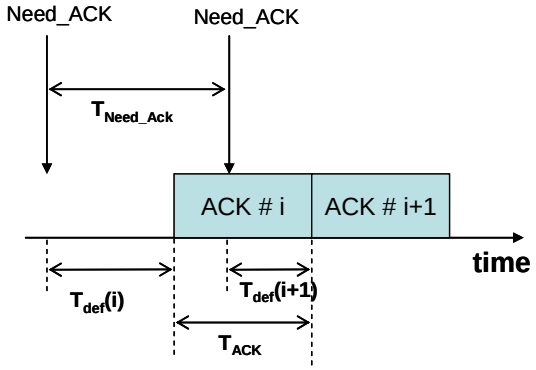

Figure 12: $T_{def}(i)$ Timing Diagram

The iterative relationship between $T_{def}(i)$ is given by:

$$T_{def}\left(i+1\right) = T_{ACK} - T_{Need\_Ack} + T_{def}\left(i\right) \tag{6}$$

where $T_{def}(i) > 0$. Note that since $T_{Ack} < T_{Need\_Ack}$, $T_{def}(i)$ is always decreasing, and the burst length of the ACK frames is the largest index *i* such that $T_{def}(i+1) > 0$, i.e., the number of consecutive frames, K, is given by $\lceil T_{def}(1)/(T_{Need\_Ack} - T_{ACK}) \rceil$. The maximum delay jitter occurs when $T_{def}(1) = T_{Need\_ACK}$:

$$D_{\max\_jitter} = T_{ACK} \max_{i>0} \left\{ T_{def}\left(i+1\right) > 0, T_{def}\left(1\right) = T_{Need\_ACK} \right\} = T_{ACK} \left\lceil \frac{T_{Need\_ACK}}{T_{Need\_ACK} - T_{ACK}} \right\rceil = T_{ACK} \left\lceil \frac{1}{1-\rho_{ACK}} \right\rceil \tag{7}$$

This result is general and includes the $\rho_{Ack} < 1/2$ case. Notice as $\rho_{Ack} \rightarrow 1$, the maximum jitter is *unbounded*, making the forward link really unsuitable for isochronous traffic. Furthermore, because the maximum number of consecutive ACKs is unbounded, significant queueing delay could be incurred by data frames carrying isochronous traffic.

3. Impact on ARQ Efficiency

The impact on ARQ efficiency on the reverse direction is similar to the light loading case. The driving factor is the transmission time of the data frame, if the data frame is sufficient large and will create ACK gaps, the window

size of the ARQ algorithm should be adjusted accordingly. When the data frame is small, the jitter on ACK frame is bounded by T_{Need_Ack}, which can be easily compensated by increasing the window size by one.

V. Recommendations and Conclusions

In this study, we analyzed the jitter performance of carrying isochronous service and its impact on ARQ efficiency on the reverse direction, when using the CCSDS Proximity-1 Space Link protocol. We presented analysis of the maximum jitter of the isochronous data frame and show that it is an inverse function of the residual bandwidth on the forward link, $(1-\rho_{Ack})$, which in term depends on the degree of asymmetry between data rates of the two directions of the full-duplex link. For example, our result shows that if the normalized loading of the acknowledgement traffic is kept under 50%, the maximum delay jitter for isochronous traffic is upper-bounded by twice the transmission time of an acknowledgement frame.

The ARQ efficiency of Proximity-1 is strongly influenced by the transmission duration of the isochronous data frame in the opposite direction of the link. Long transmission time causes delay as well as gaps in the acknowledgement frames and may trigger pre-mature re-transmission that reduced the ARQ throughput. In this study, we analyzed the ARQ efficiency as a function of the link configuration when the window size is selected *without* taking into account the impact of gaps in the acknowledgements. One can mitigate the adverse effect on ARQ performance by increase the retransmission window size by the expected acknowledgement gaps size to eliminate duplicate transmissions, or by controlling the frame size of the isochronous traffic and the data rate. The choice of either method will depend on the constraints of individual applications.

Our study showed that in general, no modification to the current proximity-1 specification is needed to deliver isochronous service, given jitter tolerance on the order of the acknowledgement frame transmission time. For applications requiring very precise transmission timing, with jitter lower than the acknowledgement frame transmission time, the strict priority mechanism used by Proximity-1 to arbitrate channel access between the acknowledgement and data frames need to be update. A potential candidate is a form of weighted-fair scheduling method that provides bandwidth guarantee to the data frames under constrained scenarios.

Acknowledgments

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1] "CCSDS Proximity-1 Space Link Protocol — Data Link Layer," Blue Book, Issue 3, May 2004, URL: http://public.ccsds.org/publications/archive/211x0b3.pdf [cited March 23, 2006]