The Consultative Committee for Space Data Systems

Draft Recommendation for
Space Data System Standards

AOS SPACE DATA LINK PROTOCOL

DRAFT RECOMMENDED STANDARD

CCSDS 732.0-P-4.1

PINK SHEETS
October 2023

The Consultative Committee for Space Data Systems

Draft Recommendation for Space Data System Standards

AOS SPACE DATA LINK PROTOCOL

DRAFT RECOMMENDED STANDARD

CCSDS 732.0-P-4.1

PINK SHEETS

October 2023

DOCUMENT CONTROL

Document	Title	Date	Status
CCSDS 732.0-B-1	AOS Space Data Link Protocol, Issue 1	September 2003	Original issue, superseded
CCSDS 732.0-B-2	AOS Space Data Link Protocol, Recommended Standard, Issue 2	July 2006	Issue 2, superseded
CCSDS 732.0-B-3	AOS Space Data Link Protocol, Recommended Standard, Issue 3	September 2015	Issue 3, superseded
CCSDS 732.0-B-4	AOS Space Data Link Protocol, Recommended Standard, Issue 4	October 2021	Current issue
CCSDS 732.0-P-4.1	AOS Space Data Link Protocol, Draft Recommended Standard, Issue 4.1	October 2023	Current draft update: – adds clarifying text to Frame Header Error Control specification; – modifies Multiplexing Protocol Data Unit header to increase the size of the First Header Pointer.

4.1.2.6.4 Once set by management, the presence or absence of the Frame Header Error Control shall be static throughout a Mission Phase.

4.1.2.6.5 The mechanism for generating the Frame Header Error Control shall be a ~~shortened Reed-Solomon (10,6) code~~Reed-Solomon (15, 11) code over GF(2^4), shortened by 5 symbols, and converted to GF(2), to form a binary (40, 24) code. The parameters of the selected code are as follows:

a) 'J=4' bits per Reed-Solomon (R-S) symbol.

b) 'E=2' symbol error correction capability within an R-S code word.

c) The field generator polynomial shall be:

$$F(X) = x^4 + x + 1$$

over GF(2)

d) The code generator polynomial shall be:

$$g(x) = (x + \alpha^6)(x + \alpha^7)(x + \alpha^8)(x + \alpha^9)$$

over GF(2^4)

where F(α) = 0,

$\alpha^6 = 1100$, $\alpha^7 = 1011$
$\alpha^8 = 0101$, $\alpha^9 = 1010$

also:

$$g(x) = x^4 + \alpha^3 x^3 + \alpha x^2 + \alpha^3 x + 1$$

over GF(2^4)

and:

$\alpha^0 = 0001$, $\alpha^3 = 1000$
$\alpha = 0010$

e) Within an R-S symbol, the transmission shall start from the bit on the left side; e.g.,

$\alpha^3 = 1000$

shall be transmitted as a 1 followed by three 0s.

f) Five symbols of virtual fill shall be used to logically complete the Reed-Solomon codeword. This virtual fill shall:

   1) consist of all zeros;

   2) not be transmitted;

   3) be inserted only at the beginning of the codeword.

g) The selected code is a systematic code.

h) The bit to R-S symbol mapping shall be:

bits in the header	symbol	function
—	1	virtual fill
—	2	virtual fill
—	3	virtual fill
—	4	virtual fill
—	5	virtual fill
0,1,2,3	~~0~~6	systematic symbol
4,5,6,7	~~1~~7	systematic symbol
8,9,10,11	~~2~~8	systematic symbol
12,13,14,15	~~3~~9	systematic symbol
40,41,42,43	~~4~~10	systematic symbol
44,45,46,47	~~5~~11	systematic symbol
48,49,50,51	~~6~~12	parity symbol
52,53,54,55	~~7~~13	parity symbol
56,57,58,59	~~8~~14	parity symbol
60,61,62,63	~~9~~15	parity symbol

NOTES

1 The purpose of this field is to provide a capability for protecting some key elements in the Transfer Frame Primary Header.

2 Whether this field should be used on a particular Physical Channel is determined based on the mission requirements for data quality and the selected options for the Channel Coding Sublayer.

3 The header error correction code can correct up to and including two symbol errors. This is sufficient to meet the performance of < ~~1×10E-07~~ 10^{-7} Data Fields missing at a ~~1×10E-05~~ 10^{-5} channel bit error rate, for random bit errors. In the case of convolutional coded channels, in particular when the convolutional coding is interleaved, the Data Field loss rate will drop to 2× ~~10E-05~~ 10^{-5} at an operating point equivalent to a channel bit error rate of ~~1×10E-05~~ 10^{-5}. This is the result of the burst errors typical of the convolutional decoders.

4.1.4.1.4 M_PDUs, B_PDUs, VCA_SDUs, and Idle Data shall not be mixed in a Virtual Channel (i.e., if a Virtual Channel transfers M_PDUs, every Transfer Frame of that Virtual Channel shall contain an M_PDU). Management shall decide whether M_PDUs, B_PDUs or VCA_SDUs are transferred on a particular Virtual Channel, and this decision shall remain static throughout a Mission Phase.

4.1.4.1.5 In the case where no valid Transfer Frame Data Field is available for transmission at release time for a Transfer Frame, a Transfer Frame with a Data Field containing only Idle Data shall be transmitted. Such a Transfer Frame is called an OID Transfer Frame.

4.1.4.1.5.1 The Virtual Channel ID of an OID Transfer Frame shall be set to the value of 'all ones'.

4.1.4.1.5.2 The Transfer Frame Data Field of an OID Transfer Frame shall be filled in by the mandatory Pseudo Noise (PN) sequence generated by a 32-cell Linear Feedback Shift Register (LFSR) with polynomial $D^0 + D^1 + D^2 + D^{22} + D^{32}$~~, as shown in annex D~~.

NOTE – Annex D contains example implementations of this LFSR.

4.1.4.1.5.2.1 An LFSR that follows the Fibonacci form (see figure D-1) shall be initialized at device start-up with an 'all-ones' seed and shall not be restarted.

4.1.4.1.5.2.2 An LFSR that follows the Galois form (see figure D-2) shall be initialized at device start-up with a '00000000001111111111111111111101' seed and shall not be restarted.

NOTES

1 Transfer Frames containing Idle Data in their Data Fields are sent to maintain synchronization at the receiver and also to transmit data in the Transfer Frame Insert Zone when there is no Data Field to send.

2 Idle Data in the Transfer Frame Data Field of an OID Transfer Frame must not be confused with the Idle Packet specified in reference [8].

3 A random pattern is used in the OID Transfer Frame Data Field to avoid frame-reception problems resulting from insufficient randomization.

4.1.4.2 Multiplexing Protocol Data Unit

4.1.4.2.1 Overview

4.1.4.2.1.1 The Multiplexing Protocol Data Unit (M_PDU) shall follow, without gap, the Transfer Frame Primary Header or the Transfer Frame Insert Zone if present.

4.1.4.2.1.2 The length of the M_PDU shall be fixed by management for any particular Virtual Channel, since it is required to fit exactly within the fixed-length Transfer Frame Data Field.

NOTE – The length of M_PDUs carried by a Physical Channel which supports the Insert Service must take into account the fixed length of the optional Insert Zone.

4.1.4.2.1.3 The M_PDU shall be divided as follows:

a) M_PDU Header (2 octets, mandatory);

b) M_PDU Packet Zone (integral number of octets, mandatory).

~~4.1.4.2.1.4 The M_PDU Header shall be sub-divided as follows:~~

~~a) Reserved Spare (5 bits, mandatory);~~

~~b) First Header Pointer (11 bits, mandatory).~~

4.1.4.2.1.4 The M_PDU Header shall contain the First Header Pointer (16 bits, mandatory).

4.1.4.2.1.5 The format of the M_PDU is shown in figure 4-3.

M_PDU HEADER		M_PDU PACKET ZONE				
~~RSVD.~~ SPARE 5 bits	~~FIRST~~ HEADER POINTER 11 bits	~~END OF~~ PREVIOUS CCSDS PACKET #k	~~CCSDS~~ PACKET #k+1	. . . . . .	~~CCSDS~~ PACKET #m	~~START OF~~ CCSDS PACKET #m+1

M_PDU HEADER	M_PDU PACKET ZONE				
FIRST HEADER POINTER 16 bits	END OF PREVIOUS CCSDS PACKET #k	CCSDS PACKET #k+1	. . . . . .	CCSDS PACKET #m	START OF CCSDS PACKET #m+1

Figure 4-3: Multiplexing Protocol Data Unit (M_PDU)

~~**4.1.4.2.2 Reserved Spare**~~

~~**4.1.4.2.2.1** Bits 0–4 of the M_PDU Header shall contain the Reserved Spare.~~

~~**4.1.4.2.2.2** This five-bit Reserved Spare field is currently undefined by CCSDS; by convention, it shall therefore be set to the reserved value of ‘00000’.~~

4.1.4.2.2 First Header Pointer

4.1.4.2.2.1 Bits ~~5~~0–15 of the M_PDU Header shall contain the First Header Pointer.

4.1.4.2.2.2 The First Header Pointer shall contain the position of the first octet of the first Packet that starts in the M_PDU Packet Zone.

4.1.4.2.2.3 The locations of the octets in the M_PDU Packet Zone shall be numbered in ascending order. The first octet in this zone is assigned the number 0. The First Header Pointer shall contain the binary representation of the location of the first octet of the first Packet that starts in the M_PDU Packet Zone.

NOTES

1 The purpose of the First Header Pointer is to facilitate delimiting of variable-length Packets contained within the M_PDU Packet Zone, by pointing directly to the location of the first Packet from which its length may be determined.

2 The locations of any subsequent Packets within the same M_PDU Packet Zone will be determined by calculating the locations using the length field of these Packets.

3 If the last Packet in the M_PDU Packet Zone of Transfer Frame *N* spills over into Frame *M* of the same Virtual Channel ($N<M$), then the First Header Pointer in Frame *M* ignores the residue of the split Packet and indicates the start of the next Packet that starts in Frame *M*.

4.1.4.2.2.4 If no Packet starts in the M_PDU Packet Zone, the First Header Pointer shall be set to ‘all ones’.

NOTE – The above situation may occur if a long Packet extends across more than one Transfer Frame.

4.1.4.2.2.5 If the M_PDU Packet Zone contains only Idle Data, the First Header Pointer shall be set to ‘all ones minus one’.

4.1.4.2.3 M_PDU Packet Zone

4.1.4.2.3.1 The M_PDU Packet Zone shall follow, without gap, the M_PDU Header.

4.1.4.2.3.2 The M_PDU Packet Zone shall contain either Packets or Idle Data (a project-specified ‘idle’ pattern).

NOTE – The idle pattern used in the M_PDU is project specific and can be fixed or variable length. A random pattern is preferred. Problems with the reception of frames have been encountered because of insufficient randomization.

ANNEX D

OID FRAME RANDOMIZATION IMPLEMENTATION OPTIONS

(INFORMATIVE)

Two possible implementations of the OID Frame Randomizer are shown in figures D-1 and D-2.

For each frame, the LFSRs are initialized at device start-up as follows:

a) with an all-one seed when the generator in figure D-1 (Fibonacci form) is used; and

b) with a '00000000001111111111111111111101' seed when the generator in figure D-2 (Galois form) is used.

In both cases LFSRs are not restarted for subsequent OID Transfer Frames.

The first 10 bytes of the OID data pattern, in hexadecimal, are:

FF FF FF FF 6D B6 D8 61 45 1F

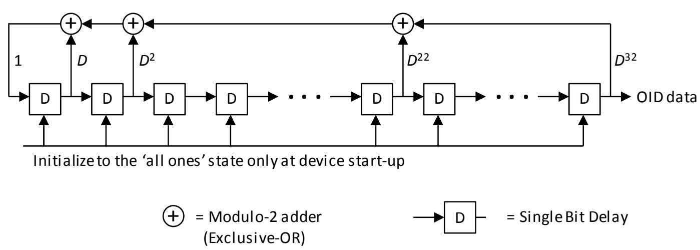

Figure D-1: OID Data Generation Logic Diagram (Fibonacci Form)

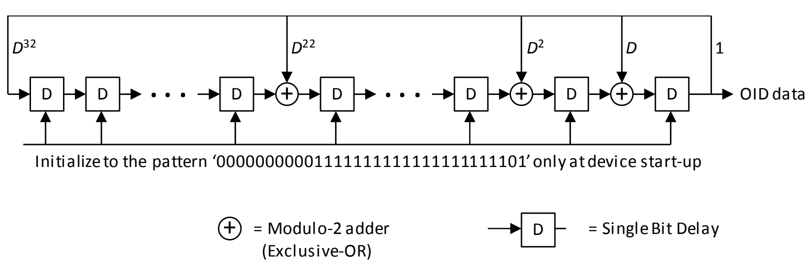

Figure D-2: OID Data Generation Logic Diagram (Galois Form)

Generated data pattern in both cases:

FF FF FF FF 6D B6 D8 61 45 1F 11 F1 97 16 72 3C BE 7E 00 B1 . . .