4910 Alameda Blvd NE
Albuquerque NM 87113

Great River Technology

ARINC 818 Implementer's Guide

An Introduction to the Avionics Digital Video Bus and Its 2013 Update

June 2, 2014

Contact Information

Telephone	1 (866) 478-4419
Company Fax	(505) 883-1375
Email	grt@greatrivertech.com
Web Address	www.greatrivertech.com

Revision history

Date	Section(s)	Description	Approval
30 Dec 2006		Initial release	J Alexander
20 Feb 2014	All	Adds general description of new capabilities under ARINC 818-2; updates table of related relevant hardware; improves illustrations and formatting.	M Gadde
02 Jun 2014	3.7.5	Corrects emphasis in Table 3.7.5 and discussion from Class C2 to Class C1.	M Gadde

Contents

1 Overview

1.1 Purpose of this guide

This guide introduces designers of high-speed video systems to the ARINC 818 Avionics Video Digital Bus (ADVB) and its 2013 supplement, ARINC 818-2. It is intended for:

- Designers of video systems hardware and engineering managers who have little or no prior knowledge of ARINC 818
- Experienced ARINC 818 designers and managers desiring an introduction to the 2013 supplement (ARINC 818-2)
- Designers and managers exploring the utility of ARINC 818-2 beyond the avionics community.

The guide covers basic ARINC 818 implementations and does delve into some of the more complex ARINC 818-2 features, such as channel bonding and regions of interest. It also surveys the available building blocks for ADVB interfaces, discusses design architectures, and lays out the early choices designers must make concerning link speed, synchronization, and receiver buffering. It is intended to accelerate the design of extremely reliable high-performance video through an understanding of the ARINC 818-2 specification.

1.2 Availability of the standard

ARINC 818-2 is available for purchase directly through ARINC at https://www.arinc.com/cf/store/catalog.cfm?prod_group_id=1&category_group_id=63.

1.3 Development of ARINC 818 and its supplement

In 2005 Airbus, Boeing, and other aerospace companies drove the effort to improve avionics architectures for the new 787 and A400M programs by developing a new standard for digital video used in cockpits. The effort was initiated through the Digital Video Subcommittee of ARINC. The subcommittee considered several available technologies as the foundation for the original ARINC 818 standard. It decided to base ARINC 818 on FC-AV (ANSI INCITS 356-2002).

Why FC-AV? Before 2006, it had already proven itself in avionics applications. It fit well with the ARINC 818 concept because it was video-centric. A primary factor was FC-AV's ability to handle high bandwidth, which ARINC 818 would require to support uncompressed digital video for the avionic environment.

There are several reasons that avionics video be distributed without compression. Although many interface standards use compression to greatly reduce the bandwidth required; avionics video involves fine lines and textual information, for which any losses

due to compression could be hazardous. Also, the real-time blending of video—for example, when symbology overlays digital map images—is a common requirement for avionics displays and a reason for avoiding compressed video. Finally, compression codecs add latency, and ARINC 818 was intended to support many time-critical functions, such as HUD-assisted takeoff and landing, where latency must be minimized.

Without compression, high-resolution video requires significant bandwidth. For example, SXGA using 24-bit RGB at 60 hertz requires a bandwidth of more than 2 gigabits per second (Gb/s). The bandwidth needs for UXGA approaches 4 Gb/s (Figure 1.3).

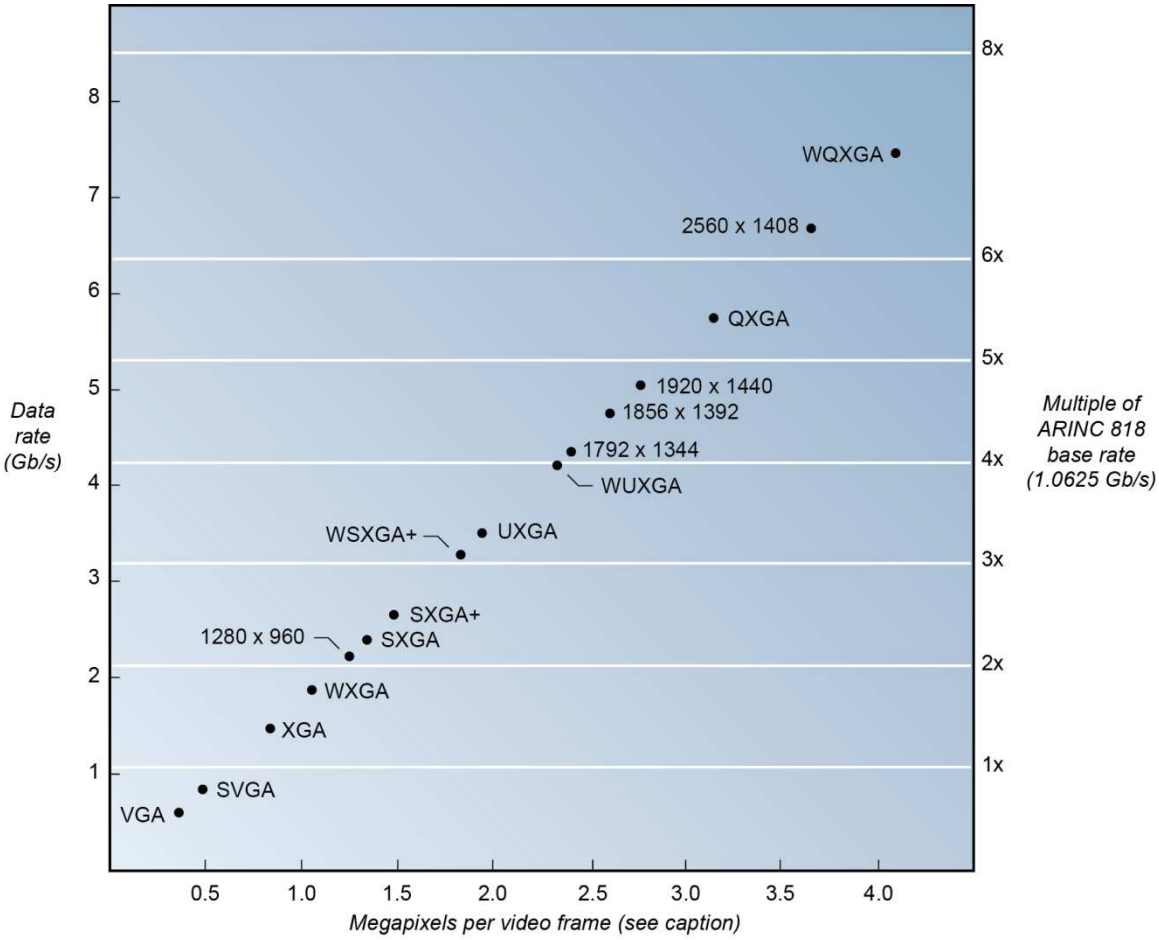

Figure 1.3. Bandwidth requirements for various display formats in 24-bit RGB at 60 Hz. (Here the word* frame *carries its conventional meaning, but note that a special use of the term is defined in Section 3.3.)

For this level of bandwidth, fiber-optic interfaces have a strong advantage over copper. Fiber has other benefits for the avionics environment, where distance and weight matter. In large aircraft, such as a cargo planes, video sources and displays might be separated by more than 30 meters. Multimode fiber can cover distances approaching 500 meters. The sum of all cables used for video distribution, were it to be copper, would add significant weight to the aircraft. For its bandwidth, distance capabilities, and weight, fiber optic was chosen.

ARINC 818 was ratified in October 2006 by the Airlines Electronic Engineering Committee (AEEC) and generated enthusiastic industry support. Since then, ARINC 818 has been used as the video transport protocol for cockpit displays on the Boeing 787 and the KC-46A tanker; Airbus A350 and A400M; the COMAC C-919; the C-17, F15, F18 upgrade programs; and numerous other commercial and military aircraft, becoming the de facto military standard worldwide as well as the commercial standard. It has contributed to systems such as infrared and wavelength sensors, optical cameras, radar, flight recorders, map/chart systems, which in turn contributed to taxi and take-off assist, cargo loading, navigation, target tracking, collision avoidance, and other critical functions.

Since 2006, as programs advanced, new requirements and applications for the ARINC 818 protocol arose. Link rates of 14.025, and 28.05 Gb/s have been released with even higher speeds planned as the market needs it. For example, a display at WQXGA resolution (2560 x 1600 pixels) with 24-bit RGB at 60 hertz would need a bandwidth of 7.372 Gb/s (Figure 1.3).

Meanwhile, custom applications of ARINC 818 led to work in areas that the original specification omitted—for example, video compression and encryption, switching, and bi-directional synchronization. This work also led to a standard means for computing prior-image cyclical redundancy checks (CRC).

Early in 2013, ARINC approved a project to advance the protocol. Representatives from Airbus, Boeing, COTSWORKS, Elbit, Thales, Honeywell, DDC, SRB Consulting, and Great River Technology proposed, discussed, and drafted the items for the supplement, completing the work that August. On October 31, 2013, the AEEC Executive Committee unanimously approved the ARINC 818-2 draft, and ARINC released ARINC 818-2 on December 18, 2013.

1.4 The first things to know about ARINC 818

ARINC 818 is a simplification of FC-AV. Because ARINC 818 began as a unidirectional point-to-point video link, many of the complexities associated with FC-AV and Fibre Channel lower layers are greatly simplified. For instance, there is no link initialization, no fabric login, no flow control, and no exchanges—all of which require a bi-directional link. Because of this, much of the FC-2 layer and some of the FC-1 layer (the bulk of the FC-PH document) are not used. This is good news for designers because meeting these requirements would require that designs always have a micro-processor and software to manage the exchanges. ARINC 818 can be implemented in a PLD or FPGA. Because it is a simplification, the learning curve for designers is reduced. Under the ARINC 818-2, bi-directional communication is possible—for example, to provide a return path for sensor control. However, it is achieved by an independent ARINC 818 path, *not* by reverting to the complexities of FC-AV. No handshaking has been added, but if needed, this could be defined in an interface control document for a particular project.

Each ARINC 818 project uses an associated interface control document (ICD) to achieve flexibility *and* interoperability. ARINC 818 allows for flexibility in the implementation of the video interface. This flexibility is desirable because of the diverse resolutions, grayscales, pixel formats, and frame rates of avionics display systems.

However, it is a potential problem for equipment venders hoping for interoperability. For that reason, each ARINC 818 serial interface is associated with an ICD. A particular piece of equipment compliant with ARINC 818 is not interoperable with another piece of equipment compliant with ARINC 818 *unless they share a common ICD.* Interfaces that are designed to a particular ICD *are* interoperable. The ICD will specify parameters, including link speed, image resolution, synchronization scheme, and frame rate. Typically, a military program or commercial avionics development program will adopt a particular ICD. The appendix to this guide gives a sample ICD using an XGA format for 24-bit color at 60 hertz. When specifying that equipment be ARINC 818–compatible, a particular ICD must be referenced.

ARINC 818 has application beyond its initial industry. ARINC 818 was developed specifically for commercial avionics, so aerospace designers have a natural choice to consider. ARINC 818's benefits, however, have also been proven in military aircraft programs. Many of the early adopters of its predecessor, FC-AV, were military aircraft programs, and ARINC 818 has followed suit in the last seven years. Furthermore, its flexibility offers significant benefits for *any* military display systems, whether on ground vehicles, aboard ships, or in ground-based command centers. It is being applied in ship-based programs. Can that flexibility contribute to advances in other industries—for example, in rapidly advancing programs in medical imaging? Very likely, and the arrival of ARINC 818-2 represents mature potential ready for additional applications.

2 Building blocks for ARINC 818

Fibre Channel is widely used and has been around for many years, so there are many vendors and devices applicable to ARINC 818 architecture.

However, some of these devices are less well suited for ARINC 818 because of unwanted complexity. This is because Fibre Channel primarily lives in the world of storage area networks (SANs). Since SAN equipment implements all aspects of FC-0 through FC-3, some devices are overkill for simpler ARINC 818 links.

ARINC 818 began as a unidirectional link with a simple well-defined Fibre Channel sequence. Specification 818-2 does provide a return path, but it is simply an independent ARINC 818 link. In contrast, SAN Fibre Channel equipment achieves bi-directionality using link initialization, port login, and more sophisticated sequences and exchanges that make sense for storage media such as SCSI protocol.

ARINC 818 designers have kept things simple and can continue to do so with 818-2. However, they should keep in mind that many silicon vendors are thinking SAN—not ARINC 818. Most of the Fibre Channel silicon vendors have chosen one of two product strategies. Some vendors offer complex Fibre Channel controllers with high-level functionality supporting the specific complexities of SAN equipment.

In contrast, others offer simple and flexible physical interface chips (PHYs) that are compatible with many high-speed serial interface standards, such as Gigabit Ethernet. PLD and FPGA vendors, recognizing the demand for high-speed serial interfaces, are also embedding PHYs in their products. Since the use of a PHY will require additional logic to implement ADVB, designers should first consider PLDs and FPGA with built-in PHYs.

2.1 PLDs and FPGAs

Xilinx and Altera dominate the landscape for PLDs, FPGAs, and CPLDs. Both vendors offer products with built in high-speed serial interfaces blocks. In most cases these are simple 8b/10b serializer/deserializers (SerDes) that can be used to fulfill existing serial standards, such as Fibre Channel, PCI Express, XAUI, SONET, Gigabit Ethernet, and SDI.

In a few cases the high-speed interface is more complex. The gigabit transceivers (referred to as GTs in Xilinx FPGAs and GXBs in Altera FPGAs) support a 16-bit wide interface or 32-bit wide interface, depending on the FPGA family. Fibre Channel uses this embedded functionality in CRC computation, elastic buffering, and EOF insertion. More common are the 16-bit wide interfaces, which require the user to design PLD logic to achieve FC-0, and FC-1 compatibility.

A few things are important in the selection of a device. First, to use an 8b/10b SerDes for ARINC 818, the SerDes must make certain control and status indicators available to the designer. Certainly, robust receivers should monitor and report on any 8b/10b disparity errors in the incoming stream.

Also, when transmitting, the running disparity of the link must be known to the supporting logic for the insertion of the proper EOF ordered sets. This EOF switching must be done in real time to comply with FC-0 and FC-1. If the disparity is not available, then only the physical interface of the block can be used, and an 8b/10b SerDes must be constructed within the PLD logic so that disparity can be tracked.

Depending on the synchronization scheme for a receiver, PLL clock recovery must be considered along with the availability of elastic buffers.

The following programmable devices have high-speed serial interfaces with the potential to support ARINC 818. The list is not exhaustive and is constantly evolving.

Table 2.1. A818-capable, high-speed programmable devices

Vendor	*Device family*	*Device*	*Max no. of transceivers*	*Max speed (Gb/s)*
Xilinx	Spartan 6	XC6SLX45T	4	3.2
Xilinx	Spartan 6	XC6SLX150T	8	3.2
Xilinx	Spartan 6	XC6SLX25T	2	3.2
Xilinx	Virtex 6	XC6VLX240T-1	24	5
Xilinx	Virtex 6	XC6VHX565T-3	48	6.6
Xilinx	Virtex 6	XC6VLX75T-3	12	6.6
Xilinx	Artix 7	XC7A200T	16	6.6
Xilinx	Artix 7	XC7A35T	4	6.6
Altera	Arria II GX	EP2AGX45DF29C5	8	3.75
Altera	Arria II GX	EP2AGX260F I3	16	6.375
Altera	Arria II GX	EP2AGX45DF29I3	8	6.375
Altera	Stratix 4	EP4SGX230KF40C2	36	3.375
Altera	Stratix 4	EP4SGX230NF40C2	48	3.375
Altera	Stratix 4	EP4SGX230DF40C2	8	3.375
Altera	Cyclone 5	5CGTFDE5F35C7N	12	6.144
Altera	Cyclone 5	5CSXFC6D6F31C8NCES	9	3.125
Altera	Cyclone 5	5CSXFC6C6F31C8NCES	6	3.125

2.2 Fibre Channel serializer/deserializers

There are several silicon vendors of stand-alone SerDes that can be used to build ARINC 818 links.

The features incorporated in these chips vary. The simplest among them will have no more than an 8b/10b encoder/decoder. The more sophisticated include the 32-bit wide organization of data, elastic buffers, and perhaps a CRC calculation.

Cypress Semiconductor, Vitesse Semiconductor, and Texas Instruments all have products. A few important considerations influence selection of a device. First, an 8b/10b SerDes for ARINC818 must make certain control and status indicators available to the designer. Robust receivers should monitor and report on any 8b/10b disparity errors. Also, when transmitting via an 8-bit wide interface, the running disparity of the link must be known to the supporting logic for the insertion of the proper EOF ordered sets.

Depending on the synchronization scheme for a receiver, PLL clock recovery must be considered along with the availability of elastic buffers.

2.3 Optical transceivers

There are numerous optical transceivers that can be used for ARINC 818. Any component compatible with Fibre Channel can be used. This includes 850nm multimode lasers for short haul (less than 500 meters) and 1310nm lasers for greater distances using single mode fiber.

The ICD should specify the media used for the implementation, whether it be single mode or multi-mode. It is not recommended that the fiber-optic transceivers designed for single mode applications be used with multi-mode fiber. Doing this risks modal cancellation that reduces optical power at the receiver.

3 ARINC 818 implementation–warm up

ARINC 818 links can be constructed from any 8b/10b SerDes that is Fibre Channel compatible. First a little background on the lowest layers of ADVB.

3.1 8b/10b encoding

At the lowest level, ARINC 818 uses 8b/10b encoding. This scheme is widely used in serial interfaces, and numerous references describe how it works.

In the simplest terms, it assigns two 10-bit codes to a single byte of data. An 8b/10b link keeps track of the running disparity of the serial transmission and seeks to balance the cumulative number of ones and zeroes transmitted. It does this choosing between the two 10-bit codes—with one having more ones than zeroes and the other more zeroes than ones. In some cases, the codes are neutral, with five ones in five zeroes.

8b/10b receivers check that the running disparity of the link is maintained. That is, the cumulative number of ones does not exceed the cumulative number of zeroes by more than one.

Moving up the layers of Fibre Channel, the link is organized in 32-bit chunks. Four 8b/10B codes are combined to make a 32-bit ordered set.

3.2 32-bit ordered sets

Ordered sets in Fibre Channel refer to 32-bit characters that demarcate such things as the beginning of a packet or the end of a packet. Ordered sets are also used as the stuffing characters in between packets.

Table 3.2. Ordered sets allowable under ADVB

Primitive signal	*Beginning RD*	*Ordered set*
SOF initiate Class 1 (SOFi1)	Negative	K28.5 D21.5 D23.2 D23.2
SOF normal Class 1 (SOFn1)	Negative	K28.5 D21.5 D23.1 D23.1
SOF initiate Class 3 (SOFi3)	Negative	K28.5 D21.5 D22.2 D22.2
SOF normal Class 3 (SOFn3)	Negative	K28.5 D21.5 D22.1 D22.1
EOF terminate (EOFt)	Negative Positive	K28.5 D21.4 D21.3 D21.3 K28.5 D21.5 D21.3 D21.3
EOF normal(EOFn)	Negative Positive	K28.5 D21.4 D21.6 D21.6 K28.5 D21.5 D21.6 D21.6
Idle	Negative	K28.5 D21.4 D21.5 D21.5
Low emissions idle*	Negative	K28.5 D20.4 D31.7 D31.7

All other ordered sets are ignored by the receiver.

***This item is also known as a Fibre Channel ARB(ff) and is an alternative form of idle. Usage results in minimization of EMI within a system.**

Here is a good place to demystify the shorthand used in Fibre Channel. For instance, the designation K28.5 is shorthand for the 8-bit code 10111100. To determine the code for a special character byte, simply take the last five bits of the eight bits, in this case 11100, which has the decimal value 28 and concatenate that with the decimal value of the first three bits, in this case 101 or 5. The preceding K designates that this character is a special character, rather than a data character.

A code for a data byte is constructed in the same way. For instance, 10110101 has the last five bits 10101 or 21, and the first three bits 101 or 5. Therefore, the shorthand value becomes D21.5.

All ordered sets begin with one special character followed by three data characters.

3.3 ADVB frames

A packet of data in Fibre Channel is referred to as a *Fibre Channel frame*. Note that the isolated use of the word *frame* risks confusion when dealing with ARINC 818. When applying the conventional meaning (as in Figure 1.3), use *video frames*. When referring to Fibre Channel packets, specify *FC frames* or *ADVB frames*. (The ARINC 818 Spec also refers to FC frame as ADVB frames. In reference to ARINC 818, as in this guide, the phrases *ADVB frames* and *FC frames* can be used interchangeably.)

Since the basic chunk in Fibre Channel is 32 bits wide, the number of bytes in an ADVB frame is always divisible by four.

Figure 3.3 shows the basic construction of all ADVB frames. ARINC Specification 818 and 818-2 show a more complex version of this figure, as it originated in Fibre Channel–Framing and Signaling Interface (FC-FS) *(ANSI / INCITS 373-2003)*. It is simplified for ADVB since there are no optional headers allowed.

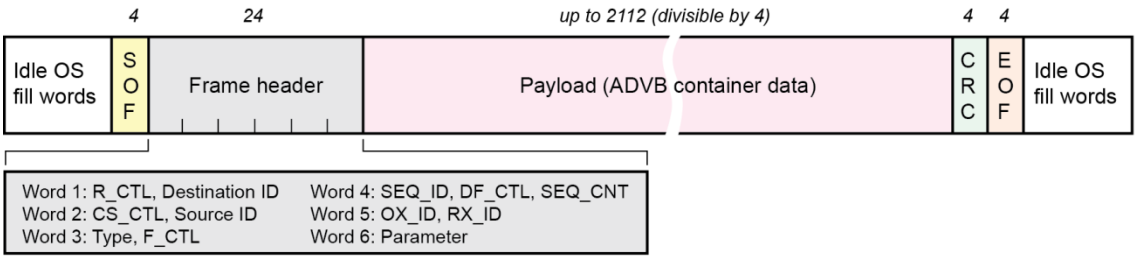

Figure 3.3. ADVB frame structure with bytes per segment.

3.4 The ADVB container

In most implementations, an ADVB container is simply a set of FC frames that make up one video frame. It's as simple as that. A single container maps exactly into a single FC sequence. Therefore, an FC sequence, a container, and a video frame, are effectively one in the same. (Exception: There are possible implementations with more than one video frame per container.)

Within a container are *objects*. Simply put, objects contain certain types of data. Object 0 is intended for header data. Object 1 contains audio information (and is not employed in most implementations), and Object 2 and Object 3 contain video payload:

Object 0: Ancillary data; object type number 5xh
Object 1: Audio data; object type number 4xh
Object 2: Video data; object type number 1xh | 2xh
Object 3: Video data; object type number 1xh | 2xh

For interlaced formats, Object 2 carries even-field data, and Object 3 carries odd-field data. For progressive scans—employed in most implementations—only Object 0 and Object 2 will be used. Object 0 requires a single ADVB frame. Object 2 will require as many ADVB frames as needed to move a single video frame over the link.

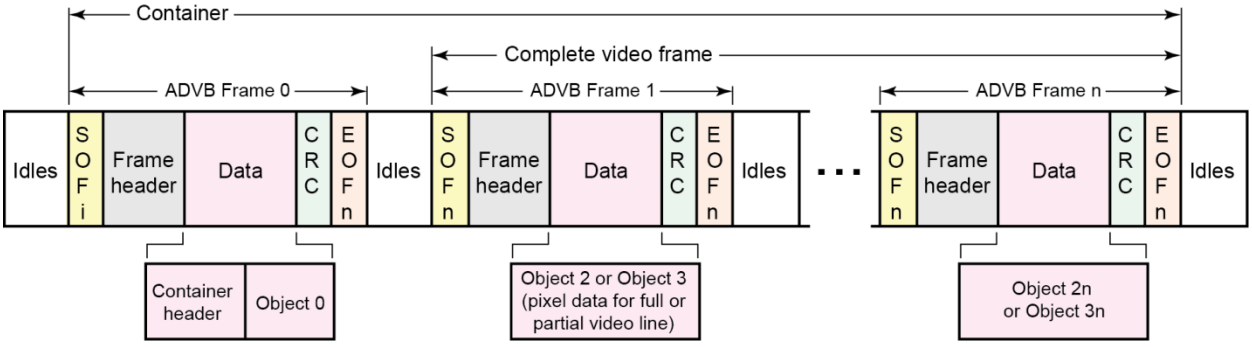

Figure 3.4. The Fibre Channel sequence and its relationships with a container, a video frame, and ADVB frames. Again, ARINC 818 nomenclature differentiates between video frames and ADVB frames.

3.5 Vertical and horizontal line timing

The ARINC 818 container rate is the same as the vertical frame rate of the transported video. The horizontal line timing may or may not be preserved in the transmission. This

depends on what is specified in the ICD. In some implementations, the ICD will have strict requirements on the timing of ADVB frame transmission, and this timing will correspond to the horizontal line timing of the video transmitted. These implementations are referred to as line-synchronous. In line-synchronous implementations, the Object 0 ADVB frame occurs during vertical blanking, and ADVB frames carrying line data are sent at the horizontal line rate.

3.6 Synchronization and buffering issues

The flexibility of ARINC 818 allows receiver implementations using either full image buffers or only line buffers. Line-buffer implementations may be desirable for cockpit display systems since they eliminate the dangerous possibility of stale video lingering in the display unit.

For either type of receiver design, synchronization issues must be considered at the pixel, line, and frame levels. At the pixel level, one solution would be to use the Fibre Channel recovered clock as an FPGA global clock. However, there are many advantages to having a local clock independent of the link—including clock stability if the Fibre Channel link goes off line for any reason. In addition, display devices will most likely need pixel-clock frequencies that are different from the Fibre Channel–clock frequency. The choice of an FPGA global clock frequency may be made with a display interface in mind, or perhaps with respect to a needed memory or bus interface.

In cases where the local clock runs asynchronously, designs may leverage elastic buffers found within the SerDes to cross the clock domain. The GTs/GXBs may include elastic buffers for this purpose.

However, even after pixels have been synchronized to the local clock, synchronization at the line and frame level may be required. For example, a receiver design with full-frame buffers that needs to drive a display at a set vertical rate will need to address vertical synchronization. Using a ping-and-pong image store scheme, the receiver will periodically need to drop or repeat a video frame presented to the display. This is because of the free drift between the incoming and outgoing vertical syncs. Switching between the ping and the pong image stores can only occur when the vertical syncs are aligned. However, the incoming frame sync based on the Fibre Channel clock will be asynchronous to, and will not stay aligned with, the outgoing sync based on the FPGA clock even though the two vertical rates are very close.

ARINC 818 supports various synchronization schemes. Line-buffer or FIFO-based receiver designs will require that the transmitter adhere to the strict line timing requirements of the display. Since horizontal scanning must be precise for some displays, the arrival times of lines will also need to be precise.

ARINC 818 intends timing requirement such as this be captured in an ICD specific to the video system.

3.7 Interoperability considerations

In the interest of supporting ADVB designs that are interoperable, designers should be mindful of the many parameters that determine interoperability. In ARINC 818, designers choose parameters as they see fit. In many instances, the choices will have little consequence for the functionality of the link. However, they will determine the degree to which ARINC 818 equipment is interoperable. This section discusses these choices.

3.7.1 Physical medium

The ARINC 818 Specification does not specify the ADVB physical medium. The ADVB physical layers are defined in the ICD. However, ARINC 818-2 does include an expanded Appendix B, where fiber optic parameters that must be considered are defined.

Developers should consider 850nm FC-compatible transceivers for any connection less than 500 meters. These are low cost and available through numerous vendors.

For systems with connections expected to exceed 500 meters, designers should consider 1310nm FC-compatible transceivers. For 1.0625 Gb/s interfaces, copper interfaces using differential twinax can be used. With these choices, designers will be able to leverage existing Fibre Channel development tools such as protocol analyzers and traffic generators.

Designers of equipment for avionics should consult the following ARINC Standards when using fiber optic:

- ARINC Specification 801-3: *Fiber Optic Connectors*
- ARINC Specification 802-2: *Fiber Optic Cables*
- ARINC Specification 803-3: *Fiber Optic Design Guidelines*
- ARINC Specification 804-1: *Fiber Optic Active Device Specification*
- ARINC Specification 805-3: *Fiber Optic Test Procedures*
- ARINC Specification 806-4: *Fiber Optic Installation and Maintenance*

3.7.2 Link speed

These link speeds, in gigabits per second (Gb/s), are used in ARINC 818- 2:

1.0625	FC base rate	
1.5		
1.62		
2.125	FC 2x rate	
2.5		
3.1875		
4.25	FC 4x rate	
5.0		Introduced in ARINC 818-2
6.375	FC 6x rate	Introduced in ARINC 818-2
8.5	FC 8x rate	
12.75	FC 12x rate	Introduced in ARINC 818-2
14.025	FC 16x rate	Introduced in ARINC 818-2
21.0375	FC 24x rate	Introduced in ARINC 818-2
28.05	FC 32x rate	Introduced in ARINC 818-2

The 6x, 12x, and 24x rates accommodate high-speed, bi-directional coax with power as a physical medium. The 5.0 Gb/s rate handles implementations using specific speeds by certain FPGAs.

Although the in-between rates and are certainly permissible, ARINC 818 implementations, where possible, should stick to rates evenly divisible by the base rate. By doing this, designers will be able to leverage existing Fibre Channel development tools, such as protocol analyzers and traffic generators. Also, designers will eliminate any potential hazards when using Fibre Channel chips and transceivers not optimized for non–Fibre Channel rates.

In addition to these rates, an ICD can specify a specific low-speed rate for return paths—a feature introduced in ARINC 818-2. For example, a camera might have a control link requiring a rate well below the FC base rate.

Rates of 12x or above will use 64b/66b encoding rather 8b/10b.

3.7.3 Choice of class

As mentioned, ARINC 818 allows the use of either Class 1 or Class 3 ordered sets. There may be arguments for using one class of ordered sets over another, but in most cases, it will make little difference—except for interoperability. In the interest of interoperability (and where the choice has little bearing on the system design), Class 1 ordered sets are recommended.

Generic receivers should be constructed such that they can operate with either class of ordered set. Transmitters should have a built-in capability to easily switch between the two classes.

3.7.4 Video payload segmentation

ARINC 818 places no constraints on how video data is loaded into ADVB frames. As long as the number of bytes is divisible by four and doesn't exceed the maximum count of 2112 bytes, any method of loading is permissible.

However, since it is foreseeable that avionics displays may rely on line-buffered receivers (rather than image buffered receivers), it is recommended that video lines be loaded into ADVB frames in a standard way.

A simple formula is to load as many integral lines as will fit into the payload limit. In cases where the video lines are larger than the payload limit, the rule is to insert half lines, quarter lines, etc. into ADVB frames. A simple Excel spreadsheet that calculates these parameters automatically based on this simple rule can be found at www.arinc818.com.

Requirements for segmentation must be captured in the ICD.

3.7.5 Synchronization and segmentation classes

ARINC 818 places no constraints on the timing of the ADVB frames during transmission. However, ARINC 818-2 Appendix C does establish synchronization segmentation classes.

To understand these classes, first consider four separate variables that characterize the delivery of video over the ADVB:

- Vertical timing – do entire video frames arrive at a set rate?
- Horizontal timing – do video lines arrive at a set rate?
- The uniformity to which line data is loaded in packets (segmentation); some classes only separate the Object 0 into a separate ADVB frame)
- The variance in arrival time (or jitter) for video line data

The table in Appendix C of the ARINC 818-2 specification begins with the least constrained ADVB case (A1 with no vertical or horizontal timing imposed and no special segmentation) and moving to the most constrained case (D3 with perfect vertical and horizontal timing and orderly line segmentation into frames). Table 3.7.5 summarizes the classes by these parameters.

Table 3.7.5. Characteristics of ordered sets

	Non–line-synchronous classes						*Line-synchronous classes*				
	A1	*A2*	*A3*	*B1*	*B2*	*B3*	*C1*	*C2*	*D1*	*D2*	*D3*
Stable vertical period?	no	no	no	yes	yes	yes	yes	yes	yes	yes	yes
Stable horizontal period?	no	no	no	no	no	no	yes	yes	yes	yes	yes
Object 0 or line segmentation?	n/a	Obj0	line	n/a	Obj0	line	line	line	line	line	line
Vertical jitter?	yes	yes	yes	yes	yes	yes	yes	ext. sync	yes	yes	no
Horizontal jitter?	yes	yes	yes	yes	yes	yes	yes	ext. sync	no	no	no

So what class should be picked for an ARINC 818 implementation? *For most systems, Great River Technology recommends C1 designs.* This offers enough constraint for the receivers to handle the video without overburdening the transmitter design.

However, some systems may require more constraints than others. For instance, some avionics programs will be concerned about hazardous stale video, so the use of full-image buffers will be prohibited. In this case, ADVB receivers will have only line FIFOs to buffer incoming data. These types of implementations are referred to a line synchronous. Classes C1, C2, D1, D2, and D3 are considered line synchronous.

Classes A1, A2, A3, B1, B2, and B3 are not line synchronous and will require that the receiver have full-image buffers for the incoming video.

Line-synchronous implementations require that video on the ADVB have vertical and horizontal timing embedded into the ADVB. Basically, this means that the transmitter will need to control the number of idle ordered sets between ADVB frames.

3.7.6 Interoperable transmitters

Although there may not be a requirement for horizontal line timing for a particular ADVB implementation, designers may opt for increased interoperability by employing horizontal

timing on the transmission. Transmitters designed to a more constrained class on the above table (further to the right), will be compatible with more receivers. That is, line-synchronous transmitters should be compatible with receivers designed for the same line-synchronous class, and in addition, be compatible will receivers designed for all less constrained non–line-synchronous classes.

Even if the targeted system does not require precise timing for the horizontal line rate, adding this to the ARINC 818 transmitter design will greatly improve the likelihood that the transmitter is compatible with other ARINC 818 receivers not designed to the same ICD.

A simple Excel spreadsheet that calculates these parameters automatically for line-synchronous implementations can be found at www.arinc818.com.

3.7.7 Interoperable receivers

Designers of ADVB receivers can also make certain choices in their designs that increase the likelihood of interoperability with other ARINC 818 equipment.

Mostly these choices fall in to the realm of the synchronization scheme used. As discussed above, there are many synchronization schemes possible for an ARINC 818 receiver.

For avionics displays that have concerns about hazardous stale video, the use of full-image buffers may be prohibited and receivers will use only line FIFOs. This will require that the transmitter maintain precise horizontal timing. Line buffered receivers will not be operable with any transmitter that does not obey the strict horizontal timing requirements.

For receivers not under this requirement, a better approach is to include full-image buffers. These designs will be operable with more transmitter classes—even with non-uniform or non-existent horizontal timing. Receivers designed to this less constrained class (further to the left in Table 3.7.5), will be compatible with more transmitters. That is, non–line-synchronous receivers will be compatible with transmitters designed for the same class and, in addition, will be compatible with transmitters designed for all line-synchronous classes.

3.8 Before you begin

3.8.1 Helpful reading

The Fibre Channel family of ANSI standards is considered an integral part of ARINC Specification 818-2 and is a valuable reference. The Fibre Channel-Audio Video standard (referenced below) is the primary document since it deals with protocol. The title is shortened to simply FC-AV in this Implementer's Guide.

Fibre Channel–Audio Video (FC-AV) (*ANSI INCITS 356-2002,* 25 Nov 2002)

Fibre Channel–Framing and Signaling Interface (FC-FS) (*ANSI / INCITS 373-2003*)

Fibre Channel–Physical Interfaces (FC-PI) (*INCITS 352-2002*)

Related ARINC documents that are valuable to review are:

ARINC Specification 801-3: *Fiber Optic Connectors.*

ARINC Specification 802-2: *Fiber Optic Cables.*

ARINC Specification 803-3: *Fiber Optic Design Guidelines.*

ARINC Specification 804-1: *Fiber Optic Active Device Specification.*

3.8.2 Creating an ICD

All ARINC 818 designs should be based on a complete ICD. For new systems, this will be the first task to undertake. ARINC Specification 818-2 makes recommendations for what an ICD should contain. It is a long list, and not every parameter is necessarily required. To help simplify this task, Appendix A is a sample ICD that can be used as a template.

3.8.3 Great River's ADVB parameter calculator

An Excel spreadsheet that calculates parameters for line-synchronous implementations can be found at www.arinc818.com.

This tool also generates a graphical representation of the ADVB frame sequence for the particular resolution and video frame rate. This graphic includes all relevant timing for the line-synchronous sequence of ADVB frames.

The calculator can then be used to determine parameters that best approximate these timings for a particular link speed, resolution, and pixel format.

The calculator can be found at http://www.arinc818.com/arinc-818-library.html

4 ARINC 818 transmitter design

This section describes in more detail an ARINC 818 transmitter design for progressive-scan video. The steps described below could be implemented in the logic of an FPGA or PLD with an internal 8b/10b serializer or a serializer external to the programmable device.

The basic job of the transmitter will be to create ADVB frames meeting the requirements of ARINC 818 and of the associated ICD.

4.1 ADVB frame creation

As shown in Figure 3.3, ADVB has a simple frame structure. In progressive scan implementations, where video line data is loaded into frames in a uniform way (such as one half line per ADVB frame as specified in the ICD of Appendix A), there are only two basic frame structures: one for the Object 0 ADVB frame and one for all other payload ADVB frames.

The Object 0 frame structure is:

SOFi:	4 bytes (1 long word [LW])
Frame header:	24 bytes (6 LWs)
Container header:	88 bytes (22 LWs)
Ancillary data:	16 bytes (4 LWs)
CRC:	4 bytes (1 LW)
EOFn:	4 bytes (1 LW)

All Object 2 payload frames take on the following structure:

SOFn:	4 bytes (1 LW)
Frame header:	24 bytes (6 LWs)
Video payload:	the size in bytes specified by the ICD
CRC:	4 bytes (1 LW)
EOFn/t:	4 bytes (1 LW)

The basic job of the transmitter will be to create and transmit these frames per the timing requirements of the ICD.

ADVB frames can be created with the steps in Sections 4.4.1–4.4.*.

4.1.1 Insert FC ordered sets

When the transmitter is not transmitting ADVB frames, it is transmitting idle ordered sets continuously. The transmitter must insert a minimum of six idle ordered sets between any two ADVB frames. Typically, there are many more idle ordered sets stuffed in the vertical blanking period as a way to adjust the video frame rate to a desired period.

ADVB allows either of two versions of the idle ordered set:

Idle:	K28.5	D21.4	D21.5	D21.5
Low emissions idle:	K28.5	D20.4	D31.7	D31.7

The first in the list is the most pervasive among Fibre Channel equipment in existence today. It is the original idle ordered set defined for Fibre Channel.

The second is a more recently introduced idle ordered set that has improved characteristics for EMI and is known as the ARB(ff). The lower EMI quality of the ARB(ff) may have little benefit for ADVB interfaces, but does have benefits for other Fibre Channel applications and may be the more common idle ordered set in future Fibre Channel equipment. ARINC 818 allows either to be used.

For interoperability and longevity of a design, ADVB receivers should be constructed such that they are compatible with either idle ordered set version. Transmitters should have a built-in capability to easily switch between the two. Designers should investigate whether 32-bit wide decoders can accommodate either ordered set. (Some may have logic that is hard coded for the original idle ordered set.)

The first step in the ADVB frame construction is to write the SOF to the 8b/10b encoder.

SOF initiate class 1 (SOFi1):	K28.5	D21.5	D23.2	D23.2
SOF normal class 1 (SOFn1):	K28.5	D21.5	D23.1	D23.1

The Object 0 ADVB frame begins with the SOFi, which denotes that it is the first ADVB frame of the sequence. For ADVB it indicates the first ADVB frame of the video frame (or container).

All subsequent payload frames will begin with the SOFn

Therefore, the Obj0 ADVB frame will begin:

BC 95 B5 B5 (idle ordered set [OS])
*
*
BC 95 B5 B5 (idle OS)
BC 95 B5 B5 (idle OS)
BC 95 B5 B5 (idle OS)
BC B5 57 57 (SOFi)
*
*

All other payload frames will begin:

BC 95 B5 B5 (idle OS)
*
*
BC 95 B5 B5 (idle OS)
BC 95 B5 B5 (idle OS)
BC 95 B5 B5 (idle OS)
BC B5 37 37 (SOFn)
*
*

4.1.2 Insert ADVB frame header fields

Next in the sequence, the transmitter will need to insert the 6 long words (24 bytes) of ADVB frame header data (Table 4.1.2).

Table 4.1.2. ADVB frame header long-word (LW) fields

	Identifier	*Byte 0*	*Byte 1*	*Byte 2*	*Byte 3*
LW 1	Frame header	0100 0100 (R_CTL)	0000 0000 (dest. ID)	0000 0000 (dest. ID)	xxxx xxxx (dest. ID)
LW2	Frame header	0000 0000 (reserved)	0000 0000 (source ID)	0000 0000 (source ID)	xxxx xxxx (source ID)
LW 3	Frame header	0110 0000 60h=FC-AV	0011 w000 (Bit 19=END_SEQ) (F_CTL)	0000 000x (F_CTL)	0000 x0xx (F_CTL)
LW 4	Frame header	xxxx xxxx (SEQ_ID)	0000 0000 (DF_CTL)	xxxx xxxx (SEQ_CNT)	xxxx xxxx (SEQ_CNT)
LW 5	Frame header	1111 1111 (OX_ID)	1111 1111 (OX_ID)	1111 1111 (RX_ID)	1111 1111 (RX_ID)
LW 6	Frame header	xxxx xxxx (parameter)	xxxx xxxx (parameter)	xxxx xxxx (parameter)	xxxx xxxx (parameter)

A description of these fields can be found in ARINC Specification 818-2. For the transmitting hardware, they may be entered as constants in the PLD's non-volatile memory or, for a more flexible design, they may be loaded by associated software at initialization.

Some of the fields are static, and some need to be dynamically changed by the transmitter (Table 4.1.2 above).

4.1.3 Static fields

The static fields are typically stored in memory in the transmitter design. They are read from memory and inserted in the same way that video payload is inserted. The static fields contain the same parameter values over the entire set of ADVB frames. Therefore, they

may be accessed during initialization and stored locally for insertion into all outgoing ADVB frame headers.

4.1.4 Dynamic fields

The dynamic fields are as follows:

SEQ_CNT—This requires a 24-bit counter in the design. The count value always starts with 000000 in the Object 0 ADVB frame and increments for every subsequent ADVB frame in the sequence. (In this case a sequence is the same as a container or a single video frame.)

SEQ_ID—This parameter takes on the lowest order bytes of the container count (another counter to be discussed shortly). This value is constant over the sequence (or single video frame) but increments on subsequent video frames.

F_CTL Bit 19—The 24-bit F_CTL field is static except for Bit 19, which is set to “1” in the very last ADVB frame of the sequence. Bit 19 is “0” for all other ADVB frames. This is a holdover from Fibre Channel as a way to signal the receiver that the last ADVB frame has been received—but nevertheless is required by ADVB.

The dynamic values will need to be updated by the hardware and multiplexed into the data path.

4.1.5 Container header and ancillary data

The container header of twenty-two 32-bit words (88 bytes) will typically be stored in memory in the transmitter design (Table 4.1.5).

Table 4.1.5. ADVB container header and ancillary data long-word (LW) fields.

	Identifier	*Byte 0 (MSB)*	*Byte 1*	*Byte 2*	*Byte 3 (LSB)*
LW 0	Container count	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx
LW 1	Clip ID	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx
LW 2	Container time stamp	0000 0000	0000 0000	0000 0000	0000 0000
LW 3	Container time stamp	0000 0000	0000 0000	0000 0000	0000 0000
LW 4	Transmission type	xxxx xxxx (video frame rate)	xxxx xxxx (transmission rate)	0000 0000 (reserved)	0000 0000 (reserved)
LW 5	Container type	0000 0000 (mode)	0000 0100 (number of objects)	0000 0000 (reserved)	0000 0000 (size of ext. header)
LW 6	Object 0 class	0101 xxxx (type ancillary)	0000 0000 (link pointer)	1101 0000 (SPDV index)	0000 0000 (SPDV index)

(continued)

Table 4.1.5 *continued*. ADVB container header and ancillary data long-word (LW) fields.

	Identifier	*Byte 0 (MSB)*	*Byte 1*	*Byte 2*	*Byte 3 (LSB)*
LW 7	Object 0 size	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx
LW 8	Object 0 offset	0000 0000	0000 0000	0000 0000	0101 1000
LW 9	Object 0 object type defined	0000 0000	0000 0000	0000 0000	0000 0000
LW 10	Object 1 class	0100 0000 (type audio)	0000 0000 (link pointer)	1101 0000 (SPDV index)	0000 0000 (SPDV index)
LW 11	Object 1 size	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx
LW 12	Object 1 offset	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx
LW 13	Object 1 object type defined	0000 0000	0000 0000	0000 0000	0000 0000
LW 14	Object 2 class	0001 xxxx (type video)	0000 0000 (link pointer)	1101 0000 (SPDV index)	0000 0000 (SPDV index)
LW 15	Object 2 size	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx
LW 16	Object 2 offset	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx
LW 17	Object 2 object type defined	0000 0000	0000 0000	0000 0000	0000 0000
LW 18	Object 3 class	0001 xxxx (type)	0000 0000 (link pointer)	1101 0000 (SPDV index)	0000 0000 (SPDV index)
LW 19	Object 3 size	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx
LW 20	Object 3 offset	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx
LW 21	Object 3 object type defined	0000 0000	0000 0000	0000 0000	0000 0000

This data can be read from memory and inserted directly into the Object 0 ADVB frame—with one exception. The container count will need to be incremented for each transmitted video frame. Therefore, the transmitter requires a 32-bit counter for this purpose.

The container count is an unsigned integer, starting at 0. The container count will be incremented by one LSB for each subsequent container (or video frame), and will wrap around to a count of 0 when an overflow occurs.

The lower byte of the container count should be used for the SEQ_ID value in the ADVB frame header.

Immediately after the container header will be the ancillary data of 16 bytes (four 32-bit words). This data can also be read from memory and inserted directly into the Object 0 frame.

Ancillary data defines the characteristics of the data being transferred from the sender to the recipient. In most cases the ancillary data will be 16 bytes (4 words); however, there

are provisions within ARINC 818 for an extended Object 0 to accommodate color palette data and cursor control. (Refer to the specification for details on extended ancillary data.)

The four words of ancillary data are:

Word 0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Number of rows														Number of columns														Frame/field			

Word 1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CI[1]				P[2]		PA[3]			PAO[4]				PTN[5]			Bits/Subpixel A				Bits/Subpixel B				Bits/Subpixel C				Bits/Subpixel D			

[1] Color information type
[2] Prior CRC valid flag
[3] Pixel aspect ratio (width:height)
[4] Pixel array order
[5] Packing table number

Word 2

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Prior image CRC																															

Word 3

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Parameter 2 type						Parameter 2 data										Parameter 1 type						Parameter 1 data									

4.1.6 Video payload insertion

The video payload is typically stored in memory in the transmitter design. Video data is read from memory and inserted directly—or perhaps after the pixel data is correctly packed. For example, 24-bit RGB pixels will need to be combined to form 32-bit words written to the serializer.

Where integer numbers of lines are inserted into frames (or perhaps half lines), the transmitter will require a counter that can track the exact payload size. In other words, the hardware will need to manage the precise number of memory accesses and the number of writes to the serializer to achieve the correct integer number of lines.

4.1.7 CRC insertion

While the header data and all payload data is being written to the serializer, the hardware must compute a 32-bit CRC for the outgoing ADVB frame. The CRC is calculated on all data after the SOFi or SOFn and before the CRC.

The CRC computations will use the following 32-bit polynomial:

$$X32 + X26 + X23 + X22 + X16 + X12 + X11 + X10 + X8 + X7 + X5 + X4 + X2 + X + 1$$

A helpful guide for the Fibre Channel CRC computation method is the FC-FS (*ANSI/INCITS 373-2003*).

4.1.8 EOF control

After the CRC is inserted, the transmitter must insert the EOFn or EOFt

EOF Terminate (EOFt)	Beg RD (negative)	K28.5 D21.4 D21.3 D21.3
	Beg RD (positive)	K28.5 D21.5 D21.3 D21.3
EOF Normal (EOFn)	Beg RD (negative)	K28.5 D21.4 D21.6 D21.6
	Beg RD (positive)	K28.5 D21.5 D21.6 D21.6

The EOFn is used to terminate all ADVB frames except for the last in the sequence—where the EOFt is used. Therefore the transmitter multiplexes in the EOFt once every video frame on the very last payload packet.

Easy enough—but there is one more trick.

ARINC 818, since it is based on Fibre Channel, requires that one of two EOF codes be inserted at the end of each ADVB frame to drive the link disparity to negative such that only one IDLE OS disparity version is used. The IDLE OS with beginning running disparity positive is not defined for Fibre Channel and the existence of this OS in the link can lead to errors when using some Fibre Channel receivers.

For instance, the GTP and Altera GXB transceivers are fully Fibre Channel–compliant and use the occurrence of the negative IDLE OS (that is, the idle ordered set version with beginning running disparity negative) for two things:

- Correct start/termination of the CRC calculation
- Re-centering of the elastic buffer in receiver

Incorrect EOF insertion leads to positive IDLE OS on the link, which causes errors in the CRC calculation and the errors in the elastic buffer.

To understand the effect of EOF insertion, consider an idle ordered set *K28.5, D21.4, D21.5, D21.5*, which begins with comma character *K28.5* and has this 32-bit code:

BC 95 B5 B5 (1011 1100 1001 0101 1011 0101 1011 0101)

Corresponding to this on the serial link are two possible 40-bit codes. When the beginning running disparity is negative (allowed by Fibre Channel), this is the result:

	K28.5		D21.4		D21.5		D21.5	
	001111	1010	101010	0010	101010	1010	101010	1010
(RD-)		(RD+)		(RD-)		(RD-)		(RD-)

When the beginning running disparity is positive (*not* allowed by Fibre Channel), this is the result:

	K28.5		D21.4		D21.5		D21.5	
	110000	0101	101010	1101	101010	1010	101010	1010
(RD +)		(RD -)		(RD+)		(RD+)		(RD+)

This non-compliant ordered set ends with positive running disparity and therefore all following ordered sets will be the same.

For a transmitter to be Fibre Channel–compliant it must never send this second case. To do this, the beginning running disparity must be negative immediately after the last

character of the EOF. If not, then the non-compliant positive idle ordered sets are sent. Because the running disparity is random at the end of the packet (the last byte of the CRC), Fibre Channel defines two possible EOF ordered sets. One of these is neutral—that is, it will not change the running disparity after it is inserted. Therefore, this one is used when the beginning running disparity is negative.

The negative EOF ordered set is to be used when the running disparity is positive. This ordered set will change the running disparity back to negative. The transmitter must switch between these two possible EOF versions based on the beginning running disparity.

4.2 XGA color progressive example

Table 4.2 XGA video format

Parameter	Value	Ancillary data value (hex)
Video frame rate	15 to 60 Hz	14
Number of rows	768	300
Number of columns	1024	400
Frame or field based	Frame	0
Color information	RGB	1
Packing table number	32-bit	0
Bits per sub-pixel	8:8:8:0	8880
Video lines per ADVB frame	0.5	na
ADVB frames per video frame	1536	na

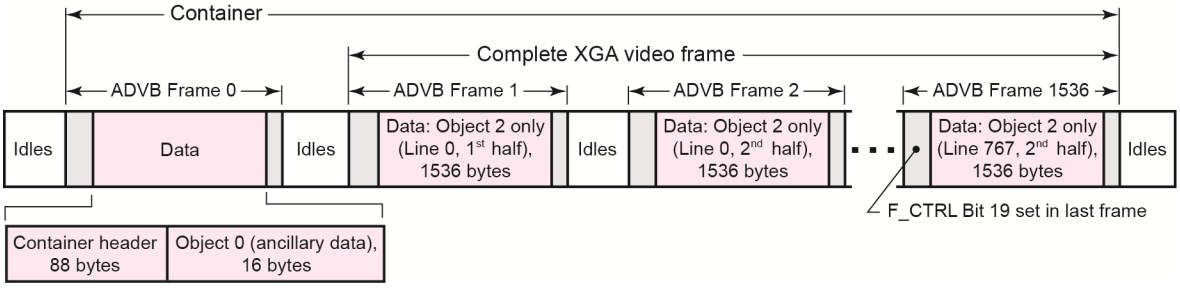

Figure 4.2. ADVB frame sequence for XGA in RGB color.

Appendix: Sample ICD

ADVB Interface Control Document

- **XGA**
- **24-bit color**
- **Progressive scan**
- **60 hertz**

1 General

This Interface Control Document (ICD) is to be used in conjunction with ARINC Specification 818 for the implementation of an ADVB link. This ICD will place additional constraints over the ADVB implementation for several parameters such as link speed, packetization, and video timing.

1.1 References and document precedence

The primary references for this ICD are:

- *ARINC Specification 818* or
- *ARINC Specification 818-2*

All standards on which ARINC Specification 818 or ARINC Specification 818-2 relies, namely The Fibre Channel family of ANSI standards, also apply. These include:

- Fibre Channel–Audio Video (FC-AV) (*ANSI INCITS 356,* 2002)
- Fibre Channel–Framing and Signaling Interface (FC-FS) (*ANSI INCITS 373,* 2003)
- Fibre Channel–Physical Interfaces (FC-PI) (*ANSI INCITS 352*, 2002)

1.2 Document precedence

Where requirements are not stated explicitly within this ICD, the requirements set forth in ARINC Specification 818 or ARINC Specification 818-2 will govern.

2 ADVB Requirements

2.1 Physical media

This section may place constraints on the physical media used such as:

- Optical transceiver requirements for wavelength, power, and sensitivity
- Cabling types, such as multi-mode fiber or copper twinax
- Connector types

2.2 Link characteristic

The ADVB shall be a single channel, 2.125 gigabits per second interface.

2.3 Video format

The ADVB shall have the following video format:

Resolution: 1024 pixels x 768 lines
Scan: Progressive (left to right, top to bottom)
Frame rate: 60 hertz
Pixel format: 24-bit RGB (8:8:8)

2.4 Audio capabilities

The ADVB will not include capabilities for transporting audio (container Object 1)

2.5 ADVB Frame Segmentation

All transmitted ADVB frames shall conform to the following segmentation rules for Object 0 and Object 2 video payload:

- Single video stream (single container)
- Object 0 segmented in the first transmitted ADVB frame in each Video frame
- Object 2 frames
  - 2048 ADVB Frames total
  - All frames shall contain exactly one half line (512 pixels)

2.6 ADVB frame timing

All transmitted ADVB frames shall conform to the following timing:

32-bit character time (ns):	18.82352941
Bytes per video line:	3072
Lines per ADVB frame:	0.5
ADVB frame payload size (bytes):	1536
Number of FC Object 2 frames:	1536

Line-synchronous timing shall be achieved using the following parameters:

Inactive lines (vertical blanking):	27
Pre–Object 0 inactive lines:	23
Post–Object 0 inactive lines:	4
Horizontal line time (µs):	20.93176471
Horizontal line rate (kHz):	47.77428058
***Actual frame rate (Hz) with synced lines*:**	**60.09343469**

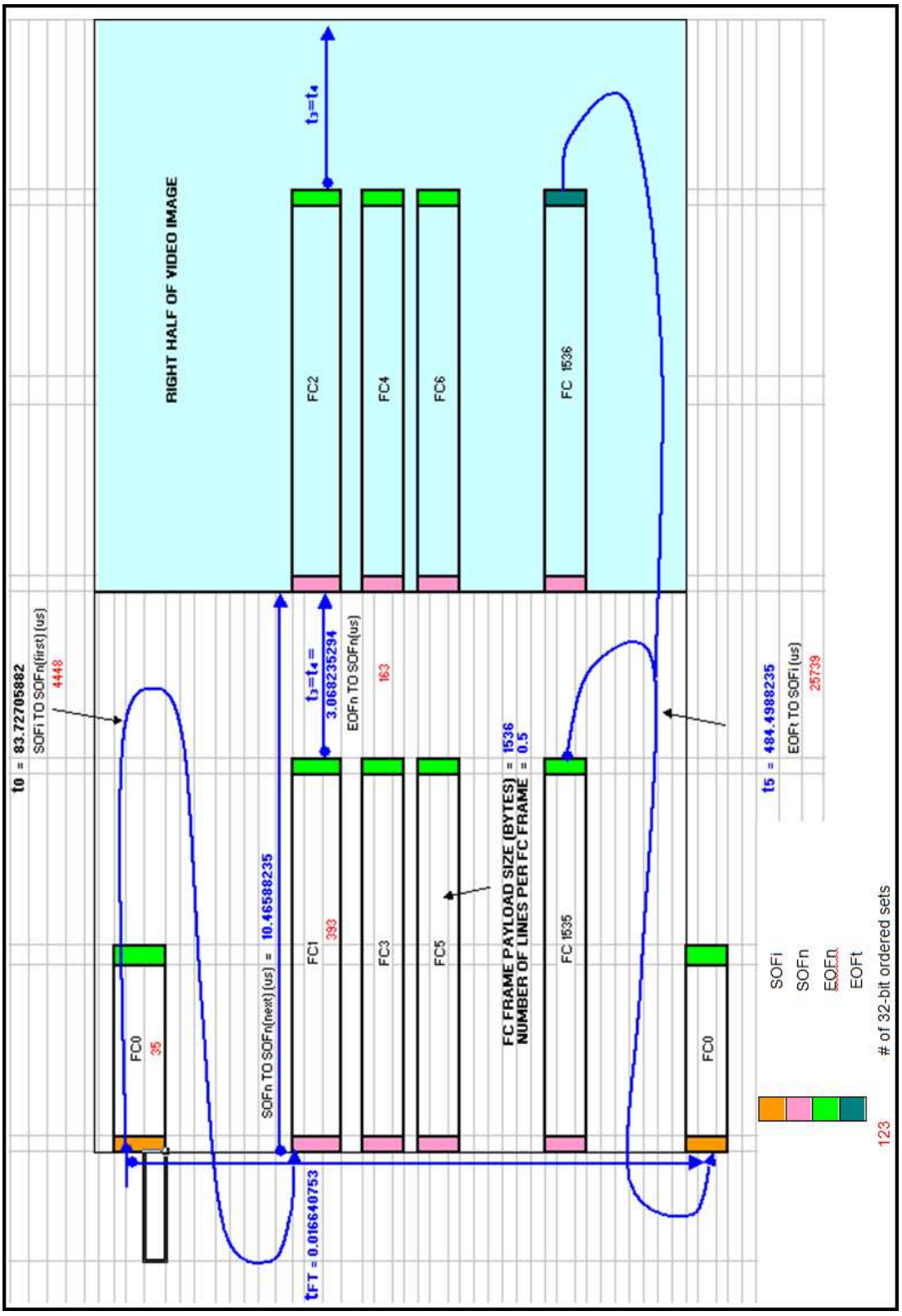

Figure 2.6. Timing diagram.

These parameters lead to the times in Table 2.6.

Table 2.6: ADVB frame timing.

	*µs or [ms]**	*32-bit character count*	*Horizontal lines*
Obj0 ADVB frame time	0.658823529	35	na
Obj2 ADVB frame time	7.397647059	393	na
SOFi to SOFi **(t_{FT})**	[16.64075294]	884040	795
SOFi to SOFn (first) **(t_0)**	83.72705882	4448	4
SOFn to SOFn (next)	10.46588235	556	0.5
SOFn (first) to SOFn (last)	[16.07559529]	854016	768
SOFn (last) to SOFi	491.8964706	26132	23.5
Video line time **(t_{LT})**	20.93176471	1112	1
EOFn to SOFn **(t_3,t_4)**	3.068235294	163	na
EOFt to SOFi **(t_5)**	484.4988235	25739	na

Times shall be achieved to +/- 6 32-bit character times.

2.7 Object 0 ADVB frame header

The Object 0 ADVB frame shall use the header values in Table 2.7.

Table 2.7: Object 0 ADVB frame header values.

Word	*Byte 0*	*Byte 1*	*Byte 2*	*Byte 3*
1	44	00	00	00
2	00	00	00	03
3	60	var*	00	00
4	var*	00	var*	var*
5	FF	FF	FF	FF
6	00	00	00	00

These values shall be updated by the transmitting hardware as described in ARINC Specification 818 or 818-2.

2.8 Object 0 ADVB frame

The Object 0 ADVB frame shall use the values in Table 2.8a for the container header.

Table 2.8a: Object 0 ADVB frame container header values.

Word	*Identifier*	*Byte 0*	*Byte 1*	*Byte 2*	*Byte 3*
0	Container count	var*	var*	var*	var*
1	Clip ID	00	00	00	00
2	Container time stamp	00	00	00	00
3	Container time stamp	00	00	00	00
4	Transmission type	09	01	00	00
5	Container type	00	04	00	00
6	Object 0 class	50	00	D0	00
7	Object 0 size	00	00	00	10
8	Object 0 offset	00	00	00	58
9	Object 0 object type defined	00	00	00	00
10	Object 1 class	40	00	D0	00
11	Object 1 size	00	00	00	00
12	Object 1 offset	00	00	00	68
13	Object 1 object type defined	00	00	00	00
14	Object 2 class	10	00	D0	00
15	Object 2 size	00	14	00	00
16	Object 2 offset	00	00	00	68
17	Object 2 object type defined	10	00	D0	00
18	Object 3 class	00	00	00	00
19	Object 3 size	00	00	00	00
20	Object 3 offset	00	00	00	00
21	Object 3 object type defined	00	00	00	00

These values shall be updated by the transmitting hardware as described in ARINC Specification 818 or 818-2.

The Object 0 ADVB frame shall use the values in Table 2.8b for ancillary data.

Table 2.8b. Object 0 ADVB frame ancillary data values.

Word	*Byte 0*	*Byte 1*	*Byte 2*	*Byte 3*
0	0C	00	40	00
1	10	05	77	70
2	var*	var*	var*	var*
3	00	00	00	00

These values shall be updated by the transmitting hardware as described in ARINC Specification 818 or 818-2.

Object 2 ADVB Frames

2.9 Object 2 ADVB frames

The Object 2 ADVB frame shall insert 24-bit RGB values per Table 2.9. Pixel data shall be loaded into ADVB frames in normal scanning order for a total of 1536 bytes.

Table 2.9. Object 2 ADVB frame 24-bit RGB values.

Word	*Description*	*Byte 0*	*Byte 1*	*Byte 2*	*Byte 3*
1	Frame header	44	00	00	00
2	Frame header	00	00	00	03
3	Frame header	60	var*	00	00
4	Frame header	var*	00	var*	var*
5	Frame header	FF	FF	FF	FF
6	Frame header	00	00	00	00
7	Video payload	R1	G1	B1	R2
8	Video payload	G2	B2	R3	G3
* * *					
390	Video payload	B511	G512	B512	R512
391	CRC	CRC	CRC	CRC	CRC
* * *					

These values shall be updated by the transmitting hardware as described in ARINC Specification 818 or 818-2.