DYNAMIC ENGINEERING
435 Park Dr., Ben Lomond, Calif. 95005
831-336-8891 Fax 831-336-3840
sales@dyneng.com
www.dyneng.com
Est. 1988

User Manual

IP BiSerial BA4

Bidirectional Serial Data Interface IP Module

Specification for the Display Buffer FPGA/IPACK Design
Dynamic Engineering IP-BiSerial-IO-BA4

The Display Buffer box (also referred to as Buffer box) will exist between the Radar Warning Receiver (RWR) and the LRU-9 display. Its purpose is to capture the display data that is serially transmitted from the RWR to the display (allowing it to pass transparently to the display). Every 100 ms one frame of display data will be captured and transmitted to the Avionics Simulator (AvSim) over another serial port. The AvSim will use the data to generate a simulated display.

This specification provides definition of the logic functionality that will be required of the IP-BiSerial-IO IPACK module. The IPACK module will reside inside the Display Buffer box. It carries a Xilinx XC4005 or Spartan family FPGA that can be customized to perform the tasks needed by Display Buffer box.

Interface

There are five signals in the interface; two inputs and three outputs. All five are physically implemented as RS-422 differential signals:

- **Data Request In** — Handshake signal, input to the Buffer box from the LRU-9 display. While this signal is active (high) the RWR will serially shift out data on the Display Data In port. This signal creates the window of time that display data is to be transmitted (shifted out).
- **Data Request Out** — Handshake signal, output from the Buffer box to the RWR. This is a buffered version of the Data Request In signal from the LRU-9, or a generated version of the signal when no LRU-9 is present.
- **Display Data In** — Manchester serial input to the Buffer box from the RWR. This is the display data arriving from the RWR. It is re-transmitted transparently to the LRU-9. Also, every 100 ms one frame of this data is captured, stored in a FIFO, and transmitted to the Avionics Simulator via the AvSim Display Data Out UART port.
- **Display Data Out** - Manchester serial output from the Buffer box to the LRU-9. This is the buffered version of the display data, as it is transparently re-transmitted to the LRU-9.
- **AvSim Display Data Out** — UART-style serial output from the Buffer box to the Avionics Simulator. This port carries one frame of display data every 100 ms.

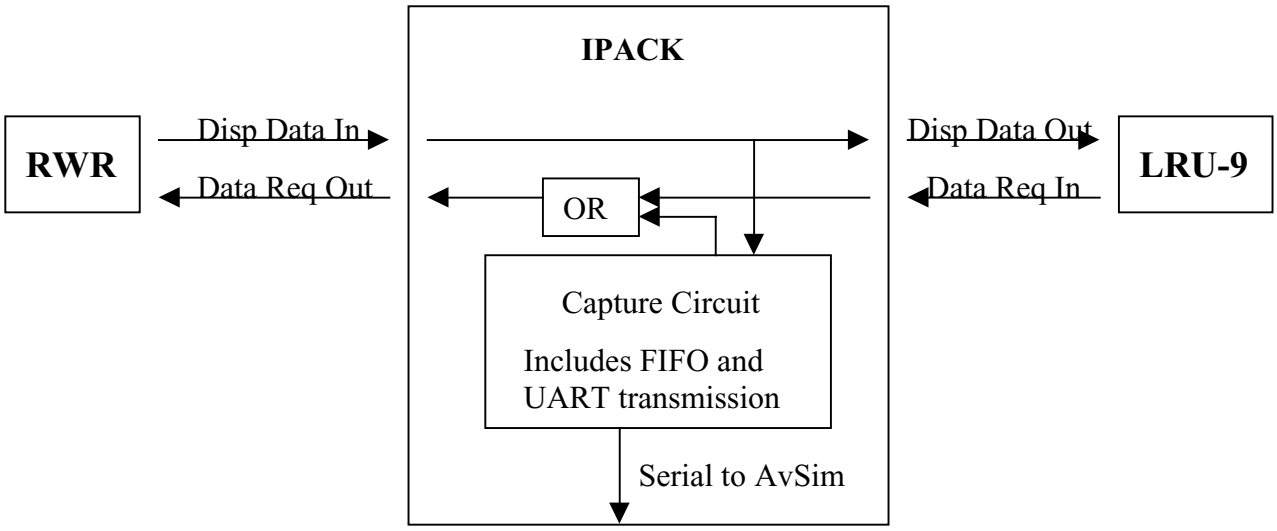
The following table summarizes the signals entering/exiting the IPACK:

Signal	From	To	Type/Protocol	Physical	Description
Data Request In	LRU-9	Buffer	Handshake	RS-422	Request for data from LRU-9
Data Request Out	Buffer	RWR	Handshake	RS-422	Buffered request, passed on the RWR
Display Data In	RWR	Buffer	Serial/Manchester	RS-422	Display data from the RWR
Display Data Out	Buffer	LRU-9	Serial/Manchester	RS-422	Buffered data, captured and passed on
AvSim Display Data Out	Buffer	AvSim	Serial/RS-232	RS-422	Captured data, 100 ms frame sent to sim

The Manchester serial protocol is implemented as a 1 MHz Manchester II (Bi-phase) serial link, with 0/1 as the idle pattern and 0/1/1 or 0/0/0 recognized as the synchronization pattern. Data is transmitted MSB first.

The RS-232 serial protocol is transmitted using a standard UART protocol. Data should be sent with RS-422 compatible line drivers. The data protocol will be 8 data bits, no parity and 1 stop bit (8N1) at a baud rate of 38400. It will be sent to a 16550-type UART device. (Data is transmitted LSB first).

Functional Diagram



Data Formatting

As mentioned, the Manchester Display Data In signal from the RWR should be buffered and sent directly out the LRU-9 Display Data Out port. Whether the display is actually connected or not, the data can always be transmitted out that port.

For the AvSim Display Data Out link, the data must be reformatted from the thirty-two 32 bit words to 128 bytes. The re-formatting that is necessary for that link is described here. Also, prior to sending the 128 bytes of each frame, a synchronization byte (0xBB) and a data byte count (0x80) should be first to the AvSim.

A frame of display data consists of 32 words of 32 bits each. Each frame begins with a unique word known as the Cross-Hair/Ring-Dot (CHRD) word. This word should be recognized by the design and noted as the start of a frame. If this is the first frame detected since startup, or 100 ms has transpired since the last transmission of a frame to the AvSim, this frame should be collected. The frame should start with the CHRD word, and continue with the next 31 data words.

The CHRD word to be detected as the beginning-of-frame breaks down as:

31-29	28	27	26	25 — 21	20 — 16	15 — 8	7 — 0
Sync	Intensify	Circ 1	Circ 2	Sym 1	Sym 2	X cord	Y cord
0 1 1 or 0 0 0	0	0	0	0x10	0x10	0x80	0x80

The circuit only needs to look at the Sym 1 field; if it is 0x10 then it is the CHRD. (Bit 31 is transmitted first.)

The data should be stored in the FIFOs as 16 bit words, with the least significant half going in first. The least significant byte of the each 16 bit word should be transmitted first, so that each 32 bit word is sent to the AvSim from bit 0 up to bit 32.

The following figures show how the data flows from the Manchester input to the UART output.

Manchester in from RWR:

Last bit in First bit in

	Dword 2		Dword 1	
	0	31	0	31

Placed in FIFO:

Last word in FIFO	Dword 2	31	16
	Dword 2	15	0
	Dword 1	31	16
First word in FIFO	Dword 1	15	0

UART Serial out to AvSim:

Last bit out First bit out

	Dword 2, byte 4	31	24
	Dword 2, byte 3	23	16
	Dword 2, byte 2	15	8
	Dword 2, byte 1	7	0
	Dword 1, byte 4	31	24
	Dword 1, byte 3	23	16
Etc	Dword 1, byte 2	15	8
3rd byte out of UART	Dword 1, byte 1	7	0
2nd byte out of UART	Byte count	0x80	
1st byte out of UART	Sync byte	0xBB	

Modes of Operation

The Display Buffer box will be used in two different modes:

1) The LRU-9 display is present
   In this configuration the Data Request from the LRU-9 will be passed transparently on to the RWR, and the data that the RWR responds with will be passed transparently to the display. On 100 ms intervals, the Display Buffer box will capture a frame of data and store it in a FIFO. As data becomes present in the FIFO it will be retrieved and transmitted to the Avionics Simulator using the UART serial port.

2) The LRU-9 display is not present
   Without the display, no Data Request will be generated. In this configuration the Buffer box shall detect that no request is being received and automatically generate the request so that at least one frame of data can be collected every 100 ms. The data that is captured will be transmitted to the AvSim.

The capture circuit should work similarly in either mode of operation. It can employ a timer that produces an edge once every 100 ms. This edge should trigger the capture circuit to begin looking for the CHRD word and subsequently collect one frame of data to be sent to the AvSim. The capture circuit would then rest until the next 100 ms trigger.

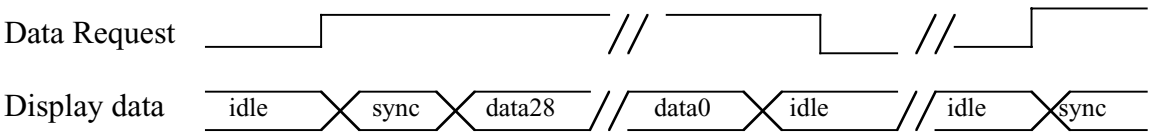
Handshake Protocol

As long as the Data Request signal from the LRU-9 to the RWR is active, the RWR will shift display data out. The request provides the window of time that the RWR will transmit the data. This implies that when one 32-bit word is desired the request must window those 32 bits in accordance with specific timing requirements. This is how the LRU-9 display requests display data. (The specific requirements are included in the attached diagram.)

However, it is conceivable that the data could be requested in other packages. A frame of display data (all 1024 bits, or 32x32 words of it) is stored in a 1Kbit shift register in the RWR. So, for example, another version of the display (other than the LRU-9) requests the whole frame with one request window (as opposed to 32 request windows). This is the preferred way to request data. (The capture circuit must be aware that the request can be issued either way. This will be discussed further in the **Frame Capture Circuit** section below.)

When the LRU-9 display is **not** present and the Buffer box determines that it must generate the request itself, the design should request the data using the whole-frame-at-once scheme, i.e. it should request all 1024 bits with one request window. The request must be removed at the correct time, or one too many or too few bits may be sent from

the RWR. Close attention must be paid to the timing restrictions put on the request line. The following diagram shows the general requirements of the link (for one word), but a more detailed attachment includes the specific timing.



Notes:
The elapsed time from beginning-of-word to beginning-of-word (within a frame) is approximately 300 us on average (assuming word-at-a-time)
The elapsed time from beginning-of-frame to beginning-of-frame is approximately 20 ms.

As mentioned earlier, when the LRU-9 is not present the design should recognize that no request is being generated and start generating the request itself. It can request data at whatever rate makes the design more manageable, as long as a frame of data can be collected and transmitted to the AvSim every 100 ms. If it makes sense to request a frame every 20 ms and to let the collection circuit pick which frame to collect, that is acceptable. (Data should not be requested more often than 20 ms to avoid over-tasking the RWR.)

NOTE: The request generation circuit should continue to monitor the Data Request In signal. If a request signal begins to appear on that port, then the generating circuit should stop producing the request.

Frame Capture Circuit

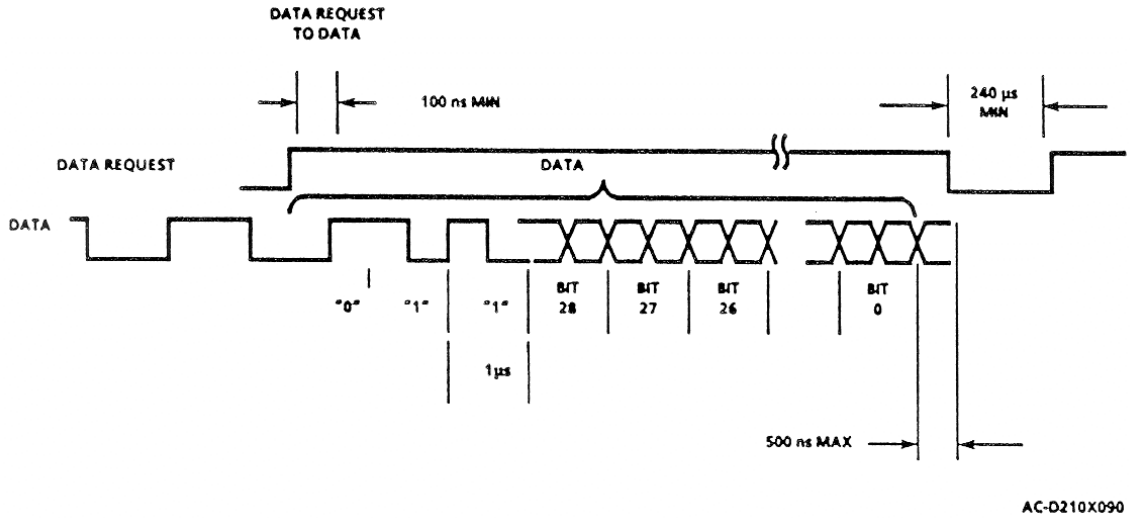
The frame capture circuit can operate independent of the knowledge of where the request is generated. It should have the simple requirement of collecting a frame of data and transmitting it to the AvSim once every 100ms.

As stated earlier, the capture circuit can employ a timer that produces an edge once every 100 ms. The edge should trigger the capture circuit to begin looking for the next frame of data to collect and send to the AvSim. Once sent, the capture circuit would then rest until the next 100 ms trigger.

An important issue to discuss is the fact that different displays may request the display data in slightly different manners. The two current ways of requesting the display data are as thirty-two separate 32-bit words, or as one long 1024 bit stream. Either way provides the exact same data.

The capture circuit should be able to collect the data without concern of how the display is requesting the data. This can be done if the request signal is simply used as an enable to indicate when non-idle data is present on the Display Data port. Once 32 bits of data are captured is should be shipped out to the FIFO for transmission to the AvSim. If the request signal is still active, then collection of the next 32 bits should proceed. This should continue until thirty-two words of data have been captured and transmitted to the AvSim. The same rule of the CHRD word still applies. The frame collection should always begin with the CHRD word and continue for the next 31 words.

Attachment A
Manchester Timing (One 32-bit Word)



Although this diagram shows the timing for one word, it does extend for as many bits as needed (up to 1024). The Data Request active/inactive timing is the same relative to the first bit collected and last bit collected. In other words, ensure that Data Request becomes active more than 100 ns prior to leading edge of the first bit (but less than 500 ns prior to the leading edge, not shown.) And ensure that the Data Request goes inactive 500 ns (or less) after the trailing edge of the last data bit.

Please note the Manchester signaling in the diagram. A binary 1 is indicated by a transition from high-to-low; a binary 0 is indicated by a transition from low-to-high. In the idle periods (when the request is inactive) the Manchester link maintains an alternating 0/1 pattern. Since this is a 1 MHz link, when the link is in the idle state the signal appears as a 500 KHz clock (high for 1 us, low for 1 us).

The sync field in each Manchester data word can either be 0/1/1 or 0/0/0. If it is 0/1/1, the word contains new display data; if it is 0/0/0, the word does not contain new valid display data (null word). Words beginning with either form of sync. should be collected and transmitted to the AvSim as part of each frame of data. The collection circuit does not need to distinguish between the two types of words. (There will usually be some null words in each frame.)

Pinouts

DATA IN +[RX_422D]	20
DATA IN -	21
DATA OUT + [TX_DATAP]	8
DATA OUT -	9
RQST IN + [RX_422C]	23
RQST IN -	24
RQST OUT + [TX_CLK]	11
RQST OUT -	12
AVSIM + [TX_STB]	14
AVSIM -	15

GROUNDS
1,4,7,10,13,16,19,22,25,28,30,32,34,36,38,39,41,42,44,45,47, 48,50

ID PROM
The BiSerial-BA4 has an IDPROM built into the Xilinx. I realize that you are not using the bus interface other than to supply a clock on this project. You can use the data for configuration control.

Current data 9/9/99
0x49
0x50
0x41
0x48
0x1E manufacturer [Dynamic Engineering]
0x01
0xA0 revision code
0x00
0x04 BA4 designator
0x00
0x0C byte count
0x7A crc

Download header pinout

1	vcc
2	gnd
3	m0-2 unused with download cable
4	cclk
5	done
6	din

Please note that pins 2-3 need to be tied together when the PROM is installed.

Order Information

The IP-BiSerial BA4 board standard configurations.
http://www.dyneng.com/ip_bis_ba4.html

IP_Bis_BA4

Tools for IP-BiSerial-BA4

IP-Debug-Bus - IP Bus interface extender with testpoints, isolated power and quickswitch technology to allow hot swapping of IPs or power cycling without powering down the host.
http://www.dyneng.com/ipdbgbus.html

IP-Debug-IO II - IndustryPack IO connector breakout with testpoints, ribbon cable headers, and locations for user circuits.
http://www.dyneng.com/ipdbgio.html

HDRterm50 - Ribbon cable compatible 50 pin header to 50 screw terminal header. Comes with DIN rail mounting capability.
http://www.dyneng.com/HDRterm50.html

PCI3IP - 1/2 length PCI card with 3 IP slots.
http://www.dyneng.com/pci_3_ip.html

IP-MTG-KIT – 4 metric stainless screw and stand-off pairs to retain IP-Parallel-IO against the carrier board. Flat head screws match IP Specification mounting requirements.

All information provided is Copyright Dynamic Engineering