APPLICATION NOTE

ABSTRACT
The scope of this application note is to make the developer aware of the difference between the L21 and the L11 and to provide the developer with the basic information needed in order to upgrade an application from the L11 to the L21. This application note concentrates mainly on the modifications that pertain to the AV Link registers and the related parts of the transaction layer.

AN2453

Half duplex AV link (PDI1394L11) vs. Full duplex AV link (PDI1394L21)

Author: Elie Boujaoude

1999 Aug 13

Philips Semiconductors

PHILIPS

Half duplex AV link (PDI1394L11) vs. Full duplex AV link (PDI1394L21)

AN2453

Author: Elie Boujaoude

TABLE OF CONTENTS

1.0 INTRODUCTION

The PDI1394L21 is an AV Link Layer Controller chip that complies with IEEE 1394-1995. It is a modification of the PDI1394L11. One of the main characteristics of the PDI1394L21 is its full duplex functionality. While the PDI1394L11 is able to transmit and receive, it can only do either one or the other at a time, thus the PDI1394L11 is characterized as half duplex. On the other hand, the PDI1394L21 is capable of transmitting and receiving at the same time on two different AV ports. Although most applications need a half duplex AV Link chip, there are several major applications that can benefit greatly from a full duplex AV Link chip; such an application is the Set Top Box.

For simplicity, in the remainder of this document, we will refer to the PDI1394L11 AV Link chip as L11 and the PDI1394L21 AV Link chip as the L21.

Code samples, pseudo code and coding suggestions shown in this application note are meant to be used as guidelines for developers. From this application note and the datasheets for the L11 and L21, developers can expect to learn the difference between these two devices. It is left to the developers to determine what modification(s), and corresponding implementation(s), is(are) needed in their end application.

In case of any questions or suggestions regarding this application note or any other related 1394 issues, please feel free to contact the Philips 1394 Applications and Marketing group at **1394@philips.com**. For information on Philips Semiconductors 1394 chipsets and tools please visit our web site at **www.semiconductors.philips.com/1394**.

2.0 APPLICATION NOTE OBJECTIVE

The scope of this application note is to make the developer aware of the difference between the L21 and the L11 and to provide the developer with the basic information needed in order to upgrade an application from the L11 to the L21. This application note concentrates mainly on the modifications that pertain to the AV Link registers and the related parts of the transaction layer.

3.0 DIFFERENCE BETWEEN THE PDI1394L21 AND THE PDI1394L11

3.1 Main Functionality

The L21 is “full duplex”, while the L11 is “half duplex”. That is, while the L11 can either transmit or receive at a time, the L21 can transmit and receive at the same time. Thus, the L21 RDK has 2 AV ports that are capable of simultaneous transmission and reception of isochronous data over the 1394 bus. The L11 either transmits or receives on one AV port.

3.2 Modified AV Link Registers

All the modifications to the AV Link registers were performed with backward compatibility with the L11 registers in mind. That is:

- When a new field is added to a register where no other field existed before (reserved), the default value of that field was set to 0.
- When the functionality/usability of a field has been modified, the previous functionality of the field was preserved.
- Fields are moved from one register to another only when they can be more useful and easier to access.
- Whenever the name of a field was modified, the name was modified for the sole purpose of making it more intuitive.

The modified Link registers and the bits are as follows:

Registers	Modified/Removed Fields	New Fields
0x004 LNKCTL		TxRDY
0x008 LNKPHYINTACK	TxRDY	
0x00C LNKPHYINTE	ETxRDY	
0x018 GLOBCSR	TXMODE	ENOUTAV1, ENOUTAV2, DIRAV1
0x020 ITXPKCTL		SYT_DELAY, ENXTMSTP
0x02C ITXINTACK		ITX512LFT, ITX256LFT, ITX100LFT
0x030 ITXINTE		EITX512LFT, EITX256LFT, EITX100LFT
0x034 ITXCTL	SYNC	SYNC: ODD/EVEN
0x038 ITXMEM		ITM512LFT, ITM256LFT, ITM100LFT
0x040 IRXPKCTL		RXAP_CLK, SNDIMM, DIS_TSC
0x04C IRXINTACK		IRX512LFT, IRX256LFT, IRX100LFT
0x050 IRXINTE		EIRX512LFT, EIRX256LFT, EIRX100LFT
0x054 IRXCTL	SYNC	SYNC: ODD/EVEN
0x058 IRXMEM		IRM512LFT, IRM256LFT, IRM100LFT
0x080 ASYCTL		DIS_BCAST

The following sections describe the specific modifications to these registers.

4.0 NEW AND MODIFIED REGISTERS

The modifications to the transaction layer are limited to those of the AV Link registers. The following subsections show the AV Link registers that have been modified and show analysis to be used when modifying the application. The next section gives coding suggestions to insure a backward compatibility and forward preparedness with other link revision.

4.1 Registers 0x004 LNKCTL, 0x008 LNKPHYINTACK, and 0x00C LNKPHYINTE

L11:
Interrupt TxRDY, bit 12 of register 0x008 LNKPHYINTACK, is used in the L11 to indicate that the transmitter is idle and ready. For each interrupt bit in the LNKPHYINTACK register, there exists a corresponding enable interrupt bit in the LNKPHYINTE. Thus the enable interrupt bit for TxRDY is bit ETxRDY, bit 12, of register 0x00C LINKPHYINTE.

L21:
In the design of the L21, it was decided that the transmitter idle and ready status is better indicated by a status bit instead of an interrupt. Thus, TxRDY was moved to bit 6 of the General Link control register, 0x004 LNKCTL.

Analysis:
The vacated locations of bits TxRDY and ETxRDY in LNKPHYINTACK and LNKPHYINTE have been replaced with reserved fields, thus writing to these bits has no effect. Also, since the new location of TxRDY was a reserved bit and since TxRDY is a read only bit, then no modification is necessary to the existing code unless the application desires to keep track of the transmitter's status.

4.2 Register 0x018 GLOBCSR

L11:
Since the L11 is "half-duplex", that is, it can either transmit or receive at a time, a control bit is used to control the mode of the chip by enabling either the AV transmitter or the AV receiver. This is bit TXMODE, bit 16 of the GLOBCSR.

L21:
The L21 is a full duplex part that provides the capability of transmitting and receiving on different AV ports at the same time. Thus the need to change the above TXMODE and replacing it with three bits: DIRAV1 (bit 16), ENOUTAV1 (bit 17), and ENOUTAV2 (bit 18).

DIRAV1 is used to specify the direction of the AV port 1 (AVPORT1), transmit or receive. When DIRAV1 is 1 then AVPORT1 is enabled as a transmitter, and thus AVPORT2 is enabled as a receiver; and vice versa when DIRAV1 is set to 0.

ENOUTAV1 and ENOUTAV2 are used to enable the AV ports 1 and 2 as outputs. When set to 1, the corresponding port is enabled as an output, and when set to 0, the corresponding port is in 3-state condition and may be used for either input or unused output based on the state of DIRAV1.

Analysis:

- Anywhere the application sets the chip to transmit or receive, the code needs to be modified to access the above new control bits.
- Since a port is enabled as either receiver or transmitter, the two ports should not be outputs at the same time. Thus ENOUTAV1 and ENOUTAV2 should not be both set to 1 at the same time. However, the link chip will cause only the port set to receive to be an output in this case of program error.

4.3 Register 0x020 ITXPKCTL

L11:
SYT Delay: If a device transmits real time data (identified by FMT) and requires time stamp in the CIP header, it shall use the SYT format. The SYT stamp shall point to a time in the future that is greater than 1 cycle in order for the packet to be transmitted. When packets that use the SYT field are transmitted using the L11, a delay of 3 cycles is added to the SYT stamp; this is called SYT delay. For more information on the CIP, FMT, and SYT please see the IEC-61883 specification.

Time stamps: On transmission of isochronous packets that required time stamping, the stamps are generated automatically in the L11 link chip.

L21:
Two new fields have been added to the ITXPKCTL register: SYT_DELAY (bits 5 and 6) and ENXTMSTP (bit 7).

SYT_DELAY: This field is added to make the SYT delay programmable by the application. This field can be set to provide a delay of 2, 3, or 4 cycles. For more information, please refer to the L21 datasheet and the above brief description under the L11 section.

ENXTMSTP: When this field is set, the link chip uses externally generated time stamps instead of generating them internally.

Analysis:
SYT_DELAY: The default value for this field is 0, which indicates a delay of 3 cycles. Thus, implementation of this field is needed by applications that may require a different delay.

ENXTMSP: When this field is set, an extra quadlet (4 bytes) is added to the packets. Thus, the application should be modified to handle the extra quadlet. For example an MPEG-2 packet using internal time stamps has a size of 188 bytes, however a packet that uses external time stamps has a size of 192 bytes.

For information on how to set control bits, please refer to section 5.2.2.

4.4 Registers 0x02C ITXINTACK and 0x030 ITXINTE

L21:
The following three interrupts and their interrupt enables have been added: IT512LFT, IT256LFT, IT100LFT, EIT512LFT, EIT256LFT, and EIT100LFT. These interrupts get asserted when the transmitter queue reaches 512, 256, and 100 quadlets from full.

Analysis:
In the L11, time stamps were generated internally. In the L21, the application can use either internally or externally generated time stamps as mentioned in the section 4.3. Some applications that use the external time stamps send data in bursts; that is, data is loaded into the FIFO and are transmitted at some time in the future. The interrupts above alert the application of the status of the FIFO so that the application will not overrun the FIFO.

For information on how to acknowledge interrupt bits and how to set interrupt enable bits, please refer to sections 5.2.1 and 5.2.2.

4.5 Register 0x034 ITXCTL

L11:
The SYNC field (bits 0 through 4) of the ITXCTL register contains the code to insert in the SY field of the isochronous bus packet header.

L21:
The ODD/EVEN bit (bit 1 of the SYNC field) is used for the encryption key (0=even, 1=odd). This is a read only bit that reflects the encryption key used for the transmitted packet. The newly added AVxENKEY pin sets the encryption key for the transmitted packets.

Analysis:
The SYNC field is formed of 4 bits as follows: bit 0 is the SY field, bit 1 is the ODD/EVEN key, bits 2 and 3 are not used in the L11 and the L21. In the L11, only bit 0 (the SY field) was used. Thus, if the existing code uses the SYNC as one field, then the developer may want to break it into 4 fields to handle the SY, the ODD/EVEN, and the two unused bits separately.

4.6 Register 0x038 ITXMEM

L21:
The following three memory status bits were added: ITM512LFT, ITM256LFT, and ITM100LFT. These memory statuses are similar to the interrupts in section 4.4, they indicate when the transmitter queue reaches 512, 256, and 100 quadlets from full. The difference is that these memory statuses are continuously modified by the link to reflect the current status of the isochronous transmitter queue.

Analysis:
Memory status bits are read only and are available for diagnostic purposes. Thus, modification to the existing code may not be necessary.

4.7 Register 0x040 IRXPKCTL

L21:
Three new fields were added; these are RXAP_CLK, SNDIMM, and DIS_TSC.

RXAP_CLK is Receiver Application Clock that allows the user to use an external input clock on the AVxCLK pin or use an internal clock (24.576, 12.288, and 6.144 MHz) and output it on the AVxCLK pin.

SNDIMM: when set to 1, a received isochronous packet containing a CRC error is sent to the output immediately (without regard to the time stamp value).

DIS_TSC is used to disable Time Stamp Checking. When time stamp checking is disabled, the time stamp accompanying a packet is output before the packet to the application for the use by the application. This adds an extra quadlet of data to the received data stream.

Analysis:
Applications that intend to disable time stamp checking must be modified to handle the extra quadlet received with the data stream.

RXAP_CLK, SNDIMM, and DIS_TSC are control bits. For information on how to set control bits, please refer to section 5.2.2.

4.8 Registers 0x04C IRXINTACK and 0x050 IRXINTE

L21:
The following three interrupts and their interrupt enables have been added: IR512LFT, IR256LFT, IR100LFT, EIR512LFT, EIR256LFT, and EIR100LFT. These interrupts get asserted when the receiver queue reaches 512, 256, and 100 quadlets from full.

Analysis:
In the L11, time stamps were generated internally. In the L21, the application can use either internally or externally generated time stamps as mentioned in the section 4.3. Some applications that use the external time stamps are designed to receive the data, store it in the FIFO, and send it in bursts out on the AV ports at a time in the future. The interrupts above alert the application of the status of the FIFO so that the application will not overrun the FIFO by waiting too long to release the data.

For information on how to acknowledge interrupt bits and how to set interrupt enable bits, please refer to sections 5.2.1 and 5.2.2.

4.9 Register 0x054 IRXCTL

L11:
The SYNC field (bits 0 through 4) of the IRXCTL register contains the last received SY code in the isochronous bus packet header.

L21:
The ODD/EVEN bit (bit 1 of the SYNC field) is used for the encryption key (0=even, 1=odd). This is a read only bit that reflects the encryption key used for the received packet. The newly added AVxENKEY pin reflects the encryption key for the received packets.

Analysis:
The SYNC field is formed of 4 bits as follows: bit 0 is the SY field, bit 1 is the ODD/EVEN key, bits 2 and 3 are not used in the L11 and the L21. In the L11, only bit 0 (the SY field) was used. Thus, if the existing code uses the SYNC as one field, then the developer may want to break it into 4 fields to handle the SY, the ODD/EVEN, and the two unused bits separately.

4.10 Register 0x058 IRXMEM

L21:
The following three memory status bits were added: IRM512LFT, IRM256LFT, and IRM100LFT. These memory statuses are similar to the interrupts in section 4.8, they indicate when the receiver queue reaches 512, 256, and 100 quadlets from full. The difference is that these memory statuses are continuously modified by the link to reflect the current status of the isochronous receiver queue.

Analysis:
Memory status bits are read only and are available for diagnostic purposes. Thus, modification to the existing code may not be necessary.

4.11 Register 0x080 ASYCTL

L11:
When the destination ID of an asynchronous packet is set to 63 (=0x3F), the packet is considered a broadcast packet and thus all the nodes on the bus report it to their application layer.

L21:
Depending on the types of applications/nodes on the network, there might exist lots of broadcast traffic in which the application may not be interested. DIS_BCAST has been introduced to allow the developer to disable the reporting of broadcast packets (1=Disable reception of broadcast packets, 0=enable reception of broadcast packets).

Analysis:
DIS_BCAST is a control bit. For information on how to set control bits, please refer to section 5.2.2.

5.0 SUGGESTED COMMON PRACTICE CODING

When developing the transaction layer for a Link chip it is necessary to insure that the code is backward compatible with existing link chips, and as ready as possible to work with future chips. Some of the most common problems that developers run into are due to the methods of accessing the shadow and the link registers. Below are common coding suggestions on how to best access (and how to not access) these registers to insure backward compatibility.

5.1 The Shadow Registers

The host interfaces of both the L21 and the L11 provide access to 64 link registers and to the asynchronous packet queues through an 8-bit interface. Since all the registers and the queues are made of quadlets (4-bytes), accessing these registers might require more than one read or write cycle. In order to insure that the value does not change during the read/write of a quadlet, two shadow registers were introduced, these are Read Shadow register and Write Shadow register.

The Read Shadow register holds a snapshot of the read link register (4 bytes), while the Write Shadow register holds the 4 bytes value until the application is ready to write to the actual link register. In order to obtain the snapshot (Read) and to cause the writing of the link register (Write) the Host Interface Address line 8 (HIF A8) is used as the update control line. That is, when the application wants to read/write a value to/from a link register, it causes the access by asserting the update control line. For more information on accessing these registers please see the datasheets.

These shadow registers should be accessed cautiously; some of the cautions are listed in the sections below.

5.1.1 Do not access the Read Shadow Register from more than one location at a time:

As described in the data sheets, this register is used to get a snapshot of the value of a link register (quadlet) and make it available to the CPU one byte at a time. That is, when the transaction layer wants to read the value of a link register, all 4 bytes of the register are placed in the shadow register and thus the transaction layer can access them one byte at time. Since this shadow register could be used by other parts of the transaction layer or the application layer, if not guarded well, the snapshot (value) inside the register can be overwritten while the 4 bytes are read one at a time.

Such a problem could occur in case of interrupts: If an interrupt occurs while a register is being read, the application is interrupted and the interrupt handling routine is called. In most cases, the interrupt routine needs to access the shadow register to check which interrupt occurred. Thus, the value that was being read by the application prior to the interrupt gets overwritten. One solution to such a problem is to disable the external interrupts while reading from the shadow register.

In the case of an 8051, the external interrupts are controlled via EX1, and thus to read a quadlet, the code might look as follows:

```
QUADLET AVLinkRead (BYTE offset)
{
    QDATA value;

    /* Disable external interrupt 1 */
    EX1 = 0;

    /* First, transfer the register value to the shadow register */
    value.byte[0] = XBYTE[AVBASE + offset + LINK_TRANSFER];

    /* Then get the remaining bytes */
    value.byte[1] = XBYTE[AVBASE + offset + 1];
    value.byte[2] = XBYTE[AVBASE + offset + 2];
    value.byte[3] = XBYTE[AVBASE + offset + 3];

    /* Enable the external interrupt */
    EX1 = 1;

return value.quadlet;
}
```

Where AVBase is the address of the shadow register in RAM, and LINK_TRANSFER is the offset for the update control line.

In the case where only one byte needs to be read, it is not necessary to disable the external interrupts since the value can be automatically transferred. For such a case, the code might look as follows:

```
BYTE AVLinkByteRead(BYTE offset)
{
    return XBYTE[AVBASE + offset + LINK_TRANSFER];
}
```

Disabling the external interrupt might not be the only solution depending on the application. Other solutions might pertain to your application as long as they insure that the shadow register is not accessed by more than one location at a time.

5.1.2 Do not access the Write Shadow Register from more than one location at a time:

This is the same case as for the Read shadow register above. In the case of an 8051, the external interrupts are controlled via EX1, and thus to write a quadlet the code might look as follows:

```
QUADLET AVLinkWrite (QUADLET quadlet, BYTE offset)
{
    QDATA value;

    /* Disable external interrupt 1 */
    EX1 = 0;

    value.quadlet = quadlet;

    /* First, write to the shadow register */
    XBYTE[AVBASE + offset] = value.byte[0];
    XBYTE[AVBASE + offset + 1] = value.byte[1];
    XBYTE[AVBASE + offset + 2] = value.byte[2];

    /* Then perform the actual write by writing to the link register */
    XBYTE[AVBASE + offset + 3 + LINK_TRANSFER] = value.byte[3];

    /* Enable the external interrupt */
    EX1 = 1;

    return (quadlet);
}
```

Where AVBase is the address of the shadow register in RAM, and LINK_TRANSFER is the offset for the update control line.

In the case where only one byte needs to be written, it is not necessary to disable the external interrupts since the value can be automatically transferred. For such a case, the code might look as follows:

```
void AVLinkByteWrite (BYTE value, BYTE offset)
{
    XBYTE[AVBASE + offset + LINK_TRANSFER] = value;
}
```

CAUTION: It is not suggested to write only one byte. Please see the next section on this matter.

5.1.3 Do not write only one byte using the Write Shadow Register:

Although it is possible to write only one byte using the Write shadow register as shown above, it is not suggested to be used since the other 3 bytes in the shadow register which might be of unknown values will be transferred as well to the link register. In the case where the other bytes of the link register are reserved or read only, it is still not suggested for forward compatibility since the reserved bits could be used in future link chips.

5.1.4 Do not assume the value in a shadow register is still unchanged from a previous access:

Since more than one location in the application can access the shadow register, the value in the register could have changed between different reads or writes.

An example of such a misconception occurs when the application is writing a series of quadlets of similar values as part of a packet to the write shadow register. If the quadlets are as follows:

```
0x00000001
0x00000002
0x00000003
0x00000004
…
0x000000FF
```

A quick look at the data could erroneously suggest that once the first quadlet is written to the shadow register and is transferred to the queue, the next quadlets can be written by only modifying byte 3 in the shadow register and transferring it to the queue. Although this might sound like a good idea, it is important to remember that the write shadow register could have been used in between the quadlet writes by another part of the application. And, as mentioned in a previous suggestion, it is always more cautious to write all 4 bytes of a value to a shadow register.

5.2 The AV Link Registers

Accessing the AV link registers is done via the shadow registers as described in the previous section and in the datasheet. Some of the most common problems encountered in porting from one link chip to another are related to setting and acknowledging interrupt registers. Below are suggestions on how to access these types of registers to insure backward compatibility.

5.2.1 How to acknowledge interrupts:

To acknowledge an interrupt bit, a ‘1’ must be written to it. A value of ‘0’ does not modify or acknowledge an interrupt. Thus, the easiest and fastest way to acknowledge an interrupt is to write a ‘1’ to that bit and ‘0’s to all the other bits in the register. For example, to acknowledge interrupt bit 9 of an interrupt acknowledge register a write of value 0x00000200 should be written to that register.

It is important to remember that a value of ‘0’ should be written to the interrupt bits that do not need to be acknowledged as well as the reserved bits. The latter is due to the fact that reserved bits might be used in future links.

5.2.2 How to set interrupt enables and general control bits

A common mistake in accessing interrupt enable bits and general control bits is that the other bits in the same registers get unintentionally overwritten. A solution to such a problem is to first read the value of the register, modify the bit in the read value and then write the whole register back.

For example, to set bit 9 in interrupt enable register 0xA4:

```
ValueASYINTE = AVLinkRead(0xA4);
AVLinkWrite (0xA4, ValueASYINTE | 0x00000200);
```

To clear bit 9 in interrupt enable register 0xA4:

```
ValueASYINTE = AVLinkRead(0xA4);
AVLinkWrite (0xA4, ValueASYINTE & 0xFFFFFDF);
```

Half duplex AV link (PDI1394L11) vs. Full duplex AV link (PDI1394L21)

AN2453

Definitions

Short-form specification — The data in a short-form specification is extracted from a full data sheet with the same type number and title. For detailed information see the relevant data sheet or data handbook.

Limiting values definition — Limiting values given are in accordance with the Absolute Maximum Rating System (IEC 134). Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Characteristics sections of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability.

Application information — Applications that are described herein for any of these products are for illustrative purposes only. Philips Semiconductors make no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Disclaimers

Life support — These products are not designed for use in life support appliances, devices or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips Semiconductors customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Philips Semiconductors for any damages resulting from such application.

Right to make changes — Philips Semiconductors reserves the right to make changes, without notice, in the products, including circuits, standard cells, and/or software, described or contained herein in order to improve design and/or performance. Philips Semiconductors assumes no responsibility or liability for the use of any of these products, conveys no license or title under any patent, copyright, or mask work right to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified.

Philips Semiconductors
811 East Arques Avenue
P.O. Box 3409
Sunnyvale, California 94088–3409
Telephone 800-234-7381

Date of release: 08-99

Document order number: 9397 750 06386

Let's make things better.

Philips Semiconductors

PHILIPS