

Embedded Systems Group LLRF Documentation

Linkport.doc

Last Saved 10/30/00

Subject: Digital Signal Processor (Analog Devices 21062) link port description and implementation in a LLRF system using a VXI bus

General: provides six link ports

data transfer: 1 nibble/DSP clock (x1 clock), 1 byte/DSP clock (x2 clock)

32 or 48 bit transfers

DMA and chaining DMA on each channel

DMA IRQ generated on block transfer complete or receive/transmit buffers

Link Service IRQ generated when accessing a disabled port

logical mapping from DSP pins to link buffers

status "fullness" for each buffer

packing error status for each buffer

Specific: MI/TV link is circular (any data may be read or added by any card)

RR link is unidirectional

(only upstream data may be read by downstream cards)

VXI local bus is used for transmission/reception path

DMA transfer complete generates IRQ which runs 100 KHz. process

100 KHz. process determines if 16.6 KHz. or 720 Hz. process are also executed

100 KHz. process is PLL to TCLK E\$07 by master module (dds)

The Embedded Systems Group front ends, for the MI,TV, and RR LLRF, contain cards which support communication among themselves. This is accomplished by using the VXI local bus and the DSPs built in link port hardware. By transferring data from card to card, data is shared. This shared data can come from internal sources (ie. calculations, summed values, feed forward curves, etc.) and external (ie. beam feedback, MDAT broadcasts etc.)

Link port activity is initiated by a master card, which is a DDS type, and repeats at a 100 KHz. rate. All other cards are slaves. Each 100 KHz. "tick" allows either 250 or 333 DSP instructions before the next link port transfer starts. There are 138 ticks/720 Hz. and each one is numbered and sent over the link port to coordinate slave activities. The master 100 KHz. is phase locked to TCLK E\$07.

The MI operates with five DSP cards linked together. The TV has three and the RR has four. The RR link port is somewhat different since it transmits in only one direction with data sharing only possible with downstream cards.

Below is a code segment that shows the variables which are sent/received on the MI DDS link port.

LinkBuf1Begin:

```
(1) .VAR ddsTickCounter;      100 KHz. tick count (0 -> 137, 138 ticks/ 720 Hz.)
(2) .VAR _ddsTimeFromReset;  seconds from cycle reset
(3) .VAR _ddsFout;          output frequency
```

LinkBuf1End:

LinkBuf0Begin:

```
(4) .VAR _xferMDat;          MDAT frame 54 & 55
(5) .VAR adcJFpll;          phase feedback for phase lock loop
(6) .VAR _adcJVgap;         HLRF cavity sum voltage
(7) .VAR adcQSum1;          phase contributions from ADCI & ADCJ to NCO1
```

Embedded Systems Group LLRF Documentation

Linkport.doc

Last Saved 10/30/00

```
(8) .VAR adcQSum2;           phase contributions from ADCI & ADCJ to NCO2
(9) .VAR adcQSum3;           phase contributions from ADCI & ADCJ to NCO3
(10) .VAR adcIFrpfb;         future feedback signal
```

LinkBuf0End:

```
(11) .VAR xFerTickCounterReturned; tick count which has traversed the link path
```

Variables #1 through #3 originate at the DDS. Data value #4 is received (and used) and retransmitted. #5 through #10 originate at other locations and are used by the DDS. Value #11 is repeated around the link loop and checked to determine if there were round trip errors.

All modules use two link ports: 1 for transmit, 1 for receive. This section describes the initialization of link ports during the "_init_dds" routine:

Map link port 0 (pins) to link buffer 0 (I/O processor space) and link port 1 to buffer 1 (make all others inactive):

```
r9 = LP0LB0 | LP1LB1 | LBUF2I | LBUF3I | LBUF4I | LBUF5I;
(defllrf.h contains LBUFxx definitions)
dm(LAR) = r9;           (LAR = Link Assignment Register)
```

Set the clock rate for buffers 0 and 1 (x2 the DSP clock):

```
r9 = LCLKX20 | LCLKX21;
(defllrf.h contains LCLKXxx definitions)
dm(LCOM) = r9;         (LCOM ≡ Link Common control register)
```

Setup DMA channel #1 for buffer #0 (receive) and DMA channel #3 for buffer #1 (transmit):

```
r9 = LinkBuf0Begin;    (start address of data block)
dm(II1) = r9;
dm(IM1) = m6;          (amount to auto increment address: m6=1)
r9 = LinkBuf0End - LinkBuf0Begin + 1;
dm(C1) = r9;           (number of variables in block)

r9 = LinkBuf1Begin;    (start address of data block)
dm(II3) = r9;
dm(IM3) = m6;          (amount to auto increment address: m6=1)
r9 = LinkBuf1End - LinkBuf1Begin + 1;
dm(C3) = r9;           (number of variables in block)
```

Enable buffers:

```
r9 = LOEN | LODEN;    (enable buffer #0 for DMA reception)
(defllrf.h contains LOENxx definitions)
dm(LCTL) = r9;        (LCTL ≡ Link buffer Control register)
```

This section describes what code is executed during the "Scheduler" routine to service a link port transmission/reception (100 KHz. internal timer interrupt driven).

Initiate link buffer #1 transfers:

```
r9 = L1EN | L1DEN | L1TRAN; (enable buffer for DMA transmission)
(defllrf.h contains LxEN and LxTRAN definitions)
dm(LCTL) = r9;
```

Embedded Systems Group LLRF Documentation

Linkport.doc

Last Saved 10/30/00

Call "LinkPortControl"

This part updates the link port tick count and stores the returned tick count for later analysis of transmission errors in "soft0isr.c".

```
r2 = b3;                (get start of scheduler table)
r1 = i3;                (get scheduler pointer)
r1 = r1 - r2, r3 = dm(xFerTickCounterReturned - varbase, i4); (r1 is the index)
dm(ddsTickCounter-varbase, i4) = r1; (write ddsTickCounter to link port data)
dm(@schedulerJumpTab, i3) = r3;    (store returned tick counter)
dm(LCTL) = m5;            (clear link control)
```

This part reinitializes the DMA channels.

```
r9 = LinkBuf0Begin;    (start address of data block)
dm(II1) = r9;
r9 = LinkBuf0End - LinkBuf0Begin + 1;
dm(C1) = r9;          (number of variables in block)

r9 = LinkBuf1Begin;    (start address of data block)
dm(II3) = r9;
r9 = LinkBuf1End - LinkBuf1Begin + 1;
dm(C3) = r9;          (number of variables in block)
RTS (db);
r9 = LOEN | LODEN;    (enable buffer #0 for DMA reception)
dm(LCTL) = r9;
```

There are timing restrictions due to the real time processing of data at the 100 KHz. rate. The time limit is a combination of the number of words (32 bit) transmitted, the number of cards involved, the software overhead, and latency effects. (At the present, the Tev link port operates at x1 because, in the past, the dsp silicon version did not support x2 operation. That silicon has been replaced with a version that does support x2 but the code has not been changed since the link has always worked at x1.)

The total time required for setup and data transmission is composed of five items:

1. start the transmission:

```
r9 = L1EN | L1DEN | L1TRAN;
dm(LCTL) = r9;                (enable buffer for DMA transmission)
```

2. I/O Processor starts filling 2 deep link FIFO with data

3. time (in dsp clock ticks) for link port transmission:

```
(assume no higher DMA or intervening requests)
(1 nibble/dsp clock)*(LCLKX21) = 1 byte/dsp clock
(dsp clock/1 byte)*(4 bytes/32 bit word) = 4 dsp clocks/word
(4 dsp clocks/word)*(4 words/MI dds block transmit) =
16 dsp clks/MI dds block transmit
```

4. DMA channel #0 transfer complete interrupt "SPR1I" on downstream card (ADCI) is latched and (assume no higher interrupt request) its' 100 KHz. routine is called.

5. reinitialize DMA beginning address and count registers for next transmission

Embedded Systems Group LLRF Documentation
Linkport.doc
Last Saved 10/30/00

Below are hyperlinks, to data tables, for each DSP link port:

[MI Linkport.pdf](#)

[TV Linkport.pdf](#)

[RR Linkport.pdf](#)

Below is a hyperlink to Analog Devices instruction set reference:

http://www.analog.com/publications/documentation/ADSP_2106x_SHARC_Users_Manual/apxa.pdf

Below is a hyperlink to Analog Devices data sheet:

http://www.analog.com/pdf/ADSP-21062_L_c.pdf

Below is a hyperlink to the Embedded Systems Group home page which includes contact information.

<http://www-rfi.fnal.gov/LLRF/llrf.html>

Embedded Systems Group LLRF Documentation
 Linkport.doc
 Last Saved 10/30/00

