Notes on S3D1 Board Design

©Duy-Ky Nguyen 99-06-01

Table of Contents

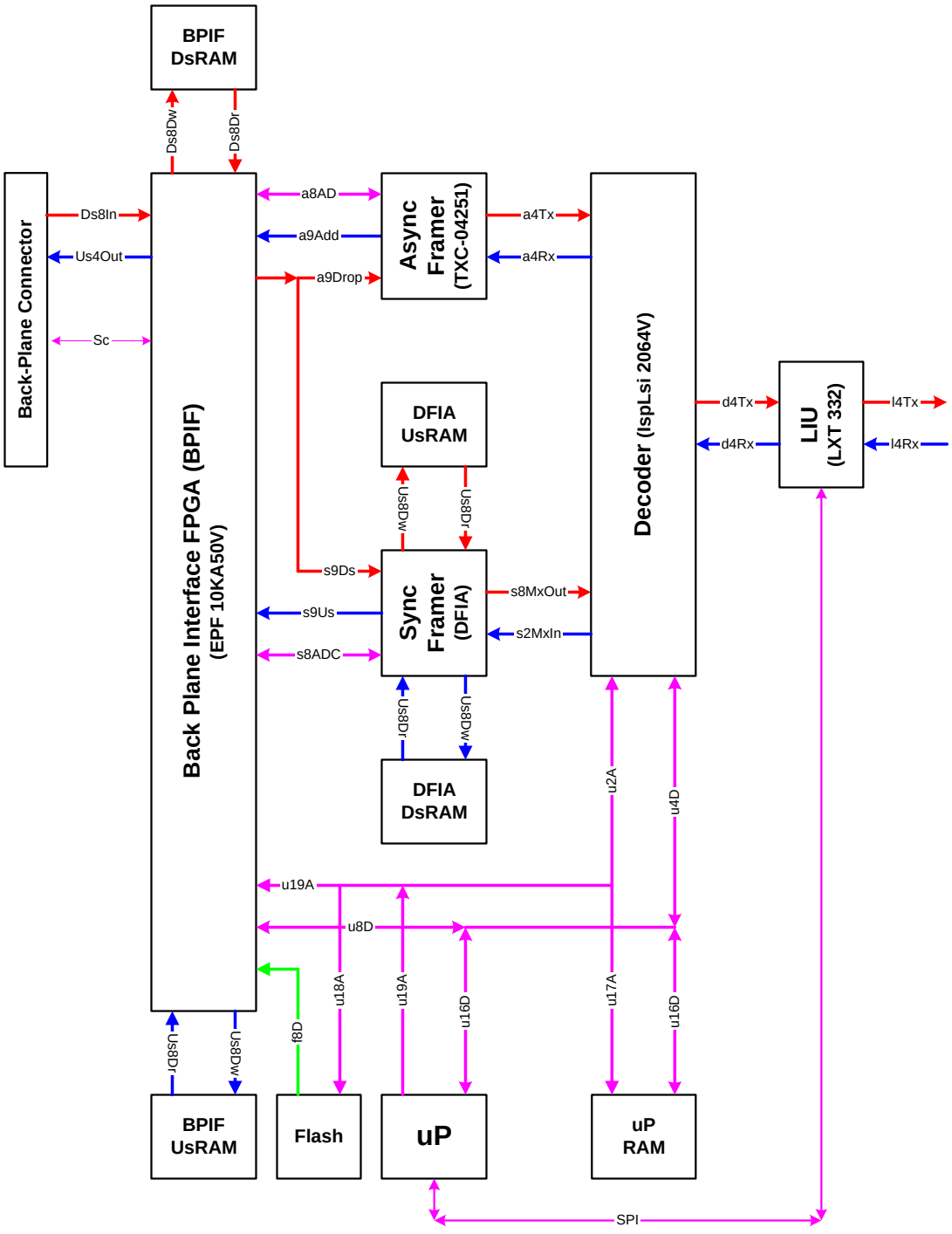
1.1.1. Abstract

This document specifies the function USAM DS1 Unit (SMDS1U). At the boundary, signal is either in synchronous TDM format or in asynchronous SONET VT1.5 format at one side (with Back-Plane), either in DSX format or in VT1.5 format, respectively, at the other side (with 4 Line Cards). This choice is set by uP.

2. Acronyms

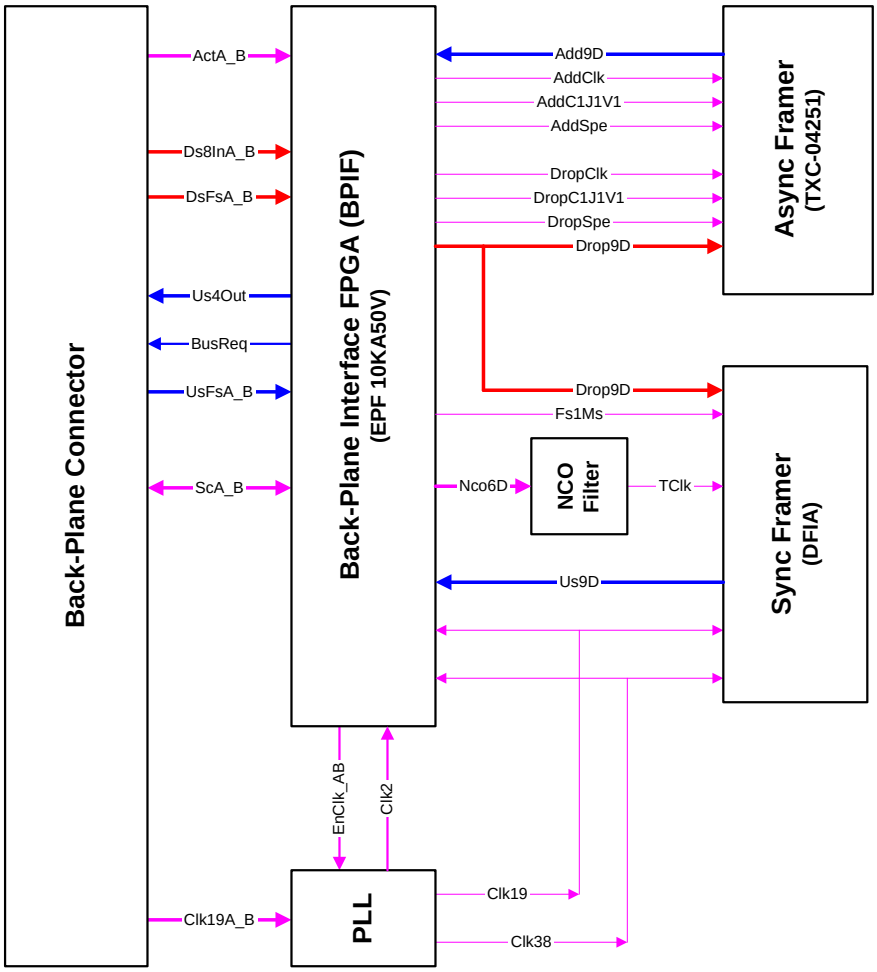
ADC	Address-Data Channel
ASIC	Application Specific Integrated Circuit
BP	Back-Plane
CuP	Central uP
DS	Down-Stream
DS0	Digital Signal 0
DS1	Digital Signal 1
DSX	Digital Signal Cross-Connect
DFIA	DS1 Framer and Interface ASIC
DuP	DFIA uP
HDLC	High-Level Data-Link Channel
LIU	Line-Card Interface Unit
LOH	Line Over-Head
POH	Path Over-Head
SL	Serial Link
SOH	Section Over-Head
TDM	Time Division Multiplexing
TOH	Transport Over-Head
US	Up-Stream
VT	Virtual Tributary

3. Block Diagram



4. Signal Interface

4.1. Back-Plane Signal Interface

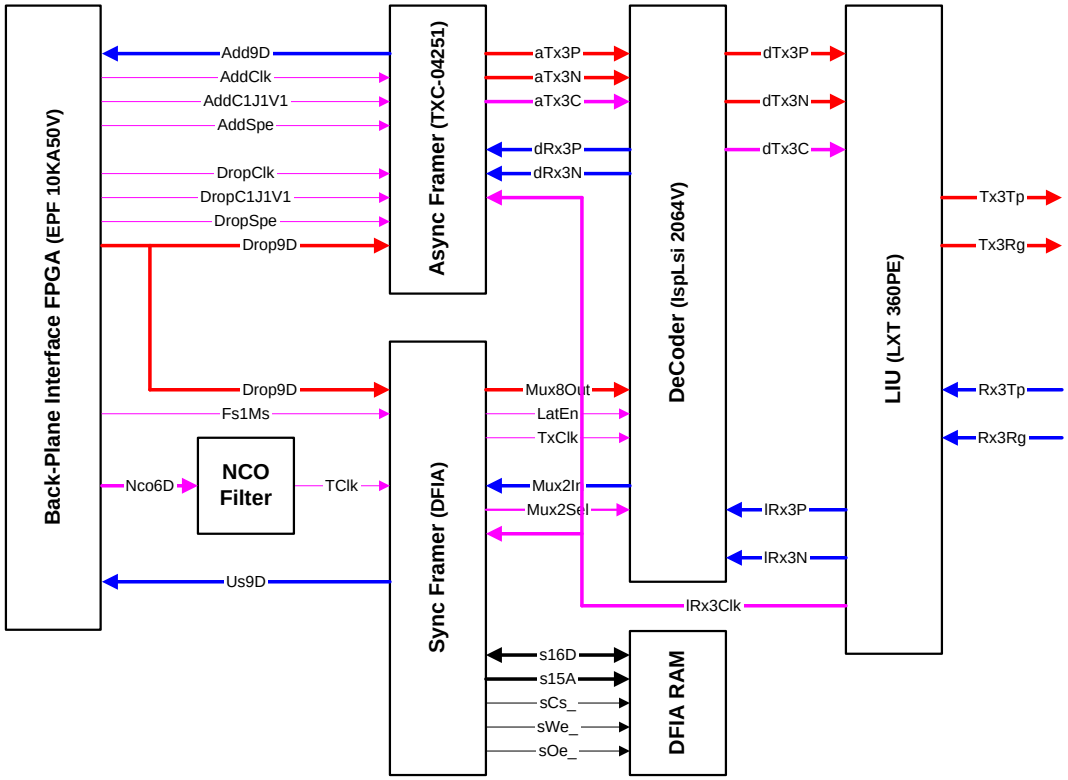


Remark 1:

- To save pin count of FPGA, 9 async Drop signals are shared with 9 sync Ds signals
- Bit 8 (MSB) of Drop bus is the parity bit.

- ActA_B select source A or B.
- DsFsA_B are used to generate frame sync of 1 mS to DFIA and to extract Ds8D from Ds8DA_B to DsRAM
- UsFsA_B is used to generate Us4D (in a frame) and BusReq from Us8D
- Clk19Ref is selected from Clk19A_B;
- TClk is extracted from Clk19A_B using NCO filter.

4.2. Line-Card Signal Interface



To save pin count in DFIA, signals between DFIA and LIU are multiplexed. The decoder is used to demultpilexed these signals to get sync signals that are counterparts of async signals aTx3P, aTx3N, dRx3P, dRx3N. This decoder is also used to select sync or async signals for LIU. Refer to Sec Control Interface below for some detail of this selection and LIU initialization.

4.3. Control Interface

4.3.1. Mode Setups

BP Interface FPGA

- MSel[1:0] = %11 (p 123-124) for PPA Config mode (from Flash under uP control).
- CS_ = 0 (p 240) to enable cfg CS (p 239) under uP control
- RS_ = 1 (p 236) to disable RdynBsy appear on cfg Dat7 pin (p 190)
- CE_ = 0 (p 178) for single FPGA (as opposed to multi FPGA)
- CLKUSR = 0 (p 11) to disable user-supplied clk for initialization

DFIA

- FrcZ_ =1 (p 1) for normal operation;
- Test = 0 (p 10) for normal operation;
- Mode = 0 (pin 144) is pulled low for Us data of 8-bit wide, rather than 4-bit;
- TbSel = 0 (pin 143). In fact, this pin is in effect if Mode of 4-bit for Us data. This case is 8-bit so it's arbitrarily pulled low (Don't Care)

LIU LXT360

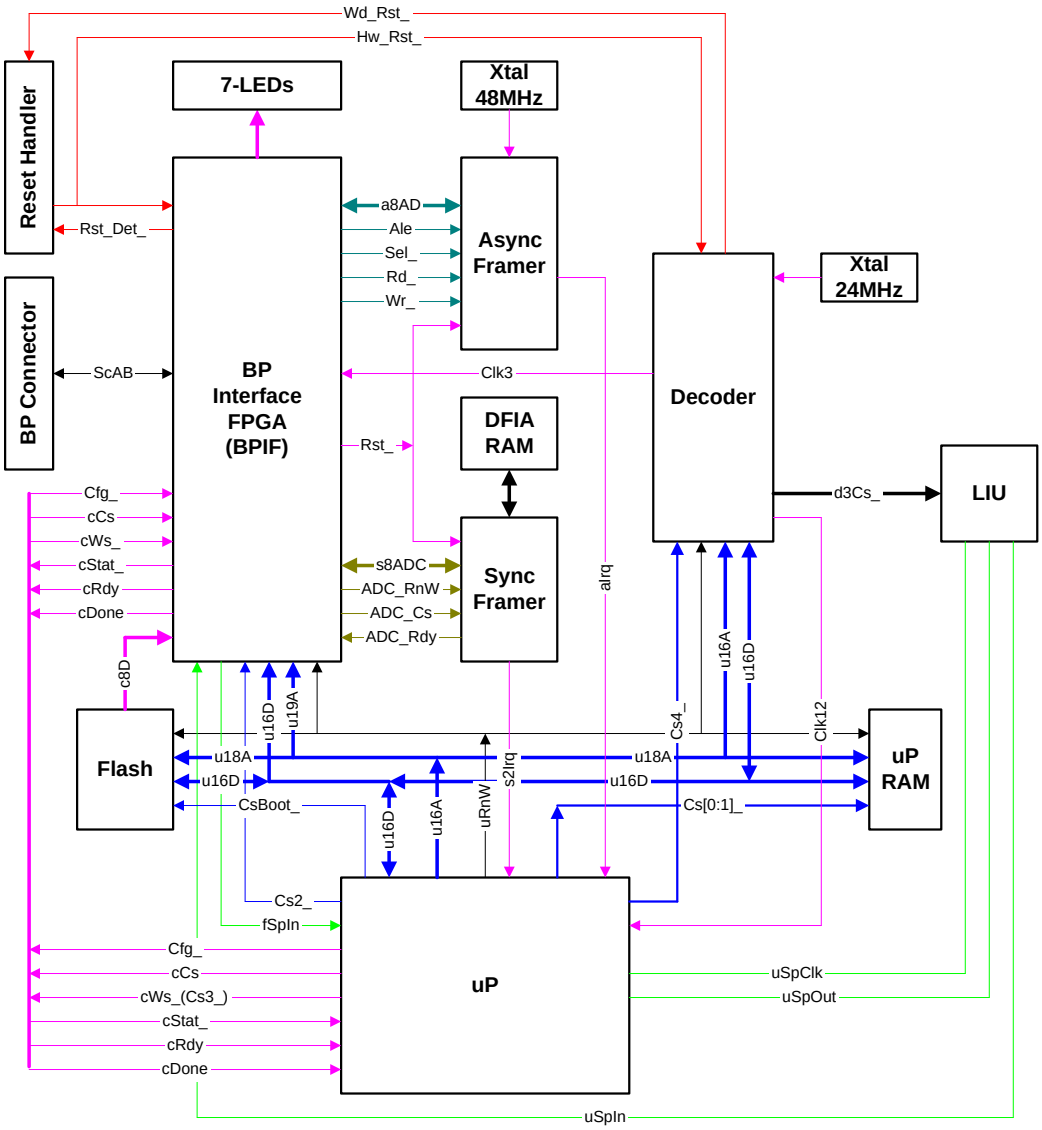
- TrSte = 0 (p 9) for normal operation;
- JaSel = 0 (p 11) for Jitter Attenuation circuit placed in the Tx path;
- ClkEdg = 1 (p 28) for RPOS/RNEG or RDATA out-puts are valid on the falling edge of RCLK, and SDO is valid on the rising edge of SCLK.

4.3.2. Reset

Common Control writes $69 @7 and $e7 @9 of BPIF to reset the board by asserting -HwRstCtl (p 184) which will activate -HwRst to reset the linecard. To reset async and sync framer, the corresponding bit must be set in HwCtlReg of BPIF (p 70). To prevent SW crashing, a watch-dog timer is implement in the decoder CPLD. To keep the line-card from reset, the watch-dog timer in CPLD must be resfreshed at most 1.4 secs.

4.3.3. Interrupts

- DFIA sends DuP_Irq and Timer_Irq to local Irq1and Irq2 of uP, respectively. Details are in Ref.2.
- BPIF Irq to Irq3 of uP
- Irq4 is reserved for BPIF HDLC Irq at the end of a message
- Async Irq to Irq5



5. On-Board Communication Protocols

There are 6 protocols used in this board:

- A bi-directional 8-bit multiplexed AD protocol is used bus between master BPIF and slave Async Framer. Its controls signals are ALE, Sel_, Rd_ and Wr_;
- Legacy NIU bi-directional 8-bit ADC protocol is used bus between master BPIF and slave DFIA. Its controls signals are ADC_RnW, ADC_CS and ADC_Rdy;
- A bi-directional serial link protocol (Ref [5]) is implemented between USAMA in master SMCC and slave BPIF. To enable SMCC to *read* from BPIF via this link, CC writes $a5 to @1 then $c3 to @3 of BPIF via this very link. Note that this requirement does not apply when SMCC writes to BPIF;
- A Mail-Box protocol is established for SMCC to access uP resource and its RAM, to initialize LIU;
- A "Mail Box" Data Link protocol is implemented in RAM at both CC and LC each with Tx DL Buf, Tx DL Reg, Rx DL Buf and Rx DL Reg;
- A SSU Data Link protocol is implemented using 23rd and 24th DS0.

5.1. Multiplexed AD

A bi-directional 8-bit multiplexed AD protocol is used bus between master BPIF and slave Async Framer. Its controls signals are ALE, Sel_, Rd_ and Wr_

1) ALE asserted in 1 clk (pulse) for latching address
2) Sel_ asserted
3) Rd_/Wr_ asserted for R/W
4) Sel_ deasserted to terminate cycle

5.2. ADC Protocol

Legacy NIU bi-directional 8-bit ADC protocol is used bus between master BPIF and slave DFIA. Its controls signals are ADC_RnW, ADC_CS and ADC_Rdy

- 8-bit bus ADC[7:0] to carry 16-bit address and 16-bit data;
- 1-bit ADC_Cs to start;
- 1-bit ADC_RnW;
- 1-bit ADC_Rdy to ack.

Remark 2: Software Transparent

This ADC protocol is hardware implemented and is transparent to software development.

Both ADC read/write cycles have 4 cycles and an additional ack cycle to complete. Both have ADC_Req asserted to start and ADC_Rdy asserted to terminate

1) ADC_Req =1 and ADC_RnW = 0 (Wr)/1(Rd) to start cycle
2) A[15:8] high address byte cycle;
3) A[7:0] low address byte cycle;

ADC Write Cycle

4w) D[15:8] high data byte write cycle;
5w) D[7:0] low data byte write cycle;
6w) Wait for ADC_Rdy =1 to terminate;

ADC Read Cycle

4r) Wait for ADC_Rdy =1 for valid read data;
5r) D[15:8] high data byte read cycle;
6r) D[7:0] low data byte read cycle;

5.3. Serial Link Protocol

Remark 3: Software Transparent

This serial link protocol is hardware implemented and is transparent to software development.

A bi-directional serial link protocol (Ref [5]) is implemented between USAMA in master SMCC and slave BPIF. To enable SMCC to *read* from BPIF via this link, CC writes $a5 to @1 then $c3 to @3 of BPIF via this very link. Note that this requirement does not apply when SMCC writes to BPIF.

The sequence is below

1. Start bit = 0
2. 4-bit Command (2: RdByte, 3: WrByte, C: RdWord, D: WrWord)
3. 16-bit address

Write Cycle

4w) 8/16-bit for byte/word data write
5w) Wait for write acked or time-out
6w) N.A.
7w) N.A.

Read Cycle

4r) Wait for read acked or time-out

If read acked

5rb)	8-bit byte read	5rw)	Status bit for word read
6rb)	N.A.	6rw)	16-bit word read
7rb)	N.A.	7rw)	Parity bit

1) Stop bit = 1

5.4. Mail-Box Protocol

Part of DFIA RAM is reserved to set up Mail-Box protocol to establish a communication between BPIF, hence SMCC via serial link, and uP resource and its peripheral such as its registers, RAM and LIU.

Offset DFIA RAM Address	Register	Size (Byte)	Description
0x200	Addr_Hi	2	A[15:8]
0x202	Addr_Lo	2	A[7:0]
0x204	Data_Wr	4	Data write
0x208	Data_Rd	4	Data read
0x20c	Cmd	2	0: Idle 1: Read byte 2: Read word 3: Read long 4: Write byte 5: Write word 6: Write long

The sequence is

1) Write A[15:8] into Addr_Hi Reg
2) Write A[7:0] into Addr_Lo Reg

Write cycle

3w) Write Data (byte/word/long) into Data_Wr Reg

Read cycle

3r) N.A.

1) Write Cmd into Cmd Reg
2) Read and Wait until Cmd Reg clear by uP to indicate that it completes the required job
3) Write cycle is completed. Read valid data to complete the read cycle.

5.5. Mail-Box Data Link Protocol

CC (via Scomm) and LC (via uP) have their own DL_Ctl, each has 3 bits: DL Enable, Message Valid and Buffer Ready. CC cannot write rDL_CTL of LC and vice versa. The Valid and Ready bits have their meaning only if Enable bit is set.

CC set CC_Vld will clear LC_Rdy, set CC_Rdy to clear LC_Vld; and vice versa.

There are Tx_DL_Buf and Rx_DL_Buf associated with Tx_DL_Ctl_Reg/ Tx_DL_Siz_Reg and Rx_DL_Ctl_Reg/ Rx_DL_Siz_Reg, respectively. This implementation has a size of 64 bytes for each buf.

This protocol is fully under software control using Tx_Ctl_Reg & Rx_Ctl_Reg for hand-shakings.

5.5.1. Message from CC to LC

CC via SComm	LC via uP
CC set Enable bit	LC set Enable bit
CC • writes message into DL_CC_Buf • set CC_Valid to tell LC message available	LC • polls CC_Valid once set; • get the message from CC_Buf if CC_Valid set • set LC_Rdy to clear CC_Valid

5.5.2. Message from LC to CC

CC via SComm	LC via uP
CC set Enable bit	LC set Enable bit
CC • polls LC_Valid once set; • get the message from LC_Buf if LC_Valid set • set CC_Rdy to clear LC_Valid	LC • writes message into DL_LC_Buf • set LC_Valid to tell CC message available

5.6. SSU Data Link Protocol

In the SSU mode, the DS1 channel will use the 23rd DS0 for the Data-Link Service Request function and the 24th DS0 (last DS0) for the byte-oriented system data-link message. There is only one data-link buffer used for both direction one at a time (half duplex data-link).

This protocol has a control reg and a status reg. The SSU DL control register (@ offset 0x4e) is used for connect this buffer to the target DS1 channel and at Tx or Rx mode (one or the other but not both). The SSU DL status register (@ offset 0x50) is used to monitor the SSU data-link buffer operation.

To transmit a message, SW needs to divide the message into segments of 64 bytes, fill up the 128-byte SSU datalink buffer before setting the SSU Tx DL enable bit. Since the buffer is capable to store 2 segments, by monitoring top bit of the buffer address (bit 6 of the SSU DL status register @ offset 0x50), SW should be able to determine the completion of a message segment so that the new segment can be refilled. The Tx DL enable bit must be cleared to "0" before a new message is transmitted before turning the buffer around for the message reception.

To receive a message, SW needs to set the SSU Rx DL enable bit. The reception won't be started until the opening flag 0x81 follows by 0x7e is detected. After that data will be sequentially written into the SSU datalink buffer. SW should read the SSU DL status register @ offset 0x50 to get the current rx pointer at bits 6:0. Bit 7 of this register must = '1' to indicate the opening flag is detected

There is only 1 byte to carry SSU message for each frame. This byte is at Ds1 specified by SSU_CTL[1:0] = 0~3 (@ 0x4e) and at Ds0 = 23 (last 24th Ds0).

5.6.1. SSU Tx (Rd Us RAM)

Data *read* from Us RAM will be sent out as TxTDM to DFIA. Their address is calculated for SSU mode counting on Blk_4 or Blk_8 mode.

4-bit SSU_CTL reg must be set with SSU_CTL [3] = 1 (ssu_txdl_en) and SSU_CTL [1:0] = 0~3 (4 Ds1) for a particular Ds1 channel to carry SSU message.

14-bit Id_Ch_x regs must be set with Id_Ch_x [13] = 1 for blk_8 or 0 for blk_4, Id_Ch_x[12:10] = 1~6 (blk_id) , Id_Ch_x [7] = 0 (sync), Id_Ch_x[6:0] < 82 (half_cell).

Given these settings, the SSU_Tx_Req will be generated at Ds1=4, Ds0=22, and this request make SSU_Tx_Cyc start at Ds0=23 to read data from Us RAM.

5.6.2. SSU Rx (Wr Us RAM)

Data received as RxTdm from DFIA will be *written* into Us RAM.

4-bit SSU_CTL reg must be set with SSU_CTL [2] = 1 (ssu_rxdl_en) and SSU_CTL [1:0] = 0~3 (4 Ds1) for a particular Ds1 channel to carry SSU message.

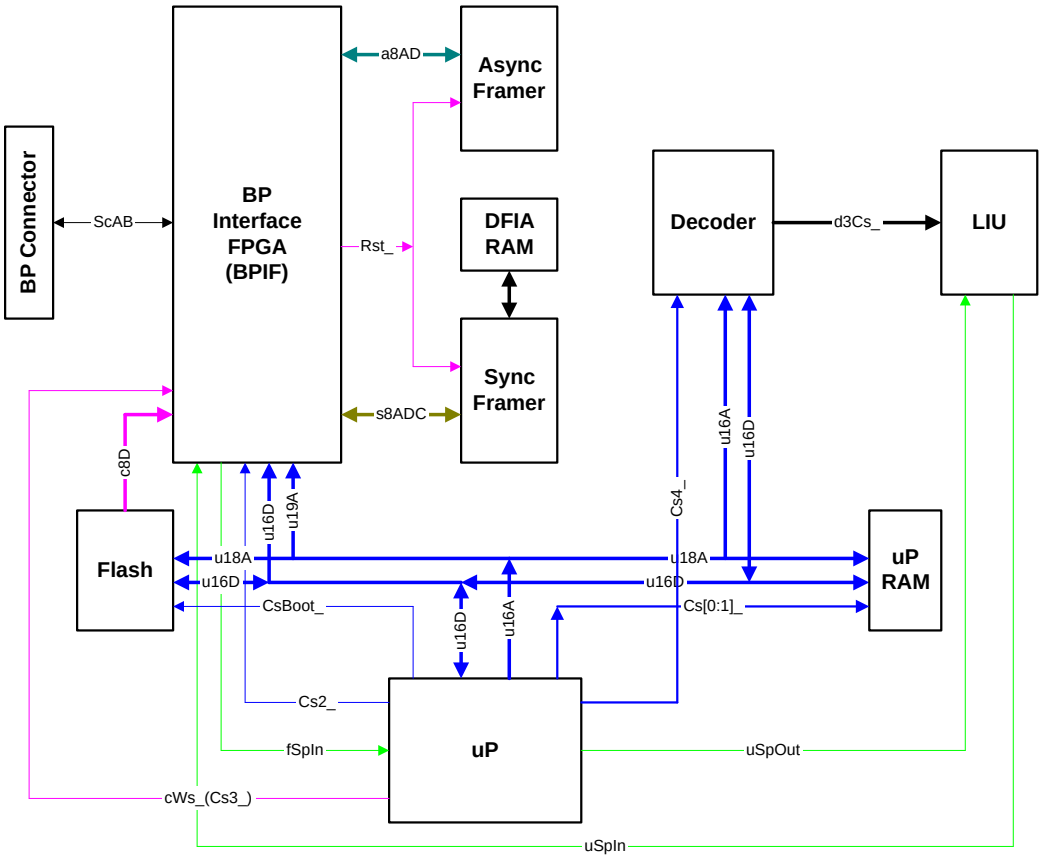
14-bit Id_Ch_x reg must be set with Id_Ch_x [13] = 1 for blk_8 or 0 for blk_4, Id_Ch_x[12:10] = 1~6 (blk_id) , Id_Ch_x [9] = 1 (UsrWe), Id_Ch_x [7] = 0 (sync), Id_Ch_x[6:0] < 82 (half_cell).

Given these settings, the SSU_Rx_Req will be generated at Ds1=5~8, Ds0=22 with *Ds0_Data = $55*. This request will generate Irq to uP; note that this request does not generate SSU_Rx_Cyc.

SSU_Tx_Cyc starts at Ds1-6 = 0~3, Ds0=23 to write data RxTdm from DFIA into Us RAM with *valid* addr is generated after receiving the opening flag that is 2 consecutive bytes of $81 and $7e.

As mentioned earlier, there is only 1 byte for SSU message each frame, so 2 frames must be elapsed before getting an opening flag. The SSU_STAT can be used to monitor SSU operation. SSU_STAT = 1 after receiving the first byte of opening flag ($81) and SSU_STAT = $82 after receiving the second ($7e). In fact, SSU_STAT[7] = SSU_Rx_Act, SSU_STAT[6:0] = SSU_Addr = 2, hence the value $82.

5.7. Resource Accessibility



uP accesses to its RAM via its bus using Cs0_ and Cs1_ for low and high memory bank. It accesses to BPIF using Cs2_. It accesses to CPLD Decoder using Cs4_. It accesses to LIU's (for LIU initialization) by Motorola SPI protocol via Decoder CPLD for Cs. It accesses to SMCC via the serial link protocol (Sec 6.3).

BPIF accesses to Async framer via the multiplexed AD bus (Sec 6.1).

The 8-bit ADC protocol (Sec.6.2) is implemented between BPIF and DFIA so BPIF can access to DFIA RAM via this link. The uP can access DFIA RAM via uP A/D buses and ADC link.

SMCC accesses to BPIF and then BPIF RAM via the serial link protocol (Sec 6.3). It can then access to DFIA and DFIA RAM through the ADC protocol.

SMCC cannot directly access uP RAM, and LIU; but it can indirectly access via the Mail-Box protocol (Sec.6.4)

6. Memory Map

Remark 4:

- All addresses and data are in Hex base.
- Detailed memory map for DFIA and its uP (DuP) should be referred to Ref [2].

Remark 5: writing both Sync and Async signals to BPIF

It's required to write both sync and async signals to BPIF to avoid floating inputs for BPIF although there is only one to be processed.

6.1. Map for MC68LK331 @ 12 MHz

Absolute Base Address	Devices	Size (Byte)	Width (Bit)	Type	Chip Select	Ack (Wait-State)
$00 0000	Flash	512 K	16	RW	CSBOOT	0(1)
$08 0000	uP RAM	256 K	16	RW	CS0: Lo-Byte CS1:Hi-Byte	0(2)
$10 0000	FPGA	64 K	16	RW	CS2	External
$20 0000	FPGA Config	128 K	8	W	CS3	0(3)
$30 0000	CPLD	2 K	3	RW	CS4	0

For 12 MHz, we have

$$t_{u\_cyc} = 84\ nS \qquad (1)$$

For CS of MC68LK331to generate inmternal DsAck, we have

$$t_{u\_rw} = (3 + n \times WS)\, t_{u\_cyc} = 252 + n \times WS \times t_{u\_cyc}\ \langle nS \rangle \qquad (2)$$

(1) A flash of 120 nS is used in this project

$$t_{f\_rd} \le 120\ \langle nS \rangle \qquad (3)$$

so the required condition is

$$t_{u\_rw} \ge 120\ \langle nS \rangle \qquad (4)$$

and we have 0 Wait-State to read this flash.

(2) For a RAM of 12 nS in reading

$$t_{m\_rd} \le 12\ \langle nS \rangle \qquad (5)$$

so the required condition is

$$t_{u\_rw} \ge 12\ \langle nS \rangle \qquad (6)$$

and we have 0 Wait-State to read this RAM.

To write this RAM

$$t_{m\_wr} \ge 10\ \langle nS \rangle \qquad (7)$$

so the required condition is

$$t_{u\_rw} \ge 10\ \langle nS \rangle \qquad (8)$$

and we have 0 Wait-State to write this RAM.

(3) For the Passive Parallel Asynchronous Configuration (PPA) of the FPGA in use, we have

$$t_{a\_wr} \ge 200\ \langle nS \rangle \qquad (9)$$

so the required condition is

$$t_{u\_rw} \ge 200\ \langle nS \rangle \qquad (10)$$

and we have 0 Wait-State to write into FPGA for configuration.

6.1.1. Flash (CS Boot @ $0)

Offset Base Address	Data	Type	Description
$00 0000	uP Boot (Top 512 K Flash)	R	Boot code for uP
$06 0000	FPGA Cfg Code (Bottom 512 K Flash)	W	Config code for FPGA (CS2)

6.1.2. CPLD Registers (CS4 @ $30,000)

Offset Base Address	Data	Type	Description
0x0001	7~4: Reserved 3~0 = 0001: CS LIU 1 0010: CS LIU 2 0011: CS LIU 3 0100: CS LIU 4	R/W	LIU CS Reg
0x0000	7~4: Reserved 3: Channel 4 2: Channel 3 1: Channel 2 0: Channel 1	R/W	Async_nSync Mode Reg 0 = Sync. 1 = Async
0x0002	7~3: Reserved 2~0 = 5	W	Watch Dog Service Reg Writing 5 to this reg at most 1.4 secs, otherwise being reset

6.2. Memory Map for FPGA

Offset Base Address	Type	Requirement
0x0 0000	R/W	FPGA Register
0x0 0E00	R/W	Async Mapper (Async Mapper Multiplexed AD Bus)
0x0 1000	R/W	Internal Ds Data/Sig RAM (within FPGA)
0x0 1180	R/W	CC to Linecard datalink buffer RAM (64 bytes within FPGA)
0x0 11c0	R/W	Linecard to CC datalink buffer RAM (64 bytes within FPGA)
0x0 1800	R/W	Internal Us Data/Sig RAM (within FPGA)
0x0 1d00	R/W	SSU datalink buffer (128 bytes within FPGA)
0x2 0000	R/W	DFIA (DFIA AD Multiplexed Bus)

FPGA Registers

Offset Base Address	Register	Data	Type	Requirement
0x0	rScomm_En_1	15:8 = 0xa5 7:0 = not used	R/W	Linecard enable byte #1
0x2	rScomm_En_2	15:8 = 0xc3 7:0 = not used	R	Linecard enable byte #2. Byte wr to address 1 = 0xa5 then addr 3 = 0xc3 will enable the linecard serial operation.
0x4	**rBoot_Code_type**	4 bit: 0x01	R	Misc Status reg. **Boot Code embedded in LC (SW)** Fixed at 0x1 for boot code available in Flash already, not downloaded from CC (16-bit support & boot code needed)
0x6	rReset_Act_1	15:8 = 0x69 7:0 = not used	R/W	Linecard reset byte #1
0x8	rReset_Act_2	15:8 = 0xe7 7:0 = not used	R	Linecard reset byte #2. Byte wr to address 7 = 0x69 then addr 9 = 0xe7 will reset the linecard.
0xE	rDs_PAR	7:0 8-bit Latched Ds Parity Error	R/W	= 0 for normal operation. Word wr of 0xFFFF to clear.
0x12	rMisc_Info	15:8 = misc info 7:0 = not used	R/W	Misc Info byte
0x14	rLC_Type	0x2	R	Linecard Type. Fixed at 0x2 (SMDS1 card)
0x40	rCfg_Ch_1	15:14 = not used 13:10 = SSU DS0 selects (1~6) 9 = US buffer wr enable 8 = DS buffer wr enable 7 = 1: Async 0: Sync 6~0 = 0~81 Valid Location	R/W	ID Channel 1. The wr enable bits set to “1” for normal operation. Clear it to “0” will stop PCM/SIG data write into buffer. This feature used during buffer test or during error condition for pattern insertion. Bits 6:0 value > 81 when channel is idle.
0x42	rCfg_Ch_2	Ch 2 ID	R/W	Same as above but for channel 2
0x44	rCfg_Ch_3	Ch 3 ID	R/W	Same as above but for channel 3
0x46	rCfg_Ch_4	Ch 4 ID	R/W	Same as above but for channel 4
0x48	rFrame_OS	10 bit 9~6 = 4-bit Sync Tx Frame Offset 5~3: 3 bit Async Rx 2~0: 3 bit Async Tx	R/W	Framer interface frame offset regs. For DFIA frame offset control. Must set to 0xa. Transwitch chip V1 Rx frame offset. Must set to 0x??; Transwitch chip V1 Tx frame offset. Must set to 0x??;

0x4A	rLED	9 bit 8~5: Ch 4:1 Yellow Alarm 4~1: Ch 4:1 Red Alarm 0: Fail	R/W	9 Alarm LEDs
0x4C	rClk_Mon_Ref	4 bit 3:2 = BP ref clk #2 select. 1:0 = BP ref clk #1 select.	R/W	Clock Reference Select to BP. = 10 for DFIA ref clk, = 11 for DFIA mon clk. Else idle = 10 for DFIA ref clk, = 11 for DFIA mon clk. Else idle Using clk select register in the DFIA for further selectioon to the channel number as source.
0x4E	rSSU_Ctl	15:4 = Not used 3 = SSU Tx DL enable. 2 = SSU Rx DL enable 1:0 = DS1 ch SSU DL connected	R/W	SSU DS1 Datalink control reg.
0x50	rSSU_Stat	15:8 = Not used 7 = RxDL in-progess flag 6:0 = Current DL buffer index	R	SSU datalink status. SW will rd DL buffer index as FIFO rd pointer during TxDL and as wr pointer durinh RxDL.
0x52-0x60		Not used		
0x62	rHw_Ctl	6 bit 5 = BNU/USAM Par Err Sel 4 = “0” to teset Framer 3 = Loopback 2 = frc_PE 1: Sc_block_en 0: PCM Enable	R/W	Control Reg = 0 for BNU, 1 for USAM = 1 for normal operation. = 1 to loop data back to framers. For HW test only. = 1 to force bus parity error for HW test only. = 1 to block the CC write into HW registers. = 1 when the linecard ready accept PCM/SIG data.
0x64	rHw_Status_rd	5 bit 15 = BP Clk 19 present 4 = Async Up-Stream Parity Error 3 = Sync Up-Stream Parity Error 2 = Power-Down 1 = Alarm Indication Signal 0 = Parity Error	R	Raw Status Read-Only Reg
0x66	rHw_Status	5 bit 15 = BP Clk 19 present 4: Async Up-Stream Parity Error 3: Sync Up-Stream Parity Error 2: Power-Down 1: Alarm Indication Signal 0: Parity Error	R/W	Latched Status Read-Write Reg. Write with “1” to the bit position will clear this bit latched error condition.
0x68	rHw_State_Ref	15:4 = N.U. 4 = 0 3 = 0 2 = 0 1 = 0 0 = 0	RW	Hardware State Reference Register User-defined PowerDown User-defined AIS User-defined ActA User-defined ActB User-defined Act
0x6A	rHw_State_Mask	15:4 = N.U. 4 = 0 3 = 0 2 = 0 1 = 0 0 = 0	RW	Hardware State Mask Register for HW State Ref Reg above PowerDown AIS ActA ActB Act 0: to mask out
0x6C	rHw_State_rd	5 bit 4: iPower_down 3: iAis 2: iActA 1: iActB 0: iOAct	R	Hardware State Reg To generate Irq if any Current State changed from Ref State based on State Mask Reg
0x6E	rIrq_En	15:11 = not used 10: DS1 ch4 DL SRQ high priority 9: DS1 ch3 DL SRQ high priority 8: DS1 ch2 DL SRQ high priority 7: DS1 ch1 DL SRQ high priority 6: DS1 ch4 DL SRQ low priority 5: DS1 ch3 DL SRQ low priority 4: DS1 ch2 DL SRQ low priority 3: DS1 ch1 DL SRQ low priority 2: StateChg_IrqEn 1: S2D_IrqEn 0: D2S_IrqEn	R/W	Interrupt Enable Reg
0x70	rIrq_State	15:11 = not used 10: DS1 ch4 DL SRQ high priority 9: DS1 ch3 DL SRQ high priority 8: DS1 ch2 DL SRQ high priority 7: DS1 ch1 DL SRQ high priority 6: DS1 ch4 DL SRQ low priority 5: DS1 ch3 DL SRQ low priority 4: DS1 ch2 DL SRQ low priority 3: DS1 ch1 DL SRQ low priority 2: StateChg_IrqEn 1: S2D_IrqEn 0: D2S_IrqEn	R/W	Interrupt State Reg Wr is used to clear only, wr 1 to a bit to clear that bit Rd is used to see Irq state
0x72	rAct_Sel	Activity Signal	R/W	Write to Clear Activity with Bit 0 = 1

		Bits 15:8 = not used 7 : iAct_bupinB 6 : iAct_usfsB 5 : iAct_dsfsB 4 : iAct_dsdatainB_all 3 : iAct_bupinA 2 : iAct_usfsA 1 : iAct_dsfsA 0 : iAct_dsdaainA_all		When = 1, these latched bits indicates the coresponding backplane signal is toggled at least one from the last clear.
0x74	rIrq_Req	2 bit 1: D2S_Irq Dup to CC interrupt req 0: S2D_Irq CC to Dup interrupt req	W Only	Write "1" to these bit position will set the coresponding interrupt bits in the interrupt reg (@0x70).
0x76 ~ 0x78		Reserved		
0x7A	rChip_Id	0x558n	R	ASIC ID rev n, n=1 to f . Current n value is 1
0x100	rDL_CC_Ctl	15:13 = not used 12 = CcDlEnable 11:10 = 00 9 = RxPkt2CcRdy 8 = TxPkt2LcRdy 7:0 = not used	CC R/W Lc R	From CC Datalink Control This provides standard USAM Common Control to Linecard datalink semaphores.
0x102	rDL_LC_Ctl	5:13 = not used 12 = LcDlEnable 11:10 = 00 9 = RxPkt2LcRdy 8 = TxPkt2CcRdy 7:0 = not used	Lc RW CC R	From Linecard Datalink Control (Same as in DSLIA) This provides standard USAM Linecard to Common Control datalink semaphores.
0x104 0x106		0x40	R	DL size = 64 bytes

7. Testing Line-Card

7.1. Programming Flash

A 512Kb flash is divided into 2 regions, the first 384Kb is for uP boot code starting from addr 0, the last 128Kb for FPGA config code starting addr $60000. This flash is of Bottom Boot Block Sector in case using Sector Erase.

Boot code in S19 format is down-loaded into RAM from addr $80000 and write into flash from addr 0. Config code is downloaded into RAM from addr $e0000 and

FPGA config is under uP control. UP asserts CsBoot_ to read config code from flash (start from addr $60000) in words and asserts Cs3_ to write to FPGA in bytes, ie. 1 uP word-read word and 2 FPGA byte-write.

Remark 6: Disable Watch-Dog Timer in CPLD

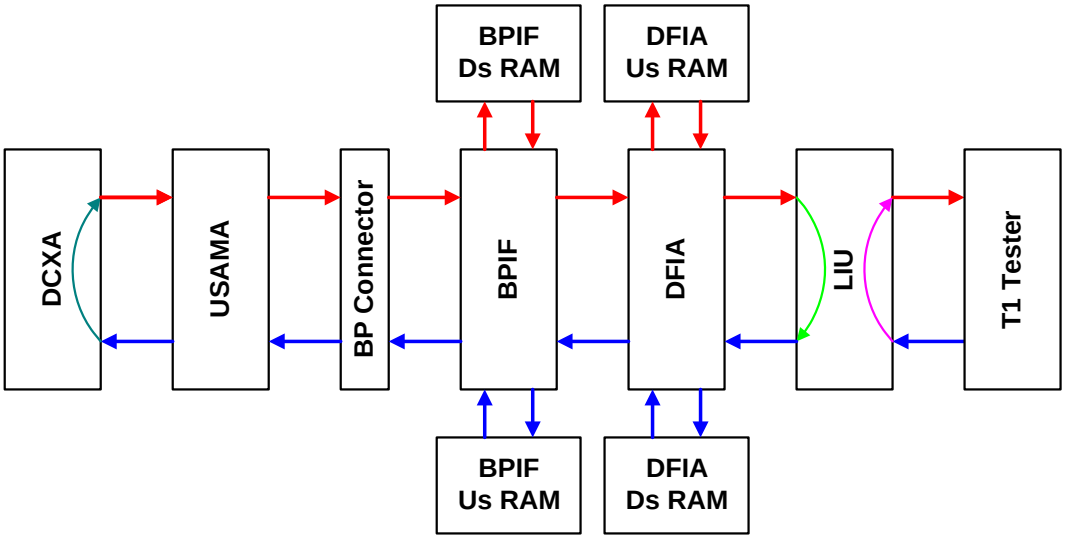
It's required to write 5 into Watch-Dog Reg in CPLD in every 1.4 secs at most; otherwise the linecard will be reset. It takes more than 1.4 secs to download boot code or config code to RAM via BDM. This process is under BDM control, ie. out of user control. Therefore, it's required to disable the watch dog timer by installing a jumper across a 2x1 header.

7.2. Hardware Test

As mentioned previously, via the serial link, SMCC can access to test BPIF RAM and DFIA RAM, but it cannot directly access to test uP RAM. As required operationally, SMCC initializes DFIA and loads application code into DFIA RAM @ physical address 0. DFIA then releases reset for uP to load code from DFIA RAM using CsBoot_. However, SMCC can indirectly testing uP RAM by including that testing code into the application code and asks uP to do it. SMCC can test uP RAM via the serial link and the Mail-Box protocols, but this test takes quite long time to complete due to the serial link.

It is possible to do a local hardware test (without SMCC) for all RAM using BDM port of uP. In doing so, it is necessary to remove a jumper (a resistor RB2 of 0 Ohm) between pin DupReset_ of DFIA (p.191) and pin Reset_ of uP (p.68); so uP can do its job by itself without waiting for the reset release from DFIA.

7.3. Functional Test



Using VxWork via the serial link, SMCC will basically do the following

- reset line-card by writing 0x69 to @0x06 and 0xe7 to @0x08 in BPIF;
- enable SMCC to read BPIF via serial link by writing 0xa5 to @0x00 and 0xc3 to @0x02 in BPIF;
- initialize HDLC_ID by writing 0x5555 to 0x78 in BPIF;
- set line-card active by writing 0xa7 then 0x59 into Local Protection Ctl Reg @0x7c to set active bit for the linecard ready for En_Us/En_LIU to take effect once enabled in Hw Ctl Reg @0x62;
- set sync/async mode via 4 Ch_ID Regs in BPIF and Async_nSync Reg in CPLD;
- initialize DFIA to have all 24 channels working with frame ESF of CRC-6, B8ZS code and 2nd column of the three is selected for TDM time-slot in a 27 row x 90 col frame;
- load boot code into DFIA RAM with code for Mail-Box protocol to establish a link between SMCC and uP (via serial link) and uP RAM testing code if so required;
- release reset for uP by setting HwCtlReg in DFIA;
- initialize LIU in dual loopback mode (local & remote loopback). Remote loopback mode will check line transformer;
- generate some random patterns as a reference and writes these patterns into Us Data Link Buf of DFIA; enables DFIA to send. Due to local loopback at LIU, these pattern will go to Ds Data Link Buf of DFIA. Read and verify this buf with reference to check the path from Us DFIA → local-loopbacked LIU → Ds DFIA;
- load a mapping between 82 half-cells and 24 DS1 into the DS1 Src Ptr (LUT) @ 0x01 1800;
- enable TDM slot in USAMA ready for DCXA to communicate with the linecard through USAMA;
- generate some random patterns as a reference and writes these patterns into Data Buf of DCXA; enables DCXA to send. Read and verify Ds Data Buf of BPIF with reference and verify with the reference to check the path from DCXA → USAMA → BP Connector → Ds BPIF. These patterns continue to go to Us DFIA, local loopbacked LIU, back to Ds DFIA and Us BPIF. Read and verify Us Data Buf of BPIF with the referenece to check this path. These patterns then continue to go back to DCXA. Read and verify Sig Buf of DCXA with the reference to check this last path;
- initialize LIU in normal mode (transparent without any loopback) and loopbacl DCXA to check the whole path T1-Tester → LIU → Ds DFIA → Us BPIF → USAMA → Loopbacked DCXA → USAMA → Ds BPIF → Us DFIA → LIU → T1-Tester.

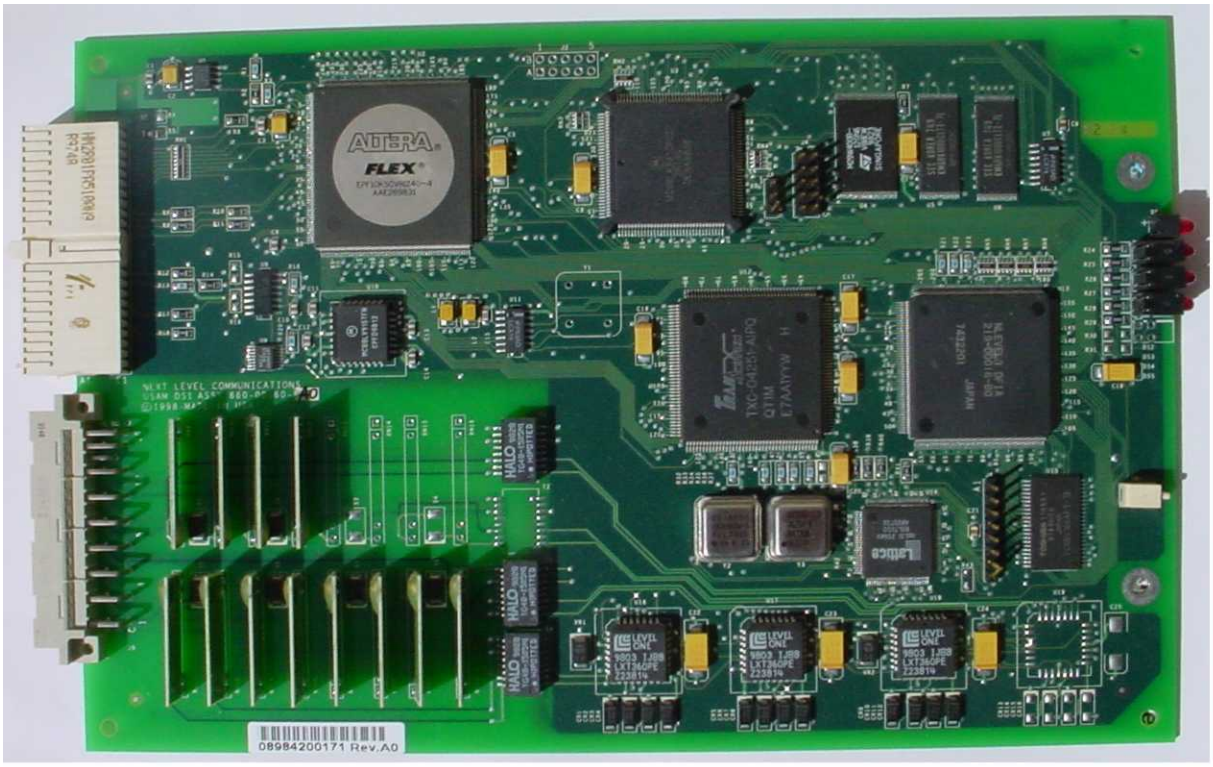
8. Pin Description for EPF 10K50VR-240

Pin	Type	Signal	Description
1	I	TCk	N.U. (Pull High)
2	IO (O.D.)	Cfg_Done	High once config is completed
3	O	-CEO	N.U.
4	O	TDO	Test Data Out: N.U.
5	P	Vdd	3.3 v
6	I	DsB4	DsB[4] from BP Conn
7	I	DsB0	DsB[0] from BP Conn
8	I	ActA	ActA from BP Conn to select side A
9	I	DsA4	DsA[4] from BP Conn
10	P	Gnd	0 v
11	I	ClkUsr	N.U. (pull low)
12	I	DsB0	DsB[0] from BP Conn
13	I	DsB5	DsB[5] from BP Conn
14	I	DsB1	DsB[1] from BP Conn
15	I	DsA5	DsA[5] from BP Conn
16	P	Vdd	3.3 v
17	I	DsA1	DsA[1] from BP Conn
18	I	DsB6	DsB[6] from BP Conn
19	I	DsB2	DsB[2] from BP Conn
20	I	DsA6	DsA[6] from BP Conn
21	I	DsA2	DsA[2] from BP Conn
22	P	Gnd	0 v
23	O	RdynBsy	Ready to get Cfg data
24	I	DsB7	DsB[7] from BP Conn
25	I	DsB3	DsB[3] from BP Conn
26	O (O.D.)	Init_Done	N.U.
27	P	Vdd	3.3 v
28	I	ActB	ActB from BP Conn to select side B
29	I	DsA7	DsA[7] from BP Conn
30	I	DsA3	DsA[3] from BP Conn
31	I	UsFsB	Us Frame Sync for side B
32	P	Gnd	0 v
33	I	UsFsA	Us Frame Sync for side A
34	I	DsFsB	Ds Frame Sync for side B
35	I	DsFsA	Ds Frame Sync for side B
36	O	Us0	Us[0] to BP Conn
37	P	Vdd	3.3 v
38	O	Us1	Us[1] to BP Conn
39	O	Us2	Us[2] to BP Conn
40	O	Us3	Us[3] to BP Conn
41	O	BusReq	Us Bus Req to BP Conn
42	P	Gnd	0 v
43	IO	ScA	Serial link with BP Conn for side A
44	IO	SCB	Serial link with BP Conn for side B
45	O	fD5	D[5] of NCO Filter for Clk1.544 MHz
46	O	fD4	D[4] of NCO Filter for Clk1.544 MHz
47	P	Vdd	3.3 v
48	O	fD3	D[3] of NCO Filter for Clk1.544 MHz
49	O	fD2	D[2] of NCO Filter for Clk1.544 MHz
50	O	fD1	D[1] of NCO Filter for Clk1.544 MHz
51	O	fD0	D[0] of NCO Filter for Clk1.544 MHz
52	P	Gnd	0 v
53	I	Clk1544	Clk 1.544 MHz after NCO Filter
54	O	IO	N.U.
55	I	EnClk19A	En Clk19 from BP Conn for side A
56	I	EnClk19B	En Clk19 from BP Conn for side B
57	P	Vdd	3.3 v
58	I	TMS	Test Mode Select: Pull High to disable JTAG Circuit
59	I	TRst	Test Reset: Pull Low to disable JTAG Circuit
60	IO (O.D.)	-Status	Cfg Status for config error
61	IO	AD0	Async Multiplexed AD[0]
62	IO	AD1	Async Multiplexed AD[1]
63	IO	AD2	Async Multiplexed AD[2]
64	IO	AD3	Async Multiplexed AD[3]
65	IO	AD4	Async Multiplexed AD[4]
66	IO	AD5	Async Multiplexed AD[5]
67	IO	AD6	Async Multiplexed AD[6]
68	IO	AD7	Async Multiplexed AD[7]
69	P	Gnd	0 v
70	O	-Rst_Ctl	Reset to Sync and Async Framers
71	O	-AD_Sel	Cs to Async AD bus
72	O	-AD_Rd	En Rd to Async AD bus
73	O	-AD_Wr	En Wr to Async AD bus
74	O	AD_Ale	En Latch Addr to Async AD bus
75	I	-AddInd	NU: VT/TU time-slot indication from Async for Add bus
76	I	-DropInd	NU: VT/TU time-slot indication from Async for Drop bus
77	P	Vdd	3.3 v

78	O	DropClk	Clk to Async Drop bus
79	O	AddClk	Clk to Add Drop bus
80	O	AddC1J1V1	C1J1V1 pulse to Async for Add bus
81	O	AddSpe	SPE indication to Async for Add bus
82	O	DropC1J1V1	C1J1V1 pulse to Async for Drop bus
83	O	DropSpe	SPE indication to Async for Drop bus
84	I	AddD0	D[0] from Async for Add bus
85	P	Gnd	
86	I	AddD1	D[1] from Async for Add bus
87	I	AddD2	D[2] from Async for Add bus
88	I	AddD3	D[3] from Async for Add bus
89	P	Vdd	3.3 v
90	I	Clk19B	Clk19 from BP side B
91	I	Clk38	Clk38 from PLL
92	I	Clk19A	Clk19 from BP side A
93	P	Gnd	0 v
94	I	AddD4	D[4] from Async for Add bus
95	I	AddD5	D[5] from Async for Add bus
96	P	Vdd	3.3 v
97	I	AddD6	D[6] from Async for Add bus
98	I	AddD7	D[7] from Async for Add bus
99	I	AddD8	D[8] from Async for Add bus
100	I	-AddVld	Valid data from Async Add bus
101	O	Ds_DrA0	Ds[0] to Sync Us and Async Drop
102	O	Ds_DrA1	Ds[1] to Sync Us and Async Drop
103	O	Ds_DrA2	Ds[2] to Sync Us and Async Drop
104	P	Gnd	0 v
105	O	Ds_DrA3	Ds[3] to Sync Us and Async Drop
106	O	Ds_DrA4	Ds[4] to Sync Us and Async Drop
107	O	Ds_DrA5	Ds[5] to Sync Us and Async Drop
108	O	Ds_DrA6	Ds[6] to Sync Us and Async Drop
109	O	Ds_DrA7	Ds[7] to Sync Us and Async Drop
110	O	Ds_DrA8	Ds[8] to Sync Us and Async Drop
111	I	UsTdm0	Us[0] from Sync Ds
112	P	Vdd	3.3 v
113	I	UsTdm1	Us[1] from Sync Ds
114	I	UsTdm2	Us[2] from Sync Ds
115	I	UsTdm3	Us[3] from Sync Ds
116	I	UsTdm4	Us[4] from Sync Ds
117	I	UsTdm5	Us[5] from Sync Ds
118	I	UsTdm6	Us[6] from Sync Ds
119	I	UsTdm7	Us[7] from Sync Ds
120	I	UsTdm8	Us[8] from Sync Ds
121	I	-Config	Start config from uP
122	P	Vdd	3.3 v
123	I	MSel1	High for PPA config from Flash
124	I	MSel0	High for PPA config from Flash
125	P	Gnd	0 v
126	IO	Adc8	Adc[8] with Sync Framer
127	IO	Adc7	Adc[7] with Sync Framer
128	IO	Adc6	Adc[6] with Sync Framer
129	IO	Adc5	Adc[5] with Sync Framer
130	P	Vdd	3.3 v
131	IO	Adc4	Adc[4] with Sync Framer
132	IO	Adc3	Adc[3] with Sync Framer
133	IO	Adc2	Adc[2] with Sync Framer
134	IO	Adc1	Adc[1] with Sync Framer
135	P	Gnd	0 v
136	IO	Adc0	Adc[0] with Sync Framer
137	O	TxClk	Clk 1.544 MHz to Sync Framer for Tx
138	O	Fs1ms	Frame sync 1 mS to Sync Framer
139	O	AdcCs	Cs to Sync Framer for Adc bus
140	P	Vdd	3.3 v
141	I	AdcRdy	Ready for Adc from Sync Framer
142	I	A18	A[18] from uP
143	I	A17	A[17] from uP
144	I	A16	A[16] from uP
145	P	Gnd	0 v
146	I	A15	A[15] from uP
147	I	A14	A[14] from uP
148	I	A13	A[13] from uP
149	I	A12	A[12] from uP
150	P	Vdd	3.3 v
151	I	A11	A[11] from uP
152	I	A10	A[10] from uP
153	I	A9	A[9] from uP
154	I	A8	A[8] from uP
155	P	Gnd	0 v
156	I	A7	A[7] from uP
157	I	A6	A[6] from uP

158	I	A5	A[5] from uP
159	I	A4	A[4] from uP
160	P	Vdd	3.3 v
161	I	A3	A[3] from uP
162	I	A2	A[2] from uP
163	I	A1	A[1] from uP
164	I	A0	A[0] from uP
165	P	Gnd	0 v
166	IO	D0	D[0] with uP & Flash
167	IO	D1	D[1] with uP & Flash
168	IO	D2	D[2] with uP & Flash
169	IO	D3	D[3] with uP & Flash
170	P	Vdd	3.3 v
171	IO	D4	D[4] with uP & Flash
172	IO	D5	D[5] with uP & Flash
173	IO	D6	D[6] with uP & Flash
174	IO	D7	D[7] with uP & Flash
175	IO	Dat0/D8	D[8] with uP & Flash, Cfg_Dat[0] for PPA (Cfg Flash)
176	P	Gnd	0 v
177	I	TDI	Test Data In: Pull Low to disable JTAG Circuit
178	I	-CE	Pull Low
179	I	DClk	Cfg Clk for PPS (Cfg Cable)
180	I	Dat0	Serial Data for PPS (Cfg Cable)
181	IO	Dat1/D9	D[9] with uP & Flash, Cfg_Dat[1] for PPA (Cfg Flash)
182	IO	Dat2/D10	D[10] with uP & Flash, Cfg_Dat[2] for PPA (Cfg Flash)
183	IO	Dat3/D11	D[11] with uP & Flash, Cfg_Dat[3] for PPA (Cfg Flash)
184	O	-HwRstCtl	To Reset Handler for HwRst
185	IO	Dat4/D12	D[12] with uP & Flash, Cfg_Dat[4] for PPA (Cfg Flash)
186	IO	Dat5/D13	D[13] with uP & Flash, Cfg_Dat[5] for PPA (Cfg Flash)
187	I	-HwRst	Reset from Reset Handler
188	IO	Dat6/D14	D[14] with uP & Flash, Cfg_Dat[6] for PPA (Cfg Flash)
189	P	Vdd	3.3 v
190	IO	Dat7/D15	D[15] with uP & Flash, Cfg_Dat[7] for PPA (Cfg Flash)
191	O	IO	NU
192	O	IO	NU
193	O	IO	NU
194	O	IO	NU
195	O	IO	NU
196	O	IO	NU
197	P	Gnd	0 v
198	O	IO	NU
199	I	-L2CIrq1	Local-to-Central Irq[1] from uP
200	I	-L2CIrq0	Local-to-Central Irq[0] from uP
201	I	-AD_Cs	Cs for A bus and D bus from uP
202	O	-DSAck0	DSAck0 to uP
203	O	-DSAck1	DSAck1 to uP
204	I	-AS	AS from uP
205	P	Vdd	3.3 v
206	I	Siz0	Siz[0] from uP
207	I	Siz1	Siz[1] from uP
208	I	RnW	RnW from uP
209	I	-Dev_Clr	High: NU
210	I	Ded_In	Must High: NU
211	I	Clk19	Clk19 from PLL
212	I	Ded_In	Must High: NU
213	I	Dev_OE	High: NU
214	I	IO	High: NU
215	I	Clk3	Clk3 from CPLD to select A/B of BP
216	P	Gnd	0 v
217	O	-M2LIrq	Central-to-Local Irq to uP
218	O	IO	NU
219	I	LSpiIn_In	SpiIn from LIU
220	O	SpiIn_Out	SpiIn to uP after TTL-CMOS adaptation
221	O	YelLed3	To Yel LED for channel 3, NU
222	O	RedLed3	To Red LED for channel 3, NU
223	O	YelLed0	To Yel LED for channel 0
224	P	Vdd	
225	O	RedLed0	To Red LED for channel 0
226	O	YelLed2	To Yel LED for channel 2
227	O	RedLed2	To Red LED for channel 2
228	O	YelLed1	To Yel LED for channel 1
229	O	RedLed1	To Red LED for channel 1
230	O	FailedLed	To Failed LED for channel 0
231	I	SMonClk	Mon Clk from Sync Framer
232	P	Gnd	0 v
233	I	SRefClk	Ref Clk from Sync Framer
234	O	MonClk	Mon Clk to BP
235	O	RefClk	Ref Clk fto BP
236	I	-RS	High, NU
237	I	IO	NU

238	I	-WS	Wr Cfg from Cs3 of uP
239	I	CS	Cs Cfg from uP
240	I	-CS	Low for Cfg


ALTERA
FLEX
08984200171 Rev.A0