PM4351

COMET

PROGRAMMER'S GUIDE

ISSUE 4: SEPTEMBER 2000

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

This document introduces the reader to the programmable features of the COMET by providing the register accesses necessary to configure the COMET. It also provides pseudo code, flow charts, figures and tables for programming the real-time aspects of the COMET.

This document is a supplement to the COMET datasheet. Due to the vast number of configurations the COMET may have, this document may not cover all possible configurations. Please contact a PMC-Sierra Applications Engineer for specific uses not covered in this document.

The COMET can be programmed for either T1 or E1 interfaces. On power-up, the default register values must be modified via software to successfully use the part. The COMET interfaces to software via registers and interrupts. Therefore, this document discusses the register accesses and interrupt processing that is necessary to apply the COMET to user applications.

Although every effort has been taken to ensure that this document is consistent with the datasheet, some errors may occur. Where there are discrepancies with this document, the datasheet and datasheet errata (if any) will take precedence.

1.1 Target Audience

This document has been prepared for Engineers that design-in the COMET and need a quick reference on how to program the COMET. It is assumed the reader is familiar with E1, T1 and framing technologies.

1.2 Numbering Conventions

Binary	0001B, 1110B
Decimal	198, 234, 2
Hexadecimal	0x2F

1.3 Pseudo-Code

The pseudo-code in this document is shown as a C-like syntax. The code segments are not intended to be compiled as shown but are shown to help the reader's understanding for a procedure or concept. The programmer may use the code as a basis for programming the COMET.

Text enclosed by the "< >" characters represent descriptions of what needs to be substituted in the location. For Example:

```
WRITE_REG(<CHANNEL_INDIRECT_ADDRESS>, <offset>);
```

2 REGISTER DESCRIPTION

Unless otherwise specified, COMET registers are shown in this document using the convention **REGISTER_NAME**(register number from datasheet).

All register accesses are shown in this document as:

```
READ_REG(<address>)

WRITE_REG(<address>, <value>)
```

whereby <address> is the register number and <value> is the data written to a register.

The COMET Registers have the following characteristics:

- All values written into unused register bits should be written with logic 0 unless otherwise stated, ensuring software compatibility with future versions of the product. Reading back unused bits can produce either logic one or a logic zero; hence, unused register bits should be masked off by software during a register read access.

- Certain register bits are reserved. To ensure that the COMET operates as intended, reserved register bits must only be written with their default values unless otherwise stated in the datasheet.

The COMET has 2 types of register spaces -- the “normal” registers, which are accessed directly by the microprocessor bus interface of the COMET, and the “Indirect” registers which are internal to the COMET. Through multiple writes to normal registers and polling of a BUSY bit, indirect registers are accessed, ensuring data is visible at the microprocessor bus or written into a normal register stored internal to the COMET.

2.1 Normal Mode Registers

Normal mode registers configure the operation of the COMET. The registers have offsets between 0x00 and 0xFF inclusive. Writable normal mode registers are cleared to logic zero upon reset unless otherwise noted. The reader should refer to the datasheet for the default values of these registers.

2.2 Indirect Registers

The indirect registers are for per-channel control, which is normally disabled by default, on power-up. The PCCE bit in the corresponding per-channel block must be set to enable per channel control.

The "IND" bit in the corresponding per-channel block must be set before the software can access indirect registers. On power-up or software reset, this bit is clear. See Table 1 for the location of the PCCE and IND bits for each per-channel block.

Table 1: Location of PCCE and IND Bits

Bits	Register Number		
	TPSC	RPSC	SIGX
PCCE and IND	0x6C	0x70	0x50

Once the "IND" bit is set, the following pseudo code shows how to read from and write to an indirect register. See Table 2 for explanation of pseudo code register names.

```
#define MAX_BUSY_READ   0x5
#define BIT_BUSY        0x80

/** to read an indirect register: **/

int ReadIndirectReg(int offset, unsigned char &value) {
  int i;
  WRITE_REG(REG_CHANNEL_INDIRECT_ADDRESS, (offset|0x80));
  for (i = 0; i < MAX_BUSY_READ; i++) {
      value = READ_REG(REG_MICRO_ACCESS_STATUS);
      if ((value&BIT_BUSY) == 0) {
          value = READ_REG(REG_CHANNEL_INDIRECT_DATA_BUFFER);
          return SUCCESS;
      }
  }
  return FAILURE;
}
```

```
/** to write to an indirect register: **/

int WriteIndirectReg(int offset, unsigned char value) {
  unsigned char temp;
  int i;
  WRITE_REG(REG_CHANNEL_INDIRECT_DATA_BUFFER, value);
  WRITE_REG(REG_CHANNEL_INDIRECT_ADDRESS, (offset&0x7F));
  for (i = 0; i < MAX_BUSY_READ; i++) {
      temp = READ_REG(REG_MICRO_ACCESS_STATUS);
      if ((temp&BIT_BUSY) == 0) return SUCCESS;
  }
  return FAILURE;
}
```

Table 2: Indirect Access Registers

Register Define	Register Number		
	TPSC	RPSC	SIGX
REG_MICRO_ACCESS_STATUS	0x6D	0x71	0x51
REG_CHANNEL_INDIRECT_DATA_BUFFER	0x6F	0x73	0x53
REG_CHANNEL_INDIRECT_ADDRESS	0x6E	0x72	0x52

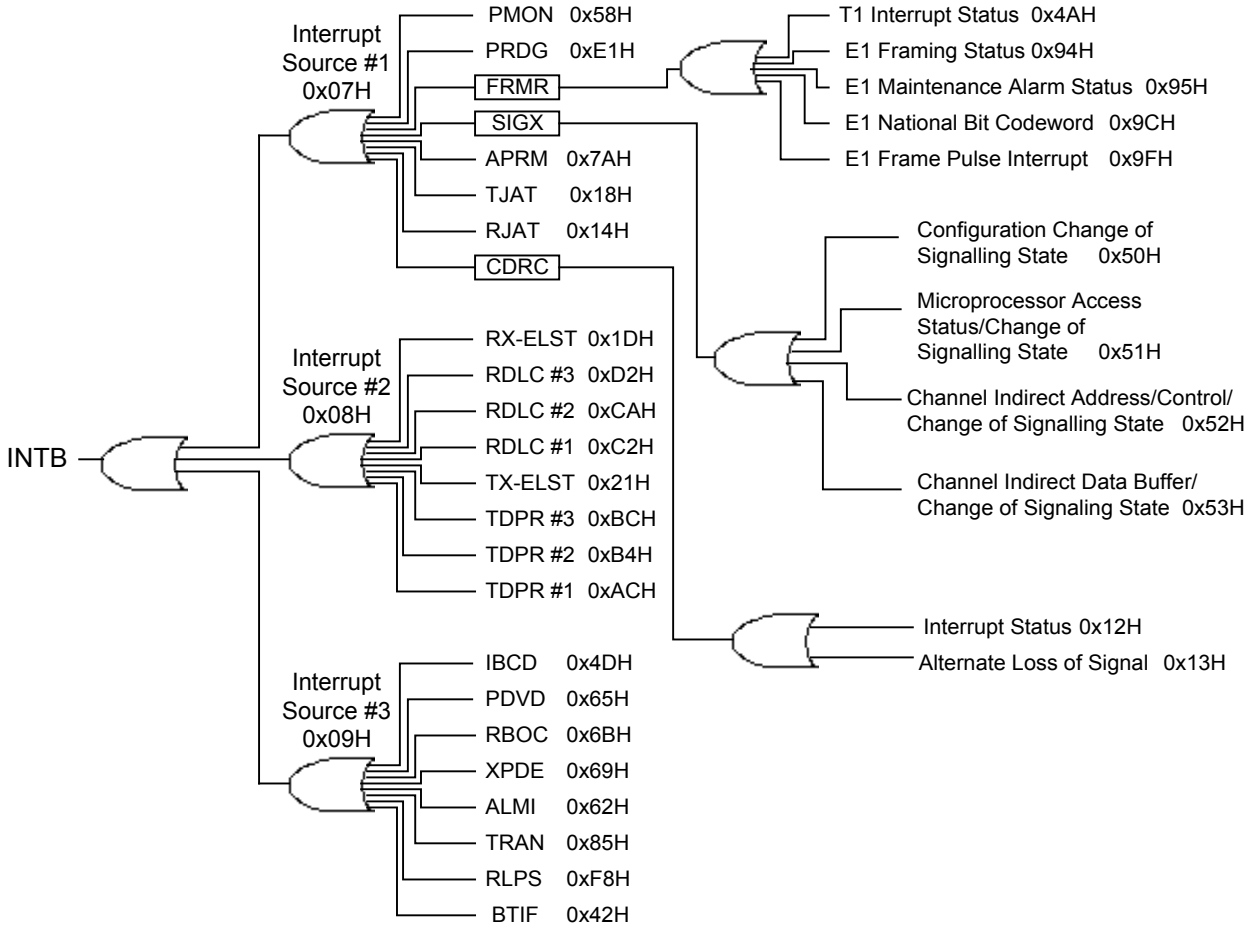
3 INTERRUPTS

The COMET has a single interrupt pin (INTB) reflecting the status of any independent interrupt sources. This section describes the COMET interrupt hierarchy.

3.1 Interrupts Overview

Figure 1 shows all registers containing interrupt bits for the COMET. An interrupt, due to any of the interrupt status bits, is reflected on the COMET INTB pin, only if the interrupt status bit associated is enabled.

Figure 1: COMET Interrupt Register Structure



The interrupt/alarm hierarchy is shown in another section in this document.

3.2 Interrupt Status and Interrupt Enables

Every interrupt source has one or more associated interrupt status bits, or "I" bit. The "I" bit is set and latched when the interrupt event occurs, and will clear only after a processor read from the register. For example: the **RLPS Configuration and Status**(0xF8) register has an enable bit (ALOSE) and an associated interrupt status bit (ALOSI). When a signal loss event occurs, ALOSI is set. ALOSI will clear only after a processor reads from register .

To filter unused interrupts, every "I" bit has an associated interrupt enable bit, or "E" bit. When the "E" bit is clear, the associated "I" bit will not contribute to the INTB pin nor to an interrupt source bit. When the "E" bit is cleared, the associated "I" bit will have no effect on INTB or top-level interrupts. Note that the "E" bit does not affect the behaviour of the "I" bit itself. For example, the LOSE bit (reg 0x11) is the interrupt enable for LOSI. If LOSE is cleared, the LOSI bit continues to indicate changes in the LOSV bit but never contributes to an INTB pin signal being active.

All interrupt "E" bits are disabled on reset of the COMET; therefore, none of the "I" bits will contribute to drive the INTB pin active unless the "E" bits are set under program control. The user would typically set the desired "E" bits on ACTIVATION of the COMET (see section 4.3).

3.3 Interrupt Value Bits

Many interrupt status bits, but not all, have an associated value bit, or "V" bit that indicates the current state (See Table 3).

3.4 Interrupt Sources

Every interrupt is grouped into one of the interrupt source bits. These top level bits represent the logical "or" of all enabled interrupts in that group, and allow an interrupt service routine to read which group of interrupts is active without clearing an associated "I" bit. Top level interrupts cannot be cleared directly; instead, they will clear when the underlying interrupts are cleared (See Table 3 for interrupt groups).

3.5 Interrupt Reference

The Table 3 below references all interrupt bits in the register they appear and the location and name of the corresponding enable/disable and status bit. The table is shown in the order displayed by Figure 1.

Table 3: Interrupt Reference

Interrupt		Enable/Disable		Status		Comments
Register	**"I" Bit or source group**	**Register**	**"E" Bit**	**Register**	**"V" Bit**	
Interrupt Source #1 0x07	PMON	------	------	------	------	This is a top level interrupt source register so no enable/disable or status bits exist. Individual "I" bits belonging to a group are shown later in this table.
	PDRG	------	------	------	------	
	FRMR	------	------	------	------	
	SIGX	------	------	------	------	
	APRM	------	------	------	------	
	TJAT	------	------	------	------	
	RJAT	------	------	------	------	
	CDRC	------	------	------	------	
Interrupt Source #2 0x08	RX-ELST	------	------	------	------	This is a top level interrupt source register so no enable/disable bits exist. Individual "I" bits belonging to a group are shown later in this table.
	RDLC #3	------	------	------	------	
	RDLC #2	------	------	------	------	
	RDLC #1	------	------	------	------	
	TX-ELST	------	------	------	------	
	TDPR #3	------	------	------	------	
	TDPR #2	------	------	------	------	
	TDPR #1	------	------	------	------	
Interrupt Source #3 0x09	IBCD	------	------	------	------	This is a top level interrupt source register so no enable/disable bits exist. Individual "I" bits belonging to a group are shown later in this table.
	PDVD	------	------	------	------	
	RBOC	------	------	------	------	
	XPDE	------	------	------	------	
	ALMI	------	------	------	------	
	TRAN	------	------	------	------	
	RLPS	------	------	------	------	
	BTIF	------	------	------	------	

Interrupt		Enable/Disable		Status		Comments
Register	"I" Bit or source group	Register	"E" Bit	Register	"V" Bit	
PMON 0x58	XFER	0x58	INTE	------	------	
	OVER			------	------	
PRDG 0xE1	SYNCI	0xE1	SYNCE	0xE1	SYNCV	
	BEI		BEE	------	------	
	XFERI		XFERE	------	------	
T1 FRMR Interrupt Status 0x4A	COFAI	0x49	COFAE	------	------	
	FERI		FERE	------	------	
	BEEI		BEEE	------	------	
	SFEI		SFEE	------	------	
	MFPI		MFPE	0x4A	MFP	
	INFRI		INFRE		INFR	
E1 FRMR Framing Status 0x94	C2NCIWI	0x92	C2NCIWE	0x96	C2NCIWV	
	OOFI		OOFE		OOFV	
	OOSMFI		OOSMFE		OOSMFV	
	OOCMFI		OOCMFE		OOCMFV	
	COFAI		COFAE	------	------	
	FERI		FERE	------	------	
	SMFERI		SMFERE	------	------	
	CMFERI		CMFERE	------	------	

Interrupt		Enable/Disable		Status		Comments
Register	"I" Bit or source group	Register	"E" Bit	Register	"V" Bit	
E1 FRMR Maintenance Alarm Status 0x95	RAII	0x93	RAIE	0x97	RAIV	
	RMAII		RMAIE		RMAIV	
	AISDI		AISDE		AISD	
	REDI		REDE		RED	
	AISI		AISE		AIS	
	FEBEI		FEBEE	------	------	
	CRCEI		SMFERE	------	------	
E1 FRMR National Bit Codeword 0x9C	Sa4I	0x9B	Sa4E	------	------	
	Sa5I		Sa5E	------	------	
	Sa6I		Sa6E	------	------	
	Sa7I		Sa7E	------	------	
	Sa8I		Sa8E	------	------	
E1 Frame Pulse Interrupt 0x9F	OOOFI	0x9E	OOOFE	0x96	OOOFV	
	RAICCRCI		RAICCRCE		RAICCRCV	
	CFEBEI		CFEBEE		CFEBEV	
	V52LINKI		V52LINKE		V52LINKV	
	IFPI		IFPE	------	------	
	ICSMFPI		ICSMFPE	------	------	
	ICMFPI		ICMFPE	------	------	
	ISMFPI		ISMFPE	------	------	
SIGX Configuration Change of Signaling State 0x50	COSS [25..30]	0x50	SIGE	------	------	The COSS bit in register 0x50 must be set to logic 1 for this register to be used as shown.

Interrupt		Enable/Disable		Status		Comments
Register	"I" Bit or source group	Register	"E" Bit	Register	"V" Bit	
SIGX Microproc. Access Status/ Change of Signaling State 0x51	COSS [24..17]	0x50	SIGE	------	------	The COSS bit in register 0x50 must be set to logic 1 for this register to be used as shown.
SIGX Channel Indirect Address/Control/Change of Signalling State 0x52	COSS [16..9]	0x50	SIGE	------	------	The COSS bit in register 0x50 must be set to logic 1 for this register to be used as shown.
SIGX Channel Indirect Data Buffer/ Change of Signaling State 0x53	COSS [8..1]	0x50	SIGE	------	------	The COSS bit in register 0x50 must be set to logic 1 for this register to be used as shown.
APRM Interrupt Status 0x7A	INTR	0x78	INTE	------	------	
TJAT Interrupt Status 0x18	OVRI	0x1B	OVRE	------	------	
	UNDI		UNDE	------	------	

Interrupt		Enable/Disable		Status		Comments
Register	"I" Bit or source group	Register	"E" Bit	Register	"V" Bit	
RJAT Interrupt Status 0x14	OVRI	0x17	OVRE	------	------	
	UNDI		UNDE	------	------	
CDRC Interrupt Status 0x12	LCVI	0x11	LCVE	------	------	
	LOSI		LOSE	0x12	LOSV	
	LCSDI		LCSDE	------	------	
	ZNDI		ZNDE	------	------	
CDRC Alternate Loss of Signal 0x13	AL2LOSI	0x13	AL2LOSE	0x13	ALTLOS	
RX-ELST Interrupt Enable Status 0x1D	SLIPI	0x1D	SLIPEH	0x1D	SLIPD	
RDLC #3 Status 0xD2	OVR	0xD1	INTE	------	------	
	COLS			------	------	
	PKIN			------	------	
RDLC #2 Status 0xCA	OVR	0xC9	INTE	------	------	
	COLS			------	------	
	PKIN			------	------	
RDLC #1 Status 0xC2	OVR	0xC1	INTE	------	------	
	COLS			------	------	
	PKIN			------	------	

Interrupt		Enable/Disable		Status		Comments
Register	"I" Bit or source group	Register	"E" Bit	Register	"V" Bit	
TX-ELST Interrupt Enable Status 0x21	SLIPI	0x21	SLIPE	0x21	SLIPD	
TDPR #3 Interrupt Status UDR Clear Count 0xBC	PRINTI	0xBB	PRINTE	------	------	
	FULLI		FULLE	0xBC	FULL	
	OVRI		OVRE	------	------	
	UDRI		UDRE	------	------	
	LFILLI		LFILLE	0xBC	BLFILL	
TDPR #2 Interrupt Status UDR Clear Count 0xB4	PRINTI	0xB3	PRINTE	------	------	
	FULLI		FULLE	0xB4	FULL	
	OVRI		OVRE	------	------	
	UDRI		UDRE	------	------	
	LFILLI		LFILLE	0xB4	BLFILL	
TDPR #1 Interrupt Status UDR Clear Count 0xAC	PRINTI	0xAB	PRINTE	------	------	
	FULLI		FULLE	0xAC	FULL	
	OVRI		OVRE	------	------	
	UDRI		UDRE	------	------	
	LFILLI		LFILLE	0xAC	BLFILL	

Interrupt		Enable/Disable		Status		Comments
Register	"I" Bit or source group	Register	"E" Bit	Register	"V" Bit	
IBCD Interrupt Enable/ Status 0x4D	LBAI	0x4D	LBAE	0x4D	LBA	
	LBDI		LBDE		LBD	
T1 PDVD Interrupt Enable/ Status 0x65	Z16DI	0x65	Z16DE	------	------	
	PDVI		PDVE	0x65	PDV	
T1 RBOC Code Status 0x6B	IDLEI	0x6A	IDLI	------	------	
	BOCI		BOCE	------	------	
T1 XPDE Interrupt Enable/ Status 0x69	STUFI	0x69	STUFE	------	------	
	Z16DI		Z16DE	------	------	
	PDVI		PDVE	0X69	PDV	
T1 ALMI Interrupt Status 0x62H	YELI	0x61	YELE	0x62	YEL	
	REDI		REDE		RED	
	AISI		AISE		AIS	
E1 TRAN Interrupt Status 0x85	SIGMFI	0x84	SIGMFE	------	------	
	NFASI		NFASE	------	------	
	MFI		MFE	------	------	
	SMFI		SMFE	------	------	

Interrupt		Enable/Disable		Status		Comments
Register	"I" Bit or source group	Register	"E" Bit	Register	"V" Bit	
	FRMI		FRME	------	------	
RLPS Configuration and Status 0xF8	ALOSI	0xF8	ALOSE	0xF8	ALOSV	
BTIF Parity Configuration and Status 0x42	TDI	0x42	TPTYE	------	------	Both of these interrupt bits are enabled by TPTYE
	TSIGI					

4 PROGRAMMING THE COMET

It is convenient to consider the COMET chip as a device within a larger subsystem. Such a device could be encapsulated within an API and managed using device states, and state transitions. The three states the COMET transitions through are RESET, CONFIGURATION, and ACTIVATION. See Figure 2 for state diagram.

The RESET state identifies the device on power-up, when the registers have the default values specified in the datasheet, or after the software has reset the COMET. The CONFIGURATION state identifies the device after the software has accessed the COMET registers to specify the chip configuration. The COMET is not fully operational, or is out-of-service, during the RESET or the CONFIGURATION state. Following the CONFIGURATION state, where the software assigns an interrupt vector to handle COMET interrupts, and the software accesses COMET registers to enable the desired interrupts to activate the INTB pin, the COMET is active. In the ACTIVATION state, the COMET is fully in-service and the user may issue in-service diagnostics, process interrupt events, or poll the COMET registers to handle COMET events.

Figure 2: State Diagram of COMET

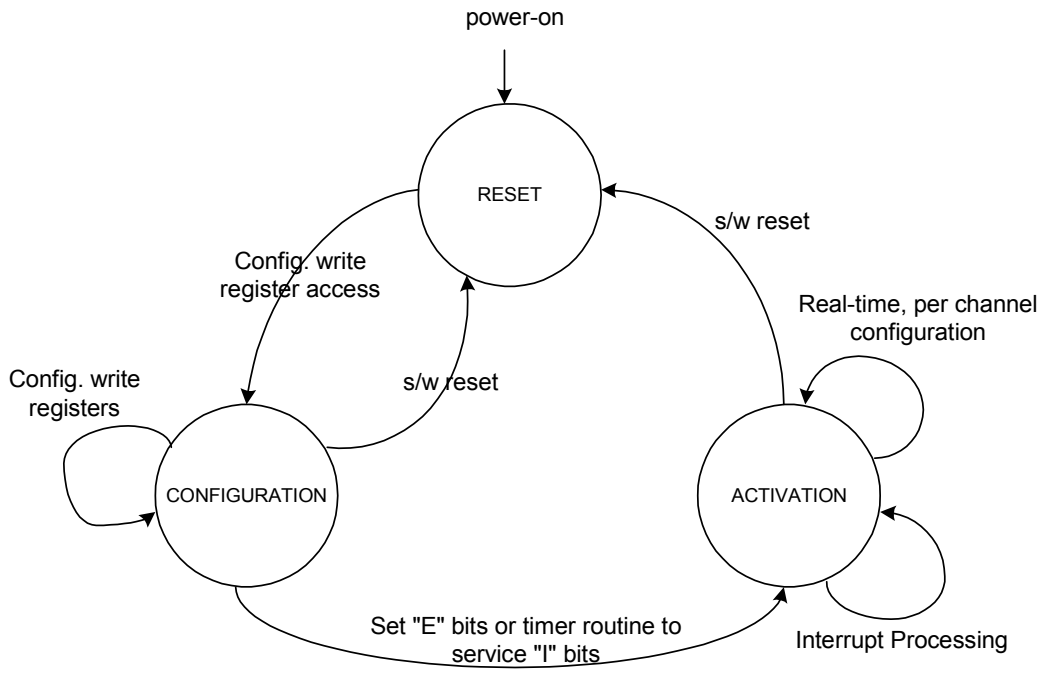


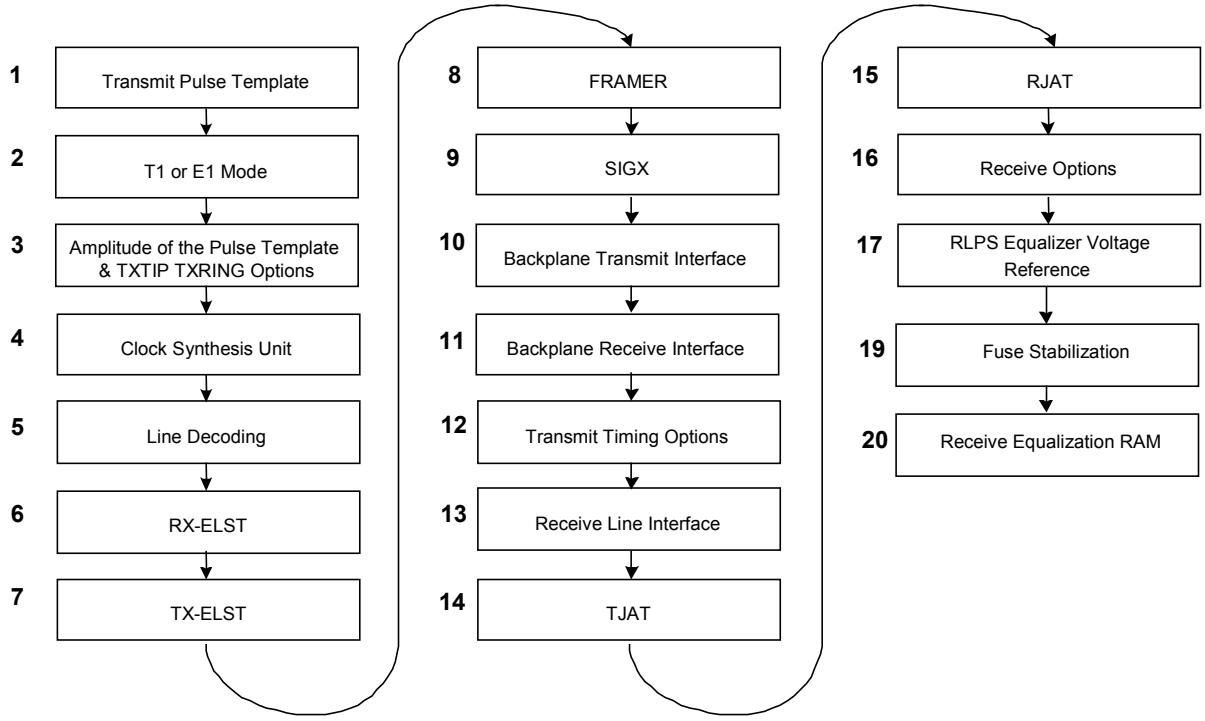
Figure 3 summarizes the configurable items necessary to transition the COMET from the RESET to the CONFIGURATION state. Table 3 identifies the interrupts that can be individually enabled via software before the ACTIVIVATION state is entered.

During the ACTIVATION state, the individual time-slots, or channels, can be reconfigured without affecting the operation of other time-slots or channels.

Note: The programmer may change some register values during the ACTIVATION state, such as enabling interrupts, but changing certain register values, such as COMET's mode of operation from T1 to E1, is unacceptable.

The subsequent sections show transition details through these states.

Figure 3: Configuration Steps



4.1 Software Reset of the COMET

The programmer may issue a software reset by writing to the register **Reset**(0x0E) with the following register access:

```
WRITE_REG(0x0E, 0x01)
```

The COMET will be held in the RESET state until the **Reset** bit is cleared with the following register access:

```
WRITE_REG(0x0E, 0x00)
```

Note: A software reset should be applied to the COMET after power-up, and whenever the configuration is changed between T1 or E1 modes of operation.

4.2 Configuring the COMET

The table below shows the chronological steps to configure the COMET. The programmer is directed to the datasheet for details on each register mentioned.

Table 4: Configuration Steps

Step	Register(s)	Description
1	See section 4.2.1	Program the Transmit Pulse Template
2	0x00	Program the COMET for T1 or E1 mode
3	0xF0	Allow transmission for the COMET by setting the HIGHZ bit to logic 0
4	0xD6	Set system clock and XCLK
5	0x10	Set line decoding
6	0x1C	Program the RX-ELST for the appropriate mode
7	0x20	Program the TX-ELST for the appropriate mode
8	For T1: 0x48	Program the framing format for the receiver
	For T1: 0x54	Set the framing format & line encoding for the transmitter
	For T1: 0x60	Program framing format and the data rate of the Facility Data Link in ESF
	For E1: 0x80	Program the transmitter framing format and line encoding
	For E1: 0x90	Configure the Receive Framer
9	0x50	Configure the SIGX Configuration Register. See "Programming the Signaling Extraction Block (SIGX)" in this document for details

Step	Register(s)	Description
10	0x40	Program the BTIF for the frame pulse mode, and when to sample the BTFP, BTPCM, and BTSIG signals. (see section 4.2.4 for fractional T1/E1 mode)
	0x41	Set the type of frame pulse on the backplane
11	0x30	Set the backplane rate and when the BRFP, BRPCM and BRSIG signals are updated. (see section 4.2.4 for fractional T1/E1 mode)
	0x31	Set frame pulse mode
	0x32	Program the data integrity checking on the BRIF
12	0x06	Select TJAT FIFO output clock signal
13	0xF8	Select long or short haul and enable the equalizer
	0xF9	Set the ALOS Detection and Clearance Thresholds
	0xFA	Set the ALOS Detection period to set the ALOS alarm
	0xFB	Set the ALOS Clearance Period to clear the ALOS alarm
	0xFC	Program to 0x00 to initiate a microprocessor access to the RAM
	0xFD	Write the value 0x80 to this register to select a write to the RAM
	0xFE	Program this register to 0x00 to reset the pointer to the RAM location to increment through the equalizer values until the signal is found.
	0xFF	Configure the Receive Line Equalizer for the correct feedback loop frequency (set the EQFBL_EN bit to logic 1).
14	0x1B	Configure the TJAT FIFO. The programmer should set the CENT bit and clear the LIMIT bit.
15	0x17	Configure the RJAT FIFO
16	0x02	Program the Receive Options

Step	Register(s)	Description
17	0xDC	Program the RLPS Equalizer Voltage reference to 0x2C for T1 or 0x34 for E1
19	See section 4.2.2	Fuse Stabilization Procedure
20	See section 4.2.3	Set up the Receive Equalization RAM

4.2.1 Using the Transmit Line Pulse Generator

Writing the value 0x00 to each sample clears the Transmit Pulse Template. The user may then program the template for the desired application. Pulse Template Table values are shown in the datasheet for the following applications:

- T1 Long Haul (LBO 0 dB)
- T1 Long Haul (LBO 7.5 dB)
- T1 Long Haul (LBO 15 dB)
- T1 Long Haul (LBO 22.5 dB)
- T1 Short Haul (0 - 110 ft.)
- T1 Short Haul (110 – 220 ft.)
- T1 Short Haul (220 – 330 ft.)
- T1 Short Haul (330 – 440 ft.)
- T1 Short Haul (440 – 550 ft.)
- T1 Short Haul (550 – 660 ft.)
- E1 120 Ohm
- E1 75 Ohm

The T1 and E1 pulse templates may be programmed via register 0xF2: **XLPG Pulse Waveform Storage Write Address** where the sample number (0..24) and the UI number (UI#0 – UI#4) are set and the data content is written or read from register 0xF3: **XLPG Pulse Waveform Storage Data**; therefore, the following instructions may be used to write to a single pulse template value:

```
WRITE_REG(0xF2, <indirect address indexed>)
WRITE_REG(0xF3, <sample value indexed>)
```

The following formula provides the indirect address based on sample number and UI:

```
Indirect_Address = (sample<<3) | UI
```

where sample = [0..23] and UI = [0..4].

The user may then use the "Transmit Waveform Table" from the COMET datasheet to determine the value to write to 0xF3(for the Sample and UI number indexed).

The programmer must specify a sample value for each sample and UI. The register access sequence for each Sample and UI is:

```
WRITE_REG(0xF2, <value from formula above for the
            indirect address for each Sample and UI>)
WRITE_REG(0xF3, <pulse template value from COMET datasheet for the
            Sample and UI>)
```

The software must repeat these two register accesses for every sample value.

For Example in the case of the T1 Long Haul (LBO 0dB) application:

Part of the Transmit Table from the COMET datasheet is shown below.

Table 5: Transmit Waveform Values for T1 Long Haul (LBO 0 dB)

Sample number	UI #0	UI #1	UI #2	UI #3	UI #4
1	00	44	00	00	00
2	0A	44	00	00	00
3	20	43	00	00	00

Sample Value

For this application, assume a two-dimensional array called "Samp_Val", stores all of the sample values (as shown in Transmit Waveform tables) when "initialize_samp_val" is called. The following piece of pseudo-code shows how the pulse template could be setup for this application:

```
#define FIRST_SAMPLE 0
#define LAST_SAMPLE 23
#define FIRST_UI 0
#define LAST_UI 4
```

```
int Samp_Val[LAST_SAMPLE-FIRST_SAMPLE+1][LAST_UI-FIRST_UI+1)] =
{<array is statically assigned from values from datasheet table>};

main() {

clear_template();
   init_template();
   /* do configurations here */
}

/* clear the pulse template */
void clear_template() {
   int i, j;
   unsigned char indirect_addr;

   for (i = FIRST_UI; i <= LAST_UI; i++) {
      for (j = FIRST_SAMPLE; j <= LAST_SAMPLE; j++) {
        indirect_addr = (j<<3)|i;
         WRITE_REG(0xF2, indirect_addr); /* set up the indirect
                                            address */
         WRITE_REG(0xF3, 0x00);
      }
}

/* initialize pulse template */
void init_template() {
   int i, j;
   unsigned int indirect_addr;

   for (i = FIRST_UI; i <= LAST_UI; i++) {
      for (j = FIRST_SAMPLE; j <= LAST_SAMPLE; j++) {
        indirect_addr = (j<<3)|i;
         WRITE_REG(0xF2, indirect_addr); /* set up the indirect
                                            address */
         WRITE_REG(0xF3, Samp_Val[j][i]);
      }
}
```

4.2.2 Fuse Stabilization

After the COMET pulse template is setup, the waveform shape may vary from the template programmed because fuses expand and collapse over time; hence, voltage levels increase and decrease. To lock the template in place, the following Fuse Stabilization Procedure (shown as pseudo-code) should be followed. This pseudo-code will setup the positive and negative fuses to a

constant state so that fuse re-growth will not vary the template through continued operation.

```
unsigned char Value;

/* write the value 0x01 to register XLPG Analog Test Positive
Control(0xF4) twice and read the result and "AND" it with 0xFE and
rewrite it to 0xF4*/
WRITE_REG(0xF4, 0x01);
WRITE_REG(0xF4, 0x01);
Value = (READ_REG(0xF4) & 0xFE);
WRITE_REG(0xF4, Value);

/* same as above except for register XLPG Analog Test Negative
Control(0xF5)*/
WRITE_REG(0xF5, 0x01);
WRITE_REG(0xF5, 0x01);
Value = (READ_REG(0xF5) & 0xFE);
WRITE_REG(0xF5, Value);

/* set up the Fuse Data Select */
WRITE_REG(0xF6, 0x01);
```

4.2.3 Using the Line Receiver

The software must assign values to RAM internal to the COMET via the normal registers. Respectively, the tables shown in the COMET datasheet under "Using the Line Interface" contain the values to be programmed into the Equalizer RAM for T1 and E1 modes. The software should program the RLPS Equalization table as the last step during the CONFIGURATION state.

The RLPS equalizer RAM content is programmed by registers 0xD8 to 0xDB **RLPS Equalization Indirect Data** for each address location. The address location is given by register 0xFC **RLPS Equalization Indirect Address**. A read or write request is done by setting the RWB bit in register 0xFD: **RLPS Equalization Read/Write Select**. NOTE: A maximum period of one line rate cycle (650ns) is required between each write access to ensure the data is correctly written.

To program the RLPS RAM content, register 0xFD must be set to 0x00 to configure a "WRITE" to the RAM. Part of the "RLPS Equalizer RAM Table" is shown in the example below. Each RAM Address has four values displayed, in each cell, under the Content column. The first four "WRITE instructions" write these values into the registers shown in the sequence below. The final "WRITE"

instruction selects the RAM address to write to. Thus initially the following instruction is used to select a "WRITE" to the RAM:

```
WRITE_REG(0xFD, 0x00) * to configure a write into the RAM address
```

The following instructions may be used to write to a single RAM address:

```
WRITE_REG(0xD8, <1st value from Content Cell>)
WRITE_REG(0xD9, <2nd value from Content Cell>)
WRITE_REG(0xDA, <3rd value from Content Cell>)
WRITE_REG(0xDB, <4th value from Content Cell>)
WRITE_REG(0xFC, <the RAM address from the table to write into>)
```

The user must repeat the above instructions for all RAM addresses, in the RLPS RAM Equalization Table, for the mode desired (T1 or E1)..

For Example:

Part of the RLPS RAM Equalization Table is shown below for T1 mode from the COMET datasheet.

RAM Address	Content (MSB..LSB)
00	0x03 0xFE 0x18 0x40
01	0x03 0xF6 0x18 0x40

Initially a "WRITE" is selected to the RAM.

```
WRITE_REG(0xFD, 0x00) *Configure a write into the RAM address
```

To write to RAM Address 00, the following instructions are used:

```
WRITE_REG(0xD8, 0x03) *1st value from Content column
WRITE_REG(0xD9, 0xFE) *2nd value from Content column
WRITE_REG(0xDA, 0x18) *3rd value from Content column
WRITE_REG(0xDB, 0x40) *4th value from Content column
WRITE_REG(0xFC, 0x00) *Initiate a write into the specified RAM
                       address
```

Similarly, to write to RAM Address 01D, the following instructions are used:

```
WRITE_REG(0xD8, 0x03) *1st value from Content column
WRITE_REG(0xD9, 0xF6) *2nd value from Content column
WRITE_REG(0xDA, 0x18) *3rd value from Content column
WRITE_REG(0xDB, 0x40) *4th value from Content column
WRITE_REG(0xFC, 0x01) *Initiate a write into the specified RAM
```

```
address
```

This must be repeated for each RAM address in the table.

4.2.4 Using the Fractional T1/E1 mode

For fractional T1/E1 mode, BRCLK and BTCLK master mode should be configured by setting the CMODE bit in registers **BRIF Configuration**(0x30) and **BTIF Configuration**(0x40) to logic 0. The NxDS0 mode selection should be determined from the datasheet for these registers. Depending on the configuration, full-frame, 56 kb/s NxDS0, 64 kb/s NxDS0, and 64 kb/s NxDS0 with F-bit (E1 mode may be selected) modes are available.

Typically the DE and FE bits in registers 0x30 and 0x40 are set to logic 0 to update the data and framing edge on the falling edge of BRCLK and BTCLK.

Note: When using the fractional T1/E1 mode, the elastic stores (RX-ELST and TX-ELST are bypassed).

For Example:

The NxDS0 64kb/s fractional T1/E1 mode may be configured to work with FREEDM. The FREEDM should connect to the COMET through the BRIF/BTIF. The registers that need to be configured for this mode are as follows:

Table 6: Register programming for use with FREEDM

Register	Value	
	T1	E1
0x30: **BRIF Configuration**	0x80	0x81
0x40: **BTIF Configuration**	0x80	0x81

4.3 Activating the COMET

For real-time activity, the COMET requires either interrupts to be enabled, or interrupt status bits to be polled. The COMET is in an ACTIVE state once the software has installed an interrupt vector, the appropriate interrupt status bits have been enabled and/or a timer initiated polling routine has been installed.

If the user intends an active interrupt status bit to drive the interrupt pin, and invoke an interrupt service routine, then the appropriate interrupt enable bits

must be set. On power-up or software reset, the interrupt status bits are cleared. The "Interrupt Reference" in section 3.5 provides a summary list of interrupt bits to find the associated enable bits.

5 USING THE PER-CHANNEL SERIAL CONTROLLERS (PCSC)

The Per-Channel Serial Controllers (TPSC and RPSC) control the per-channel functions of the Transmit and Receive PCM data.

The TPSC and RPSC configurations should be setup when the COMET is in the ACTIVATION state. The following steps should be followed in order:

1) Disable the PCCE bit and Enable the IND bit in the **TPSC (or RPSC) Configuration Register.**

2) Program the required TPSC (or RPSC) Configuration.

3) Enable the PCCE bit in the **TPSC (or RPSC) Configuration Register**.

The following sections discuss step 2.

5.1 RPSC

On power up, the per-channel control is normally disabled by default. Once the RPSC is configured, the PCCE bit in **RPSC Configuration**(0x70) must be set to activate per channel control.

The RPSC Indirect Registers (**PCM Data Control byte**(0x20-0x3F), **Data Trunk Conditioning Code byte**(0x40-0x5F), and **Signaling Trunk Conditioning byte**(0x61-0x7F)) are accessible on a per-channel basis. Table 7 provides a summary of some of the control bits (in the **RPSC Indirect Registers 20H-3FH: PCM Data Control byte**) that must be programmed to configure the channels.

Table 7: Control Bits for the PCM Data Control byte

Bit		Resulting Action
DTRKC	0	No Data Trunk Conditioning Applied
	1	BRPCM output data is replaced with content of **Data Trunk Conditioning Code Byte**(0x40-0x5F) for the channel
STRKC	0	No Signaling Trunk Conditioning Applied
	1	BRSIG output data is replaced with content of **Signaling Trunk Conditioning Byte**(0x61-0x7F) for the channel
DMW	0	No digital milliwatt pattern replaces BRPCM output data

Bit		Resulting Action
	1	BRPCM output data is replaced with a digital milliwatt pattern (based on the DMWALAW bit) for the channel
DMWALAW	0	Only applicable if DMW = 1. Digital milliwatt pattern used is the digitalA-law pattern
	1	Only applicable if DMW = 1. Digital milliwatt pattern used is the µ-law pattern
SIGNINV	0	No action
	1	Most Significant Bit of data output on BRPCM pin is inverted for the channel.

Notes:

1) RPSC indirect registers have no default values after the COMET is reset; therefore, values must be programmed in for the desired configuration.

2) If a channels DTRKC and/or STRKC bit(s) is/are set, then values must be set in the corresponding **Conditioning Code Byte**. If none are set, default values will be sent.

For further details on configurations, the programmer is directed to the section "Using the Per-Channel Serial Controllers" and the RPSC registers in the COMET datasheet.

5.1.1 RPSC Indirect Register Access

The RPSC indirect registers are for the per-channel control of the receive serial data stream. On power-up, the per-channel control is normally disabled by default. The PCCE bit in **RPSC Configuration**(0x70) must be set to enable per channel control.

The "IND" bit of the **RPSC Configuration**(0x70) must be set before the software can access RPSC indirect registers. On power-up or software reset, this bit is clear by default. Once the "IND" bit is set, the following pseudo code shows how to read a RPSC indirect register and how to write to a RPSC indirect register.

```
#define REG_RPSC_CHANNEL_INDIRECT_ADDRESS_CONTROL   0x72
#define REG_RPSC_MICRO_ACCESS_STATUS                0x71
#define REG_RPSC_CHANNEL_INDIRECT_DATA_BUFFER       0x73
#define MAX_BUSY_READ   0x5
#define BIT_BUSY        0x80
```

```
/** to read a RPSC register: **/

int ReadRpsc(int offset, unsigned char &value) {
  unsigned char temp;
  int i;
  int BusyFlag;
  // check busy bit to make sure device is ready to access
  BusyFlag = 1;
  for (i = 0; i < MAX_BUSY_READ; i++)
  {
      temp = READ_REG(REG_RPSC_MICRO_ACCESS_STATUS);
      if ((temp&BIT_BUSY) == 0)
      {
            BusyFlag = 0;
            Break;
      }
  }
  if(BusyFlag == 1)
      return FAILURE;

  WRITE_REG(REG_RPSC_CHANNEL_INDIRECT_ADDRESS_CONTROL,
                  (offset|0x80));
  for (i = 0; i < MAX_BUSY_READ; i++) {
      value = READ_REG(REG_RPSC_MICRO_ACCESS_STATUS);
      if ((value&BIT_BUSY) == 0) {
         value = READ_REG(REG_RPSC_CHANNEL_INDIRECT_DATA_BUFFER);
         return SUCCESS;
      }
  }
  return FAILURE;
}

/** to write a RPSC register: **/

int WriteRpsc(int offset, unsigned char value) {
  unsigned char temp;
  int i;
  int BusyFlag;
  // check busy bit to make sure device is ready to access
  BusyFlag = 1;
  for (i = 0; i < MAX_BUSY_READ; i++)
  {
      temp = READ_REG(REG_RPSC_MICRO_ACCESS_STATUS);
      if ((temp&BIT_BUSY) == 0)
      {
```

```
                BusyFlag = 0;
                Break;
            }
        }
        if(BusyFlag == 1)
            return FAILURE;

        WRITE_REG(REG_RPSC_CHANNEL_INDIRECT_DATA_BUFFER, value);
        WRITE_REG(REG_RPSC_CHANNEL_INDIRECT_ADDRESS_CONTROL, (offset&0x7F));
        for (i = 0; i < MAX_BUSY_READ; i++) {
            temp = READ_REG(REG_RPSC_MICRO_ACCESS_STATUS);
            if ((temp&BIT_BUSY) == 0) return SUCCESS;
        }
        return FAILURE;
    }
```

5.2 TPSC

The TPSC indirect registers provide per-channel serial control over the transmit data stream. The per-channel control is normally disabled by default, on power-up. The PCCE bit in **TPSC Configuration**(0x6C) must be set to activate per channel control once the RPSC is configured.

On power up, the per-channel control is normally disabled by default. Once the TPSC is configured, the PCCE bit in **TPSC Configuration**(0x6C) must be set to activate per channel control. See section 5.2.1 for accessing the TPSC indirect registers.

The TPSC Indirect Registers (**PCM Data Control byte**(0x20-0x3F), **IDLE Code byte**(0x40-0x5F) and **Signaling/E1 Control byte**(0x60-0x7F)) are accessible on a per-channel basis.

The **PCM Data Control byte** provides control over how to condition the data received on the BTPCM pin. The **IDLE Code byte** stores a constant value that the programmer can transmit for any channel. The **Signaling/E1 Control byte** provides data manipulation in E1 mode and signaling insertion in T1 mode.

The programmer is directed to the section “Using the Per-Channel Serial Controllers” and the TPSC registers in the COMET datasheet for the programming sequence for TPSC as well as to the section “Using the Loopback Modes” for details on configurations.

5.2.1 TPSC Indirect Register Access

The TPSC indirect registers are for the per-channel control of the transmit serial data stream. On power-up, the per-channel control is normally disabled by default; therefore, the PCCE bit in **TPSC Configuration**(0x6C) must be set to enable per channel control once the TPSC is configured.

The "IND" bit of the **TPSC Configuration**(0x6C) must be set before the software can access TPSC indirect registers. On power-up or software reset, this bit is clear by default. Once the "IND" bit is set, the following pseudo code shows how to read a TPSC indirect register and write to a TPSC indirect register.

```
#define REG_TPSC_CHANNEL_INDIRECT_ADDRESS_CONTROL   0x6E
#define REG_TPSC_MICRO_ACCESS_STATUS                0x6D
#define REG_TPSC_CHANNEL_INDIRECT_DATA_BUFFER       0x6F
#define MAX_BUSY_READ   0x5
#define BIT_BUSY        0x80

/** to read a TPSC register: **/

int ReadTpsc(int offset, unsigned char &value) {
  unsigned char temp;
  int i;
  int BusyFlag;
  // check busy bit to make sure device is ready to access
  BusyFlag = 1;
  for (i = 0; i < MAX_BUSY_READ; i++)
  {
      temp = READ_REG(REG_TPSC_MICRO_ACCESS_STATUS);
      if ((temp&BIT_BUSY) == 0)
      {
            BusyFlag = 0;
            Break;
      }
  }
  if(BusyFlag == 1)
      return FAILURE;

  WRITE_REG(REG_TPSC_CHANNEL_INDIRECT_ADDRESS_CONTROL,
                  (offset|0x80));
  for (i = 0; i < MAX_BUSY_READ; i++) {
      value = READ_REG(REG_TPSC_MICRO_ACCESS_STATUS);
      if ((value&BIT_BUSY) == 0) {
         value = READ_REG(REG_TPSC_CHANNEL_INDIRECT_DATA_BUFFER);
```

```
            return SUCCESS;
        }
    }
    return FAILURE;
}

/** to write a TPSC register: **/

int WriteTpsc(int offset, unsigned char value) {
    unsigned char temp;
    int i;
    int BusyFlag;
    // check busy bit to make sure device is ready to access
    BusyFlag = 1;
    for (i = 0; i < MAX_BUSY_READ; i++)
    {
        temp = READ_REG(REG_TPSC_MICRO_ACCESS_STATUS);
        if ((temp&BIT_BUSY) == 0)
        {
              BusyFlag = 0;
              Break;
        }
    }
    if(BusyFlag == 1)
        return FAILURE;

    WRITE_REG(REG_TPSC_CHANNEL_INDIRECT_DATA_BUFFER, value);
    WRITE_REG(REG_TPSC_CHANNEL_INDIRECT_ADDRESS_CONTROL, (offset&0x7F));
    for (i = 0; i < MAX_BUSY_READ; i++) {
        temp = READ_REG(REG_TPSC_MICRO_ACCESS_STATUS);
        if ((temp&BIT_BUSY) == 0) return SUCCESS;
    }
    return FAILURE;
}
```

6 USING THE SIGNALING EXTRACTION BLOCK (SIGX)

The Signaling Extraction (SIGX) block provides channel associated signaling (CAS) extraction from an E1 signaling multiframe or from SF, SLC®96, and ESF T1 Formats. The SIGX block extracts the signaling bits from the received datastream and serializes the results on the BRSIG output pin.

Control of CAS is normally disabled by default, on power-up or reset. The PCCE bit in **SIGX Configuration Register**(0x50) must be set to activate per channel control. The SIGX configuration should be setup when the COMET is in the ACTIVATION state. The following steps should be followed in order:

1) Disable the PCCE and enable the IND and COSS bit (if indirect registers need to be accessed) in the **SIGX Configuration Register**.

2) Program the required SIGX Configuration.

3) Enable the PCCE bit in the **SIGX Configuration Register**.

The **SIGX Configuration Register**(0x50) can represent two registers based on how the COSS register bit is set. This is discussed below.

COSS = 0

The SIGX indirect registers are disabled on reset and have no default value when enabled. Setting the IND bit to logic 1 and the COSS bit to logic 0 enables access to the indirect registers in the **SIGX Configuration Register**(0x50). When the COSS bit is set to logic 0, the SIGX register space is configured to allow indirect access to each of the 24 T1 or 30 E1 channels.

The SIGX Indirect Registers (**Current Timeslot/channel Signaling Data(0x10 – 0x1f), Delayed Timeslot/Channel Signaling Data** (0x20-0x3F) and **Per-Timeslot Configuration** (0x40-0x5F)) are accessible on a per-channel basis.

Timeslot/Channel Signaling Data registers store - signaling data to be sent to the BRSIG pin. These registers contain the A,B,C,D bits for the ESF, SF, SLC®96 and signaling multiframe format (where bits C and D are replaced with A and B for SF and SLC®96). The **Per-Timeslot Configuration** registers provide data conditioning in conjunction with the RPSC Data Control bytes.

COSS = 1

When the COSS bit is set to 1, the **SIGX Change of Signaling State Registers** (0x50-0x53) are used to indicate a change of signaling state on a channel. In T1

mode, COSS bits 1 to 24 represent each of the 24 channels. In E1 mode, COSS bits 1 to 30 represent each of the 30 timeslots. These registers may be polled to determine the channel that had a signaling state change. The COSS bits are cleared when the register is read.

6.1.1 SIGX Indirect Register Access

The SIGX indirect registers are for the channel associated signaling from framing formats. On power-up, the per-channel control is normally disabled by default. The PCCE bit in **SIGX Configuration Register (COSS=0)**(0x50) must be set to enable per channel control.

The "IND" bit of the **SIGX Configuration Register (COSS=0)**(0x50) must be set to logic 1 and the COSS bit must be set to logic 0 before the software can access SIGX indirect registers. By default, on power-up or software reset, these bits are clear. Once the "IND" bit is set, the following pseudo code shows how to read a SIGX indirect register and write to a SIGX indirect register.

```
#define REG_SIGX_CHANNEL_INDIRECT_ADDRESS_CONTROL  0x52
#define REG_SIGX_MICRO_ACCESS_STATUS               0x51
#define REG_SIGX_CHANNEL_INDIRECT_DATA_BUFFER      0x53
#define MAX_BUSY_READ   0x5
#define BIT_BUSY        0x80

/** to read a SIGX register: **/

int ReadSigx(int offset, unsigned char &value) {
  unsigned char temp;
  int i;
  int BusyFlag;
  // check busy bit to make sure device is ready to access
  BusyFlag = 1;
  for (i = 0; i < MAX_BUSY_READ; i++)
  {
      temp = READ_REG(REG_SIGX_MICRO_ACCESS_STATUS);
      if ((temp&BIT_BUSY) == 0)
      {
            BusyFlag = 0;
            Break;
      }
  }
  if(BusyFlag == 1)
      return FAILURE;
```

```
  WRITE_REG(REG_SIGX_CHANNEL_INDIRECT_ADDRESS_CONTROL,
                  (offset|0x80));
  for (i = 0; i < MAX_BUSY_READ; i++) {
      value = READ_REG(REG_SIGX_MICRO_ACCESS_STATUS);
      if ((value&BIT_BUSY) == 0) {
         value = READ_REG(REG_SIGX_CHANNEL_INDIRECT_DATA_BUFFER);
         return SUCCESS;
      }
  }
  return FAILURE;
}

/** to write a SIGX register: **/

int WriteSigx(int offset, unsigned char value) {
  unsigned char temp;
  int i;
  int BusyFlag;
  // check busy bit to make sure device is ready to access
  BusyFlag = 1;
  for (i = 0; i < MAX_BUSY_READ; i++)
  {
      temp = READ_REG(REG_SIGX_MICRO_ACCESS_STATUS);
      if ((temp&BIT_BUSY) == 0)
      {
            BusyFlag = 0;
            Break;
      }
  }
  if(BusyFlag == 1)
      return FAILURE;
  WRITE_REG(REG_SIGX_CHANNEL_INDIRECT_DATA_BUFFER, value);
  WRITE_REG(REG_SIGX_CHANNEL_INDIRECT_ADDRESS_CONTROL, (offset&0x7F));
  for (i = 0; i < MAX_BUSY_READ; i++) {
      temp = READ_REG(REG_SIGX_MICRO_ACCESS_STATUS);
      if ((temp&BIT_BUSY) == 0) return SUCCESS;
  }
  return FAILURE;
}
```

7 HDLC PROGRAMMING

The COMET provides three HDLC controllers, each with 128-byte transmit and receive buffers.

7.1 Using the Internal HDLC Transmitters

The access rate to the TDPR registers is limited by the transmit clock (TCLK) rate; therefore, it is important that they are accessed at a rate no faster than the TCLK.

To properly initialize the transmit HDLC controllers in basic frame alignment mode (FPTYP is logic 0), multiframe alignment (FPTYP is logic 1) must be configured for at least one multiframe (i.e., for at least one multiframe period in frame pulse master mode or for at least one input frame pulse in frame pulse slave mode). After this initialization, the FPTYP can be set to any desired value.

Upon reset, setting the EN bit in the **TDPR Configuration**(0xA8, 0xB0, or 0xB8) register to logic 0 (default value) disables the TDPR. After making all initial configurations to the TDPR, the EN bit should be set to logic 1 to enable the TDPR. Then the FIFOCLR bit should be set and then cleared to initialize the TDPR FIFO. The TDPR is now ready to transmit.

To initialize the TDPR, the **TDPR Configuration** Register must be properly set. If FCS generation is desired, the CRC bit should be set to logic 1. If the block is to be used in interrupt driven mode, then interrupts should be enabled by setting the FULLE, OVRE, UDRE, and LFILLE bits in the **TDPR Interrupt Enable**(0xA8, 0xB3, or 0xBB) register to logic 1. The TDPR operating parameters in the **TDPR Upper Transmit Threshold** and **TDPR Lower Interrupt Threshold** registers should be set to the desired values. The **TDPR Upper Transmit Threshold** sets the value at which the TDPR automatically begins the transmission of HDLC packets, even if no complete packets are in the FIFO. Transmission will continue until the current packet is transmitted and the number of bytes in the TDPR FIFO falls to, or below, this threshold level. The TDPR will always transmit all complete HDLC packets (packets with EOM attached) in its FIFO. Finally, setting the EN bit to logic 1 enables the TDPR. If no message is sent after the EN bit is set to logic 1, continuous flags will be sent.

The TDPR can be used in a polled or interrupt driven mode for the transfer of packet data. In the polled mode, the processor controlling the TDPR must periodically read the **TDPR Interrupt Status** register to determine when to write to the **TDPR Transmit Data** register. In the interrupt driven mode, the processor

controlling the TDPR uses the INT output to identify the interrupts which determine when writes can or must be done to the **TDPR Transmit Data** register.

7.1.1 Automatic transmission mode using interrupts:

The TDPR automatically transmits a packet once it is completely written into the TDPR FIFO. The TDPR also begins transmission of bytes once the FIFO level exceeds the programmable Upper Transmit Threshold. The CRC bit can be set to logic 1 so that the FCS is generated and inserted at the end of a packet. The TDPR Lower Interrupt Threshold should be set to such a value that sufficient warning of an underrun is given. The FULLE, LFILLE, OVRE, and UDRE bits are all set to logic 1 so an interrupt on INT is generated upon detection of a FIFO full state, a FIFO depth below the lower limit threshold, a FIFO overrun, or a FIFO underrun. The following procedure should be followed to transmit HDLC packets:

1. Wait for data to be transmitted. Once data is available to be transmitted, go to step 2.

2. Write the data byte to the **TDPR Transmit Data** register.

3. If all bytes in the packet have been sent, set the EOM bit in the **TDPR Configuration** register to logic 1. Proceed to step 1.

4. If there are more bytes in the packet to be sent, go to move on 2.

While performing steps 1 to 4, the processor should monitor for interrupts generated by the TDPR. When an interrupt is detected, the TDPR Interrupt Routine should be executed.

The TDPR will force transmission of the packet information when the FIFO depth exceeds the threshold programmed with the UTHR[6:0] bits in the **TDPR Upper Transmit Threshold** register. Transmission will not stop until the last byte of all complete packets is transmitted and the FIFO depth is at or below the threshold limit. The user should watch the FULLI and LFILLI interrupts to prevent overruns and underruns.

7.1.2 TDPR Interrupt Routine:

The following procedure should be carried out when an interrupt is detected on INT:

1. Read the **TDPR Interrupt Status** register.

2. If UDRI=1, then the FIFO has underrun and the last packet transmitted has been corrupted and needs to be retransmitted. When the UDRI bit transitions to logic 1, one Abort sequence and continuous flags will be transmitted. The TDPR FIFO is held in reset state. To re-enable the TDPR FIFO and to clear the underrun, the **TDPR Interrupt Status**/UDR Clear register should be written with any value.

3. If OVRI=1, then the FIFO has overflowed. The packet, which the last byte written into the FIFO belongs to has been corrupted and must be retransmitted. Other packets in the FIFO are not affected. When an overflow occurs, the OVR output signal is set. To indicate that the FIFO depth is at the lower threshold limit, either a timer can be used to determine when sufficient bytes are available in the FIFO or the user can wait until the LFILLI interrupt is set. The OVR output signal remains set until the next write to the **TDPR Transmit Data** register. This write contains the first byte of the next packet to be transmitted.

   If the FIFO overflows on the packet currently being transmitted (packet is greater than 128 bytes long), the OVR output signal is set, an Abort signal is scheduled to be transmitted, the FIFO is emptied, and then flags are continuously sent until there is data to be transmitted. The FIFO is held in reset until a write to the **TDPR Transmit Data** register occurs. This write contains the first byte of the next packet to be transmitted.

4. If FULLI=1 and FULL=1, then the TDPR FIFO is full and no further bytes can be written. When in this state, either a timer can be used to determine when sufficient bytes are available in the FIFO or the user can wait until the LFILLI interrupt is set, indicating that the FIFO depth is at the lower threshold limit.

   If FULLI=1 and FULL=0, then the TDPR FIFO had reached the FULL state earlier, but has since emptied out some of its data bytes and now has space available in its FIFO for more data.

5. If LFILLI=1 and BLFILL=1, then the TDPR FIFO depth is below its lower threshold limit. If there is more data to transmit, then it should be written to the **TDPR Transmit Data** register before an underrun occurs. If there is no more data to transmit, then an EOM should be set at the end of the last packet byte. Flags will then be transmitted once the last packet has been transmitted.

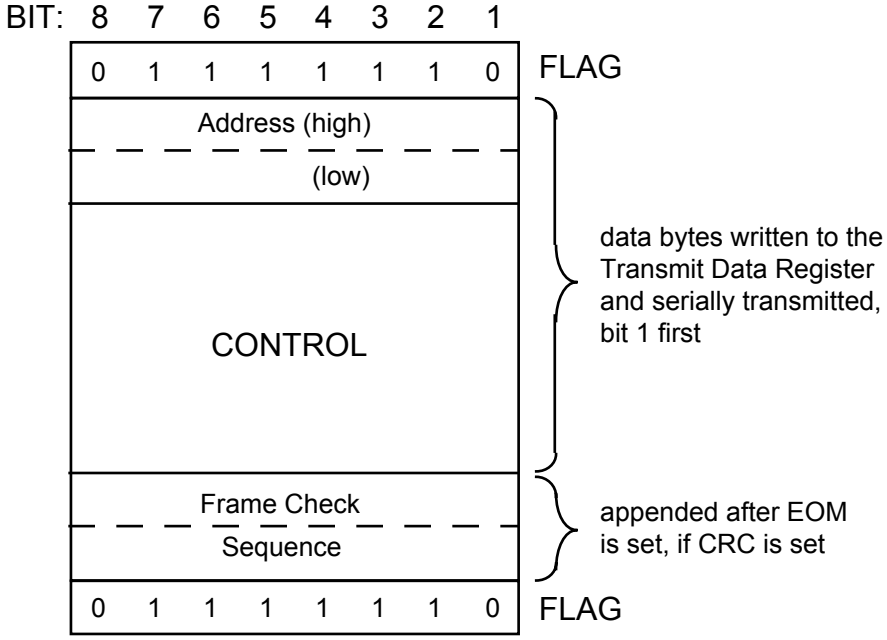
   If LFILLI=1 and BLFILL=0, then the TDPR FIFO had fallen below the lower-threshold state earlier, but has since been refilled to a level above the lower-threshold level.

7.1.3 Automatic transmission mode using polling:

The TDPR automatically transmits a packet once it is completely written into the TDPR FIFO. The TDPR also begins transmission of bytes once the FIFO level exceeds the programmable Upper Transmit Threshold. The CRC bit can be set to logic 1 so that the FCS is generated and inserted at the end of a packet. The **TDPR Lower Interrupt Threshold** should be set to such a value that sufficient warning of an underrun is given. The FULLE, LFILLE, OVRE, and UDRE bits are all set to logic 0 since packet transmission is set to work with a periodic polling procedure. The following procedure should be followed to transmit HDLC packets:

1. Wait until data is available to be transmitted, then go to step 2.
2. Read the **TDPR Interrupt Status** register.
3. If FULL=1, the TDPR FIFO is full and no further bytes can be written. Continue polling the **TDPR Interrupt Status** register until either FULL=0 or BLFILL=1. Then, proceed to either step 4 or 5 depending on implementation preference.
4. If BLFILL=1, the TDPR FIFO depth is below its lower threshold limit. Write the data into the **TDPR Transmit Data** register. Advance to step 6.
5. If FULL=0, the TDPR FIFO has room for at least 1 more byte to be written. Write the data into the **TDPR Transmit Data** register. Move on to step 6.
6. If more data bytes are to be transmitted in the packet, proceed to step 2.
7. If all bytes in the packet have been sent, set the EOM bit in the **TDPR Configuration** register to logic 1. Go to step 1.

Figure 4: Typical Data Frame



7.1.4 Effect of XBOC on HDLC Packets Being Transmitted

The **T1 XBOC Code**(0x67) register enables the XBOC to generate a bit oriented code and selects the 8-bit code to be transmitted. If the **T1 XBOC Code**(0x67) register is written with any 6-bit code other than 111111B, that code will be transmitted repeatedly in the ESF Facility Data Link overwriting any HDLC packets currently being transmitted. The XBOC is disabled when **T1 XBOC Code**(0x67) register is written with the 6-bit code 111111B.

7.2 Using the Internal HDLC Receivers

On power up of the system, setting the EN bit in the Configuration Register to logic 0 disables the RDLC. The Interrupt Control Register should then be initialized to enable the INTB output and to select the FIFO buffer fill level at which an interrupt will be generated. If the INTE bit is not set to logic 1, the Status Register must be continuously polled to check the interrupt status (INTR) bit.

After the Interrupt Control Register has been written, the RDLC can be enabled at any time by setting the EN bit in the Configuration Register to logic 1. When the RDLC is enabled, it will assume the link status is idle (all ones) and immediately begin searching for flags. When the first flag is found, an interrupt

will be generated, and a dummy byte will be written into the FIFO buffer providing alignment of link up status with the data read from the FIFO. When an abort character is received, another dummy byte and link down status is written into the FIFO providing alignment of link down status with the data read from the FIFO. The controlling processor is responsible for checking the COLS Status Register bit for a change in the link status. If the COLS Status Register bit is set to logic 1, the FIFO must be emptied to determine the current link status. The first flag and abort status, encoded in the PBS bits, are used to set and clear a Link Active software flag.

When the last byte of a properly terminated packet is received, an interrupt is generated. When the Status Register is read, the PKIN bit will be logic 1, which can either be a signal to the external processor to empty the bytes remaining in the FIFO or an increment of a number-of-packets-received count. Wait for the FIFO to fill to a programmable level. Once the Status Register is read, the PKIN bit is cleared to logic 0. If the Status Register is read immediately after the last packet byte is read from the FIFO, the PBS[2] bit will be logic 1 and the CRC and non-integer byte status can be checked by reading the PBS[1:0] bits.

When the FIFO fill level is exceeded, an interrupt is generated. The FIFO must be emptied to remove this source of interrupt.

The RDLC can be used in polled, interrupt driven or DMA-controlled mode for the transfer of frame data. In the polled mode, the INTB output is not used, and the processor controlling the RDLC must periodically read the Status Register of the RDLC to determine when to read the Data Register. In the interrupt driven mode, the processor controlling the RDLC uses the INTB output to determine when to read the Data Register.

In the case of interrupt driven data transfer from the RDLC to the processor, the INTB output of the RDLC is connected to the interrupt input of the processor. Once the processor has determined that the RDLC is the source of the interrupt, the interrupt service routine should process the data in the following order:

1. **RDLC Status Register** Read. If INTR=1 then proceed to step 2 otherwise find the interrupt source elsewhere.

2. If OVR = 1, discard last frame and go to step 1. Overrun causes a reset of FIFO pointers. Any packets that may have been in the FIFO are lost.

3. If COLS = 1, set the EMPTY FIFO software flag.

4. If PKIN = 1, increment the PACKET COUNT. If the FIFO is desired to be emptied as soon as a complete packet is received, set the EMPTY FIFO

software flag. If the EMPTY FIFO software flag is not set, FIFO emptying will be delayed until the FIFO fill level is exceeded.

5. Data Register Read.

6. Status Register Read.

7. If OVR = 1, discard last frame and go to step 1. Overrun causes a reset of FIFO pointers. Any packets that may have been in the FIFO are lost.

8. If COLS = 1, set the EMPTY FIFO software flag.

9. If PKIN = 1, increment the PACKET COUNT. If the FIFO is desired to be emptied as soon as a complete packet is received, set the EMPTY FIFO software flag. If the EMPTY FIFO software flag is not set, FIFO emptying will be delayed until the FIFO fill level is exceeded.

10. Start the processing of FIFO data. Use the PBS[2:0] packet byte status bits to decide what is to be done with the FIFO data.

    a) If PBS[2:0] = 001, discard data byte read in step 5 and set the LINK ACTIVE software flag.

    b) If PBS[2:0] = 010, discard the data byte read in step 5 and clear the LINK ACTIVE software flag.

    c) If PBS[2:0] = 1XX, store the last byte of the packet, decrement the PACKET COUNT, and check the PBS[1:0] bits for CRC or NVB errors before deciding whether or not to keep the packet.

    d) If PBS[2:0] = 000, store the packet data.

11. If FE = 0 and INTR = 1 or FE = 0 and EMPTY FIFO = 1, go to step 5 else clear the EMPTY FIFO software flag and leave this interrupt service routine to wait for the next interrupt.

The Link State is typically a local software variable. The Link State is inactive if the RDLC is receiving all ones or receiving bit-oriented codes which contain a sequence of eight ones. The Link State is active if the RDLC is receiving flags or data.

If the RDLC data transfer is operating in the polled mode, processor operation is exactly as shown above for the interrupt driven mode, except that the entry to the service routine is from a timer, rather than an interrupt.

8 COMET PROGRAMMING EXAMPLES

The following sections are provided as a quick reference for the register write sequence for commonly used configurations. The sequences do not show interrupt enable, per channel control and HDLC sequences. The reader should refer to other sections in this document for in depth discussion of these topics.

8.1 T1 Mode

Table 8 in this section shows how to configure the COMET for T1 Long Haul (0dB), B8ZS line decoding and T1-ESF mode. It is assumed the programmer will program the transmit table and receive equalization RAM table as described in sections 4.2.1 and 4.2.2.

Table 8: Using B8ZS line encoding/decoding and ESF mode

Programming COMET to use B8ZS line encoding/decoding and put into T1-ESF mode.			
#	**Register**	**Value**	**Comments**
1	0xF0	0x0C	Set SCALE[4:0]=0x0C for 0dB Line Build Out and enable XLPG by setting HIGHZ bit to logic 0
2	0xD6	0x07	Set XCLK = 2.048MHz and TCLKO = 1.544MHz
3	0x10	0x00	Set B8ZS Line Decoding
4	0x1C	0x00	Enable RX-ELST for T1 mode
5	0x20	0x00	Enable TX-ELST for T1 mode
6	0x54	0x30	Transmit T1-ESF and set B8ZS Line Encoding
7	0x48	0x30	Receive T1-ESF framing format
8	0x60	0x10	Configure framing format for facility data link
9	0x50	0x04	Set SIGX block for ESF framing mode
10	0x40	0x38	Full Framed, BTCLK = clock slave mode
11	0x41	0x01	BTFP in frame pulse slave mode
12	0x30	0x00	BRCLK=Clock Master mode, Full Framed
13	0x31	0x00	Put BRFP in frame pulse master mode
14	0x32	0x01	Enable backplane data and signaling

Programming COMET to use B8ZS line encoding/decoding and put into T1-ESF mode.			
#	**Register**	**Value**	**Comments**
15	0x06	0x01	Bypass TX-ELST and set TJAT to use BTCLK as reference clock
17	0xF9	0x00	ALOS Clearance (to make the ALOS quasi-shorthaul detectable). The value may be programmed to 0xAA in accordance with I.431.
18	0xFA	0x01	16 pulse intervals
19	0xFB	0x01	16 pulse intervals
22	0xFE	0x00	Location = 0
23	0xFF	0x0B	Configure the Receive Line by setting the reserved bits appropriately
24	0x1B	0x10	Set TJAT FIFO up
25	0x17	0x10	Set RJAT FIFO up
26	0x02	0x00	Enable Jitter attenuator on receive side
29	0xDC	0x2C	Set RLPS Equalizer Voltage Reference

Table 9 shows changes that are required from Table 8 to put COMET into SF mode. The "#" column shows the table row from table 6 that should be replaced by the corresponding table row in Table 9.

Table 9: Programming the COMET for SF format

Programming COMET to use the SF format in T1 mode.			
#	**Register**	**Value**	**Comments**
6	0x54	0x20	Transmit T1-SF and set B8ZS Line Encoding
7	0x48	0x20	Receive T1-SF framing format
8	0x60	0x00	Configure framing format for facility data link
9	0x50	0x00	Set SIGX block for SF framing mode

Table 10 shows changes that are required from Table 8 to allow COMET to use AMI line encoding/decoding. The "#" column shows the table row that should be replaced from Table 8 by the corresponding table row in Table 10.

Table 10: AMI line encoding/decoding

Programming COMET to use AMI.			
#	**Register**	**Value**	**Comments**
3	0x10	0x80	Set AMI Line Decoding
6	0x54	0x10	Transmit T1-ESF and disable B8ZS Line Encoding

8.2 E1 Mode

Table 11 shows how to configure the COMET for E1 120 ohms, HDB3 line decoding and E1-CRC Multiframe mode. It is assumed the programmer will program the transmit table and receive equalization RAM table as described in the datasheet.

Table 11: HDB3 line encoding/decoding & E1-CRC Multiframe

Programming COMET to use HDB3 line encoding/decoding and put into E1-CRC Multiframe mode.			
#	**Register**	**Value**	**Comments**
1	0x00	0x81	Put COMET into E1 mode

Programming COMET to use HDB3 line encoding/decoding and put into E1-CRC Multiframe mode.			
#	**Register**	**Value**	**Comments**
2	0xF0	0x0C	Set SCALE[4:0]=0x0C for 0dB Line Build Out and enable XLPG by setting HIGHZ bit to logic 0
3	0xD6	0x00	Set XCLK = 2.048MHz and TCLKO = 2.048MHz
4	0x10	0x00	Set HDB3 Line Decoding
5	0x1C	0x03	Enable RX-ELST for E1 mode
6	0x20	0x03	Enable TX-ELST for E1 mode
7	0x80	0x10	E1 CRC Multiframe and HDB3 line encoding
8	0x90	0xC2	Receive E1-CRC Multiframe format
9	0x50	0x00	Disable Per-Timeslot/Per-Channel Configuration
10	0x40	0x39	Full Framed, BTCLK = clock slave mode. Set Rate
11	0x41	0x01	BTFP in frame pulse slave mode
12	0x30	0x01	BRCLK=Clock Master mode, Full Framed
13	0x31	0x00	Put BRFP in frame pulse master mode
14	0x32	0x01	Enable backplane data and signaling
15	0x06	0x01	Bypass TX-ELST and set TJAT to use BTCLK as reference clock
17	0xF9	0x00	ALOS Clearance (to make the ALOS quasi-shorthaul detectable). The value may be programmed to 0x55 in accordance with I.431.
18	0xFA	0x01	16 pulse intervals
19	0xFB	0x01	16 pulse intervals
22	0xFE	0x00	Location = 0
23	0xFF	0x0B	Configure the Receive Line by setting the reserved bits appropriately
	0x19	0xFF	set TJAT jitter attenuation

Programming COMET to use HDB3 line encoding/decoding and put into E1-CRC Multiframe mode.			
#	**Register**	**Value**	**Comments**
	0x1A	0xFF	set TJAT jitter attenuation
24	0x1B	0x10	Set TJAT FIFO up
	0x15	0xFF	set RJAT jitter attenuation
	0x16	0xFF	set RJAT jitter attenuation
25	0x17	0x10	Set RJAT FIFO up
26	0x02	0x00	Enable Jitter attenuator on receive side
29	0xDC	0x34	Set RLPS Equalizer Voltage Reference

Table 12 shows what needs to change from the above table to allow the COMET to use AMI rather than HDB3.

Table 12: Using AMI line encoding/decoding

Programming COMET to use AMI line encoding/decoding			
#	**Register**	**Value**	**Comments**
4	0x10	0x80	Set AMI Line Decoding
7	0x80	0x90	E1 CRC Multiframe and AMI line encoding

9 ALARMS

The COMET can detect many alarm conditions and can transmit certain alarms based on the mode of operation. The following sections show the interrupt/alarm hierarchy, as well as how to insert alarms and the types of alarms the COMET may indicate based on T1 or E1 mode.

9.1 Interrupt/Alarm Hierarchy

The receiver side of COMET has an interrupt/alarm hierarchy shown in Figure 6 for T1 and E1 modes. The priority shown is based on order of importance for correcting alarm conditions. Interrupts/alarms shown in dotted boxes represent the same priority for the enclosed blocks. The highest priority interrupts/alarms are shown at the top and decrease in priority as the user goes down. Interrupts and alarms may propagate down so the higher level ones should be processed before processing any below. If an interrupt/alarm condition arises in the COMET receiver side, the hierarchy should generally be followed. The diagram shows blocks for all of the interrupt source register bits. Once the highest level block is found (that is active), the diagram shows arrows to the interrupt bits that caused the interrupt/alarm to occur. The user should look in the COMET datasheet for details about the particular interrupt.

Figure 6 generally shows the order to follow. In some cases the interrupt/alarm should be ignored. See Table 13 for transmit cases and see Table 14 for receiver cases.

Table 13: Cases to ignore Transmitter interrupts / alarms

TSB Group of "I bits" to ignore	Reason
TRAN	COMET mode is T1
APRM	COMET mode is E1
TX-ELST	Datacom applications

Table 14: Cases to ignore Receiver interrupt / alarms

TSB Group of "I bits" to ignore	Reason
CDRC	COMET is running in digital only mode
RLPS	COMET is running in long haul only mode

TSB Group of "I bits" to ignore	Reason
RX-ELST	Datacom applications
SIGX	Datacom applications

The transmit side of COMET does not have an interrupt/alarm hierarchy. Checking each of the blocks can identify the cause of an interrupt/alarm. Figure 5 and Figure 6 shows the interrupt bits for each interrupt/alarm case.

The next two sections show the bits that must be set to insert alarms into the transmit stream and the backplane and also show why an alarm may occur.

Figure 5: Transmitter Alarms and Interrupts

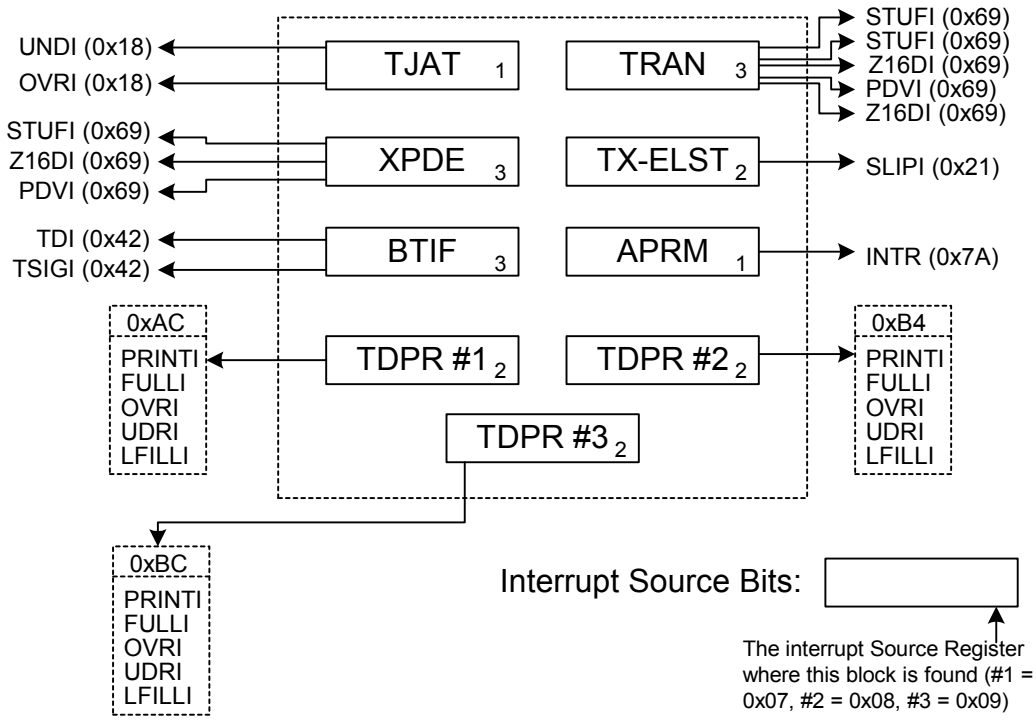
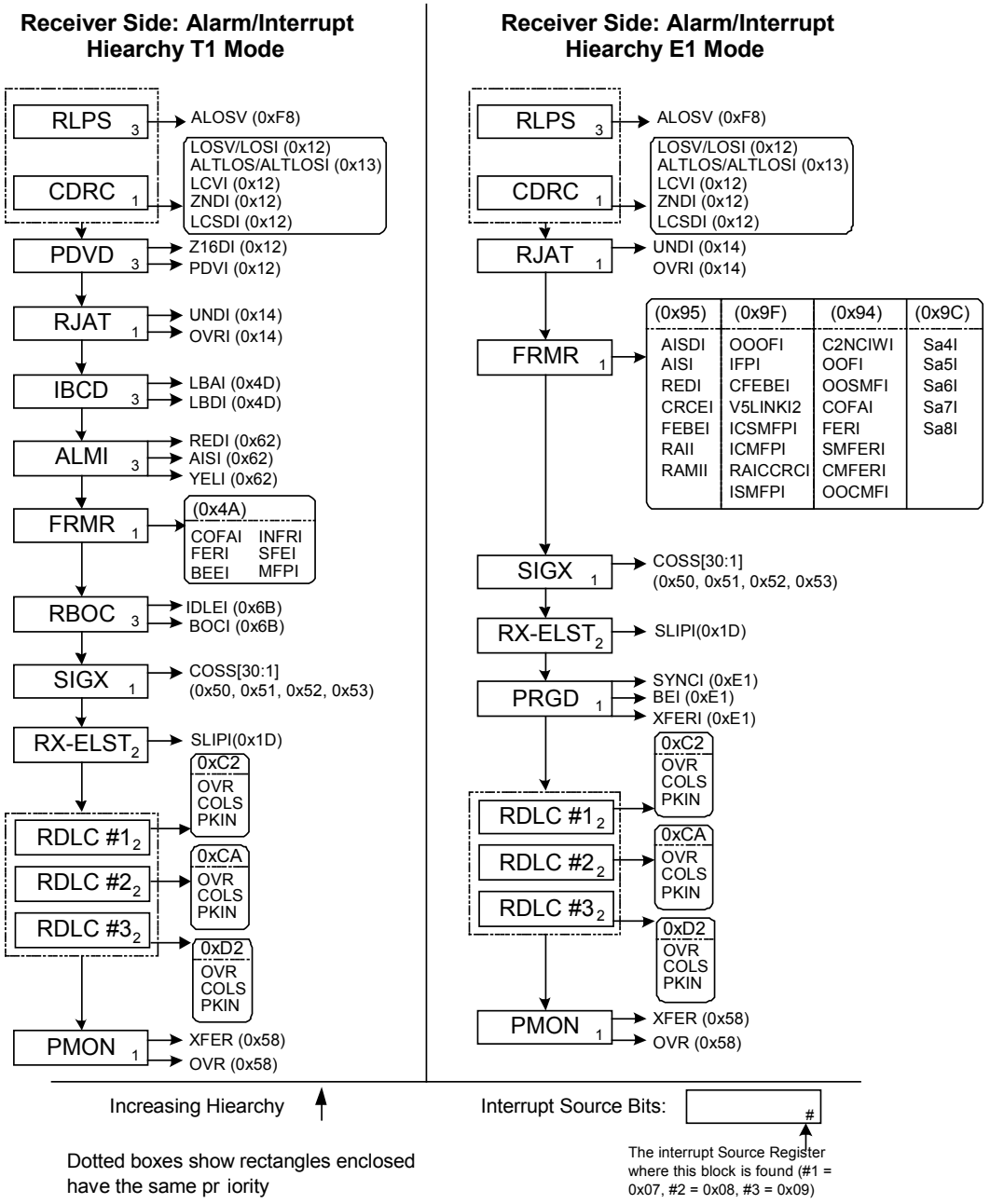


Figure 6: Receiver alarms and Interrupts for T1 and E1



9.2 T1 Mode

The sections below explain how to insert alarms into the Transmit stream and the backplane as well as why or how an alarm may occur.

9.2.1 T1 Alarm Insertion

The following tables describe how the COMET can be configured to insert T1 alarms in the transmit datastream. Each alarm may be caused by several different conditions.

The Alarm column represents the type of alarm. The "Condition" column identifies bits that have to be set to configure the alarm. The "Register" column simply states where the bit is located.

Table 15: Transmit Alarm Insertion (T1 Mode)

Alarm	Condition	Register	Comments
YELLOW	XYEL	0x55	
AIS	XAIS	0x55	
	TAISEN=1 & TUNI=0	0x04	AIS signal on TXTIP & TXRING pins
	TAISEN=1 & TUNI=1	0x04	AIS signal on TDAT pin
	TAIS & AISE	0xF1	TAIS bit (s/w AIS) is ORed with the external AIS input (h/w AIS) to enable AIS insertion.

Alarms are transparently sent to the Backplane Receive Interface. The AIS alarm may optionally be inserted based on the Condition below.

Table 16: Drop Alarm Insertion (T1 Mode)

Alarm	Condition	Register	Comments
AIS	AUTOAIS	0x03	AIS is inserted in the receive path and signaling frozen
	RAIS & (AUTOAIS (0x03) = 0)	0x0A	BRPCM pin is forced to all 1's

9.2.2 T1 Receiver Alarms

Table 17 below shows Alarms related to the Receive path of COMET. The columns show the alarms, the register bits that can affect or indicate the alarm, the register the bits are located, possible causes for the alarm and general comments.

Table 17: Receiver Alarms (T1 Mode)

Alarm	Condition	Register	Possible Cause(s)	Comments
ALOS	DET_PER[7:0] &	0xFA	Threshold is set too low; the link is physically down	ALOS Detection Threshold has been reached and ALOSI set.
	ALOSI	0xF8		
LOS	LOS[1:0] &	0x10	An incorrect threshold is set; the link is physically down	LOS Threshold has been reached
	LOSV	0x12		
PDV	PDV	0x65	16 consecutive zeros are detected; an incorrect line code is being transmitted	Pulse Density Violation
RED	REDD	0x63	An incorrect framing format is being received	An OOF condition has been present for 2.55 sec (± 40 ms)
AIS	AIS	0x62	Alarm Detected	The carry failure alarm had a state change
	AISD	0x63		AIS signal was present during the last 60 ms interval
YEL	YEL	0x62	Alarm Detected	The carry failure alarm had a state change
	YELD	0x63		Yellow signal was present during the last 40 ms interval
Out of Frame	INFR=0 (T1 Mode)	0x4A	Clock may not be synchronized	

9.3 E1 Mode

The sections below explain how to insert alarms into the Transmit stream and the backplane as well as why or how an alarm may occur.

9.3.1 E1 Alarm Insertion

The following tables describe how the COMET can be configured to insert E1 alarms in the transmit datastream. Each alarm may be caused by several different conditions.

The Alarm column represents the type of alarm. The "Condition" column identifies bits that have to be set to configure the alarm. The "Register" column simply states where the bit is located.

Table 18: Transmit Alarm Insertion (E1 mode)

Alarm	Condition	Register	Comments
Signaling Multiframe AIS	YBIT	0x81	
Timeslot 16 AIS	TS16AIS	0x81	
AIS	AIS	0x81	
	TAISEN=1 & TUNI=0	0x04	AIS signal on TXTIP & TXRING pins
	TAISEN=1 & TUNI=1	0x04	AIS signal on TDAT pin
RAIS	RAI	0x81	

Alarms are transparently sent to the Backplane Receive Interface. The AIS alarm may optionally inserted be inserted based on the Condition below.

Table 19: Drop Alarm Insertion (E1 mode)

Alarm	Condition	Register	Comments
AIS	AUTOAIS	0x03	AIS is inserted in the receive path and signaling frozen
	RAIS & (AUTOAIS (0x03) = 0)	0x0A	BRPCM pin is forced to all 1's
	OOSMFAIS=1	0x00	An OOSMF indication from the E1-FRMR will send all 1's onto the BRSIG pin

9.3.2 E1 Receiver Alarms

Table 20 below shows Alarms related to the Receiver part of COMET. The columns show the alarms, the register bits that can affect or indicate the alarm, the register location of the bits, possible causes for the alarm and general comments.

Table 20: Receiver Alarms (E1 mode)

Alarm	Condition	Register	Possible Cause(s)	Comments
ALOS	DET_PER[7:0] &	0xFA	Threshold is set too low; the link is physically down	ALOS Detection Threshold has been reached and ALOSI set.
	ALOSI	0xF8		
LOS	LOS[1:0] &	0x10	The incorrect threshold is set; the link is physically down	LOS Threshold has been reached
	LOSV	0x12		
PDV	PDV	0x65	16 consecutive zeros are detected; an incorrect line code is being transmitted	Pulse Density Violation

Alarm	Condition	Register	Possible Cause(s)	Comments
RED	RED	0x97	An incorrect framing format is being received	An OOF condition has been present for 100 ms
AIS	AISC	0x91	loss of frame	Depending on how the AISC bit is set, this alarm can cause the AISD bit (0x97) to be set.
	AIS	0x97		An out of frame all-ones condition has persisted for 100 ms
Out of Frame	OOFI (E1 Mode)	0x94	Clock may not be synchronized. Bits in register 0x90 may be incorrectly set. ie. CRCEN could be disabled; CASDIS may be incorrectly set.	
loss of signaling multiframe	OOSMFV	0x96		Bit represents the loss of signaling multiframe
loss of CRC multiframe	OOCMFV:	0x96		Bit represents the loss of CRC multiframe

It should be noted that if an ALOS or LOS alarm occurs at the receive interface, the RSYNC pin represents either the ALOS or LOS alarm based on how the RSYNC_ALOSB bit in the **Global Configuration**(0x00) register is set. When COMET is in a loss of signal state, the RSYNC output is derived from XCLK (unless RSYNCH_MEM is set in the **Receive Options**(0x02) register) rather than the recovered receiver clock.

10 PROGRAMMING THE PATTERN GENERATOR/DETECTOR

The Pattern Generator/Detector (PRDG) is used to generate and detect repetitive or pseudo-random patterns.

The **PRGD Control**(0xE0) register is used to determine the pattern type, repetitive or pseudo-random, by setting the PS bit to the required value. The register also contains other bits that may be configured to logically invert the generated pattern or received stream. The PS bit must be programmed before any other PRGD register to prevent the possibility of corruption of the pattern. This register also specifies how the Pattern Detector Registers (0xEC, 0xED, 0xEE, and 0xEF) will be used based on how the PDR[1:0] bits are programmed. See the "Pattern Detector Register Configurations" table in the COMET datasheet.

The "Pattern Detector Registers" may be configured to receive the pattern, accumulate error counts or accumulate bit counts. Bit errors and error counts are not accumulated while the pattern detector is out of synchronization. The SYNCV bit in **PRGD Interrupt Enable/Status**(0xE1) may be checked to determine if the pattern detector is synchronized.

The Pattern Generator is shown in the "Using the Test Pattern Generator" section in the COMET datasheet. See the section for programming standardized pseudo random and repetitive patterns.

Setting the desired interrupt bits from the **PRGD Interrupt Enable/Status register** may program interrupts. Interrupts may be set for bit errors, synchronization errors, accumulation intervals are complete, and overruns.

11 PERFORMANCE MONITORING

The COMET provides counters for performance monitoring and automatic performance monitoring for T1 mode.

11.1 Performance Monitoring Counters

The COMET's Performance Monitoring Counters (PMON) block accumulates: CRC error events, Frame Synchronization bit error events, Line Code Violation events, and Out Of Frame events. Optionally, Change of Frame Alignment (COFA) events, with saturating counters over consecutive intervals as defined by the period of the supplied transfer clock signal (typically 1 second), may be accumulated. When the transfer clock signal is applied, the PMON transfers the counter values into holding registers and resets the counters to begin accumulating events for the interval. The counters are reset in such a manner that error events occurring during the reset are not missed. If the holding registers are not read between successive transfer clocks, an OVERRUN register bit is asserted.

Generation of the transfer clock within the COMET chip is performed by writing to any counter register location or by writing to the Global PMON Update register. The holding register addresses are contiguous to facilitate faster polling operations.

Table 21 and Table 22 show all of COMET's performance counters for each of the counter types.

Table 21 : PMON registers for T1 mode

Counter Type	Register	Description
Framing Bit Error	0x59	Number of framing bit error events that occurred during the last accumulation interval
Out of Frame (OOF) Errors	0x5A-0x5B	Number of OOF events that occurred during the last accumulation interval
Bit Errors	0x5C-0x5D	Number of bit error events that occurred during the previous accumulation interval
Line Code Violation (LCV)	0x5E-0x5F	Number of bipolar violations or excessive zeros that occurred during the previous accumulation interval

Table 22: PMON Registers for E1 mode

Counter Type	Register	Description
Framing Bit Error	0x59	Number of framing bit error events that occurred during the last accumulation interval
Far End Block Errors	0x5A-0x5B	Number of far end block error events that occurred during the previous accumulation interval
CRC Errors	0x5C-0x5D	Number of CRC error events that occurred during the previous accumulation interval
Line Code Violation (LCV)	0x5E-0x5F	Number of bipolar violations for AMI-coded signals and HDB3-coded signals that occurred during the previous accumulation interval

11.2 Automatic Performance Monitoring

The COMET may automatically update performance reports on a per second basis if the AUTOUPDATE bit, in the **T1 APRM Configuration/Control** register, is set to logic 1. Table 22 below shows the message structure and contents of the performance report. Table 24 and Table 25 display bit explanations.

Table 23: Performance Report Message Structure and Contents

Octet No.	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
1	FLAG							
2	SAPI						C/R	EA
3	TEI							EA
4	CONTROL							
5	G3	LV	G4	U1	U2	G5	SL	G6
6	FE	SE	LB	G1	R	G2	Nm	NI
7	G3	LV	G4	U1	U2	G5	SL	G6
8	FE	SE	LB	G1	R	G2	Nm	NI
9	G3	LV	G4	U1	U2	G5	SL	G6
10	FE	SE	LB	G1	R	G2	Nm	NI
11	G3	LV	G4	U1	U2	G5	SL	G6
12	FE	SE	LB	G1	R	G2	Nm	NI
13	FCS							
14	FCS							

Octet No.	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
15	FLAG							

Notes:

1. The order of transmission of the bits is LSB (Bit 1) to MSB (Bit 8).

Table 24: Performance Report Message Structure Notes

Octet No.	Octet Contents	Interpretation
1	01111110	Opening LAPD Flag
2	00111000 00111010	From CI: SAPI=14, C/R=0, EA=0 From carrier: SAPI=14,C/R=1,EA=0
3	00000001	TEI=0,EA=1
4	00000011	Unacknowledged Frame
5,6	Variable	Data for latest second (T')
7,8	Variable	Data for Previous Second(T'-1)
9,10	Variable	Data for earlier Second(T'-2)
11,12	Variable	Data for earlier Second(T'-3)
13,14	Variable	CRC16 Frame Check Sequence
15	01111110	Closing LAPD flag

Table 25: Performance Report Message Contents

Bit Value	Interpretation
G1=1	CRC ERROR EVENT =1
G2=1	1<CRC ERROR EVENT ≤5
G3=1	5<CRC ERROR EVENT ≤10
G4=1	10<CRC ERROR EVENT ≤100
G5=1	100<CRC ERROR EVENT ≤319
G6=1	CRC ERROR EVENT ≥ 320
SE=1	Severely Errored Framing Event ≥ 1(FE shall =0)
FE=1	Frame Synchronization Bit Error Event ≥1 (SE shall=0)

Bit Value	Interpretation
LV=1	Line code violation event ≥1
SL=1	Slip Event ≥ 1
LB=1	Payload Loopback Activated
U1,U2=0	Under Study For Synchronization
R=0	Reserved (Default Value =0)
NmNI=00,01,10, 11	One second Report Modulo 4 Counter

12 DIAGNOSTICS

The COMET can perform different types of diagnostics. The following sections discuss them.

Diagnostic Modes

To perform diagnostics on COMET, the **Master Diagnostics**(0x0A) provides payload, line and diagnostic digital loopback modes(based on the PAYLB, LINELB, and DDLB bits). See the section "Using the Loopback Modes", in the COMET datasheet for information about these modes.

Diagnostic Digital Loopback

For the Diagnostic Digital Loopback Mode, the following pseudo-code could be used to verify that ABCD bits are correctly received in this mode (The pseudo-code is for timeslot 1). It is assumed that the COMET is in E1 mode for this:

```
WRITE_REG(0x0A 0x4) /* set the DDLB bit to 1 to enable the DDLB
                       loopback */
WRITE_REG(0x6C 0x1) /* set IND and PCCE to logic 1 for TPSC */
WRITE_REG(0x50 0x1) /* set IND and PCCE to logic 1 for SIGX */

while (Read 0x96 != 0) ; * wait until E1 FRMR framing statuses are
                       Correct */

WRITE_REG(0x6F 0x15) /*set the ABCD bits = 0101 for the TPSC to
                       transmit*/
WRITE_REG(0x6E 0x61) /*set the TPSC address for timeslot 1 */
WRITE_REG(0x52 0xA1) /*set the SIGX address for timeslot 1 to read
                       from */

value = READ_REG(0x53) /* read the value from SIGX time slot 1 */
if (value == 0x05) { /* hence ABCD = 0101 */
      printf ("Correct ABCD code for timeslot 1");
}
else {
      printf ("Error in ABCD code for timeslot 1");
}
```

Analog Diagnostics

To program analog diagnostics, the **Analog Diagnostics**(0x0C) may be configured. To confirm that the CDRC, RLPS and CSU are working correctly, the

user can program the Diagnostic mode for CDRC, RLPS, and CSU Monitor modes.

In one possible configuration the RDAT and RCLKI pins are reconfigured as outputs by setting the RXDIAG[1] pin to logic 0 and the RXDIAG[2] pin to logic 1 (**Analog Diagnostics** register). The RDAT output then carries RPCM output of the CDRC block, which may then be compared to verify the functionality of the CDRC block. For details on all possible configurations, see the **Analog Diagnostics** register description in the datasheet.

13 REFERENCE

- PMC-1961273, PMC-Sierra, "COMBINED E1/T1 TRANSCEIVER Standard Product Datasheet, Issue 8

CONTACTING PMC-SIERRA, INC.

PMC-Sierra, Inc.
105-8555 Baxter Place Burnaby, BC
Canada V5A 4V7

Tel: (604) 415-6000

Fax: (604) 415-6200

Document Information: document@pmc-sierra.com
Corporate Information: info@pmc-sierra.com
Application Information: apps@pmc-sierra.com
(604) 415-4533
Web Site: http://www.pmc-sierra.com

PMC-1990615 (R4) ref PMC-xxxxxx (Rx) Issue date: August 2000