



*OpenATE Inc.*  
The Open Solution for IC Tester

---

# PE16 User Manual



## Revision History

The following lists the additions, deletions and modifications in this manual at each revision.

Date	Version	Remarks
January 10, 2010	1	Release the document
April 28, 2010	2	

---



## Table of Contents

<b>No</b>	<b>Description</b>	<b>Page</b>
1	Overview	4
2	Block	5
2.1	Features	5
2.2	DC Specification	6
2.3	Digital Specification	7
2.4	Sequencer	8
2.5	Formater	9
2.6	Function Block	10
2.7	Local Memory Structure(LMSEQ+LMF)	11
2.8	Ucode Definition(LMSEQ)	12
2.9	Sequencer Structure	13
2.10	Formater Structure	14
2.11	Pin Electronics	15
3	API Functions	16
3.1	Sequencer Functions	16
3.2	Formater Functions	23
3.3	Pin Electronics	26
3.4	Calibration	30
4	Pattern Format	34
5	Pin Connector	35
	Figure1. PE16 Connector Pin Out	37



*OpenATE Inc.*

The Open Solution for IC Tester

---

## **1 Overview**

This document provides detailed description of PE16 functions.

PE16 is a PXI 3U logic IC tester with very high density and performance. It can offer the Test Solution Provider (TSP) the necessary functions to test digital channels when configuring a solution for a device. Many PE16s can be chained together to get higher pin counts. PE16 can use PXI-TRIGGER to synchronize with other PXI instruments. PE16 offers intensive APIs in C language that can control all the detailed hardware functions. TSP can use PE16 to build a customized test system or solution.



## **2 Block**

### **2.1 Features**

1. PXI 3U Logic Tester
2. Tester Per Board Architecture
3. 50MHz Test Rate TG/RVS Per Pin
4. 16 BI-Direction Logic Test Pins
5. MAX. 128 Pin,8 Site Parallel Test
6. 64 Local Memory
7. 256 I/O, Mask Set



## **2.2 DC Specification**

1. RVS Per Pin -1 ~+6V 16Bit
2. Per Pin PMU -1 ~ +6V 16Bit 32mA
3. 8 Current Range : 2UA,8UA,32UA,128UA,512UA,2MA,8 MA,32MA
4. Per Pin ACTIVE LOAD -1 ~+6V /32mA



## **2.3 Digital Specification**

1. TP MIN. 20NS(50MHz) 5NS Resolution
2. TG Per Pin 5 NS Resolution ,skew resolution 312.5pS
3. Local Memory 64M Per Channel
4. 16 TIME SET SWITCHING On the Fly
5. Window STROBE



## **2.4 Sequencer**

1. uCommand : FC,FC MATCH,FLOOP,LEND
2. FC Counter : 20 Bits
3. LMSYN to PXI TRIGGER BUS
4. Start TRIGGER from PXI TRIGGER BUS
5. Fail to PXI TRIGGER BUS
6. FT Counter : 32 Bits
7. Cycle mode
8. Ignore Fail by LM Address

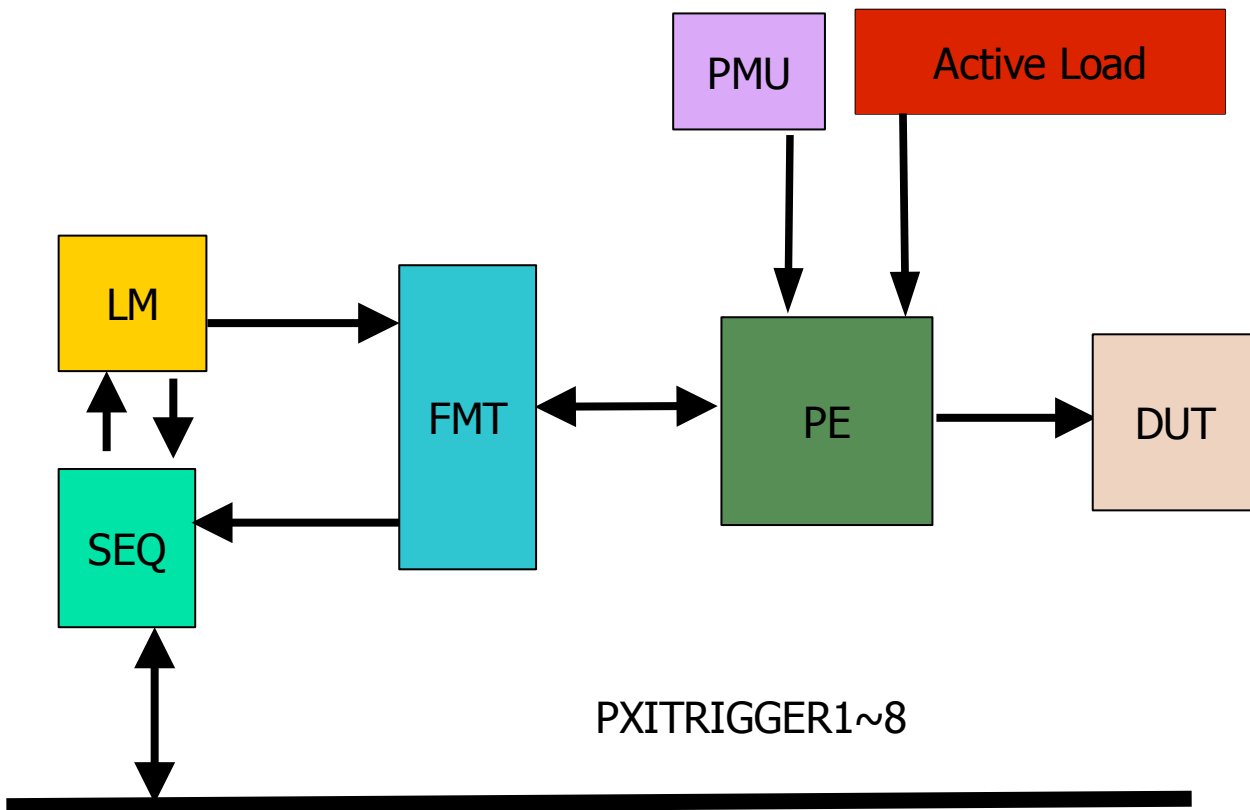


## **2.5 Formater**

1. F,RZ,RO,SBC
2. Drive : 1,0,X,J,Q
3. Receive : H,L,Z,X
4. 16 Format Set Switching on the Fly



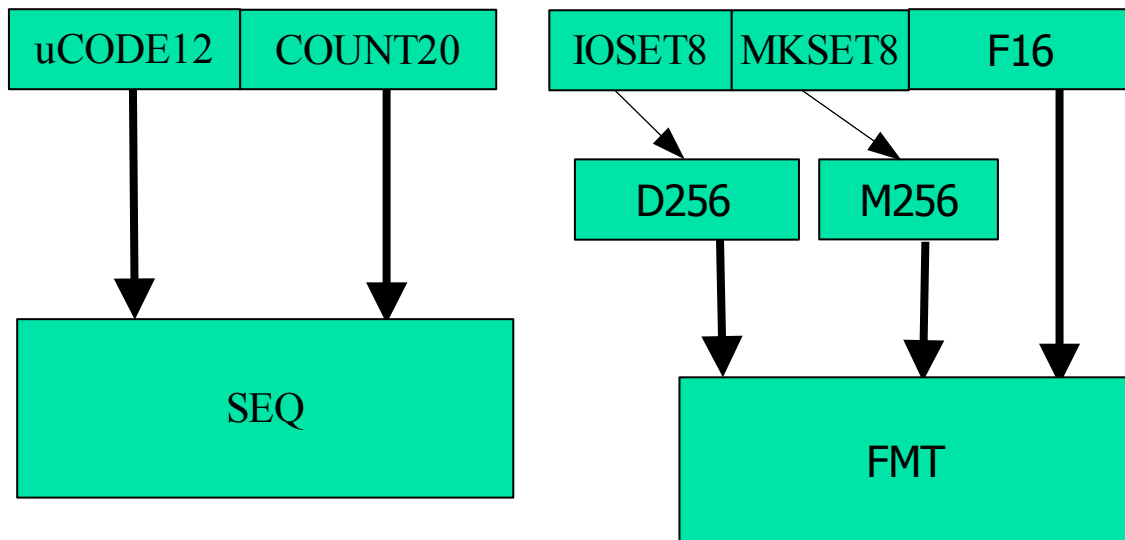
## 2.6 Function Block





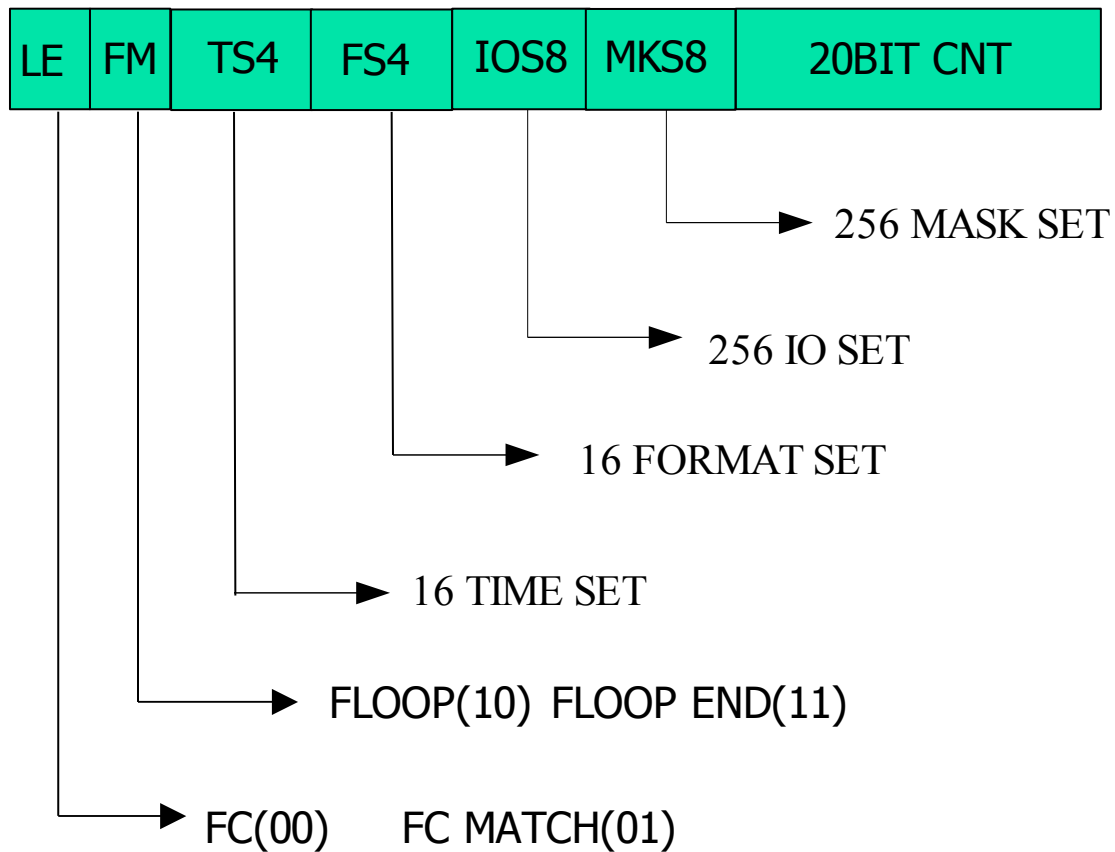
## 2.7 Local Memory Structure(LMSEQ+LMF)

32BIT X 2 X (64 M)





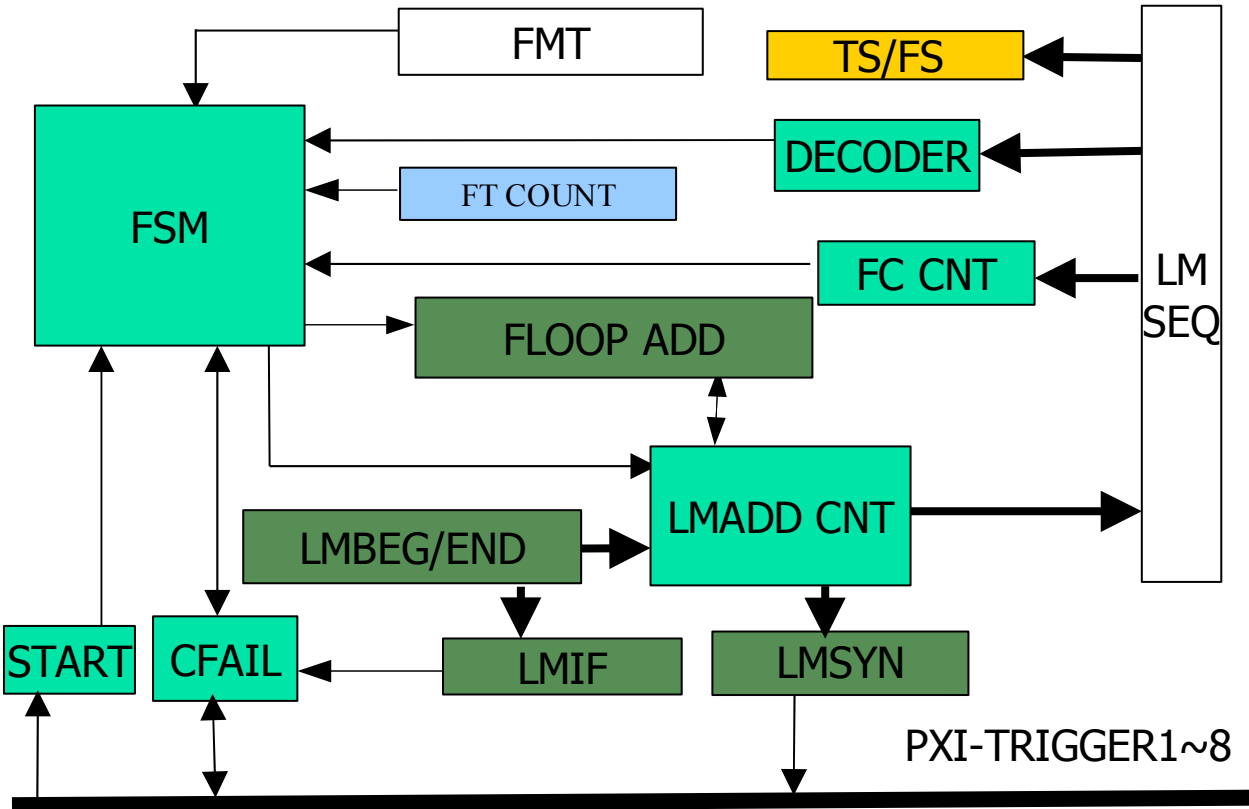
## 2.8 Ucode Definition(LMSEQ)



1. FC : REPEAT CNT TIMES
2. FC MATCH(FM) :REPEAT UNTIL PASS IN CNT TIMES
3. FLOOP (FL): LOOP BLOCK TO FLOOP END CNT TIMES
4. FLOOP END(FE) :REPEAT CNT TIMES AND END LOOP IF LOOP COUNT REACHED

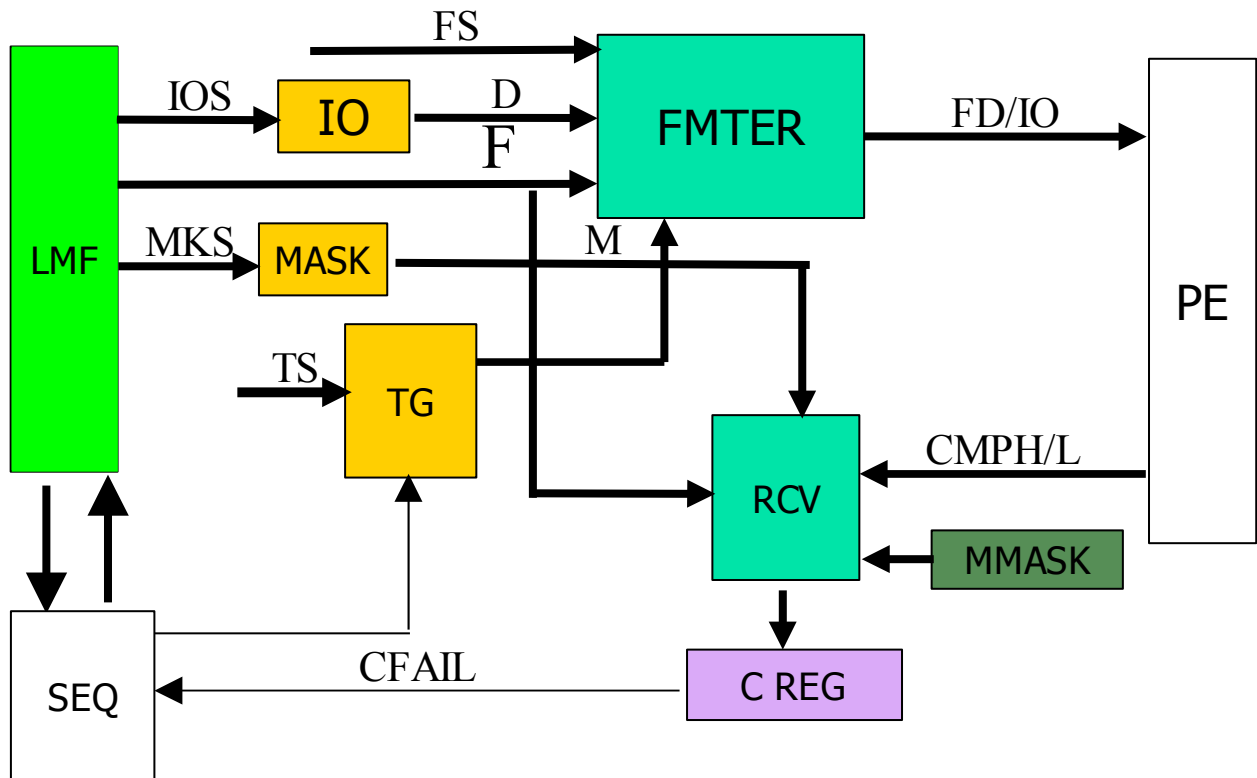


## 2.9 Sequencer Structure



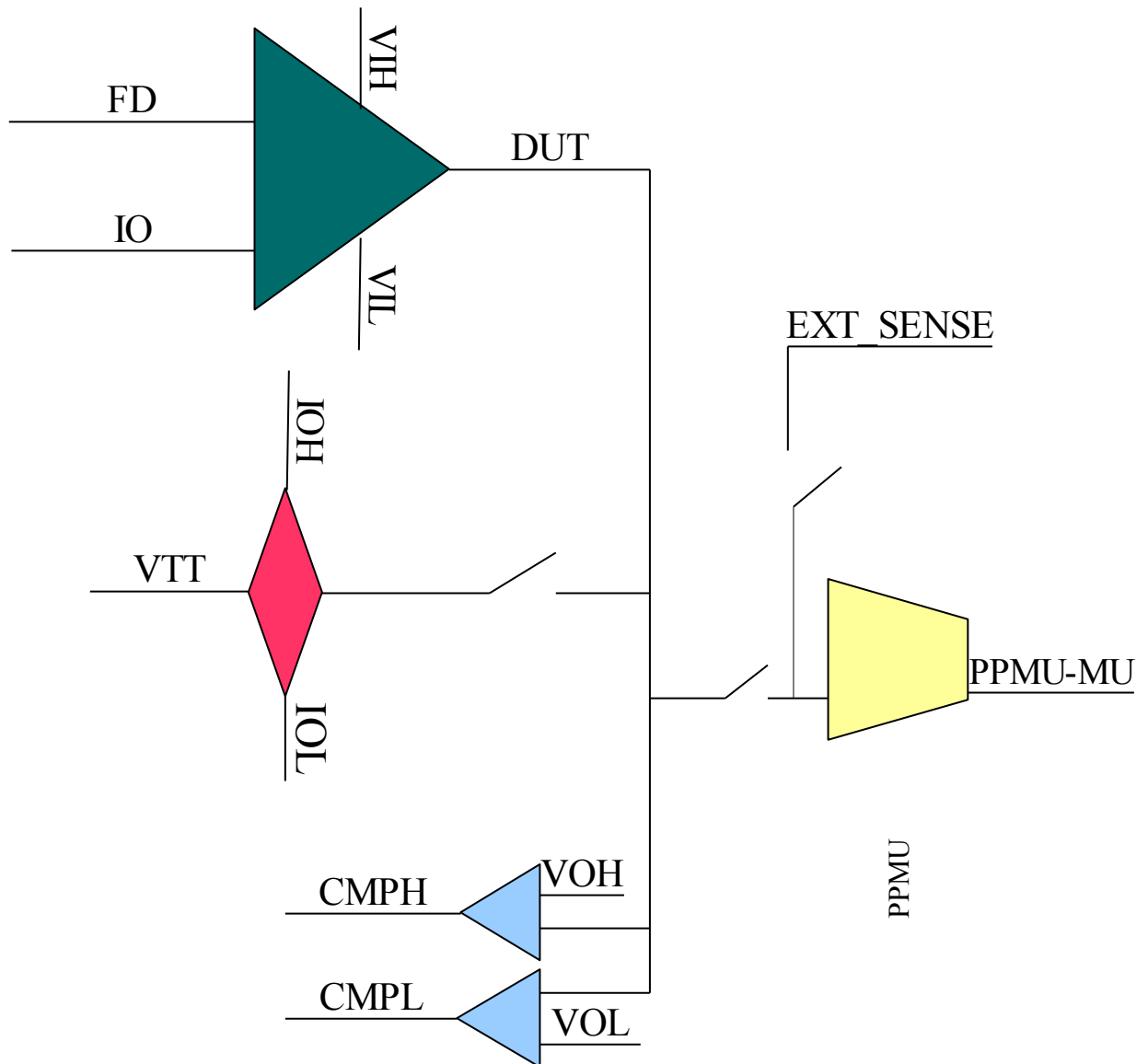


## 2.10 Formatter Structure





## 2.11 Pin Electronics





## 3 API Functions

PE16 API Functions:

### 3.1 Sequencer Functions

#### Definition

```
int pe16_init(void);
```

#### Return Value

board counts in system

#### Description

This function must be executed before any other functions.

If return value is 0, no PE16 detected.

#### Definition

```
void pe16_reset(int bdno);
```

#### Description

reset sequencer and set FTCNT to 1 , CREG,ADDBEG,ADDEND, ADDSYN ADDIF to 0,MMSKto 0xFF

if bdno=0, reset all boards

#### Definition

```
void pe16_set_ftcnt(int bdno, long cnt);
```

#### Description

set FTCNT reg(32bits) to cnt for board bdno,if bdno=0, set all boards

FTCNT is used to set loop test count.The pattern will be run from ADDBEG to ADDEND in FTCNT times.

#### Definition

```
void pe16_set_addbeg(int bdno, long add);
```

#### Description

set LMREG reg(26bits) to add for board bdno,if bdno=0, set all boards

LMADD will be loaded with LMREG;

#### Definition

```
void pe16_set_addend(int bdno, long add);
```

#### Description

set LMEND reg(26bits) to add for board bdno,if bdno=0, set all boards



**Definition**

void pe16\_set\_addif(int bdno, long add);

**Description**

set LMIF reg to add for board bdno, if bdno=0, set all boards when LMADD<= LMIF compare result will be ignored.

**Definition**

void pe16\_set\_addsyn(int bdno, long add);

**Description**

set LMSYN(26bits) reg to add for board bdno, if bdno=0, set all boards

LMSYN is used to generate LMSYNC signal which can be connect to PXI\_TRIG by setting PXI reg or output to connector 1

**Definition**

void pe16\_fstart(int bdno , int onoff);

**Description**

start/stop sequencer (run pattern) , if bdno=0, set all boards

Before using this function, LMBEG, LMEND has to be set to adequate value.

Set onoff to 0 to stop sequencer.

**Definition**

void pe16\_set\_trigmode(int bdno , int onoff);

**Description**

select TRIGMODE for board bdno, if bdno=0, set all boards

When TRIGMODE=0(default), sequencer is started by pe16\_fstart(int bdno , int onoff).

When TRIGMODE=1, the EXTTRIG signal will replce pe16\_fstart(int bdno , int onoff) to control the sequencer.

**Definition**

int pe16\_check\_tprun(int bdno);

**Return Value**

= 0 : sequencer stopped, = 1 : sequencer running

**Description**

check if sequencer running , bdno>0;

The sequencer stops in two conditions:

1. CFAIL detected from formater or PXI-TRIGGER bus.
2. LMEND reached and FT COUNTER count to the value defined by pe16\_set\_ftcnt(int bdno, long cnt);



**Definition**

int pe16\_check\_tpass(int bdn);

**Return Value**

= 1: no CFAIL detected by sequencer

**Description**

WHEN function test pass then TPASS=1, should be checked before setting tsatrt to 0, bdn>0; pe16\_fstart(bdn , 0) will reset TPASS;

**APPLICATION:**

User can make a user function for running pattern:

```
int FTEST(int bdn, unsigned long lbeg , unsigned long lend )
{
int rst;
pe16_set_checkmode(bdn, 0);
pe16_set_addbeg(bdn, lbeg);
pe16_set_addend(bdn, lend);
pe16_cycle(bdn, 0);
pe16_fstart(bdn, 1);
while (pe16_check_tprun(bdn) ); // wait for sequencer stop
rst=pe16_check_tpass(bdn);
pe16_fstart(bdn, 0);
return(rst);
}
```

This function can run pattern from lbeg to lend. If return value = 1, test pass; ###The pattern run must start from FC seq command.

**Definition**

void pe16\_cycle(int bdn , int onoff);

**Description**

set CYCLE mode on/off, if bdn=0, set all boards

In CYCLE mode , sequencer will run from LMEBG to LMEND and back to LMBEG with unlimited times. Use pe16\_fstart(bdn , 0) to stop sequencer. In CYCLE mode, any FAIL signal from receiver is ignored.

**Definition**

int pe16\_check\_sync(int bdn);

**Return Value**

LMSYNC bit

**Description**

LMSYNC=1 when LMSYN=LMADD , bdn>0;

Used for diagnosis



**Definition**

int pe16\_check\_testbeg(int bdno);

Return Value

TESTBEG bit

**Description**

return TESTBEG bit on CTRLRD reg, bdno>0;

Used for diagnosis

**Definition**

int pe16\_check\_ftend(int bdno);

Return Value

FTEND bit

**Description**

FTEND=1 when FT counter reach FTCNT set by pe16\_set\_fcnt(int bdno, long cnt), bdno>0;

Used for diagnosis

**Definition**

int pe16\_check\_lend(int bdno);

Return Value

LEND bit

**Description**

LEND=1 when LMEND=LMADD, bdno>0;

Used for diagnosis

**Definition**

void pe16\_set\_pxi(int bdno, int data);

**Description**

set PXI reg(16bits) to data for board bdno, if bdno=0, set all boards

PXI reg is for controll of PXI\_TRIG[0..7] BUS MSB and LSB of PXI reg will controll corespondent channel of PXI\_TRIG BUS

MSB[n]LSB[n]="00" : not any connection to PXI\_TRIG

MSB[n]LSB[n]="01" : output LMSYNC signal to PXI\_TRIG[n] n=0..7

MSB[n]LSB[n]="10" : in/out for CFAIL signal

MSB[n]LSB[n]="11" : in/out use PXI\_TRIG[n] as sequencer start channel (in and out)

## when using PXI\_TRIG the DIP SWITCH bit must be turn on on carrier board

**Definition**

void pe16\_set\_seq(int bdno, long data);

**Description**

set LM SEQ to data at LMADD for board bdno, if bdno=0, set all boards . Used for writing local memory



**Definition**

```
void pe16_set_lmf(int bdno, long data);
```

**Description**

set LM F to data at LMADD for board bdno,if bdno=0, set all boards . Used for writing local memory

**Definition**

```
long pe16_rd_seq(int bdno);
```

**Return Value**

LMSEQ

**Description**

return LM SEQ data at LMADD for board bdno. Used for checking local memory

**Definition**

```
long pe16_rd_lmf(int bdno);
```

**Return Value**

LMF

**Description**

return LM F data at LMADD for board bdno. Used for checking local memory

**Definition**

```
long pe16_rd_lmadd(int bdno);
```

**Return Value**

LMADD

**Description**

return LMADD for board bdno. Used for checking local memory

**APPLICATION:**

User can make a user function for checking pattern:

```
void CKLM(int bdn,unsigned long lbeg , unsigned long lend )
{
  unsigned long j;
  pe16_set_checkmode(bdn, 1);
  for (j=lbeg;j<=lend;j++)
  {
    pe16_set_addbeg(bdn,j);
    printf("BDN %d lmadd %d seq 0x%08lX lmf 0x%08lX \n",bdn,pe16_rd_lmadd(bdn),
    pe16_rd_seq(bdn),pe16_rd_lmf(bdn));
  }
  pe16_set_checkmode(bdn, 0);
}
```



**Definition**

unsigned int pe16\_rd\_fcncnt(int bdno);

Return Value

FCCNT(20bit)

**Description**

FCCNT is set by FC or FM in pattern. It counts down by every TP until 0.

Used for diagnosis

**Definition**

unsigned int pe16\_rd\_flcncnt(int bdno);

Return Value

FLCNT(20bit)

**Description**

FCCNT is set by FL in pattern. It counts down by every TP until 0.

Used for diagnosis

**Definition**

unsigned int pe16\_rd\_ftcncnt(int bdno);

Return Value

FT(32 bits)

**Description**

FT counter controls the loop counts the sequencer had run. Each LEND will increase the FT by one until reaching FTCNT

Used for diagnosis

**Definition**

int pe16\_check\_checkmode(int bdno);

Return Value

CHECKMODE bit

**Description**

CHECKMODE set by pe16\_set\_checkmode(int bdno, int onoff), bdno>0;

Used for diagnosis

**Definition**

int pe16\_check\_dataready(int bdno);

Return Value

DATAREADY bit

**Description**

Each time when LMBEG changed, DATAREADY goes to 0 until LM data updated from DDR11.

DATAREADY =1 indicate LM data available, bdno>0

Used for diagnosis



**Definition**

long pe16\_check\_ucnt(int bdno);

Return Value

UCNT(32 bits)

**Description**

UCNT counts the TP steps sequencer has run, bdno>0;

Used for diagnosis

**Definition**

void pe16\_set\_checkmode(int bdno, int onoff);

**Description**

set checkmode bit for board bdno,if bdno=0, set all boards

CHECKMODE is used to set LM data accessed without passing through cache memory. This should be enabled only when LM checking is practicing. It can save time for LM read/ write. It is reseted to '0' when system reset. Only pe16\_set\_addbeg() is affected by this controll bit.

**Definition**

void pe16\_set\_dualmode(int bdno, int onoff);

**Description**

set dualmode bit for board bdno,if bdno=0, set all boards

Dual site mode will devide 16 ch to two 8 ch site(1~8,9~16). FM command will interprete to FC in dual site mode. User can read back CREG to check out witch site pass/fail after running pattern.

**Definition**

int pe16\_check\_dualmode(int bdno);

Return Value

DUALMODE bit

**Description**

return DUALMODE bit set by pe16\_set\_dualmode(int bdno, int onoff), bdno>0;

Used for diagnosis



## 3.2 Formater Functions

### Definition

```
void pe16_set_mmsk(int bdno, long data);
```

### Description

set MMSK to data for board bdno,if bdno=0, set all boards after reset MMSK=0xFFFF when set to '0' will mask compare result for relative bit.

### Definition

```
void pe16_set_tp(int bdno, int ts,long data);
```

### Description

set TP to data at ts for board bdno,if bdno=0, set all boards,ts=1..16,if ts=0, set all time sets, tp is 16 bit counter ,unit:5ns Minimum is 4(50MHz)

### Definition

```
void pe16_set_tstrob(int bdno, int pno,int ts,long data);
```

### Description

set TSTROB to data at ts for board bdno port pno,if bdno=0, set all boards,ts=1..16,,if ts=0, set all time sets, pno=1..16,if pno=0, set all pins ,data is 16 bits,unit:5ns  
Test fail will stop at next step (run over) when TP-TSTROB<60ns  
When using FM command(matching) the nimimumTP is 100ns to working correctly(because of CFAIL recovery time).

### Definition

```
void pe16_set_dstrob(int bdno, int pno,int ts,int data1,int data2);
```

### Description

set window strobe to data1 and data2 at ts for board bdno port pno,if bdno=0, set all boards,ts=1..16,if ts=0, set all time sets,pno=1..16,if pno=0, set all pins ,data is 16 bits,unit:5ns.  
When using pe16\_set\_tstrob, the default mode is edge strobe. If useing pe16\_set\_dstrob, the window strobe will be acitved. The compare strobe will be from data1 to data2. If data=0, then edge strobe .The window strobe can be set on per pin base.

### Definition

```
void pe16_set_tstart(int bdno, int pno,int ts,long data);
```

### Description

set TSTART to data at ts for board bdno pin pno,if bdno=0, set all boards,ts=1..16,if ts=0, set all time sets,pno=1..16,if pno=0, set all pins,data is 16 bits,unit:5ns



**Definition**

void pe16\_set\_tstop(int bdno, int pno,int ts,long data);

**Description**

set TSTOP to data at ts for board bdno pin pno,if bdno=0, set all boards,ts=1..16,if ts=0, set all time sets,pno=1..16,if pno=0, set all pins,data is 16 bits,unit:5ns

**Definition**

void pe16\_set\_rz(int bdno, int fs,long data);

**Description**

set RZ to data at fs for board bdno,if bdno=0, set all boards,fs=1..16 , if fs=0, set all format sets,RZ is 16bit each bit map to one channel.

**Definition**

void pe16\_set\_ro(int bdno, int fs,long data);

**Description**

set RO to data at fs for board bdno,if bdno=0, set all boards,fs=1..16,if fs=0, set all format sets,RO is 16bit each bit map to one channel

OUTPUT format is defined by RZ and INV

RZ:RO=00 =>NRZ

RZ:RO=01 =>RTO

RZ:RO=10 =>RTZ

RZ:RO=11 =>SBC

**Definition**

long pe16\_lmload(int begbdno,int boardwidth,long begadd, char\* patternfile );

**Return Value**

last LMADD pattern loaded

**Description**

load pattern file to pe16 board from board no begboard to (begboard+boardwidth-1) ,begin at lmadd begadd .

For patternfile string, use "/" instead of "\".

Example : use "[c://OpenATE//patternfile](#)" for "c:\OpenATE\patternfile" .

**Definition**

unsigned int pe16\_rd\_cmph(int bdno);

**Return Value**

CMPH(16 bits)

**Description**

CMPH(bit)= 1 when DUT >VOH



**Definition**

unsigned int pe16\_rd\_cmpl(int bdno);

**Return Value**

CMPL(16 bits)

**Description**

CMPL(bit)=1 when DUT > VOL

**Definition**

unsigned int pe16\_rd\_creg(int bdno);

**Return Value**

CREG(16 bits)

**Description**

CREG(bit) =1 when compare result failed



### 3.3 Pin Electronics

#### Definition

`void pe16_set_vih(int bdno, int pno, float rv);`

#### Description

set VIH to rv for board bdno pin pno, if bdno=0, set all boards, if pno=0, set all pins

#### Definition

`void pe16_set_vil(int bdno, int pno, float rv);`

#### Description

set VIL to rv for board bdno pin pno, if bdno=0, set all boards, if pno=0, set all pins

#### Definition

`void pe16_set_voh(int bdno, int pno, float rv);`

#### Description

set VOH to rv for board bdno pin pno, if bdno=0, set all boards, if pno=0, set all pins

#### Definition

`void pe16_set_vol(int bdno, int pno, float rv);`

#### Description

set VOL to rv for board bdno pin pno, if bdno=0, set all boards, if pno=0, set all pins

#### Definition

`void pe16_set_pmu(int bdno, int pno, int onoff);`

#### Description

power on/off PMU for board bdno pin pno, if bdno=0, set all boards, if pno=0, set all pins onoff=1 : power on.

#### Definition

`void pe16_pmufv(int bdno, int pno, float rv);`

#### Description

set PMUFV to rv for board bdno pin pno, if bdno=0, set all boards, if pno=0, set all pins, rang 0..7  
 $-1.0 < rv < 6.0V$

#### Definition

`void pe16_pmufi(int bdno, int pno, float ri, float clampvh, float clampvl);`

#### Description

set PMUFI to ri mA and clamp V for board bdno pin pno, if bdno=0, set all boards, if pno=0, set all pins, if ri>32mA disable PPMU



**Definition**

void pe16\_pmucv(int bdno, int pno, float cvh, float cvl);

**Description**

set PMU compare V limit to  $cvh(PPMU-CH)/cvl(PPMU-CL)$  for board bdno pin pno, if bdno=0, set all boards, if pno=0, set all pins

**Definition**

void pe16\_pmuci(int bdno, int pno, float cih, float cil);

**Description**

set PMU compare I limit to  $cih/cil$  for board bdno pin pno, if bdno=0, set all boards, if pno=0, set all pins

**Definition**

void pe16\_con\_pmu(int bdno, int pno, int onoff);

**Description**

connect/disconnect PPMU to DUT for board bdno pin pno, if bdno=0, set all boards, if pno=0, set all pins

**Definition**

int pe16\_check\_pmu(int bdno, int pno);

**Return Value**

1 when pass

**Description**

PPMU-CH/CL compare result pass=1/fail=0 for board bdno pin pno  
pass when  $PPMU-CL < DUT < PPMU-CH$ .

**Definition**

int pe16\_pmuch(int bdno, int pno);

**Return Value**

1 when  $DUT > PPMU-CH$

**Description**

PPMU-CH compare result for board bdno pin pno

**Definition**

int pe16\_pmucl(int bdno, int pno);

**Return Value**

1 when  $DUT > PPMU-CL$

**Description**

PPMU-CL compare result for board bdno pin pno



## APPLICATION:

User can make a user function for PMU measurement:

```
int FVMI(int bdn, int pno , float fv, float cih, float cil )
{
pe16_set_driver(bdn,pno,0);//disable driver
pe16_set_pmu(bdn, pno,1);// power on pmu
pe16_con_pmu(bdn, pno,1);//connect pmu to DUT
pe16_pmufv(bdn, pno ,fv);// force volatge
pe16_pmuci(bdn, pno ,cih,cil);//set compare limits
_sleep(100);//wait for hardware statble
return(pe16_check_pmu(bdn,pno));
}
```

### Definition

```
void pe16_con_ext_sense(int bdno, int pno,int onoff);
```

### Description

connect/disconnect PMU V-SENSE to EXT\_SENSE for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins onoff=1 : connected

### Definition

```
void pe16_set_active_load(int bdno, int pno,int onoff);
```

### Description

connect/disconnect ACTIVE\_LOAD to DUT for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins onoff=1 : connected

### Definition

```
void pe16_set_vtt(int bdno, int pno,float rv);
```

### Description

set VTT to rv for board bdno port pno,if bdno=0, set all boards,if pno=0, set all pins

### Definition

```
void pe16_set_ioh(int bdno, int pno ,float ri);
```

### Description

set IOH to ri mA resolution 0.4883uA for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins, if ri>32mA or < 0 disable active load

### Definition

```
void pe16_set_iol(int bdno, int pno ,float ri);
```

### Description

set IOL to ri mA resolution 0.4883uA for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins, if ri>32mA or < 0 disable active load



**Definition**

void pe16\_set\_deskew(int bdno, int pno,int rt);

**Description**

set Driver deskew to rt (0..15) for board bdno port pno,if bdno=0, set all boards,if pno=0, set all pins. unit 312.5ps 0ns(0x0) to +4.6875ns(0xF)

**Definition**

void pe16\_con\_ext\_sense(int bdno, int pno,int onoff);

**Description**

connect/disconnect PMU V-SENSE to EXT\_SENSE for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins onoff=1 : connected



## 3.4 Calibration

To get an accurate DC level, calibration is necessary.

For all DC level offset adjustment,  $vr = 0x0000(-5.4\% \text{ of FullScale } 8V/32mA) \sim 0x7FFF(0) \sim 0xFFFF(+5.4\% \text{ of FS})$

For DC level gain adjustment,  $vr = 0x0000(0.875) \sim 0x7FFF(1.0) \sim 0xFFFF(1.125)$

The default offset and gain are set to  $0x7FFF$  by `pe16_rst_pe`.

In general, most calibration points will be at 20% and 80% of the full scale value for that range.

$V_{mid} = 3.0$  for voltage level,  $V_{mid} = 0.0$  for current level.

### Level Calibration

#### Initialize

- Set Gain = 1.0; Offset = 0.0V( $0x7FFF$ )

#### Measure

- Set Level 1 = Cal Point 1. Measure Output1'
- Set Level 2 = Cal Point 2. Measure Output2'

#### Calculate

- $Gain' = (Output2' - Output1') / (Level2 - Level1)$
- $Offset' = (Output2' - V_{mid}) - Gain' \cdot (Level2 - V_{mid})$

#### Finish

- Set Offset =  $- Offset' / Gain'$
- Set Gain =  $1.0 / Gain'$

#### Definition

```
void pe16_set_vih_offset(int bdno, int pno,int rv);
```

#### Description

set VIH offset to rv for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins offset and gain are for calibration.

#### Definition

```
void pe16_set_vil_offset(int bdno, int pno,int rv);
```

#### Description

set VIL offset to rv for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins

#### Definition

```
void pe16_set_voh_offset(int bdno, int pno,int rv);
```

#### Description

set VOH offset to rv for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins

#### Definition

```
void pe16_set_vol_offset(int bdno, int pno,int rv);
```

#### Description

set VOL offset to rv for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins



**Definition**

void pe16\_pmufv\_offset(int bdno, int pno ,int rv);

**Description**

set PMUFV offset to rv for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_pmufi\_offset(int bdno, int pno ,int ri);

**Description**

set PMUFI offset to ri for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_pmuvclamp\_offset(int bdno, int pno ,int cvh,int cvl);

**Description**

set PMU clamp V offset to cvh/cvl for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_pmucv\_offset(int bdno, int pno ,float cvh,float cvl);

**Description**

set PMU compare V offset limit to cvh/cvl for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_set\_vtt\_offset(int bdno, int pno,int rv);

**Description**

set VTT offset to rv for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_set\_ioh\_offset(int bdno, int pno ,int ri);

**Description**

set IOH offset to ri for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_set\_iol\_offset(int bdno, int pno ,int ri);

**Description**

set IOL offset to ri for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_set\_vih\_gain(int bdno, int pno,int rv);

**Description**

set VIH gain to rv for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins



**Definition**

void pe16\_set\_vil\_gain(int bdno, int pno,int rv);

**Description**

set VIL gain to rv for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_set\_voh\_gain(int bdno, int pno,int rv);

**Description** set VOH gain to rv for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_set\_vol\_gain(int bdno, int pno,int rv);

**Description**

set VOL gain to rv for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_pmufv\_gain(int bdno, int pno ,int rv);

**Description**

set PMUFV gain to rv for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_pmufi\_gain(int bdno, int pno ,int ri);

**Description**

set PMUFI gain to ri for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_pmuvclamp\_gain(int bdno, int pno ,int cvh,int cvl);

**Description**

set PMU clamp V gain to cvh/cvl for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_pmucv\_gain(int bdno, int pno ,float cvh,float cvl);

**Description**

set PMU compare V gain limit to cvh/cvl for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins



**Definition**

void pe16\_set\_vtt\_gain(int bdno, int pno,int rv);

**Description**

set TT gain to rv for board bdno pin pno,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_set\_ioh\_gain(int bdno, int pno ,int ri);

**Description**

set IOH gain to ri for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins

**Definition**

void pe16\_set\_iol\_gain(int bdno, int pno ,int ri);

**Description**

set IOL gain to ri for board bdno pin pno ,if bdno=0, set all boards,if pno=0, set all pins



## 4 Pattern Format

Basic Commands:

- 1.FC counts :Repeat pattern counts times and go to next pattern.
- 2.FL loops :Run pattern one time and go to next pattern. When hit FE return to this pattern loops times.
- 3.FE :Repeat pattern 1 times and go to previous FL loops times.
4. FM counts :Repeat pattern counts times before pattern pass. If pattern matched before counts, go to next pattern or test failed for others.

Note:counts and loops are from 1~1M(20 BIT)

Time Commands

- 1.TSn :Select time sets n from 1~16, if not used, compiler will use previous time set

Format Commands

- 1.FSn :Select time sets n from 1~16, if not used, compiler will use previous format set

Data Types

1. '1' :Drive 1
2. '0' :Drive 0
3. 'H' :Expect 1
4. 'L' :Expect 1
5. 'X' :Tri-state out
1. 'Z' :Tri-state in( < VOH, > VOL)
6. 'J' :Drive 1 Expect 1
7. 'Q' : Drive 0 Expect 0

Examples:

// DEFAULT PATTERN IS 'X' TS1 OR PREVIOUS IF NOT FIRST LINE

```
FC 1024    011000HLHHH1100XX;//REPEAT 1024 USE TS1 FS1
FC 2 FS2   1110111XLL1100XX;//REPEAT 2 USE TS1 FS2
FL TS4     0011HLLXHHH; //REPEAT 1 USE TS4 FS2
FC 1 TS6   11HLLL00000; //REPEAT 1 USE TS6 FS2
FC 1 TS1   0000111111HHHLLLLL;//REPEAT 1 USE TS1 FS2
FC 6 TS7   HHHHLLXXXX11110000;//REPEAT 6 USE TS7 FS2
FM 54 FS3  HHHH11111LLLLLXXXX00;//MATCH REPEAT 54 USE TS7 FS3
FE TS4     HHHLLLLL1010101X10;//REPEAT 1 USE TS4 FS3 THEN GOTO FL
FC 1       1111100101HHHLLLLL1L1L;REPEAT 1 USE TS4 FS3
```



## 5 Pin Connector

PIN OUT

	LOC
DOUT1	J1-65
DOUT2	J1-63
DOUT3	J1-61
DOUT4	J1-59
DOUT5	J1-57
DOUT6	J1-55
DOUT7	J1-53
DOUT8	J1-51
DOUT9	J1-49
DOUT10	J1-47
DOUT11	J1-45
DOUT12	J1-43
DOUT13	J1-41
DOUT14	J1-39
DOUT15	J1-37
DOUT16	J1-35
SENSE1	J1-31
SENSE2	J1-29
SENSE3	J1-27
SENSE4	J1-25
SENSE5	J1-23
SENSE6	J1-21
SENSE7	J1-19
SENSE8	J1-17
SENSE9	J1-15



SENSE10	J1-13
SENSE11	J1-11
SENSE12	J1-9
SENSE13	J1-7
SENSE14	J1-5
SENSE15	J1-3
	LOC
SENSE16	J1-1
GND	2,4,6,10,12,14,16,18,20,22,24, 28,32,34,68,66,62,58,56,54,50 ,48,46,44,42,40,38,36
LSYNC	J1-26
EXTTRIG(low active level trigger)	J1-67



Figure1. PE16 Connector Pin Out

SENSE16	1	35	DOUT16
GND	2	36	GND
SENSE15	3	37	DOUT15
GND	4	38	GND
SENSE14	5	39	DOUT14
GND	6	40	GND
SENSE13	7	41	DOUT13
	8	42	GND
SENSE12	9	43	DOUT12
GND	10	44	GND
SENSE11	11	45	DOUT11
GND	12	46	GND
SENSE10	13	47	DOUT10
GND	14	48	GND
SENSE8	15	49	DOUT9
GND	16	50	GND
SENSE8	17	51	DOUT8
GND	18	52	
SENSE7	19	53	DOUT7
GND	20	54	GND
SENSE6	21	55	DOUT6
GND	22	56	GND
SENSE5	23	57	DOUT5
GND	24	58	GND
SENSE4	25	59	DOUT4
LSYNC	26	60	
SENSE3	27	61	DOUT3
GND	28	62	GND
SENSE2	29	63	DOUT2
	30	64	
SENSE1	31	65	DOUT1
GND	32	66	GND
	33	67	EXTTRIG (low active trigger)
GND	34	68	GND