# LTR介绍

## 背景介绍

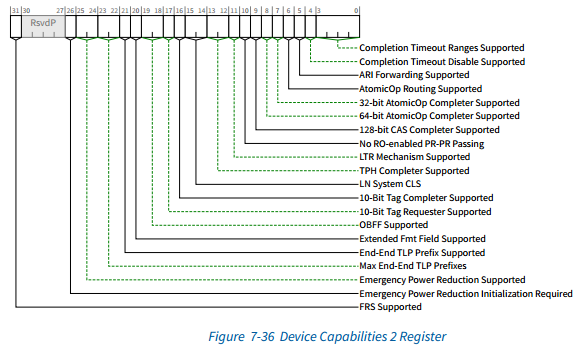
### LTR基本介绍

LTR 是 PCIe 协议中规定的一项可选特性，可用于提高系统的电源管理效率。

通过 LTR消息报文，PCIe 设备可以告知系统它们能容忍的最大响应延迟是多少，只要系统在这个时间之内对 PCIe 设备提出的请求做出响应即可。系统的电源管理策略可以根据 LTR 提供的信息做出优化。在设备需要的时候，系统快速响应设备提出的请求；在设备不需要的时候，系统可以延迟一会再对设备请求做出响应，只要不超出设备 LTR 消息中规定的时间即可。例如，如果系统在低功耗状态时收到设备的请求，那么晚一会再响应这个请求，避免频繁的从低功耗状态中唤醒。

# PCI Express Capability

## Device Capability 2 Register (Offset 24h)



### LTR Mechanism Supported（已实现）

A value of 1b indicates support for the optional Latency Tolerance Reporting (LTR) mechanism. Root Ports, Switches and Endpoints are permitted to implement this capability.

For a Multi-Function Device associated with an Upstream Port, each Function must report the same value for this bit.

For Bridges and other Functions that do not implement this capability, this bit must be hardwired to 0b.

该字段的值为1表示支持可选的LTR机制。允许RP、Switch以及EP实现此功能。对于与上游端口相关联的MFD，每个功能必须报告该bit的相同值。对于不实现此功能的Bridge和其他Function，必须将该bit硬连接到0b。

225已实现，该字段由swithc\_core\_shell模块下的k\_pexconf[`KPEXCONF\_ DEV2\_LTR]配置信号传递到trans模块下的pcie5\_pexreg模块，是k\_pexconf信号的第42bit。当该bit置1时，该信号表示支持LTR机制。

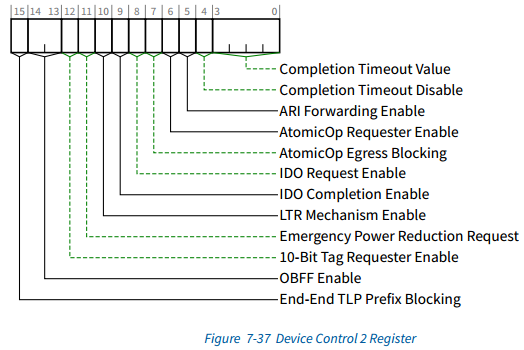
在pcie5\_pexreg模块的第1202行，该k\_pexconf[`KPEXCONF\_DEV2\_LTR]信号会和其他字段信号进行拼接并赋值给pex24信号并输出出去。pex24信号输出的LTR Mechanism Supported字段没有被其他模块所使用。

在pcie5\_pexreg模块的第1288行，k\_pexconf[`KPEXCONF\_DEV2\_LTR]用作ltr\_enable信号的判断条件，其中ltr\_enable信号表示LTR Mechanism Enable字段，ltr\_enable这部分在下面小节进行详细介绍。如果k\_pexconf[`KPEXCONF \_DEV2\_LTR]=1’b0，那么ltr\_enable=1’b0，表示如果不支持LTR机制，那么就不会启用LTR机制。

在pcie5\_extcap模块的第206行，k\_pexconf[`KPEXCONF\_DEV2\_LTR]信号用作k\_ltr信号的判断条件，当k\_pexconf[`KPEXCONF\_DEV2\_LTR]=1’b1 && FUNC\_ID=0 && (k\_ep=1 || k\_devtype==`CORE\_TYPE\_SWITCH\_UPSTREAM)时，k\_ltr=1’b1，表示当支持LTR机制且Function ID为0，并且为EP端口或Switch的上游端口时，k\_ltr置1。该k\_ltr信号用于LTR Extended Capability中Max No-Snoop和Max Snoop字段的判断条件，如果k\_ltr置0，那么Max No\_Snoop字段和Max Snoop字段都会清零。LTR Extended Capability的各个字段拼接分别赋值给ltr00和ltr04信号时，也会根据k\_ltr进行判断，只有当k\_ltr=1’b1时，才会将拼接后的字段赋值给ltr00和ltr04信号。

在hsio\_top\_pipe模块中，会用第0组的k\_pexconf[`KPEXCONF\_DEV2\_LTR]。如果第0组的k\_pexconf[`KPEXCONF\_DEV2\_LTR]置1，则表示upstream port支持LTR机制。该信号会在hsio\_ltr\_compare模块例化时候赋值给up\_ltr\_support信号。当up\_ltr\_support信号置0，也就是如果不支持LTR机制时，那么hsio\_ltr\_compare模块内部的相关信号都会初始化和清零。

## Device Control 2 Register (Offset 28h)



### LTR Mechanism Enable（已实现）

When Set to 1b, this bit enables Upstream Ports to send LTR messages and Downstream Ports to process LTR Messages.

For a Multi-Function Device associated with an Upstream Port of a device that implements LTR, the bit in Function 0 is RW, and only Function 0 controls the component’s Link behavior. In all other Functions of that device, this bit is RsvdP.

Functions that do not implement the LTR mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b.

For Downstream Ports, this bit must be reset to the default value if the Port goes to DL\_Down status.

当设置为1b时，该位允许上游端口发送LTR消息，下游端口处理LTR消息。

对于与实现LTR的设备的上游端口相关联的多功能设备，功能0中该bit位为RW，只有功能0控制该组件的Link行为。在该设备的所有其他功能中，该bit位是RsvdP。

允许不实现LTR机制的功能将此位硬连接到0b。默认值为0b。

对于下行端口，如果端口状态变为DL\_Down，则该位必须重置为默认值。

225已实现，该LTR Mechanism Enable字段表示是否启用LTR机制。该字段由pcie5\_pexreg模块中第1189行的ltr\_enable信号表示。前面小节中提到当不支持LTR机制时，ltr\_enable置0，或者FUNC\_ID不为0时，ltr\_enable同样也会置0；或者DSP进入到DL\_Down状态，那么该字段也会置0。如果前面条件都不满足的话，那么在cfg\_addr=12’h0A8时，将cfg\_wdata[10]赋值给ltr\_enable信号，cfg\_addr和cfg\_wdata是APB接口的配置信号。ltr\_enable信号会和该寄存器的其他字段进行拼接后赋值给pex28信号并发送出去。

在pcie5\_conf模块中pex28信号会将LTR Mechanism Enable字段赋值给tl\_pex\_enable[`PEXEN\_LTR\_EN]信号。该字段的tl\_pex\_enable信号会在pcie5\_cfgrw模块中使用。在pcie5\_cfgrw模块第1136行中，该字段信号打拍赋值给ltr\_en\_r寄存。在第1138行中，该字段信号tl\_pex\_enable[`PEXEN\_LTR\_EN]= 1’b1，并且tl\_report\_latency[`LAT\_REPORT]或ltr\_en\_r=1’b0时，req\_msg\_ltr置1；表示检测到LTR enable的上升沿或者在LTR启用的情况下tl\_report\_latency [`LAT\_REPORT]置1（表示downstream port更新了LTR），那么将请求LTR Message报文的发送。在pcie5\_cfgrw模块第1161行中，该字段tl\_pex\_enable [`PEXEN\_LTR\_EN]置0，并且ltr\_en\_r置1以及ltr\_requitement置1时，req\_msg\_ltr\_noreq置1；表示在检测到LTR enable的下降沿并且接收到的LTR的需求位为1时，将发起requirement为0的LTR消息报文发送请求（这个其实对应PLDA手册LTR的时序图，最后一个带有requirement置1的LTR消息报文发送完之后，发起requirement置0的LTR消息报文的发送，详见LTR协议与PLDA翻译文档的6.2.16小节）。另外在legacy电源状态从D0状态进入到非D0状态中，并且在requirement置1以及LTR enable置1时，也会起requirement置0的LTR消息报文的发送。

同时在pcie5\_conf模块中pex28信号会赋值给tl\_cfg\_regs信号并继续向上传输，在hsio\_top\_pipe模块中会将该字段信号提取出来赋值给ltr\_enable信号中。该ltr\_enable信号会用于ltr\_compare模块和decoder\_dn模块中。

在ltr\_compare模块中，ltr\_enable信号会在该模块内部产生一个下降沿检测脉冲，该脉冲会用作LTR更新事件的判断，如果发生该事件就需要向上有端口发送新的LTR消息（参考LTR协议与PLDA翻译文档的6.2.16.1小节）。

在decoder\_dn模块中，ltr\_enable信号用作ltr\_value的判断条件，ltr\_value为LTR message包头中的第4 DW的内容。当ltr\_enable置0时，ltr\_value清零；当ltr\_enable置1时，ltr\_value保持不变。这里表示当LTR机制关闭时，ltr的scale、value以及requirement都会被清除。同时代码中的注释是当下游端口的LTR enable被清除，那么各自port接收到的LTR消息报文就被视为infinity。

LTR Mechanism Enable字段只在上述模块中有所使用，在其他模块中并没有被使用到。

# LTR Extended Capability

The PCI Express Latency Tolerance Reporting (LTR) Extended Capability is an optional Extended Capability that allows software to provide platform latency information to components with Upstream Ports (Endpoints and Switches), and is required for Switch Upstream Ports and Endpoints if the Function supports the LTR mechanism. It is not applicable to Root Ports, Bridges, or Switch Downstream Ports.

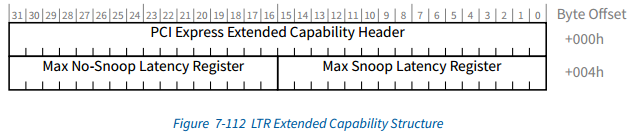
For a Multi-Function Device associated with the Upstream Port of a component that implements the LTR mechanism, this Capability structure must be implemented only in Function 0, and must control the component’s Link behavior on behalf of all the Functions of the Device.

RCiEPs implemented as Multi-Function Devices are permitted to implement this Capability structure in more than one Function of the Multi-Function Device.

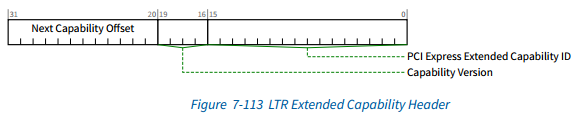
PCI Express延迟容忍报告(LTR)扩展功能是一种可选的扩展功能，允许软件向具有上游端口(EP和Switch)的组件提供平台延迟信息，如果功能支持LTR机制，则Switch的上游端口和EP需要该功能。不适用于RP、Bridge、Switch下游端口。

对于与实现LTR机制的组件的上游端口相关联的多功能设备，此Capability结构必须仅在功能0中实现，并且必须代表设备的所有功能控制组件的Link行为。

作为多功能设备实现的RCiEP允许在多功能设备的多个功能中实现此能力结构。



## LTR Extended Capability Header(Offset 00h)



### PCI Express Extended Capability ID（已实现）

This field is a PCI-SIG defined ID number that indicates the nature and format of the Extended Capability.

PCI Express Extended Capability for the LTR Extended Capability is 0018h

该字段是PCI-SIG定义的标识号，表示扩展能力的性质和格式。LTR的PCI Express扩展能力为0018h。

225已实现，具体在pcie5\_extcap模块中，366行中固定赋值为16’h0018，并传递给信号ltr00中，但是后续向其他模块传输过程中并没有再使用。

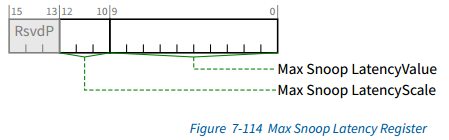
### Capability Version（已实现）

This field is a PCI-SIG defined version number that indicates the version of the Capability structure present. Must be 1h for this version of the specification.

该字段是PCI-SIG定义的版本号，指示当前Capability结构的版本。这个版本的规格必须是1h。

225已实现，具体在pcie5\_extcap模块中，366行中固定赋值为4’h1，指示当前的版本为第1版本。该域段信号传输给信号ltr00之后没有再使用过。

## Max Snoop Latency Register (Offset 04h)



### Max Snoop LatencyValue（已实现）

Along with the Max Snoop LatencyScale field, this register specifies the maximum snoop latency that a device is permitted to request. Software should set this to the platform’s maximum supported latency or less. It is strongly recommended that any updates to this field are reflected in LTR Message(s) sent by the device within 1ms. The default value for this field is 00 0000 0000b.

与Max Snoop LatencyScale字段一起，该寄存器指定设备允许请求的最大snoop延迟。软件应该将其设置为平台支持的最大延迟或更小。强烈建议对该字段的任何更新都反映在设备在1ms内发送的LTR Message(s)中。

该字段的默认值为00 0000 0000b。

225已实现但未使用，该字段的配置在pcie5\_extcap模块第355行中进行实现，当cfg\_addr=12’h10C时，将cfg\_data[7:0]赋值给ltr\_maxsnoop\_r[7:0]信号，该信号为Max Snoop LatencyValue字段。该字段和其他字段拼接后赋值给ltr04信号，ltr04信号再和ltr00信号拼接后赋值给ltr\_cap信号。ltr\_cap信号中该字段输出出去之后没有被其他模块所使用。

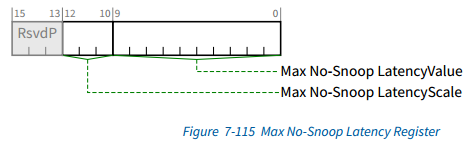
### Max Snoop LatencyScale（已实现）

This register provides a scale for the value contained within the Max Snoop LatencyValue field. Encoding is the same as the LatencyScale fields in the LTR Message. See Section 6.18. It is strongly recommended that any updates to this field are reflected in LTR Message(s) sent by the device within 1 ms. The default value for this field is 000b. Hardware operation is undefined if software writes a Not Permitted value to this field.

该寄存器为Max Snoop LatencyValue字段中包含的值提供了一个刻度。编码与LTR消息中的LatencyScale字段相同。参见6.18节。强烈建议对该字段的任何更新都反映在设备在1ms内发送的LTR Message(s)中。该字段的默认值是000b。如果软件向该字段写入不允许值，则硬件操作未定义。

225已实现但未使用，该字段的配置在pcie5\_extcap模块第357行中进行实现，当cfg\_addr=12’h10C时，将cfg\_data[12:8]赋值给ltr\_maxsnoop\_r[12:8]信号，该信号为Max Snoop LatencyScale字段。该字段和其他字段拼接后赋值给ltr04信号，ltr04信号再和ltr00信号拼接后赋值给ltr\_cap信号。ltr\_cap信号中该字段输出出去之后没有被其他模块所使用。

## Max No-Snoop Latency Register (Offset 06h)



### Max No-Snoop LatencyValue（已实现）

Along with the Max No-Snoop LatencyScale field, this register specifies the maximum no-snoop latency that a device is permitted to request. Software should set this to the platform’s maximum supported latency or less. It is strongly recommended that any updates to this field are reflected in LTR Message(s) sent by the device within 1ms. The default value for this field is 00 0000 0000b

与Max No-Snoop LatencyScale字段一起，该寄存器指定设备允许请求的最大No-Snoop延迟。软件应该将其设置为平台支持的最大延迟或更小。强烈建议对该字段的任何更新都反映在设备在1ms内发送的LTR Message(s)中。该字段的默认值为00 0000 0000b

225已实现但未使用，该字段的配置在pcie5\_extcap模块第359行中进行实现，当cfg\_addr=12’h10C时，将cfg\_data[23:16]赋值给ltr\_maxnosnoop\_r[7:0]信号，该信号为Max No-Snoop LatencyValue字段。该字段和其他字段拼接后赋值给ltr04信号，ltr04信号再和ltr00信号拼接后赋值给ltr\_cap信号。ltr\_cap信号中该字段输出出去之后没有被其他模块所使用。

### Max No-Snoop LatencyScale（已实现）

This register provides a scale for the value contained within the Max No-Snoop LatencyValue field. Encoding is the same as the LatencyScale fields in the LTR Message. See Section 6.18. It is strongly recommended that any updates to this field are reflected in LTR Message(s) sent by the device within 1 ms. The default value for this field is 000b. Hardware operation is undefined if software writes a Not Permitted value to this field.

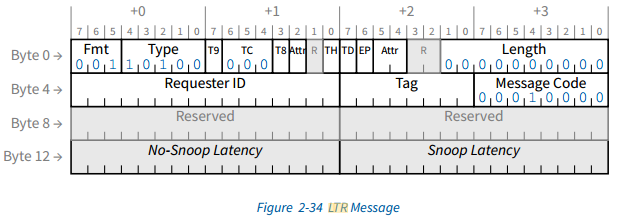
该寄存器为Max No-Snoop LatencyValue字段中包含的值提供了一个刻度。编码与LTR消息中的LatencyScale字段相同。参见6.18节。强烈建议对该字段的任何更新都反映在设备在1ms内发送的LTR Message(s)中。该字段的默认值是000b。如果软件向该字段写入不允许值，则硬件操作未定义。

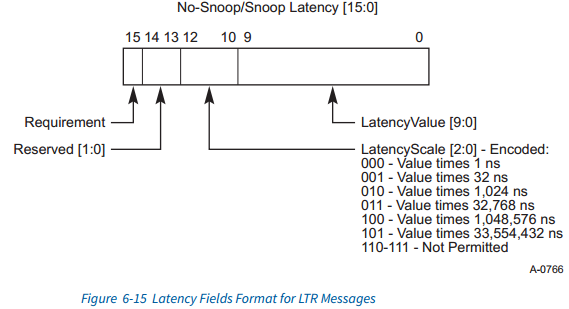
225已实现但未使用，该字段的配置在pcie5\_extcap模块第359行中进行实现，当cfg\_addr=12’h10C时，将cfg\_data[28:24]赋值给ltr\_maxnosnoop\_r[12:8]信号，该信号为Max No-Snoop LatencyScale字段。该字段和其他字段拼接后赋值给ltr04信号，ltr04信号再和ltr00信号拼接后赋值给ltr\_cap信号。ltr\_cap信号中该字段输出出去之后没有被其他模块所使用。

# LTR在225中实现过程

LTR message报文的内容在switch内部是从DSP向USP传递的。

当DSP接收到LTR message报文之后会向上传输到decoder\_dn模块中。在该模块中会对LTR message进行解析，主要是byte12~15的内容：no-snoop latency和snoop latency。snoop latency和no-snoop latency的字段主要分为requirement（1bit）、reserved（2bit）、latencyscale（3bit）、latencyvalue（10bit）。decoder\_dn模块解析出LTR message之后，在FMT\_TYPE\_MES\_ILTR状态下将该部分内容赋值给ltr\_value并发送出去，ltr\_value位宽为28bit。另外ltr\_update信号在FMT\_TYPE\_MES\_ILTR状态下如果接收到的消息报文为LTR消息，那么该信号固定置1并输出出去，ltr\_update在其他条件下置0。





第一代225中有1个USP端口和31个DSP端口，因此分别对应1个decoder\_up模块和31个decoder\_dn模块，并且decoder\_dn模块将解析出的ltr\_value信号以及ltr\_update信号输出到hsio\_top\_pipe模块中，在该模块中解析出的ltr\_value信号将按照上图6-15的字段格式赋值给dn\_ltr\_value信号，其信号格式如下表所示，该dn\_ltr\_value信号位宽为33bit：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 32 | 31 | 30:29 | 28:16 | 15 | 14:13 | 12:0 |
| ltr\_update | ltr\_value  [27] | 2’b0 | ltr\_value[26:14] | ltr\_value  [13] | 2’b0 | ltr\_value[12:0] |

每个DSP的dn\_ltr\_value信号按照端口对应顺序拼接后赋值给dn\_latch\_ltr\_ value信号，由于USP对应第0号端口，因此dn\_latch\_ltr\_value的低33bit始终为0。并且dn\_latch\_ltr\_value信号发送到hsio\_ltr\_compare模块中。

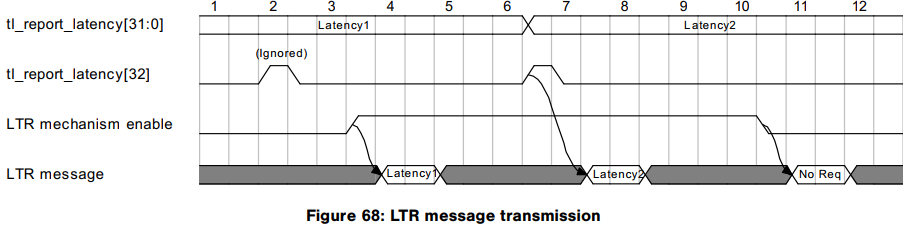
在hsio\_ltr\_compare模块中会逐次将31个DSP的dn\_latch\_ltr\_value中的snoop latencyvalue和no-snoop latencyvalue按照latencyscale对应的时间映射关系将latencyvalue转换为真实的时间值，转换为真实的时间之后再逐次将DSP的latency真实时间值进行比较，选择出最小的latency，并且只有requirement置1的latency才会参与比较（这里以snoop latency为例，no-snoop latency计算比较的流程和snoop latency的流程相同）。

同时hsio\_ltr\_compare模块中还有一个表征switch内部latency的user\_ltr\_ switch\_value信号，该信号和LTR message中的byte12~15的字段定义一致。user\_ltr\_switch\_value中的latencyvalue值也会根据latencyscale的时间映射为真是的时间值，而当前switch\_core\_shell模块的user\_ltr\_switch\_value始终赋值为32’b0。(user\_ltr\_switch\_value的的说明见LTR协议与PLDA翻译文档)

31个DSP最小的latency和Switch中的latency进行比较，如果DSP的latency大于Switch的latency时，会将DSP的latency减去Switch的latency之后赋值给ltr\_sp\_update\_real\_value\_w信号，位宽为35bit，这个信号为真实的时间值；否则DSP的latency小于Switch的latency时，ltr\_sp\_update\_real\_value\_w信号固定为35’d0。该值将根据实际时间值进行编码，转换为LTR message中snoop latency字段的latencyscale和latencyvalue，得到的新的snoop latency、no-snoop latency将会进拼接赋值给up\_tl\_report\_latency输出出去，该up\_tl\_report\_latency信号会输出到USP的trans模块下的pcie5\_cfgrw模块和pcie5\_extcap模块中。

在pcie5\_cfgrw模块存在两个状态C2T\_STATE\_LTR和C2T\_STATE\_LTR\_ NOREQ。在C2T\_STATE\_LTR状态是将ltr\_compare模块发送的up\_tl\_report\_ latency进行拼接，按照LTR message的格式进行重新组包，并且up\_tl\_report\_ latency[31:0]是位于LTR message的byte12~15的位置。这个LTR message报文将会交给txtl模块并最终发送出去。在C2T\_STATE\_LTR\_NOREQ状态是发送的一个没有up\_tl\_report\_latency[31:0]的LTR message报文。这部分是对应PLDA手册的6.2.16章节，在LTR机制启用的情况下，如果up\_tl\_report\_latency[32]置1，并且这个时候LTR message的requirement是置1的，那么在关闭LTR机制之后，下一个LTR message是一个requirement置0的消息报文，也就是no-req。

这个模块中只有USP才会进入到C2T\_STATE\_LTR和C2T\_STATE\_LTR\_ NOREQ，DSP的配置模块中并不会进入这两个状态中。



在pcie5\_extcap模块中，USP端口的up\_tl\_report\_latency传递过程中赋值给tl\_report\_latency，而DSP端口的tl\_report\_latency来自decoder\_dn模块从LTR message提取出来的snoop latency和no-snoop latency。tl\_report\_latency中的latencyvalue字段根据latencyscale的时间映射关系转换为真实时间值，包括snoop和no-snoop。L1 PM Substates Control 1 Register中的LTR\_L1.2\_THRESHOLD \_Value字段也会根据LTR\_L1.2\_THRESHOLD\_ Scale字段的时间映射关系转换为真实时间值（这里的时间映射关系和LTR message中的一致）。当requirement（snoop与no-snoop的requirement）置0或者tl\_report\_latency中的真实latency大于LTR\_L1.2\_THRESHOLD\_Value字段的真实latency，那么asl12\_latency\_ok信号就会置1，否则置0。这个信号会和L1 PM Substates Control 1 Register中的ASPM L1.2 Enable字段共同决定L1 PM状态的启用。

另外在hsio\_ltr\_compare模块中会涉及到更新并自动传输LTR message，主要由三种事件所决定的：

* 下游端口进入DL\_Down状态，或者；
* 下游端口的LTR enable位被清除，或者；
* 下游端口/功能改变延迟。

前面两种是由PCIe寄存器的字段：Link Status Register中的Data Link Layer Link Active字段和Device Control 2 Register中的LTR Mechanism Enable字段，最后一个事件由decoder\_dn发送过来的ltr\_update所决定的，当该信号拉高，就认为是该事件发生。只要上面三个事件的其中之一发生，那么就会更新并自动传输LTR message。而如果当前聚合的LTR延迟结果刚比较完成，在规定的时间段内（例如500μs）又有新的更新事件发生，那么就会重新比较LTR的延迟；否则超过规定的时间段并且之前发送的LTR延迟和新聚合的LTR延迟不相等之后，USP便会将聚合的LTR延迟值向上发送出去。

在hsio\_ltr\_compare模块中按照PLDA手册的usr\_ltr\_timer\_sel定义，设置了计数器进行计时，当达到规定的计时限制时，便可以发送新的LTR message。协议强烈建议EP在任何500μs时间段内发送不超过两个LTR消息，但是如果在500μs的时间周期内收到两个以上的LTR消息，则下游端口不能产生错误，并且必须正确处理所有的LTR消息，而不管它们之间的时间间隔。因此hsio\_ltr\_ compare模块中在没有达到指定的时间时，如果有更新事件发生，那么也会重新比较接收到的LTR message并发送的（参考LTR协议与PLDA翻译文档）。