PCIe® 3.0 Controller Case Study

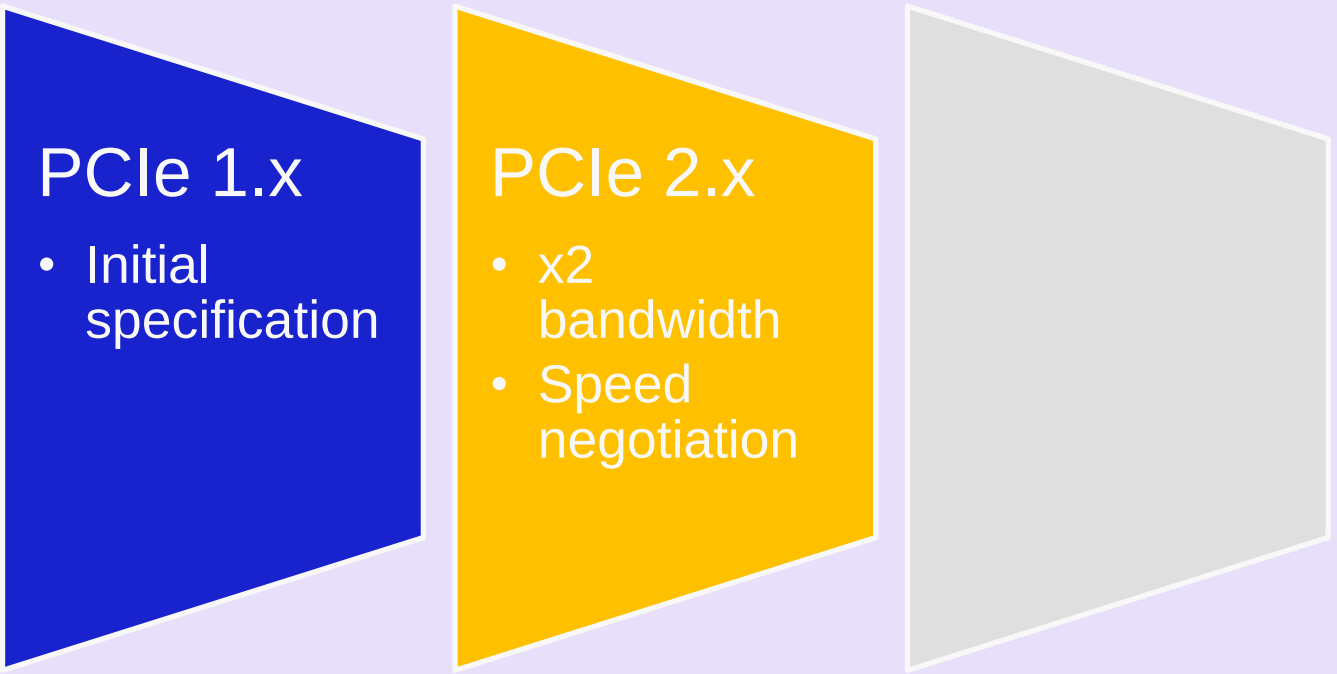
Philippe Legros
PCI Express® Design Team Manager
PLDA

Disclaimer

Presentation Disclaimer: All opinions, judgments, recommendations, etc. that are presented herein are the opinions of the presenter of the material and do not necessarily reflect the opinions of the PCI-SIG®.

The Challenge of PCIe® 3.0

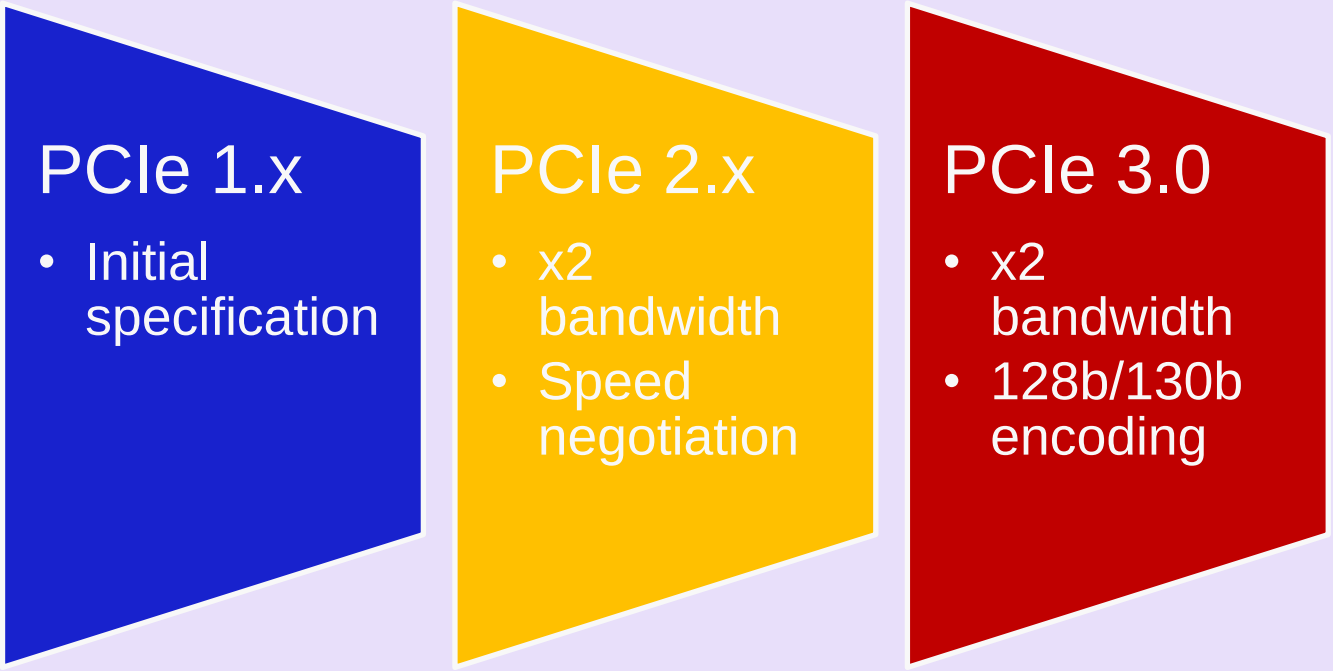
- PCIe 2.x introduced a x2.0 bandwidth increase, and speed negotiation



- Challenge mostly on electrical part, but existing PCIe logic could be adapted easily

The Challenge of PCIe 3.0

- PCI Express® 3.0 introduces x2.0 bandwidth increase, and 128b/130b encoding



- Major changes are required in PHY layer

Why a New Architecture?

- 128b/130b encoding has a major impact in PHY layer and many parts must be significantly modified:
  - ✓ Use of blocks completely changes the way packets & ordered sets are processed & transmitted
  - ✓ Scrambling is different and affects data differently
  - ✓ Different and new ordered sets (SKIP, SDS, etc.)
  - ✓ Equalization affects LTSSM & training sets

Why a New Architecture?

- Existing designs have often been defined for PCIe 1.x, then adapted for PCIe 2.x. Those technical choices are often no longer optimal with another x2.0 throughput increase

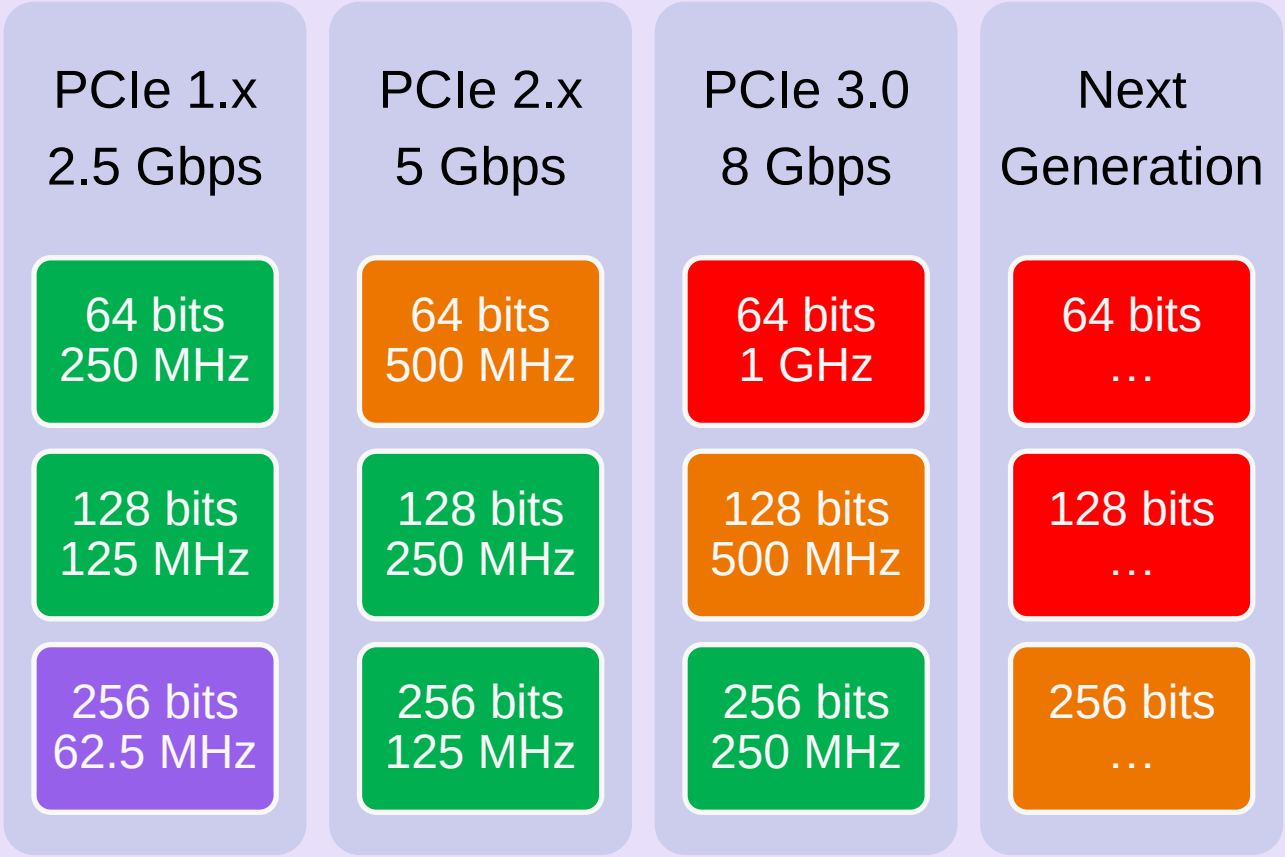
- Also a good opportunity to use experience in PCIe to design a more powerful architecture

Selecting Datapath

- Datapath width is an important choice because it strongly affects:
  - ✓ Design effort
  - ✓ Device scalability
  - ✓ Ease of use for end-user
  - ✓ Possible target devices & process
  - ✓ Cost (gate count)

Selecting Datapath

- Example datapaths for a x8 device:



Selecting Datapath

- Drawbacks of a small datapath (32/64-bit):
  - ✓ More complex in some situations
    - 3..4 clock cycles (32-bit) or 2 clock cycles (64-bit) required to get the complete header of a TLP
    - Header checking done over several clock cycles
  - ✓ Not suitable for high-throughput devices
  - ✓ Requires a faster clock than a large datapath for the same throughput: more power consumed and more expensive device/process required

Selecting Datapath

- Benefits of a large datapath (128/256-bit):
  - ✓ Requires a slower clock than a small datapath for the same throughput: less power consumed
  - ✓ Can be implemented in slower and cheaper devices & process
  - ✓ Suitable for high-throughput devices (PCIe 2.x/PCIe 3.0)

Selecting Datapath

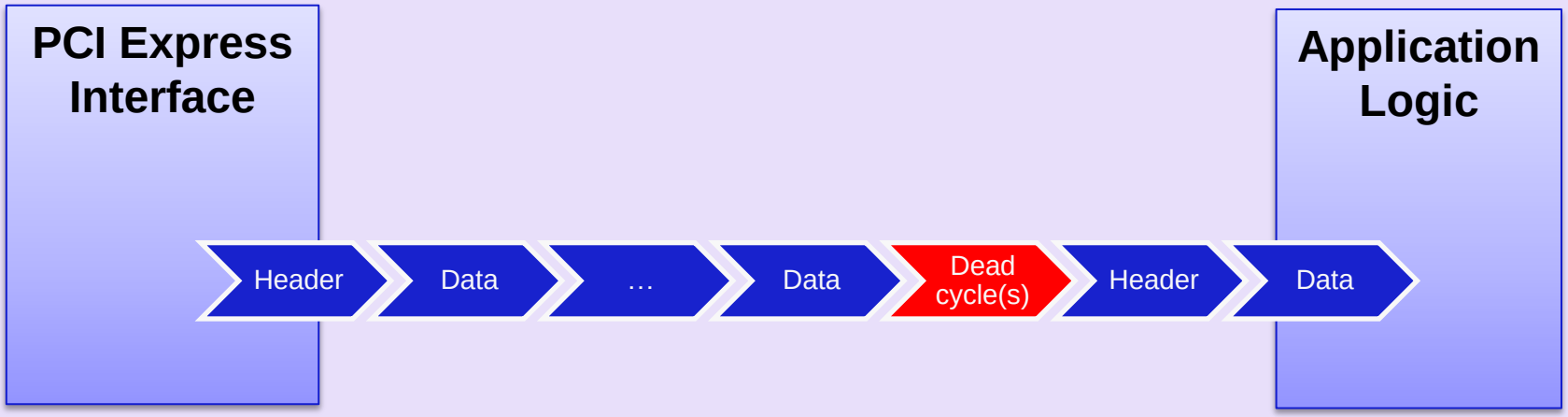
- Drawbacks of a large datapath (128/256-bit):
  - ✓ Generally more complex to design: more data alignment & corner cases to handle
    - For example: up to 4 DLLPs (256-bit) can be received on a single clock cycle
  - ✓ Larger gate count
  - ✓ More difficult to interface to legacy peripherals

Datapath Effect on Device

- Note that datapath size can adversely affect a device:
  - ✓ Example 1: throughput can decrease sharply as datapath size increases if application logic is not designed carefully
  - ✓ Example 2: datapath frequency can have a huge impact on power consumption

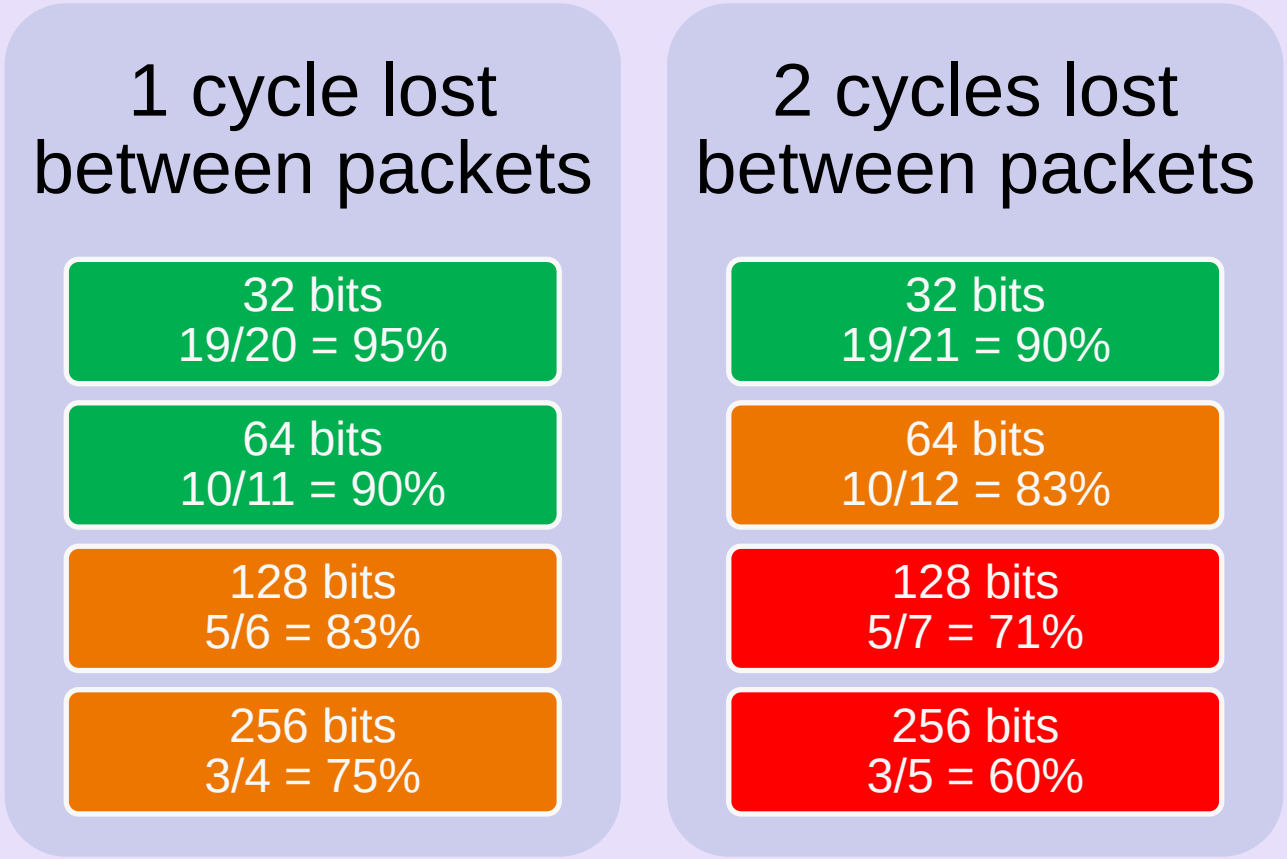
Datapath Effect on Device

- Application logic often inserts “dead cycles” in order to process incoming packets more easily:



Datapath Effect on Device

- Bandwidth efficiency on a device receiving 64-byte completions (76 bytes with header):

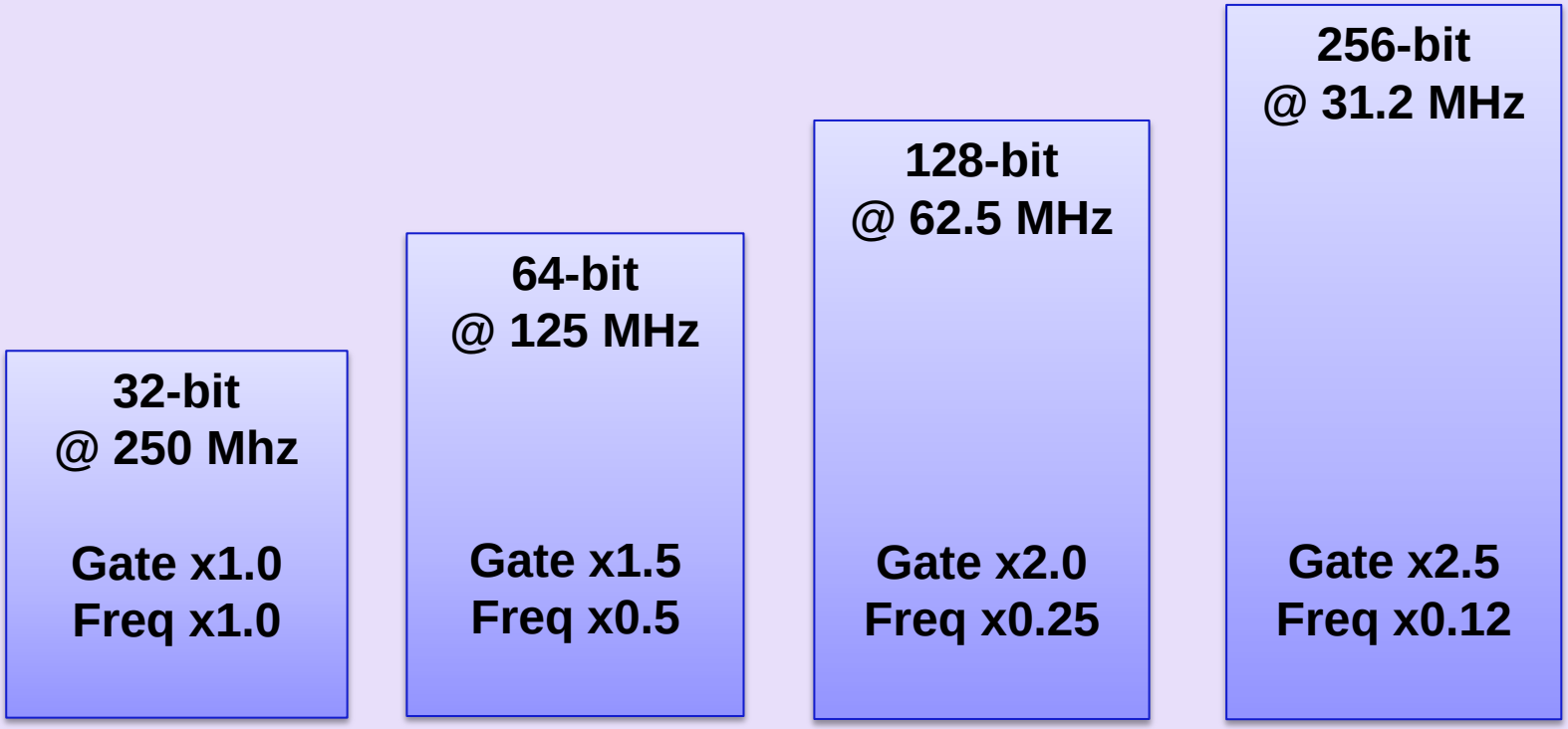


Datapath Effect on Device

- Note that datapath size can adversely affect a device:
  - ✓ Example 1: throughput can decrease sharply as datapath size increases if application logic is not designed carefully
  - ✓ Example 2: datapath frequency can have a huge impact on power consumption

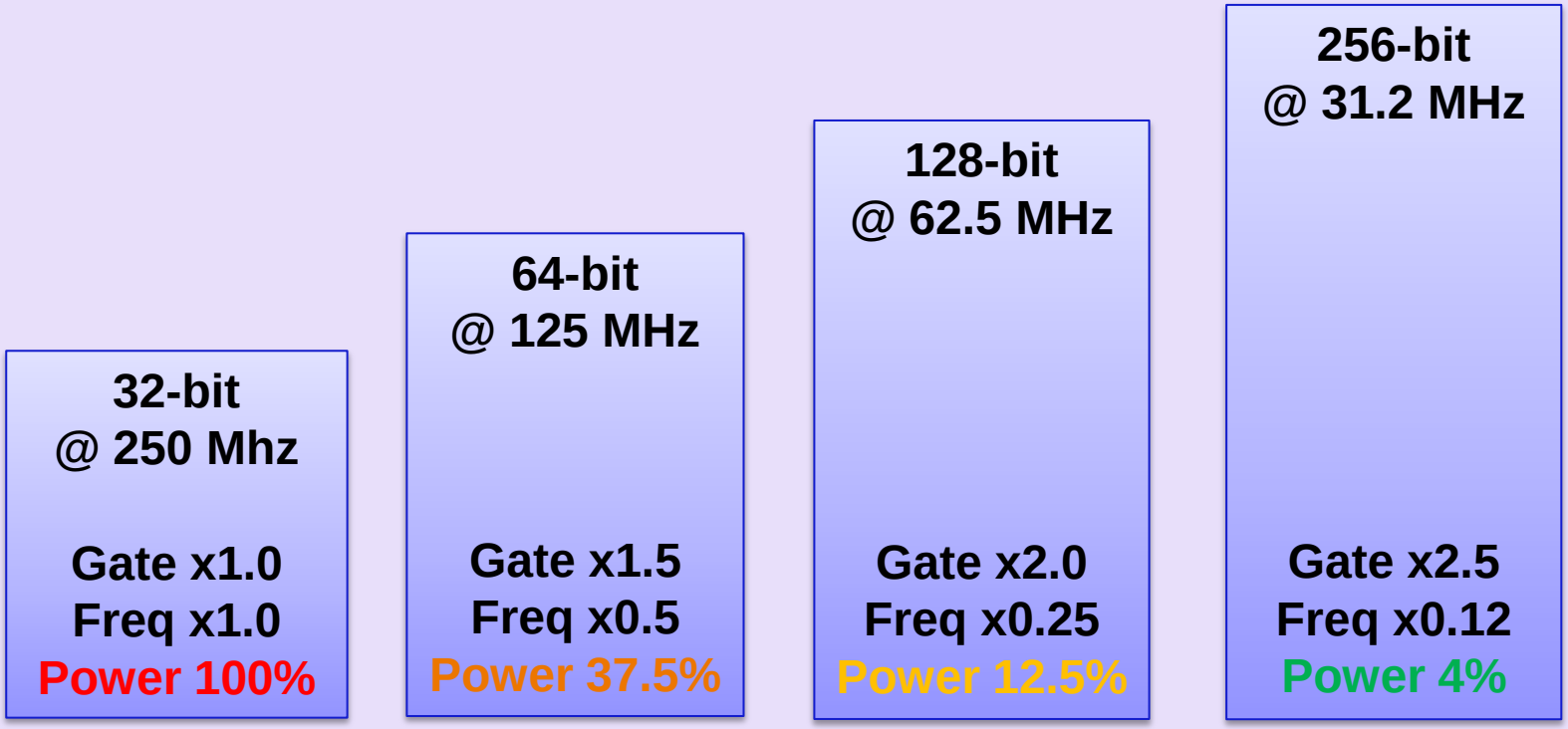
Datapath Effect on Device

- Example datapaths for a x4 PCIe 1.x device:



Datapath Effect on Device

- Power consumption is related to Gate x Freq^2:



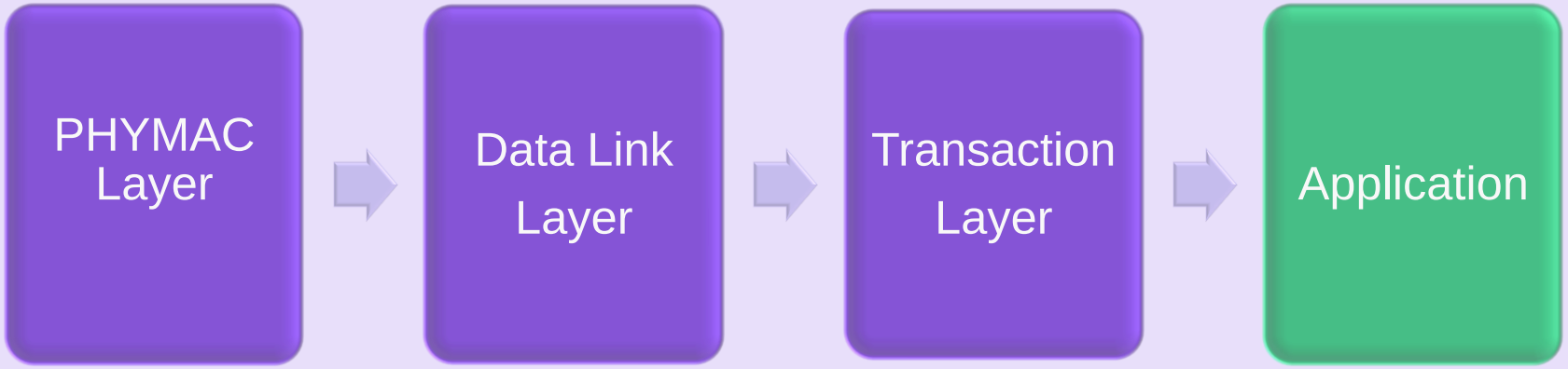
x25 factor !

Datapath Effect on Device

- So this shows that:
  - ✓ All parts of the device must be designed carefully to avoid performance degradation
  - ✓ Datapath must be chosen according to number of lanes and PCIe generation in order to:
    - sustain throughput at a reasonable frequency
    - match gate count & power requirements

Clock Domain Crossing

- PCIe logic typically runs at PIPE interface clock frequency. Clock Domain Crossing (CDC) allows part of this logic to run at an application specific clock frequency.



Clock Domain Crossing

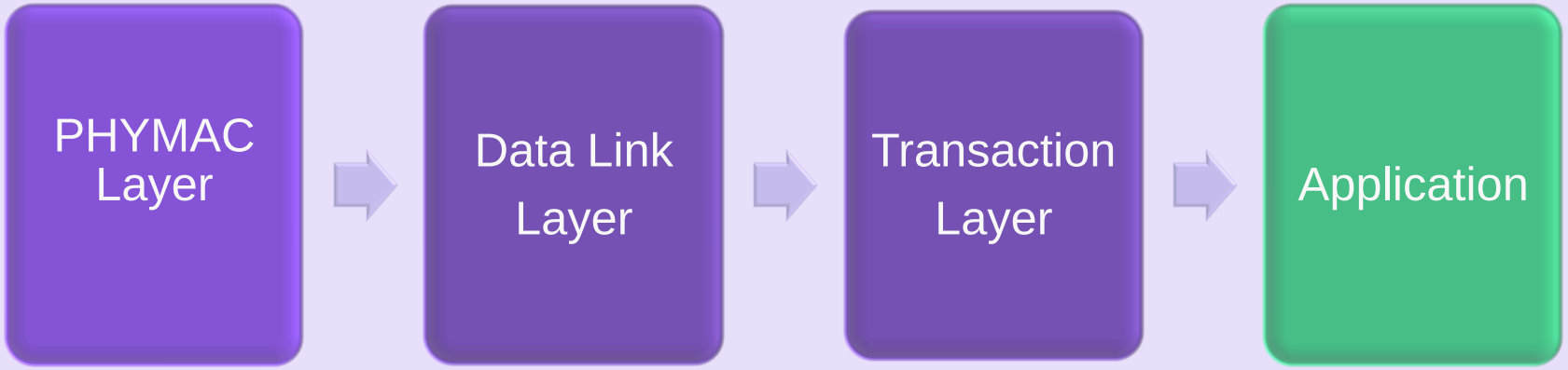
- Benefits of Clock Domain Crossing:
  - ✓ Easier integration with application logic: application front-end does not have to run at same frequency as PCIe logic
  - ✓ Application clock can be lower than PCIe clock, this can ease implementation on slow technologies
  - ✓ Application clock can be reduced or even turned-off to save power (depending on implementation)

Clock Domain Crossing

- Clock Domain Crossing location is an important choice because:
  - ✓ There can be lots of signals to re-synchronize
    - This can consume logic and add latency
  - ✓ There can be required relationships between PIPE clock and application clock
    - This can limit usability of CDC
  - ✓ There can be side effects inside PCIe logic

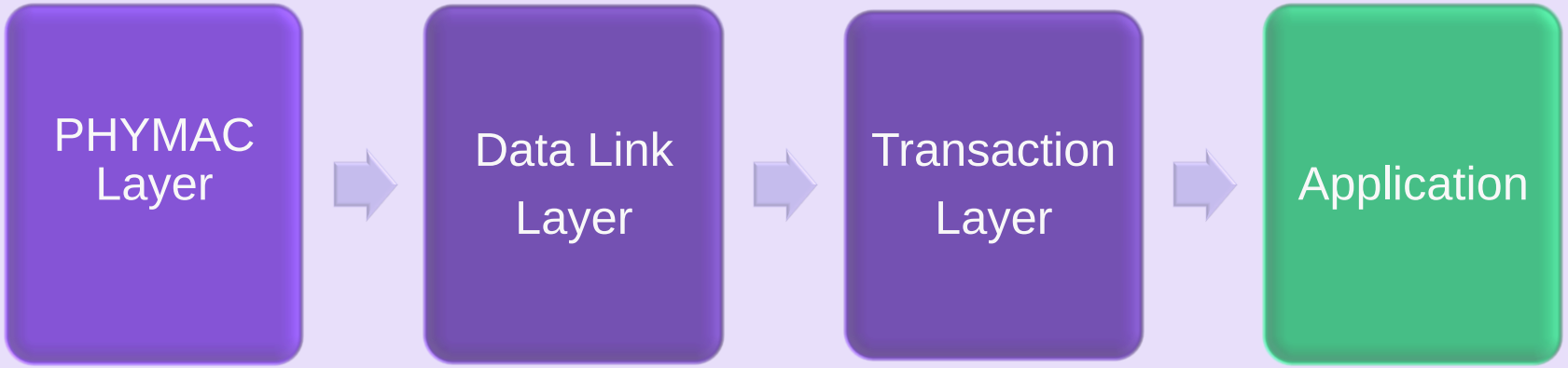
Clock Domain Crossing

- Solution #1: Change clock frequency between Transaction layer & application:



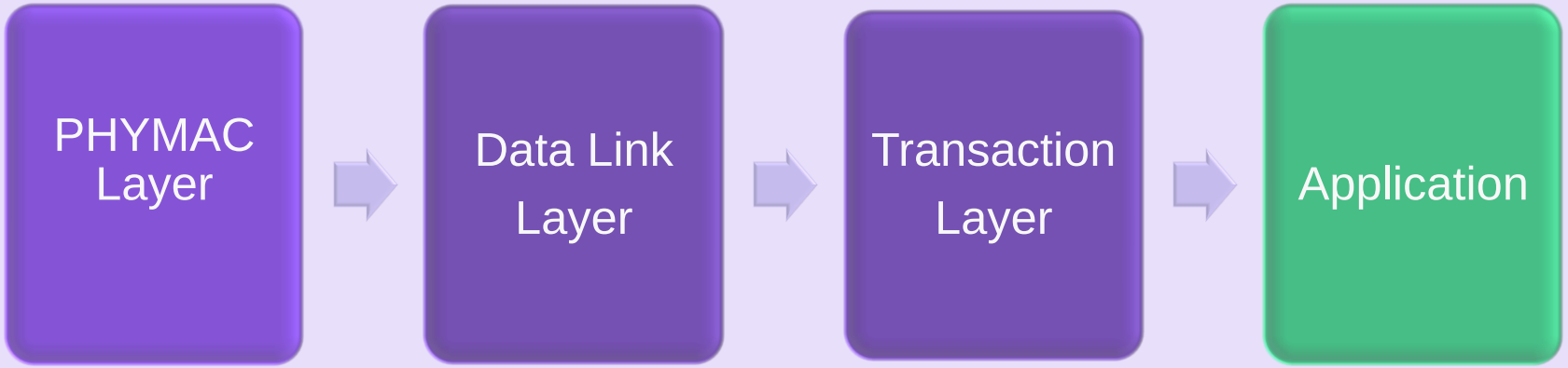
Clock Domain Crossing

- Benefits:
  - ✓ Simple to implement and there are no side-effects on PCIe logic
  - ✓ Application clock is fully independent of PIPE clock and can be freely reduced to save power



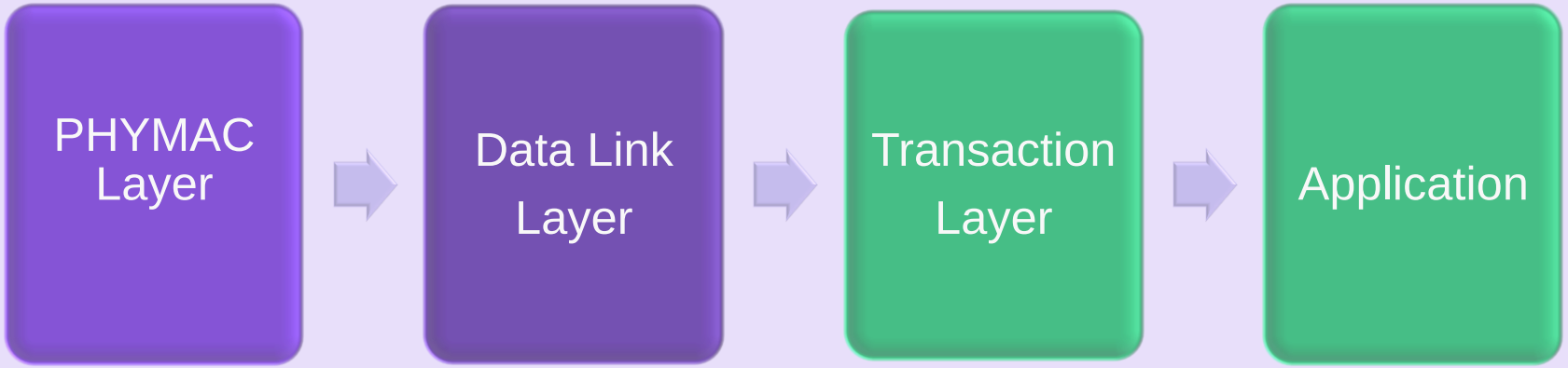
Clock Domain Crossing

- Drawbacks:
  - ✓ All receive/transmit datapath must be buffered with FIFOs or memories (consumes gates & adds latency)
  - ✓ All PCIe logic always runs at full PIPE frequency so this solution allows no power savings on PCIe logic



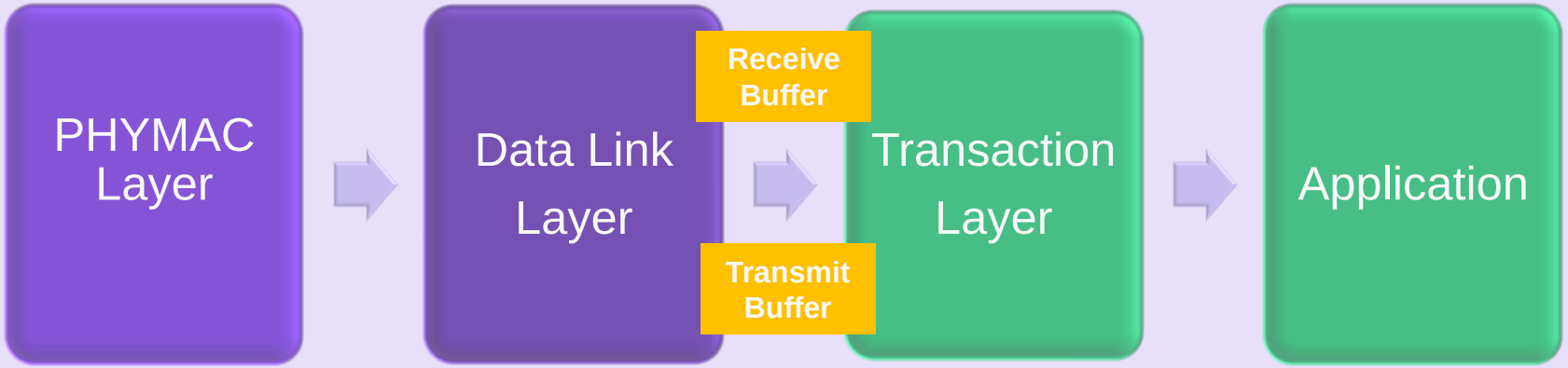
Clock Domain Crossing

- Solution #2: Change clock frequency between Data Link & Transaction layers:



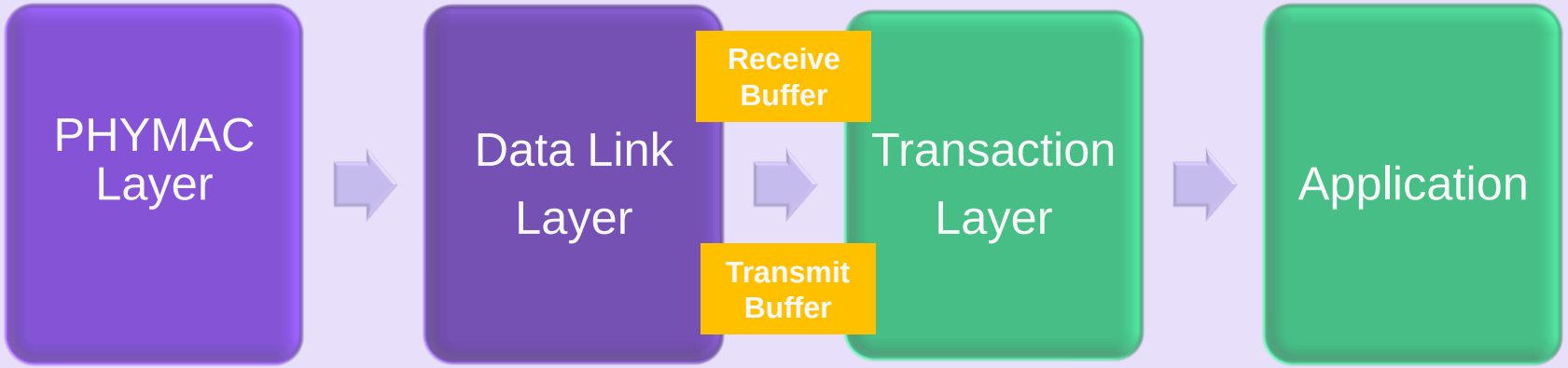
Clock Domain Crossing

- Benefits:
  - ✓ Application clock is fully independent of PIPE clock and can be freely reduced to save power
  - ✓ Receive/transmit datapaths can be buffered through receive/transmit buffers (saves gates)



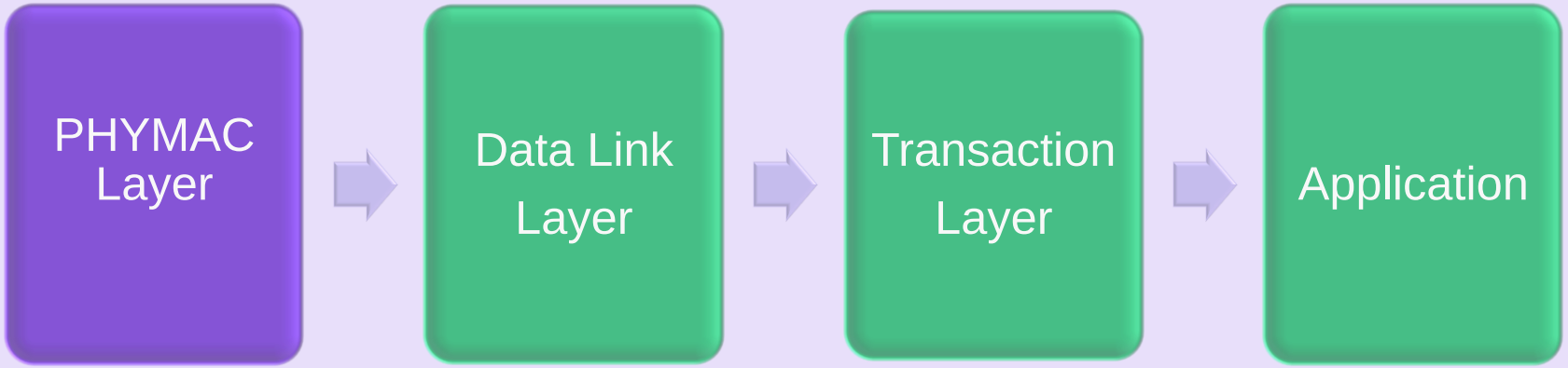
Clock Domain Crossing

- Drawbacks:
  - ✓ A part of PCIe logic always runs at full PCIe speed: this limits power savings
  - ✓ Potential side effects
    - Flow control overflow cannot be detected accurately



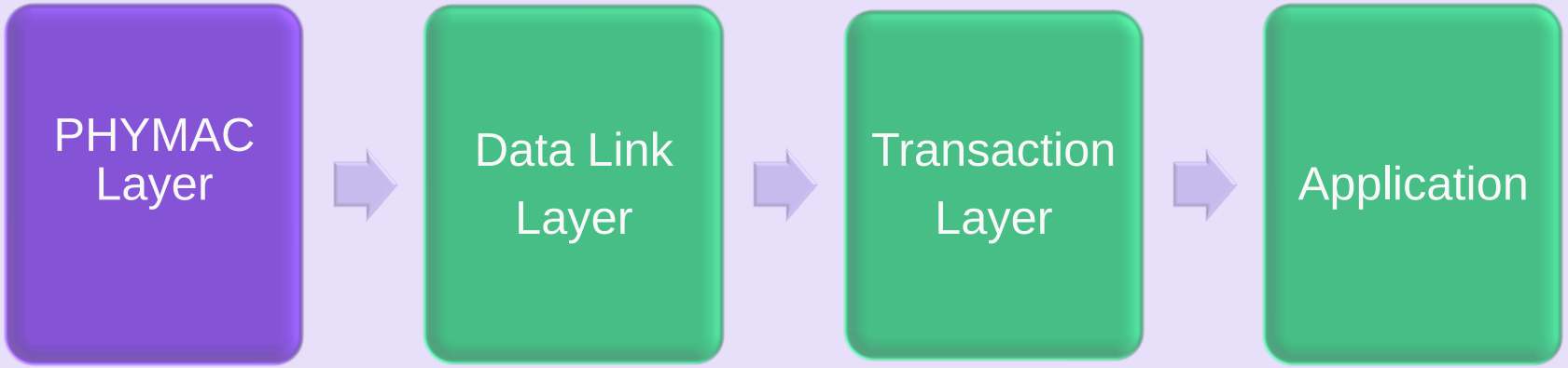
Clock Domain Crossing

- Solution #3: Change clock frequency between PHYMAC & Data Link layers:



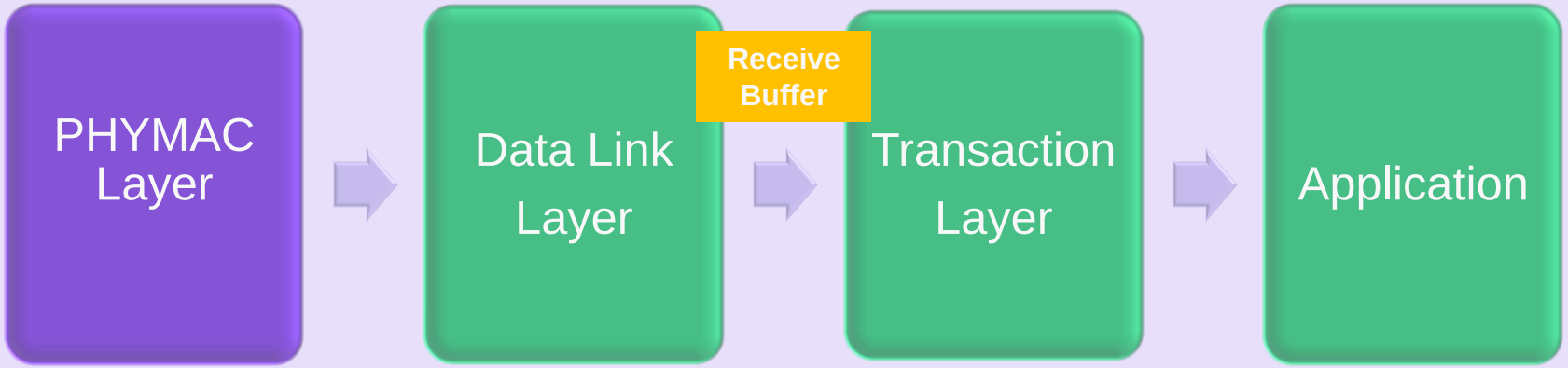
Clock Domain Crossing

- Benefit:
  - ✓ Most of PCIe logic runs at application frequency, this allows large power saving:
    - Example: with a PIPE 8-bit interface clock is 250MHz in PCIe 1.x, however with a 32-bit datapath a 62.5MHz application clock is enough



Clock Domain Crossing

- Drawbacks:
  - ✓ All receive/transmit datapath must be buffered with FIFOs or memories (consumes gate & adds latency)
  - ✓ Application clock frequency must always be high enough to match data rate
    - This is because there is no way to limit PCIe throughput before receive buffer



Case Study Conclusion

- Through these examples we can see that some key architecture choices directly impact PCIe device in terms of:
  - ✓ flexibility
  - ✓ performance
  - ✓ logic & power usage

Thank you for attending
PCI-SIG Developers Conference
Israel 2011

For more information please go to
www.pcisig.com