Single Root Resource Discovery and Allocation

Renato Recio

Contents

- Overviews
  - ✓ Function types
  - ✓ Terms
  - ✓ Initialization Flows
  - ✓ Mapping of (new) PCI IOV Capabilities to Function types
- New PCIe IO Virtualization (IOV) Capabilities register set.
  - ✓ SR-IOV Capabilities Structure

Single Root Resource Discovery and Allocation

Overviews

Function types

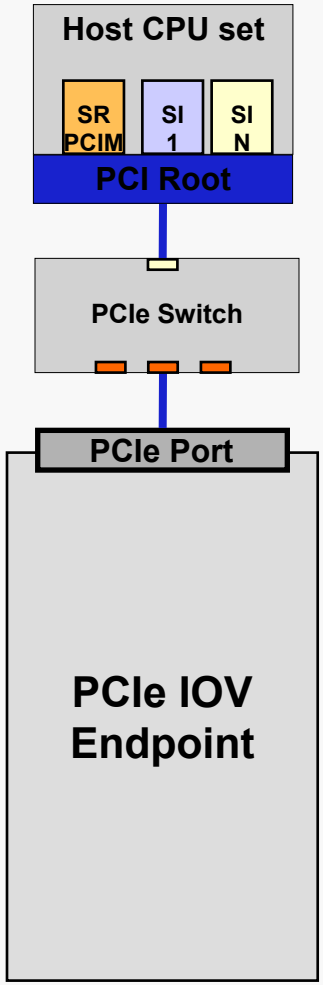
Terms

Initialization Flows

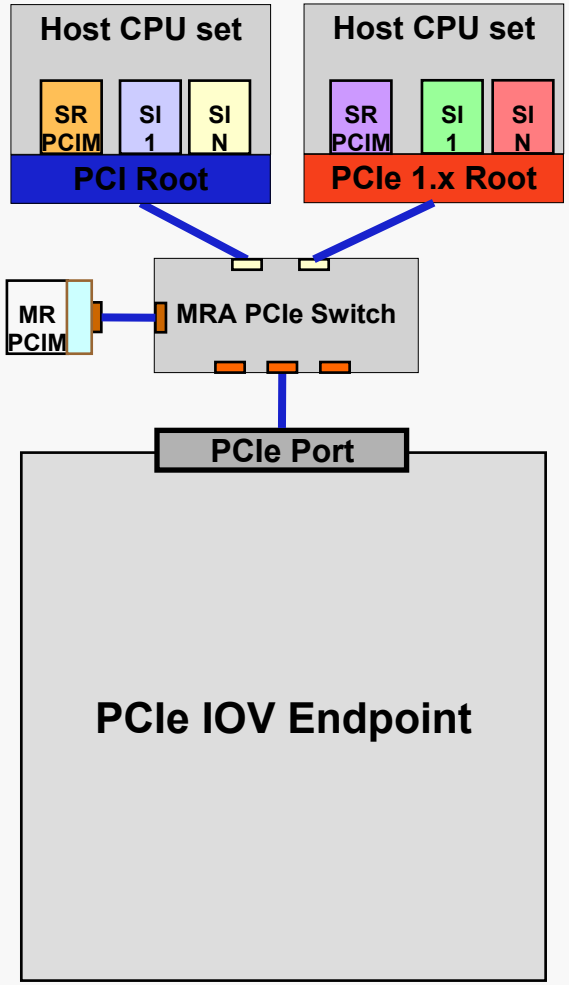
Mapping of (new) PCI IOV Capabilities to Function types

Topology Overview and Terms

SR Topology



Multi-Root Topology



Terms

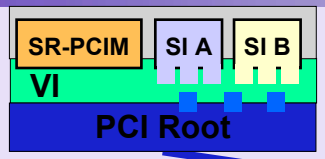
Single Root (SR) IOV Overview

- Only has one Root.
- Switches only need to support PCIe base functionality.
- To make full use of IOV, EP must support SR-IOV capabilities.
- SR-PCIM configures the EP.

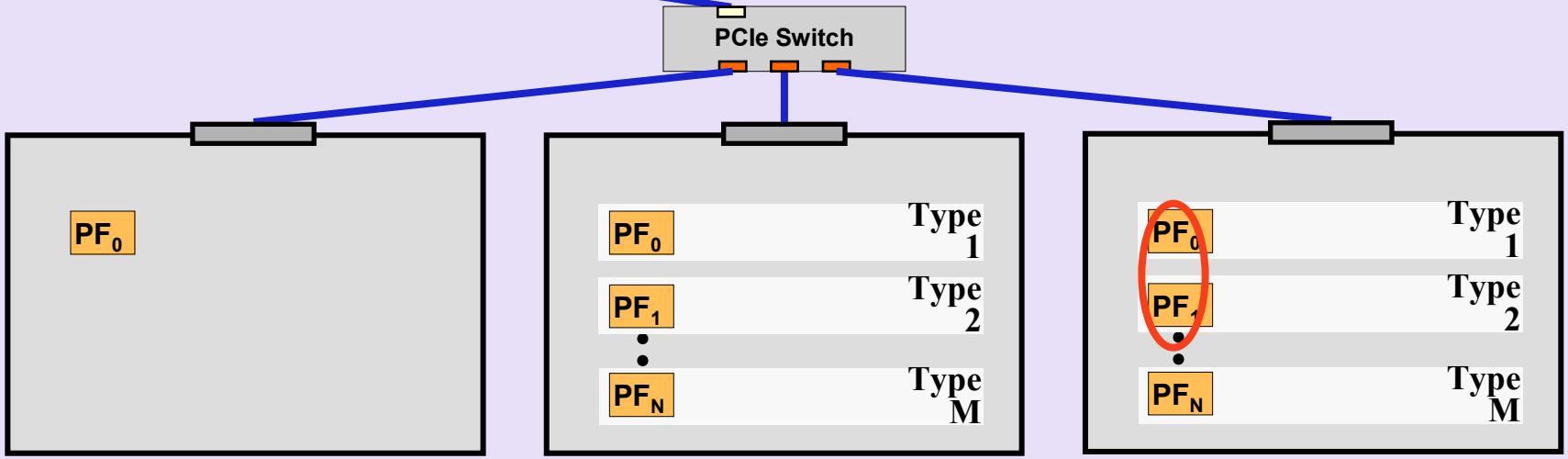
Multi-Root (MR) IOV Overview,

- One or more Roots.
- Switches with Multi-Root Aware (MRA) functionality are needed.
- To make full use of IOV, EP must support SR & MR-IOV capabilities.
- MR-PCIM assigns Virtual Endpoints (VEs) to RCs and manages PCIe components.
- SR-PCIM configures its VEs.

Today's (SR) EPs without Native Virtualization



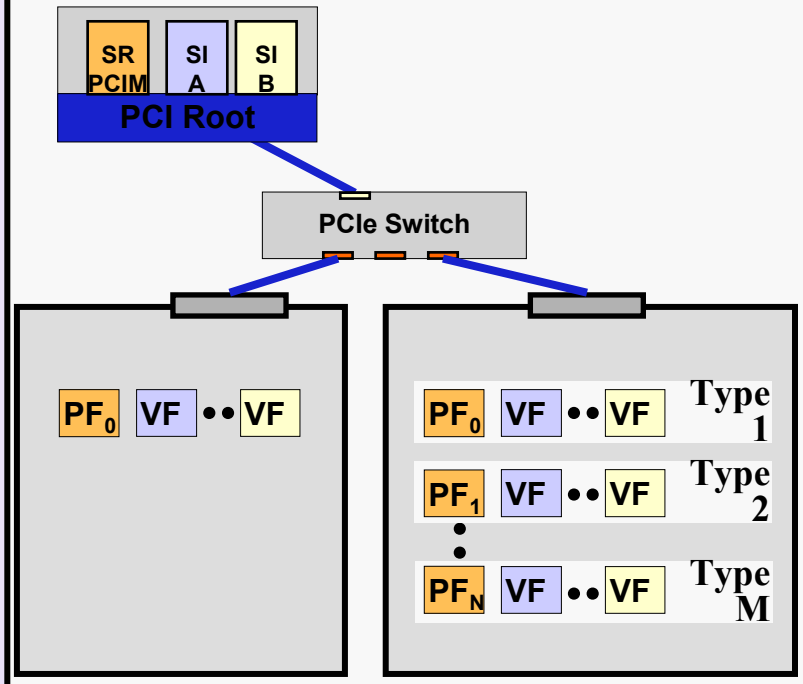
- EP shared through virtualization intermediary (VI)
- VI involved in all PCIe transactions
- EP does not need to support virtualization



- EP with one function
- EP with multiple independent functions
- EP with multiple dependent functions
  - ✓ EP has vendor defined function dependencies
  - ✓ In the above example, PF_0 and PF_1 must go together.

Single-Root IOV Function Types and Terms

SR Topology



Terms

For SR topologies,

Physical Function is used by SR-PCIM to manage a set of Virtual Functions.

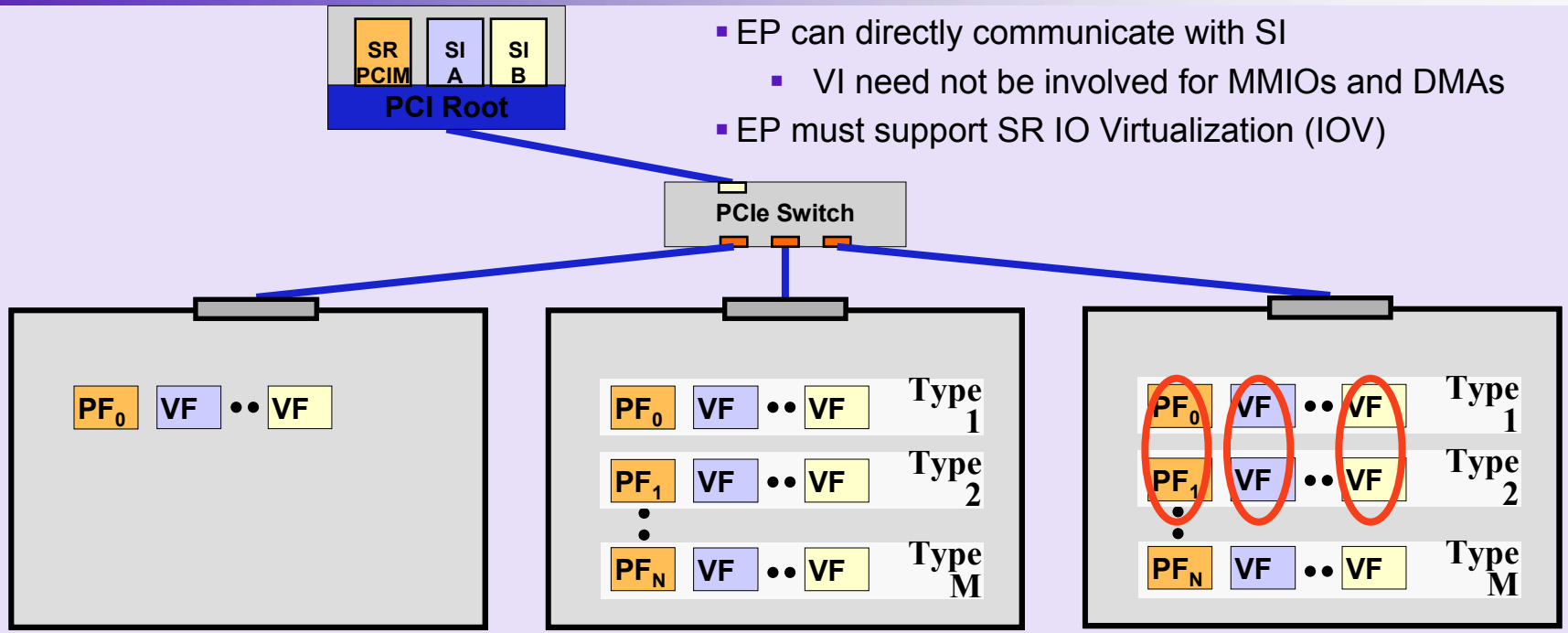
Physical Function 0 (PF_0) is also used to manage EP functions, such as physical errors and events.

Virtual Function is used by SIs to access resources on the EP.

More details on function assignment, capabilities, etc… later.

EP enabled for SR IO Virtualization

- EP can directly communicate with SI
  - VI need not be involved for MMIOs and DMAs
- EP must support SR IO Virtualization (IOV)

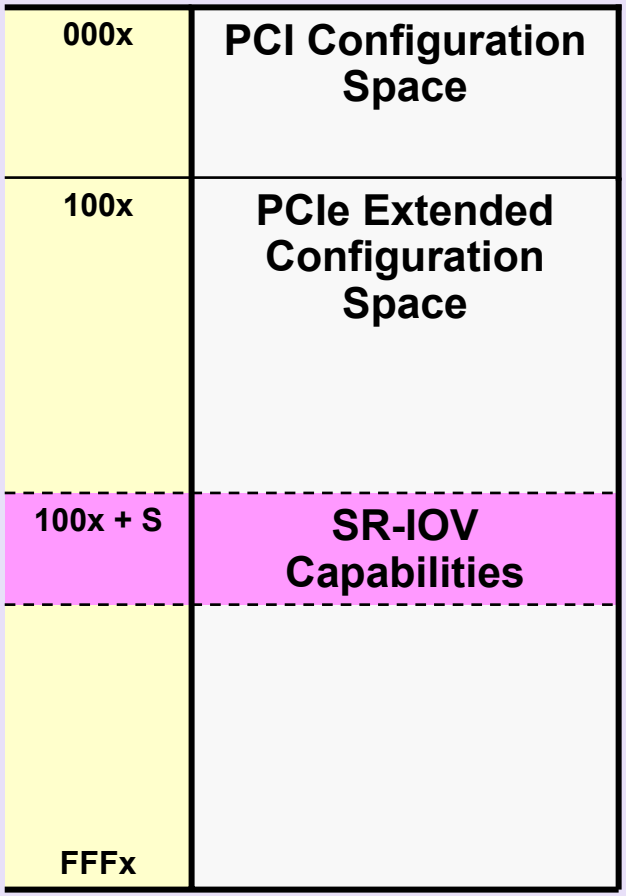


- Use by EP with one PF
- PF describes VF capabilities, i.e.:
  - ✓ Maximum number of VFs.
  - ✓ … *more later*

- Use by EP with multiple independent functions
- Each PF describes its VF capabilities, i.e.:
  - ✓ Maximum number of VFs for PF type.
  - ✓ … *more later*

- Use by EP with multiple dependent functions
- Each PF describes its VF capabilities, i.e.:
  - ✓ Maximum number of VFs for PF type.
  - ✓ Combinations
  - ✓ … *more later*

Configuration Space Overview



- IOV adds two new, optional PCIe Extended Capabilities for each Physical Function that support SR-IOV.

  ✓ SR-IOV Capability - used by SR-PCIM to:
  - Determine EP is SR-IOV capable
  - Enable/disable and configure the EP's SR-IOV capability

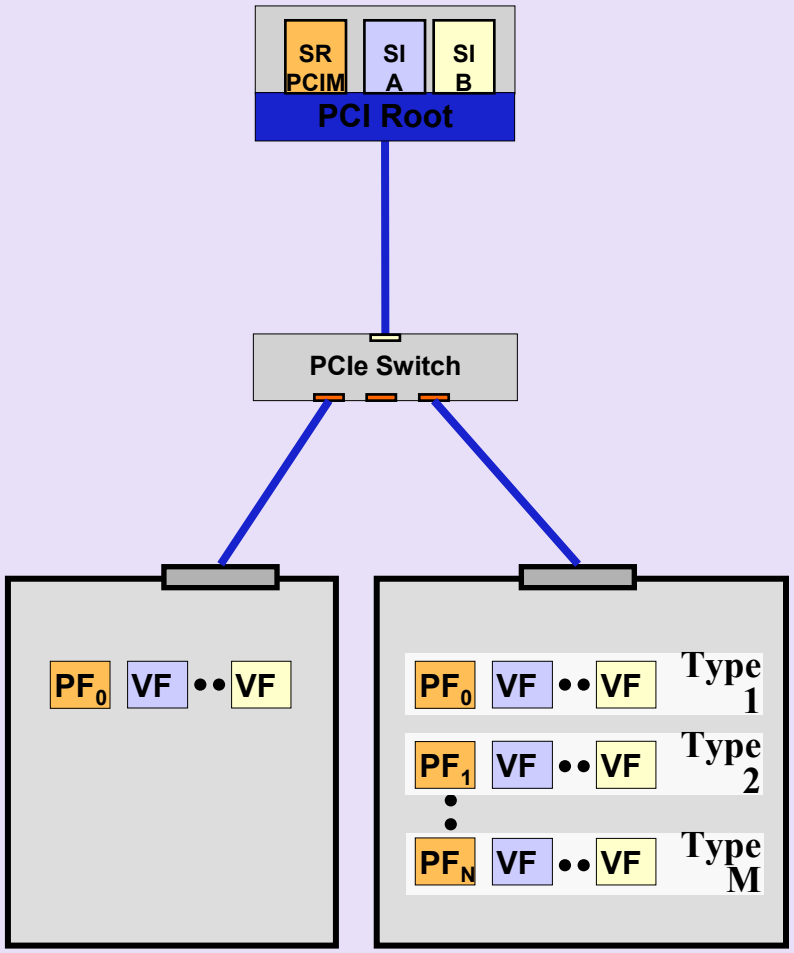
- This new capability is located in the PCIe Extended Configuration Space (offset 256 or greater).

Overview of Function Types used in Single Root Topologies

Function type	Owned by
Physical function	SR-PCIM
Virtual function	System Image

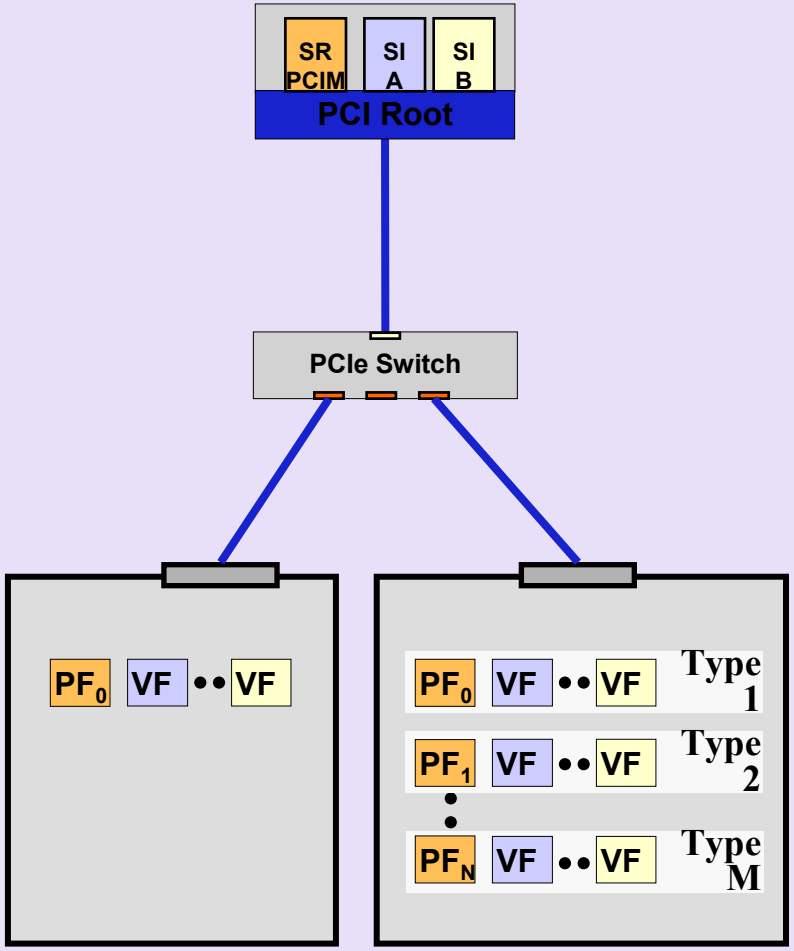
- Physical Function
  - ✓ Used by SR-PCIM to discover and set-up a function's IOV capabilities, such as configuring VFs.
  - ✓ Physical Function 0 is also used by SR-PCIM to manage the physical EP, such as physical errors and events.
- The intention is to enable isolation of Physical Functions to SR-PCIM.
- System Image sees VFs assigned to.

SR EP Initialization Overview



- SR-PCIM discovers and enumerates PCIe components.
- SR-PCIM discovers SR-IOV EP by reading SR-IOV Capabilities Register in the PFs.
  - ✓Describes SR-IOV functions supported by the EP, such as number of VFs supported by the EP
  - ✓… more details later …
- SR-PCIM owns the PF.
- SR-PCIM configures SR-IOV EP by writing SR-IOV Capabilities Register in the PFs.
  - ✓Sets SR-IOV capabilities, such as number of VFs per PF
  - ✓… more details later …
  - ✓The process SR-PCIM uses to allocate MMIO space to the EP's PCIe port is defined later in this deck.

Other SR-PCIM Responsibilities



- VI (not shown) manages SIs access to config space
  - ✓ Access to certain fields in VFs config space may be emulated by the VI.
  - ✓ B/D/F translation may be required by some implementations (e.g. for SIs that can't handle EPs that start with non-zero function numbers).
- SR-PCIM manages
  - ✓ Physical errors (i.e. errors that can't be associated with a single VF)
  - ✓ Physical events (i.e. events that can't be associated with a single VF)

Resource Discovery and Allocation

New PCIe IOV Capabilities register set

SR-IOV Capabilities Structure

SR IOV Capabilities Fields

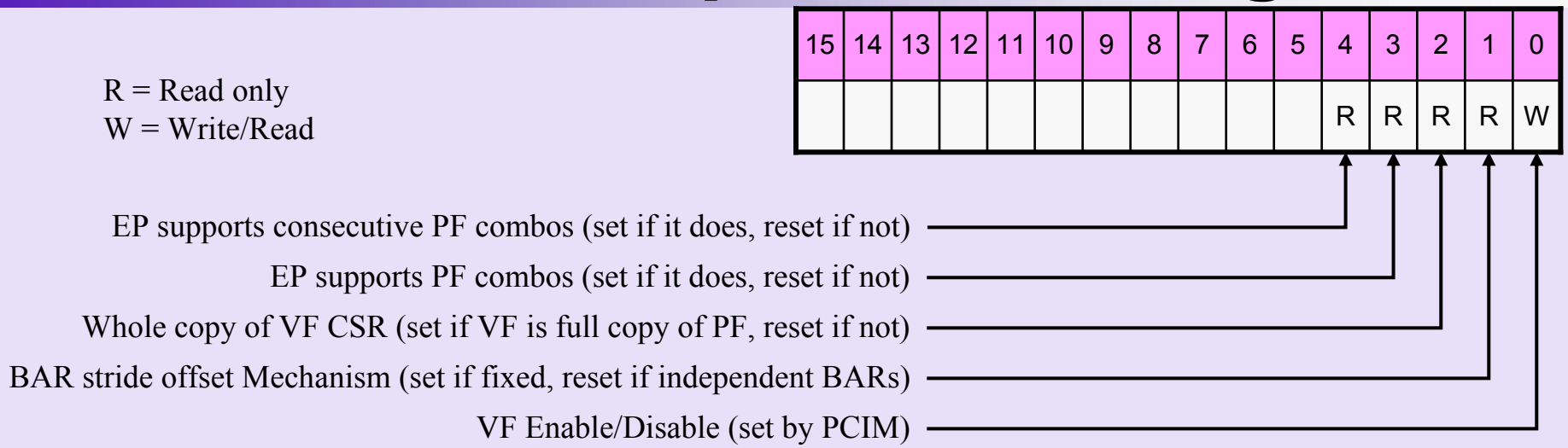
- Three EP architecture approaches will be described:
  - ✓ EPs with a single function type
  - ✓ EPs with multiple function types
  - ✓ EPs with dependent function types
- The following slides will describe the new PCIe Extended Configuration fields for SR-IOV.
  - ✓ Single-bit fields are fully defined.
  - ✓ Size of multi-bit fields has not been set … yet
    (WG still discussing scaling of the field over time).
  - ✓ Following slides represent our current direction…
    … which is subject to change as complete the specification.

SR-IOV Capabilities

SR-IOV PCIe Extended Configuration Space		
SR-IOV Capabilities	Next Cap Ptr	Version
RID Space Allocation Register(s) (TBD by RID subteam)		
Mapping Dependent Fields (covered on the slides that follow)		

- IOV Capabilities; Next Capabilities Pointer; Version
- RID Space Allocation Register(s) to be defined by RID subteam
- Mapping Dependent Fields - Contents differ depending on which EP architecture approach is used:
  - ✓ EPs with a single PF type
  - ✓ EPs with multiple independent PF types
  - ✓ EPs with multiple dependent PF combinations

SR-IOV Capabilities Register



- VF Enable/Disable - Used by PCIM to set SR-IOV mode.
- BAR Stride Offset Mechanism
  - ✓ Set if EP supports the same, fixed offset for all the VFs' BAR.
  - ✓ Reset if EP supports a Independent BARs for each VF.
- Whole copy of VF - Set if VFs share **all fields** with their associated PF.
- EP supports PF combos - Set if EP supports dependent functions.
- EP supports consecutive PF combos - For an EP with dependent functions, set if dependent functions must be sequentially assigned.

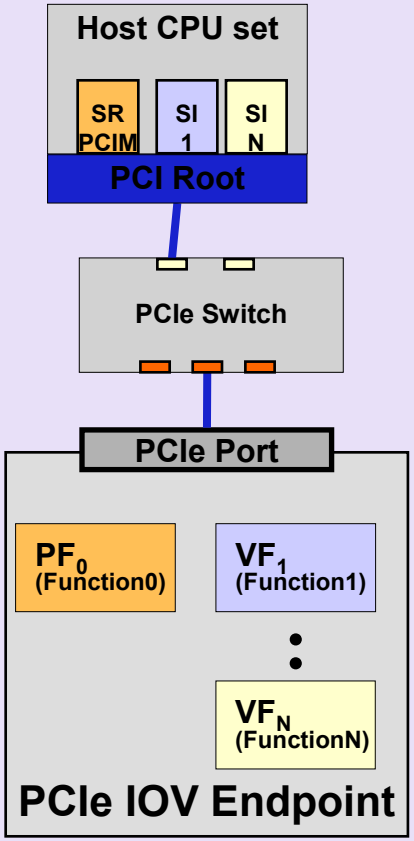
EPs with one VF type

SR-IOV PCIe Extended Configuration Space	
:	
Max VFs (RO)	Num VFs (RW)
Max Stride (RO)	
Min Stride (RO)	
Stride (RW)	

- Max VFs - Defines the maximum number of VFs supported by the EP.
- Num VFs - The number of VFs assigned to the PF by SR-PCIM (must be equal to or smaller than Max VFs.
- If EP supports fixed BAR stride:
  - ✓ Max Stride* - Defines the maximum BAR offset for all VFs.
  - ✓ Min Stride* - Defines the minimum BAR offset for all VFs (vendor defined minimum amount of MMIO space needed by each VF).
  - ✓ Stride - Defines the BAR offset for all VFs
  - ✓ Min Stride < Stride < Max Stride and all Stride fields must be a power of 2.
  - ✓ *For EPs with multiple BARs, the fixed stride applies to all BARs.*
- *If the EP supports Independent BARs, then each VF has full BAR(s) and the Stride fields are not used.*

*Note: each of these fields has a documented limit in the specification, which EPs must support.

Example of an EP with fixed Stride & one VF Type in an SR-IOV Topology



SR-IOV PCIe Extended Configuration Space		
SR-IOV Capabilities	Next Cap Ptr	Version
Max VFs	Num VFs	
Max Stride		
Min Stride		
Stride		

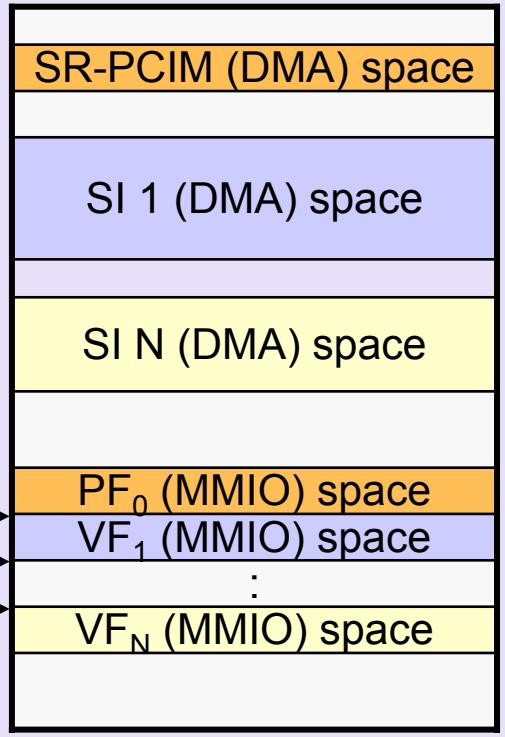
The Assigned Stride equals the offset into the PF BAR for each VF, such that:

- **VF_1 starts at base BAR + Stride**
- **VF_2 starts at base BAR + [(2) x (Stride)]**

…

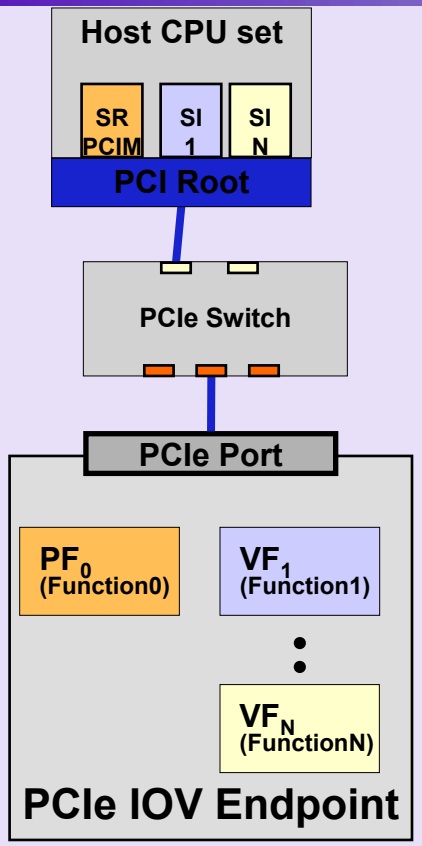
- **VF_N starts at base BAR + [(N) x (Stride)]**

Sample PCI Address Layout



Implementation note: Firmware can use the Max VFs and Max Stride fields to calculate the amount of MMIO space to allocate to the EP. For example, the MMIO space for the EP's port could be set to: (Max Stride) x (Max VFs + 1).

Example of an EP with independent BARs & 1 VF Type in an SR-IOV Topology

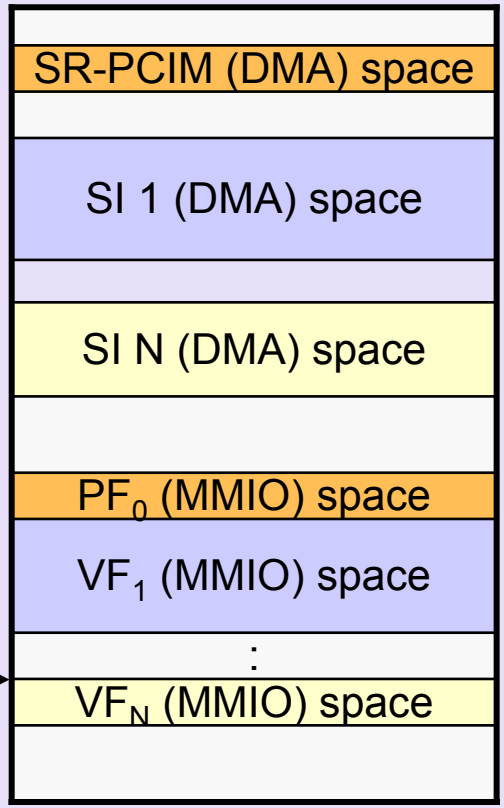


SR-IOV PCIe Extended Configuration Space		
SR-IOV Capabilities	Next Cap Ptr	Version
Max VFs	Num VFs	

VF_N PCI Configuration Space
:
BAR_1
:

Each VF has a full set of BARs in its PCI Configuration Space.

Sample PCI Address Layout



Implementation note: Firmware can use the Max VFs and BAR fields to calculate the amount of MMIO space to allocate to the EP. For example, the MMIO space for the EP's port could be set to: (Max MMIO space specified by BAR) x (Max VFs + 1).

EPs with multiple, independent VF types

SR-IOV PCIe Extended Configuration Space		
SR-IOV Capabilities	Next Cap Ptr	Version
Max VFs	Num VFs	VF Offset
Max Stride		
Min Stride		
Stride		

- Fields for an EP with multiple, independent VF types have the same definitions as for an EP with one VF type. One additional field is used:
  - ✓ VF Offset - Virtual function pointer (see next page).

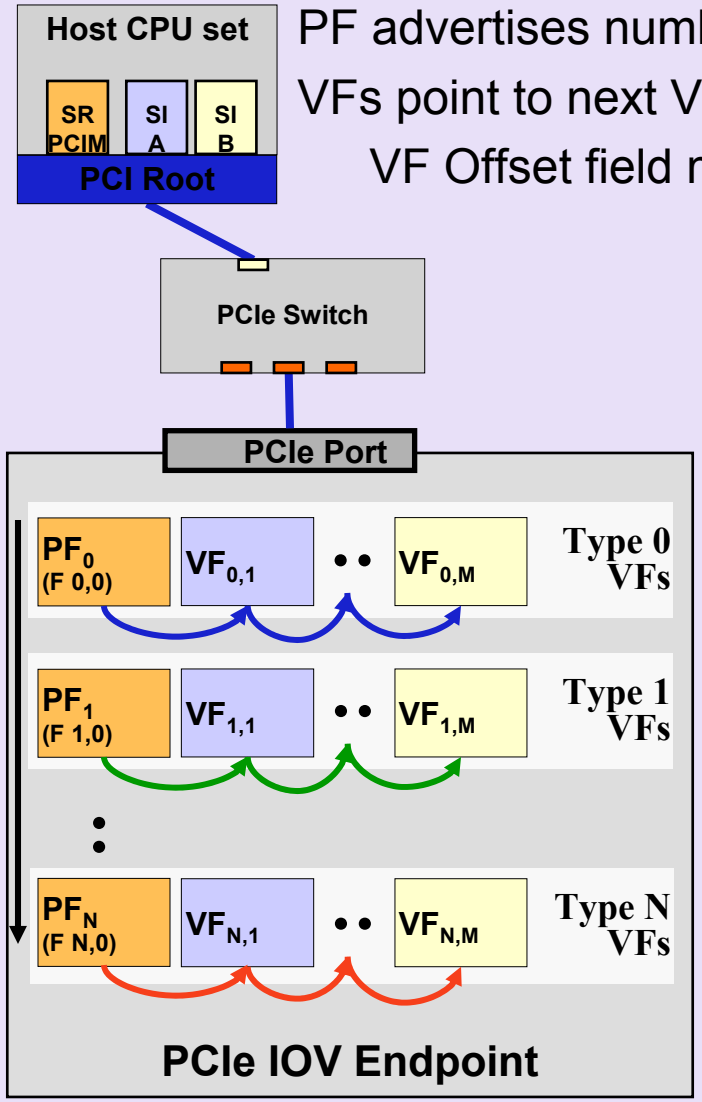
- Note: each VF type must have a physical function.
  - ✓ The PF must contains the above SR-IOV PCIe Extended Configuration Space that describes the capabilities of the VF instances of the PF.

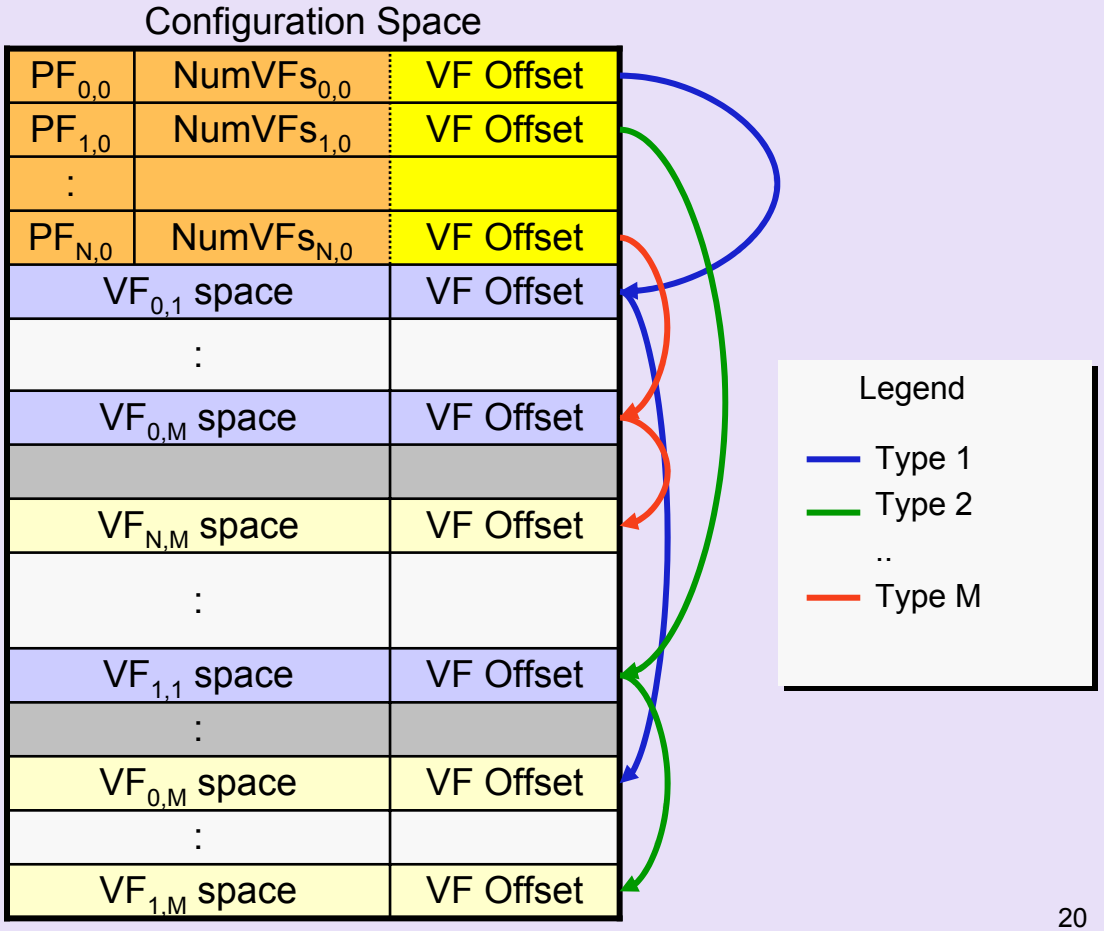
Overview of EPs with multiple, independent VF types

PF advertises number of VFs and points to 1st VF

VFs point to next VF – advertise relative offset:

VF Offset field must be in the VF's CSRs.





EPs with multiple, fixed interdependent functions

SR-IOV PCIe Extended Configuration Space	
SR-IOV Capabilities	Next Cap Ptr
RID Space Allocation Register(s) (TBD by RID subteam)	
Max VFs	Num VFs
Nominal Combo Number (RW)	
Combo Instance Number (RW)	
Supported Combinations (RO)	
$Stride_1$	
:	
$Stride_N$	

- IOV Capabilities; Next Capabilities Pointer; Version; and RID Space, Max VFs, and Num VFs are as defined before.
- Nominal Combo Number - Describes the combination to be used by the Function.
- Combo Instance Number - A unique identifier for the specific combination instance the Function is in.
- Supported Combinations - Function truth table that describes which function combinations the EP supports.
- For each VF, Stride fields are as defined previously.

Supported Combinations Combo Truth Table

Nominal Combo Number	$Type_N$	$Type_{N-1}$	. . .	$Type_1$	$Type_0$	Supported Combinations
0	0	0		0	0	0
1	0	0		0	1	0/1
2	0	0		1	1	0/1
3	0	0		0	0	0/1
…						
X-2	1	1		0	0	0/1
X-1	1	1		0	1	0/1
X	1	1		1	1	0/1

Function Types Supported by the EP

EP supported combinations are 1

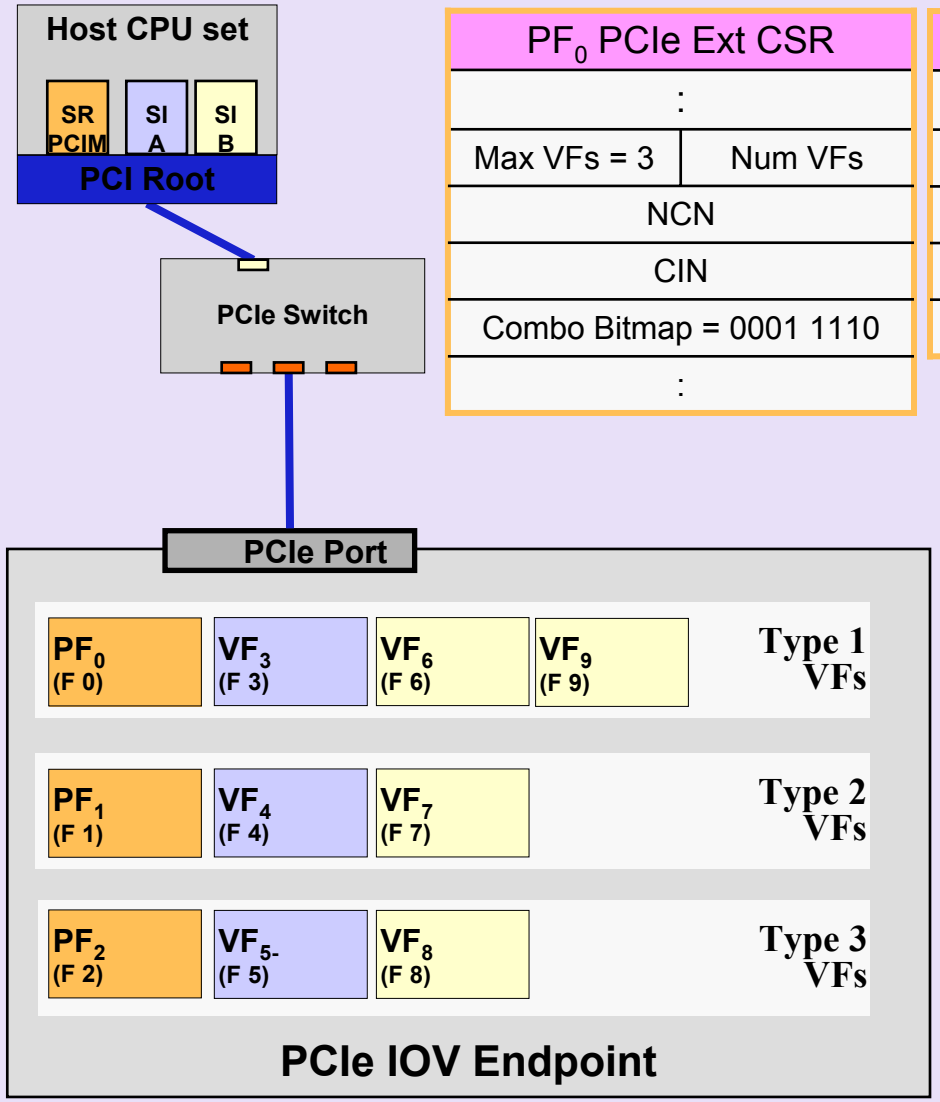
EP unsupported combinations are 0

Example

- EP with 3 Function types, where:
  - ✓ Function $Type_0$ can be used independently.
  - ✓ Function $Type_1$ can be used independently.
  - ✓ Function $Type_2$ can be used independently.
  - ✓ Function $Type_0$ and $Type_1$ can be used together.

Nominal Combo Num	2	1	0	Supported
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	0

Example - discovery



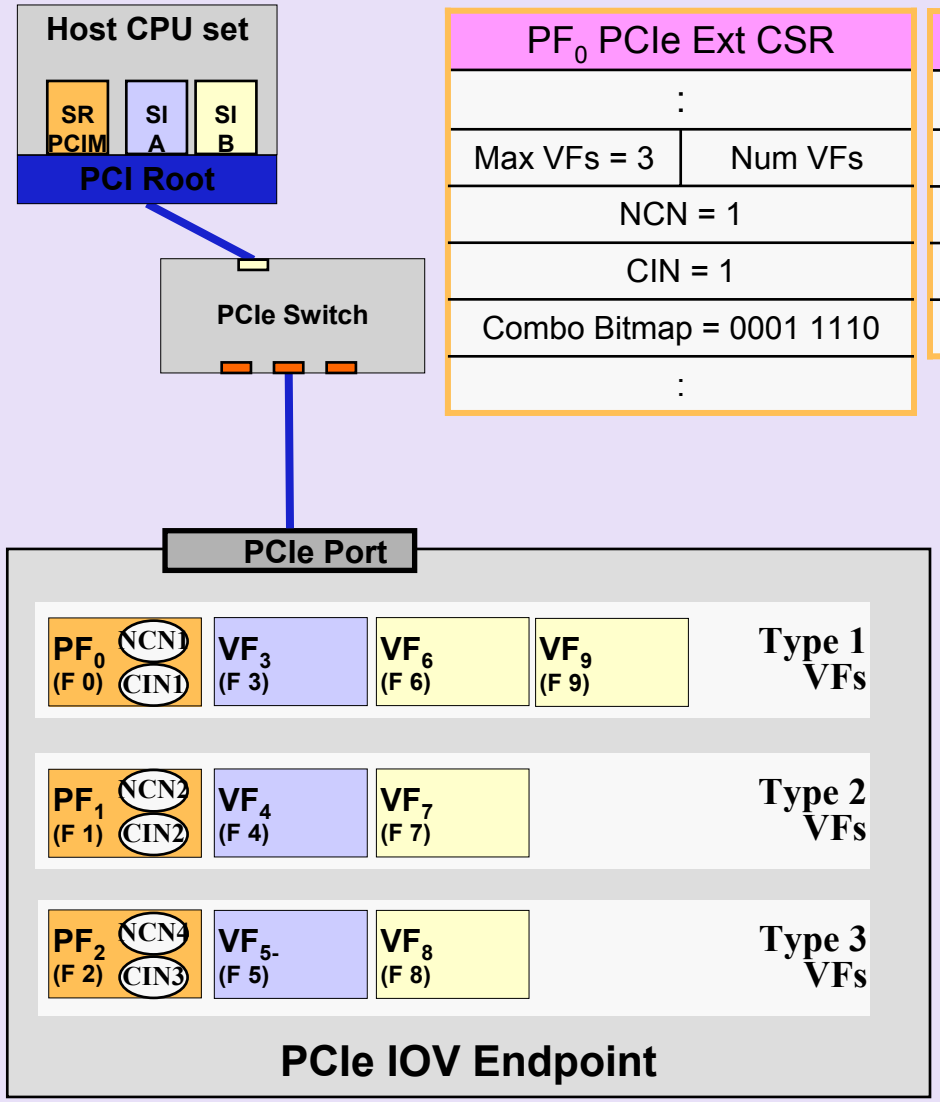
PF_0 PCIe Ext CSR	
:	
Max VFs = 3	Num VFs
NCN	
CIN	
Combo Bitmap = 0001 1110	
:	

PF_1 PCIe Ext CSR	
:	
Max VFs = 2	Num VFs
NCN	
CIN	
:	

PF_2 PCIe Ext CSR	
:	
Max VFs = 2	Num VFs
NCN	
CIN	
:	

- SR-PCIM reads EP:
  - ✓ PF_0 CSRs to determine
    - SR-IOV Cap = **Consecutive** FNs
    - Max VFs = 3
    - **PF Combinations**
    - …
  - ✓ PF_1 CSRs to determine
    - Existence of PF_1
    - Max VFs
  - ✓ PF_2 CSRs to determine
    - Existence of PF_2
    - Max VFs

Example - PF Combination setting



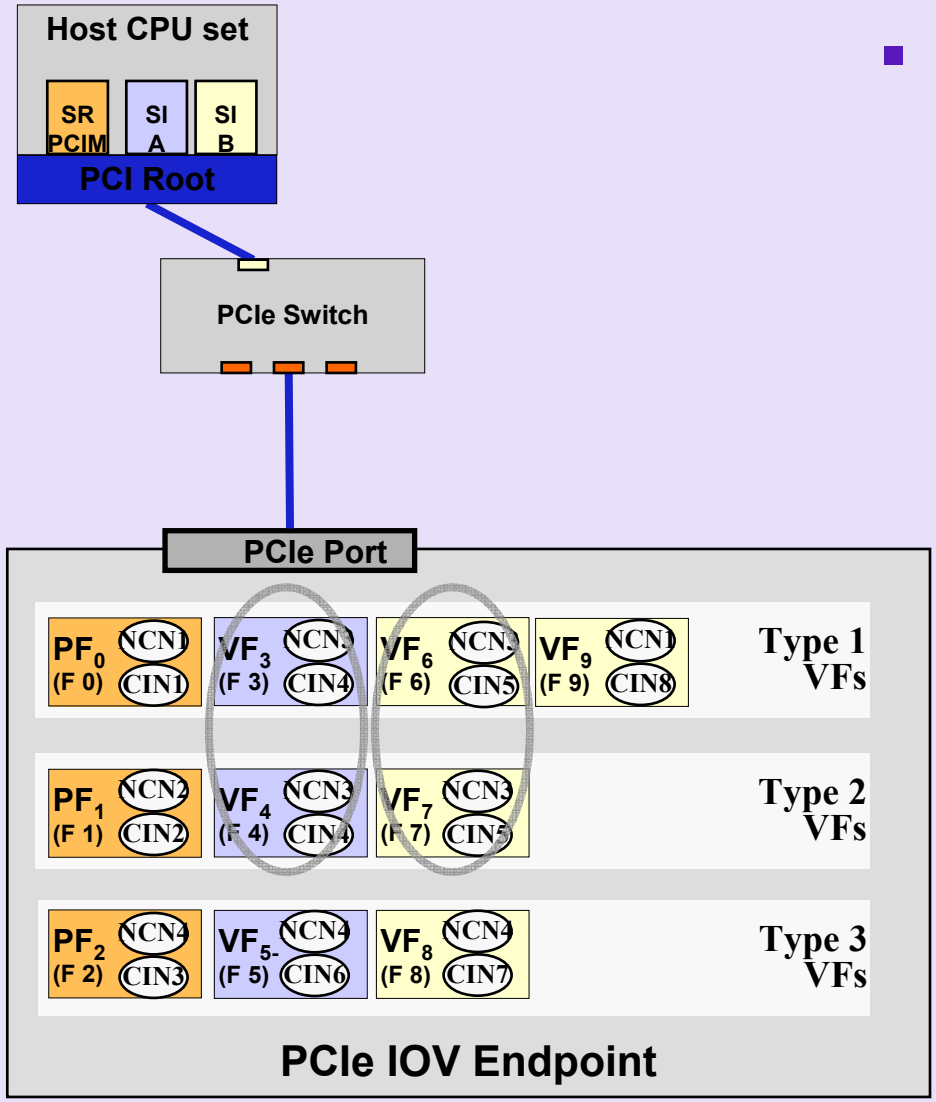
PF_0 PCIe Ext CSR	
:	
Max VFs = 3	Num VFs
NCN = 1	
CIN = 1	
Combo Bitmap = 0001 1110	
:	

PF_0 PCIe Ext CSR	
:	
Max VFs = 2	Num VFs
NCN = 3	
CIN = 2	
:	

PF_0 PCIe Ext CSR	
:	
Max VFs = 2	Num VFs
NCN = 4	
CIN = 3	
:	

- SR-PCIM sets combos for PFs:
  - ✓ Writes to PF_0 CSRs
    - Num VFs = 3
    - **NCN = 1**
    - **CIN =1**
  - ✓ Writes to PF_1 CSRs
    - Num VFs = 2
    - **NCN = 2**
    - **CIN = 2**
  - ✓ Writes to PF_2 CSRs
    - Num VFs = 2
    - **NCN = 4**
    - **CIN = 3**

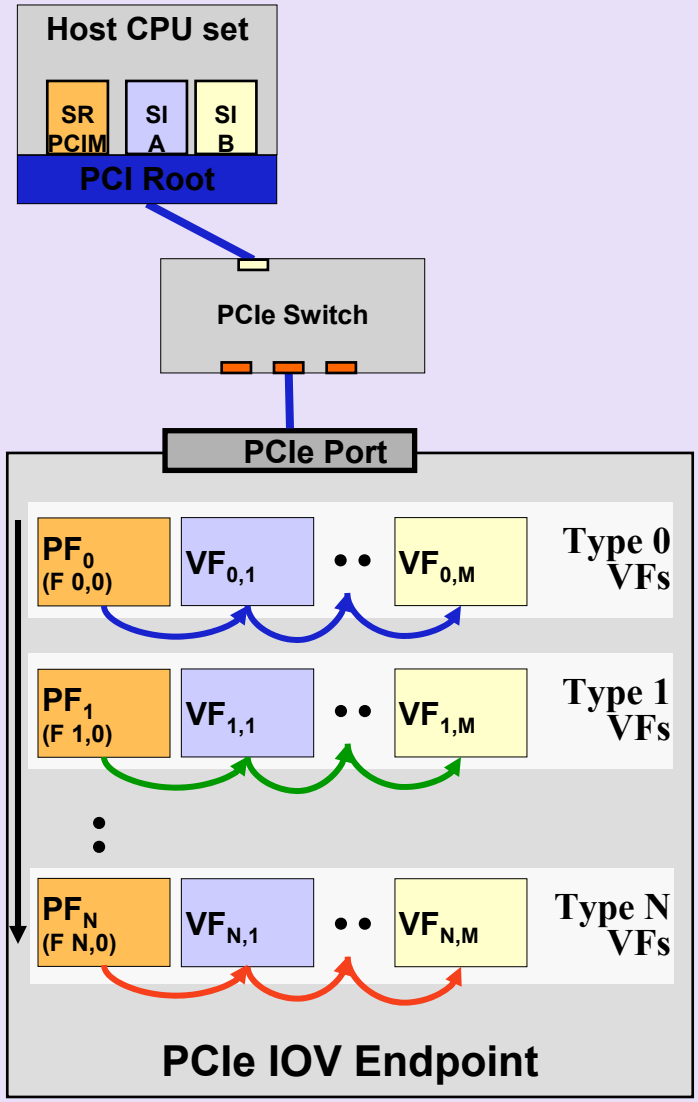
Example - VF Combination setting



- SR-PCIM sets combos for VFs:
  - ✓ Writes to VF_3 CSRs
    - **NCN = 3**
    - **CIN = 4**
  - ✓ Writes to VF_4 CSRs
    - **NCN = 3**
    - **CIN = 4**
  - ✓ …
  - ✓ Writes to VF_9 CSRs
    - **NCN = 1**
    - **CIN = 8**

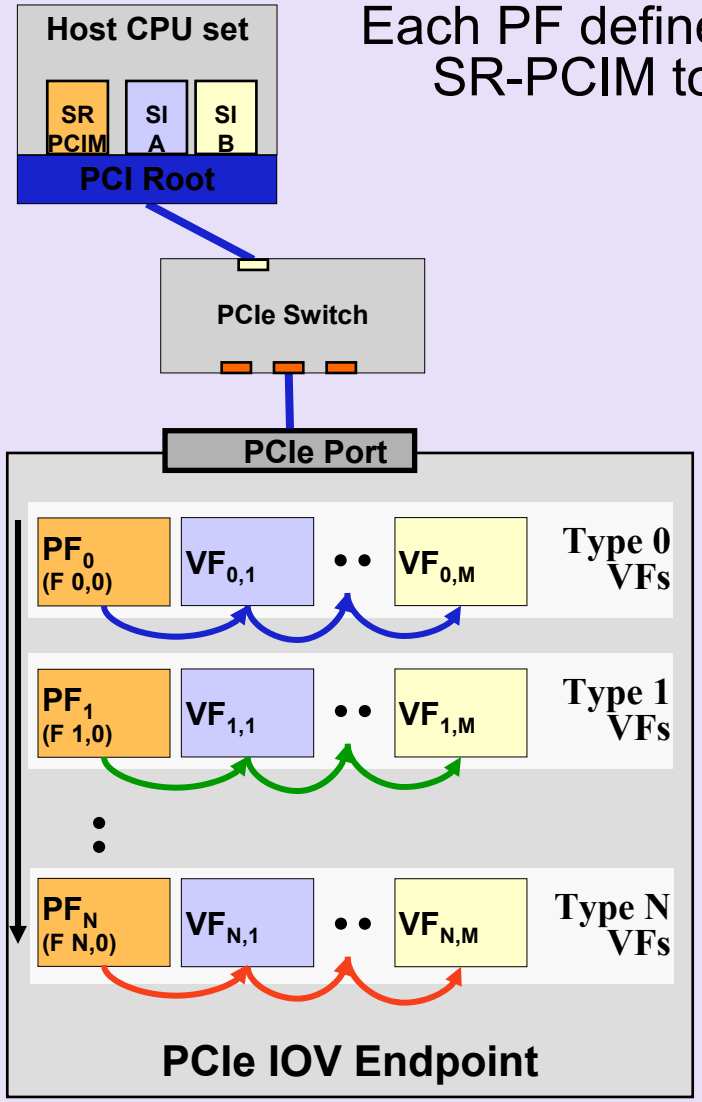
- Back-ups

Example of an EP with multiple, independent VF Types

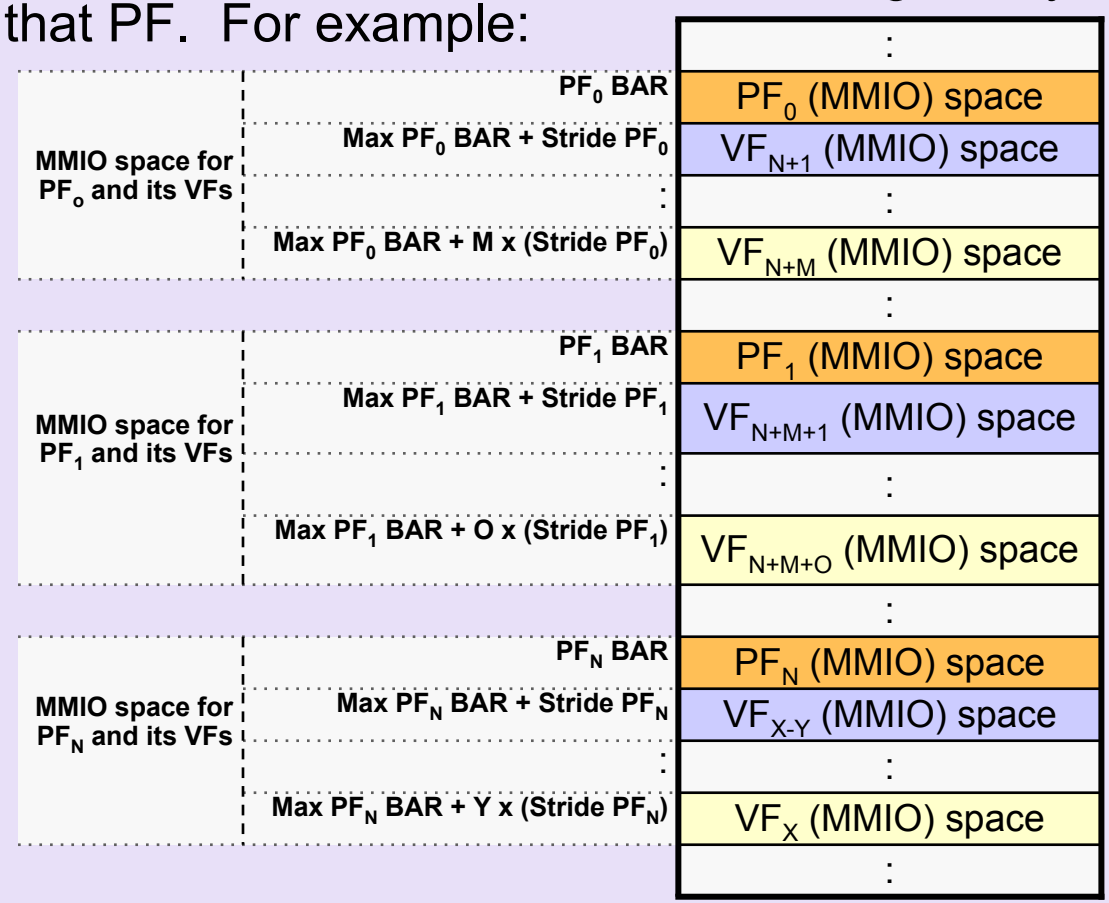


- SR-PCIM determines number of VFs supported by EP, by:
  - ✓ Reading the SR-IOV Capabilities register of each PF to determine: max VFs, BAR capability, …
- SR-PCIM then configures the EP by performing the following for each PF:
  - ✓ Setting the NumVFs and Enable VFs;
  - ✓ Setting the Stride or BARs:
    - If the EP supports a fixed stride, setting the Stride value for the EP.
    - If the EP supports Independent BARs, then sets the BARs for each VF.

EP with multiple, independent function types and fixed stride

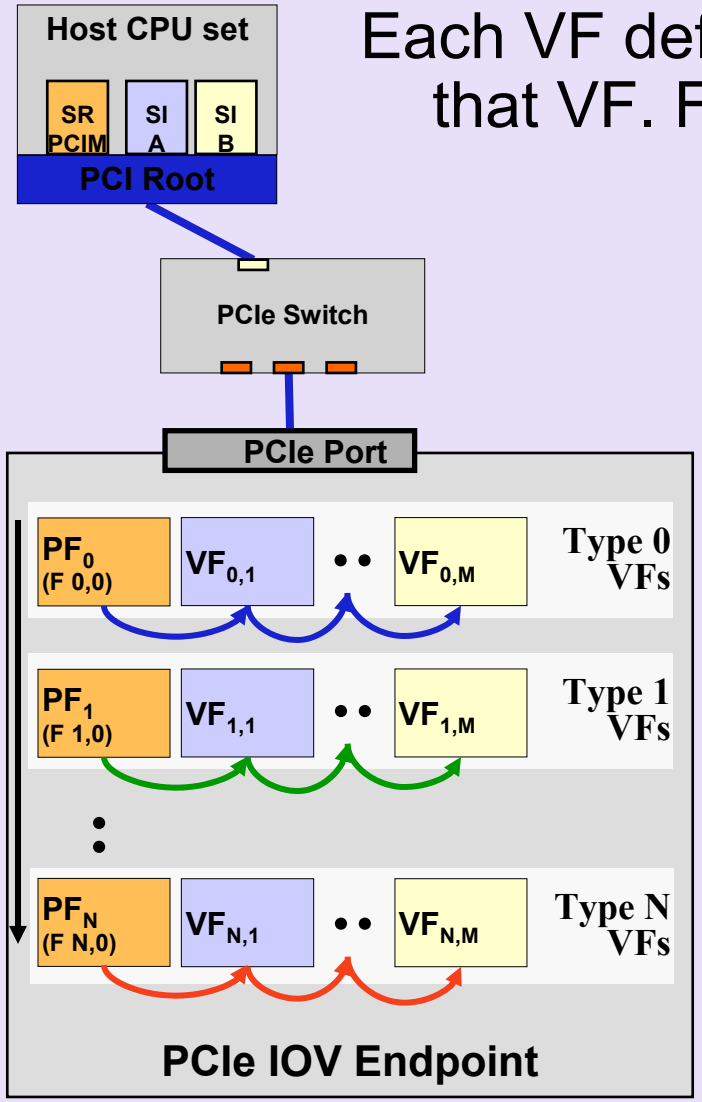


Each PF defines the fixed stride for the NumVFs assigned by SR-PCIM to that PF. For example:

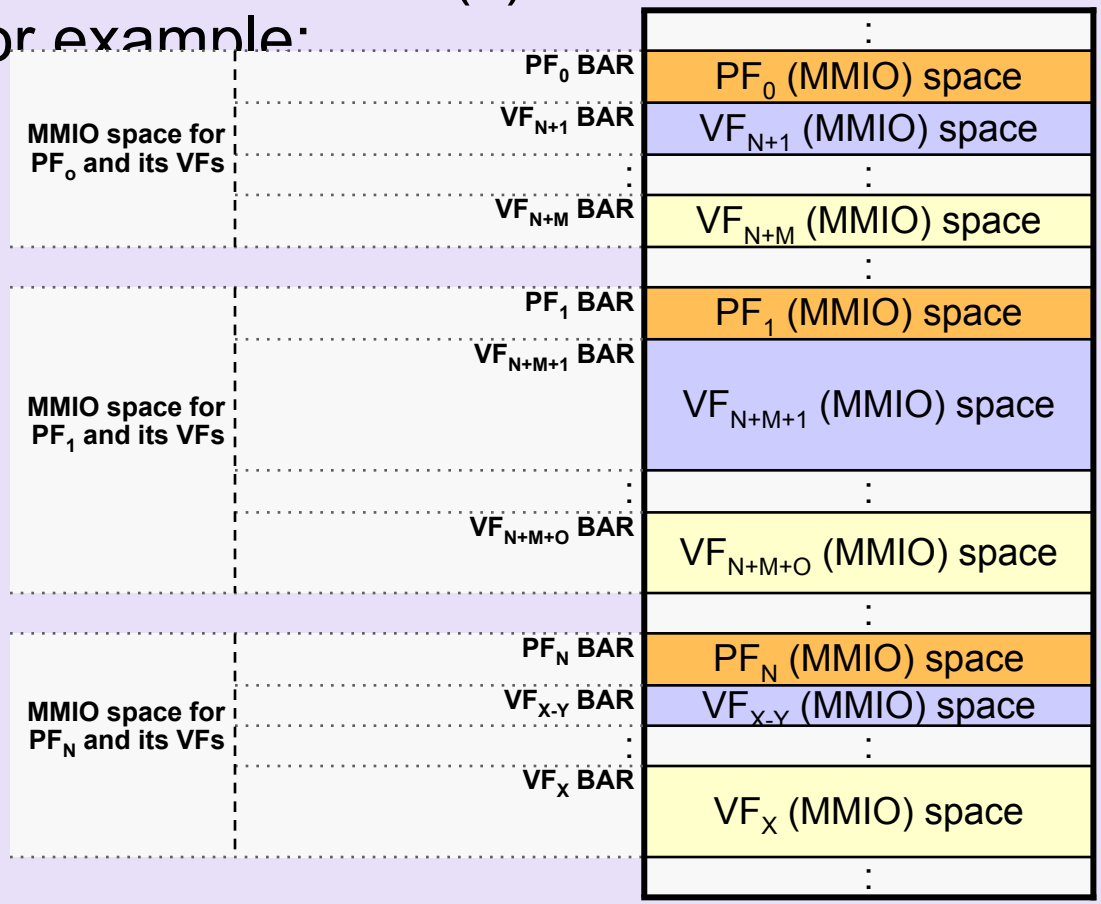


Implementation note: Firmware can use the Max VFs and Max Stride fields of each **PF** to calculate the amount of MMIO space to allocate to the EP.

EP with multiple, independent function types and independent BARs



Each VF defines the full BAR(s) associated with that VF. For example:



Implementation note: Firmware can use the Max VFs and BAR fields of each **PF** to calculate the amount of MMIO space to allocate to the EP.

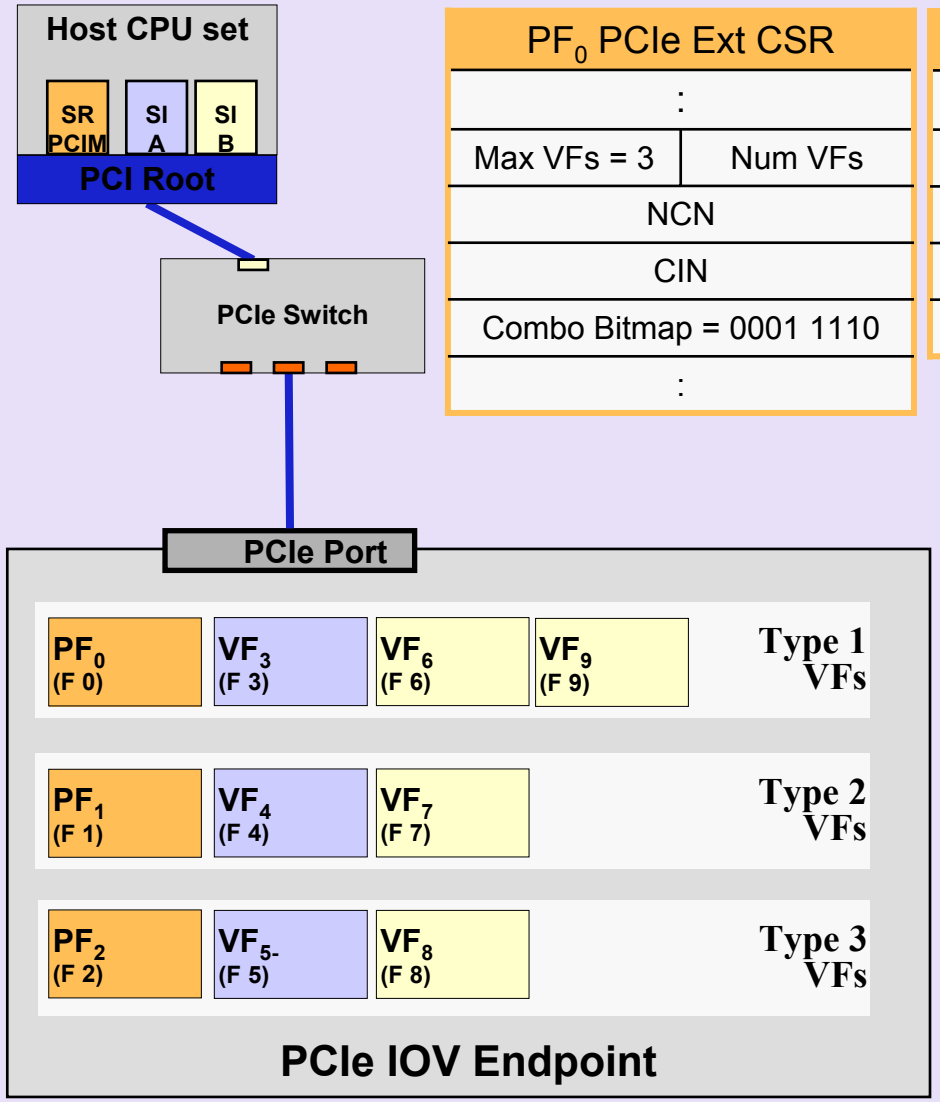
Assumptions

- Both EP and SR-PCIM need to distinguish between
  - ✓ **Potential Combinations Matrix** of Existing Function
  - ✓ **Supported Combo Matrix** of Existing Function
- Both EP and SR-PCIM need to distinguish between
  - ✓ A **Nominal Combo:** This identifies the type of combination a given Function belongs to
  - ✓ A **Combo Instance:** This identifies a particular combination that a given Function belongs to.
- PF's and VF's are allowed to belong to a common combo.
- Any given Function lives in 1 and only 1 combo.

Combo Matrix Assumptions

- Combos are bound to $2^{\text{Max PFs}} - 1$.
  - ✓ Where Max PFs is the maximum PFs the EP supports.
- Potential Combo Matrix Semantics
  - ✓ Potential Combo Matrix be documented in the spec according as an incremental combination list.
    - There is no need for an explicit Potential Combo Matrix CSR/field in the EP.
- Supported Combo Matrix Semantics
  - ✓ The Supported Combo Matrix is kept in a field in the PF0 config space.
  - ✓ The config space for all PF's and VF's have two additional fields:
    - Nominal Combo Number:
      - Identifying the Nominal Combo the Function is in.
    - Combo Instance Number:
      - Identifying the unique number belonging to the instance of a Combo the Function is in.

Example 2 of an EP with interdependent VF Types



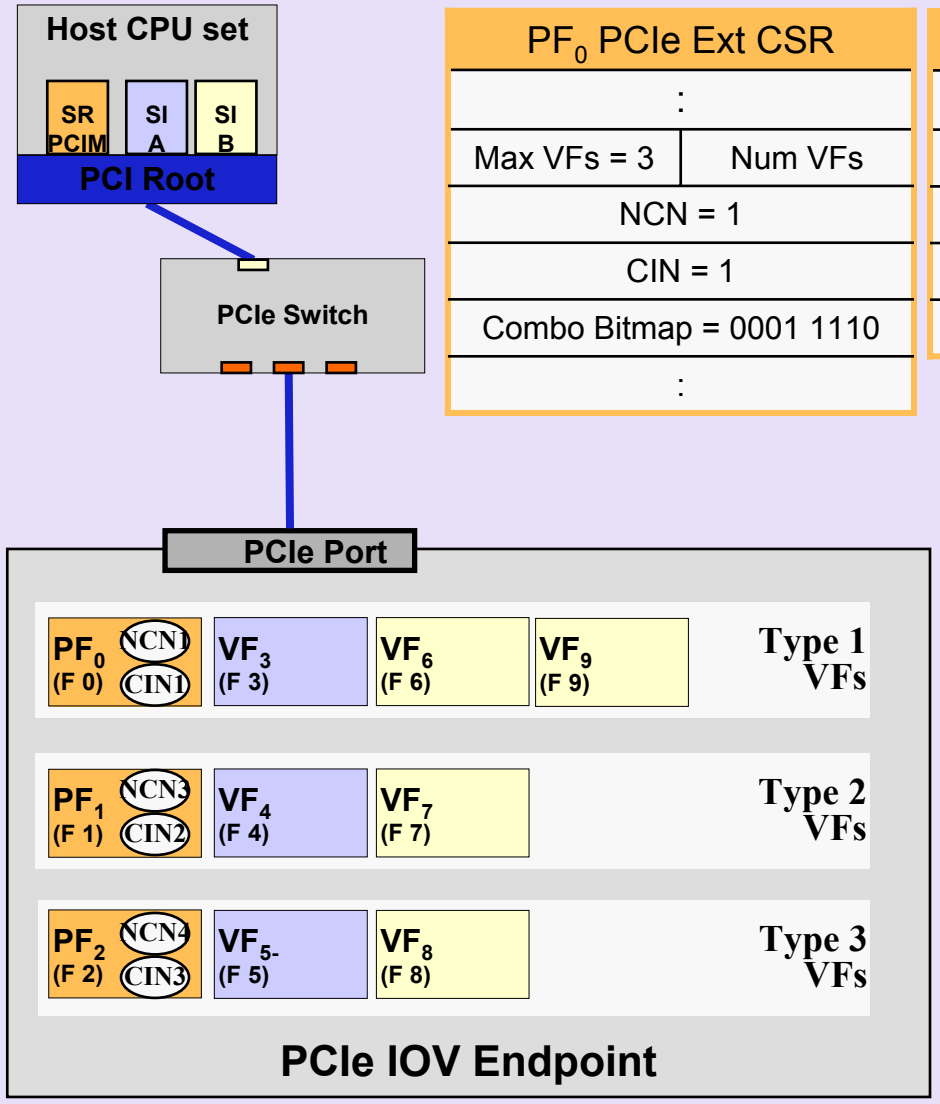
PF_0 PCIe Ext CSR	
:	
Max VFs = 3	Num VFs
NCN	
CIN	
Combo Bitmap = 0001 1110	
:	

PF_1 PCIe Ext CSR	
:	
Max VFs = 2	Num VFs
NCN	
CIN	
:	

PF_2 PCIe Ext CSR	
:	
Max VFs = 2	Num VFs
NCN	
CIN	
:	

- SR-PCIM reads EP:
  - ✓ PF_0 CSRs to determine
    - SR-IOV Cap = **Non-consecutive** FNs
    - Max VFs = 3
    - PF Combinations
    - Stride Type (not redealt with here)
  - ✓ PF_1 CSRs to determine
    - Existence of PF_1
    - Max VFs
  - ✓ PF_2 CSRs to determine
    - Existence of PF_2
    - Max VFs

Example 2 of an EP with interdependent VF Types



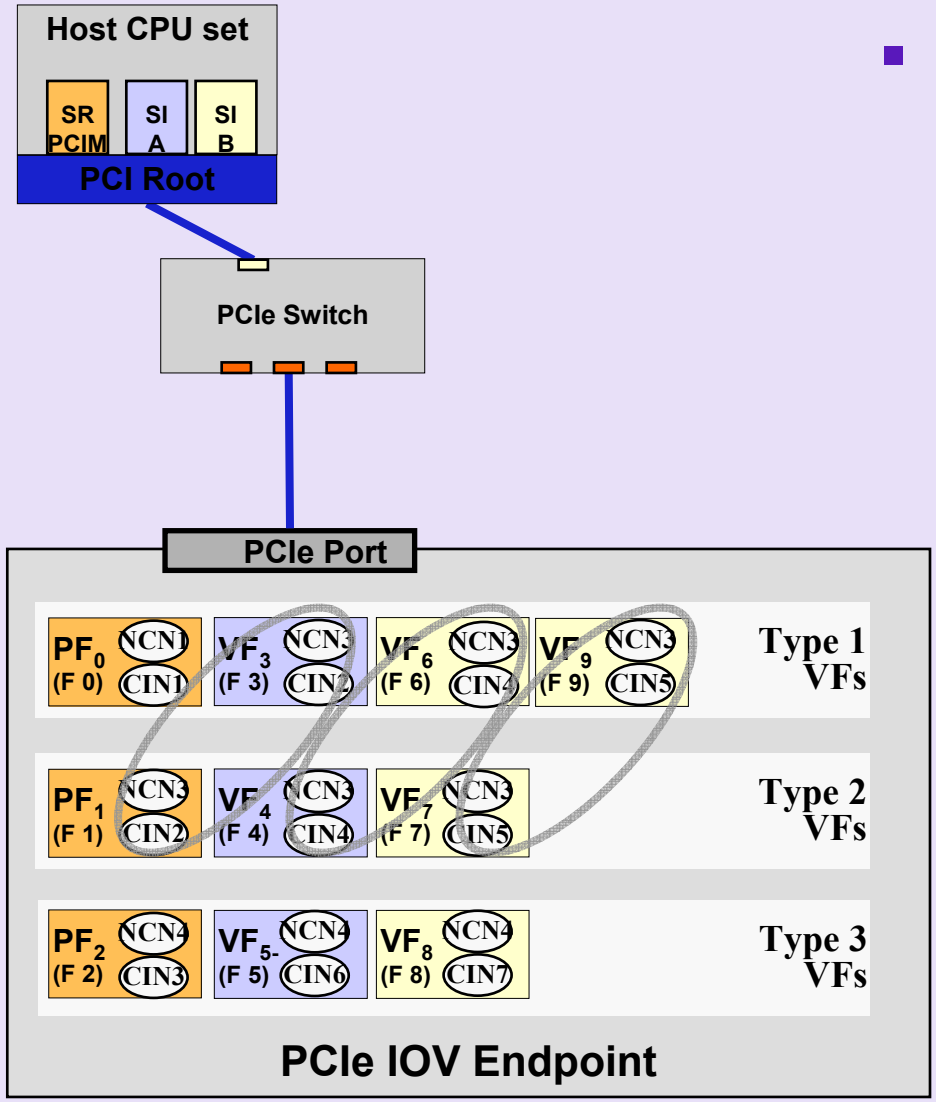
PF_0 PCIe Ext CSR	
:	
Max VFs = 3	Num VFs
NCN = 1	
CIN = 1	
Combo Bitmap = 0001 1110	
:	

PF_0 PCIe Ext CSR	
:	
Max VFs = 2	Num VFs
NCN = 3	
CIN = 2	
:	

PF_0 PCIe Ext CSR	
:	
Max VFs = 2	Num VFs
NCN = 4	
CIN = 3	
:	

- SR-PCIM sets combos for PFs:
  - ✓ Writes to PF_0 CSRs
    - Num VFs = 3
    - NCN = 1
    - CIN =1
  - ✓ Writes to PF_1 CSRs
    - Num VFs = 2
    - NCN = 3
    - CIN = 2
  - ✓ Writes to PF_2 CSRs
    - Num VFs = 2
    - NCN = 4
    - CIN = 3

Example 2 of an EP with interdependent VF Types



- SR-PCIM sets combos for VFs:
  - ✓ Writes to VF_3 CSRs
    - NCN = 3
    - CIN = 2
  - ✓ Writes to VF_4 CSRs
    - NCN = 3
    - CIN = 4
  - ✓ …
  - ✓ Writes to VF_9 CSRs
    - NCN = 4
    - CIN = 7