Cavium RVU ATF documentation

Revision Date	Author	Change
11/17/2017	Konrad Adamczyk	Version v1.1
07/24/2018	Konrad Adamczyk	Version v1.2
11/13/2018	Konrad Adamczyk	Version v1.3
05/09/2019	Tomasz Michalec	Version v1.4

1. Overview 3
2. Provisioning 4
2.1 Legacy Provisioning 5
2.2 Dynamic Provisioning 5
3. RVU types (fixed) 9
4. RVU PFs for LMACs 11
4.1 CGX Alternate RVU PFs 12
5. FDT layout expected by ARM-TF 14
5.1 ADMIN_PF 14
5.2 SSO_TIM_PF 15
5.3 NPA_PF 16
5.4 CPT_PF 17
5.5 LMAC_PF 17
5.6 SDP PF 19
5.7 REE PF 20
6. Layout of MSI-X vectors 20

1. Overview

The Resource Virtualization Unit (RVU) is a new coprocessor in the CN9xxx SoC family. This document describes the RVU initialization/configuration performed by the ARM Trusted Firmware (ARM-TF/ATF).

The RVU supports a number of Physical Functions (RVU PFs) and a number of SRIOV functions (HWVFs). The total number of HWVFs can be distributed across the RVU PFs.

The RVU PFs are allocated to various devices (each supporting an independent number of HWVFs). This process of allocating RVU PFs to various devices is referred to as Provisioning.

The RVU PFs are provisioned to various hardware functions by ARM-TF. There are three categories of devices used in provisioning:

- Fixed
- CGX LMACs
- Dynamic

In general, number of HWVFs allocated to a given RVU PF is defined in FDT. If, for particular RVU node, given property is missing, a default number of HWVFs is set for particular RVU PF. Moreover, same behavior is defined for number of MSI-X vectors for particular PF. Details (for particular PF) are presented in following sections of this document.

ARM-TF allocates contiguous physical memory region for RVU AF/PF and PF/VF mailboxes and for MSI-X vectors. This memory region starts at 16M (0x01000000) and it's size is 40M (0x02800000). The reason why memory region starts at 16M offset is that memory below 16M is allocated for ARM-TF BL31 services (which persist through OS runtime).

Audience of this document is supposed to have experience with:

- Understand RVU architecture for CN96xx (presented in HRM)
- Understand FDT (Flattened Device Tree) concept

References:

- [ATF User Guide] available at https://github.com/ARM-software/arm-trustedfirmware/blob/master/docs/userguide.md

2. Provisioning

The number of RVU PFs supported by the SoC is referred to as <rvu_pf_num> and the final RVU PF ID equals <rvu_pf_num> - 1.

Hence, the range of RVU PFs is defined as:

RVU PF [0 … <last>]

Where:

<last> = <rvu_pf_num> - 1

The value <rvu_pf_num> for each SoC is:

CN96xx: 16

CN98xx: 24

As there is a limited number of RVU PFs, some adjustments can be made to the RVU PF allocations according to the provisioning mode, which is either Legacy or Dynamic.

The allocation of RVU PFs to various devices occurs according to device type and provisioning mode.

- The <fixed> devices are placed at known RVU PFs. These are:
  Administrative Function, SSO_TIM, NPA & CPT
- The <CGX LMAC> devices are placed within a defined range of RVU PFs. Some of these devices can be overridden by <dynamic> devices. In any case, the RVU PFs provisioned for <CGX LMAC> devices are always contiguous.
- The <dynamic> devices are placed starting at a known RVU PF (relative to <rvu_pf_num>). These are:
  SDP, REE
  However, the order and placement of these devices changes depending upon the “provision-mode” property, which can be one of:
  { NONE, LEGACY, AVAILABLE, FORCE }
  Note: LEGACY is applicable ONLY to SDP devices

The default provisioning is shown below.

Table 1: default provisioning of RVU PFs

PF0	Administrative Function	<fixed>
PF1	Start of <CGX LMAC> devices (UP to <last-3>)	<CGX LMAC>/<dynamic>
…		<CGX LMAC>/<dynamic>
…		<CGX LMAC>/<dynamic>
PF<last-3>	End of reserved range for <CGX LMAC> devices Start of <dynamic> devices (DOWN to PF1)	<CGX LMAC>/<dynamic>
PF<last-2>	SSO_TIM	<fixed
PF<last-1>	NPA	<fixed>/<legacy>
PF<last>	CPT	<fixed>

2.1 Legacy Provisioning

This mode allows for a single, optional SDP device to be allocated when PCI Endpoint mode is used.

In this mode, the <fixed> NPA allocation using RVU PF<last-1> is overridden by the SDP device. This depends upon the following:

- A PCI PEM operating in Endpoint mode
- An rvu-sdp entry in the FDT, with “provision-mode” = “LEGACY”

The RVU PF provisioning will be as shown below:

Table 2: Legacy provisioning of RVU PFs

PF0	Administrative Function	<fixed>
PF1	Start of <CGX LMAC> devices (UP to <last-3>)	<CGX LMAC>
…		<CGX LMAC>
…		<CGX LMAC>
PF<last-3>	End of reserved range for <CGX LMAC> devices	<CGX LMAC>
PF<last-2>	SSO_TIM	<fixed
PF<last-1>	SDP	<legacy>
PF<last>	CPT	<fixed>

2.2 Dynamic Provisioning

Unlike legacy, which only accommodates a single, optional SDP device, this mode accommodates any type and number of optional devices (SDP, REE, etc).

NOTE: AT-F defines the count and type of supported devices per platform:

- CN98xx supports two (2) each of SDP and REE
- CN96xx supports one (1) SDP (if not used in Legacy mode)

In order for these devices to be provisioned an RVU PF, there must be a device entry in the DTS file, with an appropriate “provision-mode” property setting.

The provisioning of RVU PFs to the <dynamic> devices depends upon the "provision-mode" property setting:

{ NONE, AVAILABLE, FORCE }

If set to NONE, then no RVU PF will be provisioned to the device.

Otherwise, the provisioning of RVU PFs for the <dynamic> device starts from the TOP of the CGX RVU PFs (i.e. RVU PF<last-3>) and proceeds numerically downward (i.e. <last-4>, <last-5> etc.).

Since the RVU PF range for <dynamic> devices overlaps that of the CGX LMAC devices, it is possible for a collision to occur (see Default Provisioning).

In the case of such a collision, the "provision-mode" property dictates how the RVU PF is provisioned.

The setting of "AVAILABLE" dictates that an RVU PF for the device will only be provisioned IF one is available (i.e. not already assigned to a CGX LMAC). So, in the aforementioned collision scenario, the RVU PF will be provisioned to the CGX LMAC device, not the <dynamic> device.

The setting of "FORCE" dictates that an RVU PF for the <dynamic> device WILL be provisioned. So, in the aforementioned collision scenario, the RVU PF will be provisioned to the <dynamic> device, replacing the CGX LMAC device.

Example 1: CN98xx with twelve (12) CGX LMACs populated and an REE FDT entry present with “provision-mode” = “AVAILABLE”.

Provisioning shown below; two REE devices will be present, as there are PFs available.

PF0	Administrative Function	<fixed>
PF1	<CGX LMAC>	<CGX LMAC>
…	<CGX LMAC>	<CGX LMAC>
PF12	<CGX LMAC>	<CGX LMAC>
…	See <CGX alternate>	
PF19 (i.e. <last-4>)	REE	<dynamic>
PF20 (i.e. <last-3>)	REE	<dynamic>
PF21 (i.e. <last-2>)	SSO_TIM	<fixed
PF22 (i.e. <last-1>)	NPA	<fixed>
PF23 (i.e. <last>)	CPT	<fixed>

Example 2: CN98xx with twelve (12) CGX LMACs populated and an SDP FDT entry present with “provision-mode” = “AVAILABLE”.

Provisioning shown below; two SDP devices will be present, as there are PFs available.

PF0	Administrative Function	<fixed>
PF1	<CGX LMAC>	<CGX LMAC>
…	<CGX LMAC>	<CGX LMAC>
PF12	<CGX LMAC>	<CGX LMAC>
…	See <CGX alternate>	
PF19 (i.e. <last-4>)	SDP	<dynamic>
PF20 (i.e. <last-3>)	SDP	<dynamic>
PF21 (i.e. <last-2>)	SSO_TIM	<fixed
PF22 (i.e. <last-1>)	NPA	<fixed>
PF23 (i.e. <last>)	CPT	<fixed>

Example 3: CN98xx with twelve (12) CGX LMACs populated and FDT entries for both REE and SDP, both with “provision-mode” = “AVAILABLE”.

Provisioning shown below; two REE & SDP devices will be present, as there are PFs available.

PF0	Administrative Function	<fixed>
PF1	<CGX LMAC>	<CGX LMAC>
…	<CGX LMAC>	<CGX LMAC>
PF12	<CGX LMAC>	<CGX LMAC>
…	See <CGX alternate>	
PF17 (i.e. <last-6>)	SDP	<dynamic>
PF18 (i.e. <last-5>)	SDP	<dynamic>
PF19 (i.e. <last-4>)	REE	<dynamic>
PF20 (i.e. <last-3>)	REE	<dynamic>
PF21 (i.e. <last-2>)	SSO_TIM	<fixed
PF22 (i.e. <last-1>)	NPA	<fixed>
PF23 (i.e. <last>)	CPT	<fixed>

Example 4: CN98xx with all CGX LMACs populated and an REE FDT entry present with “provision-mode” = “AVAILABLE”.

Provisioning shown below; no REE devices will be present, as no RVU PFs are available.

PF0	Administrative Function	<fixed>
PF1	<CGX LMAC>	<CGX LMAC>
…	<CGX LMAC>	<CGX LMAC>
…	<CGX LMAC>	<CGX LMAC>
PF20 (i.e. <last-3>)	<CGX LMAC>	<CGX LMAC>
PF21 (i.e. <last-2>)	SSO_TIM	<fixed
PF22 (i.e. <last-1>)	NPA	<fixed>
PF23 (i.e. <last>)	CPT	<fixed>

Example 5: CN98xx with all CGX LMACs populated and an REE FDT entry present with “provision-mode” = “FORCE”.

Provisioning shown below; two (2) REE devices will be present, overriding CGX LMAC devices at PF19-20.

PF0	Administrative Function	<fixed>
PF1	<CGX LMAC>	<CGX LMAC>
…	<CGX LMAC>	<CGX LMAC>
PF18	<CGX LMAC>	<CGX LMAC>
PF19 (i.e. <last-4>)	REE	<dynamic>
PF20 (i.e. <last-3>)	REE	<dynamic>
PF21 (i.e. <last-2>)	SSO_TIM	<fixed
PF22 (i.e. <last-1>)	NPA	<fixed>
PF23 (i.e. <last>)	CPT	<fixed>

3. RVU types (fixed)

ARM-TF software distinguish RVU blocks as follows:

- RVU PF 0 (ADMIN_PF) – PF which has RVU_PRIV_PFX_CFG[AF_ENA] bit set.
  For ADMIN_PF, as for PCI configuration, ARM-TF sets:

Register's field	Value
PCC_XXX_ID[DEVID]<7:0>	PCC_DEV_IDL_E::RVU_AF (0x65)
PCCPF_XXX_REV[BCC,SC,PI]	PCC_DEV_IDL_E::RVU's class code (0x020000)
PCCVF_XXX_REV[BCC,SC,PI]	PCC_DEV_IDL_E::RVU's class code (0x020000)
PCC_XXX_SROIV_DEV[VFDEV]<7:0>*	PCC_DEV_IDL_E::RVU_VF (0x64)*

*if given PF has RVU_PRIV_PFX_CFG[NVF] > 0.

- RVU PF<last-2> (SSO_TIM_PF) – PF which is dedicated for SSO_TIM usage. For SSO_TIM_PF, as for PCI configuration, ARM-TF sets:

Register's field	Value
PCC_XXX_ID[DEVID]<7:0>	PCC_DEV_IDL_E::SW_RVU_SSO_TIM_PF (0xF9)
PCCPF_XXX_REV[BCC,SC,PI]	PCC_DEV_IDL_E::RVU's class code (0x020000)

PCCVF_XXX_REV[BCC,SC,PI]	PCC_DEV_IDL_E::RVU's class code (0x020000)
PCC_XXX_SROIV_DEV[VFDEV]<7:0>*	PCC_DEV_IDL_E::SW_RVU_SSO_TIM_VF (0xFA)*

*if given PF has RVU_PRIV_PFX_CFG[NVF] > 0.

- RVU PF<last-1> (NPA_PF) – PF which is dedicated for NPA usage. For NPA_PF, as for PCI configuration, ARM-TF sets:

Register's field	Value
PCC_XXX_ID[DEVID]<7:0>	PCC_DEV_IDL_E::SW_RVU_NPA_PF (0xFB)
PCCPF_XXX_REV[BCC,SC,PI]	PCC_DEV_IDL_E::RVU's class code (0x020000)
PCCVF_XXX_REV[BCC,SC,PI]	PCC_DEV_IDL_E::RVU's class code (0x020000)
PCC_XXX_SROIV_DEV[VFDEV]<7:0>*	PCC_DEV_IDL_E::SW_RVU_NPA_VF (0xFC)*

*if given PF has RVU_PRIV_PFX_CFG[NVF] > 0.

- RVU PF<last> (CPT_PF) – PF which is dedicated for CPT usage. Note that CPT node has RVU_PRIV_PFX_CFG[AF_ENA] bit set. For CPT_PF, as for PCI configuration, ARM-TF sets:

Register's field	Value

PCC_XXX_ID[DEVID]<7:0>	PCC_DEV_IDL_E::SW_RVU_CPT_PF (0xFD)
PCCPF_XXX_REV[BCC,SC,PI]	CPT_CLASS_CODE (0x108000)
PCCVF_XXX_REV[BCC,SC,PI]	CPT_CLASS_CODE (0x108000
PCC_XXX_SROIV_DEV[VFDEV]<7:0>*	PCC_DEV_IDL_E::SW_RVU_CPT_VF (0xFE)*

*if given PF has RVU_PRIV_PFX_CFG[NVF] > 0.

4. RVU PFs for LMACs

ARM-TF relies on the FDT configuration provided by BDK. ARM-TF assumes that there'll be no more than 4 PHY subnodes of given "cgx@0/1/2" node. RVU PF mapping is defined as one RVU PF for given LMAC.

As CN96xx supports up to 3 CGXs and up to 4 LMACs per CGX, the maximum number of RVU PFs is 12, hence up to 12 RVU PFs will be dedicated to LMACs.

In case that the number of LMACs for all CGXs is less than 12, number of RVU PFs that will be defined for LMACs will be the same as the number of LMACs for all CGXs, as per 1:1 mapping of RVU PF:LMAC.

For those RVU PFs (LMAC_PFs), as for PCI configuration, ARM-TF sets:

Register's field	Value
PCC_XXX_ID[DEVID]<7:0>	PCC_DEV_IDL_E::RVU (0x63)
PCCPF_XXX_REV[BCC,SC,PI]	PCC_DEV_IDL_E::RVU's class code (0x020000)

PCCVF_XXX_REV[BCC,SC,PI]	PCC_DEV_IDL_E::RVU's class code (0x020000)
PCC_XXX_SROIV_DEV[VFDEV]<7:0>*	PCC_DEV_IDL_E::RVU_VF (0x64)*

*if given PF has RVU_PRIV_PFX_CFG[NVF] > 0.

4.1 CGX Alternate RVU PFs

Any unused RVU PFs in the LMAC reserved range PF1…PF<last-3> are referred to as <CGX alternate>. These are defined as SSO_TIM/NPA PFs (see 2. RVU types (fixed)) in a manner that 0.75 of unused RVU PFs will be marked as SSO_TIM PFs and 0.25 of unused PFs will be marked as NPA PFs. The <CGX alternate> RVU PFs (either SSO_TIM or NPA) have the same configuration as listed in section 2. RVU types (fixed), but they do not have any configured VFs.
Following equations are used in calculating number of various RVU PFs:

TOTAL_SSO_TIM_PFS_NO_VFS = 0.75*(**MAX_LMAC_PFS** - **TOTAL_LMAC_PFS**

)

TOTAL_NPA_PFS_NO_VFS = 0.25*(**MAX_LMAC_PFS** - **TOTAL_LMAC_PFS**) where:

MAX_LMAC_PFS = 12 (const)

TOTAL_LMAC_PFS = number of PHY subnodes of "cgx@0/1/2" nodes in FDT

provided by BDK to ARM-TF (see section 4. FDT layout expected by ARM-TF)

Examples:

1. CGX0 has 2 LMACs, CGX1 has 1 LMAC, CGX2 has 1 LMAC RVU PFs configuration is as follows:
   a. RVU PFs 1-4 are configured as LMAC_PF.
      From the left RVU PFs (**12**[MAX_LMAC_PFS] - **4**[TOTAL_LMAC_PFS] = **8**):
   b. RVU PFs 5-10 (**0.75 * 8**[left RVU PFs] = **6**) are configured as SSO_TIM
      PFs (with NumVFs = 0)
   c. RVU PFs 11-12 (**0.25 * 8**[left RVU PFs] = **2**) are configured as
      NPA_PFs (with NumVFs = 0)
   d. Other RVU PFs (0, 13, 14 and 15) are configured as defined in section

2. RVU types - (fixed).

2. CGX0 has no LMACs, CGX1 has 3 LMACs, CGX2 has 2 LMACs RVU PFs configuration is as follows:
    a. RVU PFs 1-5 are configured as LMAC_PF.
       From the left RVU PFs (12 - 5 = 7):
    b. RVU PFs 6-10 are configured as SSO_TIM PFs (with NumVFs = 0)
    c. RVU PFs 11-12 are configured as NPA_PFs (with NumVFs = 0)
    d. Other RVU PFs (0, 13, 14 and 15) are configured as defined in section 2. RVU types - (fixed).
3. CGX0 has 1 LMAC, CGX1 has 4 LMACs, CGX2 has 4 LMACs RVU PFs configuration is as follows:
    a. RVU PFs 1-9 are configured as LMAC_PF.
       From the left RVU PFs (12 - 9 = 3):
    b. RVU PFs 10-11 are configured as SSO_TIM PFs (with NumVFs = 0)
    c. RVU PF 12 is configured as NPA_PFs (with NumVFs = 0)
    d. Other RVU PFs (0, 13, 14 and 15) are configured as defined in section 2. RVU types - (fixed).

5. FDT layout expected by ARM-TF

Besides that ARM-TF configures PCI settings to make it easier for OS to distinguish particular RVU, it gives flexibility to configure:

1. Number of VFs for given RVU PF (via **num-rvu-vfs** property)
2. Number of MSI-X vectors (via **num-msix-vec** property) via proper format of FDT.

Note that ARM-TF does not remove any of the FDT nodes/properties at runtime. ARM-TF treats FDT as read-only blob.

ARM-TF expects that FDT nodes for configuring particular RVU PF as follows:

1. For group of RVU types fixed, all nodes are expected to be placed as subnodes of *ecam2: pci@848020000000* node.
2. For group of RVU PFs for LMACs, additional properties (**num-rvu-vfs** and **num-msix-vec**) are expected to be placed in PHY (e.g. xfi/sgmii/qsgmii/rxaui etc.) nodes, which are actually subnodes of "*cgx@0/1/2*" nodes.

Please note that if sum of all **num-rvu-vfs** properties presented in runtime FDT exceeds number of hardware VFs (256), ARM-TF will report an issue about this incident and will refuse to configure all RVU blocks.

5.1 ADMIN_PF

To set given number of VFs for ADMIN_PF (RVU PF0), the following format of node is expected:

```
ecam2: pci@848020000000 {
        (...)
        /* RVU admin - PF0 */ rvuadmin@0
        { num-rvu-vfs =
        <8>;  num-msix-vec = <133>;
        };
};
```

At boot time (when ARM-TF executes), ARM-TF will parse "**rvu-admin@0** " node properties and program following registers:

Register's field	Value
RVU_PRIV_PF(0)_CFG[NVF]*	From property **num-rvu-vfs***

RVU_PRIV_PF(0)_CFG[FIRST_HWVF]	0
RVU_PRIV_PF(0)_MSIX_CFG[PF_X]	Based on property **num-msix-vec****
RVU_PRIV_PF(0)_MSIX_CFG[VF_X]	198***

*if given property does not exist, ARM-TF will set it to default value, which is specified by DEFAULT_AF_PF0_VFS (0).
**if given property does not exist, ARM-TF will set it to default value, which is specified by DEFAULT_MSIX_AF (37).
***specified by NIX LF MSI-X size (131) plus NPA LF MSI-X size (66) plus number of VF interrupts (1). In future version of RVU ATF, if required, this functionality can be extended to support defining number of MSI-X vectors per VF via FDT.

5.2 SSO_TIM_PF

To set given number of VFs for SSO_TIM_PF (RVU PF13), the following format of node is expected:

```
ecam2: pci@848020000000 {
        (...)
        /* SW_RVU_SSO_TIM - PF13 */ rvu-
        sso-tim@0 { num-rvu-vfs
        = <8>;  num-msix-vec = <128>;
        };
};
```

At boot time (when ARM-TF executes), ARM-TF will parse “**rvu-sso-tim@0**” node properties and program following registers:

Register’s field	Value
RVU_PRIV_PF(13)_CFG[NVF]*	From property **num-rvu-vfs***
RVU_PRIV_PF(13)_CFG[FIRST_HWVF]	RVU_PRIV_PF(0)_CFG[NVF]
RVU_PRIV_PF(13)_MSIX_CFG[PF_X]	Based on property **num-msix-vec**** and already defined MSI-X vectors

RVU_PRIV_PF(13)_MSIX_CFG[VF_X]	Based on property **num-msix-vec****

*if given property does not exist, ARM-TF will set it to default value, which is specified by DEFAULT_VFS (3).

**if given property does not exist, ARM-TF will set it to default value, which is specified by DEFAULT_MSIX_SW (133).

5.3 NPA_PF

To set given number of VFs for NPA_PF (RVU PF14), the following format of node is expected:

```
ecam2: pci@848020000000 {
        (...)
        /* SW_RVU_SSO_NPA - PF14 */ rvu-
        npa@0 { num-rvu-vfs = <8>
        ;
                num-msix-vec = <133>;
        };
};
```

At boot time (when ARM-TF executes), ARM-TF will parse "**rvu-npa@0** " node properties and program following registers:

Register's field	Value
RVU_PRIV_PF(14)_CFG[NVF]*	From property **num-rvu-vfs***
RVU_PRIV_PF(14)_CFG[FIRST_HWVF]	RVU_PRIV_PF(0)_CFG[NVF] + RVU_PRIV_PF(13)_CFG[NVF]
RVU_PRIV_PF(14)_MSIX_CFG[PF_X]	Based on property **num-msix-vec**** and already defined MSI-X vectors
RVU_PRIV_PF(14)_MSIX_CFG[VF_X]	Based on property **num-msix-vec****

*if given property does not exist, ARM-TF will set it to default value, which is specified by DEFAULT_VFS (3).

**if given property does not exist, ARM-TF will set it to default value, which is specified by DEFAULT_MSIX_SW (133).

5.4 CPT_PF

To set given number of VFs for CPT_PF (RVU PF15), the following format of node is expected:

```
ecam2: pci@848020000000 {
        (...)
        /* SW_RVU_CPT - PF15 */ rvu-cpt@0
        { num-rvu-vfs =
        <64>;  num-msix-vec = <133>;
        };
};
```

At boot time (when ARM-TF executes), ARM-TF will parse "**rvu-cpt@0** " node properties and program following registers:

Register's field	Value
RVU_PRIV_PF(15)_CFG[NVF]*	From property **num-rvu-vfs***
RVU_PRIV_PF(15)_CFG[FIRST_HWVF]	RVU_PRIV_PF(0)_CFG[NVF] + RVU_PRIV_PF(13)_CFG[NVF] + RVU_PRIV_PF(14)_CFG[NVF]
RVU_PRIV_PF(15)_MSIX_CFG[PF_X]	Based on property **num-msix-vec**** and already defined MSI-X vectors
RVU_PRIV_PF(15)_MSIX_CFG[VF_X]	Based on property **num-msix-vec****

*if given property does not exist, ARM-TF will set it to default value, which is specified by DEFAULT_VFS (3).
**if given property does not exist, ARM-TF will set it to default value, which is specified by DEFAULT_MSIX_SW (133).

5.5 LMAC_PF

For group of RVU PFs for LMACs, additional properties (**num-rvu-vfs** and **num-msix-vec**) are expected to be placed in PHY (e.g. xfi/sgmii/qsgmii/rxaui etc.) nodes, which are actually subnodes of "*cgx@0/1/2* " nodes. When using default num-rvu-vfs and num-msix-vec values it is expected to don't add these additional properties. To set given number of VFs for given LMAC_PF (RVU PF1-12), the following format of PHY node is expected:

```
&mrml_bridge {
        /*
        *       This configuration is just an example of usage num-rvuvfs.
        * Maximum number of HWVFs is 256, hence sum of all numrvu-vfs
        * cannot exceed 256.
        *       ATF will report an issue at boot time when trying to
        configure  * more than 256 HWVFs.
         */
        cgx@0 { ethernetA0: xfi@00
                {
                        (...)
                        num-rvu-vfs = <8>;  num-msix-vec
                        = <210>;
                };
                ethernetA1: xfi@01 {
                        (...)
                        num-rvu-vfs = <8>;  num-msix-vec
                        = <210>;
                };
                ethernetA2: xfi@02 {
                        (...)
                        num-rvu-vfs = <8>;  num-msix-vec
                        = <210>;
                };
                ethernetA3: xfi@03 {
                        (...)
                        num-rvu-vfs = <8>;  num-msix-vec
                        = <210>;
                };
        };
```

At boot time (when ARM-TF executes), ARM-TF will parse all subnodes of "**cgx@0**", "**cgx@1**" and "**cgx@2**" nodes for properties and program following registers:

Register's field	Value
RVU_PRIV_PF(1..12)_CFG[NVF]*	From property **num-rvu-vfs***
RVU_PRIV_PF(1..12)_CFG[FIRST_HWVF]	Appropriate to previously configured NVFs
RVU_PRIV_PF(1..12)_MSIX_CFG[PF_X]	Based on property **num-msix-vec**** and already defined MSI-X vectors

RVU_PRIV_PF(1..12)_MSIX_CFG[VF_X]	Based on property **num-msix-vec****

*if given property does not exist, ARM-TF will set it to default value, which is specified by DEFAULT_VFS (3).
**if given property does not exist, ARM-TF will set it to default value, which is specified by DEFAULT_MSIX_LMAC (210).

If particular PHY subnode does not exist (for instance, **cgx@0** node has only 2 PHY subnodes and both **cgx@1** and **cgx@2** will have 4 PHY subnodes), as per 3. RVU PFs for LMACs:Examples section, RVU PFs 11-12 will be configured as SSO_TIM:NPA PF in 3:1 proportion, respectively. Those RVU PFs will have 0 configured VFs (fixed).

5.6 SDP PF

This device can be provisioned in either Legacy or Dynamic mode.
The required format is as follows.
NOTE: “provision-mode” must be set appropriately.

```
ecam2: pci@848020000000 {
        (...)
        /* SW_RVU_SDP */
        rvu-sdp@0 {
                num-rvu-vfs = <n>;
                num-msix-vec = <n>;
                provision-mode = “LEGACY”;
        };
};
```

5.7 REE PF

This device can only be provisioned in Dynamic mode.
The required format is as follows.
NOTE: “provision-mode” must be set appropriately.

```
ecam2: pci@848020000000 {
        (...)
        /* SW_RVU_SDP */
        rvu-sdp@0 {
                num-rvu-vfs = <n>;
                num-msix-vec = <n>;
                provision-mode = “AVAILABLE”;
        };
};
```

6. Layout of MSI-X vectors

All MSI-X vectors must be placed within continuous block of memory. Hardware limits number of MSI-X in t RVU_PRIV_CONST[MAX_MSIX]. Each MSI-X vector require 16 bytes of memory. CN96xx supports up to 32,768 vectors which gives 512KB of memory required for MSI-X vectors. ATF allocate MSI-X vectors starting from address 0x0320 0000.

Region	Number of MSI-X vectors
PF0 vectors	RVU_PRIV_PF(0)_MSIX_CFG[PF_X]
PF0 VFs vectors	RVU_PRIV_PF(0)_MSIX_CFG[VF_X] * RVU_PRIV_PF(0)_CFG[NVF]
PF1 vectors	RVU_PRIV_PF(1)_MSIX_CFG[PF_X]
PF1 VFs vectors	RVU_PRIV_PF(1)_MSIX_CFG[VF_X] * RVU_PRIV_PF(1)_CFG[NVF]
...	...
PFx vectors	RVU_PRIV_PF(x)_MSIX_CFG[PF_X]
PFx VFs vectors	RVU_PRIV_PF(x)_MSIX_CFG[VF_X] * RVU_PRIV_PF(x)_CFG[NVF]
...	...
PF15 vectors	RVU_PRIV_PF(15)_MSIX_CFG[PF_X]
PF15 VFs vectors	RVU_PRIV_PF(15)_MSIX_CFG[VF_X] * RVU_PRIV_PF(15)_CFG[NVF]