APPS036-Codensity T408™ - Kubernetes Configuration Application Note

V2.0

2023-02-03

Table of Contents

1 Table of Abbreviations

Abbreviation	Full Form
T408	Codensity T408 Video Transcoder
AI	Artificial Intelligence
API	Application Programming Interface
AUD	Access Unit Delimiters
AV	Audio Video
AV1	Alliance Open Media Video 1 Codec
CABAC	Context Adaptive Binary Arithmetic Coding
CBR	Constant Bit Rate
CPU	Central Processing Unit
CRF	Constant Rate Factor
DSP	Digital Signal Processing
EOF	End Of File
FIXQB	Fixed Quantization Parameter
FF	Form Factor
FPS	Frames Per Second
GOP	Group of pictures
HDR	High Dynamic Range
HHHL	Half Height, Half Length
HHFL	Half Height, Full Length
HLG	Hybrid Log Gamma
HRD	Hypothetical Reference Decoder
IE	Inference Engine
Mbps	Mega Bit per second
MBps	Mega Byte per second
NLP	Natural Language Processing
NPU	Neural Processing Unit
NVMe	Non-Volatile Memory express
PCIe	Peripheral Component Interconnect express
PPS	Picture Parameter Set
QP	Quoted Printable
RGB	Red Green Blue
RGBA	Red Green Blue Alpha
RDO	Optimization
ROI	Region of interest
SDK	Software Development Kit
SEI	Supplemental Enhancement Information

SRIOV	Single Root I/O Virtualization
SPS	Sequence Parameter Set
VBR	Variable Bit Rat
VCL	Video Coding Layer
VPS	Video Parameter Set
VUI	Video Usability Information

2 References

Technical Notes:

Specifications:

3 Background

This document demonstrates the configuration of a T408 pod in a Kubernetes environment.

The installation and configuration of a Linux host is not in scope of this document.

3.1 Intended Audience

This document is intended to help engineers/technicians/developers operating NETINT transcoders better understand the usage of T408 in a Kubernetes environment.

3.2 Compatibility

Software Compatibility

This guide is intended to be used with NETINT Codensity T408 Video Transcoder software Release 2.5.0 or newer.

Hardware Compatibility

Release 2.5.0 or newer supports NETINT Codensity T408 Video Transcoder hardware.

Operation system

All OS supporting T408 Video Transcoder.

4 Pre-requisite T408 Device Plugin

1] T408 NVMe device detected on the Linux host in 'sudo nvme list', ni_rsrc_mon, ni_rsrc_list. Sample output as follows.

sudo nvme list output:

```
Node             SN                   Model                                    Namespace Usage                      Format           FW Rev
---------------- -------------------- ---------------------------------------- --------- -------------------------- ---------------- --------
-
/dev/nvme0n1     TU05-07-02-B04-0959  T408-U2                                  1         536.87  GB / 536.87  GB    512
B + 0 B   250X1F10
```

ni_rsrc_mon output:

```
NI resource init'd already ..
**************************************************
1 devices retrieved from current pool at start up
Mon May  3 22:33:33 2021 up 00:00:00 v250R1F10
Num decodes: 1
BEST INDEX LOAD MODEL_LOAD MEM  INST DEVICE        NAMESPACE
L    0     0    0          0    0    /dev/nvme0    /dev/nvme0n1
Num encodes: 1
BEST INDEX LOAD MODEL_LOAD MEM  INST DEVICE        NAMESPACE
L    0     0    0          0    0    /dev/nvme0    /dev/nvme0n1
**************************************************
```

2] Install Docker module on your Linux Host. Refer links in Section 3.1 for installation steps based on the OS type of your Linux Host.

3] Install Kubernetes on your Linux Host. Refer links in Section 3.1 for installation steps based on the OS type of your Linux Host.

4] Check the status of cluster:

```
$ kubectl get nodes
NAME                STATUS   ROLES                  AGE   VERSION
kubernetes-master   Ready    control-plane,master   45h   v1.21.0
```

Check the status of core pods for kubernetes cluster in kube-system namespace:

```
$ kubectl get pods -n kube-system

NAME                                      READY   STATUS    RESTARTS   AGE
coredns-558bd4d5db-mj669                  1/1     Running   2          45h
coredns-558bd4d5db-rhwwd                  1/1     Running   2          45h
etcd-kubernetes-master                    1/1     Running   2          45h
kube-apiserver-kubernetes-master          1/1     Running   2          45h
kube-controller-manager-kubernetes-master 1/1     Running   1          24h
kube-flannel-ds-8h2xb                     1/1     Running   12         24h
kube-proxy-x69bb                          1/1     Running   1          25h
kube-scheduler-kubernetes-master          1/1     Running   2          45h
```

5] Golang is required to compile the helm chart, which will be deployed into the Kubernetes cluster to detect the T408 device.

Check the golang:

```
$ go version
go version go1.16.4 linux/amd64
```

6] The installation of helm is required.
Check the helm:

```
$ helm version
version.BuildInfo{Version:"v3.5.4",
GitCommit:"1b5edb69df3d3a08df77c9902dc17af864ff05d1",
GitTreeState:"clean", GoVersion:"go1.15.11"}
```

4.1 Useful Links

1] https://linuxhint.com/install_configure_docker_ubuntu/

2] https://docs.docker.com/engine/install/centos/

3] https://download.docker.com/linux/centos/8/x86_64/stable/Packages/

4] https://linuxconfig.org/how-to-install-kubernetes-on-ubuntu-20-04-focal-fossa-linux

5] https://www.tecmint.com/install-a-kubernetes-cluster-on-centos-8/

6] https://kubernetes.io/docs/reference/kubectl/cheatsheet/

7] https://golang.org/dl

8] https://devopscube.com/install-configure-helm-kubernetes/

5 Kubernetes T408 Device Plugin

5.1.1 Introduction

For customers with requirements of cloud deployment, the NETINT T4XX as a hardware device does not support the cloud native deploy method in Kubernetes. To resolve this problem, a Kubernetes T408 device plugin is implemented. This device plugin lists every NETINT T4XX resource as a pod in the host. As a result, the Kubernetes can manage the NETINT T4XX pod as a usual hardware resource to accelerate the video transcoding process.

The purpose of this device plugin is to find the T4XX devices in Kubernetes as a kind of resource. With the device plugin installed on the master node, all the other nodes in the cluster will be capable of recognizing the T4XX devices. A back-end service will continuously run to check the existence of any NETINT T4XX device. After locating the device, the Kubernetes will list the corresponding T4XX device as a hardware resource available to the node. Kubernetes can manage the NETINT T4XX pod as a usual hardware resource to accelerate the video transcoding process.

Note - The document is for demonstration only, there is only one master node with T408 device on it. For Environment with multiple nodes, the steps and pre-requisites are the same. We only need to install the T408 NETINT device plugin on the master node, the Kubernetes will inform other nodes the existence of T408 cards.

5.1.2 How to get the plugin

The plugin is available on Netint's github account in a public repo:
https://github.com/NETINT-Technologies/k8_device_plugin

5.1.3 Installation

Steps to install the Kubernetes device plugin in the master node are as follows.

1] Decompress the zip file 'netint_device_plugin_t408.tar.gz'

```
tar -zxvf netint_device_plugin_t408.tar.gz
```

2] Run the make build in the directory to compile.

```
cd netint-device-plugin_release
make build
```

3] This step will build the docker image according to the previous compile results. This image will be used for kubernetes plugin.

```
make buildImage
```

4] Use the helm to install the netint image. The privilege mode will be required for this image when it is installed in kubernetes. Make sure the kubelet enables privilege mode during start-up.

```
make deploy
```

After the plugin is installed, every NETINT transcoder card will be shown as a hardware resource. The name will be shown as netint.ca/ASIC in every node (The master node will let the slave nodes know the information of NETINT device). To implement a customized device plugin, the file pod-with-netint.yaml can be used as a reference. It is based on the NETINT 2.0.0 release pod.

5] Run the following commands to confirm netint plugin is installed correctly.

Check the daemonset to see if the daemonset was deployed.

```
$ kubectl get daemonset -n kube-system
NAMESPACE      NAME                DESIRED    CURRENT    READY    UP-TO-DATE
AVAILABLE    NODE SELECTOR              AGE
kube-system    kube-proxy          1          1          1        1
1            kubernetes.io/os=linux    45h
kube-system    netint              1          1          1        1
1            netint-device=enable      2m14s
```

Check the netint pod see if it’s in healthy status.

```
kubectl get pods -n kube-system
$ kubectl get pods -n kube-system
NAME                                        READY    STATUS     RESTARTS    AGE
coredns-558bd4d5db-mj669                    1/1      Running    2           45h
coredns-558bd4d5db-rhwwd                    1/1      Running    2           45h
etcd-kubernetes-master                      1/1      Running    2           45h
kube-apiserver-kubernetes-master            1/1      Running    2           45h
kube-controller-manager-kubernetes-master   1/1      Running    1           24h
kube-flannel-ds-8h2xb                       1/1      Running    12          24h
kube-proxy-x69bb                            1/1      Running    1           25h
kube-scheduler-kubernetes-master            1/1      Running    2           45h
netint-gl9bv                                1/1      Running    0           78s
```

Check the netint container to see if it is created and running. Usually each NETINT T408 device will have a corresponding pod created by the device plugin.

```
$ sudo docker ps -a |grep netint
```

```
3eedb5d249f6   e7b97afa1ecf            "/root/netint" 5 minutes ago    Up 5 minutes
k8s_netint_netint-gl9bv_kube-system_55f8ccef-9cc6-47ad-ab1c-93c3bcd0b20e_0
f334ccfb00da   k8s.gcr.io/pause:3.4.1   "/pause"       5 minutes ago    Up 5 minutes
k8s_POD_netint-gl9bv_kube-system_55f8ccef-9cc6-47ad-ab1c-93c3bcd0b20e_0
```

6] In some environment the Kubernetes might be unaware of the T408 device even after the T408 device plugin is installed successfully. We must label the node as "netint device enabled to force the pod starts.

```
kubectl label nodes "node-name" netint-device=enable
```

5.1.4 Pod Status Check

Check using 'kubectl describe pod <podname> -n <namespace>'

```
kubectl get pods --all-namespaces
NAMESPACE     NAME                                        READY   STATUS
RESTARTS   AGE
kube-system   coredns-558bd4d5db-mj669                    1/1     Running   2
7d7h
kube-system   coredns-558bd4d5db-rhwwd                    1/1     Running   2
7d7h
kube-system   etcd-kubernetes-master                      1/1     Running   2
7d7h
kube-system   kube-apiserver-kubernetes-master            1/1     Running   2
7d7h
kube-system   kube-controller-manager-kubernetes-master   1/1     Running   1
6d11h
kube-system   kube-flannel-ds-8h2xb                       1/1     Running
12         6d11h
kube-system   kube-proxy-x69bb                            1/1     Running   1
6d11h
kube-system   kube-scheduler-kubernetes-master            1/1     Running   2
7d7h
kube-system   netint-gl9bv                                1/1     Running   1
5d10h
```

```
kubectl describe pod netint-gl9bv -n kube-system
Name:                 netint-gl9bv
Namespace:            kube-system
Priority:             2000001000
Priority Class Name:  system-node-critical
Node:                 kubernetes-master/10.0.131.88
Start Time:           Wed, 12 May 2021 23:59:25 -0700
Labels:               app.kubernetes.io/instance=netint
                      app.kubernetes.io/name=netint
```

```
                      controller-revision-hash=5fc6cd6997
                      pod-template-generation=1
Annotations:          scheduler.alpha.kubernetes.io/critical-pod:
Status:               Running
IP:                   10.244.0.11
IPs:
  IP:           10.244.0.11
Controlled By:  DaemonSet/netint
Containers:
  netint:
    Container ID:
docker://5cf315da71e361dd64150861a9f9cc99bcb7a395529719d65c38c633e955be6b
    Image:          netint/device-plugin:1.4
    Image ID:
docker://sha256:e7b97afa1ecfe98b3414c7b80439d33a8428a2d4332fdefb2208b70e891c6
2db
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Mon, 17 May 2021 20:33:50 -0700
    Last State:     Terminated
      Reason:       Completed
      Exit Code:    0
      Started:      Wed, 12 May 2021 23:59:30 -0700
      Finished:     Mon, 17 May 2021 20:33:47 -0700
    Ready:          True
    Restart Count:  1
    Environment:    <none>
    Mounts:
      /dev from devfs (rw)
      /sys from sys (rw)
      /var/lib/kubelet/device-plugins from device-plugin (rw)
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-
4m9m6 (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  device-plugin:
    Type:          HostPath (bare host directory volume)
    Path:          /var/lib/kubelet/device-plugins
    HostPathType:  Directory
  devfs:
    Type:          HostPath (bare host directory volume)
    Path:          /dev
    HostPathType:  Directory
  sys:
    Type:          HostPath (bare host directory volume)
    Path:          /sys
```

```
    HostPathType:  Directory
  kube-api-access-4m9m6:
    Type:                    Projected (a volume that contains injected data
from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              netint-device=enable
Tolerations:                 CriticalAddonsOnly op=Exists
                             netint.ca/ASIC:NoSchedule op=Exists
                             node.kubernetes.io/disk-pressure:NoSchedule
op=Exists
                             node.kubernetes.io/memory-pressure:NoSchedule
op=Exists
                             node.kubernetes.io/not-ready:NoExecute op=Exists
                             node.kubernetes.io/pid-pressure:NoSchedule
op=Exists
                             node.kubernetes.io/unreachable:NoExecute
op=Exists
                             node.kubernetes.io/unschedulable:NoSchedule
op=Exists
Events:
  Type    Reason   Age                     From     Message
  ----    ------   ----                    ----     -------
  Normal  Pulled   5m13s (x2 over 4d20h)  kubelet  Container image
"netint/device-plugin:1.4" already present on machine
  Normal  Created  5m11s (x2 over 4d20h)  kubelet  Created container netint
  Normal  Started  5m11s (x2 over 4d20h)  kubelet  Started container netint
nvme@kubernetes-master:~$ kubectl describe pod netint-gl9bv -n kube-system
Name:                 netint-gl9bv
Namespace:            kube-system
Priority:             2000001000
Priority Class Name:  system-node-critical
Node:                 kubernetes-master/10.0.131.88
Start Time:           Wed, 12 May 2021 23:59:25 -0700
Labels:               app.kubernetes.io/instance=netint
                      app.kubernetes.io/name=netint
                      controller-revision-hash=5fc6cd6997
                      pod-template-generation=1
Annotations:          scheduler.alpha.kubernetes.io/critical-pod:
Status:               Running
IP:                   10.244.0.11
IPs:
  IP:           10.244.0.11
Controlled By:  DaemonSet/netint
Containers:
  netint:
    Container ID:
docker://5cf315da71e361dd64150861a9f9cc99bcb7a395529719d65c38c633e955be6b
    Image:          netint/device-plugin:1.4
```

```
    Image ID:
docker://sha256:e7b97afa1ecfe98b3414c7b80439d33a8428a2d4332fdefb2208b70e891c6
2db
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Mon, 17 May 2021 20:33:50 -0700
    Last State:     Terminated
      Reason:       Completed
      Exit Code:    0
      Started:      Wed, 12 May 2021 23:59:30 -0700
      Finished:     Mon, 17 May 2021 20:33:47 -0700
    Ready:          True
    Restart Count:  1
    Environment:    <none>
    Mounts:
      /dev from devfs (rw)
      /sys from sys (rw)
      /var/lib/kubelet/device-plugins from device-plugin (rw)
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-
4m9m6 (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  device-plugin:
    Type:          HostPath (bare host directory volume)
    Path:          /var/lib/kubelet/device-plugins
    HostPathType:  Directory
  devfs:
    Type:          HostPath (bare host directory volume)
    Path:          /dev
    HostPathType:  Directory
  sys:
    Type:          HostPath (bare host directory volume)
    Path:          /sys
    HostPathType:  Directory
  kube-api-access-4m9m6:
    Type:                    Projected (a volume that contains injected data
from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              netint-device=enable
Tolerations:                 CriticalAddonsOnly op=Exists
                             netint.ca/ASIC:NoSchedule op=Exists
```

```
                             node.kubernetes.io/disk-pressure:NoSchedule
op=Exists
                             node.kubernetes.io/memory-pressure:NoSchedule
op=Exists
                             node.kubernetes.io/not-ready:NoExecute op=Exists
                             node.kubernetes.io/pid-pressure:NoSchedule
op=Exists
                             node.kubernetes.io/unreachable:NoExecute
op=Exists
                             node.kubernetes.io/unschedulable:NoSchedule
op=Exists
Events:
  Type    Reason   Age                    From     Message
  ----    ------   ----                   ----     -------
  Normal  Pulled   6m58s (x2 over 4d20h)  kubelet  Container image
"netint/device-plugin:1.4" already present on machine
  Normal  Created  6m56s (x2 over 4d20h)  kubelet  Created container netint
  Normal  Started  6m56s (x2 over 4d20h)  kubelet  Started container netint
```

6 Revision History

Version	Change	Create Date	Author
0.1	Document Created	May 03, 2021	Balwinder Sidhu
1.0	Kubernetes details. Issued version.	May 18th, 2021	Balwinder Sidhu Weiwen Zhao
2.0	Updated as per new app note template	December 9, 2021	Balwinder Sidhu
3.0	Updated to reference github repo for the Device-plugin	Feb 3 2023	K.Robinson

7 Legal Notice

Information in this document is provided in connection with NETINT products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in NETINT's terms and conditions of sale for such products, NETINT assumes no liability whatsoever and NETINT disclaims any express or implied warranty, relating to sale and/or use of NETINT products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right.

A "Mission Critical Application" is any application in which failure of the NETINT Product could result, directly or indirectly, in personal injury or death. Should you purchase or use NETINT's products for any such mission critical application, you shall indemnify and hold NETINT and its subsidiaries, subcontractors and affiliates, and the directors, officers, and employees of each, harmless against all claims costs, damages, and expenses and reasonable attorney's fees arising out of, directly or indirectly, any claim of product liability, personal injury, or death arising in any way out of such mission critical application, whether or not NETINT or its subcontractor was negligent in the design, manufacture, or warning of the NETINT product or any of its parts.

NETINT may make changes to specifications, technical documentation, and product descriptions at any time, without notice. The information here is subject to change without notice. Do not finalize a design with this information. The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications.

NETINT, Codensity, and NETINT Logo are trademarks of NETINT Technologies Inc. All other trademarks or registered trademarks are the property of their respective owners.