Using FPGA On YARN

Prerequisites

- The FPGA resource is supported by YARN but only shipped with “IntelFpgaOpenclPlugin” for now
- YARN node managers have to be pre-installed with vendor drivers and configured with needed environment variables
- Docker support is not supported yet

Configs

FPGA scheduling

In `resource-types.xml`

Add following properties

```
<configuration>
  <property>
     <name>yarn.resource-types</name>
     <value>yarn.io/fpga</value>
  </property>
</configuration>
```

For `Capacity Scheduler`, `DominantResourceCalculator` MUST be configured to enable FPGA scheduling/isolation.
Use following property to configure `DominantResourceCalculator` (In `capacity-scheduler.xml`):

Property	Default value
yarn.scheduler.capacity.resource-calculator	org.apache.hadoop.yarn.util.resource.DominantResourceCalculator

FPGA Isolation

In `yarn-site.xml`

```
  <property>
    <name>yarn.nodemanager.resource-plugins</name>
    <value>yarn-io/fpga</value>
  </property>
```

This is to enable FPGA isolation module on NodeManager side.

By default, YARN will automatically detect and config FPGAs when above config is set. Following configs need to be set in `yarn-site.xml` only if admin has specialized requirements.

1) Allowed FPGA Devices

Property	Default value

Property	Default value
yarn.nodemanager.resource-plugins.fpga.allowed-fpga-devices	auto

Specify FPGA devices which can be managed by YARN NodeManager, split by comma
Number of FPGA devices will be reported to RM to make scheduling decisions.
Set to auto (default) let YARN automatically discover FPGA resource from system.

Manually specify FPGA devices if admin only want subset of FPGA devices managed by YARN.
At present, since we can only configure one major number in c-e.cfg, FPGA device is identified by their minor device number. For Intel devices, a common approach to get minor device number of FPGA is using “aocl diagnose” and check uevent with device name.

2) Executable to discover FPGAs

Property	Default value
yarn.nodemanager.resource-plugins.fpga.path-to-discovery-executables	

When yarn.nodemanager.resource.fpga.allowed-fpga-devices=auto specified, YARN NodeManager needs to run FPGA discovery binary (now only support IntelFpgaOpenclPlugin) to get FPGA information.
When value is empty (default), YARN NodeManager will try to locate discovery executable from vendor plugin’s preference. For instance, the “IntelFpgaOpenclPlugin” will try to find “aocl” in directory got from environment “ALTERAOCLSDKROOT”

3) FPGA plugin to use

Property	Default value
yarn.nodemanager.resource-plugins.fpga.vendor-plugin.class	org.apache.hadoop.yarn.server.nodemanager.containermanager.resourceplugin.fpga.IntelFpgaOpenclPlugin

For now, only Intel OpenCL SDK for FPGA is supported. The IP program(.aocx file) running on FPGA should be written with OpenCL based on Intel platform.

4) CGroups mount
FPGA isolation uses CGroup devices controller to do per-FPGA device isolation. Following configs should be added to `yarn-site.xml` to automatically mount CGroup sub devices, otherwise admin has to manually create devices subfolder in order to use this feature.

Property	Default value
yarn.nodemanager.linux-container-executor.cgroups.mount	true

For more details of YARN CGroups configurations, please refer to Using CGroups with YARN

In `container-executor.cfg`

In general, following config needs to be added to `container-executor.cfg`. The fpag.major-device-number and allowed-device-minor-numbers are optional allowed devices.

```
[fpga]
module.enabled=true
fpga.major-device-number=## Major device number of FPGA, by default is 246. Strongly recommend setting this
fpga.allowed-device-minor-numbers=## Comma separated allowed minor device numbers, empty means all FPGA devices man
```

When user needs to run FPGA applications under non-Docker environment:

```
[cgroups]
# Root of system cgroups (Cannot be empty or "/")
root=/cgroup
```

```
# Parent folder of YARN's CGroups
yarn-hierarchy=yarn
```

Use it

Distributed-shell + FPGA

Distributed shell currently support specify additional resource types other than memory and vcores

Run distributed shell without using docker container (the .bashrc contains some SDK related environment variables):

```
yarn jar <path/to/hadoop-yarn-applications-distributedshell.jar> \
  -jar <path/to/hadoop-yarn-applications-distributedshell.jar> \
  -shell_command "source /home/yarn/.bashrc && aocl diagnose" \
  -container_resources memory-mb=2048,vcores=2,yarn.io/fpga=1 \
  -num_containers 1
```

You should be able to see output like

```
aocl diagnose: Running diagnose from /home/fpga/intelFPGA_pro/17.0/hld/board/nalla_pcie/linux64/libexec

------------------------- acl0 -------------------------
Vendor: Nallatech ltd

Phys Dev Name  Status   Information

aclnalla_pcie0Passed   nalla_pcie (aclnalla_pcie0)
                       PCIe dev_id = 2494, bus:slot.func = 02:00.00, Gen3 x8
                       FPGA temperature = 54.4 degrees C.
                       Total Card Power Usage = 32.4 Watts.
                       Device Power Usage = 0.0 Watts.

DIAGNOSTIC_PASSED
---------------------------------------------------------
```

Specify IP that YARN should configure before launch container

For FPGA resource, the container can have an environment variable “REQUESTED_FPGA_IP_ID” to make YARN download and flash an IP for it before launch.
For instance, REQUESTED_FPGA_IP_ID=“matrix_mul” will lead to a searching in container’s local directory for IP file(".aocx" file) whose name contains “matirx_mul” (the application should distribute it first).
We only support flashing one IP for all devices for now. If user don’t set this environment variable, we assume that user’s application can find the IP file by itself.
Note that the IP downloading and reprogramming in advance in YARN is not necessary because
the OpenCL application may find the IP file and reprogram device on the fly. But YARN do this
for the containers will achieve the quickest re-programming path.