How to build AGL Basesystem for VirtIO

This section describes the steps required to build the AGL Basesystem for VirtIO.

1. Download the AGL recipe files

```
$ repo init -u https://gerrit.automotivelinux.org/gerrit/AGL/AGL-repo
$ repo sync
```

2. Setup the build environment

```
$ source meta-agl/scripts/aglsetup.sh -f -m virtio-aarch64 -b build-virtio-aarch64 agl-demo agl-basesystem
```

3. Build the AGL Basesystem Image

```
$ bitbake agl-image-boot-basesystem
```

4. Deploying the AGL Basesystem Image

This subsection describes AGL virtio-aarch64 image deployment under virtio platform provided by QEMU aarch64 emulator on PC, or QEMU/KVM hypervisor on AGL Reference Hardware board.

4.1 QEMU on PC

If shell from which AGL was built is closed, or new shell is opened, then it is needed to re-initialize build environment:

```
$ source $AGL_TOP/build-virtio-aarch64/agl-init-build-env
$ runqemu
```

4.2 QEMU/KVM on AGL Reference Hardware

Follow these steps to run virtual AGL Basesystem on bare-metal AGL on AGL Reference Hardware board:

4.2.1 Partition eMMC or SD-Card to have two partitions, at least 1 GiB each.

Actually, can be less but just rounded up to have a nice number.
For example, SD-Card can be partitioned using fdisk utility.

4.2.2 Flash AGL minimal image root file system to the second partition on SD-Card or eMMC.

Ex. SD-Card (/dev/sdb)

```
$ cd build-virtio-aarch64/tmp/deploy/images/virtio-aarch64
$ sudo dd if=agl-image-boot-basesystem-virtio-aarch64.ext4 of=/dev/sdb2
```

4.2.3 Build AGL minimal image for AGL Reference Hardware.

```
$ source meta-agl/scripts/aglsetup.sh -m h3ulcb -b build-h3ulcb agl-demo agl-refhw-h3
```

In build-h3ulcb/conf/local.conf add

```
AGL_DEFAULT_IMAGE_FSTYPES = "ext4"
IMAGE_INSTALL_append = " \
                    qemu \
              util-linux \          // needed to enable kvm on qemu
"
```

Build image:

```
$ bitbake agl-image-minimal
```

Add virtio kernel to the AGL Reference Hardware Linux rootfs:

```
$ cp build-virtio-aarch64/tmp/deploy/images/virtio-aarch64/Image build-h3ulcb/tmp/work/h3ulcb-agl-
linux/agl-image-minimal/1.0-r0/rootfs/linux2
$ bitbake agl-image-minimal -c image_ext4 -f
$ bitbake agl-image-minimal -c image_complete
```

Flash root file system to the first partition on SD-Card or eMMC.
Ex. SD-Card (/dev/sdb)

```
$ cd build-h3ulcb/tmp/deploy/images/h3ulcb
$ sudo dd if=agl-image-minimal-h3ulcb.ext4 of=/dev/sdb1
```

4.2.4 Boot AGL Reference Hardware board using Linux located on the first partition of SD-Card or eMMC.

4.2.5 Run QEMU from Linux 1 command line

Ex.

```
taskset f qemu-system-aarch64 \
    -machine virt \
    -cpu cortex-a57 \
    -m 2048 \
    -serial mon:stdio \
    -global virtio-mmio.force-legacy=false \
    -drive id=disk0,file=/dev/mmcblk1p2,if=none,format=raw \
    -device virtio-blk-device,drive=disk0 \
    -object rng-random,filename=/dev/urandom,id=rng0 \
    -device virtio-rng-device,rng=rng0 \
    -nographic \
    -kernel /linux2 \
    -append 'root=/dev/vda rw mem=2048M' \
    -enable-kvm \
    -device virtio-mouse-device \
    -object input-linux,id=mouse1,evdev=/dev/input/by-path/platform-ee080000.usb-usb-0:1:1.0-event-
mouse \
    -device virtio-net-device,netdev=net0 \
    -netdev user,id=net0,net=192.168.10.0/24
```

NOTE: mmcblk1p2 above is used for when root file system is flashed on SD-Card.
NOTE: To enable KVM using -enable-kvm option, use taskset command to bind a process to a given set of CPUs on the system.