Zedroid OS installation steps

Note: the installation procedure uses a lot of custom scripts, which have been written for an Ubuntu host system. The scripts are no more than a list of shell commands, and can be manually executed accordingly on any other host OS.

- All scripts have been added to the `scripts/` folder. Make sure you are in correct folder before running (`~/zedroid` by default).
- Whenever a file has to be edited, there is a final version of that file in `scripts/` folder, which can be referred to in case of doubt. Please manually edit all source files, and don't copy the provided files.
- If you want to follow whole procedure in another folder, copy `scripts/` folder there and follow installation from that folder only.

1. Install dependencies

Run:

```
$ scripts/install-deps.sh
```

Installs required libraries and tools, JDK8, repo. You can use it on another host OS by changing `apt-get` to the relevant tool. Some package names may not match, and will have to be manually checked.

2. Fetch sources

Run:

```
$ scripts/fetch-sources.sh
```

Fetches all sources for Android 5. It will take a long time, around 25-30GB of source is fetched. The sources are stored in `android_sources/` folder.

If this command fails, run:

```
cd android_source
repo sync
```

3. Modify source files

Change dir to `android_source` before editing any file. All file paths are relative to that folder.

3.1 Engineering build

- Open `device/xilinx/zedboard/vendorsetup.sh'
- Add these lines at start of file (after comment block)

```
# Engineering build config
add_lunch_combo zedboard-eng
```

3.2 Enable ethernet in init.rc

- Open file `device/xilinx/zedboard/init.xilinxzynqplatform.rc`
- Add these lines at the very end of the file

```
service ethernet /eth0init.sh
    class main
    user root
    group root
    oneshot
```

3.3 Add eth0init.sh to sources

- You will find `eth0init.sh` in `scripts/` folder
- Copy `eth0init.sh` to `device/xilinx/zedboard/`
- Open `device/xilinx/zedboard/device.mk`
- Add following line to the end of the file

```
PRODUCT_COPY_FILES += device/xilinx/zedboard/eth0init.sh:root/eth0init.sh
```

4. Make Android 5 OS

Make sure you are in `android_source` directory.

Run:

```
$ source ./build/envsetup.sh
$ lunch zedboard-eng
$ make -k 2> make-errors.log | tee make.log
```

Some components of the OS may fail to build. But in that case too, we can obtain a working version of the OS. For that `-k` flag is used to continue making even with errors.

To see what failed, see `make-errors.log` file.

Troubleshoot: if errors are regarding syntax errors in python scripts, make sure the `python` *executable in your PATH is* `python2` *and not* `python3`*. Check output of* `python -V` *to make sure, and relink* `/usr/bin/python` *to correct version of python. Then run* `make` *again.*

4.1 Verify changes propagated properly

Change to `android_source` directory

- In directory `out/target/product/zedboard/`, you should see 2 files
  - uImage
  - ramdisk.img
- In same directory, there should be `system/` and `data/` directories containing android root.
- There should also be 'root/' directory which contains ramdisk root. Verify that the changes you made to `init.xilinxzynqplatform.rc` are there and `eth0init.sh` exists.
- If it doesn't exist you have to discard ramdisk.image and repack manually. Else, skip the next section

4.2 Repack ramdisk manually

Change to `android_source/out/target/product/zedboard/' directory

Make the changes said in (3.2) and (3.3) in the `root/` directory directly

Run: from `zedboard/` directory

```
$ cd root && find . | cpio -o -H newc | gzip > ../ramdisk.img && cd ..
```

4.3 Adding uboot header to ramdisk

After obtaining proper `ramdisk.image`

Run:

```
$ mkimage -A arm -O linux -T ramdisk -C none \
    -a 0x02000000 \
    -n 'Zedroid ramdisk' \
    -d ramdisk.img uramdisk.image.gz
```

5. Preparing boot medium

You will need an SD-Card of atleast 2GB, 4GB is preferable.

You need to create 4 partitions as listed in the same order with same partition type. Using GParted is most suitable.

- partition 1: FAT32 - 500MB - Label="boot"
- partition 2: EXT4 - 500MB - Label="system"
- partition 3: EXT4 - use remaining space on card, minimum 500MB - Label="data"
- partition 4: SWAP - 500MB - Label="swap"

It is important to keep order and type of partitions as mentioned above. The mounting of these is hardcoded into u-boot and boot will fail otherwise.

After successfully creating partitions on SD-Card, we will copy over the OS files.

Mount on host system

Make sure you are in `zedroid/` folder (parent of `android_source/`)

Also check which device is SD-Card by running `lsblk`. In instructions `/dev/sdX` would stand for SD-Card. It is generally `/dev/sdb`

Run:

```
mkdir -vp mnt/{boot,system,data}
sudo mount /dev/sdX1 mnt/boot
sudo mount /dev/sdX2 mnt/system
sudo mount /dev/sdX3 mnt/data
```

Copy boot binaries

In `scripts/` folder you will find `boot.bin` and `devicetree.dtb`.

For all these operations you will have to use `sudo cp` command to copy

- In `mnt/boot/`, copy
  - boot.bin
  - devicetree.dtb
  - uImage
  - uramdisk.image.gz
- In `mnt/system/`, copy
  - All contents of `android_source/out/target/product/zedboard/system/` directory
- In `mnt/data/`, copy
  - All contents of `android_source/out/target/product/zedboard/data/` directory

Then run:

```
sudo umount mnt/{boot,system,data}
```

It may take a while to unmount all the partitions.

You now have a ready-to-boot SD card with Android 5 OS.

6. Booting up

- Insert SD-Card into Zedboard and power-up.

- Connect to USB-UART port on the zedboard.
- Start serial terminal like `minicom` using 115200 baud 8N1 configuration (generally default). Running `minicom -D /dev/ttyACM0 -w` or respective device name should work. `-w` flag is needed for line-wrap to work.

You should either boot directly into Android and see linux terminal, or `zynq-uboot>` terminal of u-boot.

6.1 Properly configuring U-Boot

If you are stuck at u-boot terminal, then follow these steps to boot into Android.

```
zynq-uboot> env print sdboot
zynq-uboot> env set sdboot "mmcinfo && fatload mmc 0 0x3000000 ${kernel_image} &&
fatload mmc 0 0x2000000 ${ramdisk_image} && fatload mmc 0 0x2A00000
${devicetree_image} && bootm 0x3000000 0x2000000 0x2A00000"
zynq-uboot> env save
```

Now close minicom, and restart the Zedboard. You should boot into Android.

6.2 Connecting to ADB

Once you have boot into android and have access to the shell via minicom, connect ethernet and run `ip addr`.

- If the status if `eth0` is `DOWN` then ethernet wasn't correctly enabled.
- run `cat /init.xilinxzynqplatform.rc` and see if the lines added are present.
- Also verify that `eth0init.sh` is there and correct.
- run `dmesg | grep eth` to see if ethernet service was started, and if there were any errors.

You will have to recheck the ramdisk image if new changes are not present.

Once you have a proper ramdisk image, you will see that `eth0` is `UP` at boot and also be able to know IP. For easier use, you can also scan IP from your host using `nmap -p5555 192.168.1.1/24` or equivalent IP subnet. Port 5555 is used by ADB so whichever host shows it "open" is zedboard.

You can now connect to zedboard root shell via

```
$ adb connect <ip-of-zedboard>
$ adb shell
```