LUCY-II-yocto-odyssey - Custom Seeed STM32MP157C SOM Yocto Project

Directory Structure

- `.vscode/`
  - Settings for the IDE
  - Action buttons are used to better control the project
- `build-mp1/`
  - `conf/`
    - bitbake included layers and local user configurations for build environment
      - `bblayers.conf` & `local.conf`
  - `tmp/`
    - build system's output
  - `tmp/deploy/images/stm32mp1/`
    - end result output binaries used for flashing
  - `tmp/work-shared/stm32mp1/`
    - contains shared device tree recipes
  - `tmp/work/stm32mp1-poky-linux-gnueabi/custom-image/1.0-r0/rootfs/`
    - structure of working linux directory on the custom Seeed stm32mp157c SOM
- `meta-custom/`
  - Custom Linux application layer
  - `conf/`
  - Contains the core set of configuration files that start from bitbake.conf
  - `recipes-bsp/`
  - tfa and u-boot device tree patches
  - `recipes-kernal/`
  - Linux kernal version 5.10.10 device tree patches
- `meta-st-odyssey/`
  - Custom Seeed SOM stm32mp157c BSP layer
  - `conf/`
  - Contains the core set of configuration files that start from bitbake.conf
  - `recipes-apps/`
  - Application binaries built into our linux distribution
- `meta-st-stm32mp/`
  - Default stm32mp1 BSP layer
- `meta-openembedded/`
  - Open embedded core layer
- `meta-python2/`
  - The dunfell branch from meta-st needs the legacy python 2 layer
- `poky/`
  - Reference distribution that includes the metadata and useful tools for building the yocto project
- `SDK/`
  - Software environment framework for developing Embedded Linux applications for the Dual Core Cortex-A7 on the stm32mp157c

- notes/
  - Testing booting notes

Building Yocto Project

```
## 1. Enter the Yocto directory
-- Open a new terminal
>> cd ~/Desktop/Embedded_Systems/Software/apps/LUCY-II-yocto-odyssey/

## 2. Source Bitbake Environment
-- (menuconfig) >> bitbake -c menuconfig virtual/kernel
>> source poky/oe-init-build-env build-mp1

## 3. Build Custom Image
>> bitbake custom-image
```

Flashing Custom Yocto Project to eMMC

```
## 1. Connect Micro USB Cable to USB PROG and USB SERIAL
-- Open a new terminal (UART4)
>> minicom -D /dev/ttyACM0 115200

## 2. Enter the deploy binary directory
-- Open a new terminal (USB1)
>> cd ~/Desktop/Embedded_Systems/Software/apps/LUCY-II-yocto-odyssey/build-mp1/tmp/deploy/images/stm32mp1/

## 3. Export the STM32CubeProgrammer variable
>> export PATH=~/STMicroelectronics/STM32Cube/STM32CubeProgrammer/bin/:$PATH

## 4. Flash eMMC partitions
-- Press the reset button if fails
-- Connect BOOT MODE Header Jumper & Press the RST BTN (the Red (DL7) Led should be off)
-- DL7 shouldn't be Flashing or ON if the USB PROG cable is connected (push RST BTN when cable connected)
>> STM32_Programmer_CLI -c port=usb1 -w flashlayout_custom-image/trusted/FlashLayout_emmc_stm32mp157c-odyssey-trusted.tsv

## 5.1 In the UART4 terminal, enter u-boot and select the stm32mp1 cli programmer to connect with the stm32mp157c
-- Open the UART4 terminal
-- Enter u-boot by hitting any key until you enter (i.e., STM32MP>)
STM32MP> stm32prog usb 0

### 5.2 Option two mounting (copy partitions using 'dd') - opens eMMC as a storage device like writing to a SD card
-- Open the UART4 terminal
-- Enter u-boot by hitting any key until you enter  (i.e., STM32MP>)
STM32MP> ums 0 mmc 1

## 6. Run STM32MP157C from eMMC
-- When "Flashing service completed successfully" is displayed in the USB1 terminal, flashing has been successful
-- Remove BOOT MODE Header Jumper & Press the RST BTN
--- This will then boot into linux (u-boot is passed) - "root" is the login
--- The blue user application led will start beating
root@stm32mp1:~#
```

Building and SDK usage

```
## 1. Build Yocto Project
cd ~/Desktop/Embedded_Systems/Software/apps/LUCY-II-yocto-odyssey/
source poky/oe-init-build-env build-mp1
bitbake custom-image

## 2. Populate SDK
bitbake custom-image -c populate_sdk
cd tmp/deploy/sdk/
sh poky-glibc-x86_64-custom-image-cortexa7t2hf-neon-vfpv4-stm32mp1-toolchain-3.1.19.sh
/home/jonathan/Desktop/Embedded_Systems/Software/apps/LUCY-II-yocto-odyssey/SDK/
Y
```

Notes & Requirements

- Require Host build machine to have the linux environment
  - tested on linuxmint-20.1
- Require at least 50-100GB of free HDD/SDD space
- Depending on the computer this can take a while (i.e., 2-6hrs)
  - Change PARALLEL_MAKE & BB_NUMBER_THREADS for efficiency and performance configuration
    - modify `build-mp1/confb/local.conf`
- Change the BBLAYERS_PATH variable to LUCY-II-yocto-odyssey directory - change in `build-mp1/conf/bblayers.conf`
  - i.e.,

    ```
    BBLAYERS_PATH = "/home/jonathan/Desktop/Embedded_Systems/Software/apps/Stm32mp157/LUCY-II-yocto-odyssey"
    ```
- Yocto manual
  - https://docs.yoctoproject.org/ref-manual/index.html
- A practical guide to BitBake
  - https://a4z.gitlab.io/docs/BitBake/guide.html
- Install STM32_Programmer_CLI
  - https://wiki.st.com/stm32mpu/wiki/STM32CubeProgrammer
  - Manual - https://www.st.com/resource/en/user_manual/um2237-stm32cubeprogrammer-software-description-stmicroelectronics.pdf
- Intro to Embedded Linux tutorials
  - Can be found here `Embedded_Systems/Software/test-apps/Embedded Linux - tutorials/`
- Device Tree Patches
  - git diff --no-index <.orig> > <.patch>
  - Then update the file location i.e., --- a/stm32mp157c-odyssey.dts to --- a/arch/arm/boot/dts/stm32mp157c-odyssey.dts
- Install yocto build dependencies

  ```
  sudo apt update
  sudo apt upgrade
  sudo apt install -y bc build-essential chrpath cpio diffstat gawk git texinfo wget gdisk python3 python3-pip
  sudo apt install -y libssl-dev
  ```

- Install gcc-arm dependencies for Cortex-M4 development

  ```
  sudo apt install gcc-arm-none-eabi
  ```

- Install Linux USB driver dependencies for STM32CubeProgrammer

  ```
  sudo apt-get install libusb-1.0.0-dev
  cd ~/STMicroelectronics/STM32Cube/STM32CubeProgrammer/Drivers/rules
  sudo cp *.* /etc/udev/rules.d/
  ```