Firmware for i.MX8 with Yocto

Ziloo Port of Yocto

The initial base is to create a basic port of Yocto for i.MX 8. If easy it would be very beneficial to create an equivalent port with equivalent support for Raspberry Pi within the same repository. The port must be buildable within the ziloo-firmware GitHub repository.

The initial firmware version and camera module connectivity will be evaluated against the I-Pi SMARC i.MX 8M Plus board. After that the target will also be the UCM-iMX8M-Plus as we aim to use the UCM SoM in production.

The product specific pages for the UCM-iMX8M-Plus, DART-MX8M-PLUS for inspiration.

- Hardknott (kernel 5.10.52)
- UCM-iMX8M-Plus: Yocto Linux: How-To Guide
- UCM-iMX8M-Plus NXP iMX8M-Plus Yocto Linux
- Introducing the I-Pi SMARC
- Yocto Hardknott on I-Pi SMARC
- How to Build Yocto on I-Pi SMARC
- Varscite DART i.MX 8M Plus
- Building DART Yocto from Source
- Installing Yocto to the SOM's internal storage

Working system

A working system has a number of inputs and outputs. If one or more isn't avaiable the rest of the system must continue running. So the idea of a working system is from a QA perspective, not a booting requirement.

- Audio output over HDMI
- 4 channel stereo input via SAI5
- OmniVision OV2735 stereo cameras
- V4L2 Video sources for CSI1 and CSI2

Build, Verify & Upload firmware

The i.MX BSP Porting Guide(Locally) would be the starting point for the port.

The basic port must

- Support OV2735 and Stereo Microphone
- Support Keyboard, Mouse, HDMI
- Support Ethernet & SD Card
- Provide Core debug console over UART2 & UART4

Building the Yocto distribution should run in a Docker container to be dependable long term. Use a variation of the existing Dockerfiles in this repository. The build must create a distribution that can boot on the

Compulab UCM carrier board or the Ziloo Board.

Once built the firmware image must be loaded on the SoM eMMC. Document the commands and connections needed to update the SoM on a carrier board via the USB-C (USB1).

Related sources for i.MX Docker builds and doku,

- NXP i.MX8 Yocto Docker Enviroment
- Running Docker on an i.MX8
- Great release documentation example
- Yocto Linux BSP Ver.9 notes
- Mender setup yocto
- Meeting the Yocto Project
- iMX Working with Yocto
- i.MX Yocto Project User's Guide
- IMX LINUX USERS GUIDE
- Balena yocto scripts for reference
- DART-MX8M - Customizing U-Boot

U-Boot / Fastboot

The boot must provide a way to boot an image in RAM over USB. This should be possible with Fastboot support in U-Boot. It enables a PC running the Fastboot service to provide the Firmware for the device. When the device boots it loads the firmware OS via USB. It must also be possible to connect to the device via adb/Fastboot to update firmware installed on eMMC.

Booting must proceed normally if on Fastboot server is connected on the USB 2.0 OTG.

The partition layout should be something like:

1. SPL
2. BL31
3. U-Boot / BL33
4. Linux Kernel
5. SquashFS / SYS Linux (/)
6. F2FS User (/usr)

The idea is that the base OS can be replaced on an irregular interval by overwriting it with a standard image, while the device specific software and data can be wiped or replaced by overwriting the user partition.

At boot the device must also check if a partial firmware/data update is provided on the OTG connector and import it if present.

Linux Configuration

The Yocto Linux must be configured with the following features,

- Kernel 5.10+
- Avoid OpenEmbedded as the distro must be minimal and stable. Extract parts in dedicated BSP.
- Filesystem support for F2FS, Ext, vfat, tmpfs, squashfs
- Timezone is Copenhagen

- Base Firmware from read-only squashfs
- Wayland / Weston
- RDP support (FreeRDP)
- muslc static library based
- busybox shell
- Default bootloader setup (U-Boot, SPL, U-Boot DTB, Arm Trusted Firmware)

System Configuration and Test

The system BSP is configured to multiplex I/O to standard choices. Once configured the Ziloo Selftest can be run to verify connected chipsets and devices.

Microphones I2S

The microphone I2S mapping is done by using AL2 mode for the SAI3 pads to get SAI5 signals. Multiplexed Signal Pins. The microphones on the 6 pins and 34 pins connector use SAI5_RX_DATA0.

Misc pin	SoM pin	i.MX pad	Functionality	ALT
11	P1.26	SAI3_TXD	SAI5_RX_DATA3	ALT2
17	P1.28	SAI3_RXD	SAI5_RX_DATA0	ALT2
15	P1.30	SAI3_MCLK	SAI5_MCLK	ALT2
19	P1.32	SAI3_RXC	SAI5_RXC	ALT2
23	P1.34	SAI3_RXFS	SAI5_RX_SYNC	ALT2
13	P1.36	SAI3_TXC	SAI5_RX_DATA2	ALT2
21	P1.38	SAI3_TXFS	SAI5_RX_DATA1	ALT2

Other Connectors

Ethernet (ETH0), microHDMI (HDMI)

Configure IOMUX

- Configure signal voltages
- Configure SAI5
- Configure UART1..4
- Configure I2C3, I2C5, I2C6
- Configure PWM

Boot Selftest

At boot a selftest must be run. Make an executable called `ziloo_selftest` that writes to standard out. It is run at boot to log the output. It can otherwise be run while the system is running.

The selftest checks,

- Enumerate devices on the SYS I2C bus. Read out Device Identification of devices.
- Enumerate devices on the Stem I2C3 bus. Read out Device Identification of devices.
- Microphones / Speakers lines connected
- Left camera SCCB & CSI connection and status
- Right camera SCCB & CSI connection and status
- Enumerate devices on the I2C5 & I2C6 bus. Read out Device Identification of devices.
- EEPROM present
- RTC battery power level
- System Voltage power level
- I/O Expanders 0,1,2 present (can read configuration registers)
- Blink LEDs via I/O Expanders
- USB1 connected
- USB2 connected
-
- USB Mux 3220 attached state

Camera Modules

The systems will normally have two camera modules that are used to record stereo video.

Write drivers to support modules

Detect which type of sensor is connected and enable the appropriate driver. A driver that supports multiple sensors can be created.

- OmniVision OV2735
- OmniVision OV5647
- Sony IMX477
- ov2735.c
- ov4689.c
- rv1126 ov2735.c
- I2C ov2735.c
- Kernel 4.4 drivers/media/i2c/ov2735.c

Camera Module support

The camera module must be supported with a Device Driver including V4L2 support. Stereo vision and microphones must be supported.

- Left Camera CSI1 & I2C5(for SCCB)
- Right Camera CSI2 & I2C6(for SCCB)
- I2S stereo via Channel 0(RX0) for SAI5
- Configure stereo video recording CSI1, CSI2, SAI5 RX0
- DART-MX8M-PLUS Camera
- Camera Driver development

OpenCV, ONNX and TensorFlow Lite

OpenCV hardware abstraction layer must be installed on the firmware and shown to work. It must be of the eIQ® OpenCV NXP variant that uses acceleration hardware. This should all be installed out of the box based on i.MX Machine Learning User's Guide (IMXMLUG).

- iMX Manifest
- eIQ® OpenCV Neural Network and ML Algorithm Support
- i.MX Machine Learning User's Guide (IMXMLUG)
- eIQTM MACHINE LEARNING SOFTWARE DEVELOPMENT ENVIRONMENT
- Support OpenCV 4.5.2+
- Support Tensorflow Lite 2.6.0+
- Support TVM 0.7.0
- Support ONNX (VSI NPU/NNAPI)
- Support Arm NN(VSI NPU)
- Support Arm Compute Library
- Support PyArmNN
- Support NNAPI Delegate
- Support VX Delegate (OpenVX with extension)
- Verify OpenCV and TensorFlow using the demo apps as described in the User's Guide.

The NXP eIQTM machine learning software development environment enables the use of machine learning algorithms on i.MX family SoCs. The eIQ software for i.MX includes inference engines, optimized libraries for hardware acceleration, and an ML security package. The main eIQ toolkit is integrated in the Yocto BSP, contained in meta-imx/meta-ml layer. The following inference engines are currently supported: ArmNN, TensorFlow Lite, ONNX Runtime, OpenCV, DeepViewRT(TM), and PyTorch. See the i.MX Machine Learning User's Guide (IMXMLUG) for details about the eIQ software development environment.

MediaPipe

Media Pipe should be installable since OpenCV is available. Record any issue that stops the installation.

For reference: MediaPipe on RaspberryPi

T-USB / USB-C Gadget support

Ziloo has two USB 3.0 connectors equivalent to USB1 & USB2 on UCM-iMX8M-Plus. USB1 also called OTG must work as an USB Gadget.

It should be possible to,

- Use Ziloo as a Stereo Webcam
- Use Ziloo as a Stereo Microphone
- Provide Ziloo with an Internet Connection by connecting over USB
- Access the Ziloo filesystem over USB
- Connect a monitor over USB using DisplayLink

Full USB OTG Gadget support over T-USB OTG

The device supports USB Gadget interfaces to work like a stereo WebCam. The Kernel config must be adjusted to support these.

- Providing USB Gadget Interfaces over USB 2.0 and 3.0
- Providing USB Gadget stereo camera emulation (routed from CSI1 & CSI2)
- Providing USB Gadget stereo microphone emulation (routed from SAI5 ch 0)
- Providing MSC Mass Storage giving access to filesystem on eMMC and SD Card
- Providing Android Debug Bridge (adb), Enabling adb support, Kernel Drivers Config
- Providing g_ethernet for getting an Internet connection. How to Use USB Gadget Ethernet
- Providing RNDIS for getting an Internet connectionn
- Support on recent Linux 5? Article from 2019
- Is configfs a good avenue?
- Providing DisplayLink USB 3.0 kernel experimental support (Will try to find a monitor to test with). ArchLink ref

Further

Core OS Booting Read-only Additional applicatiosn and Data on F2FS partition FAT32 config partition Booting from SD Card or USB stick (Host USB)

When g_ethernet connection is established try establishing IP address and route to Internet. Update system time on connect.

Get eIQ GStreamer Demo apps to work