Using MIPI DSI as Main Display Interface
WITH YOU TODAY...

- Joined Toradex 2011
- Spearheaded Embedded Linux Adoption
- Introduced Upstream First Policy
- Top 10 U-Boot Contributor
- Top 10 Linux Kernel ARM SoC Contributor
- Industrial Embedded Linux Platform Torizon Fully Based on Mainline Technology
  - Mainline U-Boot with Distroboot
  - KMS/DRM Graphics with Etnaviv & Nouveau
  - OTA with OSTree
  - Docker
WHAT WE’LL COVER TODAY…

- MIPI Display Serial Interface (MIPI DSI)
- Verdin MIPI DSI Display Adapter System Design
- Introduction of the Linux DSI Subsystem
- Linux DRM Stack DSI Bridge Chip Integration
- DSI Bridge Chip Ecosystem
- Bridge Chips Supported in Mainline
- Auto-Detection of DSI Adapters Based on EEPROM Contents
- U-Boot: Reading EEPROM Contents and Selecting Applicable Device Tree Overlay
- U-Boot FIT Image: Board Specific Device Trees and Display Adapter Specific Device Tree Overlays
- Live Demo: DSI Auto-Detection
MIPI Display Serial Interface (MIPI DSI)

- Specification by the Mobile Industry Processor Interface (MIPI) Alliance
- High-speed differential signaling point-to-point serial bus interface between a host processor and a display module
- High performance, low power, low electromagnetic interference (EMI)
- Reduced pin count
- Compatibility across different vendors
- One high speed clock lane and one or more data lanes
- Low power (LP) mode or high speed (HS) mode
MIPI Display Serial Interface (cont.)

- MIPI DSI Specification
  - Initial Version: May 2006
  - Current Version: MIPI DSI v1.3.1 (December 2015)

- Successor MIPI DSI-2 Specification
  - Initial Version: January 2016
  - Current Version: v1.1 (May 2018)
  - Support for both D-PHY and C-PHY
  - Supports ultra-high definition (4K and 8k)
MIPI Display Serial Interface (cont.)

- Physical Layer:
  - MIPI D-PHY
    - D-PHY 1.01: 1.0Gbps/lane
    - D-PHY 1.1: 1.5Gbps/lane
    - D-PHY 1.2: 2.5Gbps/lane
    - D-PHY 2.0: 4.5Gbps/lane
  - MIPI Display Command Set (MIPI DCS)
  - Incorporates Display Stream Compression (DSC)
    - Standard from the Video Electronics Standards Association (VESA)
  - De-facto standard display interface featured by modern higher-end SoCs
  - No long-term available discrete MIPI DSI display panels
  - Bridge chips converting to more common display interfaces like parallel RGB, LVDS, (e)DP or HDMI
Verdin MIPI DSI Display Adapter System Design

- Generic system concept
- DSI display adapter boards integrating various bridge chips
- ST M24C02 2kb EEPROM to store identification/parametrisation
- DSI Mezzanine Connector
  - MIPI DSI: 1 clk + 4 data lanes
  - GPIOs
    - BKL1_EN
    - Touch interrupt
  - 2 x I2C: bridge chip + DDC/EDID
  - PWM: backlight
  - I2S: optional audio
  - Generic system control signals
    - PWR_EN_MOCI, SLEEP_MOCI#, RESET_MOCI#
Verdin iMX8M Mini

- NXP i.MX 8M Mini SoC
- Single display controller, LCDIF
- MIPI DSI output with up to four data lanes
- Northwest Logic MIPI DSI host controller IP
- MIPI D-PHY 1.2
- maximum data transfer per lane only 1.5Gbps
- Resolutions up to 1920x1080p60 and 1800x1200p60
Verdin DSI to HDMI Adapter

- Lontium Semiconductor LT8912B MIPI DSI to HDMI bridge
- HDMI V1.4 1080p (1920x1080), 8-bit RGB, up to 60Hz
- ST HDMI2C1-14HD ESD protection and signal conditioning
- Type A standard HDMI connector
Texas Instruments SN65DSI84 MIPI DSI to dual-link LVDS bridge

- single/dual-lane LVDS up to 1920x1200/1366x768, 60fps, 24bpp
- LVDS and touch connectors compatible with Toradex Capacitive Touch Display 10.1” LVDS
Introduction of the Linux DSI Subsystem

- DRM MIPI DSI Core: Common logic and helpers to deal with MIPI DSI peripherals
Linux DRM Stack
DSI Bridge Chip Integration

- DRM bridge
  - drm_bridge_funcs: attach, enable, disable
  - drm_bridge_add()
- DRM connector
  - drm_connector_funcs: fill_modes, detect, destroy
  - drm_connector_helper_funcs: get_modes, mode_valid
  - drm_connector_init/_helper_add/_attach_encoder(), drm_panel_attach()
- I2C device
  - Detected using regular id_table and of_match_table
  - Regmap: devm_regmap_init_i2c()
  - i2c_set_clientdata()
- MIPI DSI device
  - mipi_dsi_device_register_full()
  - mipi_dsi_attach()
DSI Bridge Chip Integration Pitfalls

- Availability of “super secret” data sheets
- Ancient downstream or bare skeleton drivers only
- Lots of hard-coded parameters
- Link bring-up sequences not well documented
  - May require a lot of trial and error
- Divider/frequency limitations on either controller side, bridge side or both
  - May require running outside of recommended range
How About Bridge Chips Used in our Current Adapters?

- Lontium Semiconductor LT8912B MIPI DSI to HDMI bridge
  - Adopted downstream driver from Rockchip Linux on GitHub
    - Forward ported to later DRM API
    - Fixed confusing use of same name for struct and instance
    - Reworked driver to be a proper I2C device
    - Full register set taken from Lontium pseudo code driver
    - Improved regmap integration
    - Properly reserve i2c sub addresses
    - Added regular I2C based DDC/EDID handling
    - Added GPIO based hot-plug detection
    - Hot-plug detect GPIO handling crashed using GPIO expander (cansleep variant of gpiod_get_value fixed it)
    - Bus_format was not properly set
  - Not pretty but hey it works (;-p)
  - Further clean-up and upstreaming pending
How About Bridge Chips Used in our Current Adapters? (cont.)

- Texas Instruments SN65DSI84 MIPI DSI to dual-link LVDS bridge
  - Downstream driver taken from a patch in CompuLab Yocto Meta Layer on GitHub
    - Luckily already adopted to usage on i.MX 8M Mini
    - Hard-coded for single-channel LVDS use-case
  - Implementing support for dual channel LVDS pending
  - Further clean-up and upstreaming pending
DSI Bridge Chip Ecosystem

- Vendors still reluctant to mainlining drivers
- Few mainline supported bridge chips
- Few examples to copy from
- Procurement of actual silicon may be difficult
- Conformance of MIPI DSI host IP vs. bridge chip silicon
Bridge Chips Supported in Mainline

- drivers/gpu/drm/bridge
- Differentiate between SoC internal IP, discrete external bridge chips and directly connected panels
- Northwest Logic MIPI DSI host controller as found on NXP i.MX 8 series
- Analog Devices ADV7533/35 MIPI/DSI Receiver with HDMI Transmitter
- Parade PS8640 MIPI DSI to eDP Converter
- Texas Instruments SN65DSI86 DSI to eDP bridge
- Toshiba TC358764 DSI/LVDS bridge
- Toshiba TC358768AXBG/TC358778XBG MIPI DSI bridge chips
- Raspberry Pi 7-inch Touch Display
- Toshiba TC358762 DSI to DPI aka parallel RGB bridge
Auto-Detection of DSI Adapters Based on EEPROM Contents

• Straight forward idea 1:
  • Regular device tree: setting bridge status to disabled vs. okay
  • Device graph: Linking endpoint and remote-endpoint nodes?
• Full flexibility requires device tree overlays
• Straight forward idea 2:
  • Just storing device tree overlay in EEPROM
  • While simple DTBOs may be below 1kb more complex ones quickly account for more than 2kb in size!
• Compromise: Just store product number as part of regular Toradex factory configuration block aka ConfigBlock
• Select device tree overlay to be applied based on product number

```c
&hdmi_bridge {
    status = "disabled";
};
&lvds_bridge {
    status = "okay";
    port {
        dsi84_in: endpoint {
            remote-endpoint = <&mipi_dsi_bridgel_out>;
        }
    }
};
&mipi_dsi {
    port@1 {
        reg = <1>;
        mipi_dsi_bridgel_out: endpoint {
            remote-endpoint = <&dsi84_in>;
        }
    }
};
```
U-Boot: Reading EEPROM Contents and Selecting Applicable Device Tree Overlay

- Generalised ConfigBlock handling from NAND/eMMC to EEPROMs
- Table with product ID to device tree overlay file name mapping
- Distroboot script to apply device tree overlays based both on auto-detection as well as overlays.txt file

- HDMI may do hot-plug detect
- DDC/EDID vs. custom display-specific parametrisation (cascading device tree overlays)

- LVDS usually requires further parametrisation
  - Single/dual-channel
  - Colour format
  - Panel resolution and timing
Device Tree Overlays

// Verdin DSI to HDMI Adapter orderable at Toradex.
...

fragment00 {
    target-path = "/12c@30a30080"; /a Verdin IIC_2_DSI */
    _overlay_ {*
        clock-frequency = <100000;
        pinctrl-names = "default";
        pinctrl-0 = <spintrcl_i2c2>;
        status = "okay";
    };
};

fragment01 {
    target-path = "/12c@30a50080"; /a Verdin IIC_1 */
    _overlay_ {*
        hdmia0 {*
            compatible = "lontium,lt8912";
            ddc-i2c-bus = <i2c2>;
            gpio-gpios = <0x103 15 GPIO_ACTIVE_HIGH>;
            pinctrl-names = "default";
            pinctrl-0 = <spintrcl_gpio_hpe, spintrcl_gpio1>,
                        <spintrcl_gpio2>;
            reg = <0x48>;
            reset-gpios = <gpio5 5 GPIO_ACTIVE_LOW>;

            port {
                lt8912_l_in: endpoint {
                    remote-endpoint = <@mpi_dsi_bridgel_out>;
                };
            };
        };
    };
};

fragment02 {
    target-path = "@mpi_dsi@32e180000";
    _overlay_ {*
        port01 {*
            reg = <0>;
            mpi_dsi_bridgel_out: endpoint {
                remote-endpoint = <@lt8912_l_in>;
            };
        };
    };
};

fragment03 {
    target-path = "@mpi_dsi@932e10000";
    _overlay_ {*
        port01 {*
            reg = <0>;
            mpi_dsi_bridgel_out: endpoint {
                remote-endpoint = <ds185_in>;
            };
        };
    };
};
FIT image allows convenient packing of Linux kernel binary together with various device trees, device tree overlays and/or ramdisks.

May be booted as follows:

```
boottm ${loadaddr}#config@$({soc}-$({fdt_module}-$ {fdt_board}).dtb#$({display_adapter_dtbo})
```

In our case fdt_module is deduced from the EEPROM on the module, fdt_board from the one on the carrier board and display_adapter_dtbo from the one on the display adapter.

U-Boot FIT Image:
Board Specific Device Trees and Display Adapter Specific Device Tree Overlays

```
$ mkeimage -l tezi.itb
FIT description: U-Boot fitImage for Toradex Easy Installer
Created: Tue Jun 9 01:57:39 2020
Image 0 (kernel@)
  Description: Linux kernel
  ...
Image 1 (fdt@freescale_fsl-imx8mm-verdin-nonwifi-dev.dtb)
  Description: Flattened Device Tree blob
  ...
Image 2 (fdt@freescale_fsl-imx8mm-verdin-wifi-dev.dtb)
  Description: Flattened Device Tree blob
  ...
Image 3 (fdt@verdin-imx8mm_lt8012_overlay.dtb)
  Description: Flattened Device Tree blob
  ...
Image 4 (fdt@verdin-imx8mm_sn65ds184_overlay.dtb)
  Description: Flattened Device Tree blob
  ...
Image 5 (ramdisk@)
  Description: tezi-initramfs
  ...
Default Configuration: "config@freescale_fsl-imx8mm-verdin-nonwifi-dev.dtb"
Configuration 0 (config@freescale_fsl-imx8mm-verdin-nonwifi-dev.dtb)
  Description: 1 Linux kernel, FDT blob, ramdisk
  Kernel: kernel@1
  Init Ramdisk: ramdisk@1
  FDT: fdt@freescale_fsl-imx8mm-verdin-nonwifi-dev.dtb
  Hash algo: sha1
  Hash value: unavailable
Configuration 1 (config@freescale_fsl-imx8mm-verdin-wifi-dev.dtb)
  Description: 0 Linux Kernel, FDT blob, ramdisk
  Kernel: kernel@1
  Init Ramdisk: ramdisk@1
  FDT: fdt@freescale_fsl-imx8mm-verdin-wifi-dev.dtb
  Hash algo: sha1
  Hash value: unavailable
```
Device Tree Overlay Pitfalls

- More complex device tree overlays may require symbols
  - Make sure device trees and overlays are all compiled with DTC_FLAGS='@'

- Referencing nodes via hex addresses from within device tree overlays proves to be case sensitive!
  - Make sure all adhere to consistent lower-case hex numbering

- Troubleshooting what really got applied
  - Use dtc -I fs on target and dump /proc/device-tree

```
root@verdin-imx8mm:~# opkg install dtc_1.5.1-r0_aarch64.ipk
Installing dtc (1.5.1) on root
Configuring dtc.
root@verdin-imx8mm:~# dtc -I fs /proc/device-tree -O dts -o dump.dts
```
Live Demo: DSI Auto-Detection
References

- https://www.mipi.org/specifications/dsi
- https://developer.toradex.com/products#verdin-som-family
- Lontium Semiconductor LT8912B MIPI DSI to HDMI bridge driver
  http://git.toradex.com/cgit/linux-toradex.git/commit/?id=331ac1cf6e09d90e7d9ab39445bc8812ff33f178
THANK YOU
FOR YOUR INTEREST

www.toradex.com
developer.toradex.com
community.toradex.com
labs.toradex.com