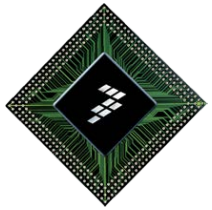
June 7th, 2011



i.MX28 Hands-On Training

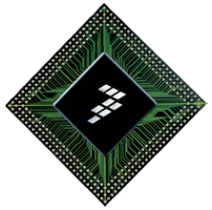
Vladan Jovanovic, Freescale, FAE
Libor Gecnuk, Freescale, FAE

freescale™ semiconductor

Agenda

- 11:30-12:30 Introduction
  - i.MX Overview and Roadmap
  - i.MX28 and i.MX28EVK overview
  - i.MX28 peripherals
- 12:30-13:30 Lunch
- 13:30-15:30 i.MX28 Hands-on
  - I/O Mux tool and Manufacturing tool
  - Linux and WinCE demo images
- 15:30-16:00 Break
- 16:00-17:00 i.MX28 Hands-on
  - Working with BSP
  - LTIB introduction

i.MX Overview and Roadmap



freescale™ semiconductor

Freescale Multimedia Markets

Portable Consumer

- Smartbook
- E-book
- Smartphone
- Portable Media Player
- Personal Navigation

Automotive Infotainment

- Audio
- Connectivity and Telematics
- Video and Navigation

Low-Power
High Integration
Advanced Performance
Platform Software

Home Consumer

- Media Phone/Terminal
- iPod accessories
- Remote controls
- Digital Photo Frame
- Appliances

Industrial

- POS/Scanners
- Security and Surveillance
- Industrial HMI
- Medical
- Metering

freescale semiconductor

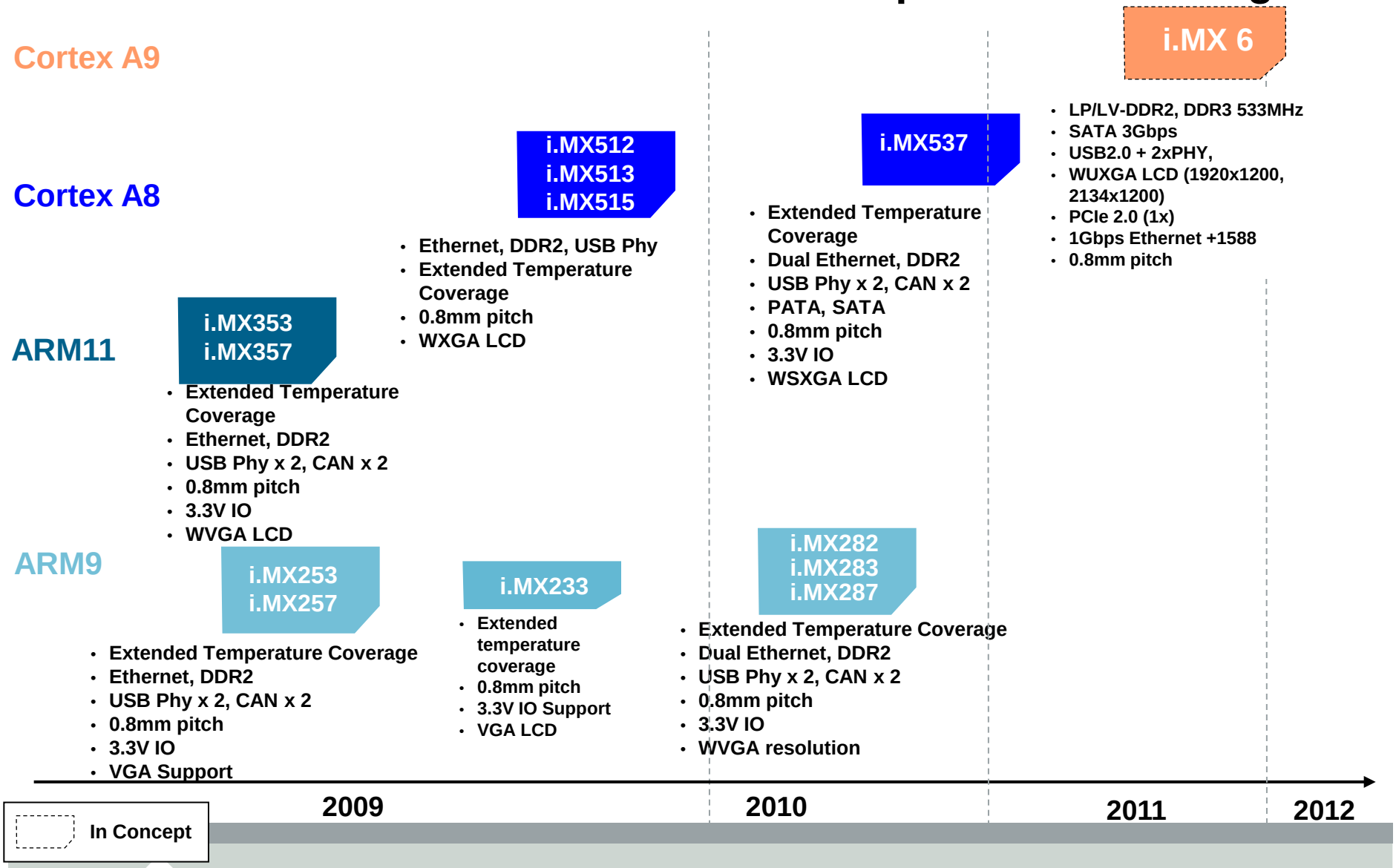
Freescale Product Longevity Program

- The embedded market needs **long-term product support**
  - Freescale has a longstanding track record of **providing long-term production support** for our products
    - Freescale is pleased to introduce a **formal product longevity program** for the market segments we serve
      - For the automotive and medical segments, Freescale will make a broad range of program devices available for a minimum of **15 years**
      - For all other market segments in which Freescale participates, Freescale will make a broad range of devices available for a minimum of **10 years**
      - **Life cycles** begin at the time of launch
- A list of participating **Freescale products** is available at: www.freescale.com/productlongevity

i.MX processors provide supply stability to customers

freescale semiconductor

ARM™ based Processor Roadmap In Industrial Segment



freescale semiconductor ™

i.MX28x Family

Key Features and Advantages

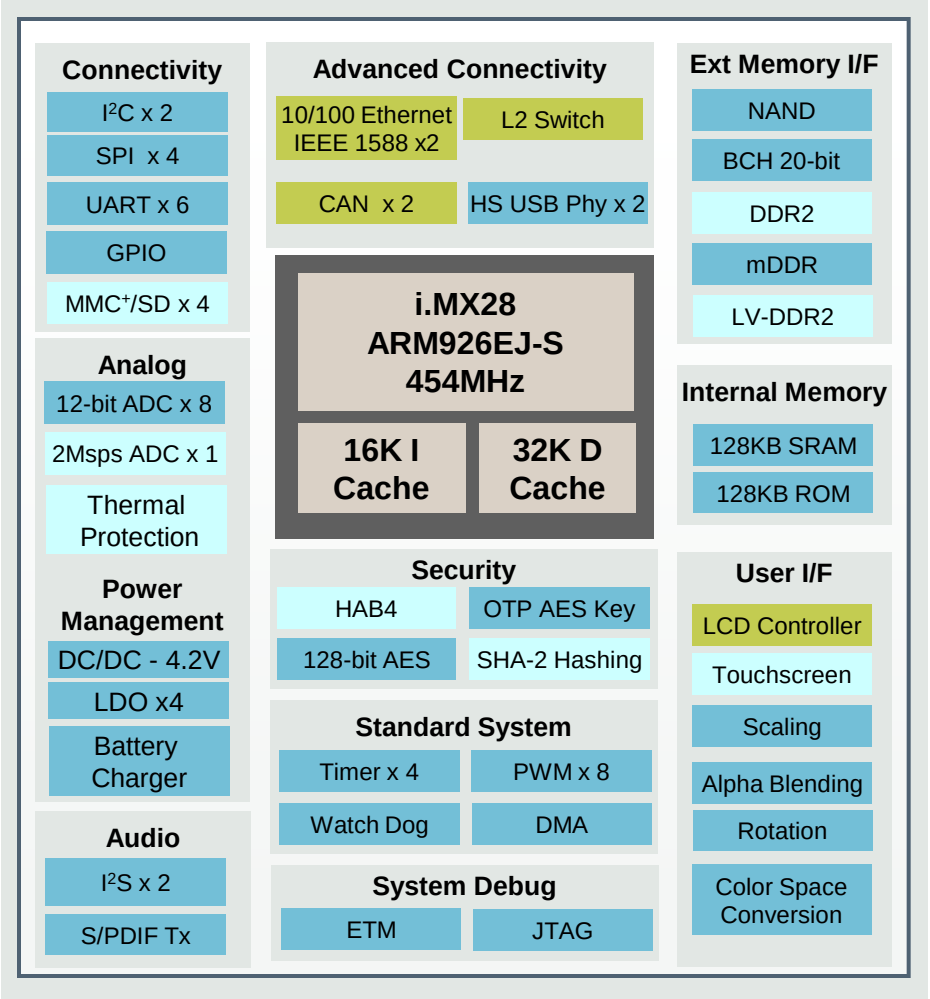
- 454MHz ARM926EJ-S core w/ 32KB Cache
- PMU with high efficiency on-chip DC/DC, supports Li-Ion batteries
- 10/100 Dual IEEE 1588 Ethernet with RMII support and L2 Switch
- Dual CAN interfaces
- LCD Controller with Touchscreen
- NAND support – SLC/MLC and eMMC 4.4 managed
- Hardware BCH (up to 20-bit correction)
- 200 MHz 16-bit DDR2, LV-DDR2, mDDR external memory support
- Dual High speed USB with embedded PHY
- Up to 8 General purpose 12-bit ADC channels and single 2 Msps ADC channel
- LCD Controller with Touchscreen
- Temperature sensor for thermal protection
- Multiple connectivity ports (UARTs, SSP, SDIO, SPI, I2C, I2S)
- Family of products supporting various feature sets

Package and Temperature

- 289 BGA 14x14mm .8mm
- -40C to +85C (Industrial, Automotive)
- -20C to +70C (Consumer)

Availability:

- Alpha Samples: Now
- Production: early Oct 2010 (consumer), end Oct 2010 (industrial)



Common IP with i.MX233 | New or enhanced from i.MX233 | Not available on all variants

freescale semiconductor

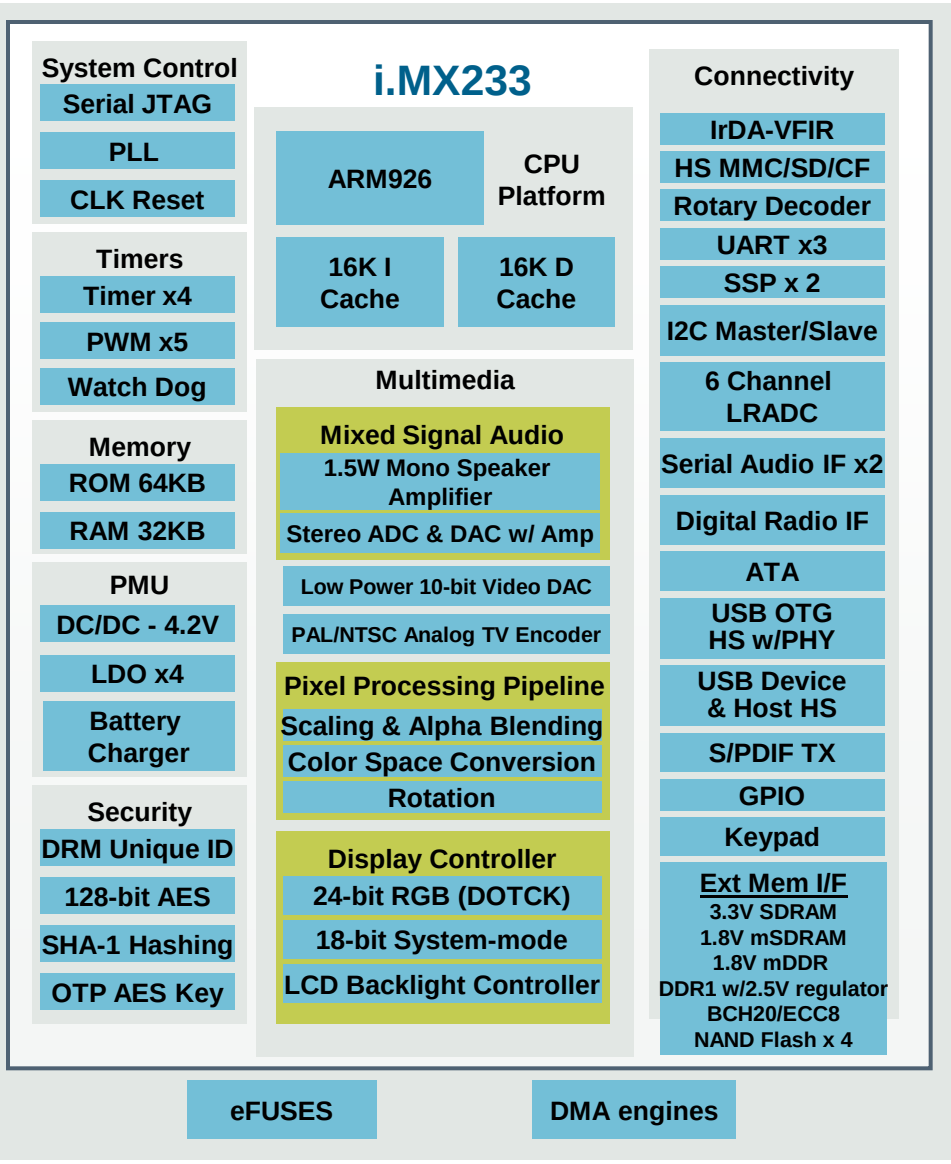
i.MX233 Applications Processor

► Specifications:

- **CPU:** ARM926, 400+ MHz
- **Process:** TSMC 90LP
- **Core Voltage:** 1.05V – 1.45V
- **Package:** 169fpBGA 11x11mm .8mm
  128LQFP 14x14mm .4mm
- **Temp Range:** -10 to 70C, -40 to 85C

► Key Features and Advantages

- Based on STMP IP platform
- ARM926 Core 400+ MHz
- **PMU with high efficiency on-chip DC/DC with 4.2V output, supports Li-Ion batteries**
- Very low video and audio power consumption
- **1.5W Mono speaker amplifier**
- **Stereo headphone DAC w/ 99dB SNR & Stereo ADC w/ 85 dB SNR with integrated amplifiers**
- Hardware BCH (up to 20-bit correction) and RS ECC8 for current and future MLC NAND support
- Power-Efficient Direct-Drive LCD Backlight Controller with Voltage or Current Feedback
- DDR1 Support with integrated 2.5V regulator
- High speed USB with embedded PHY



freescale semiconductor

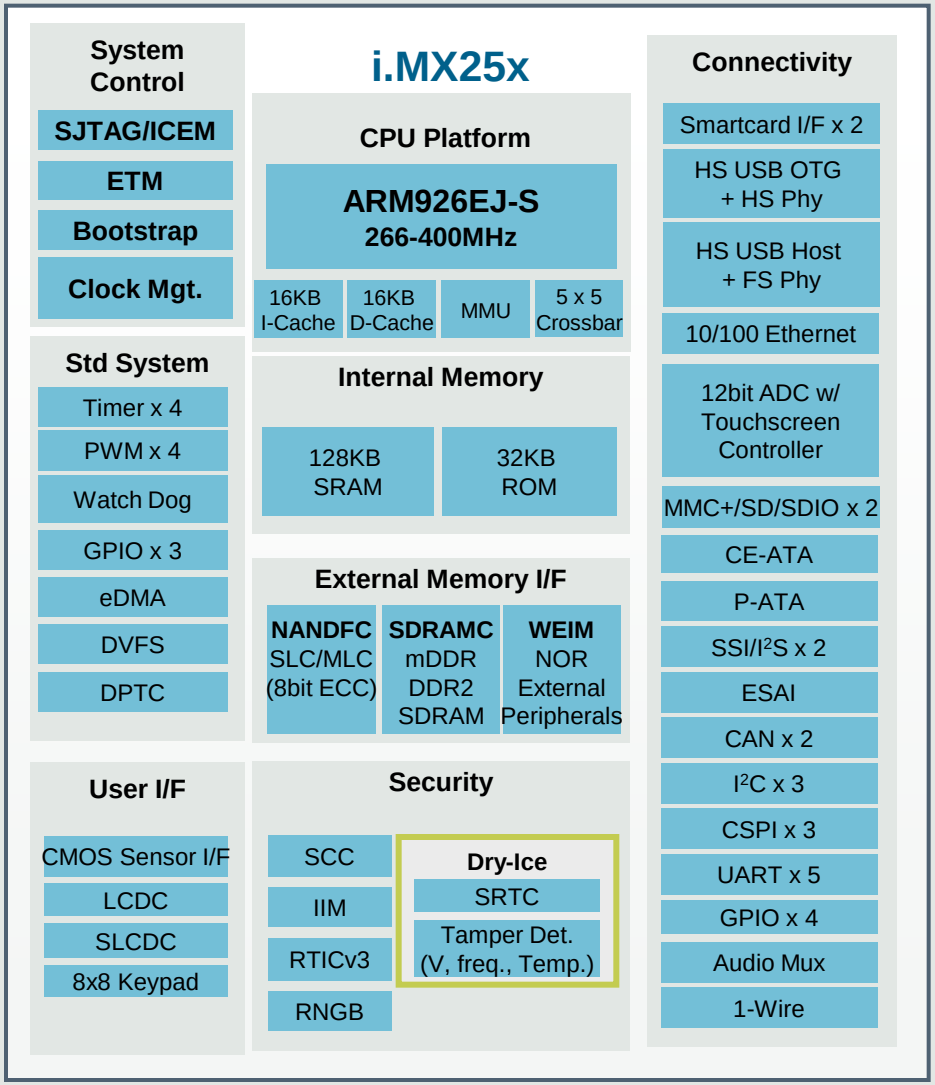
i.MX25x Multimedia Applications Processor

► Specifications:

- **CPU:** ARM926EJ-S, 266-400MHz
- **Process:** 90nm LP2
- **Core Voltage:** 266MHz @ 1.2V – 1.52V
  400MHz @ 1.38V – 1.52V

► Key i.MX25 Features and Advantages

- Same proven ARM926 platform as i.MX27
- High performance general purpose processor
- Enhanced security features, including tamper detection
- 10/100 Ethernet MAC with RMII support
- Two on-chip USB ports with PHY
  - High Speed USB OTG with HS PHY
  - High Speed USB Host with FS PHY
- 128KB on-chip SRAM – ideal for ultra low power LCD refresh
- 3 general purpose 12-bit ADC channels
- Touchscreen controller
- Two CAN interfaces
- Two smart card interfaces
- Enhanced serial audio interface
- 16-bit 1.8V mobile DDR and DDR2
- 16-bit 3.3V SDRAM
- 3.3V I/O
- 0.8mm MAPBGA package, -40C to +85C



freescale semiconductor

i.MX ARM9 Portfolio

Feature	i.MX25x	i.MX233	i.MX28x
RAM	128KB	32KB	128KB
Flash Interface	MLC/SLC NAND Flash w/ 8-bit RS, Parallel **NOR Flash**	SLC/MLC/Managed NAND Flash 20-bit BCH, 8-bit RS	SLC/MLC/Managed NAND Flash w/ 20-bit BCH
DRAM Interface	150 MHz 16-bit DDR2, mDDR, SDRAM	150 MHz 16-bit DDR1, mDDR	200 MHz 16-bit DDR2, LV-DDR2, mDDR
LCD	1 overlay, alpha blending, panning	8 overlays, alpha blending, scaling, rotation, color space conversion	8 overlays, alpha blending, scaling, rotation, color space conversion
Integrated TV-Out	-	Yes	Yes
CMOS Sensor Interface	Yes	-	-
CAN	x2	-	x2
10/100 Ethernet	Single 10/100	-	Dual 10/100 (1588 H/W Time stamping) and L2 Switch
Analog Audio	External	Integrated stereo ADC/DAC with amps, Mono speaker amp output	External
S/PDIF Interface	No	1 output	1 output
Power Management	External	Integrated	Integrated
USB 2.0	HS port (Host/Device) HS PHY x1, HS Host with FS PHY x1	HS port (Host/Device) with PHY x1	HS Host/Device with PHY x1, HS Host with PHY x1
SIM	x2	-	-
P-ATA	Yes	-	-
Security	Equivalent capabilities, Tamper Detection, RNG	Equivalent capabilities, PRNG	Equivalent capabilities, HAB4, PRNG

freescale semiconductor

i.MX53x Multimedia Applications Processor

Specifications:

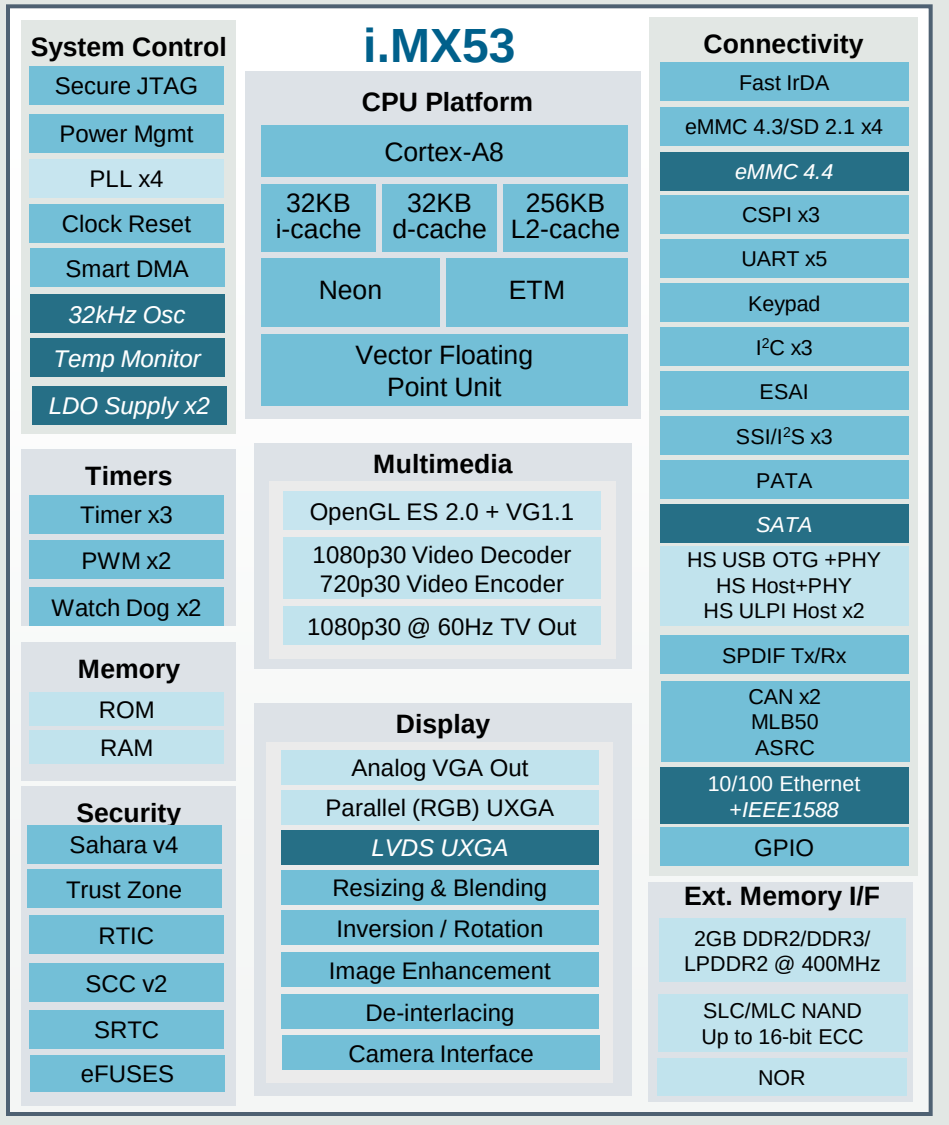
- **CPU:** Cortex-A8
  1GHz – Consumer
  800MHz – Automotive/Industrial
- **Process:** 65nm, LP/GP
- **Core Voltage:** 0.85V-1.3V
- **Package:** 19x19 0.8mm 529 ball BGA
  12x12 0.4mm PoP (Consumer)
- **Case Temp:** -20 to 70C (Consumer)
  -40 to 85C (Automotive/Industrial)

Key Features and Advantages

- High performance CPU: Cortex A8
- 2GB DDR2/3, LPDDR2 memory at 400MHz
- HDD: PATA, S-ATA interface
- One eSDHC ports supports MMC4.4 including DDR mode
- Ethernet 10/100 with IEEE1588
- Delivers rich graphics and UI in HW
  - OpenGL ES 2.0 3D accelerator (AMD Z430)
  - OpenVG 1.1 graphics accelerator (AMD Z160)
  - Neon Vector floating point co-processor
  - Display up to UXGA (1600x1200)
- Drives high resolution video in HW
  - Multi-format HD1080 video decode
  - Multi-format HD720 video encode
  - High quality video processing (resizing, de-interlacing, etc)
  - Displays: Parallel, LVDS or VGA
- Audio:
  - I2S, SPDIF Rx/Tx, ESAI
- Secure boot (HAB), cryptographic accelerators, TZ
- More analog integration: simplified system, reduced system BOM
  - Temperature Monitor for smart performance control
  - Linear supply regulators
  - 32KHz Oscillator

Availability:

- **Samples:** rev2.0 BGA – Nov 9 2010 (alpha customers)
  rev2.0 PoP – Mar 4 2011 (alpha customers)
- **Production:** Q1 2011 (BGA), Q2 2011 (PoP)



Reused IP from i.MX51x or i.MX35x

Updated IP from i.MX51x

NEW IP in i.MX53x

freescale semiconductor

Key differences

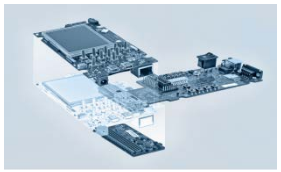
i.MX53 Family Part Numbers

	i.MX534	i.MX536	i.MX537	i.MX535	i.MX538
	Automotive	Automotive	Industrial	Consumer	Consumer
Example App.	Clusters	Video and Navigation	Industrial control, Factory automation, Video surveillance, Medical	Tablet, Video IP Phone, Connected TV, Telehealth, Digital Signage	Tablet, MID, Smartphone
Core	800MHz Cortex™-A8	800MHz Cortex™-A8	800MHz Cortex™-A8	1GHz Cortex™-A8	1GHz Cortex™-A8
Memory	2GB, x32 LPDDR2/DDR2/DDR3	2GB, x32 LPDDR2/DDR2/DDR3	2GB, x32 LPDDR2/DDR2/DDR3	2GB, x32 LPDDR2/DDR2/DDR3	2GB, x32 LP-DDR2
Video Decode	no HW acceleration	Hardware (1080p30)	Hardware (1080p30)	Hardware (1080p30)	Hardware (1080p30)
Video Encode	no HW acceleration	Hardware (720p30)	Hardware (720p30)	Hardware (720p30)	Hardware (720p30)
3D GPU	OpenGL/ES 2.0 33Mtri/s, 200Mpix/s	OpenGL/ES 2.0 33Mtri/s, 200Mpix/s	OpenGL/ES 2.0 33Mtri/s, 200Mpix/s	OpenGL/ES 2.0 33Mtri/s, 200Mpix/s	OpenGL/ES 2.0 33Mtri/s, 200Mpix/s
2D GPU	OpenVG 1.1, 200Mpix/s	OpenVG 1.1, 200Mpix/s	OpenVG 1.1, 200Mpix/s	OpenVG 1.1, 200Mpix/s	OpenVG 1.1, 200Mpix/s
LCD IF	Parallel, LVDS	Parallel, LVDS	Parallel, LVDS	Parallel, LVDS	Parallel, LVDS
Video Out	VGA HD1080p60	VGA HD1080p60	VGA HD1080p60	VGA HD1080p60	VGA HD1080p60
Camera I/F	2x 20-bit Parallel	2x 20-bit Parallel	2x 20-bit Parallel	2x 20-bit Parallel	2x 20-bit Parallel
Ethernet	10/100, IEEE1588	10/100, IEEE1588	10/100, IEEE1588	10/100 IEEE 1588	10/100
SATA	S-ATA II 1.5Gbps	S-ATA II 1.5Gbps	S-ATA II 1.5Gbps	S-ATA II 1.5Gbps	n/a
CAN	2 x FlexCAN	2 x FlexCAN	2 x FlexCAN	n/a	n/a
MLB	MLB50	MLB50	n/a	n/a	n/a
USB	Four HS USB2.0: 1xHS OTG + PHY 1xHost + PHY 2xHost + ULPI/IC-USB	Four HS USB2.0: 1xHS OTG + PHY 1xHost + PHY 2xHost + ULPI/IC-USB	Four HS USB2.0: 1xHS OTG + PHY 1xHost + PHY 2xHost + ULPI/IC-USB	Four HS USB2.0: 1xHS OTG + PHY 1xHost + PHY 2xHost + ULPI/IC-USB	Four HS USB2.0: 1xHS OTG + PHY 1xHost + PHY 2xHost + ULPI/IC-USB
SDIO I/F	3x SD/MMC 4.3 1x SD/MMC 4.4	3x SD/MMC 4.3 1x SD/MMC 4.4	3x SD/MMC 4.3 1x SD/MMC 4.4	3x SD/MMC 4.3 1x SD/MMC 4.4	3x SD/MMC 4.3 1x SD/MMC 4.4
SPI I/F	3x SPI	3x SPI	3x SPI	3x SPI	3x SPI
I2C I/F	3x I2C	3x I2C	3x I2C	3x I2C	3x I2C
Other	5x UART, P-ATA, 3x I2S, S/PDIF Tx/Rx, ESAI	5x UART, P-ATA, 3x I2S, S/PDIF Tx/Rx, ESAI	5x UART, P-ATA, 3x I2S, S/PDIF Tx/Rx, ESAI	5x UART, P-ATA, 3x I2S, S/PDIF Tx/Rx, ESAI	5x UART, P-ATA, 3x I2S, S/PDIF Tx/Rx, ESAI
Package	19x19 0.8P TE-BGA	19x19 0.8P TE-BGA	19x19 0.8P TE-BGA	19x19 0.8P TE-BGA	12x12 0.4P PoP
Qual.	Automotive AEC-Q100	Automotive AEC-Q100	Industrial	Consumer	Consumer
General Availability	Mar 2011	Mar 2011	Apr 2011	Mar 2011	May 2011

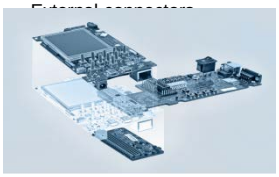
freescale semiconductor ™

i.MX DevKit Roadmap

i.MX31PDK – 1500USD



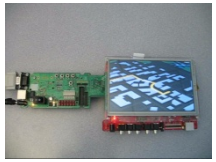
- VGA Touch-screen Display
  - USB, Ethernet
  - BT, Wifi, GPS
  - FM Receiver
  - FM Transmitter
  - TV Encoder
  - Headset Connector
  - Speaker
  - Microphone
- Camera
- Storage (HDD)
- External connectors



i.MX27PDK – 1500USD

i.MX35PDK-1500USD

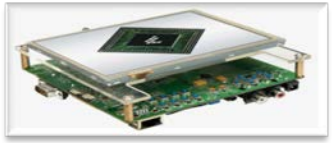
•i.MX35
•MC13892
•SGTL5000



- Auxiliary Video Input for display from external video source
- 5.1 Sound (Audio CODEC)
- FM receiver/tuner to support short range FM adapters
- CAN Connector
- CMOS Image Sensor
- USB OTG and USB Host
- 10/100 Ethernet
- Optional GPS daughtercard

i.MX25PDK-995USD

•i.MX25
•MC34704B
•SGTL500



- 5.7” VGA LCD w/ Touchscreen
- USB 2.0 OTG, Ethernet
- SD/MMC, Smartcard
- CMOS Image Sensor

i.MX51EVKJ-699USD

•i.MX51
•MC13892
•SGTL5000

- 7” WVGA Touchscreen LCD Display (add-on module)
- Expansion board (add-on module)
- 2 LVDS connectors
- DVI-I connector
- 2 SD/MMC Card Slots
- USB Host x2 / USB OTG x1
- Ethernet Port
- Mini PCIe
- SATA HDD connector
- SIM Card connector
- Keyboard connector
- Mic input, stereo headphone output (jack), V2IP Headphone
- USB Camera connector
- RGB output through DVI-I connector
- Ambient light sensor footprint
- FM receiver footprint

i.MX23EVK-399USD

•i.MX233

- 4.3” WQVGA Touchscreen LCD Display (add-on module)
- SD/MMC Card Slot
- USB Host/Device
- Ethernet supported via SPI header
- Navigation keys
- Mic input, headphone output (jack)
- Composite TV Out connector footprint
- 3-Axis Accelerometer footprint •
- Expansion Port for optional Peripheral Card

i.MX53-149USD

•i.MX51
•DA9303
•SGTL5000

- i.MX53 1Ghz Cortex-A8 Processor
- Dialog DA9053 PMIC
- 1 GB DDR3 Memory
- 3” x 3” 8-layer PCB
- LVDS connector
- VGA connector
- Parallel LCD add-on card via Expansio connector
- HDMI add-on card via Expansion connector
- SPDIF output via HDMI add-on card
- Freescale SGTL5000 Audio Codec
- Microphone jack
- Headphone jack
- Enables Parallel LCD or HDMI output
- Camera CSI port signals
- I2C, SSI, SPI signals
- Full-size SD/MMC card slot
- Micro SD card slot
- 7-pin SATA data connector
- 10/100BT Ethernet port
- 2x High-Speed USB Host port
  1x Micro USB Device port

i.MX28EVK-399USD

•i.MX28x

- i.MX28 Applications Processor (289 BGA)
- DDR2
- NAND FLASH
- SPI Flash footprint
- ETM Support
- DC/DC Converter components
- Li-Ion battery connector

freescale semiconductor

i.MX28 Evaluation Kit (EVK)

Price. Performance. Personality.		
CPU	**Debug**	**Peripherals**
► i.MX28 Applications Processor (289 BGA) ► DDR2 ► NAND FLASH ► SPI Flash footprint ► ETM Support ► DC/DC Converter components ► Li-Ion battery connector	► Debug Serial Port ► JTAG ► Reset, Interrupt, boot switches ► Debug display/LED's ► Power Source	► WVGA Touchscreen LCD Display (add-on module) ► SD/MMC Card Slot ► Dual USB Host/Device connector ► CAN connector ► Dual Ethernet with Switch for testing of features and throughput ► Navigation keys ► Line input, headphone output (jack)

MCIMX28EVK	i.MX28 Evaluation Kit	MSRP $399
MCIMX23LCD	4.3" WVGA Touchscreen LCD Display (add-on module)	MSRP $199

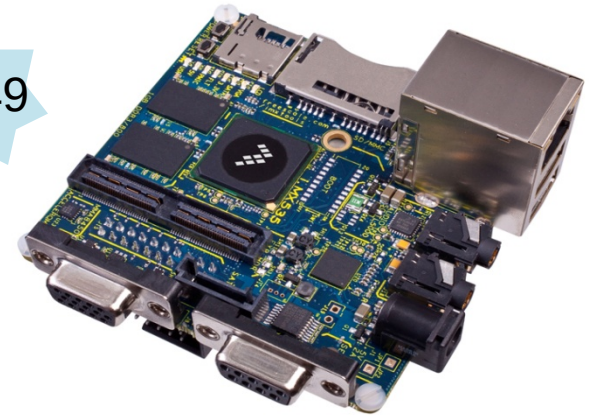
SW Tools support:

- Freescale Linux and WinCE BSPs
- Multimedia codecs
- MP4, AVI, WMA, AAC, MP3
- IEEE 1588 Support (IXXAT)

freescale semiconductor

i.MX53 Quick Start Board Features

$149



Key Features
- i.MX53 1Ghz Cortex-A8 Processor
- Dialog DA9053 PMIC
- 1 GB DDR3 Memory
- 3” x 3” 8-layer PCB

Display
- LVDS connector
- VGA connector
- Parallel LCD add-on card via Expansion connector
  - 24 bit 4.3” 800x480 WVGA with 4-wire touch screen
  - Part # MCIMX28LCD
- HDMI add-on card via Expansion connector
  - 24 bit HDMI output port also contains SPDIF audio
  - Part # MCIMXHDMICARD

Audio
- SPDIF output via HDMI add-on card
- Freescale SGTL5000 Audio Codec
- Microphone jack
- Headphone jack

Expansion Connector
- Enables Parallel LCD or HDMI output
- Camera CSI port signals
- I2C, SSI, SPI signals

Connectivity
- Full-size SD/MMC card slot
- Micro SD card slot
- 7-pin SATA data connector
- 10/100BT Ethernet port
- 2x High-Speed USB Host port
- 1x Micro USB Device port

Debug
- JTAG connector
- DB-9 UART port

Miscellaneous
- **Freescale 3-axis Accelerometer**
- Power Supply 5V, 2A
  - Included in the kit along with worldwide adapters

OS Support
- Ubuntu from Freescale
  - 4GB micro SD card with image included in the kit
- Android 2.2 from Adeneo
- Windows Compact 7 from Adeneo

Tools Support
- Segger/CodeSourcery , Macgraigor, IAR debug/IDE tool chain
- Mentor/Inflexion GUI development tool
- VMware player to bring up image on a Windows PC

Supported by *imxcommunity.org*

freescale semiconductor

Software Development Kit

Product-worthy software components to support reference design or product development.

- **Documentation:** Release notes, user/reference guides, data sheets
- **Development Tools:** FSL tools and listing of 3rd party tools
- **Demo Applications:** set of apps for demos or to serve as starting point for customers

- **Middleware:** Gstreamer or WinCE framework, multimedia codecs, connectivity protocol stacks, wireless applications, power management

- **BSP:** standard O/S optimized with additional drivers to support peripherals on Personality Module

Freescale | Third Party or Open Source

Freescale i.MX Software Expertise

- Focus on Linux®, Android and Windows® Embedded CE based solutions
- Enable customers with differentiating silicon capabilities
- Board support packages (BSP) consisting of kernel optimizations, hardware drivers and unit tests in a common i.MX code base
- Extensive portfolio of optimized multimedia codecs
- Middleware – multimedia framework plug-ins, power management, security/DRM, graphics (OpenVG/OpenGL-ES), connectivity
- Comprehensive testing and validation of the platform
- Broad ecosystem of software partners
- Well defined support through applications engineering team
- Available on external web at no cost

freescale semiconductor

i.MX Linux and Windows Embedded CE BSPs

Features

- Out-of-the-box integrated Linux/Windows Embedded CE environment – tools + kernel + drivers
- Extensively tested, hardened and validated
- Optimized for target platforms
- Accelerated Codecs support
- Common code base across different i.MX SoCs

Packages

- Boot loader binaries and source files
- Source and patches for kernel and file system
- Source code for unit tests of the drivers
- Linux Target Image Builder (LTIB) and WinCE standard build system
- Proprietary 3rd party components in binary code format
- Prebuilt Binaries
- Tool chain for ARM9/ARM11/ARM12 (Linux – Open source/Codesourcery, Windows Embedded CE – Microsoft)
- BSP Documentation (Reference manual, User's Guide, release notes)

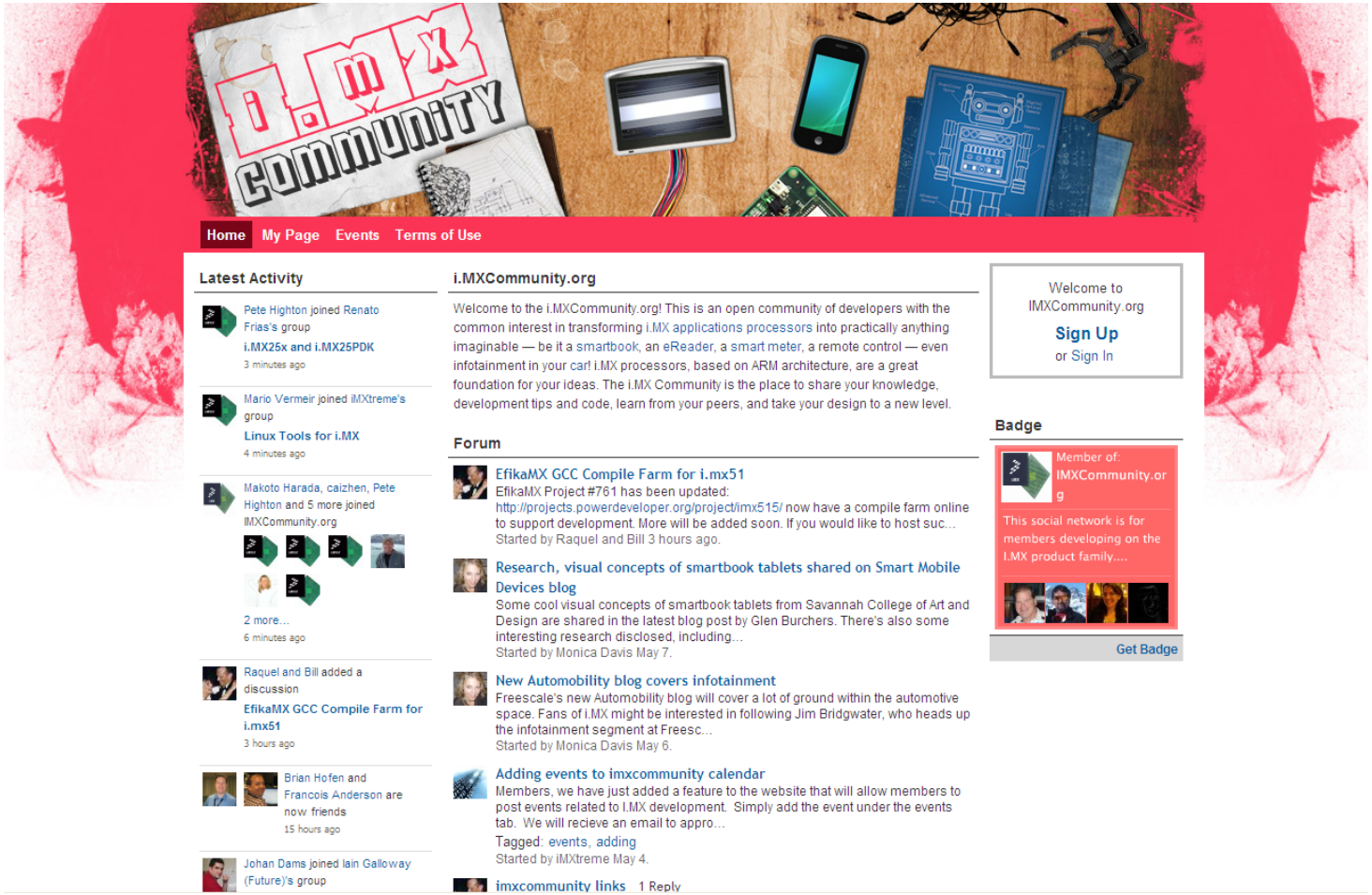
i.MX Optimized Multimedia Codecs

- Key Features
  - Comprehensive suite of optimized codecs (~40+ Audio/Video/Image Codecs)
  - Highly optimized software that is coded by Freescale processor experts
  - Consistent application programming interface (API) and frameworks across all software packages including OpenMAX support
  - Codec APIs are optimized from system design perspective and achieve optimal system performance along with related middleware wrappers
  - Codecs are supplemented with Freescale development tools, sample test streams and documentation

- Codec Software Packages include:
  - Codec libraries with a standard C-callable API
  - Gstreamer and OpenMAX plugins that provide an API layer between the multimedia framework and the codec library
    - Gstreamer is LGPL and allows for proprietary codecs integration through dynamic linking
  - Audio/video file containers (parsers) that support popular multimedia content, such as .aac, .avi, .asf, .mp3 and .mp4 files
  - Bundle of Freescale audio/video sample test streams
  - Complete documentation, including API documentation, release notes and data sheets

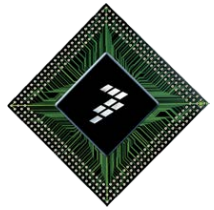
i.MX Community

http://www.iMXcommunity.org



freescale semiconductor

i.MX28 Overview



freescale™ semiconductor

i.MX28 Target Applications

Industrial

- Smart Energy Gateways/Meters
- HMI (Factory Automation & Building Control)
- Industrial Control

Home & Office

- HMI (Appliances, Security Panels, Automation)
- Portable Medical
- Media Gateways/Accessories

Point Of Sale

- Data Acquisition (Scanners)
- Fixed and Handheld Printers

Automotive

- Audio Connectivity
- CAN Gateways

freescale semiconductor

i.MX28 Family – Value Proposition

The new i.MX28 processor family reaches new levels of integration in an ARM9 device, with on-chip display, power management and connectivity features. Easy-to-use tools and software help you design differentiated industrial, automotive and consumer products in less time.

Industrial-Strength Integration

- WVGA LCD controller with touchscreen for display-centric applications
- Numerous connectivity options including dual 10/100 Ethernet (1588 capable) with L2 switch

Industry-leading Power Management

- Integrated power management simplifies customer design and saves on system cost
- <0.5 W performance under harshest conditions

Comprehensive Enablement

- Software BSPs and multimedia codecs available and supported by Freescale at no added cost
- Freescale-owned development system priced at <$400 include access to all design and layout files.

freescale semiconductor

i.MX28x Family

Key Features and Advantages

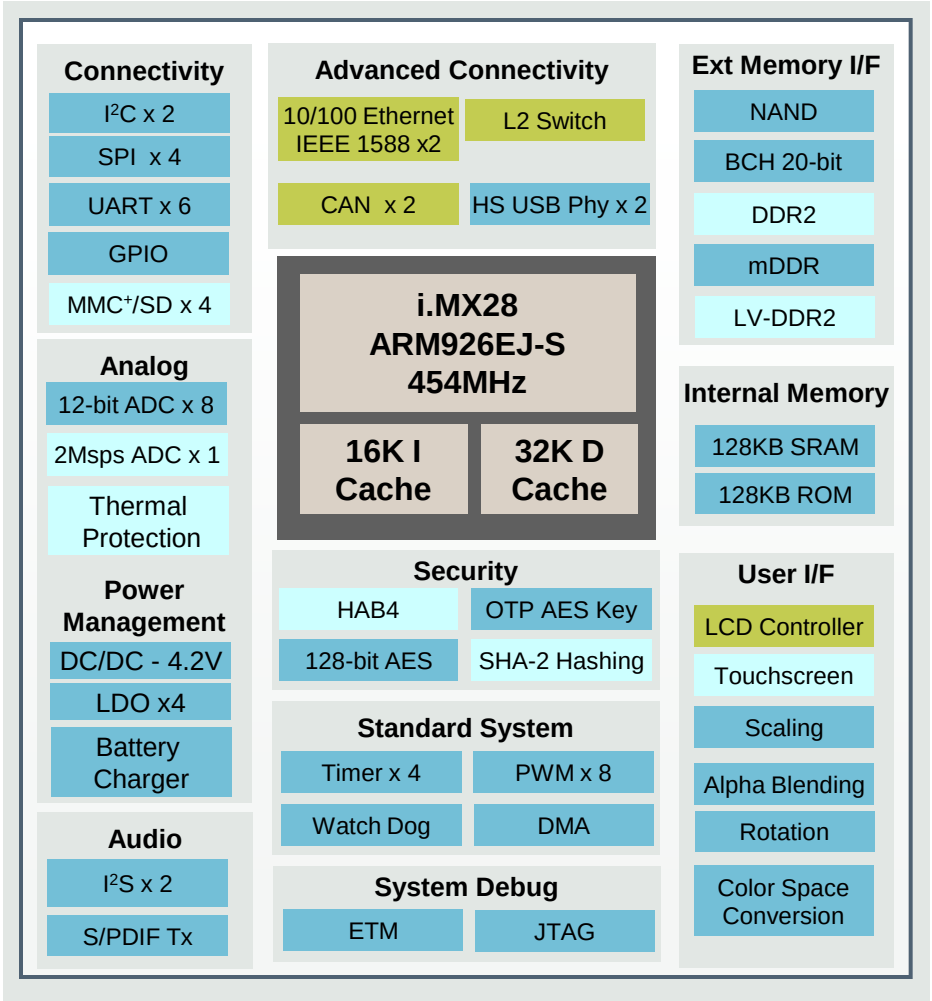
- 454MHz ARM926EJ-S core w/ 32KB Cache
- PMU with high efficiency on-chip DC/DC, supports Li-Ion batteries
- 10/100 Dual IEEE 1588 Ethernet with RMII support and L2 Switch
- Dual CAN interfaces
- LCD Controller with Touchscreen
- NAND support – SLC/MLC and eMMC 4.4 managed
- Hardware BCH (up to 20-bit correction)
- 200 MHz 16-bit DDR2, LV-DDR2, mDDR external memory support
- Dual High speed USB with embedded PHY
- Up to 8 General purpose 12-bit ADC channels and single 2 Msps ADC channel
- LCD Controller with Touchscreen
- Temperature sensor for thermal protection
- Multiple connectivity ports (UARTs, SSP, SDIO, SPI, I2C, I2S)
- Family of products supporting various feature sets

Package and Temperature

- 289 BGA 14x14mm .8mm
- -40C to +85C (Industrial, Automotive)
- -20C to +70C (Consumer)

Availability:

- Alpha Samples: Now
- Production: early Oct 2010 (consumer), end Oct 2010 (industrial)



Common IP with i.MX233 | New or enhanced from i.MX233 | Not available on all variants

freescale semiconductor

i.MX28 Family Product Comparison

Feature	i.MX283	i.MX286	i.MX287
On-chip RAM	128KB	128KB	128KB
Memory Interface	NAND Flash, DDR2, mDDR, LV-DDR2	NAND Flash, DDR2, mDDR, LV-DDR2	NAND Flash, DDR2, mDDR, LV-DDR2
LCD Interface	Yes	Yes	Yes
Touchscreen	Yes	Yes	Yes
Ethernet	x1	x1	x2
L2 Switch	-	-	Yes
CAN	-	x2	x2
12-bit ADC	x8	x8	x8
High Speed ADC	x1	x1	x1
USB2.0	OTG HS with HS PHY x1 HS Host with HS PHY x1	OTG HS with HS PHY x1 HS Host with HS PHY x1	OTG HS with HS PHY x1 HS Host with HS PHY x1
SDIO*	x3	x3	x4
SPI*	x3	x3	x4
UART*	x6	x6	x6
PWM*	x8	x8	x8
S/PDIF Tx	-	Y	Y
Package	14x14 0.8mm 289 BGA	14x14 0.8mm 289 BGA	14x14 0.8mm 289 BGA

* Represents maximum available – some pins are shared with other interfaces

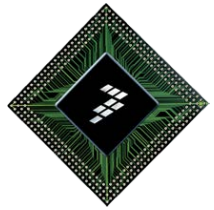
freescale semiconductor

i.MX28 Automotive Family Product Comparison

Feature	i.MX281	i.MX285
On-chip RAM	128KB	128KB
Memory Interface	NAND Flash, 16-bit 150MHz DDR2, mDDR	NAND Flash, 16-bit 200 MHz, DDR2, mDDR, LV-DDR2
LCD Interface	-	Yes
Touchscreen	-	Yes
Ethernet	x1	x1
CAN	x2	x2
12-bit ADC	x5	x5
High Speed ADC	x1	x1
USB2.0	OTG HS with HS PHY x1 HS Host with HS PHY x1	OTG HS with HS PHY x1 HS Host with HS PHY x1
SDIO	x4	x4
SPI	x4	x4
UART	x3	x3
PWM	x3	x3
S/PDIF Tx	Yes	Yes
Package	14x14 0.8mm 289 BGA	14x14 0.8mm 289 BGA

freescale semiconductor

Hands-on, Getting to know i.MX28EVK



freescale™ semiconductor

i.MX28 Evaluation Kit (EVK)

Price. Performance. Personality.

CPU	Debug	Peripherals
► i.MX28 Applications Processor (289 BGA) ► DDR2 ► NAND FLASH ► SPI Flash footprint ► ETM Support ► DC/DC Converter components ► Li-Ion battery connector	► Debug Serial Port ► JTAG ► Reset, Interrupt, boot switches ► Debug display/LED's ► Power Source	► WVGA Touchscreen LCD Display (add-on module) ► SD/MMC Card Slot ► Dual USB Host/Device connector ► CAN connector ► Dual Ethernet with Switch for testing of features and throughput ► Navigation keys ► Line input, headphone output (jack)

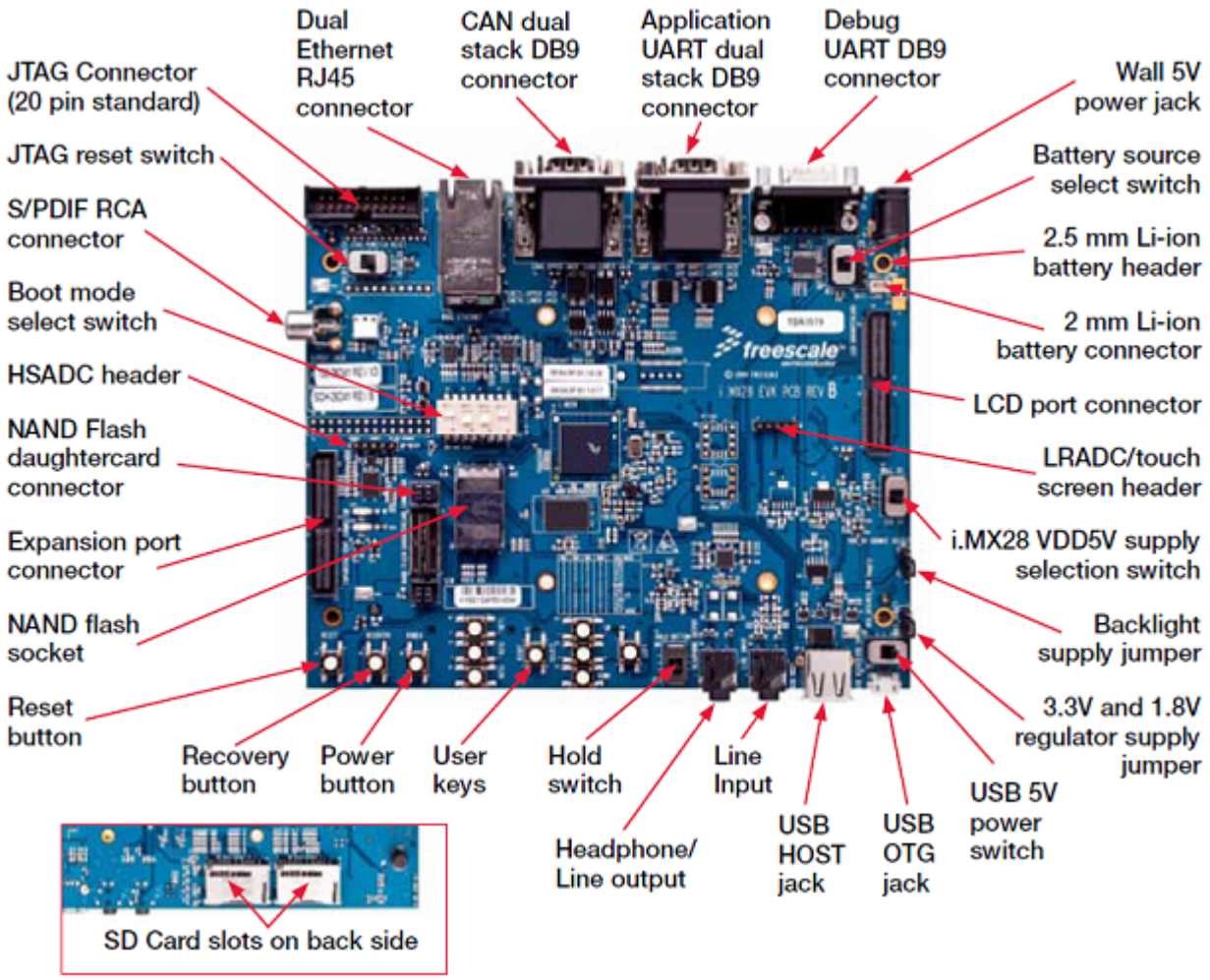
MCIMX28EVK	i.MX28 Evaluation Kit	MSRP $399
MCIMX28LCD	4.3" WVGA Touchscreen LCD Display (add-on module)	MSRP $199

Software:

- Freescale Board Support Packages (BSPs)
  - Linux
  - Windows Embedded CE
- Freescale Multimedia Codecs
  - Audio Codec: MP3, AAC, WMA
  - Video Codec: MPEG4, H264
- IEEE 1588 Demo (IXXAT)

freescale semiconductor

Get to know the i.MX28 board



freescale semiconductor

Default Switch Setting

Reference Designator	Name	Setting				Description
S1	Battery source select switch	BATT				The battery source is a real Li-ion battery connected to either J85 or J86.
		REG				The battery source is the EVK on-board 4.2V regulator.
S2	Boot mode select switch	*B3*	*B2*	*B1*	*B0*	
		1	*0*	*0*	*1*	The i.MX28 processor will boot from the i.MX28 SSP0 port, which is SD card Socket 0.
		-	-	-	-	For other boot mode switch settings, refer to the silkscreen table on the EVK board.
S3	USB 5V power switch	ON				USB 5V power from the USB micro AB jack is connected to the EVK board.
		OFF				USB 5V power from the USB micro AB jack is disconnected from the EVK board.
S7	Hold switch	ON				The user keys are locked.
		OFF				The user keys are unlocked.

Default Switch Setting, contd.

Reference Designator	Name	Setting	Description
S16	i.MX28 VDD5V pin supply selection switch	Wall 5V	The i.MX28 VDD5V pin is supplied by the 5V DC wall adapter.
		USB 5V	The i.MX28 VDD5V pin is supplied by the USB micro AB jack.
S17	JTAG reset switch	RESET ENABLED	The JTAG tool nSRST (system RESET) signal line is connected to the i.MX28 PSWITCH pin.
		RESET DISABLED	The JTAG tool nSRST (system RESET) signal line is disconnected from the i.MX28 PSWITCH pin.
J94	3.3V and 1.8V EVK regulator power source jumper	BATT	Pins 2–3 are shorted (upper position) and the battery is used to supply the power.
		VDD5V	Pins 1–2 are shorted (lower position) and the 5V DC wall adapter is used to supply the power.
J95	LCD backlight power source jumper	Wall 5V	Pins 1–2 are shorted (upper position). The 5V DC wall adapter supplies power to the LCD backlight.
		VDD4P2	Pins 2–3 are shorted (lower position). The i.MX28 VDD4P2 output supplies power to the LCD backlight.

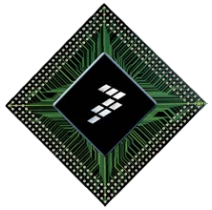
i.MX28EVK Development Kit Contents

- i.MX28 EVK Board
- SD Cards
  - Windows® Embedded CE demo image
  - Linux® demo image
- Cables
  - Ethernet straight cable
  - USB cable (micro-B to standard-A)
- Power Supply
  - 5.0 V/3.8 A Universal Power Supply
- Documentation
  - End-User License Agreement
  - Quick Start Guide
  - Warranty Card
  - Freescale Support Card
- DVD with Getting started video tutorials and useful web links

i.MX28EVK on Freescale.com

- Use www.freescale.com/imx28evk for latest documentation and BSP releases for i.MX28
- Software Releases
  - WindowsCE SDK
  - Linux SDK
  - Manufacturing Tool
- i.MX28EVK Documentation
  - Schematics, layout, and gerber files for EVK and LCD board
  - i.MX28 EVK Hardware User's Guide
  - i.MX28 Windows CE docs: Demo Image Readme, Quick Start Guide, Release Notes, User's Guide, Reference Manual, Hello World Application Note
  - i.MX28 Linux docs: Demo Image Readme, Quick Start Guide, **Release Notes**, **User's Guide**, **Reference Manual**, Hello World Application Note
- Demo images

i.MX28 Peripherals



freescale™ semiconductor

i.MX28 – CPU Subsystem

- ARM926EJ-S processor with up to 454MHz performance
- Custom caches for maximum performance and low power (16K + 32K)
  - Improves video decode modeling results
- Low-power 90LP implementation
- 128KB of On-Chip SRAM
- Vectored interrupt controller with 128 fully programmable sources and up to 4 levels of IRQ nesting
- Coresight ETM9 for higher-speed trace (DDR data, better compression) debug
- Standard 6-wire JTAG for debug
- Support wait-for-interrupt low-power mode

ARM

freescale semiconductor

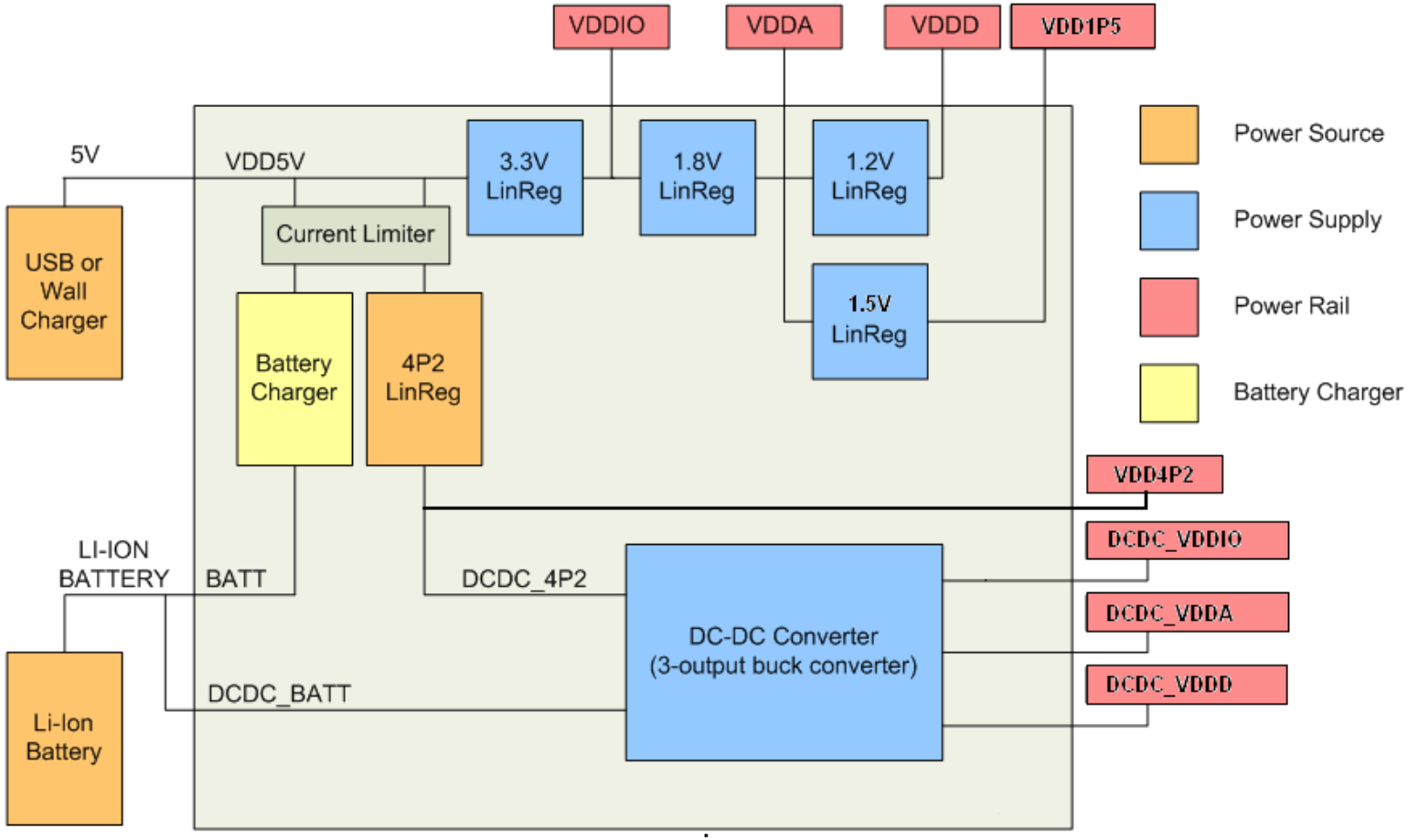
i.MX28 – Low Power Features/Characteristics

- Supports dynamic voltage frequency scaling (DVFS) which provides the most efficient power per MIPS for the application
- Architectural and automated clock gating
- External memory controller implement five levels of low-power modes (mDDR)
- Synchronous clocking mode from CPU, bus to memory controller, reduces latency and thus MHz/MIPS requirements
- Auto-slow on bus-clock (HCLK) with HW controlled slow-down/speed-up based on bus activity
- Wait-for-Interrupt standby mode system power = 2mA (~7.5mW)
  - CPU clock stopped, wakes up from interrupt
  - SRAM retained
  - Supports Interrupt from press
  - Supports wake-up from touchscreen
  - Quick power up
- Power-down (RTC-only) power = 12uA
  - Only RTC active
  - Power up longer

i.MX28 - Power Management Unit

- Integration of a DC-DC switching converter and linear regulators that provide four output rails
  - Powers digital blocks and components such as system clocks
  - Powers analog components and 1.8V DDR2
  - Powers I/O peripherals like NAND flash and SD/MMC cards
  - Powers 1.5V LV-DDR2
- Power sources
  - Li-Ion batteries (2.9V – 4.2V)
  - Direct power from 5V source (USB, wall power or other source)
  - Internal 4.2V power source generated from 5V source
- Battery charging capability
  - Allows battery to be fully charged while device is in use
  - Current and voltage sensors allows firmware to monitor the voltage and current into the battery to determine “charged” status
- On-chip silicon speed and temp sensors
  - Hardware thermal protection and shutdown circuitry

i.MX28 Power Supply Block Diagram



Notes:

1. DCDC_VDDIO connected to VDDIO on PCB
2. DCDC_VDDA connected to VDDA on PCB
3. DCDC_VDDD connected to VDDD on PCB

freescale semiconductor

Battery Boot

- Li-Ion battery attached with no 5V present.
- Triggered by PSWITCH, RTC Alarm, or AUTO_RESTART bit
- Three stages of battery boot
  - Internal circuitry performs power-on sequence for rails and DC-DC
  - DC-DC takes control and provides power to rails.
  - Software executes and finishes DC-DC initialization and configuration

5V Boot

- VDD5V > ~3.9V causes chip to begin booting.
- Starts by using linear regulators, then switches to DC-DC power
- Three stages of 5V Boot
  - Internal PMU linear regulators pull output rails up to default levels.
  - Linear regulators supply output rail power.
  - DC-DC is initialized by software

Boot Modes Introduction

- Boot mode is specified from one of two sources:
  - External resistors (development)
  - OTP bits (production)
- The ROM processes an .sb formatted *boot stream* read from the selected boot source.
- OTP bits set both general and source-specific boot options:
  - Enable unencrypted boot
  - Number of NANDs
  - Enable internal pull-ups
  - MMC/SD bus width and speed
  - Much, much more…

Boot Modes

USB	USB device boot
I²C	EEPROM
SSP0	SD/MMC
SSP1	SD/MMC
SSP2	SPI Flash
SSP3	SPI Flash/EEPROM
GPMI	NAND boot
JTAG	ROM waits for JTAG connection

- Both 3.3V and 1.8V devices
- 1-, 4-, and 8-bit SD/MMC
- x8 NAND

Secure Boot

- Two secure boot options
  - AES-128 encryption of boot stream
  - HAB4 code signing
- Both AES and HAB4 can be used simultaneously
- OTP holds AES-128 key and/or HAB4 signature
- Chip defaults
  - AES encrypted boot required (with default key of all zeros)
  - HAB4 enabled in FAB mode (IVT must be present)

Boot Stream

- Out of reset, internal ROM is executed first
- Internal ROM decides what is boot source
  - Decision is based on value of boot mode pins or eFuse settings
- Internal ROM will locate, retrieve and execute the boot stream which consists of small bootlets
- Bootlets are small pieces of code executed during boot to set up some basic system functionality (memory, clocks, etc.)
  - Each bootlet is built separately and may or may not know about others
  - Boot stream can instruct the ROM to call any number of bootlets before final jump to
- i.MX28 boot streams contain following bootlets:
  - power_prep – configures the power supply
  - Boot_prep – configures clocks and SDRAM
  - Linux_prep – prepaires to boot Linux

freescale semiconductor

Bootlets

- Any executable program designed to run in the ROM context.
- A bootlet must be linked to run in and only use OCRAM, unless the bootlet is sequenced to run after SDRAM is ready (i.e., after the **sdram_prep** bootlet).
- The intent is to perform some service or function and then return back to the ROM so the boot can continue.

- For example:
  - **power_prep** - Puts the power supply in a good, functional & protected power state.
  - **sdram_prep** - Initializes SDRAM and thus makes it available for use.
  - **linux_prep** – Helps with re-starting the Linux kernel from sleep mode.

Boot Sequence

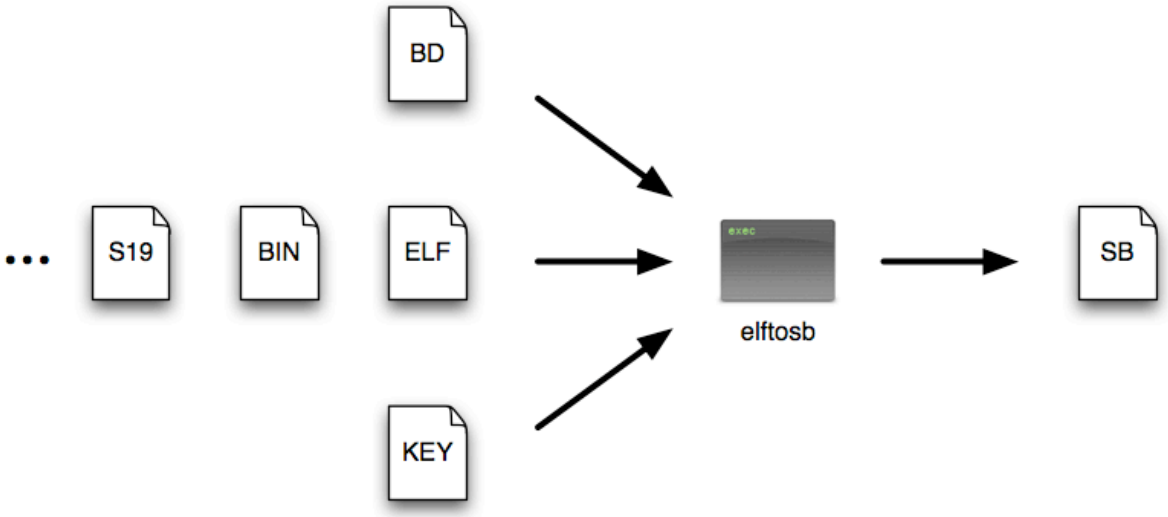
- Typical GNU/Linux boot sequence
  - power_prep bootlet
  - sdram_prep bootlet
  - linux_prep bootlet
  - zImage (loaded by linux_prep)
- Does not use a bootloader like U-boot.

.sb File

- Binary *boot stream* image format.
- Same format as used by i.MX23.
- Created by **elftosb** utility
- Consists of a header followed by a series of sections that are executed by the ROM
- Main commands are:
  - **Load** – Load text/data into RAM
  - **Call** – Function call to an address. Returns to ROM when done
  - **Jump** – Jump to an address. Cannot return to the ROM
- Usually AES-128 encrypted
- Contains encrypted SHA-1 hash of entire image

elftosb Utility

- Program (Win32/Linux/OS X) used to create .sb file.
- Uses .bd file as input to control .sb file generation.
- Optionally takes an AES-128 key stored in a small text file.
- Input data consists of various file formats such as ELF, S-record, and raw binary.
- For i.MX28, elftosb supports special commands to support HAB4.



freescale semiconductor

USB Recovery Mode

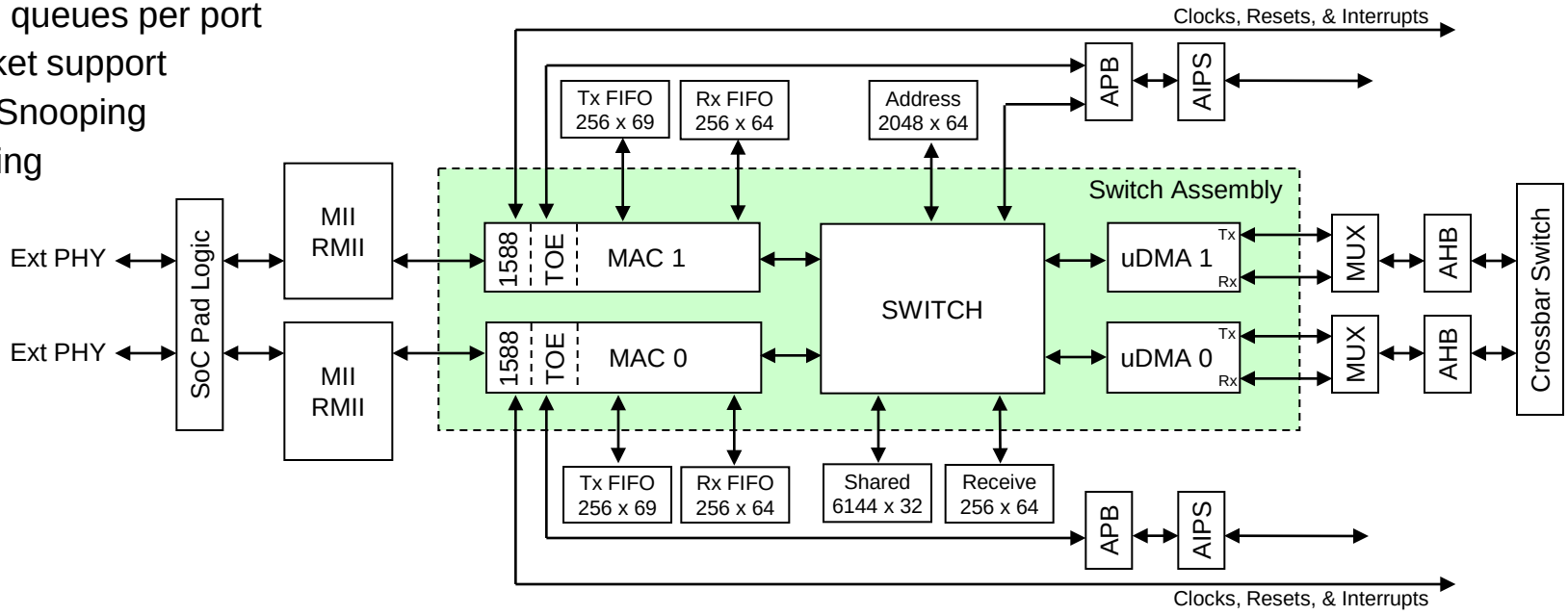
- This is a fail-safe mechanism for running a program on the system by downloading it via USB.
- Often used to write a new firmware image to the boot media.
- Entered by having USB attached and selecting USB boot mode.
- Automatically entered if the ROM cannot find a valid boot stream on the selected boot media, or if the ROM gets an error while booting.
- This boot mode can be permanently disabled by burning the `DISABLE_RECOVERY_MODE` OTP bit.
- If USB recovery mode is disabled via OTP and the ROM cannot find a valid boot stream on the selected boot media, the ROM will power down the chip.

Embedded Ethernet Assembly w/ L2 Switch

Product Features:

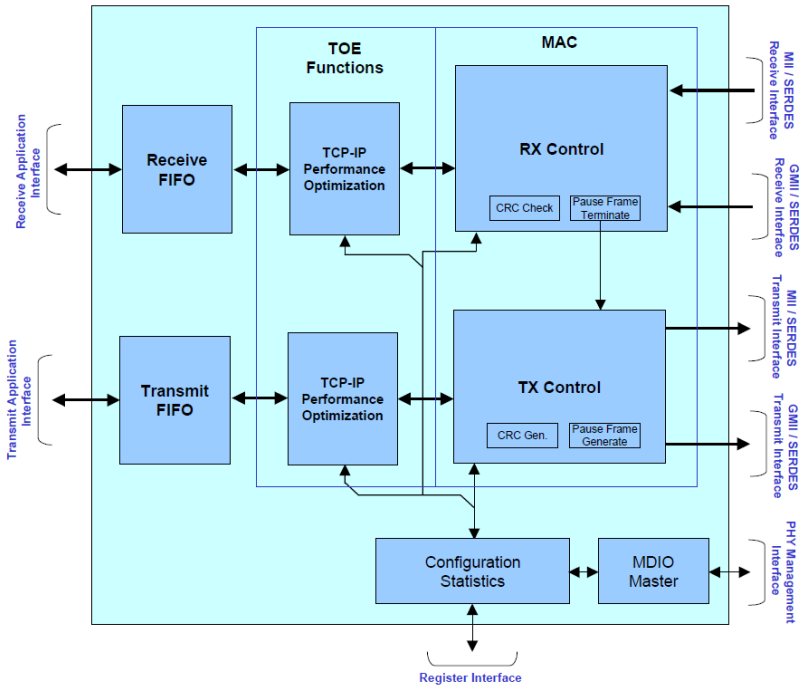
- 3-Port Switch (one internal)
- Separate dual-port FIFOs for max throughput
- TCP/IP Offload Engine (TOE)
- Hardware Time-stamping (IEEE 1588)
- Simple handshake programmable FIFO i/f
- Fast cut-through mode (MAC)
- Link aggregation, redundant backplane
- QoS with 8 queues per port
- Magic packet support
- Level 3 IP Snooping
- Port Mirroring

- **System Benefits:**
  - ***Cost-effective daisy-chain networks***
  - ***Efficient ring networks with redundancy***
  - ***Improved determinism using hardware time stamping of packets (1588)***



freescale semiconductor

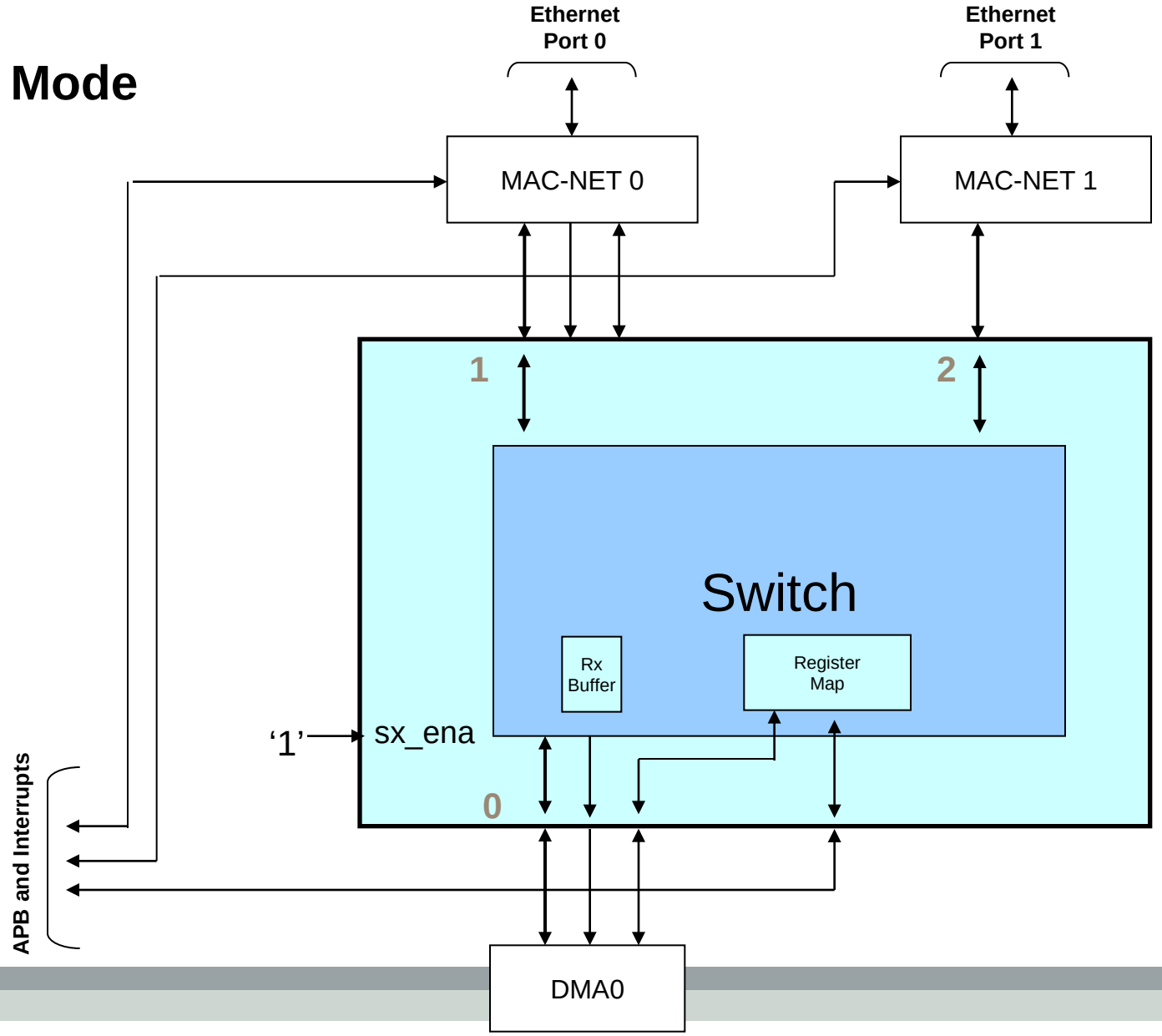
Ethernet Controller



- • **Two Ethernet controllers** with 10/100 Etthernet MAC BaseT/TX capability; half duplex or full duplex
- Hardware support for IEEE 1588
- Media independent interface (MII) and reduced media independent interface (RMII) support
- Built-in unified DMA
  - On-chip transmit and receive FIFOs
  - Supports legacy buffer descriptor programming models and functionality
  - Enhanced buffer descriptor programming model for new Ethernet functionality
- Supports wake-up from low power mode through magic packets
- During chip reset, ability to route traffic from one port to another port
- Multiple clock source options for time-stamping clock IEEE 1588 Functions
- Reference Clock can be chosen independently of the Network speed
- Software Programmable Precise Time-Stamping of Ingress Frames and Egress Frames
- Timer monitoring capabilities for System calibration and timing accuracy management
- Precise time stamping of external events with programmable interrupt generation
- Programmable event and interrupt generation for external system control
- Hardware and Software controllable timer synchronization

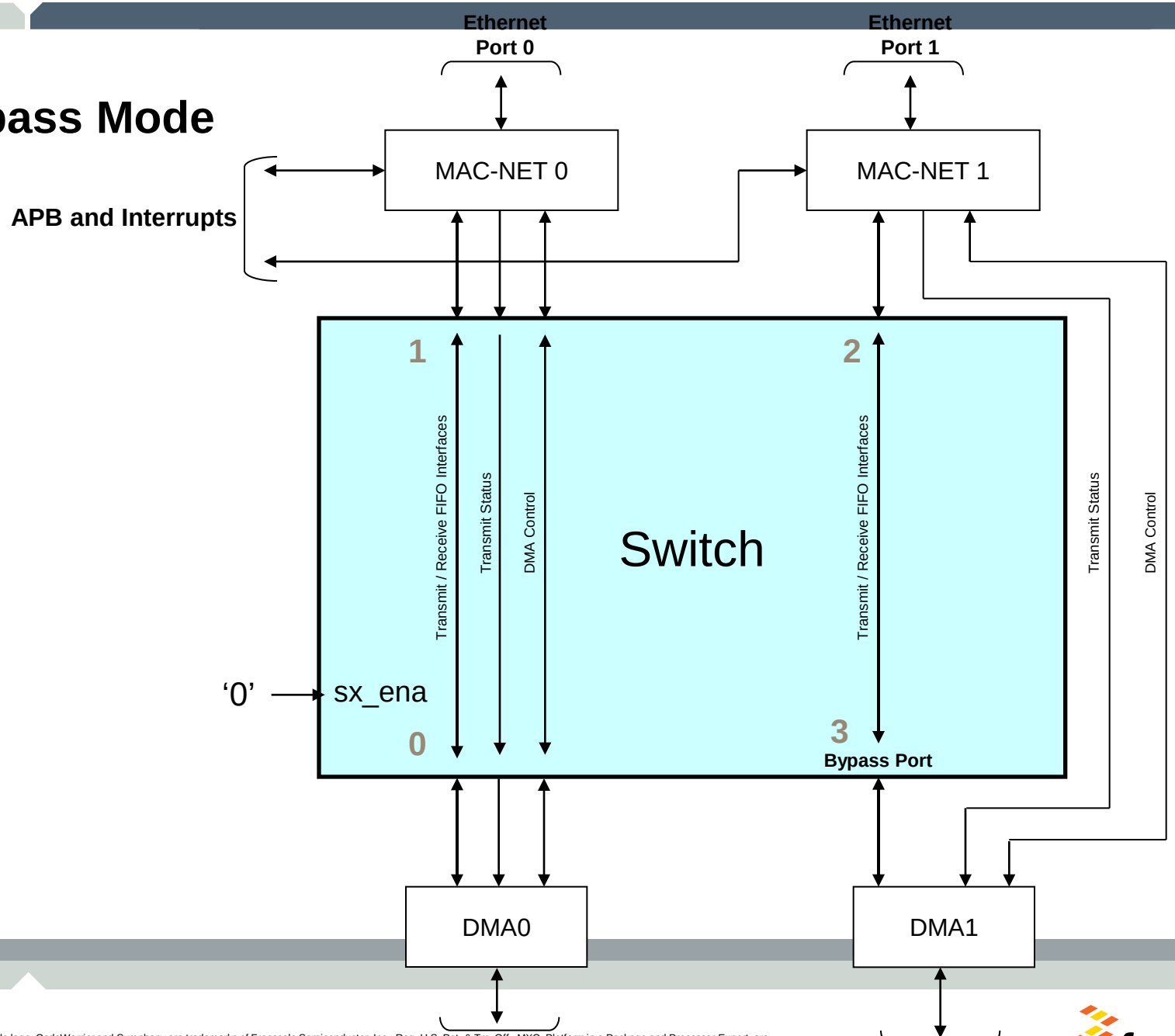
freescale semiconductor

Switch Mode



freescale semiconductor

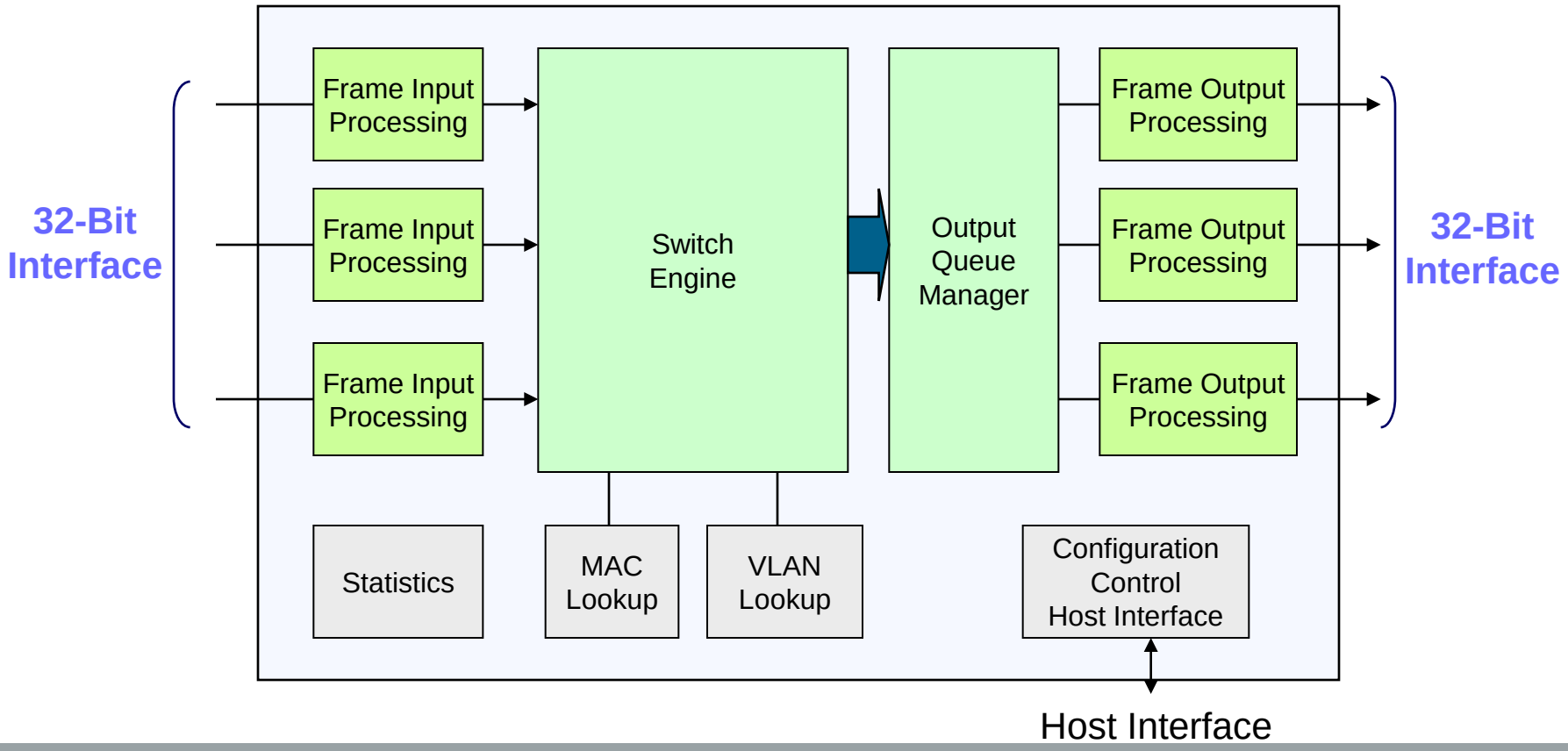
Bypass Mode



freescale semiconductor

Switch Processing Overview

PHY → Classification Scheduling Manipulation → Filtering / Forwarding Congestion → Storing Scheduling → Manipulation Shaping →



freescale semiconductor

Input Processing Options
Traffic Domain Protection and Traffic Mapping

- VLAN Manipulation

Inspect & Manipulate VLAN tag of frames entering the Switch

  - Tag Insertion (Single / Stacked VLANs)
  - Tag Verification
  - Tag Modification

- Priority Classification
  - VLAN Priority Extraction
  - IP DiffServ / Class of Service (CoS) Extraction
  - Priority Remapping to available Output Priorities
- Frame Parsing (Blocking, Broadcast Storm Protection)

Forwarding Switching Tasks

- MAC Lookup
- VLAN Lookup
- VLAN Verification
- Forwarding
  - Broadcast / Multicast Domain Resolution
  - VLAN Domain Resolution
  - Mirroring / Management Resolution
  - Output Port Selection
  - Discard
- Statistic Gathering

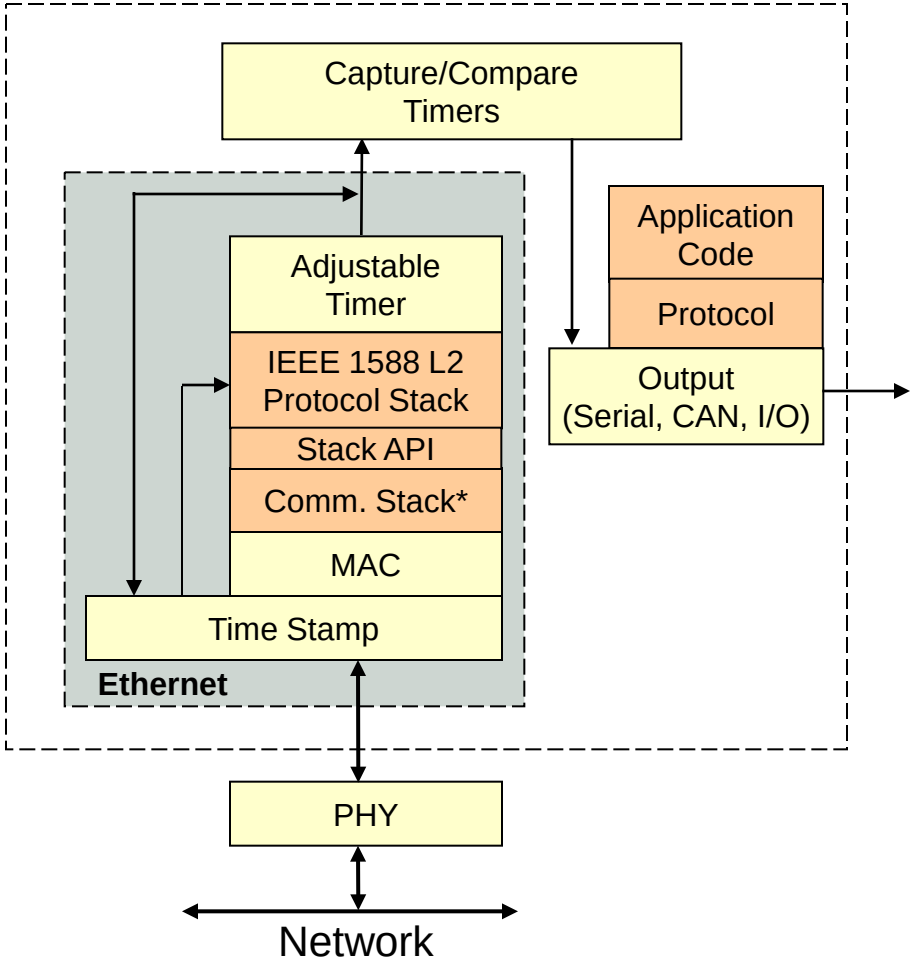
Ethernet Time-Stamping Capability

HARDWARE

- The 1588 stack can adjust and update the internal timer used as a local clock reference.
- The 1588 time stamp is implemented as close to the external MAC as possible, to provide maximum precision.
- Each network connection can run independently, together, or with embedded L2 switch.
- Internal capture/compare timers allow for other outputs to be synchronized to the network time base.

SOFTWARE

- Supports standard TCP/IP protocol, UDP protocol, and various proprietary protocols. *
- Time Stamping (1588 based) stack easily integrated into new IP stacks using the API provided.



freescale semiconductor

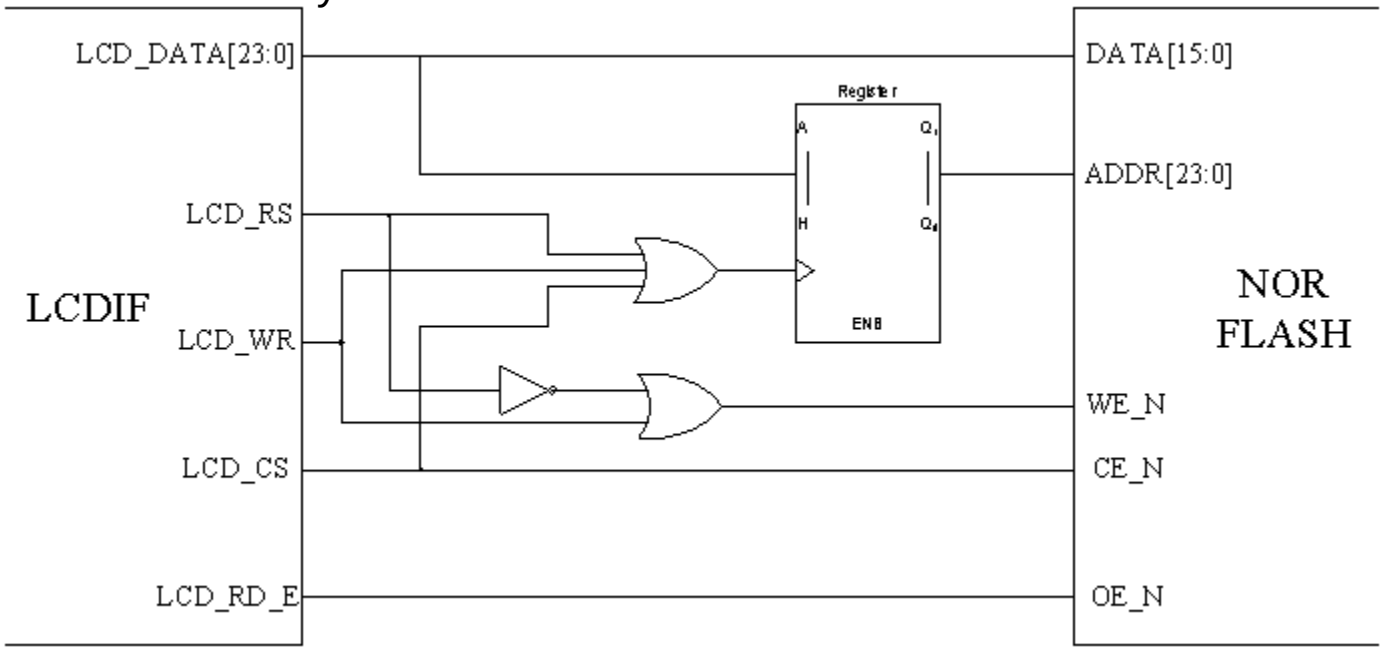
i.MX28 – Memory Support

- Off-Chip Memory Interface
  - Supports 8-bank DDR2, mDDR, LV-DDR2
  - 200MHz, 16-bit wide data bus
- Flash Memory Types Supported
  - Raw SLC NAND
  - Raw MLC NAND
  - Managed NAND – eMMC 4.4, LBA
  - SPI NOR Flash or Parallel NOR via LCDIF
  - Supports up to four NAND Flash devices
- Hardware BCH Interface
  - Provides a forward error-correction function to improve the reliability of raw NAND memory that may be attached to the i.MX28
  - BCH Engine with up to 20-bit correction (2-bit increments) with 13-bit parity.
- On-Chip 128KB SRAM
  - Ideal for low-power LCD refresh
  - Improve algorithmic performance

freescale™ semiconductor

i.MX28 – External Bus Options

- Configurations not supported via a local bus / external memory bus
  - Attaching parallel NOR memory
  - Using external buses for connecting FPGAs and ASICs
- Alternative option
  - Use the LCD Interface as a bus interface
  - Lose functionality of the LCD Controller



freescale semiconductor

i.MX28 – Security

- Data Co-Processor (DCP) Peripheral
  - Hardware accelerated encryption/decryption and hashing functions (SHA-256) typically used for security
  - High Assurance Book (HAB4) – secure boot with authenticity checking
  - AES Crypto engine – AES boot provides encrypting the boot via a shared key
  - DCP OTP key can be used to wrap (encrypt) other keys or data
- On-Chip One Time Programmable (OTP) ROM
  - Enables / disables device functionality (i.e. JTAG)
  - Holds one customer specified encryption key
- SRAM storage for four additional temporary keys.
  - SW can select from the OTP or the additional keys via the descriptor interface.
  - Once the keys have been provisioned to the SRAM they cannot be read directly by SW. (Write-Once-Only)
- On-Chip Boot ROM
  - Integrated boot loader is only method for booting device
  - Can be configured to authenticate and decrypt boot images

i.MX28 – SSP

- Up to four independent Synchronous Serial Ports (SSP)
  - SD/MMC removable cards
  - eSD/eMMC/iNAND chips
  - SPI control and communication
    - Supports Winbond SPI dual/quad read modes up to 52MHz SCK frequency
  - Peripheral chips such as Wi-Fi or Bluetooth using SDIO
  - Dedicated DMA channels
  - Maximum clock rate of 50 MHz

SSP Pin Functions For Each Mode

PIN NAME	MOTOROLA SPI MODE	WINBOND SPI MODE	TI SSI MODE	SD/SDIO/ MMC/CE-ATA MODES	MS MODE
SSP_SCK	SCK	CLK	CLK	CLK	CLK
SSP_CMD	MOSI	DI (IO0)	MOSI	CMD	BS
SSP_DATA0	MISO	DO (IO1)	MISO	DATA0	DATA0
SSP_DATA1		WPn (IO2)		DATA1/IRQ	DATA1
SSP_DATA2		HOLDn (IO3)		DATA2	DATA2
SSP_DATA3	SSn0	SSn0	SSn	DATA3	DATA3
SSP_DATA4	SSn1	SSn1		DATA4	
SSP_DATA5	SSn2	SSn2		DATA5	
SSP_DATA6				DATA6	
SSP_DATA7				DATA7	
SSP_DETECT				CARD_DETECT	

FlexCAN in i.MX28

- **Full implementation of the CAN protocol** specification, version 2.0B
  - Standard data and remote frames
  - Extended data and remote frames
  - Zero to eight bytes data length
  - Programmable bit rate up to 1 Mbit/sec
  - Content-related addressing
- Includes 1056 bytes (64 message buffers) for **message buffer storage**
- Includes 256 bytes (64 message buffers) for individual **Rx mask registers**
- Powerful Rx FIFO **ID filtering**, capable of matching incoming IDs against either 8 extended, 16 standard, or 32 partial (8 bits) IDs, with individual masking capability
- Programmable **loopback mode** supporting self-test operation
- **Programmable** transmission **priority scheme**: lowest ID, lowest buffer number, or highest priority
- **Time stamp** based on 16-bit free-running timer
  - Counts bit clocks, saves status at beginning of identifier field

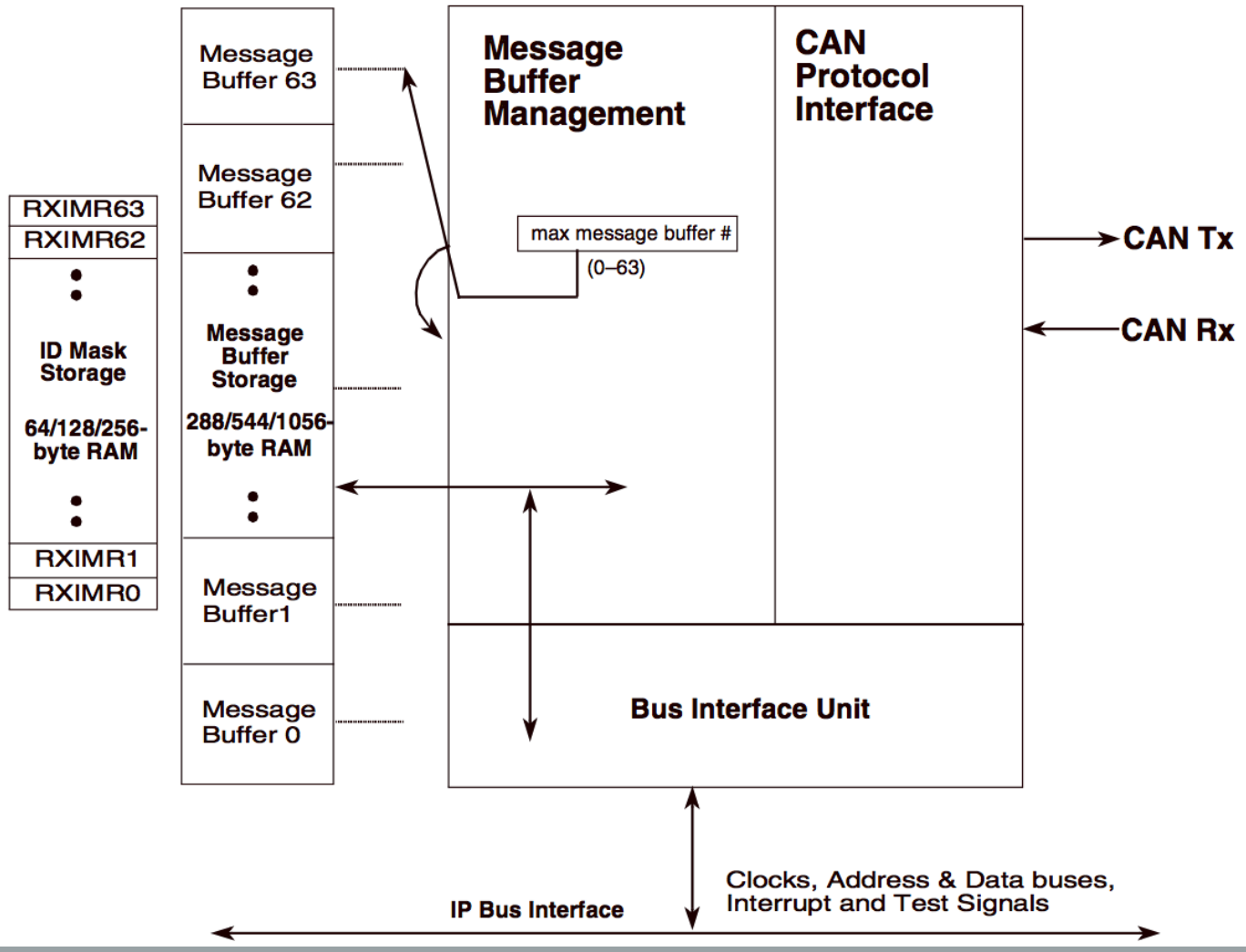
Operating Modes

- **Normal mode** (user or supervisor)
  - The module receives and/or transmits message frames, errors are handled normally, and all the CAN protocol functions are enabled
- **Freeze mode**
  - no transmission or reception of frames is done and synchronicity to the CAN bus is lost
- **Listen-only mode**
  - Only messages acknowledged by another CAN station are received
- **Loopback mode**
  - FlexCAN performs an internal loopback that can be used for self test operation
- **Module disable mode**
  - the module shuts down the clocks to the CAN protocol interface
- **Stop mode**
  - the module shuts down the clocks to the CAN protocol interface and the message buffer management sub-modules
  - Can wake up when activity is detected on the CAN bus and the self-wake-up mechanism is enabled

Hardware Signals

Signal Name	Direction	Description
CAN Rx	Input	CAN receive pin
CAN Tx	Output	CAN transmit pin

Block Diagram



freescale semiconductor

SAIF - Introduction

SAIF is a general purpose Serial Audio PCM Codec interface.

- Half duplex transmit or receive
- 3, 4, or 5 wire interface (BITCLK, LRCLK, SD0[, SD1, and SD2])
- 16- to 32- bit Data width
- Supports 2, 4, or 6 audio channels
- Supports I2S, Left justified, and Right justified data frame
- Sample rates from 8 Ksps to 192 Ksps
- Master and Slave bit clock and left-right clock modes
- DMA, PIO, and FiFo interrupt data sample transfers

SAIF – Block Diagram

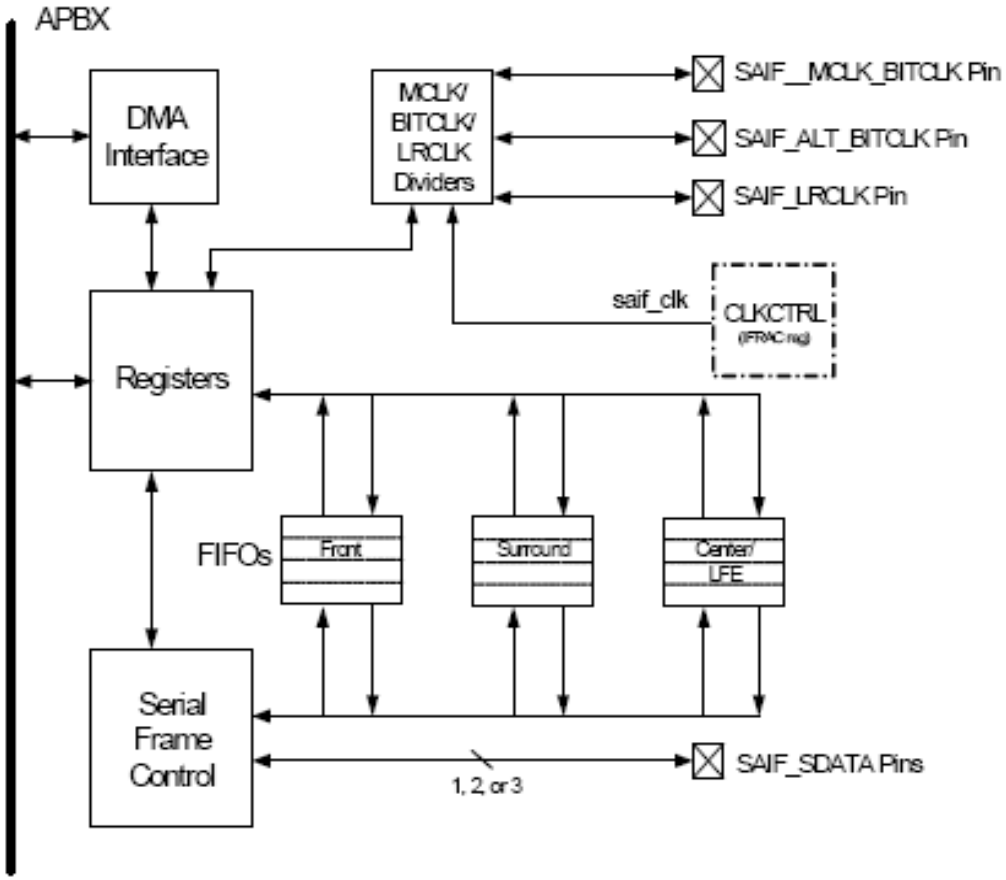


Figure 33-1. Serial Audio Interface (SAIF) Block Diagram

freescale semiconductor

SAIF – Frame Formats

SAIF supports 3 types of frame formats.

- I2S PCM format
  - MSB data bit on SAIF_DATAn is one BITCLK delayed from the transition of the LRCLK.
  - Trailing zero fills remaining data bits
- Left justified PCM format
  - MSB data bit aligns with the transitions of the LRCLK.
  - Trailing zero fills remaining data bits
- Right justified PCM format
  - LSB data bit aligns with the transition of the LRCLK
  - Leading zero bits

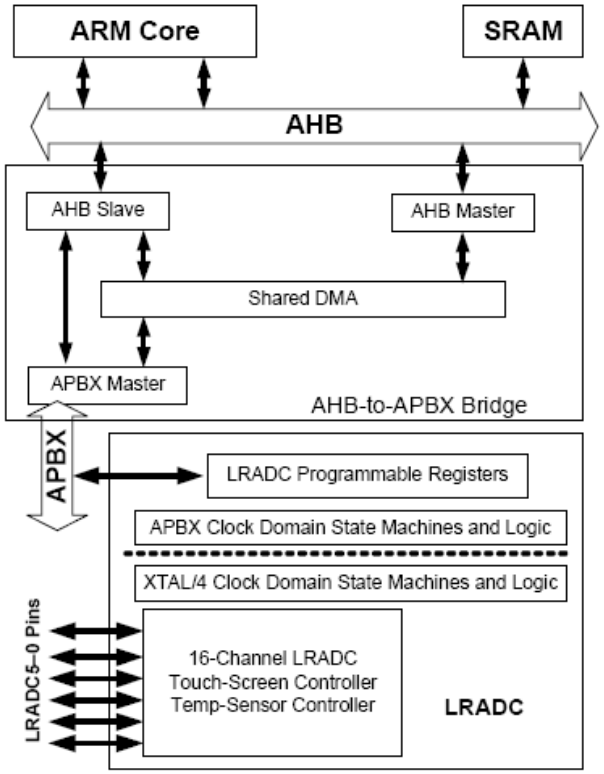
i.MX28 – Low Resolution A/D (LRADC)

- 12-bit Low-Resolution ADC, up to 0.5% battery level accuracy
- 16 total measurement nodes available
  - 8 physical channels available as external inputs
  - 8 “virtual” assignable channels for doing actual measurements can be mapped to any of the 16 measurement nodes
- Integrated 4-wire and 5-wire touch-screen controller (with wide range of impedance support, e.g. 200-400 Ohm and 50K Ohm)
- Integrated temperature sensor function (on-die, and external with diode or thermistor) to monitor the internal die temperature
  - Three sigma temperature error of +/-1.5% in degrees Kelvin
  - Temperature sampling has a 3 sigma sample-to-sample variation of 2 degrees Kelvin which can be averaged out .
  - Thermal protection on i.MX28 - Safety switch will reset the part when the shutdown temperature is reached
- Single channel high speed ADC – 2Msps at 12-bits

freescale semiconductor

ADC System Overview

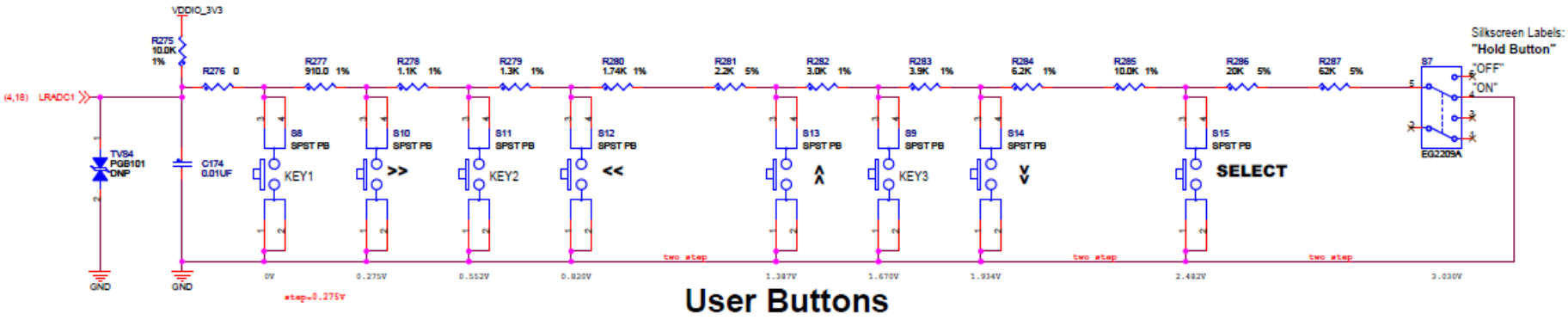
- The Low-Resolution ADC (LRADC) block has 16 physical channels for voltage measurement.
- 6 physical channels are available for general use. The other 10 physical channels are reserved for dedicated purposes.
- Only 8 “virtual” channels can be used at a time. Each of them can be mapped to any of the 16 physical channels.
- Resolution: 12 bits; Accuracy: 1.3%
  (Accuracy can be improved to better than 0.5% if the bandgap voltage reference is calibrated with fuses.)
- All channels sample on the same divided clock rate from the 24MHz crystal.
- 4 delay-control channels for timing and scheduling control events within the LRADC
- Integrated touch-screen controller with:
  - drive voltage generation for touch-screen coordinate measurement.
  - touch-detection interrupt circuit.



freescale semiconductor

ADC Application – Key Chain

- Use a resistor ladder to replace a key matrix.
- Any of the LRADC for general use can be used for the key input.
- The LRADC determine the key press by reading the voltage on input.
- It uses much less GPIO pins, which are very limited on the SoC, than key matrix.
- The circuit below is a typical key chain example (taken from i.MX28EVK).



User Buttons

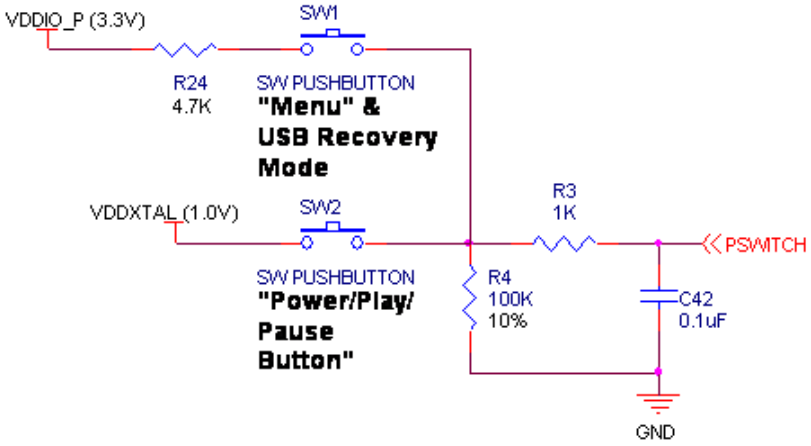
freescale semiconductor

i.MX28 Peripherals – Other I/O

- I^2C
  - EEPROM, Sensors
  - DMA controlled with M/S mode up to 400KHz
- 4-Channel 16-Bit Timers with Rotary Decoder
- Six-Channel Pulse Width Modulator (PWM)
- Real-Time Clock
  - Options for 24MHz, 32KHz or 32.768KHz
  - Storage of “persistent bits”
  - Wake from alarm
- UARTs
  - 5 x 3.25Mbps App UARTs
  - 1 x 115Kbps Debug UART
- S/PDIF Transmit

PSWITCH Operation

- The PSWITCH pin controls startup, reset, and has input pin functions.
- Startup DCDC when PSWITCH voltage >0.65V for 100ms
- Voltage levels
  - Low level : 0V- 0.30V
  - Mid level : 0.65V – 1.50V
  - High level : 2.10V – (VDDXTAL+1.575)V (recovery mode)
- Reset DCDC when PSWITCH pin voltage falling edge is <15ns.
  - Disable bit
- General purpose input
- USB Recovery mode (USB boot) when connected to VDDIO with 4.7k resistor (for 5 seconds).



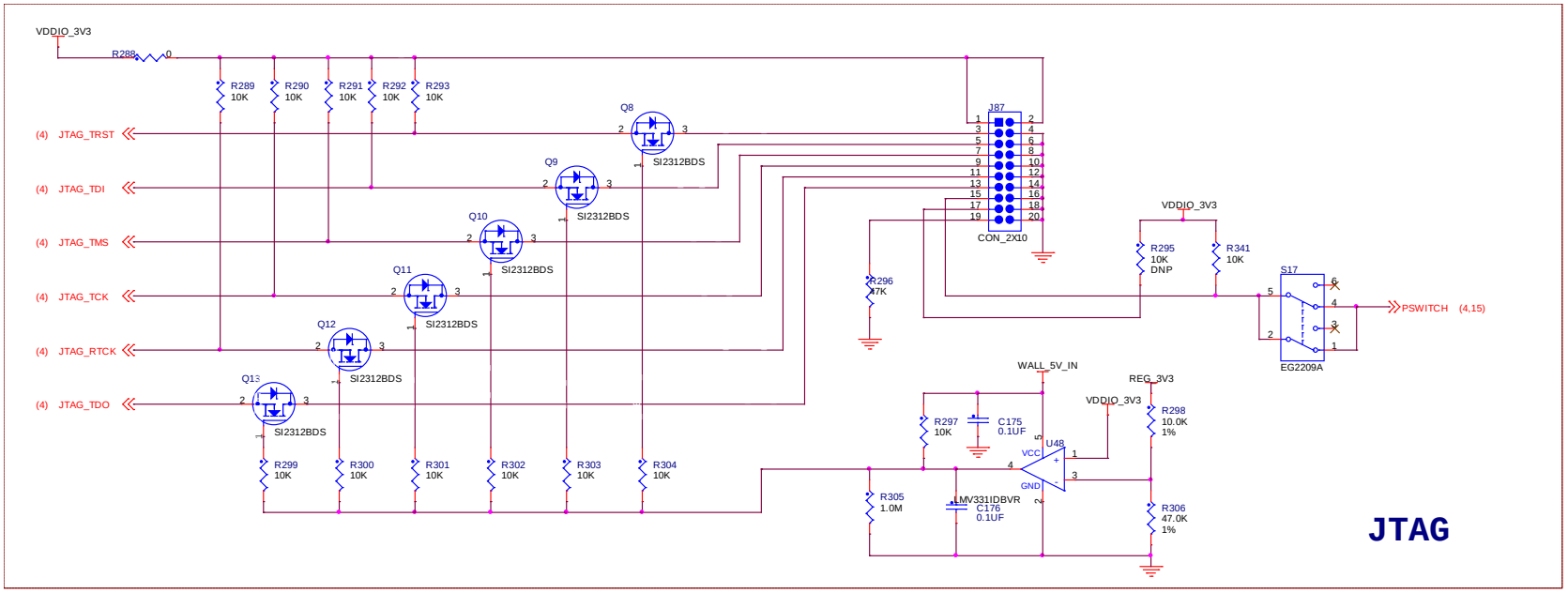
freescale semiconductor

RESET, Test mode, and Debug Pins

- RESET pin
  - Will reset and restart the processor when pulled low with 5V applied to VDD5V.
  - Will shutdown the processor and DCDC converter when powered solely from the Li-ion battery / DCDC converter.
  - Internal 10k pull-up resistor.
  - A capacitor between the RESET pin and ground should be added (for noise filtering) and the value must be 0.01uF or less.
- Test mode pin
  - Must always be connected directly to GND for normal operation.
- Debug pin
  - Must have a pull-up resistor (10k) to VDDIO_3.3V for normal (JTAG) operation.
  - Pulled low for boundary scan.

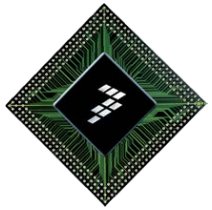
JTAG Hardware

- Standard parallel JTAG hardware is implemented on the MX28.
- To enter JTAG boot mode, LCD_D03 = 0, LCD_D2 = 1, LCD_D1 = 1, LCD_D0 = 0 during boot time (read by the ROM).
- On the EVK board, standard 20 pin JTAG connector is used.
- Additional circuitry added to prevent the back powering.



freescale semiconductor

IOMUX Tool

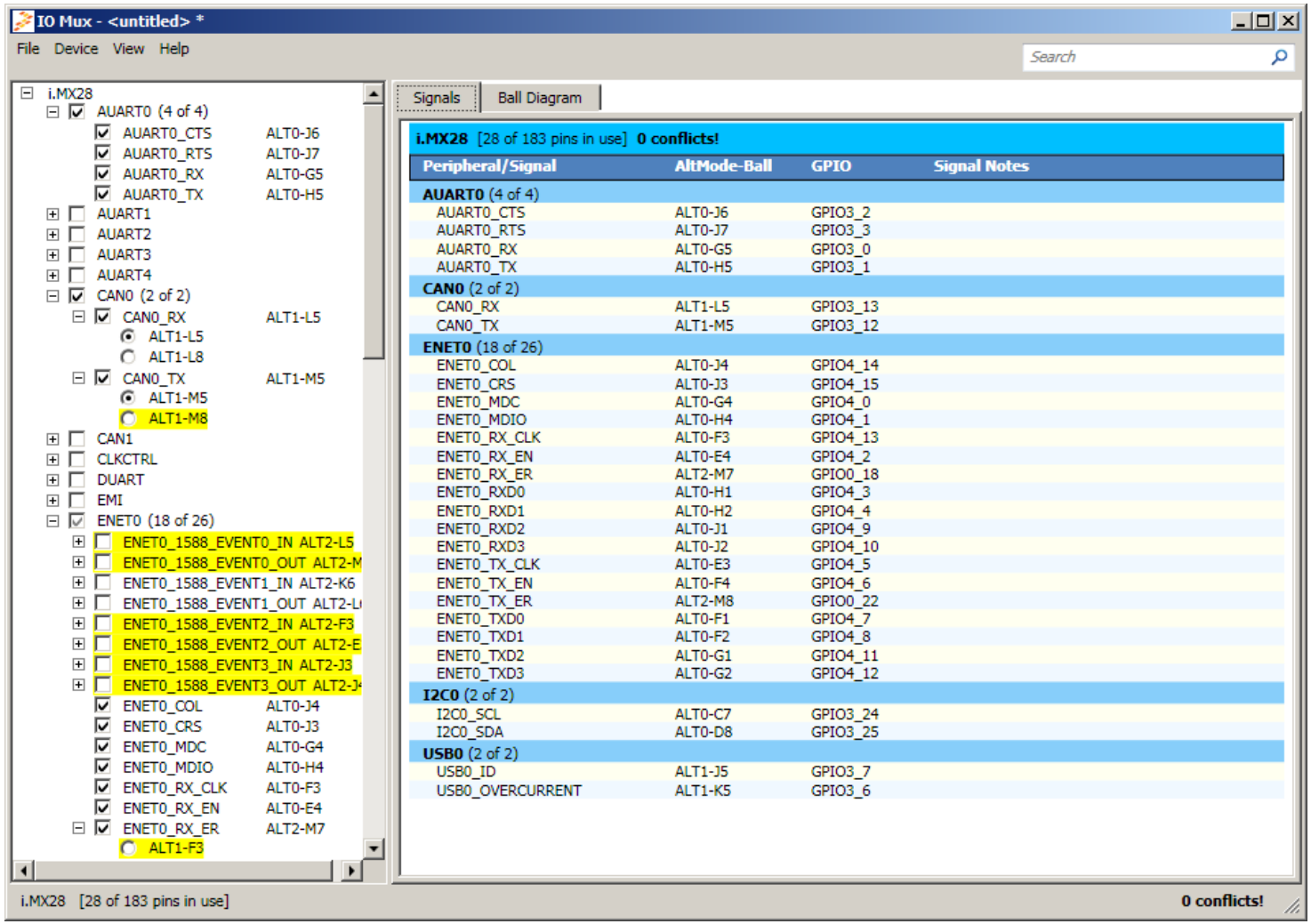


freescale™ semiconductor

Overview

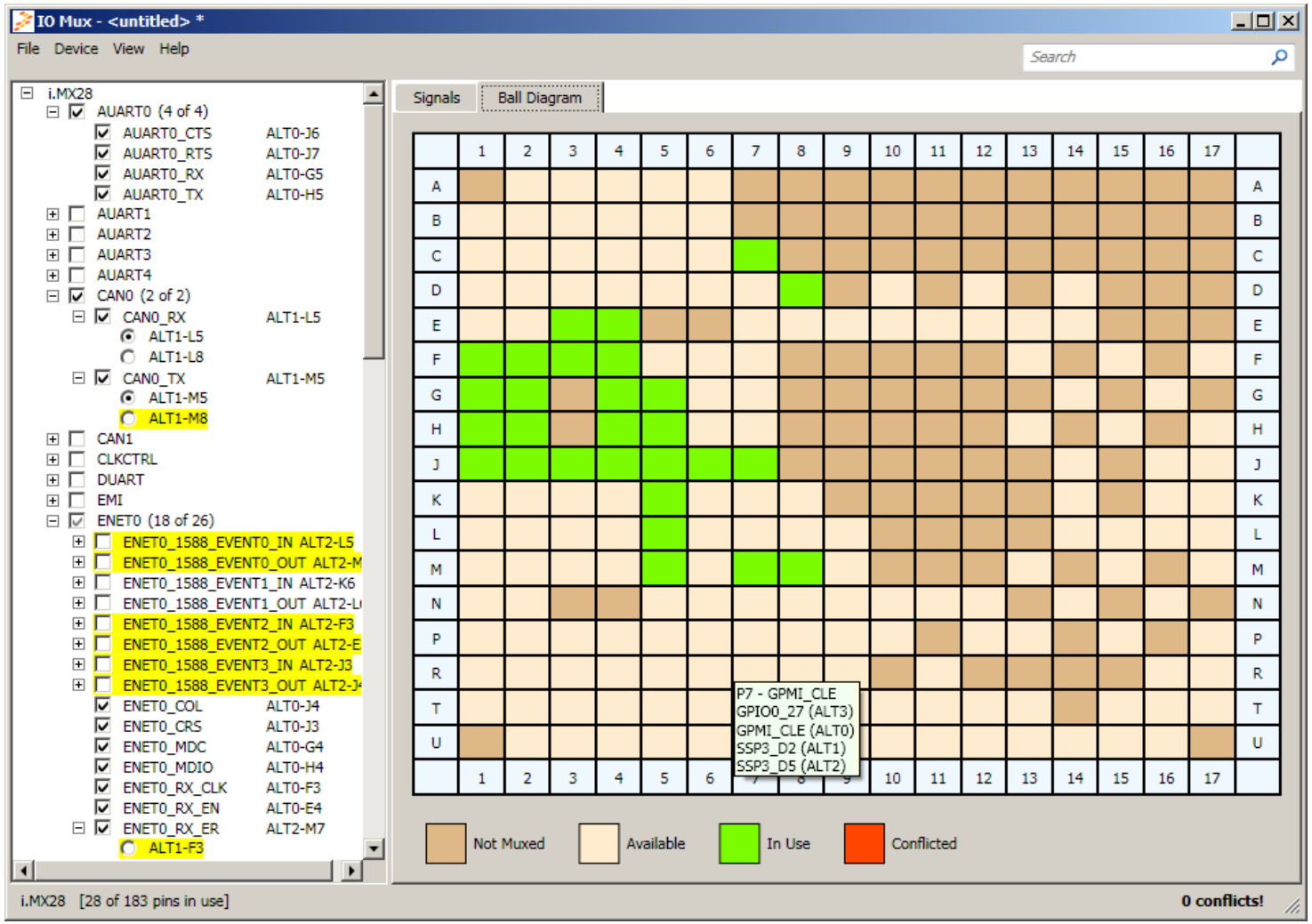
- IOMUX tool provides a simple way to
  - Verify that required interfaces can be used simultaneously
  - Find correct muxing configuration
  - Find and fix (if possible) pin sharing conflicts
  - View Ball diagram
- Available for a number of i.MX devices:
  - i.MX233
  - i.MX25
  - i.MX28
  - i.MX35
  - i.MX51
  - i.MX53

IOMUX Signal View

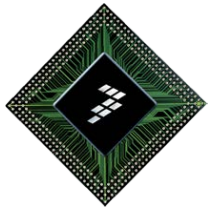


freescale semiconductor

IOMUX Ball Diagram View



freescale semiconductor



Manufacturing Tool

freescale™ semiconductor

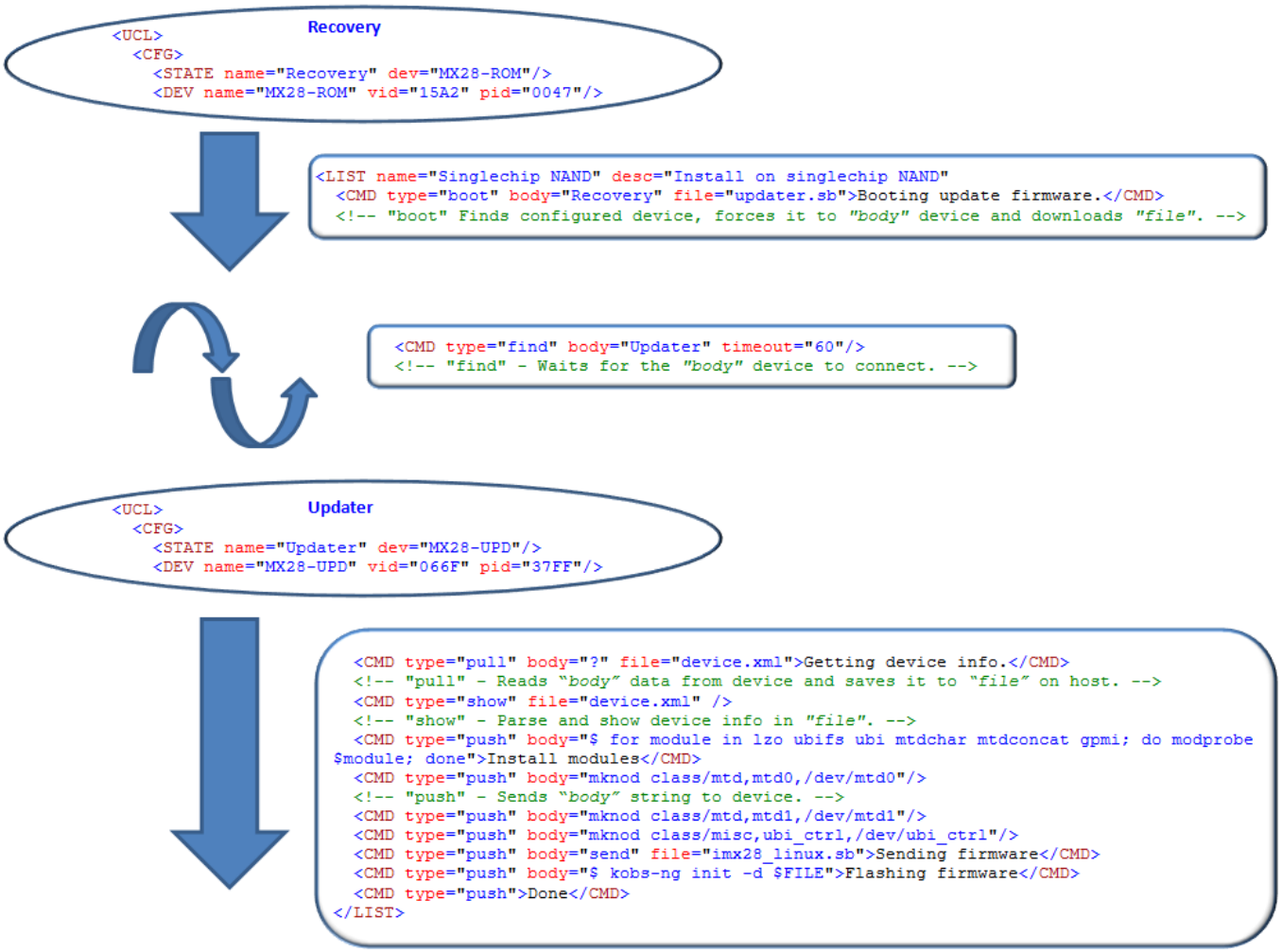
Basic Functions – Host Component

- The Mfg Tool host component is an operator friendly GUI interface for the firmware imaging process.
- The GUI associates a physical USB port to the firmware imaging operations and provides feedback to the operator.
- The Mfg Tool Framework is an architecture that supports:
  - Communication with various USB device drivers,
  - Loading firmware to ROM device enabling extended ROM functionality or complete application functionality.
  - Invoking commands supported by currently executing firmware.

Basic Functions – Firmware Component

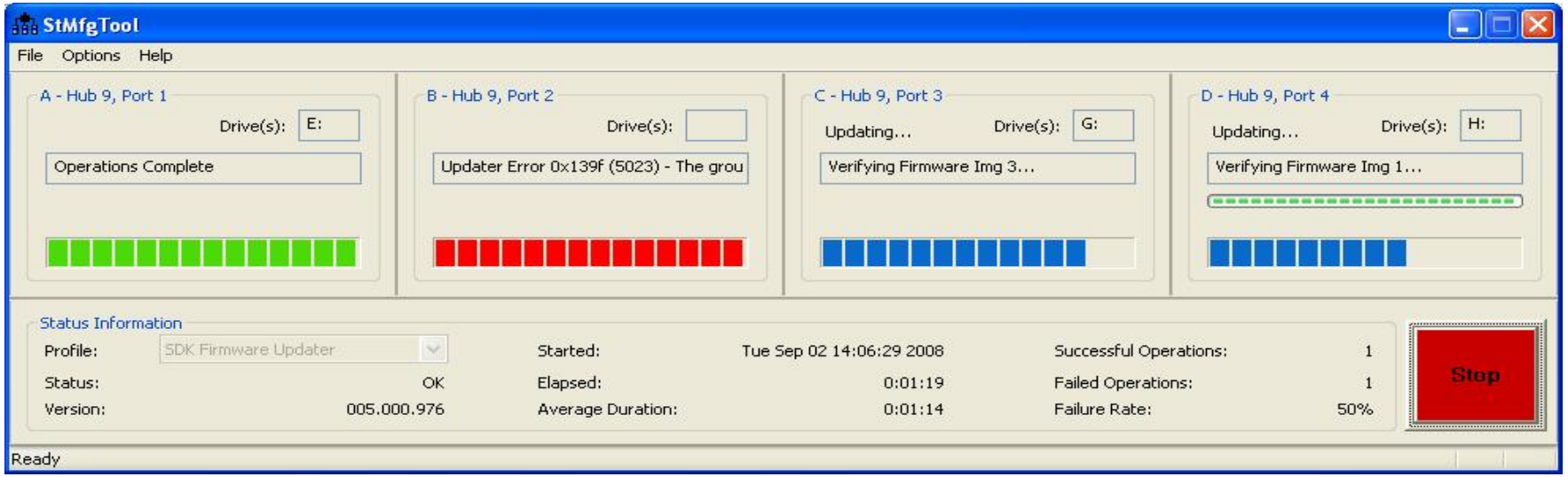
- The Mfg Tool firmware component enables these basic functions:
  - Erasing the media.
  - Allocating the media.
  - Writing firmware to the media allocation(s).
- Additional functionality is important for consumer devices:
  - Initializing the file system on the media.
  - Preloading content in data area of media

Universal Command Engine (UCE)



freescale semiconductor

GUI and Architecture

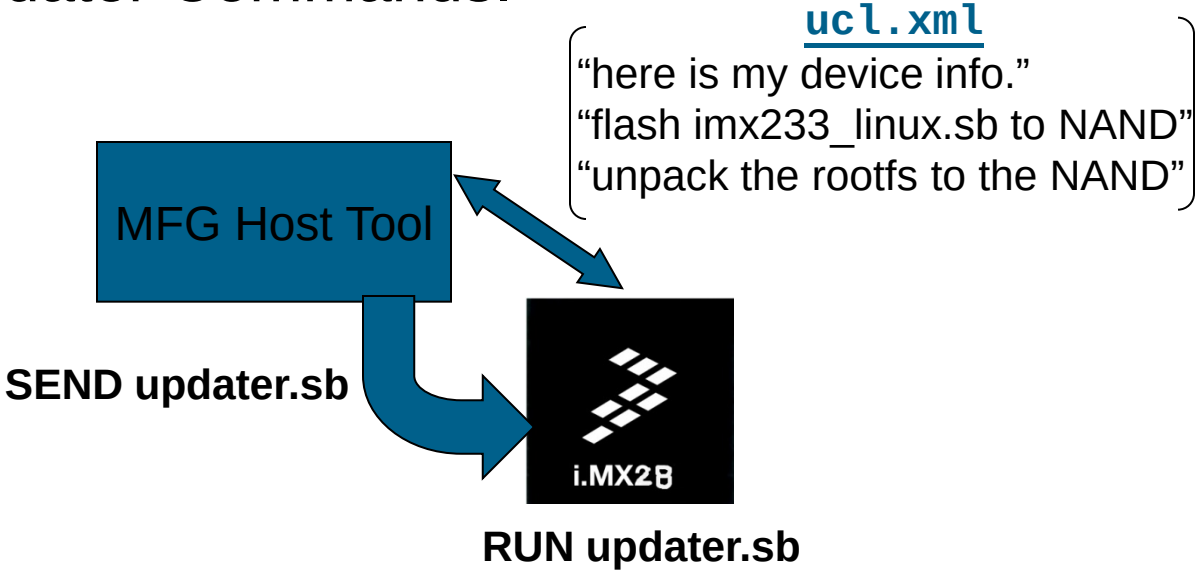


<table>
<tr><td colspan="4">Port Manager GUI</td></tr>
<tr><td colspan="4">USB Port</td></tr>
<tr><td colspan="4">Device</td></tr>
<tr><td>Recovery-Mode</td><td colspan="2">UTP Updater-Mode</td><td>MX Updater-Mode</td></tr>
<tr><td>BLTC API</td><td colspan="2">UTP API</td><td>SDP API</td></tr>
<tr><td rowspan="2">Boot Loader Transport Control (BLTC)</td><td colspan="2">Updater Transport Protocol (UTP)</td><td rowspan="2">Serial Download Protocol (SDP)</td></tr>
<tr><td>Volume (volsnap.sys)
Disk (disk.sys)</td><td>Bulk-Only Transport</td></tr>
<tr><td>USB HID (hidusb.sys)</td><td>USB MSC (usbstor.sys)</td><td colspan="2">Jungo/WinUSB/WDF Bulk I/O</td></tr>
<tr><td>ROM</td><td colspan="2">"updater.sb" (ThreadX, WinCE, Linux)</td><td>ROM</td></tr>
<tr><td>STMP 37xx/i.MX23/28</td><td colspan="2">STMP 37xx/i.MX23/i.MX51/25</td><td>i.MX51/25/35</td></tr>
</table>

freescale semiconductor

What is “Updater Firmware”?

- For i.MX28 & i.MX233, it’s an executable called **updater.sb**. This executable enables the device to “talk” to the host (via commands defined in ucl.xml).
- Manufacturing Tool sends updater.sb to device to be run. After booting updater.sb, the device can accept Host Updater Commands.



freescale semiconductor

Example of Building updater.sb

- Change ltib profile to “updater”.

  `./ltib –selectype` (and select “Mfg firmware” profile)

- Configure/Build ltib as you would for other profiles.

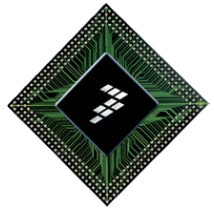
  `./ltib –c` (**note** – NEVER change the default kernel cmd line for the updater profile!)

  - Make whatever driver/package are necessary.
  - Look for new updater.sb at root of ltib.
  - Note – the 09.12 release only supports this for i.MX233. i.MX28 updater support will be in later release.

A Look at an Example Profile in ucl.xml

```
<LIST name="Singlechip NAND" desc="Install on singlechip NAND">
<CMD type="boot" body="Recovery" file="updater.sb">Booting update firmware.</CMD>
<CMD type="find" body="Updater" timeout="60"/>
<CMD type="pull" body="?" file="device.xml">Getting device info.</CMD>
<CMD type="show" file="device.xml" />
<CMD type="push" body="$ for module in lzo ubifs ubi mtdchar mtdconcat gpmi; do modprobe $module;
done"
>Install modules</CMD>
<CMD type="push" body="mknod class/mtd,mtd0,/dev/mtd0"/>
<CMD type="push" body="mknod class/mtd,mtd1,/dev/mtd1"/>
<CMD type="push" body="mknod class/misc,ubi_ctrl,/dev/ubi_ctrl"/>
<CMD type="push" body="send" file="imx23_linux.sb">Sending firmware</CMD>
<CMD type="push" body="$ kobs-ng init -d $FILE">Flashing firmware</CMD>
<CMD type="push" body="$ flash_eraseall /dev/mtd1">Erasing rootfs partition</CMD>
<CMD type="push" body="$ ubiattach /dev/ubi_ctrl -m 1 -d 0">Attaching UBI partition</CMD>
<CMD type="push" body="mknod class/ubi,ubi0,/dev/ubi0"/>
<CMD type="push" body="$ ubimkvol /dev/ubi0 -n 0 -N rootfs0 -s 48MiB">Creating UBI volumes</CMD>
<CMD type="push" body="$ ubimkvol /dev/ubi0 -n 1 -N rootfs1 -s 48MiB">Creating UBI volumes</CMD>
<CMD type="push" body="$ ubimkvol /dev/ubi0 -n 2 -N data -m">Creating UBI volumes</CMD>
<CMD type="push" body="$ mkdir -p /mnt/ubi0; mount -t ubifs ubi0_0 /mnt/ubi0" />
<CMD type="push" body="$ mkdir -p /mnt/ubi1; mount -t ubifs ubi0_1 /mnt/ubi1" />
<CMD type="push" body="send" file="files/rootfs.tar">Sending rootfs image</CMD>
<CMD type="push" body="$ tar -C /mnt/ubi0 -xvf $FILE">Unpacking to partition 0</CMD>
<CMD type="push" body="$ tar -C /mnt/ubi1 -xvf $FILE">Unpacking to partition 1</CMD>
<CMD type="push" body="$ umount /mnt/ubi0">Unmounting</CMD>
<CMD type="push" body="$ umount /mnt/ubi1">Unmounting</CMD>
<CMD type="push" body="\!3">Done</CMD>
</LIST>
```

i.MX28 Hands-on



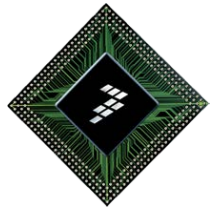
freescale™ semiconductor

Hands-on

- Use provided notes to execute hands-on steps
- Follow the instructors or proceed independently at your own pace

freescale semiconductor

A Word on LTIB



freescale™ semiconductor

What is LTIB ?

- LTIB = Freescale *GNU/**L**inux **T**arget **I**mage **B**uilder* **...**
  - … a tool to build Linux target images from source packages
  - … a mechanism to deliver Linux board support packages (BSP)
  - … a wrapper around tool chains and standard Linux commands like **`make`**, **`tar`**, **`gcc`**, **`objcopy`**, ...
- It provides…
  - functionality to configure and build Linux ***system components*** (kernel, bootloader, busybox, ……..)
  - functionality to configure and build the Linux ***target system*** (network configuration, type of file system to use, ……)

What is LTIB ?

- LTIB has been released under the terms of the GNU General Public License (GPL)
- “Standard Linux” look and feel :

  `make menuconfig, Kconfig, rpm`
- Pool of 200+ applications, originating from open source projects

What Is LTIB ?
Functionality

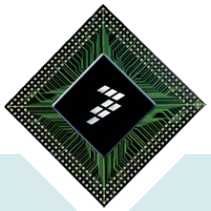
- A lightweight command line interface controls scripts and configuration menus to perform the following functions :
  - Build kernel, boot loader and application packages from source
  - Deploy built packages to a root file system (RFS) tree
  - Prepare appropriate kernel or RFS image files ready for network or flash based use on the embedded target board
  - Manage target image files using a private rpm database per LTIB instance on the host
  - Capture source modifications into patches and auto update .spec files
  - Interface directly to the network / Internet for package download and update from public CVS site
  - All package building is done as regular user (i.e. non-root)

LTIB
Philosophy (cont.)

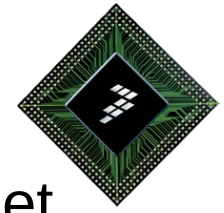
- Running LTIB on a Linux host, performs all target package configuration, build and installation tasks, that would normally take place on a self hosted Linux target platform
- Conceptually running LTIB means updating the RFS tree according to the desired configuration, including the boot loader and kernel, relying on a private per-project host based RPM management for the specific target platform
- LTIB manages changes to a package by transparently working with released or user generated .patch files

Questions & Answers

BACKUP SLIDES



freescale™ semiconductor

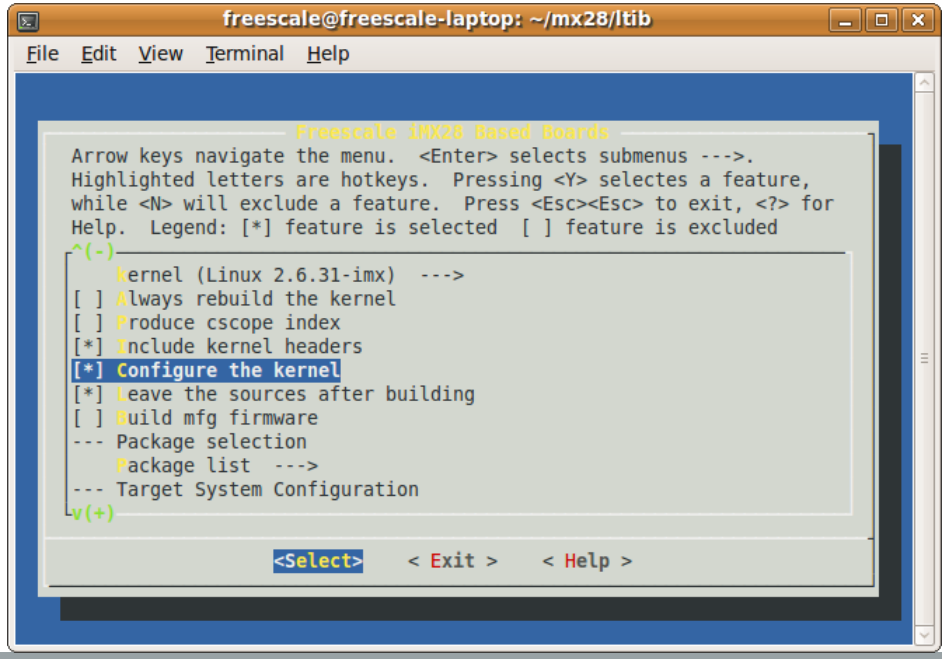


Optional Hands-on Session, L2 Switch and Dual Ethernet

freescale™ semiconductor

Changing Kernel Configuration

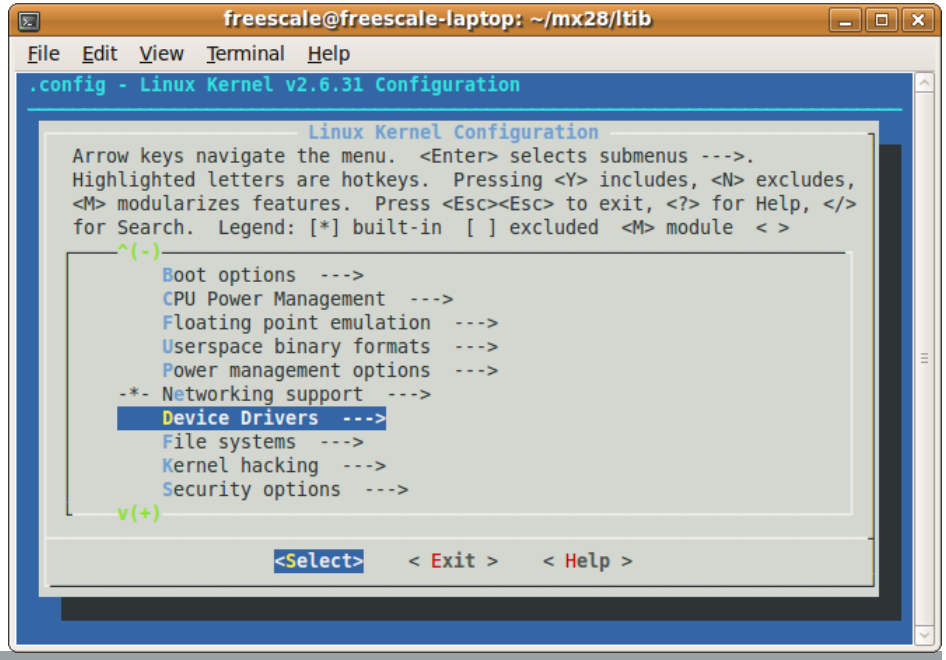
- `./ltib --configure`
- Enable the “Configure the kernel” option
  - This will trigger the menuconfig-based Linux kernel configuration later when LTIB starts to build the Linux kernel
- **Exit** and
- **Save** the configuration



freescale semiconductor

Configuring Kernel

- When LTIB starts to building the Linux kernel it will launch the menu-based configuration process
- Go to “Device Drivers”



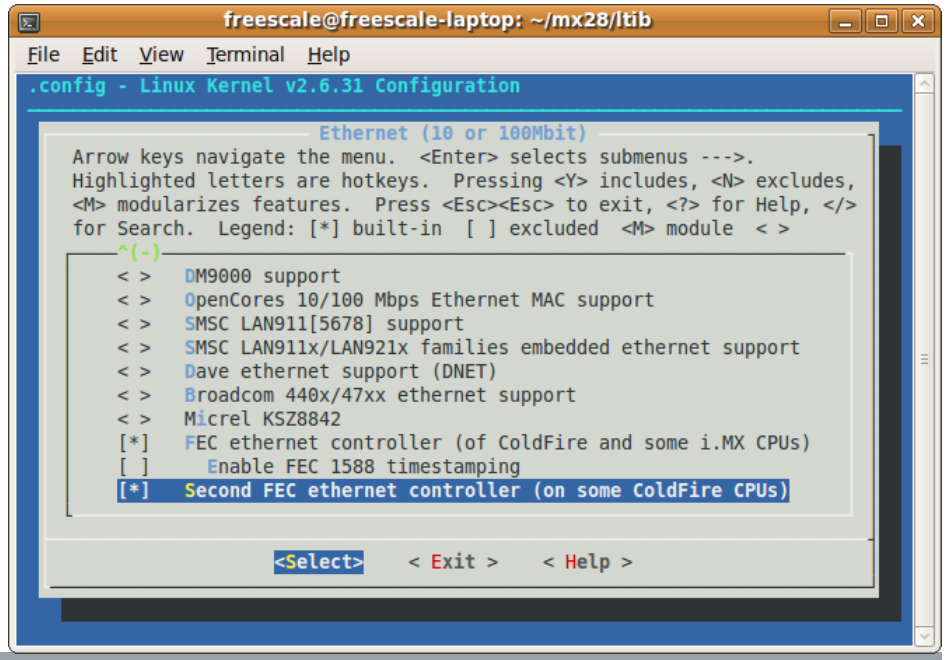
freescale semiconductor

Configuring Kernel

- Then go to “Network Device Support” and “Ethernet (10 or 100Mbit)”
- Depending on what you would like to do either:
  - Enable “Second FEC Ethernet controller (on some ColdFire CPUs)”

    or

  - Disable “FEC ethernet contr…” and
  - Enable “L2 Switch Ethernet Controller (of Coldfire CPUs)”

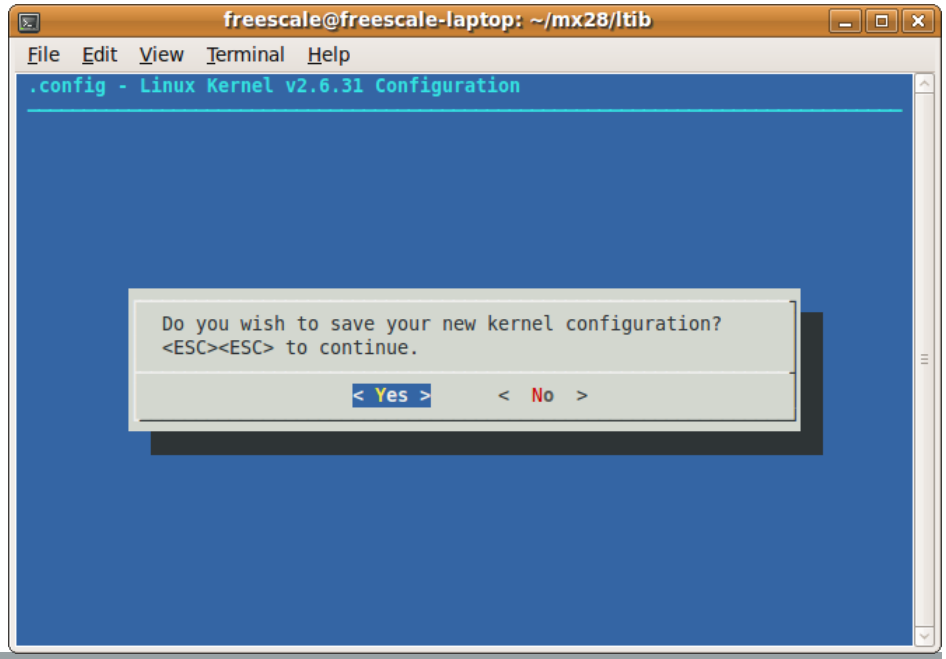


freescale semiconductor

Configuring Kernel

- Exit all the way back to main configuration screen
- Exit the main configuration screen
- Save your new kernel configuration when asked to do so

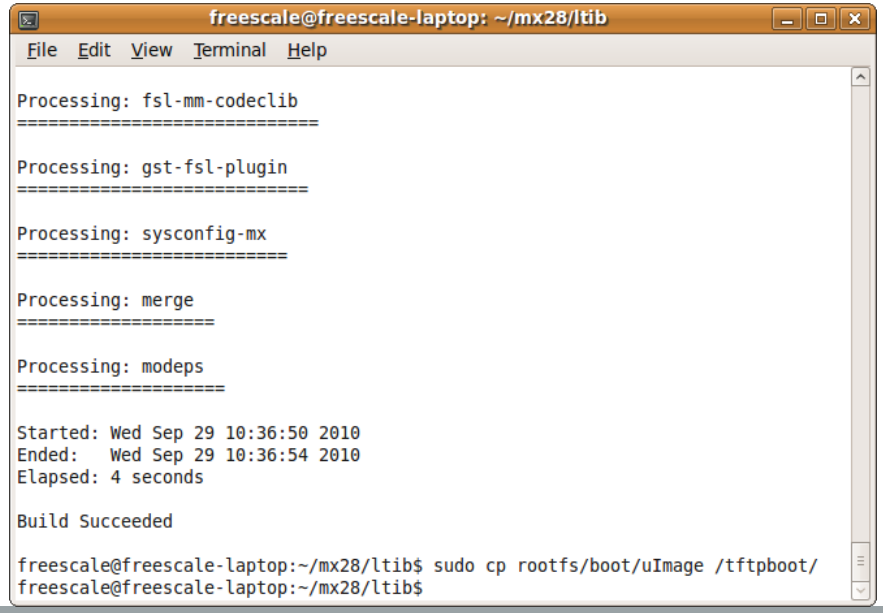
- Kernel build process will now continue



freescale semiconductor

Final Steps

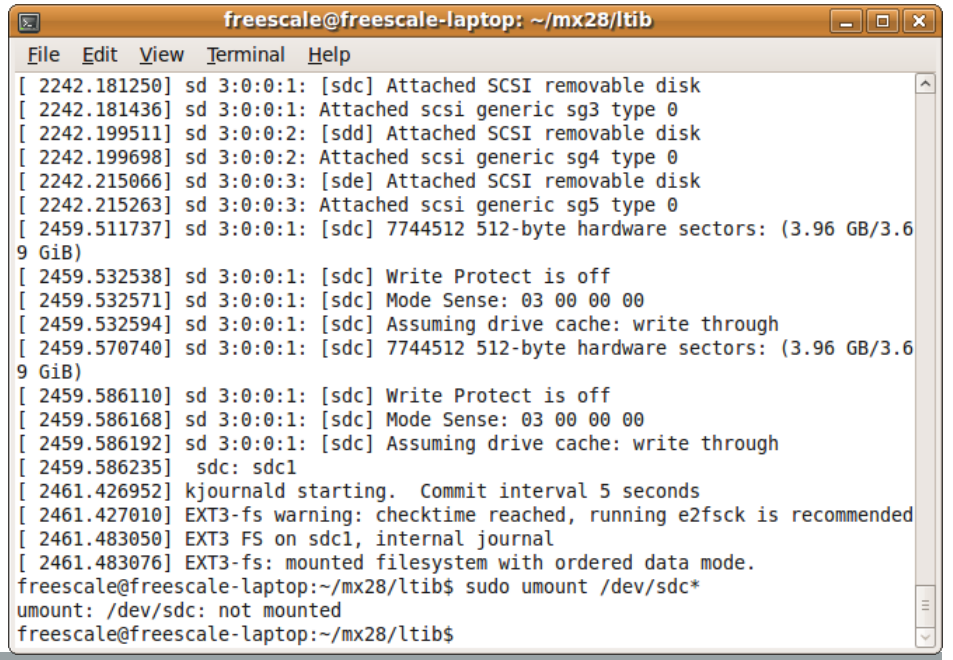
- Once kernel is build, LTIB will proceed to build all other newly selected packages
  - Build should succeed without problems.
- Kernel build **does not** trigger boot stream rebuild!
- We need to rebuild boot_stream package so it takes new kernel bin
  - `./ltib –p boot_stream.spec –f`
  - This needs to be done also when boot_stream command line is changed in LTIB
- Next we will program the binary to SD card



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Insert SD card to SD card reader
- Type `dmesg` and look for SD card device name
  - In example picture shown here it's **sdc**, so full device path is /dev/sdc
  - May change on other computers, will use **sdX** in examples here - adjust example according to your PC
- In order to create new partition structure, unmount all partitions
  - `sudo umount /dev/sdX*`



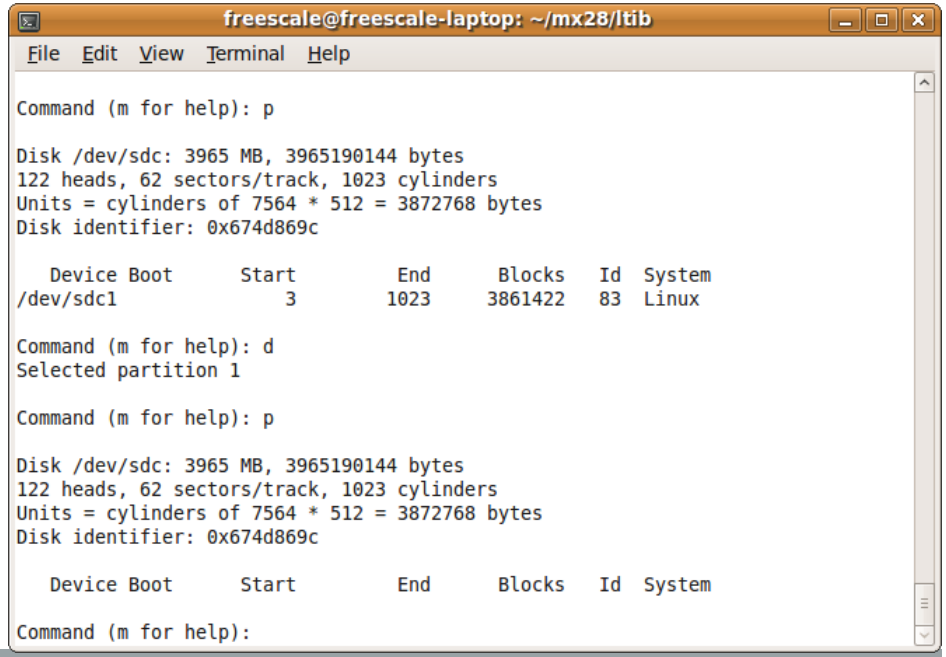
freescale semiconductor

Installing Linux BSP for i.MX Processors

- Let's NOT create the partitions on SD card
  - Card is already formatted appropriately
  - We just need to update bootstream partition (direct binary write) and (possibly) file system (copy will do)
  - Following slides marked as Example are for example only
- Default kernel is set up for following SD card structure:
  - File Allocation Table (FAT)
    - needed to prevent Windows from formatting the card
    - Not needed otherwise
  - Boot stream partition of type 0x53 (OnTrack DM6 Aux3)
    - we'll only use this partition in the training for programming the bootloader
    - In real-life linux kernel bootstream would go here
  - Linux rootfs such as ext2.
    - We won't use this partition in the training
    - In real-life, file system and all apps would go here

Installing Linux BSP for i.MX Processors, EXAMPLE

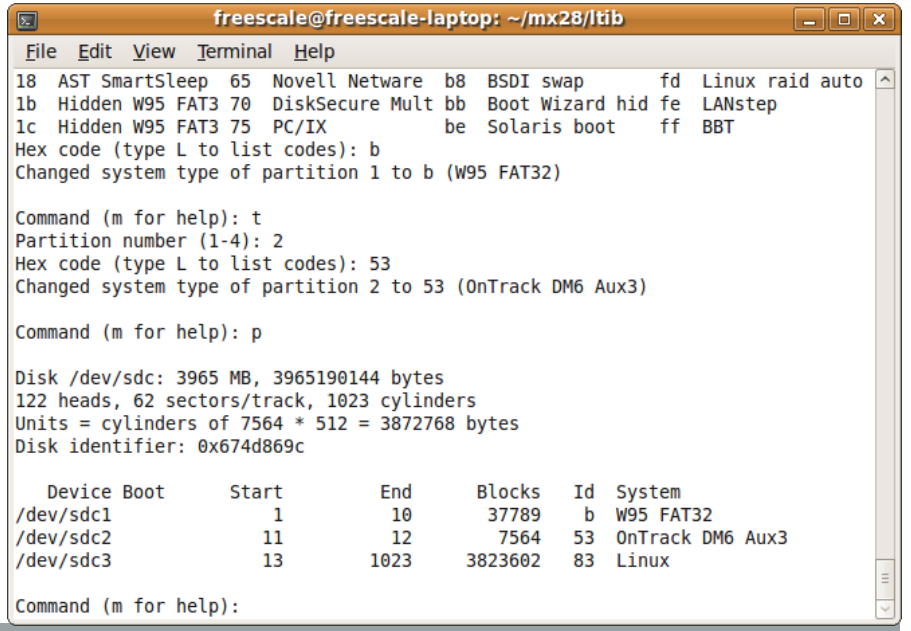
- Start fdisk on the SD card device
  - `sudo fdisk /dev/sdX`
- Type ‘p’ + <enter> to see all partitions and their numbers
- Delete all partitions on SD card
  - Type ‘d’ + <enter>
  - Enter partition number
  - Repeat above steps until there are no partitions left



freescale semiconductor

Installing Linux BSP for i.MX Processors, EXAMPLE

- Now let's create new partition structure with following sequence
  - `n <ent> p <ent> 1 <ent> <ent> +10M <ent>`
    - New primary part. 1 ending on 10th cylinder
  - `n <ent> p <ent> 2 <ent> <ent> +4M <ent>`
    - New prim. part. 2 ending on 12th cylinder
  - `n <ent> p <ent> 3 <ent> <ent> <ent>`
    - New prim. Part. 3 starting after 2nd part. and occupying rest of card
  - `t <ent> 1 <ent> b <ent>`
    - Change part. 1 type to FAT32
  - `t <ent> 2 <ent> 53 <ent>`
    - Change part. 2 type to 0x53
  - `w <ent>`
    - Write partition table and exit



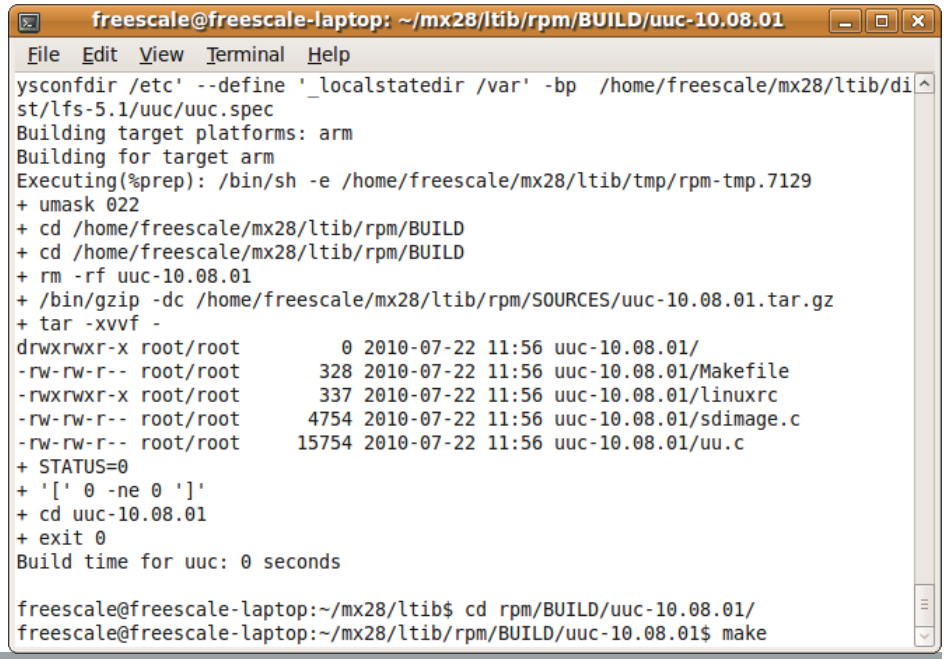
freescale semiconductor

Installing Linux BSP for i.MX Processors

- We can't write bootstream directly to /dev/sdX2 (second partition where bootstream will be located)
  - small table with some extra information is required to be pre-pended to bootstream
  - Used by internal ROM to locate proper bootstream
  - Allows for several backup copies of bootstream for unreliable media (NAND)
- Need to use program called sdimage to do that
- This program is part of uuc package in LTIB. Let's build it!
- First, unpack the uuc package
  - `./ltib –m prep –p uuc`

Installing Linux BSP for i.MX Processors

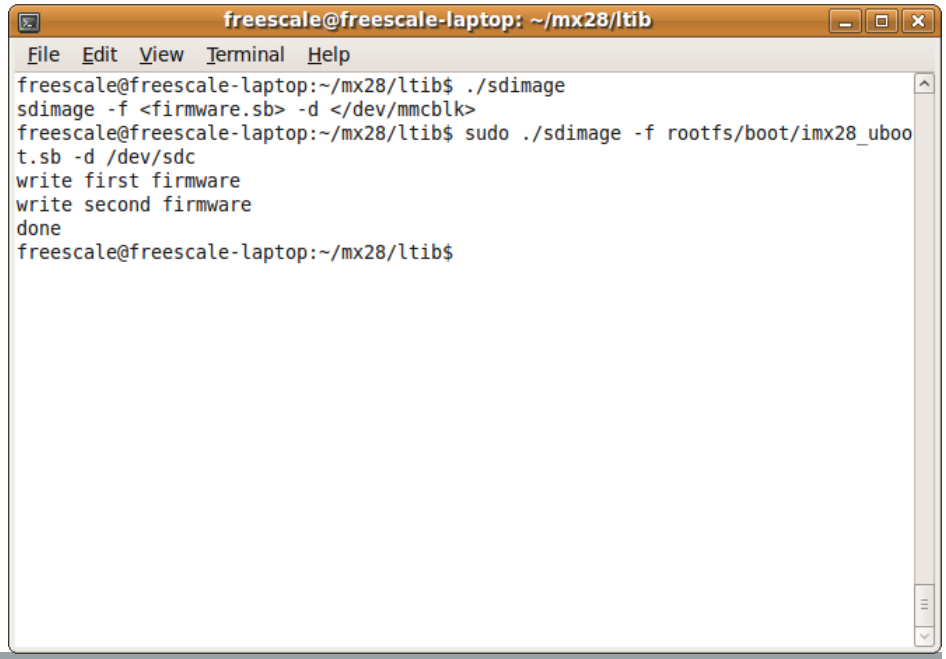
- Build the uuc package
  - `cd rpm/BUILD/uuc-10.08.01/`
  - `make`
- This will build the sdimage binary in current folder
- Note that we have built the uuc **natively**! We will run it on the host PC, not on i.MX28
  - We could have cross-compiled it for execution on i.MX28 with:
    `./ltib –m scbuild –p uuc`
    `./ltib –m scdeploy –p uuc`
- Copy the binary to LTIB install folder
  - `cp sdimage ../../..`
  - `cd ../../..`



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Now let's program the bootstream to SD card
  - `sudo ./sdimage –f rootfs/boot/imx28_ivt_linux.sb –d /dev/sdX`
- Sdimage will check the device partition table and program bootstream
- Note: there are two types of bootstream images generated:
  - for devices with HAB enabled
  - for devices with HAB disabed



freescale semiconductor

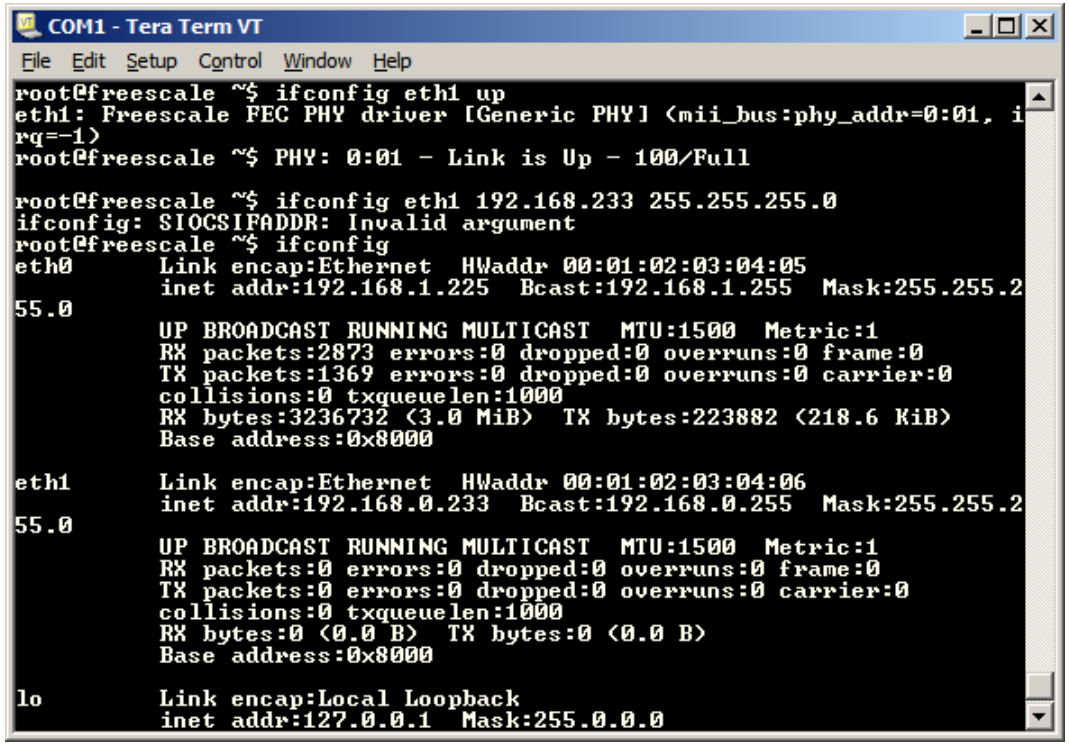
Updating File System

- In case you have added some applications/data to root file system you can update the SD card by
  - sudo mount /dev/sdX3 /mnt/tmp
  - cp –Rup rootfs/* /mnt/tmp
  - sudo umount /mnt/tmp

- That's it!
  - Take SD card out of reader
  - Insert to SD card socket 0

Second Ethernet Port

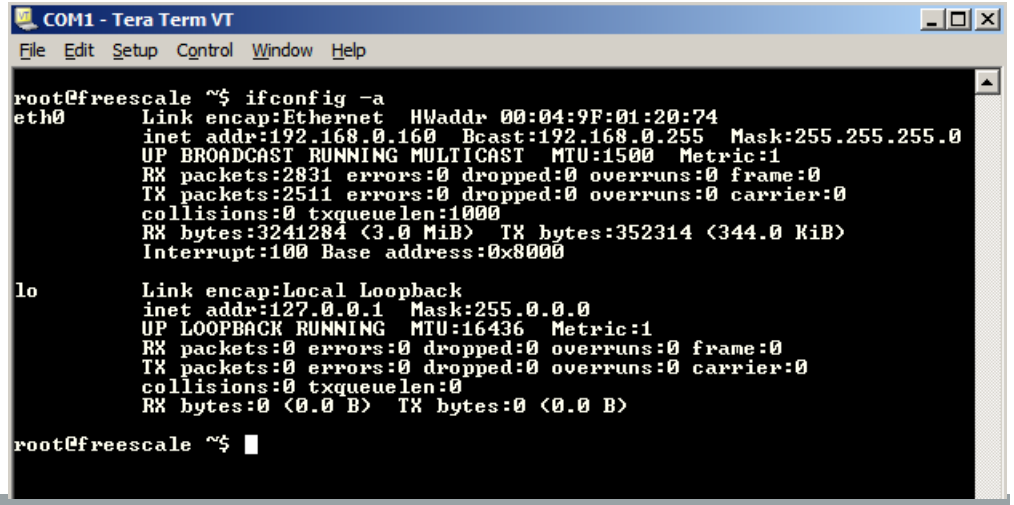
- Connect Ethernet cable to ENET1: UPPER JACK on i.MX28EVK
- `ifconfig eth1 up`
- `ifconfig eth1 IP_ADDR NETMASK`
  - IP_ADDR should belong to different subnet than current IP
  - Possible to use dhclient to obtain IP address automatically
  - Requires editing /etc/dhclient.conf not to run dhclient on eth0 but on eth1 interface
- Other network interface is now up & running



freescale semiconductor

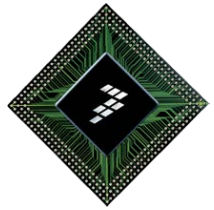
Enabling i.MX28 L2 Switch

- With L2 switch enabled there is only one network interface visible
- You can verify switch functionality by connecting a device via Ethernet cable to ENET1 port
- Device is able to communicate to Ethernet network on ENET0 port



freescale semiconductor

Host Machine Setup



freescale™ semiconductor

Linux Development Host

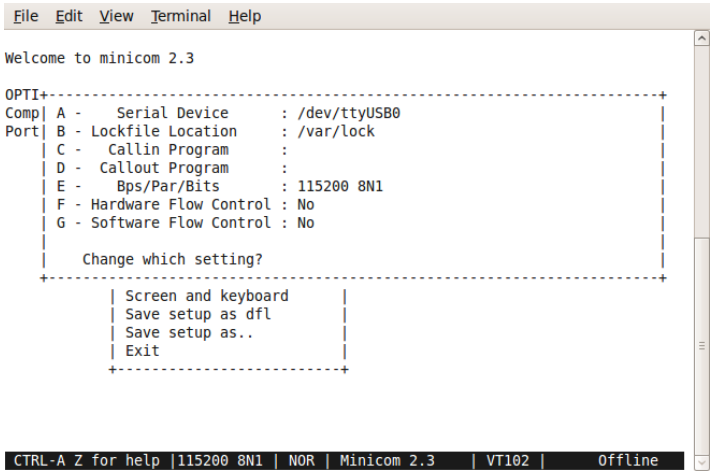
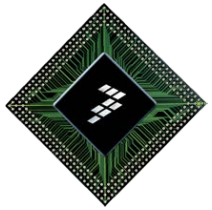
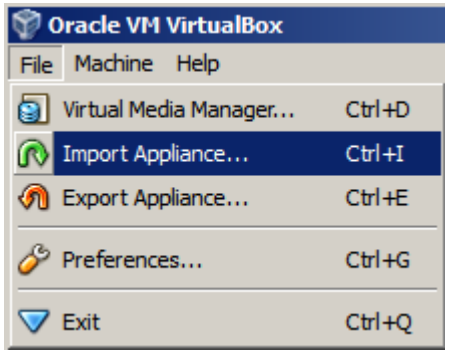
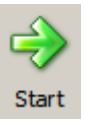
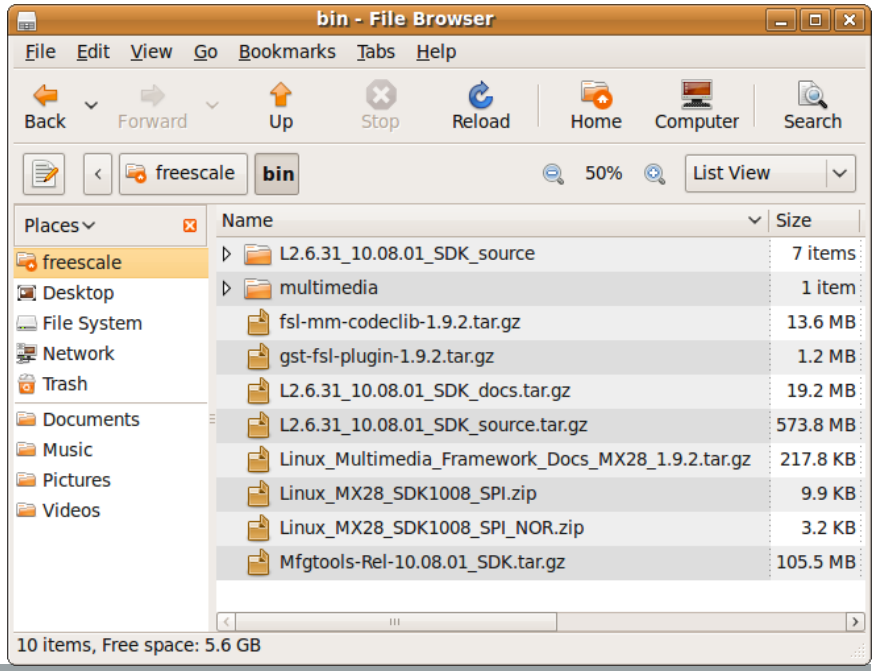
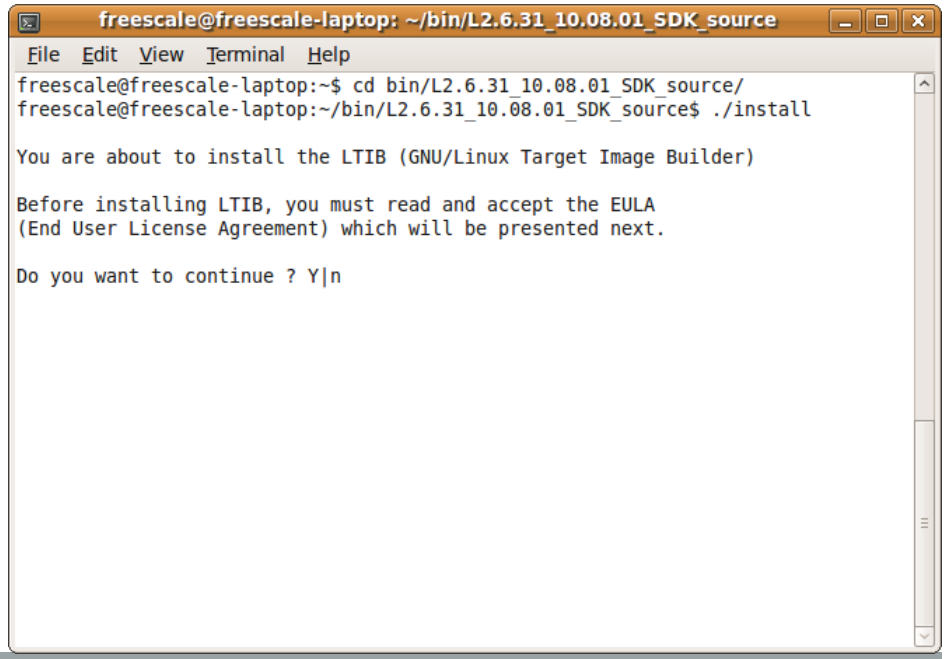
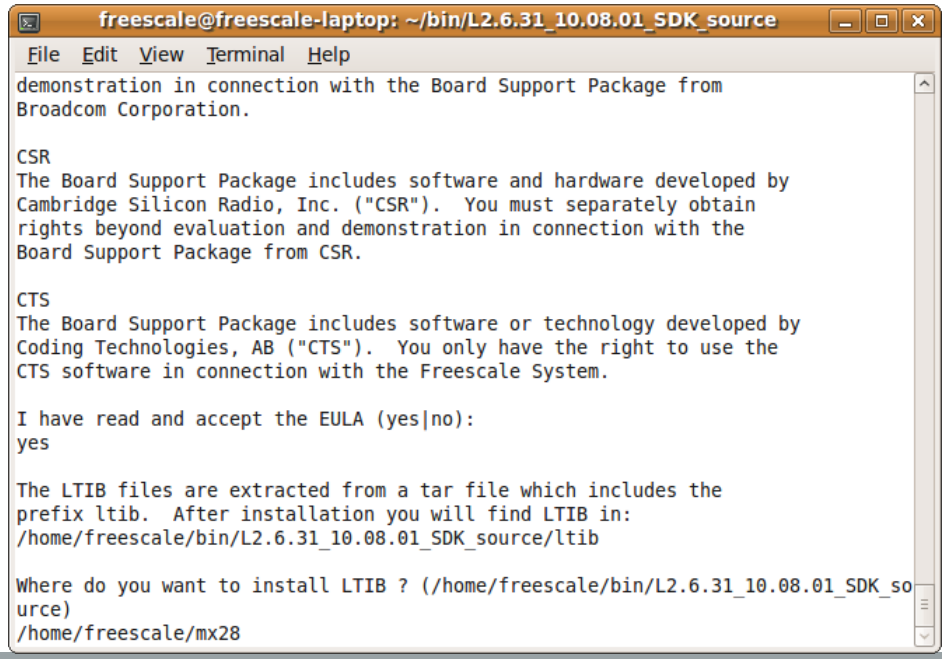
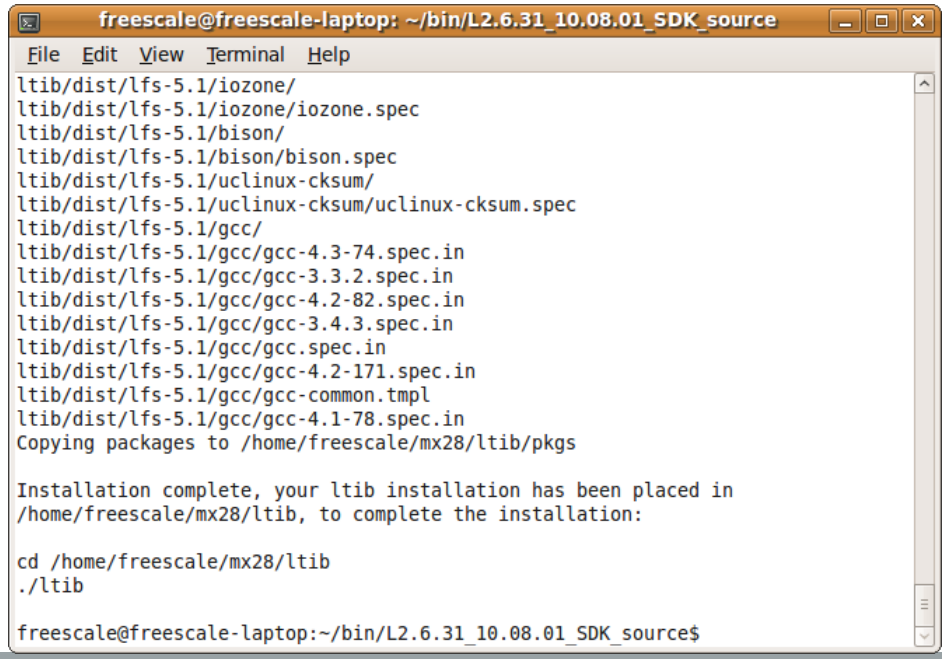
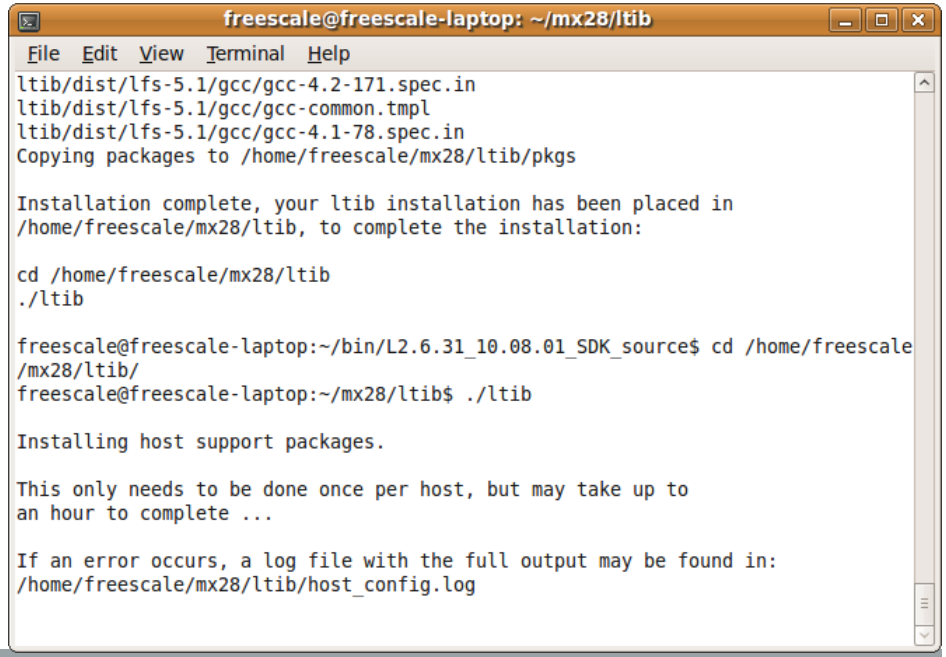
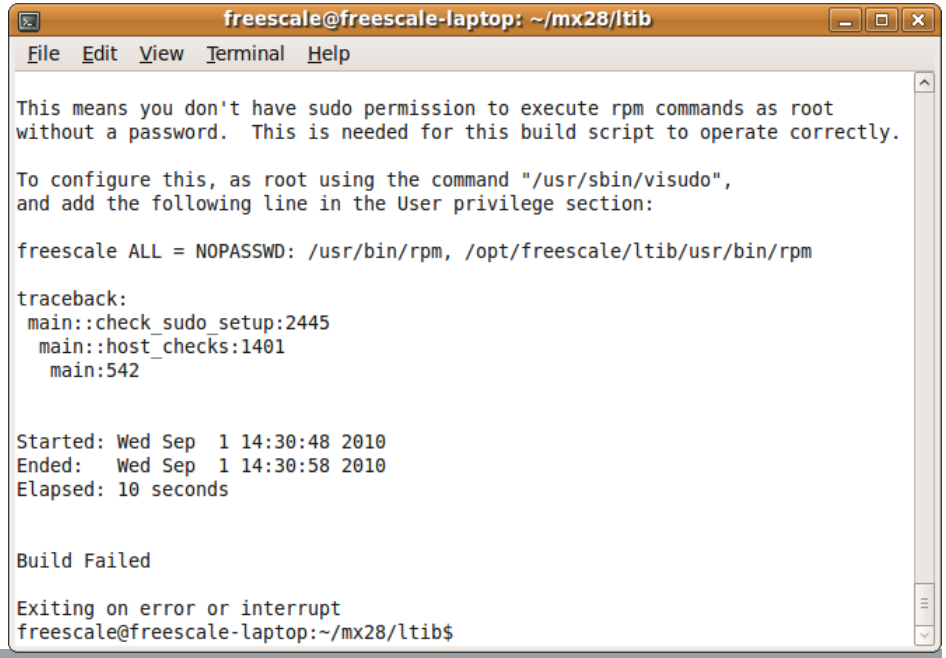
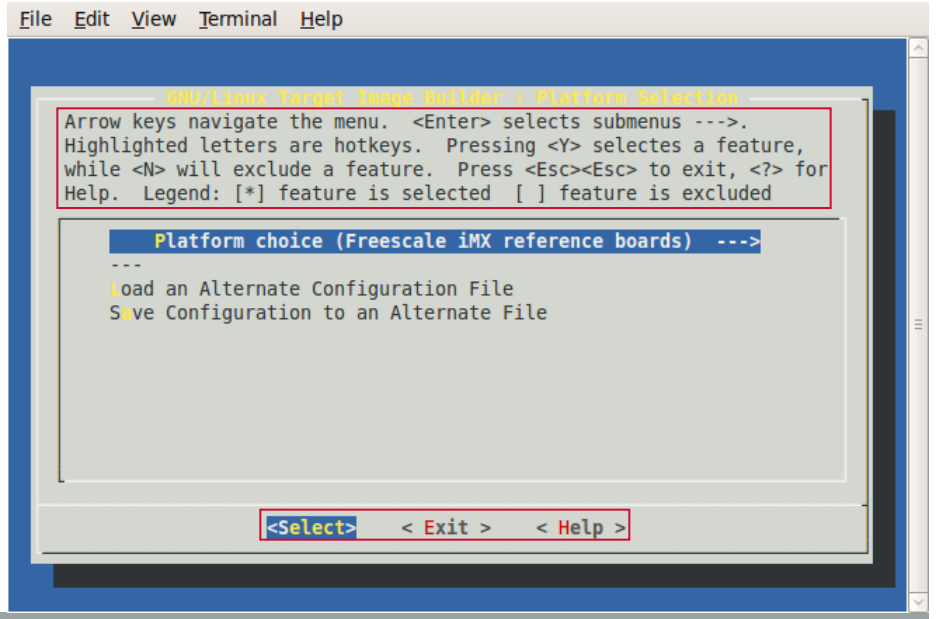
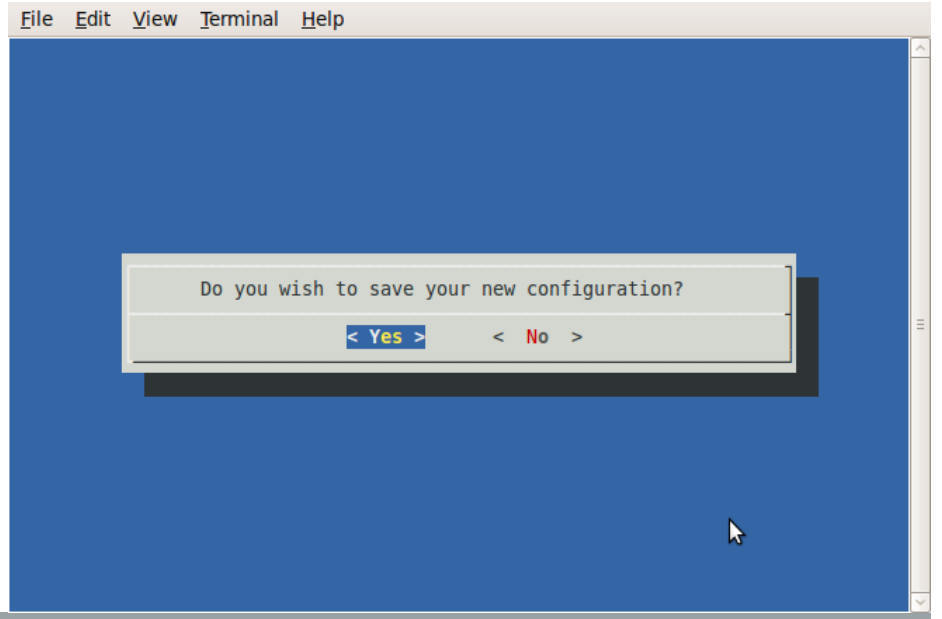
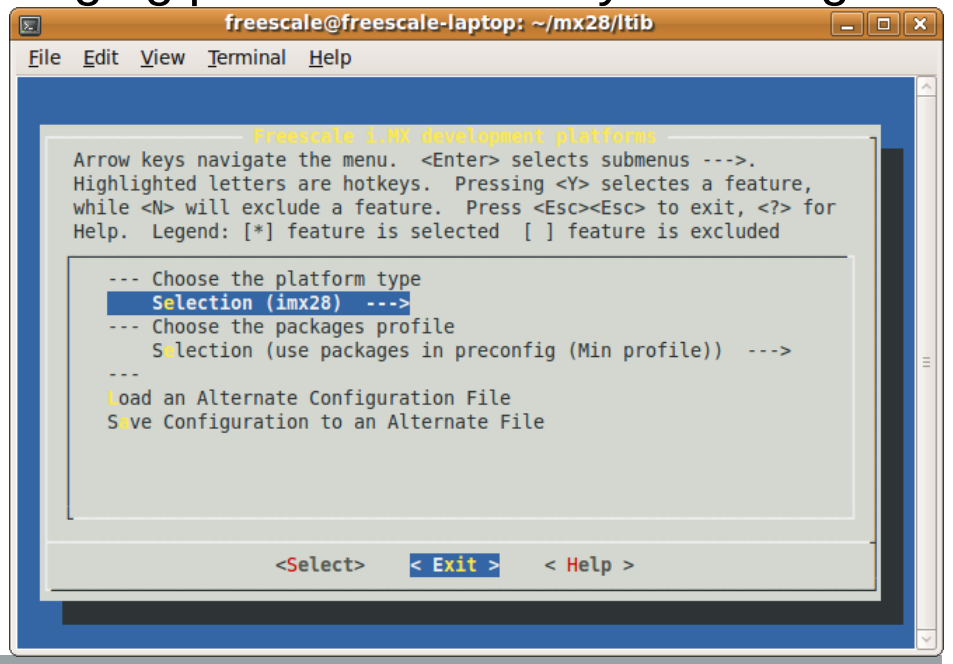
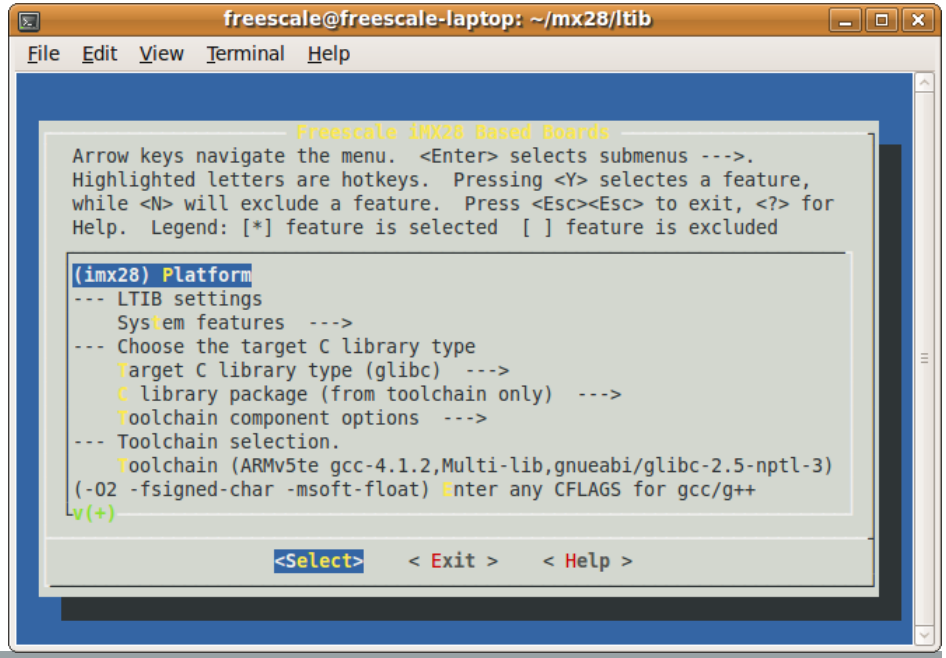
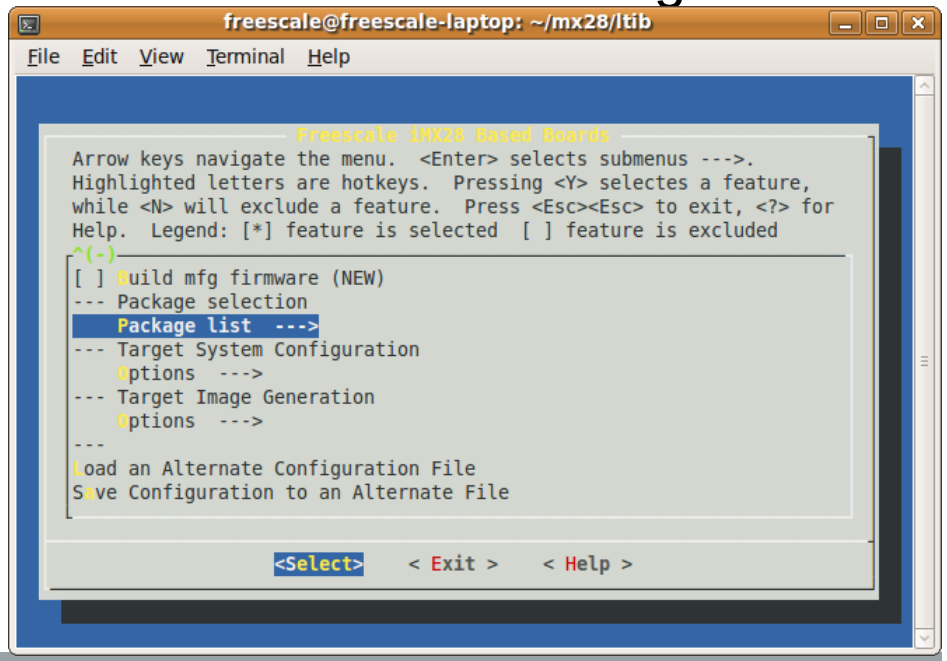
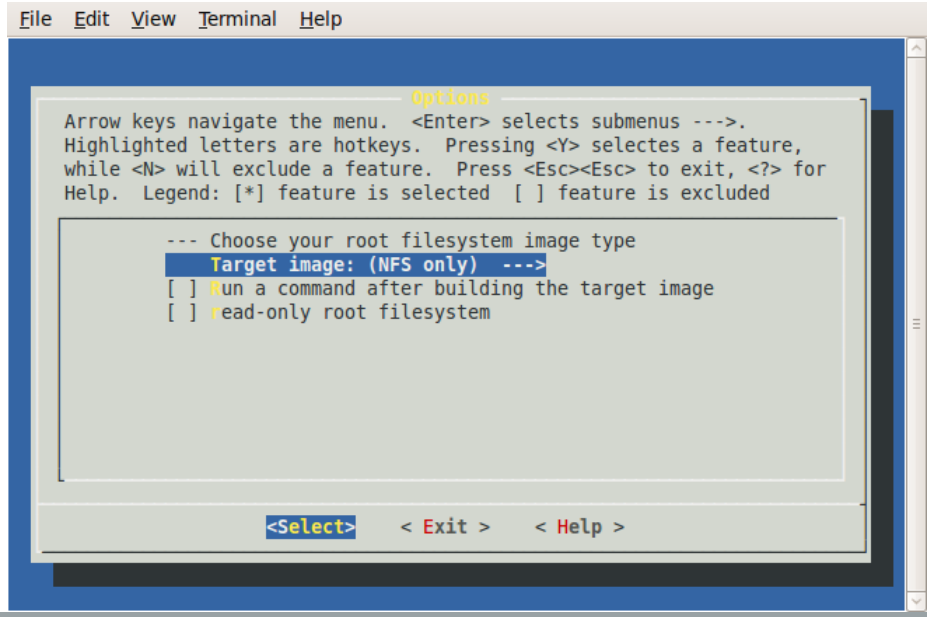
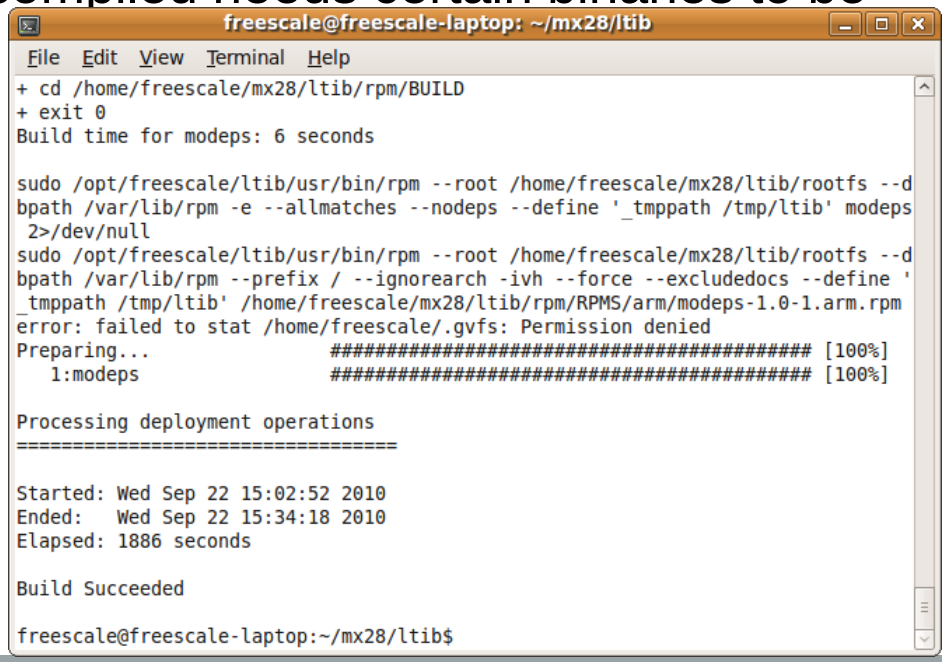
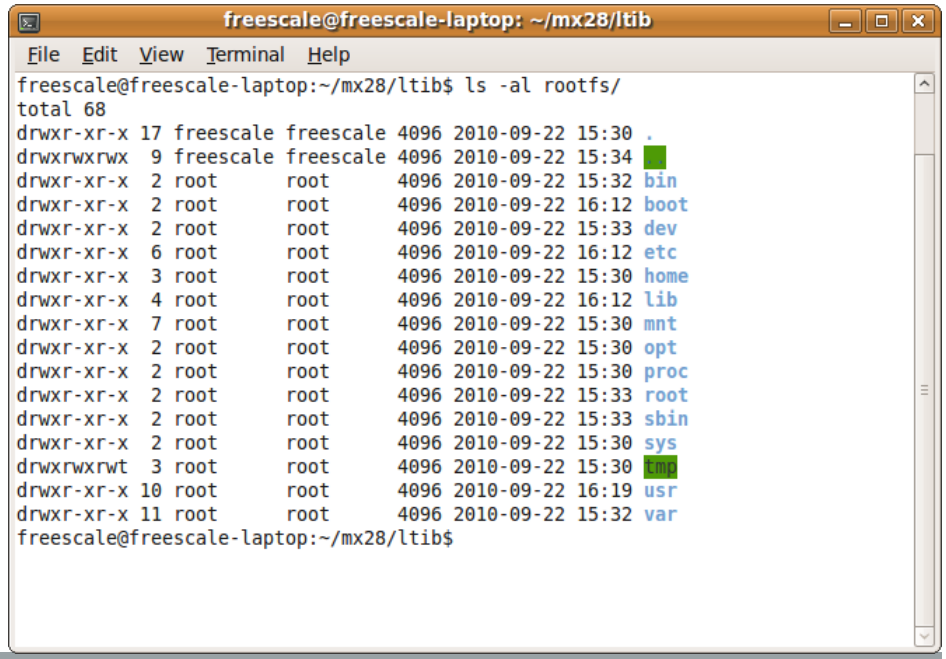
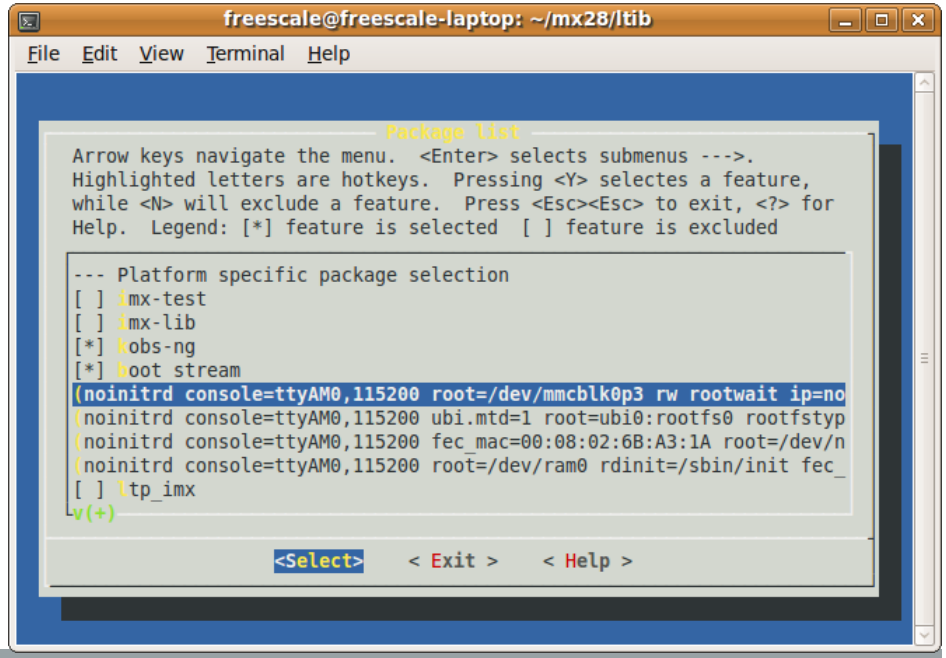
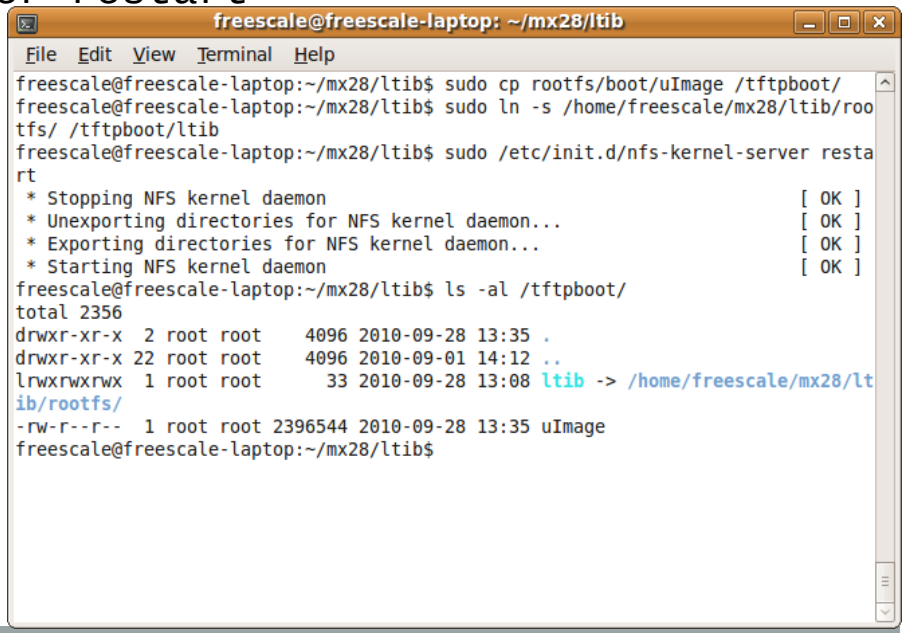
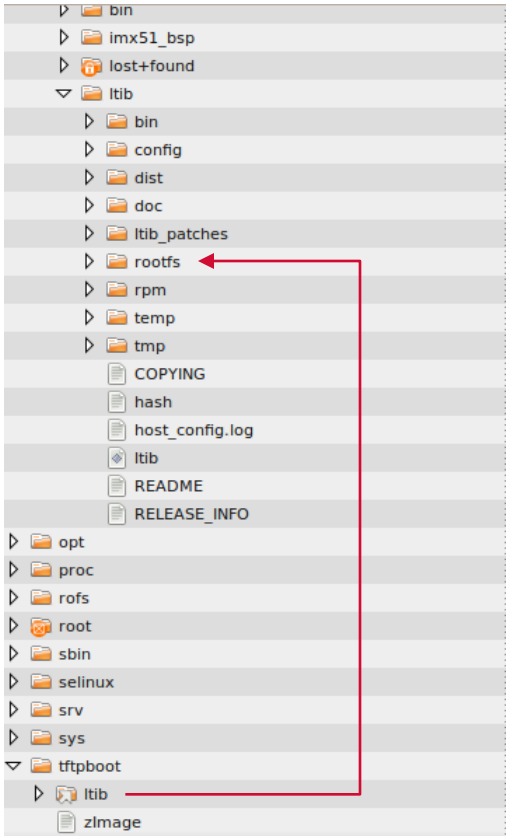
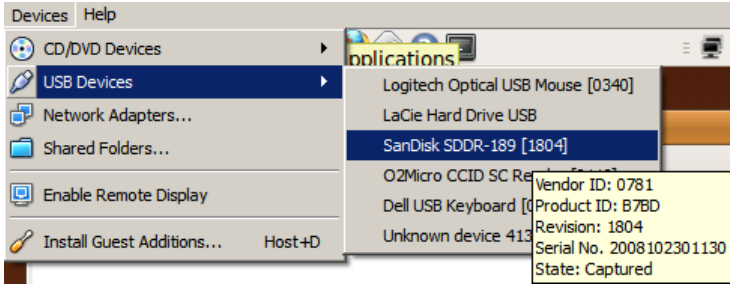
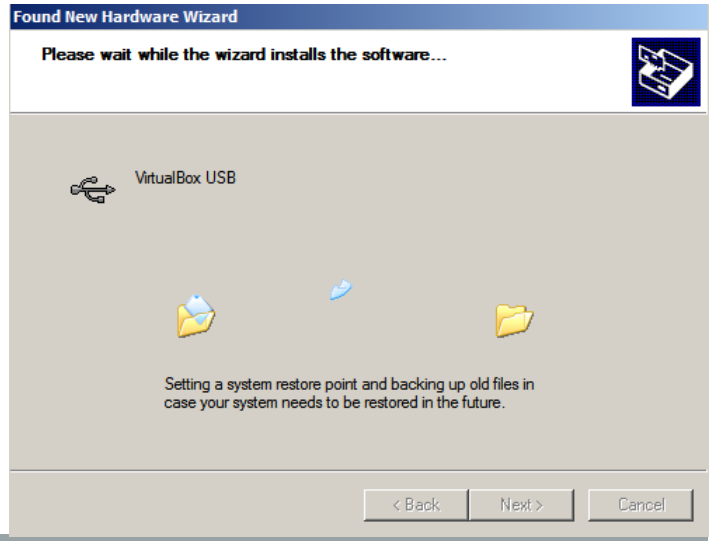
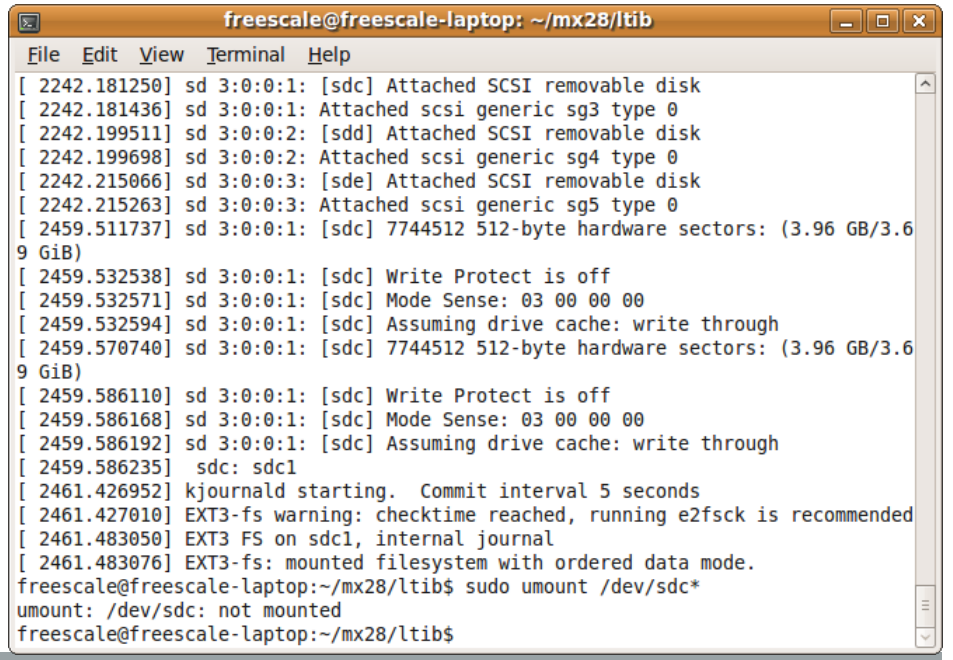
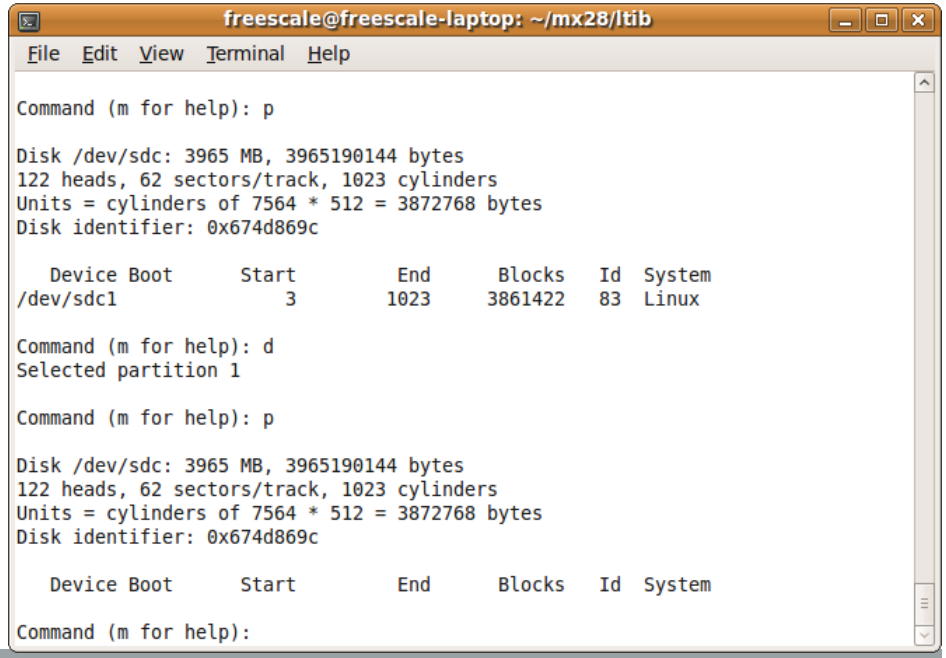
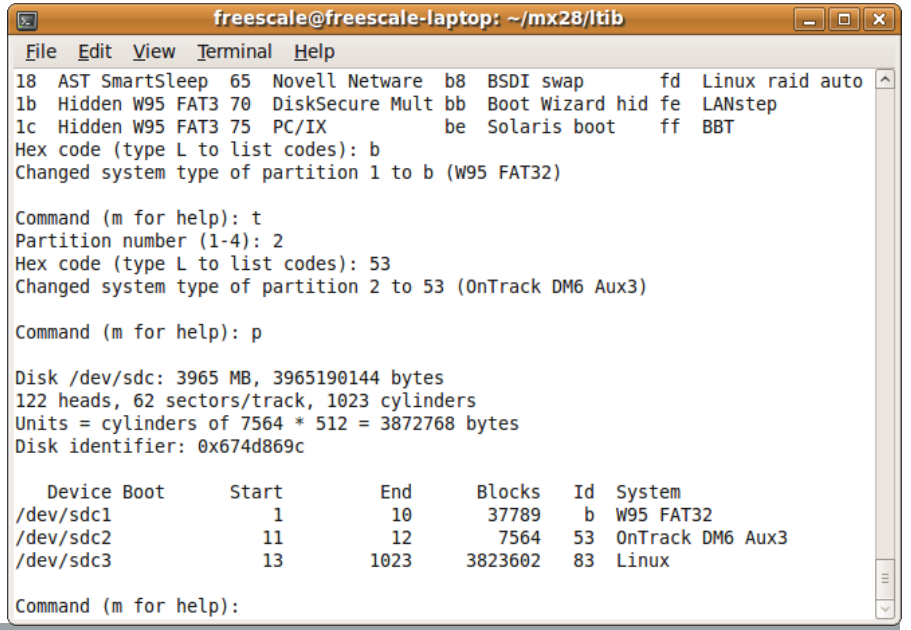
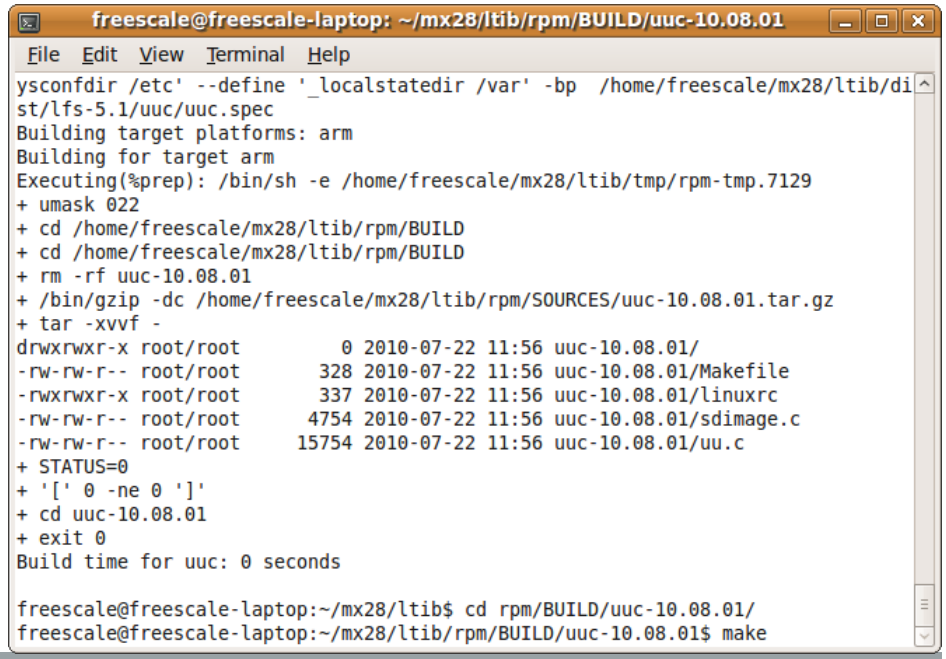
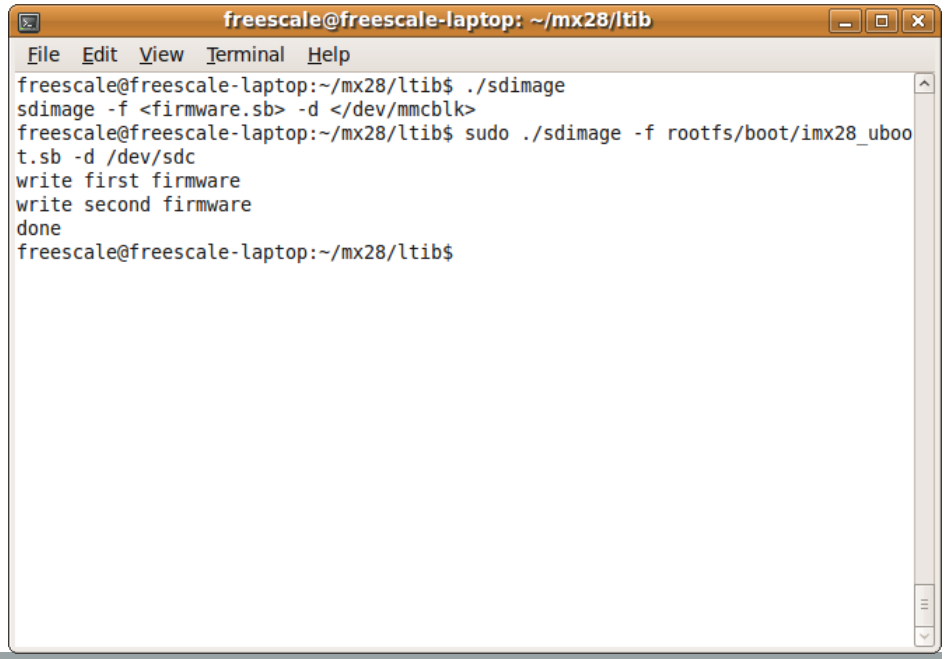
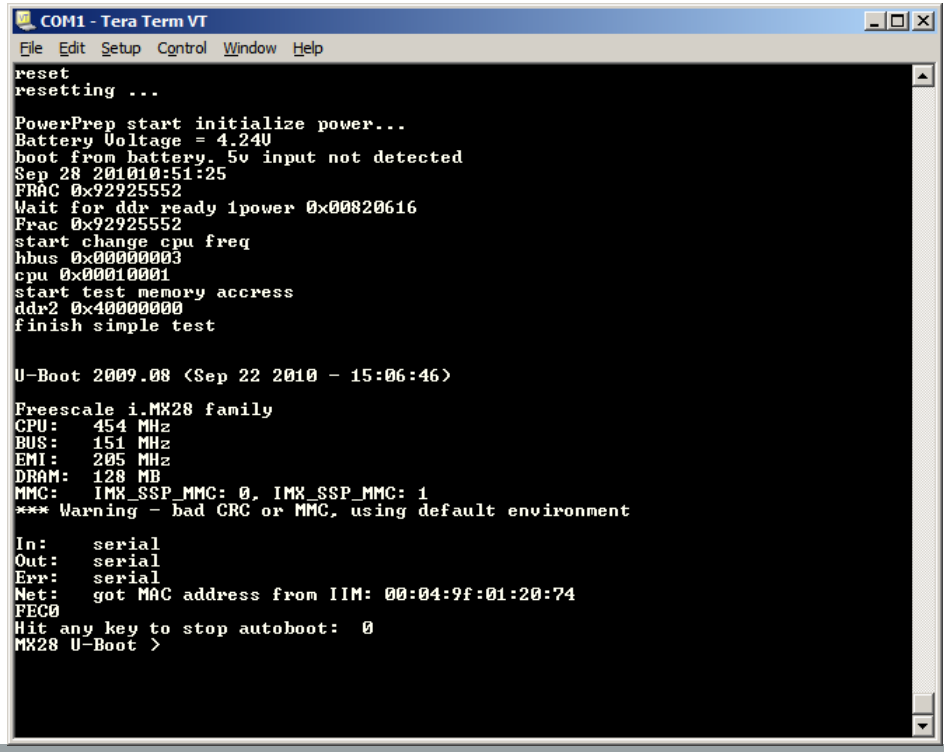
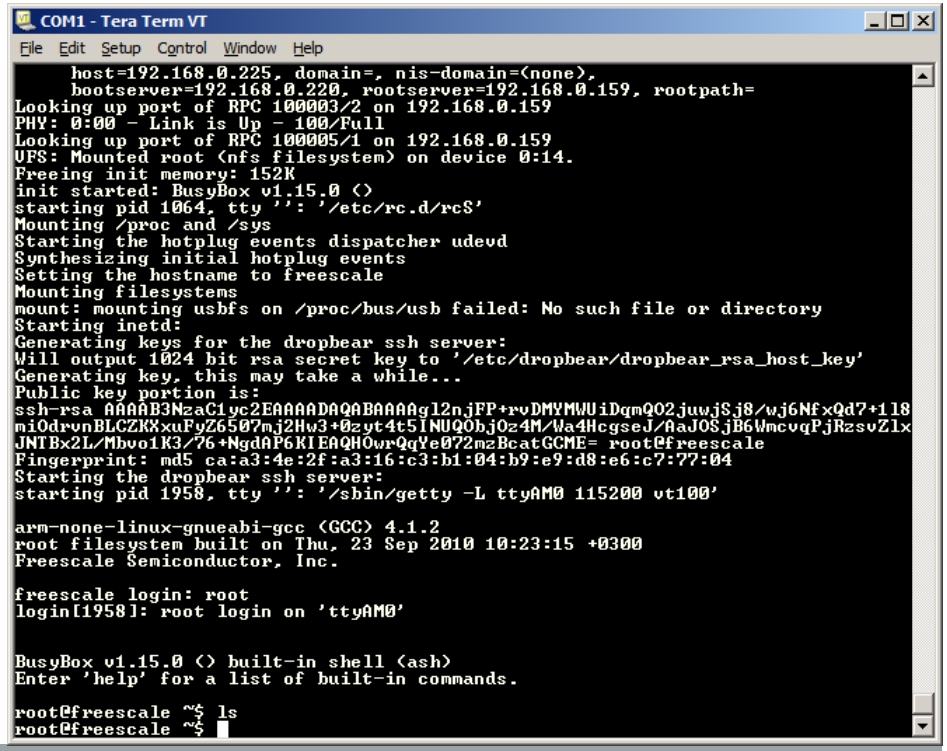
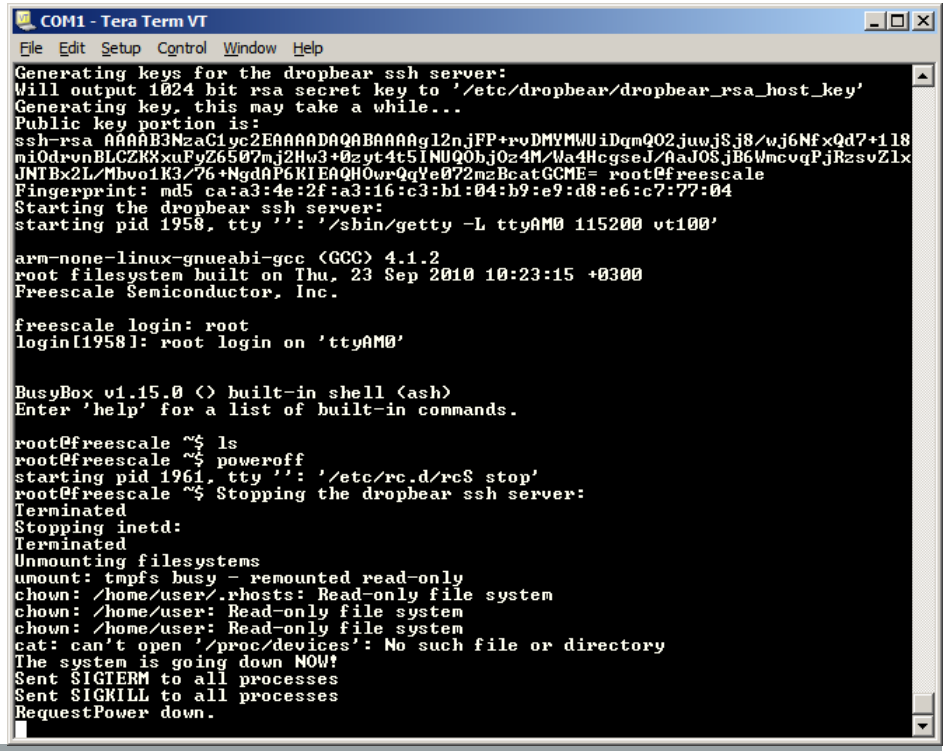
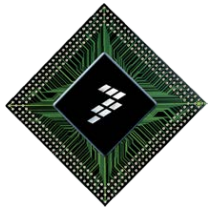
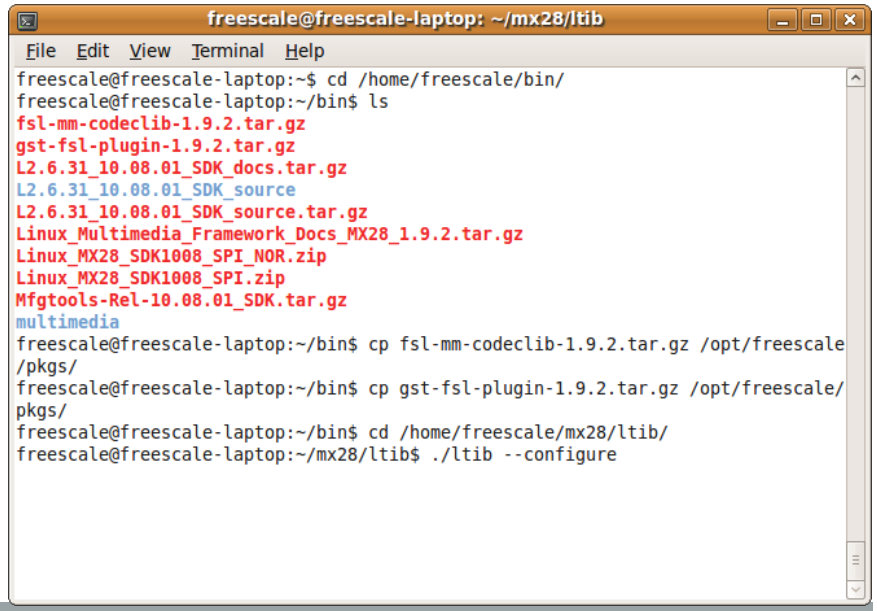
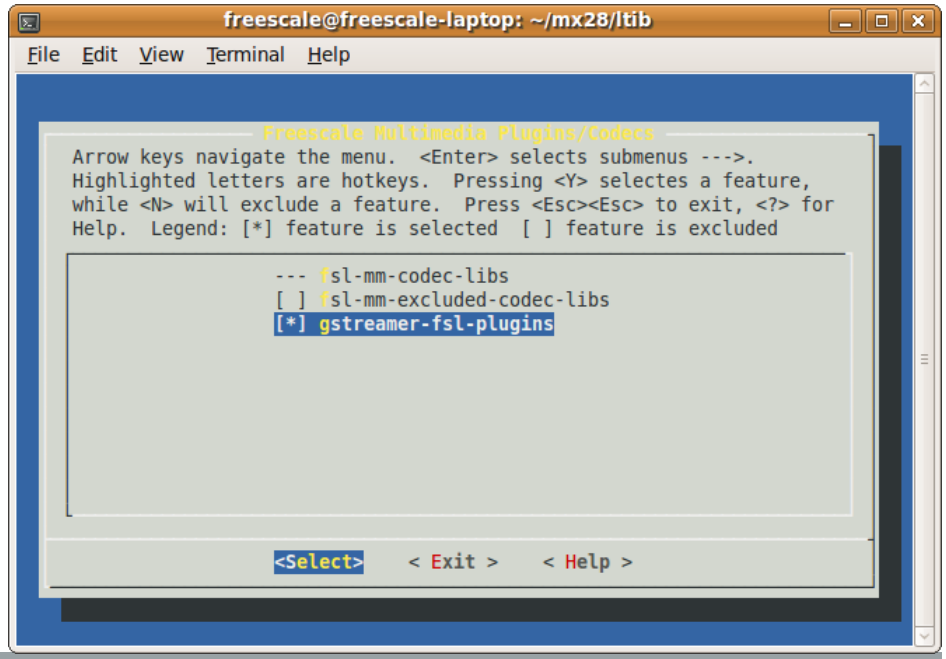
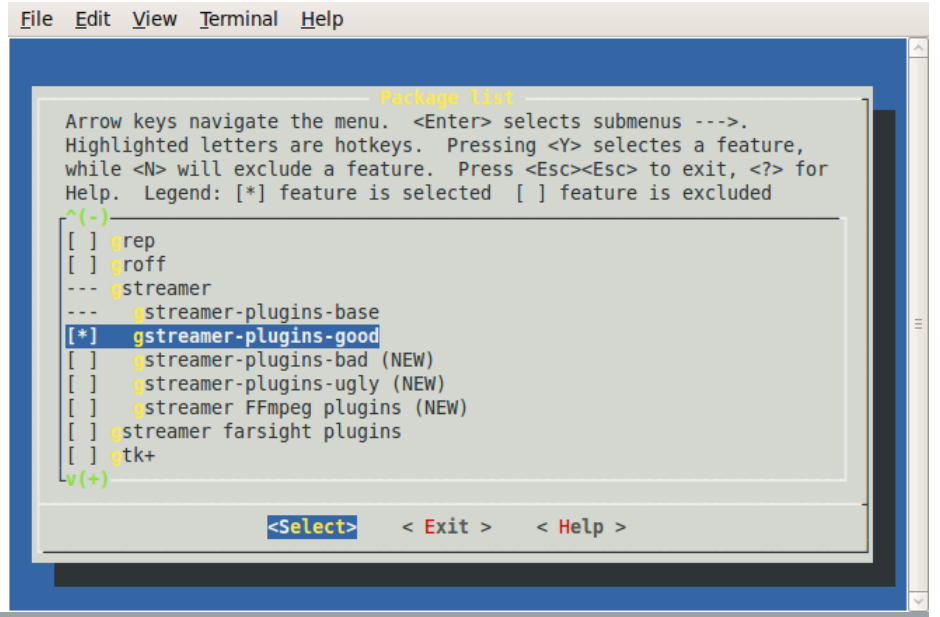
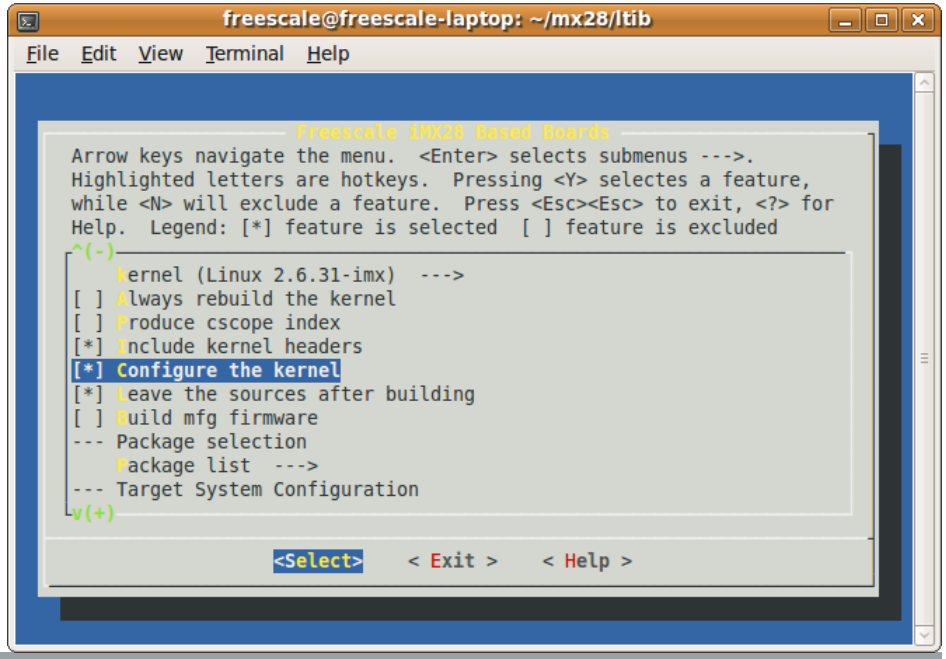
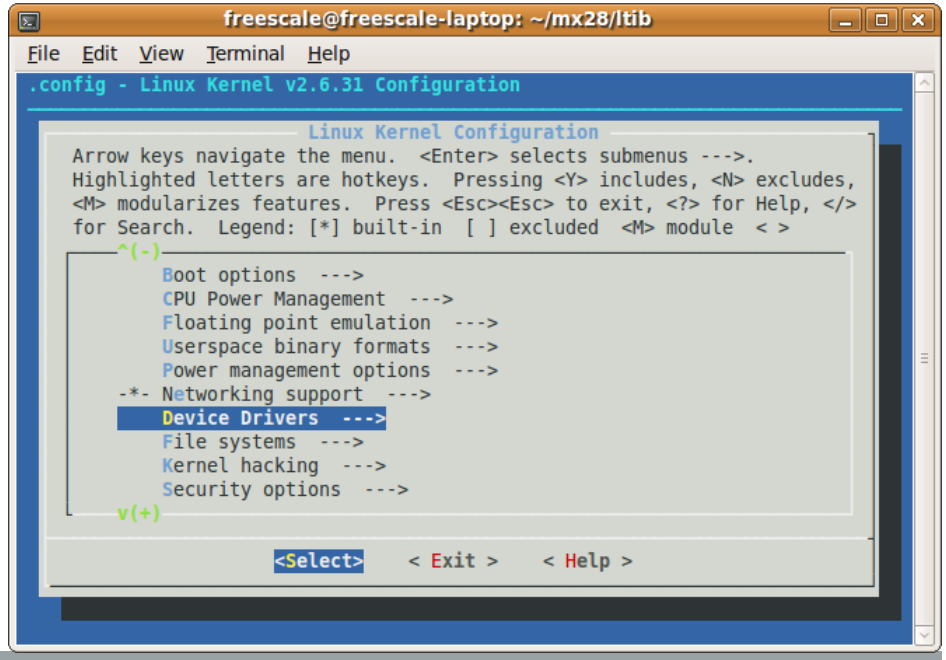
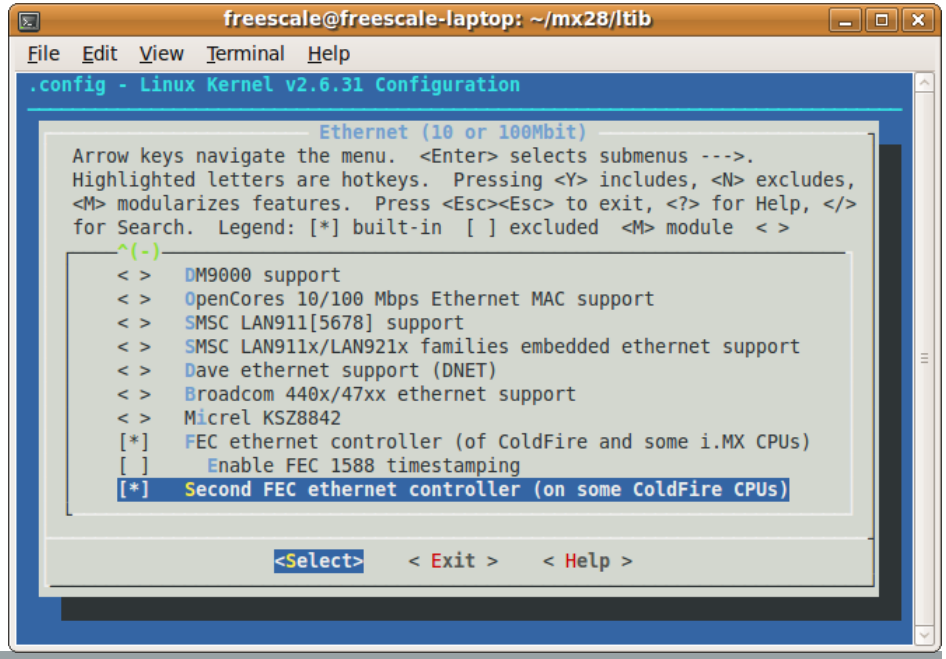
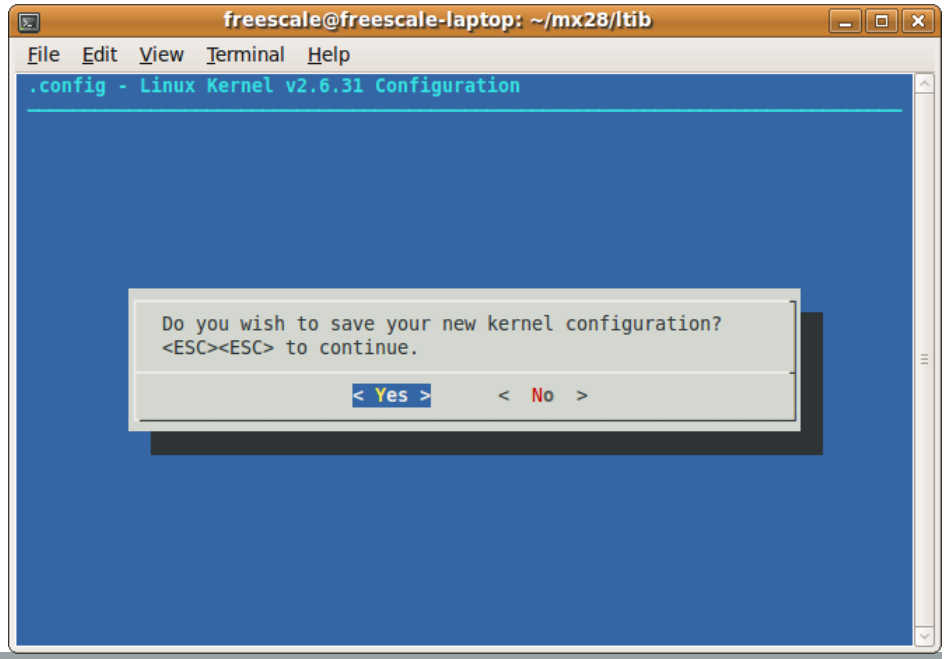
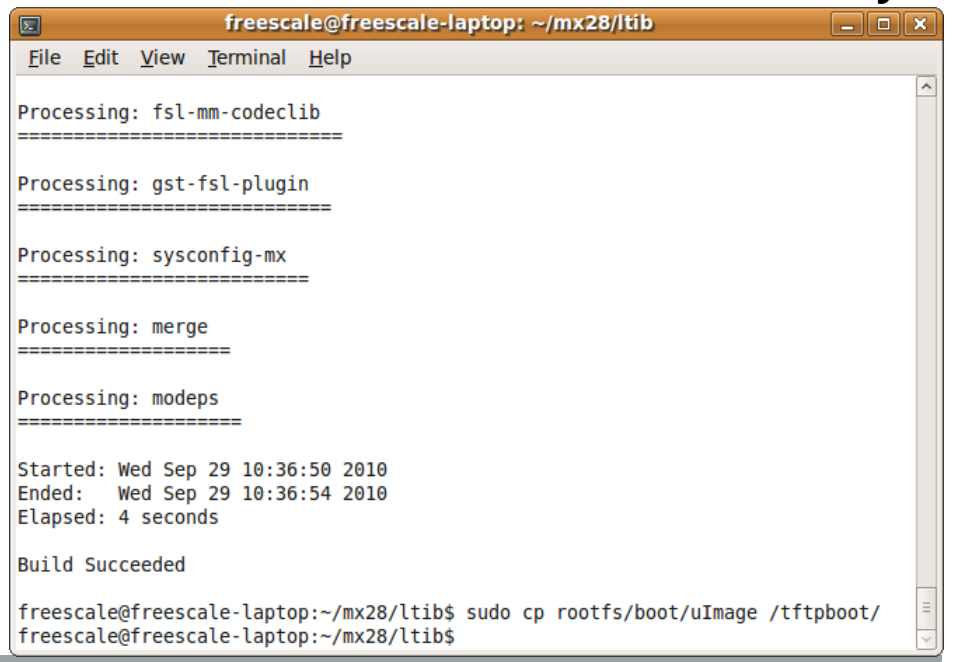
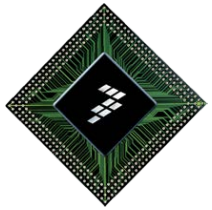
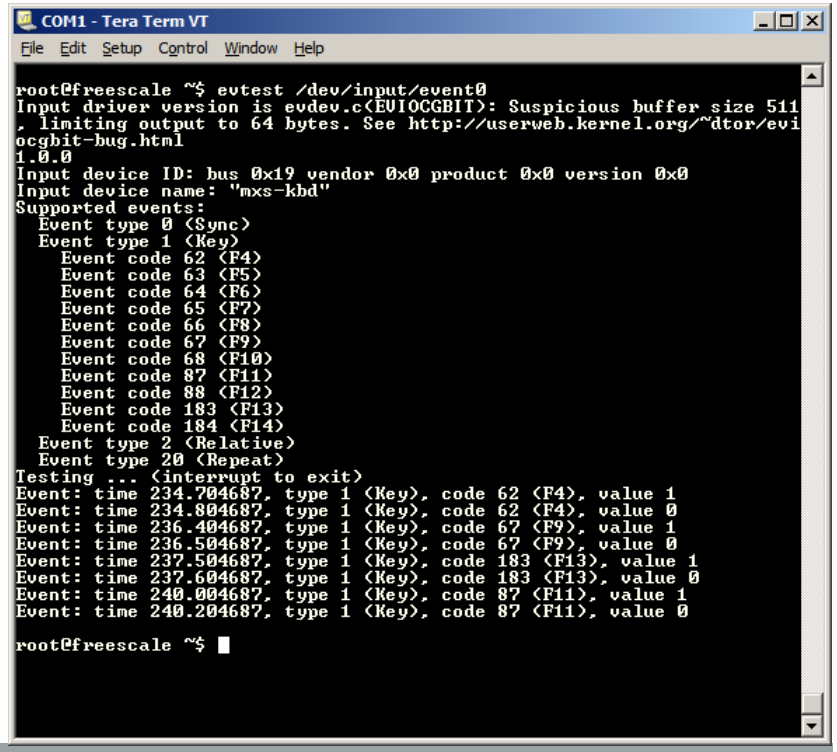
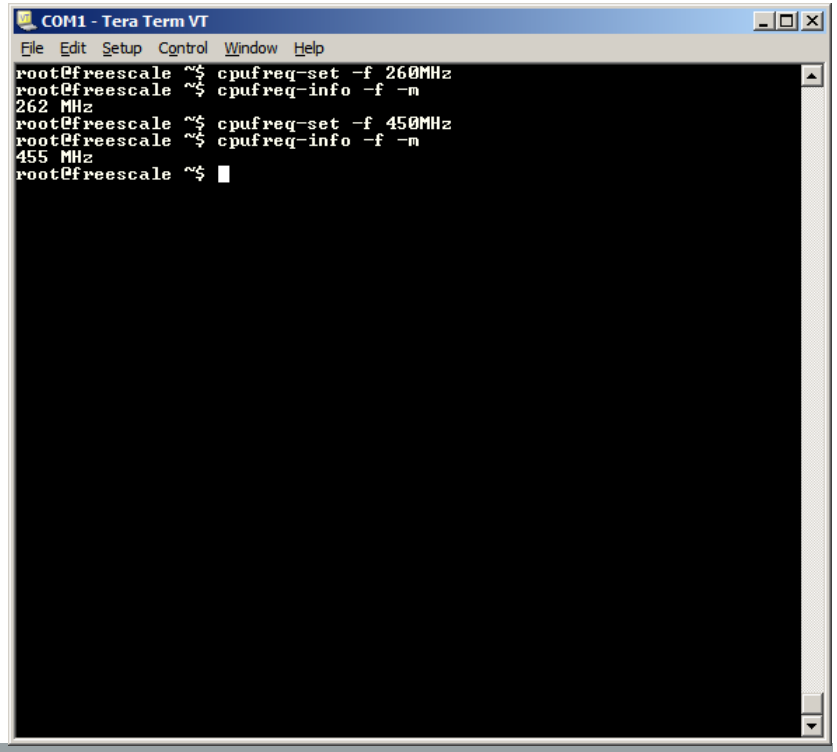
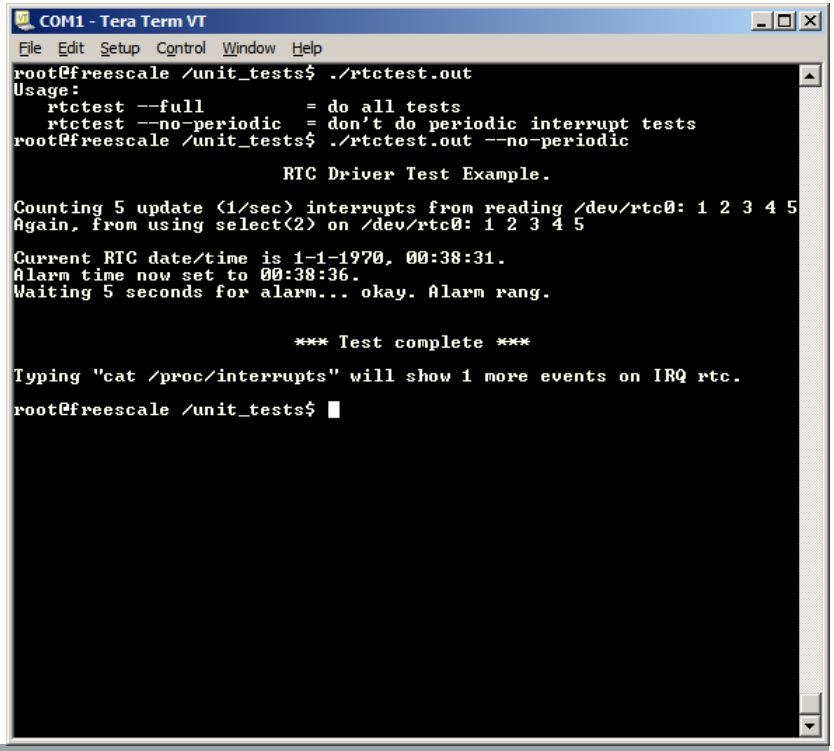
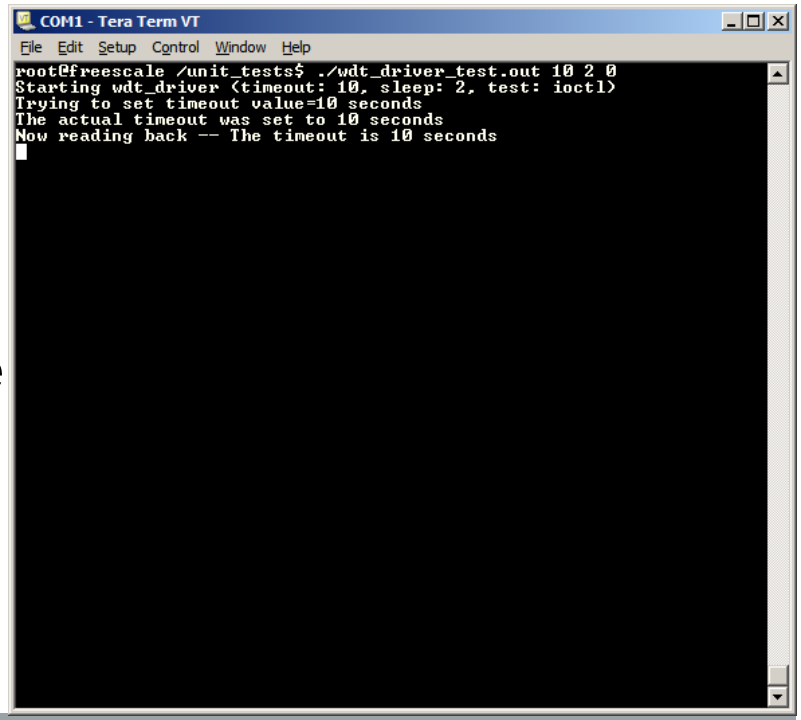
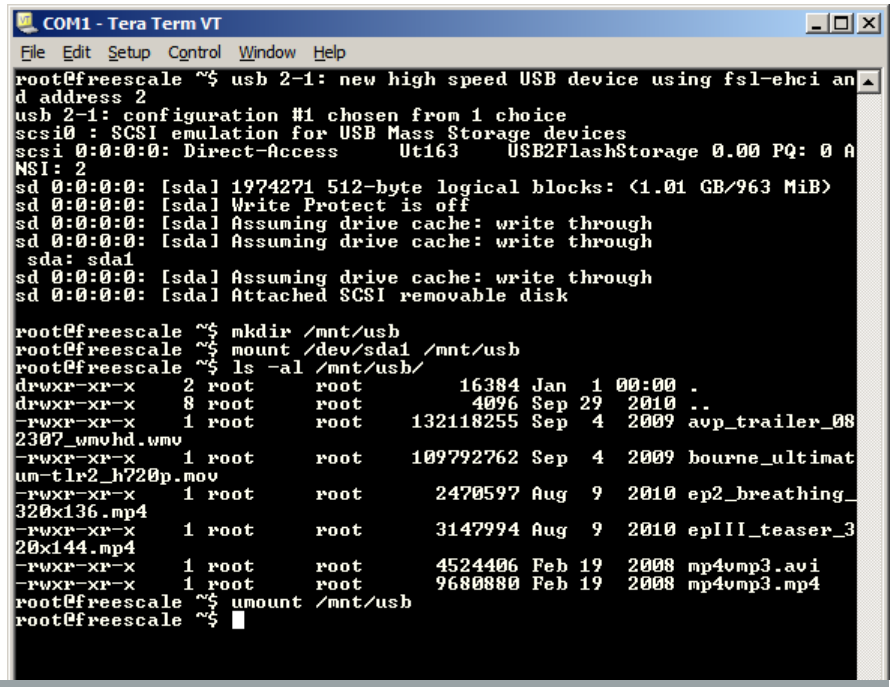
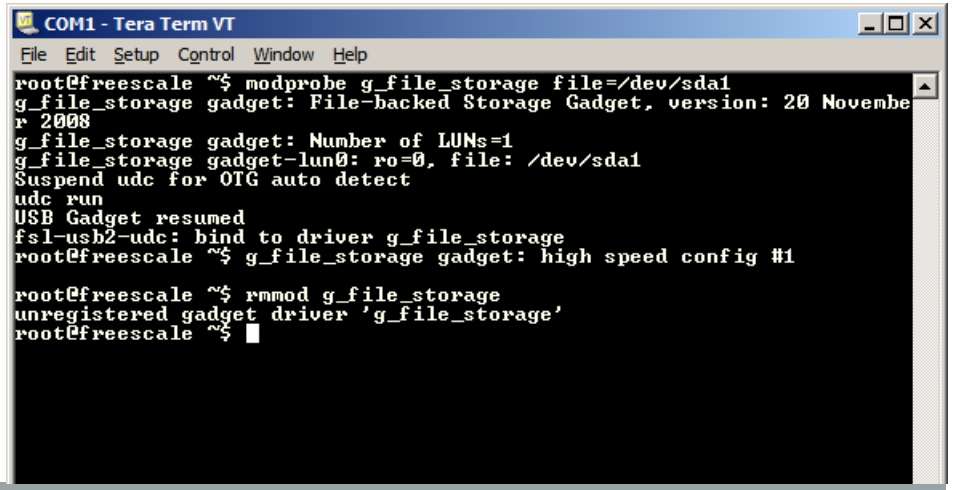
- Linux BSP can be installed on a number of Linux distributions if they are fairly recent (Fedora, Redhat, Debian, Ubuntu…)
- Most of work within Freescale is done on Ubuntu 9.04 or later and it is recommended for use
- See also ltib_build_host_setup.pdf in Linux documentation package

- For purpose of this training we will use Oracle VirtualBox virtual machine with Ubuntu 9.04 installation
- Most important when using VirtualMachine for development is to make sure “Bridged” networking is used, and that virtual machine is accessing correct Networking addapter (see Settings->Networking)
  - Linux host will act as TFTP and NFS server, so bridged networking is required
- Throughout the training following networking setup will be used:
  - Windows HOST: 192.168.0.200
  - VirtualMachine Linux host: 192.168.0.220
  - i.MX28EVK: 192.168.0.225

Ubuntu Installation

- Install Ubuntu 9.04
  - Download Ubuntu 9.04 ISO file from www.ubuntu.com
- Make sure there is enough free space available in Ubuntu after installation for working with BSP (at least 5GB)
- After Ubuntu starts, window will appear asking to install latest updates
  - Press “Install Updates” to update the system

Ubuntu Setup, Step 1

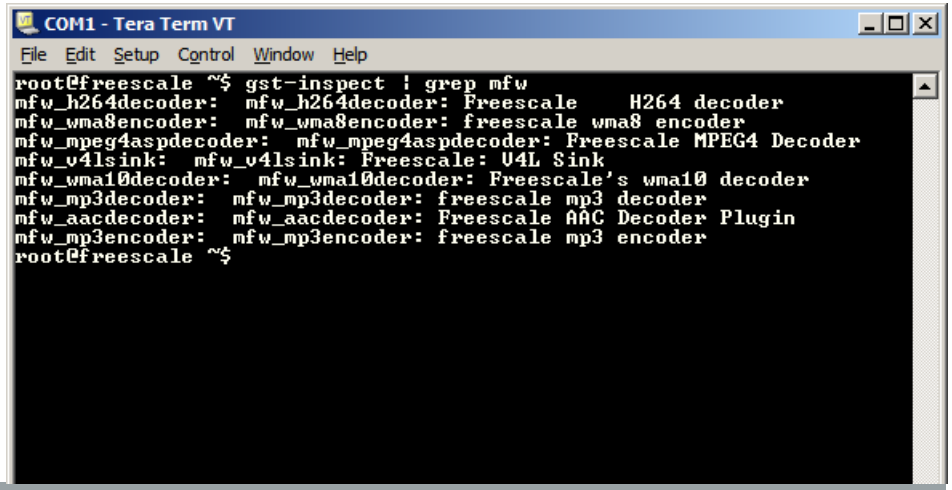
- Switch default shell to bash
  - `sudo rm /bin/sh`
  - `ln -s /bin/bash /bin/sh`
- Install additional packages needed by Linux BSP
  - `sudo apt-get install tftpd-hpa nfs-kernel-server gcc g++ make qt4-designer qt4-doc qt4-qmake qt4-qtconfig rpm ncurses-base ncurses-bin ncurses-dev m4 bison patch tcl gettext gettext-kde libdbus-glib-1-dev libgtk2.0-dev liborbit2-dev intltool minicom libtool ssh liblzo2-dev uuid-dev`
- Create directory for TFTP server
  - `sudo mkdir /tftpboot`
- Configure TFTP server by modifying configuration files (`sudo gedit _filename_`)
  - add following line to /etc/inetd.conf
    - `tftp dgram udp wait root /usr/sbin/in.tftpd /usr/sbin/in.tftpd -s /tftpboot`
  - modify /etc/default/tftpd-hpa to:
    - ```
      #Defaults for tftpd-hpa
      RUN_DAEMON="yes“
      OPTIONS="-l -s /tftpboot/"
      ```
      ```

Ubuntu Setup, Step 2

- Configure directory that NFS server will export by adding following line to /etc/exports
  - `/tftpboot/ltib *(rw,no_root_squash,no_subtree_check,async)`
- Note that /tftpboot/ltib directory doesn't exist yet. We will create a link to root filesystem later, once we create it.

Ubuntu Setup, Step 3

- As we will use serial port continuosly we need to set up terminal application (minicom)
  - type `dmesg | grep tty` to see list of serial ports
  - will most likely be `/dev/ttyS0` (when there is serial port on PC) or `/dev/ttyUSB0` (in case of USB->serial converter)
- Start minicom as `sudo minicom`
- Press Ctrl-A and then “Z” for help
- Press “O” to configure Minicom
- Go to “Serial port setup”
  - change “Serial Device” that is used (if needed)
  - change “Bps/Par/Bits” to 115200 8N1
  - disable hardware flow control



- Back in main configuration menu choose “Save setup as dfl” and exit
- Might have to enable wrapping when starting Minicom (Ctrl-A, Z, W)
- When you want to exit Minicom press Ctrl-A and then “X”

freescale semiconductor

Installing and Booting BSP



freescale™ semiconductor

Booting Development Host

- Import the Virtual Machine ("Import Appliance") to VirtualBox
- Check machine details
  - Make sure that Network adapter is "Bridged adapter" to proper network interface on the computer (if there are several)
- Press "Start" to boot up





- Ubuntu will boot up directly to "freescale" user account
- If needed during training, password is "freescale"

freescale semiconductor

What is Included in the Image

- `/home/freescale/bin` includes
  - BSP source archive (L2.6.31_10.08.01_SDK_source.tar.gz)
    - Unpacked already for convenience to L2.6.31_10.08.01_SDK_source
  - BSP documentation
  - Freescale multimedia codec libs
  - FSL Gstreamer plugins
  - Multimedia framework documents
  - Manufacturing Tools
  - Patches to enable SPI functionality
  - Multimedia folder with some video files we'll use to test video playback
  - Qt libraries



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Installation process for Linux BSP is same across all i.MX products
- Open new terminal window
- Change to the directory that contains unpacked Linux BSP
  - `cd /home/freescale/bin/L2.6.31_10.08.01_SDK_source`
- Run install script
  - `./install`
- Type Y and Enter to continue



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Press SPACE to go through EULA agreement
- In the end type “yes” and press Enter if you accept EULA
- Enter directory where LTIB will be installed
  - for this training enter /home/freescale/mx28



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Installer will start to copy files necessary files
- Once installation is finished, follow instructions shown to complete the installation
  - `cd /home/freescale/mx28/ltib`
  - `./ltib`



freescale semiconductor

Installing Linux BSP for i.MX Processors

- LTIB will start to install host support packages
  - installed to /opt/freescale
  - Only needs to be done once per host (re-used on subsequent installs - as for this training)
  - Could take up to an hour when run for first time
- If an error occurs, look at host_config.log to see what caused it
  - Usually a package or library is missing on host PC



freescale semiconductor

Configuring Access Rights

- One common failure during first install is user rights setup
  - LTIB needs to execute RPM commands with root privileges
  - Normally you want to log-in and work as normal user
  - Therefore follow the intructions shown on screen by LTIB to allow rpm to run as root without requiring password for a given user
- We don't have to do that now as it is already set up



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Once LTIB is finished, initial configuration screen will appear
- Read short instructions at the top on how to use menu system
- Pay attention to what action is selected in the bottom
- Leave “Platform choice” as is
- “Exit” this screen



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Select “Yes” to save the new configuration as default one



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Select imx28 as the platform type
- Leave “Min profile” as selected packages profile
  - Profile defines packages (applications) that are selected (enabled) and will be cross-compiled, built and deployed to target system
  - Can be changed later either by changing profile or manually selecting additional applications
  - Exit and save configuration



freescale semiconductor

Installing Linux BSP for i.MX Processors

- This is the most common LTIB screen that we will use
- Allows access to various configuration parameters and selections
- Can always come back to this menu later from command line by typing `./ltib --configure`



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Select “Package list” to view list of available and selected packages (do not enable/disable any package yet); Exit back to main menu
- Target System Configuration allows configuring of target run-time parameters (network, services started automatically, etc.)
- Target Image Generation allows to select what kind of image will LTIB generate and configure it



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Select “Target Image Generation” Options
- Change “Target image” to NFS only
  - for start, we will work with network-mounted file system only
- Exit to main screen
- Exit main screen
- Save the new configuration



freescale semiconductor

Installing Linux BSP for i.MX Processors

- After configuring is done, LTIB will start to build the target system
- If all goes well, “Build Succeeded” message will be shown ☺
- If not, error message will be shown
  - most common build error is dependency problem due to cross-compiling, when package that is compiled needs certain binaries to be installed on host computer



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Generated file system is available in `rootfs` folder
  - `ls –al rootfs/` to see content of generated file system
  - Kernel and uboot images (bootstreams) are stored in `rootfs/boot/` together with other relevant kernel files
  - This directory will be always be populated, as it is also used to generate file system image that can be programmed to end device



freescale semiconductor

Boot Stream

- Out of reset, internal ROM is executed first
- Internal ROM decides what is boot source
  - Decision is based on value of boot mode pins or eFuse settings
- Internal ROM will locate, retrieve and execute the boot stream which consists of small bootlets
- Bootlets are small pieces of code executed during boot to set up some basic system functionality (memory, clocks, etc.)
  - Each bootlet is built separately and may or may not know about others
  - Boot stream can instruct the ROM to call any number of bootlets before final jump to
- i.MX28 boot streams contain following bootlets:
  - power_prep – configures the power supply
  - Boot_prep – configures clocks and SDRAM
  - Linux_prep – prepaires to boot Linux

freescale semiconductor

A Word on Configuring Boot Stream Options

- In default setup, up to 4 different linux kernel command lines can be configured
- If Linux kernel is rebuilt OR one of kernel command line changes, need to re-built boot stream manually
  - `./ltib –p boot_stream.spec –f`
- During the hands-on we will boot U-boot first, not directly Linux kernel
  - Linux kernel command line will be configured within U-boot



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Kernel image must be copied to TFTP directory
  - `sudo cp rootfs/boot/uImage /tftpboot/`
- Network File System server exports `/tftpboot/ltib` as NFS; needs to be connected to `rootfs` that we just generated
  - `sudo ln –s /home/freescale/mx28/ltib/rootfs /tftpboot/ltib`
  - `sudo /etc/init.d/nfs-kernel-server restart`
- Every time we change NFS link, we should restart NFS server
  - `sudo /etc/init.d/nfs-kernel-server restart`
- `ls –al /tftpboot/`



freescale semiconductor

Installing Linux BSP for i.MX Processors

- It is important to understand the setup
- TFTP server will make zImage available
  - bootloader (Uboot) will run from SD card and download zImage using TFTP
- NFS server will export `/tftptpboot/ltib`
  - `/tftpboot/ltib` is pointing to `rootfs` that was just generated by LTIB
  - this will be root file system for Linux kernel running on i.MX28EVK
  - files in `/home/freescale/mx28/ltib/rootfs` are all that i.MX28EVK can access on host PC



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Next we need to program the bootloader to SD card
- We will do it manually from Linux command line
  - Other option is to use Manufacturing Tool
- Attach the USB SD card reader
  - Windows will complain if there's no card in reader, but ignore that message
- "Capture" the reader in VirtualBox by selecting it in Devices -> USB Devices
- This will start the "Found New Hardware Wizard" which should be able to install the driver automatically
  - If SD card is not detected later, you will have to shut down Ubuntu in VirtualBox and restart Windows





freescale semiconductor

Installing Linux BSP for i.MX Processors

- Insert SD card to SD card reader
- Type `dmesg` and look for SD card device name
  - In example picture shown here it's **sdc**, so full device path is /dev/sdc
  - May change on other computers, will use **sdX** in examples here - adjust example according to your PC
- In order to create new partition structure, unmount all partitions
  - `sudo umount /dev/sdX*`



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Let's create the partitions on SD card
- Default kernel is set up for following SD card structure:
  - File Allocation Table (FAT)
    - needed to prevent Windows from formatting the card
    - Not needed otherwise
  - Boot stream partition of type 0x53 (OnTrack DM6 Aux3)
    - we'll only use this partition in the training for programming the bootloader
    - In real-life linux kernel bootstream would go here
  - Linux rootfs such as ext2.
    - We won't use this partition in the training
    - In real-life, file system and all apps would go here

Installing Linux BSP for i.MX Processors

- Start fdisk on the SD card device
  - `sudo fdisk /dev/sdX`
- Type ‘p’ + <enter> to see all partitions and their numbers
- Delete all partitions on SD card
  - Type ‘d’ + <enter>
  - Enter partition number
  - Repeat above steps until there are no partitions left



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Now let's create new partition structure with following sequence
  - `n <ent> p <ent> 1 <ent> <ent> +10M <ent>`
    - New primary part. 1 ending on 10th cylinder
  - `n <ent> p <ent> 2 <ent> <ent> +4M <ent>`
    - New prim. part. 2 ending on 12th cylinder
  - `n <ent> p <ent> 3 <ent> <ent> <ent>`
    - New prim. Part. 3 starting after 2nd part. and occupying rest of card
  - `t <ent> 1 <ent> b <ent>`
    - Change part. 1 type to FAT32
  - `t <ent> 2 <ent> 53 <ent>`
    - Change part. 2 type to 0x53
  - `w <ent>`
    - Write partition table and exit



freescale semiconductor

Installing Linux BSP for i.MX Processors

- We can't write bootstream directly to /dev/sdX2 (second partition where bootstream will be located)
  - small table with some extra information is required to be pre-pended to bootstream
  - Used by internal ROM to locate proper bootstream
  - Allows for several backup copies of bootstream for unreliable media (NAND)
- Need to use program called sdimage to do that
- This program is part of uuc package in LTIB. Let's build it!
- First, unpack the uuc package
  - `./ltib –m prep –p uuc`

Installing Linux BSP for i.MX Processors

- Build the uuc package
  - `cd rpm/BUILD/uuc-10.08.01/`
  - `make`
- This will build the sdimage binary in current folder
- Note that we have built the uuc **natively**! We will run it on the host PC, not on i.MX28
  - We could have cross-compiled it for execution on i.MX28 with:
    `./ltib –m scbuild –p uuc`
    `./ltib –m scdeploy –p uuc`
- Copy the binary to LTIB install folder
  - `cp sdimage ../../..`
  - `cd ../../..`



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Now let's program the bootstream to SD card
  - `sudo ./sdimage –f rootfs/boot/imx28_ivt_uboot.sb –d /dev/sdX`
- Sdimage will check the device partition table and program bootstream
- That's it!
  - Take SD card out of reader
  - Insert to SD card socket 0
- Note: there are two types of bootstream images generated:
  - for devices with HAB enabled
  - for devices with HAB disabed



freescale semiconductor

Installing Linux BSP for i.MX Processors

- Start a new terminal window: 115200baud; parity: none; 8-bit;
- All commands issued to i.MX28EVK must be issued through this terminal window (no LTIB stuff)!
  - Do not mix terminal window for host PC and i.MX28EVK! ☺
- Connect Ethernet and serial cable between PC and EVK
  - Use lower Ethernet jack on i.MX28EVK
- Power-on i.MX28 EVK
- Interrupt EVK boot sequence by pressing any key



freescale semiconductor

Installing Linux BSP for i.MX Processors

- We need to modify default uBoot settings to match to our setup
  - `MX28 U-Boot > setenv bootargs 'console=ttyAM0,115200n8'`
  - `MX28 U-Boot > setenv bootcmd 'run bootcmd_net'`
  - `MX28 U-Boot > setenv bootdelay 2`
  - `MX28 U-Boot > setenv baudrate 115200`
  - `## Switch baudrate to 115200 bps and press ENTER ...`
  - `MX28 U-Boot > setenv serverip 192.168.0.220`
  - `MX28 U-Boot > setenv netmask 255.255.255.0`
  - `MX28 U-Boot > setenv ipaddr 192.168.0.225`
  - `MX28 U-Boot > setenv bootfile uImage`
  - `MX28 U-Boot > setenv loadaddr 0x42000000`
  - `MX28 U-Boot > setenv nfsroot /tftpboot/ltib`
  - `MX28 U-Boot > setenv bootargs_nfs 'setenv bootargs ${bootargs} root=/dev/nfs ip=${ipaddr}:${serverip}::${netmask}::eth0:off nfsroot=${serverip}:${nfsroot} fec_mac=00:01:02:03:04:05 gpmi'`
  - `MX28 U-Boot > setenv bootcmd_net 'run bootargs_nfs; tftpboot; bootm'`
  - `MX28 U-Boot > saveenv`
- Type `reset` to reset board

Installing Linux BSP for i.MX Processors

- Uboot should execute boot script, if all went well
  - Kernel will be loaded using TFTP
  - Redboot will execute the kernel, passing provided startup string to kernel
  - At some point during boot, kernel will attempt to mount file system using NFS
- If all goes well, you will be presented with a login prompt



freescale semiconductor

Installing Linux BSP for i.MX Processors

- When asked for login, enter root
- Feel free to look around the system as it is up & running
- In the end initiate power off sequence to EVK by typing:
  - `poweroff`



freescale semiconductor



Installing additional components

freescale™ semiconductor

Installing Additional Components

- Now we should all have basic system up and running
- Let's continue by making following modifications to the system
  - Install Gstreamer plugins and libraries to support audio/video playback
  - Recompile the kernel to add support for second Ethernet port
  - Install additional useful applications

Multimedia Framework Package

- Multimedia codecs are offered in a package that is downloaded separately from Freescale's web site
- Provided for free, no license or royalty payments to Freescale required
- Can be easily added to LTIB/Linux BSP installation for ease of use
- Current packages:
  - gst-fsl-plugin-1.9.2.tar.gz
  - fsl-mm-codeclib-1.9.2.tar.gz
- Documentation in a separate archive:
  - Linux_Multimedia_Framework_Docs_MX28_1.9.2.tar.gz

Adding Multimedia Packages to BSP

- Open terminal window on host and switch to folder where packages are: `cd /home/freescale/bin`
- Copy source/binary packages used during build process to Local Package Pool (/opt/freescale/pkgs)
  - `cp fsl-mm-codeclib-1.9.2.tar.gz /opt/freescale/pkgs/`
  - `cp gst-fsl-plugin-1.9.2.tar.gz /opt/freescale/pkgs/`
- Once package is in LPP, it is re-used across other LTIB installs on same machine
- Go back to LTIB install folder
  - `cd /home/freescale/mx28/ltib`



freescale semiconductor

Selecting Multimedia Codecs

- Need to select packages that we want LTIB to build
  - `./ltib --configure`
- Go to “Package list” and to “Freescale Multimedia Plugins/Codecs”
- Select fsl-mm-codec-libs and gstreamer-fsl-plugins
- Exit back to package list



freescale semiconductor

Selecting Multimedia Codecs

- Scroll down to gstreamer packages
- Select gstreamer-plugins-good package



freescale semiconductor

Installing Additional Components

- At this point lets select additional applications to install to target file system
  - imx-test
  - alsa-utils
  - tslib
  - can4linux
  - cpufrequtils
  - dhcp (enable only "Include DHCP Client Support“; disable DHCP server)
  - dropbear SSH client/server
  - evtest
  - freetype
  - iperf
  - libpng
  - netperf
  - strace
  - usbutils

Changing Kernel Configuration

- Exit to main screen
- Enable the “Configure the kernel” option
  - This will trigger the menuconfig-based Linux kernel configuration later when LTIB starts to build the Linux kernel



freescale semiconductor

Finishing the Work

- At this point we are done with system configuration and it is time to build the newly added applications

- Exit main screen
- Save the new configuration

- Build process will start automatically

Configuring Kernel

- When LTIB starts to building the Linux kernel it will launch the menu-based configuration process
- Go to “Device Drivers”



freescale semiconductor

Configuring Kernel

- Then go to “Network Device Support” and “Ethernet (10 or 100Mbit)”
- Enable “Second FEC Ethernet controller (on some ColdFire CPUs)”



freescale semiconductor

Configuring Kernel

- Exit all the way back to main configuration screen
- Exit the main configuration screen
- Save your new kernel configuration when asked to do so

- Kernel build process will now continue



freescale semiconductor

Final Steps

- Once kernel is build, LTIB will proceed to build all other newly selected packages
  - Build should succeed without problems.
- If we were going to use kernel boot stream directly (imx28_ivt_linux.sb) we would have to re-build bootstream manually
  - `./ltib –p boot_stream.spec –f`
- However, we are loading uImage binary using U-boot, so we just need to copy the new binary to place where U-boot will find it
  - `sudo cp rootfs/boot/uImage /tftpboot/`
- Power-on the board again
- Log-in as root



freescale semiconductor



Testing Various Parts of the System

freescale™ semiconductor

Event Subsystem

- To try push buttons on i.MX28
  - `evtest /dev/input/event0`
  - Press various keys on i.MX28EVK
  - You should see events generated
  - Press Ctrl+C to stop the program
- To try the touchscreen functionality
  - `evtest /dev/input/event1`



freescale semiconductor

Suspend and Wake-Up

- To suspend the system
  - `echo standby > /sys/power/state`

- Touch the screen and system wakes up

CPU Frequency

- Frequency settings available using standard Linux interfaces and tools

- Run `cpufreq-set –f 260MHz`
  - Will set CPU freq to 262MHz
- Run `cpufreq-info –f –m`
  - Will show current frequency
- Run `cpufreq-set –f 450MHz`
  - Will set CPU freq to 455MHz

- Run `cpufreq-info` to see other capabilities



freescale semiconductor

PWM Control

- There's two indication diodes on i.MX28EVK
- Connected to PWM0 and PWM1

- Diodes can be controlled
  - `echo NNN > /sys/class/leds/led-pwmX/brightness`
  - Where NNN is between 0 and 127; X is 0 or 1

RTC Test

- `cd /unit_tests/`
- `./rtctest.out --no-periodic`
  - Will run various tests on /dev/rtc0 using standard API for RTC



Watchdog Test

- `./wdt_driver_test.out 10 2 0`
  - Will enable watchdog with 10 sec timeout
  - Program will kick watchdog every 2 seconds using
  - When we stop the program with Ctrl+C board will reset after approximately 12 seconds
- `./wdt_driver_test.out 2 10 0`
  - Will enable wdog with 2 sec timeout
  - Will kick wdog every 10 seconds
  - Watchdog will kick in before first update and reset device



freescale semiconductor

Testing USB

- Insert memory stick to the USB Host port
- You will notice name of device under which it is available
  - In our example it is /dev/sda
- Create directory where you want to mount the device
  - `mkdir /mnt/usb`
- Mount the file system on device
  - `mount /dev/sda1 /mnt/usb`
- You can work with device through /mnt/usb
  - `ls –al /mnt/usb`
- When done, unmount it
  - `umount /mnt/usb`



freescale semiconductor

Testing USB OTG

- With memory stick still inserted, let's try USB OTG functionality
- We need to load gadget driver that will determine USB device functionality on the USB On-The-Go port
- When gadget is mass storage device, we need to provide memory location that will be used for mass storage
  - Can be real device (like memory stick) available under `/dev/` tree
  - Can be a file whos' content mimics a real file system

Testing USB OTG

- Load gadget driver, instructing it to use memory stick we inserted for mass storage
  - `modprobe g_file_storage file=/dev/sda1`
- Insert USB cable to PC
- EVK should be recognized as mass storage device and memory stick content should be visible!
- Remember to unmount the EVK on PC
- Remove gadget driver as well
  - `rmmod g_file_storage`
- Unplug USB cable from PC



freescale semiconductor

Testing USB OTG

- In order to create a file containing file system type following on host PC terminal in LTIB install folder:
  - `cd /home/freescale/mx28/ltib`
    `sudo mkdosfs –C rootfs/root/dosfs 8192`

- Switch back to serial terminal and check that file named **dosfs** is available
  - `cd`
    `ls –al dosfs`

Testing USB OTG

- Load gadget driver, instructing it to use dosfs file for mass storage
  - `modprobe g_file_storage file=/root/dosfs`
- Insert USB cable to PC
- EVK should be recognized as empty (~8MB) storage device
- When done, remember to unmount EVK on Host PC
- Remove g_file_storage module when not needed any more
  - `rmmod g_file_storage`
- dosfs can also be accessed from EVK
  - ```
    mkdir /mnt/fs
    mount dosfs /mnt/fs
    ls –al /mnt/fs
    umount /mnt/fs
    ```
    ```

Display Control

- To disable LCD automatic blank functionality type following
  - `echo –e “\033[9;0]” > /dev/tty0`
- If the display is off, type following to turn it back on
  - `echo 0 > /sys/class/graphics/fb0/blank`
- To control display brightness
  - `echo NNN > /sys/class/backlight/mxs-bl/brightness`
  - Where NNN is number between 0 and 100.

Using Multimedia Codecs

- Verify that codecs are installed properly
  - `gst-inspect | grep mfw`
  - You should see Freescale codecs installed
- In LINUX HOST copy the multimedia files to location where i.MX28 can see them
  - `sudo cp /home/freescale/bin/multimedia/* rootfs/root/`
  - Above command assumes you are in LTIB install folder

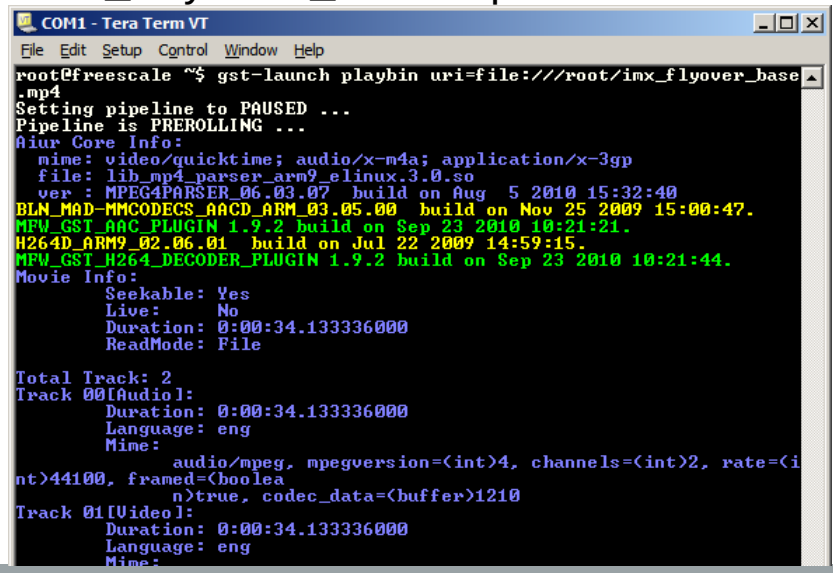


freescale semiconductor

- Plug in speakers to “Headphone Output” to hear sound

- cd /root
  - We copied multimedia files to /root folder
- Several options for video/audio playback
  - `gst-launch playbin uri=file:///root/imx_flyover_base.mp4`
  - `gplay imx_flyover_base.mp4`

- Audio too loud? Too quiet?
  - `amixer set Playback XXX`
  - Where XXX is between 0 and 192

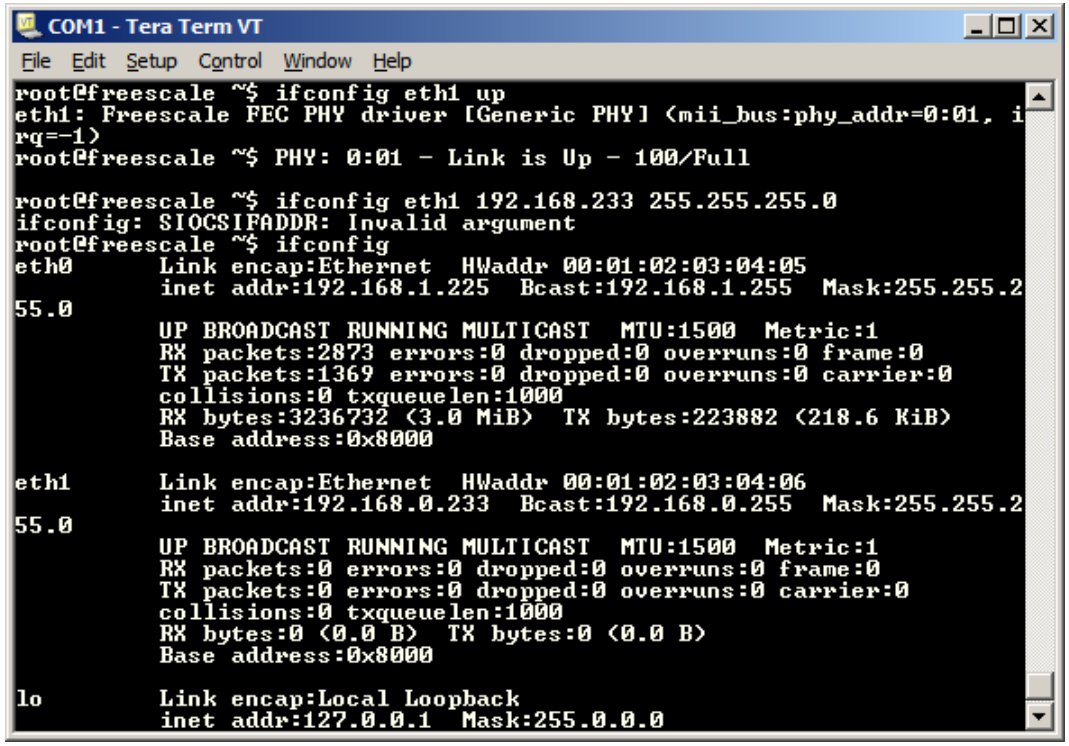


freescale semiconductor

- We've also selected and installed Dropbear SSH server on i.MX28
- You can connect from Linux host to the i.MX28 via SSH
  - `ssh root@192.168.0.225`
- This will require that you set a password for root on i.MX28
  - Default empty password doesn't work for SSH
- Can install and use other services available on Linux as well
  - Web server
  - Email server
  - Etc…

Second Ethernet Port

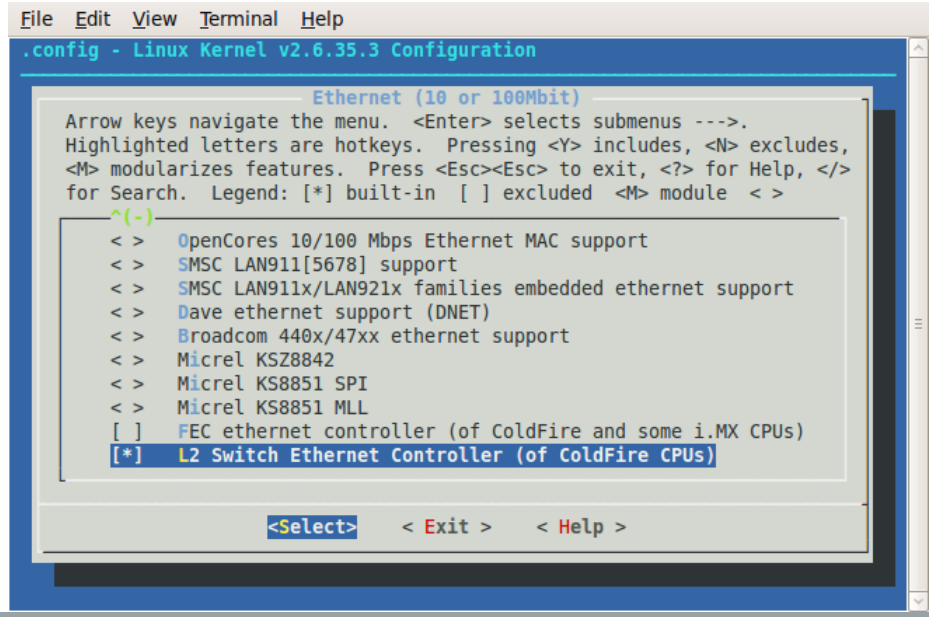
- Connect Ethernet cable to ENET1: UPPER JACK on i.MX28EVK
- `ifconfig eth1 up`
- `ifconfig eth1 IP_ADDR NETMASK`
  - IP_ADDR should belong to different subnet than current IP
  - Possible to use dhclient to obtain IP address automatically
  - Requires editing /etc/dhclient.conf not to run dhclient on eth0 but on eth1 interface
- Other network interface is now up & running



freescale semiconductor

Enabling i.MX28 L2 Switch

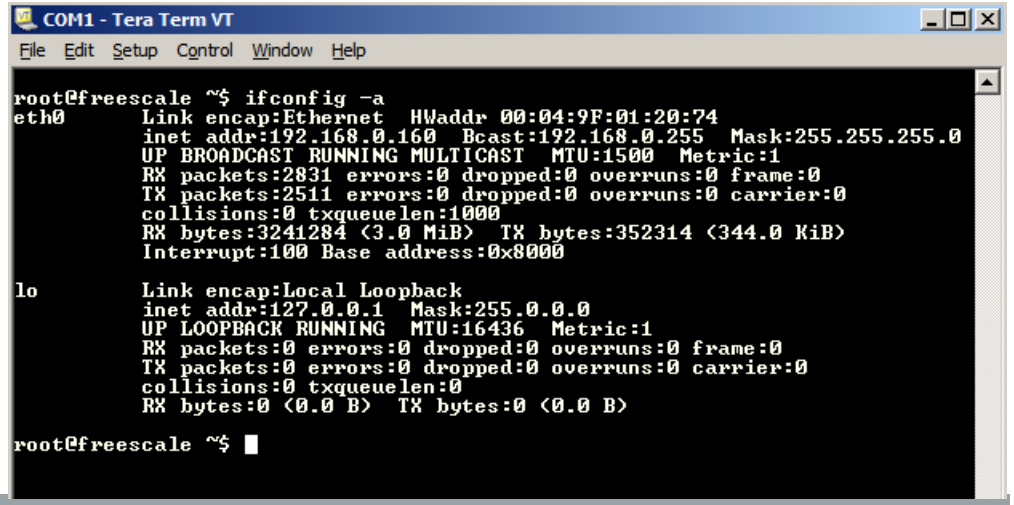
- By default setup only one Ethernet port is enabled
- Need to re-configure the kernel to enable L2 functionality
  - `./ltib --configure`
  - Enable “Configure the kernel” option in LTIB and Exit
  - Device Drivers -> Network device support -> Ethernet (10 or 100Mbit)
  - Disable “FEC Ethernet controller”
  - Enable “L2 Switch Ethernet controller…” option that will appear
- Overall, possible configs are:
  - Only one Ethernet port enabled
  - Two (separate) Eth ports enabled
  - L2 switch enabled



freescale semiconductor

Enabling i.MX28 L2 Switch

- With L2 switch enabled there is only one network interface visible
- You can verify switch functionality by connecting a device via Ethernet cable to ENET1 port
- Device is able to communicate to Ethernet network on ENET0 port



freescale semiconductor

Finishing

- Shut down the device
  - `poweroff`


freescale™
semiconductor