Creating Eco-System for R-Car LCB
How to develop BSP for SoC and what we did

Hisao Munakata

AGL advisory board member @ Linux Foundation

March 25th 2015
Why we need eco-system specially for embedded Linux?
Who am I?

- From embedded SoC provider company Renesas
- Responsible for OSS software development and delivery for R-Car series SoC
- Working with W/W car OEM and 1st tear IVI customers
- Linux Foundation CE$^1$ working Gr. Steering committee member, LF/CEWG Architecture Gr. co-chair
- One of LF/CEWG LTSI$^2$ project initial proposer
- At my company, I had been encouraging my team developers to send a patches upstream

---

$^1$CE = consumer electronics
$^2$LTSI = Long Term Support Initiative
15 years ago, my first Linux kernel port to SH7709
5 years ago, adding more peripherals for early SoC

- Video
- Audio
- GPU
- MMU
- interrupt
- DMA
- timer
- 32bit RISC SH4A CPU architecture
- serial
- USB
- I2C
- GPIO
- SATA
- Ether

Audio and video (=OpenMAX/IL) support added to BSP
### Why we need eco-system specially for embedded Linux?

Now extremely complicated drivers need integration

<table>
<thead>
<tr>
<th>VSP</th>
<th>Video</th>
<th>Audio</th>
<th>GPU</th>
<th>Display</th>
<th>CAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVDS</td>
<td>MMU</td>
<td>interrupt</td>
<td>DMA</td>
<td>timer</td>
<td>MOST</td>
</tr>
<tr>
<td>HDMI out</td>
<td>ARM cortex A15 architecture</td>
<td></td>
<td>timer</td>
<td>E-AVB</td>
<td></td>
</tr>
<tr>
<td>HDMI in</td>
<td>serial</td>
<td>USB</td>
<td>I2C</td>
<td>GPIO</td>
<td>Speed pulse</td>
</tr>
<tr>
<td>Video in</td>
<td>SATA</td>
<td>Ether</td>
<td>security</td>
<td>audio DSP</td>
<td>GPS</td>
</tr>
</tbody>
</table>

Unique device (IP) support becomes majority of Linux development work for modern SoC

### Background

PC variant vs. embedded

Why we need eco-system for embedded Linux

---

Hisao Munakata

Creating Eco-System for R-Car LCB
single Linux installer can boot for all PC variant

Commonality

Even they looks different

If you have a Linux distribution (like Debian) install DVD, you can boot PC, server and embedded PC with single amd64 binary image as they have commonality
Why we need eco-system specially for embedded Linux?

What we are doing for Renesas R-Car Linux development
embedded Linux eco-system

Background
PC variant vs. embedded
Why we need eco-system for embedded Linux

**embedded board requires dedicated kernel image**

---

**So different!**

- Reset and PMIC setting
- custom boot sequence
- boot media (eMMC, NOR, SD,..)
- original memory map
- original interrupt assignment
- unique on-chip IP feature (driver)
- complexed pin multiplex
- proprietary block (binary blobs)
- isolated power domain

---

Every embedded boards behave differently. Your previous experience might not applied to the new SoC, boards. Also, Google can not tell you how to bring your board if that is a minor one. Embedded Linux bring up is always painful work.
We need to deliver **Linux BSP for embedded hardware**

---

**Distribution** = verified collection of various Linux programs (=packages)
- per-build binary distribution = Debian, Ubuntu, Cent, Fedora
- source code distribution = Gentoo, Open Embedded (yocto), ...

**SDK (Software Development Kit)** = subset of distribution designed for specific application = Android, Tizen IVI, MPD, Drone Code

**BSP (Board Support Package)** = subset of SDK, designed for specific target hardware. Embedded SoC vendor develops BSP for their reference platform. Product producer modify BSP to fit with product hw
Battles between PC variant and embedded

Why we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system
conclusion

Background
PC variant vs. embedded
Why we need eco-system for embedded Linux

10/46 Hisao Munakata
Creating Eco-System for R-Car LCB
example (window system): X11 vs. Wayland

design policy comparison (GPU optimization point)

**X11 = conservative**
- sustainable solid API
- drawing is fully managed by X11 server side
- composition (=window manager) is out of X11 server and it cases bunch of IPC (=overhead)
- display surface allocated by X11 server side

**Wayland = aggressive**
- fully utilize GPU capability
- client (=apps.) can draw directly via DRI
- composition can be integrated to Wayland server and it makes drawing simple
- client reserver surface and send pointer
Why we need eco-system specially for embedded Linux?

What we are doing for Renesas R-Car Linux development embedded Linux eco-system

conclusion

Background

PC variant vs. embedded

Why we need eco-system for embedded Linux

example (window system) : **X11 vs. Wayland**

---

![Diagram of X11 vs Wayland](http://commons.wikimedia.org/wiki/File:Linux_graphics_drivers_DRI_current.svg)


---

Hisao Munakata

Creating Eco-System for R-Car LCB
example (window system): X11 vs. Wayland

Wayland application

OpenGL application

Wayland compositor

OpenGLES DRI driver

kms

drm

CPU (registers & L1 & L2 & L3 & L4) & main memory

GPU (registers & L1 & L2) & graphic memory

DRI drivers contains **GPU code**

munakata@muna-E450: /source/linux$ ls -l /usr/include/libdrm
total 292

drwxr-xr-x 2 munakata munakata 4096 Feb 6 14:31 .
drwxr-xr-x 67 munakata munakata 4096 Feb 6 14:11 ..
-rw-r--r-- 1 munakata munakata 25767 Feb 6 13:49 drm.h
-rw-r--r-- 1 munakata munakata 7818 Feb 6 13:49 drm_fourcc.h
-rw-r--r-- 1 munakata munakata 13152 Feb 6 13:49 drm_mode.h
-rw-r--r-- 1 munakata munakata 2629 Feb 6 13:49 drm_sarea.h
-rw-r--r-- 1 munakata munakata 29212 Feb 6 13:49 i915_drm.h
-rw-r--r-- 1 munakata munakata 7895 Feb 6 13:49 mach64_drm.h
-rw-r--r-- 1 munakata munakata 12923 Feb 6 13:49 mga_drm.h
-rw-r--r-- 1 munakata munakata 6593 Feb 6 13:49 nouveau.h
-rw-r--r-- 1 munakata munakata 5572 Feb 6 13:49 nouveau_drm.h
-rw-r--r-- 1 munakata munakata 2555 Feb 6 13:49 omap_drmif.h
-rw-r--r-- 1 munakata munakata 4221 Feb 6 13:49 qxl_drm.h
-rw-r--r-- 1 munakata munakata 9901 Feb 6 13:49 r128_drm.h
-rw-r--r-- 1 munakata munakata 13304 Feb 6 13:49 r600_pci_ids.h
-rw-r--r-- 1 munakata munakata 2839 Feb 6 13:49 radeon_bo.h
-rw-r--r-- 1 munakata munakata 1991 Feb 6 13:49 radeon_bo_gem.h
-rw-r--r-- 1 munakata munakata 1673 Feb 6 13:49 radeon_bo_int.h
-rw-r--r-- 1 munakata munakata 5121 Feb 6 13:49 radeon_cs.h
-rw-r--r-- 1 munakata munakata 1601 Feb 6 13:49 radeon_cs_gem.h
-rw-r--r-- 1 munakata munakata 2173 Feb 6 13:49 radeon_cs_int.h
-rw-r--r-- 1 munakata munakata 36015 Feb 6 13:49 radeon_drm.h
-rw-r--r-- 1 munakata munakata 5876 Feb 6 13:49 radeon_surface.h
-rw-r--r-- 1 munakata munakata 7054 Feb 6 13:49 savage_drm.h
-rw-r--r-- 1 munakata munakata 2534 Feb 6 13:49 sis_drm.h
-rw-r--r-- 1 munakata munakata 8291 Feb 6 13:49 via_drm.h

---

14/46  Hisao Munakata  Creating Eco-System for R-Car LCB
Easy solution: fork to create original kernel, but...

- fork to create original kernel, but...
- cannot maintain
- optimized for specific device but very hard to sync with future kernel migration
Why we need eco-system specially for embedded Linux?

What we are doing for Renesas R-Car Linux development

embedded Linux eco-system

conclusion

Background
PC variant vs. embedded

Why we need eco-system for embedded Linux

Linux eco-system = the power of collective wisdom

common ground = sharable board
- reasonably low-cost
- anyone can buy
- public document, BSP

To share embedded Linux experience, eco-system driven by the cheap board is the key
What we are doing for Renesas R-Car Linux development
Why we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system
conclusion

Renesas R-Car series SoC: scalable design concept

"R-Car" is the nickname for Renesas’ lineup of system-on-chips (SoCs) for car information systems.

http://www.renesas.com/applications/automotive/cis/cis_highend/index.jsp
Why we need eco-system specially for embedded Linux?

What we are doing for Renesas R-Car Linux development
embedded Linux eco-system

R-Car SoC intro
R-Car Linux BSP development

BSP is source level compatible for H2, M2 and E2

R-Car E2
- A7 x2
- PowerVR SGX540
- 2 display support

R-Car M2
- A15 x2
- PowerVR SGX544MP2
- 2 display support

R-Car H2
- A15 x4 + CA7 x4
- PowerVR G6400
- 3 display support

R-Car gen2 Linux BSP support E2, M2 and H2 with same code

R-Car gen2 devices are feature and performance scalable design, we develop single Linux BSP can support all variant so that customer can use same code for all.
Why do we need an eco-system especially for embedded Linux?

What are we doing for Renesas R-Car Linux development?

Embedded Linux eco-system

Conclusion

R-Car SoC intro

R-Car Linux BSP development

Kernel is binary level compatible across the boards

Board A

R-Car M2

Kernel for board A

Fix

Common kernel for R-Car M2 platform (binary compatible)

DTS for board A

Board A

R-Car M2

Board B

R-Car M2

Kernel for board B

DTB support

Fix for A is reflect to all boards

Board C

R-Car M2

Kernel for board C

DTS for board C

Board C

R-Car M2
# Renesas R-Car M2 SOC (for mid-range system) 1/3

## Product Specifications of the R-Car M2

<table>
<thead>
<tr>
<th>Item</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product number</td>
<td>R8A7791</td>
</tr>
<tr>
<td>Power supply voltage</td>
<td>3.3/1.8 V (IO), 1.5/1.35 V (DDR3), 1.03 V (Core)</td>
</tr>
<tr>
<td>CPU core</td>
<td>ARM® Cortex™-A15 Dual</td>
</tr>
<tr>
<td></td>
<td>SH-4A core (device option)</td>
</tr>
<tr>
<td>Maximum operating frequency</td>
<td>1.5 GHz</td>
</tr>
<tr>
<td>Processing performance</td>
<td>10500 DMIPS</td>
</tr>
<tr>
<td></td>
<td>1720 DMIPS</td>
</tr>
<tr>
<td>Cache memory</td>
<td>L1 Instruction cache: 32 KB</td>
</tr>
<tr>
<td></td>
<td>L1 Operand cache: 32 KB</td>
</tr>
<tr>
<td></td>
<td>L2 Cache: 2 MB</td>
</tr>
<tr>
<td></td>
<td>Instruction cache: 32 KB</td>
</tr>
<tr>
<td></td>
<td>Operand cache: 32 KB</td>
</tr>
<tr>
<td>External memory</td>
<td>DDR3-SDRAM</td>
</tr>
<tr>
<td></td>
<td>Maximum operating frequency: 800 MHz</td>
</tr>
<tr>
<td></td>
<td>Data bus width: 32 bits × 2 ch (6.4 GB/s × 2)</td>
</tr>
<tr>
<td>Expansion bus</td>
<td>Flash ROM and SRAM,</td>
</tr>
<tr>
<td></td>
<td>Data bus width: 8 or 16 bits</td>
</tr>
<tr>
<td></td>
<td>PCI Express2.0 (1 lane)</td>
</tr>
<tr>
<td>Graphics</td>
<td>PowerVR SGX 544MP2 (3D)</td>
</tr>
<tr>
<td></td>
<td>Renesas graphics processor (2D)</td>
</tr>
</tbody>
</table>

### Product Specifications of the R-Car M2

<table>
<thead>
<tr>
<th>Video</th>
<th>Audio</th>
<th>Storage Interface</th>
<th>In-car network and automotive peripherals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Out × 2 ch (1 ch: LVDS, 1 ch: RGB888)</td>
<td>Audio DSP</td>
<td>USB 3.0 host interface × 1 port (wPHY)</td>
<td>Media local bus (MLB) Interface × 1 ch (6-pin / 3-pin interface selectable)</td>
</tr>
<tr>
<td>Video Input × 3 ch</td>
<td>Sampling rate converter × 10 ch</td>
<td>USB 2.0 host interface × 2 ports (wPHY)</td>
<td>CAN Interface × 2 ch</td>
</tr>
<tr>
<td>Video codec module (H.264/AVC, MPEG-2/4, VC-1)</td>
<td>Serial sound interface × 10 ch</td>
<td>SD host interface × 3 ch (SDXC, UHS-I)</td>
<td>IEBus™ Interface</td>
</tr>
<tr>
<td>IP conversion module</td>
<td>MOST DTCP</td>
<td>Multimedia card interface × 1 ch</td>
<td>GPS baseband module (Galileo, GLONASS) (device option)</td>
</tr>
<tr>
<td>JPEG accelerator</td>
<td></td>
<td>Serial ATA interface × 2 ch</td>
<td>Ethernet controller AVB (IEEE802.1BA, 802.1AS, 802.1Qav and IEEE1722, GMII/MII, without PHY)</td>
</tr>
<tr>
<td>TS Interface × 1 ch</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Video image processing (color conversion, image expansion, reduction, filter processing)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distortion compensation module (image renderer) × 1 ch</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

--


---

Renesas R-CarM2 SOC (for mid-range system) 2/3

Why we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system

Conclusion

R-Car SoC intro
R-Car Linux BSP development

### Product Specifications of the R-Car M2

<table>
<thead>
<tr>
<th>Security</th>
<th>Crypto engine (AES, DES, Hash, RSA)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Secure RAM</td>
</tr>
<tr>
<td>Other peripherals</td>
<td>DMA controller</td>
</tr>
<tr>
<td></td>
<td>LBSC DMAC: 3 ch / SYS-DMAC : 30 ch /</td>
</tr>
<tr>
<td></td>
<td>RT-DMAC: 3 ch / Audio-DMAC: 26 ch /</td>
</tr>
<tr>
<td></td>
<td>Audio (peripheral)-DMAC: 29 ch</td>
</tr>
<tr>
<td></td>
<td>32 bit timer × 12 ch</td>
</tr>
<tr>
<td></td>
<td>PWM timer × 7 ch</td>
</tr>
<tr>
<td></td>
<td>I²C bus interface × 9 ch</td>
</tr>
<tr>
<td></td>
<td>Serial communication interface (SCIF) × 18 ch</td>
</tr>
<tr>
<td></td>
<td>Quad serial peripheral interface (QSPI) × 1 ch (for boot)</td>
</tr>
<tr>
<td></td>
<td>Clock-synchronized serial interface (MSIOF) × 3 ch (SPI/IIS)</td>
</tr>
<tr>
<td></td>
<td>Ethernet AVB controller (IEEE802.1BA/802.1AS/802.1Qav/IEEE1722, GMII/MII, without PHY)</td>
</tr>
<tr>
<td></td>
<td>Ethernet controller (IEEE802.3u, RMII, without PHY)</td>
</tr>
<tr>
<td></td>
<td>Interrupt controller (INTC)</td>
</tr>
<tr>
<td></td>
<td>Clock generator (CPG) with built-in PLL</td>
</tr>
<tr>
<td></td>
<td>On-chip debugger interface</td>
</tr>
<tr>
<td>Low power mode</td>
<td>Dynamic Power Shutdown (CPU core, 3D, IMP)</td>
</tr>
<tr>
<td></td>
<td>AVS and DVFS function</td>
</tr>
<tr>
<td></td>
<td>DDR-SDRAM power supply backup mode</td>
</tr>
<tr>
<td>Package</td>
<td>831-pin Flip Chip BGA (27 mm × 27 mm)</td>
</tr>
</tbody>
</table>

Why we need eco-system specially for embedded Linux?

What we are doing for Renesas R-Car Linux development

Embedded Linux eco-system

Conclusion

R-Car SoC intro

R-Car Linux BSP development

R-Car gen2 Linux BSP whole development process

1. Renesas upstream development team continuous code submission to the community ML

2. LTSI development (patch backport to 3.10)

3. We adopt LTSI 3.10 kernel as a baseline for R-Car gen2 BSP

4. R-Car BSP v.1.0

5. R-Car gen2 Linux BSP early drop

Creating Eco-System for R-Car LCB
Why we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system
conclusion

(1) Upstream _lf_pub_whowriteslinux2015.pdf_

**1. Renesas upstream development team** continuous code submission to the community ML

**2. LTSI development** (patch backport to 3.10)

**286 patches**

**3. LTSI-3.10**

We adopt LTSI 3.10 kernel as a baseline for R-Car gen2 BSP

**4. renesas backport public git**

BSP update information
- new fixes
- enhance

**5. R-Car BSP v.0.8**

**5. R-Car BSP v.0.9**

**5. R-Car BSP v.1.0**

**5. R-Car BSP early drop**

**5. R-Car BSP v.0.8**
Renesas adopts *upstream first strategy*

<table>
<thead>
<tr>
<th>pros</th>
<th>cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>- clean code (reviewed by the community)</td>
<td>- take time (roughly 6 month)</td>
</tr>
<tr>
<td>- coordinated with existing code</td>
<td>- might need iterative approach (bit by bit)</td>
</tr>
<tr>
<td>- merged to the upstream code</td>
<td>- might need code adjustment</td>
</tr>
<tr>
<td>- no need to keep in-house code</td>
<td></td>
</tr>
</tbody>
</table>
Why we need eco-system specially for embedded Linux?

What we are doing for Renesas R-Car Linux development

embedded Linux eco-system

conclusion

R-Car SoC intro

R-Car Linux BSP development

iterative approach to minimize code fragmentation

effort to minimize device uniqueness to avoid excessive code fragmentation

add device uniqueness

R-Car Linux BSP development
Why do we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system

Conclusion

R-Car SoC intro
R-Car Linux BSP development

Result of 2014: If_pub_whowriteslinux2015.pdf

<table>
<thead>
<tr>
<th>Company</th>
<th>Changes</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>11,968</td>
<td>12.4%</td>
</tr>
<tr>
<td>Intel</td>
<td>10,108</td>
<td>10.5%</td>
</tr>
<tr>
<td>Red Hat</td>
<td>8,078</td>
<td>8.4%</td>
</tr>
<tr>
<td>Linaro</td>
<td>5,415</td>
<td>5.6%</td>
</tr>
<tr>
<td>Samsung</td>
<td>4,290</td>
<td>4.4%</td>
</tr>
<tr>
<td>Unknown</td>
<td>3,842</td>
<td>4.0%</td>
</tr>
<tr>
<td>IBM</td>
<td>3,081</td>
<td>3.2%</td>
</tr>
<tr>
<td>SUSE</td>
<td>2,890</td>
<td>3.0%</td>
</tr>
<tr>
<td>Consultants</td>
<td>2,451</td>
<td>2.5%</td>
</tr>
<tr>
<td>Texas Instruments</td>
<td>2,269</td>
<td>2.4%</td>
</tr>
<tr>
<td>Vision Engraving Systems</td>
<td>2,089</td>
<td>2.2%</td>
</tr>
<tr>
<td>Google</td>
<td>2,048</td>
<td>2.1%</td>
</tr>
<tr>
<td>Renesas Electronics</td>
<td>2,004</td>
<td>2.1%</td>
</tr>
<tr>
<td>Freescale</td>
<td>1,690</td>
<td>1.8%</td>
</tr>
<tr>
<td>Free Electrons</td>
<td>1,463</td>
<td>1.5%</td>
</tr>
<tr>
<td>FOSS Outreach Program for Women</td>
<td>1,418</td>
<td>1.5%</td>
</tr>
<tr>
<td>Oracle</td>
<td>1,166</td>
<td>1.2%</td>
</tr>
<tr>
<td>AMD</td>
<td>1,109</td>
<td>1.1%</td>
</tr>
<tr>
<td>NVidia</td>
<td>1,078</td>
<td>1.1%</td>
</tr>
<tr>
<td>Broadcom</td>
<td>1,001</td>
<td>1.0%</td>
</tr>
<tr>
<td>Huawei Technologies</td>
<td>971</td>
<td>1.0%</td>
</tr>
<tr>
<td>ARM</td>
<td>788</td>
<td>0.8%</td>
</tr>
<tr>
<td>Pengutronix</td>
<td>763</td>
<td>0.8%</td>
</tr>
</tbody>
</table>
Why we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system

**conclusion**

R-Car SoC intro
R-Car Linux BSP development

(2) LTSI development process

1. Renesas upstream development team continuous code submission to the community ML

2. LTSI development (patch backport to 3.10)

3. renesas backport public git
We adopt LTSI 3.10 kernel as a baseline for R-Car gen2 BSP

4. BSP update information
- new fixes
- enhance

5. LTSI-3.4 based early release
R-Car BSP early drop

R-Car BSP v.0.8
R-Car BSP v.0.9
R-Car BSP v.1.0

LTSI-3.10
286 patches
Why we need eco-system specially for embedded Linux?  
What we are doing for Renesas R-Car Linux development embedded Linux eco-system

conclusion

(2) LTSI = cutting edge device on the solid software

<table>
<thead>
<tr>
<th>version</th>
<th>fixes</th>
</tr>
</thead>
<tbody>
<tr>
<td>v3.3 -&gt; v3.3.8</td>
<td>698</td>
</tr>
<tr>
<td>v3.4 -&gt; v3.4.95</td>
<td>4,506</td>
</tr>
<tr>
<td>v3.5 -&gt; v3.5.7</td>
<td>816</td>
</tr>
<tr>
<td>v3.6 -&gt; v3.6.9</td>
<td>676</td>
</tr>
<tr>
<td>v3.7 -&gt; v3.7.10</td>
<td>718</td>
</tr>
<tr>
<td>v3.8 -&gt; v3.8.13</td>
<td>996</td>
</tr>
<tr>
<td>v3.9 -&gt; v3.9.11</td>
<td>746</td>
</tr>
<tr>
<td>v3.10 -&gt; v3.10.69</td>
<td>4,175</td>
</tr>
<tr>
<td>v3.11 -&gt; v3.11.10</td>
<td>677</td>
</tr>
<tr>
<td>v3.12 -&gt; v3.12.38</td>
<td>4,262</td>
</tr>
<tr>
<td>v3.13 -&gt; v3.13.11</td>
<td>903</td>
</tr>
<tr>
<td>v3.14 -&gt; v3.14.33</td>
<td>2,786</td>
</tr>
<tr>
<td>v3.15 -&gt; v3.15.10</td>
<td>703</td>
</tr>
<tr>
<td>v3.16 -&gt; v3.16.7</td>
<td>871</td>
</tr>
</tbody>
</table>

Renesas R-CarM2

September 26, 2013
Renesas Electronics Corporation today announced the availability of the R-Car M2 automotive Systems-on-Chip (SoC)

kernel 3.10 development
merge open = April 28 2013
merge close = May 12 2013
release = June 30 2013

R-CarM2 was released after the release of kernel 3.10
Why we need eco-system specially for embedded Linux?

What we are doing for Renesas R-Car Linux development

embedded Linux eco-system

conclusion

R-Car SoC intro

R-Car Linux BSP development

Backported R-carM2 device support to LTSI-3.10

SoC in-house tree

later mainline ..3.12

kernel migration

feature backport from latest kernel

- new device driver
- new platform
- new kernel feature

upstreaming support

3.10-LTS

base

3.10-LTSI

LTSI = LTS + feature backport

not mainlined fix
local enhance
Why we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system

(3) continuous backport after LTSI merge closed

1. Renesas upstream development team continuous code submission to the community ML

2. LTSI development (patch backport to 3.10)

3. We adopt LTSI 3.10 kernel as a baseline for R-Car gen2 BSP

4. BSP update information
   - new fixes
   - enhance

5. LTSI-3.4 based early release

Renesas upstream development team continuous code submission to the community ML

3.10 3.11 3.12 3.13 3.14 3.15 3.16 3.17 3.18 3.19 4.0

LTSI development (patch backport to 3.10)

286 patches

We adopt LTSI 3.10 kernel as a baseline for R-Car gen2 BSP

R-Car BSP early drop
R-Car BSP v.0.8
R-Car BSP v.0.9
R-Car BSP v.1.0

BSP update information
- new fixes
- enhance

Continuous backport after LTSI merge closed
Why do we need an eco-system specially for embedded Linux?

What we are doing for Renesas R-Car Linux development

embedded Linux eco-system

R-Car SoC intro

R-Car Linux BSP development

(3) continuous backport after LTSI merge closed

https://git.kernel.org/cgit/linux/kernel/git/horms/renesas-backport.git/log/?h=bsp/v3.10.31-ltsi/rcar-gen2-1.9.2
Why we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system
conclusion

(4) super-long term security fix adoption

1. Renesas upstream development team continuous code submission to the community ML

2. LTSI development (patch backport to 3.10)

3. We adopt LTSI 3.10 kernel as a baseline for R-Car gen2 BSP

4. BSP update information
   - new fixes
   - enhance

5. LTSl-3.4 based early release

R-Car SoC intro
R-Car Linux BSP development

Hisao Munakata
Creating Eco-System for R-Car LCB
### BSP maintenance: **new bug-fix patch tracking**

**We continue check if new fixes is available**

- automated upstream patch scan from git
- crawling scope is own code or modified code
- **F**: patch severity parsing
  - **H**: crash, dead, freeze, hang, hung, leak, oops, panic
  - **M**: err, error, fix, fixes, fixed, bug
  - **L**: tidyup, typo, warn, warning
- **S**: check if this patch is cc’ed to stable@vger.kernel.org
- **L**: check if this patch is already a part of LTSI kernel
- **B**: check if this patch is send to renesas-backport git
Why we need eco-system specially for embedded Linux?

What we are doing for Renesas R-Car Linux development
embedded Linux eco-system

conclusion

R-Car SoC intro
R-Car Linux BSP development

BSP maintenance: new bug-fix patch tracking

actual scan result example (comparing upstream 3.10..3.19)

<table>
<thead>
<tr>
<th>TITLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>dmaengine: shdma: fix a race condition in __ld_cleanup()</td>
</tr>
<tr>
<td>dmaengine: Remove owning field for driver</td>
</tr>
<tr>
<td>dmaengine: sh: Remove chanent affection</td>
</tr>
<tr>
<td>dma: sh: drop owner assignment from platform_drivers</td>
</tr>
<tr>
<td>dma: rc-ar-audmapp: Fix for no corresponding slave ID</td>
</tr>
<tr>
<td>dmaengine: Remove the context argument to the prep_dma_cyclic_operation</td>
</tr>
<tr>
<td>dmaengine: shdma: Allocate cyclic sg list dynamically</td>
</tr>
<tr>
<td>dmaengine: shdma: Make channel filter ignore unrelated devices</td>
</tr>
<tr>
<td>dmaengine: sh: Rework Kconfig and Makefile</td>
</tr>
<tr>
<td>dmaengine: shdma: Use defines instead of hardcoded numbers</td>
</tr>
<tr>
<td>dma: rc-ar-audmapp: add DT support</td>
</tr>
</tbody>
</table>
Why we need eco-system specially for embedded Linux? What we are doing for Renesas R-Car Linux development embedded Linux eco-system conclusion

(5) local in-house patch adoption and elimination

1. Renesas upstream development team continuous code submission to the community ML

2. LTSI development (patch backport to 3.10)

3. renesas backport public git

4. BSP update information
   - new fixes
   - enhance

5. local in-house patch adoption and elimination
(5) local in-house patch adoption and elimination

- Due to time constraint (=up to six months needed for upstream-first attempt), we still need to manage some in-house patch.

- We start up-porting challenge (in-house code to the upstream flow) to eliminate (at least reduce) in-house code.

- It requires an extra code polish to comply with latest mainline kernel patch adoption criteria. But we believe this is the valuable challenges.
Why we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system
conclusion
Openness (hard & soft) is a key

embedded Linux eco-system
Why we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system

conclusion

Openness (hard & soft) is a key

R-Car low-cost reference board intro. @ eLinux wiki

Hardware Pages

The following hardware pages have LOTS of information on this site:

- BeagleBoard
- MinnowBoard
- OpenPhoenux
- MIPS Creator CI20
- BeagleBone
- Raspberry Pi
- Jetson TK1
- Banana Pi
- BeagleBoneBlack
- UDOO
- Mainline Linux on Tegra
- Renesas R-Car SILK
- BeagleBone Capes
- Improv
- Parallella

http://elinux.org/R-Car/Boards/Porter
Why do we need an ecosystem specifically for embedded Linux? What are we doing for Renesas R-Car Linux development?

**Renesas R-Car M2 Porter board on eLinux**

---

**R-Car/Boards/Porter**

Introduction

This is the official Wiki for Renesas R-Car M2 Porter board. Refer to R-Car page for Renesas R-Car SoC family. Information on Renesas R-Car E2 SILK board is on a separate page.

Hardware

http://elinux.org/R-Car/Boards/Porter
Renesas R-Car E2 SILK board on eLinux

Introduction
This is the official Wiki for Renesas R-Car E2 SILK board.
Refer to R-Car page for Renesas R-Car SoC family. Information on Renesas R-Car M2 Porter board is on a separate page.

Hardware

http://elinux.org/R-Car/Boards/SILK
Renesas R-CarM2/E2 yocto instruction on eLinux

R-Car/Boards/Yocto

This page contains information on building and running Yocto on Renesas R-Car E2 SILK and Renesas R-Car M2 Porter boards.

Yocto versions
Poky-1.6.1 is supported. Specific commit of meta-openembedded is required.

Preliminary steps
2. Install required packages
   Ubuntu and Debian
   ```
   sudo apt-get install gawk wget git-core diffstat unzip texinfo gcc-multilib *
   build-essential chrpath socat libssl1.2-dev xterm
   ```
   Fedora
   ```
   sudo yum install gawk make wget tar bzip2 gzip perl unzip patch *
   diffutils diffstat git cpp gcc gcc-c++ glibc-devel texinfo chrpath *
   cachetl Data-Dumper perl-Text-ParseWords perl-Thread-Queue socat *
   SDL-devel xterm
   ```
   Refer to Yocto Project Quick Start for more information.

Building the BSP for Renesas R-Car SILK and Porter
1. Create a directory and switch to it
   Warning! Yocto builds require a lot of disk space (up to 100 GB). Make sure you have got enough before starting the build.

http://elinux.org/R-Car/Boards/Yocto
Why we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system

conclusion

conclusion
Conclusion

- Introduced why embedded Linux requires BSP and what is the potential problem of too unique embedded feature adoption to the Linux kernel.

- Introduced Renesas R-Car gen2 Linux BSP development process and result.

- Introduced newly opened public web (eLinux) where you can obtain R-Car Linux BSP and related information.
Why we need eco-system specially for embedded Linux?
What we are doing for Renesas R-Car Linux development
embedded Linux eco-system

Resources

- R-CarH2 intro = http://am.renesas.com/applications/automotive/cis/cis_highend/rcar_h2/index.jsp
- R-CarM2 intro = http://am.renesas.com/applications/automotive/cis/cis_highend/rcar_m2/
- R-Car series road map = http://www.renesas.eu/products/soc/assp/automotive/index.jsp
- R-CarM2 Porter board = http://elinux.org/R-Car/Boards/Porter
- R-CarE2 SILK board = http://elinux.org/R-Car/Boards/SILK
- R-Car gen2 public yocto intro = http://elinux.org/R-Car/Boards/Yocto
- R-Car gen2 GFX/MMF evaluation download = http://www.renesas.com/secret/r_car_download/rcar_demoboard.jsp
- e-mail = Hisao Munakata (hisao.munakata vt(at)renesas.com)