



Ultra-Fast Boot Approach Using Suspend to RAM on Linux based IVI System

2017/6/1

Panasonic Corporation

Kazuomi KATO

Agenda

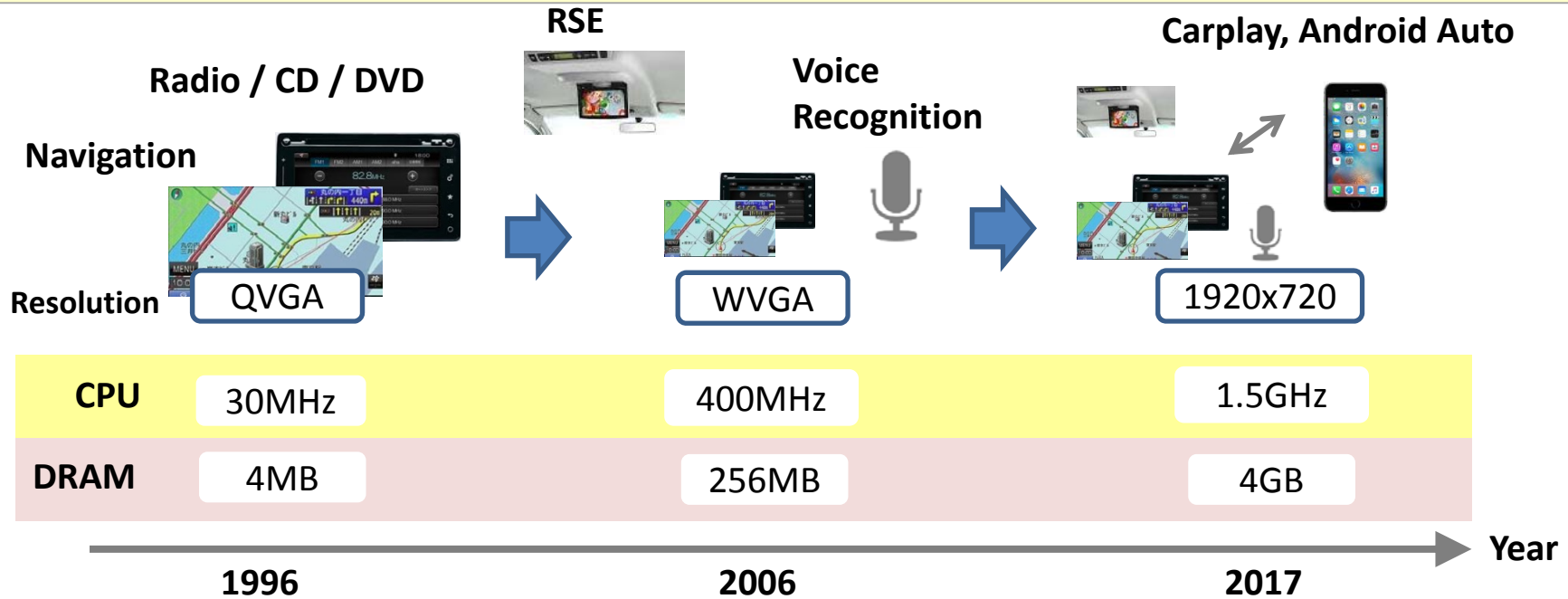
1. Introduction
2. Approaches for optimizing startup time
3. Improvement the time at the **COLD** boot
4. Improvement the time using the **WARM** boot
5. Conclusions



1. Introduction

Background

Increasing the amount of IVI system software.



- Due to the functionality such as smart phone link function, more and more functions and large-scale software are being required.
- Data size is being increased due to the expansion of the display resolution and the realization of multi-screen being equipped.
- At the same time, start-up time and quick response of the system are required by the driver much more.

What benefits for users by reducing startup time?

Obviously necessary to start up the IVI system at high speed.

- 1. Users want to start driving quickly after getting into a car and turning ignition ON.**
- 2. Before driving a car, most users expect the navigation system to guide the route for their destination.**
- 3. When going back to the car, users hope that the last display and music are immediately resumed.**

Issue

Start-up time issues of IVI system:

- 1. How to realize the quick start-up time suitable for users who are used to smart phone. IVI system needs to be activated as soon as the user gets into the car.**
- 2. How to improve the start up time of turning back the functions of music player, car navigation for user operation.**
- 3. To improve the UX and display resolution, the amount size of data are increased. As a results, reading the data from storage costs much longer. So the start-up time is increased too.**
- 4. In the application of vehicle, although large battery is equipped, the issues of dark current has to be considered, when the car not being used for several weeks.**



2. Approaches for optimizing startup time

Methods of optimizing startup time

Approach	Typical method	Pros	Cons
1. Eliminate the bottleneck and improve processing	Profiling and Analysis Improving method based on the profiling and analysis.	Fundamentally improvable.	Know-how is required to specify the bottleneck and improve.
2. Startup is quickly shown as a user's view	CAN wakeup Method of starting the system in advance by CAN signal such as unlocking the door.	It seems to start up fast when getting into a car.	It depends on OEM requirements.
3. Fast return without completely stopping the system	Suspend / Resume Method of saving the current state to DRAM, turning peripheral device power off, waiting and restarting from the state saved at that time.	Fast resume is possible at turning power ON.	The electric power for maintaining DRAM data is consumed.
4. Carry out power OFF adding to No.3 approach	Snapshot boot Method of saving the kernel image of a certain point into the storage and booting up from that point when resuming the system.	Electric power is not consumed as compared with No.3.	If boot image size becomes large, the load time from storage will become a neck.

Our Strategy

At ALS 2016

We introduced the fast boot technique using the PRAMFS* and DRAM backup and improved mainly boot process till application startup.

* Persistent RAM File System

Solution for the next IVI product at this time

- **Development on the next generation SoC.**
- **Improvement of startup including applications.**
- **Eliminating the bottleneck of the base system is significant as a baseline.**
- **Realization of the user experience of fast startup like smartphones.**

[Repeated] Methods of optimizing startup time

Approach	Typical method	Pros	Cons
1. Eliminate the bottleneck and improve processing	Profiling and Analysis Improvement including applications	Fundamentally improvable.	Know-how is required to specify the bottleneck and improve.
2. Startup is quickly shown as a user's view	CAN wakeup Method of starting the system in advance by CAN signal such as unlocking the door.	It seems to start up fast when getting into a car.	It depends on OEM requirements.
3. Fast return without completely stopping the system	Suspend / Resume Move closer to smartphone's UX <small>M D p the state saved at that time.</small>	Fast resume is possible at turning power ON.	The electric power for maintaining DRAM data is consumed.
4. Carry out power OFF adding to No.3 approach	Snapshot boot Method of saving the kernel image of a certain point into the storage and booting up from that point when resuming the system.	Electric power is not consumed as compared with No.3.	If boot image size becomes large, the load time from storage will become a neck.

Reduce the dark current

Target of the startup time

Value	Measurement Case [From a ACC-ON]	Target
The startup time	Displaying a map and playing a music on USB-memory	Under 1 sec
	Displaying rear camera image	
	Displaying a map and turning the radio on	

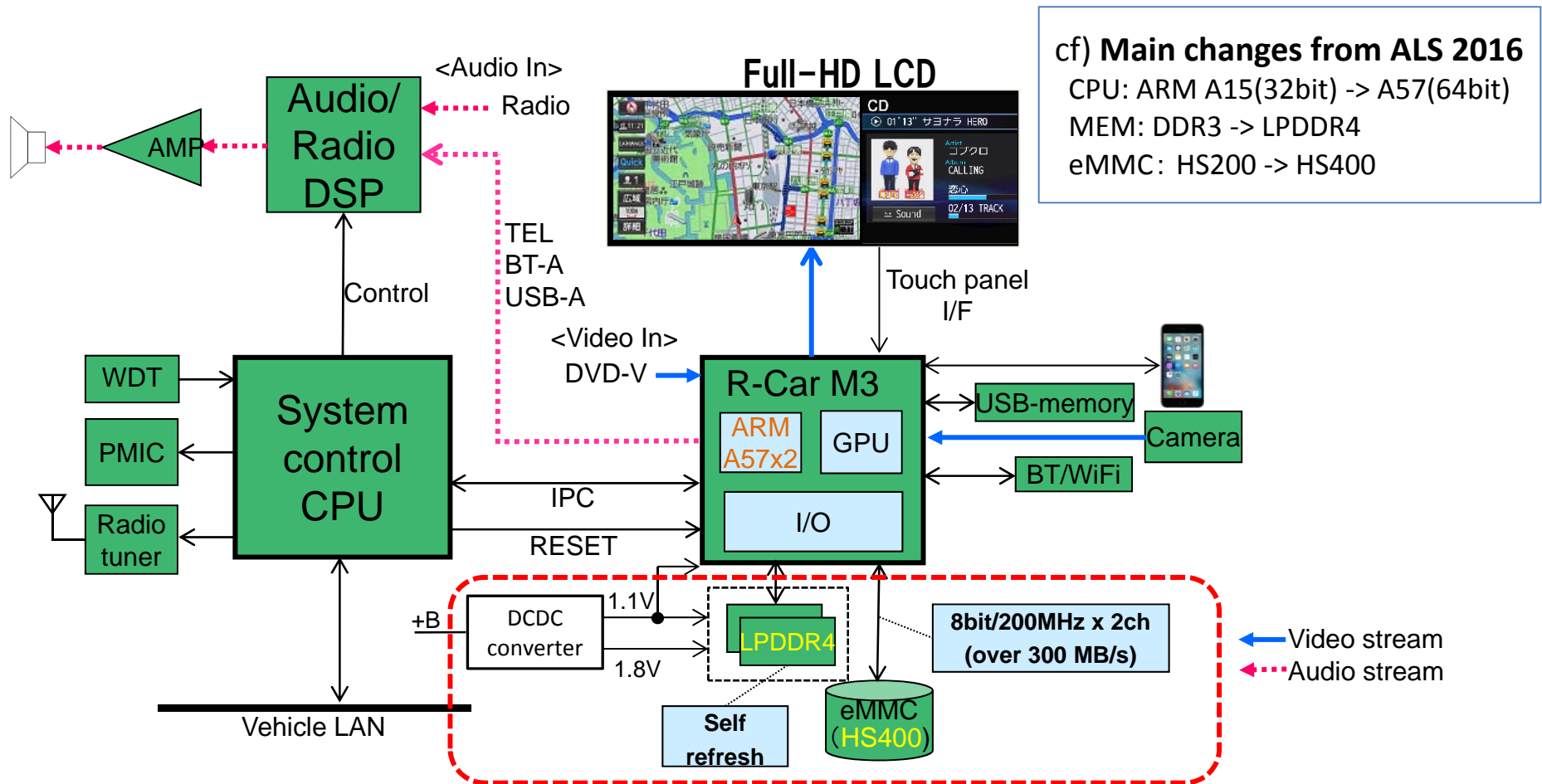
In addition to the above, we try to reduce these values:
- the dark current at the Suspend state



3. Improvement a startup time at the COLD boot

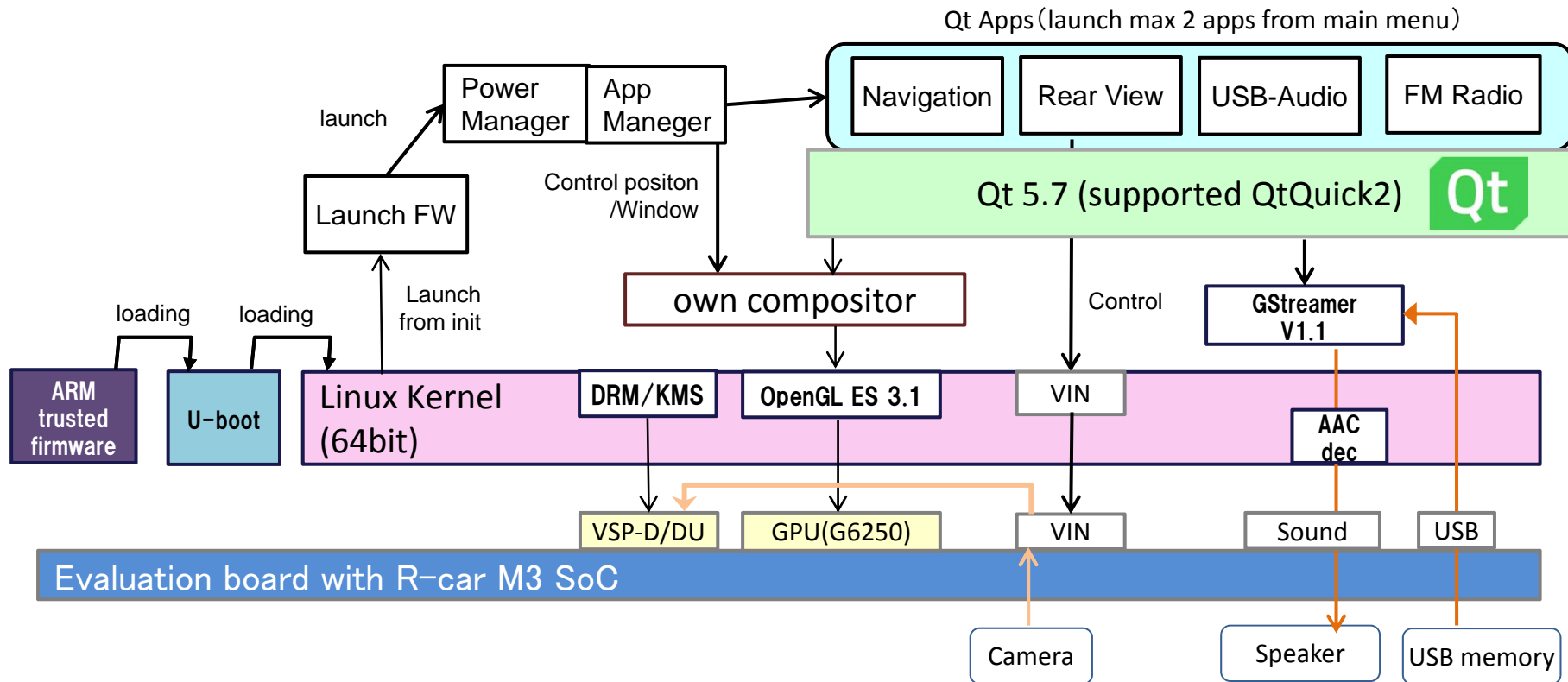
Hardware Block Diagram of the target IVI system

In order to evaluate the startup time, used the following system which is based on Renesas R-Car M3 SoC (ARM Cortex-**A57@1.5GHz** x 2)



Software Environment for Evaluation

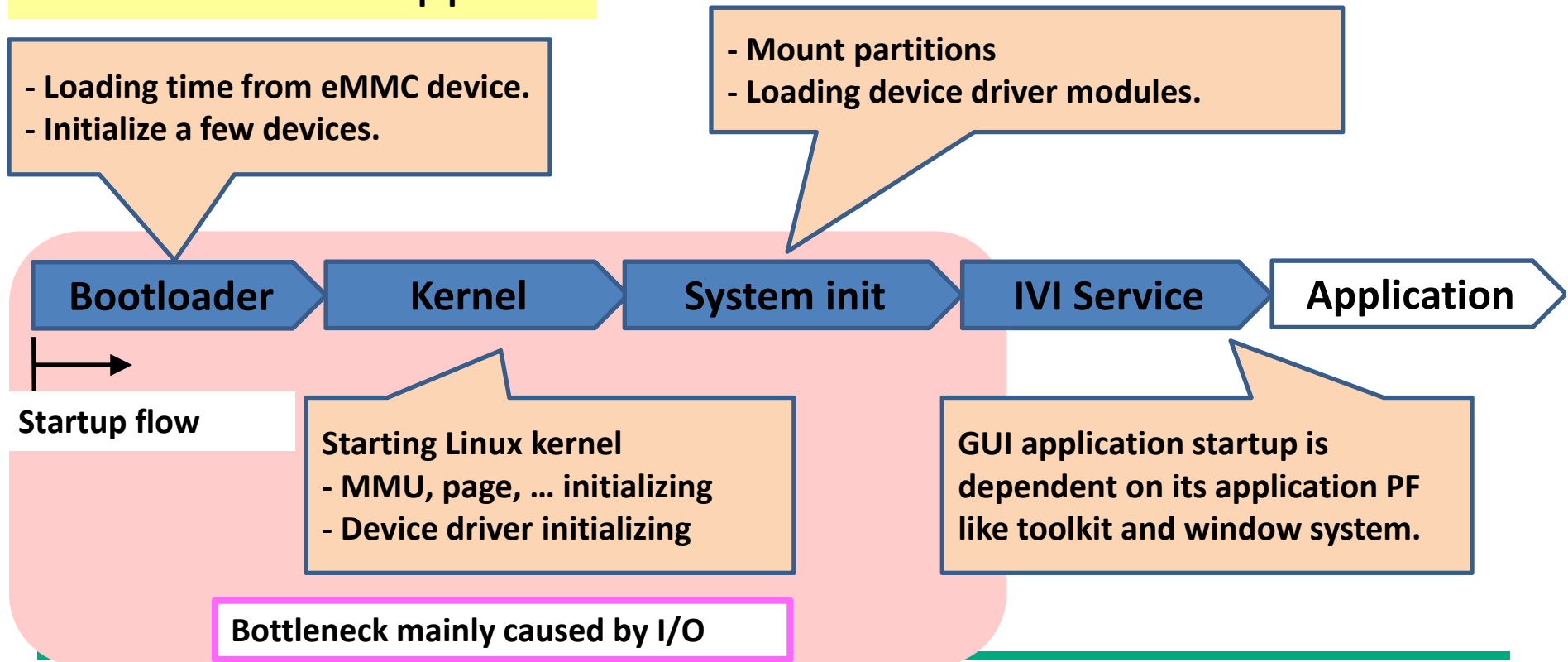
Layers	Detail
Applications	Navigation, Rear View, FM-Radio, USB-Audio (using QtQuick2)
Application PF	Qt 5.7; provided from The Qt company
OS	Linux kernel 4.6.x based on the BSP for R-car M3



Analysis points

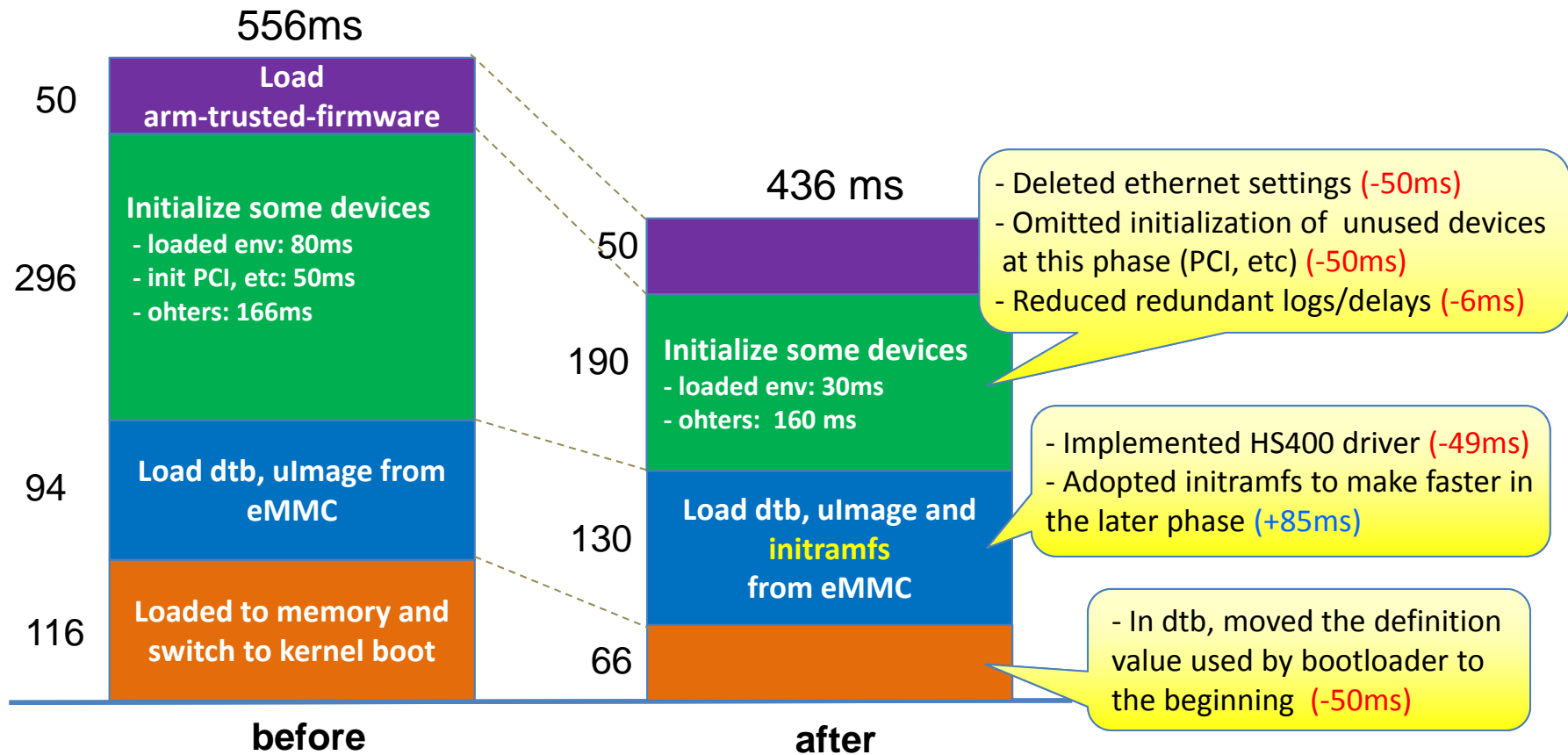
- **Divide the system startup sequence into several phases.**
- **Since the reasons which cause the bottleneck in each phase are different, considering the approach respectively is necessary.**

Overview of each startup phase



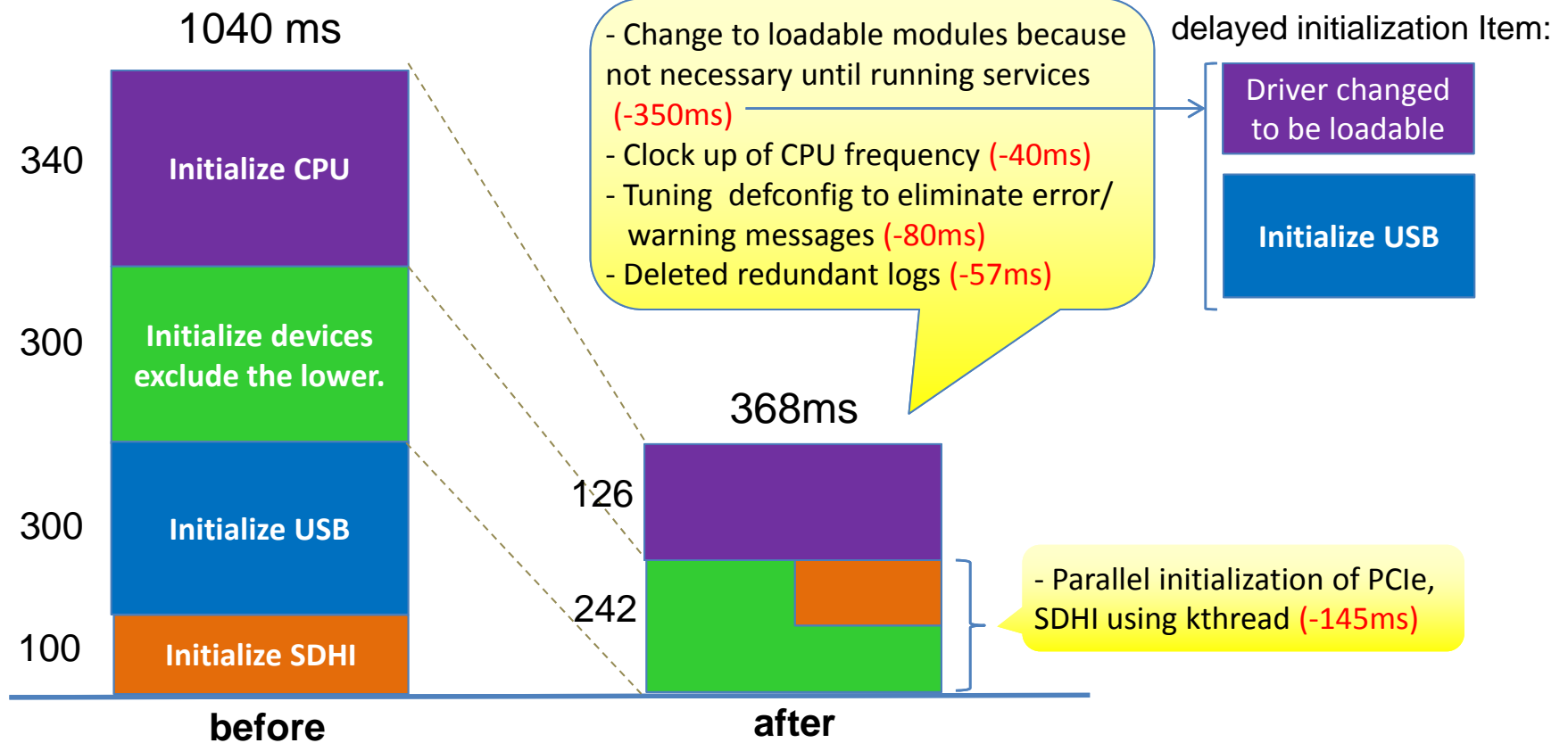
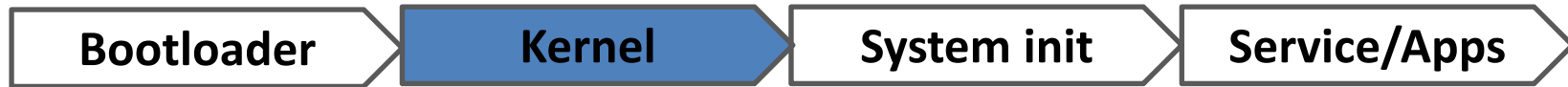
Optimize bootloader

Result: The startup time of bootloader is reduced **120ms by the followings.**



Optimize kernel boot

Result: The startup time of kernel was reduced 672ms by the followings



Optimize system init (introduction)

We use original init called **“system init”**. (Not System V init)

Why originally developed?

- Need the minimum init functionality.
- Standard init process like systemd is not necessary for our product requirements.

Own “system init” for running service process and making device environment for user land.

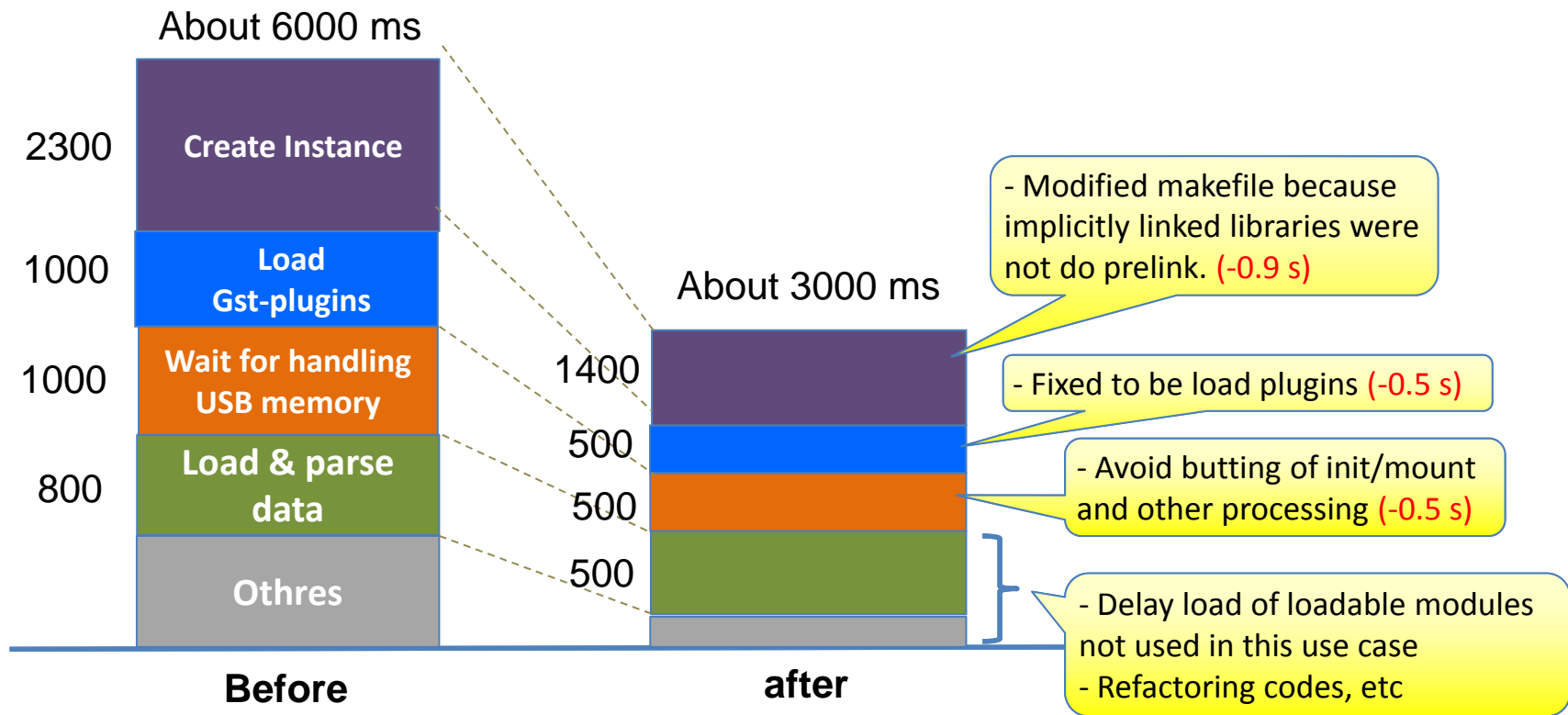
- Making device files and symbolic links
- Mount filesystem
- Activating a service process

Improvement result:

- Skip initializing eMMC because already done in bootloader. (-200msec)
- Reduce filesystem access and parallel execution (-191msec)
- Result: **900msec -> 509msec**

Optimize Service/Apps processing

Result: The startup time is reduced about 3000 msec by the followings



Ex) USB-Audio Apps

Optimize Service/Apps processing (FYI)

We faced some troubles. The causes were as follows.



- Apps launch is slow.
 - No wait for setting a network before launching apps (-3 sec)
- Application display is slow.
 - Omitted a connection verify processing for interfaces because the display unit's specific is fixed. (-400 ms)
- Navigation display is slow.
 - There was some error on prelink process. Updated version of the tool for build was fixed. (-1 sec)
- USB-Audio playback is slow.
 - It was influenced by the size and format of album arts.

Result of the startup time at COLD boot

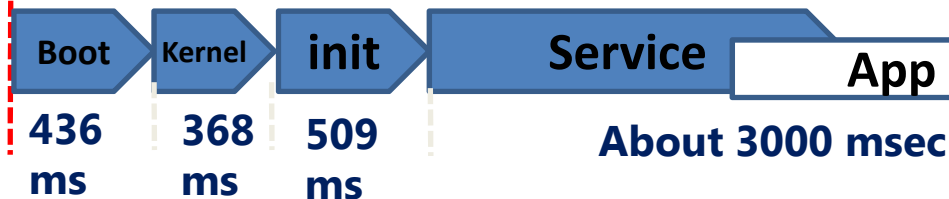
Achieved 4.5 seconds startup of car navigation and USB audio playing.

1. Before (Original boot)



← Various tunings

2. After



Demonstration
on this conference!

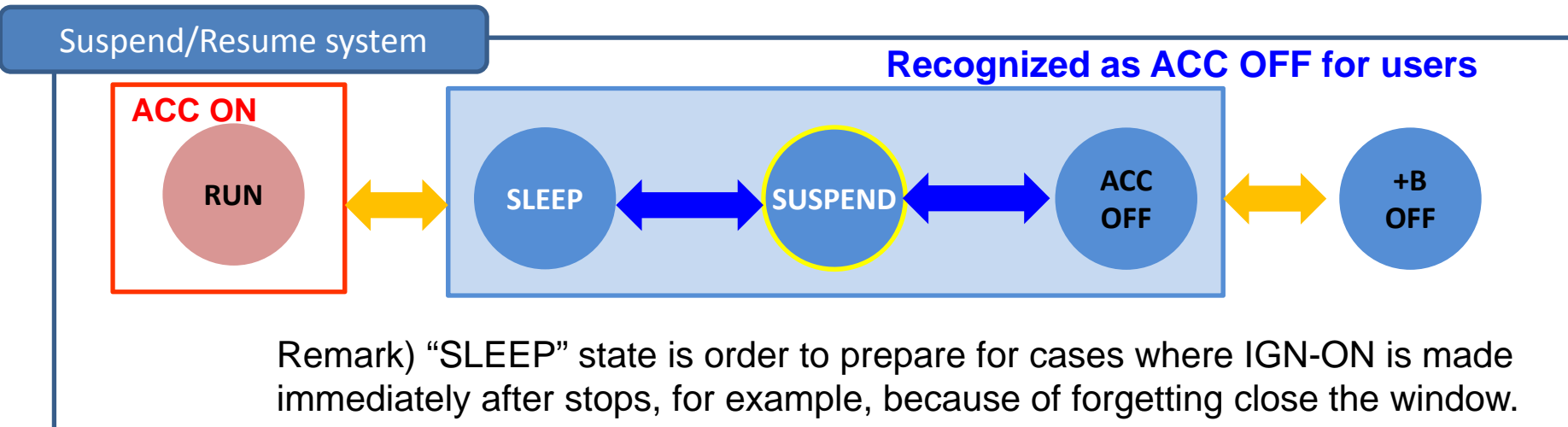
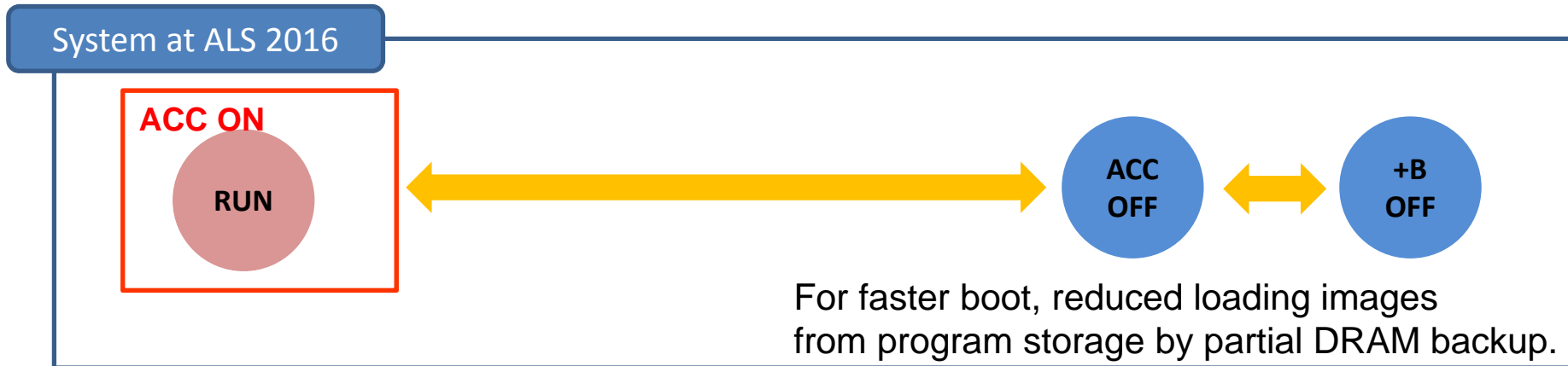
- Display the car navigation map.
- Restart the USB audio playing.

However, in order to achieve ultra fast startup (1 second), another approach is required.

**4. Improvement a startup time
using the **WARM** boot**

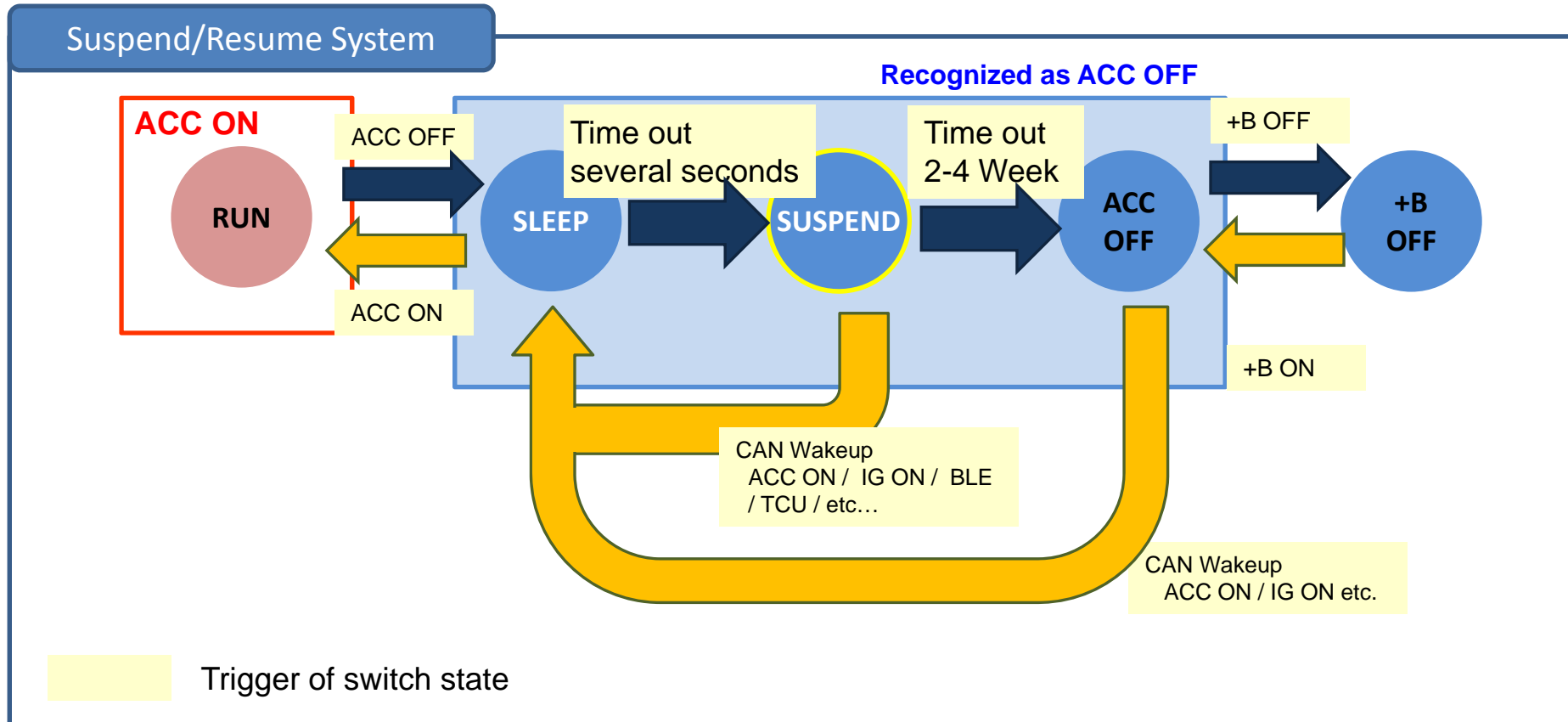
Description of Power States (1)

For ultra fast boot, we added two states before the ACC OFF state

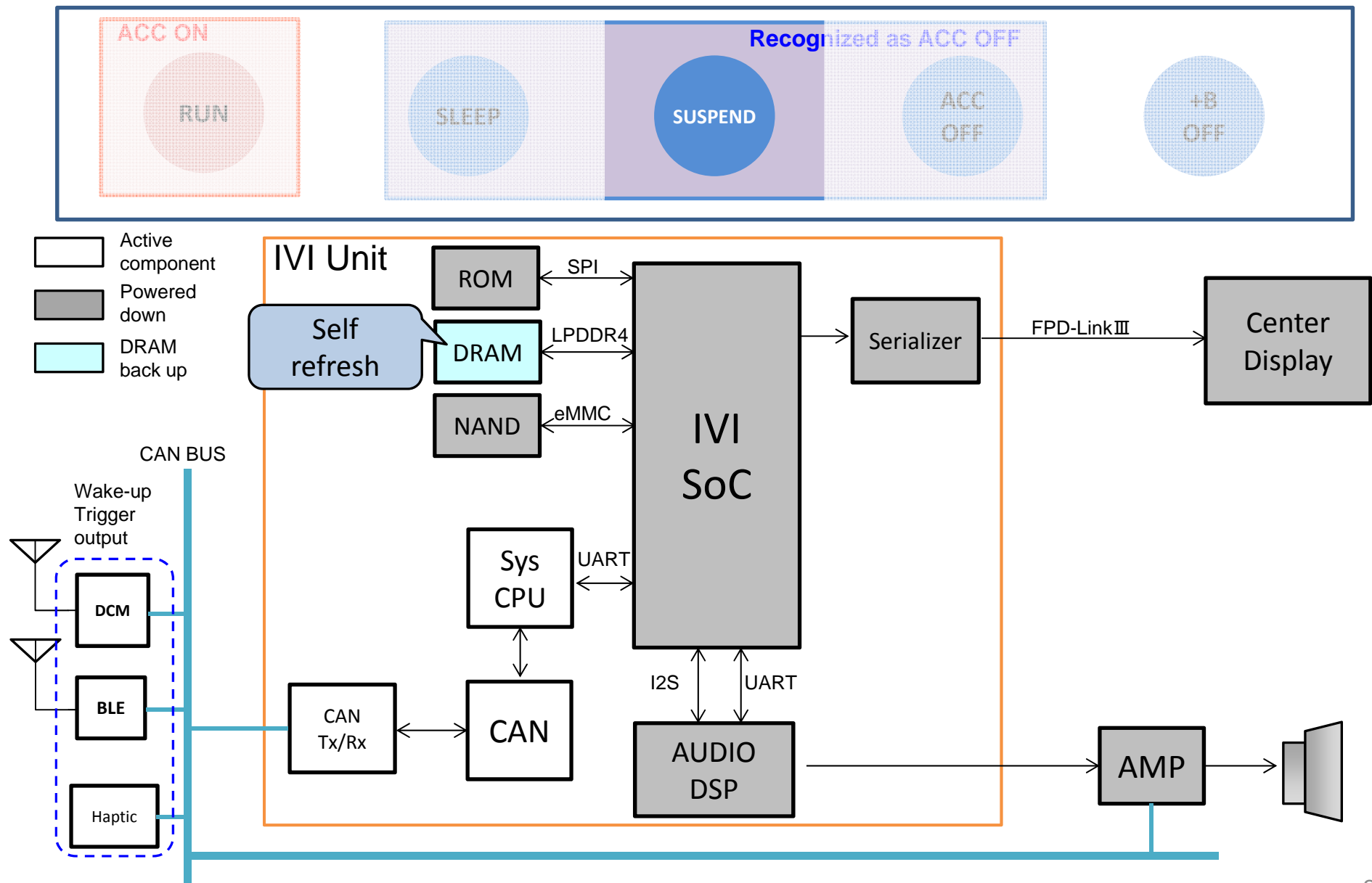


Description of Power States (2)

The transition timing between each power state is as the follows.

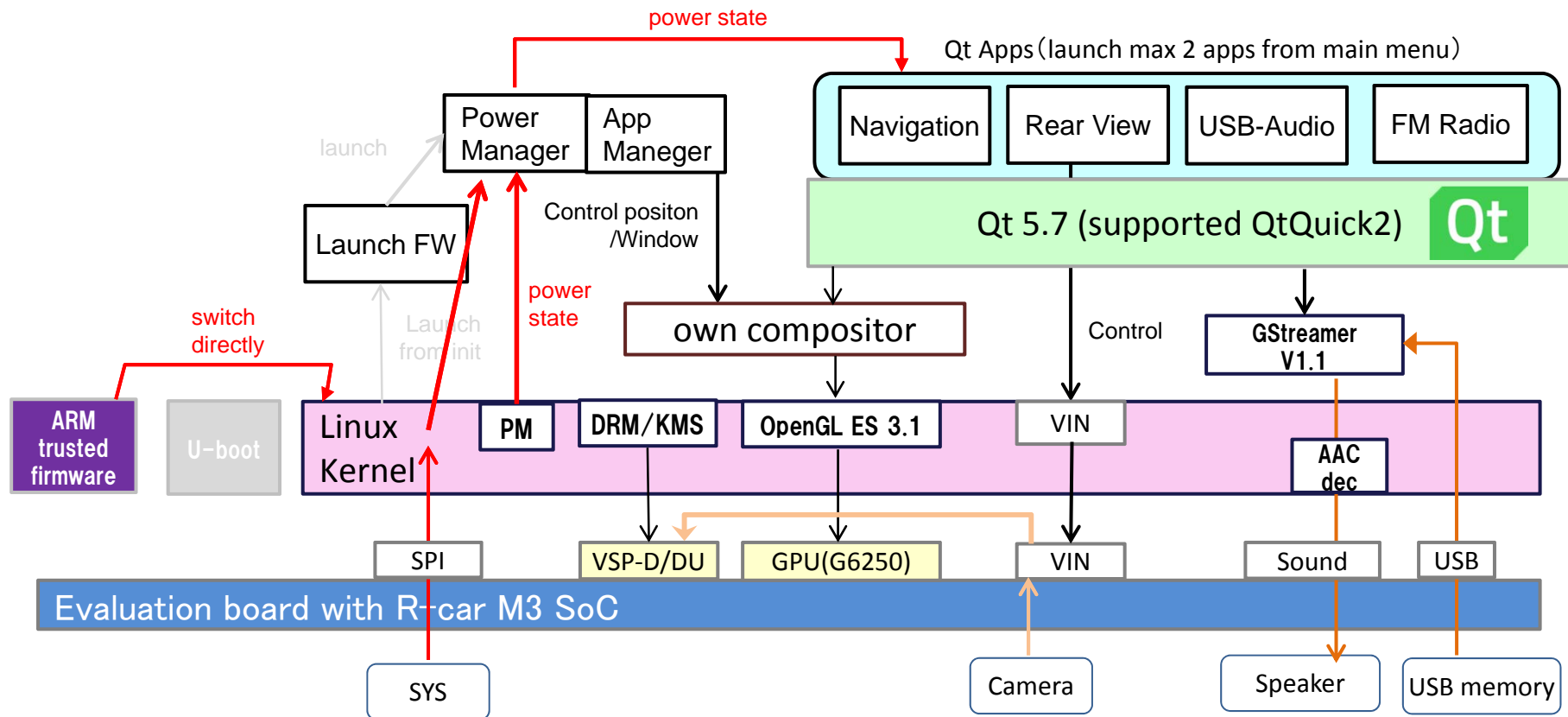


Power supply block at suspend state

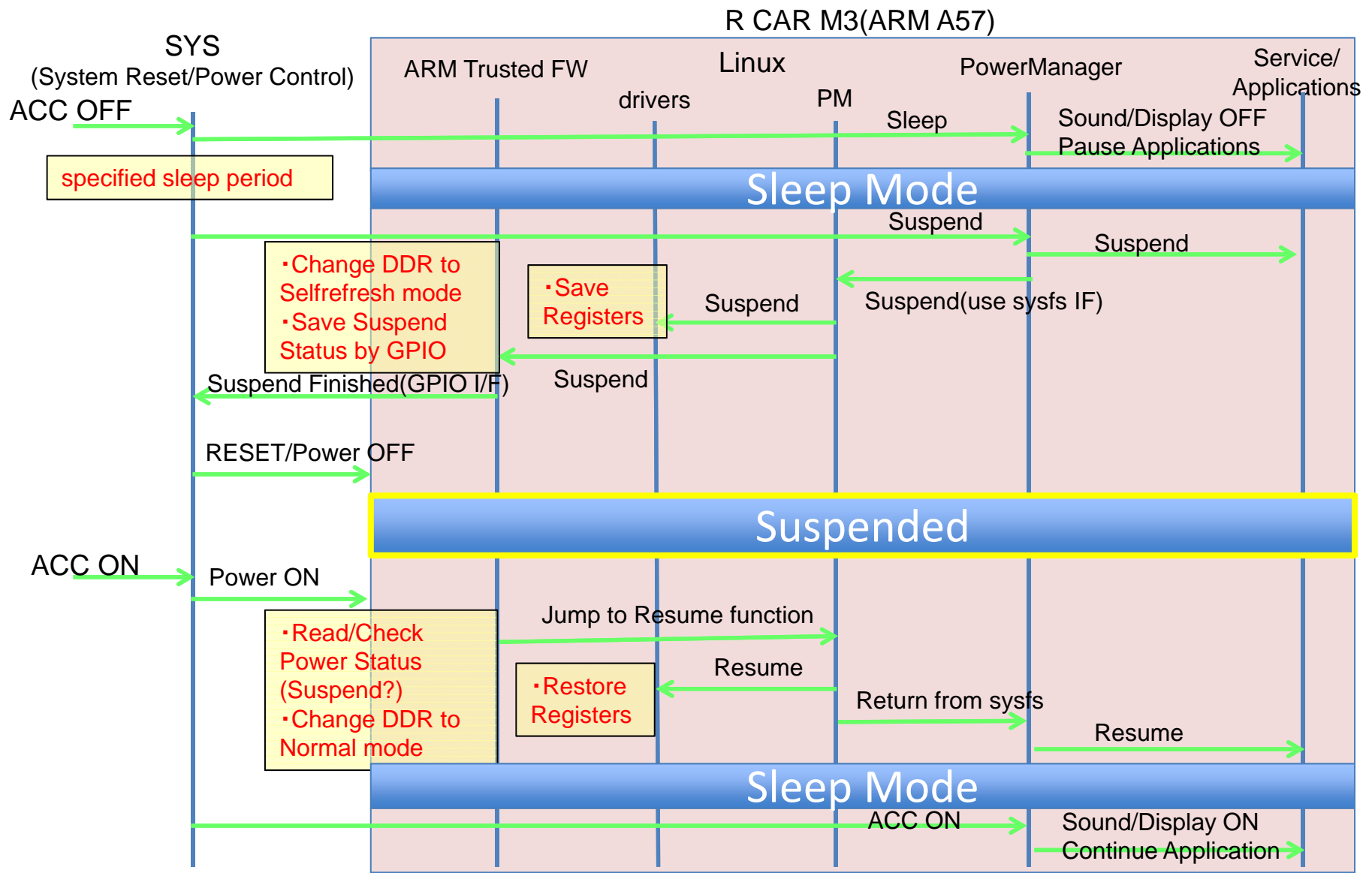


Software block for WARM boot

- ATF jumps to the kernel's resume point by setting a program counter.
- Skip launch & manager process because of already running condition.
- Kernel PM framework and power manager cooperate for restart.
- Apps sus&res methods is called with the notice from the manager.



Sequence for power management (suspend/resume)

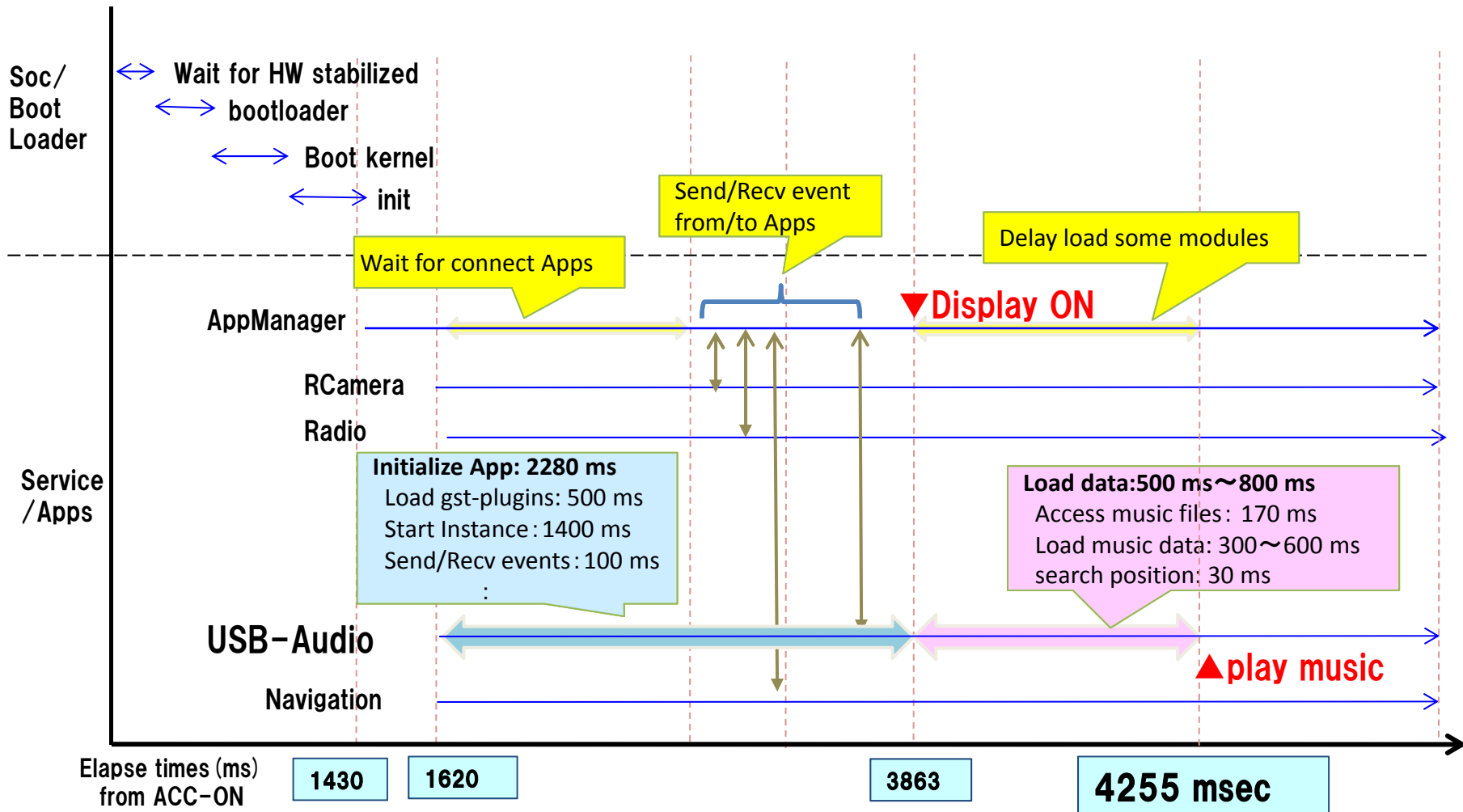


Development for the **WARM** boot.

- **Implementation of Suspend/Resume processing for device driver, application.**
 - In drivers:
 - Implemented pm_suspend/pm_resume function of custom device drivers.
 - In Apps:
 - Implemented suspend and resume process for applications such as car navigation, USB audio and radio.

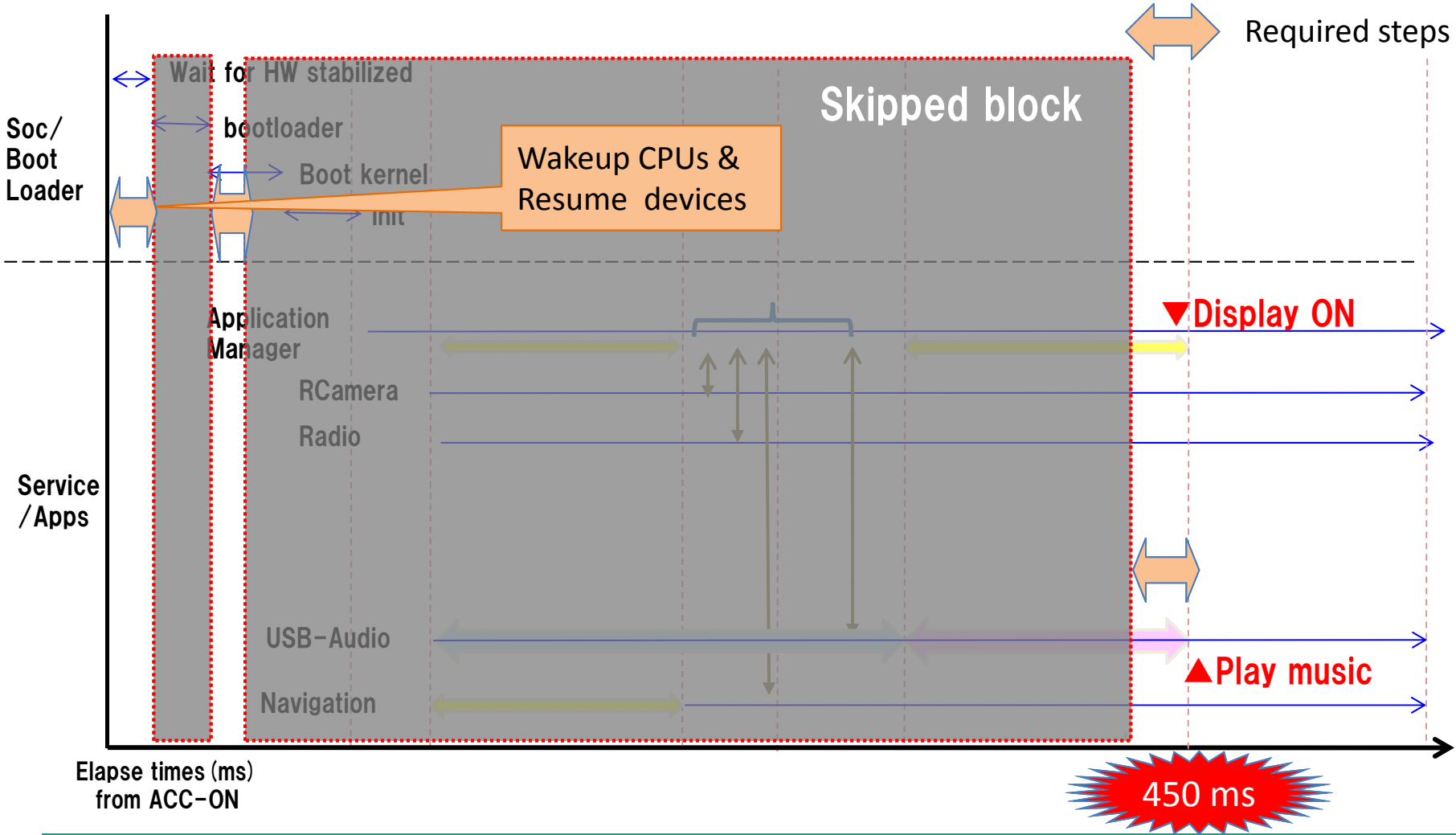
Effect of the startup time (COLD Boot)

The startup time is reduced to less than 5 sec.



Effect of the startup time (WARM boot)

The system can skip almost steps of startup with applying **WARM boot**.



Dark Current during ACC OFF

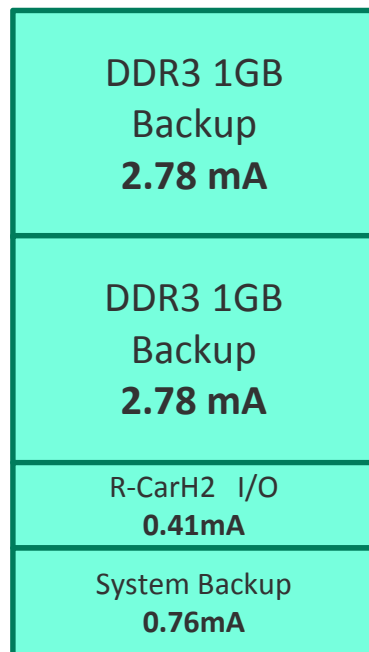
The target dark current : Under 2mA

Current System
R-Car Gen2 + DDR3

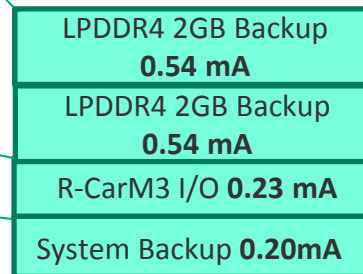


Using Next Generation
R-Car Gen3 + LPDDR4
w/ improvement system

6.73 mA



1.51 mA







- R-Car H2 → R-Car H3
- DDR3 → LPDDR4
5.56 mA → 0.54 mA (@2GB)

- Delete unused circuit block
- Less standby current PMIC

Result of the startup time

Several use case results of startup time are as follows.

Use case		[Target]	COLD boot	WARM boot [1 sec]
Use case1 Map + USB-audio		Display a map & Play the music which user listened recently.	4.50 sec	0.45 sec
Use case2 Rear view Camera		Display a camera image / with predicted course line.	0.63 sec/ 4.36 sec	0.43 sec
Use case3 Map + FM		Display a map & Play the fm-radio	4.34 sec	0.45 sec
Main Menu		Display the menu & Operation possible	4.23 sec	0.40 sec

(Remarks) Each Qt Apps is intermediate code.

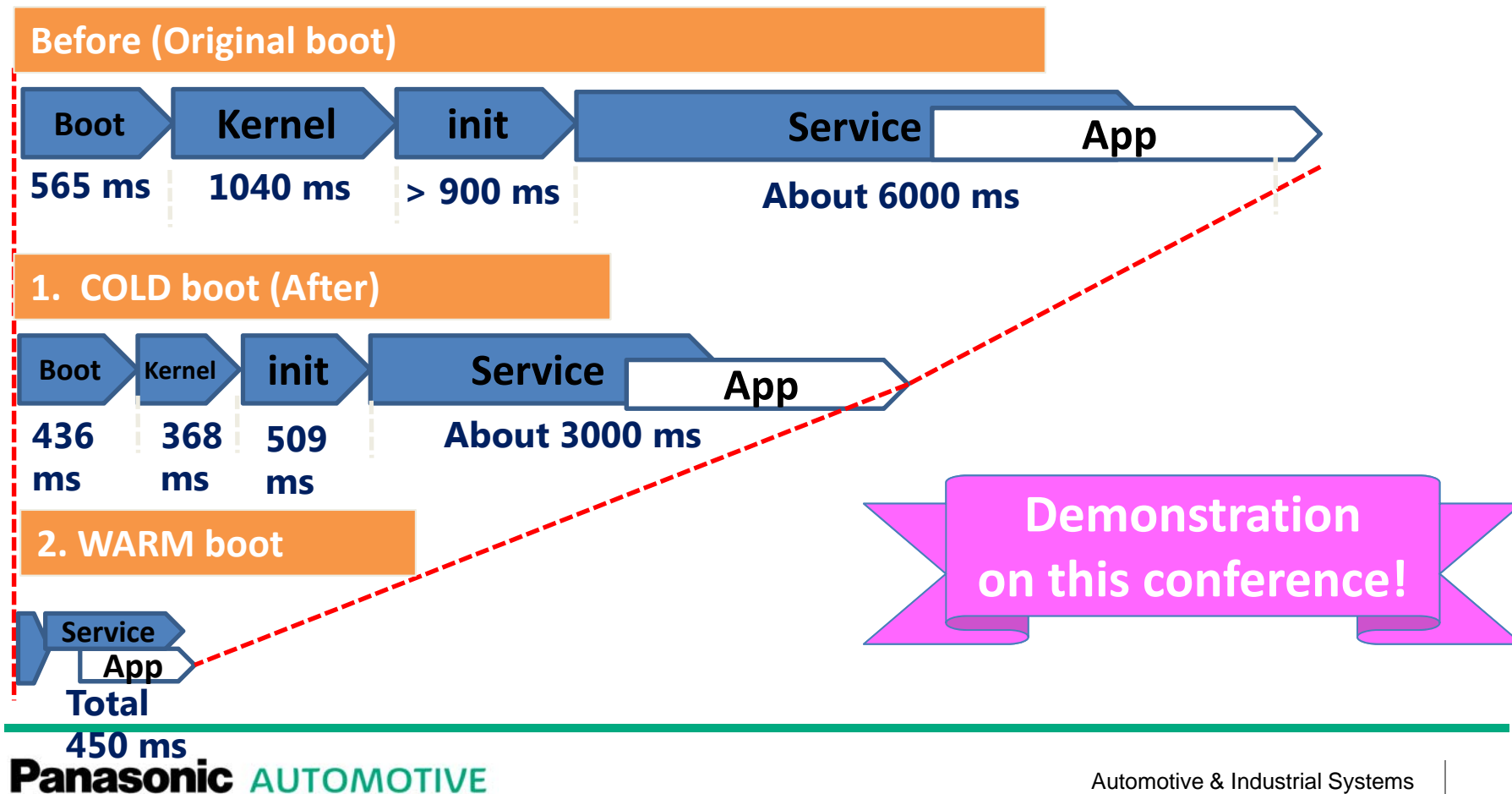
If these are machine cords made by QtQuick compiler, maybe more shorter.



5. Conclusions

Summary

1. Achieved 4.50 seconds for the **COLD** boot time includes a Map & USB-Audio.
 2. Achieved 0.45 seconds for the **WARM** boot time includes a Map & USB-Audio.
And we could improve a dark current is less than 2mA in suspended status.
- These techniques can also use in AGL based PF.



Future plan

- **Further restoration according to the outer information after resume.**
 - Need to adjust time because from the application viewpoint, it appears the time is leap.
 - Need a quick reconnection with outer devices.
- **More reduce of dark current in suspend mode in order to save the battery**
 - Adoption of next-generation low-power devices (e.g. LPDDR4X / LPDDR5)

	Current	Next target
Startup time	4.5 sec / 0.45sec with sample apps	Same level with product apps
Dark current (12V) 4GB	1.54 mA	Under 1.0mA
Drivers (on R-car gen3) ready of Suspend/Resume	Several Device Drivers (cf. exclude Ether, CAN,...)	All Drivers using on the product PF
Apps ready of Suspend/Resume	Sample Apps only	Almost product Apps



Thank you!