Fast Boot in One Second With U-Boot and Linux on Layerscape Platforms

York Sun

Presented by Joseph Byrne

September 2018 | AMF-ENT-T3135

NXP | SECURE CONNECTIONS FOR A SMARTER WORLD

Agenda

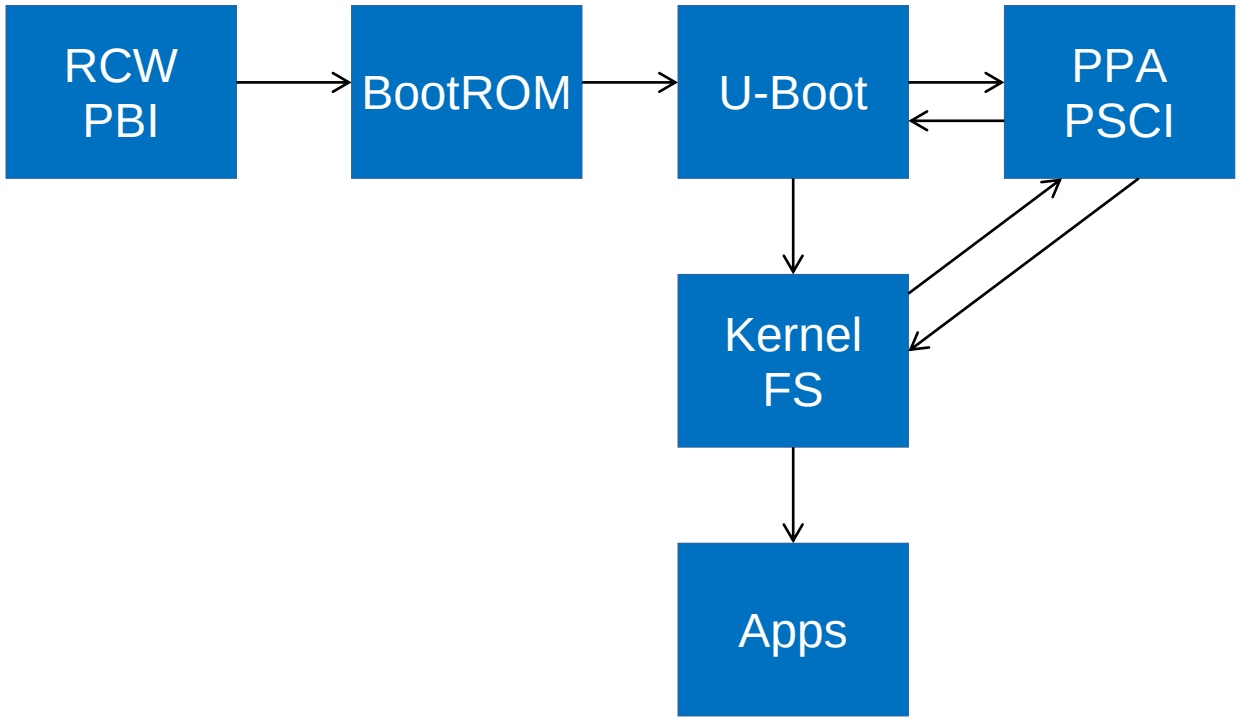
- How long does it take to boot from a cold start to OS?
- What is on the path of booting?
- What can be done to boost booting speed?

Quick Review of Current SDK

- Boot loader
  - From a few seconds to more than 10 seconds
- Prepare OS
  - Another few seconds to more than 10 seconds
- Boot OS
  - A few seconds to more than 10 seconds

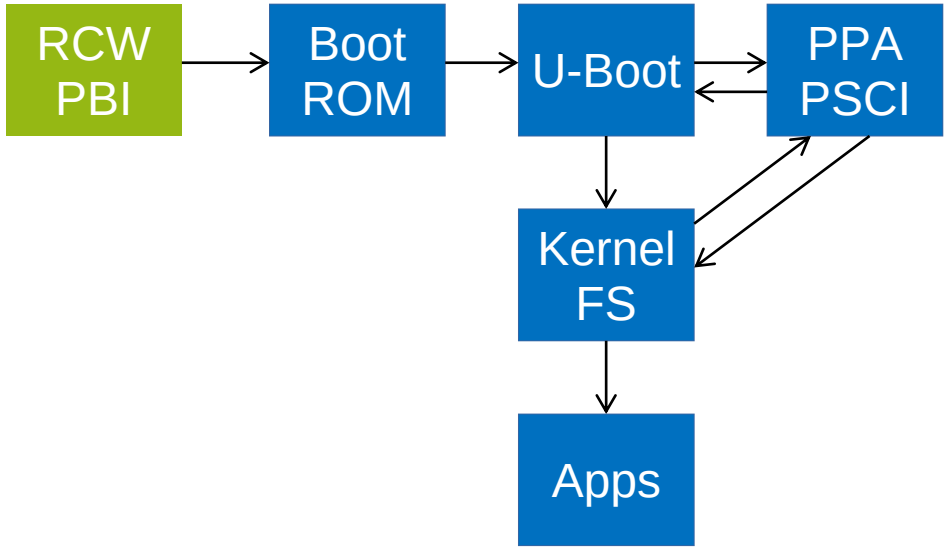
This is not wrong. SDK provides almost everything.

What Is on the Boot Path?



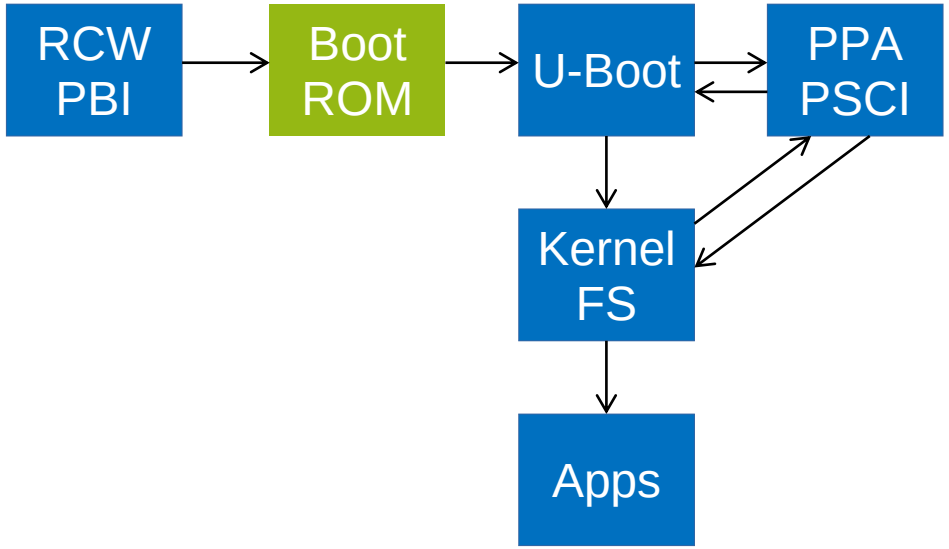
RCW & PBI

- Loading RCW varies on sources
  - NOR, NAND, QSPI, SD, I2C
- PBI payload varies on boot method
  - Simple for NOR and QSPI
  - Extra payload for SPL boot (NAND, SD)
  - Command varies (block copy)



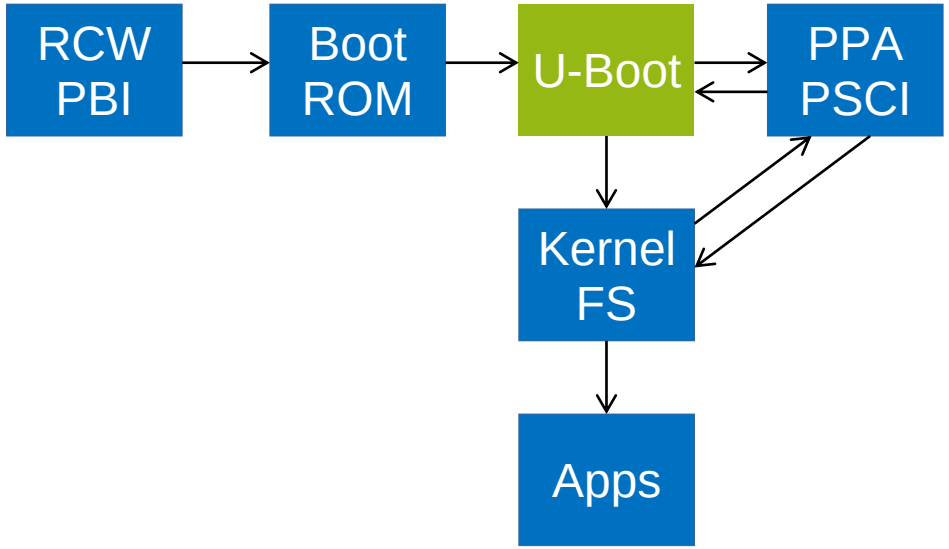
BootROM

- Fixed code in ROM
- Pass control to boot loader



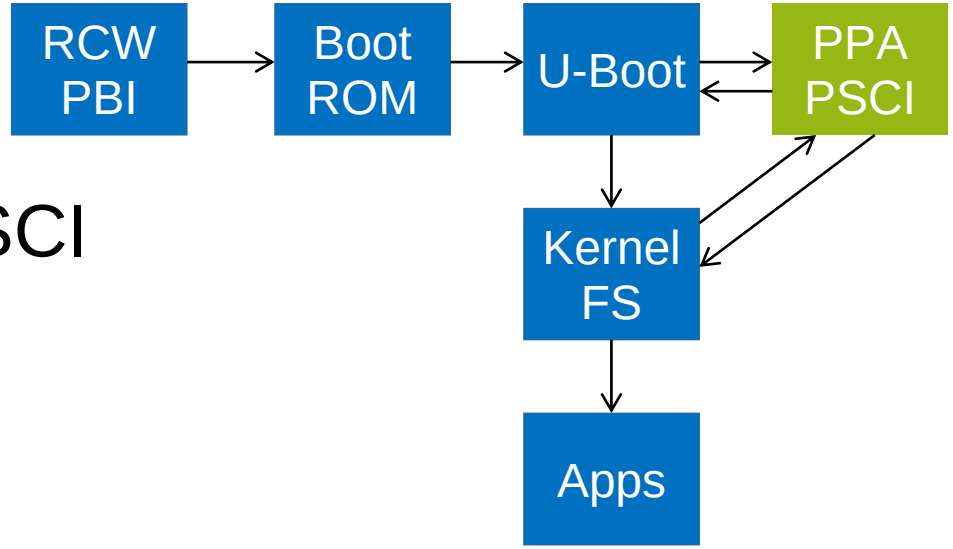
U-Boot

- Execute-in-place
  - NOR, QSPI
  - Slow at first, faster after loading driver
  - Image size about 600~700KB
- Secondary program loader (SPL)
  - NAND, SD
  - Small image loaded by PBI
  - Runs in on-chip RAM
  - Image size about 60~70KB
  - Load full-feature U-Boot image



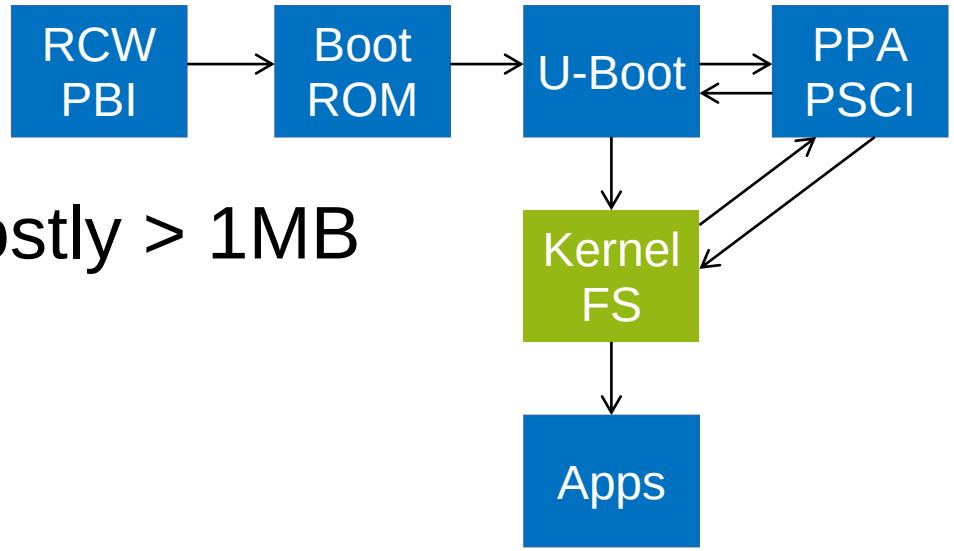
PPA

- Loaded by U-Boot
- Runs in DDR or on-chip RAM
- Provides runtime services including PSCI
- Returns to U-Boot at EL2
- Image size 20~30KB



Linux Kernel

- Loaded by U-Boot
- Runs in DDR
- Size varies based on configuration, mostly > 1MB



Root File System

- Can be in DDR, NOR, NAND, SD, USB, SATA, etc.
- Includes applications
- Size varies
- Speed varies

Where is Time Spent?

- I/O
  - Storage devices
  - Human interface devices

- Unnecessary codes/services
  - U-Boot drivers never used
  - Linux drivers never used
  - Applications never used

- Incorrectly handling images
  - Copying Linux kernel from NOR/QSPI to memory

How to Improve?

- I/O
  - Use fastest device
  - Reduce unnecessary boot messages
- Unnecessary codes/services
  - Disable unused drivers
  - Remove unused applications
- Handling images
  - Use compressed image to reduce size
  - Avoid moving images

Lowest Hanging Fruits

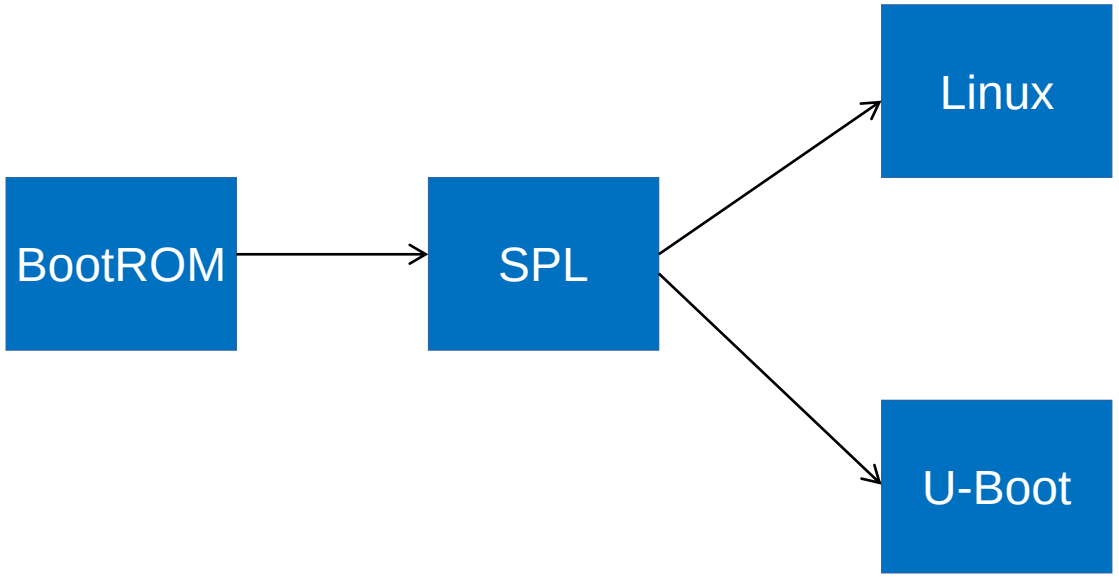
- I/O
  - Try multiple boot sources
  - Turn off kernel boot messages

- Image size
  - Disabling unused drivers can reduce kernel size to 1~2MB
  - Removing unused applications can reduce root file system size to 800KB

- These can reduce boot time down to 1~2 seconds

U-Boot Falcon Mode

- Use SPL framework
- Skip loading full-feature U-Boot
- Work on NAND, SD boot
- Can be modified to support NOR and QSPI



Caveats

- Static device tree
  - Need to be pre-processed from a regular boot

Example

- LS1046ARDB
  - Boot from SD
  - Skip DDR SPD, using fixed setting
  - Disable ECC to skip data init
  - Falcon mode enabled
  - Disable unused Linux drivers
  - loglevel=4
  - Using buildroot/busybox as root file system

Example (Cont.)

- Image size
  - U-Boot SPL 69KB
  - PPA 15KB
  - Linux with RootFS 2.3MB

Example (Cont.)

- Boot time details
  - RCW/PBI 97ms
  - U-Boot time (including loading kernel and rootfs) 430ms
  - Linux boot time 154ms

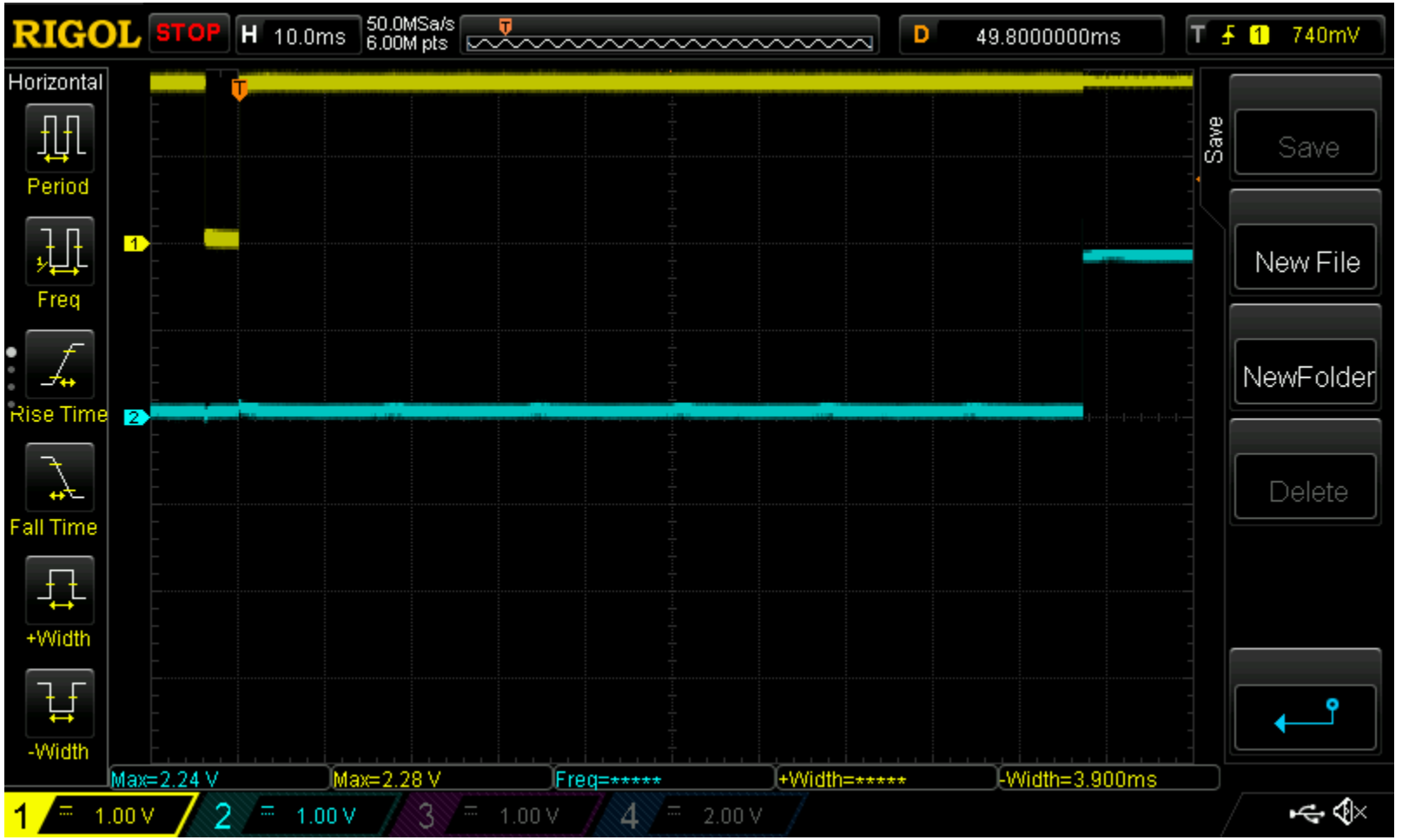
- Total boot time 681ms!

Example (Cont.)

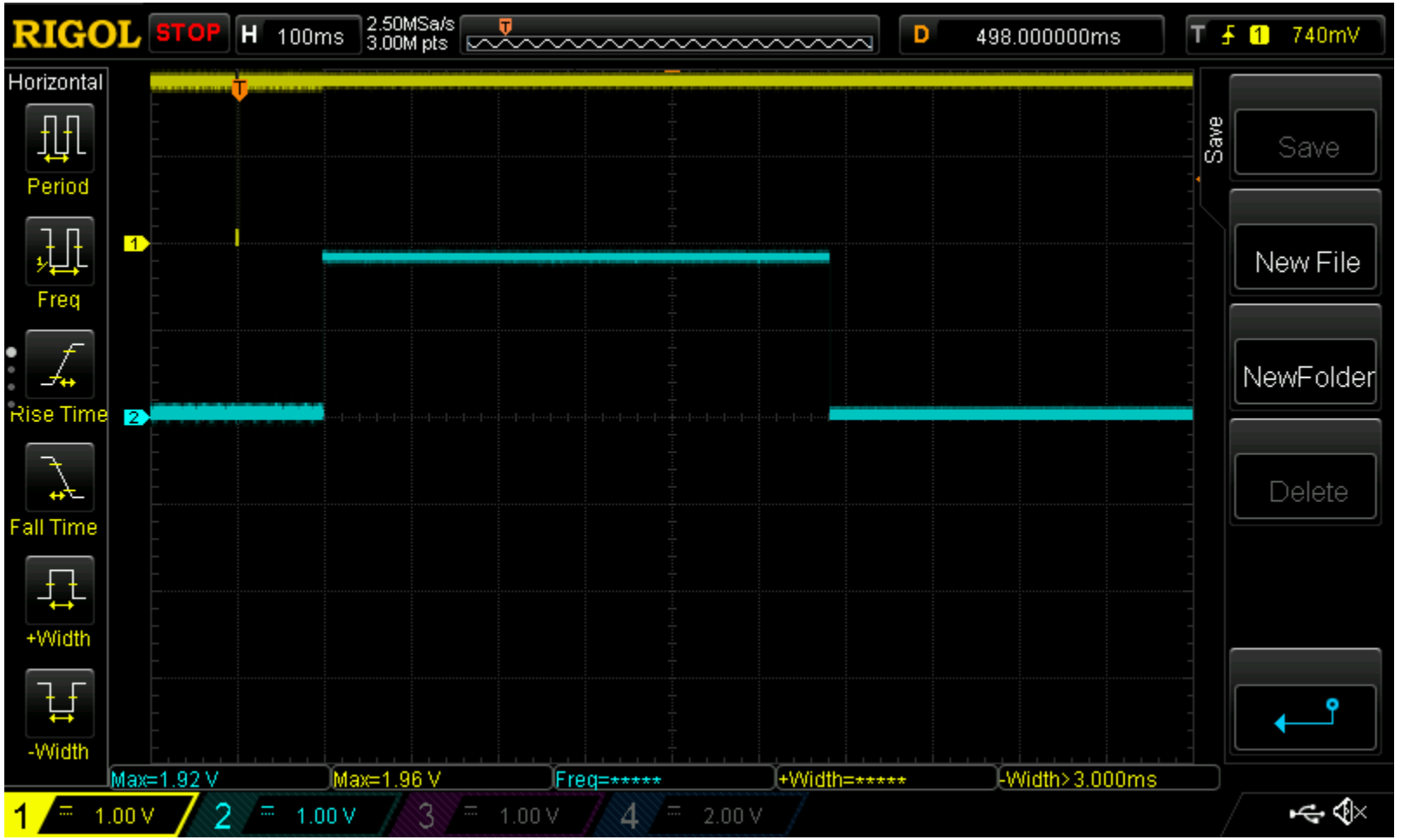
- Time measurement
  - RCW/PBI: from PORESET to GPIO toggle in start.S
  - U-Boot time: Use timer to measure
  - Linux boot time: Use timer to measure

- Total boot time confirmed from PORESET to GPIO toggled by app

RCW/PBI Time


RIGOL
STOP
H 10.0ms
50.0MSa/s
6.00M pts
D 49.8000000ms
T 1 740mV
Horizontal
Period
Freq
Rise Time
Fall Time
+Width
-Width
Save
Save
New File
NewFolder
Delete
Max=2.24 V
Max=2.28 V
Freq=*****
+Width=*****
-Width=3.900ms
1 1.00 V
2 1.00 V
3 1.00 V
4 2.00 V

Total Boot Time



Boot Log

```
U-Boot SPL 2017.03-00052-g1a7707e (May 21 2018 - 16:24:51)
York DEBUG (board_init_f 114): 5 ms
Initialzing DDR using fixed setting
Configuring DDR for 2100 MT/s data rate
York DEBUG (dram_init 904): timer: 23 ms
York DEBUG (timer_init 583): timer: 28 ms
PPA Firmware: Version fsl-sdk-v2.0-1701-47-ga1e86b4
Trying to boot from MMC1
York DEBUG (jump_to_image_linux 56): 430 ms
Starting logging: OK
Initializing random number generator... done.
Starting network: OK

Welcome to Buildroot
buildroot login: root
Jan  1 00:00:10 login[165]: root login on 'console'
#
```

Boot Log (Cont.)

```
# dmesg
[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 4.11.0-rc4-next-20170330-35547-gb2d481e (york@oslab-l16) (gcc version 6.2.1 20161016 (Linaro GCC 6.2-2016.11) ) #19 SMP Mon Apr 17 12:44:02 PDT 2017
[    0.000000] Boot CPU: AArch64 Processor [410fd082]
<snip>
[    0.000724] CPU1: Booted secondary processor [410fd082]
[    0.000856] Detected PIPT I-cache on CPU2
[    0.000869] CPU2: Booted secondary processor [410fd082]
[    0.001003] Detected PIPT I-cache on CPU3
[    0.001016] CPU3: Booted secondary processor [410fd082]
[    0.001037] smp: Brought up 1 node, 4 CPUs
[    0.001039] SMP: Total of 4 processors activated.
[    0.001041] CPU features: detected feature: 32-bit EL0 Support
[    0.001065] CPU: All CPU(s) started at EL2
[    0.001212] York DEBUG: 11984833 (479 ms)
<snip>
[    0.099073] Freeing unused kernel memory: 320K
[    0.104349] York DEBUG: Timestamp
[    0.107559] random: dd: uninitialized urandom read (512 bytes read)
[    0.135417] mmc0: new SD card at address a95c
[    0.135545] mmcblk0: mmc0:a95c SD256 241 MiB
[    0.136483]  mmcblk0: p1
#
```

SECURE CONNECTIONS
FOR A SMARTER WORLD

www.nxp.com