BCM963XX

Commengine Generic User Partition Configuration Feature

Application Note

The term “Broadcom” refers to Broadcom Limited and/or its subsidiaries. For more information, please visit www.broadcom.com.

Broadcom reserves the right to make changes without further notice to any products or data herein to improve reliability, function, or design. Information furnished by Broadcom is believed to be accurate and reliable. However, Broadcom does not assume any liability arising out of the application or use of this information, nor the application or use of any product or circuit described herein, neither does it convey any license under its patent rights nor the rights of others.

Table of Contents

Chapter 1: Introduction

The Generic User Partition Configuration feature allows Commengine users to specify partition configuration for user-defined partitions. The partitions defined using this feature are created and mounted on bootup after the rootfs has been mounted.

1.1 Supported Features

The Generic User Partition Configuration feature supports the following options:

- The feature allows for the specification of:
  - Partition name
  - Partition size
  - Partition storage device
  - Mount point and mount options
  - File system size
  - File system type
- Supports RAW (NAND) flash devices using the Linux MTD Layer.
- Supports Managed (eMMC) flash devices using the Linux Block Device Layer.
- Allows specification of a partition creation policy.
- Automatically sets up partitions on boot.

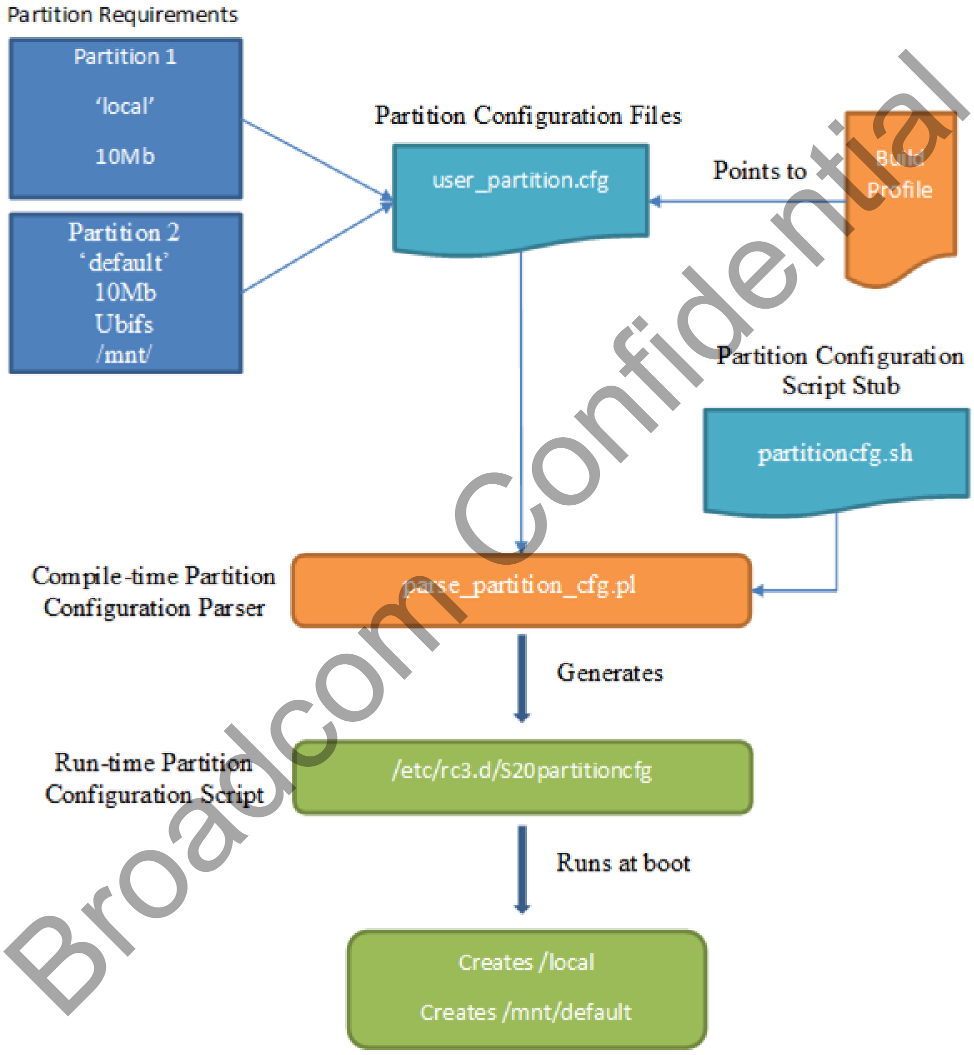
1.2 Availability

The Generic User Partition Configuration feature is available from the 5.02L05 release onwards.

Chapter 2: Implementation

The Generic User Partition Configuration allows users to create partition configuration files which are picked up by the build system and compiled into a partition configuration script. This partition configuration script is executed at bootup to create user-defined partitions. The overall all mechanism is shown in Figure 1.

Figure 1: User Partition Configuration Mechanism

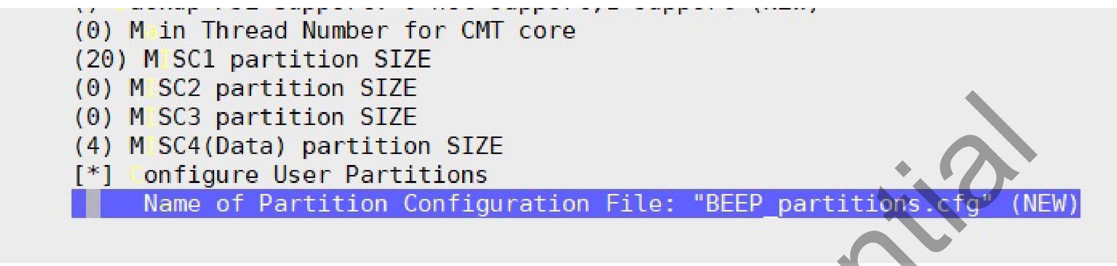


2.1 Configuring Partitions

1. **Enable the Generic User Partition Configuration feature via menuconfig.**

   Use ‘`make menuconfig`’ to edit your build profile in order to enable the feature and specify the name of the partition configuration file. The partition configuration related options are under the ‘`Chip and Board configuration selection`’ menu. The specific menu is shown in Figure 2.

Figure 2: Menuconfig Options for Partition Configuration

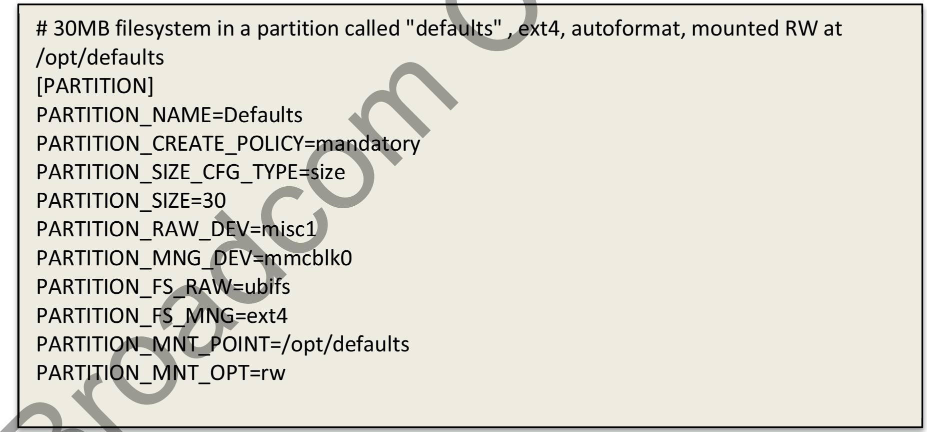


```
(0) Main Thread Number for CMT core
(20) MISC1 partition SIZE
(0) MISC2 partition SIZE
(0) MISC3 partition SIZE
(4) MISC4(Data) partition SIZE
[*] Configure User Partitions
      Name of Partition Configuration File: "BEEP_partitions.cfg" (NEW)
```

2. **Create a partition configuration file.**

   All partition configuration should be entered into a partition configuration file. The partition configuration file can be placed in either `targets/<profile name>` or `targets/arch`. Each partition in the partition configuration file is defined using a partition configuration block. Figure 3 shows a sample partition configuration block.

Figure 3: Partition Configuration Block



```
# 30MB filesystem in a partition called "defaults" , ext4, autoformat, mounted RW at
/opt/defaults
[PARTITION]
PARTITION_NAME=Defaults
PARTITION_CREATE_POLICY=mandatory
PARTITION_SIZE_CFG_TYPE=size
PARTITION_SIZE=30
PARTITION_RAW_DEV=misc1
PARTITION_MNG_DEV=mmcblk0
PARTITION_FS_RAW=ubifs
PARTITION_FS_MNG=ext4
PARTITION_MNT_POINT=/opt/defaults
PARTITION_MNT_OPT=rw
```

   There can be multiple partitions in a partition configuration file. Partitions with duplicate names are not allowed. All the available partition configuration parameters are shown in Table 1, Partition Configuration Parameters and a sample full configuration file is shown in Figure 4, Sample Partition Configuration File.

3. **Build a regular Commengine image.**

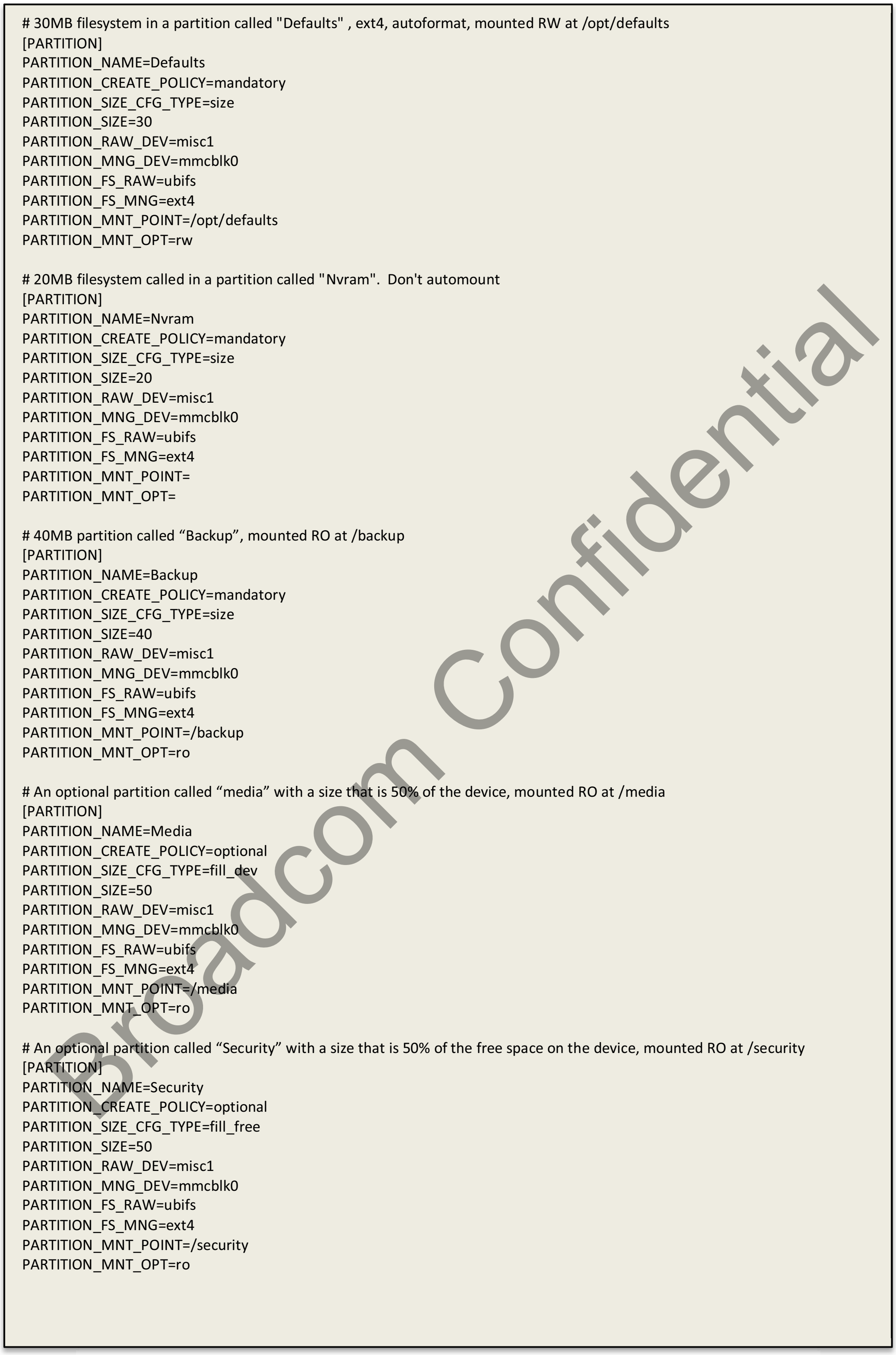
   Use the modified profile from Step 1 to build a Commengine image.

Table 1: Partition Configuration Parameters

Parameter	Description	Values
PARTITION_NAME	Name of UBI volume or GPT partition	Alphanumeric name.
PARTITION_CREATE_POLICY	Policy governing when the partition is created	**mandatory**: Partition is always created if not present. If space is used up by other partitions, then those partitions will be removed and the new one will be created. **optional**: Will not create partition if there is no space available.
PARTITION_SIZE_CFG_TYPE	How the partition size is specified	**size**: Partition size is specified in MB. **fill_free**: Partition size is specified by a fill percentage, in other words, create a partition which is a certain percentage of the free space on the specified MTD or Block device. **fill_dev**: Partition size is specified by a fill percentage, in other words, create a partition which is a certain percentage of the size of the entire MTD or Block device.
PARTITION_SIZE	Partition size specification	**>0**: When specifying size in Mb. **1–100**: When specifying a fill percentage.
PARTITION_RAW_DEV	MTD device name in which the partitions will be created	Alphanumeric name.
PARTITION_MNG_DEV	Block device name in which the partitions will be created	Alphanumeric name.
PARTITION_RAW_FS	File system for raw flash devices (NAND)	**ubifs**
PARTITION_MNG_FS	File system for managed flash devices (eMMC)	**ext4** **vfat**
PARTITION_MNT_OPT	Mount options	Mount options comma delimited.
PARTITION_MNT_POINT	Mount point	Full path to mount point. Ensure that this mount point has read-write permissions.

NOTE: In order to determine the maximum partition size, the file system overheads must be taken into account. File systems allocate a certain amount of device blocks, with the block size being dependent upon the underlying device geometry, for storage of metadata or for wear-leveling purposes. This means that a device of size ‘x’ will only have a maximum possible partition ‘x – fs overhead’. For example, if we require a UBIFS partition on a 20 MB raw NAND MTD partition with a native block size of 256 KB, the maximum possible partition size will be 17 MB. The reason for this reduction in size is that UBIFS requires eight device blocks for storage of its metadata. Similar overheads exist for the ext4 and vfat file systems.

Figure 4: Sample Partition Configuration File

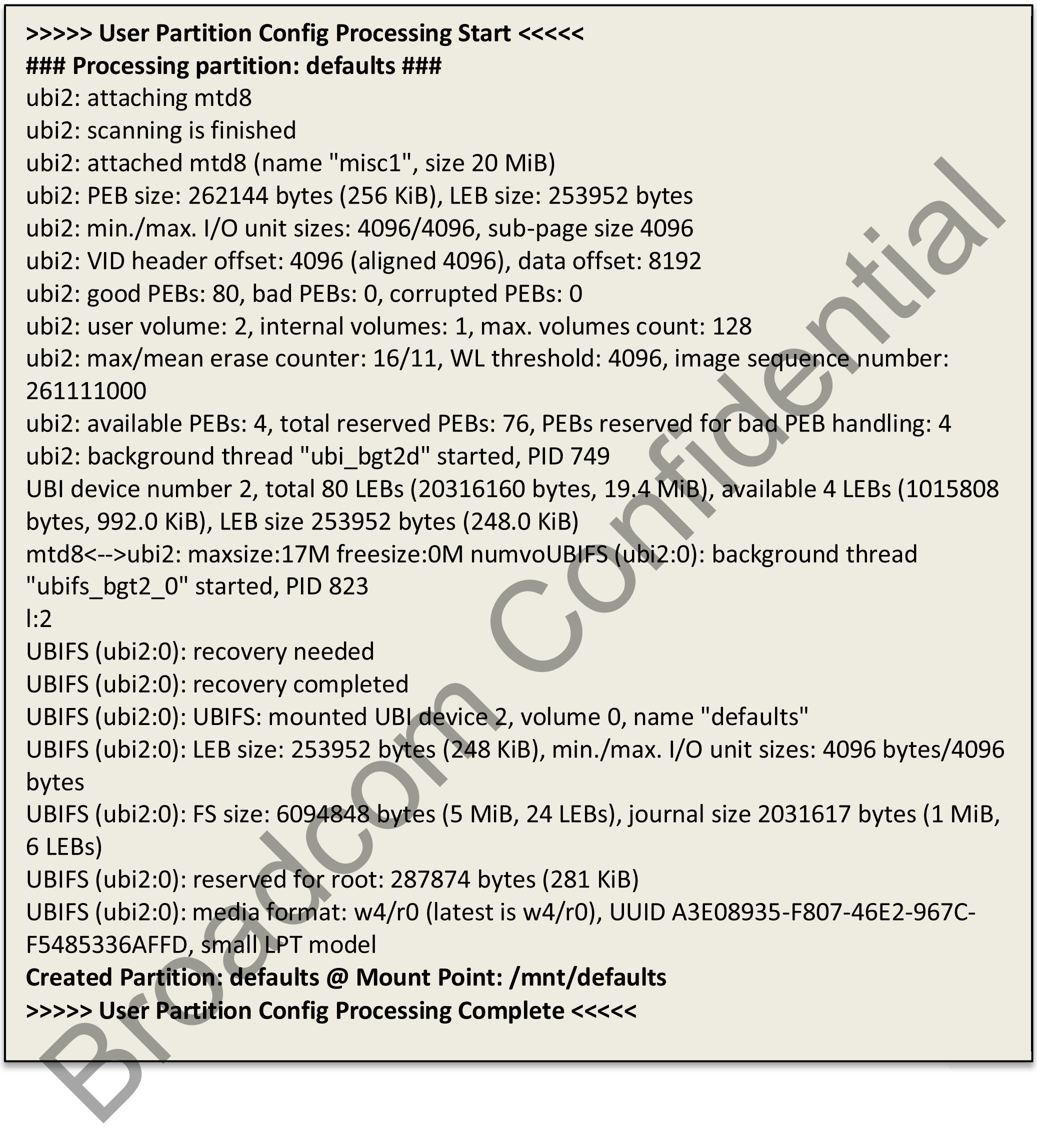


```
# 30MB filesystem in a partition called "Defaults" , ext4, autoformat, mounted RW at /opt/defaults
[PARTITION]
PARTITION_NAME=Defaults
PARTITION_CREATE_POLICY=mandatory
PARTITION_SIZE_CFG_TYPE=size
PARTITION_SIZE=30
PARTITION_RAW_DEV=misc1
PARTITION_MNG_DEV=mmcblk0
PARTITION_FS_RAW=ubifs
PARTITION_FS_MNG=ext4
PARTITION_MNT_POINT=/opt/defaults
PARTITION_MNT_OPT=rw

# 20MB filesystem called in a partition called "Nvram".  Don't automount
[PARTITION]
PARTITION_NAME=Nvram
PARTITION_CREATE_POLICY=mandatory
PARTITION_SIZE_CFG_TYPE=size
PARTITION_SIZE=20
PARTITION_RAW_DEV=misc1
PARTITION_MNG_DEV=mmcblk0
PARTITION_FS_RAW=ubifs
PARTITION_FS_MNG=ext4
PARTITION_MNT_POINT=
PARTITION_MNT_OPT=

# 40MB partition called “Backup”, mounted RO at /backup
[PARTITION]
PARTITION_NAME=Backup
PARTITION_CREATE_POLICY=mandatory
PARTITION_SIZE_CFG_TYPE=size
PARTITION_SIZE=40
PARTITION_RAW_DEV=misc1
PARTITION_MNG_DEV=mmcblk0
PARTITION_FS_RAW=ubifs
PARTITION_FS_MNG=ext4
PARTITION_MNT_POINT=/backup
PARTITION_MNT_OPT=ro

# An optional partition called “media” with a size that is 50% of the device, mounted RO at /media
[PARTITION]
PARTITION_NAME=Media
PARTITION_CREATE_POLICY=optional
PARTITION_SIZE_CFG_TYPE=fill_dev
PARTITION_SIZE=50
PARTITION_RAW_DEV=misc1
PARTITION_MNG_DEV=mmcblk0
PARTITION_FS_RAW=ubifs
PARTITION_FS_MNG=ext4
PARTITION_MNT_POINT=/media
PARTITION_MNT_OPT=ro

# An optional partition called “Security” with a size that is 50% of the free space on the device, mounted RO at /security
[PARTITION]
PARTITION_NAME=Security
PARTITION_CREATE_POLICY=optional
PARTITION_SIZE_CFG_TYPE=fill_free
PARTITION_SIZE=50
PARTITION_RAW_DEV=misc1
PARTITION_MNG_DEV=mmcblk0
PARTITION_FS_RAW=ubifs
PARTITION_FS_MNG=ext4
PARTITION_MNT_POINT=/security
PARTITION_MNT_OPT=ro
```

2.2 Runtime Output

When an image with the Generic User Partition Configuration feature enabled is run, the following output can be seen on the console. Note that this output is from a system which has raw NAND flash.

Figure 5: Sample Script Output



```
>>>>> User Partition Config Processing Start <<<<<
### Processing partition: defaults ###
ubi2: attaching mtd8
ubi2: scanning is finished
ubi2: attached mtd8 (name "misc1", size 20 MiB)
ubi2: PEB size: 262144 bytes (256 KiB), LEB size: 253952 bytes
ubi2: min./max. I/O unit sizes: 4096/4096, sub-page size 4096
ubi2: VID header offset: 4096 (aligned 4096), data offset: 8192
ubi2: good PEBs: 80, bad PEBs: 0, corrupted PEBs: 0
ubi2: user volume: 2, internal volumes: 1, max. volumes count: 128
ubi2: max/mean erase counter: 16/11, WL threshold: 4096, image sequence number:
261111000
ubi2: available PEBs: 4, total reserved PEBs: 76, PEBs reserved for bad PEB handling: 4
ubi2: background thread "ubi_bgt2d" started, PID 749
UBI device number 2, total 80 LEBs (20316160 bytes, 19.4 MiB), available 4 LEBs (1015808
bytes, 992.0 KiB), LEB size 253952 bytes (248.0 KiB)
mtd8<-->ubi2: maxsize:17M freesize:0M numvoUBIFS (ubi2:0): background thread
"ubifs_bgt2_0" started, PID 823
l:2
UBIFS (ubi2:0): recovery needed
UBIFS (ubi2:0): recovery completed
UBIFS (ubi2:0): UBIFS: mounted UBI device 2, volume 0, name "defaults"
UBIFS (ubi2:0): LEB size: 253952 bytes (248 KiB), min./max. I/O unit sizes: 4096 bytes/4096
bytes
UBIFS (ubi2:0): FS size: 6094848 bytes (5 MiB, 24 LEBs), journal size 2031617 bytes (1 MiB,
6 LEBs)
UBIFS (ubi2:0): reserved for root: 287874 bytes (281 KiB)
UBIFS (ubi2:0): media format: w4/r0 (latest is w4/r0), UUID A3E08935-F807-46E2-967C-
F5485336AFFD, small LPT model
Created Partition: defaults @ Mount Point: /mnt/defaults
>>>>> User Partition Config Processing Complete <<<<<
```

Chapter 3: Limitations

The current implementation has the following limitations:

- Mandatory partitions are processed first, before any other partitions.
- On eMMC/Managed block-based flash devices, the partition creation mechanism requires a reboot between partition creation and formatting.
- Only UBIFS is supported for RAW flash devices.
- VFAT and EXT4 are supported for eMMC.

Revision History

963XX-AN700-R, March 20, 2018

- Initial release