Getting Started Guide

RELEASE UM 3.6.1

Unmanaged-SWUM106

Corporate Headquarters: San Jose, CA

January 24, 2018

For a comprehensive list of changes to this document, see the Revision History.

Table of Contents

Broadcom's Unmanaged Software is available in three different packages: Unmanaged Web (UM-Web), Unmanaged Plus (UM+) and Unmanaged Basic (UM-Basic). This document applies to all packages.

Purpose

- Add on value of unmanaged device
- Leverage embedded resources of silicon
  - Processor
  - SRAM
- Less cost

Summary

System Services

- Single threaded
  - No OS, no interrupt used
  - CPU timer used as system tick
  - One background task to run all registered tasks
  - SAL layer for each platform
- Start-up code and BSP
  - CPU initialization (cache, MMU/MPU, timer, UART, etc.)
  - Serial flash driver

Switch Management

- Switch init sequence
  - Chip reset and IP/EP/MMU initializations
  - DMA init for packet TX/RX on CPU port
- Use MDK-PHY/SDK-PHY drivers in UM
  - Port PHY drivers and remove unnecessary configurations
  - MAC/PHY (SerDes) init sequence at start-up
  - Use SDK PHY pre-built libraries as UM PHY drivers of BCM95357x platform, and use MDK PHY library as PHY drivers for other platforms
- Linkscan task
  - Poll PHY status and program port information (speed, duplex, and pause) to MAC according to the link status when the link is up
  - Link down process for MAC/PHY
- Board API for each switch feature
  - For example, `board_vlan_crate()` to create a VLAN
- Includes uIP stack in release package

Functions

UM-Basic	Only does the basic initialization for CPU/MAC/SerDes/PHY to allow the packets to be switched among the front ports. The purpose of the UM-Basic package is to let the switch behave as a dumb switch.
UM-Web	Include more L2 features besides the basic initialization of the UM-Dump package. The user can also enable/disable those features through the web page. The firmware upgrade can also be done through the web page.
UM+	Remove WEB features and revise VLAN default settings from the UM-Web package.

Table 1 shows the features for the three packages.

Table 1: Features

Feature Support	*Description*	*UM-Web*	*UM+*	*UM-Basic*	*Default*
Console/UART	–	V	V	V	9600 bps
L2 AGE	–	V	V	V	300 seconds
EEE	–	V	V	V	Enabled
Jumbo Frame	–	V	V	V	9216 bytes
Port Flow Control	–	V	V	V	Enabled
DoS	Auto Denial of Service prevention. Check DOS_CONTROL and DOS_CONTROL_2 registers for details.	V	V	–	Enabled
IEEE 802.1p QoS	Schedule mode is Weighted Round Robin (WRR) and weighted for priority queues (0, 1, 2, and 3) is (1:2:4:8).	V	V	V	Enabled
DHCP	Note[a]	V	V	–	Enabled
VLAN	–	V	V	V	Note[b]
LAG	Maximum of four groups. Maximum eight ports per group.	V	V	–	Disabled
Mirror	–	V	V	–	Disabled
Port-Based QoS	–	V	V	–	Disabled
Rate Limit	–	V	V	–	Disabled
Storm Control	–	V	V	–	Disabled
IGMP Snooping	–	V	V	–	Enabled
Block Unknown Multicast Address	–	V	V	–	Disabled
Access Control	Covers “limit designated SIP access” and “limit single user access”.	V	V	–	Disabled
Loop Detection	–	V	V	–	Disabled
Cable Diagnostics	–	V	V	–	Triggered by user

Table 1: Features (Cont.)

Feature Support	*Description*	*UM-Web*	*UM+*	*UM-Basic*	*Default*
Port Status	Covers link status, speed duplex, flow control, and auto-negotiation status.	V	–	–	–
Statistics	Covers TX octets, RX octets, and CRC errors.	V	–	–	–
LED Display	Link/Activity LED	V	V	V	–
Firmware Upgrade	–	V	–	–	–
Password Protection	–	V	–	–	–
Snake Test	Support snake test by CLI command.	V	V	V	Triggered by user
Dual Image	–	V	–	–	Enabled
Bonjour	Supports Bonjour browser to discover device.	V	–	–	Enabled
Web	Support web browser to access the device.	V	–	–	–
Persistence	Supports persistence storage.	V	V	–	–
Vendor Config	Configure different settings without rebuilding the image	V	V	V	–

a. For UM-Web, the address is set to a random IPv4 link-local address if DHCP fails. The range of IPv4 Link-local addresses is 169.254.1.0 to 169.254.254.255. For UM+, the address is set to 192.168.0.239 if DHCP fails.
b. For UM-Web, the default setting is 802.1Q VLAN mode. For UM+ and UM-Basic, the default setting is Port-based VLAN mode. It creates 1 ~ 4094 VLANs as below and sets PVID of all ports to 1:
VLAN 1: All ports in this VLAN are untagged members.
VLAN 2 ~ 4094: All ports in this VLAN are tagged members.

You can use a web browser through HTTP to connect to the web page of a device or use Apple Bonjour to discover the device.

The code directory is shown in Figure 2.

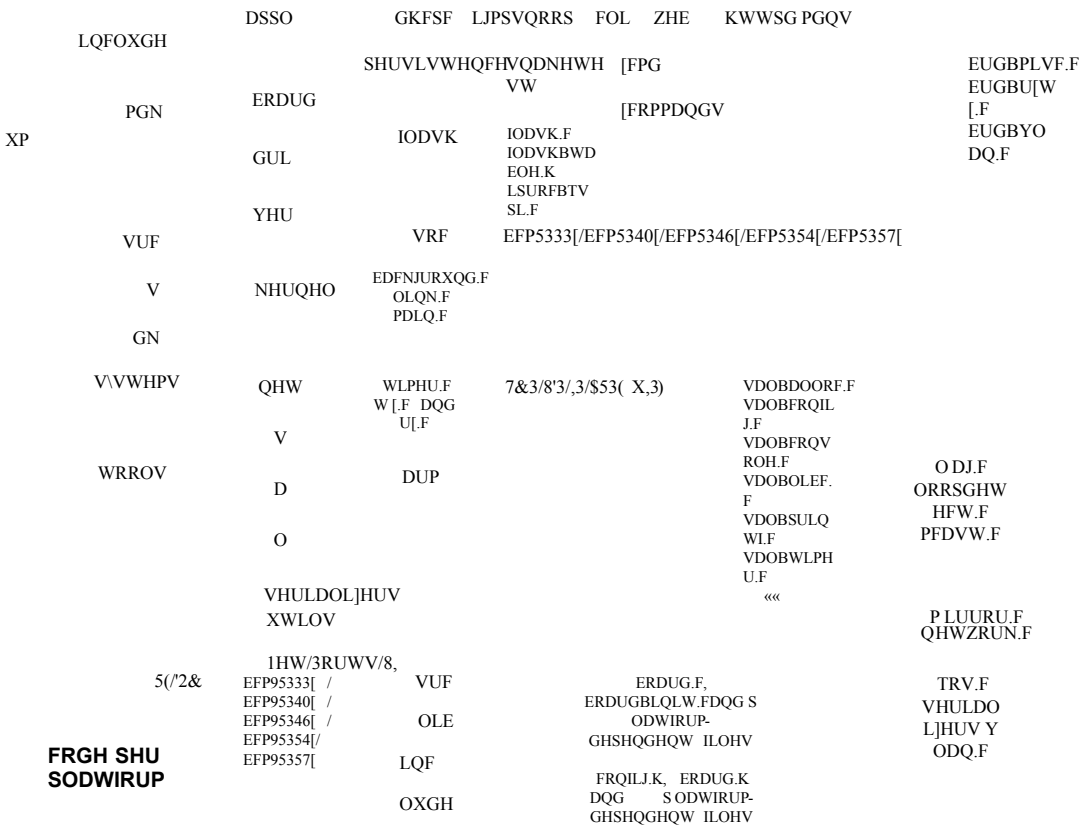


Table 2 gives a brief description of the files that make up the UM code.

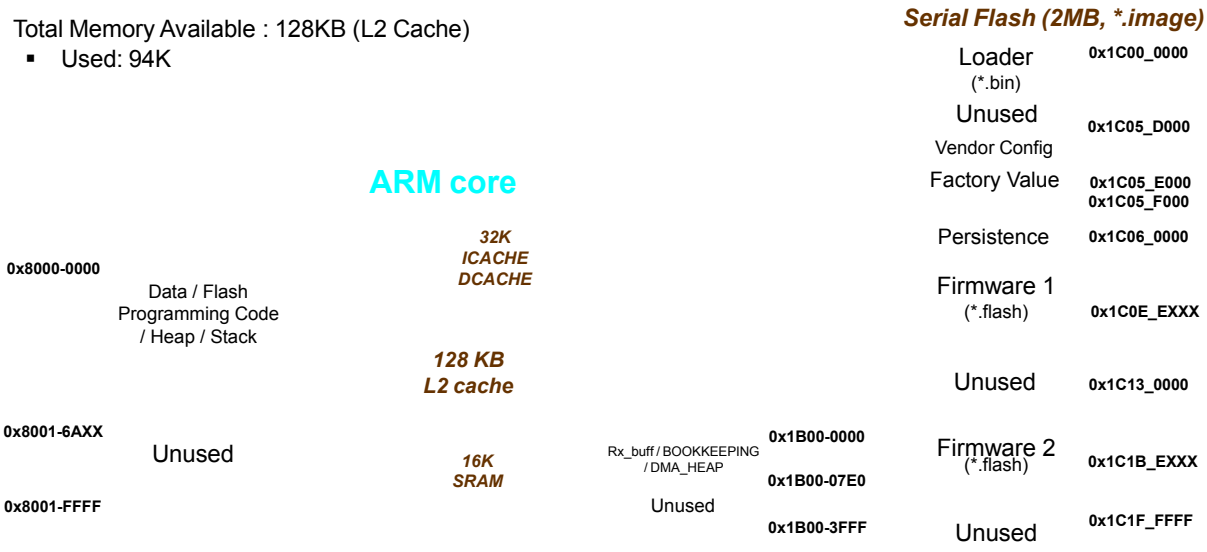
Table 2: UM Files

File	*Description*
$UM/src/sal	The SAL directory that includes printf, timer, assertion, memory allocation, C libraries, and the console API. After UM 3.1.1, the MDK PHY driver is used in UM. The MDK PHY driver is a library file under *system/per-platform-directory/lib* for each platform.

This section describes the flash and memory layout for UM-Web and UM+ on different platforms.

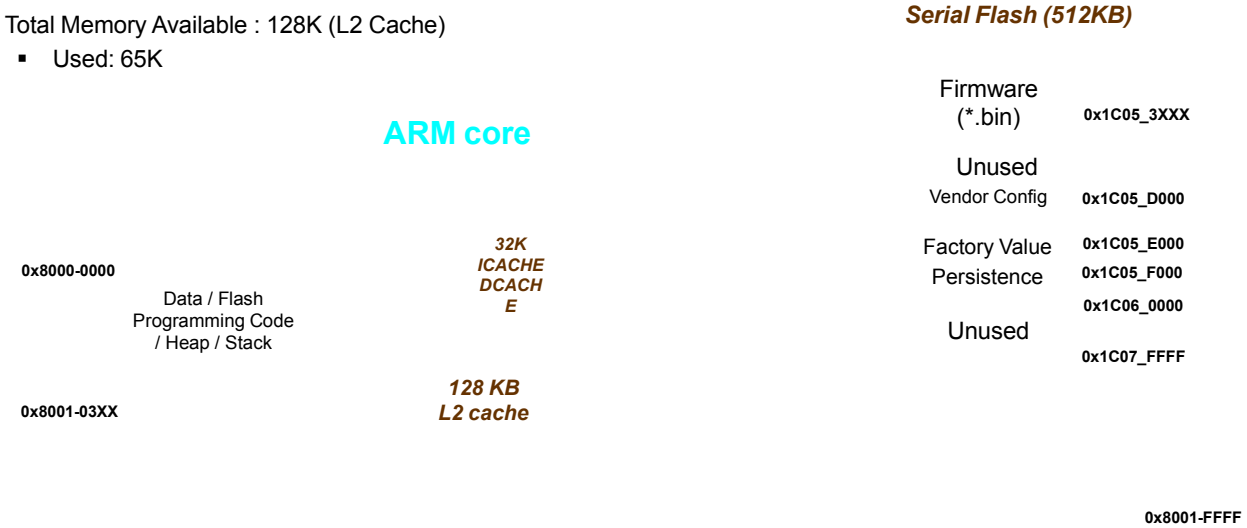
Flash Partitions/Memory Layout for UM-Web (BCM5333X)

- Total Memory Available : 128KB (L2 Cache)
  - Used: 94K



Flash Partitions/Memory Layout for UM+ (BCM5333X)

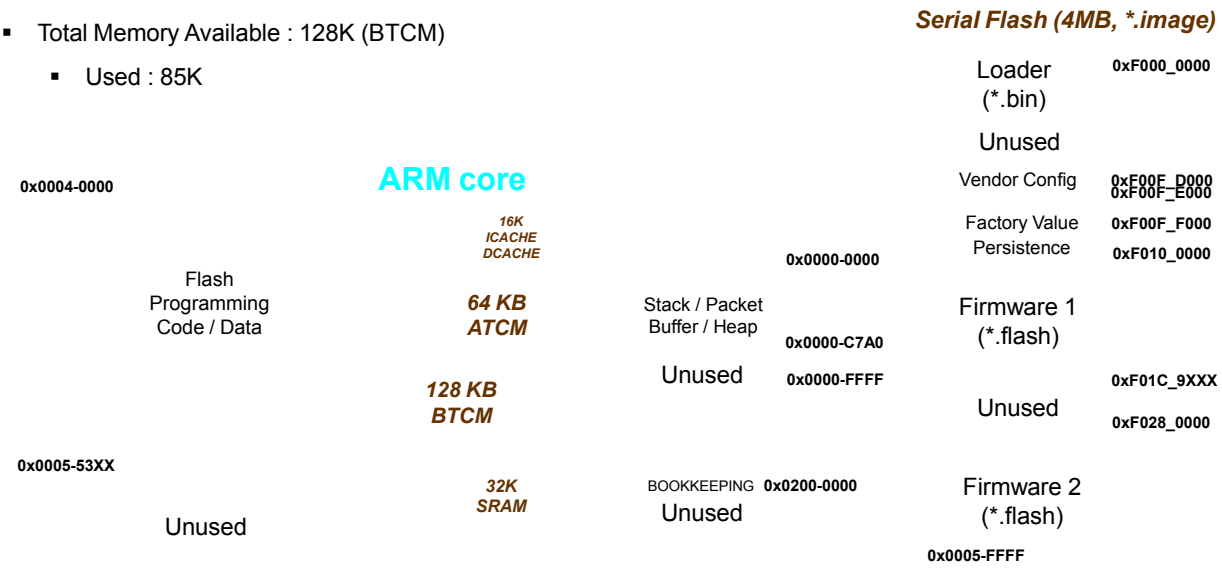
- Total Memory Available : 128K (L2 Cache)
  - Used: 65K




Unused
16K SRAM
Rx_buff
/
BOOKKEEPING
/
DMA_HEAP
0x1B00-0000
0x1B00-07E0
0x1B00-3FFF
Unused

Flash Partitions/Memory Layout for UM-Web (BCM5340X)

- Total Memory Available : 128K (BTCM)
  - Used : 85K



0x0200-7FFF

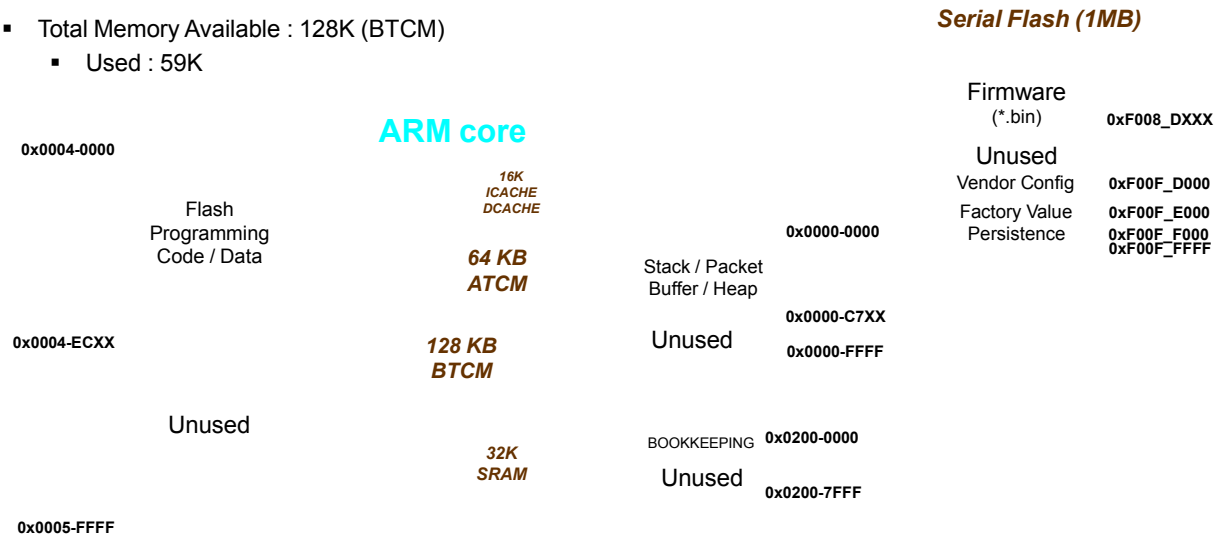
Unused

0xF034_9XXX

0xF03F_FFFF

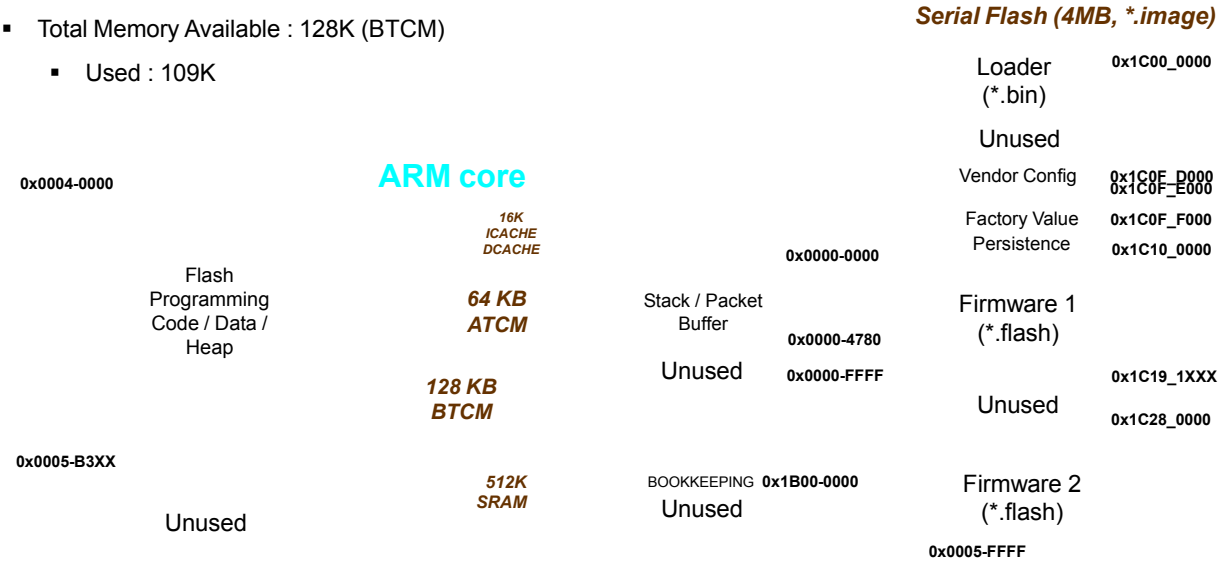
Flash Partitions/Memory Layout for UM+ (BCM5340X)

- Total Memory Available : 128K (BTCM)
  - Used : 59K



Flash Partitions/Memory Layout for UM-Web (BCM5346X)

- Total Memory Available : 128K (BTCM)
  - Used : 109K



0x1B07-FFFF

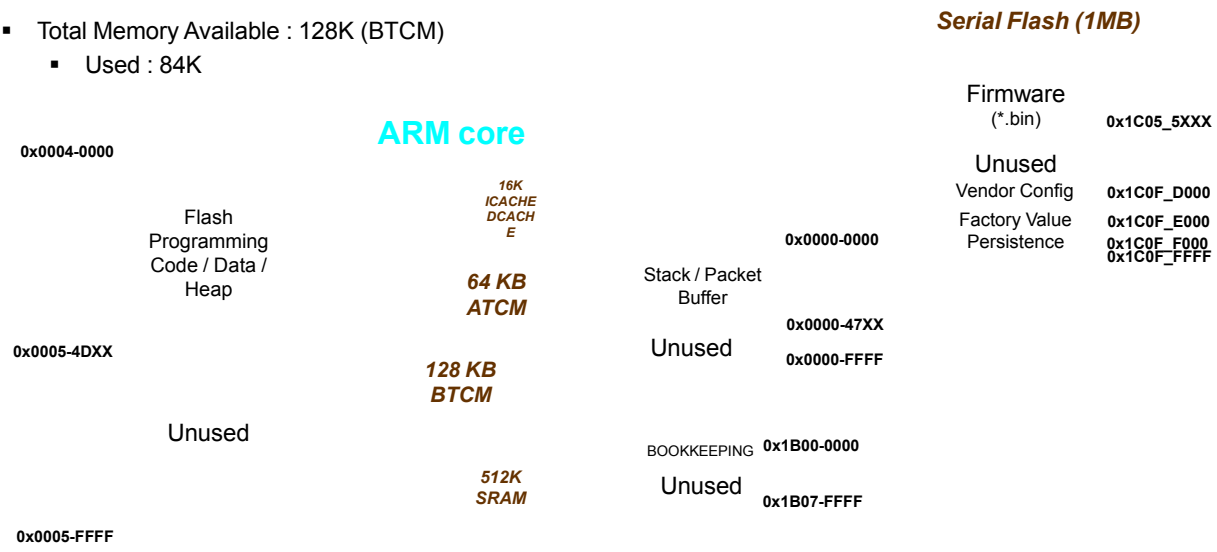
Unused

0x1C31_1XXX

0x1C3F_FFFF

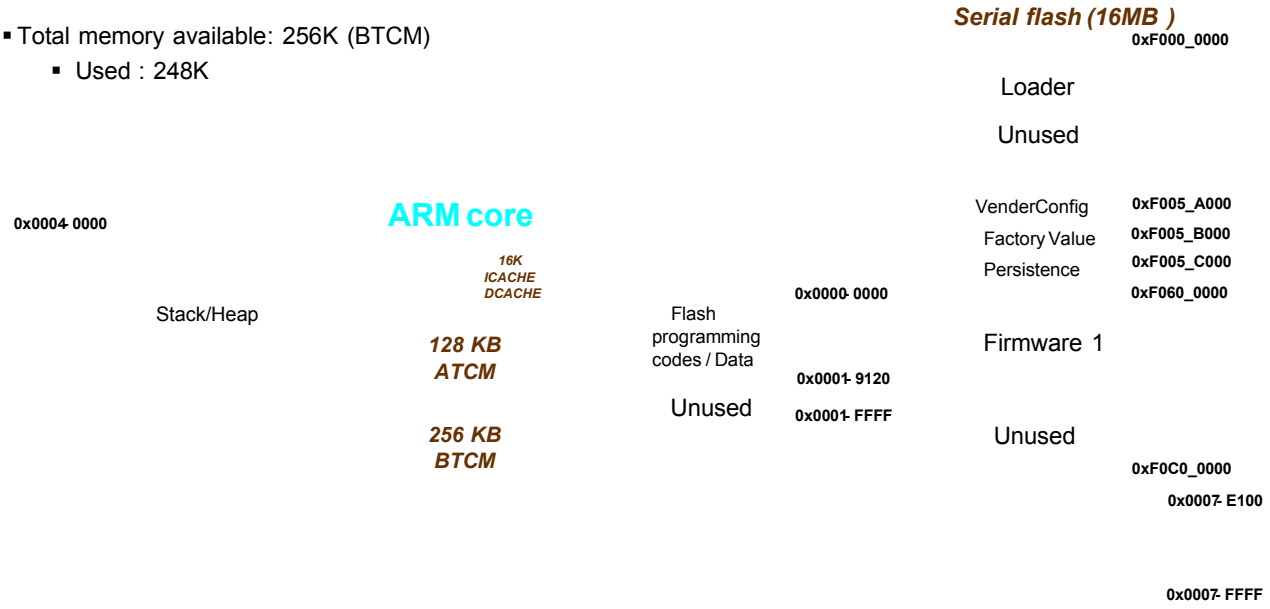
Flash Partitions/Memory Layout for UM+ (BCM5346X)

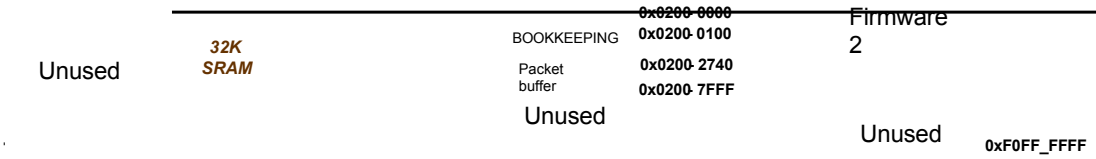
- Total Memory Available : 128K (BTCM)
  - Used : 84K



Flash Partitions/Memory Layout for UM-Web (BCM5357X)

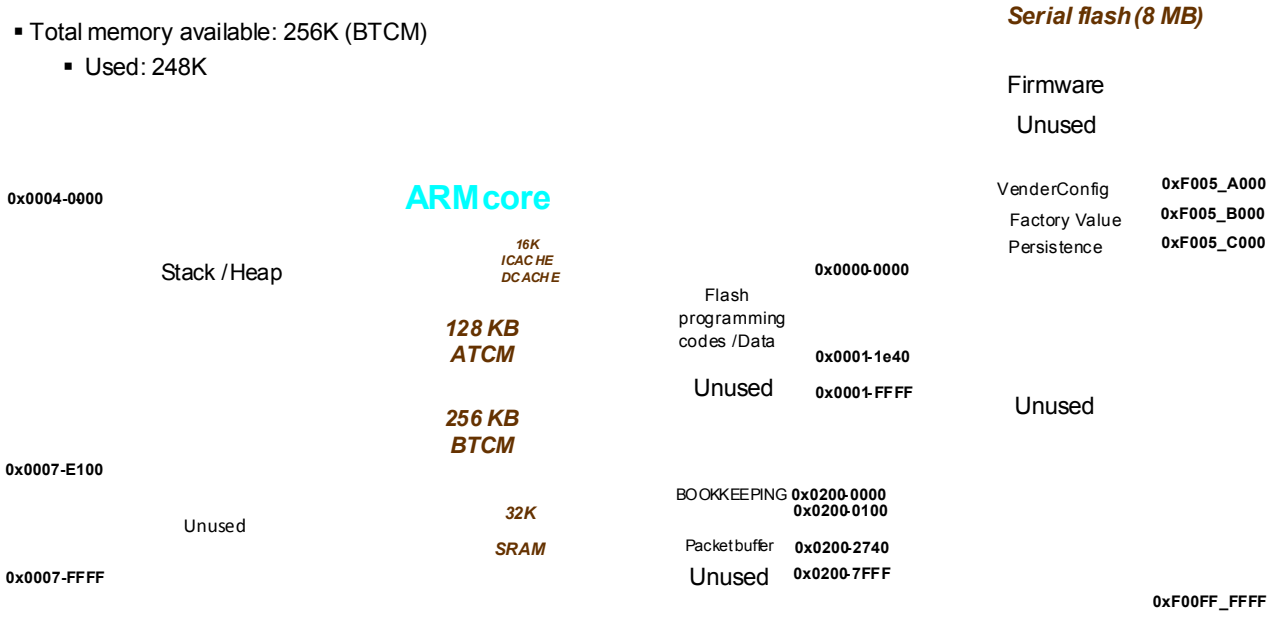
- Total memory available: 256K (BTCM)
  - Used : 248K





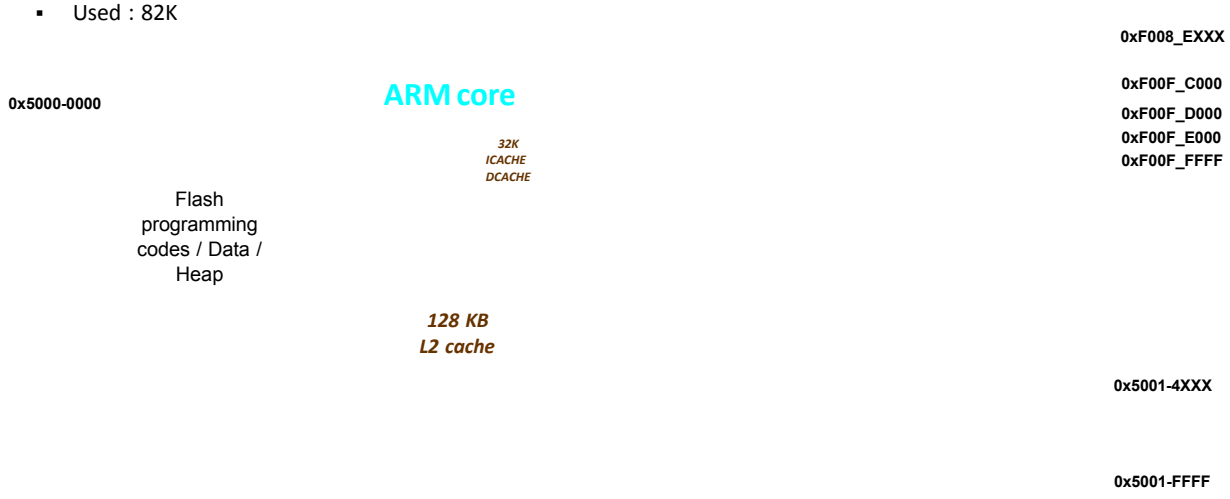
Flash Partitions/Memory Layout for UM+ (BCM5357X)

- Total memory available: 256K (BTCM)
  - Used: 248K



Flash Partitions/Memory Layout for UM-Basic (BCM5354X)

- Total memory available : 128K (L2 cache)
  - Used : 82K



Unused	*32K SRAM*	**0x0200-7FXX**	
		0x0200-7FFF	

Get the source tree in the release tar file.

Toolchain

The BCM5333X, BCM5340X, BCM5346X, BCM5354X, and BCM5357X platforms use the same toolchain.

1. Download the GNU GCC ARM Embedded 4.8 update, and install it.
   https://launchpad.net/gcc-arm-embedded/4.8/4.8-2014-q1-update
2. Change TOOLCHAIN_DIR and TOOLPREFIX in *$UM/systems/$PLATFORM/src/tools.mk* to the location where the toolchain is installed. *$PLATFORM* could be bcm95333x, bcm95340x, bcm95346x, bcm95354x, or bcm95357x.

Build the MDK PHY Library

You can rebuild the MDK PHY library before building the image if you change the PHY code under *um/mdk*.

1. Go to the directory *$UM/systems/$PLATFORM*. *$PLATFORM* could be bcm95333x, bcm95340x, or bcm95346x.
2. To generate MDK PHY library under *$UM/systems/$PLATFORM/lib*. *$PLATFORM* could be bcm95333x, bcm95340x, or bcm95346x. Type the following:

```
make phylibs
```

Build the Image File for UM-Web

BCM5333X Platform

1. Go to the *$UM/systems/bcm95333x* directory, and create an output directory (outputs).
   ```
   mkdir outputs
   ```
2. Type the following to build the bootloader (*bcm95333x-loader.bin*), and save the bootloader to the output directory.
   ```
   make clean; make target=loader
   cp bcm95333x-loader.bin outputs/
   ```
3. Type the following to build the firmware (*bcm95333x-umweb.flash*), and save the firmware to the output directory.
   ```
   make clean; make target=umweb
   cp bcm95333x-umweb.flash outputs/
   ```
4. Go to output directory and generate a single image (bcm95333x.image) for the BCM5333X platform.
   ```
   cd outputs
   ../../../tools/mkflashimage.pl bcm95333x-loader.bin bcm95333x-umweb.flash bcm95333x.image
   ```
5. Use `flash programmer` to program bcm95333x.image into the flash. Currently, the BCM953334K and BCM953394K use Micro N25Q256A13 serial flash.

BCM5340X Platform

1. Go to the *$UM/systems/bcm95340x* directory, and create an output directory (outputs).
   ```
   mkdir outputs
   ```
2. Type the following to build the bootloader (*bcm95340x-loader.bin*), and save the bootloader to the output directory.
   ```
   make clean; make target=loader
   cp bcm95340x-loader.bin outputs/
   ```
3. Type the following to build the firmware (*bcm95340x-umweb.flash*), and save the firmware to the output directory.
   ```
   make clean; make target=umweb
   cp bcm95340x-umweb.flash outputs/
   ```
4. Go to output directory and generate a single image (*bcm95340x.image*) for the BCM5340X platform.
   ```
   cd outputs
   ../../../tools/mkflashimage.pl bcm95340x-loader.bin bcm95340x-umweb.flash bcm95340x.image
   ```
5. Use `flash programmer` to program *bcm95340x.image* into the flash. Currently, the BCM953406K and BCM953456K use Micro N25Q512A83 serial flash.

BCM5346X Platform

1. Go to the *$UM/systems/bcm95346x* directory, and create an output directory (outputs).
   ```
   mkdir outputs
   ```
2. Type the following to build the bootloader (*bcm95346x-loader.bin*), and save the bootloader to the output directory.
   ```
   make clean; make target=loader
   cp bcm95346x-loader.bin outputs/
   ```
3. Type the following to build the firmware (*bcm95346x-umweb.flash*), and save the firmware to the output directory.
   ```
   make clean; make target=umweb
   cp bcm95346x-umweb.flash outputs/
   ```
4. Go to the output directory and generate a single image (*bcm95346x.image*) for the BCM5346X platform.
   ```
   cd outputs
   ../../../tools/mkflashimage.pl bcm95346x-loader.bin bcm95346x-umweb.flash
   bcm95346x.image
   ```
5. Use flash programmer to program *bcm95346x.image* into the flash. Currently, the BCM956270K uses Macronix MX25L12835FMI-10G serial flash.

BCM5357X Platform

1. Go to the *$UM/systems/bcm95357x* directory, and create an output directory (outputs).
   ```
   mkdir outputs
   ```
2. Check EXT_PHY_LIST in *$UM/systems/bcm95357x/um_gen_phylibs.mk* for the external PHY drivers, and make sure the PHY part number, which is used by the hardware board, is explicitly listed/defined.
3. Type the following to build the bootloader (*bcm95357x-loader.bin*), and save the bootloader to the output directory.
   ```
   make clean; make target=loader
   cp bcm95357x-loader.bin outputs/
   ```
4. Type the following to build the firmware (*bcm95357x-umweb.flash*), and save the firmware to the output directory.
   ```
   make clean; make target=umweb
   cp bcm95357x-umweb.flash outputs/
   ```
5. Go to the output directory and generate a single image (*bcm95357x.image*) for the BCM5357X platform.
   ```
   cd outputs
   ../../../tools/mkflashimage.pl bcm95357x-loader.bin bcm95357x-umweb.flash
   bcm95357x.image
   ```

   Note: Some vendor config examples about BCM5357X SKUs are placed at *$UM/systems/bcm95357x/vendor_config* for reference.

6. Use flash programmer to program *bcm95357x.image* into the flash. Currently, the BCM953570K uses Micron MT25QL512ABB serial flash.

   Note: Files with the extension **.image* are used for the flash programmer, and files with the extension **.flash* are used for the web firmware upgrade.

   Note: Features defined in *$UM/systems/$PLATFORM/include/configs/config_loader.h* and *$UM/systems/$PLATFORM/include/configs/config_umweb.h* can be selectively added or removed before building the firmware. Check the string “Features defined below can be selectively add or removed before building the image” in *config_loader.h* and *config_umweb.h*. *$PLATFORM* can be bcm95333x, bcm95340x, bcm95346x, or bcm95357x.

Build the Image File for UM+

BCM5333X Platform

1. Go to the *$UM/systems/bcm95333x* directory.
2. Type the following to build the firmware (*bcm95333x-umplus.bin*).
   ```
   make clean; make target=umplus
   ```
3. Use flash programmer to program *bcm95333x-umplus.bin* into the flash. Currently, the BCM953334K and BCM953394K use Micro N25Q256A13 serial flash.

BCM5340X Platform

1. Go to the *$UM/systems/bcm95340x* directory.
2. Type the following to build the firmware (*bcm95340x-umplus.bin*).
   ```
   make clean; make target=umplus
   ```
3. Use flash programmer to program *bcm95340x-umplus.bin* into the flash. Currently, the BCM953406K and BCM953456K use Micro N25Q512A83 serial flash.

BCM5346X Platform

1. Go to the *$UM/systems/bcm95346x* directory.
2. Type the following to build the firmware (*bcm95346x-umplus.bin*).
   ```
   make clean; make target=umplus
   ```
3. Use flash programmer to program *bcm95346x-umplus.bin* into the flash. Currently, the BCM956270K uses Macronix MX25L12835FMI-10G serial flash.

Note: Features defined in *$UM/systems/$PLATFORM/include/configs/config_umplus.h* can be selectively added or removed before building the firmware. Check the string "Features defined below can be selectively add or removed before building the image" in *config_umplus.h*. *$PLATFORM* can be bcm95333x, bcm95340x, bcm95343x, or bcm95346x.

BCM5357X Platform

1. Go to the *$UM/systems/bcm95357x* directory.
2. Check EXT_PHY_LIST in *$UM/systems/bcm95357x/um_gen_phylibs.mk* for the external PHY drivers, and make sure that the PHY part number, which is used by the hardware board, is explicitly listed/defined.
3. Type the following to build the firmware (*bcm95357x-umplus.bin*).
   ```
   make clean; make target=umplus
   ```

   Note: Some vendor config examples about BCM5357X SKUs are placed at *$UM/systems/bcm95357x/vendor_config* for reference.

4. Use flash programmer to program *bcm95357x-umplus.bin* into the flash. Currently, the BCM953570K uses Micron MT25QL512ABB serial flash.

Build the Image File for UM-Basic

BCM5333X Platform

1. Go to the *$UM/systems/bcm95333x* directory.
2. Type the following to build the firmware (*bcm95333x-umdumb.bin*).
   ```
   make clean; make target=umdumb
   ```
3. Use flash programmer to program *bcm95333x-umdumb.bin* into the flash. Currently, the BCM953334K and BCM953394K use Micro N25Q256A13 serial flash.

BCM5340X Platform

1. Go to the *$UM/systems/bcm95340x* directory.
2. Type the following to build the firmware (*bcm95340x-umdumb.bin*).
   ```
   make clean; make target=umdumb
   ```
3. Use flash programmer to program *bcm95340x-umdumb.bin* into the flash. Currently, the BCM953406K and BCM953456K use Micro N25Q512A83 serial flash.

BCM5346X Platform

1. Go to the *$UM/systems/bcm95346x* directory.
2. Type the following to build the firmware (*bcm95346x-umdumb.bin*).
   ```
   make clean; make target=umdumb
   ```
3. Use flash programmer to program *bcm95346x-umdumb.bin* into the flash. Currently, the BCM956270K uses Macronix MX25L12835FMI-10G serial flash.

BCM5354X Platform

1. Go to the *$UM/systems/bcm95354x* directory.
2. Type the following to build the UM-Basic image (*bcm95354x-umdumb.bin*):
   ```
   make clean; make target=umdumb
   ```
3. By default, the command inserts the vendor config setting file *$UM/systems/ vendor_configs/ config.um.53547option6* into the output binary (*bcm95354x-umdumb.bin*).

   To change the setting from the default, modify the *$UM/systems/bcm95354x/Makefile* to replace */vendor_configs/config.um.53547option6* with your desired option setting file.

   For example, if you change the setting to run option 1, modify the Makefile as follows:
   ```
   yes | ../../tools/um_config_insert.pl -c ./vendor_configs/config.um.53547option1 -i
   ./bcm95354x-umdumb.bin -force
   ```

   Note: For reference, vendor config examples for BCM5354X options are located at *$UM/systems/ bcm95354x/vendor_config/*.

4. Use the flash programmer to program *bcm95354x-umdumb.bin* into the flash. Currently, both the BCM953547K and BCM953549K use Micron MT25QL512ABB serial flash.

BCM5357X Platform

1. Go to the *$UM/systems/bcm95357x* directory.
2. Check EXT_PHY_LIST in *$UM/systems/bcm95357x/um_gen_phylibs.mk* for the external PHY drivers, and make sure that the PHY part number, which is used by the hardware board, is explicitly listed/defined.
3. Type the following to build the firmware (*bcm95357x-umdumb.bin*).
   ```
   make clean; make target=umdumb
   ```

   Note: Some vendor config examples about BCM5357X SKUs are placed at *$UM/systems/bcm95357x/vendor_config* for reference.

4. Use flash programmer to program *bcm95357x-umdumb.bin* into the flash. Currently, the BCM953570K uses Micron MT25QL512ABB serial flash.

   Note: Features defined in *$UM/systems/$PLATFORM/include/configs/config_umdumb.h* can be selectively added or removed before building the firmware. Check the string "Features defined below can be selectively add or removed before building the image" in *config_umdumb.h. $PLATFORM* can be bcm95333x, bcm95340x, bcm95346x, bcm95354x, or bcm95357x.

Device Bootup

1. Use `flash programmer` to program the image (*bcm95333x.image, bcm95340x.image, bcm95343x.image* or *bcm95346x.image*) into the flash.
2. After the device boots-up, change the device's MAC address using CLI commands. Type `F->f`.
3. Enter the new MAC address (e.g., `00-10-18-55-01-02`), then reboot the device to allow the new MAC address to take effect.

Web Interface

2. If you do not know the device's IP address or the device does not have a console, you can use the web browser with Bonjour support to discover the device’s home page.

For Safari Web Browser

1. Open Safari.
2. Click **Show all bookmarks**.
3. Select Bonjour. You will see the device's HTTP service instance name under **Bookmarks**.

For Internet Explorer Web Browser

1. Install Bonjour for Windows (http://download.cnet.com/Bonjour-for- Windows/3000-18507_4-93550.html) to support Bonjour on Microsoft's Internet Explorer (IE) web browser.
2. Open the IE web browser and enable Bonjour from **Manage Add-ons**.
3. Click the device's HTTP service name from the left frame of Bonjour.

Firmware Upgrade

1. Use the web browser to connect to the device's home page.

3. Click **OK** when it asks to confirm the firmware operation.
4. Wait until the button becomes **Continue**, and then click it.
5. If you do not want to upgrade the firmware, click **Abort** (and skip the remaining steps).
6. To upgrade the firmware, choose the firmware image file (*bcm95333x-umweb.flash, bcm95340x-umweb.flash, bcm95343x-umweb.flash* or *bcm95346x-umweb.flash*) and click **Upgrade**.
7. Wait until **Firmware Upgrade Completed** appears, then click **Continue**.

Note: Do not close the browser or power off the device before the upgrade is complete. Otherwise, the firmware image or serial flash could become corrupted.

Note: The IP address might be changed after reboot. See "Web Interface" on page 25 for information on the IP address.

Generate Code for Web Pages

1. Prepare the HTML file, for example, *$UM/src/appl/web/html/left.htm*.
2. Go to the directory *um/tools/web* and copy file *left.html* to this folder.
3. Type the following to generate files *left_htm.c*, *left_htm.h,* and *sspmacro_feature.h* based on the HTML file.
   `perl sspgen.pl left.htm`
4. Manually copy generated files *left_htm.c* and *left_htm.h* to *$UM/src/appl/web/content*.
5. Manually add new defines SSPMACRO_FEATURE_XXX in the generated file *sspmacro_feature.h* into same file under directory *$UM/src/appl/web/content*.

Generate Code for Xcommands

1. Define command syntax and describe the syntax in the XML file. For example, *$UM/src/appl/xcommands/switch/config.xml*.
2. Go to the directory *$UM/systems/$PLATFORM*. *$PLATFORM* could be *bcm95333x, bcm95340x, bcm95343x* or *bcm95346x*.
3. Type the following to generate files *xccxt_global_builders.c*, *xccxt_global_enums.h*, *xccxt_global_handlers.c* and *xccxt_table_global.c* under directory *$UM/src/appl/xcommands/generated* based on the XML file:
   `make xcommands`
4. Manually copy the functions or paths in the generated files *xccxt_global_builders.c* and *xccxt_global_handlers.c* to the same files under callback directory *$UM/src/appl/xcommands/callback* and revise it.

Cisco-Like CLI Interface

A user can display a Cisco-like CLI shell by typing `x` under the prompt `CMD>`. The Cisco-like CLI provides two accounts: admin (password is “password”) and guest (no password). The admin account can execute all levels of commands. The guest account can execute guest-level commands only, which includes the show commands in our example. UM software provides a few examples of Cisco-like CLI commands for reference.

BCM5333X Platform

Figure 12 shows the front panel port numbers for the BCM953334K board.

Ports 1–24 are 1G copper mode with auto-negotiation.

2	4	6	8	10	12	14	16	18	20	22	24
1	3	5	7	9	11	13	15	17	19	21	23

Figure 13 shows the front panel port numbers for the BCM953394K board.

Ports 1, 3, 5, 7, 9, and 11 are 1G copper mode with auto-negotiation. Ports 25–28 are 10G fiber mode with forced speed. Ports 29–32 are 1G fiber mode with auto-negotiation, clause 37. Ports 2, 4, 6, 8, 10, and 12 are unused ports.

2	4	6	8	10	12								
1	3	5	7	9	11	25	26	27	28	29	30	31	32

BCM5340X Platform

Figure 14 shows the front panel port numbers for the BCM953406K board.

Ports 1–12 are 1G fiber mode with auto-negotiation, clause 37. Ports 13–24 are 10G fiber mode with forced speed.

1	3	5	7	9	11	13	15	17	19	21	23
2	4	6	8	10	12	14	16	18	20	22	24

Figure 15 shows the front panel port numbers for the BCM953456K board.

The left 16 ports are 1G copper mode with auto-negotiation. The right ports 1–8 are 1G fiber mode with auto-negotiation, clause 37. Right ports 9, 11, 13, and 15 are 10G fiber mode with forced speed. Right ports 10, 12, 14, 16, and 17–20 are unused ports.

2	4	6	8	10	12	14	16
1	3	5	7	9	11	13	15

1	3	5	7	9	11	13	15	17	19
2	4	6	8	10	12	14	16	18	20

BCM5346X Platform

Figure 16 shows the front panel port numbers for the BCM956270K board with BCM53460 SKU. Ports 1–4 are 1G fiber mode with auto-negotiation, clause 37. Ports 5–12 are 10G fiber mode with forced speed.

UART1

UART0

MDC2/
MDIO2

BCM5354X Platform

Figure 17 shows the front-panel port numbers for the BCM953547K. Figure 18 on page 31 shows the front-panel port numbers for the BCM953549K. Left ports 1 through 24 of the BCM953547K are 1G copper mode with auto-negotiation. Right ports 1 through 8 are SGMII ports. QGPHY5 (left ports 21 through 24) of BCM953547K and SGMII 0 (right port 1 through 4) are alternative ports. Likewise, QGPHY5 (left ports 5 through 8) of BCM953549K and SGMII 0 (right port 1 through 4) are alternative ports.

Use a strap to select the usage of QGPHY5 and SGMII 0, as described in Table 3. For example with option 1, if GPHY_SGMII_SEL[0] and GPHY_SGMII_SEL[1] are set to 0, QGPHY 5 is disabled, and SGMII 0 is enabled. As another example (option 6), if GPHY_SGMII_SEL[0] and GPHY_SGMII_SEL[1] are set to 1, QGPHY 5 is enabled, and SGMII 0 is disabled.

Some of the options, like option 3 and option 4, partially enable port 1 and port 2 of QGPHY 5 and port 3 and port 4 of SGMII 0. For the detailed strap setting toward each option and the enabled ports, see Table 3.

Note: The BCM53547 and BCM53548 devices can be used on the BCM953547K reference board, while the BCM53549 device is for the BCM953549K reference board only.

Table 3: QGPHY 5 and SGMII 0 Strap Settings

Option	GPHY_SGMII_SEL[1]	GPHY_SGMII_SEL[0]	QGPHY 5				SGMII 0			
			1	2	3	4	1	2	3	4
Option 1/7/13	0	0	–	–	–	–	S[a]	S	S	S
Option 2/8/14	1	1	S	S	S	S	–	–	–	–
Option 3/9/15	0	1	S	S	–	–	–	–	S	S
Option 4/10/16	0	1	S	S	–	–	–	–	S	S
Option 5/11/17	0	0	–	–	–	–	S	S	S	S
Option 6/12/18	1	1	S	S	S	S	–	–	–	–

a. S: Switch Port

BCM5357X Platform

BCM53570 contains four kinds of port macros, including SGMIIPX4, QTC, TSCE and TSCF. Unlike before, the front panel of BCM953570K does not have the front port numbers printed on it. Only the index numbers of each port macro instance are printed on the front panel instead. Figure 19 below shows the layout of the BCM953570K front panel.

BCM5357X supports BCM53570 option5_0 with 56 × 2.5G ports, 2 × 40G port plus 4 × 25G ports. SGMII0 ~ SGMII5 provide 24 × 2.5G capabilities, QTC 0 and QTC 1 provide 8 × 2.5G capabilities, and TSCE0, TSCE1, TSCE2, TSCE3 and TSCE5 provide 20 × 2.5G capabilities. TSCE4 and TSCE6 provide 2 × 40G capabilities. TSCF0 provides 4 × 25G capabilities. Other SKUs (port options) are in preview stage for the BCM5357X release.

Table 4: Port Mapping of BCM53570 Option 5_0

Front Panel	***Port Number***

Table 4: Port Mapping of BCM53570 Option 5_0 (Cont.)

Front Panel	***Port Number***

Dual image is enabled by default. To build the loader and firmware without dual image ability, remove CFG_DUAL_IMAGE_INCLUDED in *config_loader.h* and *config_umweb.h*.

Table 5 lists the start address of firmware 1 and firmware 2 partitions. See “Memory Layout ” on page 11 for details.

Flash Partition	***BCM5333X Platform***	***BCM5340X Platform***	***BCM5346X Platform***	***BCM5357X Platform***

The following is a simple test of enabling dual image through a firmware upgrade on the BCM5357x platform. The red information is related to dual image.

1. First, boot after using the flash programmer to program the um-3.5.0 image:

```
Quartz loader-3.5.0
Flash detected: N25Q512
Flash image is 1880572 bytes, chksum A4A9, version 3.5.0 for board BCM95357X
Load program at 0xF0600040...
Quartz umweb-3.5.0
Flash detected: N25Q512
devid = 0x8750, revid = 0x1
TX/RX support enabled.
Some of current saved settings are invalid. Loading factory defaults.....

CMD> D - Show dual image info
Image Version Active
-------------------------
1 3.5.0 Y
2 NA -
```

2. During the first firmware upgrade via the UM-Web:

```
Quartz loader-3.5.0
Flash detected: N25Q512
devid = 0x8570, revid = 0x1
TX/RX support enabled.
System IP : 192.168.0.4 netmask : 255.255.255.0 gateway : 0.0.0.0
CMD>
lport 12 (P:12,U:11), speed = 1000, duplex = 1, tx_pause = 1, rx_pause = 1, an = 1
IPv6 address: fe80:0:0:0:210:18ff:fe55:444b
Upgrading firware at partition 2 address 0xf0c00000
Flash image is version 3.5.0 for board BCM95357X
```

3. After the first firmware upgrade via UM-Web again:

```
Quartz loader-3.5.0
Flash detected: N25Q512
Flash image is version 3.5.0 for board BCM95357X
Flash image is 1880572 bytes, chksum A4A9, version 3.5.0 for board BCM95357X
Flash image is version 3.5.0 for board BCM95357X
Flash image is 1880572 bytes, chksum A4A9, version 3.5.0 for board BCM95357X
Load program at 0xF0C00040...

Quartz umweb-3.5.0
Flash detected: N25Q512

devid = 0x8570, revid = 0x1
TX/RX support enabled.

CMD> D - Show dual image info
Image   Version   Active
-------------------------
  1     3.5.0       N
  2     3.5.0       Y
```

4. During the second firmware upgrade:

```
Quartz loader-3.5.0
Flash detected: N25Q512
```

```
devid = 0x8570, revid = 0x1

TX/RX support enabled.
System IP : 192.168.0.4 netmask : 255.255.255.0 gateway : 0.0.0.0
CMD>
lport 12 (P:12,U:11), speed = 1000, duplex = 1, tx_pause = 1, rx_pause = 1, an = 1
IPv6 address: fe80:0:0:0:210:18ff:fe55:444b
Upgrading firmware at partition 1 address 0xf0600000
Flash image is version 3.5.0 for board BCM95357X
```

5. After the second firmware upgrade:

```
Quartz loader-3.5.0
Flash detected: N25Q512
Flash image is version 3.5.0 for board BCM95357X
Flash image is 1880572 bytes, chksum A4A9, version 3.5.0 for board BCM95357X
Flash image is version 3.5.0 for board BCM95357X
Flash image is 1880572 bytes, chksum A4A9, version 3.5.0 for board BCM95357X
Load program at 0xF0600040...

Quartz umweb-3.5.0
Build Date: Feb 16 2017
Flash detected: N25Q512

devid = 0x8570, revid = 0x1
TX/RX support enabled.

CMD> D - Show dual image info
Image   Version   Active
------------------------
  1     3.5.0       Y
  2     3.5.0       N
```

Table 6 lists the minimum storage requirements, based on theUM 3.6.1release.

Table 6: UM 3.6.1 Minimum Storage Requirements

	UM+ (*.bin)			***UM-Web (*.image)***			***UM-Basic (*.image)***
	BCM5333X	***BCM5340X/ BCM5346X***	***BCM5357X***	***BCM5333X***	***BCM5340X/ BCM5346X***	***BCM5357X***	***BCM5354X***
Minimum flash size (Includes one firmware only)	512 KB	1 MB	8 MB	–	–	–	2M
Minimum flash size (Includes one loader and one firmware)	–	–		1 MB	2 MB	16 MB	–
Minimum flash size with dual image (Includes one loader and two firmware)	–	–		2 MB	4 MB	16 MB	–

The Unmanaged Software provides a tool to insert the configuration into the precompiled firmware image (*.image) as well as a set of APIs to retrieve the configuration at runtime. The firmware image may perform different settings in the initialization based on the configuration.

Configuring the APIs

`$UM/include/$(CPU)/sal_config.h` reveals the APIs to get the value of certain configuration items which can be regarded as a specific type, such as uint8, uint16, and port bitmap.

The caller should give the name of the configuration item and use the proper function to get the configure value. If the given configuration item is not found at configuration space or an error occurs when getting the configuration, the APIs will return an error and not affect the content of the output pointer.

Function prototypes for the firmware APIs are shown below.

```
sys_error_t sal_config_pbmp_get(const char *name, pbmp_t *p)
int sal_config_bytes_get(const char*name, uint8* buf, int len)
sys_error_t sal_config_uint8_get(const char*name, uint8* byte)
sys_error_t sal_config_uint16_get(const char*name, uint16* hword)
sys_error_t sal_config_uint32_get(const char*name, uint32* word)
```

Using the Configuration Insert Tool

`$UM/tool/um_config_insert.pl` is the configuration insert tool. When given the firmware image and the configuration file, this tool will translate this configuration file into binary and insert it into the firmware image. An example of the tool is shown below.

```
$UM/tool/um_config_insert.pl -i bcm95340x.image -c config.um -force
```

`bcm95340x.image` is the firmware image. `config.um` is the configuration file. The option, `-force`, will insert the configuration regardless of the existence of the previous inserted configuration. For more information, execute `$UM/tool/um_config_insert.pl -h` to see the usage of the insert tool.

Usage: `um_config_insert.pl -image <image_file> [options]`:

- -image, `-i`: This specifies the image file where the configuration will be inserted.
- -config, `-c`: This optionally specifies the configuration file name. The default name is `config.um`.
- -force: This forces the previous configuration in the image file to be overwritten.
- -verbose: This shows the debug log.
- -h: This shows the configuration file usage.

A sanity check is performed at the beginning of tool execution to ensure that the configuration insert procedure is correct. The sanity check includes:

1. Checking the configuration file syntax.
2. Verifying the availability of the firmware image space for configuration storage.

3. Checking if board_name is defined in configuration file. The board_name in the configuration space should be equal to the CFG_BOARDNAME defined in Makefile.

$UM/tools/config.um

```
#
# Vendor Configuration
#
# Each entry in the file consists of a single line of the form:
# <Parameter>=<Value>
# UM software provides a tool (um_config_insert.pl) to insert the Vendor
# configuration(config.um) into the precompiled firmware image (*.image).
# Then the firmware image may do different setting in the intialization
# based on the configuration.
#
# Usage to insert Vendor Configuration into the precompiled firmware
#      um_config_insert.pl -image  <image_file> [options]
#          -image, -i: specify the image file where the config will be inserted
#          -config, -c: optionally specify config file name,
#                           default name is "config.um"
#          -force: force to overwrite previous configuration in the image file
#          -verbose: show more debug log
#          -generate, -g: only generate the config binary file for web update
#          -h: show usage
#

#     Board Name : Board name checking will be enabled if board_name is set.
#                  For board name checking, it would check whether it is equal
#                  to the value of CFG_BOARDNAME defined in
#                  system/bcmxxxxx/Makefile.
#board_name=BCM95340X

#     SKU Option :
#
#         - Purpose: Specifies option of SKU.
#
#         - Syntax: value is from X[_Y]
#                  X is option number which is a positive integer.
#                  Y is sub-option number which is a integer greater or equal to 0.
#                  The sub-option number, Y only is supported in BCM95357X platform
#                  Please get sku option number from Programmer's Register Reference
#                  Guide.
#sku_option=2

#     Serial LED
#         - led_option : Option of serial LED micro code, value is from 1 ~ 3.
#             1: Left LED : Link              Right LED : TX/RX activity
#             2: Left LED : TX/RX activity    Right LED : Link
#             3: Customer LED uCode
#         - led_program : Customer LED uCode
#led_option=3
#led_program=led.hex
#led_1_program=led1.hex

#     Parallel LED Setting
#         - Overide the setting of register LED_CONTROL and LED_SELECTOR
```

```
#           for external PHY.
#phy_led1_mode=0xF
#phy_led2_mode=0xF
#phy_led3_mode=0xF
#phy_led4_mode=0xF
#phy_led_ctrl=0x8
#phy_led_select=0x0

#     Reset Button
#         - reset_button_enable :  0 (disable) / 1 (enable reset button feature)
#         - reset_button_gpio_bit : Set GPIO bit for reset button, value is
#                                   from 0 ~ 7.
#         - reset_button_polarity : 0 (active low) / 1 (active high)
#reset_button_enable=1
#reset_button_gpio_bit=4
#reset_button_polarity=1


#     Valid Ports:
#
#     - Purpose: Disable/enable ports through valid ports.
#
#     - Syntax: Valid_logical_ports=<port list>
#               Where <port list> could be 1,2,4,5-10,11

#valid_logical_ports=2-25

#
#    Port Speed: Specifies forced port speed or max port speed in the autonegotioation
#
#     - Syntax: speed_<speed>_logical_ports=<port list>
#               Where <port list> could be 1,2,4,5-10,11,
#                     <speed> could be 1000, 2500, 5000, 10000, 25000, 40000, 50000
#

#speed_1000_logical_ports=2-25
#speed_2500_logical_ports=2-25
#speed_5000_logical_ports=2-25
#speed_10000_logical_ports=2-25
#speed_25000_logical_ports=2-25
#speed_40000_logical_ports=2-25
#speed_50000_logical_ports=2-25


#
#    Auto-negotioation
#
#       - Purpose: Enable/disable auto-negotioation and specifies the mode of auto-negotioation
#
#

#phy_an_logical_ports=2-25
#phy_cl73_logical_ports=2-25
#phy_cl37_logical_ports=2-25
```

```
#     SGMIIPX4 interface
#
#         - Purpose: Specifies interface mode of SGMIIPX4.
#
#         - Syntax: sgmiipx4_interface[_<core_num>][_<lane_num>]=<value>
#               <value>: 1: SGMII mode
#                        2: Fiber mode
#          <core_num> is optional and indicates the core number of SGMIIPX4 instance.
#          <lane_num> is optional and indicates the lane number of a SGMIIPX4 instance.
#
#         - Note: <core_num> and <lane_num> are only valid on BCM95357X platform
#
#         - Example:
#          Per system setting: sgmiipx4_interface=1 means setting every SGMIIPX4s in system to
#             mode 1
#          Per core setting: sgmiipx4_interface_0=2 means setting SGMIIPX4 Core 0  to mode 2
#          Per lane setting: sgmiipx4_interface_0_1=3 means setting SGMIIPX4 lane 1 of SGMIIPX4
#             Core 0 to mode 2
#
#sgmiipx4_interface=1
#
#     QTC interface
#
#         - Purpose: Specifies interface mode of QTC.
#
#         - Syntax: qtc_interface[_<core_num>][_<lane_num>]=<value>
#               <value>: 1: QSGMII mode
#                        2: SGMII mode
#                        3: Fiber mode
#              <core_num> is optional and indicates the core number of QTC instance.
#               <lane_num> is optional and indicates the lane number of a QTC instance.
#
#         - Note: <core_num> and <lane_num> are only valid on BCM95357X platform
#
#         - Example:
#              Per system setting: qtc_interface=1 means setting every QTCs in system to mode 1
#              Per core setting: qtc_interface_0=2 means setting QTC Core 0 to mode 2
#              Per lane setting: qtc_interface_0_1=3 means setting lane 1 of QTC Core 0 to mode 3
#
#qtc_interface=1
#
#     TSCE interface
#
#         - Purpose: Specifies interface mode of TSCE.
#
#         - Syntax: tsce_interface[_<core_num>][_<lane_num>]=<value>
#               <value>: 1: SGMII/XFI mode
#                        2: Fiber mode
#                        3: XAUI (only valid on BCM5357X platform)
#
#              <core_num> is optional and indicates the core number of TSCE instance.
#              <lane_num> is optional and indicates the lane number of a TSCE instance.
#
#         - Note: <core_num> and <lane_num> are only valid on BCM95357X platform
#
```

```
#         - Example:
#              Per system setting: tsce_interface=1 means setting every TSCEs in system to mode 1
#              Per core setting: tsce_interface_0=2 means setting TSCE Core 0  to mode 2
#              Per lane setting: tsce_interface_0_1=3 means setting lane 1 of TSCE Core 0 to
#                  mode 2

#tsce_interface=1


#     TSCF interface
#
#         - Purpose: Specifies interface mode of TSCF.
#
#         - Syntax: tsce_interface[_<core_num>][_<lane_num>]=<value>
#                <value>: 1: SGMII/XFI mode
#                         2: Fiber mode
#              <core_num> is optional and indicates the core number of TSCF instance.
#              <lane_num> is optional and indicates the lane number of a TSCF instance.
#
#         - Note: <core_num> and <lane_num> are only valid on BCM95357X platform
#
#         - Example:
#              Per system setting: tscf_interface=1 means setting every TSCFs in system to mode 1
#              Per core setting: tscf_interface_0=2 means setting TSCF Core 0  to mode 2
#              Per lane setting: tscf_interface_0_1=3 means setting lane 1 of TSCF Core 0 to
#                  mode 2

#tscf_interface=1


#     VIPER interface
#         - viper_interface : Different modes for viper interface, value is 1 or 2.
#             1: SGMII mode
#             2: Fiber mode
#viper_interface=1

#     Config "static IP address" or dhcp
#         - ifconfig=IP_ADDR/NETMASK/GATEWAY or ifconfig=dhcp
#
#           For example1 ,ifconfig=192.168.0.239/255.255.255.0/192.168.0.254
#           For example2 ,ifconfig=dhcp, it is same as if ifconfig is not configured
#
#ifconfig=192.168.0.239/255.255.255.0/192.168.0.254

#     Lossless Mode
#         - By default, 53570 will be configured to accept the maximum number of
#           packets per port, but may drop them if resources are oversubscribed due
#           to activity from other ports.  If lossless mode is enabled, 53570 will
#           instead be configured to accept packets only if sufficient processing
#           resources are guaranteed for all ports.  This may decrease overall
#           throughput, but no accepted packets will be dropped.
#             0: lossy mode
#             1: lossless mode
#mmu_lossless=1
```

```
#   mmu_lossless_logical_ports
#       The BCM5354X platform has support for a maximum of 4 MMU lossless ports.
#       When Lossless Mode is enabled, the user may also need to specify which ports are to be in
#       lossless mode. For example, value 0x3C is the logical port's port bit map,
#       which represents 2b'0011,1100 in binary. That means logical port 2, 3, 4, 5 would be
#       configured in lossless mode. For the user port and logical port number mapping,
#       refer to the BCM5354X Port Mapping section in the UM Porting Guide. If no value is
#       specified, the default value is 0x3C when in MMU Lossless mode.
#mmu_lossless_logical_ports=0x3C
```

Note: For logical port numbers, refer to the “Port and PHY Configuration” section of document “Unmanaged Software Porting Guide” (“References”) for detailed procedures.

Example 1

The following section provides an example to insert the configuration `valid_logical_ports=5-8` into the release image `bcm95346x.image`.

1. Go to the tools directory `$UM/tools` and copy the `bcm95346x.image` from the image directory.
2. Add `valid_logical_ports=5-8` to config.um.

```
In config.um:
#     Valid port list:
#
#        Description: specifies valid ports
#
#        Syntax: valid_logical_ports=<port list> where <port list> could be 1,2,4,5-10,11
valid_logical_ports=5-8
```

3. Use the `um_config_insert.pl` tool to insert the configuration into the precompiled firmware image (*.image). The output file will be the same name *.image. It is `bcm95343x.image` in this example.

```
um_config_insert.pl -i bcm95343x.image
======================================================================
UM signature is found at bcm95346x.image
Detail parameters of image are shown below
Board name = BCM95346X
Firmware Type = loader
Version = 0x3040000
Config base address = 0xfd000
Config maximun size = 0x1000
======================================================================
Perform config.um Parsing ...
valid_logical_ports=>5-8
======================================================================
Config inserted successfully
======================================================================
```

4. Use the flash programmer to program the output image `bcm95346x.image` into serial flash.

5. Use the **F->I** command to list the configuration after the system boots up. It is for debug only.

```
CMD> F - Flash utilities
f - Write factory mac address
n - Write the serial number
d - Dump the mac address and serial number
s - Set nvram variable
g - Get nvram variable
l - list all nvram variable
r - Remove nvram variable
c - Commit nvram variable bindings
Enter your choice: l
valid_logical_ports=5-8
```

Example 2

The following section provides an example of how to use an LED uCode. In this case, two configurations (`led_option=3` and `led_program=led_example.hex`) must be inserted into the precompiled image bcm95346x.image.

1. Prepare the LED uCode (*.hex).
   a. Prepare an `.asm` file. It is `led_example.asm` in this case.
   b. Generate an assembler that can convert the `.asm` file into .hex files.

```
make ledasm' under $UM/tools/led/tools
```

   c. Use assembler to covert the `.asm` file into .hex files. The `.asm` file needs to be copied to `$UM/tools/led/tools`.

```
ledasm led_example' under $UM/tools/led/tools
```

2. Go to the tools directory `$UM/tools` and copy the `bcm95346x.image` from the image directory.
3. Copy the `.hex` file generated in Step 1 to `$UM/tools` also.
4. Add `led_option=3` and `led_program=led_example.hex` to config.um. In config.um:

```
# Serial LED
# - led_option : Option of serial LED micro code, value is from 1 ~ 3.
# 1: Left LED : Link Right LED : TX/RX activity
# 2: Left LED : TX/RX activity Right LED : Link
# 3: Customer LED uCode
# - led_program : Customer LED uCode
led_option=3
led_program=led_example.hex
```

5. Use the `um_config_insert.pl` tool to insert these two configurations into the precompiled firmware image (*.image). The output file will be the same name *.image. It is bcm95346x.image in this example.

```
um_config_insert.pl -i bcm95346x.image
=====================================================================
UM signature is found at bcm95346x.image
Detail parameters of image are shown below
Board name = bcm95346x
Firmware Type = loader
Version = 0x3030000
Config base address = 0xfd000
Config maximun size = 0x1000
=====================================================================
```

```
=====================================================================
Perform config.um Parsing ...
led_option=>3
led_program=>led_example.hex
led_example.hex is loaded
=====================================================================
Config inserted successfully
=====================================================================
```

6. Use the flash programmer to program the output image `bcm95346x.image` into serial flash.
7. Use the `F->l` command to list the configuration after the system boots up. It is for debug only.

```
CMD> F - Flash utilities
f - Write factory mac address
n - Write the serial number
d - Dump the mac address and serial number
s - Set nvram variable
g - Get nvram variable
l - list all nvram variable
r - Remove nvram variable
c - Commit nvram variable bindings
Enter your choice: l
led_option=3
led_program=led_example.hex
```

The GENERIC-SERIAL-LED mechanism provides a unified interface to control the serial LED processor and the serial LED circuit on different boards. This mechanism supports the BCM5357X device. GENERIC-SERIAL-LED and handles the port status remapping with its own remapping rule.

In general, the features of GENERIC-SERIAL-LED are as follows:

- Supports up to three LEDs per port.
- Supports up to four user-defined LED colors (up to four kinds of color).
- Supports four kinds of generic LED behaviors such as LINK, ACTIVITY, BLINK, and FORCE_ON for each LED.
- Supports a configurable timer for flexible BLINK periods and ACT extension time.

Note: For more information about the GENERIC-SERIAL-LED mechanism, refer to the *Unmanaged Software Porting Guide* (see "References").

802.1Q VLAN

Click the VLAN hyperlink to display the VLAN web page. Choose **IEEE 802.1Q VLAN** to start setting the 802.1Q VLAN. The default setting is 802.1Q VLAN for UM-Web. By changing the VLAN type, the configuration related to the VLAN will be erased.

Creating a New 802.1Q VLAN

2. Type the new VLAN ID in **New VLAN ID** and select the ports belonging to this new VLAN with a tag or untag attribute. Then click **Create**.

4. Click on **VLAN ID 20** to change the member ports of this VLAN or to remove this VLAN.

Modify or Delete an 802.1Q VLAN

To change member ports of a VLAN, select the port number to select the desired state: Not member, Tag member, or Untag member.

To delete a VLAN, select **VLAN_ID** and then click **Remove This VLAN**.

Port-Based VLAN

Click the VLAN hyperlink to display the VLAN web page. Choose **Port-Based VLAN** to start setting the Port-Based VLAN. By changing the VLAN type, the configuration related to the VLAN will be erased.

Create a New Port-Based VLAN

Modify or Delete a Port-Based VLAN

System Page

The system page is the first page after the login page.

Table 7: System Page Parameter Description

Parameter/Button	*Description*
	upgrade. After confirming this request, the device will be rebooted to bootloader for upgrade process.

Port Status and Configuration Page

Port functions provide an overview of the system. The Port Status and Configuration page displays each port's status, such as link status, speed, duplex, flow control, and auto-negotiation.

Table 8: Port Status and Configuration Page Parameter Description

Parameter/Button	*Description*
Speed/Duplex	Indicates the duplex and speed of the port when it is linked. If the port link is down, there is no status display.

Statistics Page

Table 9: Statistics Page Parameter Description

Parameter/Button	*Description*

VLAN Page—IEEE 802.1Q VLAN

Table 10: VLAN Page — IEEE 802.1Q VLAN Parameter Description

Parameter/Button	*Description*

VLAN Page—Port-Based VLAN

Table 11: VLAN Page—Port-Based VLAN Parameter Description

Parameter/Button	*Description*

Trunk Setting Page

Trunking aggregates multiple ports into a trunk. It uses a distribution algorithm to balance traffic between trunk members.

Table 12: Trunk Setting Page Parameter Description

Parameter/Button	*Description*

Mirror Setting Page

Mirroring monitors traffic from some given ports to a single mirror-to port. Ingress and/or egress traffic is copied from the mirroring port to the mirror-to port.

Table 13: Mirror Setting Page Parameter Description

Parameter/Button	*Description*
Mirror	Specifies an ingress/egress mirror port to which ingress/egress traffic will be mirrored.

QoS Setting Page-IEEE 802.1P QoS

IEEE 802.1P QoS operation sets the queues and priority relationship. Currently, UM only supports the default IEEE 802.1P setting value.

Table 14: QoS Setting Page—IEEE 802.1P QoS Parameter Description

Parameter/Button	*Description*
	queues.

QoS Setting Page-Port-Based QoS

Port-Base QoS operation sets the port and queue relationship, selects the scheduling method for these queues and {1, 2, 4, 8} weight for mapping from low to high queue.

Table 15: QoS Setting Page—Port-Based QoS Parameter Description

Parameter/Button	*Description*
	queues.

Port Rate Page

The Port Rate page is used for rate limit and storm control.

Table 16: Port Rate Page Parameter Description

Parameter/Button	*Description*
Egress Rate	Rate limitation of outgoing traffic in this port. See"Per Port Rate Limit" on page 64.

Loop Detection Page

Multicast Page

IGMP is a standard defined in RFC1112 for IGMPv1, and in RFC2236 for IGMPv2. IGMP specifies how a host can register to receive specific multicast traffic from a multicast server. IGMP snooping can reduce multicast traffic at Layer 2 by configuring Layer 2 LAN ports dynamically to forward multicast traffic only to those ports that are configured to receive it.

Table 17: Multicast Page Parameter Description

Parameter/Button	*Description*
	Enable blocking of unknown multicast address when selected.

Cable Diagnostic Page

Access Control Page

You can use this page to limit access to the web GUI to a designated source IP address and to a single user only.

Password Page

The references in this section may be used in conjunction with this document.

Note: Broadcom provides customer access to technical documentation and software through its Customer Support Portal (CSP) and Downloads and Support site.

For Broadcom documents, replace the “xx” in the document number with the largest number available in the repository to ensure that you have the most current version of the document.

Document (or Item) Name	*Number*	*Source*
Broadcom Items		
[1] Unmanaged Software Porting Guide	Unmanaged-PG1*xx*-R	Broadcom CSP

Revision	Date	Change Description
		• Figure 2: “Code Directory,” on page 9 (Added BCM95354X platform) • “Toolchain” on page 16 (Added BCM5354X information) • Removed vendor config step and changed Micron serial flash part number in: – “Build the Image File for UM-Web” instructions for the “BCM5357X Platform” on page 19 – “Build the Image File for UM+” instructions for the “BCM5357X Platform” on page 21 – “Build the Image File for UM-Basic” instructions for the “BCM5357X Platform” on page 23 • “BCM5357X Platform” on page 31 in “Port Configuration” • Table 6: “UM 3.6.1 Minimum Storage Requirements,” on page 35 (Added UM-Basic column) • “$UM/tools/config.um” on page 37 (Added `mmu_lossless_logical_ports` information) **Added:** • “Flash Partitions/Memory Layout for UM-Basic (BCM5354X)” on page 15 • “BCM5354X Platform” on page 22 to “Build the Image File for UM-Basic” • “BCM5354X Platform” on page 30 to “Port Configuration”

Revision	*Date*	*Change Description*
Unmanaged-SWUM105	09/20/17	**Updated:** • "Switch Management" on page 6 • Figure 2: "Code Directory," on page 9 • Table 2: "UM Files," on page 10 • "Build the MDK PHY Library" on page 16 • "Dual Image" on page 31 • "$UM/tools/config.um" on page 36 • "Example 1" on page 40 • "Example 2" on page 40 **Added:** • "Flash Partitions/Memory Layout for UM-Web (BCM5357X)" on page 14 • "Flash Partitions/Memory Layout for UM+ (BCM5357X)" on page 15 • BCM5357X to "Toolchain" on page 16 • "BCM5357X Platform" on page 19 in "Build the Image File for UM-Web" • "BCM5357X Platform" on page 21 in "Build the Image File for UM+" • "BCM5357X Platform" on page 23 in "Build the Image File for UM-Dumb" • "BCM5357X Platform" on page 30 in "Port Configuration" • BCM5357X to Table 4: "Start Address of Firmware 1 and Firmware 2 Partitions," on page 31 • BCM5357X to Table 5: "UM 3.6.1Minimum Storage Requirements," on page 34 **Removed:** • Flash Partitions/Memory Layout for UM-Web (BCM5343X) from "Memory Layout " on page 11 • Flash Partitions/Memory Layout for UM+ (BCM5343X) from "Memory Layout " on page 11 • BCM5343X from "Toolchain" on page 16 • BCM5343X from "Build the MDK PHY Library" on page 16 • BCM5343X Platform from "Build the Image File for UM-Web" on page 17 • BCM5343X Platform from "Build the Image File for UM+" on page 20 • BCM5343X Platform from "Build the Image File for UM-Dumb" on page 22 • BCM5343X Platform from "Port Configuration" on page 28 • BCM5343X Platform from Table 4: "Start Address of Firmware 1 and Firmware 2 Partitions," on page 31
Unmanaged-SWUM104-R	05/19/16	Document updated for the 3.4.0 release.
Unmanaged-SWUM103-R	01/19/16	Document updated for the 3.3.X release.

Revision	*Date*	*Change Description*
Unmanaged-SWUM102-R	07/30/15	Document updated for the 3.3.0 release.
Unmanaged-SWUM101-R	02/23/15	Document updated for the 3.2.2 release.
Unmanaged-SWUM100-R	12/18/14	Initial release

Corporate Headquarters: San Jose, CA

Unmanaged-SWUM106 January 24, 2018