Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

AN214935

OTP Programming and NVRAM Development Process for SDIO Embedded WLAN Devices

Associated Part Family: CYW4330

The Cypress CYW4330 is a single-chip Secure Digital I/O (SDIO) device intended for embedded applications. The CYW4330 supports IEEE 802.11 a/b/g/n. The device has a One-Time Programmable (OTP) nonvolatile memory for storing board-specific information such as Product ID, Manufacturer ID, MAC address, and more.

Contents

1 About this Document

1.1 Purpose and Audience

The Cypress CYW4330 is a single-chip Secure Digital I/O (SDIO) device intended for embedded applications. The CYW4330 supports IEEE 802.11 a/b/g/n. The device has a One-Time Programmable (OTP) nonvolatile memory for storing board-specific information such as Product ID, Manufacturer ID, MAC address, and more.

The OTP memory content and an NVRAM file (referred to throughout this document as the nvram.txt file) combine to create a complete Card Information Structure (CIS) that hosts use to bring up Cypress SDIO embedded devices.

For designs where the host and device are permanently connected together, which is typically done with a hard-wired SDIO interface, programming the OTP memory in production is optional. It is equally acceptable to store all NVRAM parameters in host firmware and keep the OTP blank in production. For devices that may be installed on different hosts, the OTP is programmed to protect the unique MAC address and to prevent end-users from altering power control parameters (such as maximum output power and other PA parameters).

For host platforms running the Linux® or Windows® XP operating system, it is not necessary to program the OTP memory during board bring up and hardware tuning. Instead, store all required CIS information in the nvram.txt file. Although OTP programming is not required for devices used on these host operating systems, nvram.txt file development is still required. Developing an nvram.txt file is a secondary purpose of this document.

Note: To use this application note, users must retrieve the following from Cypress's Customer Support Portal, contacting the cypress.com/support if necessary:

- A CYW4330 design package, which contains:
  - A chip-specific development board, board schematic, bill of materials, and layout
  - An nvram.txt template file
- A Windows XP or Linux device driver for the relevant SDIO device
- Cypress Transmit Signal Strength Indicator (TSSI) calibration tools

1.2 Cypress Part Numbering Scheme

Cypress is converting the acquired IoT part numbers from Broadcom to the Cypress part numbering scheme. Due to this conversion, there is no change in form, fit, or function as a result of offering the device with Cypress part number marking. The table provides Cypress ordering part number that matches an existing IoT part number.

Table 1. Mapping Table for Part Number between Broadcom and Cypress

Broadcom Part Number	Cypress Part Number
BCM4330	CYW4330

1.3 Acronyms and Abbreviations

In most cases, acronyms and abbreviations are defined upon first use. For a more complete list of acronyms and other terms used in Cypress documents, go to: http://www.cypress.com/glossary.

2 IoT Resources

Cypress provides a wealth of data at http://www.cypress.com/internet-things-iot to help you to select the right IoT device for your design, and quickly and effectively integrate the device into your design. Cypress provides customer access to a wide range of information, including technical documentation, schematic diagrams, product bill of materials, PCB layout information, and software updates. Customers can acquire technical documentation and software from the Cypress Support Community website (http://community.cypress.com/).

3 Introduction

The initial state of all OTP bits in an unprogrammed device is 0. Individual bits can be programmed to 1, but once programmed, they can never be reprogrammed back to 0. The entire OTP array can be programmed in a single write cycle using wl commands provided with the SDIO driver. Alternatively, multiple write cycles can be used to selectively program specific fields, but only bits that are still in the 0 state can be programmed to the 1 state during each programming cycle.

Because the OTP programming process is irreversible, Cypress recommends that board designers finalize all parameters before programming the OTP memory. Boards should be tested using the editable nvram.txt file. The nvram.txt parameters can be loaded into on-chip RAM, allowing the chip to be tested even if the OTP memory has not yet been programmed. This method lets board designers tune RF components and alter critical parameters while testing boards using different versions of the nvram.txt file. After the critical parameters have been finalized, they can be programmed into the SDIO device OTP memory space.

Note:

- For designs in which the host and device are permanently connected together, which is typically done with a hardwired SDIO interface, programming the OTP memory in production is optional. It is equally acceptable to store all NVRAM parameters in host firmware and keep the OTP blank in production. For devices which may be installed on different hosts, program the OTP to protect the unique MAC address and to prevent end-users from altering power control parameters such as maximum output power and other PA parameters.
- If a parameter is present in both the on-chip OTP and the nvram.txt file, the value from the OTP will override the value from the nvram.txt file; the WLAN driver will ignore the corresponding value in the nvram.txt file.

This document describes the method for creating an nvram.txt file and using the file to test a new board design, optimize all nvram values, and program the OTP. The document contains:

- An NVRAM Content Development and OTP Programming Flow
- SDIO Windows XP Driver Installation
- Information on Customizing the nvram.txt File Template
- Information on Finalizing the nvram.txt File
- An OTP Programming Procedure for an SDIO device

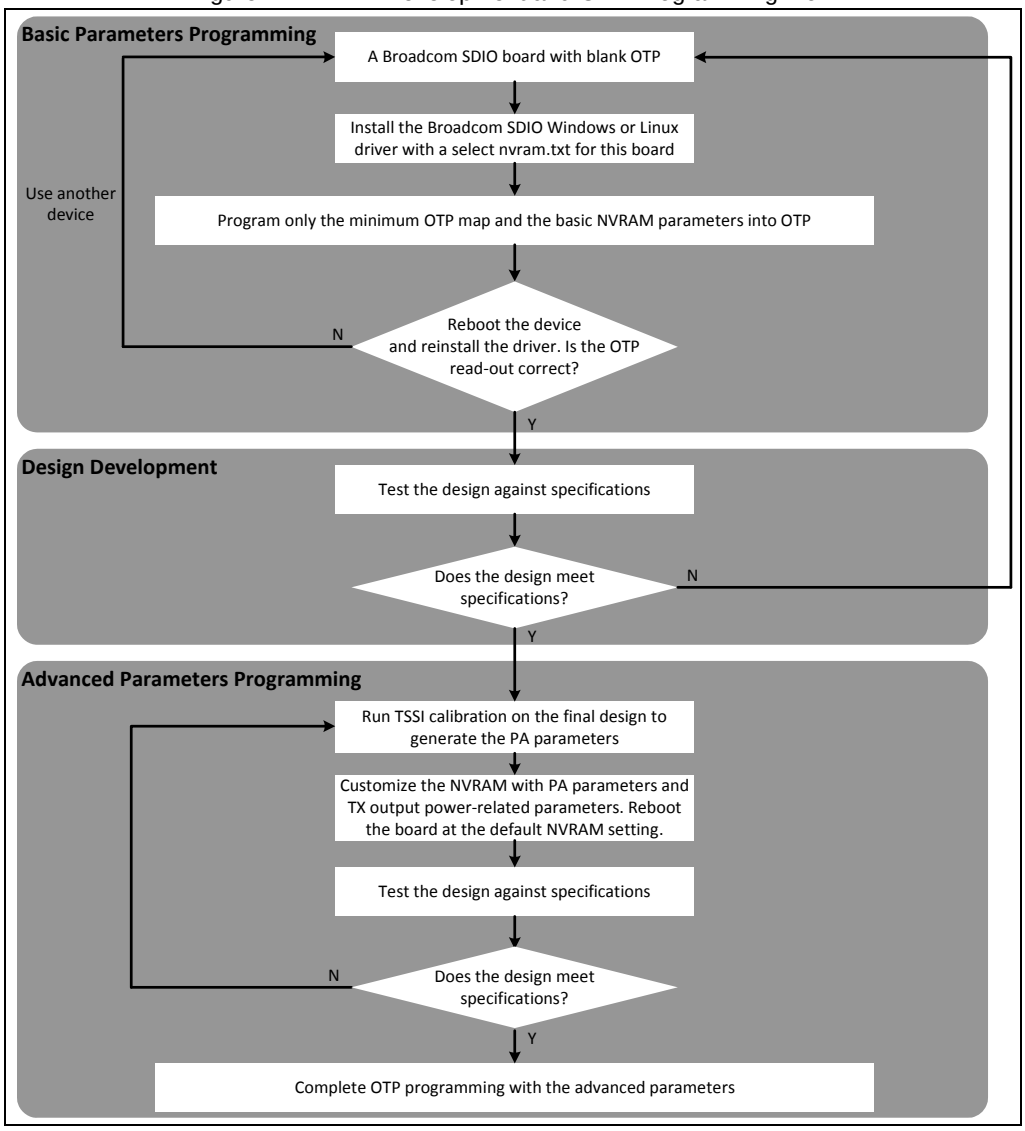
Note: Due to the irreversible OTP programming process, board development should be done on boards with blank

OTP memory spaces using the parameters in the editable nvram.txt file. Do not program the OTP memory until the contents of the nvram.txt file have been verified.

4 NVRAM Content Development and OTP Programming Flow

The nvram.txt file content development and the OTP programming flow is shown in Figure 1. Parameters in the nvram.txt can be divided into two groups: basic parameters and advanced parameters. Pertinent OTP programming details for each phase can be found in OTP Programming Procedure .

Figure 1. NVRAM Development and OTP Programming Flow



Note: The OTP programming flow shown in Figure 1 is only used during the development stages of the program on small quantities of boards. Once this process is complete and a “golden” nvram.txt file or OTP file is established, the development phase can be bypassed, and the programming can be done in high volume for mass production, following the correct manufacturing procedure defined by each manufacturer.

5 SDIO Windows XP Driver Installation

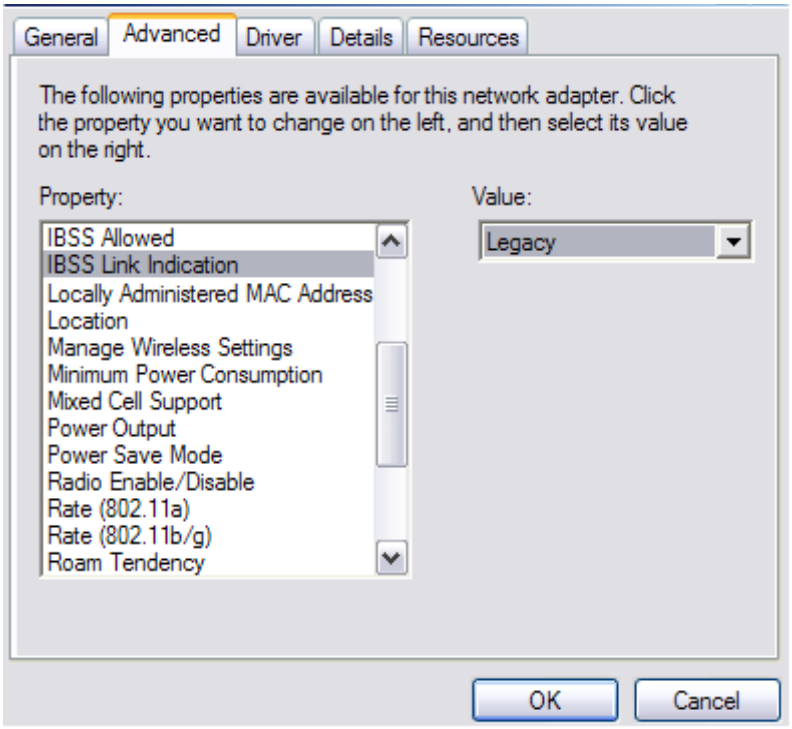
A preliminary default version of the nvram.txt file for each Cypress reference board type and board revision is released with the hardware reference design package available from the docSAFE tab of Cypress's customer support portal (cypress.com/support). Typically the file is named after the board it supports, such as bcm94330sdg.txt. The content in this default NVRAM file may likely change as the design goes through testing and tuning during the development stage.

Note: In development environments where previous drivers have been installed, it may be necessary to uninstall a previously installed driver before proceeding with driver installation. If so, refer to Appendix A: "Driver Removal," on page 13.

5.1 Driver Installation

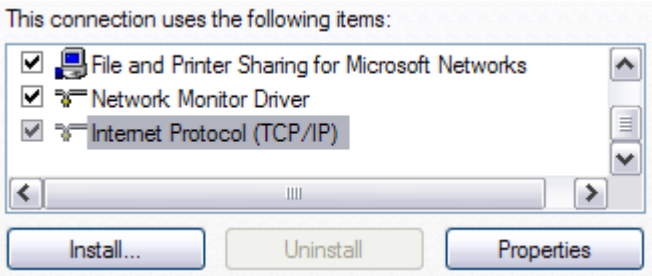
To install an SDIO device driver:

1. Rename the NVRAM file to "nvram.txt" and copy it to C:\Windows\system32\drivers\
2. Turn off the power of a test Windows XP-based PC.
3. Install the adapter into the PC.
4. Turn PC power on and allow time for Windows XP to start.
5. Start the Windows Device Manager:
   a. Click the **Start** button  .
   b. Right-click **My Computer**, and then click **Manage**.
   c. In the left pane of **Computer Management** , double-click **Device Manager**.
6. In the right pane of **Computer Management**, right-click **Network adapters**, and click **Scan for hardware changes**.
7. Follow the Windows on-screen instructions to install the SDIO device driver.
8. Double-click on the newly installed network adapter to view the adapter properties.
9. On the **Advanced** tab of **Adapter Properties**, set the **IBSS Link Indication** property to **Legacy**, set the **IBSS 54g™ Mode** property to **54g-Auto**, and then click **OK**.

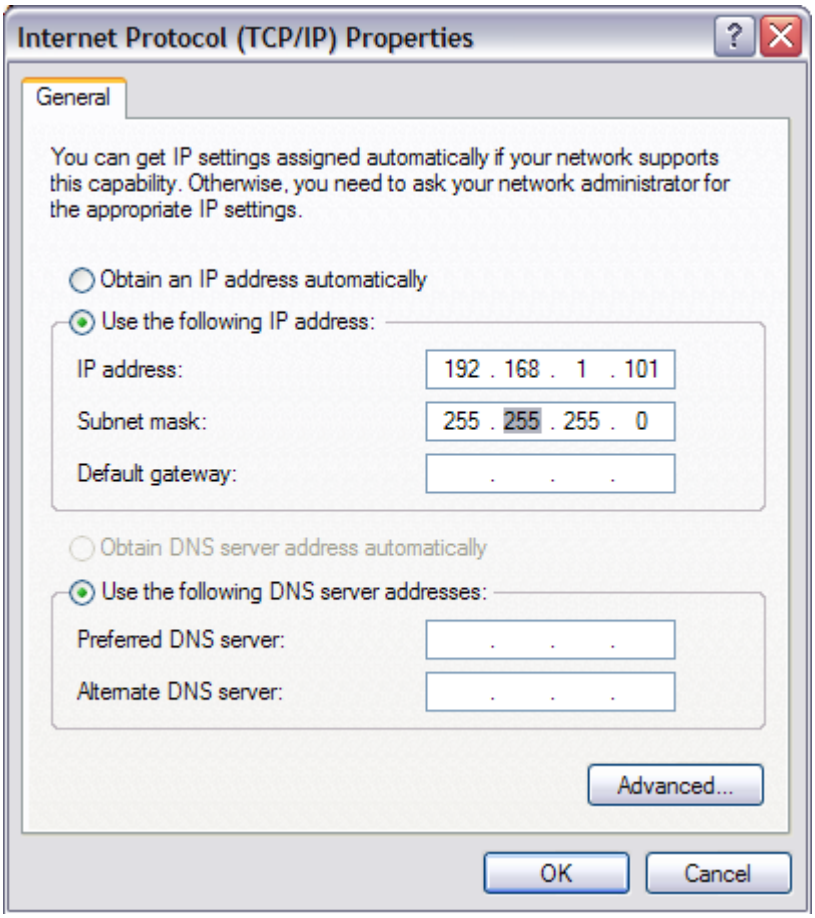


To set the static IP address for the 802.11g SDIO WLAN adapter:

1. Click the **Start** button, then select **Control Panel**.
2. In the Control Panel, double-click **Network Connections**.
3. Right-click **Wireless Network Connection**, then select **Properties**.
4. In the **Wireless Network Connection Properties** window, click **Internet Protocol (TCI/IP),** then click the **Properties** button .



5. Select the **Use the following IP address** option.
6. Set the IP address to **192.168.1.101** and the Subnet mask to **255.255.255.0**, and then and click **OK**.



To copy the WLAN test tools to enable driver test commands:

1. Copy wl.exe to C:\Windows\system32\
2. Copy brcm_wlu.dll to C:\Windows\system32\

6 Customizing the nvram.txt File Template

Testing and tuning a board with a schematic copied from a Cypress reference design can take place using the nvram.txt file template provided with the design package. To test and tune a board with a schematic that differs from a Cypress reference design, customize the nvram.txt file template per the instructions in this section.

Note: When a change is made in nvram.txt, the file must be saved and the wireless device must be disabled and re-enabled in the Device Manager for the change to take effect. Be sure to save the file as “nvram.txt” and place it in C:\Windows\system32\drivers\. Delete or replace any previous file with the same name in this directory.

During the board development phase, start with the default PA parameters provided in the nvram.txt template. The PA parameters will eventually be optimized using Cypress’s Transmit Signal Strength Indicator (TSSI) calibration tools.

A sample nvram.txt file, with parameters that are common to Cypress’s SDIO single-band, 2.4 GHz reference design boards, is shown in Table 2 and Table 3. Parameters listed in Table 2 are used and specified by Cypress only and not to be changed by customer. Parameters listed in Table 3 are design variables and are generally modified by customer to fit design specifications during development.

Note: For some board designs, if the OTP memory space is not large enough to store all of the parameters that designers deem necessary for nonvolatile storage, designs can maintain an nvram.txt file external to the chip. Upon chip initialization, a combination of parameters from OTP and the nvram.txt file will be downloaded to on-chip RAM.

Note: Customers should not modify the parameters in Table 2.

Table 2. Cypress-Specified Single-Band 2.4 GHz NVRAM Parameters

NVRAM Parameter	Example Data	Description
boardtype	0x0532	Board type. This is a critical parameter that should be copied from a similar reference board design.
boardflags	0x10080a00	Board configuration flags that define power topology, external components (ePA, eLNA), etc.
sromrev	3	SROM revision.
pa0itssit	0x20	Power detector dynamic range upper limit. Should be fixed at 0x20.
triso2g	0	Defines T/R switch isolation (0 means default).
rssismf2g	0xa	2.4 GHz RSSI midpoint select and board switch architecture (fixed values).
rssismc2g	0x3	
rssisav2g	0x7	
rssismf5g	0xa	5 GHz RSSI midpoint select and board switch architecture (fixed values).
rssismc5g	0x7	
rssisav5g	0x1	
swctrlmap_2g	0x0c05, 0x0a03, 0x0a03, 0x0, 0x0	Defines front-end RF switch or front-end module (FEM) control logic.
RAW1	4a 0b ff ff 20 04 d0 02 36 43	Defines SDIO hardware header per chipset.
otpimagesize	172	OTP memory size, in unit bytes.

Table 3. Customer-Modified Single-Band 2.4 GHz NVRAM Parameters

NVRAM Parameter	Example Data	Description
boardrev	0x1301	Board revision tracked by the internal test tool (optional). Examples: • 0x1301 converts to P301 • 0x1208 converts to P208
xtalfreq	37400	On board XTAL or oscillator frequency in kHz.
maxp2ga0	0x46	Maximum output power in hexadecimal format. Units of 0.25 dB. This applies to all CCK rates as measured at antenna port. The nominal target power in dBm for CCK packets is (0.25 × maxp2ga0 in decimal) – 1.5 dB (with a tolerance of ±1.5 dB).
ofdmgpo	0x66666666	The OFDM back-off from the maximum output power as defined by maxp2ga0. Resolution is 0.5 dB per step. Values are applied to the eight transmission rates: 54, 48, 36, 24, 18, 12, 9, and 6 Mbps. Rate 6 = LSB.
mcs2gpo0	0x2222	MCS0 to MCS3 per rate transmit power offset from maxp2ga0 for 2.4 GHz. One nibble per rate. Step size is 0.5 dB. MCS0 = LSB.
mcs2gpo1	0x2222	MCS4 to MCS7 per rate transmit power offset from maxp2ga0 for 2.4 GHz. One nibble per rate. Step size is 0.5 dB. MCS4 = LSB.
pa0b0	5836	PA parameters based on TSSI calibration.
pa0b1	–705	
pa0b2	–186	
aa2g	3	Number of antennas in bit-mapped binary format: 1 = 01b for one antenna 3 = 11b for two antennas
ag0	0x82	Antenna gain (in dBi) defined as: Converts hex to 8 bits binary: lower 0–5 bits = signed 2's compliment in units of dB; higher 6–7 bits unsigned number in units of quarter dB. Examples: • 0x82 = 2.5 dB (2 + 2 × 0.25) • 0x7f = –.75 dB (–1 + 1 × 0.25)
ccode	US	Country code.
il0macaddr[a]	00:90:4c:fe:${maclo}	The il0 MAC address format enables the firmware to use a default (dummy) MAC address if the OTP is blank (that is, if no valid MAC address has previously been programmed into the OTP).

a. When devices with blank OTPs are used, the firmware may fail to load unless a MAC address is provided. During board development, Cypress recommends using an il0macaddr value, which provides the driver with a “dummy” MAC address. With il0macaddr in nvram.txt, the driver loads even when the OTP is blank. In production, each board should have a valid, unique MAC address programmed into its OTP. il0macaddr is ignored by the driver when the MAC address is programmed in the OTP.

6.1 Editing the nvram.txt File

The nvram.txt file content should be edited in a properly-formatted text editor, such as Windows WordPad, so that the original format of the file is preserved. Using a non-formatted text editor (such as Notepad) destroys the format of the nvram map, causing the driver to fail to read the nvram.txt correctly.

7 Finalizing the nvram.txt File

After final PA parameters for the design have been generated, update `pa0b0`, `pa0b1`, and `pa0b2`, then adjust the Tx output power-related parameters in the nvram.txt file. Run output power tests (using the updated nvram.txt) to verify that these parameters are providing the correct output power. Verify that the RF performance (such as EVM, spectral mask, and rxper) meets design specifications.

Cypress recommends running a regulatory prescan to verify that the required output power can be delivered without violating the band-edge limits. If the band-edge limits cannot be met, it may be necessary to reduce the output power at the band-edge channels.

After all prototype tests have passed and all nvram.txt file parameters have been optimized and frozen, users can select the desired parameters to program to the OTP.

The CYW4330 has 172 bytes of space in the OTP memory. Given the limited space in the OTP, it is impossible to program the entire nvram.txt to the OTP. The programmer must be very careful to select only the necessary parameters to go into the OTP. Parameters that typically go into the OTP are those that are unique to the board (such as MAC address) and those that are required to satisfy local regulatory requirements, which are usually output power-related parameters (such as maximum output power, power offset per-rate, PA parameters, country code, etc).

8 OTP Programming Procedure

There are two ways to program OTP:

- Method One: Programming OTP Using the Driver WL Command. This method requires the user to manually create an OTP binary map file and use a driver wl command to program the OTP.
- Method Two: Programming OTP Using the Cypress Manufacturing Test Program[1]. This method only requires the NVRAM file, but the programming is done by the Cypress Manufacturing Test Program, so the installation and usage of the tool is required on the user's system.

8.1 Method One: Programming OTP Using the Driver WL Command

Prior to OTP programming, an OTP binary map file must be prepared and edited with correct values. An OTP binary map completely defines the parameters that are to be programmed to the OTP. The SDIO OTP data format is based on the Card Information Structure (CIS) as defined by the PCMCIA/SD Card Association. CIS data contains the hardware header followed by one or more data blocks, where each data block (or tuple) contains the type, length, and value of the tuple.

8.1.1 Minimum OTP Map

A set of parameters called the SDIO hardware header must be present in the OTP binary map as the header. The SDIO hardware header is required to boot up the CYW4330 via the SDIO interface when the driver detects there are contents in the OTP. Therefore, the SDIO hardware header is a minimal set of parameters when programming an OTP. Any other desired parameters to be programmed to the OTP are appended after the SDIO hardware header. When an OTP binary map contains only the SDIO hardware header, the binary map is called a minimal OTP map.

1. Confirm with Cypress engineering teams before using any Manufacturing Test Program to write OTP.

Table 4 shows the minimum OTP map for the CYW4330.

Table 4. CYW4330 Minimum OTP Map

172 Bytes OTP Map																
Offset	**0x0**	**0x1**	**0x2**	**0x3**	**0x4**	**0x5**	**0x6**	**0x7**	**0x8**	**0x9**	**0xa**	**0xb**	**0xc**	**0xd**	**0xe**	**0xf**
0x0000	4a	0b	ff	ff	20	04	d0	02	30	43	80	02	00	03	80	03
0x0010	02	40	00	80	03	1b	32	05	00	00	00	00	00	00	00	00
0x0020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x0030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x0040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x0050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x0060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x0070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x0080	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x0090	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x00A0	00	00	00	00	00	00	00	00	00	00	ff	ff				

For the CYW4330, the OTP map is terminated with 0xff 0xff.

8.1.2 Programming Basic NVRAM Parameters

Parameters in the nvram.txt file that are to be programmed to the OTP follow after the SDIO hardware header in the OTP binary map. These parameters form the CIS tuple in the CIS structure. Most parameters in the nvram.txt have a unique identifier called the CIS tuple tag. The driver recognizes and parses each CIS tuple by its tag number.

Table 5 lists the common basic nvram.txt parameters with their tag numbers and the byte size they occupy in the OTP memory space. Basic parameters are typically fixed values to a specific device or board, and tend to retain their values across the life of the device/board. For this reason, it is generally acceptable to program these basic parameters to the OTP early in the development, before the design is frozen.

Table 5. Basic NVRAM Parameters CIS Tuple Tags

NVRAM Parameter	CIS Tuple Tag	Length of Value (in bytes)
sromrev	0x00	1
boardrev	0x02	2
boardtype	0x1b	2
macaddr	0x19	6
ccode[a]	0x0a	2

a. The value for ccode in the nvram.txt is in ASCII format. It must be converted to hex format before entering it into the OTP map (for example, “US” = “0x55 0x53”).

In the OTP binary map, each tuple is formed by the four fragments described in Table 6.

Table 6. CIS Tuple Format

Fragment	Description
80	Indicates the beginning of a new tuple.

Table 6. CIS Tuple Format

Fragment	Description
Length	Defines the total size (in bytes) of the tag plus the value of the tuple that would occupy the OTP memory space.
Tag	The identifier of the parameter in nvram.txt. A tag usually takes one byte in memory.
Value	The value of the parameter. The format is in little endian (i.e., the first byte is the least significant byte).

For example, a tuple that looks like the following is defined by the fragments listed in Table 7:

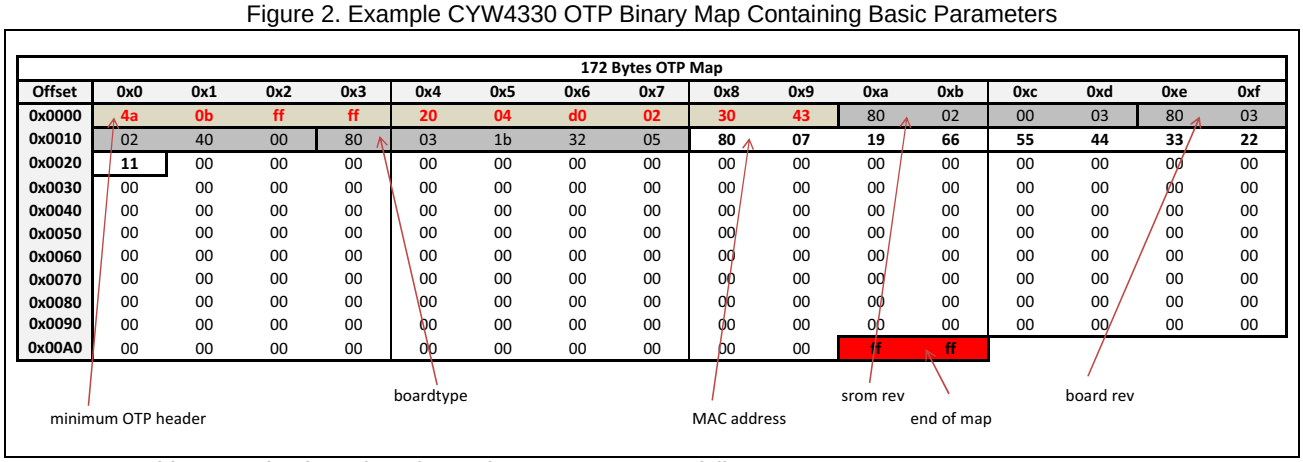
80	03	02	40	00

Table 7. Example Tuple Definition

Fragment	Description
80	Beginning of the new tuple.
03	The tag (1 byte) and the value (2 bytes) would occupy 3 bytes total in the OTP memory.
02	Tag of 0x02 is the identifier for `boardrev` in the nvram.txt.
00 40	The value of `boardrev` in reverse binary byte, or 0x4000.

Figure 2 shows an example of the OTP binary map for the CYW4330 that contains some of the nvram.txt parameters listed in Table 5.

Figure 2. Example CYW4330 OTP Binary Map Containing Basic Parameters



In this example, the values for each parameter are as follows:

- sromrev = 0x03
- boardrev = 0x40
- boardtype = 0x0532
- macaddr = 66:55:44:33:22:11

Note:

- CIS tuples do not have to be in any particular order, because each tuple begins with a unique identifier.
- OTP bytes can only be written to once, so only blank or zero-programmed bytes can be programmed on subsequent write cycles.

8.1.3 Creating and Editing the OTP Binary Map

To create and edit the OTP binary map, use a hexadecimal text editor of your choice. Cypress recommends a hexadecimal text editor called Notepad++, which is freeware and can be downloaded from the web at:

http://notepad-plus-plus.org/. Download the latest revision of the software, and follow the instructions to install the software.

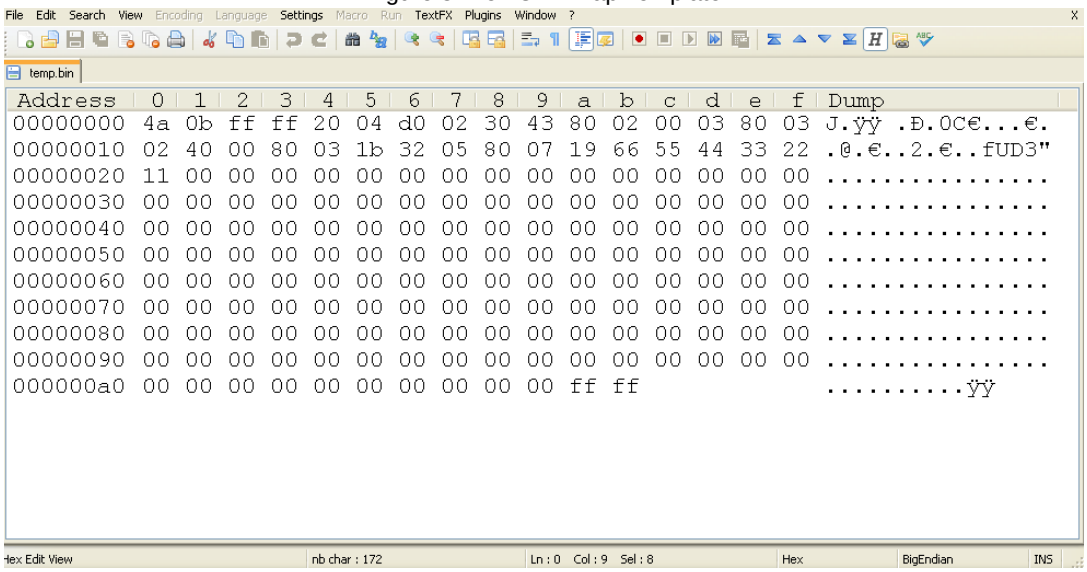
Additionally, a hex plugin is needed to edit text in hex format:

1. On the Notepad++ webpage, click the Resources tab.
2. In the Plugins section, click the link for “A list of plugins”.
3. On the new page, search for “Hex Editor”, then click on Hex Editor (Unicode) to download the plugin. (The link to download this plugin is: http://sourceforge.net/projects/npp-plugins/files/Hex%20Editor/Hex%20Editor%20Plugin%20v0.9.5/)
4. Save the plugin file at C:\Program Files\Notepad++\plugins.

After the installation is complete, open Notepad++.

1. To create a new OTP map, select **New** from the **File** menu; or, to open and edit an existing OTP map, select **Open** from the **File** menu.
2. After creating or opening a file, click the “H” button on the toolbar to show the hex template:

Figure 3. Hex OTP Map Template



3. Add or edit each byte in the map to fill in the SDIO hardware header and the CIS tuple according to the OTP binary map instructions described earlier in this section. The map shown in Figure 3 has been edited so that it matches the example CYW4330 OTP binary map in Figure 2.
4. When editing is completed, select **Save As** from the **File** menu to save the file. Manually type “.bin” at the end of the file name as the file extension. The file name must have .bin as the extension so it can be programmed to the OTP. Store this file in the working directory that contains wl.exe.

   For example purposes, this file is referred to as “4330_OTP.bin” in the following instructions.

8.1.4 Programming Procedure Using Driver WL Command

To program the OTP binary map to the OTP of the device:

1. Load the driver to the CYW4330 device with the desired nvram.txt file. Confirm the driver installation by giving a few wl commands (e.g., `wl ver`).
2. Type the following wl command to program 4330_OTP.bin to the OTP:

```
> wl ciswrite 4330_OTP.bin
```

3. In Windows Device Manager, disable the wireless device, and then enable it.
4. Confirm OTP is programmed successfully by running `> wl cisdump` (the following MAC address is used as an example: 12 34 56 78 90 11). The output should look like the following, matching exactly the OTP binary map created in Figure 3.

```
C:\WINDOWS\system32>wl ver
5.90 RC113.12
wl0: Mar  4 2011 20:59:28 version 5.90.113.12 (WLTEST)

C:\WINDOWS\system32>wl cisdump
Source: 2 (Internal OTP)
Maximum length: 172 bytes
Byte   0: 0x4a 0x0b 0xff 0xff 0x20 0x04 0xd0 0x02
Byte   8: 0x30 0x43 0x80 0x02 0x00 0x03 0x80 0x03
Byte  16: 0x02 0x40 0x00 0x80 0x03 0x1b 0x32 0x05
Byte  24: 0x80 0x07 0x19 0x12 0x34 0x56 0x78 0x90
Byte  32: 0x11 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte  40: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte  48: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte  56: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte  64: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte  72: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte  80: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte  88: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte  96: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte 104: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte 112: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte 120: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte 128: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte 136: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte 144: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte 152: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte 160: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Byte 168: 0x00 0x00 0xff 0xff

C:\WINDOWS\system32>_
```

If the cisdump is verified to match the OTP binary map, the OTP programming is complete. Once programmed, additional blank spaces (00) in the OTP can still be written by filling in those corresponding blank spaces in the OTP binary map. There is no restriction on how many times a device can be programmed, provided that each programming is writing to only blank, unwritten spaces. Follow the same procedure to program the additional blank spaces.

8.1.5 Programming Advanced NVRAM Parameters

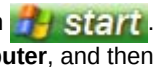
This information is under development.

8.2 Method Two: Programming OTP Using the Cypress Manufacturing Test Program

This information is under development.

Appendix A: Driver Removal

A.1. Uninstalling a Driver

1. Start the Windows Device Manager.
   a. Click the **Start** button  .
   b. Right-click **My Computer**, and then click **Manage**.
   c. In the left pane of **Computer Management**, double-click **Device Manager**.
2. In the right pane of **Computer Management** under **Network adapters**, right-click **Broadcom 802.11g Network Adapter #10**, and then click **Uninstall**.
3. Click **OK** in **Confirm Device Removal**.

4. Delete the following files, referring to Customizing the nvram.txt File Template , if necessary:

Filename	Location
bcmsddhd.sys	C:\Windows\system32\drivers\
nvram.txt	C:\Windows\system32\drivers\
oem#.inf	C:\Windows\inf\
oem#.pnf	C:\Windows\inf\
wl.exe	C:\Windows\system32\
brcm_wlu.dll	C:\Windows\system32\

The following files are typical SDIO driver files released with the Windows XP driver or design package:

bcm94329sdagb.txt
bcmsddhd.inf
bcmsddhd.sys
brcm_wlu.dll
wl.exe

A.2. Deleting the Correct oem#.inf and oem#.pnf Files

In some cases, multiple instances of Broadcom network adapters are installed. To locate the correct oem#.inf and oem#.pnf files for deletion, take the following steps with help from the Windows search facility:

1. Click the **Start** button  and then click **Search**  .
2. In the **Search Companion** pane of **Search Results**, type **C:\Windows\inf\oem*.inf** in the **All or part of the file name** box, type **Broadcom Corporation** in the **A word or phrase in the file** box, select **Local Hard Drives (C:)** from the **Look in** list, and then click **Search**.
3. If more than one INF file appears in the **Search Results**, open each in a text editor.

Delete the Broadcom oem#.inf file and the associated Broadcom oem#.pnf file for the oem#.inf file that looks similar to the file shown here.

```
;; bcmsddhd.inf
;;
;; Copyright 1998-2005, Broadcom Corporation.
;; All Rights Reserved.
;;
;; This is UNPUBLISHED PROPRIETARY SOURCE CODE of Broadcom Corporation;
;; the contents of this file may not be disclosed to third parties, copied or
;; duplicated in any form, in whole or in part, without the prior written
;; permission of Broadcom Corporation.
;;

[version]
    Signature   = "$Windows NT$"        ; Combined Win9x/Win2k inf
    Class=Net
    ClassGUID   = {4d36e972-e325-11ce-bfc1-08002be10318}
    Provider    = %V_BCM%
    Compatible  = 1
DriverVer=06/10/2009, 4.218.84.1
    CatalogFile=BCM43XX.CAT
    CatalogFile.NTamd64=BCM43XX64.CAT

[Manufacturer]
    %V_BCM% = BROADCOM, NTamd64
```

Document History Page

Document Title: AN214935 - OTP Programming and NVRAM Development Process for SDIO Embedded WLAN Devices Document Number: 002-14935				
Rev.	**ECN No.**	**Orig. of Change**	**Submission Date**	**Description of Change**
**	-	-	05/17/2011	4330-AN500-R Initial release
*A	5459425	UTSV	10/14/2016	Added Cypress Part numbering Scheme. Updated in Cypress template
*B	5834576	BENV	07/27/2017	Updated logo and copyright

Worldwide Sales and Design Support

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturers' representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC®Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Video | Blogs | Training | Components

Technical Support

cypress.com/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709