WICED® Studio

CYW20706, CYW20735, and CYW20719 Low-Power Modes

Associated Part Family: CYW20706, CYW20735, and CYW20719

Doc. No.: 002-22800 Rev. *B

Cypress Semiconductor

198 Champion Court

San Jose, CA 95134-1709

www.cypress.com

Contents

About This Document

This document explains Power Management Unit (PMU) features, Low Power Oscillators (LPOs), and power modes supported in WICED Bluetooth devices CYW20706, CYW20735, and CYW20719. This document will help you design low-power IoT applications in WICED Studio with Cypress' dual-mode Bluetooth devices.

Purpose and Audience

This document is intended for application developers designing and creating low power applications on CYW20706, CYW20735 and CYW20719 devices using Cypress WICED Studio.

Scope

This document explains the functionalities of the PMU and various sources of Low Power Oscillators (LPOs), and describes the low power modes supported in CYW20706, CYW20735, and CYW20719. This document does not include the detailed steps to create low-power applications on WICED Bluetooth devices. However, low-power examples are provided in the WICED Studio SDK. This document might refer to certain terminologies specific to the devices; see the device datasheets for more information.

Acronyms and Abbreviations

In most cases, acronyms and abbreviations are defined on first use.

For a comprehensive list of acronyms and other terms used in Cypress documents, go to www.cypress.com/glossary.

IoT Resources and Technical Support

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/).

1 Power Management Unit (PMU)

The PMU is responsible for lowering the average power consumption of the device. It is mainly a software implementation and has necessary supporting hardware. The PMU module is executed from the context of the lowest priority thread. Whenever the PMU gets the CPU, it tries to put the system into one of the supported low-power modes. The primary tasks of the PMU include:

- Implementing low-power modes, transitions, and timings
- Communicating with different blocks and modules to register callbacks for pre-sleep and post-sleep
- Implementing the wake ISR, which wakes up the device at the required time
- Detecting, switching, and calibrating of LPOs

1.1 Low-Power Oscillator (LPO)

LPO is a clock that runs in low-power modes to source some blocks. It is responsible for the timing for wakeup. This clock has low power consumption, which makes it ideal for operation in low-power modes. The Bluetooth Specification specifies an accuracy of ±250 ppm for Basic Rate/Enhanced Data Rate (BR/EDR) and ±500 ppm for Bluetooth Low Energy (BLE). Therefore, the LPO accuracy must lie within this range if there is an active Bluetooth operation.

There are multiple possible sources for the LPO in all three devices. They are:

- **Low Power Xtal (LPX):** This clock is derived from the external main crystal oscillator, but runs in low-power mode during sleep. LPX typically has a frequency of 1 MHz and provides a greater clock accuracy of 20 ppm or better, but consumes relatively more current compared to other LPO sources. LPX is useful when high accuracy is required without increasing BOM because it can serve as a substitute for the external 32-kHz crystal.
- **External LPO clock:** All three devices have pins to which an external digital clock or a 32.768-kHz crystal can be connected. The accuracy should be within ±250 ppm as specified by the Bluetooth specification. If an external LPO is used, then the PMU will most likely use this under all conditions when a 20-ppm clock is not required. If a 20-ppm clock is required at any time, then the LPX will be used instead of the external LPO clock.
- **Internal LPOs:** The internal LPOs belong to the Lean High Land (LHL) domain; therefore, they are always powered ON. There are two internal LPOs: one which has lower accuracy (~500 ppm) and consumes less current (~2-3 µA), and another with higher accuracy (~50 ppm) but consumes more current (~10 µA). The PMU calibrates these clocks at regular intervals to improve their accuracy.

1.2 PMU Tasks with Respect to LPOs

Detection: For CYW20706, the PMU detects the presence or absence of an external crystal or external digital clock. On the other hand, CYW20719 does not support auto-detection. An entry in the *patch.cgs* file[1] determines whether an external LPO source is present. In WICED Studio, the entry is present by default, so an external LPO source is expected. If an external LPO source is absent, remove the following code entry from the *patch.cgs* file.

```
ENTRY "Data" = "g_aon_flags"
{
  "Address" = 0x0028067c
  "Data" =
   COMMENTED_BYTES
     {
        <hex>
        81
     } END_COMMENTED_BYTES
}
```

[1] File path: *\20719-B1_Bluetooth\WICED\internal\20719B1\patches*

With CYW20735, the only option is to use an external digital clock or a 32.768-kHz crystal.

Switching: The PMU switches between different LPOs. Active peripherals can request a clock with a particular ppm value so that the PMU can select an appropriate LPO source based on the required ppm. The PMU might also turn OFF unused LPO sources to save power. The usual preference for the LPO source is in the following order: External LPO clock > Internal LPO > LPX. Although it is possible for WICED Bluetooth devices to enter low-power modes using internal LPOs, it is recommended to use an external crystal.

Calibration: The PMU calibrates internal LPOs at regular intervals to improve accuracy. The calibration is performed against the external high-accuracy (20 ppm) clock. The PMU calibrates internal LPOs every 500 ms when active. If in low-power mode, the calibration is done when the chip wakes up. The calibration takes place only if the internal LPOs are selected as the source.

1.3 PMU Architecture

The PMU module executes from the context of the lowest-priority thread. It interacts with other modules to determine whether it can put the system to sleep. The PMU collects necessary information to decide whether to go to sleep and the duration of sleep. The PMU manages the states of the clocks and the LDO voltages in low-power modes.

The differences between the different power modes lies in the following aspects:

- Blocks that are turned OFF
- Duration for which the system goes to sleep
- Accuracy of the internal clock while sleeping
- Wakeup sequences

2 Power Modes

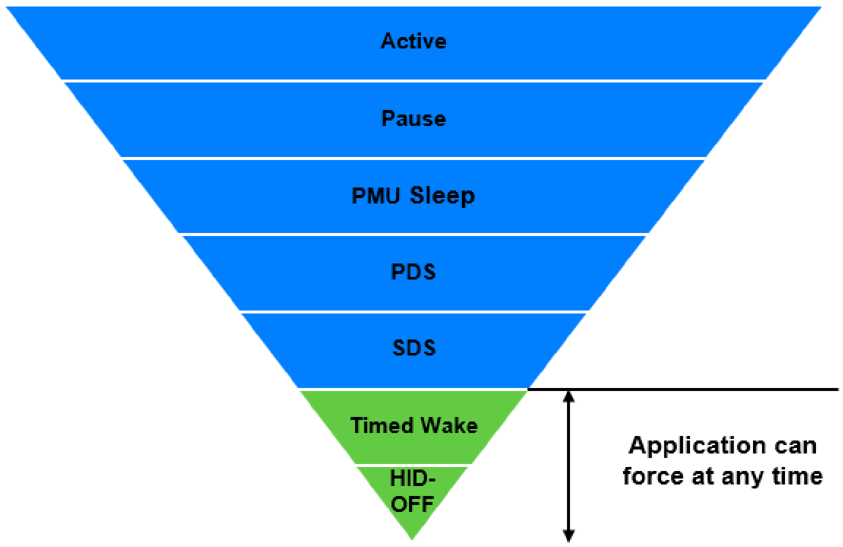
WICED Bluetooth devices support different power modes. However, it is important to note that all devices do not support all modes.

- **Active**: Active mode is the normal operating mode in which all peripherals are available and the CPU is active.
- **Pause**: In this mode, the CPU is in Wait for Interrupt (WFI) and the HCLK, which is the high-frequency clock derived from the main crystal oscillator, is running at a lower clock speed. Other clocks are active and the state of the entire chip is retained. Pause mode is chosen when other lower-power modes are not possible.
- **PMU Sleep**: In this mode, the CPU is in WFI and the HCLK is not running. The PMU determines whether other clocks can be turned OFF, and does so accordingly. The state of the entire chip is retained including SRAM; internal LDO regulators run at a lower voltage (voltage is managed by the PMU).
- **Power Down Sleep (PDS)**: This mode is an extension of PMU Sleep wherein most of the peripherals such as UART, and SPI are turned OFF. The entire memory is retained; on wakeup, execution resumes from where it was stopped. This mode is available in CYW20706 and CYW20719 but not in CYW20735.
- **Shut Down Sleep (SDS)**: Everything is turned OFF except LHL, RTC, and LPO. The device can come out of this mode due to Bluetooth activity, timer expiration, or an LHL interrupt. This mode makes use of the micro Bluetooth Core Scheduler (µBCS) which is a compressed scheduler different from the regular BCS. Before going into this mode, the application can store up to 256 bytes of data into the Always On (AON) RAM. When the device comes out of this mode, the data from the AON RAM is retained. After waking from SDS, the application will start from the beginning (using a fast boot which is described later) and will restore its state based on information stored in AON RAM. In the SDS mode, a single Bluetooth task with no data activity, such as an ACL connection, BLE connection, or BLE advertisement can be maintained. If there is data activity during these tasks, the system will undergo a fast boot and the normal BCS will be called. This mode is available in CYW20719 and CYW20735, but not in CYW20706.

  There are some guidelines and restrictions on how the SDS mode should be used:

  - The device address type should only be public.
  - The connection interval should be greater than 100 ms.
  - If the device is doing connectable advertisement in SDS mode, two connection requests need to be sent to the device to connect to it: the first one to wake the device and the second one to actually connect to it. If the device is already connected, there is no need to send any command twice.
  - The pairing process cannot be done in SDS mode.
  - High-duty-cycle advertisement with a low interval will not allow the device to enter SDS. If the device is advertising in low duty cycle and the advertisement time is set for infinity (i.e., no timeout), the device will enter SDS immediately. If the low-duty-cycle advertisement is configured for a finite time, the device will enter SDS only after this time expires (i.e., when the advertisement stops).
  - SDS should be allowed only if the next activity is more than 100 ms away; otherwise use PDS because frequent wakeup from SDS will consume more power than using PDS.
  - To get a GPIO interrupt callback upon wake up, the interrupt callback needs to be registered in the application.
- **Timed-Wake**: The device can enter this mode asynchronously. That is, the application can force the device into this mode at any time without asking for permission from other blocks. LHL, RTC, and LPO are the only active blocks in Timed-Wake. A timer that runs off the LPO is used to wake the device up after a predetermined time.

- **HID-OFF**: This mode is similar to Timed-Wake, but in HID-OFF mode, even the LPO and RTC are turned OFF. Therefore, the only wakeup source is a LHL interrupt.



2.1 Power Modes Supported by CYW20706, CYW20735, and CYW20719

Table 2-1 lists the power modes supported by CYW20706, CYW20735, and CYW20719.

	Active	Pause	PMU Sleep	PDS	SDS	Timed Wake and HID-OFF
CYW20706	✓		✓	✓		✓
CYW20719	✓	✓	✓	✓	✓	
CYW20735	✓	✓	✓		✓	

Table 2-1. Supported Power Modes

3 WICED APIs

The PMU functionality and low-power modes might seem complex, but WICED Studio offers drivers that take care of low-power operation. You will have to call a few functions to be able to observe the low power consumption.

3.1 CYW20706

The *wiced_power_save.h* header file contains the API functions related to low-power operation for CYW20706.

`wiced_sleep_config()`: Use this function to enable or disable low-power operation of the device. The PMU will determine when the system can go to low-power operation and will put the device in the appropriate low-power state. The application will resume on waking from sleep.

`wiced_power_save_start()`: Use this function to put the device into HID-OFF or Timed-Wake. The wakeup source should be passed as an argument. The wakeup source can be an LHL GPIO interrupt (`WICED_WAKE_SOURCE_GPIO`), timer interrupt (`WICED_WAKE_SOURCE_TIMEOUT`), or both. The application will be restarted on exiting the power save mode. If the application wants to save any state before going into power save and then restore that state when coming out of power save, it can use `wiced_power_save_store_state()` and `wiced_power_save_retrieve_state()` respectively.

See the CYW20706 - BLE Low Power Beacon example which demonstrates the low-power operation in CYW20706.

3.2 CYW20719 and CYW20735

The *wiced_sleep.h* header file contains the API functions related to the low-power operation of CYW20719 and CYW20735.

`wiced_sleep_configure()`: Use this function to enable the low-power operation of the device. The parameter passed to this function is a structure of type wiced_sleep_config_t. In addition to wake sources, it contains a callback that will be called by the PMU to poll for sleep permission and sleep time. In the callback, if the parameter passed is WICED_SLEEP_POLL_SLEEP_PERMISSION, the application should return one of the following values based on the requirements:

- `WICED_SLEEP_NOT_ALLOWED` – The application can return this value if it does not want the device to go to sleep mode.
- `WICED_SLEEP_ALLOWED_WITHOUT_SHUTDOWN` – The application can return this value if low-power is allowed, but if the device should not enter SDS. This means that the lowest power mode that the device can enter is PDS. This value should be passed if data exchange over Bluetooth is expected and entering SDS will be irrelevant.
- `WICED_SLEEP_ALLOWED_WITH_SHUTDOWN` – When this value is returned, the device can enter any of the low-power modes including SDS. Note that this value should not be passed for the CYW20735 device because it does not support PDS mode.

If the parameter passed to the callback is WICED_SLEEP_POLL_TIME_TO_SLEEP, the application can return one of the following values:

- WICED_SLEEP_MAX_TIME_TO_SLEEP – The application can return this value if it wants the device to sleep for the maximum possible time.
- 0 – The application can pass this value if you don't want the device to sleep.

If you want the device to wake up from SDS after a particular time interval, it is best to use an application timer. The device will retain only the shortest timer while going in to SDS; therefore, it is advisable to keep only one timer running while going in to SDS. See the low_power_sensor example included in WICED Studio,which demonstrates how low-power operation can be configured in CYW20719 and CYW20735.

4 Conclusion

WICED Bluetooth devices support ultra-low-power operation, which is essential for many IoT applications. Although architecturally complex, WICED Studio provides an easy to use API to configure these devices for low-power operation. Examples of low-power applications are included in the SDK; you can easily adopt them in your projects.

Document Revision History

Document Title: CYW20706, CYW20735, and CYW20719 Low-Power Modes

Document Number: 002-22800

Revision	ECN	Issue Date	Description of Change
**	6062087	01/18/2018	Initial release
*A	6314400	09/19/2018	Updated document to include CYW20735 device. Added guidelines for SDS mode
*B	6341837	10/11/2018	Removed PDS mode for CYW20735

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's 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 MCU

Cypress Developer Community

Community | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

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