wakeup_pin current consumption

Answered

p b
asked this on March 10, 2014, 18:46

Hi

I'd like to ask about the current consumption increase once a wakeup_pin is added to a project's hardware.xml.

With the wakeup_pin is defined current consumption after pressing the button goes up to around 9mA for the whole time the button is being pressed and not only at module startup. Is there any way to wake the BLE112 and reduce the current draw? I tried disabling the IRQ after waking up but this didn't change anyhing.

What is actually needed is to wake up the BLE112 module and execute normally (which takes ~1mA in my case). Just holding the button increases the energy usage nine times.

Comments

Jeff Rowberg
Bluegiga Technologies

Answer

Hi PB,

The wake-up pin is designed to do precisely this; it prevents the CPU from being in any other mode except **active**, and this is what results in the higher consumption. The low-power PM1/2/3 modes become unavailable. That's the purpose of the wake-up pin functionality as opposed to a normal interrupt. A regular interrupt will wake the module only to handle the interrupt signal. But the wake-up pin disables sleep mode as long as it is asserted. This is mainly (and only) useful if you need to do something that cannot be done in sleep mode, such as PWM output or UART input.

March 10, 2014, 19:00

p b

Is there any way to reduce the current consumption? I need to use the button connected to wakeup_pin for a push&hold functionality but 9mA is a lot and i'm looking for ways to reduce it.

March 11, 2014, 00:25

Chung Mui
Bluegiga Technologies

Hi PB

Perhaps, the push button circuity causes the extra unnecessary current consumption.

According to CC254x specification, min. logic-1 input voltage for PIO is 2.4v. You can increase the resistance of the serial resistor for the push-button circuity to reduce the current consumption. The resistance can be as high as possible as long as the voltage at the wakeup_pin is higher than 2.4V when you press and hold the button. You may need some margin for safety, it is better for you to set the voltage to 2.7V. You can try 10kohm or 100kohm first and then fine tune its value.

I hope my suggestion work.

March 11, 2014, 05:31

p b

Hi Chung,

I tested the circuitry and it takes about 0.1mA by itself. I'm measuring in both power supply input to the circuit and the BLE112 and it's clear that BLE112 is using almost all of the energy. It's understandable that there are short peaks of increased current consumption but i don't see why the wakeup pin has to take so much energy. It seems reasonable to wake up the device and let it resume the code execution. Is there any config option to allow the wakeup_pin to wake up the device and let it do it's thing instead of forcing it to stay active for the time that the button is pressed?

Or maybe i should approach the problem in a different way? What I'm trying to achieve is to keep the device in PM3 until an external interrupt is triggered (eg. button is pressed). The same input source is then used to turn the BLE off (after holding the button for ~2sec)

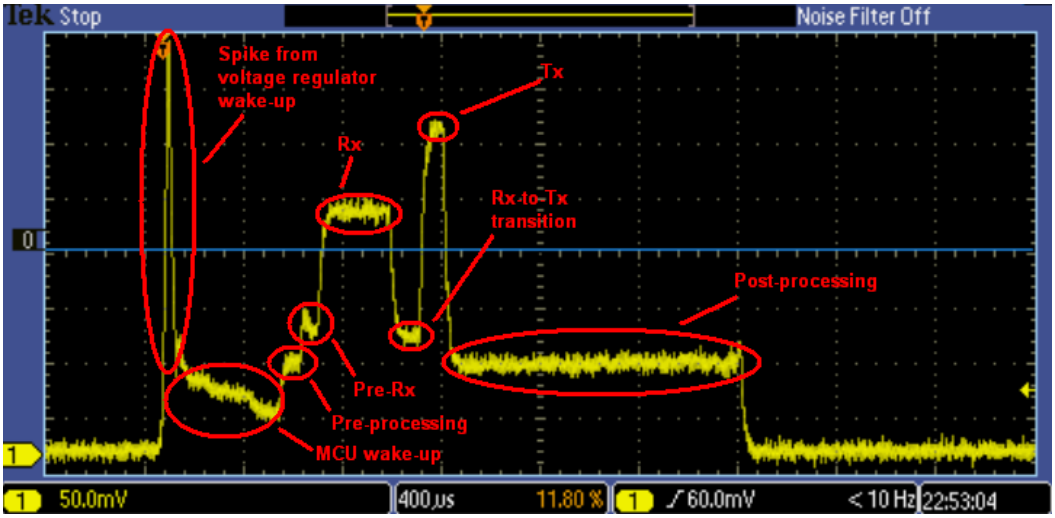
March 11, 2014, 09:52

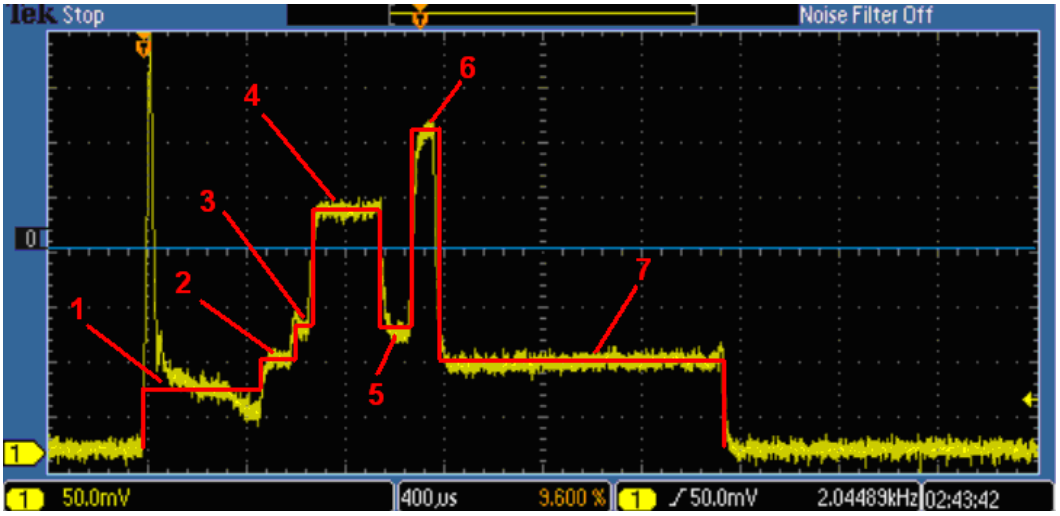
Hi PB

Support

Chung Mui
Bluegiga
Technologies

Below is the current consumption profiles for CC2540:





	Time (µs)	Current (mA)
State 1 (wake-up)	496	6.1
State 2 (pre-processing)	80	8.1
State 3 (pre-Rx)	80	12.3
State 4 (Rx)	288	22.3
State 5 (Rx-to-Tx)	120	11.1
State 6 (Tx)	104	29.3
State 7 (post-processing)	1180	8.1

Table 1- Measurements from Capture in Figure 14

You may notice when looking at the capture is a large spike in current at the moment when the MCU wakes up from sleep. This spike is caused by the digital voltage regulator inside the CC2540 re-powering up. The regulator contains capacitors that must be re-charged, and thus quickly draw current when the device wakes up. This spike normally would not appear with 47uF capacitor at VDD populated on the board; and therefore while testing power consumption this spike can be ignored.

In addition to the voltage spike, you will notice that the current draw changes as the CC2540 goes through several different states as a part of the connection event:

1. MCU wake-up – upon waking up, the current level drops slightly
2. Pre-processing – the BLE protocol stack prepares the radio for sending and receiving data

-- It's understandable that there are short peaks of increased current consumption but i don't see why the wakeup pin has to take so much energy.

Chung>> Actually, the current you saw was "MCU wake-up process" and/or "pre-process" but not the pin. The wakeup pin just causes a spike for a very short duration and it can be managed by 47uF capacitor at VDD.

-- Is there any config option to allow the wakeup_pin to wake up the device and let it do it's thing instead of forcing it to stay active for the time that the button is pressed?

Chung>> I am afraid not. You have to keep wake-up stay high to prevents the CPU from being in **sleep mode.** If you have tested that your push button circuity just draws 0.1mA, then problem is not caused by **"holding the wake-up pin to high"**

Back to your previous questions:

-- With the wakeup_pin is defined current consumption after pressing the button goes up to around 9mA for the whole time the button is being pressed.

Chung>> It sounds normal because the MCU and RF of the module is working at that moment and the peak current could be 36mA.

-- Is there any way to wake the BLE112 and reduce the current draw?

Chung>>

1. You can reduce the Tx power if you do not need long communication (but this will reduce the Tx peak current only)

2. You can use TI's TPS62730 (The internal LDOs in the CC254x regulate the supply voltage to 1.8V. At high battery voltage the efficiency takes a hit as a large amount of the battery is wasted on the LDOs. The TPS62730 can convert from 1.9V to 3.9V to 2.1V to avoid wasting power)

3. <slow_clock> and <throughput> may also help but there are downsides. For the details, please refer to the Configuration Guide

March 11, 2014, 11:30

p b

Chung

Thank you for all of the reference. This is a great deal of information.

I managed to overcome the problem with a BC847 transistor which triggers the wakeup_pin. Immediately after waking up the transistors emitter voltage is cut off (i've connected it to P2_0) and the code continues to run at around 1mA no matter how long the button is pressed. There's a bit more involved but the following describes the general approach to the problem.

March 11, 2014, 17:00