LPC2387 Timer/Counter Explained

This article is an attempt to summarize, explain, and show examples of the LPC2387FBD100 Timer/Counter functionality. For detailed information on the LCP23xx timers, please refer to chapter 23 of the LPC23XX User manual (Rev. 02 — 11 February 2009), which can be found at

<http://www.keil.com/dd/docs/datashts/philips/lpc23xx_um.pdf>

According to the LPC23xx User’s Manual, timer applications include:

* Interval Timer for counting internal events.
* Pulse Width Demodulator via Capture inputs.
* Free running timer.

In addition to these applications, I’m sure we’ll be able to add to this list after we have a better understanding of the functionality they include.

So, let’s start from the top. Are we talking about a timer or a counter, both, or neither? This is a bit of loaded question, considering that both timers and counters are functions of the topic we’re discussing. Instead of debating on the appropriate labeling of our topic, I thinks it’s best to simply describe the artifact in terms of what it really is. That is, the Timer Counter (TC) is simply a 32bit register that increments by one from x0000 0000 (0) to xFFFF FFFF (4,294,967,295). Well, that’s pretty simple but there’s a bit more to it than just being a register, so I would urge you to read on!

Before reading on, I did use terms such as: register, bit, and even threw in some Hex notation, so you may want to familiarize yourself with these terms before venturing much farther. Also, for demonstration in this article, I’ll be discussing timer3 on the LPC2387FBD100 assembled on a FEZ Mini. This chip is also used on the Panda and Panda II, which works exactly the same (aside from the pins we’ll discuss later). The Domino and Rhino use the 144 pin model, which is very similar to the 100 pin we’ll be discussing.

The LPC23xx chips have 4 timers [0/1/2/3] on them, and we’ll be using timer3 (T3) since it is generally unused and the CAP and MAT pins for it are mapped on the Mini. We’ll discuss CAP and MAT pins a bit later. For our discussion of T3, we’ll break it into three segments: incrementing the timer, monitoring the timer, and output/interrupts from it.

So, let’s begin by describing how to increment TC3 through either the Peripheral Clock (PCLK – see chapter 4 of the User’s Manual) or a CAP pin. For the first portion of the incrementing discussion, there are four registers we’ll want to talk about:

* TCR – Timer Control Register (T3TCR – 0XE007 4004)
* TC – Timer Counter (T3TC – 0xE007 4008)
* PR – Prescale Register (T3PR – 0xE007 400C)
* PC – Prescale Counter (T3PC – 0xE007 4010)

The Timer Control Register (TCR) is used to enable/disable, or reset the both the Timer Counter (TC) and the Prescale Counter (PC). The Timer Counter (TC) is as discussed earlier just a 32 bit register that holds the value of T3. Now the more interesting bit, let’s talk about the Prescale Register (PR) and Prescale Counter (PC). The Prescale Register (PR) is 32 bit value that we select, let’s say we set it to the value of 10 (0xA). Then the Prescale Counter (PC) is another 32bit register that increments with every incoming interval (PCLK or CAP). When the Prescale Counter equals the Prescale Register (remember that we set this to the value of 10), the Timer Counter (TC) is incremented by one. So, in our case, the Timer Counter will increment by one on every 10 input intervals. Consequently, by setting the Prescale Register to a value of 0, the Timer Counter will increment on every input interval. So, as the name suggest, the Prescale Register and Counter can be used to scale increments to the Timer Counter via the input pulses. So, keep in mind that the input intervals are actually affecting the Prescale Counter, and not the Timer Counter. The Timer Counter is only incremented when the Prescale Counter and the Prescale Register match.

The figure below might help better understand.



Before we move on, let’s talk a little more about the Timer Control Register. Unlike the Timer Counter and Prescale registers, that just hold a numeric values, the Timer Control Register has specific bits that are used to control specific items.



|  |  |  |
| --- | --- | --- |
| **Bit** | **Purpose** | **Value** |
| 0 | Enable the timer | 0 - Timer is disabled 1 - Timer is enabled |
| 1 | Reset | 0 - Work normally 1 - Reset the TC and PC on the next input |
| 7:2 | Reserved | Should not be used |

Note that the reset will continue to reset as long as the value is set to 1. So, bit 1 should be set back to 0 after the reset has been performed.

Now that we understand how the Timer Counter is incremented, let’s discuss the use of Capture Controls used to receive the input intervals. Remember that there are two different input intervals we can toggle our Timer Counter with: PCLK, and CAP. In this discussion, we will focus on CAP as our input interval. CAP is our Capture Input, which provides us with access to a pin to tie to our external input interval generator (Could be a PWM, or an external device such as an encoder).

There are two CAP – Capture Inputs – for each timer: CAP3.0 and CAP3.1 for timer3 (T3).



