arm Education

Embedded Systems Essentials with Arm: Getting Started

Module 6

SV1 (6): Module 6 Lab: Before you begin

Hello and welcome to this lab, which will cover timers and pulse-width modulation.

In this lab, we will continue exploring the capabilities of embedded microcontrollers by learning how to use two common modules that can be found in many embedded microcontrollers: Timers and Pulse-width Modulation. We will apply this practically by designing an audio player that will play a pre-saved melody. The volume of the music, as well as its speed, can be controlled by two potentiometers.

By the end of this lab you will gain some insight into the Mbed API and how to utilize Timers and Tickers, as well as Pulse-width Modulation.

Before starting the exercise, let us have a short review of these topics.

Timers are obviously very useful when we need to measure a period of time with precision, however this is not the only way to utilize this module.

Timers use an input clock source with a known frequency as a reference. This frequency can be modified using dividers to make it more suitable for a particular task. The timer register is incremented or decremented with each “tick” of the clock. Then the value of the timer register can be read to know how much time has passed since the timer was last reset or can even generate an interrupt when it reaches a certain value.

Often timers have comparators. An interrupt is called when the timer value and the comparator values are equal.

With this, timers usually have three modes of operation: Compare, capture and Pulse-Width Modulation.

The compare mode is the one that compares the timer value with another preset value. The value is loaded into the compare register and the timer is configured to increment or decrement at a specific speed. Then the process keeps running automatically until the two values are equal. At this point an interrupt is called.

This mode is especially useful if you’re wanting some action to fire periodically.

The capture mode is used to measure how long it takes to perform a specific task. On it the timer is configured to run at a specific frequency, and will capture the value of the counter when an event occurs, for example when a task starts. By capturing multiple values and calculating the difference, we can know how much time has passed between events.

The third one is pulse-width modulation mode or PWM mode. This mode is often classified separately as its purpose is very different.

In this mode the timer is used to regulate the duty cycle of the signal, in other words, the percentage of time that the signal is logic level high, with the intention of controlling the average power of the output. This is represented in the following graphic. The blue line is the signal generated by the PWM module. See that the frequency doesn’t change. Only the time the signal is logic level high does. The red line represents the average power of the signal.

The PWM mode is similar to the compare mode except that, in this instance, the timer is not automatically reset after reaching the compare value, but instead counts to the end or maximum value of the timer before it resets.

This means that two interrupts are generated: one compare interrupt and one timer-overflow-interrupt. These two interrupts can be used to switch the PWM and its duty cycle on and off.

We can use the Mbed Timer API to create, start, stop, and read a timer for measuring precise times, better than millisecond precision, illustrated in this example.

You can independently create, start, and stop any number of Timer objects simultaneously using the timer functions provided by the API which can be found in the table displayed.

We can use the Mbed Ticker API to set up a recurring interrupt; it calls a function repeatedly and at a specified rate, for example:

You can create any number of Ticker objects, allowing multiple outstanding interrupts at the same time. The function can be a static function, a member function of a particular object, or a Callback object. This can be done using the ticker functions provided by the API which can be found in the table displayed.

Additionally, the table here displays some member functions of the PwmOut API which will be helpful for this lab.