PWM Hub

Federico Balaguer

June 1, 2016

Abstract

Pulse-Width Modulation is a technique used to encode a message into a pulsing signal. It is a technique commonly used on microcontroller programming. A PWM solution usually is evaluated given two aspects: number of pulse outputs and precision of the duty-cycle.

Microcontrollers usually include a module that implements PWM. A PWM module sometimes comes short as a solution because of two reasons. First, it implements only a pair of pulse outputs. Second, the precision of the duty-cycle are bound to the internal oscillator settings.

PWM Hub is a library that implements PWM module by software that makes it possible to have a number of pulse outputs and it decouples the precision of the duty-cycle from the frequency of the internal oscillator.

1 Pulse-Width Modulation Paradox

Many times when developing an embedded system on a microcontroller, developers want to use the highest oscillator frequency which in turn affects the kind of PWM signal that the system can generate. For example, RC Analog servos expects a PWM signal that has a small high state.

Pulse-Width Modulation encodes a message into a pulsing signal that has the following parameters:

Period It acts as a reference to the output signal, the output signal will be on high state at most the period time.

Output Signal It has two possible states: High and Low. It represents a given datum as a high state time.

Duty-cycle It is the relationship betwen Output Signal high state time and the Period.

Dead bandwidth It is the minimum time that represents different data.

2 PWM Hub

PWM Hub is a ANSI C library that makes it possible to generate multiple PWM signals that have the same Period.

PWM Hub defines three structures and functions for defining PWM signals and for handling tick counts.

```
struct pwm_counter
{
    PWMCOUNTER_TYPE reset_value;
    PWMCOUNTER_TYPE tick_count;
    unsigned char* port;
    unsigned char pin;
};

struct pwm_period
{
    PWMPERIOD_TYPE reset_value;
    PWMPERIOD_TYPE tick_count;
};

struct pwm_array
{
    struct pwm_counter pulse_array[PWMARRAY_SIZE];
    struct pwm_period period; //period has to be equal or greater than pulse
};
```

The structure *pwm_array* is the highest level entity of the library. It defines: a period and an array of pulses (or pwm signals). The period is common to all pulses. It means that all signals of one *pwm_array* will be reset at the same time.

The structure *pwm_period* has two fields: tick_count and reset_value. The field tick_count holds -at any time- the current tick count. This field is used to know whether the period needs to be reset. The field reset_value holds the value that needs to be set in the field tick_count when under reset condition.

The structure