Pulse-Width Modulator

Features

Four-channel 10 bit pulse-width modulator

Spreading function for easier filtering

Servo mode via pulse swallowing

DMA request and dedicated DMA bus interface

Standard byte-wide interface

Description

The Pulse Width Modulator consists of a ten-bit free running counter, and four width registers. Each PWM output is High for "n+1" counts out of the 1024-clock count cycle, where "n" is the value held in the width register. The PWM output High time can optionally be spread throughout the cycle to reduce ripple on the externally filtered PWM output.

The spreading function is implemented by dividing each 1024-clock cycle into four quadrants of 256 clocks each. Within each quadrant, the Pulse Width Modulator uses the eight MSBs of each pulse-width register to select the base width in each of the quadrants. This is the equivalent to dividing the contents of the pulse-width register by four and using this value in each quadrant.

Options are available to suppress the PWM output for seven-of-eight, three-of-four and one-of-two iterations of the PWM counter The one-of-eight option works nicely with R/C servos, which require a 1mS to 2mS pulse width and a 20mS period. This option gives the full resolution for the pulse width while still meeting the period requirements.

DMA channels are usually designed to work with fixed I/O addresses. To allow DMA control of the PWM, a separate PWM Block Access Register (PWBAR) and PWM Block Pointer Register (PWBPR) are available. The pointer register contains the address of the PWM register to be accessed via the access register. Each read or write of the access register automatically increments the pointer register. This allows the DMA to write to a fixed internal I/O location but still program all of the PWM registers.

Interface

```
module pwm_top (dreq_pwm, dreq2_pwm, pwblk_rbus, pwm_int, pwm_out, pwm_rbus, clkp, peri_addr,
                pwblk_rd, pwblk_wr, pwm_inc, pwm_wr, pwrite_bus, resetb);

  input         clkp;          /* main peripheral clock                                */
  input         pwblk_rd;      /* pwm block access peripheral read strobe              */
  input         pwblk_wr;      /* pwm block access peripheral write strobe             */
  input         pwm_inc;       /* pwm counter increment                                */
  input         pwm_wr;        /* pwm peripheral write strobe                          */
  input         resetb;        /* internal reset                                       */
  input   [3:0] peri_addr;     /* internal peripheral address bus                      */
  input   [7:0] pwrite_bus;    /* internal peripheral write bus                        */
  output        dreq_pwm;      /* pwm dma request                                      */
  output        dreq2_pwm;     /* pwm dma request 2                                    */
  output  [3:1] pwm_int;       /* pwm interrupt request bus                            */
  output  [3:0] pwm_out;       /* pwm output bus                                       */
  output  [7:0] pwblk_rbus;    /* pwm block read data bus                              */
  output  [7:0] pwm_rbus;      /* pwm read data bus                                    */
```