Using a fido Timer Counter Unit as a 4-Channel Pulse Width Modulator

An Innovasic Semiconductor Application Note
fido1100 Application Note 130

Version 1.1

January 2007

Innovasic Semiconductor

Table of Contents

Abstract

This note describes how to set up a fido TCU to provide 4 channels of pulse-width modulation.

Approach

TCU Description

The **fido 1100** has two identical Timer Counter Units. The following describes the registers associated with each TCU.

Register Name	*Description*
Preload Value Register	Contains a 16-bit preload value – used to control timer frequency.
Status Register	Contains flags for various events – not used in this example
TCU Mode Register	Used to set up basic modes/operation (auto-reload, prescale, etc.).
Channel X Mode Register	Defines the behavior of a specific channel (0-3)
Channel X Input Capture Register	Value captured on a given channel – not used in this example.
Channel X Output Compare Register	Value at which to generate an output compare event – used in this example to set pulse width

Setting the Frequency

The following elements determine timer frequency:

- Clock Source
- Prescale Setting
- Preload value
- Count Direction

The first two are set in the TCU Mode Register, the third is set in the Preload Value Register. The formula for calculating frequency for a down-counter is as follows:

PWM_Frequency = Input_Clock_Frequency / Prescale / Preload_Value

Input_Clock_Frequency = 66 Mhz

desired PWM_Frequency = 500 Hz

Input_Clock_Frequency/PWM_Frequency = 66 Mhz/500 Hz = Prescale * Preload_Value

Clock_Divider * Preload_Value = 132,000 => Prescale > 2

Preload_Value = 132,000/4 = 33,000 **= 0x80E8**

We'll use a prescale of 4 and preload of 33,000.

Now we can define the TCU Mode register:

Field Name	***Value***	***Comments***
TC_IRQEN	0	Interrupt enable – disable all interrupts
Context	N/A	Without interrupts enabled, this field doesn't matter (should generally be set to the same context that is handling the timer).
Priority	N/A	Interrupts not used
TE_IRQEN	N/A	Interrupts not used
UP/DN	0	Use as down counter
EN	1	Enable the timer
Autoreload	10	Enable auto-reload
Reload	0	A 1 causes a single reload of the timer, this register will only be written once, to start the PWM, this bit is irrelevant.
Prescaler	0010	Set prescale to 4
Clk_mode	00	Use System Clock
Overall Register	**0x00006100**	

Now we have a timer that will load the preload value into the timer, count down once every 4 system clocks until it reaches zero, then repeat the process. The following table shows some examples of frequencies that can be readilly generated with the timer.

Selected Frequencies on TCU with 66 Mhz System Clock		
Desired Frequency	***Prescale***	***Preload***
.05 Hz	32768	40,283 = 0x9D5B
1 Hz	1024	64,453 = 0xFBC5
60 Hz	32	34,375 = 0x8647
100 Hz	16	41,250 = 0xA122
1000 Hz	2	33,000 = 0x80E8
50 KHz	1	1,320 = 0x528
100 KHz	1	660 = 0x294

Selected Frequencies on TCU with 66 Mhz System Clock		
Desired Frequency	***Prescale***	***Preload***
1 MHz	1	66 = 0x42
2 MHz	1	33 = 0x21
6 MHz	1	11 = 0xB

Generally speaking, choose the smallest prescale you can, this gives better resolution on adjusting the preload and/or the pulse-width (e.g. at 1 Hz you have 1/64,453 resolution). Of course, for higher frequencies, the resolution is substantially reduced – though it is still 1/660 at 100 KHz.

Enabling Output Compare Channels

Timer/Counter Channel X Mode Register															
ICM		ICM IRQ EN	ICM PRE LOAD	OCM		OCM IRQ EN	PM	PWM	PWM Level			PIN CTRL	ICM SRC	COC	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Field Name	***Value***	***Comments***
ICM	0b00	Input Capture Mode disabled
ICM_IRQEN	0	No interrupts
ICM_PLOAD	0	No Input Capture
OCM	0b10	Drive output low on output compare
OCM_IRQEN	0	No interrupts
PWM	0	No Pulse-Width-**Measurement**
PWM_Level	0	
PM	1	Enable toggle on timer underflow
PIN_CTRL	1	Drive pin as output
ICM_SRC	0	
COC	0	Don't clear timer on output compare
Full Reg	**0x0908**	

The settings above will set up a channel to drive its output high on timer underflow, when it counts down to the value in the corresponding output compare register, it will drive the output low. To

invert the waveform, change the **OCM** field to **0b01**.

Now, all that remains is to set the appropriate value in the corresponding output compare register.

To calculate the desired value in the Output_Compare_Value:

*Output_Compare_Value = Preload_Value – (duty_cycle * Preload_Value);*

Output_Compare_Value = Preload_Value(1-duty_cycle);

A typical software implementation would have as input an unsigned fractional number representing the duty_cycle (i.e. 0x10000 == 1, 0x8000 == 0.5, etc.). So we have something like the following:

```
#include "FIDOmemmap.h"
int preloadValue = 0x80E8;
int *tcuCh0OutputCompare =  (int *)FIDO_TCU00_CH0_OUTPUTCOMPARE;
int UpdatePulseWidth (int dutyCycle, int channel)
{
  int factor;
  int compareValue;

  factor = 0x10000 – dutyCycle;
  compareValue = ((preloadValue * factor)+0x1000)>>16;

  tcuCh0OutputCompare[channel] = compareValue;
}

void InitPulseWidthModulation()
{

  // set up channel modes
  *FIDO_TCU_00_CH0_IOMODE = 0x0908
  *FIDO_TCU_00_CH1_IOMODE = 0x0908
  *FIDO_TCU_00_CH2_IOMODE = 0x0908
  *FIDO_TCU_00_CH3_IOMODE = 0x0908

  // set up compare values before enabling timer
  *FIDO_TCU_00_CH0_OUTPUTCOMPARE = INIT_PW_VALUE;
  *FIDO_TCU_00_CH1_OUTPUTCOMPARE = INIT_PW_VALUE;
  *FIDO_TCU_00_CH2_OUTPUTCOMPARE = INIT_PW_VALUE;
  *FIDO_TCU_00_CH3_OUTPUTCOMPARE = INIT_PW_VALUE;

  // load prescale
  *FIDO_TCU_00_CH0_COUNTER = 0x80E8;

  // load mode register and start operation
  *FIDO_TCU_00_CH0_MODE = 0x6100;
}
```

Conclusions

This Application Note shows how simple it is to use the **fido1100** Timer Counter as a 4-channel PWM.

Thank You

Thank you for taking the time to review this application note. We hope you have found the information included in this application useful and east to understand. Please feel free to contact the Innovasic Support Team any time with questions or comments.

Innovasic Support Team
3737 Princeton NE
Suite 130
Albuquerque, NM, 87107

(505) 883-5263

support@innovasic.com
http://www.innovasic.com