AM335x PWM Driver's Guide

AM335x PWM Driver's Guide

Translate this page to cs - Česky Translate

Google™ Custom Search Search

Navigation

- Main Page
- All pages
- All categories
- Popular pages
- Popular authors
- Popular categories
- Category stats
- Recent changes
- Random page
- Help
- Google Search

Print/export

- Create a book
- Download as PDF
- Printable version

Toolbox

- What links here
- Related changes
- Special pages
- Permanent link
- Browse properties

AM335X PWMSS Driver's Guide

Linux PSP

Contents [hide]

Introduction

The Pulse-Width Modulation Subsystem (PWMSS) consists of an instance of Enhanced High Resolution Pulse Width Modulator (eHRPWM), Enhanced Capture (eCAP), and Enhanced Quadrature Encoded Pulse (eQEP) modules. AM335x has 3-instances of PWMSS.

eHRPWM

The eHRPWM is an effective PWM peripheral, able to generate complex pulse width waveforms with minimal CPU overhead or intervention. The ePWM module represents one complete PWM channel composed of two PWM outputs: EPWMA and EPWMB.

1. Dedicated 16 bit time-base with Period / Frequency control
2. Two PWM outputs (EPWMA and EPWMB) that can be used in the following configurations
    1. Two independent PWM outputs with single-edge operation
    2. Two independent PWM outputs with dual-edge symmetric operation
    3. One independent PWM output with dual-edge asymmetric operation
3. Programmable event prescaling minimizes CPU overhead on interrupts.

eCAP

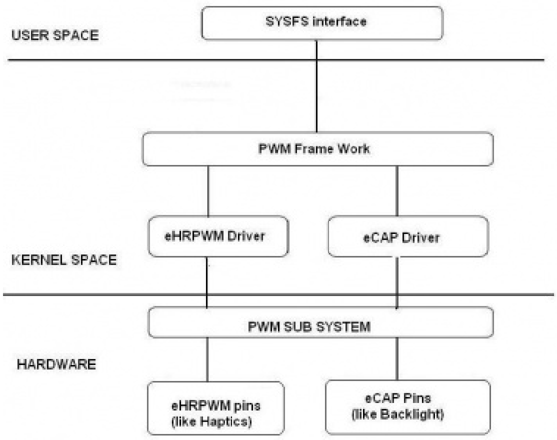
The eCAP module can function either in PWM mode or Capture mode.

1. Dedicated external pin
2. 32 bit Time Base counter
3. 4 x 32 bit Time-stamp Capture registers (CAP1-CAP4)
4. Input Capture signal pre-scaling
5. Interrupt capabilities on any of the 4 capture events

Current PWM Driver only supports below Features

1. only eHRPWM & eCAP is supported
2. only PWM mode is supported for eCAP.

PWMSS software architecture



Driver Configuration

Procedure to build eHRPWM driver

```
        Device Drivers --->
              <*> PWM Support --->
                 <*> Davinci eHRPWM support
```

Procedure to build eCAP driver

```
        Device Drivers --->
              <*> PWM Support --->
                 <*> eCAP PWM support
```

Driver Usage

IMPORTANT

- Due to heavy pin-muxing, few instances have pin-outs available on selected AM335x EVMs & profiles. Details about the availability of the peripherals can be found from the the EVM reference manual.

eHRPWM

eHRPWM can be controlled from the user space through SYSFS interface. SYSFS interface for eHRPWM is available at

```
/sys/class/pwm/ehrpwm.i:j
```

Where,

```
‘i’ is the eHRPWM instance
‘j’ is the eHRPWM ouptut 0 or 1[Each instance of eHRPWM has 2 channel outputs]
```

Various SYSFS Attributes

2 types of SYSFS attributes are available

1. Request and Control attributes
2. Configuration attributes

Note

- Below examples uses eHRPWM instance 2 and output channel 0 (i = 2, j = 0).

Type 1 attributes

- ***request*** Attribute.

Request PWM-SS for permission to use the module. Writing 1 to the request attribute Acquires the device and writing 0 to the request attribute Frees/Releases the device. Before performing any operations, device has to be requested first.

Example

- Request the Device:

```
target$ echo 1 > /sys/class/pwm/ehrpwm.2:0/request
```

- free the device:

```
target$ echo 0 > /sys/class/pwm/ehrpwm.2:0/request
```

- ***run*** Attribute

Start or Stop the PWM Module. 0 – Stop Module, 1 – Start module.

Example

- Start the PWM

```
target$ echo 1 > /sys/class/pwm/ehrpwm.2:0/run
```

- Stop the PWM

```
target$ echo 0 > /sys/class/pwm/ehrpwm.2:0/run
```

CAUTION
Before enabling the module, the module needs to be configured using below configuration attributes. Else proper operation is not assured.

Type 2 attributes

i.**Setting the Period**
Following attributes set the period of the PWM waveform.

- ***frequency*** Attribute

Enter the period in HZ. Once the frequency is configured, make sure to set the duty cycle value.

Example
Setup PWM to generate a pwm signal of 100 HZ with 50% duty cycle.

```
target$ echo 100 > /sys/class/pwm/ehrpwm.2:0/period_freq
target$ echo 50 > /sys/class/pwm/ehrpwm.2:0/duty_percent
```

- ***period_ns*** Attribute

Enter the period in nano seconds value.

Example
if the period is 1 sec , enter

```
target$ echo 1000000000 > /sys /class/pwm/ehrpwm.2:0/period_ns
```

ii.**Setting the Duty**
Following attributes set the duty of the PWM waveform.

- ***duty_percent*** Attribute

Enter the Duty cycle value in percentage.

Example
To configure for 50% duty cycle, enter

```
target$ echo 50 > /sys/class/pwm/ehrpwm.2:0/duty_percent
```

- ***duty_ns*** Attribute

duty_ns takes the duty cycle value in nano seconds.

Example
To configure for 0.5s, enter

```
target$ echo 500000000 > /sys /class/pwm/ehrpwm.2:0/duty_ns
```

iii.**Setting the Polarity**

- ***Polarity*** Attribute.

Setup Signal Polarity

Example
To set the polarity to Active High, Enter

```
target$ echo 1 > /sys /class/pwm/ehrpwm.2:0/polarity
```

Example
To set the polarity to Active Low, Enter

```
target$ echo 0 > /sys /class/pwm/ehrpwm.2:0/polarity
```

Example Configuration:
Following example shows steps to configure the PWM for 100 HZ with 50% duty cycle.

```
target$ echo 1 >  /sys/class/pwm/ehrpwm.2:0/request
target$ echo 100 > /sys/class/pwm/ehrpwm.2:0/period_freq
target$ echo 50 > /sys/class/pwm/ehrpwm.2:0/duty_percent
target$ echo 1 > /sys/class/pwm/ehrpwm.2:0/run
```

Issuing above commands will generate PWM wave. This can be verified by probing to eHRPWM pin.

IMPORTANT

1. To change the frequency, first stop the PWM waveform, reset the duty to zero and then follow the last 3 steps in the above

Example Configuration.

2. eHRPWM 2 instance channel 1 pwm pin is hooked to Haptics AM335x EVM. Changing the duty percentage/frequency will affect the speed of the Haptics.

Controlling haptics

Haptics on Evaluation Module can be controlled through eHRPWM sysfs interface.

> **IMPORTANT**
>
> In order to run Haptics using eHRPWM interface, the Evaluation Module needs to be in Profile 4.

```
target$ echo 1 > /sys/class/pwm/ehrpwm.2:1/request
target$ echo 250 > /sys/class/pwm/ehrpwm.2:1/period_freq
target$ echo 50 > /sys/class/pwm/ehrpwm.2:1/duty_percent
target$ echo 1 > /sys/class/pwm/ehrpwm.2:1/run
```

eCAP

The current release of the driver supports only PWM mode. eCAP can be controlled from the user space through SYSFS interface. SYSFS interface for eCAP is available at

```
target$ cat /sys/class/pwm/ecap.i
```

Where,

```
‘i’ is the eCAP instance.
```

Various SYSFS Attributes

2 types of SYSFS attributes are available

1. Request and Control attributes
2. Configuration attributes

Note

- Below examples uses eCAP instance 0 (i = 0).

Type 1 attributes

- ***request*** Attribute.

Request PWM-SS for permission to use the module. Writing 1 to the request attribute Acquires the device and writing 0 to the request attribute Frees/Releases the device. Before performing any operations, device has to be requested first.

Example

- Request the Device:

```
target$ echo 1 > /sys/class/pwm/ecap.0/request
```

- free the device:

```
target$ echo 0 > /sys/class/pwm/ecap.0/request
```

- ***run*** Attribute

Start or Stop the PWM Module. 0 – Stop Module, 1 – Start module.

Example

- Start the PWM

```
target$ echo 1 > /sys/class/pwm/ecap.0/run
```

- Stop the PWM

```
target$ echo 0 > /sys/class/pwm/ecap.0/run
```

CAUTION

Before enabling the module, the module needs to be configured using below configuration attributes. Else proper operation is not assured.

Type 2 attributes

i.**Setting the Period**

Following attributes set the period of the PWM waveform.

- ***frequency*** Attribute

Enter the period in HZ. Once the frequency is configured, make sure to set the duty cycle value.

Example

Setup PWM to generate a pwm signal of 100 HZ with 50% duty cycle.

```
target$ echo 100 > /sys/class/pwm/ecap.0/period_freq
target$ echo 50 > /sys/class/pwm/ecap.0/duty_percent
```

- ***period_ns*** Attribute

Enter the period in nano seconds value.

Example
if the period is 1 sec , enter

```
target$ echo 1000000000 > /sys /class/pwm/ecap.0/period_ns
```

ii.**Setting the Duty**
Following attributes set the duty of the PWM waveform.

- ***duty_percent*** Attribute

Enter the Duty cycle value in percentage.

Example
To configure for 50% duty cycle, enter

```
target$ echo 50 > /sys/class/pwm/ecap.0/duty_percent
```

- ***duty_ns*** Attribute

duty_ns takes the duty cycle value in nano seconds.

Example
To configure for 0.5s, enter

```
target$ echo 500000000 > /sys /class/pwm/ecap.0/duty_ns
```

iii.**Setting the Polarity**

- ***Polarity*** Attribute.

Setup Signal Polarity

Example
To set the polarity to Active High, Enter

```
target$ echo 1 > /sys /class/pwm/ecap.0/polarity
```

Example
To set the polarity to Active Low, Enter

```
target$ echo 0 > /sys /class/pwm/ecap.0/polarity
```

Example Configuration:
Following example shows steps to configure the PWM for 100 HZ with 50% duty cycle.

```
target$ echo 1 > /sys/class/pwm/ecap.0/request
target$ echo 100 > /sys/class/pwm/ecap.0/period_freq
target$ echo 50 > /sys/class/pwm/ecap.0/duty_percent
target$ echo 1 > /sys/class/pwm/ecap.0/run
```

Issuing above commands will generate PWM wave. This can be verified by probing to eCAP pin.

IMPORTANT

1. To change the frequency, first stop the PWM waveform, reset the duty to zero and then follow the last 3 steps in the above Example Configuration.
2. In this EVM, eCAP instance 0 pin is hooked to backlight of the LCD. Changing the duty percentage will affect the brightness of the LCD screen.

Controlling backlight

Following are the 2 procedures to vary brightness of the LCD screen.

i. Setting duty percentage of pwm wave from eCAP sysfs files

```
target$ echo val > /sys/class/pwm/ecap.0/duty_percent
```

'val' can range from 0 to 100.
ii. Setting brightness from backlight sysfs files

```
target$ echo val > /sys/class/backlight/pwm-backlight/brightness
```

'val' can range from 0 to 100.

For technical support please

*post your questions at http://e2e.ti.com. Please post only comments about the article **AM335x PWM Driver's Guide** here.*

Links

ARM Microcontroller MCU
ARM Processor
Digital Media Processor
Digital Signal Processing
Microcontroller MCU
Multi Core Processor
Ultra Low Power DSP
8 bit Microcontroller MCU
16 bit Microcontroller MCU
32 bit Microcontroller MCU

Categories: AM335x | Linux | PSP

Leave a Comment

This page was last modified on 19 March 2012, at 10:22.

This page has been accessed 3,048 times.