



LPTIM device tree configuration



Contents

1. LPTIM device tree configuration	3
2. LPTIM Linux driver	7
3. PWM overview	12
4. IIO overview	17
5. Device tree	22
6. LPTIM internal peripheral	27
7. Pinctrl overview	32
8. Pinctrl device tree configuration	37
9. GPIO internal peripheral	42
10. STM32CubeMX	47
11. ADC internal peripheral	52
12. DAC internal peripheral	57
13. DFSDM internal peripheral	62



LPTIM device tree configuration

Stable: 15.10.2019 - 14:30 / Revision: 15.10.2019 - 14:26

A quality version of this page, [accepted](#) on 15 October 2019, was based off this revision.

Contents

1 Article purpose	3
2 DT bindings documentation	3
3 DT configuration	4
3.1 DT configuration (STM32 level)	4
3.2 DT configuration (board level)	5
3.3 DT configuration examples	5
3.3.1 LPTIM1 configured as PWM and trigger source	5
3.3.2 LPTIM2 configured as counter and quadrature encoder	6
4 How to configure the DT using STM32CubeMX	7
5 References	7

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux[®]OS**, and in particular:

- how to configure the **LPTIM peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

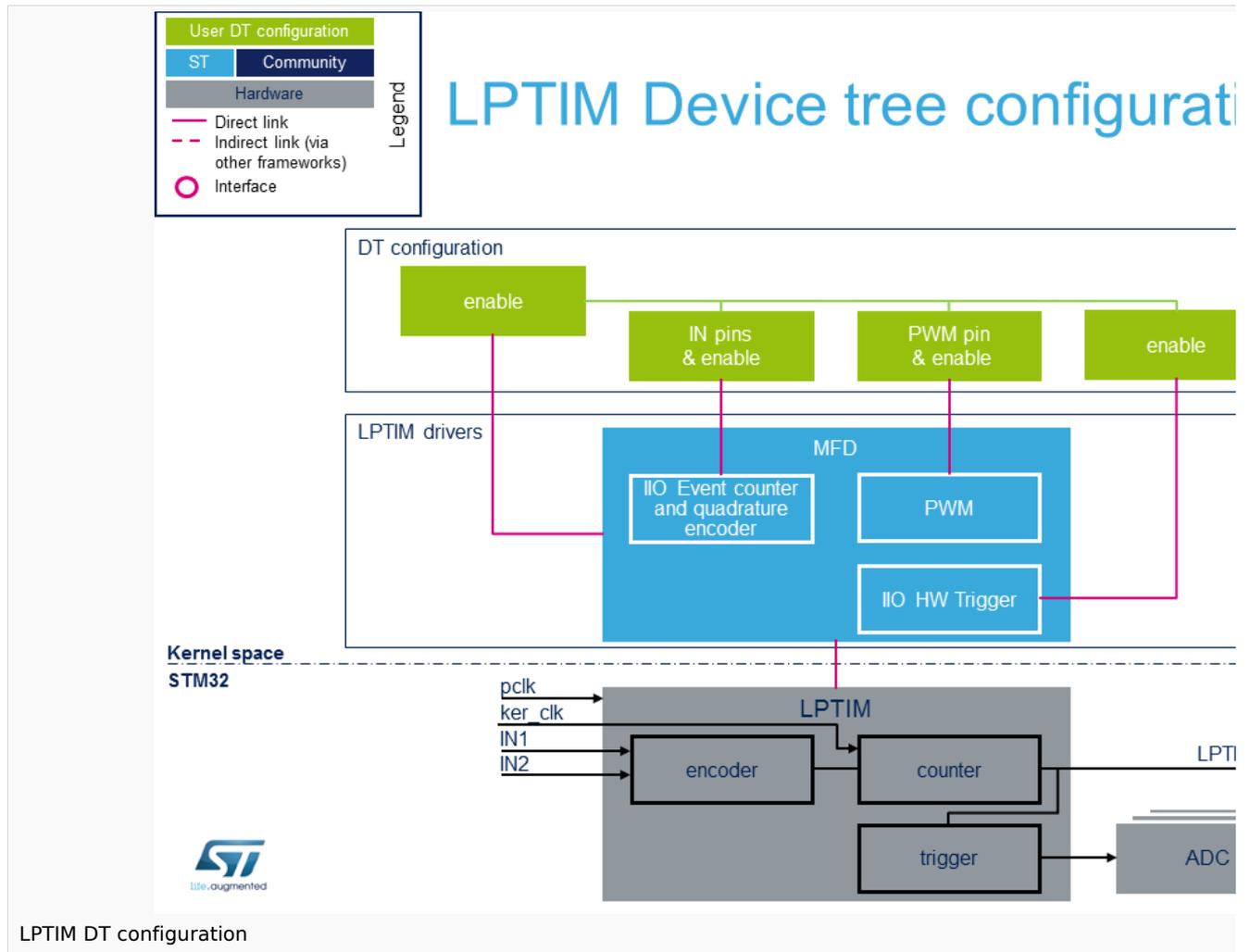
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";                                /* Lptimer's common
resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";                            /* PWM part of LPTIM */
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";                    /* trigger part of LPTIM
*/
        reg = <0>;                                                /* trigger identifier (e.
g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    };
    counter {
        compatible = "st,stm32-lptimer-counter";                    /* quadrature encoder &
event counter part of LPTIM */
    };
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via `pinctrl`, with `pinctrl-0`, `pinctrl-1` and `pinctrl-names`.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See [pinctrl device tree configuration](#) and [GPIO internal peripheral](#))
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as ADC^[8], DAC^[9], and DFSDM^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>;    /* configure 'PG13' as
alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```



LPTIM device tree configuration

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

For additional information, refer to the following links:

- [1.01.1 LPTIM internal peripheral](#)
- [Device tree](#)
- [Documentation/devicetree/bindings/mfd/stm32-lptimer.txt](#) , STM32 LPTIM MFD device tree bindings
- [Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt](#) , STM32 LPTIM PWM device tree bindings
- [Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt](#) , STM32 LPTIM trigger device tree bindings
- [Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt](#) , STM32 LPTIM counter/encoder device tree bindings
- [STM32MP157C device tree file](#)
- [ADC internal peripheral](#)
- [DAC internal peripheral](#)
- [DFSDM internal peripheral](#)

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	8
2 DT bindings documentation	8
3 DT configuration	9
3.1 DT configuration (STM32 level)	9
3.2 DT configuration (board level)	10
3.3 DT configuration examples	10
3.3.1 LPTIM1 configured as PWM and trigger source	10
3.3.2 LPTIM2 configured as counter and quadrature encoder	11
4 How to configure the DT using STM32CubeMX	12
5 References	12

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux®OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

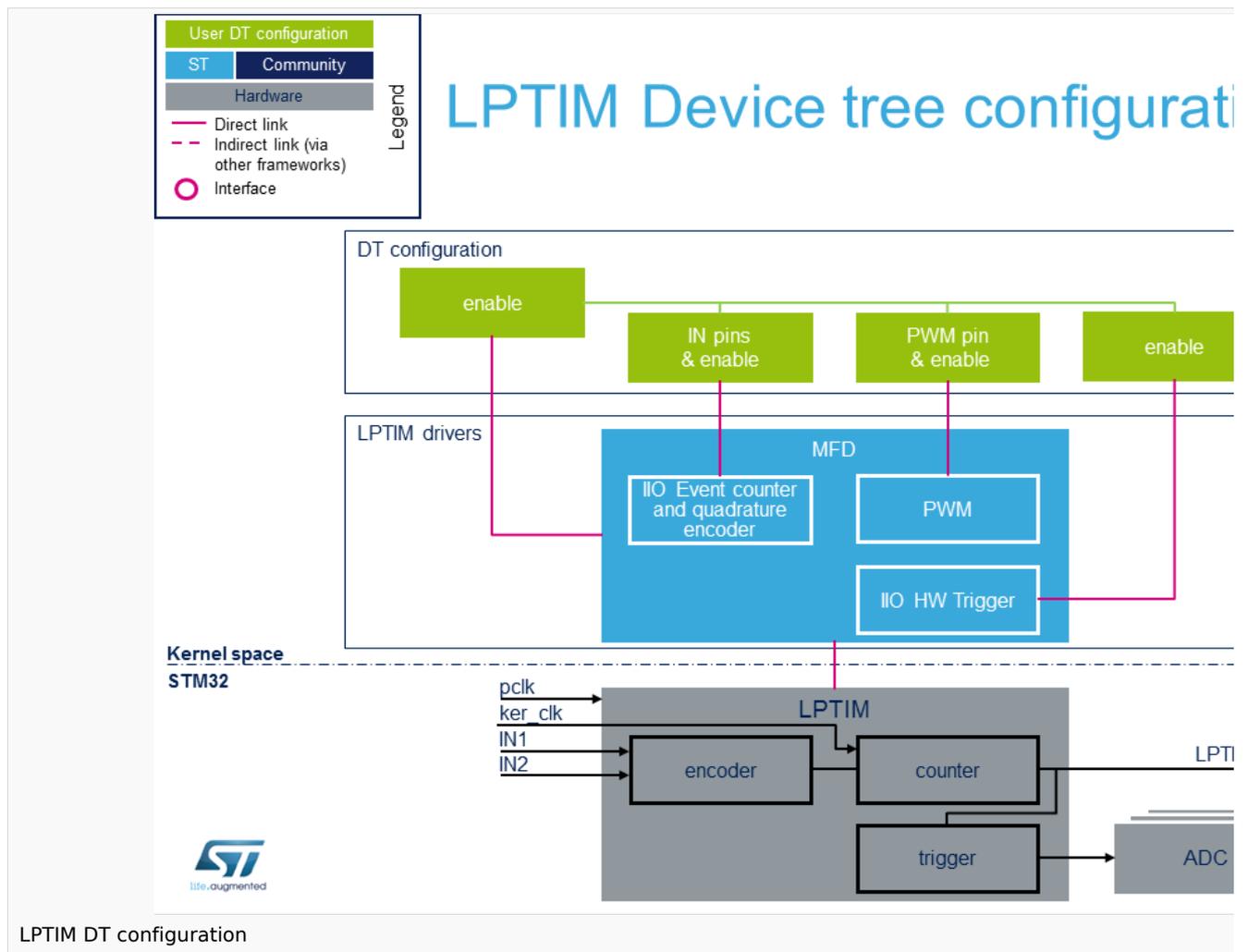
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See pinctrl device tree configuration and GPIO internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as ADC^[8], DAC^[9], and DFSDM^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

For additional information, refer to the following links:

- [1.01.1 LPTIM internal peripheral](#)
- [Device tree](#)
- [Documentation/devicetree/bindings/mfd/stm32-lptimer.txt](#) , STM32 LPTIM MFD device tree bindings
- [Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt](#) , STM32 LPTIM PWM device tree bindings
- [Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt](#) , STM32 LPTIM trigger device tree bindings
- [Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt](#) , STM32 LPTIM counter/encoder device tree bindings
- [STM32MP157C device tree file](#)
- [ADC internal peripheral](#)
- [DAC internal peripheral](#)
- [DFSDM internal peripheral](#)

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	13
2 DT bindings documentation	13
3 DT configuration	14
3.1 DT configuration (STM32 level)	14
3.2 DT configuration (board level)	15
3.3 DT configuration examples	15
3.3.1 LPTIM1 configured as PWM and trigger source	15
3.3.2 LPTIM2 configured as counter and quadrature encoder	16
4 How to configure the DT using STM32CubeMX	17
5 References	17

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux®OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

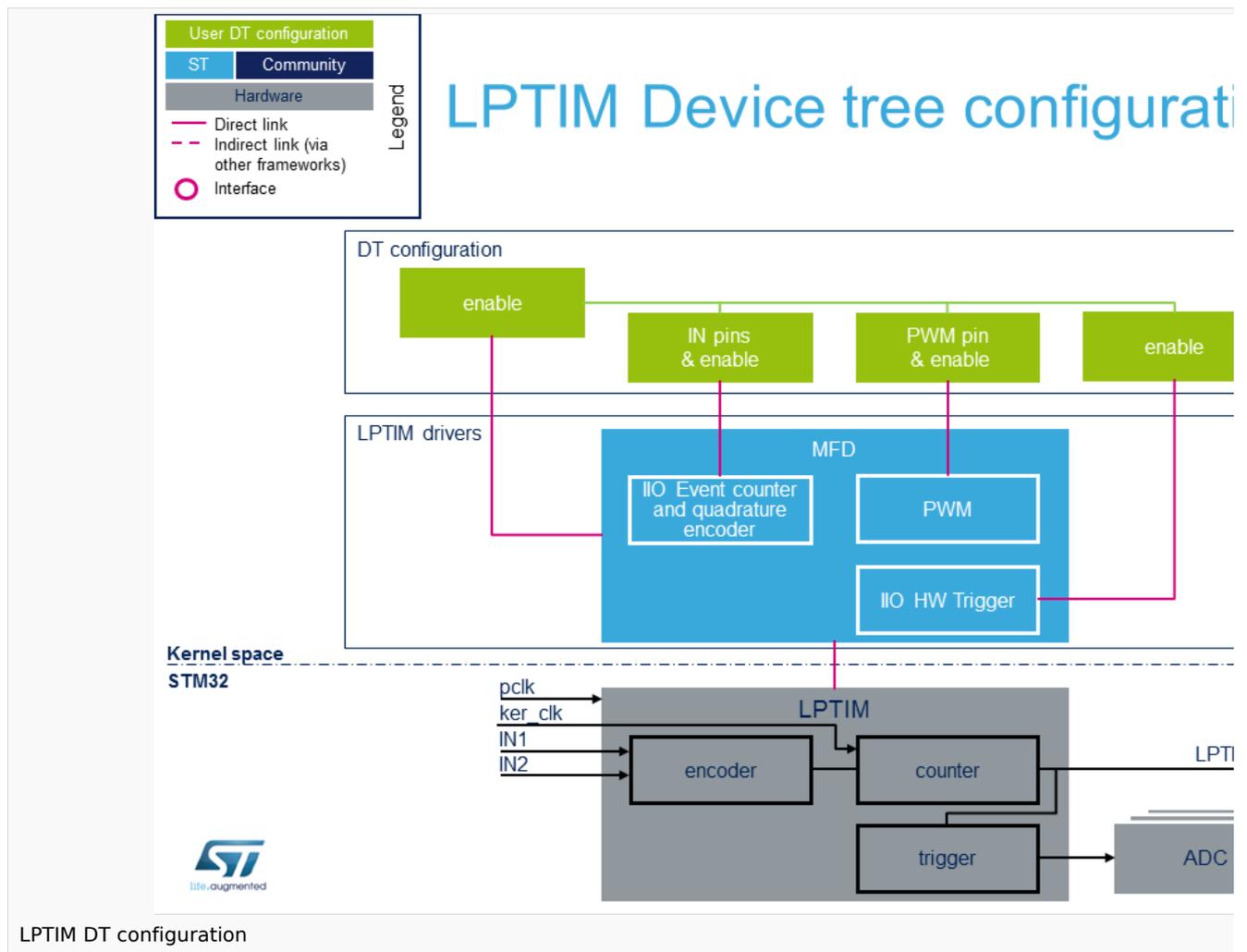
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See pinctrl device tree configuration and GPIO internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as ADC^[8], DAC^[9], and DFSDM^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```



LPTIM device tree configuration

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The [STM32CubeMX](#) tool can be used to configure the STM32MPU device and get the corresponding [platform configuration device tree files](#).

The [STM32CubeMX](#) may not support all the properties described in the above [DT bindings documentation](#) paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to [STM32CubeMX user manual](#) for further information.

5 References

For additional information, refer to the following links:

- [1.01.1 LPTIM internal peripheral](#)
- [Device tree](#)
- [Documentation/devicetree/bindings/mfd/stm32-lptimer.txt](#) , STM32 LPTIM MFD device tree bindings
- [Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt](#) , STM32 LPTIM PWM device tree bindings
- [Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt](#) , STM32 LPTIM trigger device tree bindings
- [Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt](#) , STM32 LPTIM counter/encoder device tree bindings
- [STM32MP157C device tree file](#)
- [ADC internal peripheral](#)
- [DAC internal peripheral](#)
- [DFSDM internal peripheral](#)

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	18
2 DT bindings documentation	18
3 DT configuration	19
3.1 DT configuration (STM32 level)	19
3.2 DT configuration (board level)	20
3.3 DT configuration examples	20
3.3.1 LPTIM1 configured as PWM and trigger source	20
3.3.2 LPTIM2 configured as counter and quadrature encoder	21
4 How to configure the DT using STM32CubeMX	22
5 References	22

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux®OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

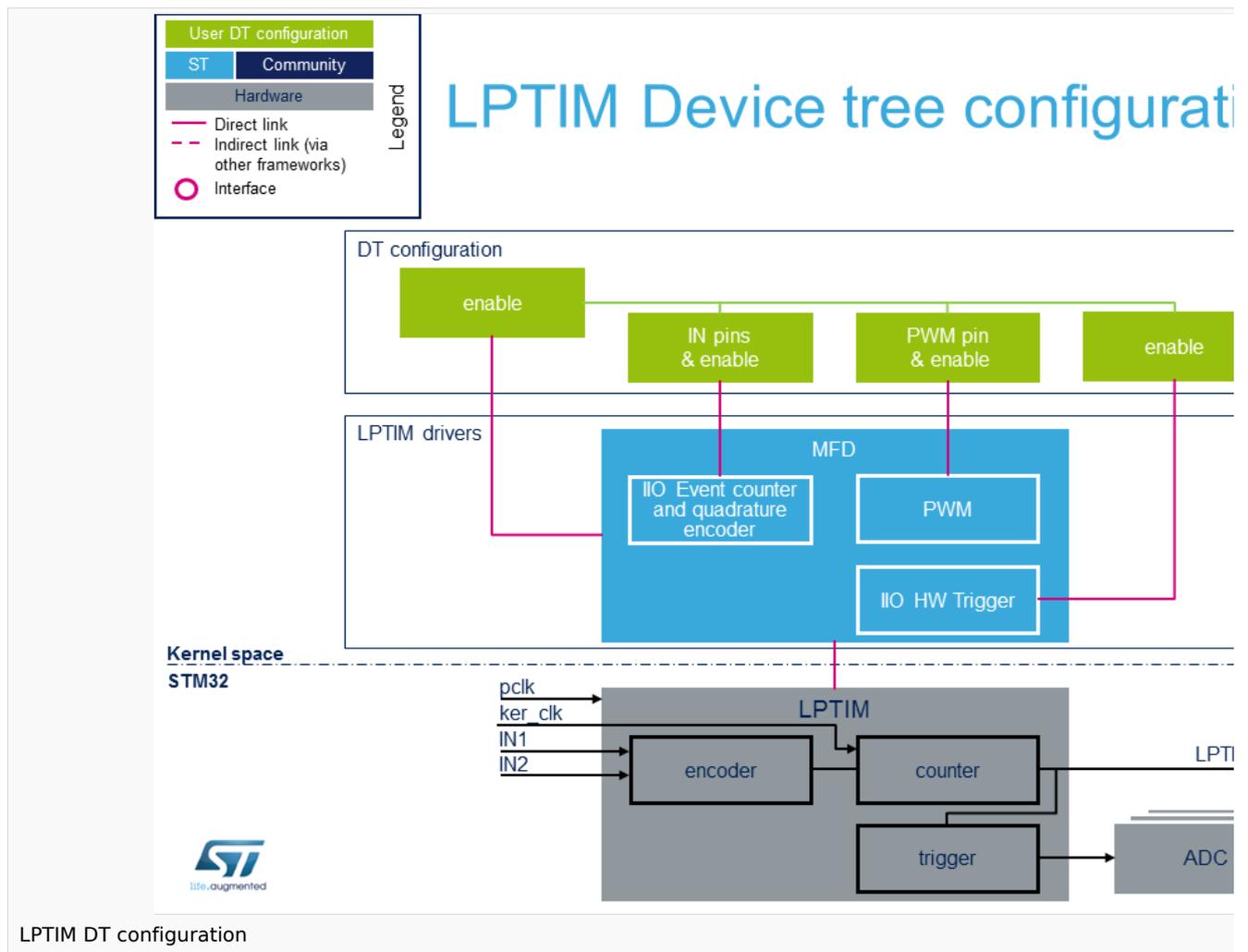
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See pinctrl device tree configuration and GPIO internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as ADC^[8], DAC^[9], and DFSDM^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

For additional information, refer to the following links:

- 1.01.1 LPTIM internal peripheral
- Device tree
- Documentation/devicetree/bindings/mfd/stm32-lptimer.txt , STM32 LPTIM MFD device tree bindings
- Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt , STM32 LPTIM PWM device tree bindings
- Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt , STM32 LPTIM trigger device tree bindings
- Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt , STM32 LPTIM counter/encoder device tree bindings
- STM32MP157C device tree file
- ADC internal peripheral
- DAC internal peripheral
- DFSDM internal peripheral

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	23
2 DT bindings documentation	23
3 DT configuration	24
3.1 DT configuration (STM32 level)	24
3.2 DT configuration (board level)	25
3.3 DT configuration examples	25
3.3.1 LPTIM1 configured as PWM and trigger source	25
3.3.2 LPTIM2 configured as counter and quadrature encoder	26
4 How to configure the DT using STM32CubeMX	27
5 References	27

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux®OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

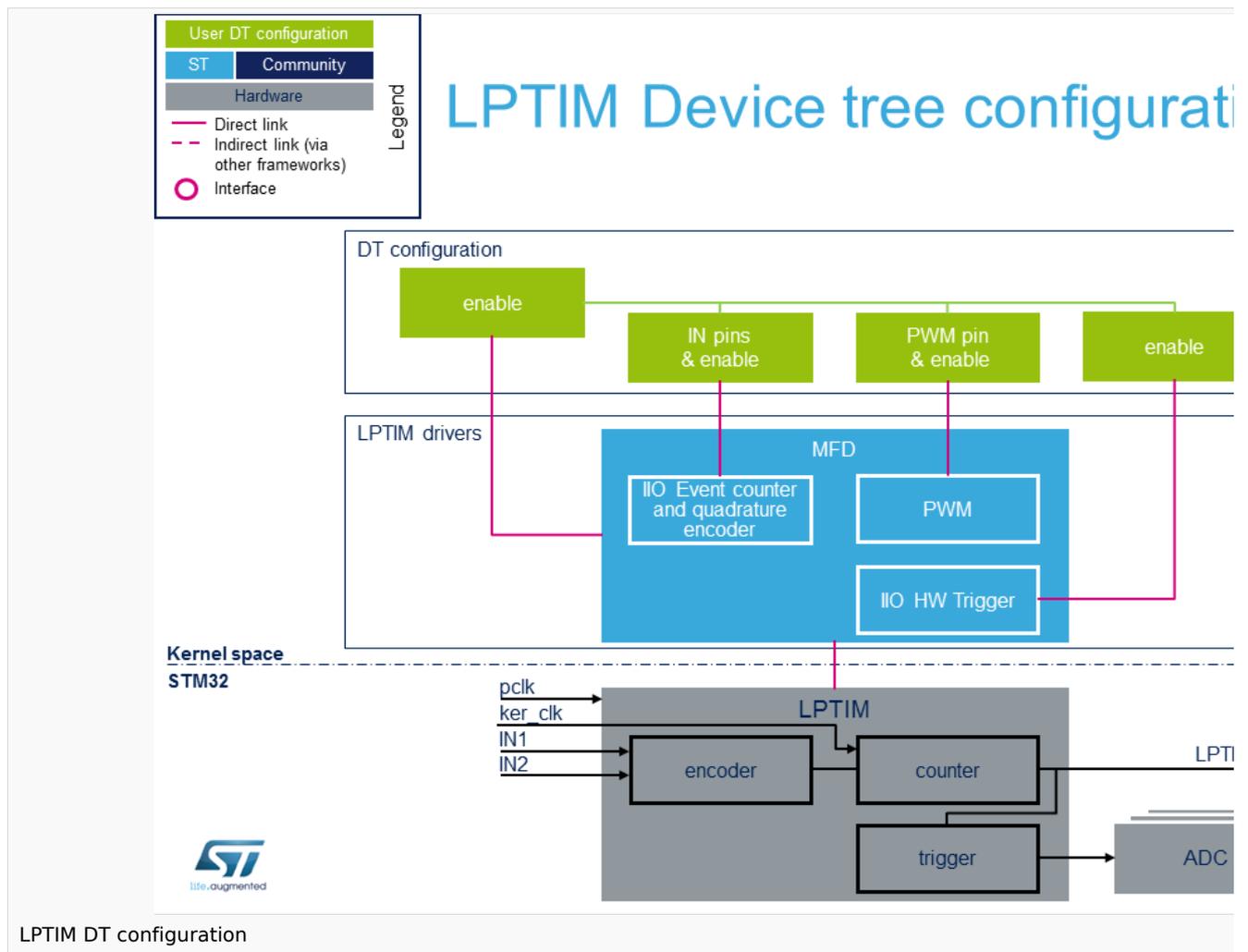
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See pinctrl device tree configuration and GPIO internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as ADC^[8], DAC^[9], and DFSDM^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```



LPTIM device tree configuration

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding [platform configuration device tree files](#).

The STM32CubeMX may not support all the properties described in the above [DT bindings documentation](#) paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to [STM32CubeMX user manual](#) for further information.

5 References

For additional information, refer to the following links:

- [1.01.1 LPTIM internal peripheral](#)
- [Device tree](#)
- [Documentation/devicetree/bindings/mfd/stm32-lptimer.txt](#) , STM32 LPTIM MFD device tree bindings
- [Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt](#) , STM32 LPTIM PWM device tree bindings
- [Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt](#) , STM32 LPTIM trigger device tree bindings
- [Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt](#) , STM32 LPTIM counter/encoder device tree bindings
- [STM32MP157C device tree file](#)
- [ADC internal peripheral](#)
- [DAC internal peripheral](#)
- [DFSDM internal peripheral](#)

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	28
2 DT bindings documentation	28
3 DT configuration	29
3.1 DT configuration (STM32 level)	29
3.2 DT configuration (board level)	30
3.3 DT configuration examples	30
3.3.1 LPTIM1 configured as PWM and trigger source	30
3.3.2 LPTIM2 configured as counter and quadrature encoder	31
4 How to configure the DT using STM32CubeMX	32
5 References	32

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux®OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

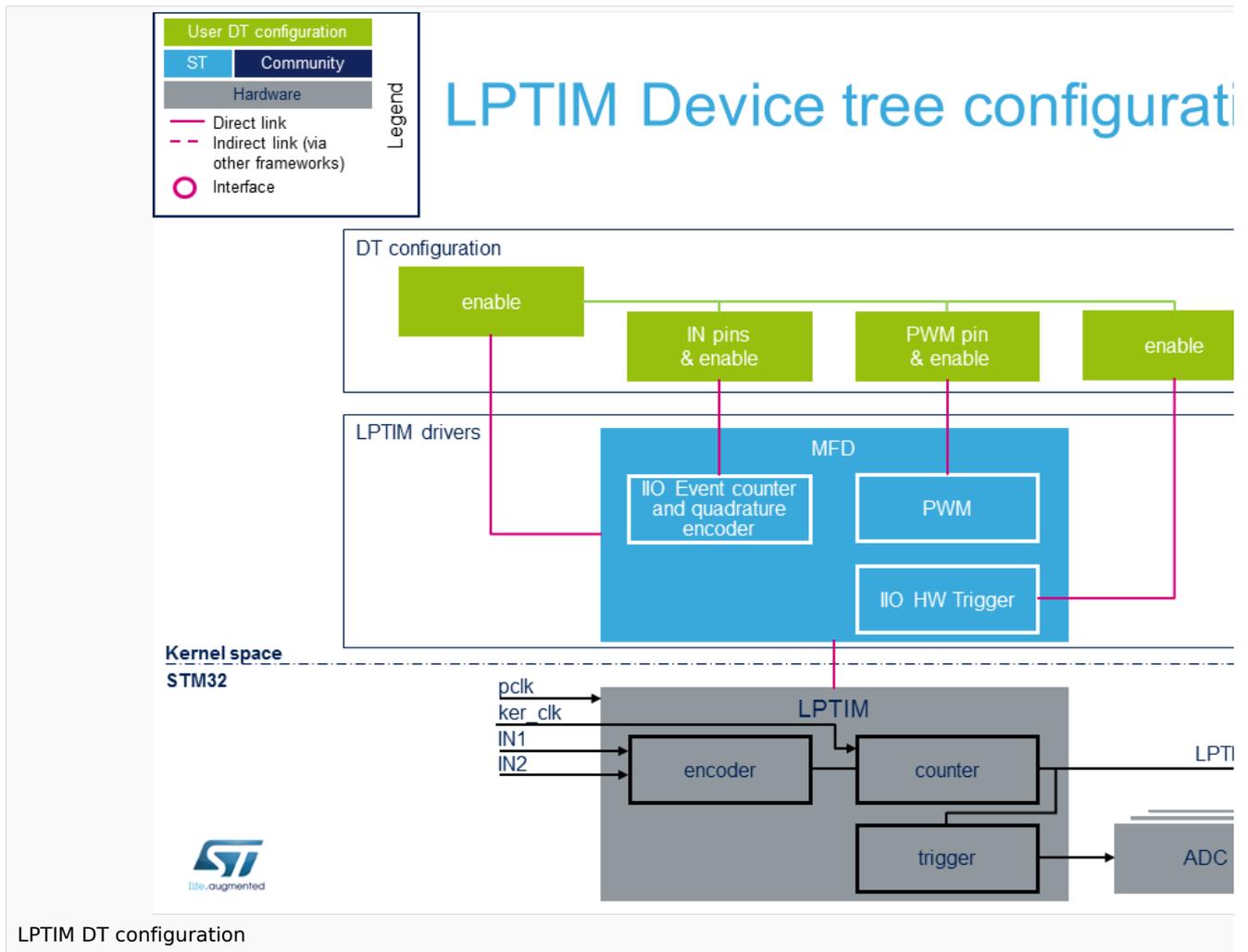
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See **pinctrl** device tree configuration and **GPIO** internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as **ADC**^[8], **DAC**^[9], and **DFSDM**^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

For additional information, refer to the following links:

- 1.01.1 LPTIM internal peripheral
- Device tree
- Documentation/devicetree/bindings/mfd/stm32-lptimer.txt , STM32 LPTIM MFD device tree bindings
- Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt , STM32 LPTIM PWM device tree bindings
- Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt , STM32 LPTIM trigger device tree bindings
- Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt , STM32 LPTIM counter/encoder device tree bindings
- STM32MP157C device tree file
- ADC internal peripheral
- DAC internal peripheral
- DFSDM internal peripheral

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	33
2 DT bindings documentation	33
3 DT configuration	34
3.1 DT configuration (STM32 level)	34
3.2 DT configuration (board level)	35
3.3 DT configuration examples	35
3.3.1 LPTIM1 configured as PWM and trigger source	35
3.3.2 LPTIM2 configured as counter and quadrature encoder	36
4 How to configure the DT using STM32CubeMX	37
5 References	37

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux®OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

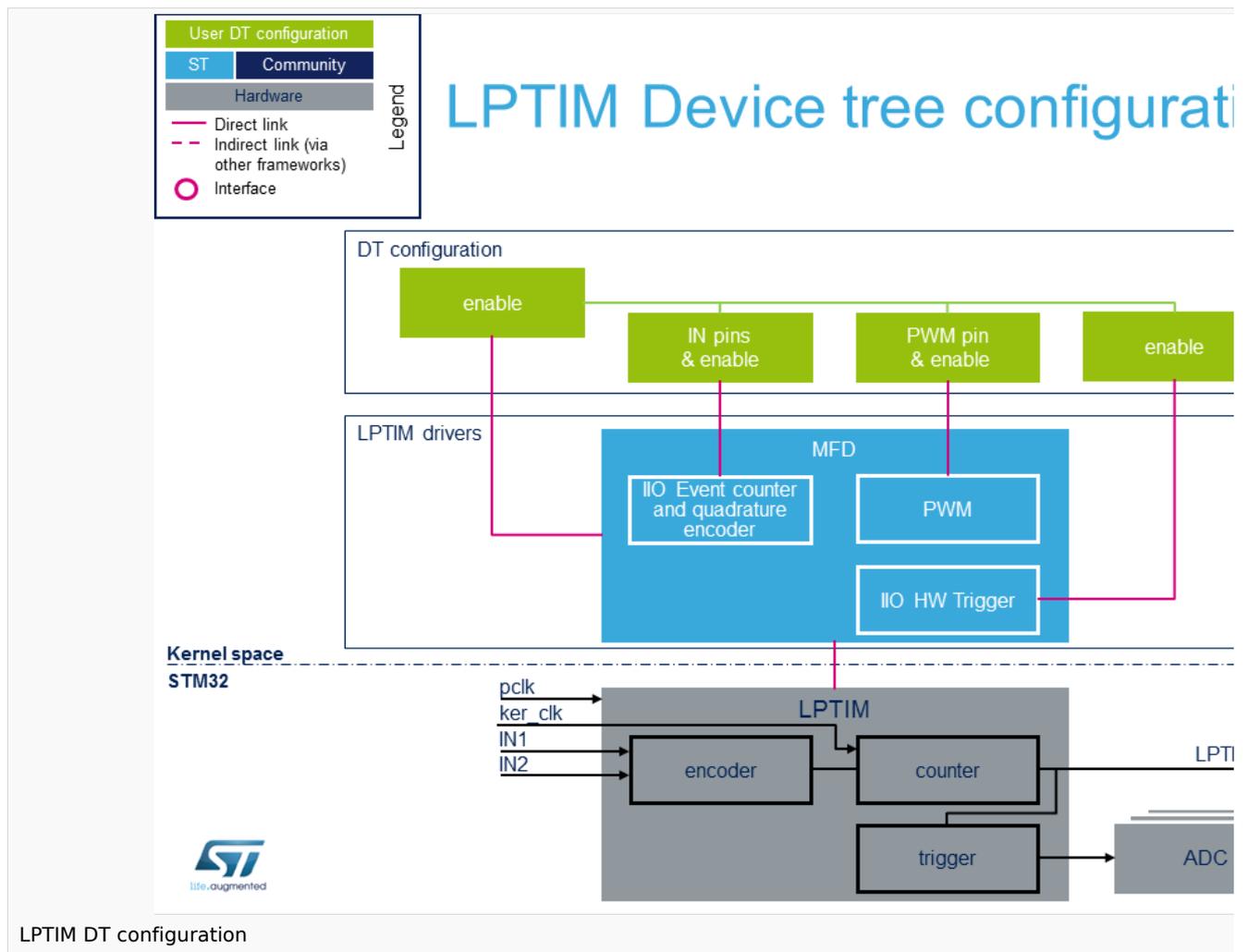
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
    resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See pinctrl device tree configuration and GPIO internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as ADC^[8], DAC^[9], and DFSDM^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
    alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

For additional information, refer to the following links:

- 1.01.1 LPTIM internal peripheral
- Device tree
- Documentation/devicetree/bindings/mfd/stm32-lptimer.txt , STM32 LPTIM MFD device tree bindings
- Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt , STM32 LPTIM PWM device tree bindings
- Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt , STM32 LPTIM trigger device tree bindings
- Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt , STM32 LPTIM counter/encoder device tree bindings
- STM32MP157C device tree file
- ADC internal peripheral
- DAC internal peripheral
- DFSDM internal peripheral

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	38
2 DT bindings documentation	38
3 DT configuration	39
3.1 DT configuration (STM32 level)	39
3.2 DT configuration (board level)	40
3.3 DT configuration examples	40
3.3.1 LPTIM1 configured as PWM and trigger source	40
3.3.2 LPTIM2 configured as counter and quadrature encoder	41
4 How to configure the DT using STM32CubeMX	42
5 References	42

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux®OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

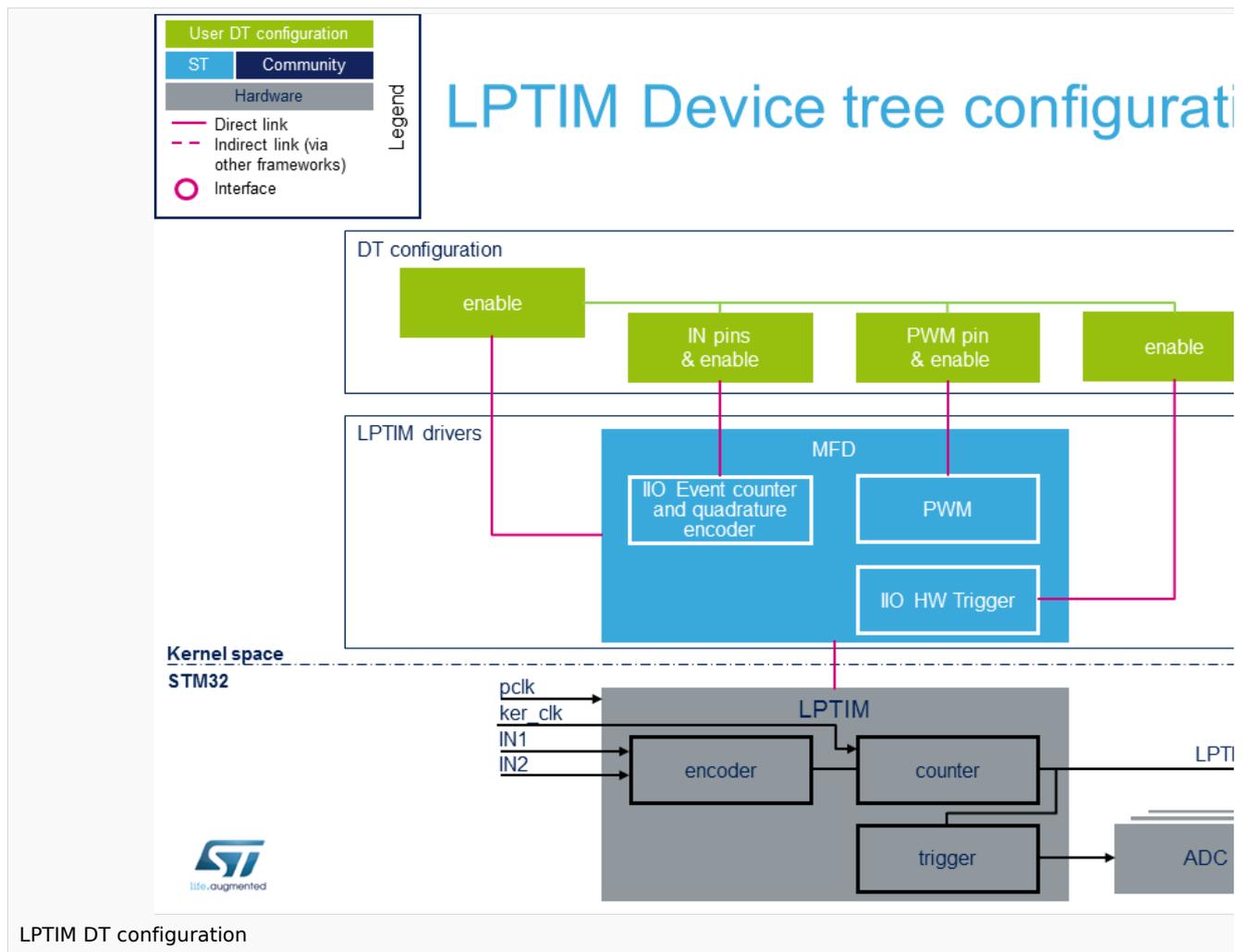
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
    resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See pinctrl device tree configuration and GPIO internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as ADC^[8], DAC^[9], and DFSDM^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
    alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```



LPTIM device tree configuration

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

For additional information, refer to the following links:

- 1.01.1 LPTIM internal peripheral
- Device tree
- Documentation/devicetree/bindings/mfd/stm32-lptimer.txt , STM32 LPTIM MFD device tree bindings
- Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt , STM32 LPTIM PWM device tree bindings
- Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt , STM32 LPTIM trigger device tree bindings
- Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt , STM32 LPTIM counter/encoder device tree bindings
- STM32MP157C device tree file
- ADC internal peripheral
- DAC internal peripheral
- DFSDM internal peripheral

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	43
2 DT bindings documentation	43
3 DT configuration	44
3.1 DT configuration (STM32 level)	44
3.2 DT configuration (board level)	45
3.3 DT configuration examples	45
3.3.1 LPTIM1 configured as PWM and trigger source	45
3.3.2 LPTIM2 configured as counter and quadrature encoder	46
4 How to configure the DT using STM32CubeMX	47
5 References	47

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux®OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

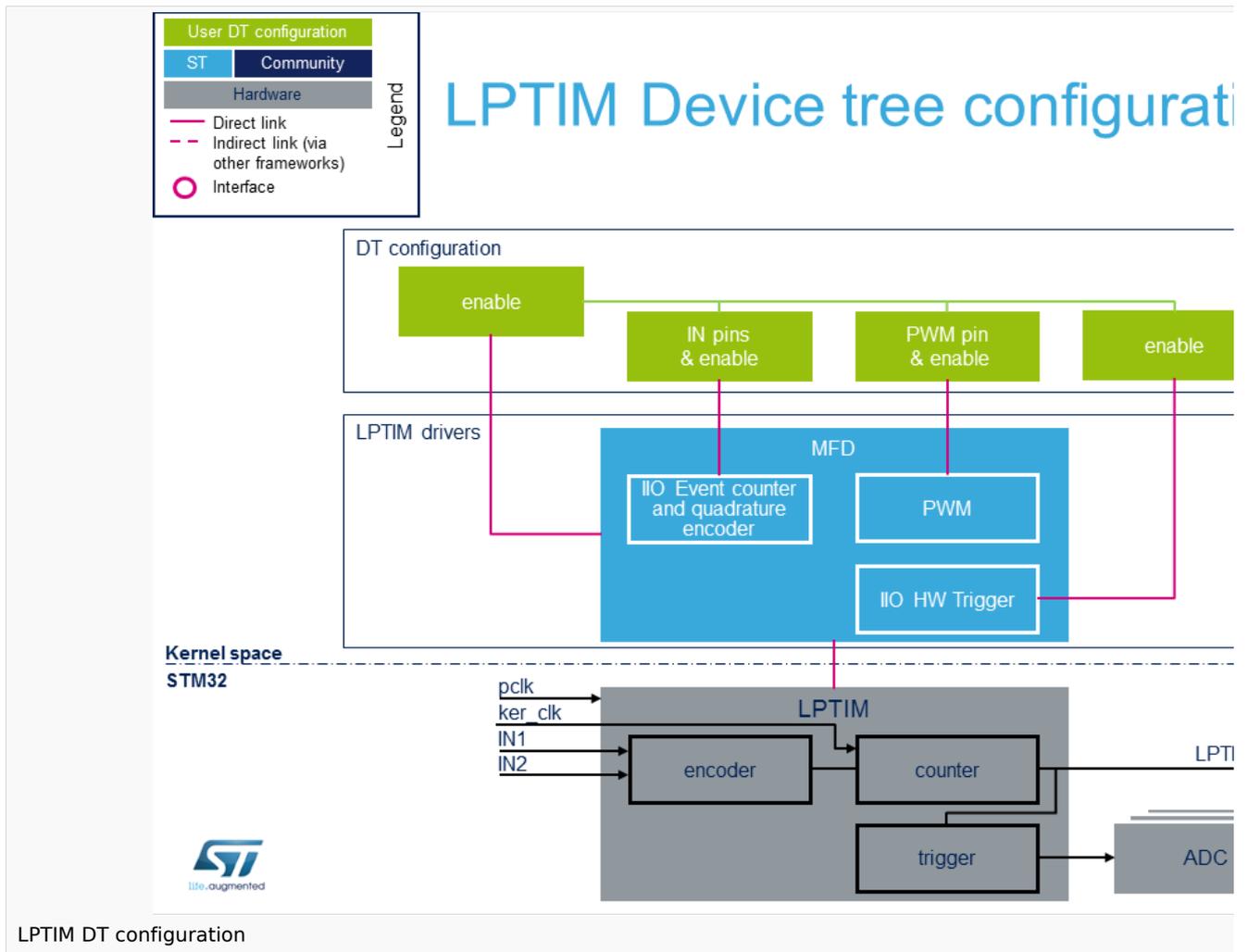
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See pinctrl device tree configuration and GPIO internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as ADC^[8], DAC^[9], and DFSDM^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```

```

lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};

```

```

&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
        status = "okay"; /* enable LPTIM1_OUT trigger
source */
    };
};

```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```

# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};

```

```

&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};

```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

For additional information, refer to the following links:

- [1.01.1 LPTIM internal peripheral](#)
- [Device tree](#)
- [Documentation/devicetree/bindings/mfd/stm32-lptimer.txt](#) , STM32 LPTIM MFD device tree bindings
- [Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt](#) , STM32 LPTIM PWM device tree bindings
- [Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt](#) , STM32 LPTIM trigger device tree bindings
- [Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt](#) , STM32 LPTIM counter/encoder device tree bindings
- [STM32MP157C device tree file](#)
- [ADC internal peripheral](#)
- [DAC internal peripheral](#)
- [DFSDM internal peripheral](#)

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	48
2 DT bindings documentation	48
3 DT configuration	49
3.1 DT configuration (STM32 level)	49
3.2 DT configuration (board level)	50
3.3 DT configuration examples	50
3.3.1 LPTIM1 configured as PWM and trigger source	50
3.3.2 LPTIM2 configured as counter and quadrature encoder	51
4 How to configure the DT using STM32CubeMX	52
5 References	52

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux[®]OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

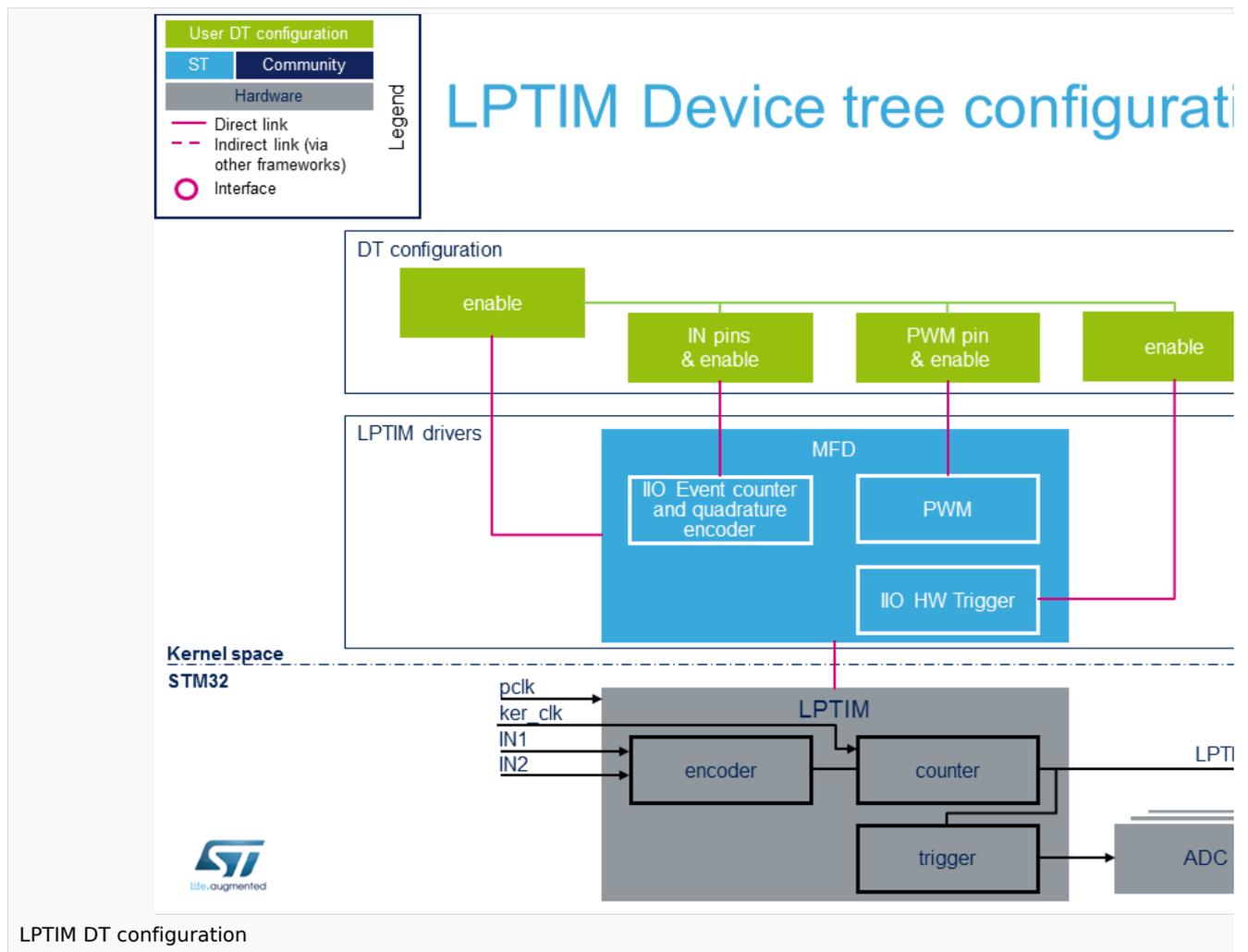
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
    resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See **pinctrl** device tree configuration and **GPIO** internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as **ADC**^[8], **DAC**^[9], and **DFSDM**^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
    alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

For additional information, refer to the following links:

- 1.01.1 LPTIM internal peripheral
- Device tree
- Documentation/devicetree/bindings/mfd/stm32-lptimer.txt , STM32 LPTIM MFD device tree bindings
- Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt , STM32 LPTIM PWM device tree bindings
- Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt , STM32 LPTIM trigger device tree bindings
- Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt , STM32 LPTIM counter/encoder device tree bindings
- STM32MP157C device tree file
- ADC internal peripheral
- DAC internal peripheral
- DFSDM internal peripheral

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	53
2 DT bindings documentation	53
3 DT configuration	54
3.1 DT configuration (STM32 level)	54
3.2 DT configuration (board level)	55
3.3 DT configuration examples	55
3.3.1 LPTIM1 configured as PWM and trigger source	55
3.3.2 LPTIM2 configured as counter and quadrature encoder	56
4 How to configure the DT using STM32CubeMX	57
5 References	57

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux®OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

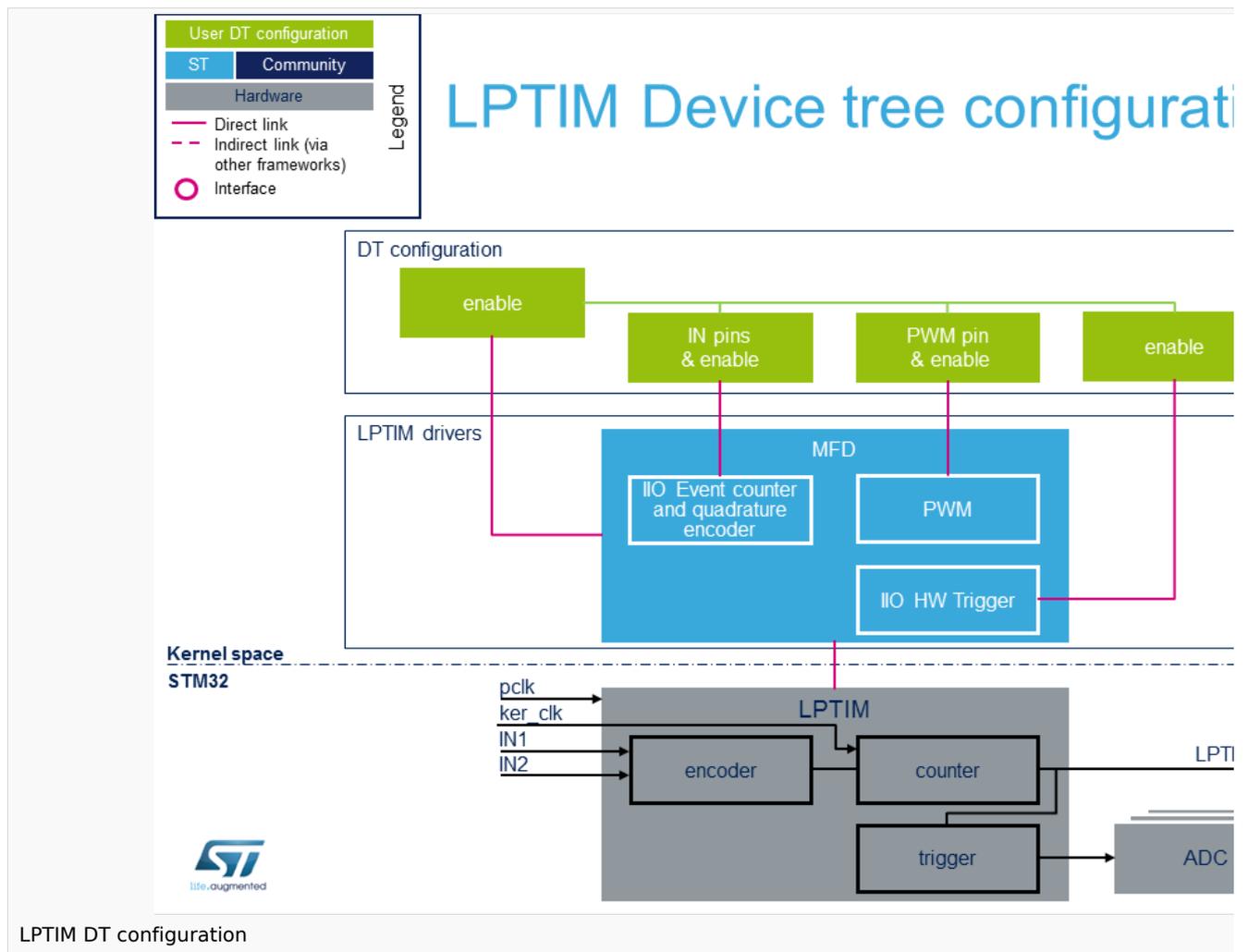
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See pinctrl device tree configuration and GPIO internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as ADC^[8], DAC^[9], and DFSDM^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```



LPTIM device tree configuration

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

For additional information, refer to the following links:

- [1.01.1 LPTIM internal peripheral](#)
- [Device tree](#)
- [Documentation/devicetree/bindings/mfd/stm32-lptimer.txt](#) , STM32 LPTIM MFD device tree bindings
- [Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt](#) , STM32 LPTIM PWM device tree bindings
- [Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt](#) , STM32 LPTIM trigger device tree bindings
- [Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt](#) , STM32 LPTIM counter/encoder device tree bindings
- [STM32MP157C device tree file](#)
- [ADC internal peripheral](#)
- [DAC internal peripheral](#)
- [DFSDM internal peripheral](#)

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	58
2 DT bindings documentation	58
3 DT configuration	59
3.1 DT configuration (STM32 level)	59
3.2 DT configuration (board level)	60
3.3 DT configuration examples	60
3.3.1 LPTIM1 configured as PWM and trigger source	60
3.3.2 LPTIM2 configured as counter and quadrature encoder	61
4 How to configure the DT using STM32CubeMX	62
5 References	62

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux®OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

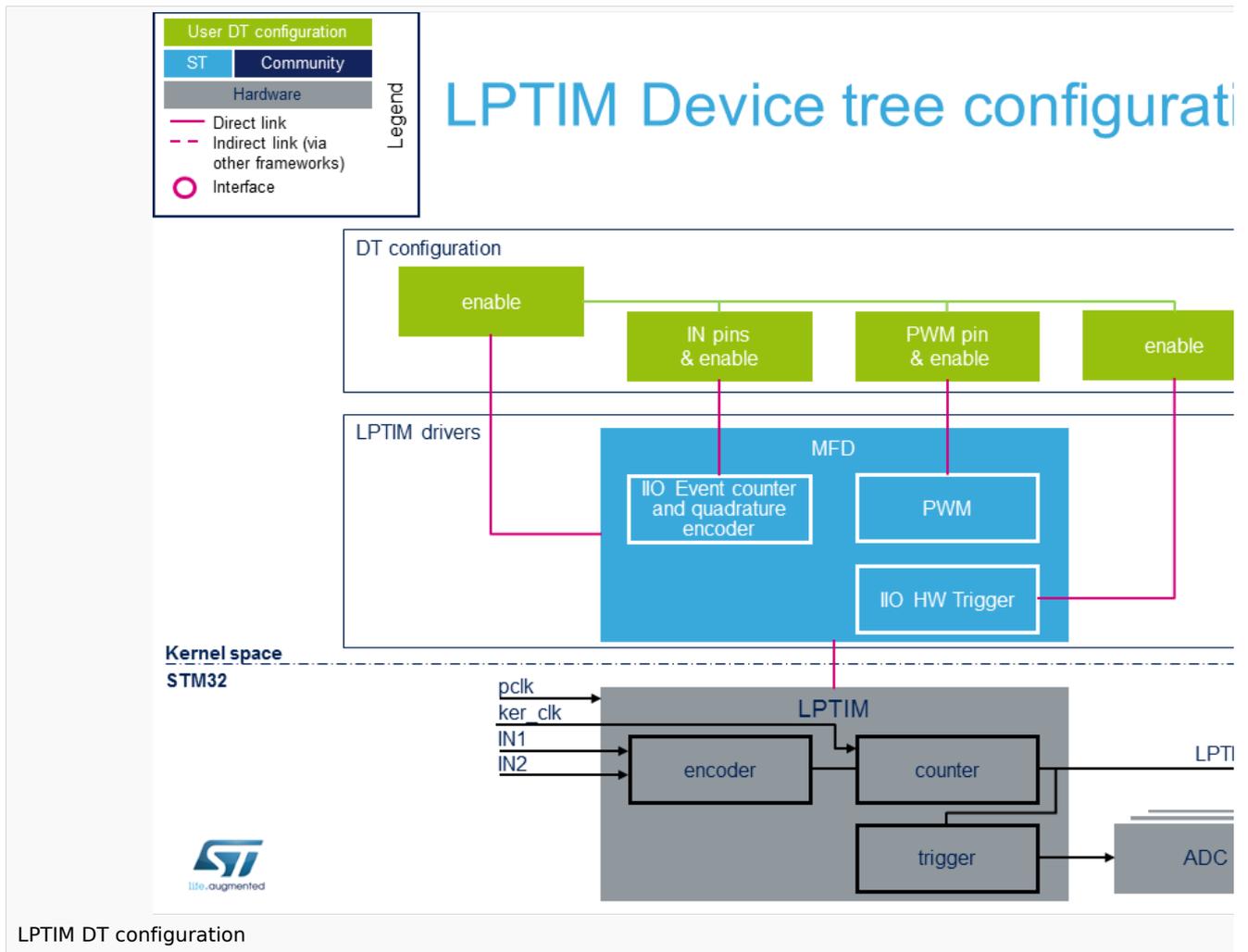
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See pinctrl device tree configuration and GPIO internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as ADC^[8], DAC^[9], and DFSDM^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

For additional information, refer to the following links:

- [1.01.1 LPTIM internal peripheral](#)
- [Device tree](#)
- [Documentation/devicetree/bindings/mfd/stm32-lptimer.txt](#) , STM32 LPTIM MFD device tree bindings
- [Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt](#) , STM32 LPTIM PWM device tree bindings
- [Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt](#) , STM32 LPTIM trigger device tree bindings
- [Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt](#) , STM32 LPTIM counter/encoder device tree bindings
- [STM32MP157C device tree file](#)
- [ADC internal peripheral](#)
- [DAC internal peripheral](#)
- [DFSDM internal peripheral](#)

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator

LPTIM device tree configuration



Contents

1 Article purpose	63
2 DT bindings documentation	63
3 DT configuration	64
3.1 DT configuration (STM32 level)	64
3.2 DT configuration (board level)	65
3.3 DT configuration examples	65
3.3.1 LPTIM1 configured as PWM and trigger source	65
3.3.2 LPTIM2 configured as counter and quadrature encoder	66
4 How to configure the DT using STM32CubeMX	67
5 References	67

1 Article purpose

The purpose of this article is to explain how to configure the low-power timer (*LPTIM*)^[1] **when the peripheral is assigned to Linux®OS**, and in particular:

- how to configure the LPTIM **peripheral** to enable PWM, trigger, event counter and quadrature encoder
- how to configure the **board**, e.g. LPTIM input/output pins

The configuration is performed using the **device tree mechanism**^[2].

It is used by the *LPTIM Linux driver* that registers relevant information in *PWM* and *IIO* frameworks.

2 DT bindings documentation

The *LPTIM*^[1] is a multifunction device (MFD).

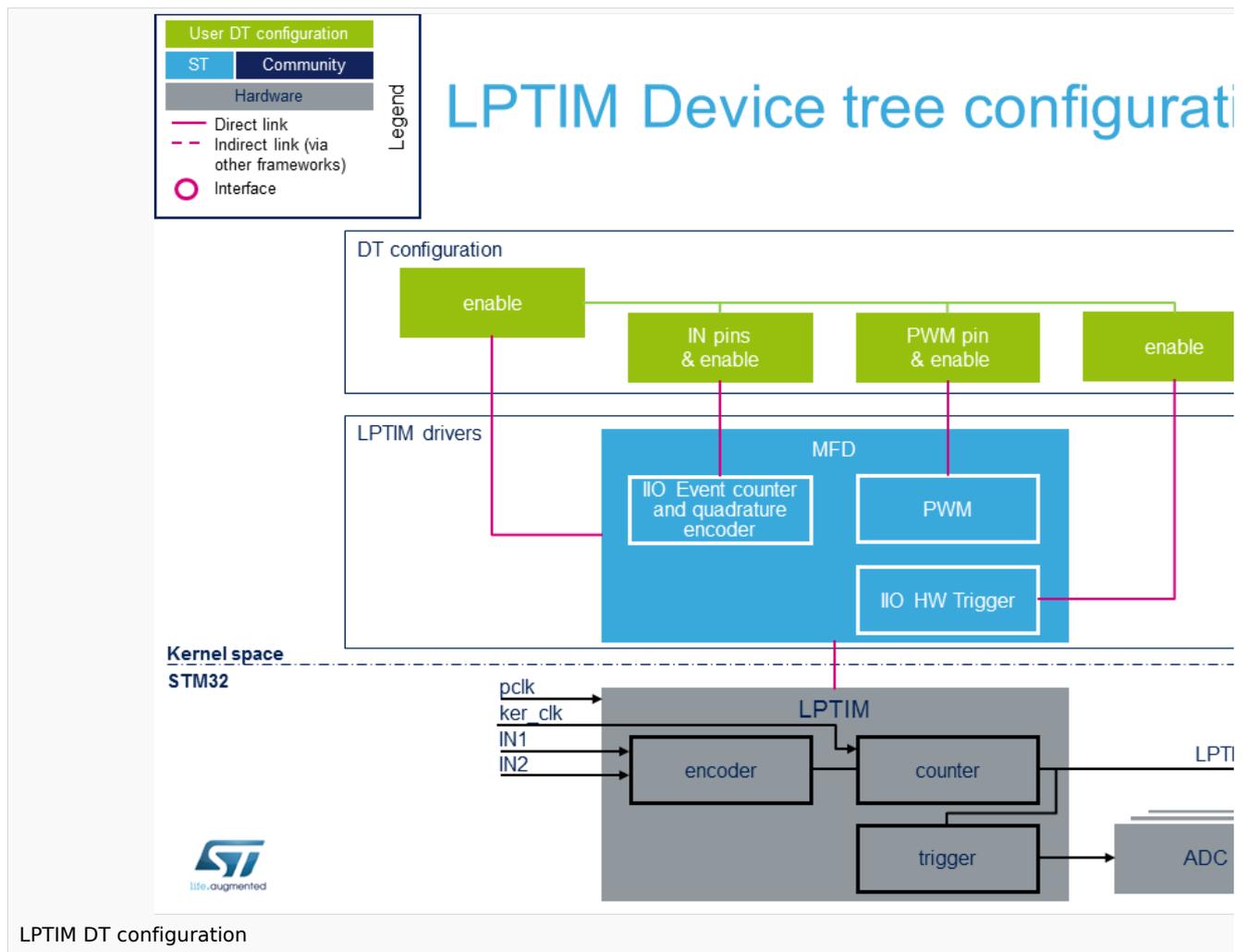
Each function is represented by a separate binding document:

- *STM32 LPTIM MFD device tree bindings*^[3] deals with core resources (e.g. registers, clock)
- *STM32 LPTIM PWM device tree bindings*^[4] deals with **PWM** interface resources (e.g. PWM pins)
- *STM32 LPTIM trigger device tree bindings*^[5] deals with **LPTIM triggers** resources (e.g. trigger output connected to other STM32 internal peripherals)
- *STM32 LPTIM counter device tree bindings*^[6] deals with **LPTIM counter and quadrature encoder** resources (e.g. counter/encoder IN[1..2] pins)

3 DT configuration

This hardware description is a combination of STM32 microprocessor and board device tree files. See [Device tree](#) for more explanations on device tree file split.

The **STM32CubeMX** can be used to generate the board device tree. Refer to [How to configure the DT using STM32CubeMX](#) for more details.



3.1 DT configuration (STM32 level)

LPTIM nodes are declared in `stm32mp157c.dtsi`^[7].

DT root node (aka `lptimer1`, ..., `lptimer5`) and **DT child nodes** describe the LPTIM features such as:

- PWM
- trigger
- event counter and quadrature encoder.

They also describe hardware parameters such as register addresses and clock.

```

lptimer1: timer@40009000 {
    compatible = "st,stm32-lptimer";
resources */
    ...
    pwm {
        compatible = "st,stm32-pwm-lp";
    };
    trigger@0 {
        compatible = "st,stm32-lptimer-trigger";
    };
    /*
    reg = <0>;
    g. 0 for LPTIM1 trigger, 1 for LPTIM2... */
    counter {
        compatible = "st,stm32-lptimer-counter";
    };
    /* quadrature encoder &
    event counter part of LPTIM */
};

```



This device tree part is related to STM32 microprocessors. It should be kept as is, without being modified by the end-user.

3.2 DT configuration (board level)

Follow the below sequence to configure and enable the LPTIM on your board STM32MP15 microprocessor:

- Enable **DT root node** for the LPTIM instance in use (e.g lptimer1, ..., lptimer5), with **status = "okay"**;
- Enable **DT child node(s)** for the feature(s) in use (PWM output, trigger, event counter and quadrature encoder), with **status = "okay"**;
- Configure the pins in use via **pinctrl**, with **pinctrl-0**, **pinctrl-1** and **pinctrl-names**.

3.3 DT configuration examples

3.3.1 LPTIM1 configured as PWM and trigger source

The example below shows how to configure LPTIM1 to act as:

- PWM output on PG13 (See pinctrl device tree configuration and GPIO internal peripheral)
- trigger source (in Synchronous mode with PWM) for other internal peripherals such as ADC^[8], DAC^[9], and DFSDM^[10]

```

lppwm1_pins_a: lppwm1-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, AF1)>; /* configure 'PG13' as
alternate 1 for LPTIM1_OUT mode of operation */
        bias-pull-down;
        drive-push-pull;
        slew-rate = <0>;
    };
};

```

```
lppwm1_sleep_pins_a: lppwm1-sleep-0 {
    pins {
        pinmux = <STM32_PINMUX('G', 13, ANALOG)>; /* configure 'PG13' as
analog for low power mode */
    };
};
```

```
&lptimer1 {
    status = "okay";
    pwm {
LPTIM1_OUT */
        pinctrl-0 = <&lppwm1_pins_a>; /* configure PWM on
        pinctrl-1 = <&lppwm1_sleep_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable PWM on LPTIM1 */
    };
    trigger@0 {
source */
        status = "okay"; /* enable LPTIM1_OUT trigger
    };
};
```

3.3.2 LPTIM2 configured as counter and quadrature encoder

The example below shows how to configure LPTIM2 to act as counter and/or quadrature encoder, with LPTIM2_IN1 and LPTIM2_IN2 pins configured as inputs on PD12 and PD11, respectively (See pinctrl device tree configuration and GPIO internal peripheral)

```
# part of pin-controller dt node
lptim2_in_pins_a: lptim2-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, AF3)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, AF3)>; /* LPTIM2_IN2 */
        bias-disabled;
    };
};
lptim2_sleep_in_pins_a: lptim2-sleep-in-pins-0 {
    pins {
        pinmux = <STM32_PINMUX('D', 12, ANALOG)>, /* LPTIM2_IN1 */
                <STM32_PINMUX('D', 11, ANALOG)>; /* LPTIM2_IN2 */
    };
};
```

```
&lptimer2 {
    status = "okay";
    counter {
/encoder pins */
        pinctrl-0 = <&lptim2_in_pins_a>; /* configure LPTIM2 counter
        pinctrl-1 = <&lptim2_sleep_in_pins_a>;
        pinctrl-names = "default", "sleep";
        status = "okay"; /* enable counter/encoder on
LPTIM2 */
    };
};
```



4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph. If so, the tool inserts **user sections** in the generated device tree. These sections can then be edited to add some properties and they are preserved from one generation to another. Refer to STM32CubeMX user manual for further information.

5 References

For additional information, refer to the following links:

- [1.01.1 LPTIM internal peripheral](#)
- [Device tree](#)
- [Documentation/devicetree/bindings/mfd/stm32-lptimer.txt](#) , STM32 LPTIM MFD device tree bindings
- [Documentation/devicetree/bindings/pwm/pwm-stm32-lp.txt](#) , STM32 LPTIM PWM device tree bindings
- [Documentation/devicetree/bindings/iio/timer/stm32-lptimer-trigger.txt](#) , STM32 LPTIM trigger device tree bindings
- [Documentation/devicetree/bindings/iio/counter/stm32-lptimer-cnt.txt](#) , STM32 LPTIM counter/encoder device tree bindings
- [STM32MP157C device tree file](#)
- [ADC internal peripheral](#)
- [DAC internal peripheral](#)
- [DFSDM internal peripheral](#)

low-power timer (STM32 specific)

Operating System

Pulse Width Modulation

Device Tree

Multifunction device

also known as

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Digital Filter for Sigma-Delta Modulator