AN12713

Generating PWM and PFM by Using FlexIO

Rev. 0 — 14 February, 2020

Application Note

Contents

1 Introduction

This application note illustrates how to use FlexIO module to generate the pulse-width modulation (PWM) and the pulse-frequency modulation(PFM) waveform.

FlexIO is an on-chip peripheral available on NXP I.MXRT series. It is a highly configurable module capable of emulating a wide range of communication protocols, such as UART, I2C,SPI, and I2S. Users can also use FlexIO to generate PWM and PFM waveform.

The standalone peripheral module FlexIO is not used for replacement of the PWM and PFM generator, but as an additional peripheral module of the MCU. This module enable users to build their own peripheral directly in the microcontroller.

This application creates a simple software demo based on the SDK and i.MX RT series platform for users to use FlexIO module generating PWM and PFM waveform with related configurations.

2 Development platform

This document describes the example application based on the i.MX RT1010 EVK board shown in Figure 1. Users can easily enable this application on other i.MX RT Series EVK board.

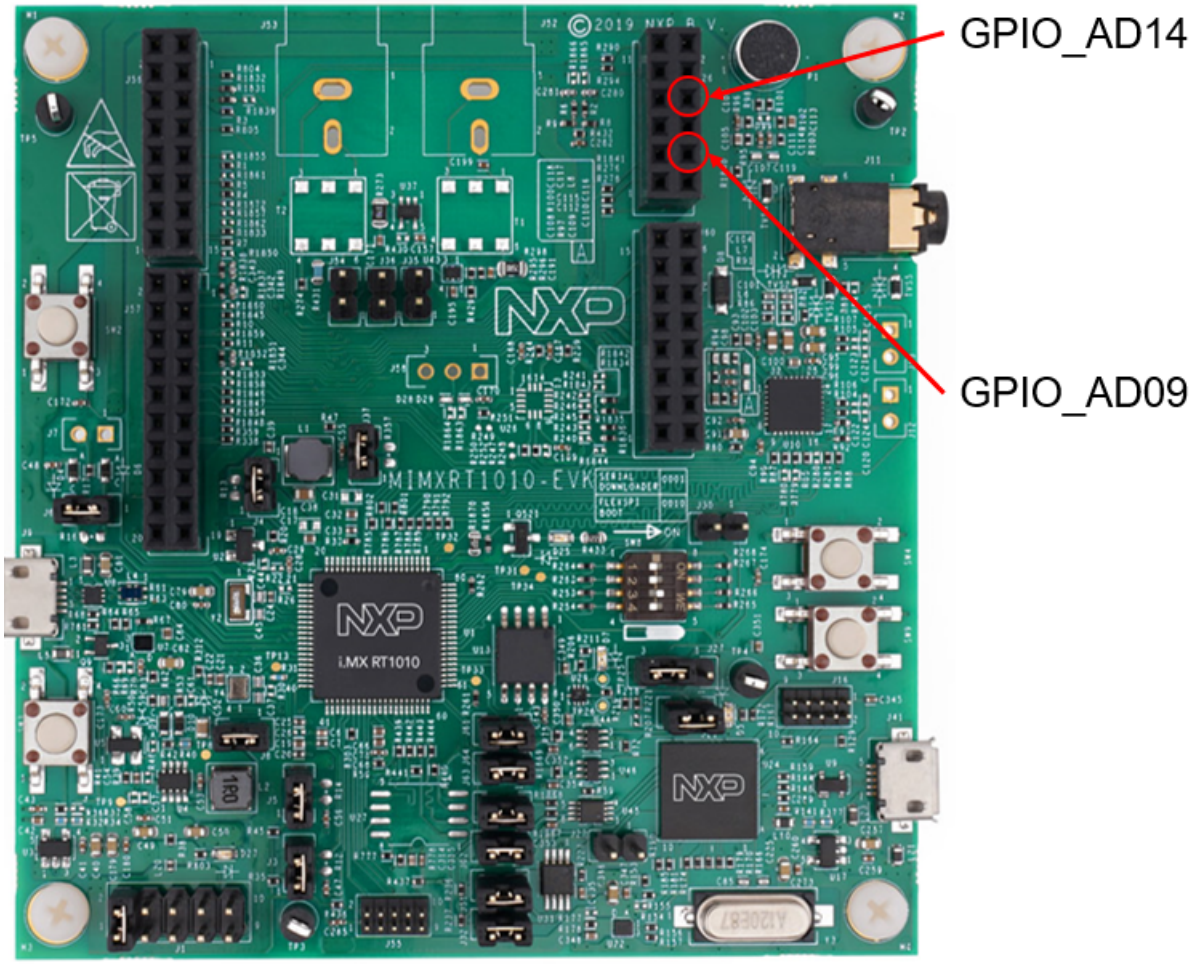

Figure 1. i.MX RT1010 EVK Board

In this application, FlexIO generates a PWM waveform on the FlexIO1_IO26 (GPIO_AD14) pin and generates a PFM waveform on the FlexIO1_IO21(GPIO_AD09) pin. Connect these two pins to a logic analyzer separately and observe the waveform on analysis software.

3 FlexIO module overview

The FlexIO module of the i.MX RT1010 provides the following key features:

- Array of 32-bit shift registers with transmit, receive, and data match modes
- Double buffered shifter operation for continuous data transfer
- Shifter concatenation to support large transfer sizes
- Automatic start/stop bit generation
- 1, 2, 4, 8, 16 or 32 multi-bit shift widths for parallel interface support
- Interrupt, DMA, or polled transmit/receive operation
- Programmable baud rates independent of bus clock frequency, with support for asynchronous operation during stop modes
- Highly flexible 16-bit timers with support for various internal or external trigger, reset, enable, and disable conditions

- Programmable logic mode for integrating external digital logic functions on-chip or combining pin/shifter/timer functions to generate complex outputs
- Programmable state machine for offloading basic system control functions from CPU with support for up to 8 states, 8 outputs, and 3 selectable inputs per state

Figure 2 gives a high-level overview of the configuration of FlexIO timers and shifters.

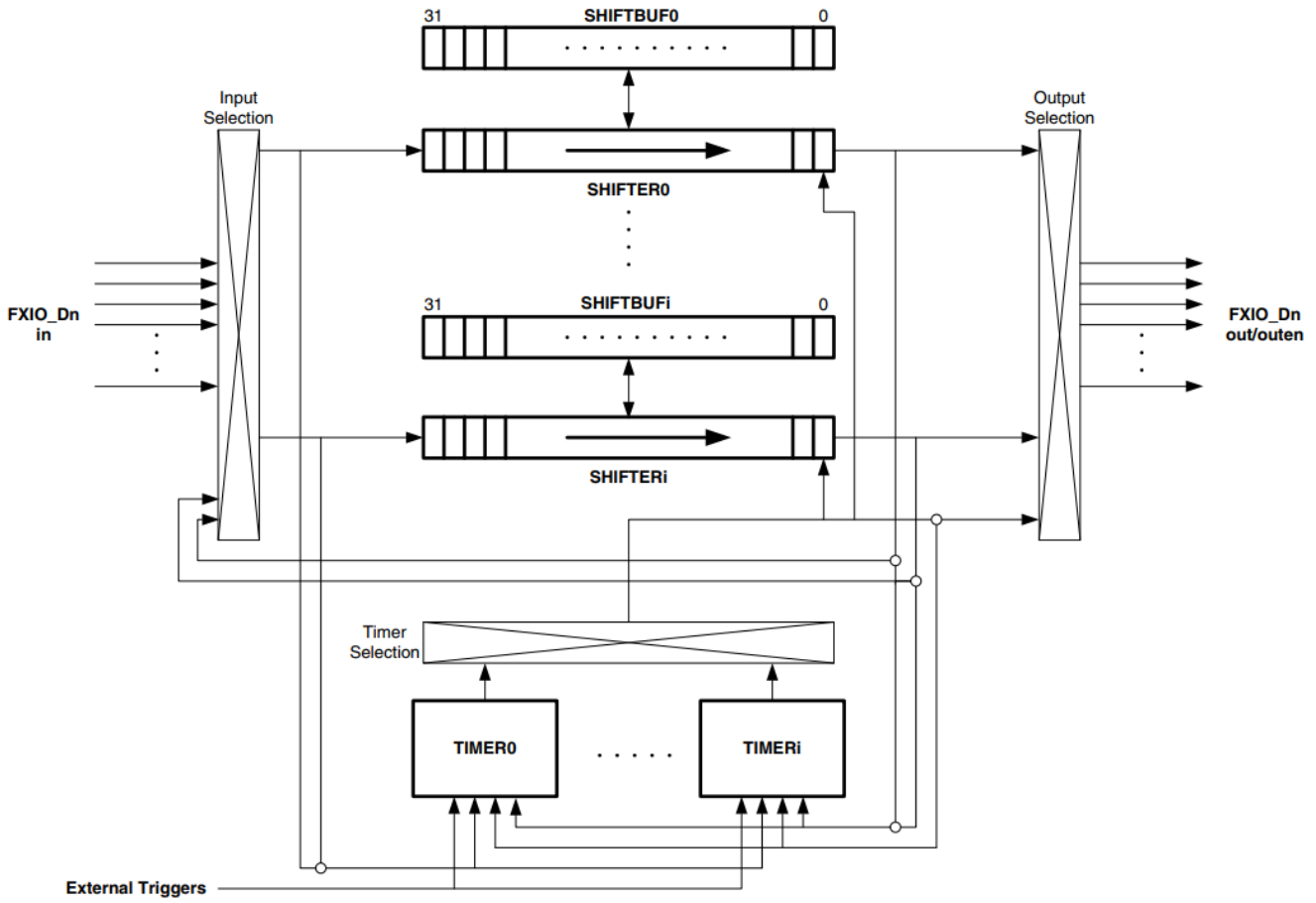

Figure 2. FlexIO Block Diagram

From the FLEXIO_PARAM register, users can read the amount of these resources like shifter, timer, pin, and trigger. For instance, there are 8 shifters, 8 timers, 32 pins, and two external triggers in i.MX RT1010 (In this device, limited by the number of pins, FlexIO only has 27 pins).

4 Generating PWM and PFM using FlexIO

To generate PWM and PFM waveform, following resources are needed:

- Two timers --– one for PWM configured as 8-bit PWM mode, one for PFM configured as 16-bit counter mode
- Two pins – configured as output and export PWM and PFM waveform

Figure 3 shows a diagram of a FlexIO timer. In this document, generating PWM and PFM waveform need to use two timers and enable their output pins to export waveform.

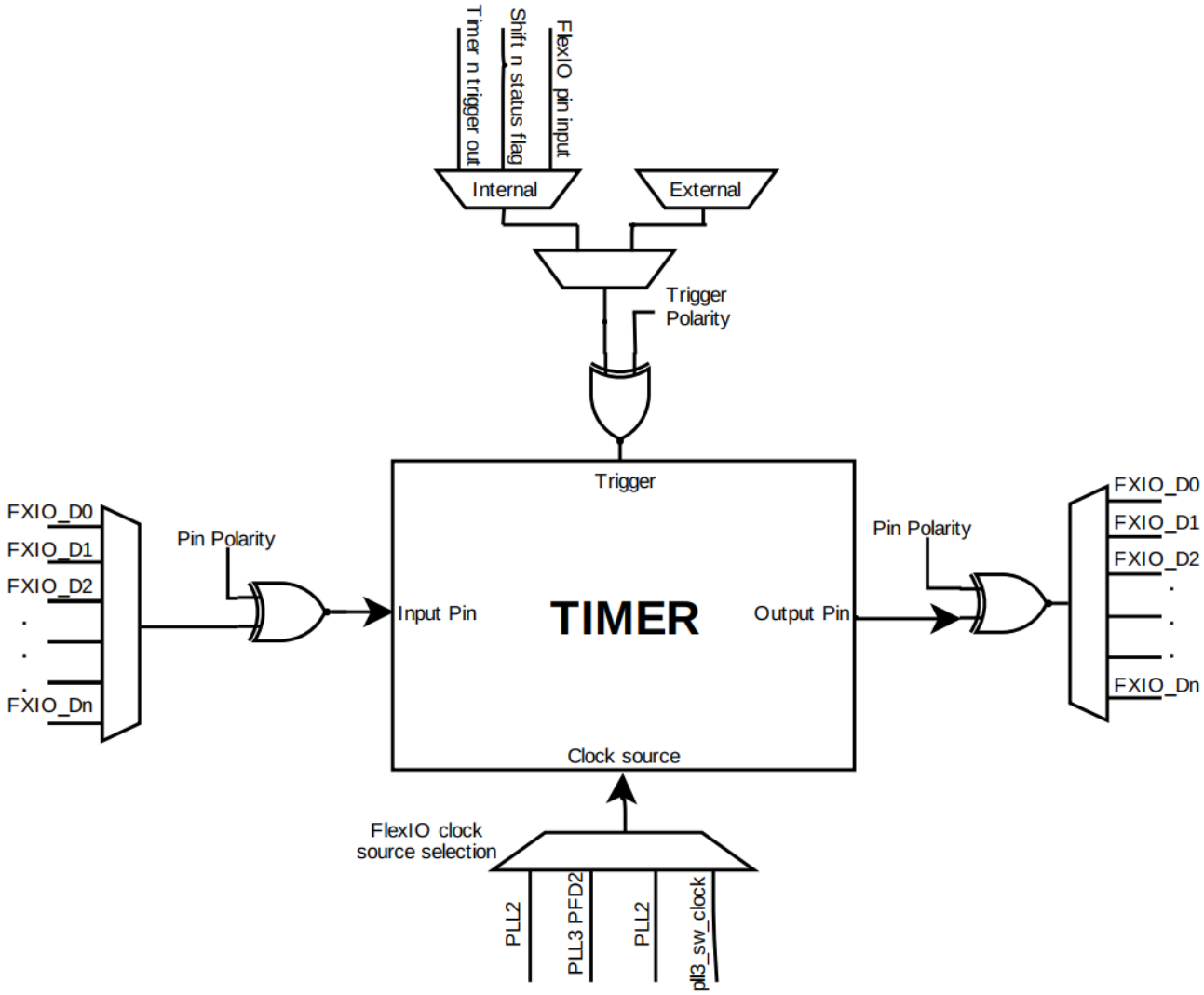

Figure 3. FlexIO Timer Diagram

This application use Timer 0 generating PWM output to pin 26 and Timer 1 generating PFM output to pin 21. There are some differences on these two waveform's key configuration:

- Timer mode select (TIMCTLn[TIMOD])
  - For PWM, select 8-bit PWM high mode. In this mode, the lower 8-bits of the counter and compare register are used to configure the high period of the timer shift clock, and the upper 8-bits are used to configure the low period of the timer shift clock. Which means the 16-bit counter split into two parts, the lower 8-bit control the PWM high pulse width and the upper 8-bits control the low pulse width.
  - For PFM, select 16-bit counter mode. In this mode, the full 16-bits of the counter and compare register are used to configure the baud rate. Which means the frequency of output waveform can be configured by the source clock and the compare value.
- Timer Decrement select (TIMCFGn[TIMDEC])
  - For both PWM and PFM, select FlexIO clock (FLEXIO_CLK[1]) as the timer decrement source. In this mode, the counter decreases on every FlexIO clock. FlexIO clock is configurable, configure CCM_CSCMR2[FLEXIO1_CLK_SEL], CCM_CS1CDR[FLEXIO1_CLK_PODF] and

CCM_CS1CDR[FLEXIO1_CLK_PRED] can change the source clock frequency, for details, See CCM chapter of reference manual.

- Timer compare value set (TIMCMPn[CMP])
  - For PWM, the lower 8-bits configure the high period of the output to (CMP[7:0] + 1) and the upper 8-bits configure the low period of the output to (CMP[15:8] + 1). The dual 8-bits value is loaded into timer counter when the counter decreased to 0. Configure these two values can control the frequency and duty of PWM waveform. Timer compare value should be calculated from this formula: TIMCMPn[CMP] = (((FLEXIO_CLK / freq[2]) * (100-duty[3]) / 100-1) << 8) | ((FLEXIO_CLK / freq) * duty / 100 - 1)
  - For PFM, the whole 16-bits of CMP are used as counter reload value. And the PFM frequency can calculate by this formula: $freq_F$[4] = FLEXIO_CLK /((CMP + 1) * 2)

[1] FLEXIO_CLK is the clock from CCM.

[2] The frequency of the PWM waveform.

[3] The duty of the PWM waveform by percentage.

[4] The frequency of the PFM frequency.

Figure 4 shows how the PWM and PFM waveform generate with above configuration.

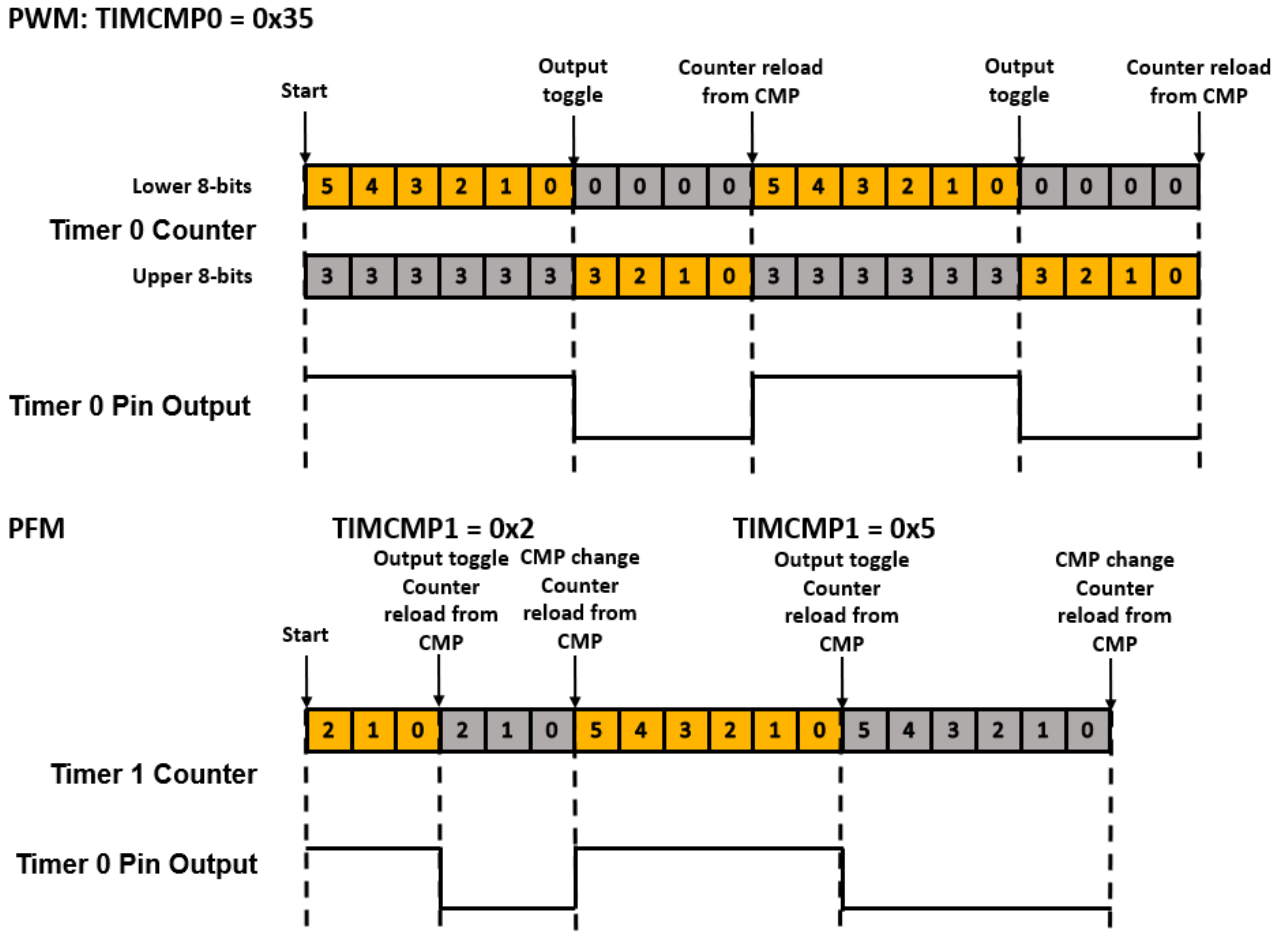

Figure 4. Waveform Generating Diagram

5 Run the example

Users can download the software from *nxp.com*. Find the IAR project *flexio_pwm_pfm*, build, download, and run the demo. Using the logic analyzer software in PC to capture the waveform from the GPIO_AD09 and GPIO_AD14 pins.

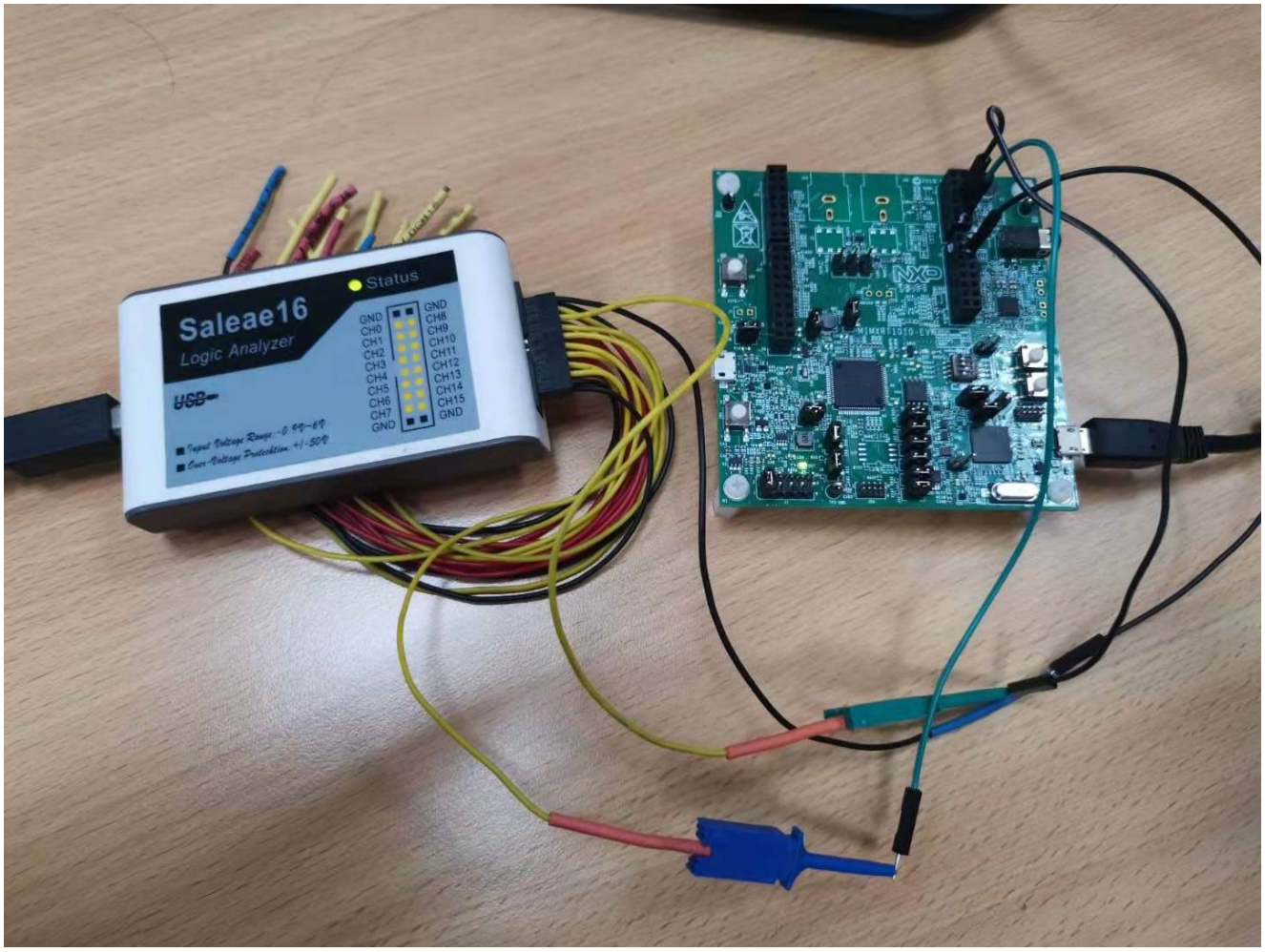

Figure 5. Run the Demo

Figure 6 shows the waveform that logic analyzers capture.

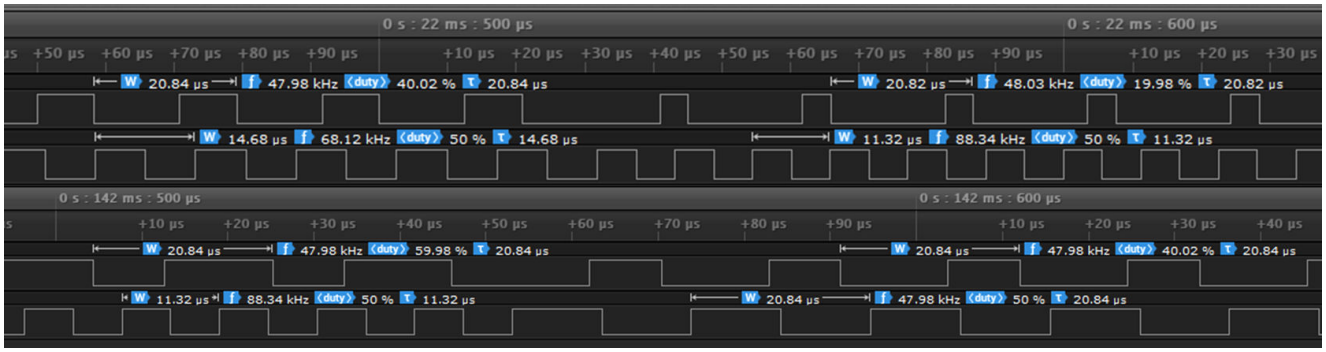

Figure 6. PWM and PFM waveform

Users can change the duty and frequency of PWM by changing the parameters of *flexion_pwm_init()*, and configure the frequency of PFM by using *flexion_pfm_configfre()*. All these APIs in this demo can be directly used in users' own code with few changes.

Compared with the PWM module, using FlexIO to generate PWM still exist some limitations. For instance, the frequency range of the PWM is limited by the FlexIO clock frequency; it can only support edge-aligned PWM and do not support center and asymmetrical PWM; it also does not support advanced features like deadtime and fault protection.

6 References

1. i.MX RT1010 Processor Reference Manual (document I.MXRT1010RM)
2. MCUXpresso SDK (Software Development Kit for NXP MCUs)

7 Revision history

Revision number	Date	Substantive changes
0	02/2020	Initial release

8 Appendix. Waveform captured by Oscilloscope

Figure 7 is the waveforms captured by an oscilloscope.

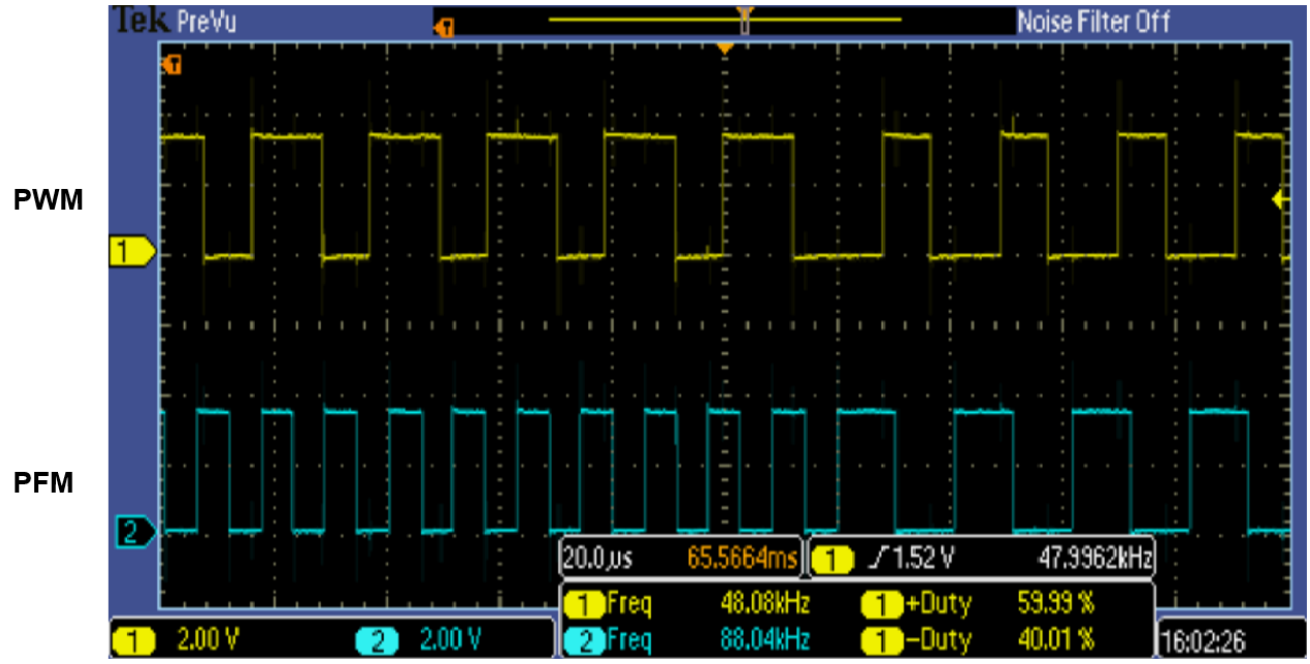

Figure 7. PWM and PFM waveform

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer’s applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 14 February, 2020

Document identifier: AN12713

arm