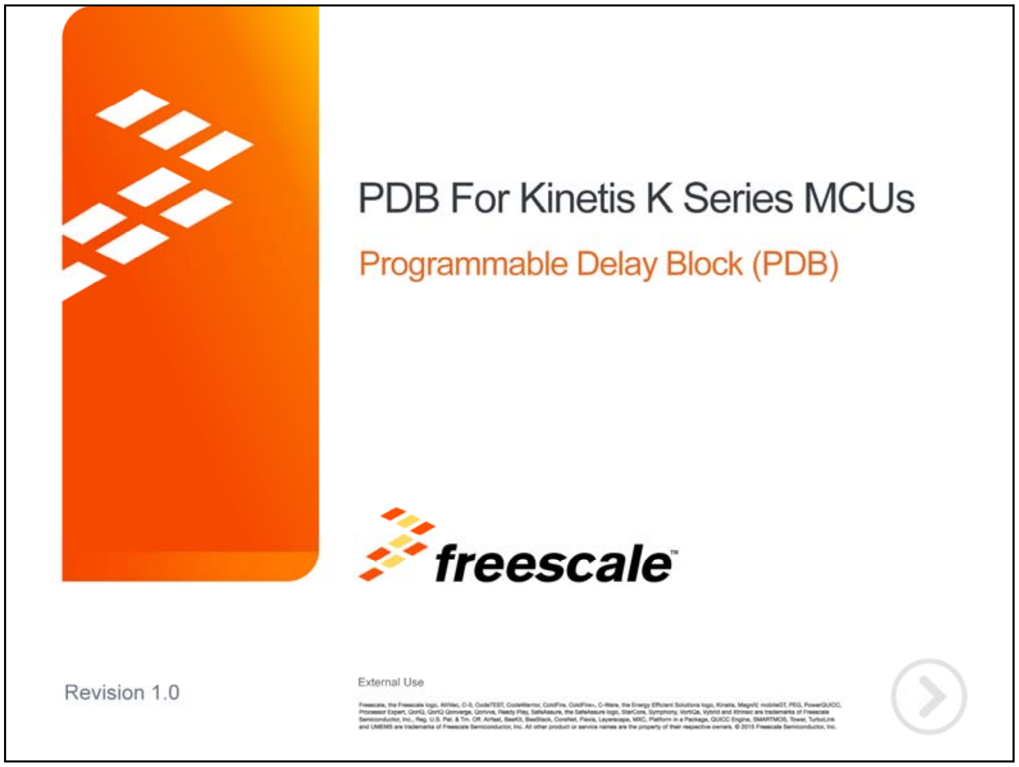


PDB For Kinetis K Series MCUs

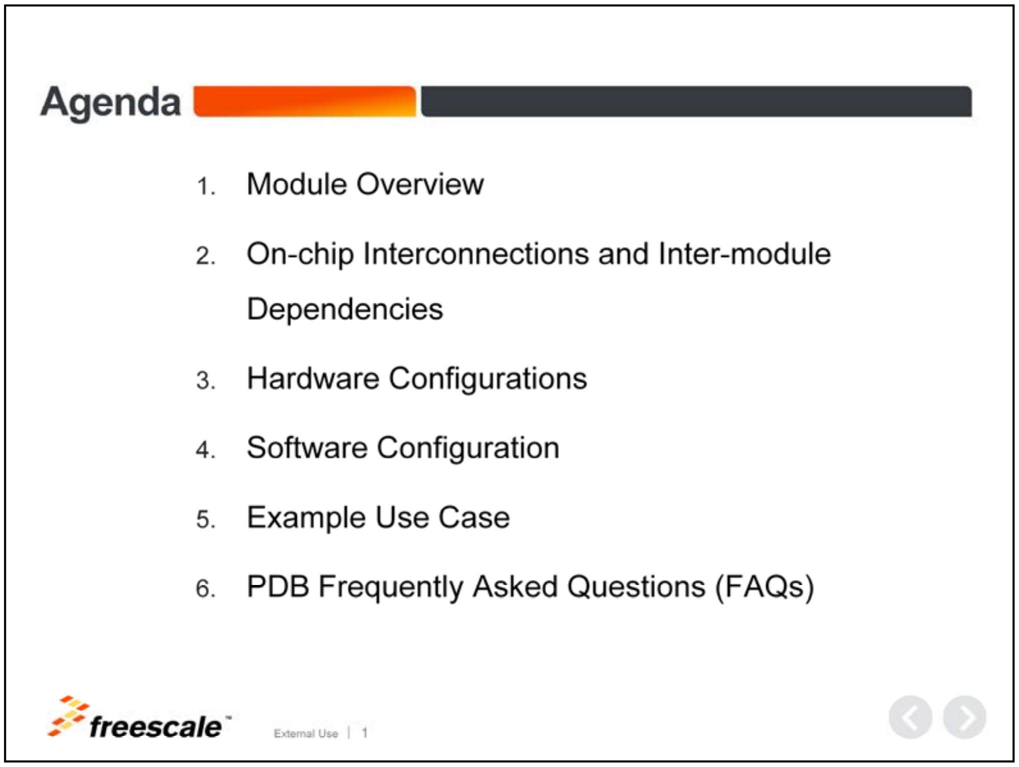
Programmable Delay Block (PDB)

Revision 1.0

External Use

Freescale, the Freescale logo, AltiVec, C-5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, Qorivva, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid and Xtrinsic are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. Airfast, BeeKit, BeeStack, CoreNet, Flexis, Layerscape, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink and UMEMS are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. © 2015 Freescale Semiconductor, Inc.

Hello and welcome to this presentation of the Programmable Delay Block, or PDB, module for Kinetis K series MCUs. In this session you'll learn about the PDB module, its main features and the application benefits of leveraging this function.



Agenda

1. Module Overview
2. On-chip Interconnections and Inter-module Dependencies
3. Hardware Configurations
4. Software Configuration
5. Example Use Case
6. PDB Frequently Asked Questions (FAQs)

The agenda for this training includes:

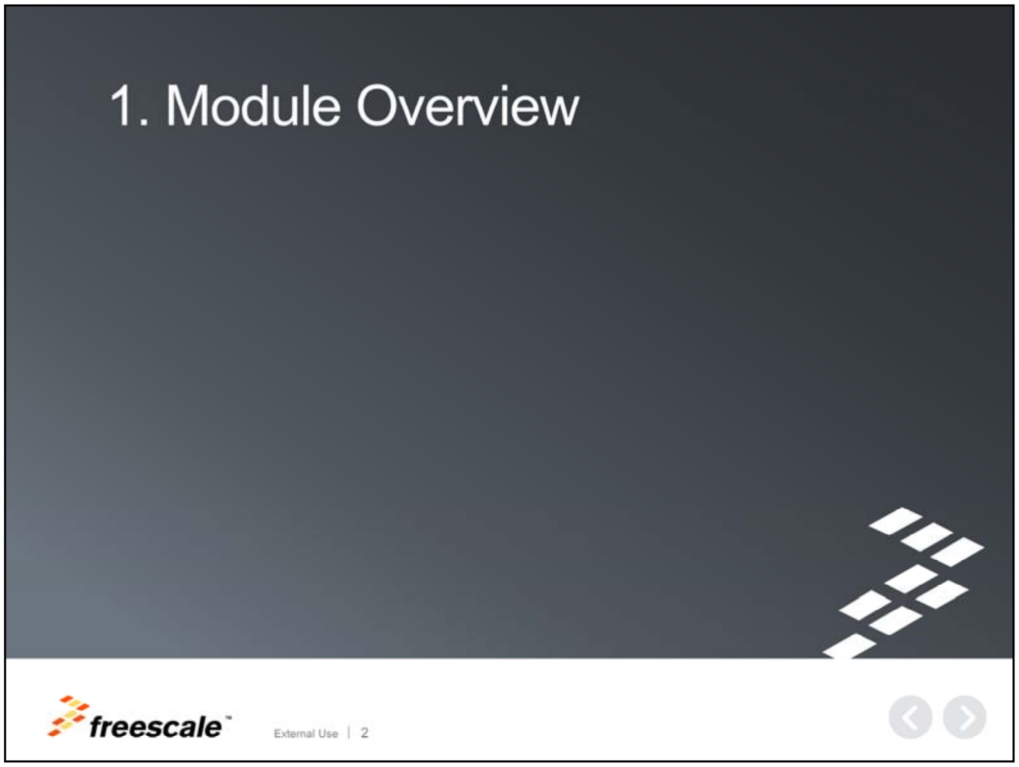
• A module overview

• The on-chip interconnections and inter-module dependencies.

• Hardware configuration and software configuration

• An example use case

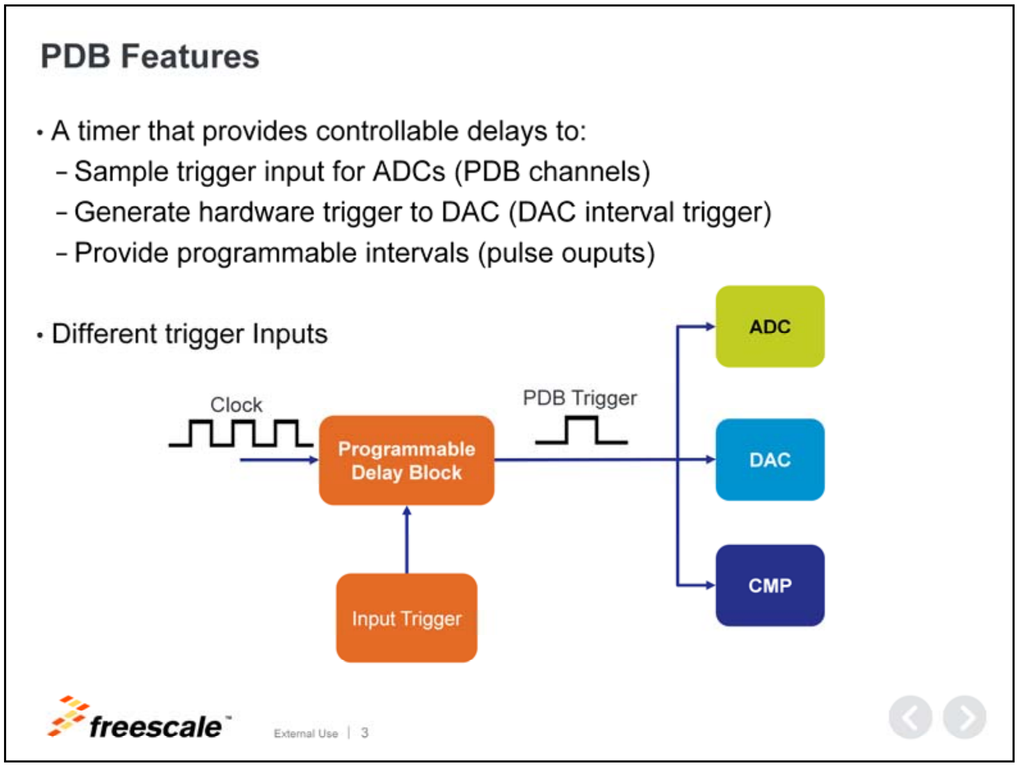
• And a few frequently asked questions



1. Module Overview

External Use | 2

We first begin with a module overview.



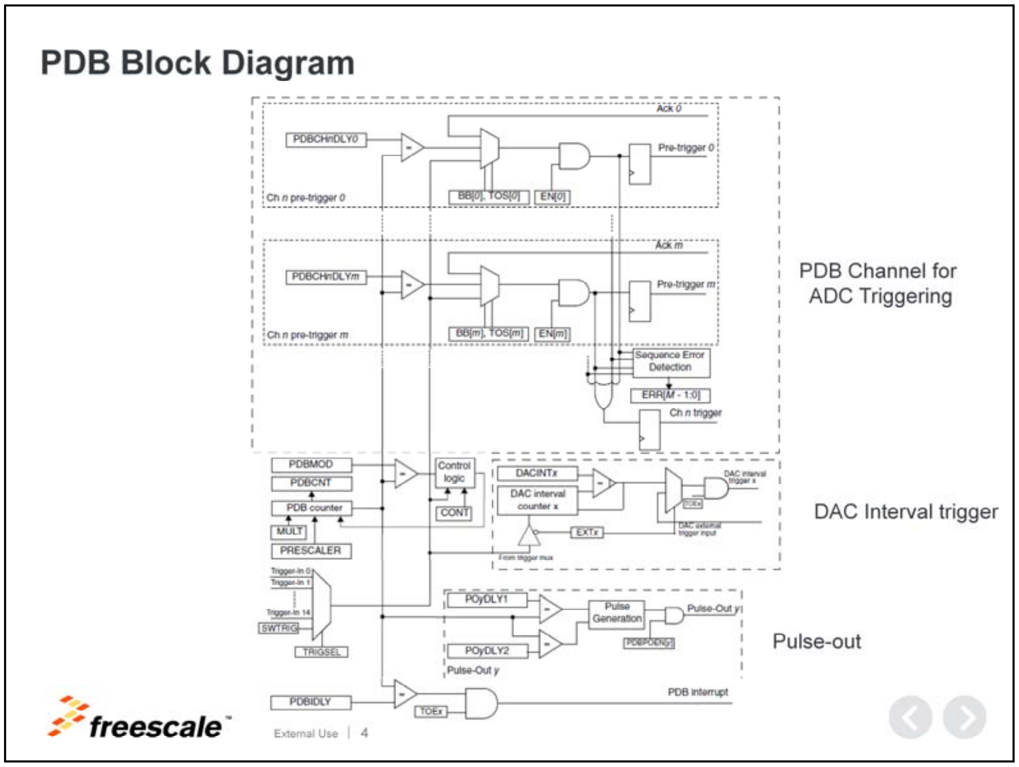
PDB Features

- A timer that provides controllable delays to:
  - Sample trigger input for ADCs (PDB channels)
  - Generate hardware trigger to DAC (DAC interval trigger)
  - Provide programmable intervals (pulse ouputs)
- Different trigger Inputs

PDB Features

The Programmable Delay Block, or PDB, is a timer that provides controllable delays to the hardware trigger inputs of ADCs and/or generates the interval triggers to DACs; so that the precise timing between ADC conversion and/or DAC updates can be achieved. It can also provide pulse outputs that are used as the sample window to the comparator, or CMP, module.

The PDB uses either an internal or external trigger input in order to start counting.

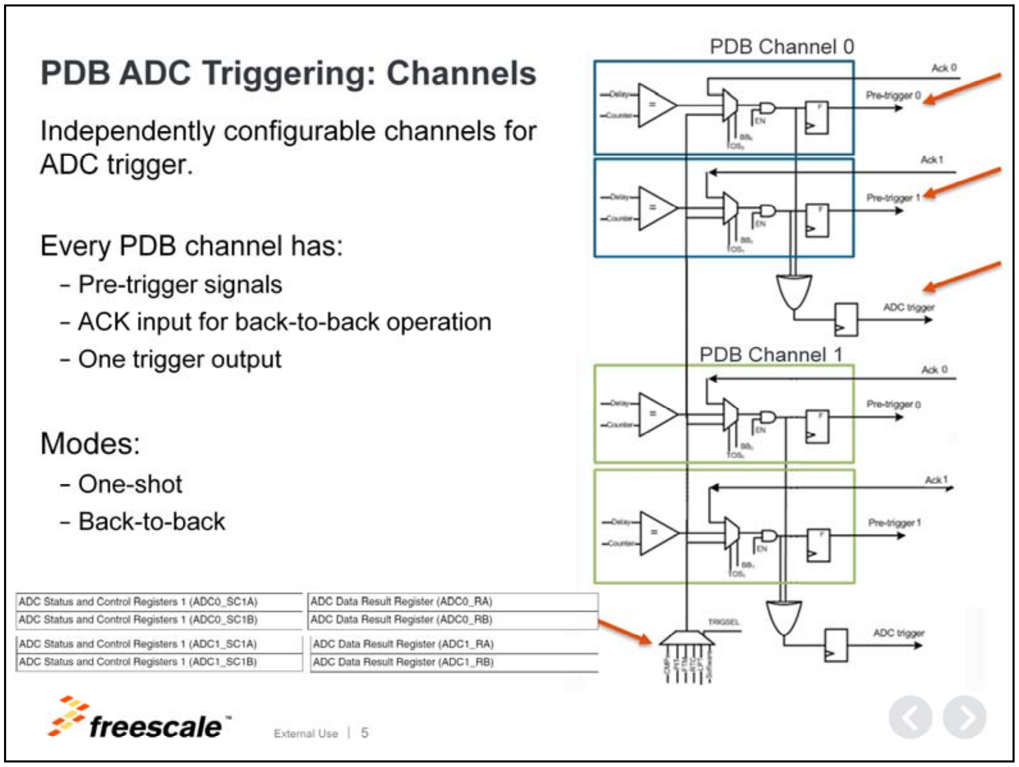


PDB Block Diagram

Here is a high-level block diagram of the PDB available on the majority of Kinetis MCUs.

The PDB module is composed of common registers that allow you to select the PDB clock signal value, input trigger, and interrupt settings register.

- PDB channels used for ADC triggering can be controlled independently. To better understand which PDB channel interconnects with the ADC hardware trigger, visit your reference manual.
- The DAC interval trigger block generates triggers for the DAC module either by selecting the input trigger or setting specific timing intervals.
- The pulse-out block generates controllable-width pulses that are used by the comparator module.



PDB ADC Triggering: Channels

Independently configurable channels for ADC trigger.

Every PDB channel has:
- Pre-trigger signals
- ACK input for back-to-back operation
- One trigger output

Modes:
- One-shot
- Back-to-back

ADC Status and Control Registers 1 (ADC0_SC1A)	ADC Data Result Register (ADC0_RA)
ADC Status and Control Registers 1 (ADC0_SC1B)	ADC Data Result Register (ADC0_RB)
ADC Status and Control Registers 1 (ADC1_SC1A)	ADC Data Result Register (ADC1_RA)
ADC Status and Control Registers 1 (ADC1_SC1B)	ADC Data Result Register (ADC1_RB)

External Use | 5

PDB ADC Triggering: Channels

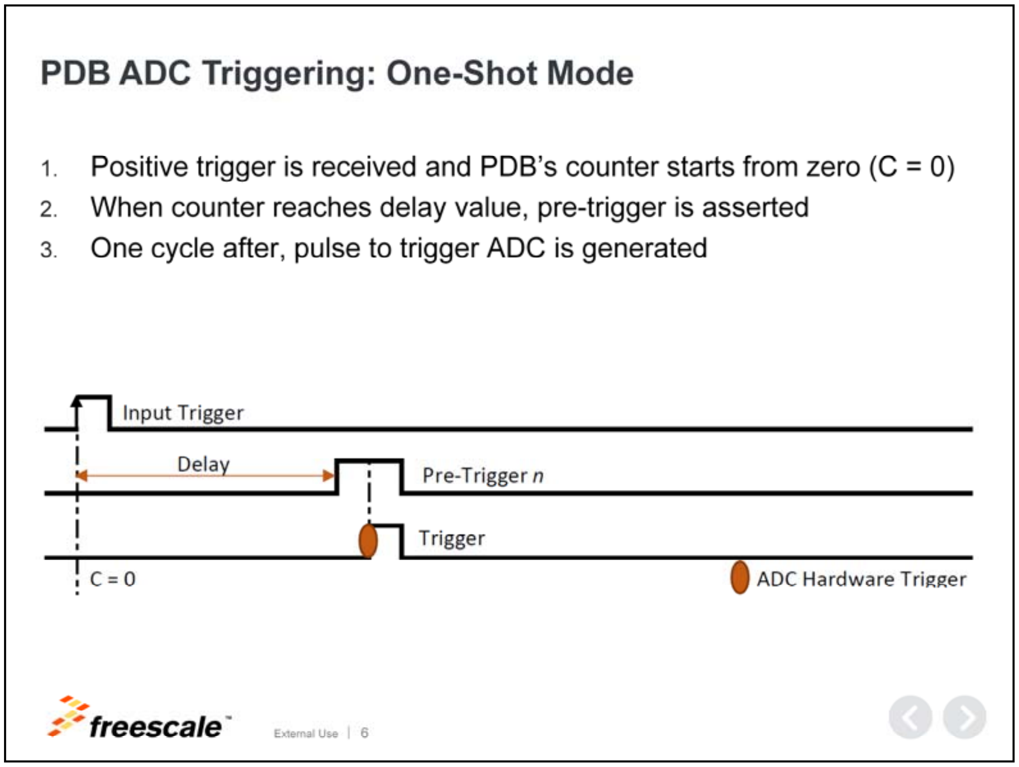
PDB is mainly used for ADC triggering. For this purpose, PDB uses channels that can be configured independently. Each of these channels is associated with one ADC.

A PDB channel is formed with pre-trigger signals; one acknowledgement input signal for each pre-trigger signal and one trigger output that serves as the hardware trigger for ADC. These channels can be configured as a one-shot mode or back-to-back mode.

Each ADC contains two sets of configuration and result registers: A and B. The pre-trigger outputs are used to specify which signal will be sampled next. The pre-triggers are used to precondition the ADC block prior to the actual trigger. When the ADC receives the rising edge of the trigger, the ADC will start conversion at the precondition determined by the pre-trigger.

Using a Kinetis K64 MCU in the example shown here, we see that there are two PDB channels available to trigger ADC0 and ADC1. PDB channel 0 triggers ADC0, and PDB channel 1 triggers ADC1. Each of these channels have two pre-trigger signals: the pre-trigger and one trigger output.

Although each channel can be configured independently, both channels use the same input trigger.



PDB ADC Triggering: One-Shot Mode

1. Positive trigger is received and PDB's counter starts from zero (C = 0)
2. When counter reaches delay value, pre-trigger is asserted
3. One cycle after, pulse to trigger ADC is generated

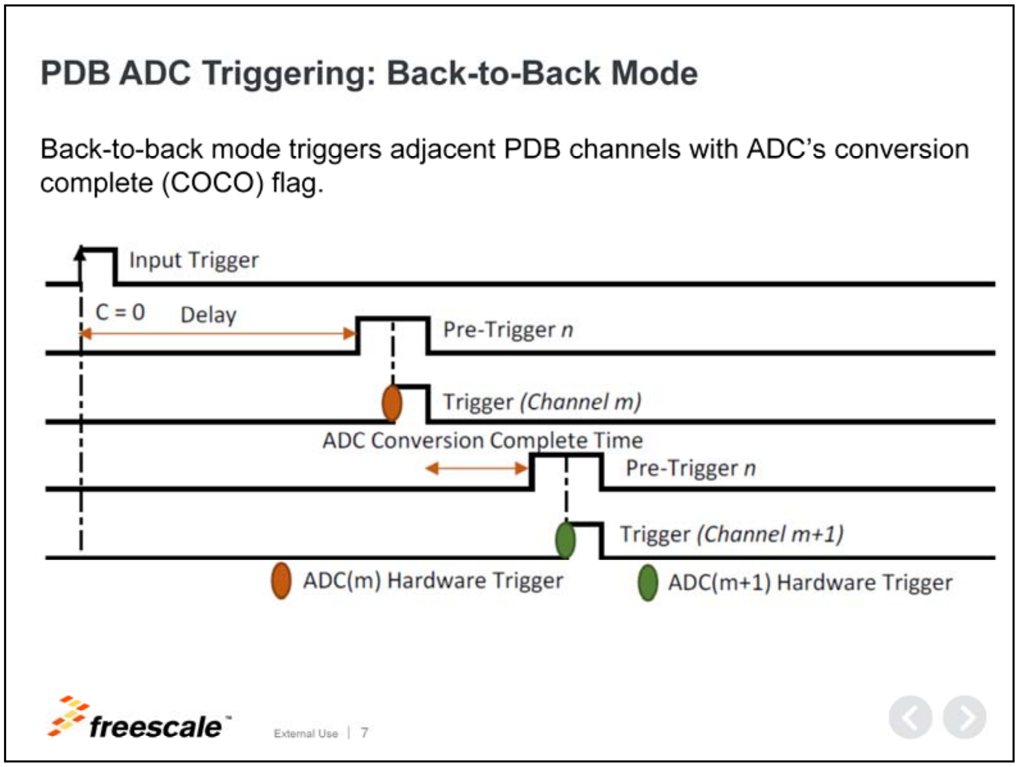
Input Trigger
Delay
Pre-Trigger n
Trigger
C = 0
ADC Hardware Trigger

PDB ADC Triggering: One-Shot Mode

Let's take a look at an example of a time diagram demonstrating the one-shot configuration functionality.

- First, the PDB module receives a positive input trigger that resets the PDB counter and counting begins.
- Once the counter reaches the PDB delay value, the pre-trigger output is asserted.
- Then, following one additional peripheral clock cycle, the trigger pulse is generated. When the ADC receives the trigger rising edge, it will start a new conversion.

If continuous mode is enabled, the counter will re-start counting from zero any time the counter reaches the modulus register.



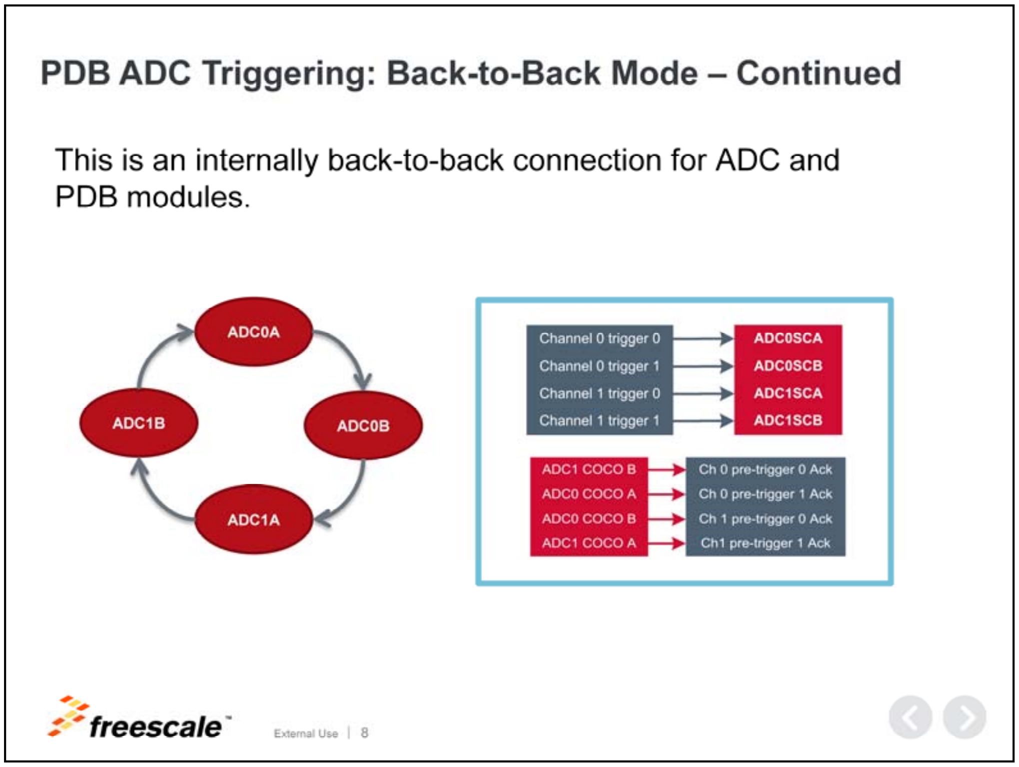
PDB ADC Triggering: Back-to-Back Mode

Back-to-back mode triggers adjacent PDB channels with ADC's conversion complete (COCO) flag.

PDB ADC Triggering: Back-to-Back Mode

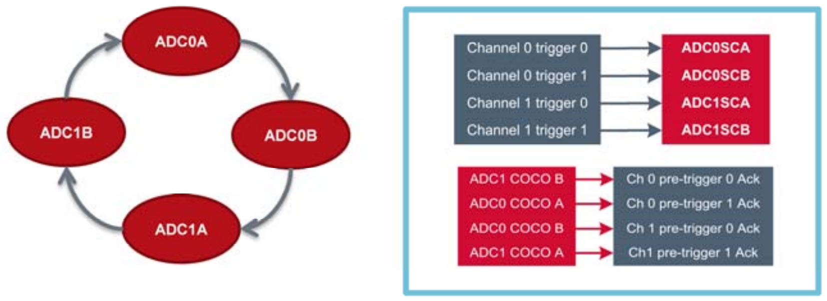
The main difference between back-to-back mode and one-shot mode is that back-to-back mode needs to receive an ADC conversion complete flag to trigger the next PDB channel.

- Similar to one-shot mode, the PDB will receive an input trigger to begin counting.
- When the counter reaches the pre-configured delay value, pre-trigger and trigger outputs for ADC(m) occur.
- When the ADC(m) conversion complete flag occurs, it will acknowledge back to the PDB module, generating pre-trigger and trigger outputs for ADC(m+1).



PDB ADC Triggering: Back-to-Back Mode – Continued

This is an internally back-to-back connection for ADC and PDB modules.

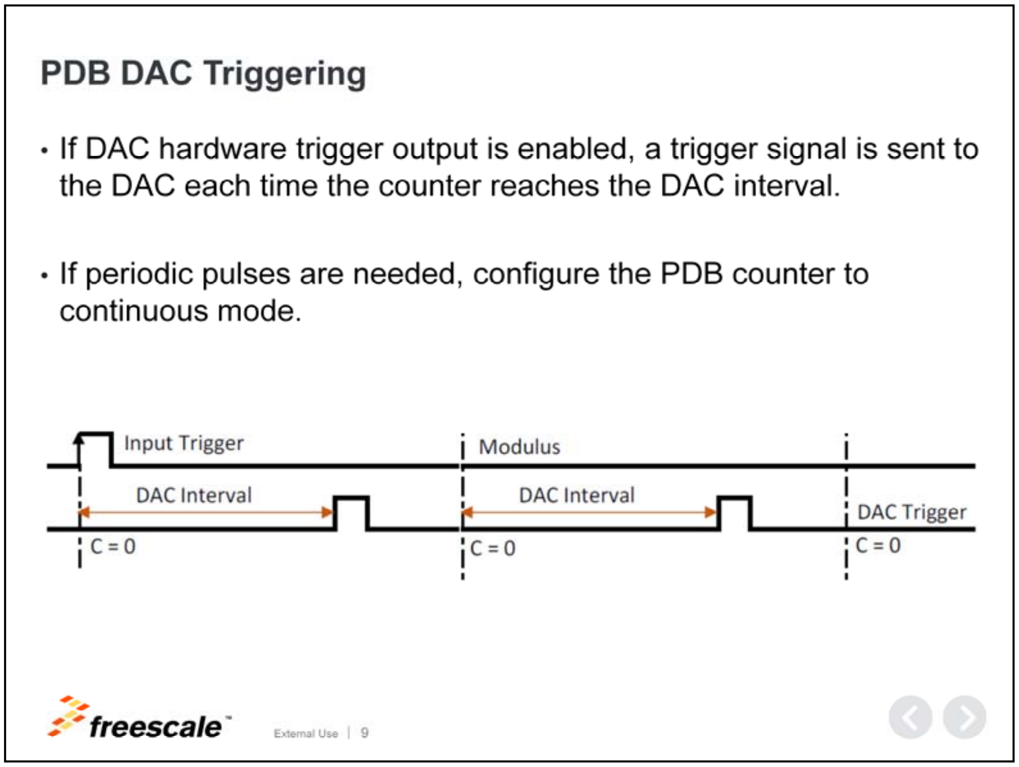


freescale External Use | 8

PDB ADC Triggering: Back-to-Back Mode – Continued

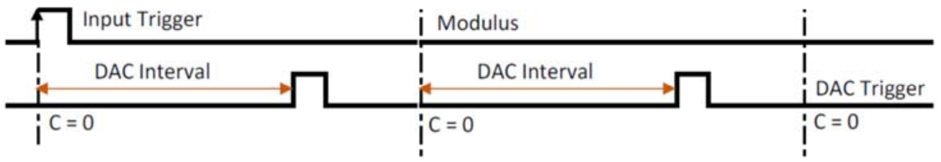
Internally, ADC conversion complete flags are connected to acknowledge the next pre-trigger and trigger output. When an ADC conversion hasn’t finished, a new trigger from PDB cannot be accepted by ADC. If this condition is overridden, a sequence error occurs.

This sequence error typically happens because the delay is set too short and pre-trigger asserts before the previously triggered ADC conversion is completed.



PDB DAC Triggering

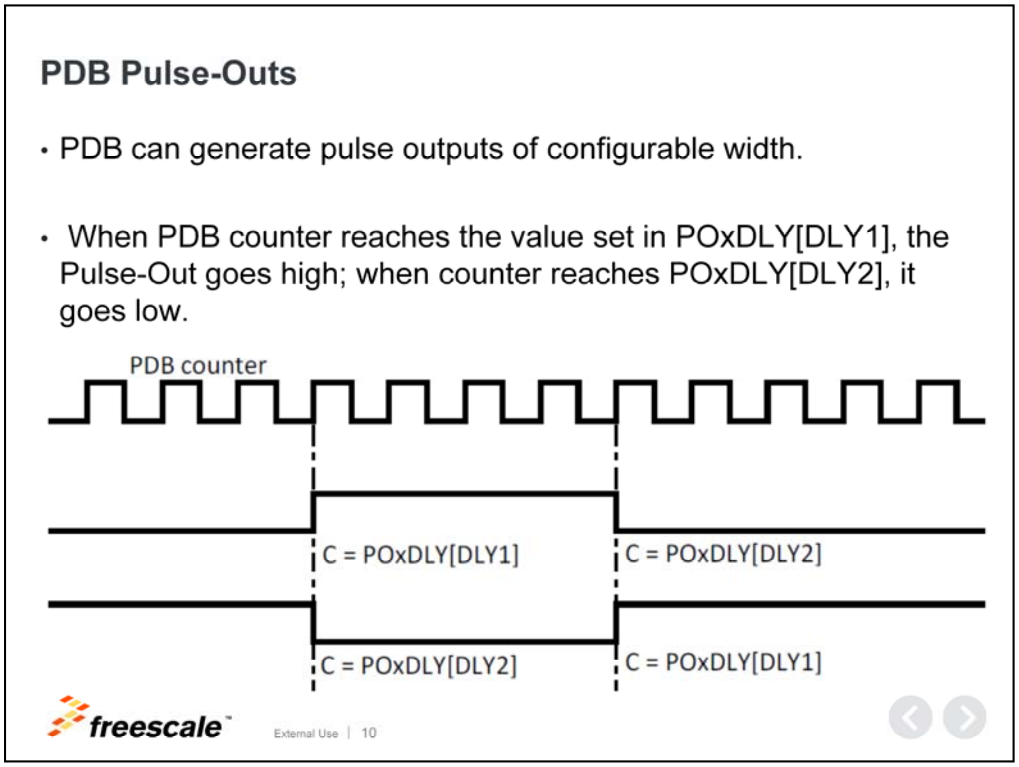
- If DAC hardware trigger output is enabled, a trigger signal is sent to the DAC each time the counter reaches the DAC interval.
- If periodic pulses are needed, configure the PDB counter to continuous mode.



PDB DAC Triggering

PDB can generate interval triggers for DACs to update their outputs periodically.

- If DAC hardware trigger output is enabled, a trigger signal is sent to the DAC each time the counter reaches the DAC interval.
- If periodic pulses are needed, configure the PDB counter to continuous mode.



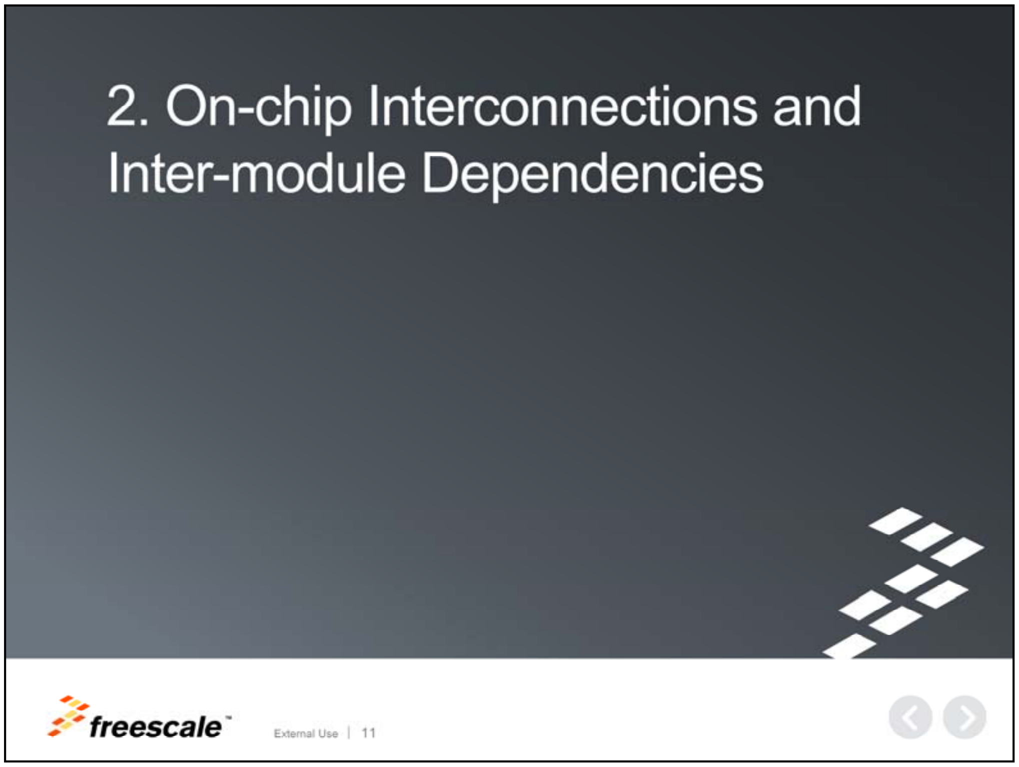
PDB Pulse-Outs

- PDB can generate pulse outputs of configurable width.
- When PDB counter reaches the value set in POxDLY[DLY1], the Pulse-Out goes high; when counter reaches POxDLY[DLY2], it goes low.

PDB counter
C = POxDLY[DLY1]
C = POxDLY[DLY2]
C = POxDLY[DLY2]
C = POxDLY[DLY1]
freescale
External Use | 10

PDB Pulse-Outs

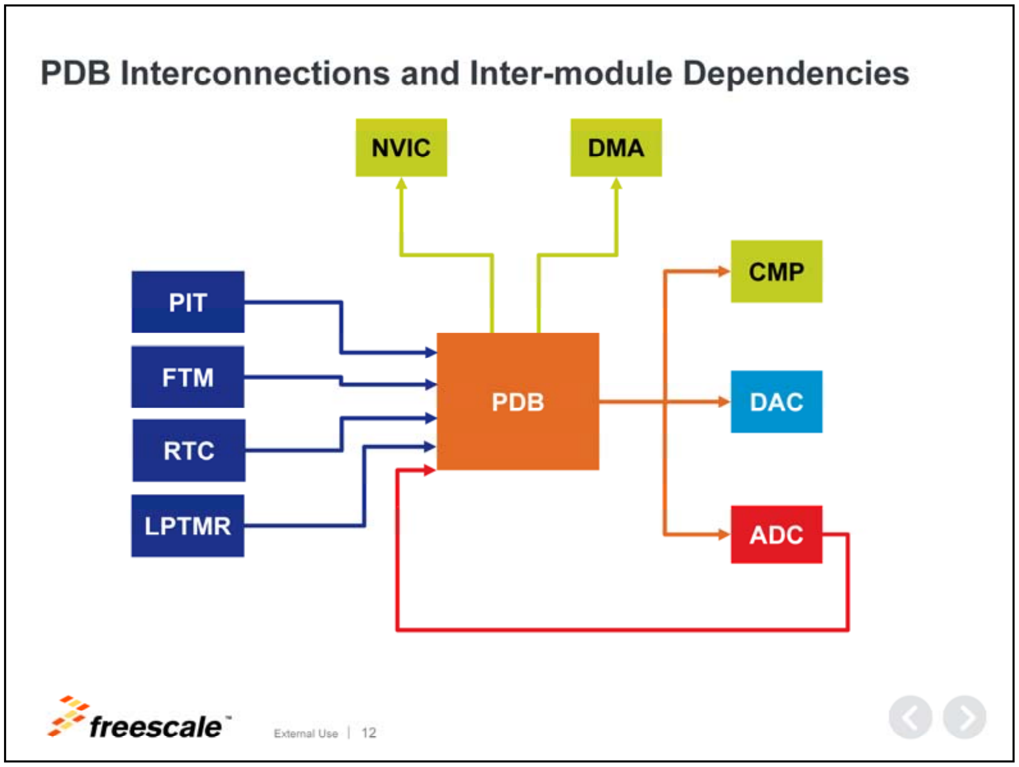
- PDB can generate pulse outputs of configurable width. This signal is connected to the comparator module and can be used for the sample window.
- When the PDB counter reaches the value set in delay-1 (DLY1), the Pulse-Out goes high. When the counter reaches DLY2, it goes low. DLY2 can be set either greater or less than DLY1. This condition allows the user to generate either: Positive-width signals when DLY2 is greater than DLY1 or negative-width signals when DLY2 is less than DLY1



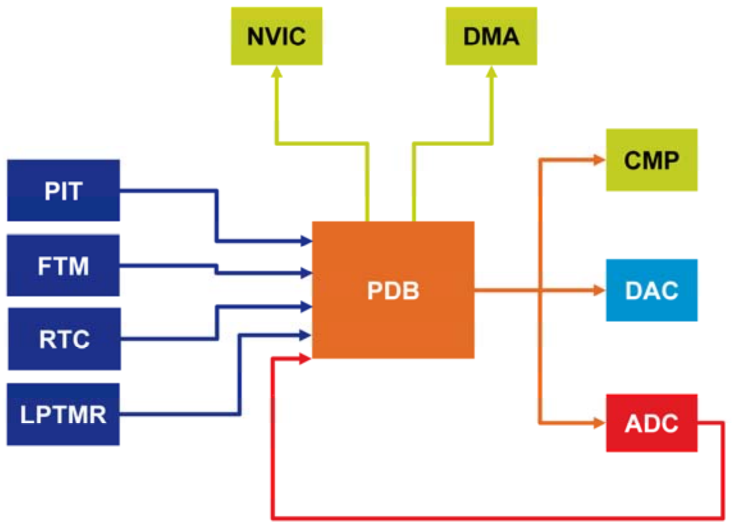
2. On-chip Interconnections and Inter-module Dependencies

freescale External Use | 11

Let’s move onto on-chip interconnection and inter-module dependencies.



PDB Interconnections and Inter-module Dependencies



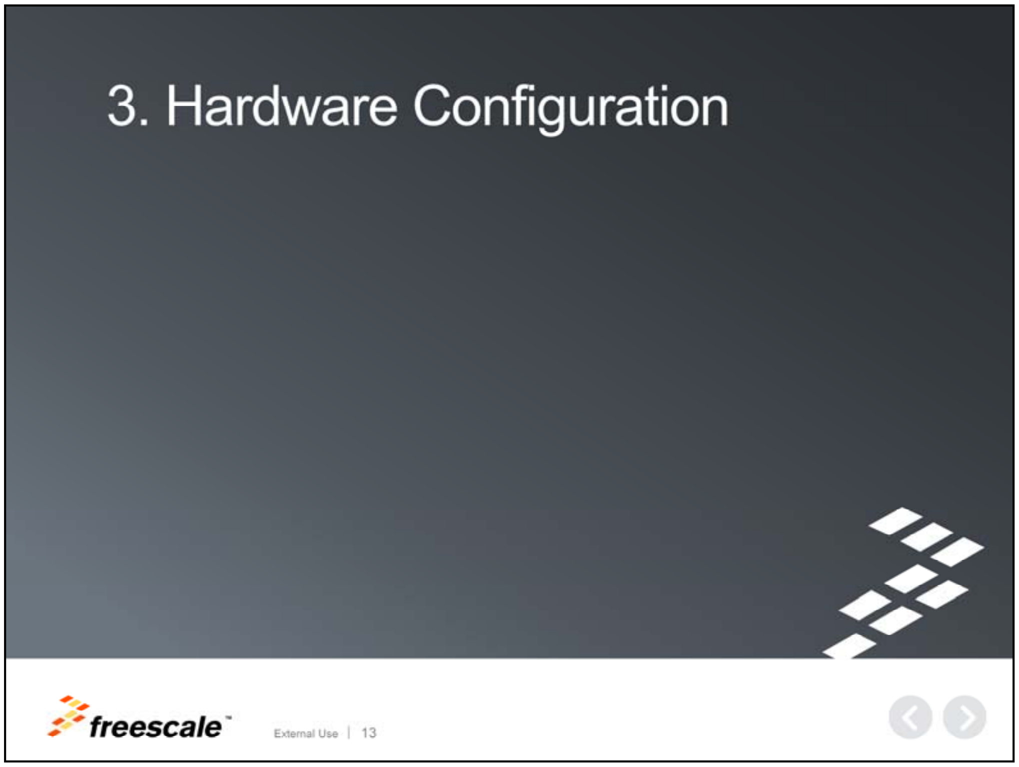
freescale External Use | 12

PDB Interconnections and Inter-module Dependencies

Like we saw previously, the PDB is capable of generating trigger outputs to analog modules such as the ADC, DAC and comparator.

To achieve this, the PDB can be triggered by internal modules like the ADC or TIMER modules like the PIT, FTM, RTC and LPTMR.

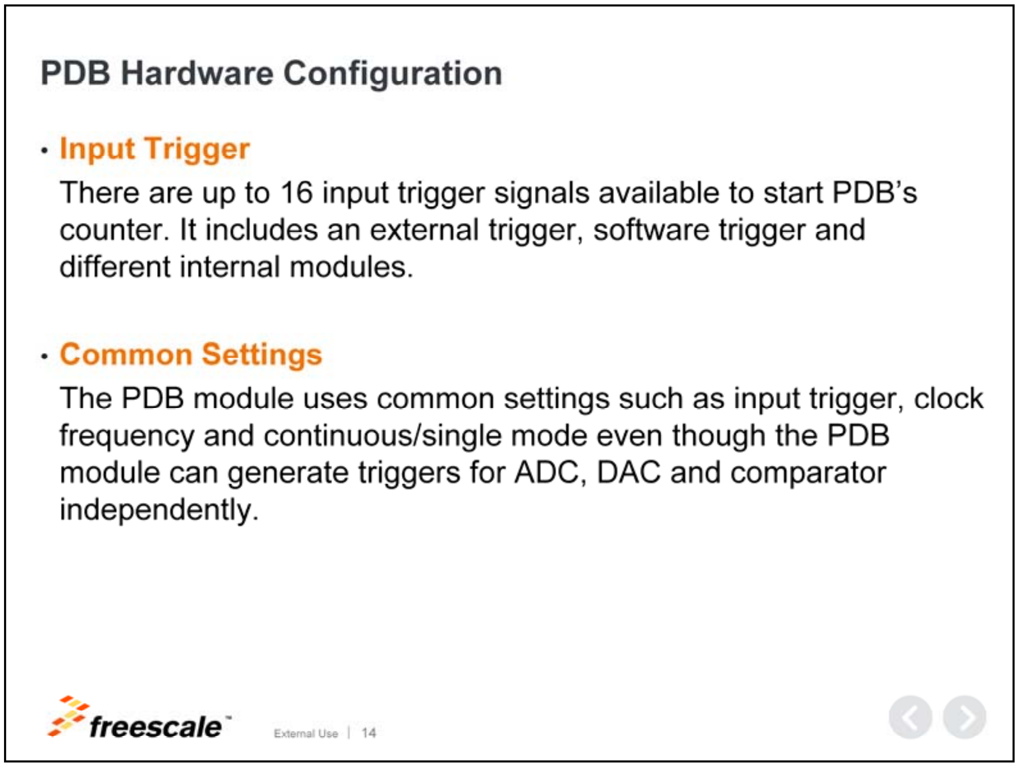
The PDB also has a dedicated channel, which can generate an interrupt in the NVIC module or request a DMA transfer.



3. Hardware Configuration

External Use | 13

Now, let’s talk about hardware configuration.


PDB Hardware Configuration

- **Input Trigger**
  There are up to 16 input trigger signals available to start PDB's counter. It includes an external trigger, software trigger and different internal modules.

- **Common Settings**
  The PDB module uses common settings such as input trigger, clock frequency and continuous/single mode even though the PDB module can generate triggers for ADC, DAC and comparator independently.

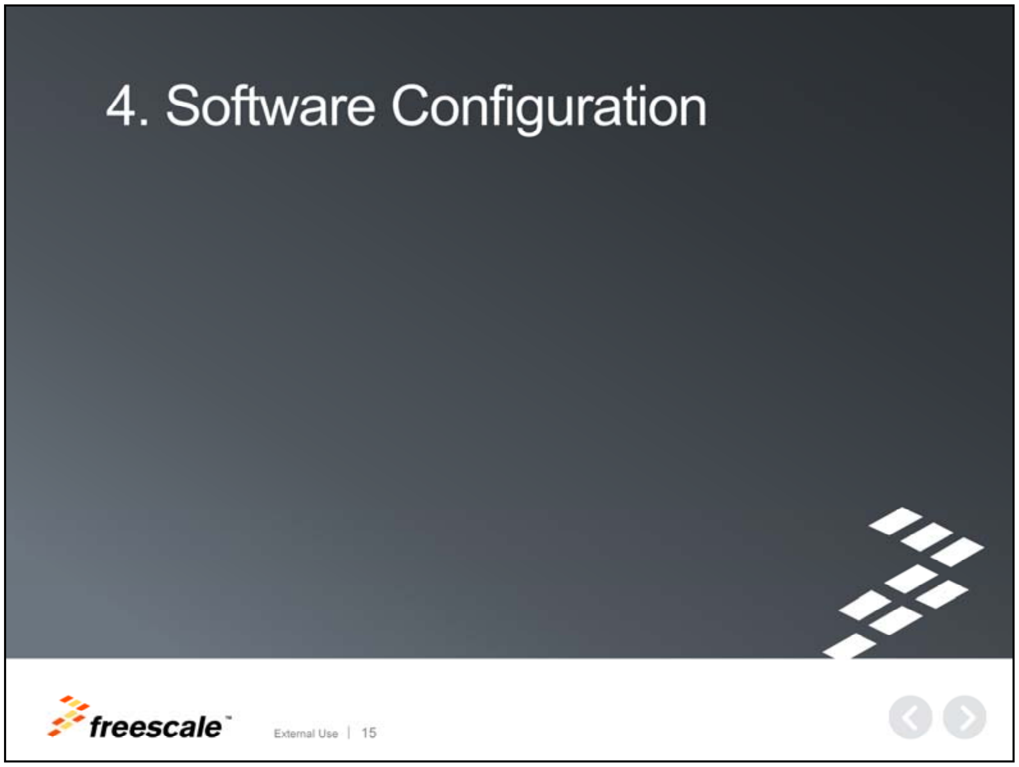
PDB Hardware Configuration

Input Trigger

The PDB can select up to 16 input trigger signals, including: an external trigger, a software trigger, and other internal modules.

Common Settings

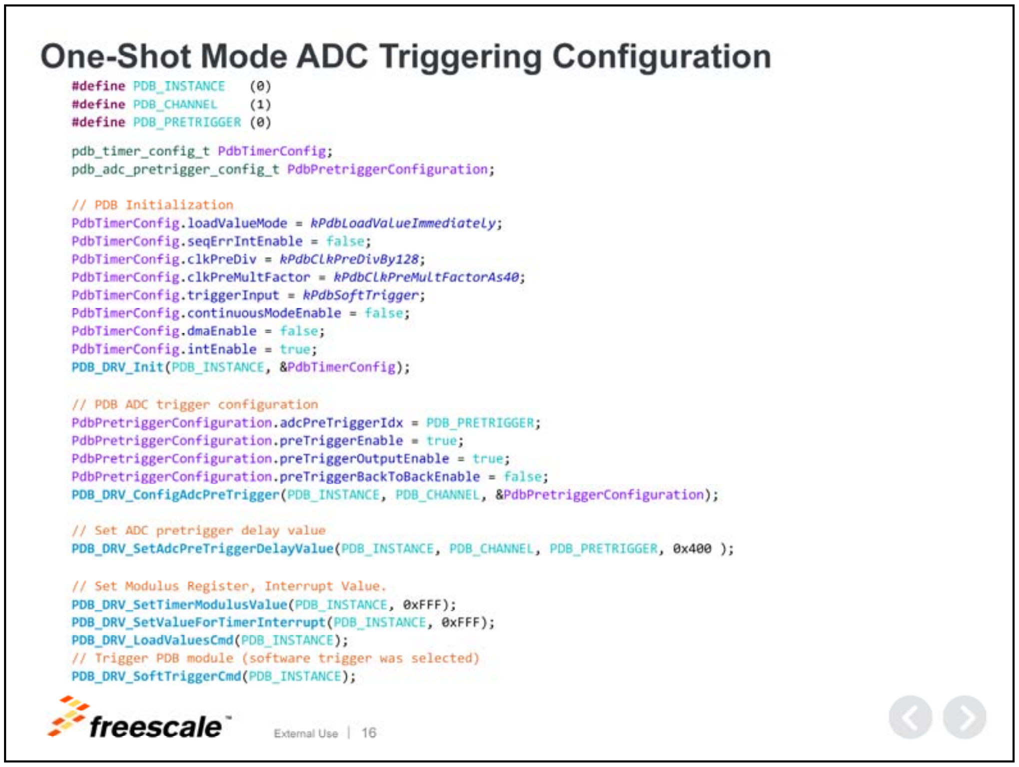
The PDB module uses common settings such as input trigger, clock frequency and continuous or single mode even though it can generate triggers for ADC, DAC and comparator independently.



4. Software Configuration

freescale™ External Use | 15

Let’s move onto software configuration.



One-Shot Mode ADC Triggering Configuration

```c
#define PDB_INSTANCE   (0)
#define PDB_CHANNEL    (1)
#define PDB_PRETRIGGER (0)

pdb_timer_config_t PdbTimerConfig;
pdb_adc_pretrigger_config_t PdbPretriggerConfiguration;

// PDB Initialization
PdbTimerConfig.loadValueMode = kPdbLoadValueImmediately;
PdbTimerConfig.seqErrIntEnable = false;
PdbTimerConfig.clkPreDiv = kPdbClkPreDivBy128;
PdbTimerConfig.clkPreMultFactor = kPdbClkPreMultFactorAs40;
PdbTimerConfig.triggerInput = kPdbSoftTrigger;
PdbTimerConfig.continuousModeEnable = false;
PdbTimerConfig.dmaEnable = false;
PdbTimerConfig.intEnable = true;
PDB_DRV_Init(PDB_INSTANCE, &PdbTimerConfig);

// PDB ADC trigger configuration
PdbPretriggerConfiguration.adcPreTriggerIdx = PDB_PRETRIGGER;
PdbPretriggerConfiguration.preTriggerEnable = true;
PdbPretriggerConfiguration.preTriggerOutputEnable = true;
PdbPretriggerConfiguration.preTriggerBackToBackEnable = false;
PDB_DRV_ConfigAdcPreTrigger(PDB_INSTANCE, PDB_CHANNEL, &PdbPretriggerConfiguration);

// Set ADC pretrigger delay value
PDB_DRV_SetAdcPreTriggerDelayValue(PDB_INSTANCE, PDB_CHANNEL, PDB_PRETRIGGER, 0x400 );

// Set Modulus Register, Interrupt Value.
PDB_DRV_SetTimerModulusValue(PDB_INSTANCE, 0xFFF);
PDB_DRV_SetValueForTimerInterrupt(PDB_INSTANCE, 0xFFF);
PDB_DRV_LoadValuesCmd(PDB_INSTANCE);
// Trigger PDB module (software trigger was selected)
PDB_DRV_SoftTriggerCmd(PDB_INSTANCE);
```

External Use | 16

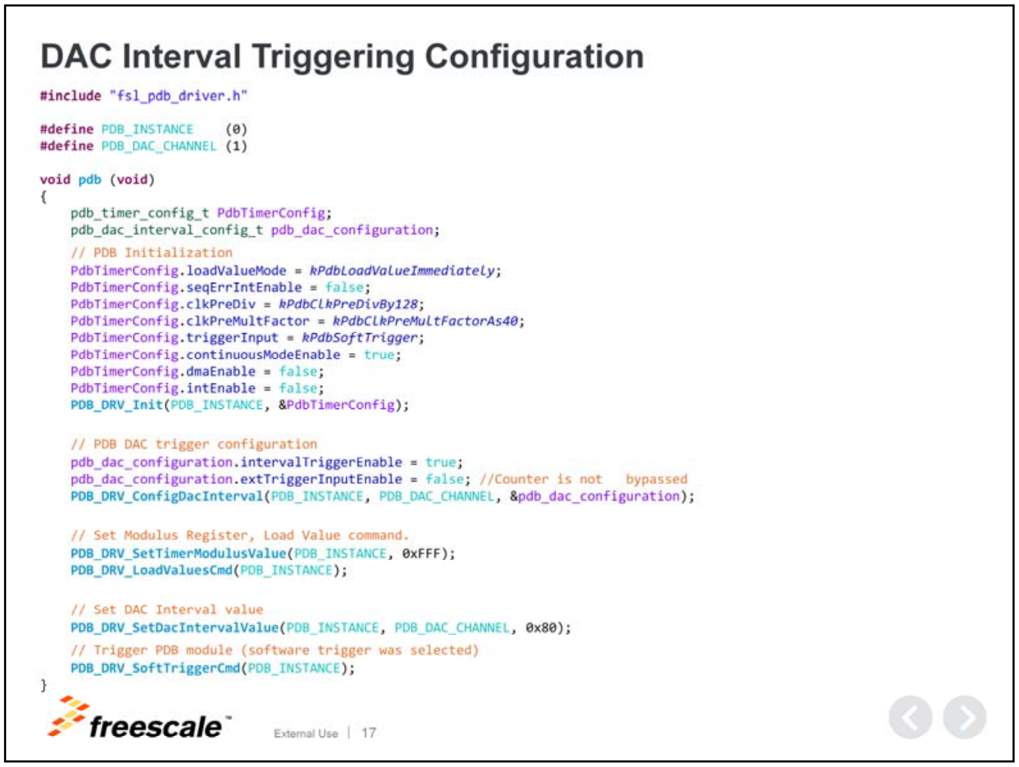
One-Shot Mode ADC Triggering Configuration

To initialize the PDB driver, prepare configuration structure PdbTimerConfig and populate it with the desired configuration. After the configuration structure is available, initialize the PDB module by calling the API function PDB_DRV_Init().

For ADC triggering, prepare pdb_pretrigger_configuration and populate with desired values. After the configuration structure is available, initialize the ADC trigger channels by calling API function PDB_DRV_ConfigAdcPreTrigger().

Set the ADC pre-trigger delay value. When the counter reaches this value, the pre-trigger will be asserted and sets the modulus register and interrupt value.

If the software trigger is enabled in the PDB configuration, it will trigger the PDB module. Otherwise, it will wait until an input trigger is received.



DAC Interval Triggering Configuration

```c
#include "fsl_pdb_driver.h"

#define PDB_INSTANCE    (0)
#define PDB_DAC_CHANNEL (1)

void pdb (void)
{
    pdb_timer_config_t PdbTimerConfig;
    pdb_dac_interval_config_t pdb_dac_configuration;
    // PDB Initialization
    PdbTimerConfig.loadValueMode = kPdbLoadValueImmediately;
    PdbTimerConfig.seqErrIntEnable = false;
    PdbTimerConfig.clkPreDiv = kPdbClkPreDivBy128;
    PdbTimerConfig.clkPreMultFactor = kPdbClkPreMultFactorAs40;
    PdbTimerConfig.triggerInput = kPdbSoftTrigger;
    PdbTimerConfig.continuousModeEnable = true;
    PdbTimerConfig.dmaEnable = false;
    PdbTimerConfig.intEnable = false;
    PDB_DRV_Init(PDB_INSTANCE, &PdbTimerConfig);

    // PDB DAC trigger configuration
    pdb_dac_configuration.intervalTriggerEnable = true;
    pdb_dac_configuration.extTriggerInputEnable = false; //Counter is not   bypassed
    PDB_DRV_ConfigDacInterval(PDB_INSTANCE, PDB_DAC_CHANNEL, &pdb_dac_configuration);

    // Set Modulus Register, Load Value command.
    PDB_DRV_SetTimerModulusValue(PDB_INSTANCE, 0xFFF);
    PDB_DRV_LoadValuesCmd(PDB_INSTANCE);

    // Set DAC Interval value
    PDB_DRV_SetDacIntervalValue(PDB_INSTANCE, PDB_DAC_CHANNEL, 0x80);
    // Trigger PDB module (software trigger was selected)
    PDB_DRV_SoftTriggerCmd(PDB_INSTANCE);
}
```

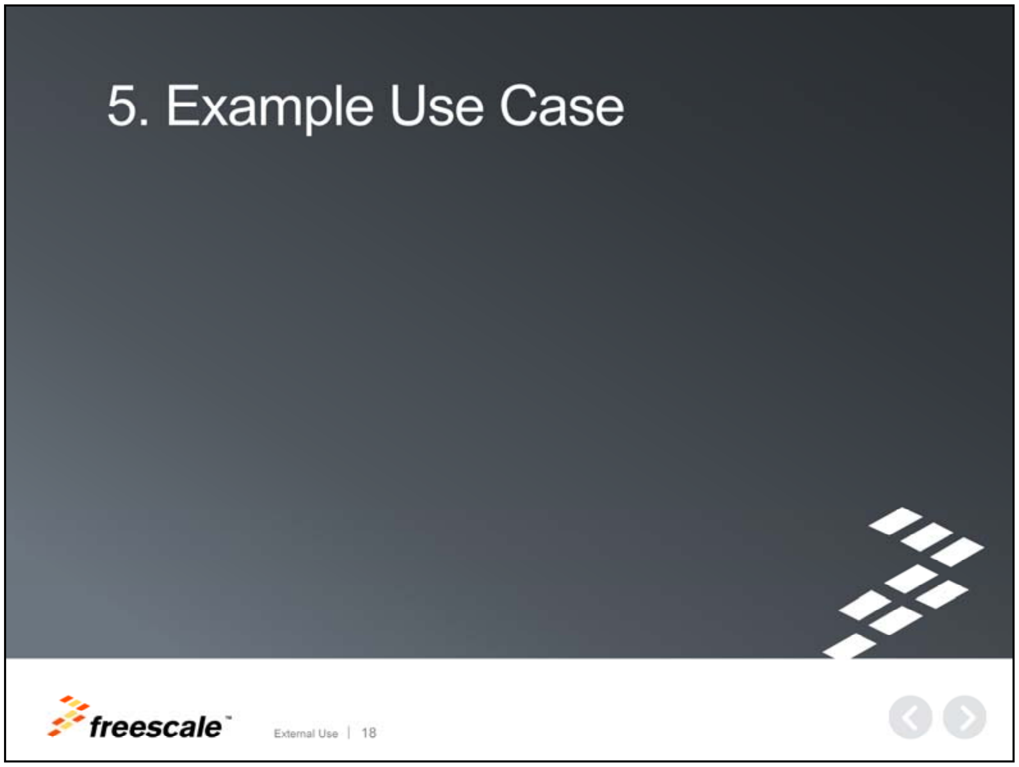
External Use | 17

DAC Interval Triggering Configuration

First, initialize the PDB common settings, such as the PdbTimerConfig structure and modulus register.

For the DAC Interval trigger configuration, prepare the pdb_dac_configuration structure and populate it with desired values. After the configuration structure is filled, initialize the DAC interval channel by calling the API function PDB_DRV_ConfigDacInterval().

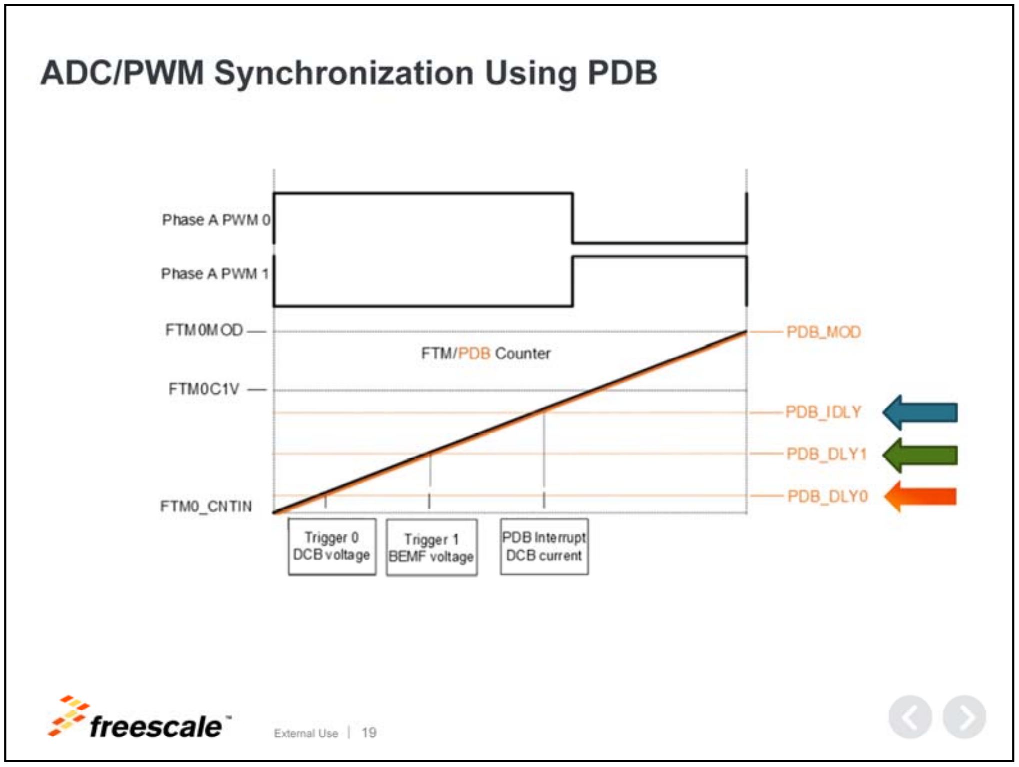
Set the DAC interval value by calling API function PDB_DRV_SetDacIntervalValue(), specifying which DAC channel interval will be written and trigger the PDB counter.



5. Example Use Case

freescale External Use | 18

Now, let’s go review an example use case.



ADC/PWM Synchronization Using PDB

freescale External Use | 19

ADC/PWM Synchronization Using PDB

In this example, we demonstrate how the PDB block can be used to synchronize FTM/PWM and ADC channels. This specific technique is commonly used for motor control applications.

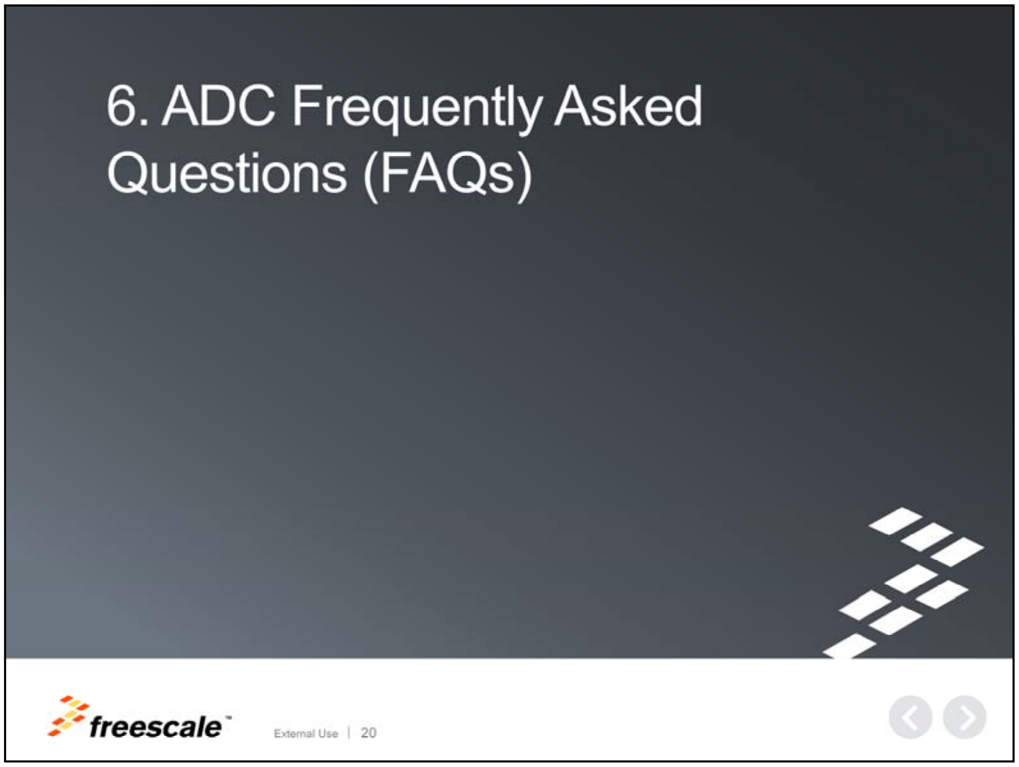
The PDB module is configured to use an FTM input trigger source. PDB channel 1 will generate two triggers, Trigger 0 and Trigger 1 and an interrupt.

Trigger 0 is used for DC bus voltage measurement.

Trigger 1 is used for back-EMF voltage of phase A and phase B.

DC bus current measurement is done by a PDB interrupt using an ADC software trigger. In the PDB interrupt service routine, one ADC channel is switched to measure DC bus current and then ADC is triggered by software. After conversion has completed and the result is read, the ADC channel is switched back into PDB triggering mode.

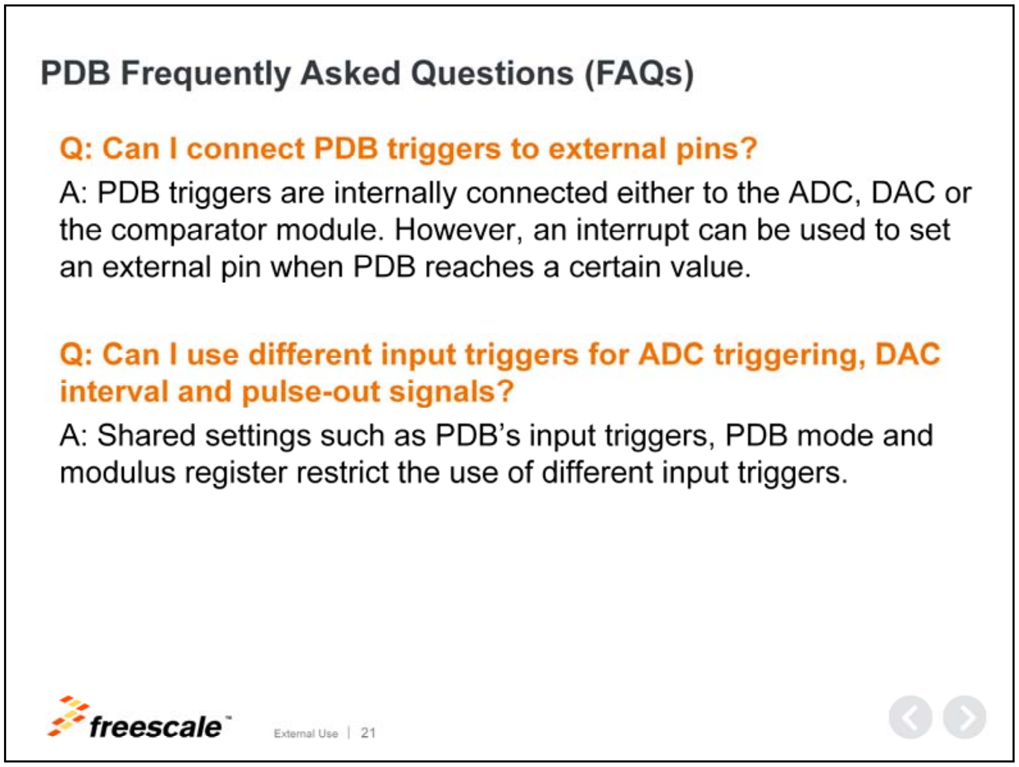
PDB’s modulus register is set according to the FTM MOD register, so both counters are reset at the same time.



6. ADC Frequently Asked Questions (FAQs)

External Use | 20

Finally, some frequently asked questions.


PDB Frequently Asked Questions (FAQs)

Q: Can I connect PDB triggers to external pins?

A: PDB triggers are internally connected either to the ADC, DAC or the comparator module. However, an interrupt can be used to set an external pin when PDB reaches a certain value.

Q: Can I use different input triggers for ADC triggering, DAC interval and pulse-out signals?

A: Shared settings such as PDB’s input triggers, PDB mode and modulus register restrict the use of different input triggers.

freescale External Use | 21

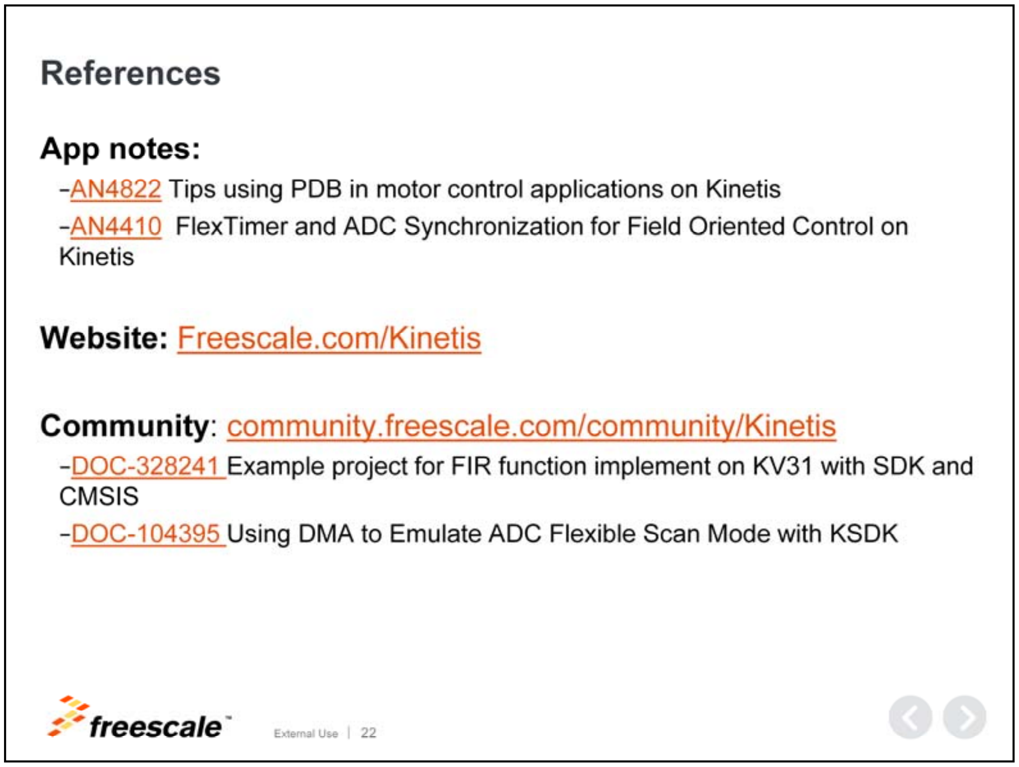
PDB Frequently Asked Questions (FAQs)

Q: Can I connect PDB triggers to external pins?

A: PDB triggers are internally connected either to the ADC, DAC or the comparator module. However, an interrupt can be used to set an external pin when PDB reaches a certain value.

Q: Can I use different input triggers for ADC triggering, DAC interval and pulse-out signals?

A: Shared settings such as PDB’s input triggers, PDB mode and modulus register restrict the use of different input triggers.



References

App notes:

-AN4822 Tips using PDB in motor control applications on Kinetis

-AN4410 FlexTimer and ADC Synchronization for Field Oriented Control on Kinetis

Website: Freescale.com/Kinetis

Community: community.freescale.com/community/Kinetis

-DOC-328241 Example project for FIR function implement on KV31 with SDK and CMSIS

-DOC-104395 Using DMA to Emulate ADC Flexible Scan Mode with KSDK

External Use | 22

This concludes our presentation on the PDB module for Kinetis K series MCUs.

For more information on the PDB module, please visit the resources listed here.

We also invite you to visit us on the web at Freescale.com/Kinetis and check out our Kinetis community page.

www.Freescale.com

© 2015 Freescale Semiconductor, Inc. | *External Use*