AN13608

Chaining FlexTimers on Layerscape Devices

Rev. 0 — 24 March 2022 Application Note

Contents

1 Introduction

Most Layerscape product families feature FlexTimers (FTMs) which individually have 16-bit count capability. When longer counts are required, it is possible to chain/cascade multiple FTMs to count with 32 bits or more. How many FTMs can be chained into a longer count and which combinations are possible depends on the SoC family as shown in the following table.

Table 1. FlexTimer Chaining Options

Layerscape SoC Family	FlexTimers	Cascade options (least significant first)	Chain Control Register
LS1021A	8	5+1, 6+2, 7+3, 8+4	SCFG_FTM_CHAIN_CONFIG
LS1043A			
LS1046A			
LS1028A	8	1-8 and subranges	FTMCR
LS1088A	4	1-4 and subranges	FTMCR
LS2088A			
LX2160A	2	1-2	FTMCR

2 Documentation Inconsistencies

Before we describe how to properly cascade the FTMs, it should be noted that some Reference Manual releases have inconsistent content which may be confusing with respect to chaining FTMs. NXP strives to correct this going forward, but you may find the following issues in documentation currently available to you:

- The FTMCR documentation may incorrectly show more bits than you have FTMs on the device.
- The FTMCR or SCFG_FTM_CHAIN_CONFIG documentation may fail to mention that CH7 of an FTM with lesser significance feeds PHA of the chained FTM with more significance as done early in the FlexTimer chapter only for SCFG_FTM_CHAIN_CONFIG based devices.
- The QDCTRL register at offset 0x80 in the FTM memory map may not be documented, nor are bit 0, called QUADEN, or bit 3, called QUADMODE documented then. You can find missing descriptions in, for example, the current LS1046ARM.
- Where quadrature decoding support is shown in the FlexTimer chapter, it may not be made clear that it would refer to external pin access specifically. Internally, all FTMs that can be added to a chain have that support, even though it is then not usable for anything but chaining.
- SCFG_FTM_RESET[FTMCHNx] bits may be incorrectly documented as chaining bits when SCFG_FTM_CHAIN_CONFIG[FTM_CHNx] should have been referenced.

- SCFG_FTM_CHAIN_CONFIG[FTM_CHNx] documentation does not state that the higher numbered FTM is the least significant part and the lower numbered FTM the most significant part of a chained 32-bit counter.

Keep checking for the latest documentation to ensure that you will over time see necessary corrections.

3 What does the Chain Control Register do for you?

Whenever an FTM overflows, the overflow must be propagated as an appropriate counter signal to the FTM forming the next stage of the chain. The control register enables the SoC logic that externally connects FTMs. What happens is that the output of FTM<x> channel 7 is connected to the quadrature decoder logic of FTM<y>. The quadrature decoder then clocks the pre-scaler of that FTM. SoCs with FTMCR as control register in general permit more flexibility in chaining FTMs.

So to make the chain work, we not only must set the control register, but we also must program FTM outputs and inputs correspondingly.

4 Configuring the FlexTimer output

In a chain, every FTM but the last one needs a special output configuration. To configure channel 7 of an FTM for chaining, we might think that we need a toggle whenever the timer overflows. The problem with that is however that the next timer would only count with half the intended frequency as it only gets the right edge to count every second toggle. What we want is to generate a pulse so that the next timer will count up every time the current one overflows.

One option to do that is to set C7SC[MSB] and C7SC[ELSB], which in combination select the mode of “Edge-Aligned PWM” and a configuration of “High-true pulses (clear output on match-up)”. To complete the setup we also want to set C7V to 0xffff, so that our pulse gets generated at the right time.

5 Configuring the chained FlexTimer input

In a chain every FTM but the first one needs a special input configuration. On the FTM<y> that should count via its quadrature decoder, we want to set it to decode count and direction by setting QDCTRL[QUADMODE] in addition to enabling it via QDCTRL[QUADEN]. This allows us to count the pulses generated by the FTM<x> that feeds us via the PHA quadrature decoder input.

6 Reading a chained counter

As the multiple 16b CNT registers cannot be read atomically, a wrap from one value to the next can occur at any time, leading to an inconsistent readout of the current value. The usual procedure to overcome it is the following:

- Read all 16b values of the chained CNT registers in the first set of variables starting with the most significant counter and ending with the least significant counter
- Read all 16b values again in the same order into a second set of variables, excluding the least significant CNT register
- Compare the first set with the second set. If they differ, start again from the top.
- Use the first set of variables as the consistent chained counter value.

This sequence exploits the fact that reads are much faster than the least significant counter can ever overflow. The likelihood of encountering an overflow during the read sequence is therefore low. If an overflow occurs it will be detected and the next attempt would then succeed to read all values without any overflows in the chain. Unconditionally retrying on any difference is a necessity though as, for example, interrupts could affect the timing and add unexpected delays.

7 Chaining example with SCFG_FTM_CHAIN_CONFIG

The following describes a simple example to chain FTMs 2 and 6 on an LS1046A into a 32-bit counter as an example for SoCs built with SCFG_FTM_CHAIN_CONFIG. FTM6 is counting with a reduced RTC clock in this example to make the count easily observable and the overflow will then clock FTM2 to form a 32-bit timer. The example can also be used on the LS1043A or LS1021A families, provided the appropriate clocking for the hardware is selected.

The example only shows the steps necessary for chaining timers and basic counter enablement. Other required initialization steps or application necessities are omitted and must be applied as described in the SoC reference manual.

To permit reproducing the example easily, simple U-Boot commands are used to directly write the registers. Note that register access in this example is Big Endian to match the memory map, that is, the 32-bit words written are byte swapped.

Table 2. Chaining of FTM2 and 6 on LS1046A

U-Boot command	Description
=> mw.l 29e0054 05000000	Enable FTM mode on FTM2 (msb counter)
=> mw.l 2a20054 05000000	Enable FTM mode on FTM6 (lsb counter)
=> mw.l 29e0008 ffff0000	Set Modulo to max for a free running counter
=> mw.l 2a20008 ffff0000	Set Modulo to max for a free running counter
=> mw.l 1570154 00400000	Use SCFG_FTM_CHAIN_CONFIG to chain FTM2 and FTM6
=> mw.l 2a20044 28000000	Set FTM6 C7SC[MSB|ELSB] to generate a pulse
=> mw.l 2a20048 ffff0000	Set FTM6 C7V to 0xffff to generate a pulse at wrap time
=> mw.l 29e0080 09000000	Set FTM2 QDCTRL[QUADMODE|QUADEN]
=> mw.l 2a20000 12000000	Enable RTC clock / 4 for FTM2 as example reference clock

The choice of the reference clock is arbitrary. A slow clock was chosen so that when dumping the `CNT` registers it is easier to observe the operation of the timer chain.

8 Chaining example with FTMCR

The following describes an example to chain `FTMs` 5 through 7 on an `LS1028A` into a 48-bit counter as example for SoCs built with `FTMCR`. To permit reproducing it easily, simple U-Boot commands are used to directly write the registers.

The example only shows the steps necessary for chaining timers and basic counter enablement. Other required initialization steps or application necessities are omitted and must be applied as described in the SoC reference manual.

Table 3. Chaining of FTMs 5 through 7 on an LS1028A

U-Boot command	Description
=> mw.l 2860054 00000005	Enable FTM mode on FTM7 (msb counter)
=> mw.l 2850054 00000005	Enable FTM mode on FTM6
=> mw.l 2840054 00000005	Enable FTM mode on FTM5 (lsb counter)
=> mw.l 2860008 0000ffff	Set Modulo to max for a free running counter
=> mw.l 2850008 0000ffff	Set Modulo to max for a free running counter
=> mw.l 2840008 0000ffff	Set Modulo to max for a free running counter
=> mw.l 1e00960 00000030	Use FTMCR to chain FTM5-7
=> mw.l 2840044 00000028	Set FTM5 C7SC[MSB|ELSB] to generate a pulse
=> mw.l 2840048 0000ffff	Set FTM5 C7V to 0xffff to generate a pulse at wrap time
=> mw.l 2850080 00000009	Set FTM6 QDCTRL[QUADMODE|QUADEN]
=> mw.l 2850044 00000028	Set FTM6 C7SC[MSB|ELSB] to generate a pulse

Table continues on the next page...

Table 3. Chaining of FTMs 5 through 7 on an LS1028A (continued)

U-Boot command	Description
=> mw.l 2850048 0000ffff => mw.l 2860080 00000009	Set FTM6 C7V to 0xffff to generate a pulse at wrap time Set FTM7 QDCTRL[QUADMODE|QUADEN]
=> mw.l 2840000 0000000f	Enable System Clock / 128 for FTM5 as example reference clock

The choice of the reference clock is arbitrary. A somewhat slow clock was chosen so that when dumping the `CNT` registers it is easier to observe the operation of the timer chain.

The general method to chain multiple `FTMs` should be clearly visible here. As other devices with `FTMCR` like `LS1088A, LS2088A,` or `LX2160A` have fewer timers, this example needs to be properly adjusted to write the right timer registers and chain only up to the maximum amount available.

The necessary changes to run this example of an `LX2160A` would be, for example, to only write the registers of `FTM1` at 0x2800000 and `FTM2` at 0x2810000 instead of `FTM5` and `FTM7`, and set `FTMCR` to 0x01 to only connect those two `FTMs`. The same modification for the example works for an `LS1088A` or `LS2088A`.

Table 4. Chaining FTM1 and 2 on LX2160A

U-Boot command	Description
=> mw.l 2810054 00000005 => mw.l 2800054 00000005 => mw.l 2810008 0000ffff => mw.l 2800008 0000ffff	Enable FTM mode on FTM2 (msb counter) Enable FTM mode on FTM1 (lsb counter) Set Modulo to max for a free running counter Set Modulo to max for a free running counter
=> mw.l 1e00960 00000001 => mw.l 2800044 00000028 => mw.l 2800048 0000ffff => mw.l 2810080 00000009	Use FTMCR to chain FTM1-2 Set FTM1 C7SC[MSB|ELSB] to generate a pulse Set FTM1 C7V to 0xffff to generate a pulse at wrap time Set FTM2 QDCTRL[QUADMODE|QUADEN]
=> mw.l 2800000 0000000f	Enable System Clock / 128 for FTM1 as example reference clock

9 Revision history

Table 5. Revision history

Revision Number	Date	Substantive Changes
0	24 March 2022	Initial release

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this data sheet expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile — are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

Airfast — is a trademark of NXP B.V.

Bluetooth — the Bluetooth wordmark and logos are registered trademarks owned by Bluetooth SIG, Inc. and any use of such marks by NXP Semiconductors is under license.

Cadence — the Cadence logo, and the other Cadence marks found at www.cadence.com/go/trademarks are trademarks or registered trademarks of Cadence Design Systems, Inc. All rights reserved worldwide.

CodeWarrior — is a trademark of NXP B.V.

ColdFire — is a trademark of NXP B.V.

ColdFire+ — is a trademark of NXP B.V.

EdgeLock — is a trademark of NXP B.V.

EdgeScale — is a trademark of NXP B.V.

EdgeVerse — is a trademark of NXP B.V.

eIQ — is a trademark of NXP B.V.

FeliCa — is a trademark of Sony Corporation.

Freescale — is a trademark of NXP B.V.

HITAG — is a trademark of NXP B.V.

ICODE and I-CODE — are trademarks of NXP B.V.

Immersiv3D — is a trademark of NXP B.V.

I2C-bus — logo is a trademark of NXP B.V.

Kinetis — is a trademark of NXP B.V.

Layerscape — is a trademark of NXP B.V.

Mantis — is a trademark of NXP B.V.

MIFARE — is a trademark of NXP B.V.

MOBILEGT — is a trademark of NXP B.V.

NTAG — is a trademark of NXP B.V.

Processor Expert — is a trademark of NXP B.V.

QorIQ — is a trademark of NXP B.V.

SafeAssure — is a trademark of NXP B.V.

SafeAssure — logo is a trademark of NXP B.V.

StarCore — is a trademark of NXP B.V.

Synopsys — Portions Copyright © 2021 Synopsys, Inc. Used with permission. All rights reserved.

Tower — is a trademark of NXP B.V.

UCODE — is a trademark of NXP B.V.

VortiQa — is a trademark of NXP B.V.

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 24 March 2022
Document identifier: AN13608