



Utilizing the Trigger Routing Unit for System Level Synchronization

Contributed by Akash Agarwal and Joe Tarkoff

Rev 1 – October 9, 2013

Introduction

In embedded processor systems, synchronization of operation between events is of paramount importance. Interrupts are a common way of synchronizing operation between events. These can be timed by enabling the first event as an interrupt, which gets detected by the core. On detection of the interrupt, the core can start or stop a new process or event. Interrupts provide a reliable synchronization medium as they are serviced through the processor core.

However, interrupts can add latency due to the time needed for core detection, context saving, and event servicing. The event processing latency depends on many conditions, such as priority levels or handling of simultaneous interrupts.

Several Analog Devices Processors provide the so called Trigger Routing Unit (TRU). The TRU is particularly well suited for system events synchronization, helping reduce the delays involved in the synchronization process, as it removes the core processing of the event and increases determinism. Therefore, the TRU provides an efficient alternative to core interrupts for event synchronization in embedded systems.



Although the concepts described in this EE-Note are TRU generic, this document focuses on how to utilize the Trigger Routing Unit on ADSP-BF60x Blackfin® Processors. For processor specific details, refer to the device Data Sheet and Hardware Reference.

The following items are discussed in this Engineer-to-Engineer Note:

- Trigger Routing Unit
- Trigger Masters and Slaves
- TRU Programming Guidelines
- Code Examples

TRU Description

The TRU provides a system level sequence control without core intervention. It maps the trigger masters (trigger generators) to trigger slaves (triggers receivers).

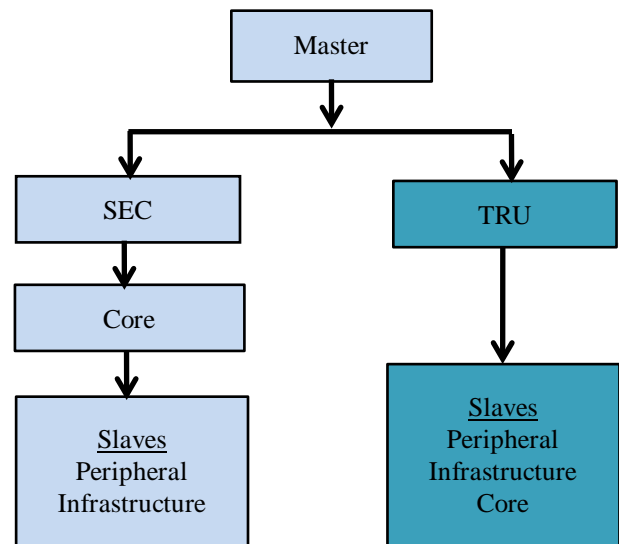


Figure 1. Interrupt and trigger execution sequence

As shown in Figure 1, a System Event generated at a master may be forwarded to the System Event Controller (SEC). At the SEC, the interrupt source and priority are determined based on the programmed configuration. The interrupt is then forwarded to the assigned core, which executes the corresponding interrupt service routine (ISR).

The processing of the interrupt at the SEC and the ISR execution by the core add further latency. In case of triggers, the TRU acts as a switch forwarding the request without any further processing. Upon receiving a trigger, this gets routed to the slave based on the programmed configuration.

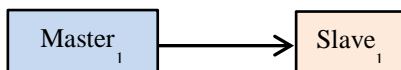
Trigger Masters and Trigger Slaves

On ADSP-BF60x Processors trigger slaves need to be pre-configured in order to respond to a trigger. In most cases, trigger masters generate a trigger when the corresponding interrupt is enabled. However, in certain cases triggers need to be specifically enabled^[1].

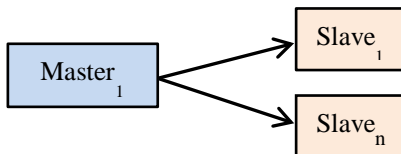
The TRU supports routing of any trigger master to any trigger slave, while also supporting routing of a single master trigger to multiple slaves.

The possible trigger propagation sequences are shown next:

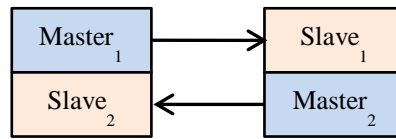
- Single Master & Single Slave



- Single Master & Multiple Slaves



- Multiple Roles (Masters & Slaves)



- Chain Trigger



On ADSP-BF60x processors, there are many modules, which can be programmed to act as both trigger master and slave. Table 1 provides a detailed list of modules that can serve in this dual role, or as a trigger master or trigger slave only.

Master & Slave	Master Only	Slave Only
GP Timer	CGU Event	RCU System Reset
SPORT DMA	PINT (GPIO)	NMI
SPI DMA	GP Counter Status	TRU Interrupt
RSI DMA (SDIO)	PWM 0/1	ACM Trigger
SDU DMA	ACM Event	SDU Slave
LP DMA	EMAC Status	Core Wakeup
UART DMA	USB DMA	
MDMA	SEC Fault	
EPPI DMA	PVP Status	
PIXC DMA	Software Driven	
PVP DMA	SWU Events	

Table 1. ADSP-BF60x Trigger Masters and Slaves

Furthermore, each of the modules listed above have specific events, which can be configured to work as either masters or slaves. [Table 2](#) and [Table 3](#) provide a verbose description of these

events. For a more detailed description of events, please refer to the *ADSP-BF60x Blackfin Processor Hardware Reference*^[1].

Trigger Master	Events
GP Timer	Active Edges (PININT mode)
	Delay Expired (PWMOUT mode)
	Width Plus Delay Expired (Watchdog mode)
	Period Expired (Watchdog)
DMA	1-D trigger at the end of the whole work unit
	2-D end of each row, or at the end of whole work unit
CGU	PLL locked and clocks synchronized
	CGU Event
PINT (Port)	User-programmable port activity (PINT0-PINT5)
PWM	Timer's period boundary reached
GP Counter	Illegal Gray/Binary Code Events
	Up/Down Count Events
	Zero-Count Events
	Overflow Events
	Boundary Match Events
	Zero Marker Events
ACM	Sampling completion
EMAC Status	Time Stamp Status
	MMC Receive Checksum Offload status
	MMC Transmit/Receive interrupt status
SEC Faults	Faults generated through the SEC by an interrupt-capable peripheral, core, or infrastructure block
Software Driven	Software writes (by master ID) to MMR register
SWU	User-determined bus monitoring conditions
USB DMA Status / Transfer Complete	VBUS drops below the VBUS valid threshold during session (A device only)
	SRP signaling detected (A device only)
	Device connected /disconnected detected (host mode)
	Babble detected (host mode)
	Session ends/ reset signaling detected (peripheral mode)
	Resume signaling detected in suspend mode
	Start-of-frame

PVP Status	Daisy chain completion
	Write error, completion of drain operation
	Pipe ready
	Data events
	Output/Status FIFO overflow
	Input overflow
	Histogram ready
	Integral unsigned/signed overflow
	Divide by zero
	ACU Output / Multiplication / Addition Saturation

Table 2. Trigger Master Events

Trigger Slave	Events
GP Timer	Start/Stop timer
DMA & MDMA	Start DMA transfer
RCU	System reset all functional units (except RCU_STAT, RCU_BCODE, RCU_CRCTL registers and units on the VDD_EXT power domain)
ACM	Initiate ADC sampling event
Wake-up Core	Breaks idle state
Interrupt Core	Trigger Routing Unit interrupts (through SEC)
	Raise 2 (Non-Maskable Interrupt - NMI)

Table 3. Trigger Slave Events

TRU Programming Guidelines

Here next are a few important points to be considered when programming the Trigger Routing Unit:

- The TRU operates in the System Clock (SYSCLK) domain.
- Software triggering may be used for trigger assertion.
- When using the GPIO or PWM as trigger masters for back to back triggers, the trigger assertions latched in the corresponding latch registers (PINT_ILAT and PWM_ILAT) must be cleared in software for back to back triggers to work.

For more details on the TRU programming model, please refer to the *ADSP-BF60x Blackfin Processor Hardware Reference*^[1].

Code Examples

The associated .ZIP file contains two code examples. Example 1 uses the TRU to time a definite frequency signal. Example 2 uses the TRU to time an unknown frequency signal.

Both of these examples have been implemented using CrossCore® Embedded Studio development tools and the ADSP-BF609 EZ-Board™ evaluation platform^[3].

Code Example 1

Experimental Setup

Figure 2 shows the experimental setup for code example 1. This example uses the TRU to time a definite external event without utilizing the core.

In order to do so, the peripherals are configured in the following roles:

- GP PB_12 as master
- GP TMR0 as slave
- MDMA2 (channels 25 and 26) as slave

An Agilent 33250A 80 MHz Function/Arbitrary Function generator^[4] is used to generate an external pulse frequency of 1 KHz and period of 1 millisecond (ms).

GPIO Port B is configured in GPIO mode and in input direction. Pin Interrupt Port 0 (PINT0) and

Port 1 (PINT1) are configured to detect the rising and falling edge of the input signal respectively.

The GP Timer 0 is configured in continuous PWM out mode counting based on SCLK0. The timer is configured to start on a receiving trigger.

The MDMA channels 25-26 are configured to initiate a write to `TIMER_RUN_CLR` register on a receiving trigger.

The TRU is configured to map the `PINT0` trigger to the timer and the `PINT1` trigger to MDMA channel.

The `TMR0_TMR0` pin is observed for the continuous PWM wave on a Tektronix, TDS 7404 Digital Phosphor Oscilloscope^[5] and captured using a Tektronix P7330 3.5 GHz differential probe^[5].

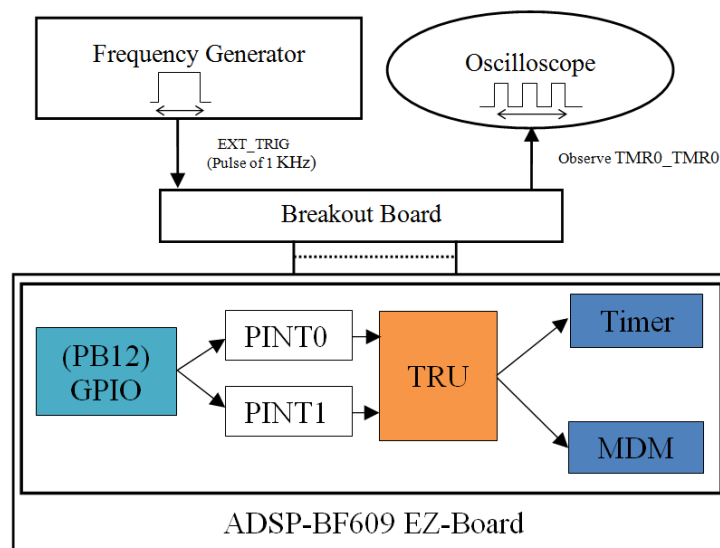


Figure 2. Experimental setup to time a definite frequency signal

Operation

The GPIO pins are applied with a single pulse of an external frequency signal of 1 KHz. PB12 pin detects the rising edge and generates PINT0 trigger. The TRU forwards the PINT0 trigger to `Timer0`, which initiates the timer counting `SCLK0`

periods. PB12 detects the falling edge of the input signal and generates the PINT1 trigger. The TRU forwards the PINT1 trigger to the MDMA channel 25-26. The MDMA channels then initiate a write on bit 0 of `TIMER_RUN_CLR` register, which then stops `Timer0`.

Results

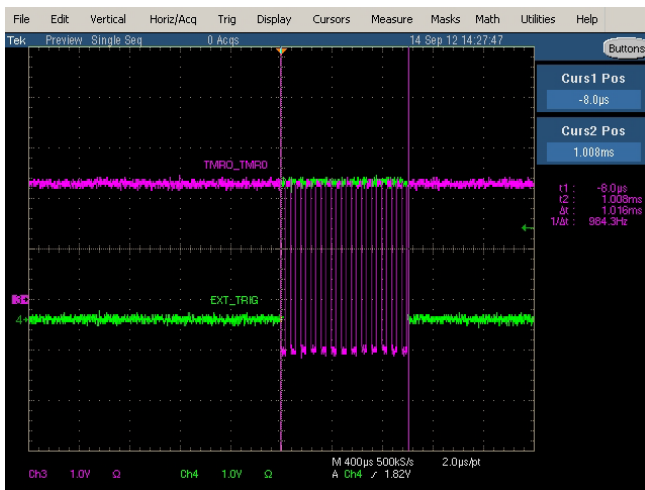


Figure 3. Example 1 scope plot

Figure 3, shows the result obtained from the code. The external trigger is indicated in green and the PWM wave is indicated in pink. The oscilloscope plot shows that as the EXT_TRIG signal goes high, Timer0 starts counting, and that as the EXT_TRIG signal goes low, Timer0 stops. The time difference as captured on the oscilloscope is 1.016 ms.

Code Example 2

Experimental Setup

Figure 4 shows the experimental setup for code example 2. This example uses the TRU to time an external event without utilizing the core.

In order to do so, the peripherals are configured in the following roles:

- GP PB_10 as master
- GP TMR0 as slave
- MDMA2 (channels 25 and 26) as slave

GPIO Port B is configured in GPIO mode and in input direction. Pin Interrupt Port 0 (PINT0) and Port 1 (PINT1) are configured to detect the rising and falling edge of the input signal respectively.

The GP Timer 0 is configured in continuous PWM out mode counting based on SCLK0. The timer is configured to start on a receiving trigger.

The MDMA channels 25-26 are configured to initiate a write to TIMER_RUN_CLR register on a receiving trigger.

The TRU is configured to map the PINT0 trigger to the timer and the PINT1 trigger to MDMA channel.

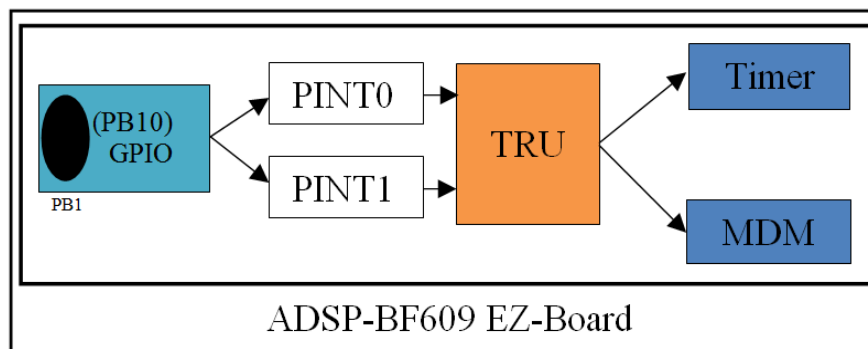


Figure 4. Experimental setup to time an unknown frequency signal

Operation

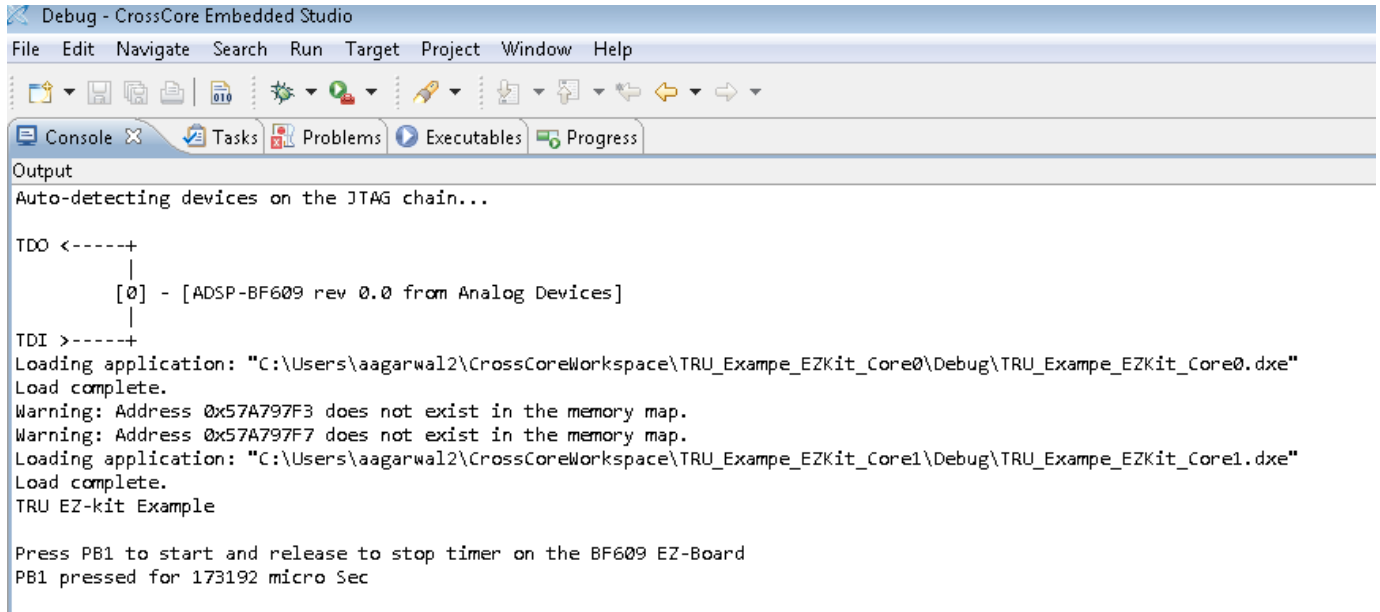
When pressing Push Button 1 (PB1) on the ADSP-BF609 EZ-Board evaluation platform, pin PB10 detects the rising edge and generates the PINT0 trigger. The TRU forwards the PINT0

trigger to Timer0, which initiates the timer counting on SCLK0. When releasing PB1, PINT1 signal detects the falling edge of the input signal and generates the PINT1 trigger. The TRU forwards the PINT1 trigger to the MDMA

channels 25-26. The MDMA channels then initiate a write on bit 0 of `TIMER_RUN_CLR` registers, which then stops `Timer0` and prints the duration, in milliseconds, for which the push button was pressed.

Results

Figure 5, shows the result obtained from the code. It shows that the Push button (PB1) was pressed for 17,3192 ms.



```

Debug - CrossCore Embedded Studio
File Edit Navigate Search Run Target Project Window Help
[Icons]
Console Tasks Problems Executables Progress
Output
Auto-detecting devices on the JTAG chain...

TDO <-----+
      |
      | [0] - [ADSP-BF609 rev 0.0 from Analog Devices]
      |
TDI >-----+
Loading application: "C:\Users\aaagarwal2\CrossCoreWorkspace\TRU_Exampe_EZKit_Core0\Debug\TRU_Exampe_EZKit_Core0.dxe"
Load complete.
Warning: Address 0x57A797F3 does not exist in the memory map.
Warning: Address 0x57A797F7 does not exist in the memory map.
Loading application: "C:\Users\aaagarwal2\CrossCoreWorkspace\TRU_Exampe_EZKit_Core1\Debug\TRU_Exampe_EZKit_Core1.dxe"
Load complete.
TRU EZ-kit Example

Press PB1 to start and release to stop timer on the BF609 EZ-Board
PB1 pressed for 173192 micro Sec
  
```

Figure 5. Example 2 console output

Conclusion

This EE-Note discussed a possible way of using the Trigger Routing Unit to off-load event timing activity from the processor's core. As described, the TRU can provide a more deterministic way of synchronizing critical events compared to using interrupts. By off-loading processing tasks from the core, this can be preserved for performing critical operations.

This document also provided a couple of examples for using the TRU to time a definite and unknown frequency signals. In these examples, it was also demonstrated how the TRU can be used to trigger memory transfers via de Memory-to-Memory DMA channels and how to write to a given Memory Mapped Register. The capability to write to MMR space using the DMA engine provides a unique way of changing peripheral behavior on-the-fly without processor core intervention.

References

- [1] *ADSP-BF60x Blackfin Processor Hardware Reference*. Rev 0.5, February 2013. Analog Devices, Inc.
- [2] *ADSP-BF606/ADSP-BF607/ADSP-BF608/ADSP-BF609 Blackfin Dual Core Embedded Processors Data Sheet*. Rev 0, June 2013. Analog Devices Inc.
- [3] *ADSP-BF609 EZ-KIT Lite Evaluation System Manual*. Rev 1.0, March 2012. Analog Devices, Inc.
- [4] *33250A Function / Arbitrary Waveform Generator, 80 MHz* (<http://www.home.agilent.com/en/pd-1000000803%3Aeps%3Apro-pn-33250A/function-arbitrary-waveform-generator-80-mhz?&cc=ES&lc=eng>). Agilent Technologies, Inc.
- [5] *TDS 7404 Digital Phosphor Oscilloscope* (<http://www.tek.com/datasheet/digital-phosphor-oscilloscopes-6>). Tektronix, Inc.
- [6] *P7330 3.5 GHz High-Performance Differential Probe* (<http://www.tek.com/datasheet/node/796147-high-performance-differential-probes>). Tektronix, Inc.

Document History

Revision	Description
<i>Rev 1 – October 9, 2013 by A. Agarwal and J. Tarkoff</i>	Initial release.