Freescale Semiconductor
Application Note

Document Number: AN4978
Rev 0, 08/2014

Generating a Quick and Controlled Waveform With the DAC

by: Arpita Agarwal

Contents

1 Overview....................1
2 Digital-to-analog converter (DAC)....................1
3 Direct memory access (DMA) controller....................3
4 Implementation....................3
4.1 CPU configuration....................4
4.2 DAC configuration....................4
4.3 DMA configuration....................5
5 Test setup and results....................6

1 Overview

This application note describes how to enable a Digital to Analog Converter (DAC) to generate a quick and controlled waveform (rise time and wave shapes) from a predefined set of digital data. The rise time of a waveform generated using Automatic mode can be much quicker, but generation of all waveform shapes is not possible through Automatic mode. Automatic mode can generate square, triangle, and sawtooth output waveforms only.

The DAC is used in Normal mode so that other wave-shapes can also be generated. Direct memory access (DMA) is used for transferring data to the DAC. This enables fast transfers of data, providing an efficient way to move blocks of data with minimal processor interaction.

2 Digital-to-analog converter (DAC)

The 12-bit digital-to-analog converter (DAC) provides a voltage reference to on-chip modules or an output to a package pin. It can also be used as a waveform generator to generate square, triangle, and sawtooth waveforms through automatic mode. The DAC can be put in power-down mode if needed. The maximum settling time will not exceed two microseconds with a maximum output load (3 kΩ || 400 pf) when the output swings from minimum output to maximum output or vice-versa. The DAC provides the ability to select either high-speed or low-speed mode and also supports two digital formats. It has a glitch filter to suppress output glitches during data conversion.

The DAC supports two conversion modes, asynchronous and synchronous. In asynchronous conversion mode, data can be immediately presented to the DAC and converted to an analog output when written to the DAC buffered data register. In synchronous conversion mode, data in the DAC buffered data register is controlled by the SYNC_IN signal when the buffered data is presented to the input of the DAC.

The DAC supports two operation modes: Normal and Automatic mode. In Normal mode, it generates an analog representation of digital words. In Automatic mode, it generates sawtooth, triangle, and square waveforms without CPU intervention, with programmable period, update rate, and range.

A quick waveform can be generated through Automatic mode, but not all wave-shapes can be generated. Normal mode can generate all the waveforms, but direct writing of the data register will be slow because of CPU intervention. A fast rise time cannot be achieved and it is difficult to have tight control. Since the DAC also has a DMA support mode, DMA is used for this purpose.

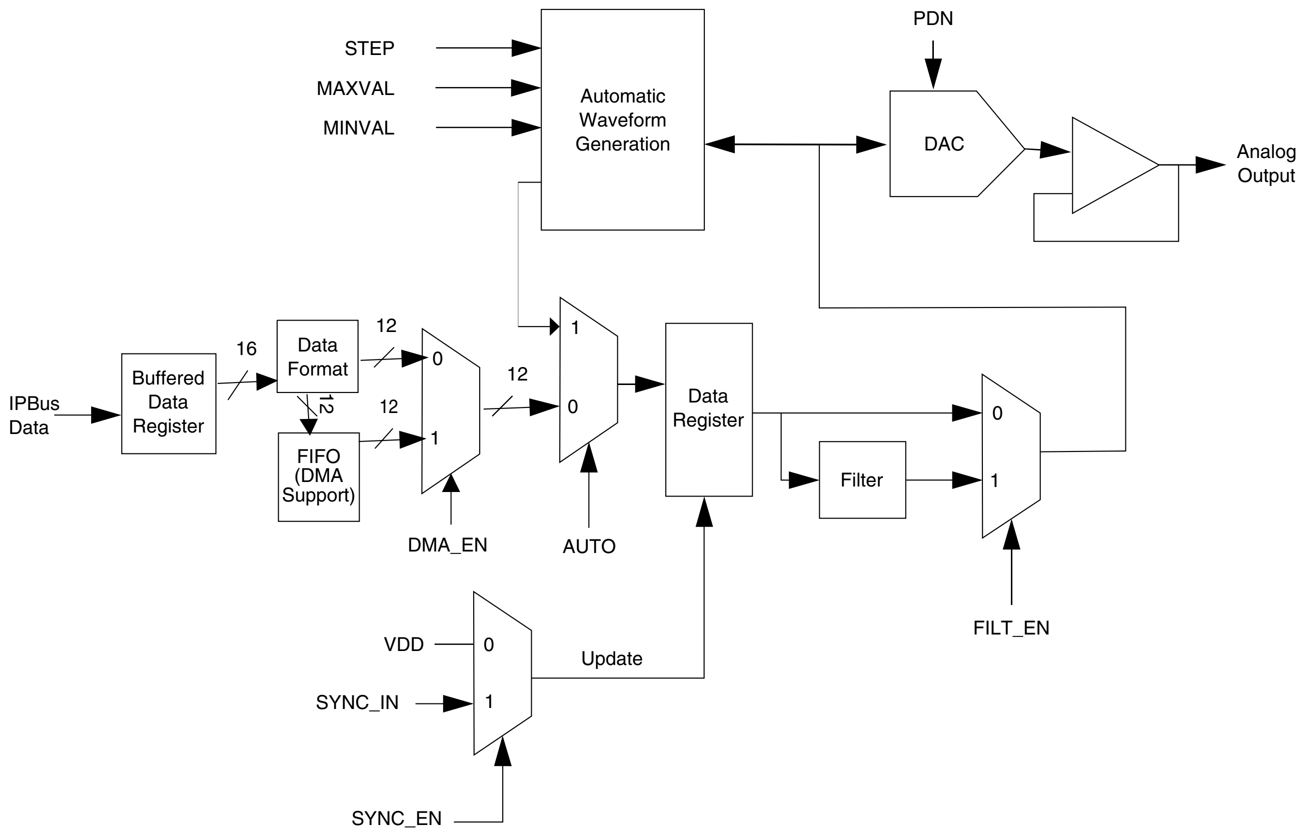


Figure 1. DAC block diagram

3 Direct memory access (DMA) controller

The DMA controller module enables fast transfers of data, providing an efficient way to move blocks of data with minimal processor interaction.

The DMA module has four channels that allow 8-bit, 16-bit, or 32-bit data transfers. Each channel has a dedicated Source Address register (SAR*n*), Destination Address register (DAR*n*), Status register (DSR*n*), Byte Count register (BCR*n*), and Control register (DCR*n*). Collectively, the combined program-visible registers associated with each channel define a transfer control descriptor (TCD).

All transfers are dual-address, moving data from a source memory location to a destination memory location with the module operating as a 32-bit bus master connected to the system bus. The programming model is accessed through a 32-bit connection with the slave peripheral bus.

DMA data transfers may be explicitly initiated by software or by peripheral hardware requests. The terms peripheral request and DREQ refer to a DMA request from one of the on-chip peripherals or package pins.

The DMA provides hardware handshake signals: either a DMA acknowledge (DACK) or a done indicator back to the peripheral.

The DMA has a Continuous mode or cycle-steal transfers from software or peripheral initiation.

One programmable input is selected from 16 possible peripheral requests per channel. Automatic hardware acknowledge/ done indicator from each channel is supported. There are independent source and destination address registers.

Optional modulo addressing, automatic updates of source and destination addresses, and independent transfer sizes for source and destination are a few of its features.

4 Implementation

Basic implementation in the DSC to achieve quick and controlled waveform is explained in this section. The example shown is based on the MC56F82748 DSC. Software requirements to demo this code are Code Warrior 10.5 or above and the Processor Expert utility. DAC and DMA peripherals are initialized through Processor Expert. These modules are interfaced as shown in Figure 2.

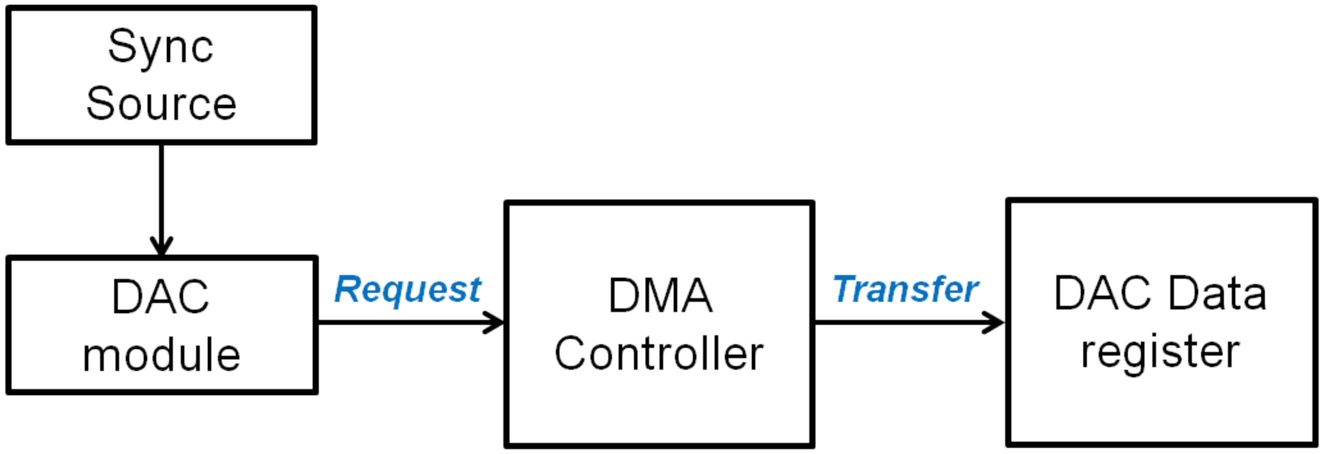
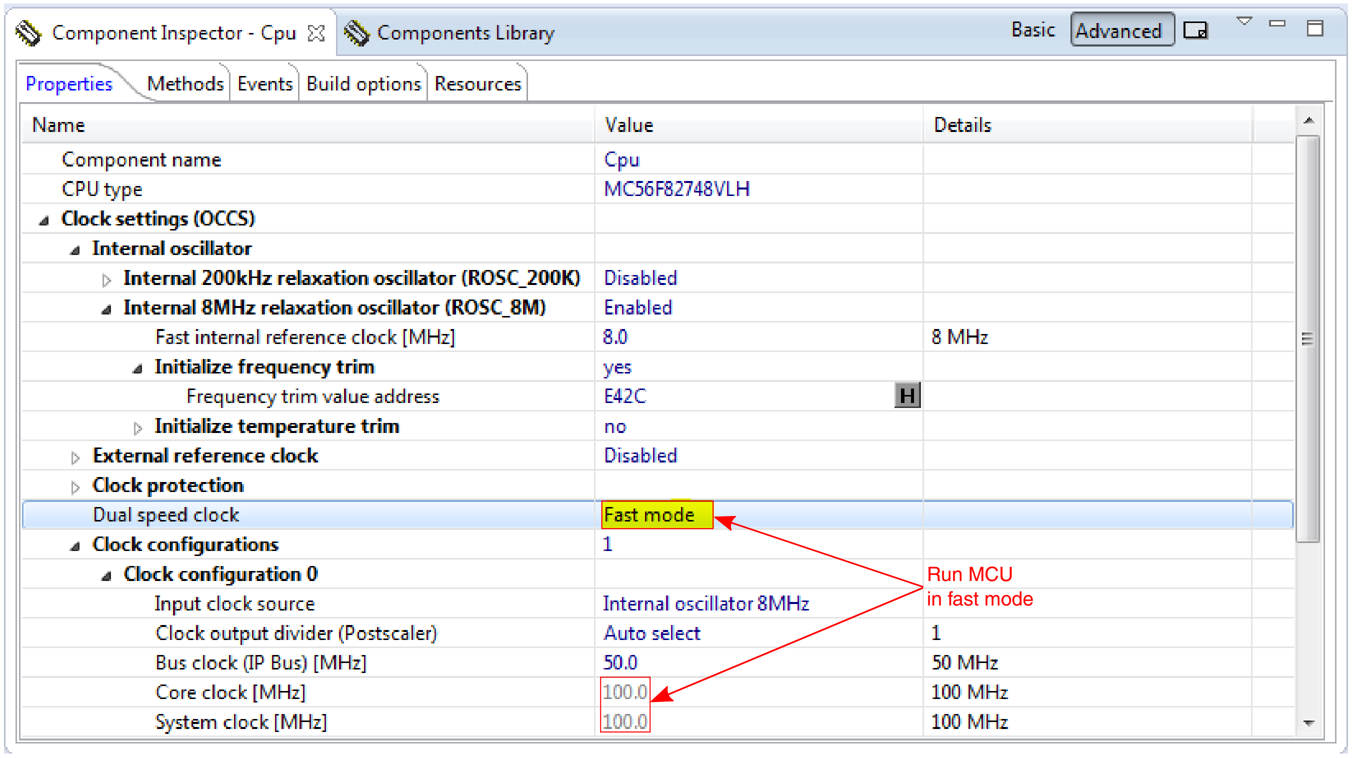


Figure 2. DAC DMA interface block diagram

4.1 CPU configuration

The MC56F82XX series of DSC supports two speed modes. Here, the CPU is configured in fast mode so that core and system clocks will be 100 MHz.

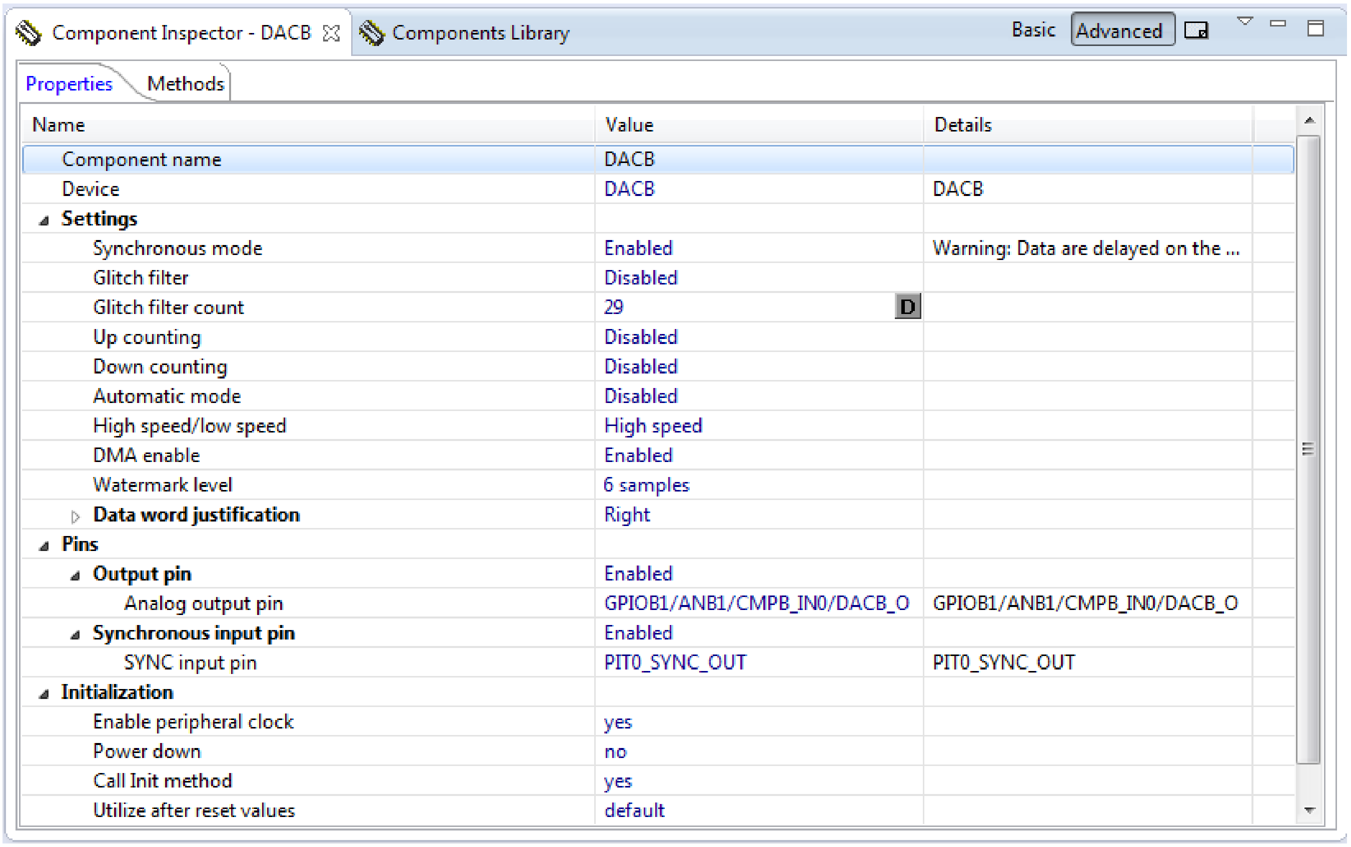


Name	Value	Details
Component name	Cpu	
CPU type	MC56F82748VLH	
Clock settings (OCCS)		
Internal oscillator		
Internal 200kHz relaxation oscillator (ROSC_200K)	Disabled	
Internal 8MHz relaxation oscillator (ROSC_8M)	Enabled	
Fast internal reference clock [MHz]	8.0	8 MHz
Initialize frequency trim	yes	
Frequency trim value address	E42C	
Initialize temperature trim	no	
External reference clock	Disabled	
Clock protection		
Dual speed clock	Fast mode	
Clock configurations	1	
Clock configuration 0		
Input clock source	Internal oscillator 8MHz	
Clock output divider (Postscaler)	Auto select	1
Bus clock (IP Bus) [MHz]	50.0	50 MHz
Core clock [MHz]	100.0	100 MHz
System clock [MHz]	100.0	100 MHz

Figure 3. CPU configuration

4.2 DAC configuration

DMA operations are enabled in the DAC with synchronous mode to maintain a controlled rise rate. The DAC is configured in high speed mode. A watermark level is defined. Sync source (such as PITx, TAx, etc.) through XBar is enabled.



Component Inspector - DACB | Components Library | Basic | Advanced

Properties | Methods

Name	Value	Details
Component name	DACB	
Device	DACB	DACB
Settings		
Synchronous mode	Enabled	Warning: Data are delayed on the ...
Glitch filter	Disabled	
Glitch filter count	29 D	
Up counting	Disabled	
Down counting	Disabled	
Automatic mode	Disabled	
High speed/low speed	High speed	
DMA enable	Enabled	
Watermark level	6 samples	
Data word justification	Right	
Pins		
Output pin	Enabled	
Analog output pin	GPIOB1/ANB1/CMPB_IN0/DACB_O	GPIOB1/ANB1/CMPB_IN0/DACB_O
Synchronous input pin	Enabled	
SYNC input pin	PIT0_SYNC_OUT	PIT0_SYNC_OUT
Initialization		
Enable peripheral clock	yes	
Power down	no	
Call Init method	yes	
Utilize after reset values	default	

Figure 4. DAC configuration

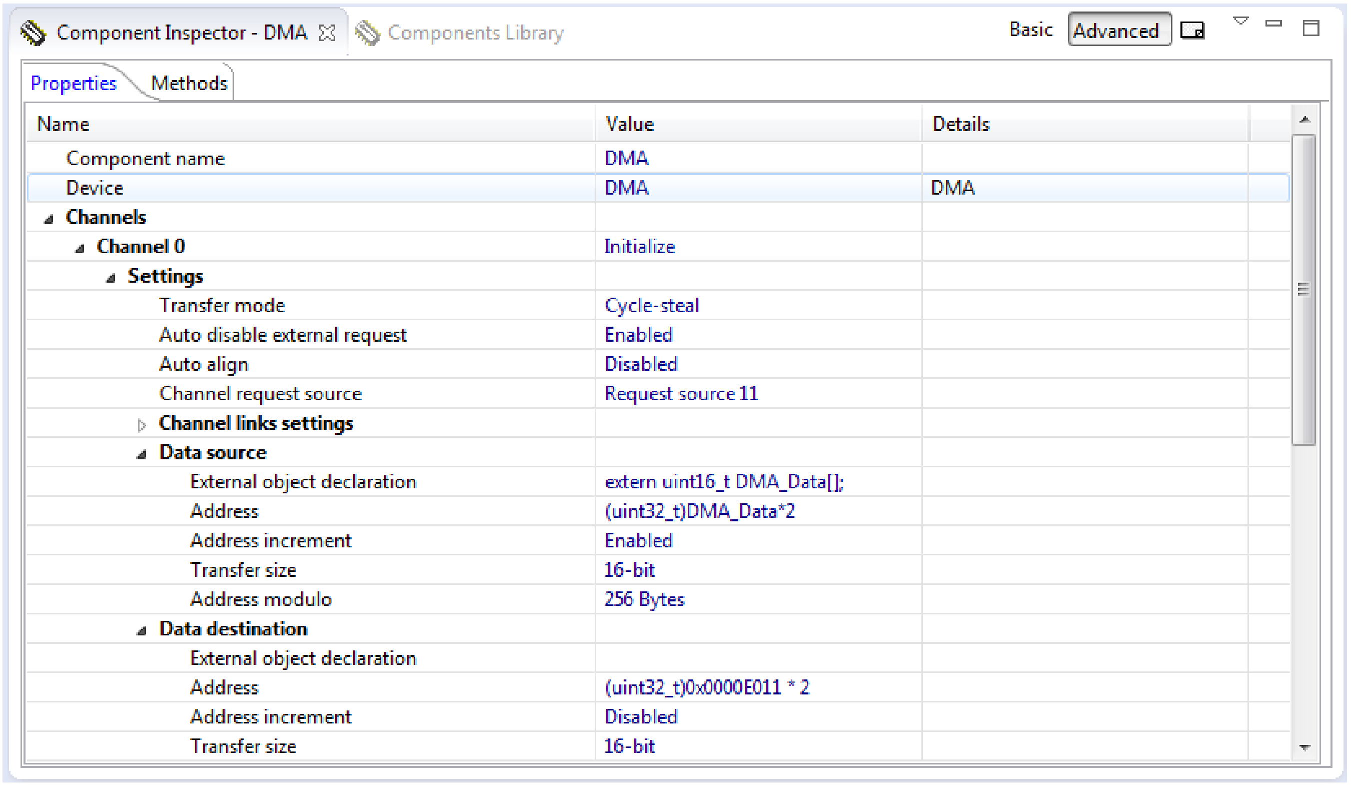
4.3 DMA configuration

Transfer mode “Cycle-steal” is set. External request and auto-disable external request features are enabled. A channel request source for DAC FIFO is assigned. Data source address (can be an array address where waveform is saved), size, other properties of source, data destination as DAC data register address, and other data destination properties are assigned.

Most peripheral modules on this chip are addressed in terms of 16-bit words. However, the DMA controller expects a byte address for every module. As a result, in the SAR*n* and DAR*n* registers, the user must always specify a byte address (which is double the value of a 16-bit word address).

The DMA Transfer Complete ISR should clear the interrupt flag by writing 1 to it. For example, DMA_DSR_BCR0 |= DMA_DSR_BCR_DONE_MASK.

Reinitialize the BCR value and source address (if cyclic buffer is not used) in the ISR and also enable a peripheral request.

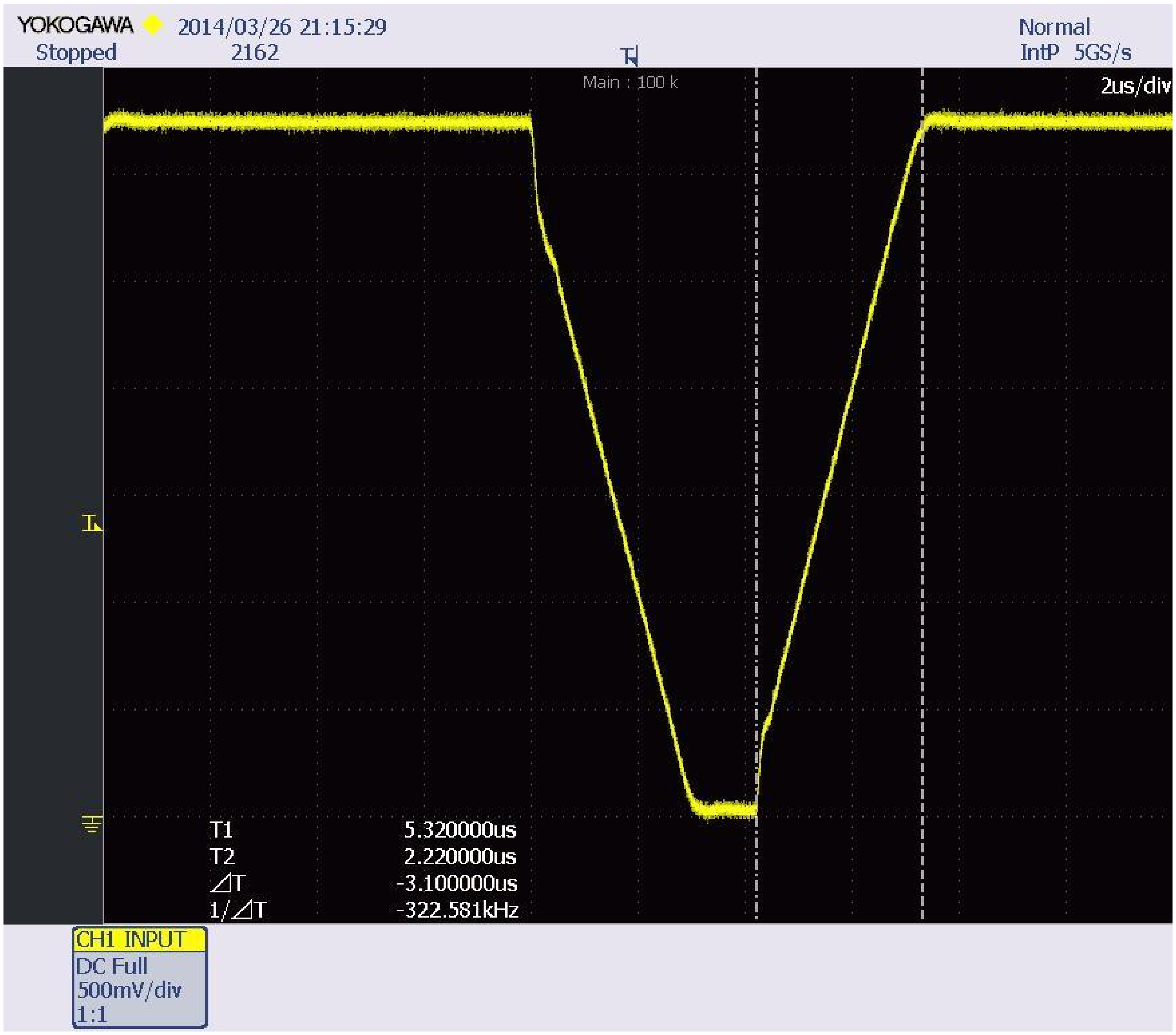


Name	Value	Details
Component name	DMA	
Device	DMA	DMA
Channels		
Channel 0	Initialize	
Settings		
Transfer mode	Cycle-steal	
Auto disable external request	Enabled	
Auto align	Disabled	
Channel request source	Request source 11	
Channel links settings		
Data source		
External object declaration	extern uint16_t DMA_Data[];	
Address	(uint32_t)DMA_Data*2	
Address increment	Enabled	
Transfer size	16-bit	
Address modulo	256 Bytes	
Data destination		
External object declaration		
Address	(uint32_t)0x0000E011 * 2	
Address increment	Disabled	
Transfer size	16-bit	

Figure 5. DMA configuration

5 Test setup and results

In the setup, PIT0 provides a sync signal to the DAC every 125 ns. Predefined steps for the waveform are stored in an array where 24 steps from 0x000 to 0xFFF are given. The watermark level is 6. Thus the expected rise time equals 125 ns × 24 = 3 µs. See Figure 6 for a graphic representation.



```
T1        5.320000us
T2        2.220000us
⊿T       -3.100000us
1/⊿T     -322.581kHz
```

Figure 6. Waveform output

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number AN4978
Revision 0, 08/2014