Interfacing the DAC8574 to the MSP430F449

Jojo Parguian *Data Acquisition Products*

ABSTRACT

This application report describes how to interface the DAC8574 digital-to-analog converter to the MSP430F449 mixed signal microcontroller.

Contents

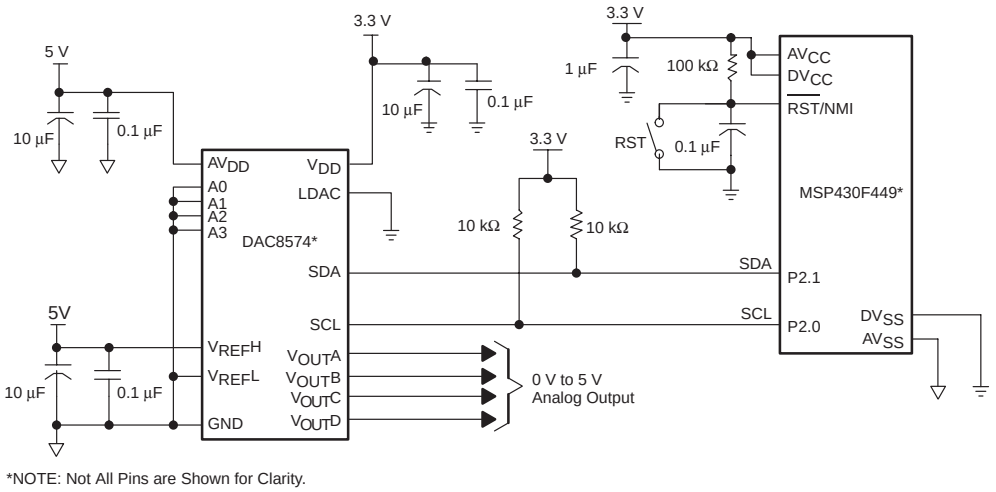
List of Figures

Trademarks are the property of their respective owners.

1 Introduction

The DAC8574 is a quad channel, low power, 16-bit voltage buffered output DAC which features an I2C two wire serial interface with triple buffering. The DAC8574 is serially interfaced with the MSP430F449 by bit-banging the selected GPIO pins to generate a serial clock and a data bit stream. The DAC8574 digital logic circuitry is powered by V_{DD} = 3.3 V to match the MSP430 logic voltage. The DAC8574 analog supply can range from a minimum of 2.7 V to a maximum of 5.5 V.

A simple diagram of the interface connection between the DAC8574 and the MSP430F449 is shown in Figure 1.



*NOTE: Not All Pins are Shown for Clarity.

Figure 1. Interface Circuit Diagram

2 Principle of Operation

The MSP430F449 µC drives the DAC8574 by simulating the I2C-bus protocol (also referred to as bit-banging) using selected GPIO pins of the MSP430 µC as shown in Figure 1. Hence, the MSP430 µC acts as the master and the DAC8574 works as a slave. The GPIO pin P2.0 is selected to generate the serial clock function (SCL), and the GPIO pin P2.1 is selected for the data transmission function (SDA). The LDAC pin of the DAC8574 is permanently tied to ground so that the software can control the LDAC function for DAC output updates. The address pins A0 to A3 are also tied to ground for simplicity.

Figures 2 and 3 show the timing diagrams for an address write, configuration write, and data write for interfacing the DAC8574 to the host processor. The address and configuration (control byte) writes need only be performed once and can be followed by consecutive data writes. Refer to the data sheet for more information on the I2C interface protocol of this device. For more information on the I2C-bus specification, refer to Philips Semiconductor.

An I2C transmission is initiated by generating a start condition as shown in Figure 2. The valid address shown in Figure 2 should be sent first to the DAC8574 to begin the communication. If the correct address is decoded by the DAC8574, it acknowledges by pulling the SDA line low on the 9th SCL pulse. A stop condition should not be initiated at this point and communication can be continued by sending the configuration (control byte) word to set the operational mode of the DAC8574. As soon as the acknowledge bit is again generated by the DAC8574, the MSB byte (data high byte) can be sent out. The DAC8574 acknowledges once more and the LSB byte (data low byte) should be sent. The DAC8574 acknowledges again, and in addition to this, the negative edge of the acknowledge pulse updates the DAC outputs. If a stop condition is performed, this concludes a single DAC update sequence.

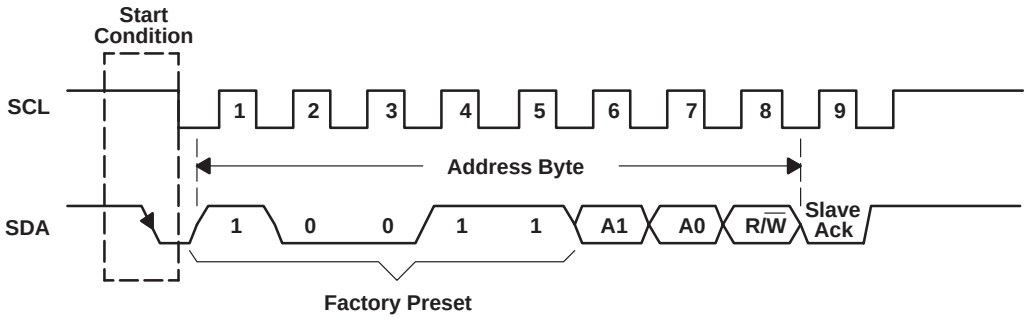


Figure 2. DAC8574 Address Write

If more than one data is to be written to the DAC8574, a stop condition does not need to occur. The MSB byte and LSB byte writes can be performed, as described above, consecutively until all of the data is written to the DAC8574. Also, the control byte does not have to be written to the DAC8574 again once it has been written to it unless a stop condition is performed or there is a requirement to change the operational mode of the DAC8574 (e.g., multichannel operation).

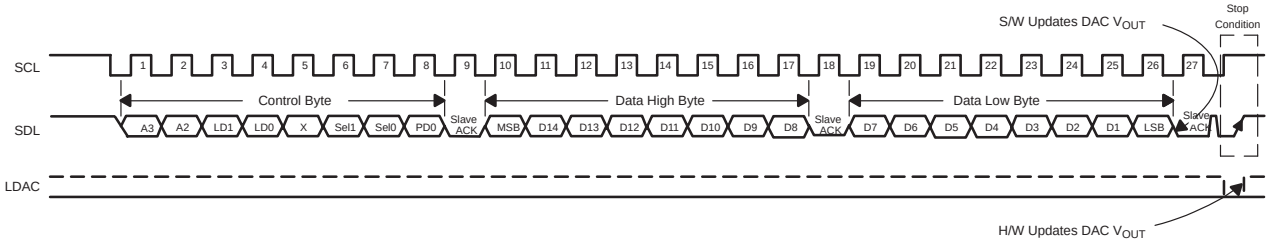


Figure 3. DAC8574 Configuration and Data Writes

3 The I2C-Bus Interface

The I2C-bus interface is a two-wire serial interface consisting only of data (SDA) and clock (SCL) lines, which are open-drain structures. A pull-up resistor is required for the SDA line and the SCL line in order for the bus to work properly. The size of the pull-up resistors used for the open-drain I2C-bus affects the signal integrity of the SDA and SCL lines depending on the bus operating speed and the bus capacitance. For a lower bus operating speed (i.e., 100 kHz), a higher pull-up resistor (such as 10 kΩ) can be used since the bus signal transition times are normally slow anyway. For the fast-speed mode, a 4.7-kΩ or 3-kΩ pull-up resistor is suggested, and for the high-speed mode, a 1 kΩ pull-up resistor is typical. These values are suggested for the pull-up resistors and should always be tested in the system to ensure proper I2C-bus operation.

An I^2C read or write cycle consists of nine periods of the SCL clock but only 8-bits of data are being read back or written to. The 9th cycle of the SCL clock is used for slave device acknowledgement. Hence, the master device should release the SDA line so that the slave device can pull the line low to generate its acknowledge signal. The master device must still generate the 9th SCL pulse to detect the slave device acknowledge pulse. Assuming the address and control bytes have already been written to the DAC8574, it takes eighteen SCL cycles for a DAC update since the MSB byte and the LSB byte along with their respective acknowledge signals are required. So, the maximum update rate of the DAC8574 is 5.3 KSPS when operating in standard-speed (100 kHz) mode. The maximum update rate is 18.2 KSPS for fast-speed (400 kHz) mode and 65.4 KSPS for high-speed (3.4 MHz) mode. These rates include the settling time of the DAC8574.

4 The LDAC Function

Although the DAC8574 offers a hardware and software simultaneous DAC output update function, this application report only utilizes the software function as mentioned earlier. The software simultaneous update function is controlled through the Load 1 (LD1) and Load 0 (LD0) control bits. By setting LD1 equal to 1 and LD0 equal to 0 all of the DAC registers are updated on the falling edge of the acknowledge pulse following the LSB byte transmission. To update only the specific DAC output that is addressed by Sel0 and Sel1, LD0 must be set to 1 and LD1 must be set to 0.

5 Multiple-Channel Operation

The address pins A0 and A1 are used to set the device I^2C address while the address pins A2 and A3 are used as an address extender so that a maximum of sixteen DAC8574s can be connected to a single I^2C serial bus. Basically there are only four different I^2C addresses in the I^2C bus, so up to four DAC8574s with the same I^2C address can be connected to the same bus. When an I^2C address is issued by the master device, all four DAC8574s with the same address acknowledge at the same time. However, the corresponding control bits DB23 (A3) and DB22 (A2) in the control byte are used to specifically address the intended DAC8574 out of the sixteen DACs that are connected to the I^2C bus.

The select bits, DB18 (Sel1) and DB17 (Sel0), in the control byte are also used to select a specific DAC channel to address. Therefore, the combination of the address pins A0, A1, A2, and A3 and the select bits Sel0 and Sel1, provides the 6-bit address decoding of the DAC8574 to allow up to 64 channels to be operated on the same I^2C bus. The flow of address decoding using the 6-bit combination mentioned above is shown in Figure 4.

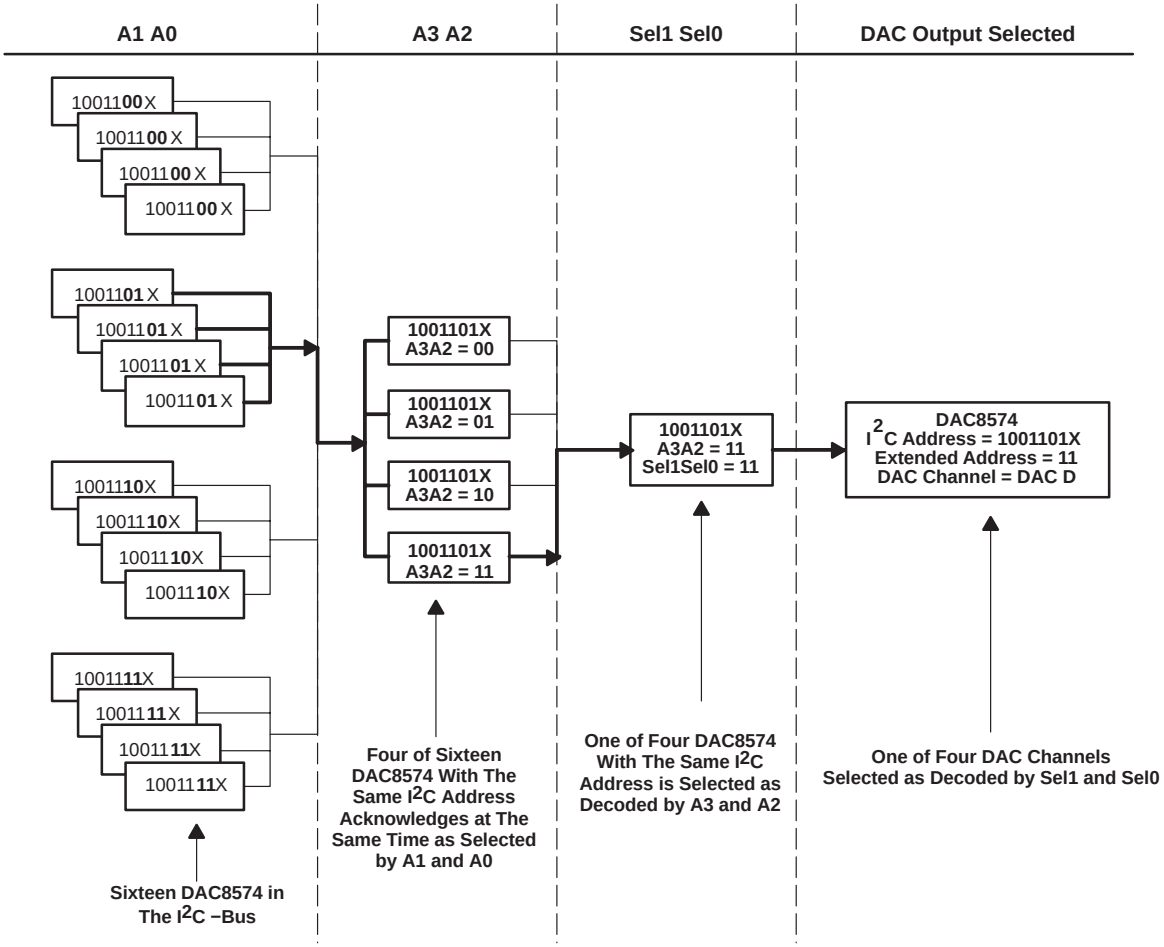


Figure 4. Address Decoding Stages of the 6-Bit (A3, A2, A1, A0, Sel1, and Sel0) Address

If there is a need to address all of the DAC8574s on the I2C-bus for synchronous update or power down mode, broadcast addressing can be used by simply issuing the broadcast address byte (0x90) instead of the I2C address byte. In broadcast mode, where LD1 = LD0 = 1, all of the DAC8574 devices connected to the I2C serial bus respond regardless of the address matching.

6 Generating a Sinewave Output

The actual timing diagram of the 2-wire I2C serial interface is shown in Figure 5. Channel 3 shows the SCL running at approximately 350 Kbps (fast mode) while channel 4 shows the SDA transmitting the address bytes followed by a stop condition. The stop condition is performed just to show how the DAC8574 responds to this condition; hence a repeated start is needed to continue communicating with the DAC8574. The I2C address of the DAC should be sent again when a restart is performed, which is shown in Figure 6.

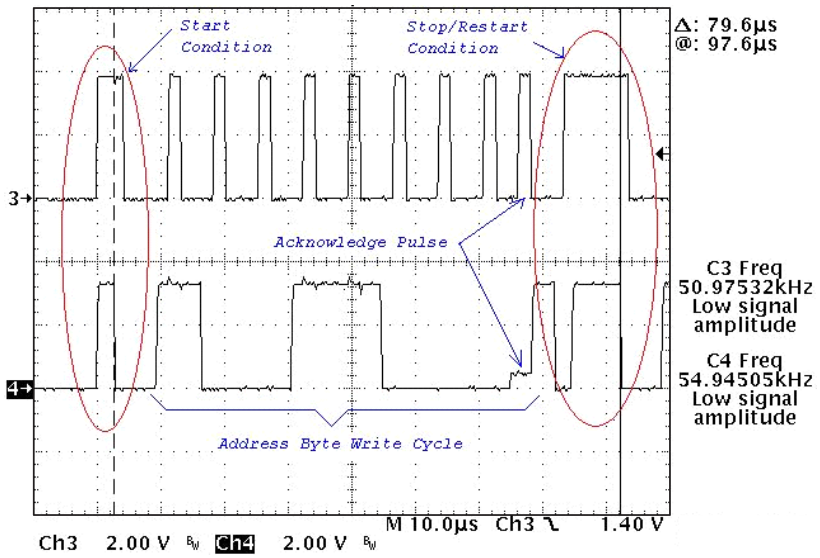


Figure 5. Actual Timing of the 2-Wire I²C Serial Interface Showing Start and Stop Conditions

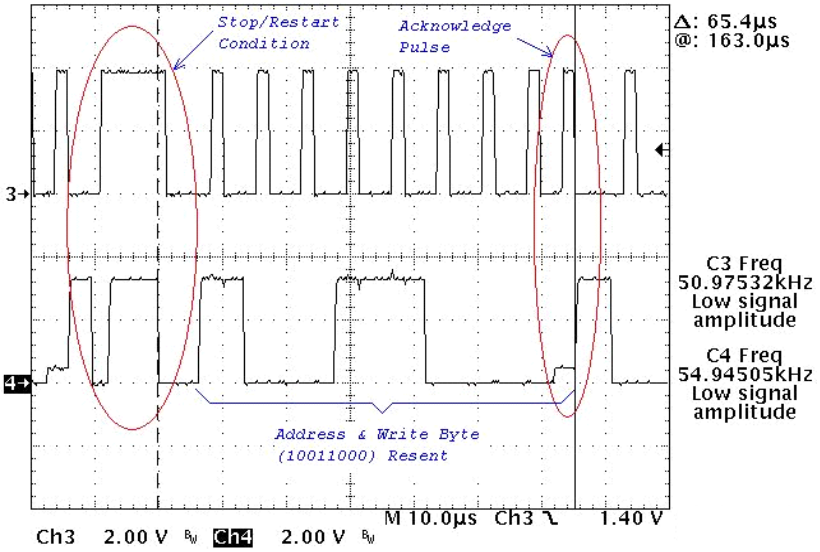


Figure 6. Repeated Start Timing Diagram

The DAC8574 must be configured first to the desired mode of operation by sending the control byte following the address and R/$\overline{W}$ byte cycle. Figure 7 shows the control byte cycle to configure the DAC8574 to operate in broadcast mode so that all DAC channels are updated with the MSB and LSB data bytes.

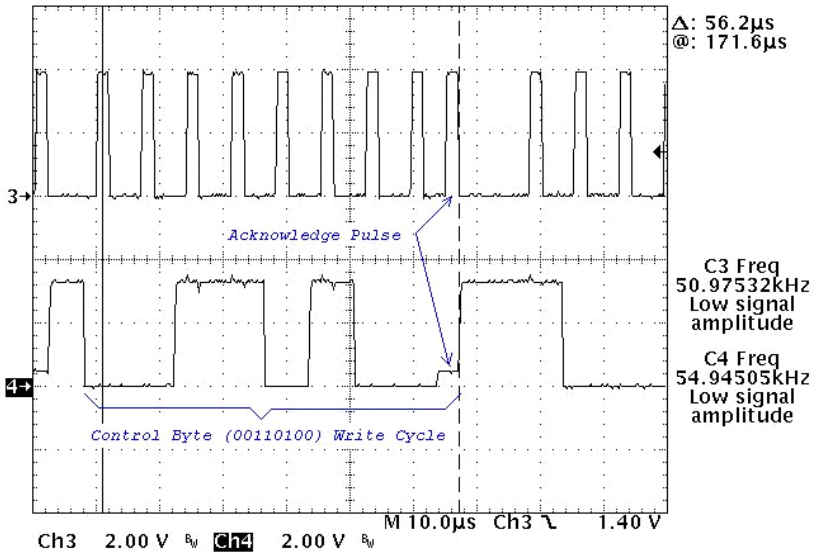


Figure 7. Control Byte (0x34) Write Cycle Timing Diagram

After the DAC8574 is configured, the MSB and LSB data bytes can be transmitted in succession and repeated continuously to generate the sinusoidal waveform. Figure 8 shows the MSB byte (0x80) of the first data from the table (Sin_tab) being sent first followed by the LSB byte (0x00) as shown in Figure 9. The combination of Figure 8 and Figure 9 makes up the 16-bit word format of the DAC8574 data format. All succeeding data are split in two and transmitted as described above. As a result of the timing diagrams in Figure 8 and Figure 9, the DAC output shown in Figure 10 displays a 256-step sinusoidal waveform. All DAC channels are active in this case as the DAC8574 was configured for broadcast mode. Therefore, all DAC channels display the same output signal.

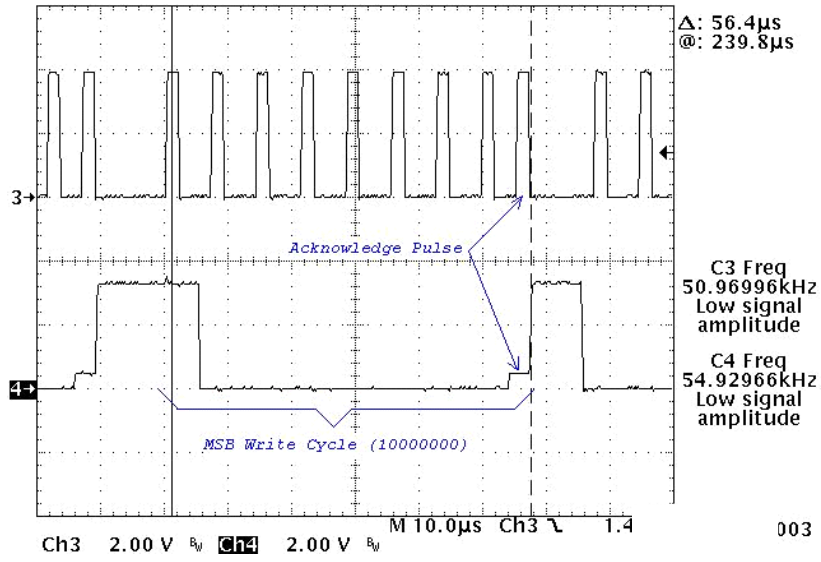


Figure 8. MSB Byte (0x80) Write Cycle Timing Diagram

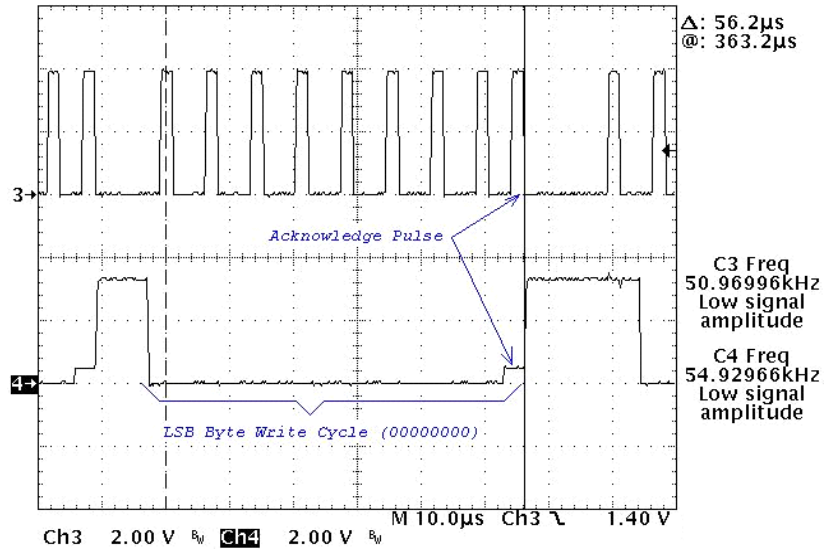


Figure 9. LSB Byte (0x00) Write Cycle Timing Diagram

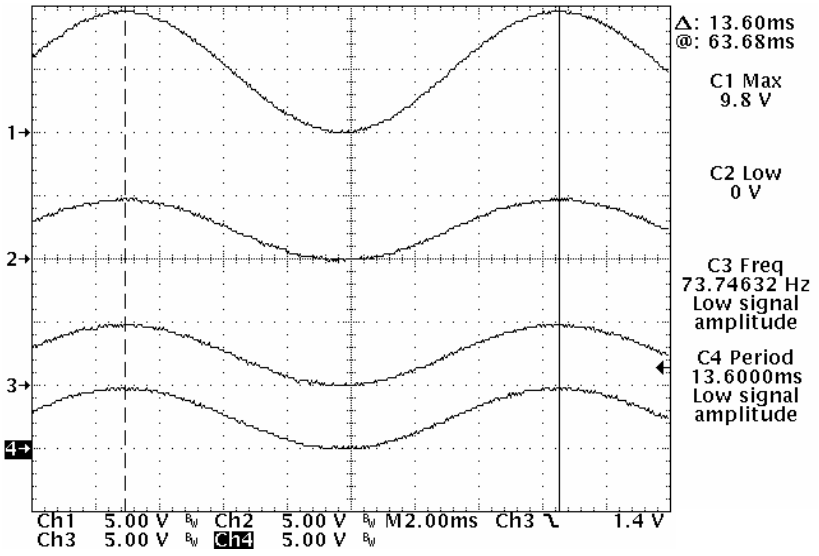


Figure 10. 4-Channel Output of the DAC8574

As shown in Figure 10, the DAC outputs channel 1 with an amplitude of 10 V_{p-p} while the rest of the channels are 5 V_{p-p}. The signal amplitude of channel 1 is 10 V_{p-p} because the DAC A channel output of the DAC8574 is connected to an external output amplifier with a gain of two as shown in Figure 11.

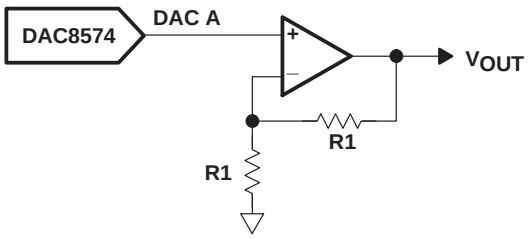


Figure 11. DAC A Output With a Gain of 2 (1 of 4 DACs)

This application report shows the ease of use of the DAC8574 using the I^2C-bus serial interface, which was simulated by the MSP430F449 host microcontroller, to perform a simple routine of signal generation. For more detailed information regarding the DAC8574, refer to the data sheet, SBAS377A. You can also contact TI's Data Acquisition Product group for further support by sending an e−mail to **dataconvapps@list.ti.com**.

7 References

1. *DAC8574 16-Bit, Voltage Output, Serial Input DAC* data sheet (SBAS377A)
2. *MSP430F449* data sheet (SLAS344C)
3. *MSP430X4XX Family User's Guide Manual* (SLAU056C)

8 MSP430F449 Software Code

```
;*******************************************************************************
;   MSP430F449 Demo - Serial communication with the DAC8574 (I2C)
;
;   Description:  Example code for interfacing the DAC8574 to the MSP430F449. This
;   code only assumes that the slave device acknowledges properly and keeps sending
;   data continuously without checking for a valid acknowledge response. This code
;   can always be modified to include that feature if necessary.
;
;   Jojo Parguian
;   Texas Instruments, Inc.
;   Data Acquisition Product
;   November 19, 2003
;*******************************************************************************
;     CPU registers used
#define     RXTXI2C     R7    ; I2C Receive/Transmit Register
#define     ADDRI2C     R8    ; I2C Address Register
#define     DATAI2C     R9    ; I2C Data Register
#define     BITI2C      R10   ; I2C Bit Counter Register
;
;     Definitions for I2C bus
DACADDR     equ   098h        ; DAC8574 HW Address (A0=A1=0+WR)
DACCTRL     equ   034h        ; DAC Control Word (Broadcast to all DAC Channels)
SCL         equ   001h        ; P2.0 controls SCL line (pull-up)
SDA         equ   002h        ; P2.1 controls SDA line (pull-up)
TRIG        equ   001h        ; Trigger point
;*******************************************************************************
#include    "msp430x44x.h"
;-------------------------------------------------------------------------------
; 16-bit Sine Lookup table with 256 steps
;-------------------------------------------------------------------------------
       ORG  1000h
;-------------------------------------------------------------------------------
Sin_tab  DW   32768,33572,34376,35178,35980,36779,37576,38370,39161,39947,40730,41507,42280
         DW   43046,43807,44561,45307,46047,46778,47500,48214,48919,49614,50298,50972,51636
         DW   52287,52927,53555,54171,54773,55362,55938,56499,57047,57579,58097,58600,59087
         DW   59558,60013,60451,60873,61278,61666,62036,62389,62724,63041,63339,63620,63881
         DW   64124,64348,64553,64739,64905,65053,65180,65289,65377,65446,65496,65525,65535
         DW   65525,65496,65446,65377,65289,65180,65053,64905,64739,64553,64348,64124,63881
         DW   63620,63339,63041,62724,62389,62036,61666,61278,60873,60451,60013,59558,59087
         DW   58600,58097,57579,57047,56499,55938,55362,54773,54171,53555,52927,52287,51636
         DW   50972,50298,49614,48919,48214,47500,46778,46047,45307,44561,43807,43046,42280
         DW   41507,40730,39947,39161,38370,37576,36779,35980,35178,34376,33572,32768,31964
         DW   31160,30358,29556,28757,27960,27166,26375,25589,24806,24029,23256,22490,21729
         DW   20975,20229,19489,18758,18036,17322,16617,15922,15238,14564,13900,13249,12609
         DW   11981,11365,10763,10174,9598,9037,8489,7957,7439,6936,6449,5978,5523,5085,4663
         DW   4258,3870,3500,3147,2812,2495,2197,1916,1655,1412,1188,983,797,631,483,356,247
         DW   159,90,40,11,1,11,40,90,159,247,356,483,631,797,983,1188,1412,1655,1916,2197
         DW   2495,2812,3147,3500,3870,4258,4663,5085,5523,5978,6449,6936,7439,7957,8489,9037
         DW   9598,10174,10763,11365,11981,12609,13249,13900,14564,15238,15922,16617,17322
         DW   18036,18758,19489,20229,20975,21729,22490,23256,24029,24806,25589,26375,27166
         DW   27960,28757,29556,30358,31160,31964
```

```
;-------------------------------------------------------------------------------
      ORG   0F000h
;-------------------------------------------------------------------------------
RESET       mov.w #300h,SP                ; Initialize stack-pointer
            call        #Init_Sys         ; Initialize system
            clr.w       R6
One_Shot
;           bis.b       #TRIG,&P4OUT      ; Toggle Trigger
;           bic.b       #TRIG,&P4OUT
            mov.b       #DACADDR,RXTXI2C  ; Load DAC8574 Address
            call        #I2C_Start        ; Generate Start Condition
            call        #I2C_TX
            call        #I2C_Stop         ; Generate Stop Condition
            mov.b       #DACADDR,RXTXI2C  ; Reload DAC8574 Address
            call        #I2C_Start        ; Repeat Start
            call        #I2C_TX
            mov.b       #DACCTRL,RXTXI2C  ; Load DAC8574 Control Byte
            call        #I2C_TX
Write_Data
            mov.w       #0FFh,R6
            mov.w       #0,R5
Again       mov.w       Sin_tab(R5),DATAI2C
            swpb        DATAI2C           ; MSB first
            call        #Write_I2C
            swpb        DATAI2C           ; LSB next
            call        #Write_I2C
            incd.w      R5
            sub.w       #1,R6
            and.w       #0FFh,R6
            jnz         Again
            jmp         Write_Data
;-------------------------------------------------------------------------------
Init_Sys;   Subroutine sets up Modules and Control Registers
;-------------------------------------------------------------------------------
            mov.w       #WDTPW+WDTHOLD,&WDTCTL  ; Stop WDT
SetupFLL2   bis.b       #FN_4,&SCFI0      ; x2 DCO, 8MHz nominal DCO
            bis.b       #DCOPLUS+XCAP14PF,&FLL_CTL0   ; DCO+, configure load caps
            mov.b       #121,&SCFQCTL     ;(121+1) x 2 x 32768 = 7.99 MHz
SetupPorts  mov.b       #0FFh,&P1DIR      ; Set port to outputs
            clr.b       &P1OUT            ; P1OUTs = 0
            mov.b       #0FFh,&P2DIR      ; Set port to outputs
            clr.b       &P2OUT            ; P2OUTs = 0
            mov.b       #0FFh,&P3DIR      ; Set port to outputs
            clr.b       &P3OUT            ; P3OUTs = 0
            mov.b       #0FFh,&P4DIR      ; Set port to outputs
            clr.b       &P4OUT            ; P4OUTs = 0
            mov.b       #0FFh,&P5DIR      ; Set port to outputs
            clr.b       &P5OUT            ; P5OUTs = 0
            mov.b       #0FFh,&P6DIR      ; Set port to outputs
            clr.b       &P6OUT            ; P6OUTs = 0
            ret                           ; Return to Main
```

```
;-------------------------------------------------------------------------------
Write_I2C
;-------------------------------------------------------------------------------
            mov.b       DATAI2C,RXTXI2C   ; Load Out-Going Data
            call        #I2C_TX           ; Send Data and Acknowledge
            ret                           ; Return from subroutine
;-------------------------------------------------------------------------------
I2C_Start;  enter SDA=1, SCL=x
;           exit  SDA=1, SCL=0
;-------------------------------------------------------------------------------
            bic.b       #SCL+SDA,&P2DIR   ; SCL and SDA to input direction
            bic.b       #SCL+SDA,&P2OUT   ; SCL=1, SDA=1
            bis.b       #SDA,&P2DIR       ; SDA=0
            bis.b       #SCL,&P2DIR       ; SCL=0
            ret
;-------------------------------------------------------------------------------
I2C_TX;     enter SDA=x, SCL=0
;           exit  SDA=1, SCL=0
;-------------------------------------------------------------------------------
            mov         #08,BITI2C        ; number of bits to transfer
I2C_TX_Bit  rla.b       RXTXI2C           ; data bit -> carry
            jc          I2C_TX1           ; test carry for 1 or 0
I2C_TX0     bis.b       #SDA,&P2DIR       ; SDA=0
            jmp         I2C_TXx           ; Toggle SCL
I2C_TX1     bic.b       #SDA,&P2DIR       ; SDA=1
            jmp         I2C_TXx
I2C_TXx     bic.b       #SCL,&P2DIR       ; SCL=1
            nop                           ; delay to meet I2C spec
            nop                           ;
            bis.b       #SCL,&P2DIR       ; SCL=0
            dec         BITI2C            ; all bits read?
            jnz         I2C_TX_Bit        ; continue until 8 bits are sent
            bic.b       #SDA,&P2DIR       ; SDA=1, release SDA line for acknowledge
                                          ;
TX_Ackn     bic.b       #SCL,&P2DIR       ; SCL=1
            nop                           ; delay to meet I2C spec
            nop                           ;
            bis.b       #SCL,&P2DIR       ; SCL=0
            ret                           ; Return from subroutine
;-------------------------------------------------------------------------------
I2C_Stop;   enter SDA=x, SCL=0
;           exit  SDA=1, SCL=1
;-------------------------------------------------------------------------------
            bis.b       #SDA,&P2DIR       ; SDA = 0
            bic.b       #SCL,&P2DIR       ; SCL = 1
            bic.b       #SDA,&P2DIR       ; SDA = 1
I2C_End     ret                           ; Return from subroutine
;-------------------------------------------------------------------------------
;           Interrupt Vectors Used MSP430F449
;-------------------------------------------------------------------------------
            ORG         0FFFEh            ; MSP430 RESET Vector
            DW          RESET             ;
            END                           ;
```

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265

Copyright © 2005, Texas Instruments Incorporated