TSW3100 High Speed Digital Pattern Generator

Contents

List of Figures

List of Tables

1 Hardware Configuration

The TSW3100 EVM can be set up in a variety of configurations to accommodate a specific mode of operation. Before starting the evaluation, you should decide on the configuration and make the appropriate connections or changes. The demonstration board comes with this factory-set configuration:

- Board set to the Ethernet IP 192.168.1.123 address. This address is controlled be switch SW2, using DIP0 and DIP1 switches. (See Figure 18 and Table 7.)
- SW2 switch DIP2 set to *OPEN*. This switch is not currently used.
- SW2 switches DIP3-DIP7 set to *OPEN*. These switches are used to set the sync delay when operating two TSW3100 boards in the Master/Slave mode.
- FPGA Input Clock select jumper J50 jumper installed between pins 2-3. This directs the Field Programmable Gate Array (FPGA) to use the on-board 100 MHz oscillator. For external CLK operation, set the jumper to pins 1-2 and provide a CMOS level clock source to connector J41 (FPGA INPUT CLK).

1.1 Power Input Source

Use the provided 5V–6V AC-DC switching power supply to apply power to the TSW3100. The incoming power is regulated down to 0.9 V, 1.2 V, 1.8 V, 2.5 V, and 3.3 V with on-board LDO regulators to generate the necessary voltages. The input power to these regulators is controlled by SW1.

1.2 Output Power Regulators

The TSW3100 provides two output power sources with these default settings:

- +3.3 V @ 1 A at J10 and the return at J38.
- +1.8 V @ 1 A at J7 and the return to J39.

Both power supplies are derived using low noise LDO regulators and controlled by switch SW5. This switch is independent of the operation of the main board power switch SW1. Both LDOs are adjustable regulators and can be modified by changing one resistor. To change the output voltage of the +1.8 V supply, replace R27 with the appropriate value. To change the output voltage of the +3.3 V supply, replace R31 with the appropriate value. See the TI TPS76701 data sheet (SGLS157) for more information regarding these devices.

1.3 Switches and LEDs

The TSW3100 provides an eight-position DIP switch and four push-button switches you can use during the EVM operation. Table 1 describes the DIP switch functionality.

Table 1. Push-button and DIP Switch Functions

Reference Designator	Switch Name	Description
S3	SYNC	Sends a one-time SYNC pulse at the start of the test pattern.
S7	START/STOP	Stops a test pattern that is running. When pressed again, starts the test pattern.
S8	SPARE	Not used
S9	FPGA CONFIG	Reconfigures the FPGA when pressed.
SW2	DIP0	Sets the Board Ethernet IP address(1)
SW2	DIP1	Sets the Board Ethernet IP address(1)
SW2	DIP2	Adjust SYNC when in CMOS mode (Master/Slave operation only)
SW2	DIP3-DIP7	Adjust SYNC when in LVDS mode (Master/Slave operation only)

(1) See Table 7 to set the TSW3100 board IP address using these switches.

Ten LEDs display the TSW3100 EVM status during its operation. Table 2 describes the meaning of each LED status.

Table 2. LED Status Descriptions

Reference Designator	LED Name	Description(1)
D13	PAT GEN IDLE	When power is applied, this LED should light, indicating the board is ready to load test pattern information.
D14	PAT GEN CLK	When pattern generator starts, this LED lights, if the required clock is present. LED is OFF during idle mode.
D15	PAT GEN RUN	When the pattern generator starts, this LED lights. LED is OFF during idle mode.
D16	FIFO EMPTY ERROR	ON—error when loading the internal FIFO of the FPGA.
D17	FIFO FULL ERROR	ON—error when unloading the internal FIFO of the FPGA.
D18	LVDS PLL LOCK	ON—indicates feedback LVDS clock present on J74. Should always be ON when using LVDS outputs with an EVM plugged into J74.

(1) See Table 8 and Table 9 for LED patterns during TSW3100 operations.

Table 2. LED Status Descriptions (continued)

Reference Designator	LED Name	Description[1]
D19	DDR2 PLL LOCK	ON—indicates the presence of the FPGA clock used for the DDR2 interface. Should always be ON.
D20	NIOS PLL LOCK	ON—indicates the FPGA clock is locked to the input clock. Should always be ON.
D21	CMOS MODE	When pattern generator starts, this LED lights when the EVM is set for CMOS output mode. This LED is OFF during idle mode.
D22	LVDS MODE	When pattern generator starts, this LED lights when the EVM is set for LVDS output mode. This LED is OFF during idle mode.

1.4 Input and Output Connectors

Table 3 describes the input and output connectors.

Table 3. Input and Output Connectors

Reference Designator	Connector Type	Description
J9	Power Connector	5V-6V VDC input power from AC-to-DC power supply
J24	240 DIMM	DDR2 dual in-line memory module connector
J13	CONN MAGJACK	10/100 Ethernet Connector
J74	160 pin 0.5mm-pitch QSH-DP series Samtec High Speed Connector	LVDS output data connector
J63	40 pin male header connectors	Data Bus A CMOS output data
J64	40 pin male header connectors	Data Bus B CMOS output data
J55	10 pin male header	JTAG interface to FPGA and serial PROM
J44	10 pin male header	JTAG interface to FPGA and FLASH
J10	Banana Jack	+3.3 V out @ 1 A
J38	Banana Jack	+3.3 V Return
J7	Banana Jack	+1.8 V out @ 1 A
J39	Banana Jack	+1.8 V Return
J47	SMA	Sync Out (Master Mode only)
J48	SMA	Sync In. Used only in Slave Mode.
J73	SMA	CMOS CLK. Required when board is generating CMOS output data.
J45	SMA	CLK OUT. Spare output clock. Same clock used by the FPGA.
J41	SMA	FPGA INPUT CLK. Required when jumper J50 is set to external clock mode (1-2).
J49	SMA	Spare IO. Spare input or output if assigned to FPGA firmware. Default firmware does not assign this.

1.4.1 Output Data Connectors

The TSW3100 provides CMOS outputs to drive existing TI HSDAC EVMs. The CMOS outputs use two connectors which interface directly to the TI DAC5687 and DAC5688 EVMs when using the provided adapter board. Table 4 and Table 5 define the pinout of CMOS output connectors J63 and J64.

Table 4. CMOS Output Data Bus A Connector J63

Pin	Description	Pin	Description
1	CMOS Data Bit 15 (MSB)	21	CMOS Data Bit 5
2	GND	22	GND
3	CMOS Data Bit 14	23	CMOS Data Bit 4
4	GND	24	GND

Table 4. CMOS Output Data Bus A Connector J63 (continued)

Pin	Description	Pin	Description
5	CMOS Data Bit 13	25	CMOS Data Bit 3
6	GND	26	GND
7	CMOS Data Bit 12	27	CMOS Data Bit 2
8	GND	28	GND
9	CMOS Data Bit 11	29	CMOS Data Bit 1
10	GND	30	GND
11	CMOS Data Bit 10	31	CMOS Data Bit 0 (LSB)
12	GND	32	GND
13	CMOS Data Bit 9	33	Sync
14	GND	34	GND
15	CMOS Data Bit 8	35	Spare
16	GND	36	GND
17	CMOS Data Bit 7	37	Spare
18	GND	38	GND
19	CMOS Data Bit 6	39	Spare
20	GND	40	GND

Table 5. CMOS Output Data Bus B Connector J64

Pin	Description	Pin	Description
1	CMOS Data Bit 15 (MSB)	21	CMOS Data Bit 5
2	GND	22	GND
3	CMOS Data Bit 14	23	CMOS Data Bit 4
4	GND	24	GND
5	CMOS Data Bit 13	25	CMOS Data Bit 3
6	GND	26	GND
7	CMOS Data Bit 12	27	CMOS Data Bit 2
8	GND	28	GND
9	CMOS Data Bit 11	29	CMOS Data Bit 1
10	GND	30	GND
11	CMOS Data Bit 10	31	CMOS Data Bit 0 (LSB)
12	GND	32	GND
13	CMOS Data Bit 9	33	TXENABLE
14	GND	34	GND
15	CMOS Data Bit 8	35	Spare
16	GND	36	GND
17	CMOS Data Bit 7	37	Spare
18	GND	38	GND
19	CMOS Data Bit 6	39	Spare
20	GND	40	GND

The TSW3100 provides LVDS level outputs to drive existing TI HSDAC EVMs. The LVDS outputs use a high speed, 0.5 mm-pitch connector from Samtec, which interfaces directly to the TI DAC5682 EVM. Table 6 defines the pinout for the LVDS Output Connector J74.

Table 6. LVDS Output Connector J74

Pin	Description	Pin	Description
1	+1.8VD	21	
2	+1.8VD	22	
3	+1.8VD	23	
4	+1.8VD	24	
5		25	
6	GND	26	DSP3
7		27	
8	GND	28	DSP4
9		29	
10	GND	30	
11		31	
12	GND	32	
13	+3.3VD	33	
14	+3.3VD	34	DSP5
15	+3.3VD	35	
16	+3.3VD	36	DSP6
17	DSP7	37	
18	DSP1	38	
19	DSP8	39	
20	DSP2	40	
41		61	DA13N
42		62	DB13N
43		63	
44		64	
45		65	DA12P
46		66	DB12P
47	DA15P	67	DA12N
48	DB15P	68	DB12N
49	DA15N	69	
50	DB15N	70	
51		71	DA11P
52		72	DB11P
53	DA14P	73	DA11N
54	DB14P	74	DB11N
55	DA14N	75	
56	DB14N	76	
57		77	DA10P
58		78	DB10P
59	DA13P	79	DA10N
60	DB13P	80	DB10N
81		101	DA7P
82		102	DB7P
83	DA9P	103	DA7N
84	DB9P	104	DB7N
85	DA9N	105	
86	DB9N	106	

Table 6. LVDS Output Connector J74 (continued)

Pin	Description	Pin	Description
87		107	DA6P
88		108	DB6P
89	DA8P	109	DA6N
90	DB8P	110	DB6N
91	DA8N	111	
92	DB8N	112	
93		113	DA5P
94		114	DB5P
95	DCLKP	115	DA5N
96	FPGA_CLKP	116	DB5N
97	DCLKN	117	
98	FPGA_CLKN	118	
99		119	DA4P
100		120	DB4P
121	DA4N	141	
122	DB4N	142	
123		143	DA0P
124		144	DB0P
125	DA3P	145	DA0N
126	DB3P	146	DB0N
127	DA3N	147	
128	DB3N	148	
129		149	
130		150	DBCLKP
131	DA2P	151	
132	DB2P	152	DBCLKN
133	DA2N	153	
134	DB2N	154	
135		155	SYNCP
136		156	
137	DA1P	157	SYNCN
138	DB1P	158	
139	DA1N	159	
140	DB1N	160	
161	GND	167	GND
162	GND	168	GND
163	GND	169	GND
164	GND	170	GND
165	GND	171	GND
166	GND	172	GND

1.4.2 JTAG Connectors

Two JTAG headers (10-pin key shrouded header J55 and J44) are provided for configuring the Stratix II™ FPGA and the FLASH memory device. The programming is done through using an Altera ByteBlaster II™ or USB-Blaster™cable. The board comes with operational firmware stored in a serial PROM device that loads the FPGA at power up. You do not need to download any firmware.

1.4.3 Ethernet Connector

The TSW3100 provides a 10/100 Ethernet interface for Ethernet connections up to 100 Mbps. The reference designator for this interface is J13.

2 Software Installation

TI provides several software tools to help you use the TSW3100 for evaluation of TI DACs. The user can follow the interface protocol discussed in Section 4.2.

2.1 USB to Ethernet Adapter Installation

The USB interface adapter is provided to allow an additional, dedicated PC IP address to connect to the fixed TSW3100 IP address. To install this adapter:

1. Connect the included USB to Ethernet adapter to a spare USB port of the host PC. The Windows *Found New Hardware Wizard* (Figure 1) displays. If this does not happen, ensure the cable is connected properly. Select the *No, not this time* option button and click *Next*.

Figure 1. Do not Use Windows Update to Find Adapter Software

2. Insert the USB to Ethernet Adapter installation CD. The installation should start automatically (Figure 2). When it starts, select the *Install the software automatically (Recommended)* option and click *Next*.

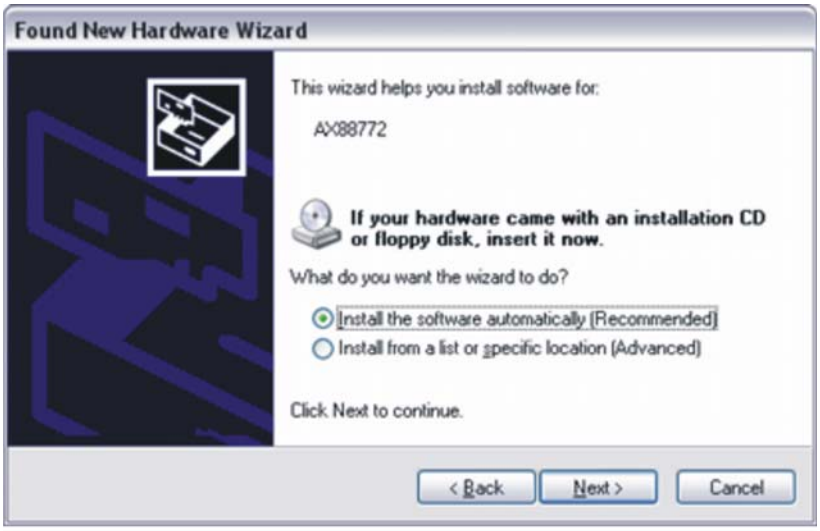

Figure 2. Install USB to Ethernet Adapter Software

3. Wait for the *Found New Hardware Wizard* to complete (Figure 3). Press *Finish*.

Figure 3. USB to Ethernet Adapter Software Installation Complete

4. Restart the host PC.

2.2 Configure the USB to Ethernet network

1. Select the Windows *Start* menu, select the *Control Panel*, and choose the *Network Connections* item.
2. Double-click the *Local Area Connection* whose device name is *ASIX AX88772 USB2.0 to Fast Ethernet Adapter*. The *Local Area Connection Properties* dialog (Figure 4) displays.

Figure 4. Configure USB to Ethernet Connection

3. Double-click the *Internet Protocol (TCP/IP)* item (Figure 4) found under the *General* dialog tab and listed in the *This Connection uses the following items* selection list.
4. Select the *Use the following IP address* option (Figure 5). Type 192.168.1.1 for the *IP address* and 255.255.255.0 for the *Subnet Mask*.

Figure 5. Specify IP Address and Subnet Mask

5. Click *OK* for both the Internet Protocol (TCP/IP) Properties and Local Area Connection Properties dialogs.

2.3 Installing the MATLAB Runtime Engine

This section helps you install the MATLAB Runtime engine which is used to run the provide MATLAB executable code.

1. Double-click on the *MCRInstaller.exe* file located on the TSW3100 installation CD. The Choose Setup Language (Figure 6) displays. Click *OK* for English (United States).

Figure 6. Choose Setup Language

2. When the *MATLAB Component Runtime 7.5* screen (Figure 7) displays, click *Next*.

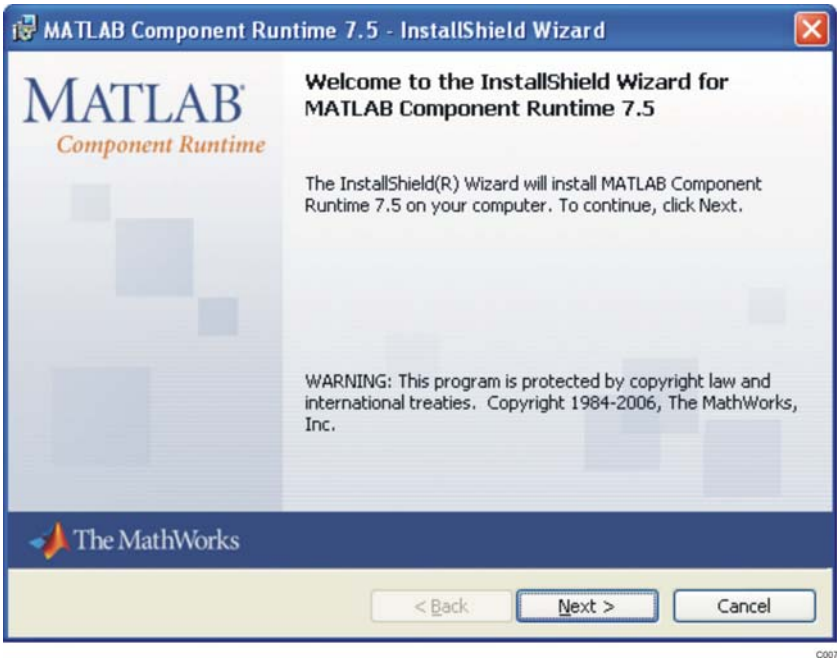

Figure 7. MATLAB Welcome Screen

3. For the *Customer Information* (Figure 8) screen, specify the *User Name*, *Organization*, select the desired user option button, and click *Next*.

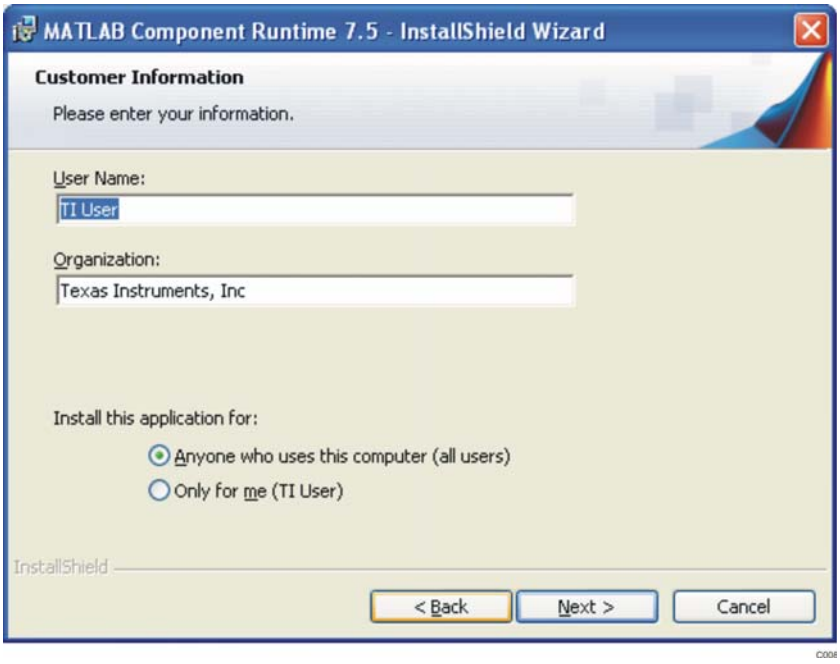

Figure 8. Customer Information

4. When the *Destination Folder* screen (Figure 9) displays, click *Next* to install the MATLAB software in the default directory.

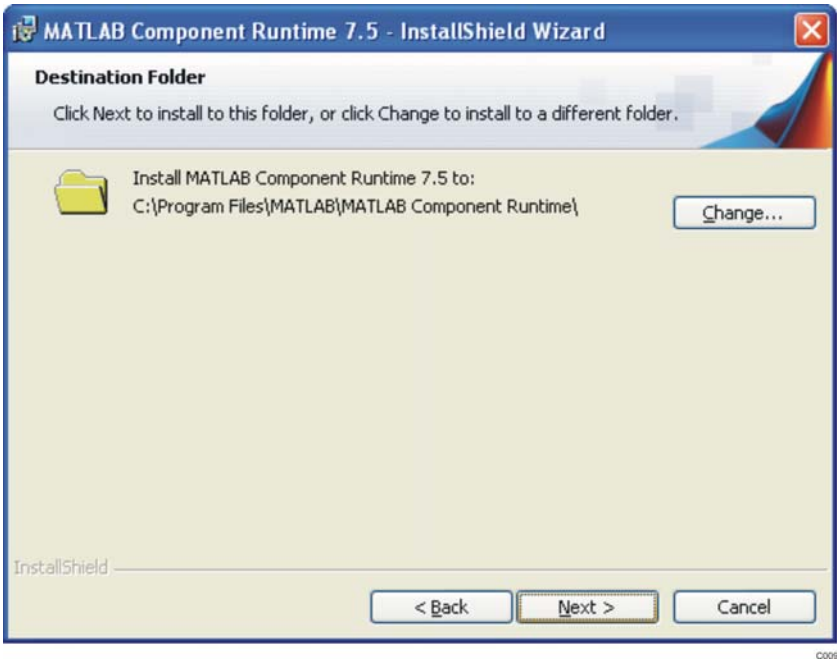

Figure 9. Destination Folder

5. When the *Ready to Install the Program* screen (Figure 10) displays, click *Install* to begin the installation. The installation lasts approximately five minutes.

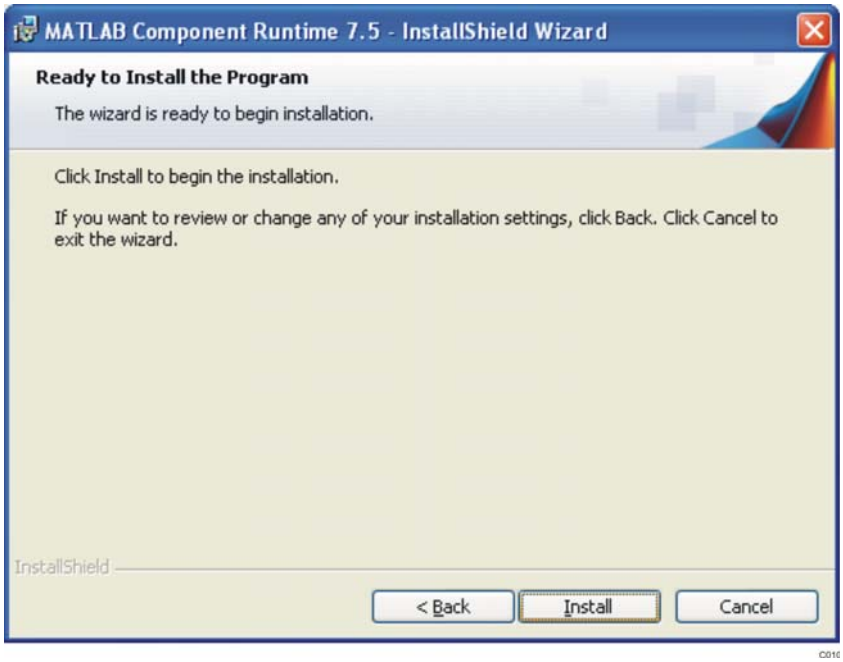

Figure 10. Ready to Install the Program

6. Click *Finish* once the *InstallShield Wizard Completed* screen (Figure 11) displays.

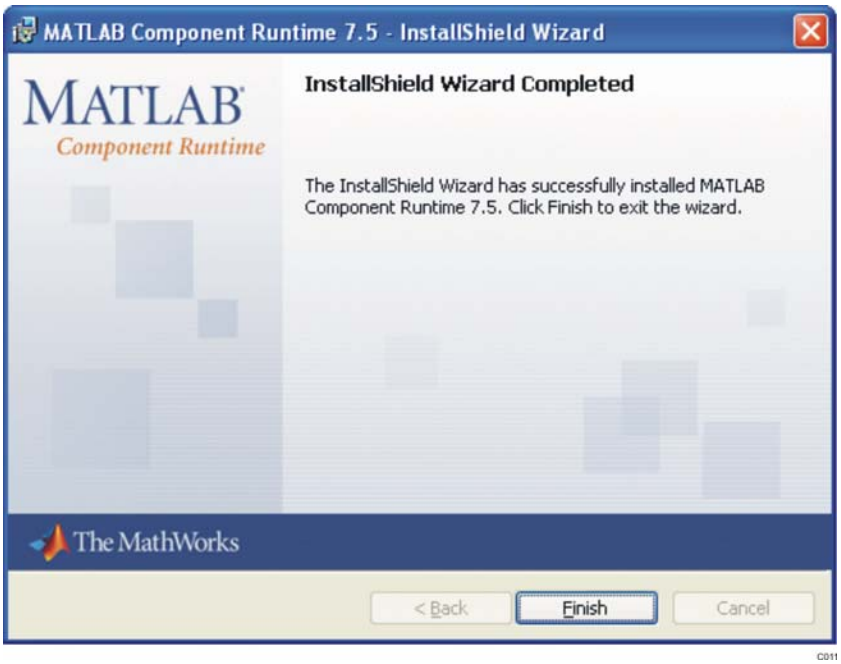

Figure 11. InstallShield Wizard Completed

2.4 Installing the TSW3100 Application Software

1. Double-click on the *TSW3100Installer.exe* file located on the TSW3100EVM installation CD. The TSW3100 Installation Wizard (Figure 12) displays. Click *Next*.

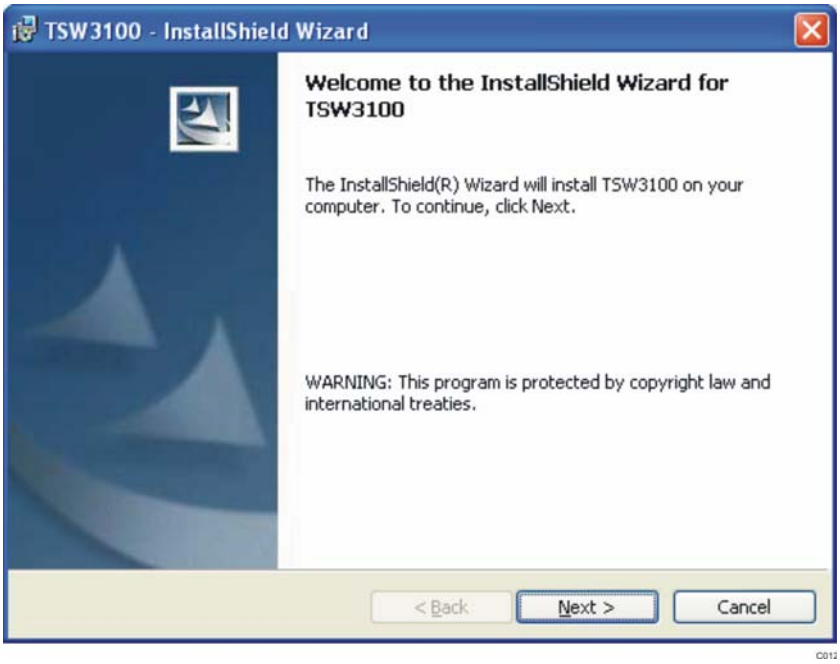

Figure 12. TSW3100 Installation Welcome

2. When the License Agreement (Figure 13) displays, select the *I accept the terms in the License agreement* option and click *Next* to accept the TSW3100 Software License Agreement.

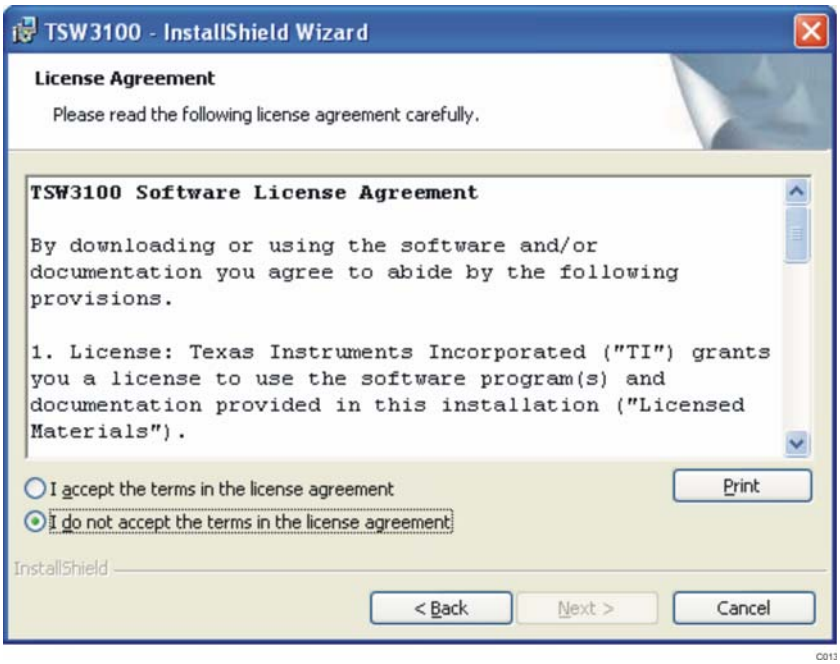

Figure 13. TSW3100 License Agreement

3. On the Customer Information (Figure 14) display, provide *User Name*, *Organization* information, and

select the appropriate *Install this Application for* option. Click *Next*.

Figure 14. Customer Information

4. On the Setup Type (Figure 15) display, select the *Complete* Setup Type option and click *Next*.

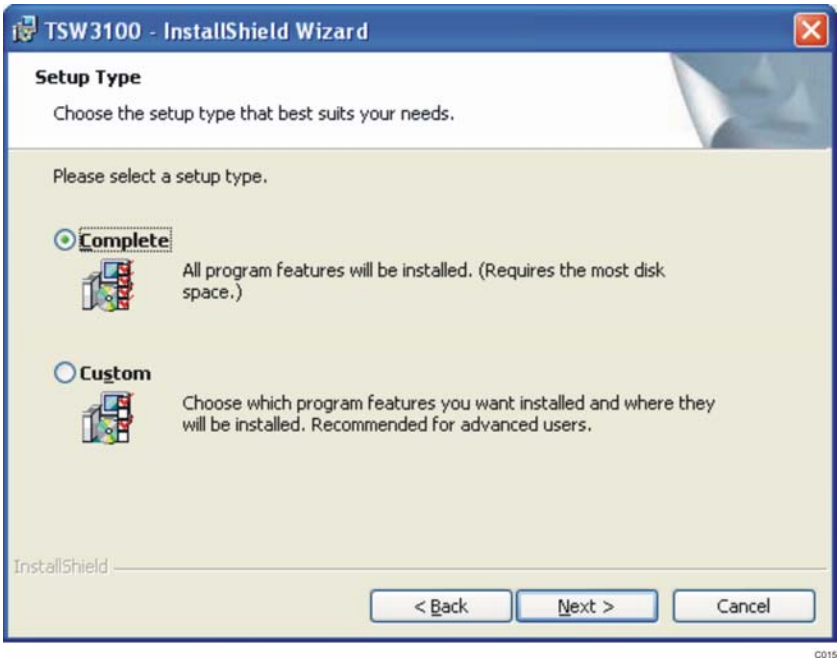

Figure 15. Setup Type

5. On the Ready to Install the Program (Figure 16) display, click *Install*. The installation takes between one and three minutes to complete.

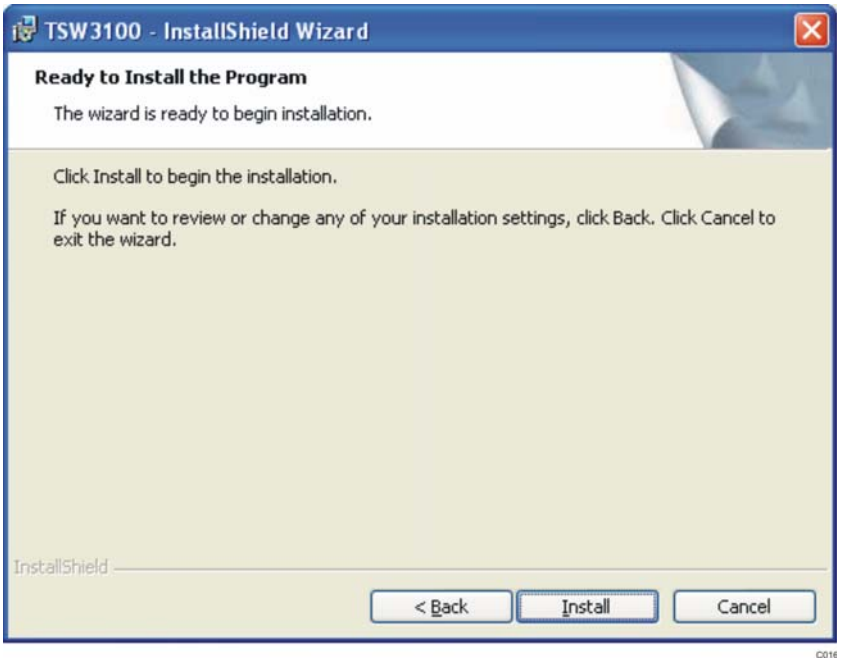

Figure 16. Ready to Install the Program

6. Click *Finish* once the InstallShield Wizard Completed screen (Figure 17) displays.

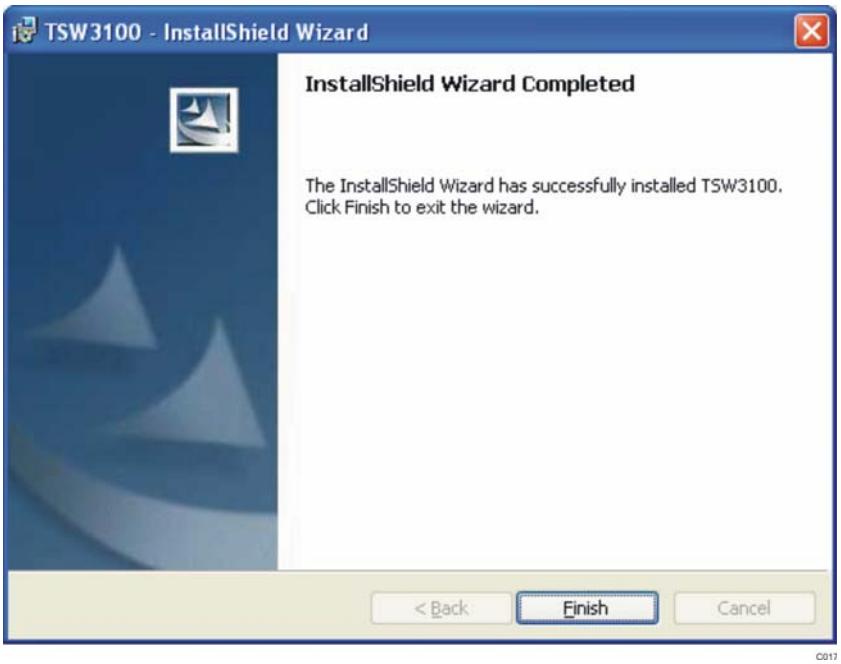

Figure 17. InstallShield Wizard Completed

2.5 Starting the TSW3100 Application Software

The TSW3100 software is now ready to use. To start the application programs, click the Windows menu sequence **start** → **All Programs** → **Texas Instruments** → **TWS3100Install_IS12**. You can select to display the application interface for the *TSW3100_CommSignalPattern.exe* (Section 6.3) software or the *TSW3100_MultiTonePattern.exe* (Section 6.1) software.

3 Apply Power to TSW3100 and Connect to a Host

To power the TSW3100, connect the 5–6 V power supply to J9. Move switch SW1 to the ON position. The four LEDs D3-D6 should now light. In addition, D13, D19 and D20 should also light.

Now, connect the TSW3100 Ethernet connector with a crossover cable to the PC or USB Ethernet adapter. Within approximately 5 seconds, the green Ethernet connector should also light, indicating a connection to the host (usually PC).

4 Host Interface

The TSW3100 uses simple interface protocols with TCP/IP over Ethernet with control and data transfer by Trivial FIle Transfer Protocol (TFTP). The protocols are host operating system agnostic (Windows, Linux, and so forth), although all examples and software provided by Texas Instruments are developed for Windows™ XP.

4.1 TSW3100 IP Address

The TSW3100 has a fixed IP address: 192.168.1.12x. The final digit x is defined by the DIP0 and DIP1 switch positions (Table 7) on SW2 (Figure 18) whenever power is applied or the FPGA is reconfigured.

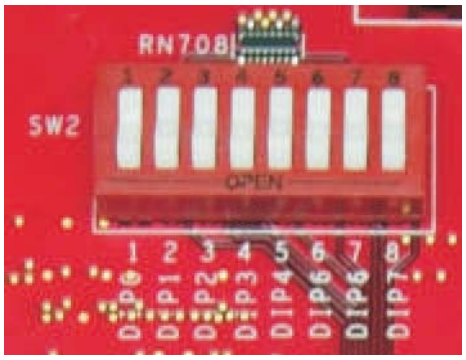

Figure 18. SW2 DIP Switches

Table 7. IP Address Digit Selection Using SW2

DIP0 Position	DIP1 Position	IP Address
Closed	Closed	192.168.1.12**0**
Closed	Open	192.168.1.12**1**
Open	Closed	192.168.1.12**2**
Open	Open	192.168.1.12**3**

For convenience, a USB to Ethernet adapter is provided for the host PC to maintain a dynamic IP address allocation and still connect to the TSW3100 using a separate, fixed IP address. See installation instructions for the USB Ethernet adapter found in Section 2.1.

4.2 TSW3100 Control Files

The TSW3100 is controlled by transferring short files with four 32-bit control words. The content of these control words:

```
Word 1 – Function code
           Bit 0 – Off
                    Turns off pattern generator
           Bit 1 – Error reset
           Bit 2 – Vector write
                    Start writing pattern vector to TSW3100
           Bit 3 – Reserved
           Bit 4 – Pattern gen master cmos start
                    Start CMOS pattern output in Master mode
           Bit 5 – Pattern gen master lvds start
                    Start LVDS pattern output in Master mode
           Bit 6 – Pattern gen slave cmos start
                    Start CMOS pattern output in Slave mode
           Bit7 – Pattern gen slave lvds start
                    Start LVDS pattern output in Slave mode

           Bit8-Bit31 Not used

Word 2 – Intro vector number
                    Starting vector number during 1st pass
                    through pattern (defaults to zero)

Word 3 – Start vector number
                    Vector number during 2nd and later passes through
                    pattern (defaults to zero)

Word 4 – Finish vector number
                    End vector for the pattern, which returns to
                    Start vector number
```

Words 2–4 and the data pattern must be a multiple of 4 vectors for LVDS output.

4.3 TSW3100 Data Pattern Format

The TSW3100 data pattern for the LVDS output consists of 16-bit little-endian words in a sequence representing the 16 differential outputs. Note, the Low Voltage Differential Signaling (LVDS) SYNC and DATA CLK signals are generated in firmware and are not stored in memory. The TSW3100 data pattern for Complementary Metal Oxide Semiconductor (CMOS) outputs uses 36-bits of a 64-bit little-Endian word, with the final 28-bits set to zero.

These data files are easily generated with programs such as MATLAB™ or LabVIEW™, with MATLAB functions described in Section 5.

4.4 TSW3100 Operation Sequence

The TSW3100 operation consists of several file transfers to load and start a pattern. The basic steps are (assuming an IP address of 192.168.1.123):

1. Control off

   `tftp -i 192.168.1.123 put control_off /tmp/control`

   control_off is a file containing the 32-bit words:
   0x 00000000 00000000 00000000 00000000
2. Vector Write Start

   tftp -i 192.168.1.123 put control_vwn /tmp/control
   control_vmn is a file containing the 32-bit words:
   0x 00000002 00000000 00000000 00000000

3. Data Vector Pattern

   The data vector pattern must be transferred in files with sizes less than 5 MBytes, which equals 2.5M Vectors for LVDS output or 1.25Mvectors for CMOS outputs. Larger patterns are transferred in multiple steps using this sequence:

   (a) Each file <5MBytes is first transferred to the TSW3100 processor memory.
   tftp -i 192.168.1.123 put data_pattern.bin /tmp/vector

   (b) Transfer a **ready_rx** file to indicate that the processor should transfer the pattern from the processor memory to the pattern memory. The ready_rx file is any non-zero file size. We use a file contain the 32-bit word: 0x 20090120
   tftp -i 192.168.1.123 put ready_rx /tmp/ready_rx

   (c) We recommend generating a pause of 0.5 seconds per Mbyte, to allow the TSW3100 processor to transfer the pattern to pattern memory.

4. Control Pattern Generator Start

   tftp -i 192.168.1.120 put control_file /tmp/control
   The TSW3100 pattern is started by the transfer of the control file words shown in Section 4.2 .

4.5 TSW3100 Connection to LVDS HSDAC EVM

For an LVDS output to a TI LVDS interface high-speed DAC EVM (DAC5682Z EVM), connect the DAC EVM to connector J74 (see Figure 19). This connection provides the 16 LVDS differential data bits, an LVDS DATA CLK at the data rate, and the LVDS SYNC signal to the DAC EVM. On the same connector, the high speed DAC EVM provides a clock to the TSW3100 to clock the output pattern. This clock must be at 1/8th the data rate of the LVDS data, or 1/4th the DATA CLK frequency, and have a minimum frequency of 25 MHz, for a minimum LVDS data rate of 200 MHz.

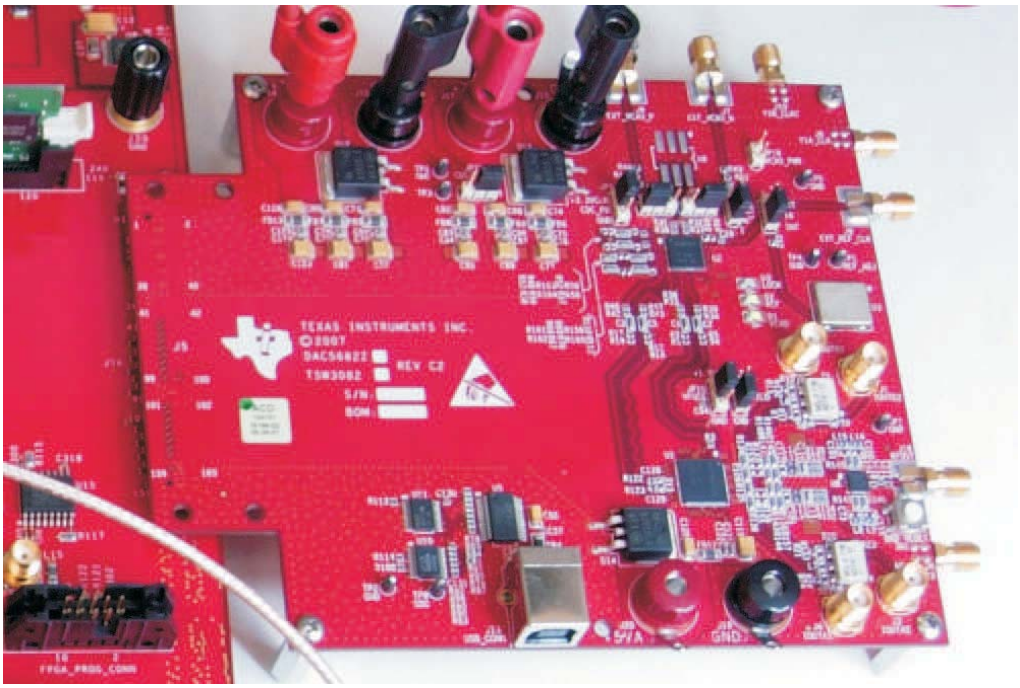

Figure 19. Connection of the DAC5682Z EVM to the TSW3100

When power is applied, LEDs D13 (PATT GEN IDLE), D19 (DDR2 PLL LOCK) and D20 (NIOS PLL LOCK) should light. When an LVDS clock signal is provided on connector J74, D18 (LVDS PLL LOCK) should light.

After the LVDS pattern starts, using the sequence in Section 4.4, LEDs D14 (PATT GEN CLK), D15 (PATT GEN RUN), and D24 (LVDS MODE) should light (Table 8).

Table 8. TSW3100 LEDs for LVDS Patterns

LED Name	Power Applied	LVDS Pattern Starts
D13	ON	ON
D14		ON
D15		ON
D18	ON (clock signal)	ON
D19	ON	ON
D20	ON	ON
D24		ON

4.6 TSW3100 Connection to CMOS HSDAC EVMs

For CMOS output to a TI CMOS interface high speed DAC (Figure 20) EVM (DAC5688 EVM), connect the DAC EVM to connectors J63 and J64 using the provided adapter PCB. This connection provides the 32 LVCMOS (3.3V) data bits to the DAC EVM. A clock at the CMOS output data rate must be provided to SMA connector J73 (CMOS CLK). This clock has a minimum frequency of 25 MHz, for a minimum CMOS data rate of 25 MHz. When using existing TI HSDAC EVMs, the TSW3100 CMOS CLK can be provided as follows:

- TI HSDAC EVMs (DAC5687EVM, DAC5688EVM or TSW3003) using external clock mode - Use the PLL LOCK output SMA.
- TI HSDAC EVMs that include the CDCM7005 clock buffer using PLL Clock mode - Use a spare CDCM7005 clock buffer output at the DAC data rate.
- Other TI HSDAC EVMs - Provide two synchronous clock sources or split an external clock source to provide a clock for both the DAC and TSW3100.

Note: The user must verify the timing of the DAC clock relative to the data to assure setup and hold times are met. These may require additional delay between the DAC EVM and TSW3100 clocks (easily accomplished by adding cable length).

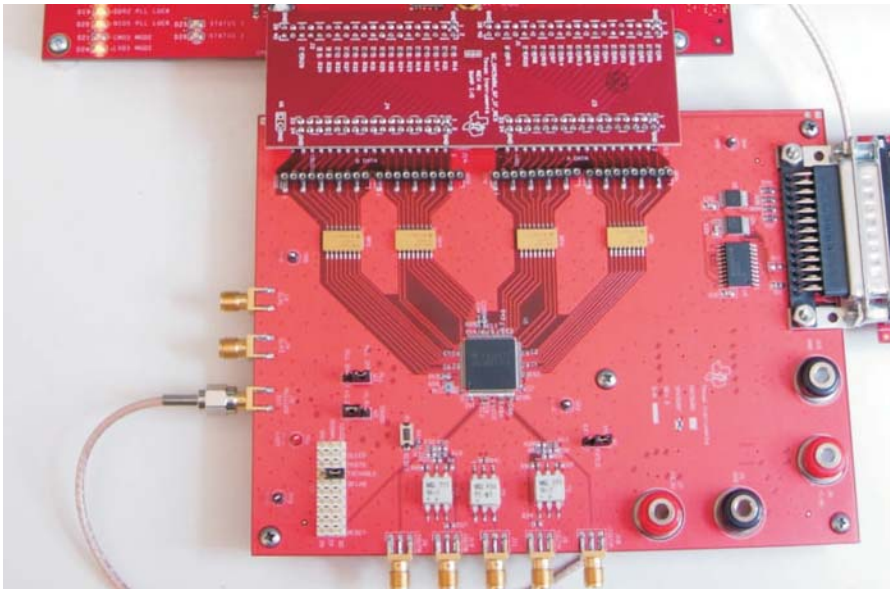

Figure 20. CMOS HSDAC Connection to the TSW3100

When power is applied, LEDs D13 (PATT GEN IDLE), D19 (DDR2 PLL LOCK) and D20 (NIOS PLL LOCK) should light. There is no LED indication for the presence of the CMOS CLK.

After the CMOS pattern starts using the sequence in Section 4.4, LEDs D14 (PATT GEN CLK), D15 (PATT GEN RUN) and D21 (CMOS MODE) should light (Table 9).

Table 9. TSW3100 LEDs for CMOS Patterns

LED Name	Power Applied	CMOS Pattern Starts
D13	ON	ON
D14		ON
D15		ON
D19	ON	ON
D20	ON	ON
D21		ON

4.7 TSW3100 Master/Slave Operation

The TSW3100 includes the ability to synchronize multiple boards using a master/slave synchronization, However, this mode is not documented.

5 Example MATLAB Functions for TSW3100 Control

Texas Instruments provides several functions in MATLAB for generation of pattern and control files and interfacing to the TW3100. These functions are provided as *.m files with the TSW3100.

These functions include:

- Section 5.1 LVDS Pattern File Generation
- Section 5.2 CMOS Pattern File Generation
- Section 5.3 Pattern File Loading to the TSW3100
- Section 5.4 Running the TSW3100

5.1 LVDS Pattern File Generation

The function **TSW3100writer_lvds** is used to generate the 16-bit words for the LVDS pattern. **File_Name** is a text string with file path and name for the output pattern file. The input data is assumed to be real or complex 16-bit integers, scaled between -32768 and 32767. The input variable **twos_or_offset** is a string that must start with a **t** (twos-compliment) or **o** (offset binary) to signify the format of the output pattern. The input variable **complex_or_real** is a string that must start with **c** or **r** (can be longer) to signify if the input vector is real or complex. The function returns the length of the pattern, which would be double the length of the input data for complex data since the output is interleaved complex

```
function vector_length=TSW3100writer_lvds(File_Name, data, twos_or_offset, complex_or_real);
% TSW3100writerfast_complex(File_Name, data, twos_or_offset)
% File_Name = text string with file path and name
% data = real or complex integer data scaled between
%        -32768 (full scale negative) and
%        32767 (full scale positive)
% twos_or_offset = the matlab string 'two' for twos complement and
%                  'off' for offset binary
% writes in little endian for TSW3100 LVDS output format
% 16-bits per vector, two vectors used for interleaved complex signal

if complex_or_real(1:1)=='c'
    %data is complex so interleave real and imaginary into array to write
    if twos_or_offset(1:1)=='t'
        data_interleaved(1:2:2*length(data))=real(data);
        data_interleaved(2:2:2*length(data))=imag(data);
    elseif twos_or_offset(1:1)=='o'
        data_interleaved(1:2:2*length(data))=real(data)+32768;
        data_interleaved(2:2:2*length(data))=imag(data)+32768;
    else
        error_msg = 'twos_or_offset must be string two... or off...'
    end
elseif complex_or_real(1:1)=='r'
```

```
    %data is real so just copy into array to write
    if twos_or_offset(1)=='t'
        data_interleaved=data;
    elseif twos_or_offset(1)=='o'
        data_interleaved=data+32768;
    else
        error_msg = 'twos_or_offset must be string starting with t or o'
    end
else
        error_msg = 'twos_or_offset must be string starting with t or o'
end

vector_length=length(data_interleaved);

% write the little endian binary file
fp = fopen(File_Name,'wb');
fwrite(fp,data_interleaved,'ubit16')
fclose(fp);
```

5.2 CMOS Pattern File Generation

The function **TSW3100writer_cmos** is used to generate the 64-bit words for the CMOS pattern. **File_Name** is a text string with file path and name for the output pattern file. The **data** is assumed to be real or complex 16-bit integers, scaled between -32768 and 32767. The input variable **twos_or_offset** is a string that must start with a **t** (twos-compliment) or **o** (offset binary) to signify the format of the output pattern. The function returns the length of the pattern.

```
function vector_length=TSW3100writer_cmos(File_Name, data, twos_or_offset);
% TSW3100writerfast_cmos_complex_twos(File_Name, data, twos_or_offset)
% File_Name = text string with file path and name
% data = complex integer data scaled between
%        -32768 (full scale negative) and
%        32767 (full scale positive)
% twos_or_offset = a matlab string starting with 't' for twos complement and
%                  'o' for offset binary
% writes in little endian for TSW3100 CMOS output format
% 64-bits per vector, I = 16 MSBs, Q = next 16 bits, bits 33-36 are for
% the extra 4 sync signals (not used here)

vector_length=length(data);

if twos_or_offset(1:1)=='t'
% interleave the complex data with odd being real
    data_interleaved(1:4:4*length(data))=real(data);
    data_interleaved(2:4:4*length(data))=imag(data);
    data_interleaved(3:4:4*length(data))=0;
    data_interleaved(4:4:4*length(data))=0;
elseif twos_or_offset(1:1)=='o'
    data_interleaved(1:4:4*length(data))=real(data)+32768;
    data_interleaved(2:4:4*length(data))=imag(data)+32768;
    data_interleaved(3:4:4*length(data))=0;
    data_interleaved(4:4:4*length(data))=0;
else
    error_msg = 'twos_or_offset must be string  starting with t or o'
end

% write the little endian binary file
fp = fopen(File_Name,'wb');
fwrite(fp,data_interleaved,'ubit16')
fclose(fp);
```

5.3 Pattern File Loading to TSW3100

The function **TSW3100_vectorwrite_load** is used to process a complete MATLAB data pattern and does the complete procedure to load it into the TSW3100 pattern memory. As needed, it breaks a large data pattern into smaller pattern segments to transfer sequentially. The input arguments are the data pattern **data, lvds_or_cmos** (a string starting with either **l** or **c**) indicating an LVDS or CMOS pattern, **twos_or_offset** (a string starting with either **t** or **c**), and **IPdigit**, the last digit of the IP address 192.168.1.12x.

The output argument is the data pattern length, which can be 2x the input pattern length for LVDS interleaved complex data.

This function includes two sub-functions:

- **TSW3100_vectorwrite_end**—transfers each pattern segment to TSW3100 process memory
- **transfer_file**—transfers the segment from processor memory to pattern memory

In addition, the functions TSW3100writer_lvds (Section 5.1) and TSW3100writer_cmos (Section 5.2). The function TSW3100_vectorwrite_load performs these operations:

1. Check if input data is complex
2. Calculate the maximum pattern segment length that can be transferred
3. If less them the maximum length, transfer once
4. If more than the maximum length, break into segments and transfer each sequentially

```
Function vector_length = TSW3100_vectorwrite_load (data, lvds_or_cmos, twos_or_offset, IPdigit)
% TSW3100_vectorwrite_load(data,lvds_or_cmos,twos_or_offset,IPdigit)
% data = real or complex integer data scaled between
%        -32768 (full scale negative) and
%        32767 (full scale positive)
% lvds_or_cmos = the matlab string starting with 'l' for twos complement and
%                  'c' for offset binary
% twos_or_offset = the matlab string starting with 't' for twos complement and
%                  'o' for offset binary


    %automatically checks of the data vector is complex or real
    if max(abs(imag(data)))>0
        complex = 2;
        complex_or_real = 'c'
    else
        complex = 1;
        complex_or_real = 'r'
    end

    % finds the # of pattern vectors that result in 5MByte file which is
    % the maximum for a single
    if lvds_or_cmos(1) == 'l'
        maxlength = 2500*1024/complex;
        vector_length=complex*length(data);
    else
        maxlength = 2500*1024/4;
        vector_length=length(data);
    end

%convert matlab vector to binary format to load to pattern generator
    if lvds_or_cmos(1) == 'l'

        %calculate the # of loads needed to transfer the data
        numloads=ceil(length(data)/maxlength);

        if numloads == 1
            %Pattern is less than the maximum pattern size, so we can
            %transfer all at once
            v_length=TSW3100writer_lvds('tsw3100_tempvector.bin', data, twos_or_offset,
complex_or_real);
            transfer_file(IPdigit);
            TSW3100_vectorwrite_end(v_length,lvds_or_cmos,IPdigit);

        else
            %Pattern is more than the maximum pattern size, so we must
            %break the pattern into separate files and load sequentially

            %sequence through the # of loads - 1 at maximum size
            for index = 1:numloads-1
                %calculate min and max of pattern segment
                array_min_index = 1+(index-1)*maxlength;
                array_max_index = index*maxlength;
                %transfer the file
                v_length=TSW3100writer_lvds('tsw3100_tempvector.bin',
data(array_min_index:array_max_index), twos_or_offset,complex_or_real);
                transfer_file(IPdigit);
                TSW3100_vectorwrite_end(v_length,lvds_or_cmos,IPdigit);

            end
            %now we need to transfer the final pattern segment

            %calculate min and max of the final pattern segment
            array_min_index = 1+(numloads-1)*maxlength;
            array_max_index = length(data);
            %transfer the file
            v_length=TSW3100writer_lvds('tsw3100_tempvector.bin',
data(array_min_index:array_max_index), twos_or_offset,complex_or_real);
            transfer_file(IPdigit);
            TSW3100_vectorwrite_end(v_length,lvds_or_cmos,IPdigit);
        end

else
```

```
        %calculate the # of loads needed to transfer the data
        numloads=ceil(length(data)/maxlength);

        if numloads == 1
            %Pattern is less than the maximum pattern size, so we can
            %transfer all at once
            v_length=TSW3100writer_cmos('tsw3100_tempvector.bin', data, twos_or_offset);
            transfer_file(IPdigit);
            TSW3100_vectorwrite_end(v_length,lvds_or_cmos,IPdigit);
        else
            %Pattern is more than the maximum pattern size, so we must
            %break the pattern into separate files and load sequentially

            %sequence through the # of loads - 1 at maximum size
            for index = 1:numloads-1
                %calculate min and max of pattern segment
                array_min_index = 1+(index-1)*maxlength;
                array_max_index = index*maxlength;
                %transfer the file
                v_length=TSW3100writer_cmos('tsw3100_tempvector.bin',
data(array_min_index:array_max_index), twos_or_offset);
                transfer_file(IPdigit);
                TSW3100_vectorwrite_end(v_length,lvds_or_cmos,IPdigit);
            end
            %now we need to transfer the final pattern segment

            %calculate min and max of the final pattern segment
            array_min_index = 1+(numloads-1)*maxlength;
            array_max_index = length(data);
            %transfer the file
            v_length=TSW3100writer_cmos('tsw3100_tempvector.bin',
data(array_min_index:array_max_index), twos_or_offset);
            transfer_file(IPdigit);
            TSW3100_vectorwrite_end(v_length,lvds_or_cmos,IPdigit);
        end
end

    %sub-function to transfer the data file
    function transfer_file(IPdigit)
    % transfer_file(IPdigit)
    % IPdigit = x=0,1,2,3 - the last digit of IP address 192.168.1.12x
        cmd_str = ['tftp -I 192.168.1.12' int2str(IPdigit) ' put tsw3100_tempvector.bin
/tmp/vector']
        dos(cmd_str) % write the command string to matlab window
        pause(0.1); % pause a short time after tftp to allow processor to catchup

    %sub-function to signal the end of the data file transfer. Signals for
    %the TSW3100 processor to transfer the data from the processor memory
    %to pattern memory
    function TSW3100_vectorwrite_end(vector_length,lvds_or_cmos,IPdigit)
    % TSW3100_vectorwrite_end(vector_length,lvds_or_cmos,IPdigit)
    % signal end of vector load.
    % Pause (~ second/2 MB) required as TSW3100 loads from processor memory into SDRAM.
        control(1)=537461024;
        fp = fopen('ready_rx','wb');
        fwrite(fp,control,'ubit32');
        fclose(fp);
        cmd_str = ['tftp -I 192.168.1.12' int2str(IPdigit) ' put ready_rx /tmp/ready_rx']
        dos(cmd_str)

        %Insert pause to allow TSW3100 processor to transfer pattern
        if lvds_or_cmos(1)=='l'
            pause(vector_length/1e6);
        else
```

5.4 Running the TSW3100

The function **TSW3100_vectorwrite_load** load a pattern files and start the pattern display. The input arguments are the data pattern array **data, lvds_or_cmos** a string starting with either **l** (LVDS) or **c** (CMOS) indicating the pattern type, **twos_or_offset** a string starting with either **t** (twos compliment) or **o** (offset binary) indicating output pattern format, **IPdigit**, the last digit of the IP address 192.168.1.12x, and **master_or_slave** a string starting with either **m** (master) or **s** (slave) defines how the TSW3100 operates.

The function returns an error message if the input arguments are out of range. The main body of the function includes all the basic steps outlined in Section 2.4 Section 4.4.

```
function error_msg=TSW3100_run(data, lvds_or_cmos, twos_or_offset, IPdigit, master_or_slave)
% error_msg = TSW3100_run(data, lvds_or_cmos, twos_or_offset, IPdigit,master_or_slave)
%            data = complex integer data scaled between
%                   -32768 (full scale negative) and
%                   32767 (full scale positive)
%    lvds_or_cmos = a matlab string starting with 'l' for LVDS output or 'c'
%                   for CMOS output
%  twos_or_offset = a matlab string starting 't' for twos complement or
%                  'o' for offset binary
%         IPdigit = IP address 192.168.1.12x where x= 0,1,2 or 3
% master_or_slave = a matlab string starting 'm' for master or 's' for slave
error_msg =[];

%round and check input data
data=round(data);
if min(min(real(data)),min(imag(data))) < -32768 | max(max(real(data)),max(imag(data))) > 32767
    error_msg = 'data must be between -32768 and 32767'
end

if lvds_or_cmos(1) ~= 'l' & lvds_or_cmos(1) ~= 'c'
    error_msg = 'lvds_or_cmos must be a matlab string starting with l for LVDS output or c for
CMOS output'
end

if twos_or_offset(1) ~= 't' & twos_or_offset(1) ~= 'o'
    error_msg = 'twos_or_offset must be a matlab string starting with t for twos complement or o
for offset binary'
end

if master_or_slave(1) ~= 'm' & master_or_slave(1) ~= 's'
    error_msg = 'master_or_slave must be a matlab string starting with m for master or s for
slave'
end

if IPdigit ~= 0 & IPdigit ~= 1 & IPdigit ~= 2 & IPdigit ~= 3
    error_msg = 'IPdigit must be an integer = 0, 1, 2 or 3'
end

if length(error_msg) == 0
    %stop pattern generator
    TSW3100_stop(IPdigit);

    %signal beginning of vector load
    TSW3100_vectorwrite_begin(IPdigit);

    %load vector for lvds or cmos
    vector_length=TSW3100_vectorwrite_load(data,lvds_or_cmos,twos_or_offset,IPdigit);

    %write control file
    TSW3100_start(vector_length,lvds_or_cmos,IPdigit,master_or_slave);

    error_msg = 'no error'
end
```

6 Generating LVDS and CMOS Test Patterns

TI provides two programs to generate test patterns for the TSW3100: **TSW3100_MultitonePattern** (Section 6.1) and **TSW3100_CommSignalPattern** (Section 6.3). Section 2.5 describes how to start these two TSW3100 software applications.

6.1 TSW3100_MultitonePattern Software

The **TSW3100_MultitonePattern** program can automatically generate a test pattern with single or multiple tones. The patterns can be complex or real for LVDS or CMOS outputs. The TSW3100 can be controlled directly from software interface, including loading, starting, and stopping the pattern.

Figure 21 shows the TSW3100_MultiTonePattern Software GUI generating a pattern by using the default settings and clicking the **Create and Save/Run TSW3100** button.

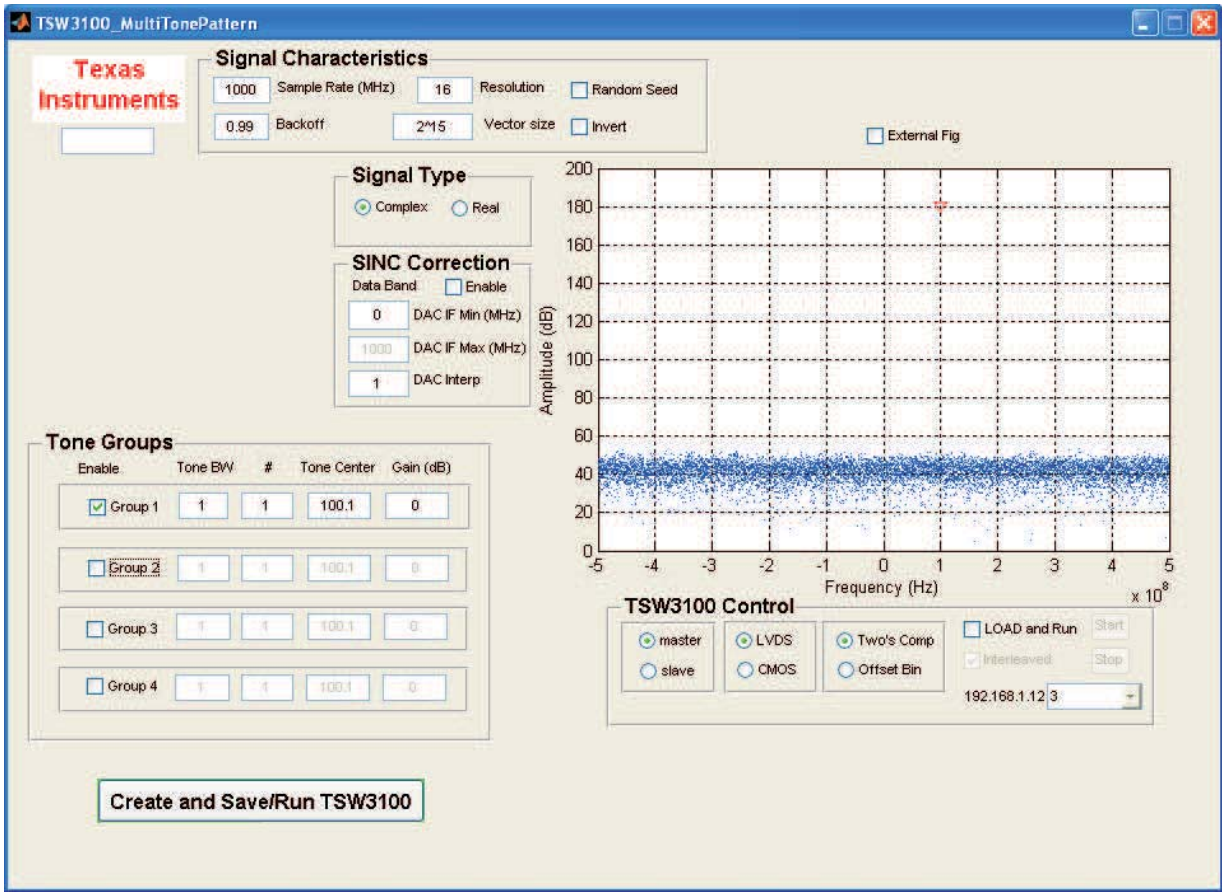

Figure 21. TSW3100_MultiTonePattern Graphical User Interface

The graphical user interface controls for the TSW3100_MultiTonePattern window divide into these areas:

Signal Characteristics area

- **Sample Rate (MHz)**—sample rate of the pattern in *MHz*. Rate is independent of whether the pattern is interleaved or not. For interleaved data, such as complex data for the LVDS pattern or interleaved CMOS data will have an interface rate of twice this sample rate.
- **Backoff**—linear backoff of the maximum signal from full scale. TI recommends using a value of less than 0.999 for the backoff.
- **Resolution**—number of bits of the pattern.
- **Vector size**—number of vectors in the pattern. For interleaved data, such as complex data for the LVDS pattern or interleaved CMOS data will have an interface rate of twice this number of vectors.
- **Random Seed**—selecting the Random Seed check box generates a different set of random phases each time the pattern is generated. If not selected, the exact same phases are used each time and therefore the patterns are identical. In generating the multi tone pattern, the phase of each tone is generated randomly to prevent aligning of the phase and generation of a very large peak to average ratio.

- **Invert**—multiplies (inverts) the signal by –1.

Signal Type option

- **Complex**—signal is complex.
- **Real**—signal is real.

SINC Correction area

- **Enable**—enables SINC correction, which applies a gradual increasing slope to compensate for the SINC rolloff of the HSDAC zero order hold output.
- **DAC IF Min (MHz)**—DAC IF Min is the minimum frequency of band at the DAC output. **DAC IF MAX** is calculated automatically using the formula, IF MIN plus the pattern bandwidth. The data pattern has a bandwidth that is equal to the sample rate for a complex signal and = the sample rate for a real signal. With interpolating DAC that includes mixer capabilities, this band is often interpolated and mixed to a higher frequency.
- **DAC Interp**—specifies the interpolation used in the HSDAC. With the pattern sample rate, this defines the DAC sample conversion rate and therefore the SINC rolloff effect.

Tone Groups area

There can be up to four group of tones combined into the final pattern. The **Enable** check box is used to select each desired group. Each tone group is defined by these input fields:

- **ToneBW**—total bandwidth (maximum frequency – minimum frequency) of this tone group. If there is only one tone in the group, the tone is at the **Tone Center** of the group and this parameter is *ignored*.
- **#**—number of tones in the group.
- **Tone Center**—center frequency of the tone in MHz. To avoid a pattern that is repetitive over a very short time scale, TI recommends you set this value slightly off from a round value. This is why 100.1 MHz is used rather than 100 MHz, which would repeat every 10 samples.
- **Gain (dB)**—amplitude in dB of each tone in the group, relative to tones in other groups (not to fullscale – the backoff parameter in **Signal Characteristics** is used to set the power of the combined pattern relative to fullscale). It is not the combined power of all the tones in the group, but for each tone. This can be a positive or negative value. If one group is set to 10 dB and a second group to –20 dB, the power difference for a tone in the first group compared to a tone in the second group is 30 dB.

TSW3100 Control area

These option buttons and other controls are used to load, start, and stop patterns with the TSW3100.

- **Master/Slave option**—operates TSW3100 in master or slave mode.
- **LVDS/CMOS option**—generates LVDS or CMOS pattern.
- **Two's Comp/Offset Bin option**—selects two's compliment or offset binary pattern output format.
- **LOAD and Run**—check to load the pattern to the TSW3100 and start the pattern.
- **Interleaved**—check to generate interleaved complex data for CMOS pattern. For LVDS, this check box has no effect since LVDS data must be interleaved.
- **Start**—restarts the TSW3100 pattern output, which started from the intro vector and sends a new SYNC for LVDS patterns.
- **Stop**—stops the TSW3100 pattern output.
- **192.168.1.12x**—select fixed IP address for the USB to Ethernet adapter.
- Note: The Start and Stop functions can also be executed by using the Switch S7 on the TSW3100EVM. If the test pattern is currently running, pressing this switch once will stop the pattern. Pressing the switch again will then re-start the pattern from the beginning.

External Figure

When checked, a separate window will display the amplitude in dB vs. frequency of the pattern. For real patterns, only the positive frequency amplitudes displays. A red, inverted triangle (Figure 21, Figure 23, and Figure 24) identifies the largest amplitude tone. If there are multiple tones with the same power, the lowest frequency is identified with the triangle.

Note: When you select the **External Fig** check box, a separate window with the amplitude vs. frequency range graphic displays. This permits you to save, copy, and print the multi-tone pattern output.

Create and Save/Run TSW3100

This button creates the pattern, and if the TSW3100 **LOAD and Run** check box is selected, loads the file to the TSW3100.

6.2 TSW3100_MultitonePattern Examples

6.2.1 Four Tone Groups Pattern

Overview: Let's set up four tone groups (Figure 22), change the sample rate to 500 MHz, and keep the other parameters at the default values displayed in Figure 21. To generate the pattern, click the *Create and Save/Run TSW3100* button. The amplitude spectral plot for this pattern displays in Figure 23. The spectra for tone groups three and four do not show the individual tones, because the spacing is less than the pixel spacing for the display. The standard MATLAB figure control (magnifying glass) can be used to zoom in on the displayed tone group and see the individual tones (Figure 24).

This example illustrates the TSW3100_MultiTonePattern software's ability to:

- set different tone bandwidths.
- select a negative tone center (Group 4).
- use positive and negative gains.
- employ a large number of tones.

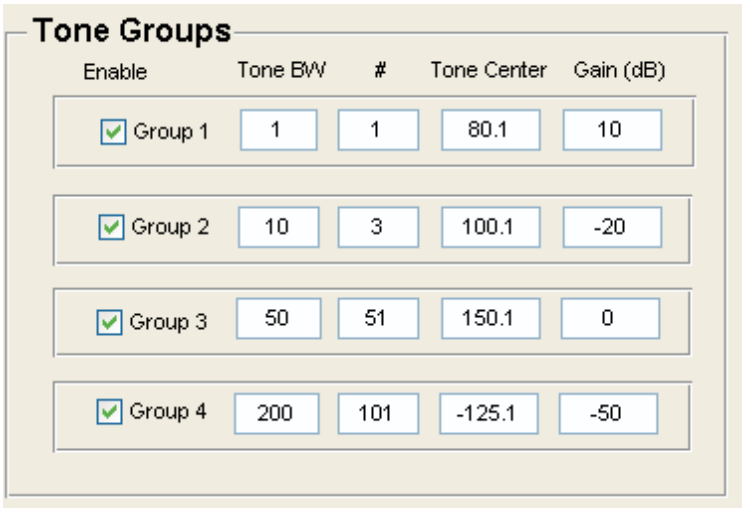

Figure 22. Tone Groups Settings

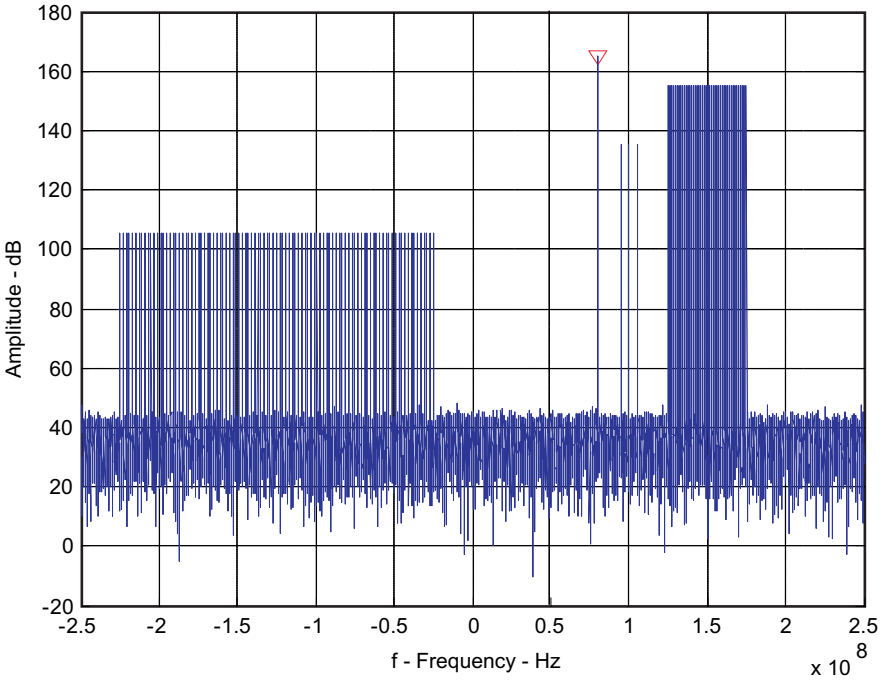

Figure 23. Spectral Plot of the Four Tone Groups Pattern

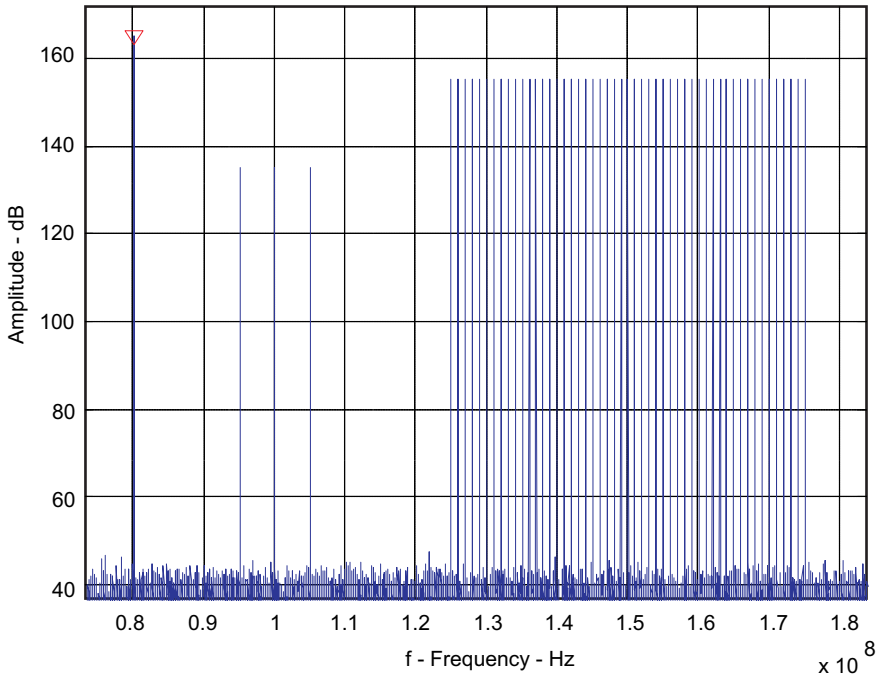

Figure 24. Magnify Tone Groups 1-3 Shown in Previous Figure

6.2.2 Download Four Tone Groups Pattern to TSW3100 / DAC5682Z EVM

Now lets download the pattern to the TSW3100 and send it to the DAC5682Z EVM. This sets the DAC5682Z with twice interpolation rate, increasing the data rate to 1 GSPS, and enables fs/4 mixing, which quadrature mixes the IQ signal to an output signal centered at 250 MHz. Following the test setup procedure in the Section two of the DAC5682Z/TSW3082EVM User's Guide:

1. Provide a 1 GHz clock to the DAC5682Z EVM. Apply power to the TSW3100 and DAC5682Z EVM's.
2. Connect to the host computer using the procedure in the DAC5682Z/TSW3082EVM User's Guide.
3. Load the following setup file for the CDCM7005: C:\Program Files\Texas Instruments\TSW3100\Example Register Files\Example_1.reg7005
4. Load the following setup file for the DAC5682Z: C:\Program Files\Texas Instruments\TSW3100\Example Register Files\Example_1.reg5682
5. Select the **LOAD and Run** check box.
6. Use the **TSW3100 Control** settings to select the *Master*, *LVDS*, and *Two's Comp* options.
7. Regenerate the pattern by clicking **Create and Save/Run TSW3100** button.

The DAC output spectrum (10–490 MHz) should display similar to Figure 25.

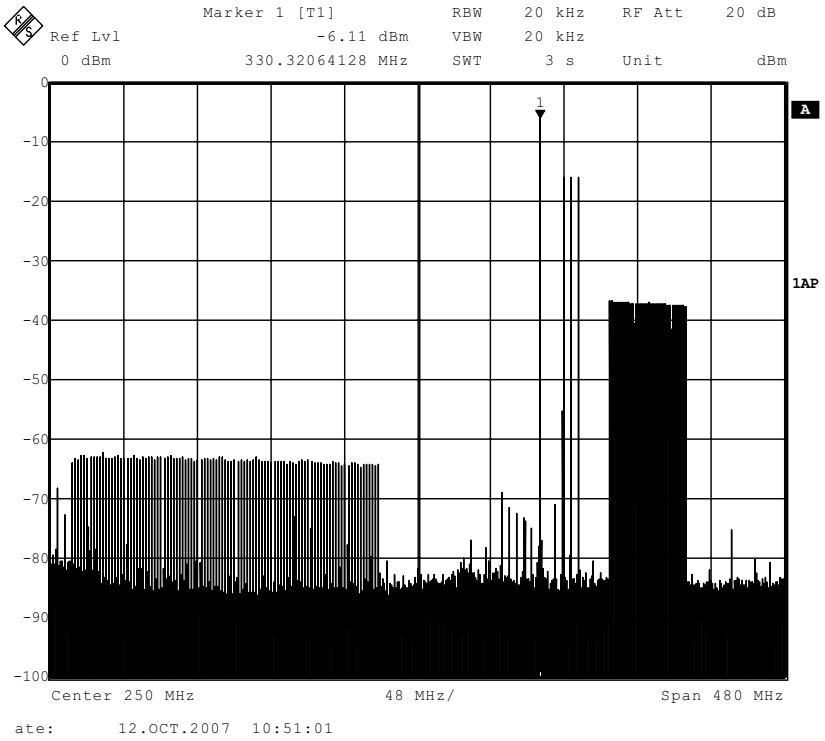

Figure 25. DAC5682Z Output Spectrum for Four Tone Groups

6.2.3 Convert Four Tone Groups Pattern to Real IF

To convert the pattern to a real IF:

1. Select *Real* option in the **Signal Type** area.
2. De-select the **Enable** check box for Group 4, so that all tone groups generate positive frequencies.
3. Click the **Create and Save/Run TSW3100** button.

The spectral plot in Figure 26 displays.

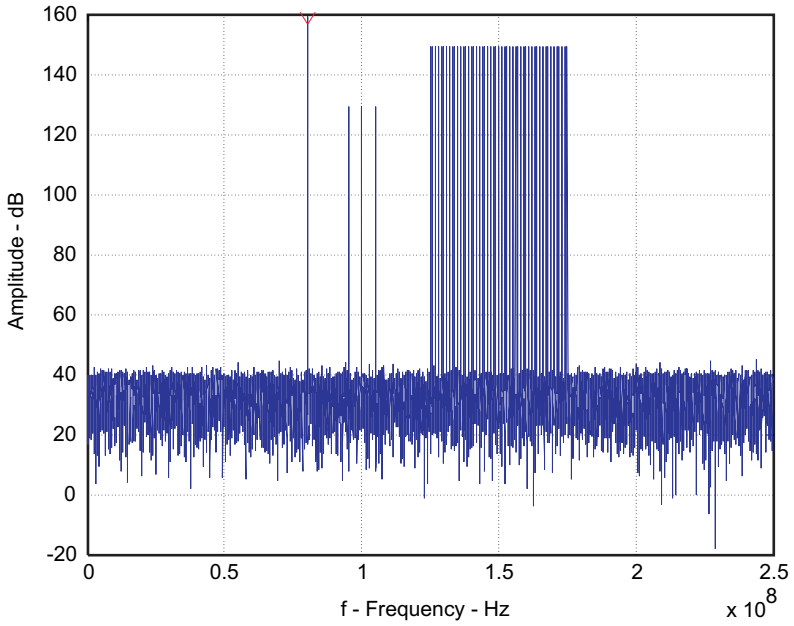

Figure 26. Spectral Plot of Real IF Pattern

6.2.4 Download Real IF Pattern to TSW3100 / DAC5682Z EVM

Now lets download the IF pattern to the TSW3100 and send the pattern to the DAC5682Z EVM. This sets the DAC5682Z with double interpolation, increasing the data rate to 1 GSPS. Following the test setup procedure in the DAC5682Z/TSW3082EVM User's Guide:

1. Provide a 1 GHz clock to the DAC5682Z EVM.
2. Load the CDCM7005 with the following:C:\Program Files\Texas Instruments\TSW3100\Example Register Files\Example_2.reg7005 setting file.
3. Load the DAC5682Z with the following:C:\Program Files\Texas Instruments\TSW3100\Example Register Files\Example_2.reg5682 settingfile.
4. Select the **LOAD and Run** check box.
5. Use the **TSW3100 Control** to select the *Master, LVDS*, and *Two's Comp* options.
6. Regenerate the pattern by clicking **Create and Save/Run TSW3100** button.

The DAC output spectrum (10–490 MHz) should display similar to Figure 27.

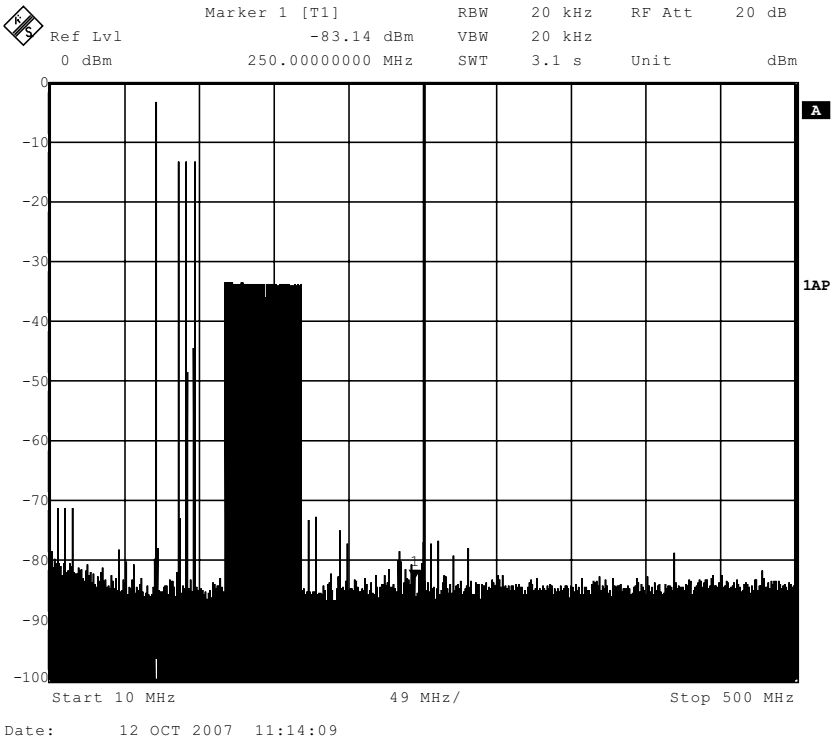

Figure 27. DAC5682Z Output Spectrum for Example 2

6.3 TSW3100_CommSignalPattern Software

The **TSW3100_CommSignalPattern.exe** program automatically generates a test pattern for several modulated communications signals such as Wideband Code Division Multiple Access (WCDMA), Time Division - Synchronous Code-Division Multiple Access (TD-SCDMA), and a generic Citriodora Amplitude Modulation (QAM) modulated signal. The patterns can be complex or real for LVDS or CMOS outputs. The TSW3100 can be controlled directly from the **TSW3100_CommSignalPattern** software, including Loading, Starting, and Stopping the pattern.

Figure 28 shows the TSW3100_CommSignalPattern Software GUI generating a pattern by using the default settings and clicking the **Create** button.

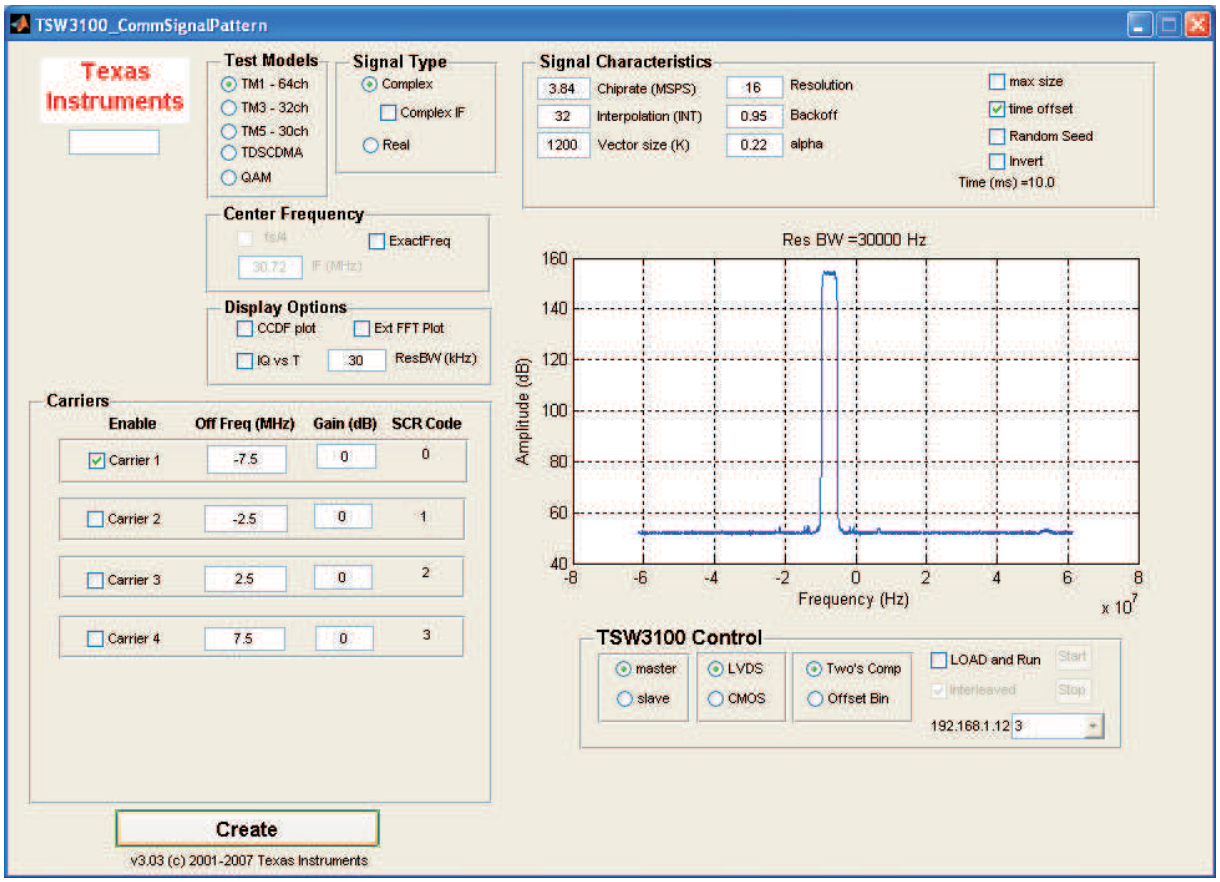

Figure 28. TSW3100_CommSignalPattern Graphical User Interface

The graphical user interface controls for the TSW3100_CommSignalPattern window divide into these areas:

Test Models area

This section defines the chip or symbol data used for the pattern generation. The data for the WCDMA TM1, WCDMA TM3, WCDMA TM5, TD-SCDMA, and QAM test models were generated with the Agilent Advanced Digital System and typically demodulate with less than 0.3% EVM. See the file *TI WCDMA GUI v3 Test Model Stats.pdf* for pictures of the demodulated signals in Agilent Visual Studio Analyzer.

- **TM1 – 64 ch**—WCDMA TM1 with 64 channels per 3GPP specification.
- **TM3 – 32 ch**—WCDMA TM3 with 32 channels per 3GPP specification.
- **TM5 – 30 ch**—WCDMA TM5 with 30 channels per 3GPP specification.
- **TD-SCDMA**—TD-SCDMA Downlink signal with 16 user codes active.
- **QAM**—Citriodora Amplitude Modulation.

Signal Type area

- **Complex**—signal is complex.
- **Complex IF**—select check box to modulate the combined group of carriers to a complex IF frequency, using the values in the **Center Frequency** pane. When unchecked, the combined group of carriers are centered at 0 Hz.
- **Real**—signal is modulated to a real IF frequency per the values in the **Center Frequency** pane.

Signal Characteristics area

- **Chiprate (MSPS)**—chip or symbol rate of the baseband data in MSPS.
- **Interpolation (INT)**—Integer value of the oversample rate from the chip or symbol data. The final pattern data rate is the chip rate × Interpolation. For example, 3.84 MSPS * 32 = 122.88 MSPS.
- **Vector size (K)**—number of K vectors in the pattern (× 1024). This is independent of whether the pattern is interleaved or not. For interleaved data, such as complex data for the LVDS pattern or interleaved CMOS data, you have twice this number of vectors.
- **Pilot Gain**—(TD-SCDMA test model only) linear gain of TD-SCDMA pilot relative to data. Typically used to reduce the peak power of the pilots which can be quite large when several carriers are combined as the pilots for each carrier add coherently.
- **Resolution**—number of bits in the pattern.
- **Backoff**—linear backoff of the maximum signal from full scale. TI recommends using a value of 0.95 or less for the backoff.
- **Alpha**—RRC filter characteristic. Usually 0.22 for WCDMA and TD-SCDMA.
- **QAM width**—(QAM test model only) width in resolution of the square QAM constellation, equal to the square root of the number of constellation points. For example, QAM64 has a width of 8 and QAM256 has a width of 16.
- **Max size**—sets the vector to the largest size possible, which uses all the baseband vector symbols (or chips).
- **Time offset**—slightly offsets the WCDMA carriers in time by 1/(N***Chiprate**), where N is the number of active carriers. This slightly reduces the PAR of a multicarrier signal. Displays only for TM1, TM3, TM5, or QAM test models
- **Random Seed**—selecting the Random Seed check box generates a different set of random phases each time the pattern is generated. If not selected, the exact same phases are used each time and therefore the patterns are identical. In generating the QAM patterns, the baseband symbol is generated randomly.
- **Invert**—multiplies (inverts) the signal by –1.
- **Time (ms)**—displays the total time of the pattern in milliseconds, which is VectorSize×1024/**Chiprate**.

Center Frequency area

This pane controls the center frequency of the group of carriers. Each carrier is offset from this center frequency by **Offset Freq (MHz)** value in the Carriers area.

- **fs/4**—sets the center frequency exactly to the sample rate divided by 4, or Chip Rate ×interpolation/4.
- **ExactFreq**— uses the exact frequency specified in **IF (MHz)** and Carrier **Off Freq (MHz)**. When unselected, the frequency is rounded to the closest frequency that has a prime integer number of periods in the pattern time. When using the exact frequency, if there is not an integer # of periods in the pattern time, there may be a glitch in the pattern as it wraps from back to front. This is seen in the FFT display as *skirts* on the carrier (Figure 29). Typically this control is unselected. The rounded frequency for each carrier is stored in a log file in the subfolder /testfiles.
- **IF (MHz)**—center frequency for the carrier group. Note, this frequency is rounded to the lowest frequency that has an integer number of periods in the pattern time when **ExactFreq** is unchecked.

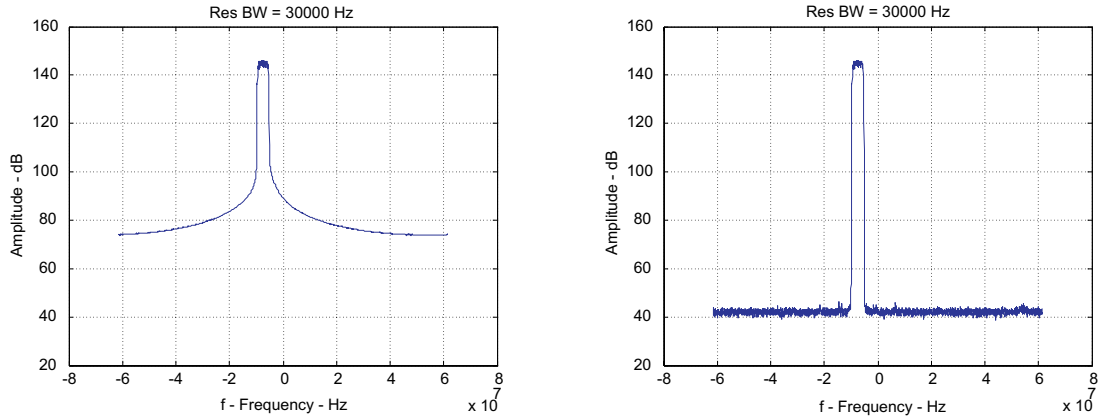

Figure 29. Comparison of Using the Exact Frequency (left) vs Rounded Frequency (right)

Carriers area

There can be up to four carriers for WCMDA/QAM and six carriers for TD-SCDMA that are combined into the final pattern. The **Enable** check box is used to select individual carriers, which are described with these fields:

- **Off Freq (MHz)**—offset frequency of the carrier in MHz from the center frequency. Note, this offset may be slightly shifted if the ExactFreq check box is unselected. When using the exact frequency, if there is not an integer number of periods in the pattern time, there may be a glitch in the pattern as it wraps from back to front. This is seen in the FFT display as *skirts* on the carrier (Figure 29). Typically the rounded frequency is used. The rounded frequency for each carrier is stored in a log file in the subfolder /testfiles.
- **Gain (dB)**—amplitude in dB of each carrier relative to other carriers (not to fullscale). The **Backoff** parameter in the Signal Characteristics pane is used to set the power of the combined pattern relative to fullscale. The Gain can be a positive or negative value. If one carrier is set to 10 dB and a second carrier to –20 dB, the power difference between the first carrier and the second carrier is 30 dB.
- **SCR Code**—carrier SCR code that can be used to set up the demodulation properties in a spectrum analyzer.

Display Options area

- **CCDF plot**—displays the pattern CCDF in a separate window when selected. Note, the zero time (during the uplink slots) of the TD-SCDMA pattern is included in the average power, so for TD-SCDMA, the downlink average power is ~ 2.5 dB lower than displayed if an integer number of slots are used.
- **IQ vs T**—displays the real and complex time series of the pattern in a separate window when selected.
- **Ext FFT Plot**—displays the spectral plot in a separate window when selected. Useful to save, copy, and print spectral plot output.
- **Res BW (kHz)**—specifies the averaging window for the FFT plot, similar to the resolution bandwidth function of a spectrum analyzer.

TSW3100 Control area

These option buttons and other controls are used to load, start, and stop patterns with the TSW3100.

- **Master/Slave option**—operates TSW3100 in master or slave mode.
- **LVDS/CMOS option**—generates LVDS or CMOS pattern.
- **Two's Comp/ Offset Bin option**—selects two's compliment or offset binary output format.
- **LOAD and Run**—check to load the pattern to the TSW3100 and start the pattern.
- **Interleaved**—check to generate interleaved complex data for CMOS pattern. For LVDS, this check box has no effect since LVDS data must be interleaved.

- **Start**—restarts the TSW3100 pattern output, which started from the intro vector and sends a new SYNC for LVDS patterns.
- **Stop**—stops the TSW3100 pattern output.
- **192.168.1.12x**—select fixed IP address for the USB to Ethernet adapter.
- **Create**—generates the composite signal pattern and loads it to the TSW3100 when **LOAD and run** check box is selected.
- Note: The Start and Stop functions can also be executed by using the Switch S7 on the TSW3100EVM. If the test pattern is currently running, pressing this switch once will stop the pattern. Pressing the switch again will then re-start the pattern from the beginning.

6.4 TSW3100_CommSignalPattern Examples

6.4.1 Three Carrier WCDMA TM1 Pattern

Let's do a three carrier, WCDMA TM1, complex baseband example.

1. Select carriers at –7.5, 2.5 and 7.5 MHz.
2. Keep all the default values and select the **Enable** check boxes for **Carrier 3** and **Carrier 4** (Figure 30).

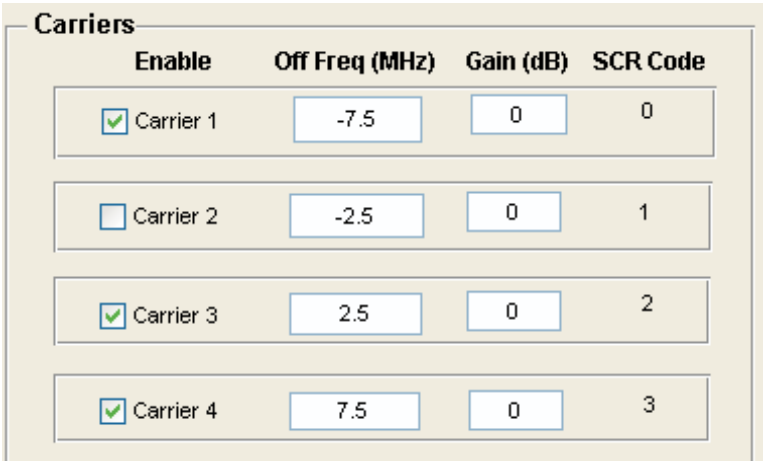

Figure 30. Carrier Input Parameters for WCDMA TM1 Example

3. Select the **CCDF** and **Ext FFT** check boxes.
4. Click the **Create** button. The CCDF and FFT windows display the signal characteristics shown in Figure 31 and Figure 32.

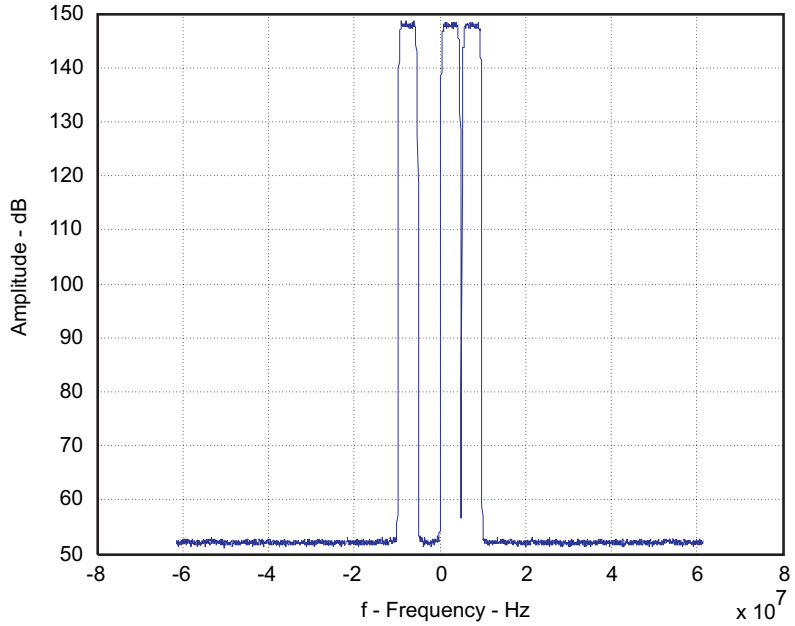

Figure 31. FFT of Three Carrier WCDMA TM1 Pattern

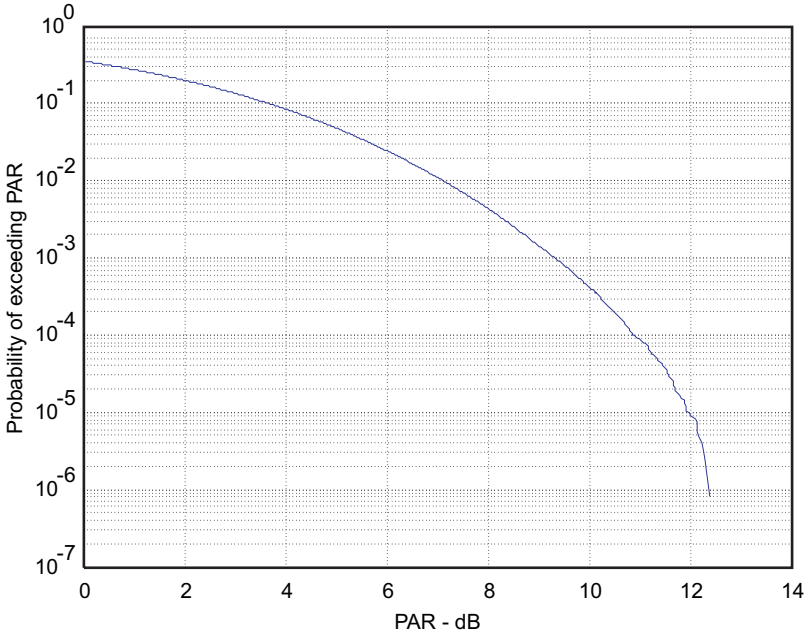

Figure 32. CCDF of Three Carrier WCDMA TM1 Pattern

6.4.2 Download Three Carrier WCDMA TM1 Pattern to TSW3100 / DAC5687 EVM

Download the three carrier WCDMA TM1 example to the TSW3100 and send the pattern to the DAC5687 EVM, which is a CMOS input HS DAC. Following the DAC5687 EVM user's guide:

1. Provide a 491.52 MHz clock to the DAC5687 EVM on CLK2. Connect a SMA-to-SMA cable between J73 (CMOS CLK) of the TSW3100 evm and J2 (PLLLOCK) of the DAC5687 evm.
2. Apply power to the TSW3100 and DAC5687 EVMs. Connect to the host using the procedure in the DAC5687EVM User's Guide.
3. Load the DAC5687 with the following: C:\Program Files\Texas Instruments\TSW3100\Example

Register Files\Example_3.reg5687 This sets the DAC5687 to use quadrature (×4) interpolation and provide an output clock of 122.88 MHz on PLLLOCK OUT. The DAC has its fs/4 mixer enabled, which will quadrature mix the complex signal to 122.88 MHz.

4. Select the *LOAD and Run* check box.
5. Use the **TSW3100 Control** to select the *Master*, *CMOS*, and *Two's Comp* options.
6. Regenerate the pattern by clicking **Create**.

The DAC output spectrum (122.88 ±50 MHz) should display similar to Figure 33.

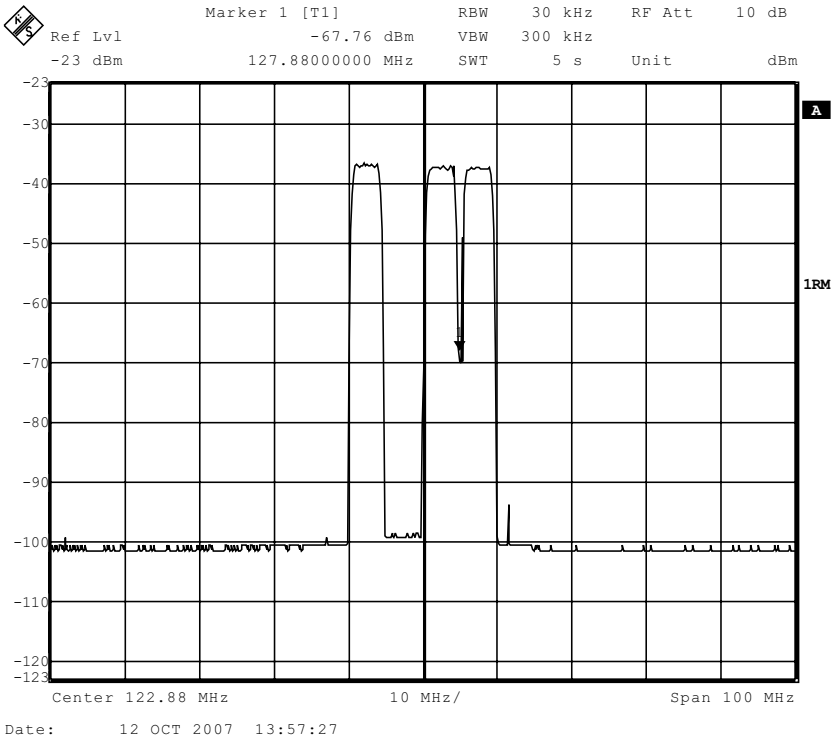

Figure 33. DAC5687 Output Spectrum for WCDMA TM1 Example

6.4.3 Four Carrier QAM256 Pattern

Let's generate a four carrier QAM256 signal, symbol rate of 8 MSPS, 20× oversampled, alpha = 0.12, 1000K vectors, offsets ±5 and 15 MHz, and a real IF with a center frequency of 40 MHz.

1. Select *QAM* from the **Test Models** area.
2. Set the *Chiprate* to 8 MSPS, *Vector Size* to 1000, and *Alpha* to 0.12 in the **Signal Characteristics** area.
3. Set the **Signal Type** to *Real*.
4. Specify a **Center Frequency** of 40 MHz.
5. For **Carriers** 1 through 4, set the *Gains* to 0, –10, –20, and –40 dB, respectfully.
   The GUI interface should looks like Figure 34.

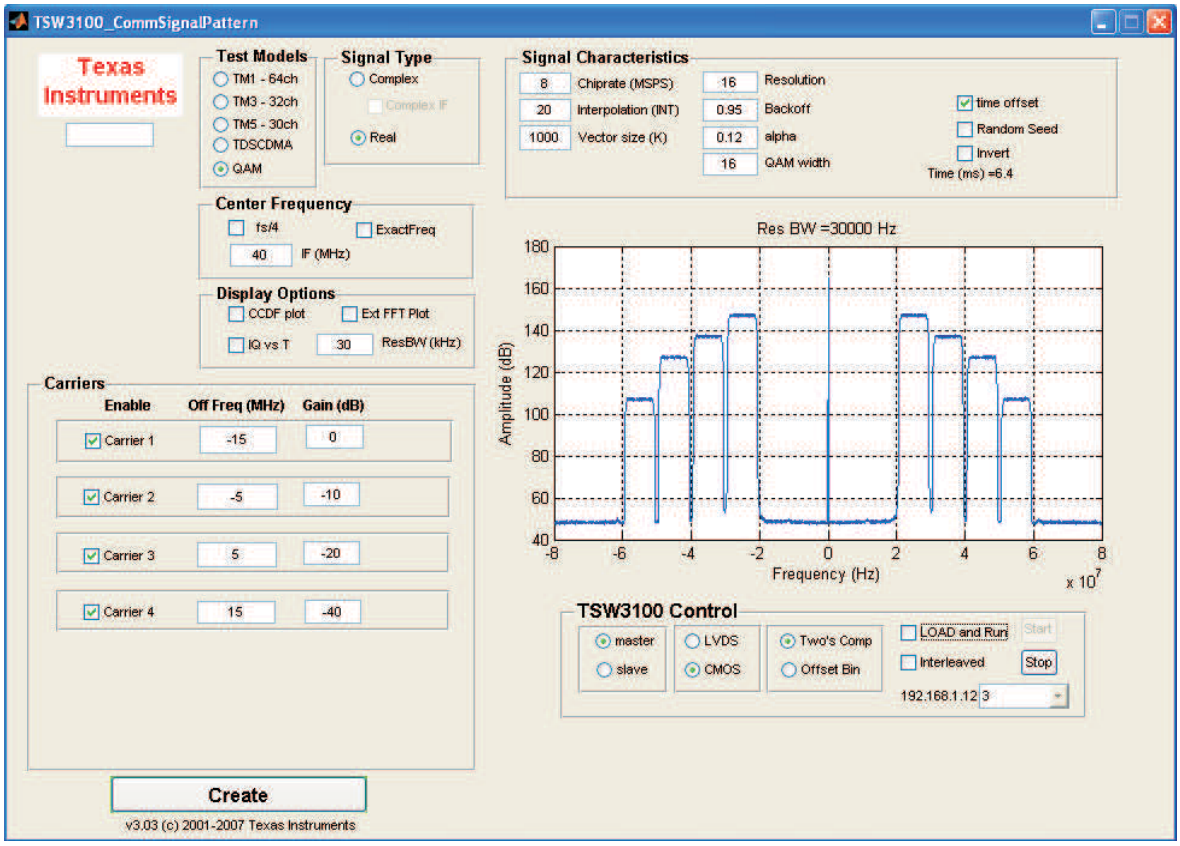

Figure 34. GUI Interface for the Four Carrier QAM256 Pattern

6. Press the **Create** button.

The output FFT should be similar to Figure 35. Note, the spectrum shows the negative frequencies to be a mirror image of the positive frequencies as it is a real signal, rather than a complex signal.

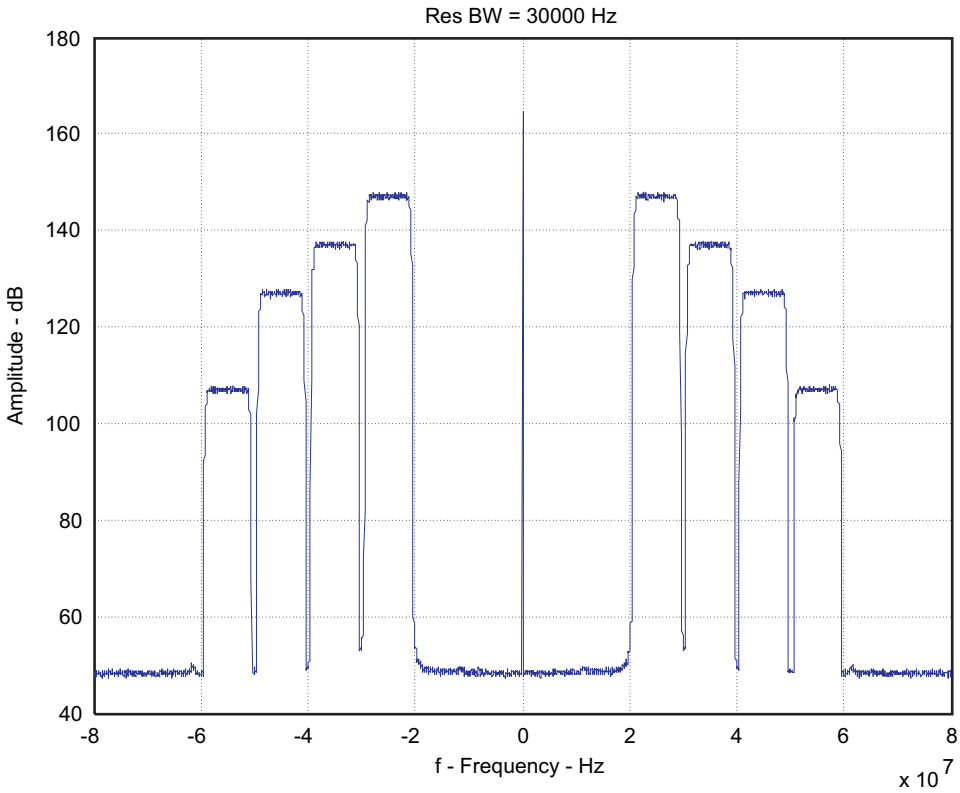

Figure 35. Four Carrier QAM256 Pattern Spectral Plot

6.4.4 Download Four Carrier QAM Pattern to TSW3100 / DAC5687 EVM

Lets download the QAM signal to the TSW3100 and send the pattern to the DAC5687 EVM. Following the DAC5687 EVM user's guide:

1. Provide a 256 MHz clock to the DAC5687 EVM on CLK2.
2. Load the following file: C:\Program Files\Texas Instruments\TSW3100\Example Register Files\ Example_4.reg5687 settings file. This sets the DAC5687 with double interpolation and provides an output clock at 128 MHz on PLLLOCK OUT. In this configuration the DAC is not mixing, and so the DAC output frequency will match the frequency represented in the digital pattern.
3. Select the *LOAD and Run* check box.
4. Use the **TSW3100 Control** to select the *Master*, *CMOS*, and *Two's Comp* options.
5. Regenerate the pattern by clicking **Create**.

The DAC output spectrum (56 ±50 MHz) should display similar to Figure 36.

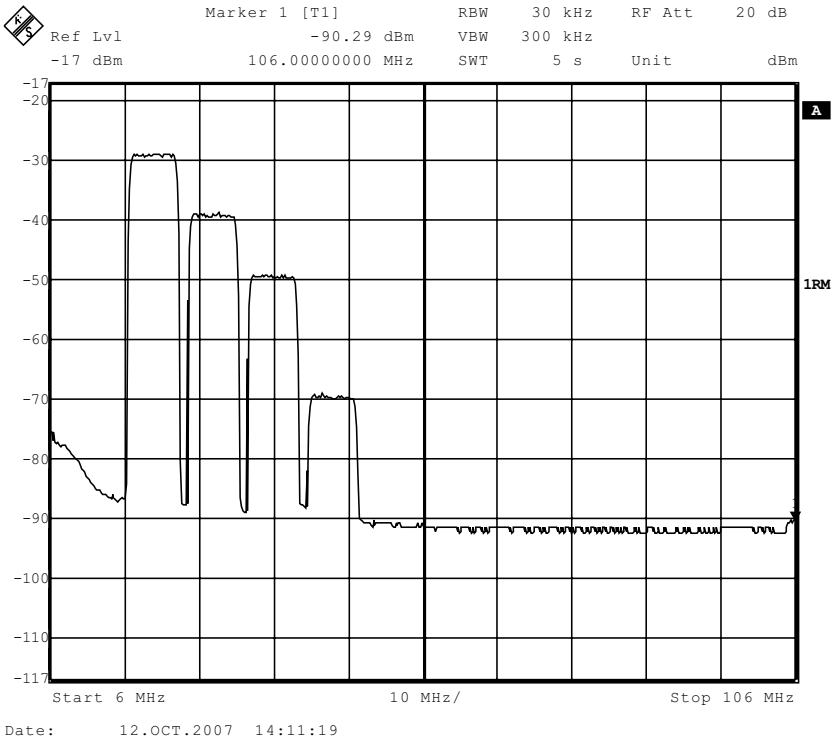

Figure 36. DAC5687 Output Spectrum for Four Carrier QAM256 Pattern

EVALUATION BOARD/KIT IMPORTANT NOTICE

Texas Instruments (TI) provides the enclosed product(s) under the following conditions:

This evaluation board/kit is intended for use for **ENGINEERING DEVELOPMENT, DEMONSTRATION, OR EVALUATION PURPOSES ONLY** and is not considered by TI to be a finished end-product fit for general consumer use. Persons handling the product(s) must have electronics training and observe good engineering practice standards. As such, the goods being provided are not intended to be complete in terms of required design-, marketing-, and/or manufacturing-related protective considerations, including product safety and environmental measures typically found in end products that incorporate such semiconductor components or circuit boards. This evaluation board/kit does not fall within the scope of the European Union directives regarding electromagnetic compatibility, restricted substances (RoHS), recycling (WEEE), FCC, CE or UL, and therefore may not meet the technical requirements of these directives or other related directives.

Should this evaluation board/kit not meet the specifications indicated in the User's Guide, the board/kit may be returned within 30 days from the date of delivery for a full refund. THE FOREGOING WARRANTY IS THE EXCLUSIVE WARRANTY MADE BY SELLER TO BUYER AND IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESSED, IMPLIED, OR STATUTORY, INCLUDING ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE.

The user assumes all responsibility and liability for proper and safe handling of the goods. Further, the user indemnifies TI from all claims arising from the handling or use of the goods. Due to the open construction of the product, it is the user's responsibility to take any and all appropriate precautions with regard to electrostatic discharge.

EXCEPT TO THE EXTENT OF THE INDEMNITY SET FORTH ABOVE, NEITHER PARTY SHALL BE LIABLE TO THE OTHER FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES.

TI currently deals with a variety of customers for products, and therefore our arrangement with the user **is not exclusive.**

TI assumes **no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein.**

Please read the User's Guide and, specifically, the Warnings and Restrictions notice in the User's Guide prior to handling the product. This notice contains important safety information about temperatures and voltages. For additional information on TI's environmental and/or safety programs, please contact the TI application engineer or visit www.ti.com/esh.

No license is granted under any patent right or other intellectual property right of TI covering or relating to any machine, process, or combination in which such TI products or services might be or are used.

FCC Warning

This evaluation board/kit is intended for use for **ENGINEERING DEVELOPMENT, DEMONSTRATION, OR EVALUATION PURPOSES ONLY** and is not considered by TI to be a finished end-product fit for general consumer use. It generates, uses, and can radiate radio frequency energy and has not been tested for compliance with the limits of computing devices pursuant to part 15 of FCC rules, which are designed to provide reasonable protection against radio frequency interference. Operation of this equipment in other environments may cause interference with radio communications, in which case the user at his own expense will be required to take whatever measures may be required to correct this interference.

EVM WARNINGS AND RESTRICTIONS

It is important to operate this EVM within the input voltage range of 5.0 V to 6.0 V and the output voltage range of 1.8 V to 3.3 V.

Exceeding the specified input range may cause unexpected operation and/or irreversible damage to the EVM. If there are questions concerning the input range, please contact a TI field representative prior to connecting the input power.

Applying loads outside of the specified output range may result in unintended operation and/or possible permanent damage to the EVM. Please consult the EVM User's Guide prior to connecting any load to the EVM output. If there is uncertainty as to the load specification, please contact a TI field representative.

During normal operation, some circuit components may have case temperatures greater than 60° C. The EVM is designed to operate properly with certain components above 60° C as long as the input and output ranges are maintained. These components include but are not limited to linear regulators, switching transistors, pass transistors, and current sense resistors. These types of devices can be identified using the EVM schematic located in the EVM User's Guide. When placing measurement probes near these devices during operation, please be aware that these devices may be very warm to the touch.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright 2008, Texas Instruments Incorporated

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 TI 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. Information of third parties may be subject to additional restrictions.

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.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

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
Clocks and Timers	www.ti.com/clocks	Digital Control	www.ti.com/digitalcontrol
Interface	interface.ti.com	Medical	www.ti.com/medical
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
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

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