TI Designs
QVGA 3-D Graphics on MSP430™ Microcontrollers

TI Designs

TI Designs provide the foundation that you need including methodology, testing and design files to quickly evaluate and customize the system. TI Designs help *you* accelerate your time to market.

Design Resources

TIDM-3DGRAPHICS-QVGA	Tool Folder Containing Design Files
MSP430ware™	Product Folder
MSP430F5529 USB LaunchPad™	Product Folder
MSP430FR5969 LaunchPad™	Product Folder
3.5" QVGA LCD BoosterPack™	Product Folder
MSP430F5529	Product Folder
MSP430FR5969	Product Folder

TI E2E™ Community

ASK Our E2E Experts
WEBENCH® Calculator Tools

Design Features

- Real-time 3-D Graphics on a QVGA TFT LCD with 262-K Colors from Kentec Display
- Resistive Touch Screen Interface with Built-In Calibration Routine
- Driver Library and Graphics Library Components Simplify Code Development
- IQmathLib Fixed-Point Software Libraries and MSPMATHLIB Floating-Point Software Libraries Provide Optimized Math Performance on MSP430™ Microcontrollers
- Source Code Contains Projects for Code Composer Studio™ and IAR Embedded Workbench

Featured Applications

- Consumer Electronics
- Appliances
- Electronic Control Panels
- Medical Equipment
- Wearable Electronics

An IMPORTANT NOTICE at the end of this TI reference design addresses authorized use, intellectual property matters and other important disclaimers and information.

1 System Description

This reference design demonstrates how to implement three-dimensional (3-D) graphics on a QVGA LCD using the low-power and performance MSP430F5529 microcontroller as well as the ultra-low-power MSP430FR5969 FRAM microcontroller. This implementation is made possible by using free and optimized software drivers and math libraries available in MSP430ware™. Using the floating-point and fixed-point libraries in MSPMATHLIB and IQmathLib respectively, developers can improve the math performance in MSP430™ and potentially increase battery life in a number of applications.

2 Getting Started

This TI Design demonstrates how to use the MSPMATHLIB and IQmathLib libraries combined with a color QVGA LCD from Kentec Display to perform real-time 3-D graphics on MSP430 devices. This design supports both the MSP430F5529 and MSP430FR5969 LaunchPad™ Development Kits and comes with projects configured for building and loading the application with Code Composer Studio and IAR Embedded Workbench.

Figure 1. MSP430 Accelerated Math Demo

2.1 Code Composer Studio

The Code Composer Studio (CCS) projects can be imported into an existing CCS workspace using the "Project → Import CCS Projects…" menu option shown in Figure 2.

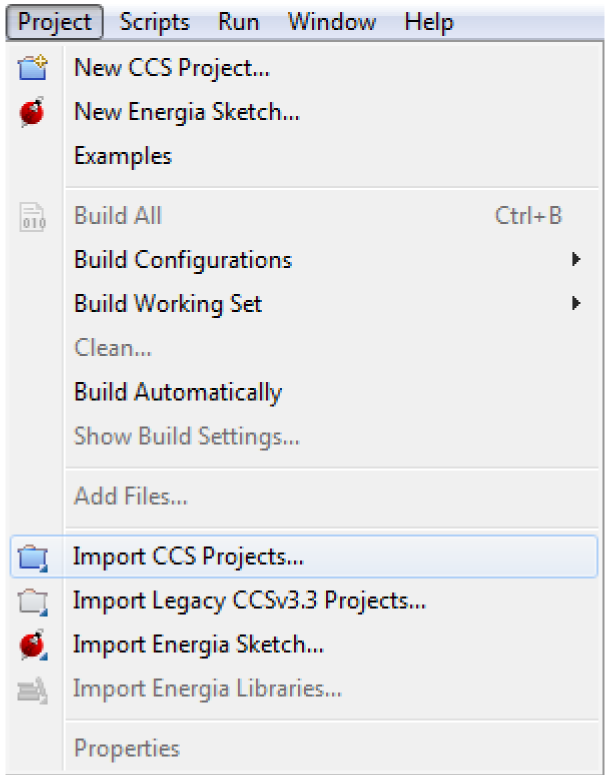

Figure 2. CCS Import Project Menu

Navigate to the installation directory of the design files to auto-discover the projects. Select the desired projects to import and if the projects should be copied to the local workspace, then click Finish to complete the operation. The project-imports window is shown in Figure 3.

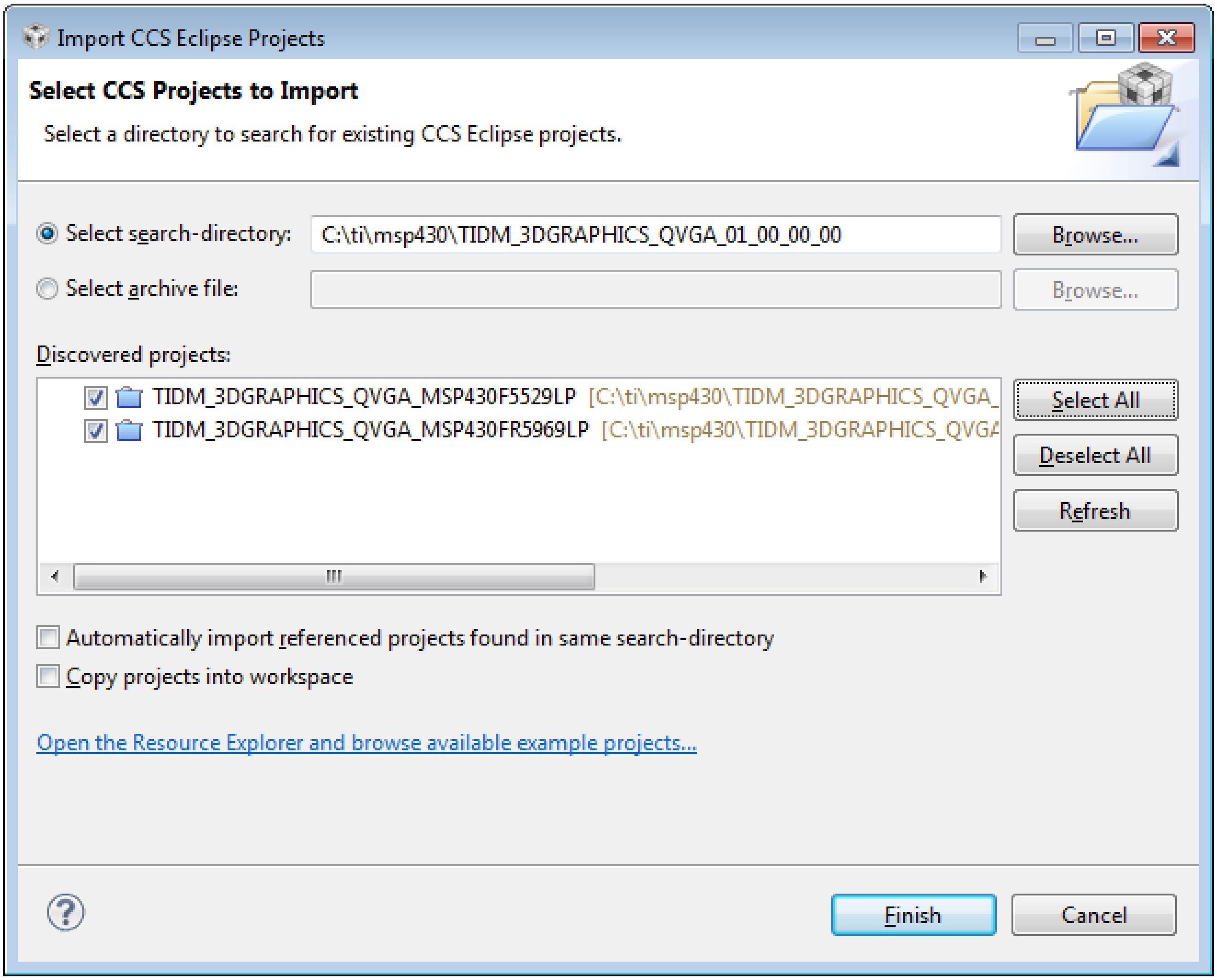

Figure 3. CCS Import Project Settings

2.2 IAR Embedded Workbench

The IAR Embedded Workbench for MSP430 projects come with a configured IAR workspace that can be easily opened in the software. To open the workspace, select the “File → Open → Workspace…” option shown in Figure 4.

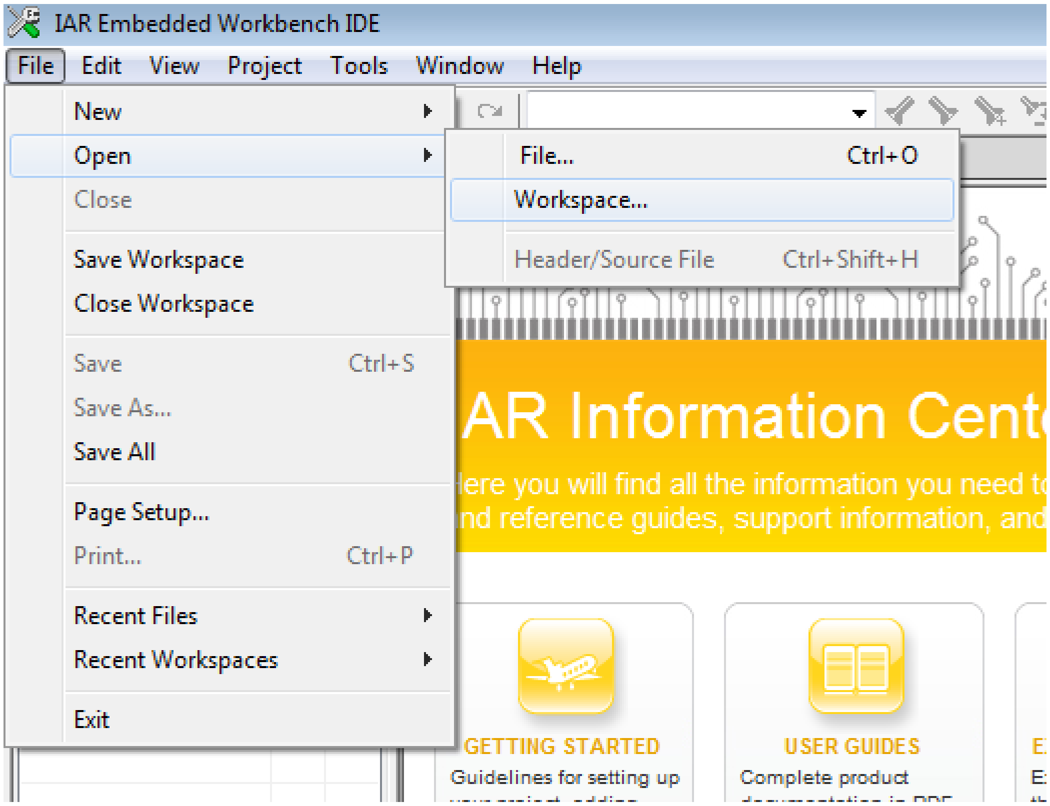

Figure 4. IAR Embedded Workbench Import Workspace Menu

Navigate to the directory containing the installed design files and open the included IAR .eww workspace file. This action will open the IAR workspace as well as the two associated projects shown in Figure 5. These projects can be used to build and load the TI Design to the LaunchPad.

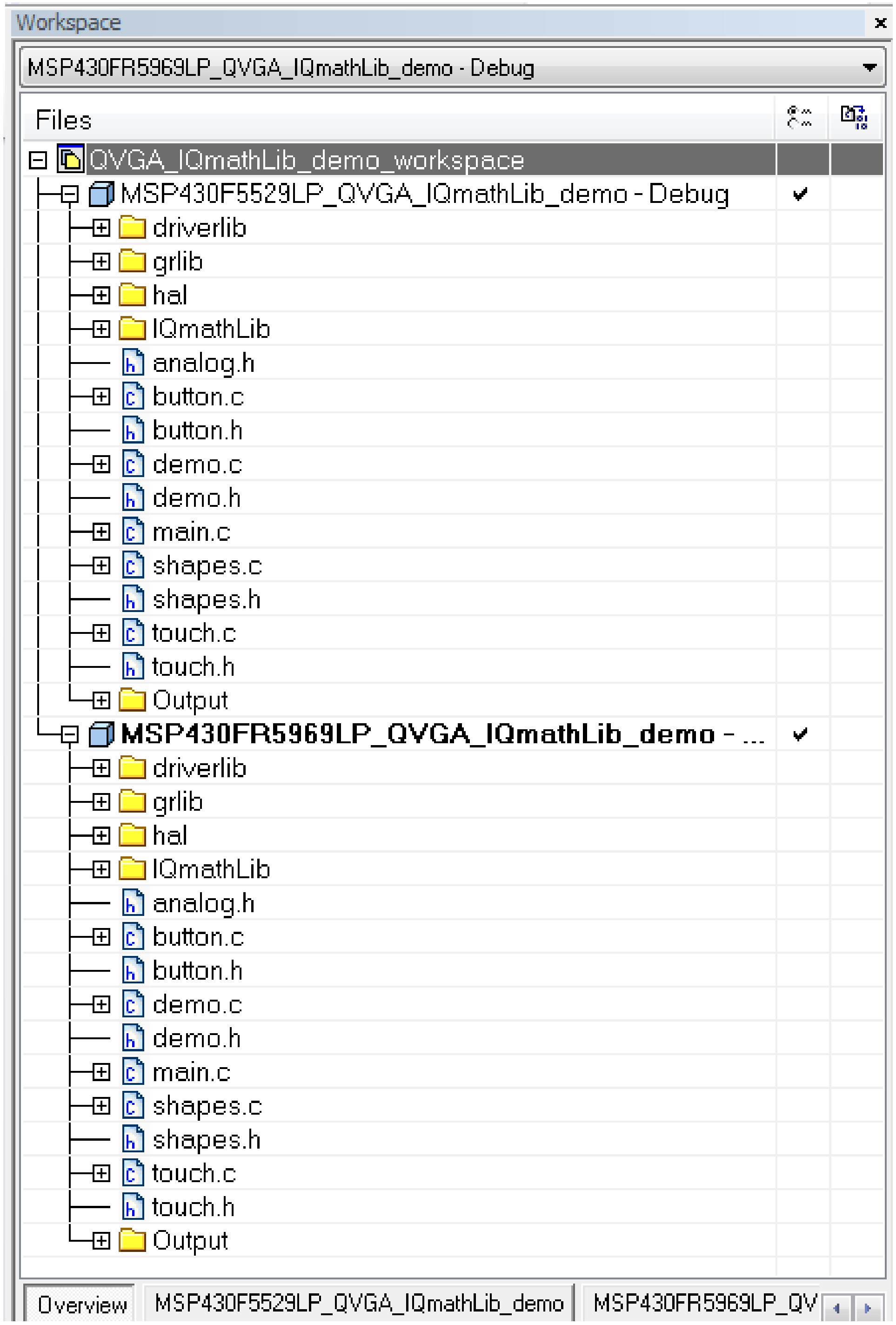

Figure 5. IAR Embedded Workbench Workspace with Design Projects

2.3 Calibration

When the application is first initialized on a device, the resistive touch screen must be calibrated. Follow the on-screen directions and touch each red dot in the four corners of the LCD. Once all four values have been measured, the calibration values are calculated and saved to the information memory, and the demo will begin.

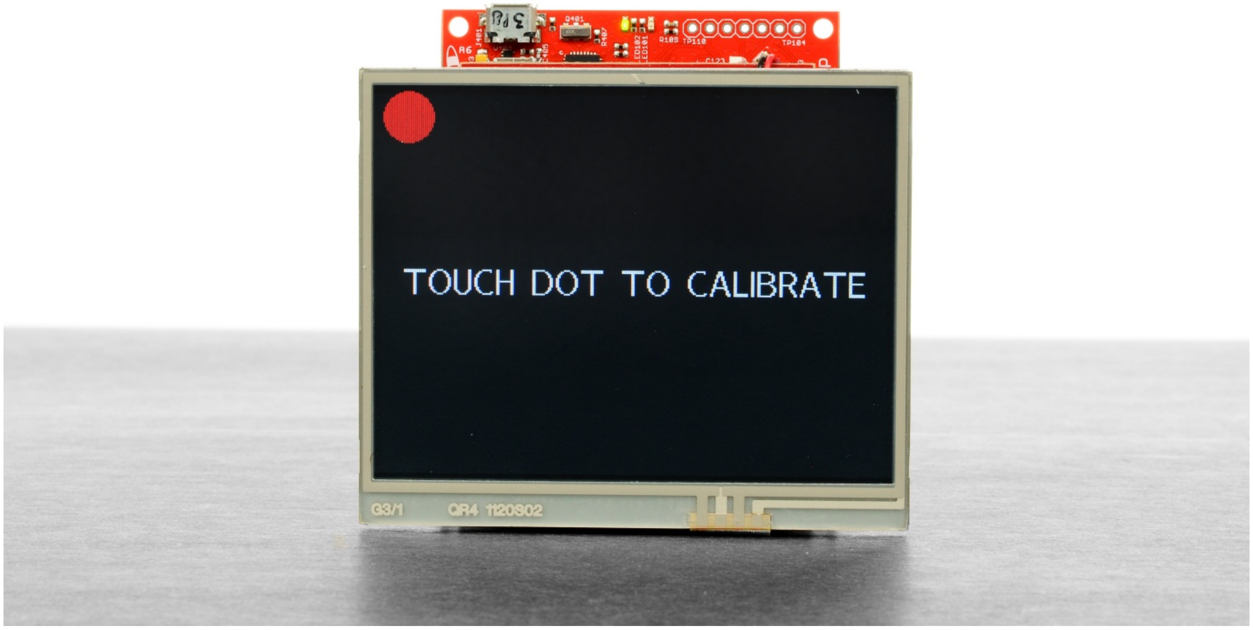

Figure 6. Built-in Touch Screen Calibration Routine

On future device initialization, the demo will read the calibration data stored in the **information memory** and skip the calibration routine. To manually overwrite the calibration data, press and hold the left switch button labeled S1 on the LaunchPad during device initialization. Once the device is operating, wait until the screen has been cleared to black and release the switch button. The calibration values stored in the information memory will be erased and the calibration routine will begin.

3 Design Hardware

This design uses the QVGA LCD BoosterPack™ from Kentec Display and is provided with software support for both the MSP430F5529 USB LaunchPad and the MSP430FR5969 FRAM LaunchPad. Additionally, this TI Design can be combined with the FuelTank BoosterPack to provide battery power and wireless operation.

3.1 MSP430F5529 LaunchPad Development Kit

The MSP-EXP430F5529LP LaunchPad (or the F5529 LaunchPad) is an inexpensive, simple evaluation module for the MSP430F5529 USB-enabled microcontroller. The LaunchPad offers quick evaluation of the MSP430 microcontroller, with an on-board emulator for programming and debugging, as well as buttons and LEDs for a simple user interface.

The 40-pin BoosterPack expansion headers allow for rapid prototyping, as well as a wide range of available BoosterPack plug-in modules. Other features such as wireless displays and sensors can also be added. Users can design their own BoosterPack or choose among many already available from TI and elsewhere. The 40-pin interface is compatible with any 20-pin BoosterPack that is compliant with the standard.

For more information, please visit the product page at http://www.ti.com/tool/msp-exp430f5529lp.

3.2 MSP430FR5969 LaunchPad Development Kit

The MSP-EXP430FR5969 LaunchPad development kit is an easy-to-use evaluation module for the MSP430FR5969 microcontroller. The kit contains everything needed to start developing on MSP430’s ultra-low-power FRAM platform, including on-board emulation for measuring energy, programming, and debugging. The board features on-board buttons and LEDs for quick integration of a simple user interface as well as a Super Cap, which allows applications to stand alone without an external power supply. The MSP430FR5969 device features embedded Ferroelectric Random Access Memory (FRAM), a non-volatile memory known for its ultra-low power, high endurance, and high write speeds.

For more information, please visit the product page at http://www.ti.com/tool/msp-exp430fr5969.

3.3 3.5” QVGA LCD BoosterPack

The BoosterPack includes a 3.5-inch QVGA TFT LCD module with a built-in LED backlight driver circuit and resistive touch screen. The 320-by-240 resolution display is driven by a SSD2119 controller with integrated power circuits and GDDRAM. A display driver is available for use with the software graphics library in MSP430ware.

For more information, please visit the product page at http://www.ti.com/tool/BOOSTXL-K350QVG-S1.

3.3.1 MSP430FR5969 LaunchPad: Hardware Changes

The MSP430FR5969 LaunchPad uses the 20-pin BoosterPack standard and requires a hardware modification to the QVGA BoosterPack. To enable operation with the 20-pin LaunchPad, the following pins on the BoosterPack must be routed from the inner row (J3/J4) of the BoosterPack header to the outer row (J1/J2). The table below lists all of the pins that must be modified.

Table 1. Pins for Modification

FUNCTIONALITY	MSP430FR5969 LP	BOOSTERPACK PIN (J3/J4)	BOOSTERPACK PIN (J1/J2)
GND	GND	22	20
PWM	P1.2	40	19
Y+	P2.4	23	6
X+	P4.2	24	2
X-	P1.4	31	12
LCD_RESET	P3.0	32	18

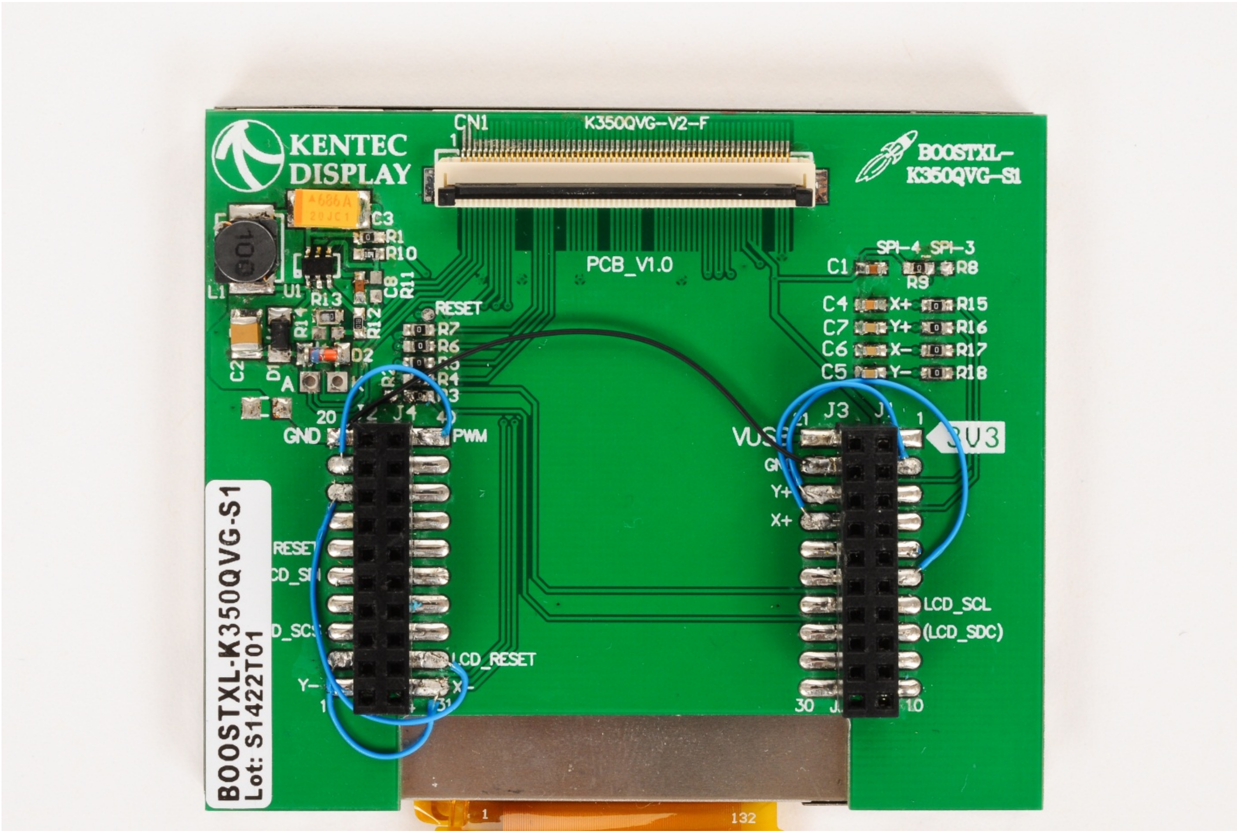

Figure 7. MSP430FR5969 BoosterPack Hardware Modifications

Additionally, the 5-V pin next to VCC on the MSP430FR5969 LaunchPad will need to be populated to supply 5 V to the display.

3.4 FuelTank Battery BoosterPack

The Fuel Tank BoosterPack from element14 and Texas Instruments allow LaunchPads to be powered from a rechargeable lithium polymer battery BoosterPack, further enabling mobile application development and evaluation.

The FuelTank BoosterPack works with TI LaunchPads and includes an on-board lithium polymer battery charger and gas gauge that can output the critical parameters of the battery, including temperature, charging state, capacity, and more. The BoosterPack comes with a lithium polymer battery cell, which is charged through an on-board USB connector.

For more information, please visit the product page at http://www.ti.com/tool/boostxl-battpack.

4 Design Software

4.1 Libraries

This design uses several of the software libraries provided in MSP430ware to enable quick software development. These components have been included in the software design files but can also be downloaded together with the MSP430ware package or downloaded individually through the TI tool pages linked below.

The latest version of MSP430ware can be downloaded from http://www.ti.com/tool/msp430ware.

4.1.1 Driverlib

Driverlib provides a full API for selected MSP430 device families for configuring, enabling, and using integrated peripherals. Each API function is fully documented through a user's guide, API guide, and code examples. Driver Library for MSP430 uses easy-to-understand function calls, which helps users master MCUs and get to market faster.

The latest version of Driver Library for MSP430 can be downloaded from http://www.ti.com/tool/msp430driverlib.

4.1.2 GrLib

Graphics Library offers an easy-to-use API for rapid development with any LCD paired with MSP430. The consistency across Graphics Library allows the same application code to be used across all LCD types. MSP430 Graphics Library is powerful enough to run a QVGA while using low power.

The latest version of GrLib for MSP430 can be downloaded from http://www.ti.com/tool/MSP430-GRLIB.

4.1.3 IQ Math Lib

The TI MSP430 IQmath and Qmath Libraries are a collection of highly optimized and high-precision mathematical functions for C programmers to seamlessly port a floating-point algorithm into fixed-point code on MSP430 devices. These routines are typically used in computationally intensive real-time applications where optimal execution speed, high accuracy, and ultra-low energy are critical. By using the IQmath and Qmath libraries, users can achieve execution speeds considerably faster and energy consumption considerably lower than equivalent code written using floating-point math.

The latest version of IQmathLib can be downloaded from http://www.ti.com/tool/msp430-iqmathlib.

4.1.4 MSPMATHLIB

MSPMATHLIB is an accelerated floating-point math library for MSP430 devices. This library replaces many of the functions found in the standard “math.h” header file including square root, trigonometric, exponential functions, and more.

MSPMATHLIB is integrated into the latest versions of Code Composer Studio, and IAR Embedded Workbench and does not require a separate download.

4.2 Files

4.2.1 Main

This file contains the main routine and makes calls to the other files included in the project. The main function will initialize the device, peripheral, and display. An infinite loop will continuously update the 3-D graphics, statistics on the display, and the graphical user interface when one of the five buttons is selected.

4.2.2 Demo

This file contains routines to update the display with one of three shapes using either floating-point or fixed-point math. Routines are called to rotate the 3-D or 4-D data arrays and then draw them to the LCD. The data arrays consist of multiple lines where each line is defined by two coordinate points at either end. During a line's rotation, both coordinate points are rotated around the origin. After rotating all of the lines, the coordinate points are drawn to the LCD using the line-draw functions provided by GrLib.

To rotate the coordinate points, calculate the sine and cosine terms by using MSPMATHLIB for the floating-point routines and IQmathLib for the fixed-point routines. In addition, the IQmathLib is used for many of the conversion functions when mapping the 2-D image for the LCD.

4.2.2.1 Cube and Dodecahedron

The cube and dodecahedron shapes are three-dimensional and consist of x, y, and z coordinate points. The constant data arrays are rotated around the origin by angles θ_x, θ_y, and $\theta_z$by calculating the rotation matrix for each angle and multiplying each coordinate point by the resulting matrices. The angles are incremented with each frame to achieve the rotation over time. The rotation matrix for each angle is defined below.

$$R_x(\theta_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta_x) & -sin(\theta_x) \\ 0 & sin(\theta_x) & cos(\theta_x) \end{bmatrix} \tag{1}$$

$$R_y(\theta_y) = \begin{bmatrix} cos(\theta_y) & 0 & sin(\theta_y) \\ 0 & 1 & 0 \\ -sin(\theta_y) & 0 & cos(\theta_y) \end{bmatrix} \tag{2}$$

$$R_z(\theta_z) = \begin{bmatrix} cos(\theta_z) & -sin(\theta_z) & 0 \\ sin(\theta_z) & cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

The matrix multiplication routine has been simplified for each axis and is implemented in the code as shown below.

$$R_x(\theta_x)\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta_x) & -sin(\theta_x) \\ 0 & sin(\theta_x) & cos(\theta_x) \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x \\ y*cos(\theta_x) - z*sin(\theta_x) \\ y*sin(\theta_x) + z*cos(\theta_x) \end{bmatrix} \tag{4}$$

$$R_y(\theta_y)\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} cos(\theta_y) & 0 & sin(\theta_y) \\ 0 & 1 & 0 \\ -sin(\theta_y) & 0 & cos(\theta_y) \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x*cos(\theta_y) + z*sin(\theta_y) \\ y \\ -x*sin(\theta_y) + z*cos(\theta_y) \end{bmatrix} \tag{5}$$

$$R_z(\theta_z)\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} cos(\theta_z) & -sin(\theta_z) & 0 \\ sin(\theta_z) & cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x*cos(\theta_z) - y*sin(\theta_z) \\ x*sin(\theta_z) + y*cos(\theta_z) \\ z \end{bmatrix} \tag{6}$$

Each coordinate point is multiplied by the three rotation matrices to obtain the rotated three-dimensional shape. The rotated shape is projected to two dimensions when drawing to the display by zeroing all z-axis values.

4.2.2.2 Hypercube

The hypercube is a four-dimensional analogue of a cube, also commonly referred to as a tesseract. The four-dimensional coordinate points consist of the same x-, y-, and z-axes with an added w-axis to create the fourth dimension. During rotations in four dimensions, a coordinate point is rotated around a plane defined by two of the axes. The hypercube demo performs two rotations: one along the XY plane and one along the ZW plane by angles θ_{xy} and θ_{zw} respectively. The rotation matrix for each angle is defined in the following equations.

$$R_{xy}(\theta_{xy}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & cos(\theta_{xy}) & -sin(\theta_{xy}) \\ 0 & 0 & sin(\theta_{xy}) & cos(\theta_{xy}) \end{bmatrix} \tag{7}$$

$$R_{zw}(\theta_{zw}) = \begin{bmatrix} cos(\theta_{zw}) & -sin(\theta_{zw}) & 0 & 0 \\ sin(\theta_{zw}) & cos(\theta_{zw}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (8)$$

The matrix multiplication routine has been simplified for each axis and is implemented in the code as shown below.

$$R_{xy}(\theta_{xy})\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & cos(\theta_{xy}) & -sin(\theta_{xy}) \\ 0 & 0 & sin(\theta_{xy}) & cos(\theta_{xy}) \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} x \\ y \\ z * cos(\theta_{xy}) - w * sin(\theta_{xy}) \\ z * sin(\theta_{xy}) + w * cos(\theta_{xy}) \end{bmatrix} \quad (9)$$

$$R_{zw}(\theta_{zw})\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} cos(\theta_{zw}) & -sin(\theta_{zw}) & 0 & 0 \\ sin(\theta_{zw}) & cos(\theta_{zw}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} x * cos(\theta_{zw}) - y * sin(\theta_{zw}) \\ x * sin(\theta_{zw}) + y * cos(\theta_{zw}) \\ z \\ w \end{bmatrix} \quad (10)$$

The 4-D result of the rotation operation described above is then projected to three dimensions. This operation is similar in concept to flattening the 3-D image to 2-D when drawing to the display. Because simply zeroing the w-axis values would cause lines to overlap, instead we scale x, y, and z values by w as shown in the following equation.

$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \rightarrow \begin{bmatrix} 2 * (x/(w+3)) \\ 2 * (y/(w+3)) \\ 2 * (z/(w+3)) \end{bmatrix} \quad (11)$$

Finally, the resulting three-dimensional shape is rotated by 30° along the y-axis to give a better view of the result. The 3-D shape is again projected to two dimensions when drawing to the display by zeroing all z-axis values. The result of the 4-D and 3-D rotations gives the impression that the hypercube is rotating around its center while simultaneously turning inside out.

4.2.3 Shapes

This file contains the constant data arrays for each of the three shapes with two arrays per shape for fixed-point and floating-point formats.

4.2.4 Buttons

This file implements common functions for drawing a button to the LCD.

4.2.5 Touch

This file implements functions to initialize the touch interface as well as update the current context of the touch screen. A hardware layer implements ADC12_A and ADC12_B analog functions to read the touch-screen values for MSP430F5529 and MSP430FR5969 devices respectively.

5 Design Benchmarks

The included applications have been compiled for Code Composer Studio and IAR Embedded Workbench for MSP430 using the following compiler optimization settings:

- Code Composer Studio
  - -03
  - --opt_for_speed=5
  - --code_model=small
  - --data_model=small
- IAR Embedded Workbench for MSP430
  - -0hs
  - --data_model=small
  - --code_model=small

5.1 Timing Measurements

The benchmarks below represent the time needed to calculate the rotations using either floating-point math with MSPMATHLIB or fixed-point math with IQmathLib, and the time required to draw to the display (this is identical for both methods). The total represents the sum of the calculation time and the display draw time, and is how long it takes to update the entire frame. Each measurement is taken as an average of eight frames, and the fastest average is recorded below.

The benchmarks demonstrate the advantage of using fixed-point over floating-point math on MSP430. Leveraging the IQmathLib functions and native fixed-point CPU instructions can accelerate highly complex math operations by up to 1200%.

5.1.1 MSP430F5529

Table 2 and Table 3 were measured with MCLK set to 8 MHz and represent the measured time in milliseconds.

Table 2. Floating-Point with MSPMATHLIB

	CODE COMPOSER STUDIO			IAR EMBEDDED WORKBENCH		
	Cube	**Dodecahedron**	**Hypercube**	**Cube**	**Dodecahedron**	**Hypercube**
Demo	18.71	42.06	60.94	13.14	27.45	43.76
Display	18.94	32.85	44.27	18.23	31.31	42.65
Total	37.65	74.91	105.21	31.37	58.76	86.41

Table 3. Fixed-Point with IQmathLib

	CODE COMPOSER STUDIO			IAR EMBEDDED WORKBENCH		
	Cube	**Dodecahedron**	**Hypercube**	**Cube**	**Dodecahedron**	**Hypercube**
Demo	1.59	3.53	7.51	1.36	2.94	6.98
Display	18.94	32.85	44.27	18.23	31.31	42.65
Total	20.53	36.38	51.78	19.59	34.25	49.63

5.1.2 MSP430FR5969

Table 4 and Table 5 were measured with MCLK set to 8 MHz using zero-FRAM wait states and represent the measured time in milliseconds.

Table 4. Floating-Point with MSPMATHLIB

	CODE COMPOSER STUDIO			IAR EMBEDDED WORKBENCH		
	Cube	Dodecahedron	Hypercube	Cube	Dodecahedron	Hypercube
Demo	18.71	42.06	60.94	13.14	27.45	43.76
Display	17.37	31.65	42.28	16.47	30.35	40.85
Total	36.08	73.71	103.22	29.61	57.8	84.61

Table 5. Fixed-Point with IQmathLib

	CODE COMPOSER STUDIO			IAR EMBEDDED WORKBENCH		
	Cube	Dodecahedron	Hypercube	Cube	Dodecahedron	Hypercube
Demo	1.59	3.53	7.51	1.36	2.94	6.98
Display	17.37	31.65	42.28	16.47	30.35	40.85
Total	18.96	35.18	49.79	17.83	33.29	47.83

5.2 Code and Data Size

Code and data-size measurements in bytes are presented in Table 6. Data variables are allocated into the device RAM while both code and constant data use flash or FRAM on MSP430F5529 and MSP430FR5969 respectively.

Table 6. Code and Data Size Measurements

	CODE COMPOSER STUDIO			IAR EMBEDDED WORKBENCH		
	Code	Data	Constant	Code	Data	Constant
MSP430F5529	25476	1626	7530	24860	1628	7508
MSP430FR5969	25048	1722	7550	25320	1628	7508

6 Design Files

6.1 Schematics

To view the schematics for this document, please see the files at http://www.ti.com/tool/TIDM-3DGRAPHICS-QVGA.

6.2 Layer Plots

To download the layer plots, see the design files at http://www.ti.com/tool/TIDM-3DGRAPHICS-QVGA.

6.3 CAD Files

To download the CAD files, see the design files at http://www.ti.com/tool/TIDM-3DGRAPHICS-QVGA.

6.4 Gerber Files

To download the Gerber files, see the design files at http://www.ti.com/tool/TIDM-3DGRAPHICS-QVGA.

6.5 Software Files

To download the software files, see the design files at http://www.ti.com/tool/TIDM-3DGRAPHICS-QVGA

7 About the Author

BRENT PETERSON is an MSP430 software applications engineer at Texas Instruments. Brent is responsible for developing ultra-low-energy math libraries for existing and next-generation devices and is the lead software developer for the MSPMathlib and IQmathLib software libraries. Brent joined TI in 2011 after graduating from Rensselaer Polytechnic Institute with a degree in computer systems engineering.

IMPORTANT NOTICE FOR TI REFERENCE DESIGNS

Texas Instruments Incorporated ("TI") reference designs are solely intended to assist designers ("Buyers") who are developing systems that incorporate TI semiconductor products (also referred to herein as "components"). Buyer understands and agrees that Buyer remains responsible for using its independent analysis, evaluation and judgment in designing Buyer's systems and products.

TI reference designs have been created using standard laboratory conditions and engineering practices. **TI has not conducted any testing other than that specifically described in the published documentation for a particular reference design.** TI may make corrections, enhancements, improvements and other changes to its reference designs.

Buyers are authorized to use TI reference designs with the TI component(s) identified in each particular reference design and to modify the reference design in the development of their end products. HOWEVER, NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY THIRD PARTY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT, IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license 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.

TI REFERENCE DESIGNS ARE PROVIDED "AS IS". TI MAKES NO WARRANTIES OR REPRESENTATIONS WITH REGARD TO THE REFERENCE DESIGNS OR USE OF THE REFERENCE DESIGNS, EXPRESS, IMPLIED OR STATUTORY, INCLUDING ACCURACY OR COMPLETENESS. TI DISCLAIMS ANY WARRANTY OF TITLE AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, QUIET ENJOYMENT, QUIET POSSESSION, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS WITH REGARD TO TI REFERENCE DESIGNS OR USE THEREOF. TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY BUYERS AGAINST ANY THIRD PARTY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON A COMBINATION OF COMPONENTS PROVIDED IN A TI REFERENCE DESIGN. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, SPECIAL, INCIDENTAL, CONSEQUENTIAL OR INDIRECT DAMAGES, HOWEVER CAUSED, ON ANY THEORY OF LIABILITY AND WHETHER OR NOT TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, ARISING IN ANY WAY OUT OF TI REFERENCE DESIGNS OR BUYER'S USE OF TI REFERENCE DESIGNS.

TI reserves the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI's terms and conditions of sale of semiconductor products. Testing and other quality control techniques for TI components are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

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

Reproduction of significant portions of TI information in TI data books, data sheets or reference designs is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards that anticipate dangerous failures, monitor failures and their consequences, lessen the likelihood of dangerous failures and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in Buyer's safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI's goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed an agreement specifically governing such use.

Only those TI components that TI has specifically designated as military grade or "enhanced plastic" are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components that have ***not*** been so designated is solely at Buyer's risk, and Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

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