



APPLICATION NOTE 4065

Implementing a MAX1385-Based Control Loop in C/C++

Abstract: Maxim's MAX1385 evaluation kit (EV kit) software includes a Windows® graphical user interface (GUI) program; however, the time cost of updating this display interferes with the control loop. This application note shows a more optimal control-loop program, using a console menu system instead of a GUI.

When used with the MAX1385EVKIT+ demo board, the control loop achieves a regulation accuracy of $\pm 2\%$. This accuracy is limited by the gate driver output step size and the FET transconductance. Drain-current regulation step size is determined by the MAX1385's gate-voltage increment multiplied by the FET's effective transconductance. Because the MAX1385EVKIT uses an IRFZ44N MOSFET to close the loop for demonstration purposes, the regulation may not be the same as it would be when used with an LDMOS FET.

Required Hardware

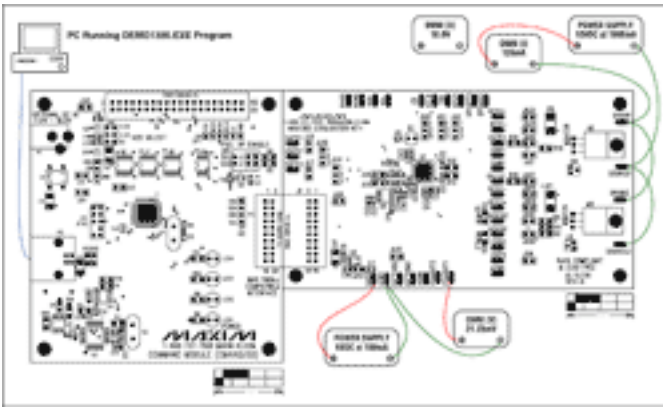
- Maxim MAX1385EVKIT+
- Maxim CMAXQUSB+ (includes USB A-B cable)
- Windows 2000/XP PC with USB port
- 5VDC at 100mA power supply
- 10VDC at 1000mA power supply
- DMM for measuring drain current
- DMM for measuring drain voltage
- DMM for measuring PGAOUT amplified current-sense voltage
- Optional: oscilloscope for monitoring GATE1 voltage and PGAOUT1 drain current

Setup

Download and unzip the necessary [executable and source code files](#) (ZIP, 736kB).

Assemble the hardware according to **Figure 1**.

- Plug the CMAXQUSB header P3 into the MAX1385EVKIT connector J1.
- Connect the MAX1385EVKIT's DRAIN1 and DRAIN2 pins to the current meter (-).
- Connect the current meter (+) to the power supply (+).
- Connect the MAX1385EVKIT's SOURCE1 and SOURCE2 pins to the power supply (-).
- Connect the voltage meter (+) to the MAX1385EVKIT's DRAIN1 pin.
- Connect the voltage meter (-) to the MAX1385EVKIT's SOURCE1 pin.
- Connect the MAX1385EVKIT's AVDD pin to the DVDD pin (or, optionally, connect to an external 5V DC supply).



[More Detailed Image](#) (PDF, 387kB)

Figure 1. MAX1385EVKIT hardware configuration.

Procedure

Set the CMAXQUSB's **VDD Select** jumper to the 5V position.

Connect the CMAXQUSB to the PC's USB port. If this is the first time a CMAXQUSB has been attached to the PC, the plug-and-play wizard will appear. Guide the GUI to the installed location of the device driver, in **MAX1385_Apnote_src.zip\src\USB_driver**.

Start the DEMO1385.EXE program. A console will appear on the screen. Enter the following series of commands at the console:

Command	Action
C	Connects to the CMAXQUSB module. Verify that the software reports: Board connected. Got board banner: Maxim CMAXQUSB V01.04.32 > Searching for MAX1385... Found MAX1385 at 0x4e Note: when using MAX1385EVKIT with CMAXQUSB, connect 5V DVDD supply to AVDD.
T V P	Test menu/verify power-up values
T S O FCT1 0300	Test menu/servo mode/output register/FineCalThru1 register, initial value 0x0300
T S I F F	Test menu/servo mode/input register/FIFO register
T S A 2	Test menu/servo mode/ADC command/trigger channel 2 (current CS1)
T S T 0020	Test menu/servo mode/target value 0x0020
T S C 1	Test menu/servo mode/convergence step positive 1
T S H 1	Test menu/servo mode/hysteresis one step
T S M 60000	Test menu/servo mode/maximum loop duration set to 60 seconds
T S R	Test menu/servo mode/run
T W FCT1 0300	Test menu/write register/FineCalThru1 register, value 0x0300

Monitor the regulation by watching the DMM.

The voltage on PGAOUT1 regulates between 20.8mV and 21.7mV, which is a variation of 0.45mV (2%) around an average of 21.25mV.

Source Code Walk-Through

The source code was developed with the free dev-cpp IDE, which uses the GNU gcc-3.4.2 C++ compiler.

Listing 1 shows a simplified version of the C++ code performed inside the regulation loop. Output statements and error handling have been removed for clarity.

Listing 1. Simplified C++ code.

```
// read InputRegAddr8 into DeviceDataBuf16
if (InputRegAddr8 == MAX1385::rd_FIFO) {
    myDevice.ReadWord_HiLo(InputRegAddr8, &DeviceDataBuf16);
    if ((DeviceDataBuf16 & MAX1385::FIFO_Tag_Mask) == MAX1385::FIFO_F) {
        // FIFO contains FLAG, not the channel we were looking for.
        if ((DeviceDataBuf16 & MAX1385::FLAG_ADCBUSY) == 0) {
            // trigger another conversion
            myDevice.WriteWord_HiLo(MAX1385::wr_ADCCON,
                (1 << ADCCONselectedBit_0_6));
        }
        continue; // this C++ statement means go back to the start of the loop
    }
    // FIFO contains some channel data. Assume it's the correct channel.
    DeviceDataBuf16 &= MAX1385::FIFO_Data_Mask;
} else {
    // our input register is not the FIFO, so we just read the register directly
    myDevice.ReadWord_HiLo(InputRegAddr8, &DeviceDataBuf16);
}
// compare DeviceDataBuf16 to InputTargetData16, applying Hysteresis: adjust OutputData16
if (DeviceDataBuf16 > InputTargetData16 + Hysteresis) {
    OutputData16 -= ConvergeStep;
}
else if (DeviceDataBuf16 < InputTargetData16 - Hysteresis) {
    OutputData16 += ConvergeStep;
}
// Write the output value
myDevice.WriteWord_HiLo(OutputRegAddr8, OutputData16);
```

Menu System

The complete source code implements the console menu system seen in **Listing 2**, which connects to the CMAXQUSB module.

Listing 2. Console menu system.

```
=====
CmodComm test program main menu when not connected
A) adjust timing parameters
L) CmodLog... functions
C) connect
```

- D) Debug Messages
- X) exit

C

Board connected.

Got board banner: Maxim CMAXQUSB V01.04.32 >

Searching for MAX1385...

Found MAX1385 at 0x4e

Note: when using MAX1385EVKIT with CMAXQUSB,
connect 5V DVDD supply to AVDD.

=====
CmodComm test program main menu after successful connect

- T) Test the device
- 8) CmodP8Bus... functions
- A) adjust timing parameters
- L) CmodLog... functions
- P) CmodPin... functions
- S) CmodSpi... functions
- M) CmodSMBus... functions
- \$) CmodCommStringWrite list of hex codes
- R) CmodBoardReset
- D) Disconnect

=====
T Test menu

- T ? Hunt for active devices
- T R Read register
- T W Write register
- T S Servo loop
- T VP Verify Power-On Register Values
- T VM reg mask Verify Register Memory Persistence, All Combinations ...
- T VW reg mask Verify Register Memory Persistence, Walking-One's test ...

=====
Write register:

- T W AD Write ADCCON
- T W AH Write ALMHCFG
- T W AS Write ALMSCFG
- T W FI1 Write FINE1
- T W FI2 Write FINE2
- T W FC1 Write FINECAL1
- T W FC2 Write FINECAL2
- T W FCT1 Write FINECALTHRU1
- T W FCT2 Write FINECALTHRU2
- T W FT1 Write FINETHRU1
- T W FT2 Write FINETHRU2
- T W HC Write HCFG
- T W HT1 Write THRUHI1
- T W HT2 Write THRUHI2
- T W HW1 Write HIWIPE1
- T W HW2 Write HIWIPE2
- T W IH1 Write IH1

```

T W IH2           Write IH2
T W IL1           Write IL1
T W IL2           Write IL2
T W LD            Write LDAC
T W LT1           Write THRULO1
T W LT2           Write THRULO2
T W LW1           Write LOWIPE1
T W LW2           Write LOWIPE2
T W P             Write PGACAL
T W SC            Write SCLR
T W SS            Write SSHUT
T W TH1           Write TH1
T W TH2           Write TH2
T W TL1           Write TL1
T W TL2           Write TL2
T W X /hexRegAddr/ Write any register by its hexadecimal address

```

```

=====
Read register:

```

```

T R AH           Read ALMHCFG
T R AS           Read ALMSCFG
T R FF           Read FIFO
T R FI1          Read FINE1
T R FI2          Read FINE2
T R FL           Read FLAG
T R HC           Read HCFG
T R HW1          Read HIWIPE1
T R HW2          Read HIWIPE2
T R IH1          Read IH1
T R IH2          Read IH2
T R IL1          Read IL1
T R IL2          Read IL2
T R LW1          Read LOWIPE1
T R LW2          Read LOWIPE2
T R TH1          Read TH1
T R TH2          Read TH2
T R TL1          Read TL1
T R TL2          Read TL2
T R X /hexRegAddr/ Read any register by its hexadecimal address

```

```

=====
T S Test Servo menu
T S O FCT1 0300 output register [wr_FINECALTHRU1, initial value 0x0300]
T S I FF         input register [rd_FIFO]
T S A 2          ADC input channel [ bit 2 = 0x0004 = ADCCON_CURRENT_CS1 ]
T S T 0020      target value [0x0020]
T S C 1          ConvergeStep [1]
T S H 1          hysteresis [1]
T S M 60000     max_loop_duration_msec [60000]
T S R           servo loop run

```

Windows is a registered trademark of Microsoft Corp.

Application Note 4065: www.maxim-ic.com/an4065

More Information

For technical questions and support: www.maxim-ic.com/support

For samples: www.maxim-ic.com/samples

Other questions and comments: www.maxim-ic.com/contact

Keep Me Informed

Preview new application notes in your areas of interest as soon as they are published. Subscribe to [EE-Mail - Application Notes](#) for weekly updates.

Related Parts

MAX11008: [QuickView](#)

MAX11014: [QuickView](#) -- [Full \(PDF\) Data Sheet](#) -- [Free Samples](#)

MAX1385: [QuickView](#)

MAX1386: [QuickView](#)

AN4065, AN 4065, APP4065, Appnote4065, Appnote 4065

Copyright © by Maxim Integrated Products

Additional legal notices: www.maxim-ic.com/legal