LTC2949
Battery Stack Monitor

DESCRIPTION

The DC2732A demonstrates a high voltage battery pack monitor based on the LTC®2949. The LTC2949 is a high precision current, voltage, temperature, charge and energy meter for electrical and hybrid vehicles and other isolated current sense applications. It infers charge and energy flowing in and out of the battery pack by simultaneously monitoring the voltage across two sense resistors and the battery pack voltage.

Due to its compatible protocol, the LTC2949 can share the same communication bus with ADI battery stack monitors that contain the isoSPI™ interface.

The LTC2949 is typically powered from an isolated supply and directly connected to the battery stack on the low or high side depending on the shunt position. Resistive high voltage dividers provide connections to high voltage terminals that need to be supervised.

Design files for this circuit board are available.

BOARD PHOTO

TABLE OF CONTENTS

Description 1
Board Photo 1
Performance Summary 3
Hardware Setup 4
Hardware Description 5
Jumper Functions 5
Connector Functions 6
Connector J3 Pin Assignment 7
Pin Functions 7
Solder Jumpers and Other Functions 8
Hardware Setup Examples 9
Option 1: Nonisolated SPI Setup 9
Option 2: Isolated isoSPI DC2617 Setup 10
Option 3: Isolated isoSPI DC1941 Setup 11
Option 4: Isolated isoSPI DC2792 Setup 12
Option 5: Isolated Reversible isoSPI DC2792, DC2350 Setup 13
Connecting a High Voltage Battery 14
Software Setup Overview 15
Arduino IDE Setup 15
LTC2949 Windows GUI Usage 18
LTC2949 GUI Setup 18
Operation Control 19
Multimeters 21
Plots/Data Collection 22
Plot Zoom Fit 23
Plot Pan, Zoom, Labels 23
Change Thresholds On-The-Fly 24
Plot Trackers 25
Export Plot Data 25
Plot Time Axis 25
Register Map 25
Edit Register Values 25
Register Context Menu 26
Display Format Selection 26
Register Map Tooltips 28
Register Details 28
Auto Read Menu 28
Tools Menu 29
Basic Operation Example 29
Appendix A: Usage of SPI Isolator Instead of isoSPI 32
Appendix B: CAN Based Evaluation 33
Hardware Requirements 33
Software Requirements 33
Setup the Hardware 33
DC2732A_CAN: LTC2949 CAN Firmware 34
CAN Messages Overview 34
Basic Operation Examples 40
Current, Power, BAT (100ms) 40
I1, P1, BAT (100ms), I2fast, BATFast (1ms to 2ms) 40
I1, P1, BAT (100ms), I2fast, BATFast (average 10ms) 41
Additional Operation Examples 43
Usage of RAWRW 43
Fast Measurement 46
Serial Monitor via Linduino's USB Port 49
Convert .DBF to .DBC File 51
Resistive Divider Equation 52
Definitions 52
Appendix C: Isolation Measurement with LTC2949 52
Case I: Switch M1 Open 53
Case II: Switch M1 Closed 53
Combine Equations of Both Cases 53
Calculate Y_{ISO}– from Case II: 54
Simulation with Switch Leakage Error 54
Appendix D: Measure Hall Sensor with DC2732A_Basic 54
Appendix E: Synchronous Measurements with Cell Monitors and LTC2949 56
Appendix F: GUI Troubleshooting & Linduino Programming 62
Appendix G: Log Measurements with Tera Term 66
Log Measurement Data from DC2732A_BASIC to TEXT/.CSV – File 66
Appendix H: LTC2949.CPP/.H Basic Library Functions .. 69

PERFORMANCE SUMMARY

The ● denotes the specifications which apply over the full operating temperature range (–40°C to 125°C), otherwise specifications are at T_A = 25°C. The test conditions are V_{CC} = 12.0V, JP11 set to 12.4V operation, unless otherwise noted. JP4–JP6 set to isoSPI, unless otherwise noted.

SYMBOL	PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
Isolated Supply (Flyback LT8301)							
$\lvert V_{GND-LGND} \rvert$	Isolation Working Voltage				800		V
V_{CC}	Input Supply Voltage		●	5.0		12.0	V
				5.0		32.0	V
I_{VCC}	Input Supply Current	Sleep (Note 1) Standby (Note 1)			10 25		mA mA
V_{UVLO}^{+}	Under Voltage Lockout Rising	(Note 2)			4.2		V
V_{UVLO}^{-}	Under Voltage Lockout Falling	(Note 2)			3.1		V
V_{OUT}	Output Voltage	JP11 = 5.3 JP11 = 12.4 JP11 = 9.0			5.3 12.4 9.0		V V V
Nonisolated ADV_{CC} Supply, V_{CC} = 0V/JP9 = Dis							
ADV_{CC}	Input Supply Voltage		●	4.5		14.5	V
I_{ADVCC}	Input Supply Current	Sleep (Note 1), ADV_{CC} = 12V Standby (Note 1), ADV_{CC} = 12V			10 25		mA mA
isoSPI Interface J1, J2, ISO+/ISO–							
$\lvert V_{GND-LGND} \rvert$	Isolation Working Voltage				800		V
R_{TERM}	Differential Bus Termination Resistance				100		Ω
Operation Mode Indication LEDs							
D2	Sleep LED	LTC2949 Core Sleep			Red		
D3	Active LED	LTC2949 Core Standby/Measure			Green		
NTC Temperature Sensor NTCG164KF104FT/TDK							
R1	Shunt Temperature Resistor	Resistance at 25°C			100k		Ω
R2	Board Temperature Resistor	Resistance at 25°C			100k		Ω
A, B, C	Steinhart-Hart Parameters	$\frac{1}{T} = A + B \bullet \ln R_{NTC} + C \bullet (\ln R_{NTC})^3$		A = 9.85013754e-4 B = 1.95569870e-4 C = 7.69918797e-8			
Current Sense Resistor (Note 3)							
Manufacturer: Isabellenhuette Tolerance of Nominal Resistance: 5%, Temperature Range of Given TC: 20°C–60°C							
Manufacturer Part Number							
	BAS-M-R0001-R-5.0	DC2732A-B: 50e–6Ω, R_{THI} = 1.5K/W, TC = 100 ppm/K, P140°C = 20W ($I_{140°C}$ ≈ 630A)					
	BAS-M-R00005-AEU-5.0	DC2732A-A: 100e–6Ω, R_{THI} = 2.0K/W, TC = 50 ppm/K, P140°C = 15W ($I_{140°C}$ ≈ 380A)					
	BAS-M-R0002-R-5.0	DC2732A-C: 200e–6Ω, R_{THI} = 3.0K/W, TC = 50 ppm/K, P140°C = 10W ($I_{140°C}$ ≈ 220A)					

Note 1: The sleep current does not reflect the sleep current of LTC2949, which is in the µA range. The DC2732A uses the LT8301 isolated flyback converter, that requires a minimum load current for stable regulation. To maintain this minimum load and to have some visual feedback on operation mode of LTC2949, the DC2732A has a red LED that indicates sleep and a green LED that indicates standby/measure state.

Note 2: The undervoltage lockout is calculated according to LT8301's data sheet and the nominal resistance values of R16 (R1), R23 (R2).

Note 3: All given specifications of the current sense resistor are from Isabellenhuette's data sheet of the BAS series precision and power resistors.

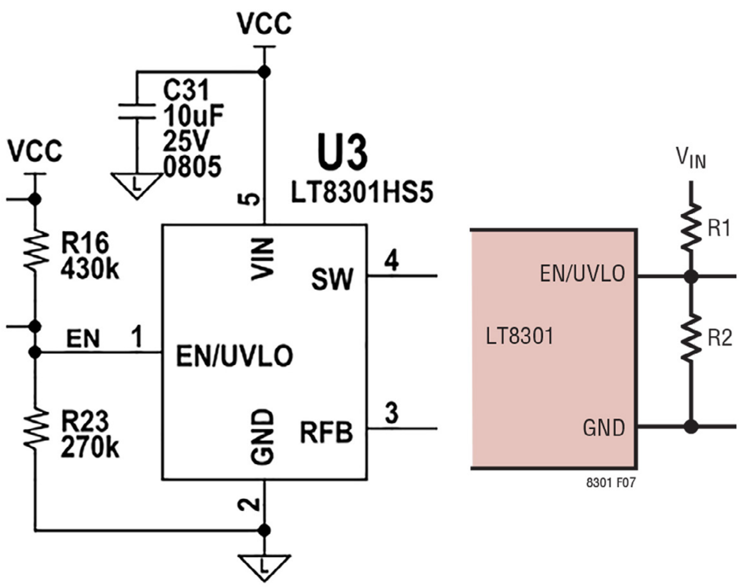
Note 4: When evaluating the current measurement accuracy of the DC2732A the shunt resistance must be calibrated. Also the remaining temperature dependency of the resistance (TC) can be calibrated. To allow precise temperature measurement of the current sense resistor, the DC2732A contains an NTC (R1) placed close to the shunt with good thermal connection (also electrically connected to one pad of the shunt).

HARDWARE SETUP

The DC2732A can be setup for different applications. The power supply and communication interface can be a nonisolated connection via J4 or the individual turrets IOV_{CC}, MISO, MOSI, SCK, CS, ADV_{CC}, GND. An isolated connection is possible using the onboard flyback converter LT8301 sourced by V_{CC}, LGND and using the isoSPI communication via J1, J2 or iso+, iso– test points. If not needed, the onboard power supply can be left unpowered, or be disabled by connecting the enable signal to LGND (put jumper JP9 to DIS). If not using the onboard power supply, any external (also isolated) power supply can be connected to ADV_{CC} and GND (e.g., Analog Devices integrated isolated DC-to-DC converters – isoPower).

The communication mode (SPI or isoSPI) must be set using jumpers JP3–JP6. In case J4 is directly connected to a Linduino®, it provides 7V and JP3–JP6 must be set to SPI operation.

The Hardware Description section gives more details about different hardware setup options. In most cases, the DC2732A is operated together with a DC2026, which is referred to as a Linduino.



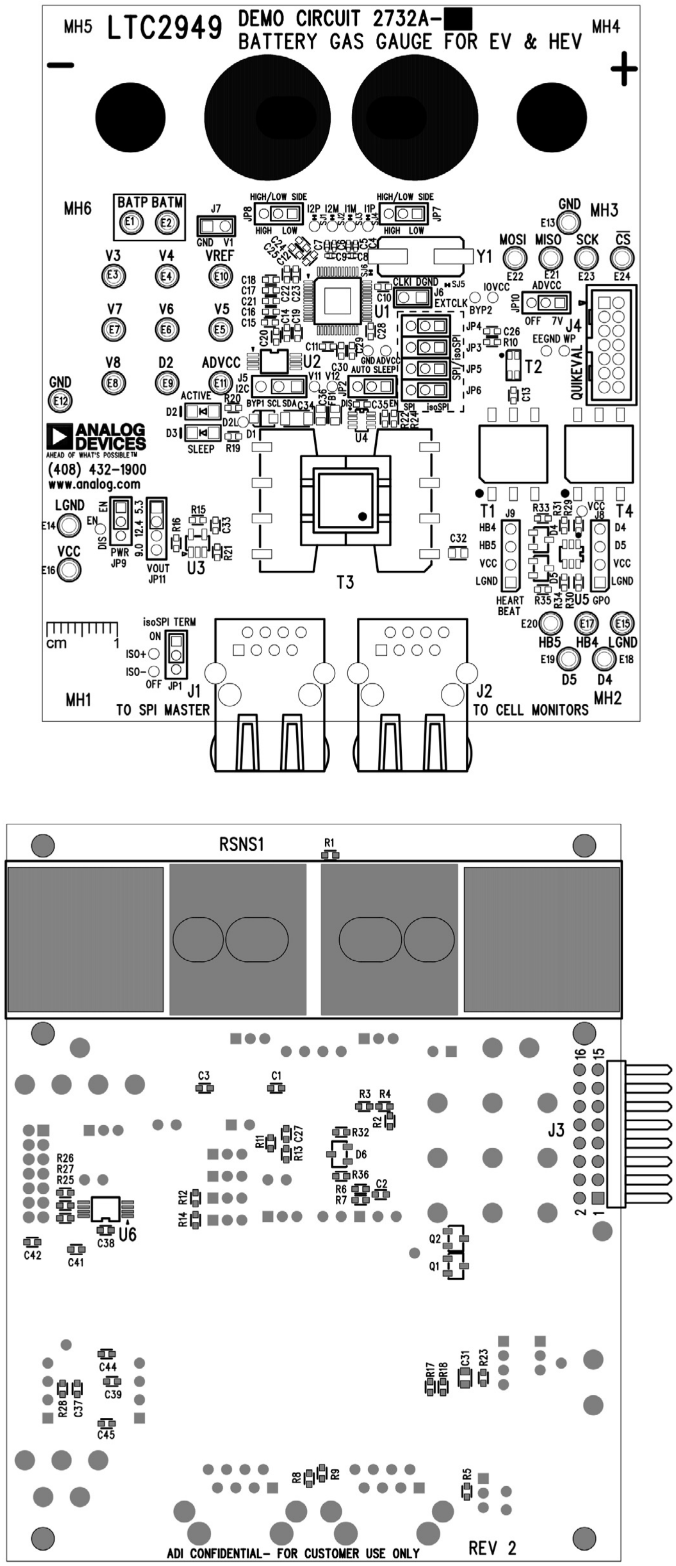
$$V_{IN(UVLO+)} = \frac{1.242V \bullet (R1 + R2)}{R2} + 2.5\mu A \bullet R1$$

$$V_{IN(UVLO-)} = \frac{1.228V \bullet (R1 + R2)}{R2}$$

R1 = 430e3; R2 = 270e3; UVLOP = 1.242 • (R1 + R2)/R2 + 2.5E-06 • R1;
UVLON = 1.228 • (R1 + R2)/R2; UVLON, UVLOP (3.18, 4.30)

Figure 1. Undervoltage Lockout Calculation for LT8301 According to the Schematic of DC2732A

HARDWARE DESCRIPTION



a) Top

b) Bottom

Figure 2. DC2732A2

HARDWARE DESCRIPTION

JUMPER FUNCTIONS

JUMPER	DEFAULT	NAME	DESCRIPTION
JP1	On	isoSPI Term	**Enable/disable 100e isoSPI bus termination.** If LTC2949 is the only device on the isoSPI bus or connected on top of a cell monitor isoSPI daisy chain, the termination must be enabled. If LTC2949 is connected in parallel to a daisy chain (LTC2949 is fed into the input of a daisy chain via J1, J2) of cell monitors, the termination must be disabled.
JP2	EN	Auto Sleep	**Enable/disable automatic entering of sleep state after power-up.** The LTC2949 automatically returns to sleep state if no wake-up confirmation command is received within 1 second after entering standby state, after power-up. Wake-up confirmation can be either writing 0x00 to register 0x70 or starting of a measurement. LTC2949 will not go automatically to sleep if SDA is pulled low (JP2 at position DIS) during power-up. As SDA is also used for the optional external EEPROM, it must be released from GND to allow communication to the EEPROM. In most applications, the Auto Sleep will stay enabled and the muster will send the wake-up acknowledge. Still, for debugging purposes, this jumper may be set to DIS to force LTC2949 to stay in standby mode.
JP3–JP6	isoSPI	SPI/isoSPI	**Select between nonisolated SPI or isolated isoSPI communication mode.** All jumpers must be either set to the left (SPI) or the right (isoSPI) position. In case the QuikEval™ connector J4 is connected to a Linduino, the interface must be set to SPI.
JP7, JP8	Low	High/ Low Side	**Select between high or low side current sensing.** To avoid the sense resistor being floating, it can be tight to either LTC2949's GND (low side sensing) or to LTC2949's A/DV$_{CC}$ (high side sensing). It is also possible to remove both jumpers (or set JP7 to low and JP8 to high) if the shunt is applied somewhere between LTC2949's supply rails by external connections. See following sections for example setups.
JP9	EN	PWR	**Enable/disable the onboard flyback converter LT8301**. The isolated onboard supply can also be disabled by disconnecting V$_{CC}$, LGND or setting V$_{CC}$, LGND to less than 3V (see V$_{UVLO}$–).
JP10	7V	ADV$_{CC}$	**Enable (7V)/disable (off) connection of LTC2949's A/DVCC supply input to V$^+$ of J4.** If Linduino is connected to J4, V$^+$ is supplied with 7V from a SMPS with post LDO on the Linduino. The Linduino allows to override this voltage up to LTC2949's max. operating voltage of 14.5V via turrets ADV$_{CC}$ and GND. If JP10 is set to off, it is also possible to apply any voltage between 4.5V and 14.5V to ADV$_{CC}$ and still use the Linduino connected via J4.
JP11	5.3V	V$_{OUT}$	**Onboard isolated flyback converter LT8301 output voltage selector.** Select one of three (5.3V, 9.0V, 12.4V) pre-configured output voltages for the flyback converter. Higher supply voltages are useful to take advantage of LTC2949's GPOs being able to drive the output to one of LTC2949's supply rails. This allows for example to ensure sufficient gate-source voltage when using MOSFETs to switch high voltage resistive dividers connected to LTC2949's voltage inputs. See Hardware Setup Examples section and LTC2949 data sheet for more details.

HARDWARE DESCRIPTION

CONNECTOR FUNCTIONS

CONNECTOR	NAME	DESCRIPTION
J1, J2	SPI Master, Cell Monitors	**isoSPI connectors.** Pin 1 (iso–) and Pin 2 (iso+) of both connectors are connected in parallel. Thus, any of the two connectors can be used as the interface to the isoSPI master. For clarity, they are still named differently to indicate one connector can be used to interface to the isoSPI master and one to a cell monitor. Having two connectors allows the DC2732A to be easily inserted into the isoSPI bus between a LTC6820 and a LTC681x/ADBMS681x cell monitor by using two standard Ethernet cables. Physically the LTC2949 is then connected in parallel to this isoSPI bus and the isoSPI termination resistor (see JP1) must be disabled.
J3	EXT	**General purpose I/O connector.** Allows connection to LTC2949's analog inputs V2 to V7, VBATP, VBATM, general purpose I/Os V8 to V10, reference output V_{REF} and supply voltage A/DV_{CC}. See Pin Functions or schematic for pin assignment.
J4	QuikEval	**Linduino QuikEval connector.** Connect the 14-pin flat ribbon cable between this connector and Linduino for nonisolated SPI operation. Keep in mind that the Linduino has an isolated USB interface, meaning also in this setup the DC2732A is isolated from the PCs USB port. Also note, that the Linduino's default VCCIO output voltage setting (Pin 2) is 5V, whereas LTC2949's max. IOV_{CC} operating voltage is 4.5V. The absolute maximum voltage for IOV_{CC} of LTC2949 is 5V plus some safety margin Analog Devices puts on such parameters, thus no damage will happen even with Linduino's VCCIO tolerances. Still, for operation within LTC2949's specifications, it is recommended to set the jumper JP3 of the Linduino to any of EXT, 2.5V or 3.3V. It is also allowed to remove the jumper which sets VCCIO to 1.8V. In case EXT is selected, an external voltage of 1.8V to 4.5V must be applied to JP1 of Linduino or to test point IOV_{CC} of DC2732A.
J5	I^2C	**I2C test points.** Allows to connect a scope or 2nd I^2C master to the onboard I^2C EEPROM for debugging purposes.
J6	EXTCLK	**External clock interface.** In case the internal oscillator and the optional onboard 4MHz crystal is not used, an external oscillator between 10kHz and 25MHz can be connected to those test points.
J7	NTC (V1)	**External NTC connection.** The DC2732A, as default, has an onboard NTC that is connected to V1 and to IM (negative shunt terminal). This allows a good thermal (via big copper plane) connection to the shunt for sensing its temperature. As the NTC is put into a voltage divider between V_{REF} and GND, it requires the shunt to be configured for low side sensing (IM connected to GND via JP8). To allow the temperature measurement of the shunt also in high side current sensing applications, it is possible to remove the onboard NTC R1 and connect an external, wired NTC to J7. For tight thermal coupling the external NTC can then be clued to the shunt (electrical isolated). If shunt temperature measurement is not required in high side sensing applications, it is not necessary to remove the onboard NTC. Still, if leaving the NTC R1 populated, the analog input V1 can't be used for other purposes. To allow usage of V1 for other analog input signals, the NTC R1 and the reference resistor R3 must be removed.
J8	GPO	**Isolated digital interface to LTC2949's alert signals (heart-beat signals DO4, DO5).** If enabled, the heart beat signals of LTC2949 toggle at 400kHz at normal operation. They are transferred via capacitors and the dual isolation transformer T4 over the isolation barrier, rectified and buffered via the dual comparator LT6700. The open-drain output signals are driven to V_{CC} by weak 150k pull-up resistors, as long the heart beat signals are toggling. The weak pull-up resistors allow to connect low-ohmic (e.g., 3.3k) external pull-up signals to a different I/O voltage if required. In case of an alert (e.g., over current), the associated heart beat signals stop, and the comparator's output is pulled low indicating the alert on the isolated, low voltage side. See LTC2949's data sheet for details on the heart beat signals.
J9	Heart Beat	**Test points to the heart beat signals after the isolation transformer and before the rectifier.** For debugging purposes only.
J8, J9		In addition to the alert and heart-beat signals, those connectors also allow access to the supply voltage input V_{CC} and LGND of the onboard flyback converter.

HARDWARE DESCRIPTION

CONNECTOR J3 PIN ASSIGNMENT

PIN	NAME	DESCRIPTION
15	GND	LTC2949's GND
13	VBATM	Negative BAT Input
11	NTC2+V2	Analog Input, a 100k NTC/R_{REF} is Connected Onboard
9	V4	Analog Input V4
7	V6	Analog Input V6
5	V8_DO1	Analog Input V8/Digital Output DO3 (Note 4)
3	V10_DO3	Analog Input V10/Digital Output DO3 (Note 4)
1	GND	LTC2949's GND

PIN	NAME	DESCRIPTION
16	V_{REF}	LTC2949's 3V Reference Voltage Output
14	VBATP	Positive BAT Input
12	V3	Analog Input V3
10	V5	Analog Input V5
8	V7	Analog Input V7
6	V9_DO2	Analog Input V9/Digital Output DO2 (Note 4)
4	GND	LTC2949's GND
2	ADV_{CC}	LTC2949's ADV_{CC} Supply Voltage

Note 1. All dual purpose pins (GPIOs) do not have their filter capacitor populated on the PCB. This was done to not interfere with the digital output function (including the 400kHz toggling mode) of those pins. It is recommended to assemble an input filter capacitor (C14, C19, C20) to the GPIOs that are used in analog input only mode for best noise filter performance.

PIN FUNCTIONS

PIN/ TURRET	NAME	DESCRIPTION
E1	BATP	See J3 VBATP
E2	BATM	See J3 VBATM
E3	V3	See J3 V3
E4	V4	See J3 V4
E5	V5	See J3 V5
E6	V6	See J3 V6
E7	V7	See J3 V7
E8	V8	See J3 V8_DO1
E9	D2	See J3 V9_DO2
E18	D4	Isolated, Rectified and Buffered Heart Bit Signal V11_DO4
E19	D5	Isolated, Rectified and Buffered Heart Bit Signal V12_DO5 (Over Current Comparator Output)
E14, E15	LGND	Negative Supply Input to Onboard Isolated Flyback Converter LT8301

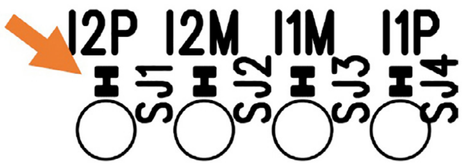
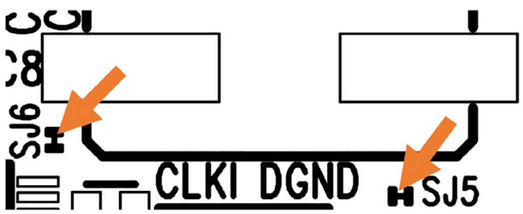
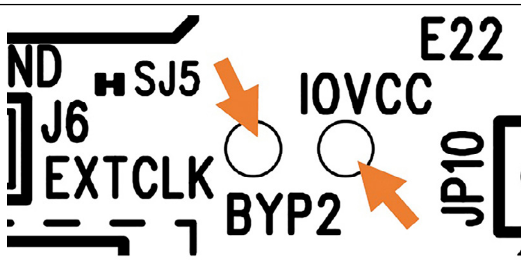
PIN/ TURRET	NAME	DESCRIPTION
E10	VR_{EF}	See J3 V_{REF}
E11	ADV_{CC}	See J3 VBATP
E12, E13	GND	See J3 VBATP
E21	MISO	LTC2949's SPI Interface (When Configured to SPI Mode, See JP3–JP6)
E22	MOSI	
E23	SCK	
E24	CS	
E17	HB4	Isolated Heart Bit Signal V11_DO4
E20	HB5	Isolated Heart Bit Signal V12_DO5 (Over Current Comparator Output)
E16	V_{CC}	Positive Supply Input to Onboard Isolated Flyback Converter LT8301

Note: [shaded] = Signals on the low voltage side, isolated from LTC2949

Rev. 0

HARDWARE DESCRIPTION

SOLDER JUMPERS AND OTHER FUNCTIONS

REF	PCB PICTURE	DESCRIPTION
SJ1–SJ4		Normally closed solder jumpers to allow disconnection of onboard shunt RSNS1. Those are small PCB footprints that can be cut with a small knife. SJ1, SJ2 allow to disconnect I2P, I2M from the shunt. SJ3, SJ4 allow to disconnect I1P, I1M from the shunt. Once a channel is disconnected, it is possible to connect an external shunt to the test points right below the solder jumpers. After being cut, it is still possible to close them again by applying some solder.
SJ5, SJ6		Normally closed solder jumpers to allow disconnection of onboard 4MHz crystal.
BYP2, IOV_{CC}		Test points to LTC2949's IOV_{CC} supply input and BYP2 3.3V supply output. The BYP2 supply output can be used to load external circuits with up to 10mA, for example to supply an external SPI isolator like ADuM141E or ADuM4154. In such applications IOV_{CC} can be connected to BYP2 via those test points. See also Appendix A: Usage of SPI Isolator Instead of isoSPI for more details.

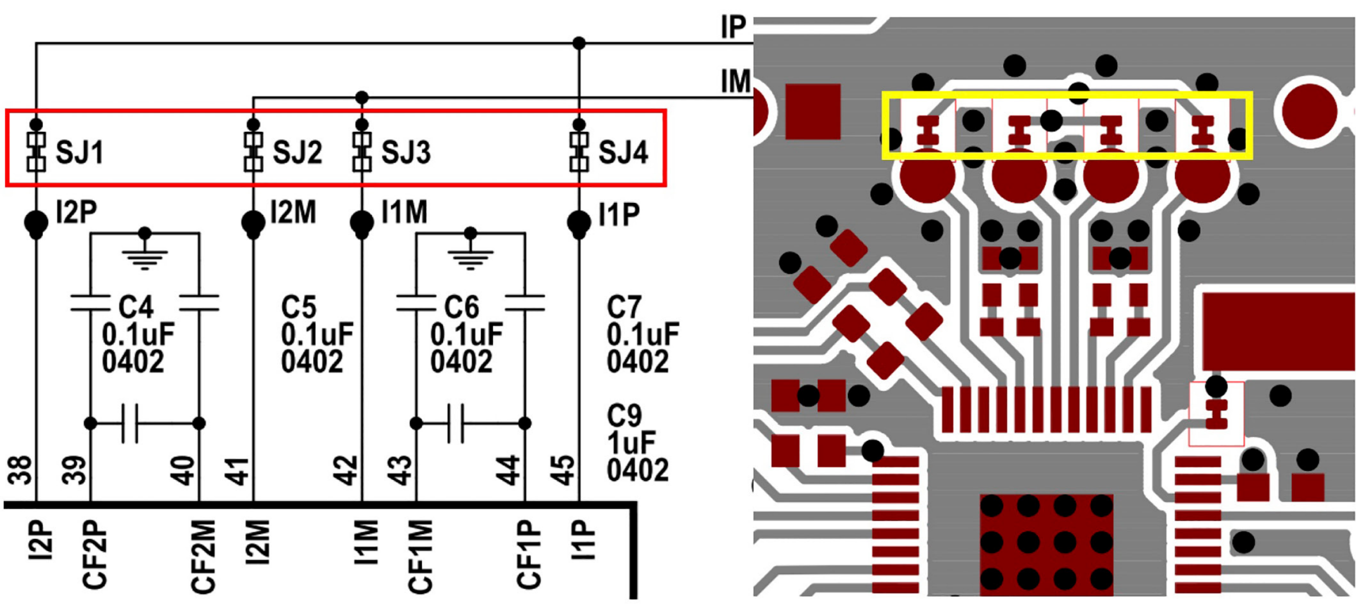


Figure 3. Solder Jumpers SJ1–SJ4, Allow Separation of the Two Current Channels

HARDWARE SETUP EXAMPLES

The DC2732A demo board can be operated in four different setup options.

1. Nonisolated SPI interface with DC2026 (Linduino)

Options 2 thru 5 are isolated.

2. isoSPI interface with DC2617 (CAN to isoSPI shield) plugged on top of DC2026 (Linduino)
3. isoSPI interface with DC1941 (LTC6820 SPI to isoSPI bridge) connected to DC2026 (Linduino)
4. isoSPI interface with DC2792 (dual LTC6820 SPI to isoSPI bridge) connected to DC2026 (Linduino)
5. isoSPI interface parallel to a reversible daisy chain of cell monitors (as above with DC2792, DC2026)

For option 1, the LTC2949 is supplied with 7V by Linduino. In all other cases a 5V to 12V (or up to 32V, see performance summary) supply needs to be connected to turrets LGND and V_{CC} on the lower left side of the demo board.

For option 1, the LTC2949 is galvanically connected to the Linduino. Still, the Linduino has a galvanic isolation to its USB port. For the other two options, the communication and supply to LTC2949 is isolated by an isoSPI transformer and flyback converter.

Make sure to set the SPI/isoSPI selection jumpers J3–J6 to the correct position depending on the chosen option (SPI for option 1, isoSPI for all other options).

In all setups it is possible to operate LTC2949 together with cell monitors ADBMS68xx/LTC681X. As the DC2732A has two RJ45 isoSPI connectors, this is especially easy when operating in isoSPI mode, as shown in option 5.

Figure 4 thru Figure 6 show how DC2732A is configured and connected with above mentioned demo circuits.

OPTION 1: NONISOLATED SPI SETUP

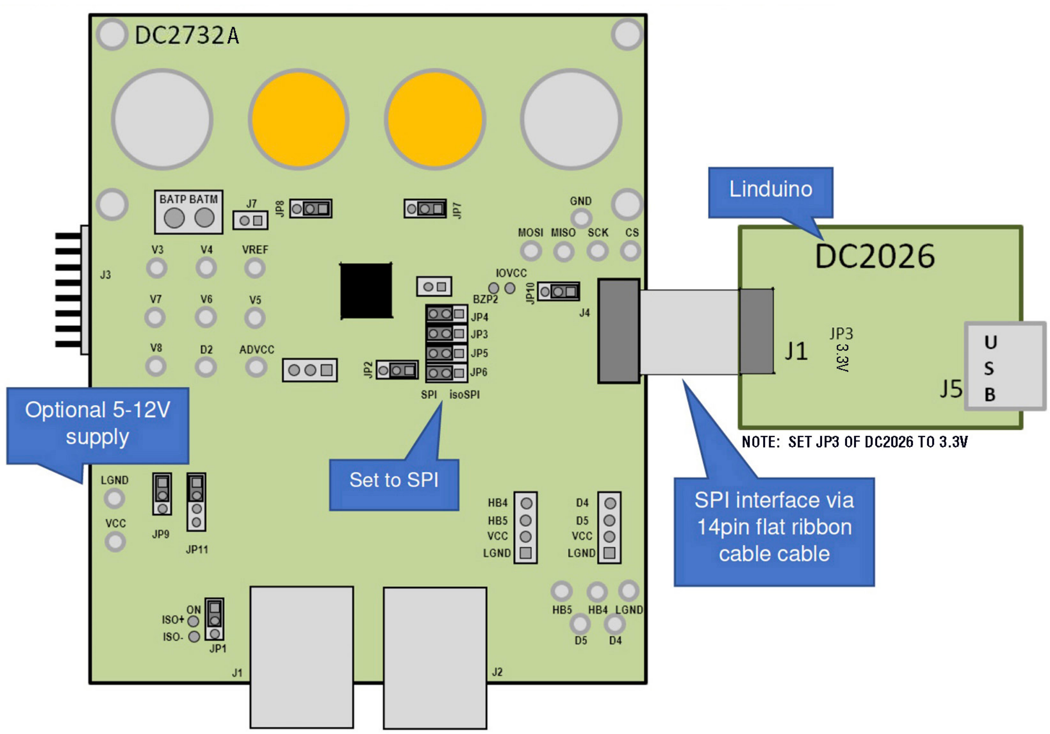


Figure 4. Nonisolated SPI Communication and Supply via Linduino (Linduino's USB Port is Still Isolated). Supply via Turrets V_{CC}, LGND is Not Necessary in This Setup; This is Also the Recommended Initial Setup when Using the GUI Software, the First Time as It Allows Easy GUI Installation via QuikEval; After the GUI is Installed Any Other Setup Can Also Be Used

HARDWARE SETUP EXAMPLES

OPTION 2: ISOLATED ISOSPI DC2617 SETUP

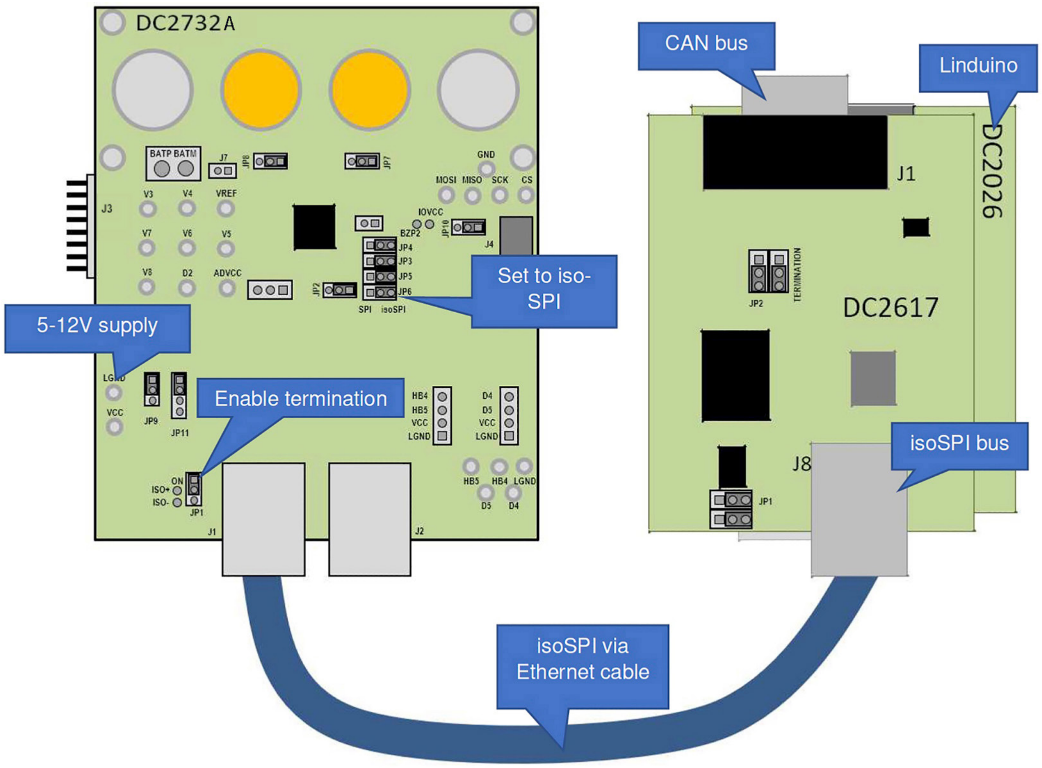


Figure 5. Isolated isoSPI Communication via LTC6820 on DC2617, Isolated Supply via Turrets V_{CC}, LGND. Optional CAN Interface via DC2617; The Linduino Sketchbook for LTC2949/DC2732A Also Contains a Software Example, that Allows Measurements Done by LTC2949 to Be Controlled and Communicated Into a CAN Environment; See Appendix A: Usage of SPI Isolator Instead of isoSPI

HARDWARE SETUP EXAMPLES

OPTION 3: ISOLATED ISOSPI DC1941 SETUP

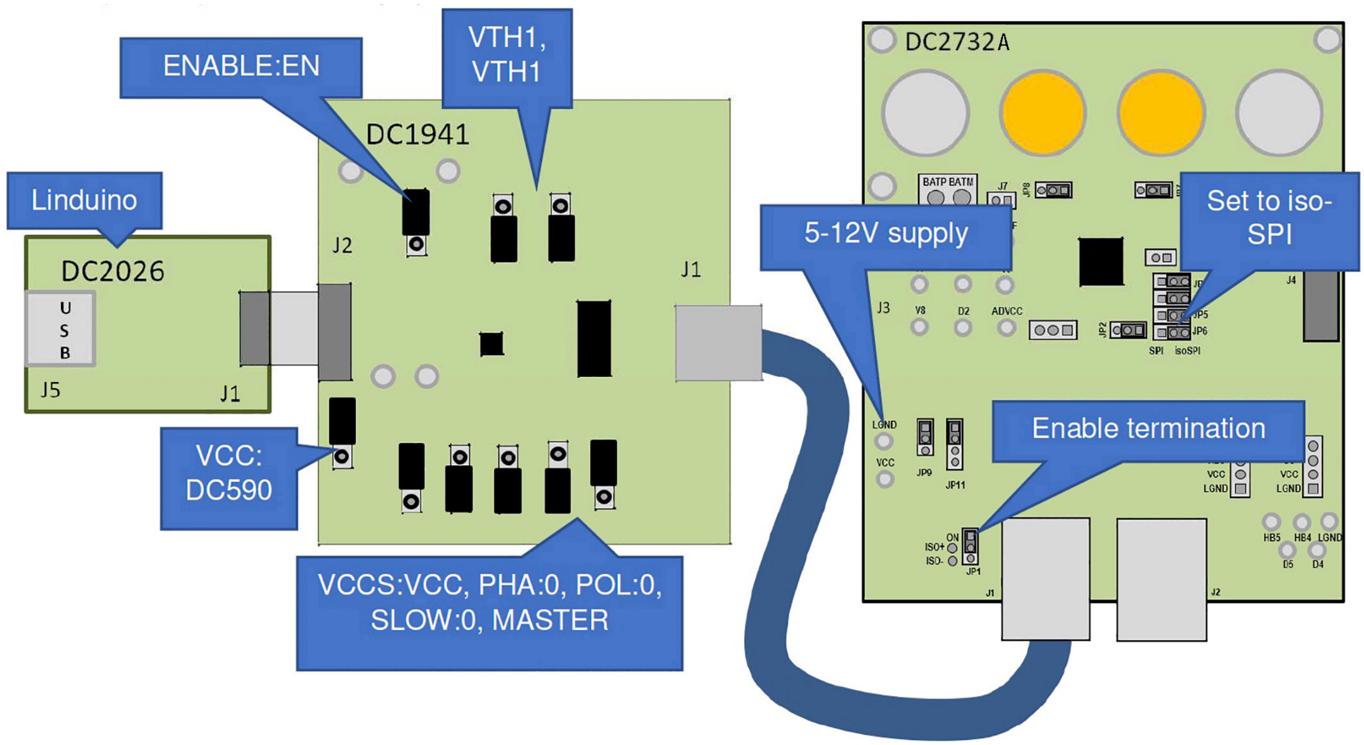


Figure 6. Isolated isoSPI Communication via LTC6820, Isolated Supply via Turrets V_{CC}, LGND; Note the Jumper Settings of DC1941 (LTC6820) Demo Board for Proper isoSPI Communication

Rev. 0

HARDWARE SETUP EXAMPLES

OPTION 4: ISOLATED ISOSPI DC2792 SETUP

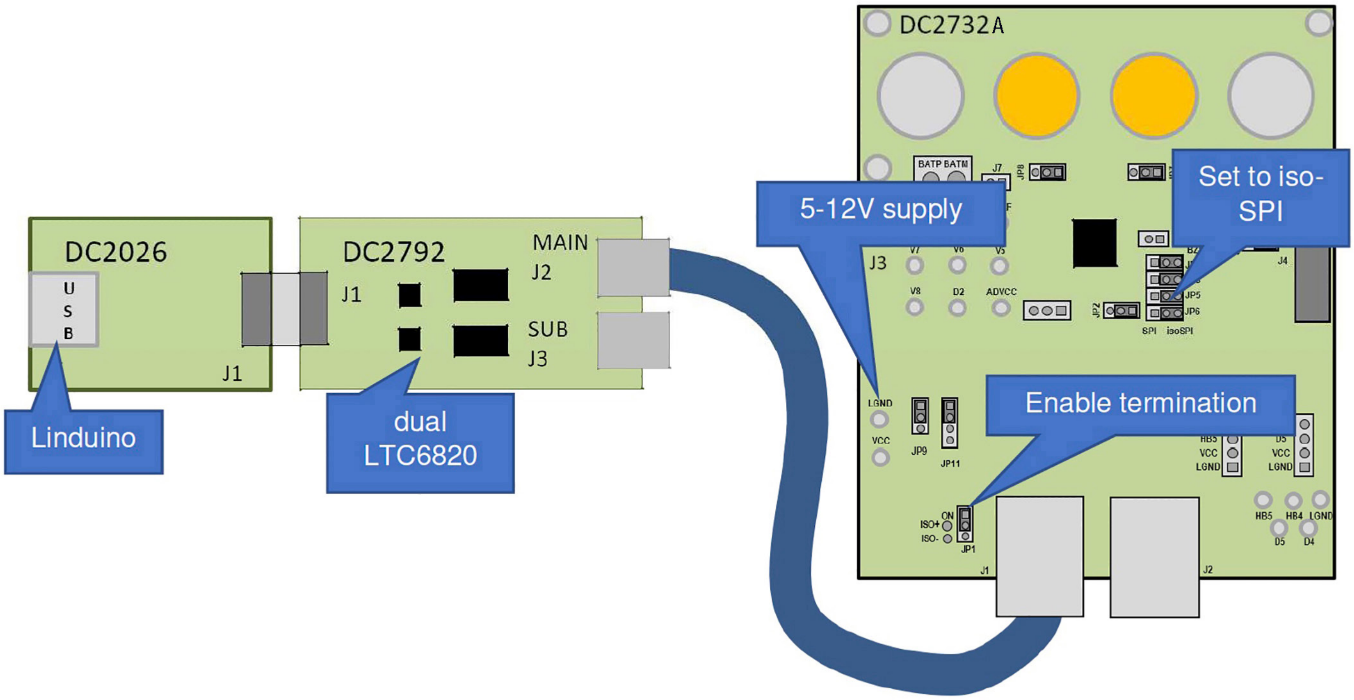


Figure 7. Isolated isoSPI Communication via Dual LTC6820 Demo Board (DC2792), Isolated Supply via Turrets V_{CC}, LGND; Note: DC2792 Can Also Be Plugged on Top of DC2026 Without Using the 14-Pin Flat Ribbon Cable

HARDWARE SETUP EXAMPLES

OPTION 5: ISOLATED REVERSIBLE ISOSPI DC2792, DC2350 SETUP

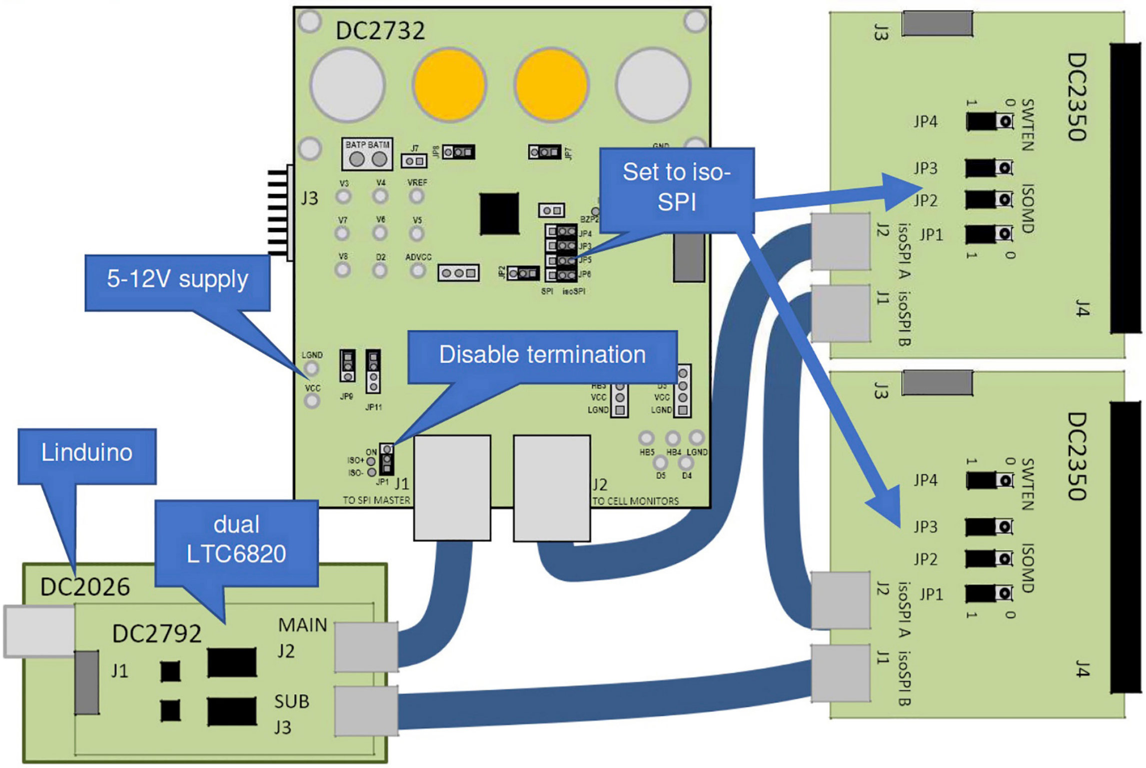


Figure 8. Isolated, Reversible isoSPI Communication via Dual LTC6820 Demo Board (DC2792) Together with DC2350 (LTC6812-1/LTC6813-1)

HARDWARE SETUP EXAMPLES

CONNECTING A HIGH VOLTAGE BATTERY

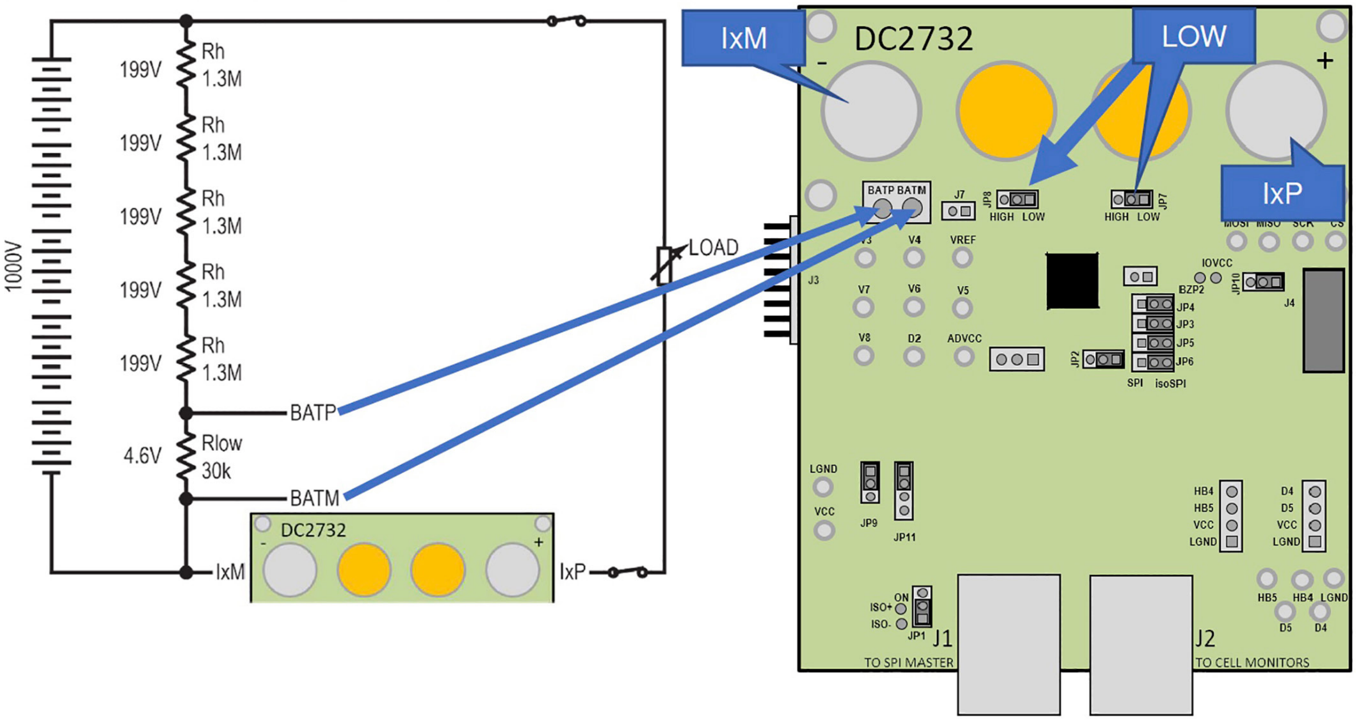


Figure 9. Connecting a High Voltage Battery with Resistive Divider to DC2732 for Low Side Current Sense Operation

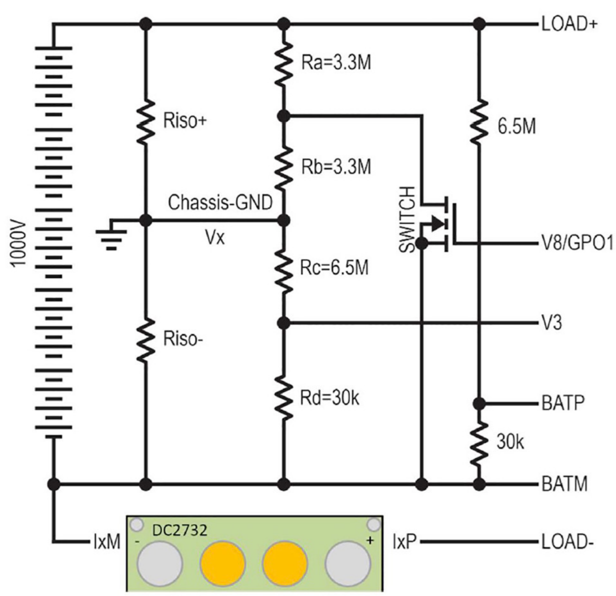


Figure 10. Connecting a High Voltage Battery with Resistive Divider to DC2732 for Low Side Current Sense Operation with Chassis-GND Isolation Measurement; See Appendix C: Isolation Measurement With LTC2949 for Operation and Calculation Details

SOFTWARE SETUP OVERVIEW

The DC2732A can be controlled by the Linduino One (DC2026) board. The Linduino is an Arduino compatible platform with example code that will demonstrate how to control multicell battery stack monitor ICs and the stack monitor LTC2949. Compared to most Arduino compatible microcontroller boards, the Linduino offers conveniences such as an isolated USB connection to the PC, built-in SPI MISO line pull-up to properly interface with LTC2949's open-drain SDO, and an easy ribbon cable connection for SPI communication through the DC2732A 14-pin "QuikEval" J4 connector. Please see the Linduino web page for more details.

Besides using the Linduino as a host controller, the DC2732A can also operate with any other host controller, that offers a SPI interface. Linux based evaluation, for example using a Raspberry Pi, is also supported; please contact Analog Devices for details.

ARDUINO IDE SETUP

1. Download and install the Arduino IDE to the PC. Detailed instructions can be found under the quick start tab.
2. Set the Arduino IDE to open LTC2949 Sketchbooks. From within the Arduino IDE, click on File menu select Preferences. Then under Sketchbook location: select Browse and locate the path to the extracted LinduinoSketchbook2949.zip file that was provided by ADI.
   a. If there is already a BMS Sketchbook, it can be extracted into the same folder as the bmsSketchbookBeta.zip.
   b. Also, if there is already a local copy of LinduinoScketchbook2949.zip, it can be extracted into the LTSketchbook folder.

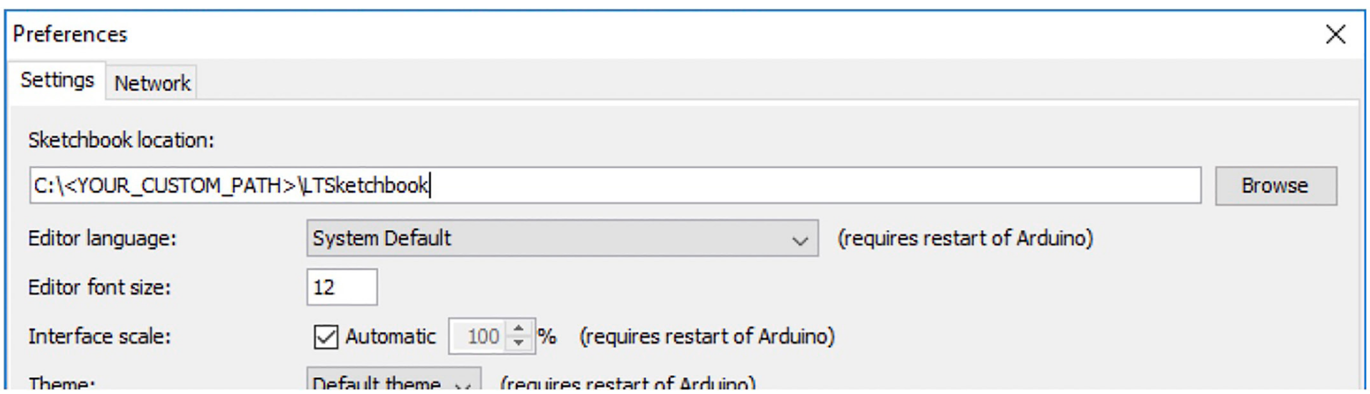


Figure 11. Arduino IDE Preferences, Sketchbook Location

3. Close then re-open the Arduino IDE to enable the use of the Sketchbook Location that was previously set.
4. Select the correct COM port to allow communication to Linduino through USB. Under the Tools menu, select Port → Select the highest number COMxx with the "√" check mark symbol. There may be more than one option; Linduino is usually the highest COM port number. The PC screenshots used in this example show the Linduino connected to COM6. To identify the right COM port, unplug the USB cable and check which port disappears from the Tools/Port menu.

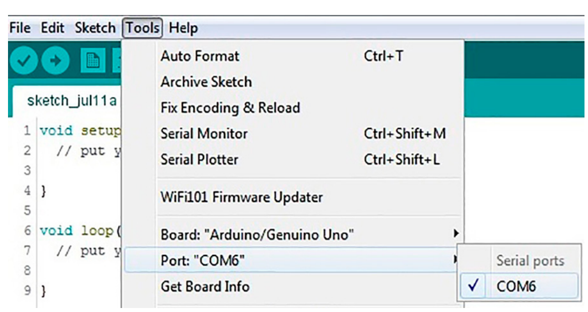


Figure 12. Arduino IDE, COM-Port Setting

SOFTWARE SETUP OVERVIEW

5. Select the correct Arduino compatible microcontroller board. Under the Tools menu, select Board → Arduino/Genuino Uno with the "•" black dot symbol.

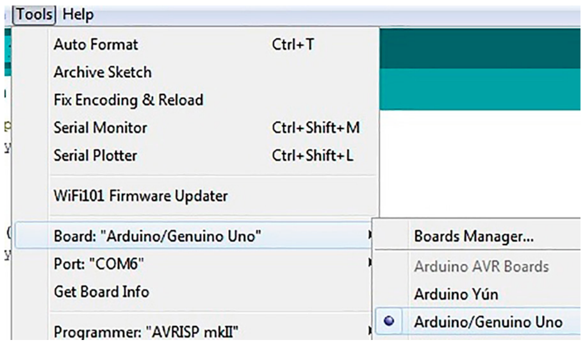


Figure 13. Arduino IDE, Board Setting

6. Open one of the programs, called "Sketches," associated with the DC2732A. In this example DC2732A_BASIC Sketch will be opened. Under the File menu, select Sketchbook → Part Number → 2000 → 2900 → 2949 → DC2732A_BASIC.

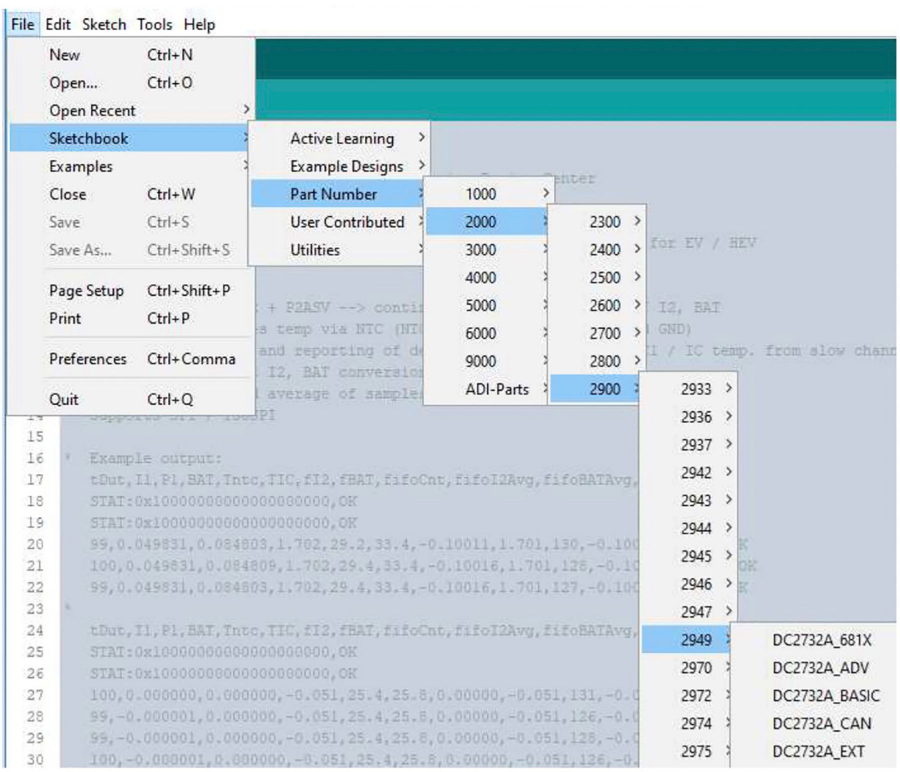


Figure 14. Arduino IDE, Sketchbooks for LTC2949/DC2732A

SOFTWARE SETUP OVERVIEW

7. Upload the DC2732A_BASIC Sketch onto the Linduino by clicking on the Upload button on the top left corner. When this process is completed there will be a "Done Uploading" message on the bottom left corner.

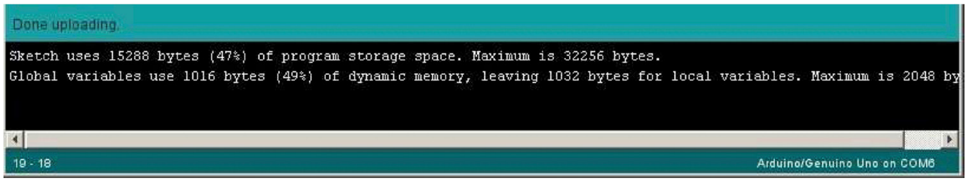


Figure 15. Arduino IDE, Sketchbook Upload

8. Open the Arduino Serial Monitor tool. Click on the Serial Monitor button on the top right corner then the Serial Monitor window will open and show on the top left corner the COMxx used.

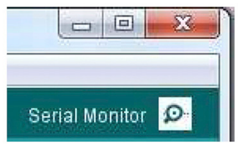


Figure 16. Arduino IDE, Serial Monitor

9. Configure the Serial Monitor to allow communication to the Linduino through USB. On the bottom of the Serial Monitor window, set the following starting from bottom left to bottom right.

   a. Enable "Autoscroll"

   b. Select Both NL and CR on the left dropdown menu.

   c. Select 1000000 baud on the right dropdown menu (see Serial.begin within DC2732A_BASIC.ino for the baud-rate setting).

   Note: In case Arduino DUE is used, the max. supported baud rate is 250000.

   d. As shown in Figure 17, when configured correctly the DC2732A_BASIC Sketch will start to output measurement data.

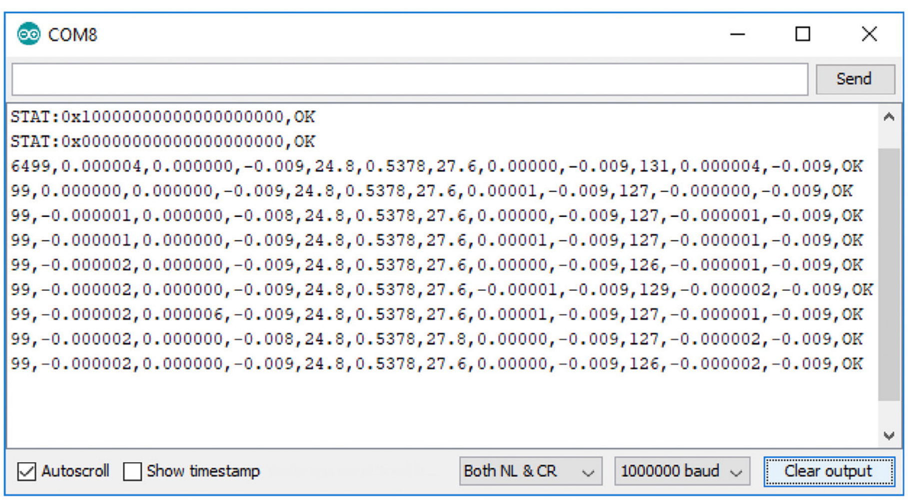


Figure 17. Arduino IDE, Serial Monitor Output of DC2732A_BASIC Sketch

SOFTWARE SETUP OVERVIEW

LTC2949 GUI SETUP

Various features of the DC2732A can be demonstrated by using Analog Devices' QuikEval Software. QuikEval is a USB-based product demonstration and data acquisition software meant to be used in conjunction with the Linduino programmed with the DC590B Sketch (factory default) that connects to individual daughter cards for specific Analog Devices products.

Connect the Linduino to a PC using the USB cable provided with the Linduino. Now, connect the Linduino to the DC2732A configured to SPI mode, see Hardware Setup section.

Once setup is complete (also wait for Windows installing the drivers after Linduino is connected the first time to the PC), run the QuikEval Software. QuikEval should auto-detect the DC2732A and check if the LTC2949 GUI is already installed. QuikEval will automatically download and install the GUI if necessary and launch it. Once the LTC2949 GUI is installed, QuikEval is not necessary anymore and the GUI can be opened directly from the Windows Start Menu (named LTC2949). In case the DC2732A is operated in isoSPI mode in which the Linduino is not connected via the 14-pin ribbon cable directly to DC2732A but to some isoSPI demo board (DC1941, DC2792, DC2617), QuikEval is unable to detect the DC2732A as it will only read the identification EEPROM on the isoSPI demo board and thus try to launch other GUIs that are not compatible with LTC2949. Still, if the LTC2949 is opened directly from the Windows start menu it will operate normally (QuikEval should be closed before opening the LTC2949 GUI).

LTC2949 WINDOWS GUI USAGE

The DC2732A software user interface was designed to allow users to quickly evaluate the LTC2949. The user has the ability to plot/monitor voltage, current, power, charge and energy, and fully access LTC2949's register map to make any configuration. Also, operation of LTC2949 together with ADBMS68xx/LTC681x cell monitors e.g., to do synchronous measurements of cell voltage and battery stack current is supported.

Once the graphical user interface (GUI) is started, the user can connect to the LTC2949 via Linduino and the GUI will initially read out all the register values from the LTC2949.

After the connection is established, most typical basic operation steps are:

1. Starting continuous conversion.
2. Enabling Register Auto Read (data transfer from device to PC).
3. Clearing device's accumulators and trackers and GUI's plots.

The LTC2949 GUI is split into several sub-windows that can be rearranged and even detached from the main window by dragging the title, moving around and dropping to a new position. Some sub-windows are hidden and can be shown by enabling them in the View menu. All this allows

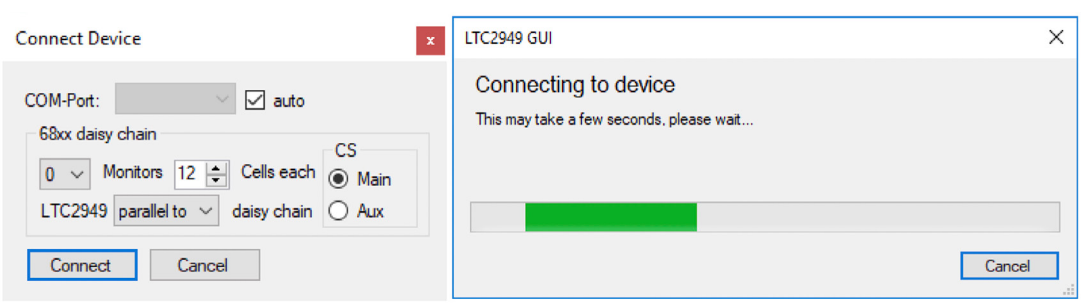


Figure 18. GUI: Connect Device

LTC2949 WINDOWS GUI USAGE

the GUI to be adjusted to any screen size and to show only the information that is of interest for a given application without overwhelming the user with the large feature-set of the LTC2949.

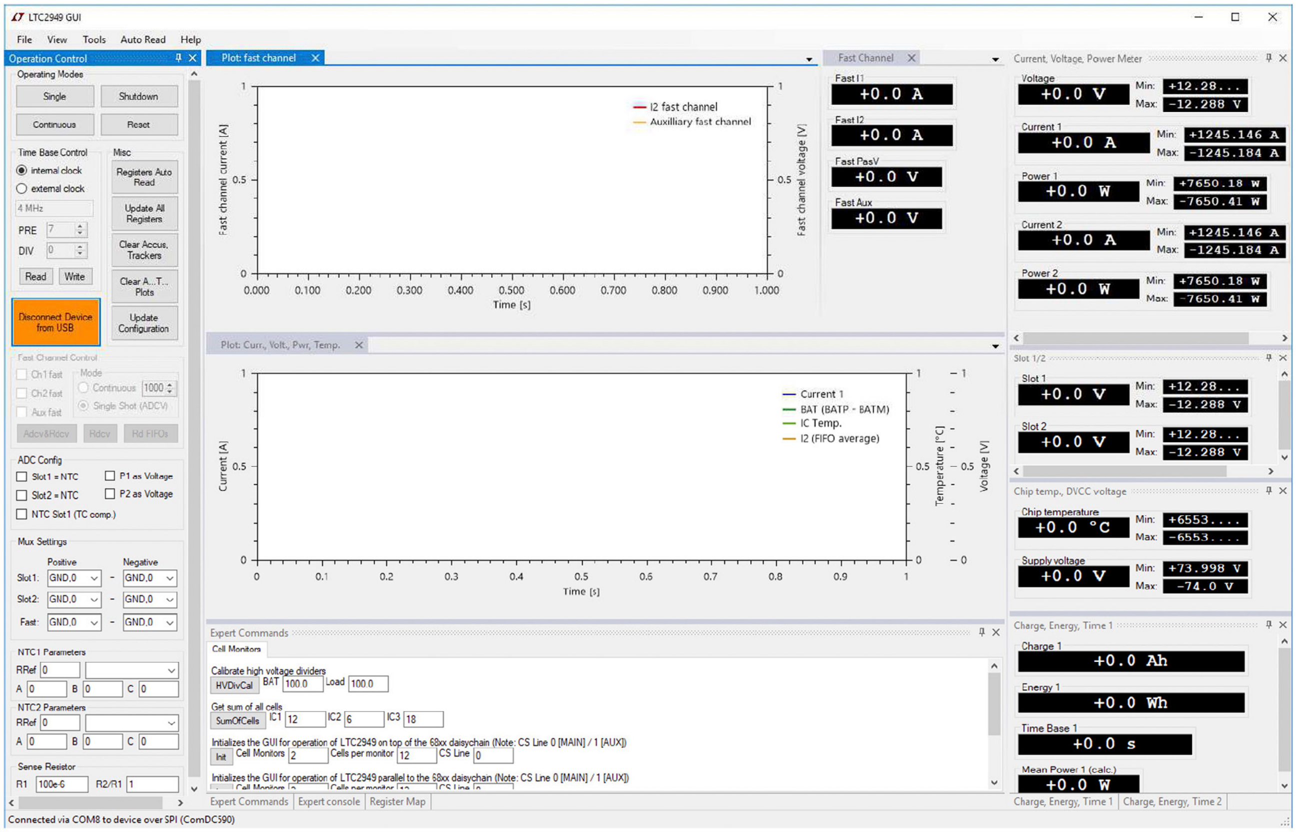


Figure 19. GUI: Main Window

Any custom GUI layout may be set as the default (View → Set Default Layout), stored to a specific file (View → Save Layout...) or reloaded (View → Get Default Layout or Load Layout...). The default layout will always be reloaded on start-up. The GUI may be reset to the factory default layout by View → Reset Default Layout.

OPERATION CONTROL

The Operation Control functions are:

1. Connect/Disconnect Device via USB (Linduino/DC2026).
2. Enable/Disable Continuous measurement.
3. Make Slow Channel Single conversion.
4. Shutdown Device.
5. Reset Device.
6. Time Base Control to set internal or external clock frequency including calculation of PRE/DIV values for a given clock (see also Tools Menu section).
7. Enable/Disable Register Auto Read. If enabled data is automatically transferred from device to PC every 100ms (default), see Auto Read Menu section.
8. Manually Update All Registers (manually transfer all data from device to PC).
9. Clear Accumulator and Tracking registers.
10. Clear Accumulator and Tracking registers and reset all data in all plot windows.
11. Update Configuration from 2nd memory page (ADJUPD which is necessary when changing gain correction factors, ADC configuration etc.).
12. Configure ADCs (requires ADJUPD to become effective).

LTC2949 WINDOWS GUI USAGE

13. Configure Fast Channel (only possible when in Continuous mode).
14. Configure AUX multiplexer.
15. Set NTC parameters (requires ADJUPD to become effective).
16. Configure sense resistor's nominal value and ratio in case two separate sense resistors are used (only change of ratio requires ADJUPD to become effective).

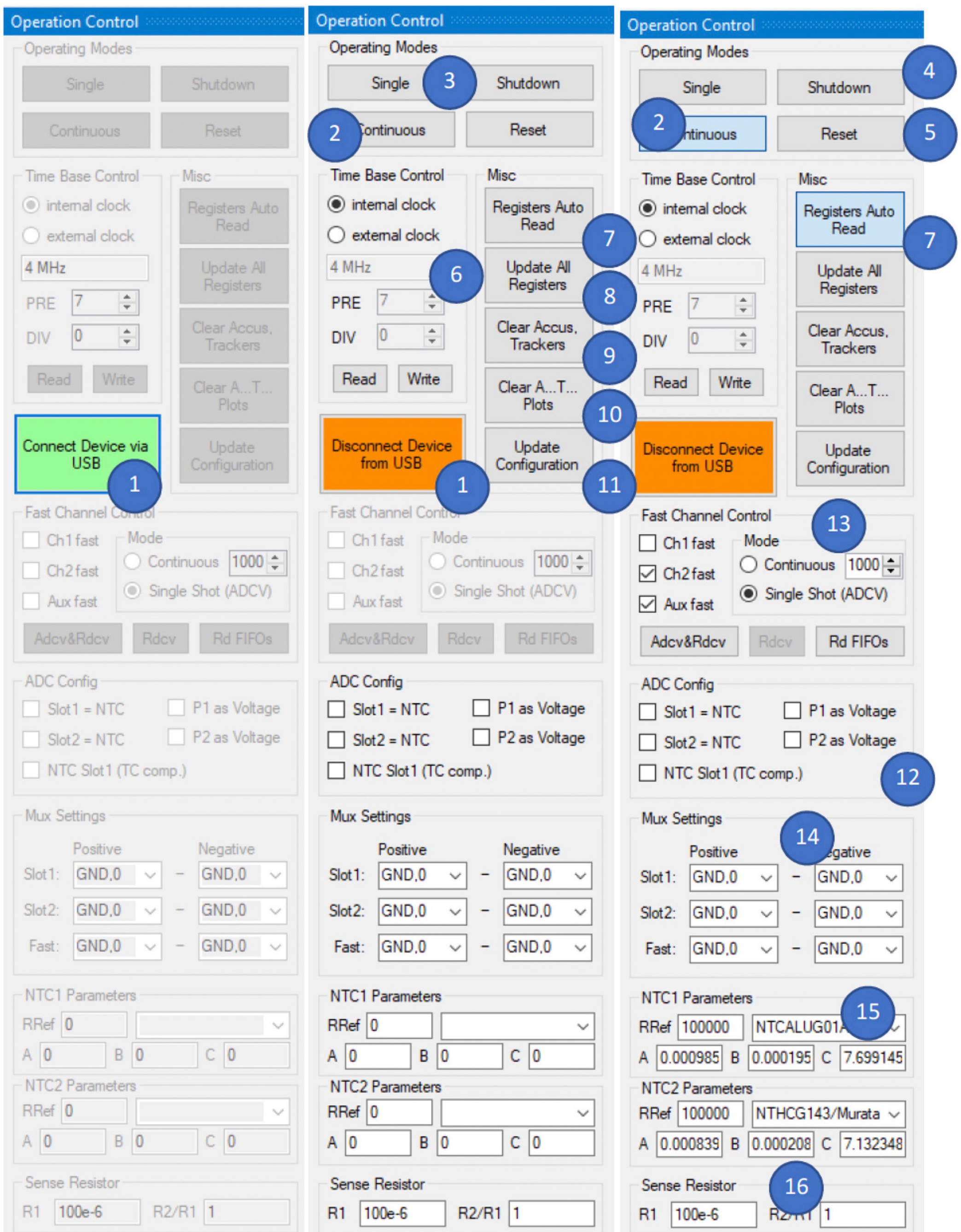


Figure 20. Operation Control Functions

LTC2949 WINDOWS GUI USAGE

In case the GUI is not connected to the device, all controls but the green "Connect Device via USB" button are disabled. Hit the green button to establish the connection. A window is shown allowing to specify the COM port of the Linduino, the used chip select (CS) line and the optional configuration of attached cell monitors. In case the dual LTC6820 demo board (DC2792) is used, the CS configuration can be set to Aux or Main to allow usage of one of the two isoSPI ports of DC2792. For any other hardware setups, where DC2792 is not used, CS must be set to Main.

Once the connection is established all sub-windows are enabled and the GUI will be ready to control the device. The COM port used for communication will be reported in the Status Bar.

The buttons labeled "Continuous", "Shutdown" and "Register Auto Read" are two-state-buttons, meaning, they change color if they are enabled (highlighted, e.g., blue) or disabled (default e.g., grey). Figure 20 shows the GUI running continuous measurements and Register Auto Read enabled.

Shutdown will put the device into a low power state. Any serial transaction to the device will wake it up immediately. For this reason, Register Auto Read will always be disabled before issuing the shutdown command to the device. The GUI will report this as shown in Figure 21.

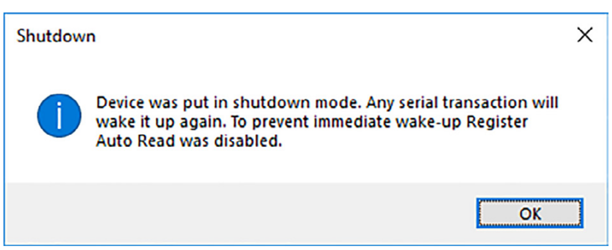


Figure 21. GUI: Sleep/Shutdown Note

Most of the GUI controls will take effect immediately once they are clicked or a value is entered. The only exception is the Time Base Control. Any changed configuration and entered value will only take effect after the Write button is clicked. If Register Auto Read is enabled, the Time Base Control values won't be updated if the user is making changes to it. Any changed values will be discarded and replaced by the current device values if the Write button is not clicked. If Register Auto Read is disabled, a click on the Read button will get the current values from the device.

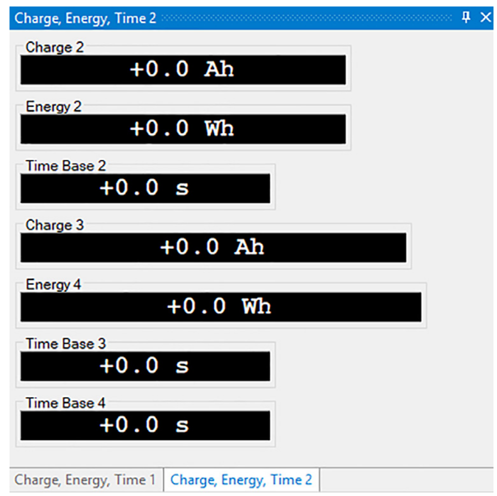


Figure 22. GUI: Multimeters

MULTIMETERS

The GUI shows all measurement quantities in digital multimeter styles. As default all multimeter views are enabled in the View menu, but accumulators C2, E2, TB2, C3, E4, TB3, TB4 are hidden under the Charge, Energy, Time 2 tab.

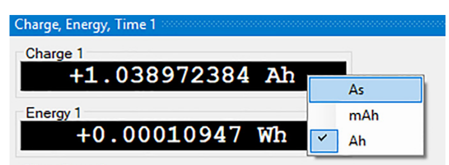


Figure 23. GUI: Charge, Energy, Time

Accumulated quantities are shown in units of h, Wh, Ah by default. The units can be changed by right mouse clicking on a specific multimeter and selecting a different unit from the opened context menu. The Charge, Energy, Time

Rev. 0

LTC2949 WINDOWS GUI USAGE

1 multimeter in Figure 23 shows the context menu for unit selection.

The Charge, Energy, Time multimeters include a calculated mean power display. This value is calculated by the GUI using Energy E1 and Time Base TB1 from the device and equals E1/TB1.

PLOTS/DATA COLLECTION

The GUI provides several plots of collected data. There is a plot for slow channel non-accumulated measurements such as current, voltage, power and temperature. Another plot is for accumulated charge. There is also a plot for energy over time and a plot for fast channel measurements. These plots will collect data whenever the device is in continuous measurement mode and measurement results are read from the device either by a manual read (e.g., Update All Registers or read via the register map) or by the Registers Auto Read function.

The basic steps to start data collection in Register Auto Read mode are:

- Set continuous conversion.
- Enable Register Auto Read.
- Optional: clear device's accumulators, trackers and GUI's plots to have a clean start for data acquisition.

The order of those steps does not matter. Any time Continuous and Auto Register Read are activated, data collection is performed automatically.

By default, the plots do not show all available inputs/channels. Some channels are hidden but may be enabled via the plot's context menu, which can be shown by right mouse clicking in the plot area. The sub-menu Channels will list all available data inputs, see Figure 24 as an example for the fast and slow channel plots.

The plot context menu's functions are:

1. Clear All plot data.
2. Zoom.
3. Zoom All Plots.
4. Export plot data to Clipboard as CSV (Comma-separated values).
5. Export plot data to File in CSV format.

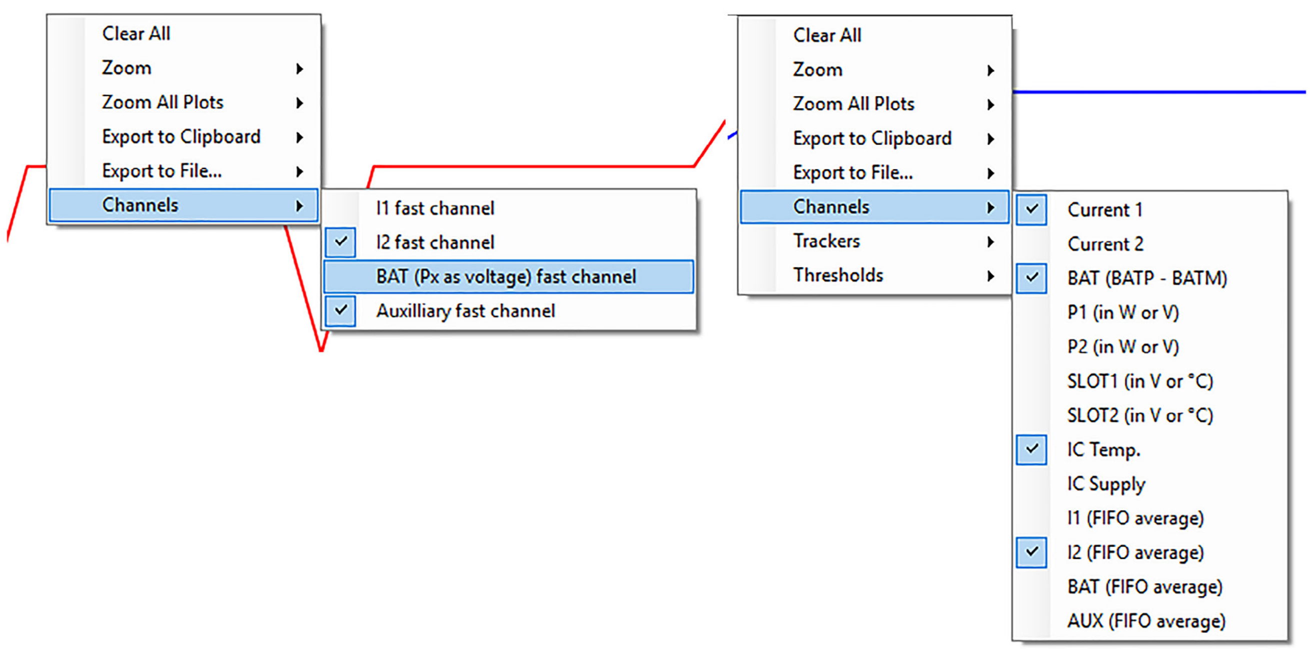


Figure 24. GUI: Plot Window's Context Menu, Select Channels

LTC2949 WINDOWS GUI USAGE

6. Display/Hide Channels.
7. Display/Hide Trackers (minimum/maximum values, not available for fast channel and not for accumulated quantities like charge, energy).
8. Display/Hide Thresholds (low/high alert values, not available for fast channel).

PLOT ZOOM FIT

The plots may be zoomed in different ways. "Zoom Fit All" will fit all data except trackers and thresholds into the plot area. "Zoom Fit X-Axis/Y-Axis" will do the same for X/Y axis only. "Zoom Fit Limits" will again fit all axes but will include trackers and thresholds. The latter is helpful to show the device's thresholds which may be far above/below typical measurement values. The low thresholds, for example, are set to the most negative register value. In case of power the threshold's low (P1TL) default value is –7650.41 Watts (assuming 100µΩ shunt).

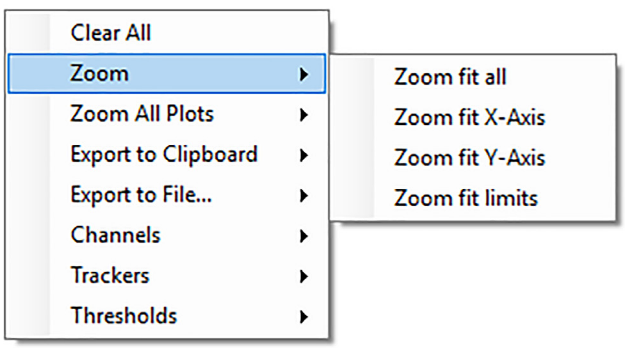


Figure 25. GUI: Plot Window'S Context Menu, Zoom

The Menu "Zoom All Plots" provides the same functionality but applies the zooming to all plots at the same time.

PLOT PAN, ZOOM, LABELS

The following mouse gestures are supported by the plot windows.

1. Pan vertically by right mouse clicking the Y-Axis.
2. Pan horizontally by right mouse clicking the X-Axis.
3. Pan vertically and horizontally by right mouse clicking the plot area. This will pan the first visible channel only.
4. Show label with data point values by left mouse click on a data line.
5. Zoom vertically by rotating mouse wheel over the Y-Axis.

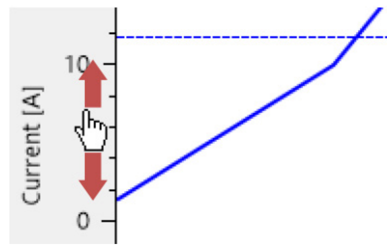


Figure 26. Plot Pan Vertically by Right Mouse Clicking on Y-Axis

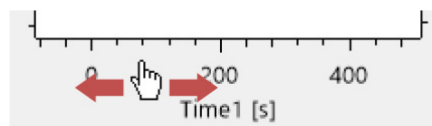


Figure 26. Plot Pan Horizontally by Right Mouse Clicking X-Axis

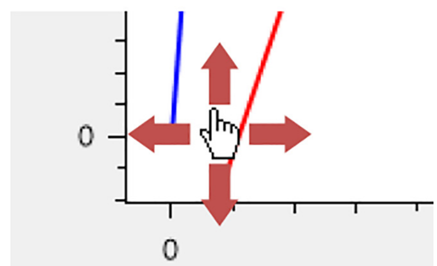


Figure 27. Plot Pan Vertically and Horizontally by Right Mouse Clicking in Plot Area

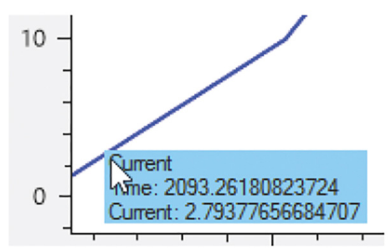


Figure 28. Plot Show Label by Left Mouse Clicking on Data Line

LTC2949 WINDOWS GUI USAGE

6. Zoom horizontally by rotating mouse wheel over the X-Axis.
7. Zoom vertically and horizontally by rotating mouse wheel over the plot area. This will zoom the first visible channel only.

See also Figure 26 thru Figure 28 on how to pan, zoom and show labels.

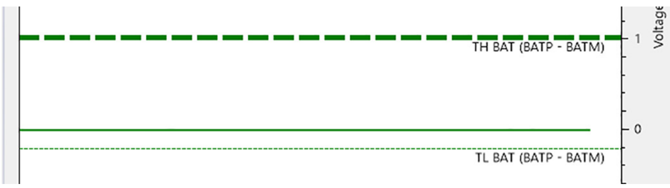


Figure 29. Voltage Plot, Dragging of TH/TL BAT Indicated by Bold Threshold Line

CHANGE THRESHOLDS ON-THE-FLY

Once the thresholds are visible, they can be dragged by the mouse and set to a new desired value. Dragging is indicated by a bold threshold line in the plot, see Figure 29. The new value is written to the device immediately. The threshold's visibility may be changed in the context submenu Thresholds.

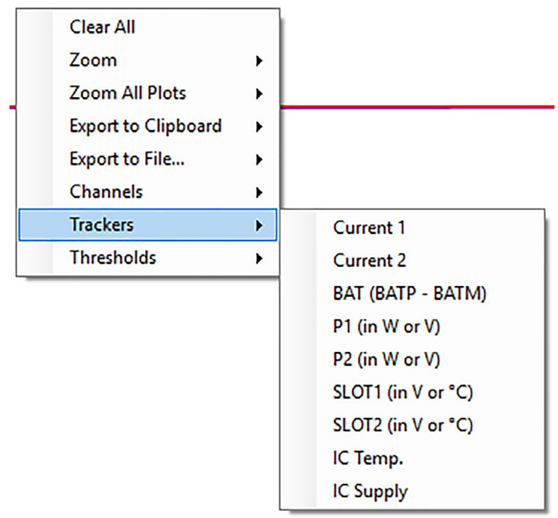


Figure 30. Plot Window Show Min/Max Trackers

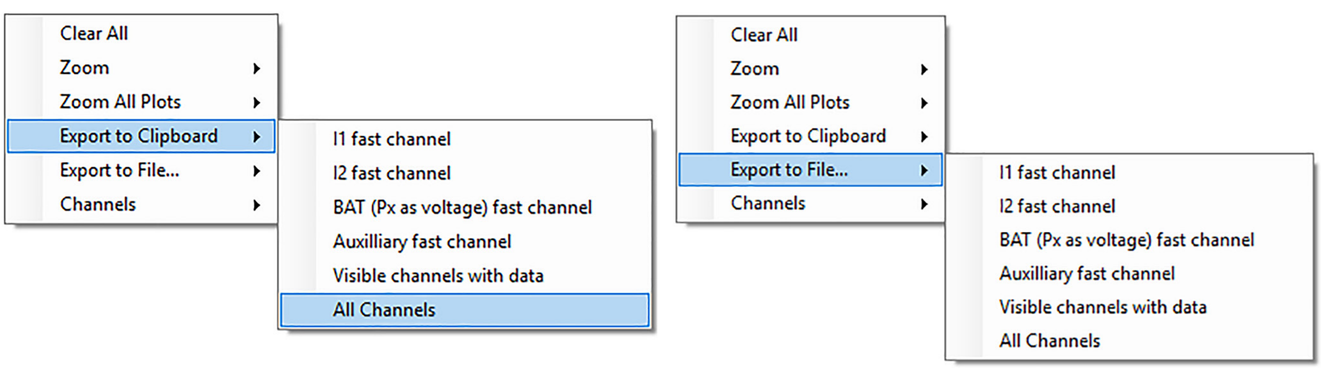


Figure 31. Plot Window Export Data

LTC2949 WINDOWS GUI USAGE

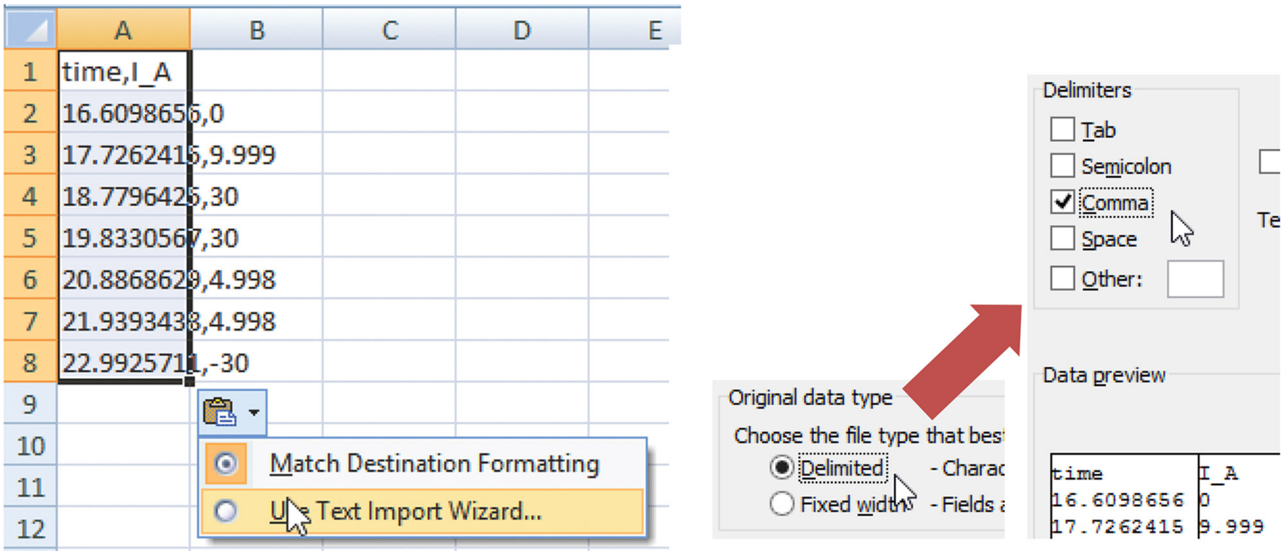


Figure 32. Import Plot CSV Data Into Excel

PLOT TRACKERS

The trackers show the minimum and maximum values of the non-accumulated quantities (Current, Voltage, etc.). Overwriting trackers is not possible from the plot window but can be done in the register map control, see the Register Map section.

EXPORT PLOT DATA

There are two ways to export the collected data as CSV (comma-separated values) from the plot. Export to Clipboard will copy the selected channel data to the clipboard to be able to paste it to some other program like MS Excel. Export to File allows the user to choose a file to store the data. See Figure 31.

The CSV data can be easily imported into MS Excel using the Text Import Wizard as shown in Figure 32.

PLOT TIME AXIS

All data is plotted over time, but the meaning of time differs for the two plots. Non-accumulated quantities (current, voltage, etc.) are plotted over real time of the PC. The time starts when the first values are read from the device while in continuous mode or when the plots are cleared. The accumulated quantities charge and energy are plotted over the corresponding time base. Whenever a triplet of charge, energy and time of any of the two accumulator sets is read, it is added to the plot. Charge, energy and time may be modified during measurement either by a direct write or by issuing the Clear (Clear Accus, Trackers) command. Doing so will result in a jump in the plot. If the time base after this modification is lower than the last time read, the plot will even go backwards. If this behavior is not desired, it is recommended to clear accumulators and trackers together with the plot data (Clear Accus, Trackers, Plots) or clear the plot data manually (Clear All from the plot's context menu).

REGISTER MAP

The Register Map allows direct access to all device Registers/Values (see Figure 33). All data is shown in a table where each cell corresponds to a single-byte register or multiple-byte registers grouped to a single device value like charge, energy and current.

EDIT REGISTER VALUES

Register values can be edited either by selecting a cell and typing a new value or by double clicking on the cell to modify the current value. New values are written to the

LTC2949 WINDOWS GUI USAGE

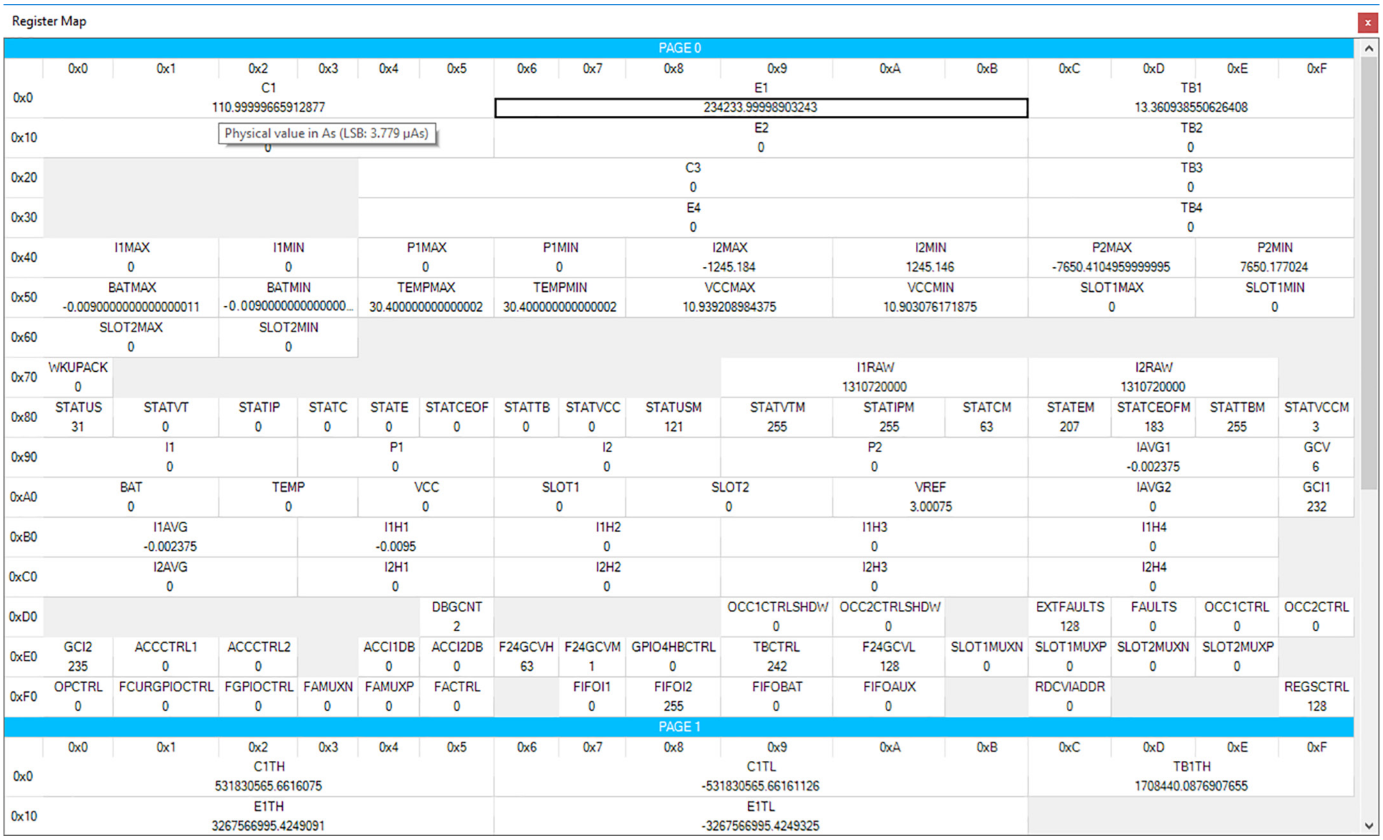


Figure 33. Register Map Control

LTC2947 after pressing enter or leaving the cell. An entered value may be discarded by pressing the Escape button.

REGISTER CONTEXT MENU

By clicking on any register header (e.g., Status) or right clicking on any register value (shown underneath the header), a context menu with the functions is shown:

1. Change cell's display format.
2. Show cell/register details.
3. Write cell/register value to device.
4. Read cell/register value from device.
5. Select all cells.
6. Export all to Clipboard.

Format and Details will be explained in the following paragraphs. The Write and Read command will immediately write or read the selected cell value to or from the device.

Select All will select all the cells. It is also possible to select a range of cells or some individual cells by using Ctrl or Shift key plus the mouse button. The procedure is the same as for other Windows programs like MS Excel, for example. The purpose of selecting more than one cell is to read and write or change the formats of several cells at once.

DISPLAY FORMAT SELECTION

Each cell's display format may be set to either Hexadecimal, Decimal, Binary or Physical. The Physical format (in A, As, W, V, etc.) is only available for cells having a physical value representation as the measurement, tracking and threshold values.

LTC2949 WINDOWS GUI USAGE

The default cell format is Decimal. All cell values may be changed at once to the physical format, for example. This is done by first selecting all cells, either using the shift key plus mouse button or by using the Select All function from the context menu. Then right click on any of the selected cells to open the context menu and set the new format.

The binary and hexadecimal formats are indicated by a 0b and 0x suffix. The physical and decimal format can be distinguished by the tooltip that is shown after the mouse has rested for a short while over the cell value. This tooltip will show the corresponding unit if it is a physical value or the corresponding LSB value for all the other formats.

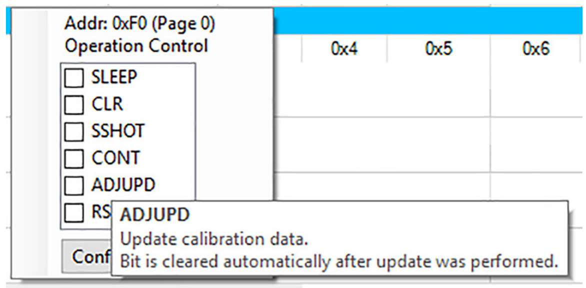


Figure 34. Register Details View, OPCTRL Example

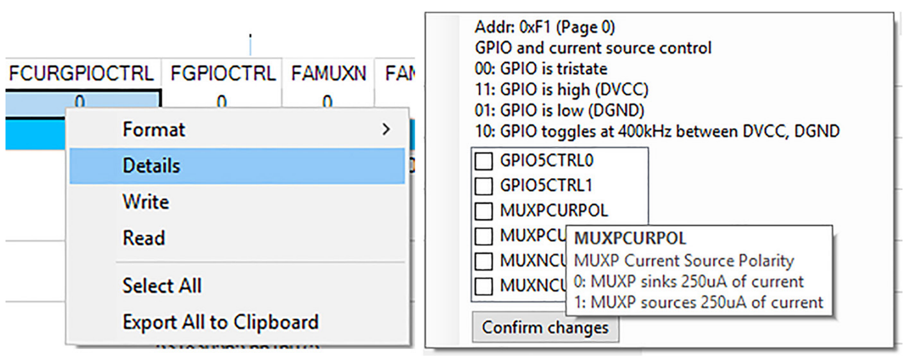


Figure 35. Register Details View, FCURGPIOCTRL Example

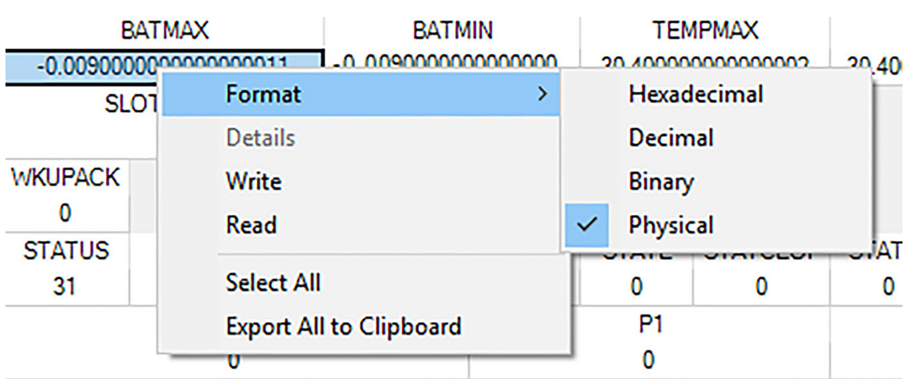


Figure 36. Change Format of Register Values

Rev. 0

LTC2949 WINDOWS GUI USAGE

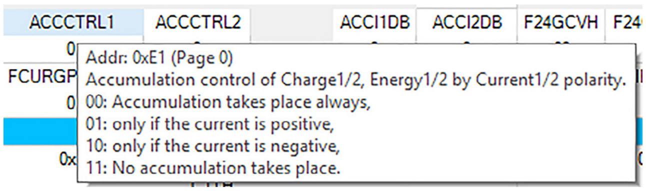


Figure 37. Tooltip Showing Register Description

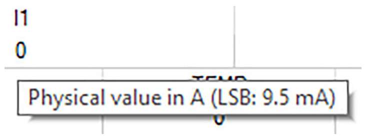


Figure 38. Tooltip Showing Register Value LSB Size

REGISTER MAP TOOLTIPS

The register map includes a lot of helpful information shown in tooltips. They appear after the mouse has rested for a short while over an element of the register map.

REGISTER DETAILS

For registers that have bits assigned to functions, the register context menu provides access to a details view.

Figure 34 shows as an example the access to the details view of OPCTL. Click on the name OPCTL or right click on the cell value, select Details from context menu. All bits with defined functions are listed with check boxes next to them indicating if the corresponding bit is set or not. Tooltips show more detailed descriptions for each bit. Any changes may be performed by clicking on the check boxes. After clicking on Confirm changes the new value is assigned to the cell, still allowing to make changes to it. The new value will be written to the device after hitting the ENTER key or leaving the cell (e.g., clicking somewhere else in the register map).

AUTO READ MENU

The Auto Read menu allows users to specify which registers/values are read from the device if Register Auto Read is enabled. The update time may also be specified here.

Once Register Auto Read is clicked the first time, it will automatically make the configuration to read Current, Voltage, Power, Temp. from slow channel and set the update time to 100ms.

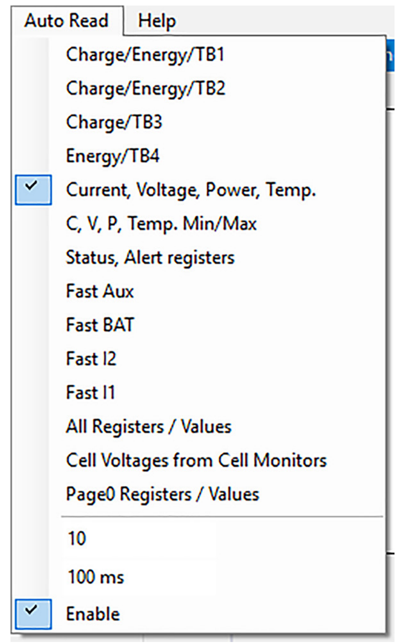


Figure 39. Auto Read Menu

The update time may be adjusted between 1ms and 10s. An error will be reported if the entered value is outside this range.

Note, that the update time is not the period at which data is read from LTC2949, but rather the delay between performed updates. This means the period will always be longer than this time and depends also on the performance of the used PC. Still, the main limitation of update speed is the serial interface to the Linduino. Update speed can usually be increased by selecting only the measurements from the Auto Read menu that are of interest for the current evaluation purpose and disable all the others. It is then still possible to manually read other values from the Register Map, even while Auto Read is enabled.

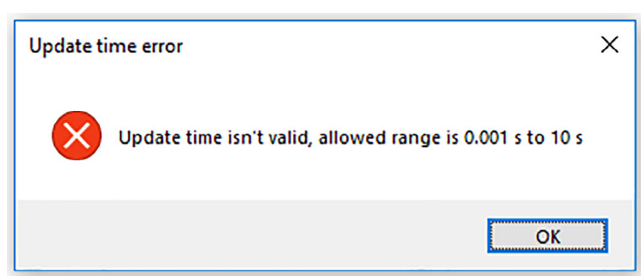


Figure 40. Valid Range of Update Time

LTC2949 WINDOWS GUI USAGE

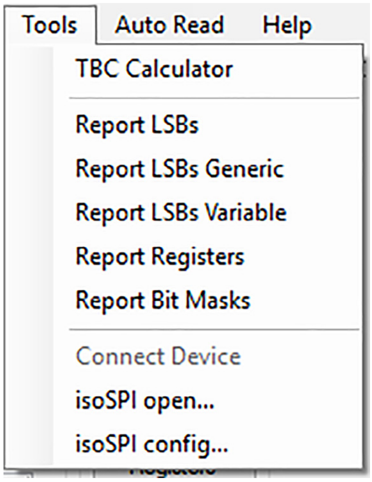


Figure 41. Tools Menu

TOOLS MENU

The Tools menu provides access to a stand-alone TBC Calculator and to report functions that generate C-compliant source code for register, bit and LSB definitions.

BASIC OPERATION EXAMPLE

The following shows an example of basic configuration and operation of LTC2949 with the GUI.

1. Enable default Layout of the GUI.

   a. View → Reset Default Layout.

2. Connect to device from operation control.
3. Enable P2 as voltage from ADC config.
4. Click Update Configuration from Operation Control/Misc.
5. Click Continuous from Operation Modes.
6. Enable Ch2 fast and AUX fast from Fast Channel Control.
7. From MUX Settings set Fast to V_{REF2} – GND (any other channel is also possible, e.g., if some external signal is applied).
8. Go to "Auto Read" menu and select.

   a. Fast all.

   b. Page0 Registers/Values.

   c. Enable.

9. The Plot: "Fast Channel" and the "Fast Channel Multimeter" will show the fast measurements. From the fast channel plot context menu select "Channels" →"BAT…" to also enable plot of battery voltage.
10. From MUX Settings set Fast to V8 – GND (Note: V8 is one of the dual purpose pins GPIO1).
11. From the center low Element of the GUI Window select "Register Map", from which all registers of LTC2949 can be accessed.
12. Scroll down little bit to see the register row "OPCTRL, FCURGPIOCTRL, FGPIOCTRL, …."
13. Right click on FGPIOCTRL.
14. Enable GPIO1CTRL1 (all other disabled).
15. Hit Enter (from you keyboard) two times to confirm changes.
16. The new configuration is now written to LTC2949 and activates the toggling on GPIO1 (V8)
17. Watch the effect in the Plot: fast channel.
18. Change GPIO1 setting from register map from FGPIOCTRL: Values can also be just entered in the cell of FGPIOCTRL and confirmed by pressing the enter key (Figure 47).

    a. Alternate between 0, 1, 2, 3 and watch the effect in the Plot: fast channel, Auxiliary fast channel (Figure 48).

19. Note: Setting the GPIO control bits to 1,1 (decimal 3) will enable the output driver to go to DVCC which can be above the full-scale range of the ADC, depending on the power supply voltage of AVCC, DVCC. The ADC will just saturate to its full-scale value.

LTC2949 WINDOWS GUI USAGE

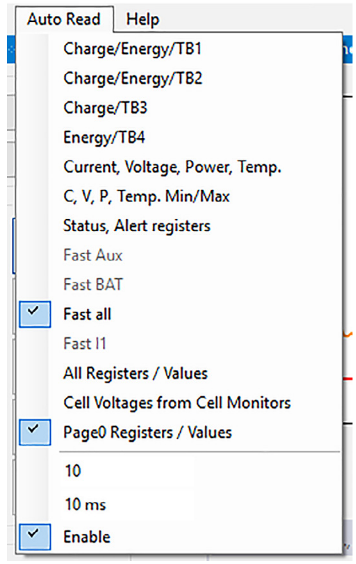


Figure 42. Auto Read Menu for Basic Operation Example

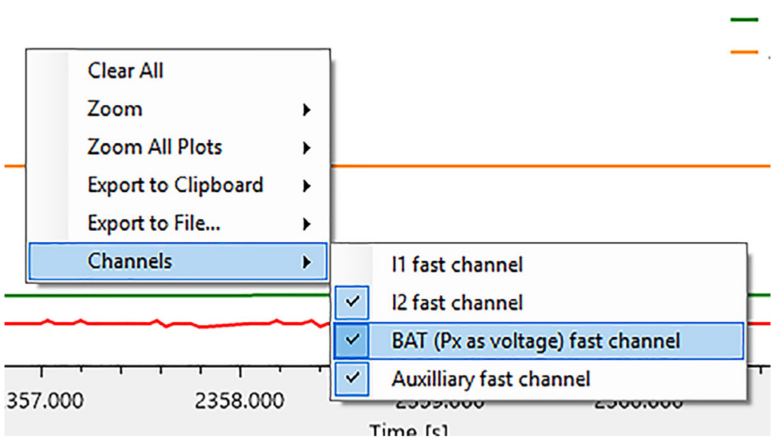


Figure 44. Plot Channels Menu for Basic Operation Example

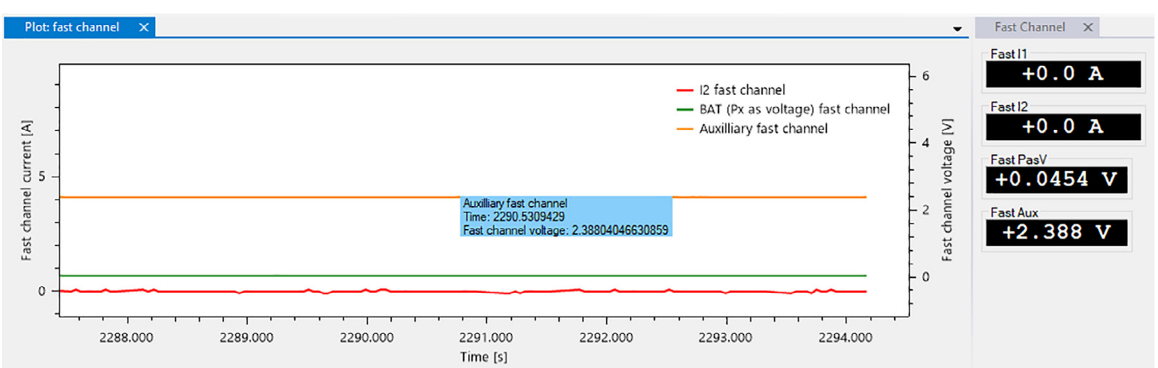


Figure 43. Fast Channel Plot for Basic Operation Example

LTC2949 WINDOWS GUI USAGE

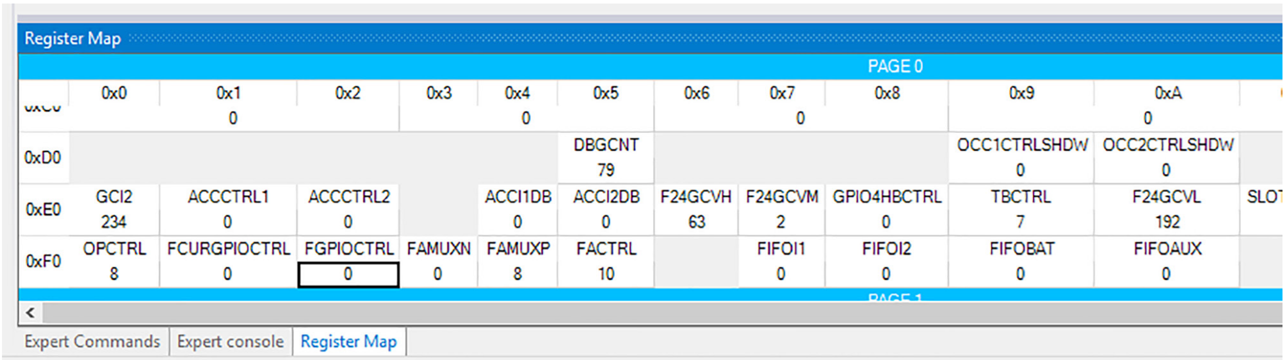


Figure 45. Register Map Access for Basic Operation Example

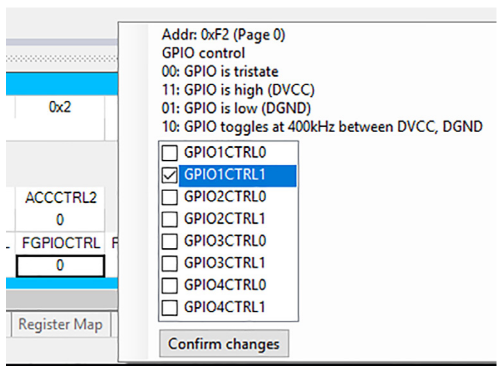


Figure 46. FGPIOCTRL Register Details View to Control GPO1 for Basic Operation Example

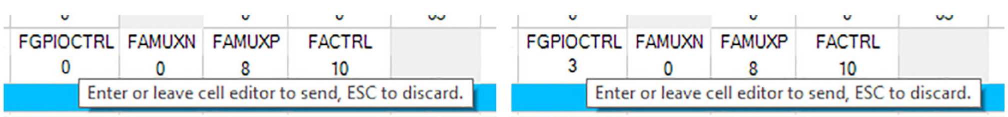


Figure 47. FGPIOCTRL Register Direct Access for Basic Operation Example

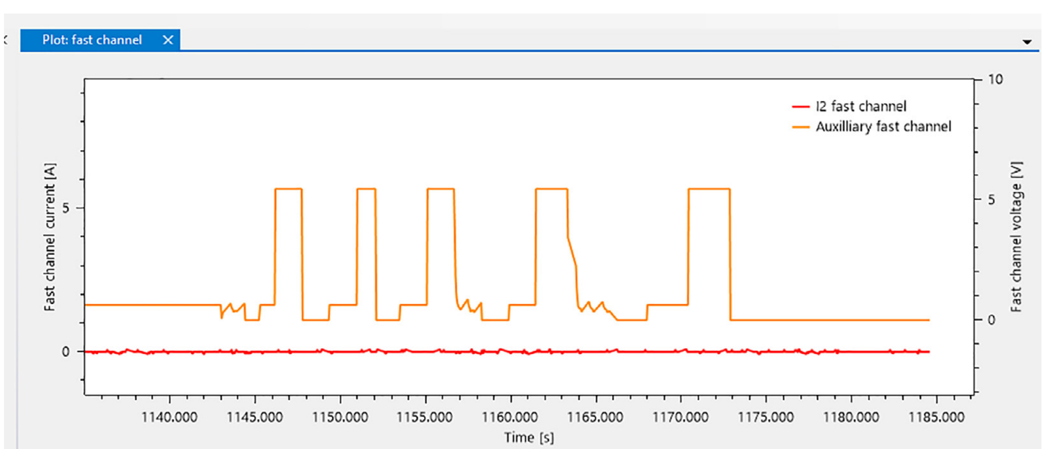


Figure 48. Fast Channel Plot for Basic Operation Example Showing Alternating AUX Measurements

Rev. 0

APPENDIX A: USAGE OF SPI ISOLATOR INSTEAD OF ISOSPI

If required by customer, it is also possible to use a SPI isolator instead of LTC2949's integrated isoSPI. One example is the ADuM141E which could be inserted into the SPI interface using the Linduino QuikEval connector J4 and then connect to the Linduino or any other host controller, as shown in in Figure 49. In this scenario IOV_{CC} of the LTC2949 is shorted to BYP2 which will then also supply V_{DD1} of the ADuM141E. Usage of other SPI isolators with additional chip select lines is also possible, which would allow to connect additional SPI slaves together with LTC2949. Please contact Analog Device for details on this evaluation setup.

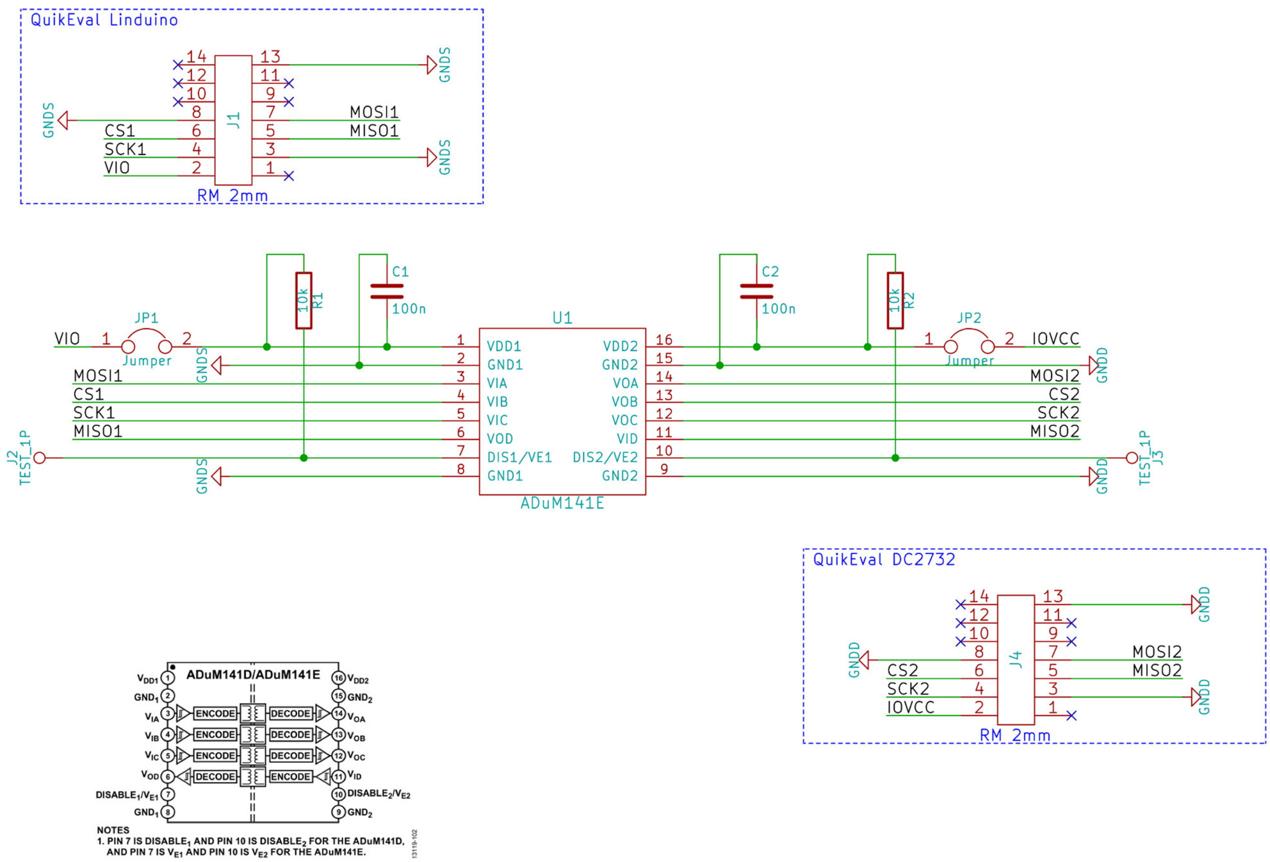


Figure 49. ADUM141E SPI Isolator Schematic

APPENDIX B: CAN BASED EVALUATION

The DC2732A can be used in conjunction with the Linduino and the DC2617A in a CAN based evaluation and test environment, which is described in the following paragraphs.

HARDWARE REQUIREMENTS

- DC2617A (CAN to isoSPI shield).
- DC2732A (LTC2949 demo board).
- Ethernet cable.
- 5V power supply.
- Linduino (DC2026).
- Power to the Linduino (either via USB or AC ADAPTOR IN).
- Optional: 120Ω reset pull-up on Linduino between RST and IOREF to prevent Linduino reset in case Linduino is connected to the USB port of a PC, but its COM port is not opened.

SOFTWARE REQUIREMENTS

- Some CAN analyzer or CAN master to control the DC2732A_CAN and receive messages (e.g., BUSMASTER).
- DC2732A_CAN.ino programmed to the Linduino.
- DC2732A_CAN.dbf loaded to the CAN analyzer software.
- Optional: Serial terminal software (e.g., Arduino IDE's serial monitor) connected to the Linduino's COM port.

SETUP THE HARDWARE

Put the DC2617A on top of the Linduino. Connect LTC2949 demo board via an Ethernet cable to the RJ45 connector. Connect power to the Linduino (either via USB or AC ADAPTOR IN). Set JP1 on DC2617A to ARD and JP2 to ON to enable 120R CAN bus termination.

Note 1: If the Linduino is connected to a USB port of a PC, depending in the operating system, the board will make periodic resets. This is caused by Windows and the USB COM port driver accessing the COM port assigned to the Linduino. This can be prevented by either making a connection via a serial terminal to the COM port or by connecting a 120R pull-up on Linduino between RST and IOREF. If a plain external USB power supply or the "AC ADAPTOR IN" is used to power the Linduino, this problem does not appear.

Note 2: 5V supply input of the LTC2949 demo board may be connected to V_{CC}, GND turrets of DC2617A. Still this is not recommended as it shorts the galvanic isolation between LTC2949 and DC2617A.

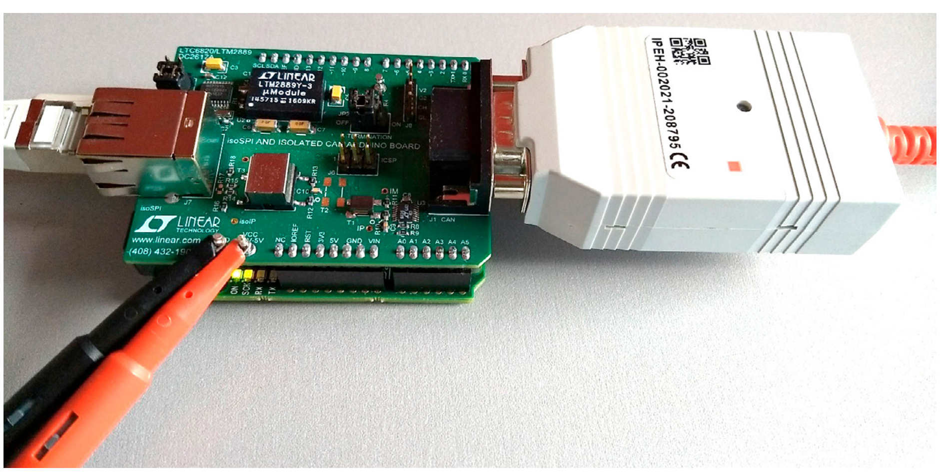


Figure 50.

Rev. 0

APPENDIX B: CAN BASED EVALUATION

Connect 5V to 12V supply to DC2732A demo board. Connect Ethernet patch cable between RJ45 connectors of DC2732A and DC2617A.

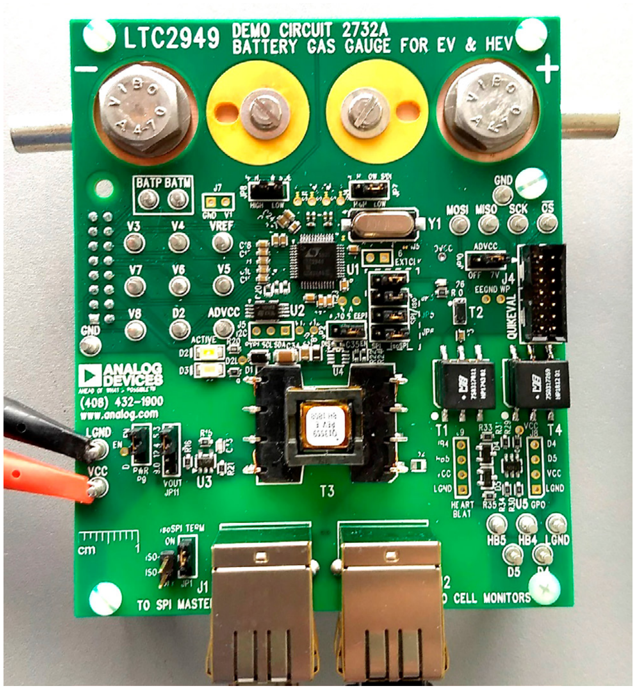


Figure 51. DC2732A Demoboard

DC2732A_CAN: LTC2949 CAN FIRMWARE

DC2732A_CAN's CAN baud rate is set to 500k Bit/s but may be changed via the Serial monitor.

CAN Messages Overview

The DC2732A_CAN uses CAN messages listed in Table 1. See also DC2732A_CAN.dbf.

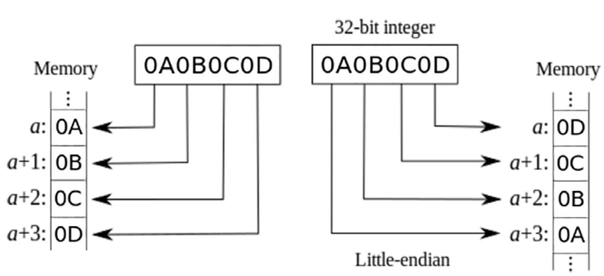


Figure 52. FSSHT, FIFOAUX, FIFOBAT, FIFOI2, FIFOI1, FIFOAUXAVG, FIFOBATAVG, FIFOI2AVG, FIFOI1AVG Use the Little-Endian Representation for Their Signal Data (All Other Messages Use Big-Endian Format)

The DC2732A_CAN uses CAN messages listed in Table 1. See also DC2732A_CAN.dbf / DC2732A_CAN.dbc for scaling factors and units.

All current dependent measurement quantities have to be scaled with $1/R_{SNS}$, where R_{SNS} is the resistance of the shunt resistor.

Example ACC_C1/ACC_E1:

The scaling factors and units from the .dbc file are:

```
BO_ 293 ACC_C1: 6 Vector__XXX
 SG_ C1 : 7|48@0- (0.377887,0) [-5.31829e+013|5.31829e+013] "uVms" Vector__XXX
BO_ 294 ACC_E1: 6 Vector__XXX
 SG_ E1 : 7|48@0- (2.32175,0) [-3.26757e+014|3.26757e+014] "uVVms" Vector__XXX
BO_ 295 ACC_TB1: 4 Vector__XXX
 SG_ TB1 : 7|32@0- (0.397777,0) [-8.5422e+008|8.5422e+008] "ms" Vector__XXX
```

The factors are taken from LTC2949 data sheet: *Table 26. Accumulated Results Register Parameters for Use with Crystal or Internal Clock* but scaled for units ms and µV. Charge is given in microvolts times milliseconds, Energy is given in microvolts times volts times millisecond. The conversion to As/Ws in case of a 100µΩ shunt is:

```
Charge = ACC_C1[uV*ms] / 1e6[uV/V] / 1000[ms/s] / 100e-6[Ohms] = ACC_C1 * 1e-5 [Vs/Ohms = As]
Energy = ACC_E1[uV*V*ms] / 1e6[uV/V] / 1000[ms/s] / 100e-6[Ohms] = ACC_E1 * 1e-5 [V2 s / Ohm = VAs = Ws]
```

Rev. 0

APPENDIX B: CAN BASED EVALUATION

Table 1. CAN Messages with Measurement Values Send by DC2732A_CAN

ID	NAME	SIGNAL	T, [BYTES], BITS	CFG2949 FLAG	DESCRIPTION
0x110	IP1V	I1	Int, [2:0], 24	MEAS_I1	Current1
		P1	Int, [5:3], 24	MEAS_P1	Power1
		BAT	Int, [7:6], 16	MEAS_BAT	Battery Voltage
P1 Will Be Reported by This Message If P1ASV Flag is Not Set. Otherwise Battery Voltage by Power ADC Will Be Reported by PASV Message.					
0x111	IP2T	I2	Int, [2:0], 24	MEAS_I2	Current2
		P2	Int, [5:3], 24	MEAS_P2	Power2
		TEMP	Int, [7:6], 16	MEAS_TEMP	IC Temperature
P2 Will Be Reported by This Message If P2ASV Flag Is Not Set. See Also Note Above.					
0x112	SVR	SLOT1	Int, [1:0], 16	MEAS_SLOT1	Slot1 voltage
		SLOT2	Int, [3:2], 16	MEAS_SLOT2	Slot2 voltage
		V_{REF}	Int, [5:4], 16	MEAS_VREF	Reference Voltage (V_{REF} Pin)
		V_{CC}	Int, [7:6], 16	MEAS_VCC	IC Supply Voltage
SLOT1/SLOT2 Voltage Will Be Reported by This Message If SLOT1NTC/SLOT2NTC Flag Is Not Set. Otherwise Temperature by NTC Measurement Will Be Reported by SLOTSNTC Message.					
0x113	SLOTSNTC	NTC1	Int, [1:0], 16	MEAS_SLOT1, SLOT1NTC	NTC Temperature Measurement via Slot1
		NTC2	Int, [3:2], 16	MEAS_SLOT2, SLOT2NTC	NTC Temperature Measurement via Slot2
NTC1/NTC2 Temperature Will Be Reported by This Message If SLOT1NTC/SLOT2NTC Flag Is Set.					
0x114	PASV	P1V	Int, [2:0], 24	MEAS_P1, P1ASV	Power1 as Voltage
		P2V	Int, [5:3], 24	MEAS_P2, P2ASV	Power2 as Voltage
P1V/P2V (Power ADC in Voltage Mode) Will Be Reported by This Message If P1ASV/P2ASV Flag Is Set.					
0x115	No used				
0x116	No used				
0x125	ACC_C1	C1	Int, [5:0], 4 8	ACC	Charge1
0x126	ACC_E1	E1	Int, [5:0], 48	ACC	Energy1
0x127	ACC_TB1	TB1	Int, [3:0], 32	ACC	Time1
0x128–0x12E					Optional, See Below.
C2, C3, E2, E4, TB2, TB3, TB4 are not reported as default. Values can still be read via RAWRW if required. Alternatively, if dedicated CAN messages for those additional accumulators are needed, they can be enabled within the DC2732A_CAN.ino source code via the preprocessor definition ENABLE_CAN_ID_ACC_C2_TO_TB4. Those messages will have the CAN IDs 0x128–0x12E.					

Notes:

1. "T, [Bytes], Bits' indicates the type, byte order (typically big-endian) and number of bits. For example, "Int, [0:2], 24" refers to a 24-bit big-endian integer value stored in CAN message data bytes 0 to 2 where byte 0 is the MSB.

2. CFG2949 flag refers to the Boolean signal within CFG2949 that enables the related measurement signal. Above CAN messages will be send if any of the related CFG2949 flags are enabled. Signals within CAN messages that are not enabled will report the most negative value.

APPENDIX B: CAN BASED EVALUATION

Table 2. CAN Messages Send to DC2732A_CAN to Configure Fast Measurements

ID	NAME	SIGNAL	T, [BYTE], BITS	DESCRIPTION
0x11D	FASTRQ	SAMPLES	UInt, [1:0], 16	Number of samples to be read on demand or per cycle. Or number of samples of moving average filter for I2.
		FAST_MEAS_ PERIOD	UInt, [2], 8	Fast measurement cycle time in milliseconds. If greater than zero, fast measurements are reported periodically. If zero a single measurement (on demand) is reported.

Fast conversion request. For fast continuous measurements SAMPLES is the number of samples to be read from FIFO. If SAMPLES is 0 or 1, no samples will be read from the FIFO, instead only the latest converted sample is read via RDCV command and reported.

For fast continuous measurements with FAST_MEAS_PERIOD = 1, no samples will be read from the FIFO, instead only the latest converted sample is read via RDCV command and reported.

For fast single shot measurements SAMPLES sets the number I2 samples moving average filter. Set to 0 or 1 to disable. If set to >1 and <129 the moving average filter of I2 fast samples is enabled and filtered I2 fast single shot result will be transmitted via FSSHTMA. This is for a typical configuration where I1 is doing slow precision measurements for coulomb counting and only I2 (and optional BAT and AUX) is doing fast conversions.

Table 3. CAN Messages Send by DC2732A_CAN to Report Fast Measurement Results

<table>
<tr><th>ID</th><th>NAME</th><th>SIGNAL</th><th>T, [BYTE], BITS</th><th>DESCRIPTION</th></tr>
<tr><td>0x118</td><td>FIFOI1</td><td>S0</td><td>Int, [1:0], 16</td><td rowspan="4">Four fast continuous conversion results.</td></tr>
<tr><td>0x119</td><td>FIFOI2</td><td>S1</td><td>Int, [3:2], 16</td></tr>
<tr><td>0x11A</td><td>FIFOBAT</td><td>S2</td><td>Int, [5:4], 16</td></tr>
<tr><td>0x11B</td><td>FIFOAUX</td><td>S3</td><td>Int, [7:6], 16</td></tr>
<tr><td colspan="5">N samples read from the FIFO will be reported by N/4 CAN messages. If N is not a multiple of 4, the last message will be truncated accordingly (e.g.,if 10 samples are reported, the last message will contain only S0 and S1 and thus only have 4 bytes). Signal format is little endian.</td></tr>
<tr><td>0x11C</td><td>FSSHT</td><td>I1</td><td>Int, [1:0], 16</td><td rowspan="4">Fast single shot conversion result or last conversion result in fast continuous mode.</td></tr>
<tr><td></td><td></td><td>I2</td><td>Int, [3:2], 16</td></tr>
<tr><td></td><td></td><td>BAT</td><td>Int, [5:4], 16</td></tr>
<tr><td></td><td></td><td>AUX</td><td>Int, [7:6],16</td></tr>
<tr><td colspan="5">This message is used to report fast single shot conversion results. In case fast continuous mode is activated and 0 or 1 (= signal SAMPLES of message FASTRQ) samples are requested, this message will be used to report the last acquired sample from the fast channel. Signal format is little endian.</td></tr>
<tr><td>0x120</td><td>FIFOI1AVG</td><td rowspan="2">…AVG</td><td rowspan="2">Int, [3:0], 32</td><td rowspan="2">Average of all samples multiplied by 1024 (= fixed point representation with 10 fractional bits)</td></tr>
<tr><td>0x121</td><td>FIFOI2AVG</td></tr>
<tr><td>0x122</td><td>FIFOBATAVG</td><td rowspan="2">LEN</td><td rowspan="2">UInt, [5:4], 16</td><td rowspan="2">Number of samples acquired to calculate the average</td></tr>
<tr><td>0x123</td><td>FIFOAUXAVG</td></tr>
<tr><td colspan="5">The LSB size of the above average signals is those of FIFOI1, FIFOI2, FIFOBAT, FIFOAUX Divided by 1024. Signal format is little endian.</td></tr>
<tr><td>0x124</td><td>FSSHTMA</td><td>I2MA</td><td>Int, [3:0], 32</td><td>Moving average of fast single shot I2 conversion results</td></tr>
<tr><td colspan="5">This message is used to report moving average of fast single shot I2 conversion results. Signal format is little endian.</td></tr>
</table>

APPENDIX B: CAN BASED EVALUATION

Table 4. CAN Message Send by DC2732A_CAN to Signal Errors

ID	NAME	SIGNAL	T, [BYTE], BITS	DESCRIPTION
0x11E	ERR	COMMERR	Bool, [0] (4)	
		FIFOWROVR	Bool, [0] (5)	
		FIFOTAGERR	Bool, [0] (6)	
		OTHER	Bool, [0] (7)	
		TIMEOUT	Bool, [0] (3)	
		FIFOALLERR	Bool, [0] (2)	
		PECERR	UInt, [0] (1:0)	
Error message. Combination (ORing) of error codes (bits [7:2]) and PEC error counter (bits [1:0]). The PEC counter counts the PEC errors when receiving data from the LTC2949 and will stop counting at 3.				

Note: "T, [Byte], Bits" indicates the type and number of bits. For example, "Int, [0], 8" refers to an 8-bit integer value stored in CAN message data byte 0.

Table 5. CAN Message Received by DC2732A_CAN to Configure Measurements

ID	Name	Signal	T, [Byte], Bits	Description
0x11F	CFG2949	CH1FAST	Bool, [0] (0)	CH1 fast enable flag
		CH2FAST	Bool, [0] (1)	CH2 fast enable flag
		AUXFAST	Bool, [0] (2)	AUX fast enable flag
		FCONT	Bool, [0] (3)	fast continuous enable flag (fast measurements will be stored within FIFO, otherwise fast single shot measurements will be performed upon request, see FASTRQ)
		SLOT1NTC	Bool, [0] (4)	SLOT1 temperature measurement via NTC enable flag
		SLOT2NTC	Bool, [0] (5)	SLOT2 temperature measurement via NTC enable flag
If enabled an NTC must be connected to the voltage inputs selected via SLOTxP/N which then will be used for temperature measurements.				
		P1ASV	Bool, [0] (6)	Power1 as voltage enable flag
		P2ASV	Bool, [0] (7)	Power2 as voltage enable flag
		SLOT1P	UInt, [1] (3:0)	Positive voltage input channel for SLOT1
		SLOT1N	UInt, [1] (7:4)	Negative voltage input channel for SLOT1
		SLOT2P	UInt, [2] (3:0)	Positive voltage input channel for SLOT2
		SLOT2N	UInt, [2] (7:4)	Negative voltage input channel for SLOT2
		SLOTFP	UInt, [3] (3:0)	Positive voltage input channel for SLOTF (AUX fast)
		SLOTFN	UInt, [3] (7:4)	Negative voltage input channel for SLOTF (AUX fast)
		MEAS_I1	Bool, [4] (0)	Current1 measurement enable flag
		MEAS_I2	Bool, [4] (1)	Current2 measurement enable flag
		MEAS_P1	Bool, [4] (2)	Power1 measurement enable flag
		MEAS_P2	Bool, [4] (3)	Power2 measurement enable flag
		MEAS_SLOT1	Bool, [4] (4)	Slot1 measurement enable flag
		MEAS_SLOT2	Bool, [4] (5)	Slot2 measurement enable flag

APPENDIX B: CAN BASED EVALUATION

Table 5. CAN Message Received by DC2732A_CAN to Configure Measurements

ID	Name	Signal	T, [Byte], Bits	Description
		MEAS_BAT	Bool, [4] (6)	Battery voltage measurement enable flag
		MEAS_TEMP	Bool, [4] (7)	IC temperature measurement enable flag
		MEAS_VCC	Bool, [5] (0)	IC supply voltage measurement enable flag.
		MEAS_VREF	Bool, [5] (1)	Reference voltage (V_{REF} pin) measurement enable flag.
Any combination of measurement enable flags may be set to				
		NTC1_TYPE	Bool, [5] (3:2)	Selects on of 3 predefined NTC parameters.
		NTC2_TYPE	Bool, [5] (5:4)	

NTCx_TYPE	MPN	A	B	C	R_{REF}
0	NTCALUG01A104F/TDK	9.85013754e-4	1.95569870e-4	7.69918797e-8	100e3
1	NTHCG143/Murata	8.39126e-4	2.08985e-4	7.13241e-8	100e3
2	Reserved	8.39126e-4	2.08985e-4	7.13241e-8	100e3

For NTCx_TYPE = 3, no configuration of NTC parameters will be written to LTC2949 with the CFG2949 message. This allows user defined values provided via a RAWRW message.

		ACC	Bool, [5] (6)	Battery voltage measurement enable flag. If set the accumulators C1, E1, TB1 are reported via CAN messages ACC_C1, ACC_E1, ACC_TB1.
		MEAS_ ENABLE	Bool, [5] (7)	Global measurements enable. Only if this flag is set DC2732A_CAN will do measurements and react on FASTRQ.
		MEAS_ PERIOD	UInt, [6],8	Measurement period in multiples of 100ms. For example, if set to 10 all enabled measurement CAN messages will be send every second. If set to zero messages will be send at the update rate of LTC2949 which is ~100ms. A value of 1 (100ms) may result in a jitter of the actual send message period as the internal update rate of LTC2949 may vary and messages will never be sent before LTC2949 updates its measurement registers. Setting a value of 255 will completely disable reading of slow channel registers. Fast channel values may still be reported.
		FAST_MEAS_ PERIOD	UInt, [7],8	Fast measurement cycle time in milliseconds. If greater than zero, fast measurements are reported periodically. Same as signal FAST_MEAS_PERIOD of CAN message FASTRQ.

Notes:

1.“T, [Byte], Bits” indicates the type and number of bits. For example ,“Int, [0], 8” refers to an 8-bit integer value stored in CAN message data byte 0.

2. For Boolean signals the bit index is given in round brackets, for example, “Bool, [4] (7)” refers to a flag stored at bit position 7 within CAN data byte 4.

3. For signals covering only part of a byte the bit range is given in round brackets, for example, “UInt, [3] (7:4)” refers to an unsigned integer value covering bits [7:4] within CAN data byte 3.

APPENDIX B: CAN BASED EVALUATION

Table 6. CAN Message Send and Received by DC2732A_CAN for RAW LTC2949 Register Access

<table>
<tr><th>ID</th><th>NAME</th><th>SIGNAL</th><th>T,[BYTE],BITS</th><th>DESCRIPTION</th></tr>
<tr><td>0x117</td><td>RAWRW</td><td>WRITE</td><td>Bool, [0] (7)</td><td>Write flag. Set for write commands, clear for read commands. In the responding message send by DC2732A_CAN this flag will be inverted (e.g., after a write send to DC2732A_CAN the WRITE flag will be cleared in the response message)</td></tr>
<tr><td></td><td></td><td>COUNTDOWN</td><td>UInt, [0], (6:1)</td><td>Countdown tag for multi RAWRW CAN message bursts. For RAWRW messages with up to 6 data bytes COUNTDOWN must be zero. If more than 6 bytes must be written in a burst to LTC2949 COUNTDOWN must be decremented with every message till zero for the last message. If the number of bytes is not a multiple of 6 bytes only the first message of the multi RAWRW CAN message bursts may have less than 6 bytes. WRITE, PAGE and ADDR must be equal for all sub-messages of a burst. The following table shows an example how to write a burst with 16 bytes (e.g., a full row in LTC2949's register map).
<table>
<tr><th>MESSAGE</th><th>COUNTDOWN</th><th>CAN DATA LENGTH</th><th>DX SIGNALS</th></tr>
<tr><td>1</td><td>2</td><td>6</td><td>D0 – D1 (4)</td></tr>
<tr><td>2</td><td>1</td><td>8</td><td>D0 – D5 (6)</td></tr>
<tr><td>3</td><td>0</td><td>8</td><td>D0 – D5 (6)</td></tr>
</table>
Messages send by DC2732A_CAN in response to a read command will make the same usage of COUNTDOWN to pack more than 6 bytes into several CAN messages.</td></tr>
<tr><td></td><td></td><td>PAGE</td><td>Bool, [0] (0)</td><td>Register page flag. Set for page 1, clear for page 0.</td></tr>
<tr><td></td><td></td><td>ADDR</td><td>Int, [1], 8</td><td>Register address</td></tr>
<tr><td></td><td></td><td>D0</td><td>Int, [2], 8</td><td>Write: Data byte 0/Read: Length (number of bytes to be read)</td></tr>
<tr><td></td><td></td><td>D1</td><td>Int, [3], 8</td><td>Register byte 1</td></tr>
<tr><td></td><td></td><td>D2</td><td>Int, [4], 8</td><td>Register byte 2</td></tr>
<tr><td></td><td></td><td>D3</td><td>Int, [5], 8</td><td>Register byte 3</td></tr>
<tr><td></td><td></td><td>D4</td><td>Int, [6], 8</td><td>Register byte 4</td></tr>
<tr><td></td><td></td><td>D5</td><td>Int, [7], 8</td><td>Register byte 5</td></tr>
</table>

Notes:

1. Page as MSB followed by 8-bit address is equivalent to a 9-Bit address value as defined in DC2732A_CAN.dbf and in the register constants in LTC2949.h.

2. The number of register bytes Dx to be send in a RAWRW CAN message is defined by the CAN message data length – 2, as two bytes are always reserved for WRITE, PAGE, COUNTDOWN and ADDR.

APPENDIX B: CAN BASED EVALUATION

BASIC OPERATION EXAMPLES

Current, Power, BAT (100ms)

Examples shows measurement of Current (I1), Power (P1) and battery voltage (BAT) at LTC2949's update rate of 100ms.

```
***BUSMASTER Ver 3.2.2***
***PROTOCOL CAN***
***NOTE: PLEASE DO NOT EDIT THIS DOCUMENT***
***[START LOGGING SESSION]***
***HEX***
***RELATIVE MODE***
***CHANNEL 1 - PCAN-USB Driver Id 16 - 500000 bps***
***START DATABASE FILES***
***DC2732A_CAN\DC2732A_CAN.dbf***
***END DATABASE FILES***
***<Time><Tx/Rx><Channel><CAN ID><Type><DLC><DataBytes>***
```

- Send CFG2949 with flags MEAS_BAT, MEAS_I1, MEAS_P1, MEAS_ENABLE set and MEAS_PERIOD = 0

```
00:01:36:4389 Tx 1 0x11F s 8 00 01 00 00 45 80 00 00
```

- Measurement results reported every ~0.1 second (see above CAN message ID description)

```
00:00:00:3140 Rx 1 0x110 s 8 00 00 04 00 00 01 FF D7
00:00:00:0968 Rx 1 0x110 s 8 00 00 00 00 00 01 FF D7
00:00:00:0969 Rx 1 0x110 s 8 FF FF FE 00 00 01 FF D7
00:00:00:0977 Rx 1 0x110 s 8 FF FF FE 00 00 01 FF D7
...
```

- Stop measurement

```
00:00:00:0137 Tx 1 0x11F s 8 00 01 00 00 45 00 00 00
***END DATE AND TIME 17:4:2018 11:46:33:344***
***[STOP LOGGING SESSION]***
```

I1, P1, BAT (100ms), I2fast, BATFast (1ms to 2ms)

Examples shows measurement of Current (I1), Power (P1) and battery voltage (BAT) at LTC2949's update rate of 100ms plus fast current and battery voltage measurement every 1ms to 2ms (nominal 0.8ms but here limited by host controller speed)

```
***BUSMASTER Ver 3.2.2***
...
***<Time><Tx/Rx><Channel><CAN ID><Type><DLC><DataBytes>***
```

- Start of measurement: MEAS_BAT, MEAS_I1, MEAS_P1, CH2FAST, FCONT, P2ASV, MEAS_ENABLE, MEAS_PERIOD = 0, FAST_MEAS_PERIOD = 1

```
00:00:12:8961 Tx 1 0x11F s 8 8A 01 00 00 45 80 00 01
```

APPENDIX B: CAN BASED EVALUATION

- Fast measurement results (I2, BAT, others: don't care!)

```
00:00:00:1989 Rx 1 0x11C s 8 00 C0 01 00 D7 FF 04 C0
00:00:00:0013 Rx 1 0x11C s 8 00 C0 02 00 D7 FF 04 C0
00:00:00:0014 Rx 1 0x11C s 8 00 C0 01 00 D7 FF 04 C0
00:00:00:0018 Rx 1 0x11C s 8 00 C0 01 00 D7 FF 04 C0
00:00:00:0013 Rx 1 0x11C s 8 00 C0 01 00 D8 FF 04 C0
00:00:00:0014 Rx 1 0x11C s 8 00 C0 FF FF EE FF 04 C0
```

- Slow measurement results (I1, P1, BAT)

```
00:00:00:0019 Rx 1 0x110 s 8 FF FF FE 00 00 01 FF EB
```

- Fast measurement results (I2, BAT, others: don't care!)

```
00:00:00:0010 Rx 1 0x11C s 8 00 C0 FF FF EE FF 04 C0
00:00:00:0014 Rx 1 0x11C s 8 00 C0 00 00 EE FF 04 C0
...
```

- Slow measurement results (I1, P1, BAT)

```
00:00:00:0019 Rx 1 0x110 s 8 FF FF FD 00 00 01 FF F2
```

- Fast measurement results (I2, BAT, others: don't care!)

```
00:00:00:0010 Rx 1 0x11C s 8 00 C0 FF FF F2 FF 04 C0
00:00:00:0014 Rx 1 0x11C s 8 00 C0 00 00 F2 FF 04 C0
...
```

- Fast measurement results (I2, BAT, others: don't care!)

```
00:00:00:0018 Rx 1 0x11C s 8 00 C0 FF FF EB FF 04 C0
00:00:00:0014 Rx 1 0x11C s 8 00 C0 00 00 EB FF 04 C0
00:00:00:0013 Rx 1 0x11C s 8 00 C0 00 00 EB FF 04 C0
```

- Stop measurement

```
00:00:00:0007 Tx 1 0x11F s 8 8A 01 00 00 45 00 00 01
```

- Last fast measurement results before stop of measurement

```
00:00:00:0011 Rx 1 0x11C s 8 00 C0 00 00 EB FF 04 C0
***END DATE AND TIME 17:4:2018 11:51:46:257***
***[STOP LOGGING SESSION]***
```

I1, P1, BAT (100ms), I2fast, BATFast (average 10ms)

Examples shows measurement of Current (I1), Power (P1) and battery voltage (BAT) at LTC2949's update rate of 100ms plus fast current and battery voltage measurements averaged over 10ms. Every 10ms roughly 13 samples (10ms/0.8ms = 12.5) each are read from FIFOI1 and FIFOBAT and reported as averaged values.

```
***BUSMASTER Ver 3.2.2***
...
***<Time><Tx/Rx><Channel><CAN ID><Type><DLC><DataBytes>***
```

Rev. 0

APPENDIX B: CAN BASED EVALUATION

- Start of measurement: MEAS_BAT, MEAS_I1, MEAS_P1, CH2FAST, FCONT, P2ASV, MEAS_ENABLE, MEAS_PERIOD = 0, FAST_MEAS_PERIOD = 0

```
00:03:58:2797 Tx 1 0x11F s 8 8A 01 00 00 45 80 00 00
```

- Slow measurement results (I1, P1, BAT)

```
00:00:00:2915 Rx 1 0x110 s 8 00 00 02 00 00 01 FF D7
00:00:00:0967 Rx 1 0x110 s 8 FF FF FF 00 00 00 FF DF
```

…

- Enable fast measurement results read-out from FIFO (max. 128 samples each cycle, cycle period = 10ms)

```
00:00:00:0403 Tx 1 0x11D s 3 00 80 0A
```

- Report of FIFOI2 raw values (only done once initially to show report of currently available FIFO RAW values. Afterwards only the average of the FIFO samples will be reported)

```
00:00:00:0148 Rx 1 0x119 s 8 00 00 FF FF FF FF FF FF
00:00:00:0003 Rx 1 0x119 s 8 FF FF FF FF 00 00 FF FF
```

…

- Average of FIFOI2 and number of samples used for average (here 128)

```
00:00:00:0004 Rx 1 0x121 s 6 00 00 00 80 80 00
```

- Report of FIFOBAT raw values (only done once initially to show report of currently available FIFO RAW values. Afterwards only the average of the FIFO samples will be reported)

```
00:00:00:0136 Rx 1 0x11A s 8 E2 FF E2 FF E2 FF E2 FF
00:00:00:0005 Rx 1 0x11A s 8 E2 FF E2 FF E2 FF E2 FF
```

…

- Average of FIFOBAT and number of samples used for average (here 128)

```
00:00:00:0004 Rx 1 0x122 s 6 00 00 00 80 80 00
```

- Slow measurement results (I1, P1, BAT)

```
00:00:00:0022 Rx 1 0x110 s 8 FF FF FD 00 00 00 FF E1
```

- Average of FIFOI2 and number of samples used for average (here 13)

```
00:00:00:0112 Rx 1 0x121 s 6 00 08 00 00 0D 00
00:00:00:0028 Rx 1 0x122 s 6 00 00 00 80 10 00
00:00:00:0077 Rx 1 0x121 s 6 B1 07 00 00 0D 00
00:00:00:0025 Rx 1 0x122 s 6 00 00 00 80 0E 00
00:00:00:0069 Rx 1 0x121 s 6 9D 04 00 00 0D 00
```

…

- Stop measurement

```
00:00:00:0047 Tx 1 0x11F s 8 8A 01 00 00 45 00 00 00
```

APPENDIX B: CAN BASED EVALUATION

- Last fast measurement results before stop of measurement

```
00:00:00:0030 Rx 1 0x121 s 6 00 00 00 80 0D 00
00:00:00:0025 Rx 1 0x122 s 6 00 00 00 80 0E 00
***END DATE AND TIME 17:4:2018 11:56:19:410***
***[STOP LOGGING SESSION]***
```

ADDITIONAL OPERATION EXAMPLES

Usage of RAWRW

- Send CFG2949 with flags CH2FAST, AUXFAST, P2ASV, MEAS_I1, MEAS_ENABLE set and MEAS_PERIOD set to 10

```
00:00:27:6178 Tx 1 0x11F s 8 86 01 73 74 01 80 0A 01
```

- See Measurement results of Current1 (I1) every second (here: 3LSBs)

```
00:00:00:9990 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:9979 Rx 1 0x110 s 3 00 00 03
```

- Send FASTRQ at any time to make a fast single shot conversion

```
00:00:00:6957 Tx 1 0x11D s 0
```

- See FSSHT as the response

```
00:00:00:0033 Rx 1 0x11C s 8 00 80 00 00 65 FF 3C 02
```

- Send CFG2949 with flag MEAS_ENABLE cleared to stop measurement

```
00:00:00:8610 Tx 1 0x11F s 8 86 01 73 74 01 00 0A 01
```

- Send RAWRW with COUNTDOWN = 0, WRITE = 0, ADDR = 0, D0 = 16, CAN-Message-Data-Length = 3 to read C1, E1, TB1 (see LTC2949's data sheet for LSB values. DC2732A_CAN is configured to use the internal clock)

```
00:01:58:4164 Tx 1 0x117 s 3 00 00 10
00:00:00:0385 Rx 1 0x117 s 6 04 00 00 00 00 03
00:00:00:0004 Rx 1 0x117 s 8 02 00 F4 0F 00 00 00 00
00:00:00:0004 Rx 1 0x117 s 8 00 00 4F C2 00 01 58 7B
```

- Send three RAWRW messages to clear C1. E1 and TB1: (for all messages WRITE = 1, ADDR = 0)

MESSAGE	COUNTDOWN	CAN DATA LENGTH	DX SIGNALS
1	2	6	D0 – D1 All Zero
2	1	8	D0 – D5 All Zero
3	0	8	D0 – D5 All Zero

```
00:02:00:0434 Tx 1 0x117 s 6 84 00 00 00 00 00
00:00:01:8720 Tx 1 0x117 s 8 82 00 00 00 00 00 00 00
00:00:01:5680 Tx 1 0x117 s 8 80 00 00 00 00 00 00 00
00:00:00:0493 Rx 1 0x117 s 4 00 00 10 00
```

APPENDIX B: CAN BASED EVALUATION

- Send RAWRW with COUNTDOWN = 0, WRITE = 0, ADDR = 0, D0 = 16, CAN-Message-Data-Length = 3 to read C1, E1, TB1 again and check all data is now zero.

```
00:00:09:9925 Tx 1 0x117 s 3 00 00 10
00:00:00:0279 Rx 1 0x117 s 6 04 00 00 00 00 00
00:00:00:0004 Rx 1 0x117 s 8 02 00 00 00 00 00 00 00
00:00:00:0004 Rx 1 0x117 s 8 00 00 00 00 00 00 00 00
```

- See below for the full log file of this example recorded with BUSMASTER

```
***BUSMASTER Ver 3.2.0***
***PROTOCOL CAN***
***NOTE: PLEASE DO NOT EDIT THIS DOCUMENT***
***[START LOGGING SESSION]***
***START DATE AND TIME 8:5:2017 15:21:16:468***
***HEX***
***RELATIVE MODE***
***START CHANNEL BAUD RATE***
***CHANNEL 1 - PCAN-USB Driver Id 16 - 500000 bps***
***END CHANNEL BAUD RATE***
***START DATABASE FILES***
***… \LTSketchbook\DC\DC2732A_CAN\DC2732A_CAN.dbf***
***END DATABASE FILES***
***<Time><Tx/Rx><Channel><CAN ID><Type><DLC><DataBytes>***
00:00:27:6178 Tx 1 0x11F s 8 86 01 73 74 01 80 0A 01
00:00:01:2543 Rx 1 0x110 s 3 00 00 03
00:00:00:9990 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:9979 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:9979 Rx 1 0x110 s 3 00 00 03
00:00:00:9990 Rx 1 0x110 s 3 00 00 03
00:00:00:9979 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:9980 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:9979 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:9980 Rx 1 0x110 s 3 00 00 03
```

APPENDIX B: CAN BASED EVALUATION

```
00:00:00:9988 Rx 1 0x110 s 3 00 00 03
00:00:00:9980 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:6957 Tx 1 0x11D s 0
00:00:00:0033 Rx 1 0x11C s 8 00 80 00 00 65 FF 3C 02
00:00:00:2989 Rx 1 0x110 s 3 00 00 03
00:00:00:9990 Rx 1 0x110 s 3 00 00 03
00:00:00:9979 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 02
00:00:00:9979 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:9990 Rx 1 0x110 s 3 00 00 03
00:00:00:9979 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 02
00:00:00:9980 Rx 1 0x110 s 3 00 00 04
00:00:00:9988 Rx 1 0x110 s 3 00 00 03
00:00:00:9980 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:9979 Rx 1 0x110 s 3 00 00 03
00:00:00:9989 Rx 1 0x110 s 3 00 00 03
00:00:00:8610 Tx 1 0x11F s 8 86 01 73 74 01 00 0A 01
00:01:58:4164 Tx 1 0x117 s 3 00 00 10
00:00:00:0385 Rx 1 0x117 s 6 04 00 00 00 00 03
00:00:00:0004 Rx 1 0x117 s 8 02 00 F4 0F 00 00 00 00
00:00:00:0004 Rx 1 0x117 s 8 00 00 4F C2 00 01 58 7B
00:02:00:0434 Tx 1 0x117 s 6 84 00 00 00 00 00
00:00:01:8720 Tx 1 0x117 s 8 82 00 00 00 00 00 00 00
00:00:01:5680 Tx 1 0x117 s 8 80 00 00 00 00 00 00 00
00:00:00:0493 Rx 1 0x117 s 4 00 00 10 00
00:00:09:9925 Tx 1 0x117 s 3 00 00 10
00:00:00:0279 Rx 1 0x117 s 6 04 00 00 00 00 00
00:00:00:0004 Rx 1 0x117 s 8 02 00 00 00 00 00 00 00
00:00:00:0004 Rx 1 0x117 s 8 00 00 00 00 00 00 00 00
***END DATE AND TIME 8:5:2017 15:26:27:342***
***[STOP LOGGING SESSION]***
```

APPENDIX B: CAN BASED EVALUATION

FAST MEASUREMENT

- Enable Measurement:

```
00:01:07:2305 Tx 1 0x11F s 8 9E 01 05 05 F5 87 0A 00
```

- Slow channel measurement results 0

```
00:00:01:0022 Rx 1 0x110 s 8 FF FF FE 00 00 01 FF EB
00:00:00:0005 Rx 1 0x111 s 8 80 00 00 80 00 00 00 7B
00:00:00:0010 Rx 1 0x112 s 8 80 00 0A 75 1F 06 09 76
00:00:00:0003 Rx 1 0x113 s 2 00 71
```

- Slow channel measurement results 1

```
00:00:00:9980 Rx 1 0x110 s 8 FF FF FE 00 00 00 FF EB
00:00:00:0005 Rx 1 0x111 s 8 80 00 00 80 00 00 00 7B
00:00:00:0010 Rx 1 0x112 s 8 80 00 0A 75 1F 06 09 76
00:00:00:0004 Rx 1 0x113 s 2 00 71
```

- Slow channel measurement results 2

```
00:00:00:9979 Rx 1 0x110 s 8 FF FF FE 00 00 01 FF EB
00:00:00:0005 Rx 1 0x111 s 8 80 00 00 80 00 00 00 7B
00:00:00:0010 Rx 1 0x112 s 8 80 00 0A 75 1F 06 09 76
00:00:00:0004 Rx 1 0x113 s 2 00 71
o .... (repeated every second)
```

- Enable report of fast channel (I2, BAT, AUX every 50ms)

```
00:00:00:7683 Tx 1 0x11D s 3 00 50 32
```

- Fast channel FIFO samples I2 report:

```
00:00:00:0128 Rx 1 0x119 s 8 00 00 FF FF FF FF FF FF
00:00:00:0004 Rx 1 0x119 s 8 FF FF FF FF FF FF 00 00
00:00:00:0004 Rx 1 0x119 s 8 00 00 00 00 FF FF 00 00
00:00:00:0004 Rx 1 0x119 s 8 00 00 00 00 00 00 00 00
00:00:00:0004 Rx 1 0x119 s 8 01 00 01 00 01 00 01 00
00:00:00:0005 Rx 1 0x119 s 8 01 00 01 00 00 00 00 00
00:00:00:0004 Rx 1 0x119 s 8 01 00 00 00 00 00 01 00
00:00:00:0004 Rx 1 0x119 s 8 02 00 01 00 01 00 02 00
00:00:00:0004 Rx 1 0x119 s 8 01 00 01 00 01 00 02 00
00:00:00:0004 Rx 1 0x119 s 8 02 00 02 00 02 00 02 00
00:00:00:0004 Rx 1 0x119 s 8 02 00 02 00 02 00 02 00
00:00:00:0004 Rx 1 0x119 s 8 02 00 03 00 02 00 02 00
00:00:00:0004 Rx 1 0x119 s 8 02 00 02 00 02 00 02 00
00:00:00:0004 Rx 1 0x119 s 8 02 00 01 00 01 00 02 00
00:00:00:0004 Rx 1 0x119 s 8 01 00 01 00 02 00 02 00
00:00:00:0004 Rx 1 0x119 s 8 01 00 02 00 01 00 01 00
```

APPENDIX B: CAN BASED EVALUATION

```
00:00:00:0004 Rx 1 0x119 s 8 01 00 00 00 01 00 00 00
00:00:00:0004 Rx 1 0x119 s 8 01 00 00 00 00 00 00 00
00:00:00:0004 Rx 1 0x119 s 8 00 00 00 00 00 00 FF FF
00:00:00:0004 Rx 1 0x119 s 8 FF FF 00 00 00 00 FF FF
```

- Fast channel average measurement results I2:

```
00:00:00:0003 Rx 1 0x121 s 6 19 03 00 00 50 00
```

- Fast channel FIFO samples BAT report:

```
00:00:00:0115 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EF FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EF FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EF FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EF FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EF FF EE FF EE FF
00:00:00:0004 Rx 1 0x11A s 8 EE FF EE FF EE FF EE FF
```

- Fast channel average measurement results BAT:

```
00:00:00:0004 Rx 1 0x122 s 6 00 00 00 80 50 00
```

- Fast channel FIFO samples AUX report:

```
00:00:00:0114 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 79 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
```

APPENDIX B: CAN BASED EVALUATION

```
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
00:00:00:0004 Rx 1 0x11B s 8 78 0A 78 0A 78 0A 78 0A
```

- Fast channel average measurement results AUX:

```
00:00:00:0004 Rx 1 0x123 s 6 0C E0 29 00 50 00
```

(FIFO samples are reported only once, afterwards the average of FIFO samples is reported at the given cycle time)

- Fast channel average measurement results 0 (0x121: I2, 0x122: BAT, 0x123: AUX)

```
00:00:00:0596 Rx 1 0x121 s 6 41 00 00 00 3F 00
00:00:00:0111 Rx 1 0x122 s 6 00 00 00 80 4B 00
00:00:00:0116 Rx 1 0x123 s 6 CC E1 29 00 50 00
```

- Fast channel average measurement results 1 (0x121: I2, 0x122: BAT, 0x123: AUX)

```
00:00:00:0277 Rx 1 0x121 s 6 30 06 00 00 40 00
00:00:00:0095 Rx 1 0x122 s 6 00 00 00 80 40 00
00:00:00:0105 Rx 1 0x123 s 6 24 E2 29 00 47 00
```

- Fast channel average measurement results 2 (0x121: I2, 0x122: BAT, 0x123: AUX)

```
00:00:00:0300 Rx 1 0x121 s 6 00 00 00 80 3F 00
00:00:00:0095 Rx 1 0x122 s 6 00 00 00 80 40 00
00:00:00:0096 Rx 1 0x123 s 6 20 E3 29 00 40 00
```

- Slow channel measurement results.

```
00:00:00:0019 Rx 1 0x110 s 8 00 00 03 00 00 01 FF EB
00:00:00:0005 Rx 1 0x111 s 8 80 00 00 80 00 00 00 7B
00:00:00:0011 Rx 1 0x112 s 8 80 00 0A 75 1F 06 09 76
00:00:00:0003 Rx 1 0x113 s 2 00 71
```

- Fast channel average measurement results 3 (0x121: I2, 0x122: BAT, 0x123: AUX)

```
00:00:00:0273 Rx 1 0x121 s 6 A0 04 00 00 40 00
00:00:00:0096 Rx 1 0x122 s 6 00 00 00 80 40 00
00:00:00:0096 Rx 1 0x123 s 6 10 E3 29 00 40 00
...
```

- Enable report of fast channel (I2, BAT, AUX every 1ms)

```
00:00:00:0065 Tx 1 0x11D s 3 00 50 01
...
```

APPENDIX B: CAN BASED EVALUATION

- Fast channel last sample report (every 1ms to 2ms):

```
00:00:00:0024 Rx 1 0x11C s 8 00 00 02 00 F0 FF 7A 0A
00:00:00:0016 Rx 1 0x11C s 8 00 00 02 00 F0 FF 7A 0A
00:00:00:0018 Rx 1 0x11C s 8 00 00 02 00 F0 FF 7A 0A
00:00:00:0014 Rx 1 0x11C s 8 00 00 02 00 F0 FF 7A 0A
00:00:00:0010 Rx 1 0x11C s 8 00 00 03 00 EF FF 7A 0A
00:00:00:0017 Rx 1 0x11C s 8 00 00 03 00 F0 FF 7A 0A
...
```

- Stop measurement

```
00:00:00:0006 Tx 1 0x11F s 8 9E 01 05 05 F5 07 0A 00
```

SERIAL MONITOR VIA LINDUINO'S USB PORT

The serial monitor is not necessary for operation but can give additional debug information and allows to change the CAN bus baud rate.

The serial terminal baud rate must be set to 1000000 and the newline termination must be set to "Both NL & CR". See Arduino IDE's Serial Monitor screenshot as an example.

Note: In case Arduino DUE is used, the max. supported baud rate is 250000.

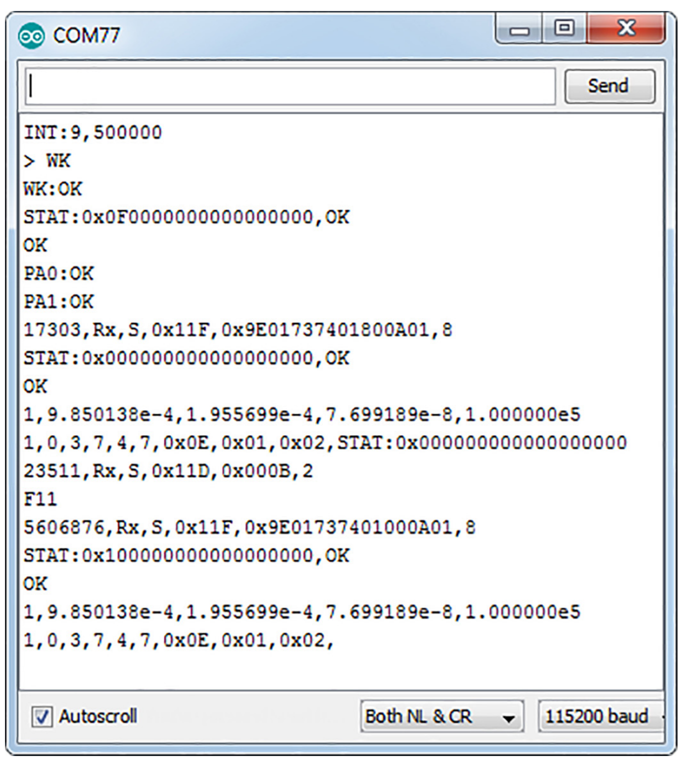


Figure 53. Arduino IDE's Serial Monitor

APPENDIX B: CAN BASED EVALUATION

Table 7. Serial Monitor Command Overview

CMD	DESCRIPTION
RST	Reset the LTC2949

INT<CS, BR>	Assign chip select pin to the CAN controller (must be 9 for the DC2617A) and set the CAN baud rate.
	For example:
	`> INT9,500000`
	`INT:9,500000`

T<ms>	Configure the CAN message send period in milliseconds. (Equivalent to MEAS_PERIOD•100 which is configured via CAN message CFG2949)

WK	Wakeup LTC2949, read and clear status, alerts and faults registers. This is done automatically with CAN message CFG2949.
	Typical output:
	`> WK`
	`WK:OK`
	`STAT:0x0F0000000000000000,OK`
	`OK`
	`PA0:OK`
	`PA1:OK`
	0x….: Content of status and alert registers in the order of their addresses as hexadecimal string (first two characters are the content of the STATUS register; last two characters are the content of the FAULTS register)

DB<Val>	Val = 0: Disable debug output (default) Val = 1: Enable debug output (default) Enabled debug output of LTC2949's SPI communication. Attention: If enabled this will drastically impact operation timing due to processing time needed to generate the debug output and send it via the serial monitor. Certain functions may not work as expected.

Note:

After a reset of the Linduino the DC2732A_CAN will send the following message via the serial COM port: INT:9,500000 which is the default setting for the CAN controller (CS-Pin = 9, 500kBit/s)

APPENDIX B: CAN BASED EVALUATION

CONVERT .DBF TO .DBC FILE

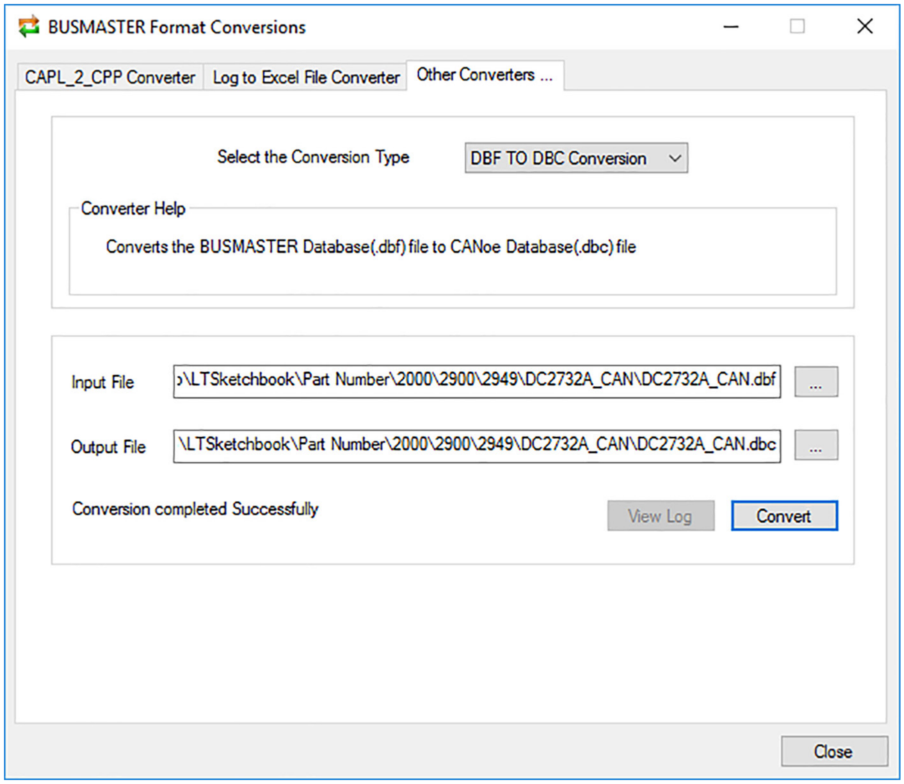
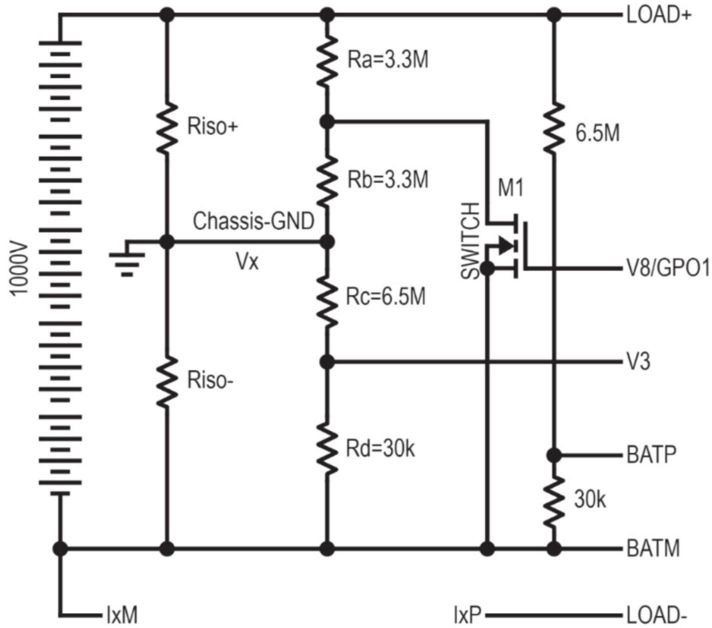


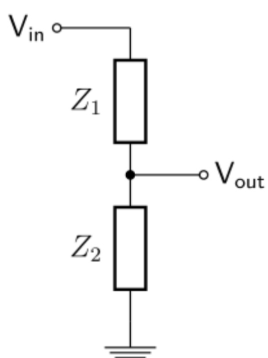
Figure 54. Busmaster → Tools → Format Converter → Other Converters

APPENDIX C: ISOLATION MEASUREMENT WITH LTC2949

The equations to calculate the isolation fault resistors Riso+ and Riso- depending on two chassis-GND meas-urements VxI and VxII and two battery stack voltage measurements VBATI and VBATII for the two cases 'switch open' (I) and 'switch closed' (II) are derived as following.



RESISTIVE DIVIDER EQUATION



$$V_{out} = V_{in}\frac{Z_2}{Z_2+Z_1} = V_{in}\frac{Y_1}{Y_1+Y_2}; Z = \frac{1}{Y}$$

DEFINITIONS

$$R_a = R_p * (1-p);\ R_b = R_p * p;\ \ R_a + R_b = R_p;\ Y_p = \frac{1}{R_p}$$

$$R_c = R_n * (1-n);\ R_d = R_n * n;\ \ R_c + R_d = R_n;\ Y_n = \frac{1}{R_n}$$

$$V_{V1} = n * V_x;\ \ V_x = \frac{V_{V1}}{n}$$

APPENDIX C: ISOLATION MEASUREMENT WITH LTC2949

CASE I: SWITCH M1 OPEN

$$V_{xI} = V_{BATI}\frac{Y_{iso+} + Y_p}{Y_{iso+} + Y_p + Y_{iso-} + Y_n}$$

$$V_{xI}\left(Y_{iso+} + Y_p + Y_{iso-} + Y_n\right) = V_{BATI} * \left(Y_{iso+} + Y_p\right)$$

$$V_{xI} * Y_{iso+} + V_{xI} * Y_p + V_{xI} * Y_{iso-} + V_{xI} * Y_n = V_{BATI} * Y_{iso+} + V_{BATI} * Y_p$$

$$(V_{xI} - V_{BATI}) * Y_{iso+} + V_{xI} * Y_{iso-} + V_{xI} * \left(Y_p + Y_n\right) - V_{BATI} * Y_p = 0$$

CASE II: SWITCH M1 CLOSED

$$V_{xII} = V_{BATII}\frac{Y_{iso+}}{Y_{iso+} + Y_{iso-} + \frac{Y_p}{p} + Y_n}$$

$$V_{xII} * \left(Y_{iso+} + Y_{iso-} + \frac{Y_p}{p} + Y_n\right) = V_{BATII} * Y_{iso+}$$

$$V_{xII} * Y_{iso+} + V_{xII} * Y_{iso-} + V_{xII} * \frac{Y_p}{p} + V_{xII} * Y_n = V_{BATII} * Y_{iso+}$$

$$(V_{xII} - V_{BATII}) * Y_{iso+} + V_{xII} * Y_{iso-} + V_{xII} * \left(\frac{Y_p}{p} + Y_n\right) = 0$$

COMBINE EQUATIONS OF BOTH CASES

$$\text{'case I'} * V_{xII}:\quad (V_{xI} - V_{BATI}) * V_{xII} * Y_{iso+} + V_{xI} * V_{xII} * Y_{iso-} + V_{xI} * V_{xII} * \left(Y_p + Y_n\right) - V_{BATI} * Y_p * V_{xII} = 0$$

$$\text{'case II'} * V_{xI}:\quad (V_{xII} - V_{BATII}) * V_{xI} * Y_{iso+} + V_{xII} * V_{xI} * Y_{iso-} + V_{xII} * V_{xI} * \left(\frac{Y_p}{p} + Y_n\right) = 0$$

$$\text{'case I'} * V_{xII} - \text{'case II'} * V_{xI}:$$

$$(V_{xI} - V_{BATI}) * V_{xII} * Y_{iso+} + V_{xI} * V_{xII} * \left(Y_p + Y_n\right) - V_{BATI} * Y_p * V_{xII} - (V_{xII} - V_{BATII}) * V_{xI} * Y_{iso+} - V_{xII} * V_{xI} * \left(\frac{Y_p}{p} + Y_n\right) = 0$$

$$(V_{BATII} * V_{xI} - V_{BATI} * V_{xII}) * Y_{iso+} + V_{xI} * V_{xII} * Y_p\left(1 - \frac{1}{p}\right) - V_{BATI} * Y_p * V_{xII} = 0$$

$$Y_{iso+} = \frac{V_{xI} * V_{xII} * Y_p\left(\frac{1}{p} - 1\right) + V_{BATI} * Y_p * V_{xII}}{V_{BATII} * V_{xI} - V_{BATI} * V_{xII}}$$

APPENDIX D: MEASURE HALL SENSOR WITH DC2732A_BASIC

CALCULATE Y_{ISO}– FROM CASE II:

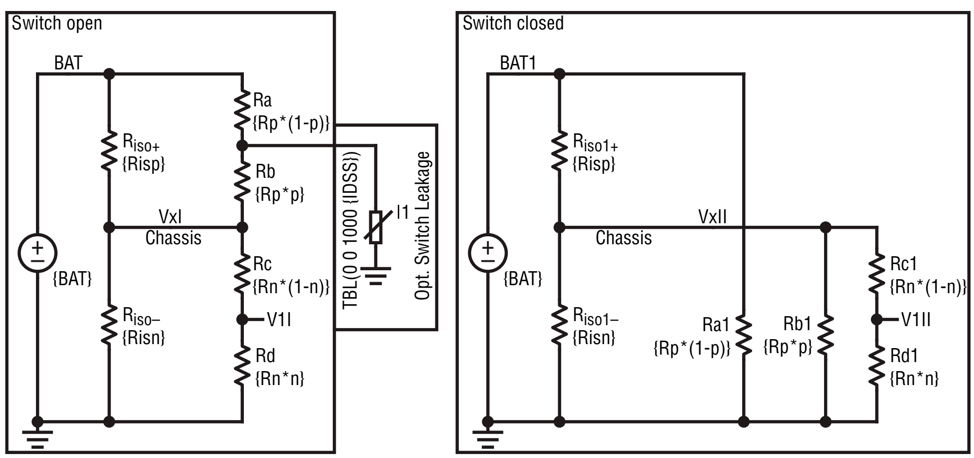
$$(V_{xII} - V_{BATII}) * Y_{iso+} + V_{xII} * Y_{iso-} + V_{xII} * \left(\frac{Y_p}{p} + Y_n\right) = 0$$

$$Y_{iso-} = \frac{-V_{xII} * \left(\frac{Y_p}{p} + Y_n\right) - (V_{xII} - V_{BATII}) * Y_{iso+}}{V_{xII}}$$

$$Y_{iso-} = \left(\frac{V_{BATII}}{V_{xII}} - 1\right) * Y_{iso+} - Y_n - \frac{Y_p}{p}$$

Note: The final two equations used to calculate R_{iso+} and R_{iso-} are put inside rectangles ($R_{iso\pm} = 1 / Y_{iso\pm}$)

SIMULATION WITH SWITCH LEAKAGE ERROR



Note: Typical Rn = Rp, p = 0.5. Due to tolerances the real values might be different and could be calibrated as shown in the example values above.
IDSS: MOSFET Zero gate voltage drain current (VGS = 0), e.g. STD4NK100Z 50µA@125dC

Table: Error of Isolation Resistance Measurement (IDSS = 10µA, Rd = 38k)

Error of R_{iso-} Measurement

	2.50E+05	1.00E+06	5.00E+06	1.00E+07
2.50E+05	–2%	–2%	–3%	–3%
1.00E+06	–2%	–2%	–2%	–2%
5.00E+06	–2%	–2%	–2%	–2%
1.00E+07	–2%	–2%	–2%	–2%

Error of R_{iso+} Measurement

	2.50E+05	1.00E+06	5.00E+06	1.00E+07
2.50E+05	–3%	–2%	–2%	–2%
1.00E+06	–3%	–3%	–3%	–3%
5.00E+06	–6%	–5%	–5%	–5%
1.00E+07	–10%	–9%	–8%	–8%

Table: Error of Isolation Resistance Measurement (IDSS = 50µA, Rd = 10k)

Error of R_{iso-} Measurement

$R_{iso\pm}$	2.50E+05	1.00E+06	5.00E+06	1.00E+07
2.50E+05	–10%	–10%	–10%	–10%
1.00E+06	–12%	–11%	–11%	–11%
5.00E+06	–23%	–21%	–21%	–21%
1.00E+07	–33%	–31%	–31%	–31%

Error of R_{iso+} Measurement

$R_{iso\pm}$	2.50E+05	1.00E+06	5.00E+06	1.00E+07
2.50E+05	–10%	–9%	–10%	–11%
1.00E+06	–9%	–8%	–8%	–9%
5.00E+06	–8%	–8%	–8%	–8%
1.00E+07	–8%	–8%	–8%	–8%

For more details please contact Analog Devices.

APPENDIX D: MEASURE HALL SENSOR WITH DC2732A_BASIC

For easy evaluation without changing the Linduino sketch, the external sensor (e.g., hall sensor) could be connected to the BAT (BATP-BATM) input of the demoboard.

```
tDut,I1,P1,BAT,Tntc,TIC,fI2,fBAT,fifoCnt,fifoI2Avg,fifoBATAvg,OK/ERR
```

Measurements of BAT are reported twice by DC2732A_BASIC. BAT is from the slow channel, f_{BAT} is from the fast channel (average of 128 samples). As also the current is reported from the fast channel as average of 128 samples, it is reasonable to compare f_{BAT} and fI2 for evaluation of LTC2949's current channel and compare it to a hall sensor.

Alternatively, the Sketch can be changed to measure also SLOT2 and configure this slot to measure the hall sensor voltage output e.g., on V3(hall+)-V4(hall-). Within DC2732A_BASIC.ino search for following code and comment "**#undef ADD_SLOT2_MEASUREMENT**" as shown here.

```
// enable measurement of Vx-Vy via SLOT2
#define ADD_SLOT2_MEASUREMENT
 // disable measurement of Vx-Vy via SLOT2
 //#undef ADD_SLOT2_MEASUREMENT
#define SLOT2_MEAS_POS 3
#define SLOT2_MEAS_NEG 4
```

As default now also SLOT2 measurement is reported. See also CSV header:

```
tDut,I1,P1,BAT,Tntc,S2,TIC,fI2,fBAT,fifoCnt,fifoI2Avg,fifoBATAvg,OK/ERR
```

S2 is the report of slot2 in volts.

Example output:

```
tDut,I1,P1,BAT,Tntc,S2,TIC,fI2,fBAT,fifoCnt,fifoI2Avg,fifoBATAvg,OK/ERR
STAT:0x10000000000000000000,OK
STAT:0x10000000000000000000,OK
6499,0.010881,0.013448,1.236,26.2,0.5966,27.6,0.00336,1.235,132,0.008124,1.236,OK
99,0.002108,0.002609,1.234,26.2,0.1725,27.6,0.00014,1.233,126,0.000994,1.234,OK
99,0.001046,0.001296,1.233,26.2,0.0030,27.6,0.00526,1.232,127,0.002169,1.233,OK
99,0.008098,0.009969,1.231,26.2,0.1714,27.6,0.01690,1.230,126,0.010967,1.231,OK
99,0.020582,0.025308,1.229,26.2,0.6131,27.6,0.03084,1.228,129,0.024189,1.229,OK
99,0.033752,0.041447,1.228,26.2,1.1603,27.6,0.04123,1.227,126,0.036727,1.228,OK
99,0.042602,0.052245,1.226,26.2,1.6050,27.6,0.04466,1.225,129,0.043759,1.226,OK
```

APPENDIX E: SYNCHRONOUS MEASUREMENTS WITH CELL MONITORS AND LTC2949

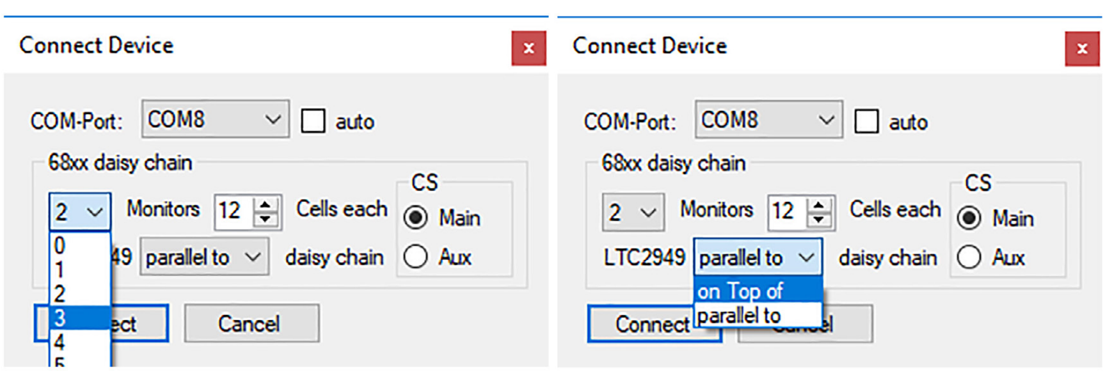
The LTC2949 GUI also supports synchronous measurements of cell voltages via cell monitors LTC68xx together with current, voltage measurements by LTC2949:

1. Click Connect.

a.

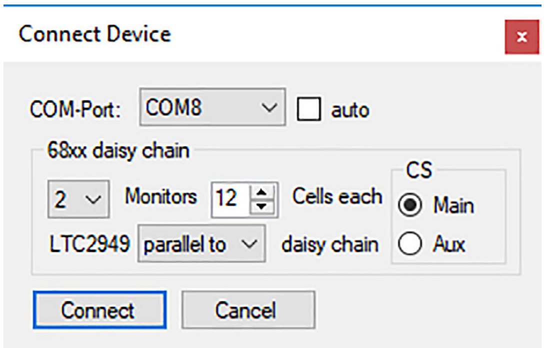
2. Select number of cell monitors, cells per cell monitor, topology of the isoSPI bus (LTC2949 on top of the daisy chain or parallel to the bottom of the daisy chain) and the CS (chip select) line. The latter can be set to Aux to use the AUX channel of the dual LTC6820 isoSPI board (DC2792). In most other cases it will be set to Main.

a.



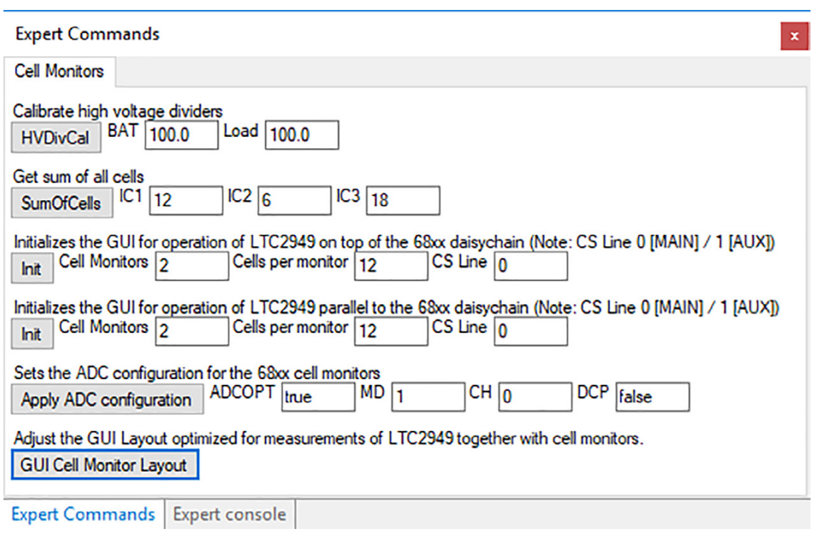
APPENDIX E: SYNCHRONOUS MEASUREMENTS WITH CELL MONITORS AND LTC2949

3. Click Connect.

a. 

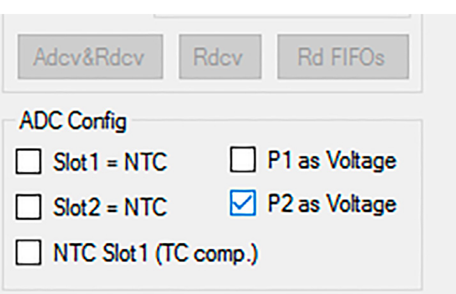
b. **Note:** Keep in mind to disable the termination resistor on DC2732A (JP1) in case LTC2949 is connected in parallel to an isoSPI bus.

4. Adjust GUI Layout optimized for Cell Monitor integration (click on GUI Cell Monitor Layout within Expert Commands window).

a. 

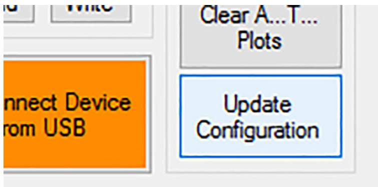
5. Configure fast voltage and current measurements of LTC2949.

a. Enable P2 as Voltage.

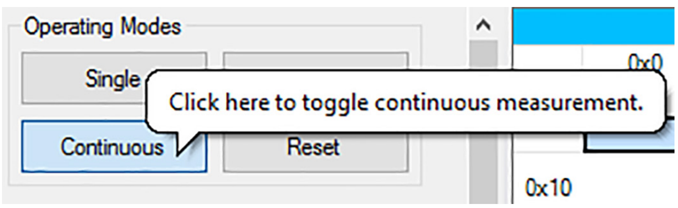
b. 

APPENDIX E: SYNCHRONOUS MEASUREMENTS WITH CELL MONITORS AND LTC2949

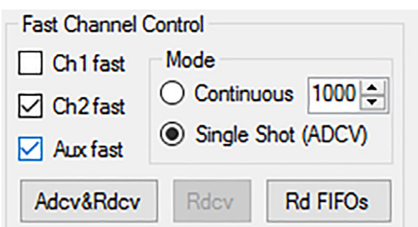
c. Update Configuration.

d. 

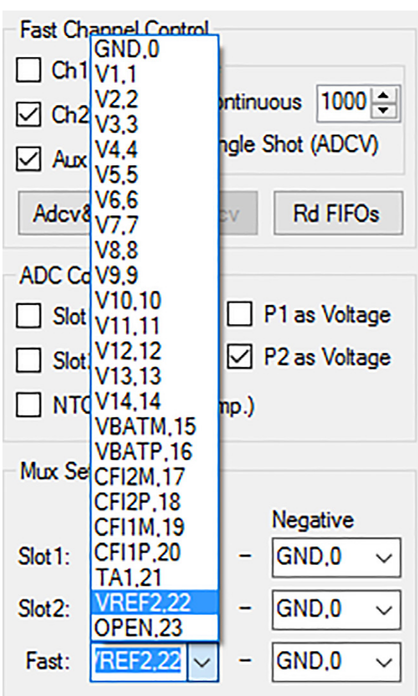
e. Enable Continuous Mode.

f. 

g. Enable Fast Measurements CH2, Aux.

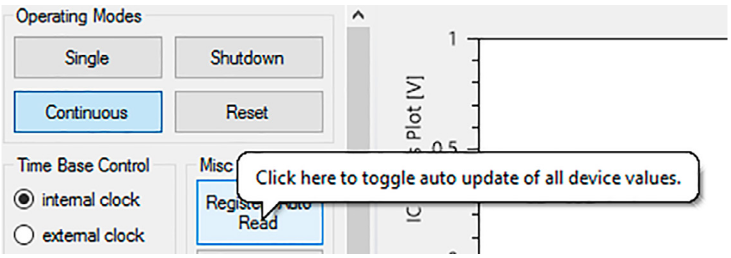
h. 

6. Select Channel for Fast AUX Conversion (e.g., V_{REF2} vs GND).

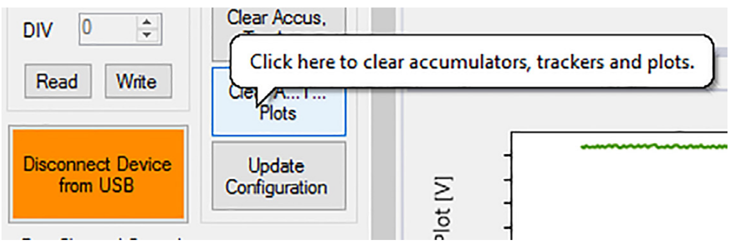
a. 

APPENDIX E: SYNCHRONOUS MEASUREMENTS WITH CELL MONITORS AND LTC2949

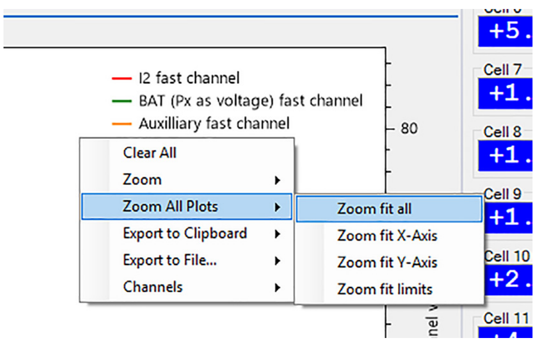
7. Enable "Register Auto Read" to Get Measurement Results.

a. 

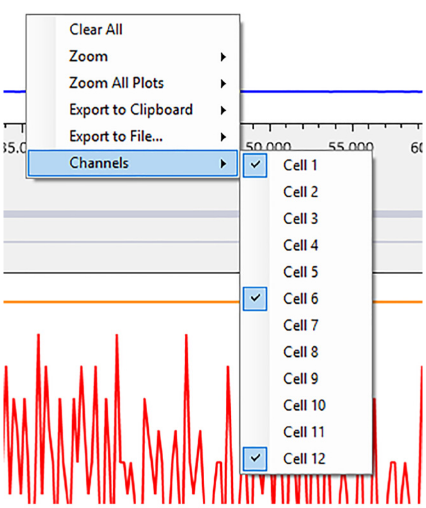
8. Click on "Clear A… T… Plots" to Clear the Plots and Start Over at Time = 0.

a. 

9. Right Click in Any Plot Window and Select "Zoom All Plots" → "Zoom Fit All" to See the Plotted Results.

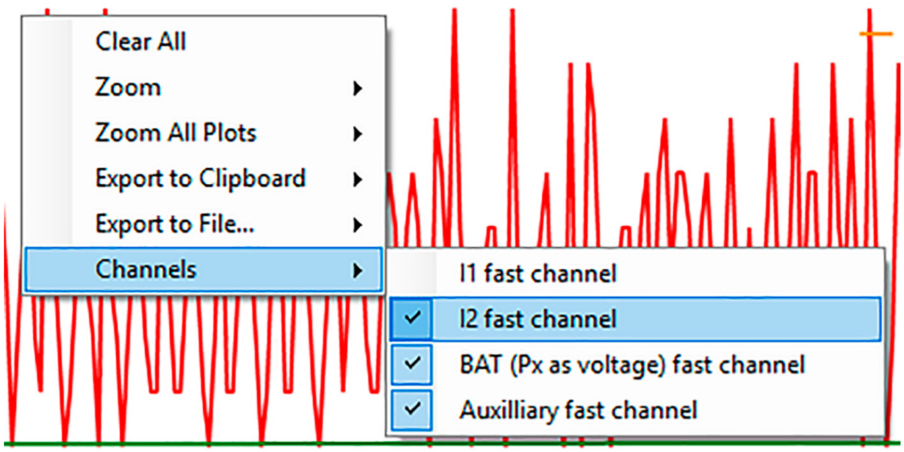
a. 

10. Right Click in Any Plot Window and Select More Channels to View:

a. 

Rev. 0

APPENDIX E: SYNCHRONOUS MEASUREMENTS WITH CELL MONITORS AND LTC2949



b.

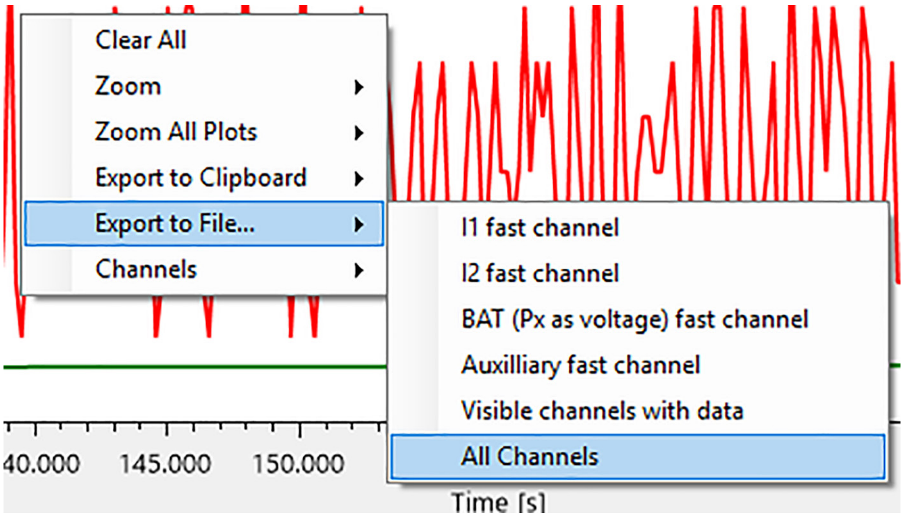
11. Watch the Measurements.

a.

APPENDIX E: SYNCHRONOUS MEASUREMENTS WITH CELL MONITORS AND LTC2949

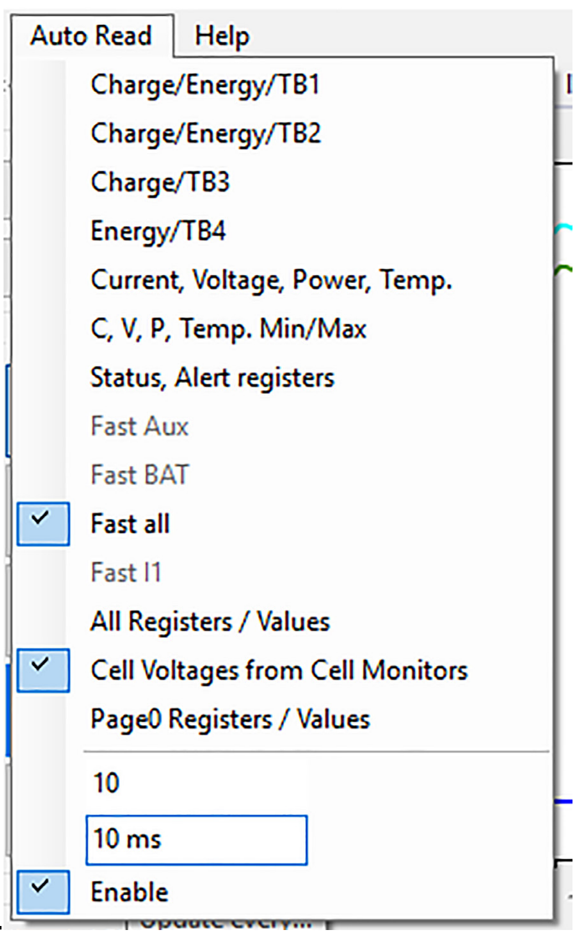
12. Export the Measurements if Required.

a.



13. Change the Measurement Update Rate if Required.

a.



APPENDIX F: GUI TROUBLESHOOTING & LINDUINO PROGRAMMING

The latest LTC2949 GUI version is distributed with Analog Devices QuikEval software.

For QuikEval to be able to detect the DC2732A, the demoboard must be configure to SPI mode (JP3–JP6) and connect to the Linduino via the 14-pin flat ribbon cable.

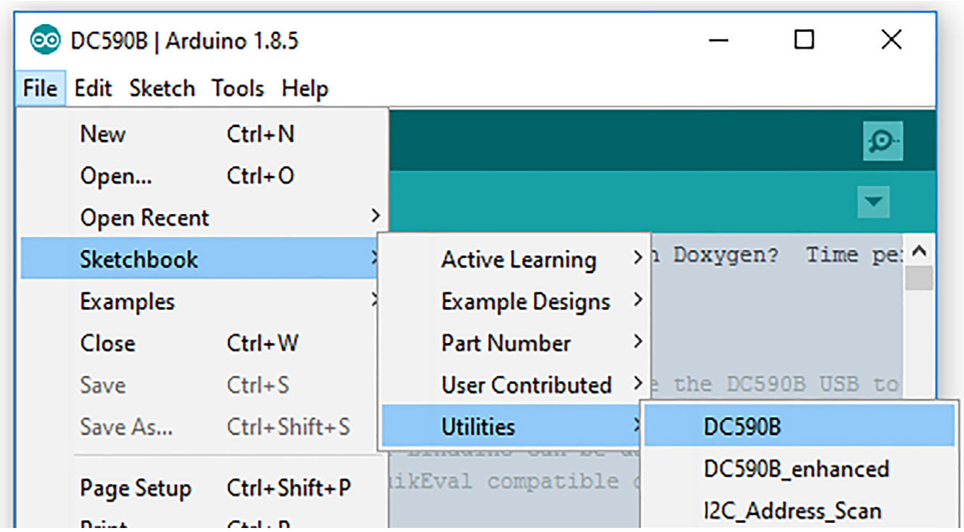
After that plug the Linduino's USB cable to the PC and launch QuikEval. It should detect the board and ask to download and install the LTC2949 GUI software. SPI mode is only mandatory for QuikEval to detect the board. Afterwards, isoSPI mode can be used again with the LTC2949 GUI (but the GUI must be launched manually from the Windows start menu, as QuikEval will not identify LTC2949 demoboard when connected via isoSPI, e.g., with LTC6820 demoboard, to the PC).

Alternatively, the GUI can be downloaded directly from:

LTC2949 GUI Tools & Simulations

Both files must be place in one local folder and then setup2949.exe must be executed (depending on the operating system settings, the context menu command "Run Elevated" must be used).

The LTC2949 GUI expects the Linduino to be loaded with the DC590B Sketch (factory default). If the GUI does never connect, very likely the Linduino is programmed with the wrong Sketch. Please make sure the DC590B sketch is programmed (factory default of the Linduino).



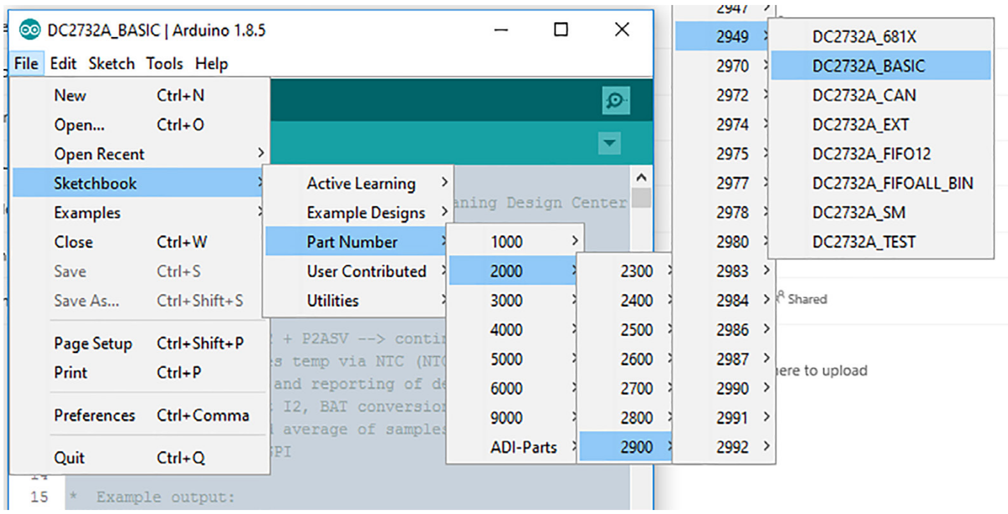
The green LED on the DC2732A (LTC2949 demoboard) indicates, there was/is communication to LTC2949, it's in IDLE and not in sleep mode.

To check, that the hardware setup is fine, another sketch for LTC2949 can be programmed. All the Linduino sketches for LTC2949 can be found in:

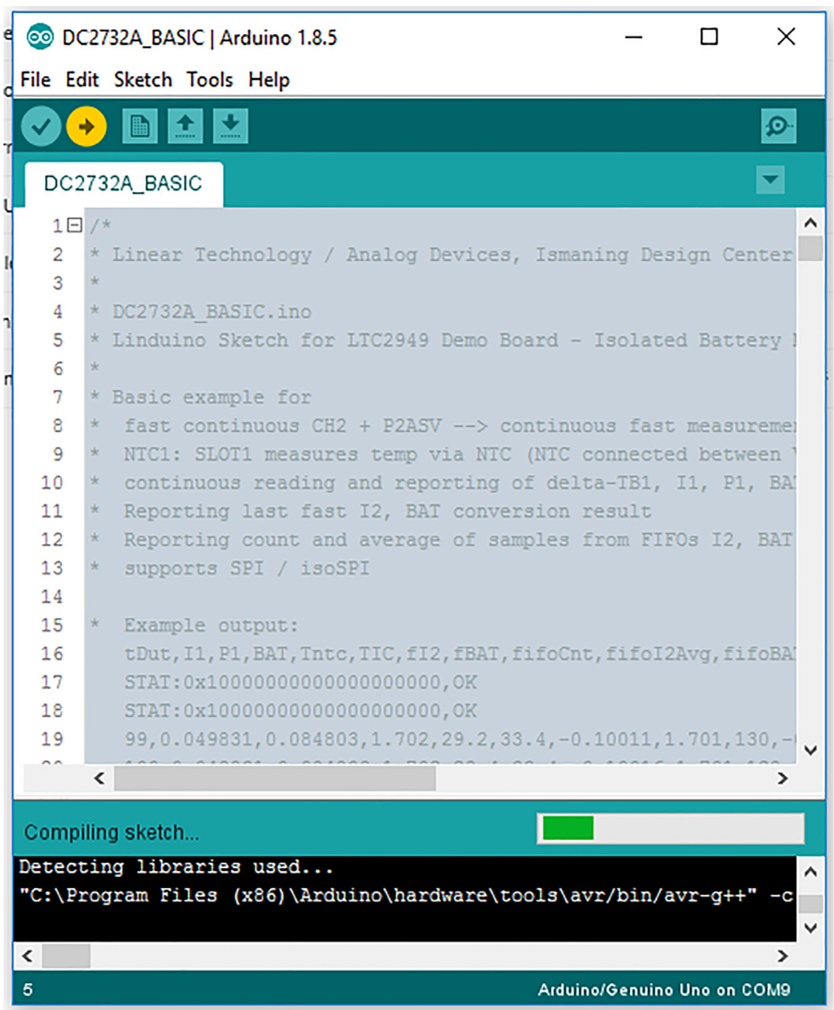
LinduinoSketchbook2949.zip, provided by Analog Devices.

APPENDIX F: GUI TROUBLESHOOTING & LINDUINO PROGRAMMING

To do a basic hardware test, the DC2732A_BASIC sketch can be loaded.



Click on the "right arrow" to compile and upload.



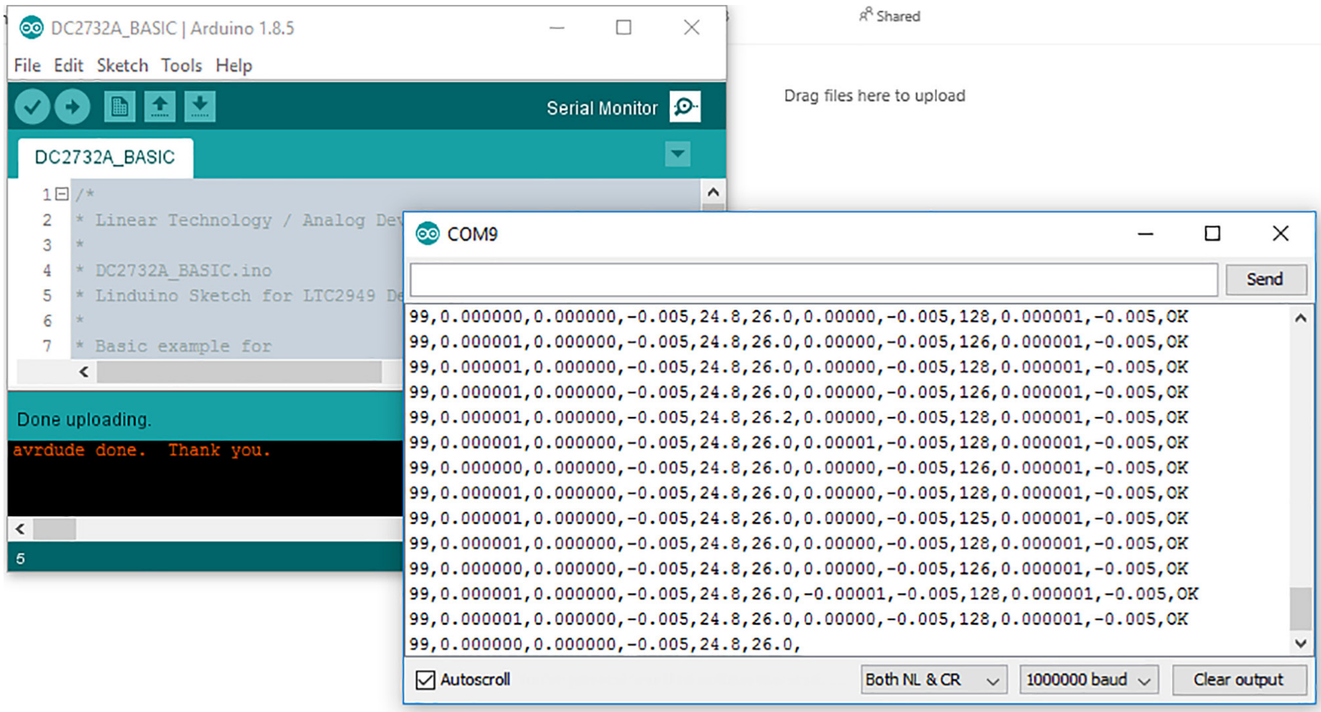
Rev. 0

APPENDIX F: GUI TROUBLESHOOTING & LINDUINO PROGRAMMING

Click on the Serial Monitor Icon, make sure Serial Monitor is set to 1000000 baud.

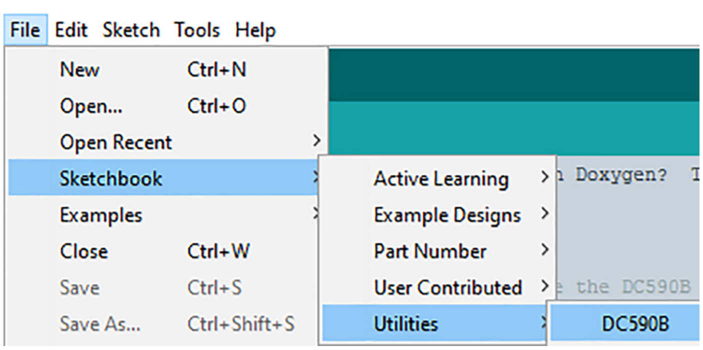
Note: In case Arduino DUE is used, the max. supported baud rate is 250000.

See output:



If the output is like above screenshot, without any message "…ERR…" the hardware works fine. Now the Linduino can be program again with the DC590B sketch to use the LTC2949-GUI.

1. Open DC590B.

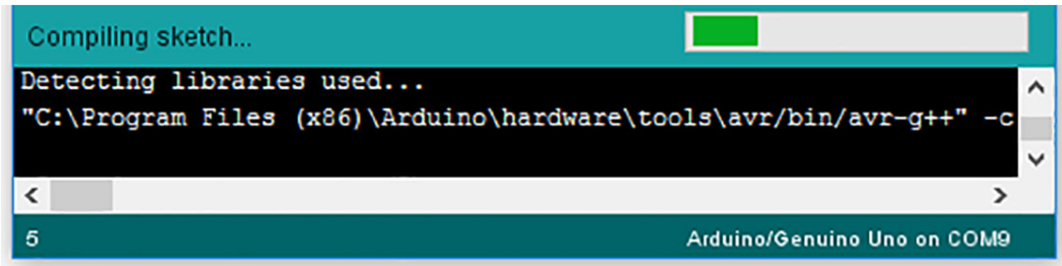
a. 

APPENDIX F: GUI TROUBLESHOOTING & LINDUINO PROGRAMMING

2. Click the Upload button.

a. 

3. Watch the Sketch being compiled….

4. 

a. …and wait till the upload is done.

5. 

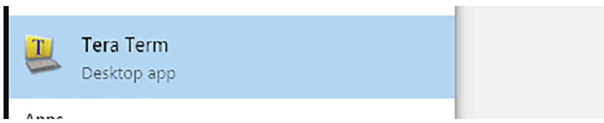
6. Now the Arduino IDE can be closed and the LTC2949 GUI can be opened.

APPENDIX G: LOG MEASUREMENTS WITH TERA TERM

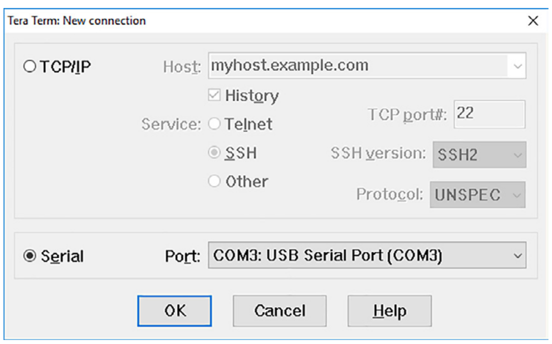
Most of the DC2732A example Sketches, for example, the DC2732A_BASIC Sketch, send the measurement data via the serial monitor to the PC. Any serial terminal software can be used to record this data. For example, the open source tool Tera Term and its log feature can be used as described here.

LOG MEASUREMENT DATA FROM DC2732A_BASIC TO TEXT/.CSV – FILE

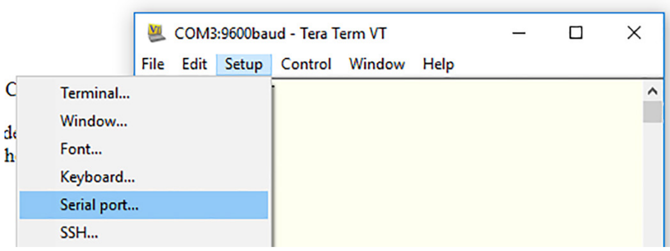
1. Download Tera Term.
2. Start Tera Term.

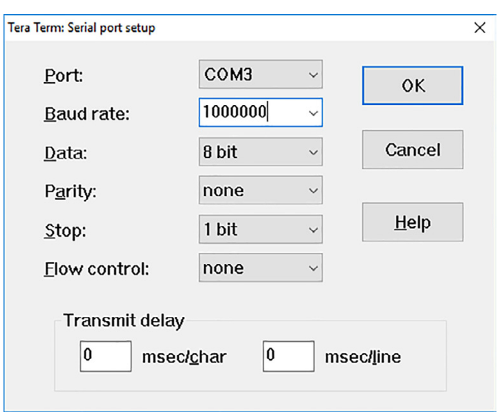
a. 

3. Select right com port (close the serial monitor of the Arduino IDE before!)

a. 

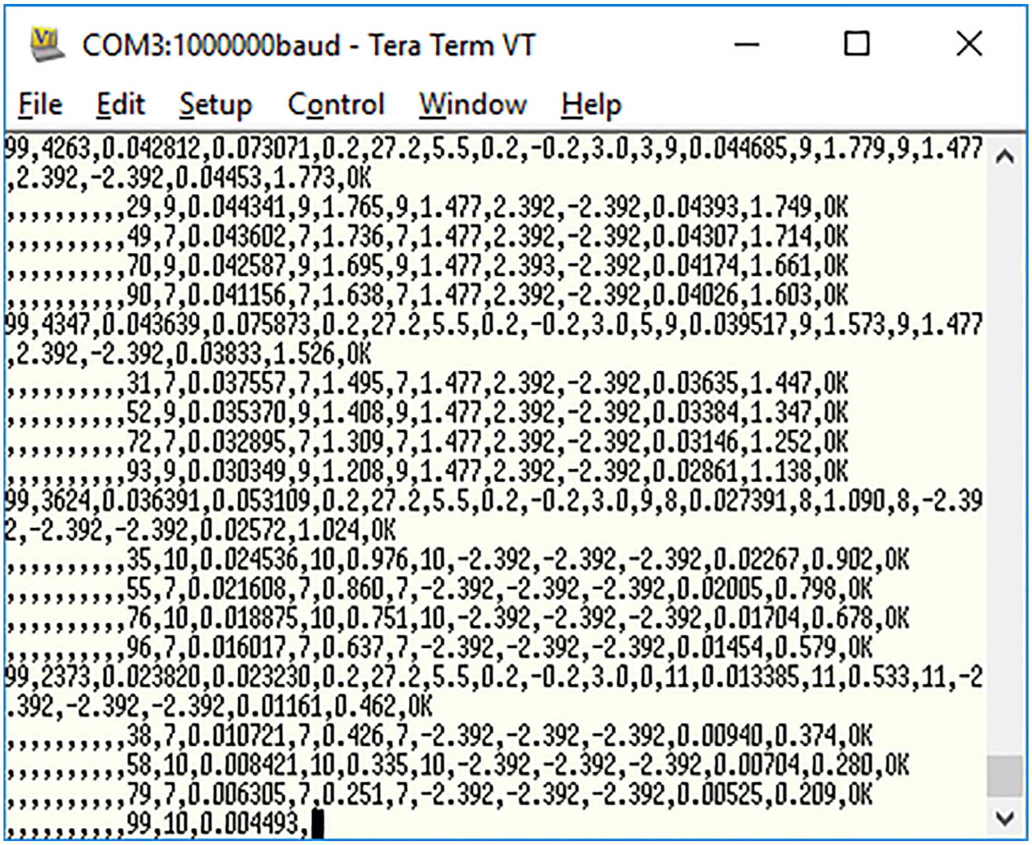
4. Set the right baud rate (check .ino file for baud rate setting).

a. 

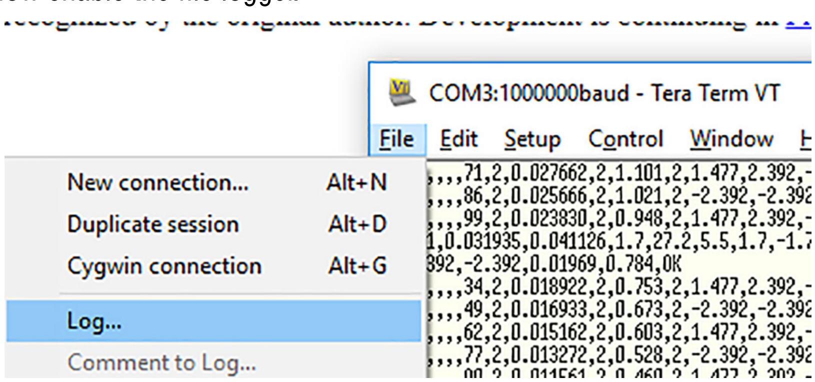
b. 

APPENDIX G: LOG MEASUREMENTS WITH TERA TERM

5. Now you see the output printed to the window.

a. 

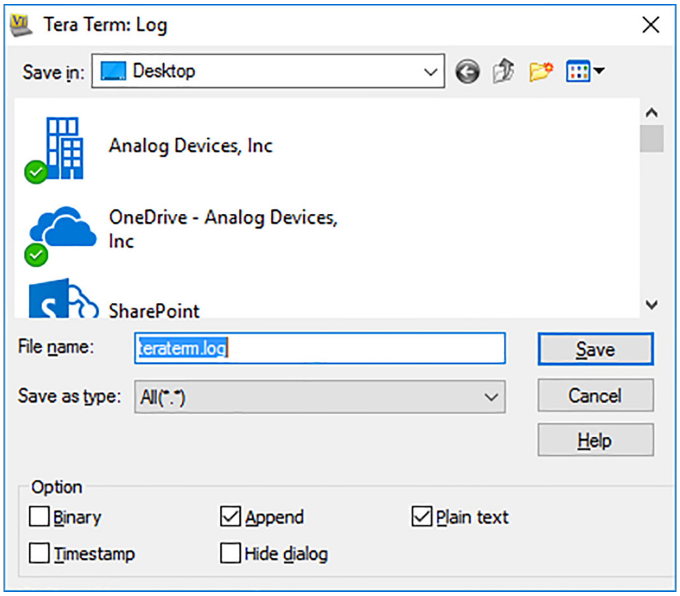
6. Now enable the file logger.

a. 

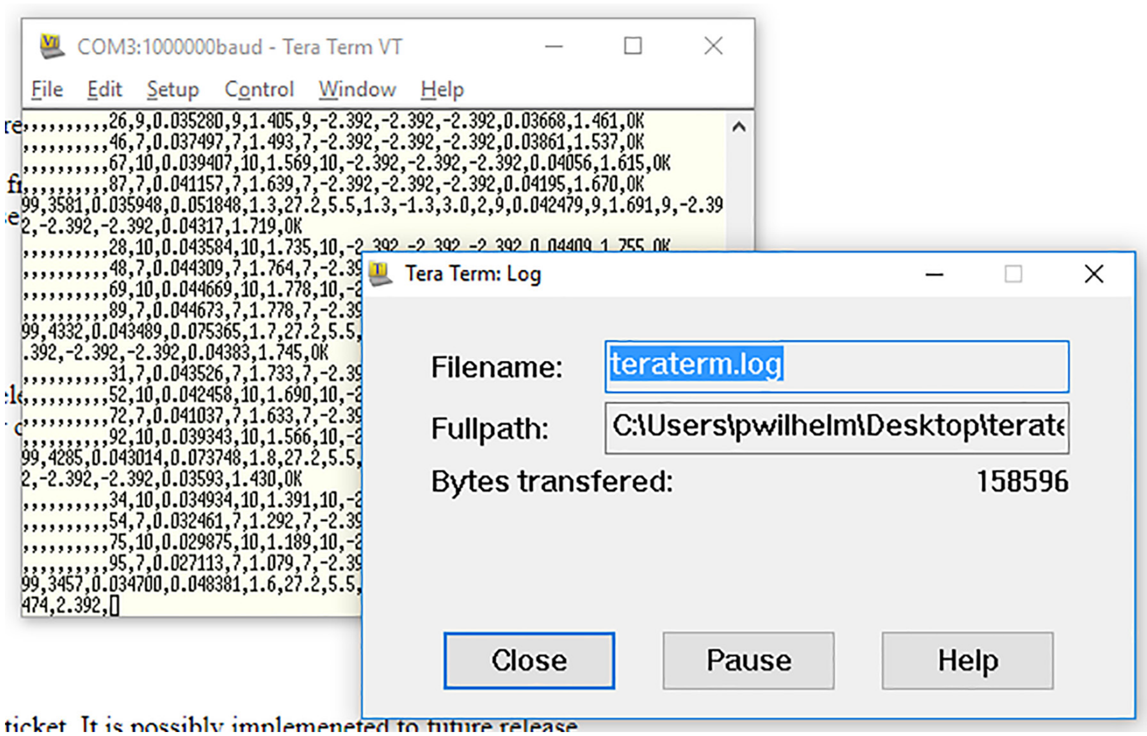
Rev. 0

APPENDIX G: LOG MEASUREMENTS WITH TERA TERM

7. Choose a file name (it is recommended to use the ending .csv, for easy opening the file with excel…)



8. Click save. All data is now stored to a file. To show the status of the log go to File → "Show Log dialog…"



The file could now be opened in a text editor or in excel. If the file ending .csv was used, the file can be easily opened in excel with a double click.

APPENDIX H: LTC2949.CPP/.H BASIC LIBRARY FUNCTIONS

The following gives an overview and description of the low level communication functions available in the LTC2949 C-Code library LTC2949.cpp/.h (within Sketchbook folder: LTSketchbook\libraries\LTC2949).

1. Configure library for communication topology (top of daisy chain or parallel to daisy chain/single device).

```
LTC2949_init_lib(
/*byte cellMonitorCount,   */LTCDEF_CELL_MONITOR_COUNT,
/*boolean ltc2949onTopOfDaisychain, */false,
    /*boolean debugEnable    */false
);
```

2. Set library to default settings of LTC2949 after power-up (library mirrors some of LTC2949's register settings to calculate e.g., charge/energy/time LSB sizes, SLOT LSB which can be temperature or voltage, power ADC settings which can be power of voltage…).

```
LTC2949_init_device_state();
```

3. WRITE single byte.

```
LTC2949_WRITE(LTC2949_REG_WKUPACK, 0x00); // write wake up acknowledge
// is Equal to
byte data = 0;
LTC2949_WRITE(LTC2949_REG_WKUPACK, 1, &data);
```

4. WRITE burst of bytes.

```
void LTC2949_SlotsCfg(byte slot1P, byte slot1N, byte slot2P, byte slot2N)
{
  byte data[4] = { slot1N, slot1P, slot2N, slot2P };
  LTC2949_WRITE(LTC2949_REG_SLOT1MUXN, 4, data);
}
```

5. READ single byte

```
byte data;
byte error = LTC2949_READ(LTC2949_REG_FACTRL, 1, &data);
6. READ burst of bytes
byte LTC2949_GetSlotsCfg(byte * slot1P, byte * slot1N, byte * slot2P, byte * slot2N)
{
  byte data[4];
  byte error = LTC2949_READ(LTC2949_REG_SLOT1MUXN, 4, data);
  *slot1N = data[0];
  *slot1P = data[1];
  *slot2N = data[2];
  *slot2P = data[3];
  return error;
}
```

APPENDIX H: LTC2949.CPP/.H BASIC LIBRARY FUNCTIONS

7. Check EEPROM control and status register, make sure to be in the initial state

```
byte error = LTC2949_EEPROMIsReady()
```

8. Initialize EEPROM

```
byte error = LTC2949_EEPROMCommand(LTC2949_BM_EEPROM_INIT)
```

9. Check if EEPROM was initialized correctly

```
byte error = LTC2949_EEPROMCommand(LTC2949_BM_EEPROM_CHECK)
```

10. Store LTC2949 memory to EEPROM

```
byte error = LTC2949_EEPROMCommand(LTC2949_BM_EEPROM_SAVE)
```

11. Restore LTC2949 memory from EEPROM

```
byte error = LTC2949_EEPROMCommand(LTC2949_BM_EEPROM_RESTORE)
```

12. EEPROM access, higher level functions
13. All-in-one EEPROM read (do all checks and restore from EEPROM)

```
byte error = LTC2949_EEPROMRead()
```

14. All-in-one EEPROM write (Initialize, check and write to EEPROM)

```
byte error = LTC2949_EEPROMWrite()
```

15. Initialize only and check EEPROM (checks if EEPROM is connected and write is possible, but does not write any of LTC2949's configuration)

```
byte error = LTC2949_EEPROMInitCheck()
```

Most functions LTC2949_ typically return an error code. Return value is 0
.in case of no error.

Most functions LTC2949_ typically return an error code. Return value is 0 in case of no error.

ESD Caution

ESD (electrostatic discharge) sensitive device. Charged devices and circuit boards can discharge without detection. Although this product features patented or proprietary protection circuitry, damage may occur on devices subjected to high energy ESD. Therefore, proper ESD precautions should be taken to avoid performance degradation or loss of functionality.

02/20

www.analog.com