Application Note AN-102
InnoSwitch3-Pro/InnoSwitch4-Pro Family

Arduino Code Library Overview and Guide

Introduction

The application of InnoSwitch3-Pro and InnoSwitch4-Pro Family Arduino Library is discussed in this document. This code was designed to be highly portable with different microcontroller platforms. The use of Arduino compatible C++ language will make it easy for users to understand and modify the code according to their needs. This guide will allow the user to get sufficient knowledge on how to operate the devices with a use of a simple microcontroller such as Arduino.

InnoSwitch3-Pro

InnoSwitch3-Pro devices are ideal for AC/DC power supply applications where fine (10 mV, 50 mA) output voltage and current adjustment are necessary. Typical implementations comprise a system microprocessor or dedicated microcontroller with an I2C port that is used to configure, control and supervise the operation of the power sub-system. The uVCC pin provides a bias supply for the microprocessor in stand-alone implementations such as USB PD adapters and chargers.

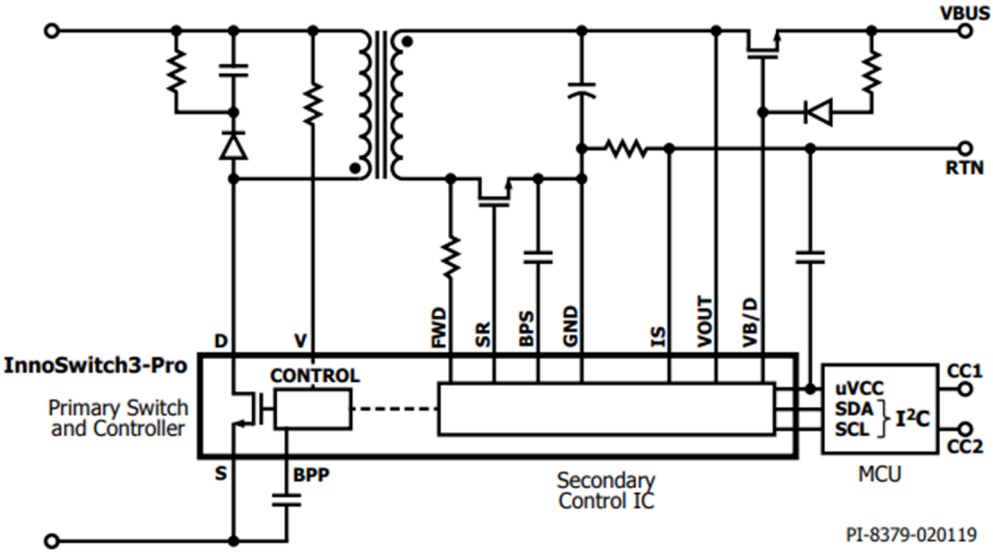


Figure 1. InnoSwitch3-Pro Schematic

InnoSwitch4-Pro

The command and telemetry registers on InnoSwitch4-Pro are updated compared to InnoSwitch3-Pro. These features add flexibility and improve fault response.

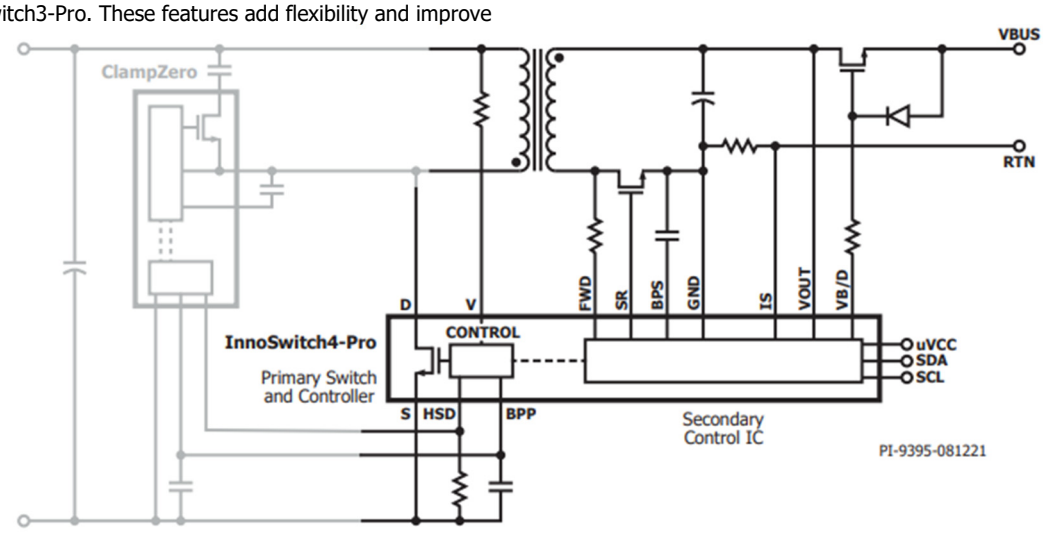


Figure 2. InnoSwitch4-Pro Schematic

System Requirements

Hardware

The InnoSwitch3-Pro and InnoSwitch4-Pro can be controlled using its on board microcontroller or by an external I²C Master through the interface header. This Demo Application does not use the on board microcontroller but an Arduino Uno as an I²C Master and InnoSwitch3-Pro/InnoSwitch4-Pro as slave device. This demonstration will be conducted with the Reference Design RDK-641 for InnoSwitch3-Pro and RDR-961 for InnoSwitch4-Pro.

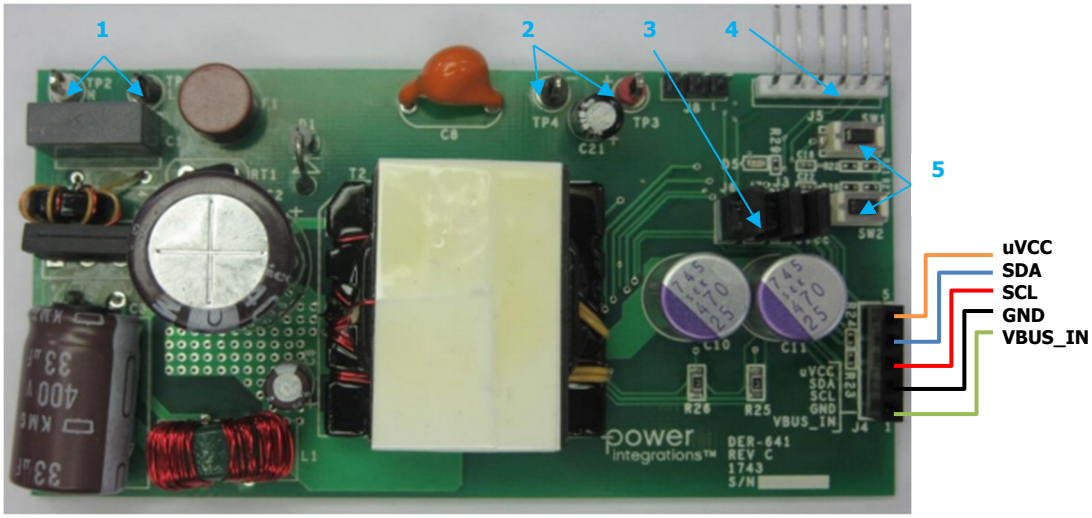


Figure 3. RDK-641

Number	Description	Label
1	AC Input Terminals	TP1, TP2
2	DC Output Terminals	TP3, TP4
3	uVcc and I²C Isolation Jumpers	J3, J6, J7
4	PIC Programming Header	J5
5	Push Buttons	SW1, SW2

Table 1. RDK-641 Part Description

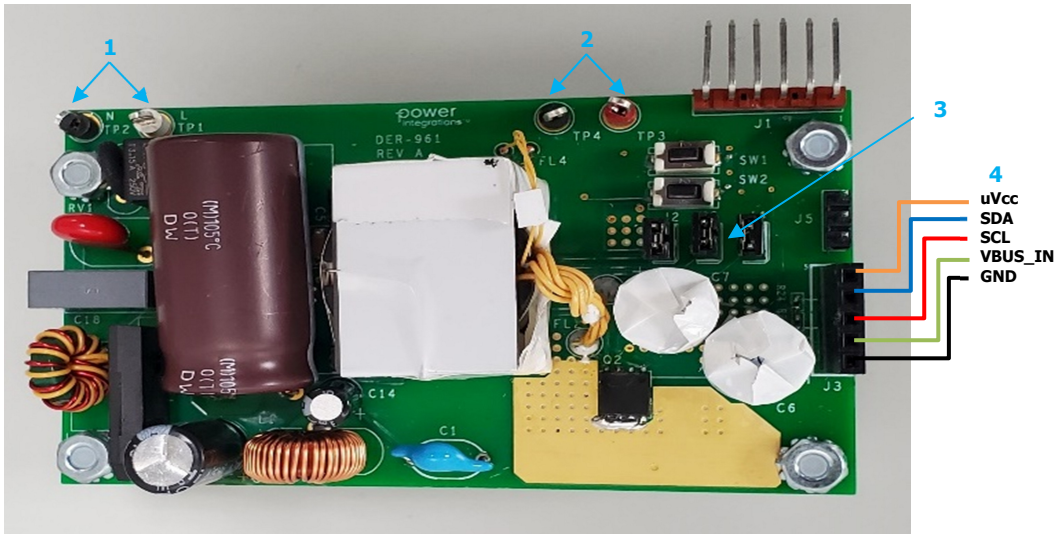


Figure 4. DER-961

Number	Description	Label
1	AC Input Terminals	TP1, TP2
2	DC Output Terminals	TP3, TP4
3	uVcc and I²C Isolation Headers	J2, J4, J6
4	External I²C Interface Header	J3

Table 2. RDR-961 Part Description

Arduino

Arduino IDE version 1.8.16 was used in this document with Arduino Mega 2560 microcontroller board because of its higher flash memory needed for other certain library examples. Arduino Uno board may also be used for InnoSwitch3-Pro and InnoSwitch4-Pro library examples that do not require a lot of memory.

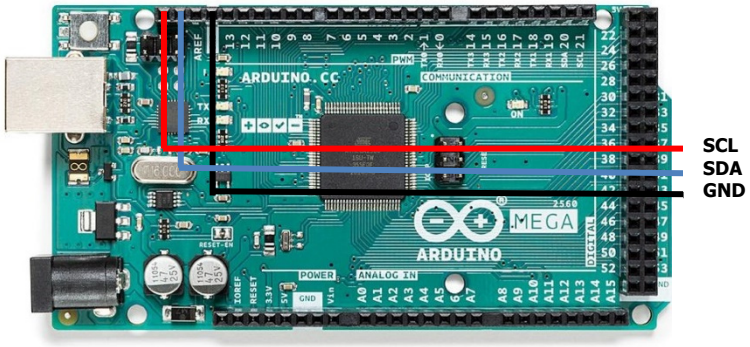


Figure 5. Arduino Mega 2560

Code Library

InnoSwitch3-Pro and InnoSwitch4-Pro Arduino code library contains the drivers and sketch examples of InnoSwitch3-Pro and InnoSwitch4-Pro. The function of these examples ranges from simply initializing the device up to allowing the user direct access to each individual Command and Telemetry registers. The InnoSwitch3-Pro and InnoSwitch4-Pro Arduino library can be downloaded from the link below.

InnoSwitch3-Pro and InnoSwitch4-Pro Family Code Library and API for Arduino | Power Integrations, Inc.
https://www.power.com/design-support/downloads/innoswitch3-pro-code-library-and-api-arduino

Library Installation

Head on to Arduino IDE and go to *Sketch menu > Include Library > Add .ZIP Library.* Select and open the Arduino .zip library. The user can check the if the library has been installed in the *Sketch menu > Include Library* menu. In addition, the library can also be seen in the file path below.

- *C:|Users|username|Documents|Arduino|libraries|*

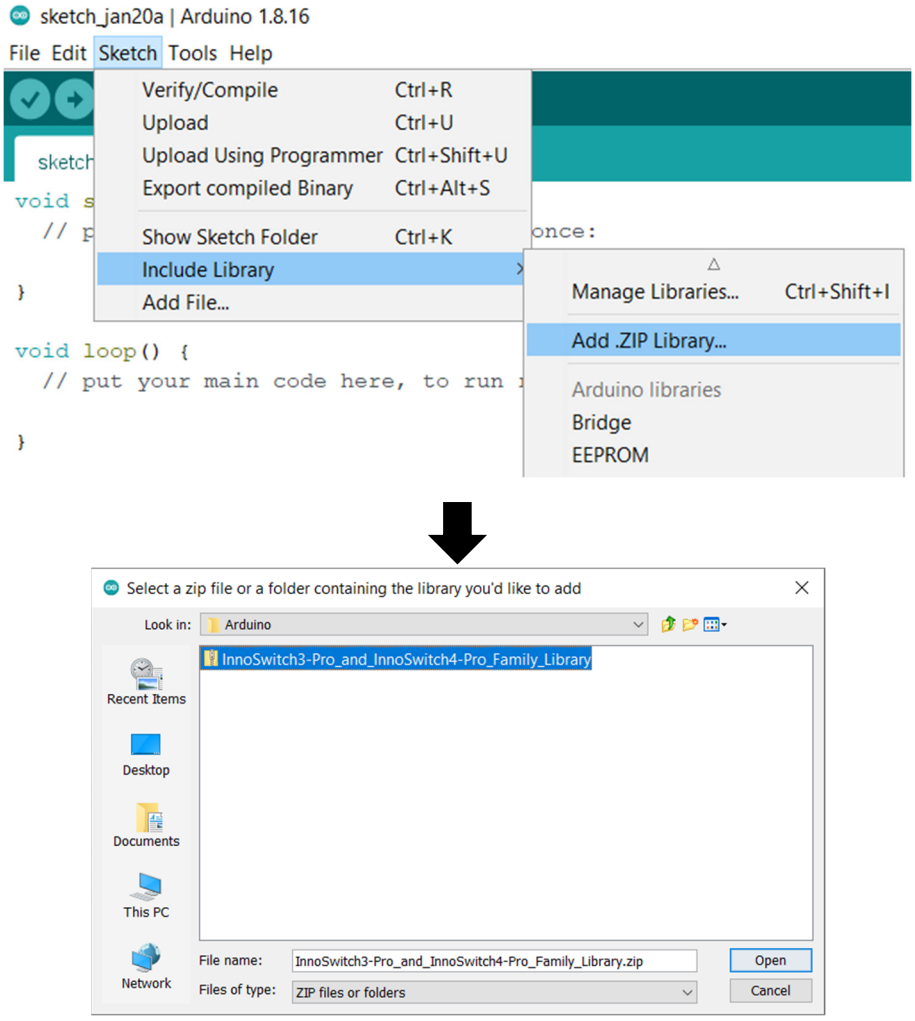


Figure 6. Adding Arduino Library

Removing Other InnoSwitch3-Pro / InnoSwitch4-Pro Libraries

There is a need to remove older versions of the InnoSwitch3-Pro / InnoSwitch4-Pro Libraries since it may cause the Arduino IDE to compile the wrong source files with the new library. Using old source files may cause different issues and it is better to avoid that possibility by having only one version of the Arduino library. To remove other Arduino Libraries, go to the file path found in *File > Preferences* and delete the folder. In the example below, the file path is at *C:|Users|users|Documents|Arduino*. Go to the library and delete the folder of the old library likely named "InnoSwitch3-Pro_Library". This will avoid conflicting header files when compiling.

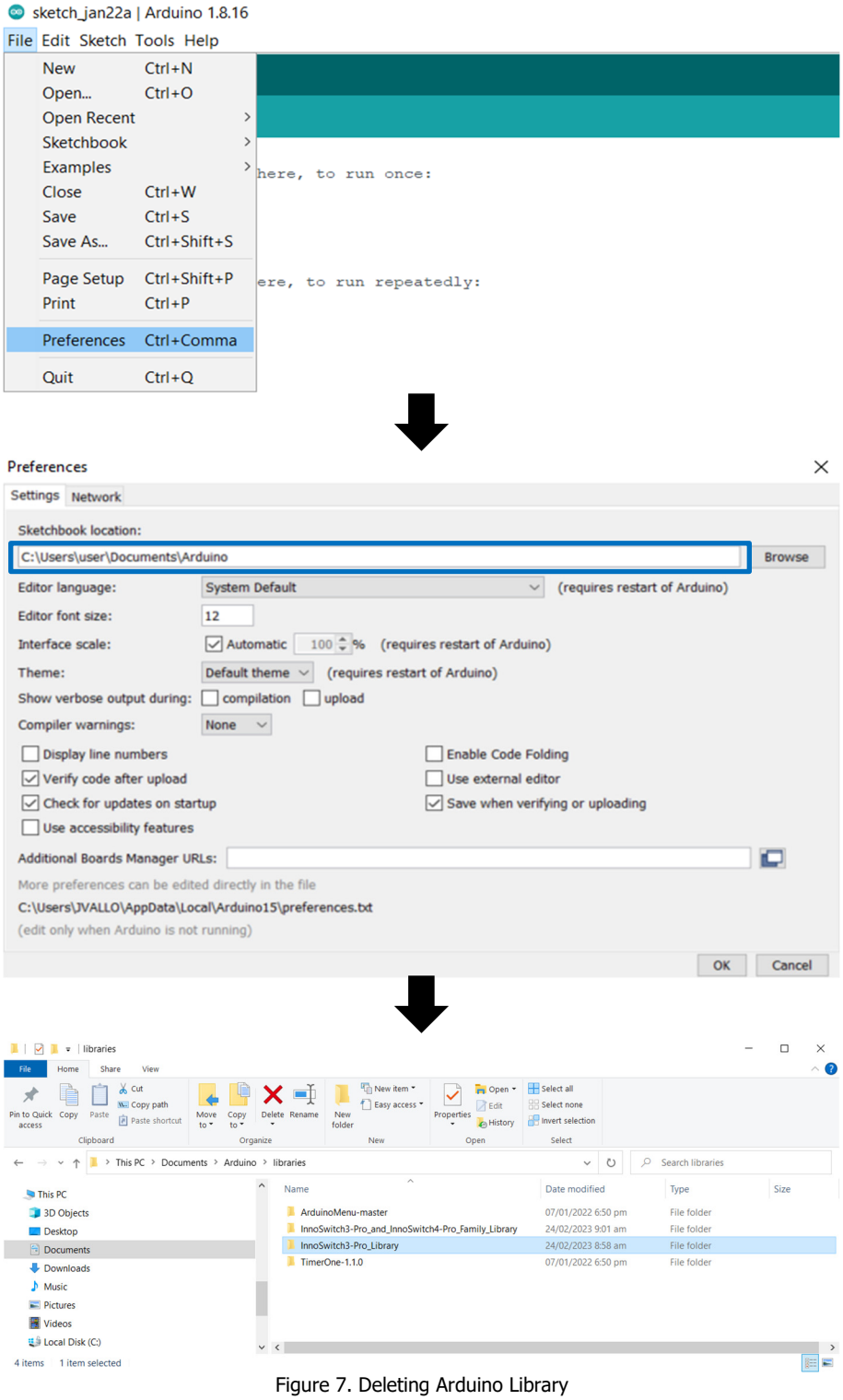
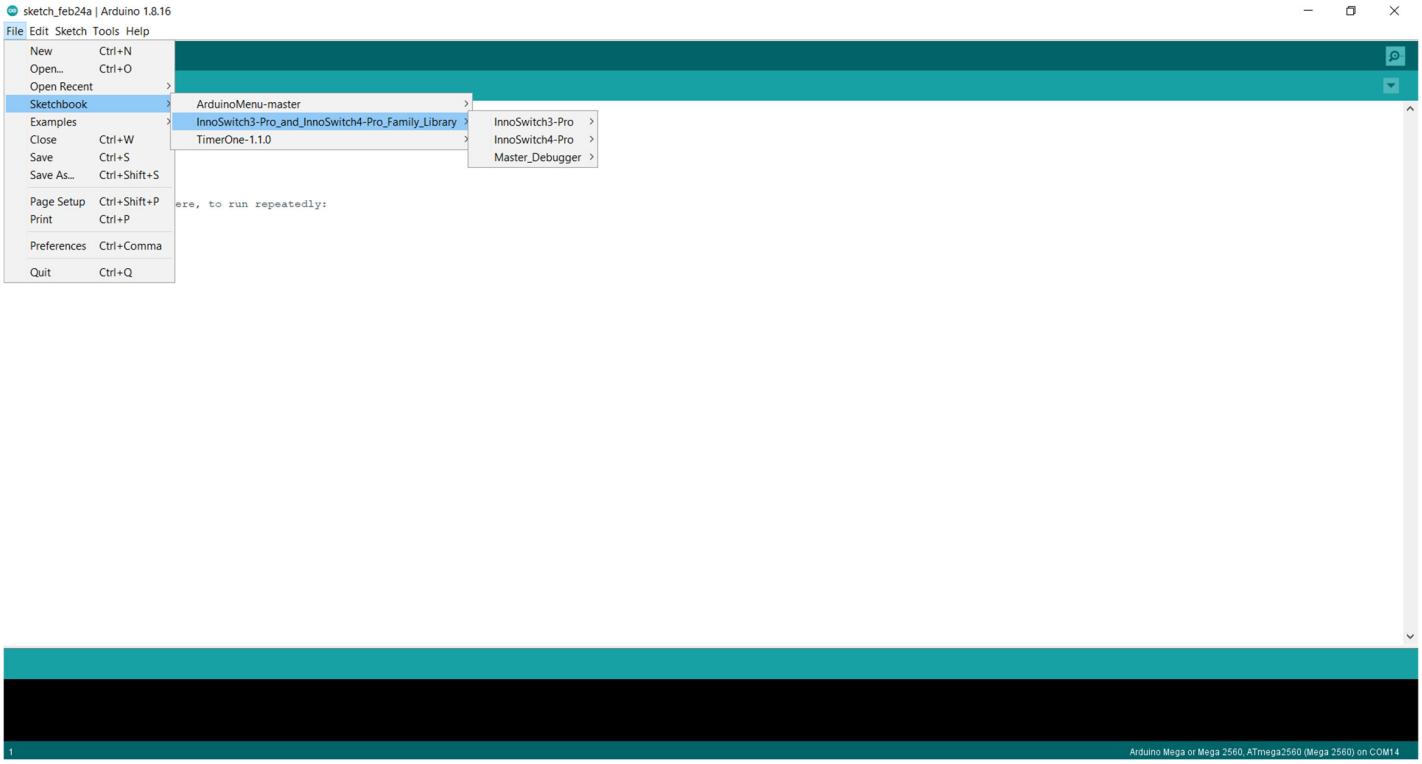


Figure 7. Deleting Arduino Library

Examples

Installing the library will allow the user to use the examples found in *File Sketchbook > InnoSwitch3-Pro and InnoSwitch4-Pro Library.* These sketches provide basic code and can serve as the framework and guide more complex implementations for controlling InnoSwitch3-Pro and InnoSwitch4-Pro devices. Note that InnoSwitch3-Pro sketches do not work with InnoSwitch4-Pro devices and vice versa.



InnoSwitch3-Pro Sketches

- Inno3Pro_APDOs.ino
- Inno3Pro_Basic.ino
- Inno3Pro_Basic_Volts_Amps_OV_UV.ino
- Inno3Pro_PD_Hard_Reset.ino
- Inno3Pro_PDOs.ino
- Inno3Pro_Plotter.ino
- Inno3Pro_Ramp.ino
- Inno3Pro_Random_Volt_Time.ino
- Inno3Pro_Serial.ino
- Inno3Pro_SineWave.ino

InnoSwitch4-Pro Sketches

- Inno4Pro_APDOs.ino
- Inno4Pro_Basic.ino
- Inno4Pro_Basic_Volts_Amps_OV_UV.ino
- Inno4Pro_PD_Hard_Reset.ino
- Inno4Pro_PDOs.ino
- Inno4Pro_Plotter.ino
- Inno4Pro_Ramp.ino
- Inno4Pro_Random_Volt_Time.ino
- Inno4Pro_Serial.ino
- Inno4Pro_SineWave.ino

InnoSwitch3-Pro Examples

Example 1 - Inno3Pro_Basic.ino

Inno3Pro_Basic.ino sketch implements the essential commands to run the InnoSwich3-Pro IC. The adapter will output 5V 3.1A while using this code. There are five commands in this sketch to note:

- Inno3Pro_Initialization(); - Function for initializing the InnoSwitch3-Pro device
- Inno3Pro_Write_VI(); - Function for setting the output voltage and current
- Inno3Pro_Write_Volt_Peak(); - Sets the knee voltage (V_{KP})
- Inno3Pro_Vbus_Switch_Control(); - Controls the BUS switch

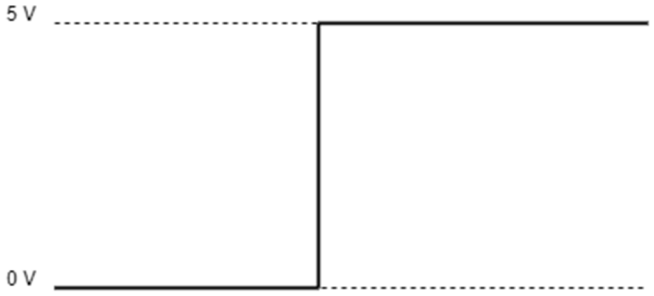


Figure 8. Inno3Pro_Basic.ino Output Voltage Waveform

Example 2 - Inno3Pro_PDOs.ino

This sketch cycles through the multiple output voltages. This mimics the USBPD standard PDO's for 60W adapters.

Main functions used:

- Inno4Pro_Initialization(); - Initializes InnoSwitch4-Pro
- Inno4Pro_Write_Cable_Drop_Comp(); - Sets the cable drop voltage in mV
- Inno4Pro_Write_Volt_Peak() - Sets the knee voltage (V_{KP})
- Inno4Pro_Vbus_Switch_Control() – Controls the BUS switch
- clock_HasTimeElapsedMs(); - Returns 1 when a certain amount of time in milliseconds has elapsed.
- Clock_GetTimeStampMs(); - Returns the current time in milliseconds
- Inno4Pro_PD_Write_VI(); - Sets the output voltage and current

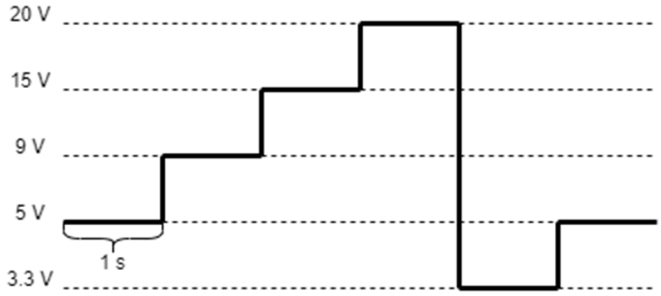


Figure 9. Inno3Pro_PDOs.ino Output Voltage Waveform

```
#include <Drv_Rtc.h>
#include <Drv_i2c.h>
#include <InnoProBase.h>
#include "Inno3Pro.h"
#include "Inno3ProConfig.h"

//Step  2  :  Create the class instance
Inno3Pro_Application Inno3ProApp;

//Step 3 :  Write Initial Commands to Inno Pro
void setup()
{
   Inno3ProApp.Inno3Pro_Initialization();
}

//Step 4 :  Call the Functions on the Main Loop
void loop()
{
            //Control Functions Set-Up
            // 5V  Voltage SetPoint
            // 3.1A Constant Current
            Inno3ProApp.Inno3Pro_PD_Write_VI(5,3.1);

            // 300mV    , Cable Drop Compensation
            Inno3ProApp.Inno3Pro_Write_Cable_Drop_Comp(300);

            // 7V       , Constant Output Power Knee Voltage
            Inno3ProApp.Inno3Pro_Write_Volt_Peak(7);

            // ON       , Vbus Enable
            Inno3ProApp.Inno3Pro_Vbus_Switch_Control(1);
}
```

Figure 10. Inno3Pro_Basic.ino Code

```
#include <Drv_Rtc.h>
#include <Drv_i2c.h>
#include <InnoProBase.h>
#include "Inno3Pro.h"
#include "Inno3ProConfig.h"

//Step  2  :  Create the class instance
InnoProBase_Rtc Inno3ProClk;
Inno3Pro_Application Inno3ProApp;

//Step 3 :  Write Initial Commands to Inno Pro
void setup()
{
            Inno3ProApp.Inno3Pro_Initialization();
            Inno3ProApp.Inno3Pro_Write_Cable_Drop_Comp(300);     // CDC  = 300mV
            Inno3ProApp.Inno3Pro_Write_Volt_Peak(24);            // VKP  = 24V
            Inno3ProApp.Inno3Pro_Vbus_Switch_Control(1);         // VBEN = ON
}

//Step 4 :  Call the Functions on the Main Loop
void loop()
{

   // Main Loop Variables
   static uint16_t u16_Main_State = 0;                //Initialize Main State
   static uint16_t u16_Request_Timer = 0;               //Initialize Request Timer

   //Timer Routine For Automatic Activation of  Requests
   if(Inno3ProClk.clock_HasTimeElapsedMs(u16_Request_Timer,1000))    //Delay Time
   {

     u16_Main_State++;                        //Change State
     u16_Request_Timer = Inno3ProClk.clock_GetTimeStampMs();     //Reset Timer
   }

   // Main Loop States
   switch(u16_Main_State)
   {
         case 0:
         u16_Main_State = 1;
         break;

     case 1:   //Activate 5V Configuration
          // CV = 5V and CC = 3.1A
         Inno3ProApp.Inno3Pro_PD_Write_VI(5,3.1);
         break;

     case 2:     //Activate 9V Configuration
         Inno3ProApp.Inno3Pro_PD_Write_VI(9,3.1);
         break;

     case 3:   //Activate 15V Configuration
         Inno3ProApp.Inno3Pro_PD_Write_VI(15,3.1);
         break;

     case 4:     //Activate 20V Configuration
         Inno3ProApp.Inno3Pro_PD_Write_VI(20,3.1);
         break;

     case 5:   //Activate 3.3V Configuration
         Inno3ProApp.Inno3Pro_PD_Write_VI(3.3,3.1);
         break;

     default:
         u16_Main_State = 1;
         break;
   }

 }
```

Figure 11. Inno3-Pro_PDOs.ino Code

InnoSwitch4-Pro Examples

Example 1 – Inno4Pro_Basic.ino

Inno3Pro_Basic.ino sketch implements the essential commands to run the InnoSwich3-Pro IC. The adapter will output 5V 3.1A while using this code. There are five commands in this sketch to note:

- Inno4Pro_Initialization(); - Function for initializing the InnoSwitch4-Pro device
- Inno4Pro_Write_VI(); - Function for setting the output voltage and current
- Inno4Pro_Write_Volt_Peak(); - Sets the knee voltage (V_{KP})
- Inno4Pro_Vbus_Switch_Control(); - Controls the BUS switch

Example 2 – Inno4Pro_PDOs.ino

This sketch cycles through the multiple output voltages. This mimics the USBPD standard PDO's for 60W adapters.

Main functions used:

- Inno4Pro_Initialization(); - Initializes InnoSwitch4-Pro
- Inno4Pro_Write_Cable_Drop_Comp(); - Sets the cable drop voltage in mV
- Inno4Pro_Write_Volt_Peak() - Sets the knee voltage (V_{KP})
- Inno4Pro_Vbus_Switch_Control() – Controls the BUS switch
- clock_HasTimeElapsedMs(); - Returns 1 when a certain amount of time in milliseconds has elapsed.
- Clock_GetTimeStampMs(); - Returns the current time in milliseconds
- Inno4Pro_PD_Write_VI(); - Sets the output voltage and current

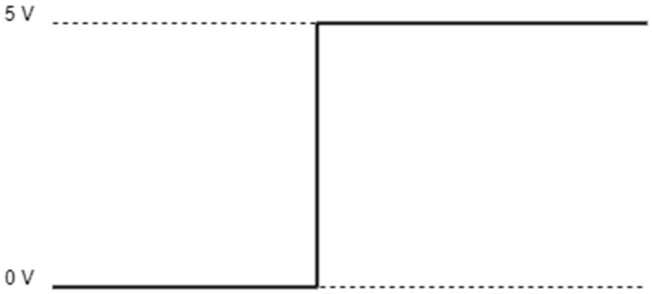


Figure 12. Inno4Pro_Basic.ino Output Voltage Waveform

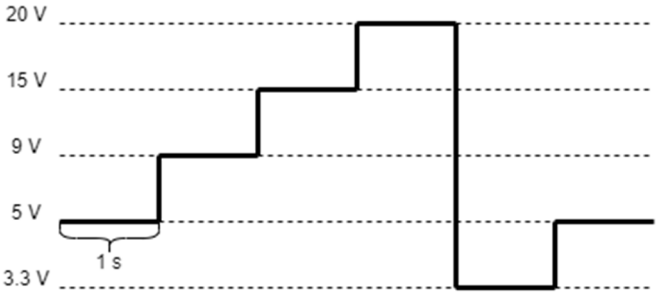


Figure 13. Inno4Pro_PDOs.ino Output Voltage Waveform

```
//Step  1  :  Add the Header Files
#include <Drv_Rtc.h>
#include <Drv_i2c.h>
#include <InnoProBase.h>
#include "Inno4Pro.h"
#include "Inno4ProConfig.h"

//Step  2  :  Create the class instance
Inno4Pro_Application Inno4ProApp;

//Step 3 :  Write Initial Commands to InnoPro
void setup()
{
            Inno4ProApp.Inno4Pro_Initialization();
}

//Step 4 :  Call the Functions on the Main Loop
void loop()
{
    //Control Functions Set-Up
    // 5V  Voltage SetPoint
    //3.1A Constant Current
    Inno4ProApp.Inno4Pro_PD_Write_VI(5,3.1);

    // 300mV    , Cable Drop Compensation
    Inno4ProApp.Inno4Pro_Write_Cable_Drop_Comp(300);

    // 24V      , Constant Output Power Knee Voltage
    Inno4ProApp.Inno4Pro_Write_Volt_Peak(24);

    // ON       , Vbus Enable
    Inno4ProApp.Inno4Pro_Vbus_Switch_Control(1);
}
```

Figure 14. Example 1 - Inno4Pro_Basic.ino Code

```
//Step  1  :  Add the Header Files
#include <Drv_Rtc.h>
#include <Drv_i2c.h>
#include <InnoProBase.h>
#include "Inno4Pro.h"
#include "Inno4ProConfig.h"

//Step  2  :  Create the class instance
InnoProBase_Rtc Inno4ProClk;
Inno4Pro_Application Inno4ProApp;

//Step 3 :  Write Initial Commands to InnoPro
void setup()
{
        Inno4ProApp.Inno4Pro_Initialization();
        Inno4ProApp.Inno4Pro_Write_Cable_Drop_Comp(300);
        Inno4ProApp.Inno4Pro_Write_Volt_Peak(24);
        Inno4ProApp.Inno4Pro_Vbus_Switch_Control(1);
}

//Step 4 :  Call the Functions on the Main Loop
void loop()
{

   // Main Loop Variables
   static uint16_t u16_Main_State = 0;          //Initialize Main State
   static uint16_t u16_Request_Timer = 0; //Initialize Request Timer

   //Timer Routine For Automatic Activation of Requests
   if(Inno4ProClk.clock_HasTimeElapsedMs(u16_Request_Timer,1000))
   {
     u16_Main_State++;                  //Change State

     u16_Request_Timer = Inno4ProClk.clock_GetTimeStampMs ();
   }

   // Main Loop States
   switch(u16_Main_State)
   {

     case 0:
          u16_Main_State = 1;
          break;

     case 1:   //Activate 5V Configuration
          Inno4ProApp.Inno4Pro_PD_Write_VI(5,3.1);
          break;

     case 2:    //Activate 9V Configuration
          Inno4ProApp.Inno4Pro_PD_Write_VI(9,3.1);
          break;

     case 3:   //Activate 15V Configuration
          Inno4ProApp.Inno4Pro_PD_Write_VI(15,3.1);
          break;

     case 4:    //Activate 20V Configuration
          Inno4ProApp.Inno4Pro_PD_Write_VI(20,3.1);
          break;

     case 5:   //Activate 3.3V Configuration
          Inno4ProApp.Inno4Pro_PD_Write_VI(3.3,3.1);
          break;

     default:
          u16_Main_State = 1;
          break;
   }

}
```

Figure 15. Example 2 - Inno4Pro_PDOs.ino Code

Source Files

Name	Date modified	Type	Size
Drv_i2c	16/12/2021 2:34 pm	CPP File	4 KB
Drv_i2c	16/12/2021 2:34 pm	H File	4 KB
Drv_Rtc	16/12/2021 2:34 pm	CPP File	3 KB
Drv_Rtc	16/12/2021 2:34 pm	H File	4 KB
Inno3Pro	22/12/2021 4:11 pm	CPP File	45 KB
Inno3Pro	22/12/2021 4:11 pm	H File	67 KB
Inno3ProConfig	23/12/2021 9:19 am	H File	26 KB
Inno4Pro	06/01/2022 11:09 am	CPP File	50 KB
Inno4Pro	22/12/2021 4:11 pm	H File	69 KB
Inno4ProConfig	23/12/2021 9:19 am	H File	28 KB
InnoProBase	16/12/2021 2:34 pm	CPP File	10 KB
InnoProBase	16/12/2021 2:34 pm	H File	14 KB
LcdKeypad	16/12/2021 2:34 pm	CPP File	4 KB
LcdKeypad	16/12/2021 2:34 pm	H File	3 KB

API – Handles command sequences, timings, register settings, threshold calculations, parity implementations, telemetry, and etc.
Code core used by both InnoSwitch3-Pro and InnoSwitch4-Pro

- *InnoProBase.h*
- *InnoProBase.cpp*

Code core limited to InnoSwitch3-Pro

- *Inno3Pro.h*
- *Inno3Pro.cpp*

Code core limited to InnoSwitch4-Pro

- Inno4Pro.h
- Inno4Pro.cpp

Code core for controlling LCD Keypad Arduino Shield

- *LcdKeypad.h*
- *LcdKeypad.cpp*

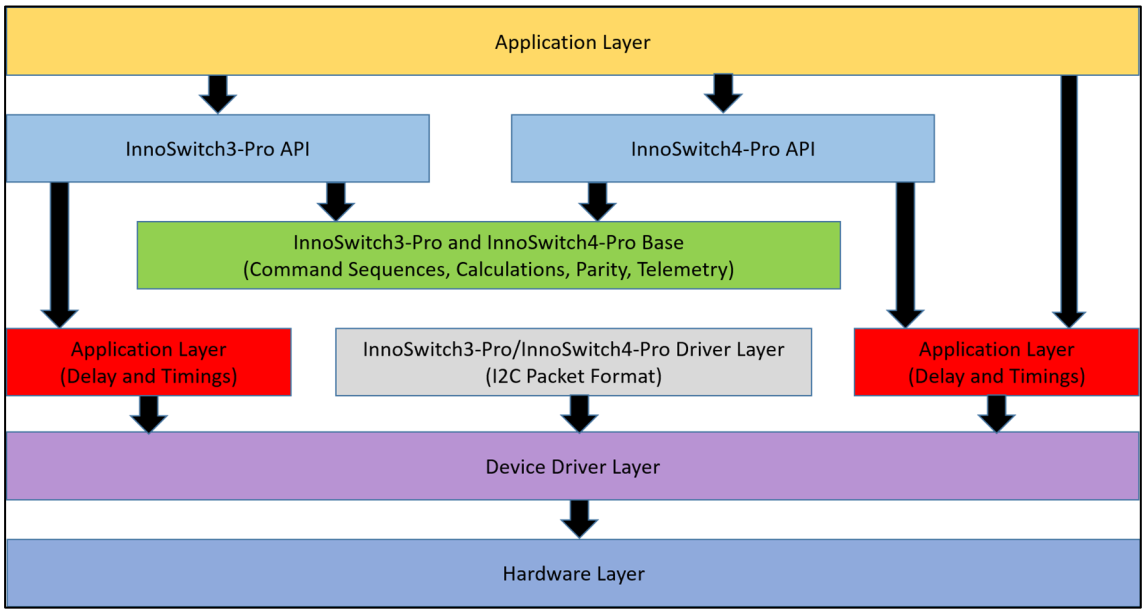
InnoSwitch Driver – Manages I^2C packet format based on InnoSwitch3-Pro/InnoSwitch4-Pro datasheet for write and read transactions. Arduino Wire library was used as the lower level library.

- *Drv_I2C.h*
- *Drv_I2C.cpp*

Clock Driver – A module for generating delays and timings

- *Drv_Rtc.h*
- *Drv_Rtc.cpp*

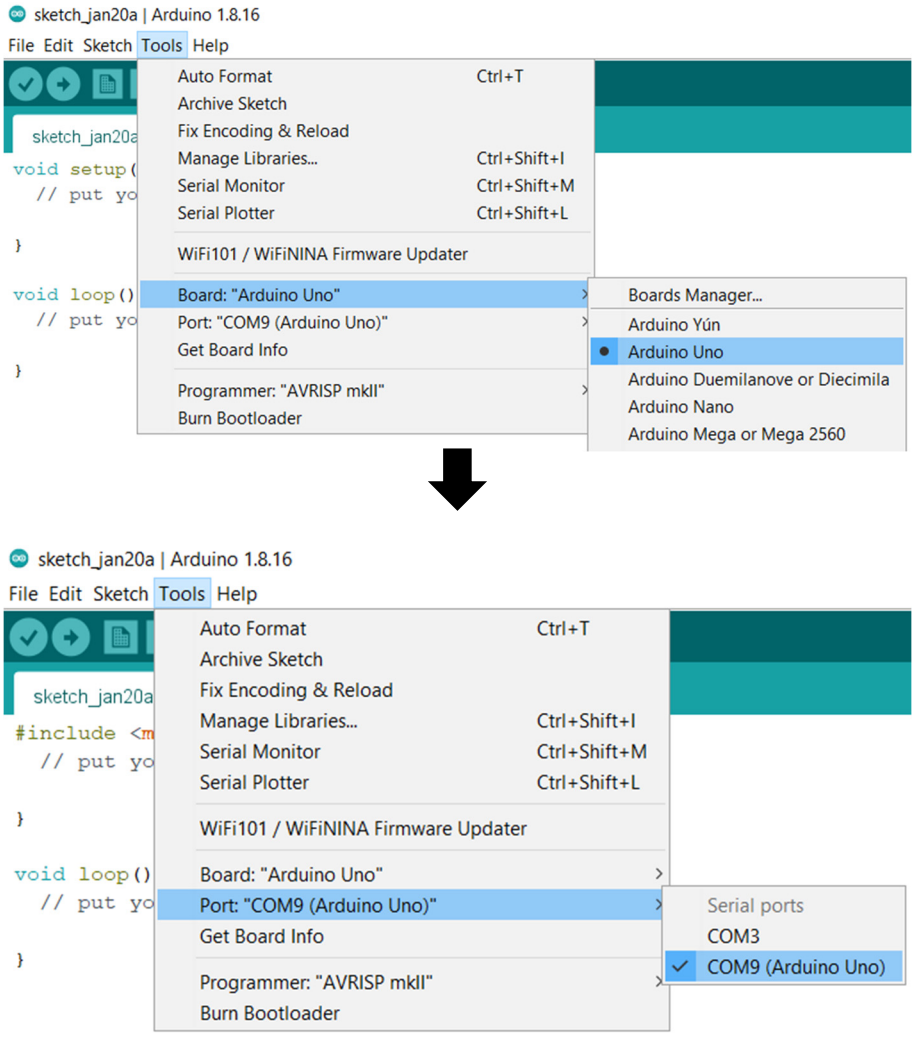
The figure below shows how each of the layers interact with each other. The application layer consists of the InnoSwitch3-Pro/InnoSwitch4-Pro Arduino sketches that implement the function of the InnoSwich3-Pro/InnoSwitch4-Pro and Clock Driver functions.



Building the Project

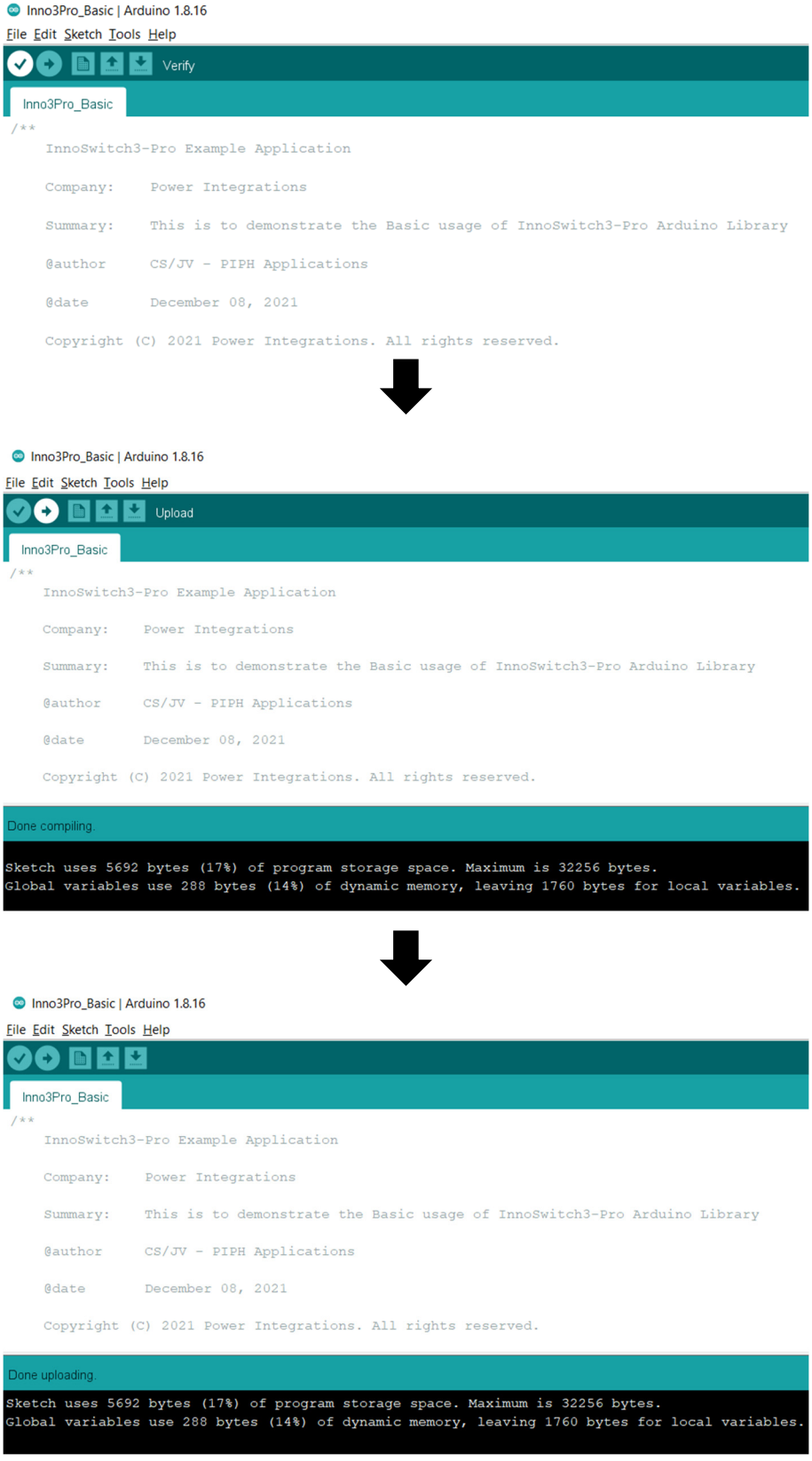
Board Selection

1. Go to *Tools menu > Board:* and select the Arduino device used.
2. Select also the active COM port number from *Tools menu > Port* which determines which USB port the Arduino board is connected.



Sketch Upload

1. Select the desired sketch to use from *File menu > Examples > InnoSwitch3-Pro and InnoSwitch4-Pro Library.* Click on the Verify icon on the upper left corner of the IDE.
2. An indicator will show on the bottom of the screen that says "Done compiling" along with the amount of memory used by the sketch.
3. Click on the Arrow icon beside the Verify button to upload the compiled sketch onto the Arduino board. Once done, an indicator on top of the debug log will show "Done Uploading".

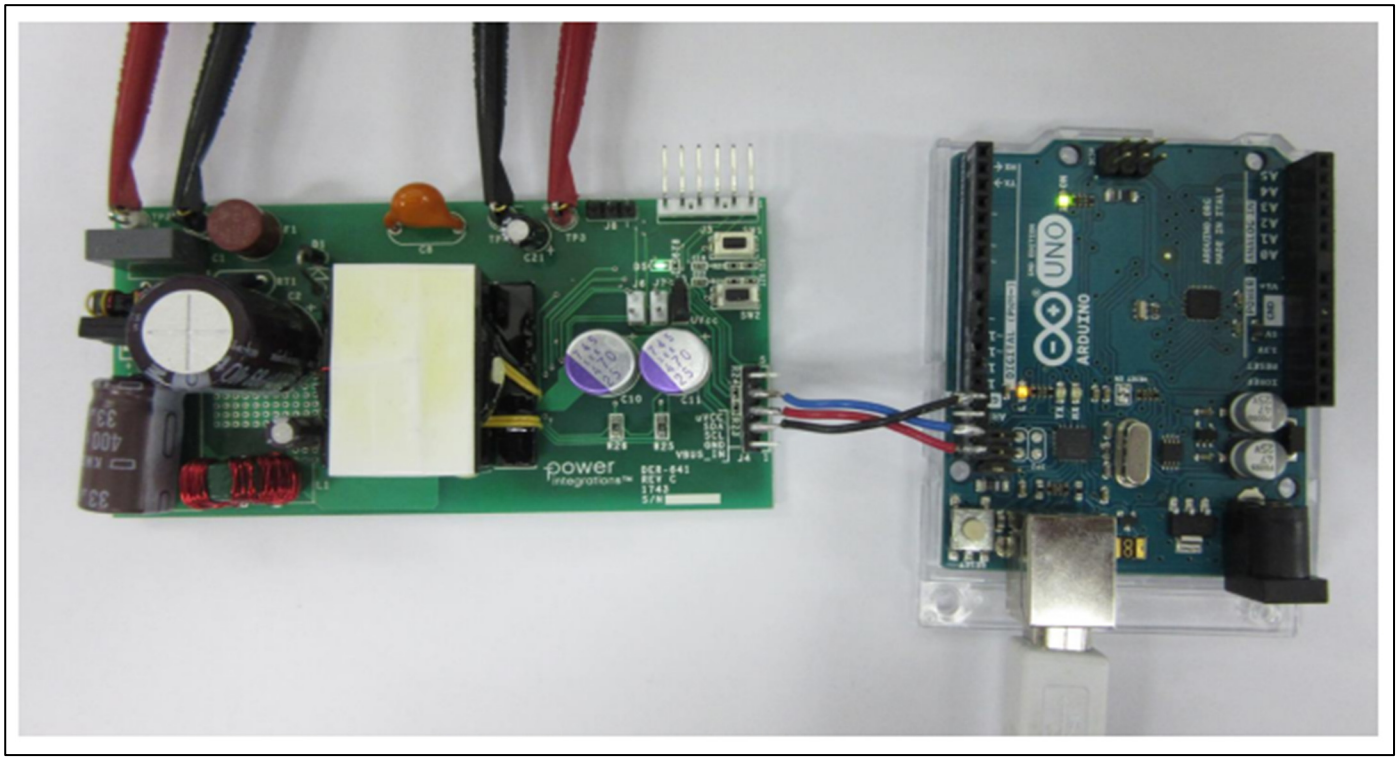


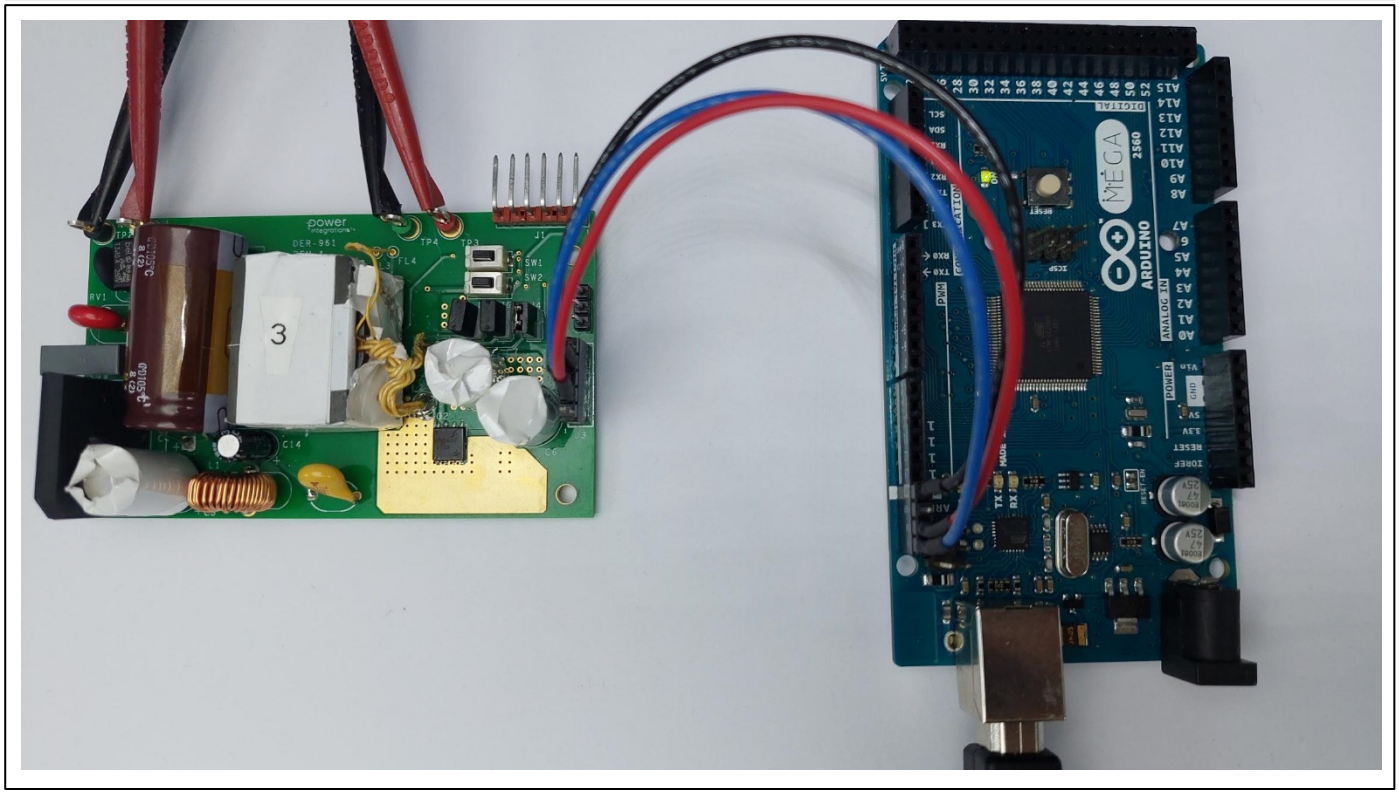
Hardware Setup

This section demonstrates the use of the Arduino library in controlling the InnoSwitch3-Pro device. The sketch used in this example is

Inno3_Basic.ino where it initializes the InnoSwitch3-Pro to output 5V and 3.1A. Remove jumpers J6 and J7 on the RDK-641 board. Connect the I^2C lines of the Arduino board to the RDK-641 as shown in the figure below. Upon power up of RDK-641, the sketch is uploaded to the Arduino board. The output of the RDK-641 should be 5V with a 3.1A current limit.

InnoSwitch4-Pro setup uses the RDR-961 board instead of the RDK-641. Inno4Pro_PDOs.ino sketch changes the output voltage every second from 5 V> 9 V> 15 V> 20 V> 5 V through the use of multiple Inno4Pro_PD_Write_VI() commands.

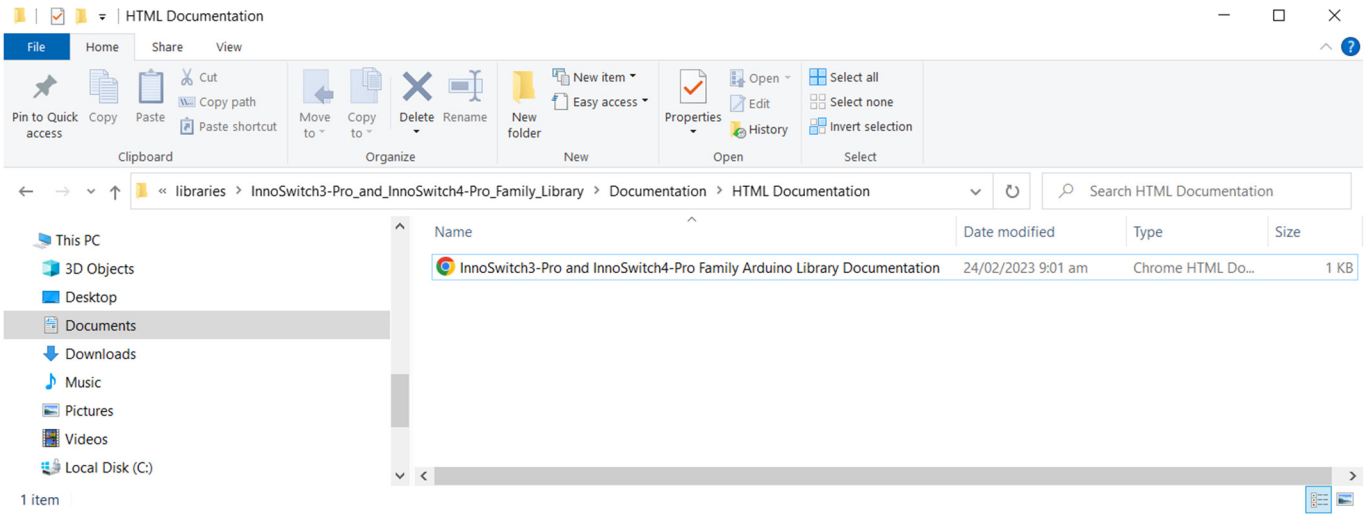




Doxygen Documentation

There are compiled HTML (.chm) and HTML (.html) files in the Documentation folder. These files contain the documentation of the InnoSwitch3-Pro and InnoSwitch4-Pro Arduino Library. It contains brief descriptions on how to use each function in the API and core drivers. The examples in the document

shows a brief overview on how the code works as well as a guide on how to use each sketch.

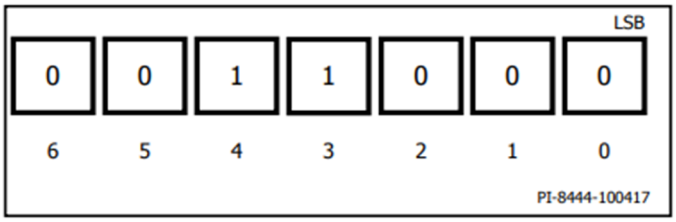


Appendix

Register Definition

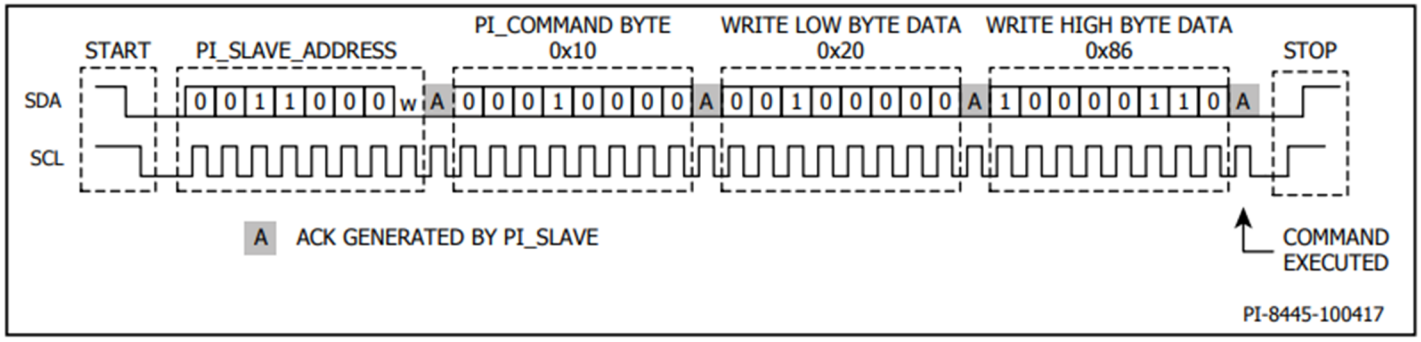
I²C Slave Address

The InnoSwitch3-Pro and InnoSwitch4-Pro 7-bit slave address is 0x18 (7'b001 1000).

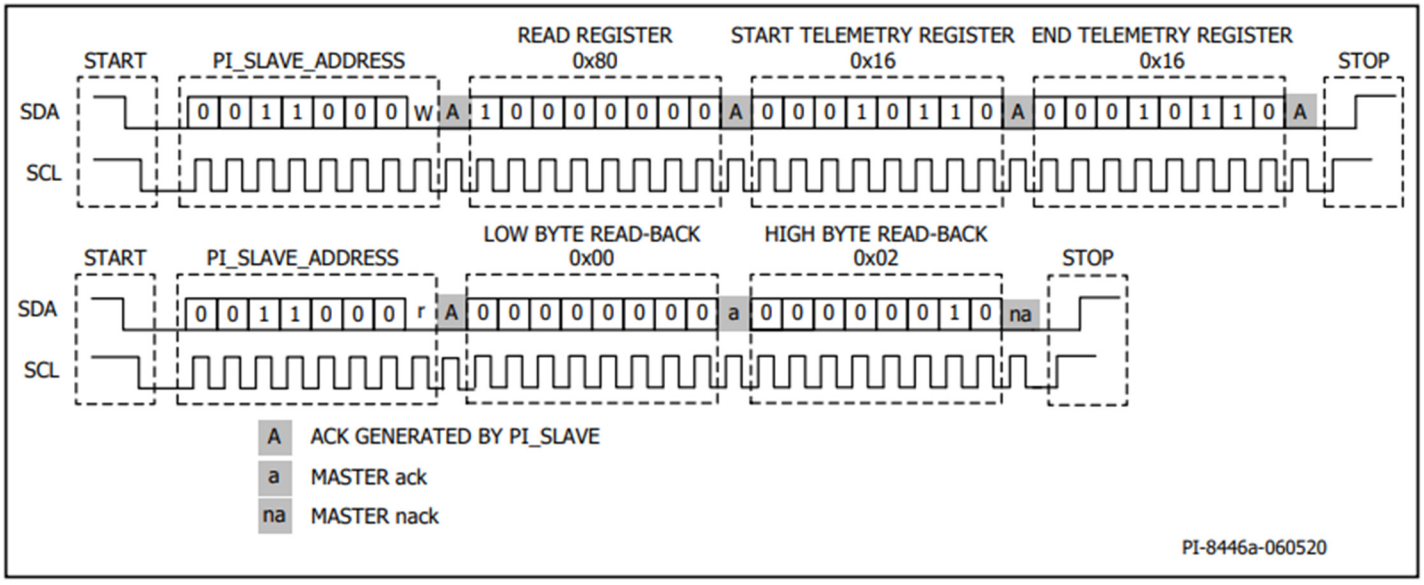


I²C Protocol Format

3-Byte Write Commands:
[PI_SLAVE_ADDRESS][W][A][PI_COMMAND][A][Byte][A] or
[PI_SLAVE_ADDRESS][W][A][PI_COMMAND][A][Low Byte][A][High Byte][A]



2-Byte Read Commands:
[PI_SLAVE_ADDRESS][W][A][PI_COMMAND][A][START_TELEMETRY_REGISTER_ADDRESS][A][END_TELEMETRY_REGISTER_ADDRESS][A]
[PI_SLAVE_ADDRESS][r][A][PI Slave responds Low Byte][a][PI Slave responds High Byte][na]



Revision	Notes	Date
A	Initial release.	01/20/23

For the latest updates, visit our website: www.power.com
Power Integrations reserves the right to make changes to its products at any time to improve reliability or manufacturability. Power Integrations does not assume any liability arising from the use of any device or circuit described herein. POWER INTEGRATIONS MAKES NO WARRANTY HEREIN AND SPECIFICALLY DISCLAIMS ALL WARRANTIES INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF THIRD PARTY RIGHTS.

Patent Information
The products and applications illustrated herein (including transformer construction and circuits external to the products) may be covered by one or more U.S. and foreign patents, or potentially by pending U.S. and foreign patent applications assigned to Power Integrations. A complete list of Power Integrations patents may be found at www.power.com. Power Integrations grants its customers a license under certain patent rights as set forth at www.power.com/ip.htm.

Life Support Policy
POWER INTEGRATIONS PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF POWER INTEGRATIONS. As used herein:

1. A Life support device or system is one which, (i) is intended for surgical implant into the body, or (ii) supports or sustains life, and (iii) whose failure to perform, when properly used in accordance with instructions for use, can be reasonably expected to result in significant injury or death to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

Power Integrations Worldwide Sales Support Locations

World Headquarters
5245 Hellyer Avenue
San Jose, CA 95138, USA
Main: +1-408-414-9200
Customer Service:
Worldwide: +1-65-635-64480
Americas: +1-408-414-9621
e-mail: usasales@power.com

China (Shanghai)
Rm 2410, Charity Plaza, No. 88
North Caoxi Road
Shanghai, PRC 200030
Phone: +86-21-6354-6323
e-mail: chinasales@power.com

China (Shenzhen)
17/F, Hivac Building, No. 2, Keji Nan
8th Road, Nanshan District,
Shenzhen, China, 518057
Phone: +86-755-8672-8689
e-mail: chinasales@power.com

Germany (AC-DC/LED Sales)
Einsteinring 24
85609 Dornach/Aschheim
Germany
Tel: +49-89-5527-39100
e-mail: eurosales@power.com

Germany (Gate Driver Sales)
HellwegForum 1
59469 Ense
Germany
Tel: +49-2938-64-39990
e-mail: igbt-driver.sales@power.com

India
#1, 14th Main Road
Vasanthanagar
Bangalore-560052 India
Phone: +91-80-4113-8020
e-mail: indiasales@power.com

Italy
Via Milanese 20, 3rd. Fl.
20099 Sesto San Giovanni (MI) Italy
Phone: +39-024-550-8701
e-mail: eurosales@power.com

Japan
Yusen Shin-Yokohama 1-chome Bldg.
1-7-9, Shin-Yokohama, Kohoku-ku
Yokohama-shi,
Kanagawa 222-0033 Japan
Phone: +81-45-471-1021
e-mail: japansales@power.com

Korea
RM 602, 6FL
Korea City Air Terminal B/D, 159-6
Samsung-Dong, Kangnam-Gu,
Seoul, 135-728, Korea
Phone: +82-2-2016-6610
e-mail: koreasales@power.com

Singapore
51 Newton Road
#19-01/05 Goldhill Plaza
Singapore, 308900
Phone: +65-6358-2160
e-mail: singaporesales@power.com

Taiwan
5F, No. 318, Nei Hu Rd., Sec. 1
Nei Hu Dist.
Taipei 11493, Taiwan R.O.C.
Phone: +886-2-2659-4570
e-mail: taiwansales@power.com

UK
Building 5, Suite 21
The Westbrook Centre
Milton Road
Cambridge
CB4 1YG
Phone: +44 (0) 7823-557484
e-mail: eurosales@power.com