SEED TECHNOLOGY INC (SEEEDUINO)
Seeeduino ADK Main Board
Model: ARD52028P

Introduction

Seeeduino ADK Main Board is an Android Open Accessory Development Kit(**ADK**). Compared to the Google's ADK Reference design, Seeeduino ADK has many added features like support of both **5V** and **3.3V I/O** (logic with a switch), **smaller form-factor**, **better placement of reset button**, **pads for more pin-headers** and **sparkling RED PCB with Golden finish**. Using ADK Main Board build your own Mobile accessories. Open Source Android development platform and **Seeeduino ADK Main Board** is an ideal solution for mobile based Home Automation.

Seeeduino ADK Main Board supports Android devices v1.5 using **MicroBridge** and v2.3.4 and above with Google Open Accessories API (ADK).

Seeeduino ADK Main Board works like Arduino **Mega 2560** with inbuilt USB Host Shield. Hence, this Main Board can be connected to any USB devices. Users have to write their own drivers for using this feature.

What is Android Open Accessory Development Kit?

The Android 3.1 platform (also backported to Android 2.3.4) introduces Android Open Accessory support, which allows external USB hardware (an Android USB accessory) to interact with an Android-powered device in a special "accessory" mode. When an Android-powered powered device is in accessory mode, the connected accessory acts as the USB host (powers the bus and enumerates devices) and the Android-powered device acts as the device.

Android USB accessories are specifically designed to attach to Android-powered devices and adhere to a simple protocol (Android accessory protocol) that allows them to detect Android-powered devices that support accessory mode. Accessories must also provide 500mA at 5V for charging power. Many previously released Android-powered devices are only capable of acting as a USB device and cannot initiate connections with external USB devices. Android Open Accessory support overcomes this limitation and allows you to build accessories that can interact with an assortment of Android-powered devices by allowing the accessory to initiate the connection.

Features

- Android Open Accessories development Kit (**ADK**) compatible:
- Supports Android v2.3.4 and above devices.
- Works with Android Debug Bridge (ADB) using MicroBridge :
- Supports Android v1.5 and above devices.
- Arduino Mega 2560 compatible (**256K** Flash MCU)
- Simply works like an Arduino Mega with an integrated USB Shield
- 56 Digital IOs
- 16 Analog inputs
- 14 PWM outputs
- 4 Hardware serial ports (UART)
- 1 Hardware TWI (I2C)
- 1 Hardware SPI (up to 8Mbps)
- On board USB host(**MAX3421**), and breakout for all I/Os pins.
- On board USB slave(**FT232RL**), and IOs breakout
- Build-in 5V-1A switched power regulator (input range 6V - 18V)
- Build-in 3.3V-500mA LDO power regulator.
- Red PCB, ROHS compatible and Golden finish.
- 5v/3v3 IO Level selectable (Atmega2560 running on 16MHz@3.3v is a bit over-clock, but according to the test result, works fine).

Application Ideas

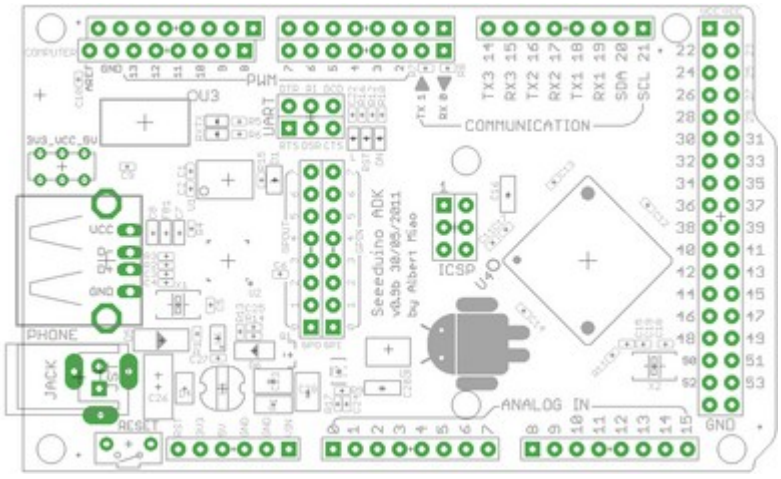
- Develop accessories for Android (v2.3.4 and above) devices using ADK.
- For Android devices not supporting ADK, use MicroBridge to develop accessories.
- Make Android devices interact with the physical world.
- As an Arduino compatible platform with 256K Flash.
- On board USB host makes it easy to interact with USB devices like pen drivers, keyboard, mouse, Bluetooth dongles.

Cautions

- Do not connect external DC power-supply while Seeeduino ADK Main Board is connected to PC.

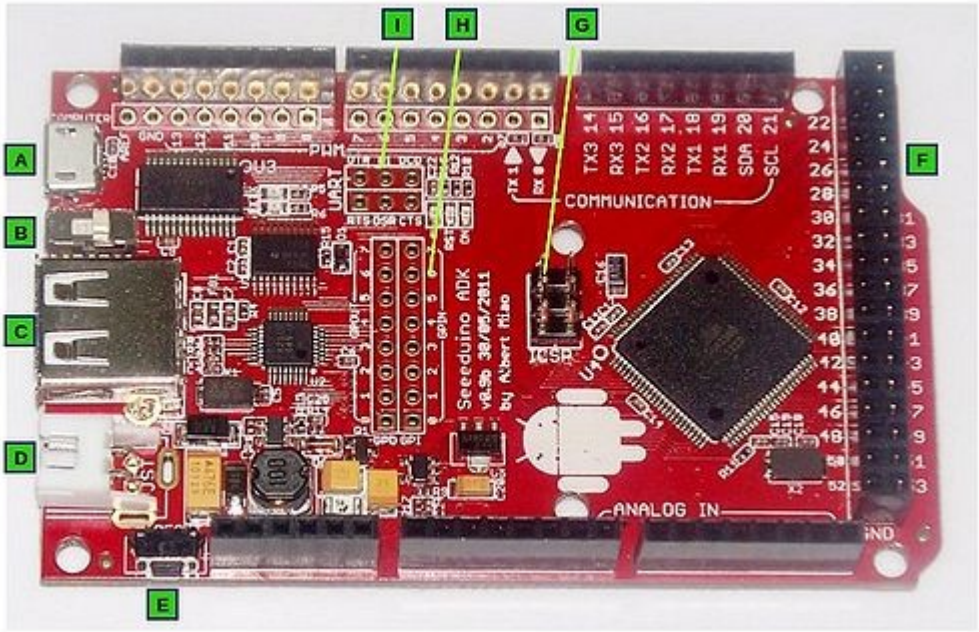
Mechanical Dimensions

- Dimensions: 3.4" x 2.1"



Usage

Hardware



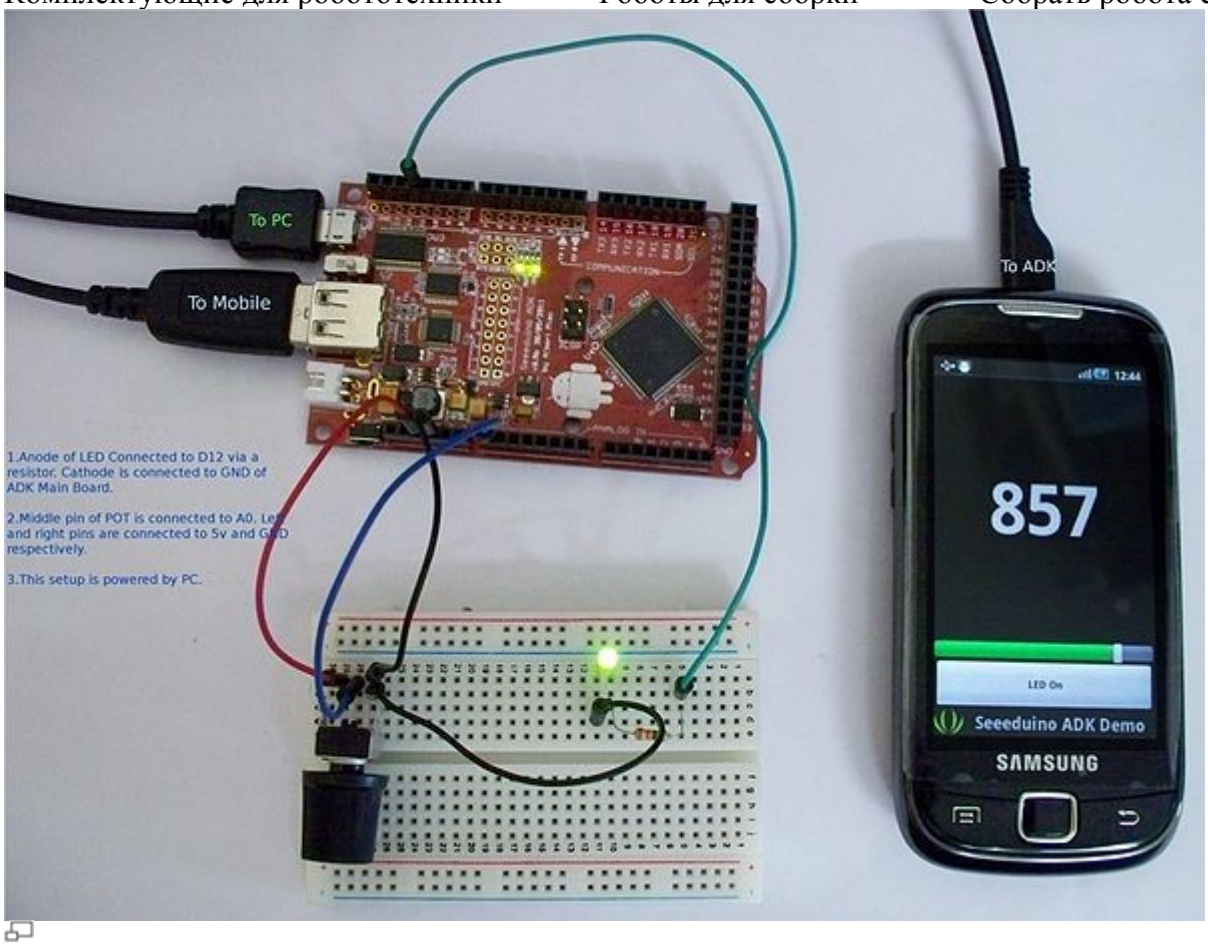
Seeeduino ADK Salient Features

Seeeduino ADK Hardware blocks are listed below:

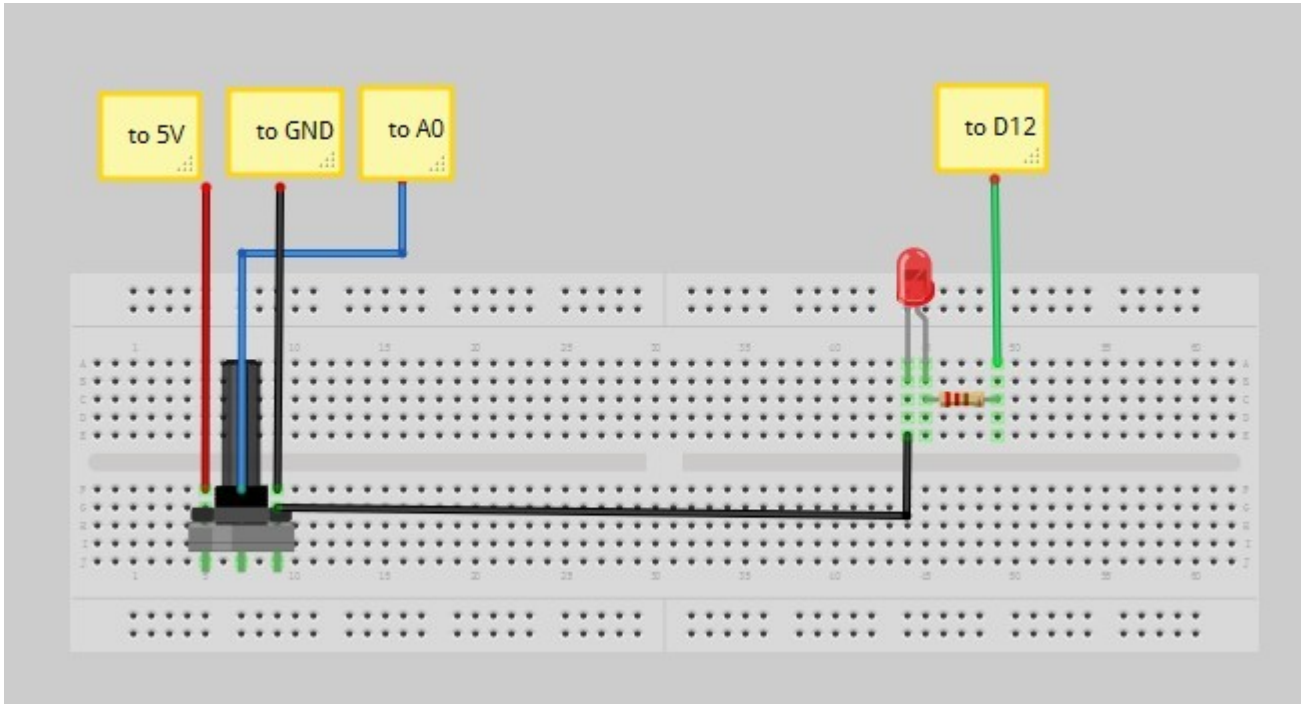
Item	Description
A	**Micro USB Socket**: Connects main board to PC. Used for uploading sketch using Arduino IDE.
B	**Slide switch** to choose operating or I/O voltage: 3.3V or 5V
C	**USB A Plug** : Connects to Android Mobile Device.
D	**JST Conector / DC Jack** : For external DC power supply. Do not connect PC while using external DC.
E	**Reset Button**: Conveniently placed at the side to allow using reset while using shields.
F	**I/O pins**
G	**ICSP**: For programming Arduino Bootloader using AVR ICSP
H	Breakouts pins of **Max3421E GPIO**
I	Breakouts pins of **FT232RL**

Hardware Installation

The following image illustrates an application example of **Seeeduino ADK Main Board** with an Android Mobile. It runs the demo application based on **MicroBridge** provided illustrated this page. All basic electronic components are taken from Arduino Sidekick Basic Kit.



Seeeduino ADK Main Board and Android Mobile Connection



- **Uploading Firmware**
  - Set the VCC slide switch to 5V.
  - Connect the Seeeduino ADK Main Board - Micro USB to PC USB port.
  - Set the Board type in Arduino IDE to **Arduino Mega 2560**.
  - Compile the Demo Sketch and upload to Main Board.

- **Android App**
  - Install Android Platform Development Software.
  - Import the demo Android app to Eclipse Workspace.

- Connect the mobile device to PC and upload the application

- Connect the mobile to ADK Main Board.
- Enable ADB in your Mobile device if not already enabled. This is only for MicroBridge.
- Push the Reset button.

Programming

Using MicroBridge

The sample applications and library are available in the resources section. The following Arduino Sketch and Android code are commented well explaining the usage.

- **SeeeduinoADKDemo.pde**

```
//Seeeduino ADK Demo using Niels Brouwers' MicroBridge library.
//Connect a LED to D12 and a variable resistor(POT) to A0

#include <SPI.h>
#include <Adb.h>

// Adb connection.
Connection * connection;

// Elapsed time for ADC sampling. The rate at which ADC value is sent to Android device.
long lastTime;

//State of LED. Initially OFF.
uint8_t LEDState=0;

// Event handler for the shell connection.
// This event handler is called whenever data is sent from Android Device to Seeeduino ADK.
// Any data / command to be sent to I/O of ADK has to be handled here.
//
// For eg: 1.Controlling an ouput port 2.Interacting with a device connected
// to ADK via IIC or Serial Port.

void adbEventHandler(Connection * connection, adb_eventType event, uint16_t length, uint8_t
* data)
{

  // In this example Data packets contain one byte and it decides the state of a LED
connected to D12
  // The size of data is predetermined for this application. Android device also uses the
same size.

  if (event == ADB_CONNECTION_RECEIVE)
  {
     if(LEDState != data[0])
     {
         digitalWrite(12, data[0]);   // Change the state of LED
         Serial.println(data[0],DEC);
        LEDState = data[0];          // Store the State of LED
     }
  }

}

void setup()
{
  //Serial port debug purpose
  Serial.begin(57600);

  // Note start time
  lastTime = millis();
```

```
  // Set Digital pin 12 (LED is connected) as output
  pinMode(12,OUTPUT);

  // Initialise the ADB subsystem.
  ADB::init();

  // Open an ADB stream to the phone's shell. Auto-reconnect. Use any unused port number
eg:4568
  connection = ADB::addConnection("tcp:4568", true, adbEventHandler);

}

void loop()
{
  //Check if ADC needs to be sampled.
  if ((millis() - lastTime) > 20)
  {
    //Read ADC value
    uint16_t data = analogRead(A0);

    //Send the ADC value to Android device as two bytes of data.
    connection->write(2,(uint8_t*)&data);
    lastTime = millis();
  }

  // Poll the ADB subsystem.
  ADB::poll();
}
```

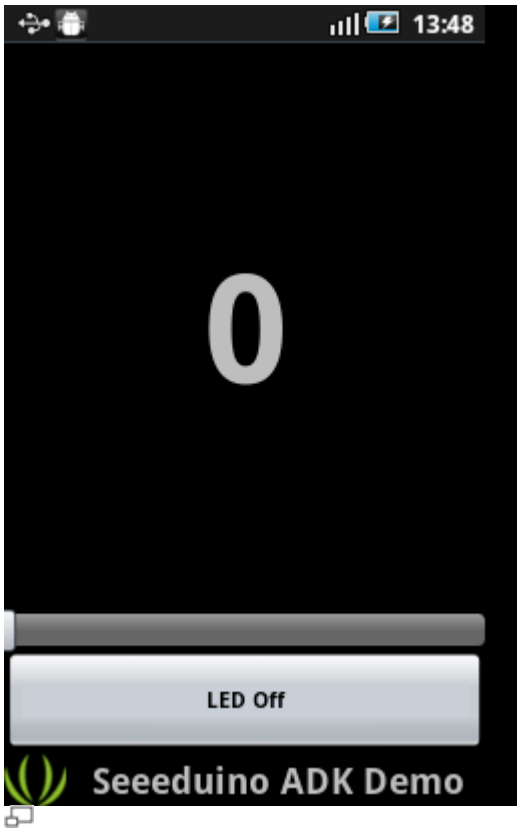
- **Android Application**
- Download the complete Android application from Seeeduino ADK Demo application package. The main java file is listed below with usage comments :

```
/* Application demonstrates the interaction between Seeeduino ADK and Android Device
 * using Niels Brouwers' MicroBridge library.
 *
 * Android Device: Any device with Android v1.5 which supports ADB(Android Debug Bridge).
 *
 * This application uses a very simple (or a trivial) design to make it understandable.
 *
 * Overview:
 * 1.Seeeduino ADK Main Board periodically samples Analog Channel 0 and sends it
 *   to Android Device for display. This value is displayed using a TextView and SeekBar
Widgets
 *
 * 2.Android device controls the state of a LED connected to Digital Pin 12 of ADK Main
Board.
 *   A Button Widget used for this.
 *
 * Microbridge uses ADB based client-server implementation. The Server code that runs on
Android
 * device runs in a separate thread. Hence any update to UI widgets value has to be carried
out
 * in UI thread. This application uses XML based UI creation as it is easier for adding
addition
 * UI Widgets.
 *
 */
package com.seeedstudio.SeeeduinoADKDemo;

import java.io.IOException;

import org.microbridge.server.AbstractServerListener;
import org.microbridge.server.Server;

import android.view.View;
import android.view.Window;
import android.view.View.OnClickListener;
```

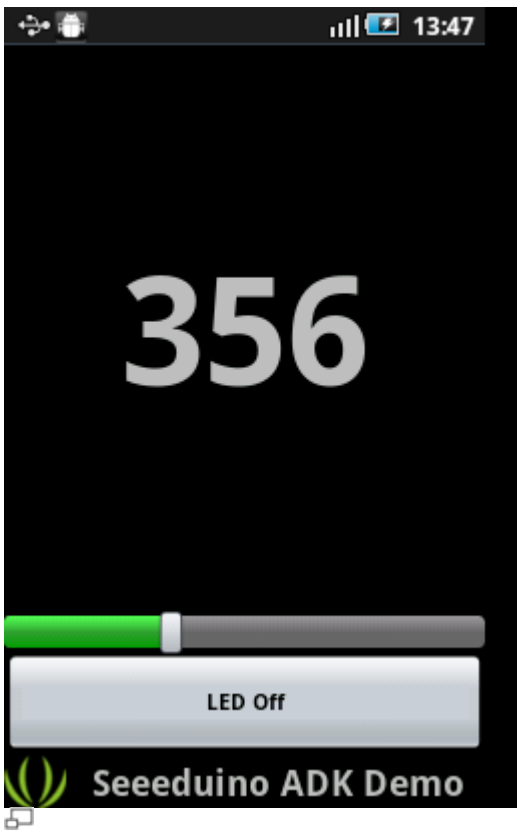
```
import android.app.Activity;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Button;


public class SeeeduinoADKDemo extends Activity implements OnClickListener {
        private int adcSensorValue=10;

        //UI Widgets
        TextView tvAdcvalue;
        SeekBar sbAdcValue;
        Button bOutPutLED;

        boolean LEDState = false ; //initially OFF

        // Create TCP server (based on  MicroBridge LightWeight Server).
        // Note: This Server runs in a separate thread.
        Server server = null;

        /** Called when the activity is first created. */
        @Override
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                requestWindowFeature(Window.FEATURE_NO_TITLE);
                setContentView(R.layout.main);

                bOutPutLED = (Button) findViewById(R.id.buttonOuputLED);
                bOutPutLED.setOnClickListener(this);


                // Create TCP server (based on  MicroBridge LightWeight Server)
                try
                {
                        server = new Server(4568); //Use the same port number used in ADK
Main Board firmware
                        server.start();
                } catch (IOException e)
                {
                        Log.e("Seeeduino ADK", "Unable to start TCP server", e);
                        System.exit(-1);
                }

                server.addListener(new AbstractServerListener() {

                        @Override
                        public void onReceive(org.microbridge.server.Client client, byte[]
data)
                        {

                                if (data.length<2) return;
                                adcSensorValue = (data[0] & 0xff) | ((data[1] & 0xff) << 8);

                                //Any update to UI can not be carried out in a non UI thread
like the one used
                                //for Server. Hence runOnUIThread is used.
                                runOnUiThread(new Runnable() {
                                        @Override
                                        public void run() {
                                                new UpdateData().execute(adcSensorValue);

                                        }
                                });

                        }

                });
```

```
	}	//End of TCP Server code

	// UpdateData Asynchronously sends the value received from ADK Main Board.
	// This is triggered by onReceive()
	class UpdateData extends AsyncTask<Integer, Integer, String> {
		// Called to initiate the background activity
		@Override
		protected String doInBackground(Integer... sensorValue) {

			//Init SeeekBar Widget to display ADC sensor value in SeekBar
			//Max value of SeekBar is set to 1024
			SeekBar sbAdcValue = (SeekBar) findViewById(R.id.sbADCValue);
			sbAdcValue.setProgress(sensorValue[0]);
			return (String.valueOf(sensorValue[0]));  //This goes to result

		}

		// Called when there's a status to be updated
		@Override
		protected void onProgressUpdate(Integer... values) {
			super.onProgressUpdate(values);
			// Not used in this case
		}

		// Called once the background activity has completed
		@Override
		protected void onPostExecute(String result) {
			//Init TextView Widget to display ADC sensor value in numeric.
			TextView tvAdcvalue = (TextView) findViewById(R.id.tvADCValue);
			tvAdcvalue.setText(String.valueOf(result));

		}
	}

	//Called when the LED button is clicked
	@Override
	public void onClick(View v) {
		byte data;


		// Toggle the state of LED
		if(LEDState == true)
		{
			LEDState = false;
			data = 0;
			bOutPutLED.setText("LED Off");
		}
		else
		{
			LEDState = true;
			data = 1;
			bOutPutLED.setText("LED On");
		}

		try
		{
			//Send the state of LED to ADK Main Board as a byte
			server.send(new byte[] {(byte) data});
		} catch (IOException e)
		{
			Log.e("Seeeduino ADK", "problem sending TCP message", e);
		}

	}

}
```

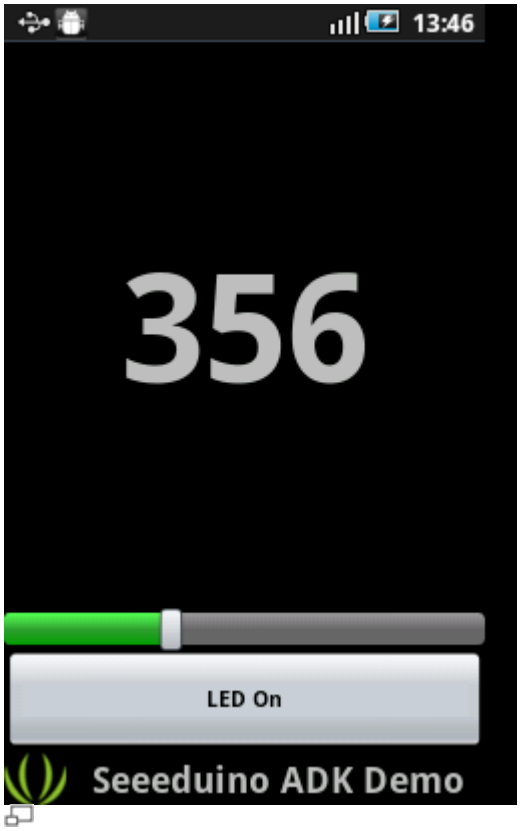
Application Screenshots



LED is switched Off and ADC measures 0



POT is rotated. LED is still Off



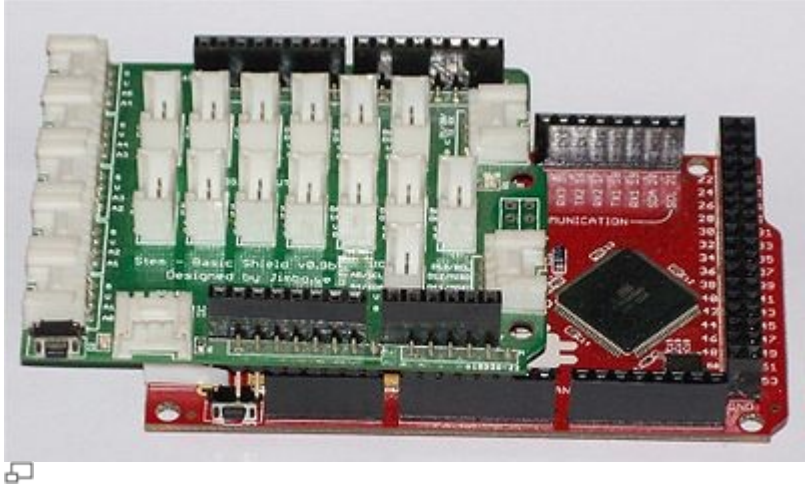
LED is swiched On. This changes the Button's caption.

Using Google ADK

Visit Android ADK Developer page for complete documentation on how to use Accessory API.

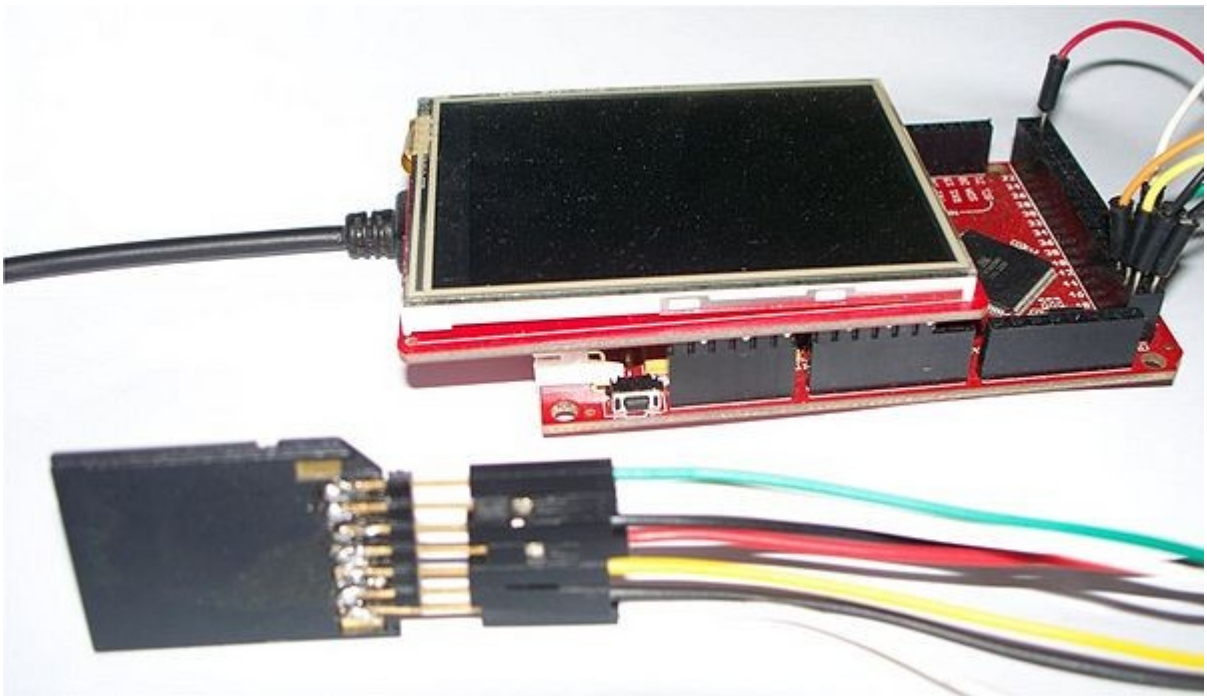
As Mega 2560

- **Seeeduino ADK Main Board** can be used as **Mega 2560**. It also works well with GROVE System. The Grove - Base Shield can be used to connect the numerous Grove modules available.

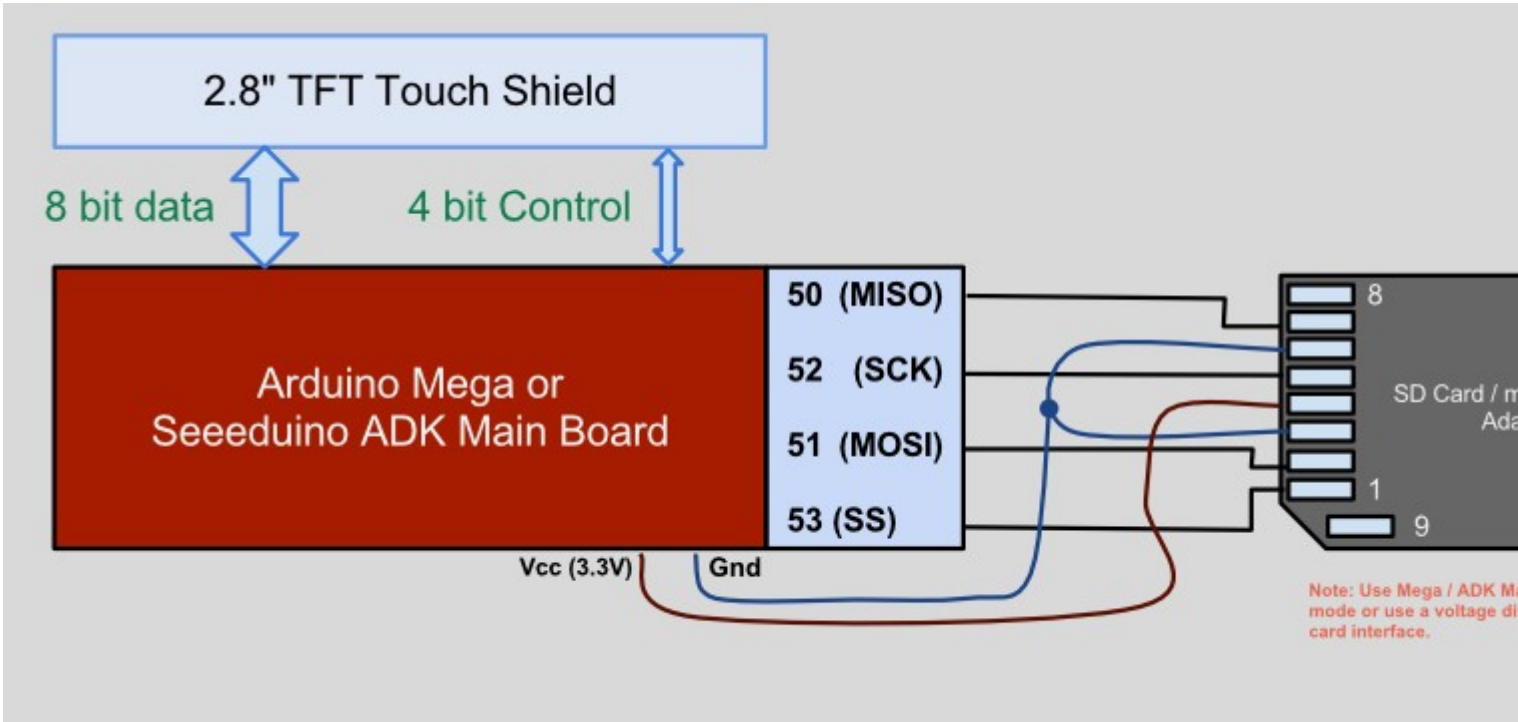


Seeeduino ADK Main Board with Grove - Base Shield

2.8" TFT Touch Shield -- Digital Photo Frame Demo



- Connect **Seeeduino ADK Main Board** to 2.8" TFT Touch Shield
- Format SD card in FAT mode
- Set the operating voltage slide switch to 3.3V
- Copy few 24-bit Bitmap Images (.bmp) files of 240 x 320 size to SD Card. Few samples are present in bmp demo application archive
- Connect the SD card / microSD Card Adapter (with a microSD card) as shown in the illustration to **Seeeduino ADK Main Board**



- Download and install TFT Touch Library
- Download SD Card library and install it to Arduino folder.

- Compile and upload this sketch.

Output

- Refer 2.8" TFT Touch Shield for more information.

- **Oleg Mazurov** of Circuits@Home is the one who originally designed USB Host Shield based on MAX3421E. This was adapted by Google's ADK Referece Board. His site has tons of information and code examples to use MAX3421E based USB Host Shield to with USB keyboard, Mouse, Bluetooth Dongle, Wii Remote, etc.

Bill of Materials (BOM) /parts list

Part	Quantity	Value	Package
3V3_VCC_5V	1	SWITCH-2CH-6P-2.54	2.54SW-2CH
C1-C3,C6-C3,C11-C15,C17-C15,C24,C24	13	100nF	0402
C4,C5,C18,C19	4	13pF	0402
C7,C8	2	10v_1uF	0603
C10,C27	2	10nF	0402
C16	1	10v_10uF	AVX-A
C20	1	22uF_6.3v	AVX-A
C23,C28	2	10v_47uF	AVX-B
C25	1	1nF	0402
C26	1	25v_47uF	AVX-C
COMPUTER	1	Micro USB (MOLEX-47364-0001)	MOLEX-47364-0001
D+,D-,GND,U5V	4	DNP	TP_3535
D1	1	1N4448WS	D0805
D4	1	SS1P3L	DO-220AA
D5	1	1N4007	D2010
D6	1	LL4148	D1206
F1	1	500mA	1206
FB1	1	Ferrite Bead 120R 500mA	0603

GPI,GPO,J3E,J4E	4	DNP	CK_1X8
GPX,HOST_RST,INT	3	DNP	JUMPER
ICSP	1	2.54 male 2*3p	CK_2X3
INTERFACE,J3,J4,J5,J6	5	2.54 female 8p	CK_1X8
J1	1	2.54 female 6p	2.54_6P
J2	1	2.54 Female 2*18p	2.54-2X18P
JACK	1	DNP	3.5MM_JACK
JST	1	JST	PWR_JST
L,ON	2	GREEN	D0603
L1	1	IDCP-2218-10uH	2281
MARK1,MARK2,MARK3,MARK4	4	DNP	MARK_30
PHONE	1	USB-A Port	PN87520
Q1	1	si2305DS	SOT23_
R1,R9,R17	3	100k	0402
R2,R3	2	33R	0402
R4	1	4K7	0402
R5,R6,R10,R12,R16	5	1K	0402
R7,R8	2	1k	0402
R11,R15	2	10k	0402
R13	1	8.2k_1%	0402
R14	1	43k_1%	0402
RESET	1	BOTTON-2P-DP	BTN-5.0-RST
RST,RX,TX	3	RED	D0603
U1	1	TXB0108PW	TSSOP20
U2	1	MAX3421EEHJ+	TQFP32-05
U3	1	FT232RL	SSOP28
U4	1	ATMEGA2560-16AU	TQFP100
U5	1	LD1117-3.3	SOT89
U6	1	LM2734YMK	TSOT6
U7	1	LM321MF	SOT23-5
UART	1	DNP	CK_2X3
X1	1	12.000MHz	5032
X2	1	16.000MHz	5032

FAQ

Please list your question here:

Support

If you have questions or other better design ideas, you can go to our forum or wish to discuss.

Version Tracker

Revision	Descriptions	Release
v0.9b	Initial public release	July 2011

Bug Tracker

Bug Tracker is the place you can publish any bugs you think you might have found during use. Please write down what you have to say, your answers will help us improve our products.

Additional Idea

The Additional Idea is the place to write your project ideas about this product, or other usages you've found. Or you can write them on Projects page.

Resources

- Modified MicroBridge Arduino Library
- Seeeduino Demo - Android App
- Seeeduino ADK Eagle Schematic PDF
- Seeeduino ADK Eagle Schematic and Board Files

See Also

- Seeeduino Mega based on ATMega1280
- Arduino Sidekick Basic Kit
- GROVE_System
- Grove - Base Shield

Licensing

External Links

- Android Open Accessory Development Kit Page
- Android Debug Bridge Page
- MicroBridge Page
- Game controllers using USB Host Shield