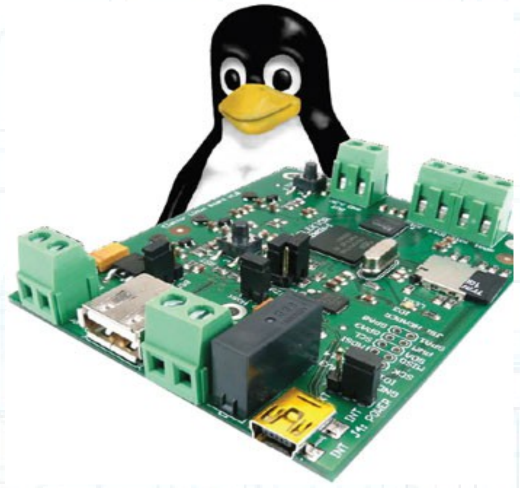


Android Open Accessory Protocol

Turn Your Linux machine as ADK

By
Rajesh Sola,
CDAC ACTS,Pune

OSI Days 2014,
Nimhans Convention
Centre, Bengaluru

Outline

- Need for AOA Protocol/ADK
- Introduction,History
- Available ADK hardware
- Initializing Android device
- Using driver skeleton
- Communication – ADK Side
- Sample Apps
- Developing your own apps
- Communication – Android Side

Need for AOA Protocol/ADK Host

- Peripheral support of typical android gadget
- Adding more peripherals like temperature sensor,CANBus support,SPI,I2C,PWM etc?
- With/wthout rooting of android device
- Scope of Android USB Host capabilities
- USB device capabilities of android gadget
- Alternatives for USB – Bluetooth,WiFi etc.
- Designing rich UI for your hardware control.
- Lets talk about the solution for all...

ADK Host

- In this protocol android gadget is kept in device mode and external hardware will be chosen in host mode
- Any hardware with USB host capabilities and capable of supplying a power of 5V@500mA to the android device can be chosen for this purpose,which is called as Android Accessory Development Kit(ADK) or Accessory Host
- Simple USB communication with two bulk end points

History

- AOA v1.0
  - Supported from Android 3.1(API Level 12) onwards
  - With add on library from Android 2.3.4(API level 10)
  - Announced in Google I/O 2011
  - Demonstrated through Custom shield for Arduino Mega board
- AOA v2.0
  - Supported from Android 4.1(API Level 16) onwards
  - Comes with additional features like audio,hid support and bluetooth connectivity
  - Announced in Google I/O 2012
  - Demonstrated through Arduino Due based custom board

Available Hardware for ADK

- Arduino Mega
- Arduino Due
- AOA kit from Embedded Artists
- FTDI Vinculum II
- Microchip PIC24F ADK
- Sparkfun IOIO
- BeagleBone powered with TI Starterware

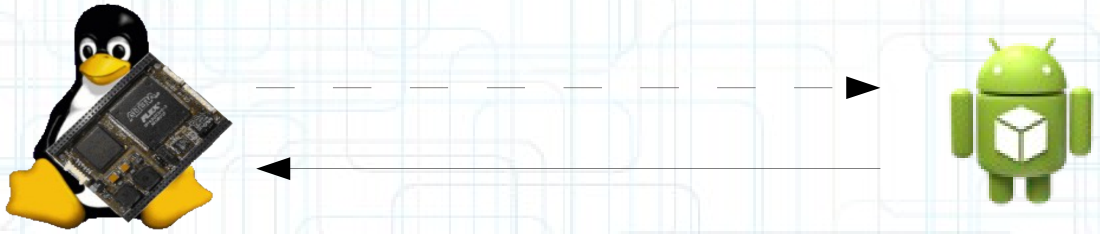
Any Linux machine with USB Host support

Not limited to this hardware,any board with USB Host support can be designed as ADK with suitable USB Host APIs. The only advantage with this listing is availability of some sample code and certain tested apps.

Initialization

Step1:-

- Send Control request 51(0x33) to get the protocol version
- Returns 1 for AOA 1.0, 2 for AOA 2.0

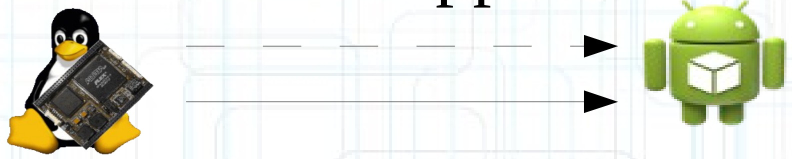


bmRequestType	TYPE_VENDOR| DIR_DEVICE_TO_HOST
bRequest	51
wValue	0
wIndex	0
Data	Address of two byte buffer
wLength	2

Initialization

Step2:-

- Send identity strings through control request 52
- Essentially one should send manufacturer, model, version to communicate ADK with an app
- Rest all are optional



bmRequestType	TYPE_VENDOR|DIR_HOST_TO_DEVICE
bRequest	52
wValue	0
wIndex	String id
Data	Address of string
wLength	Length of the string

String IDs	
Manufacturer	0
Model	1
Description	2
Version	3
Url	4
Serial id	5

Initialization

Step3:-

- Start in accessory mode through control request 53(0x35)
- Launch an app matched by essential identity strings or start without app just for audio/hid support.

bmRequestType	TYPE_VENDOR| DIR_HOST_TO_DEVICE
bRequest	53
wValue	0
wIndex	0
Data	NULL
wLength	0

Audio Support

- Send control request 0x58 for audio support with value=1 for 2 channel 16 bit PCM @ 44100 Kh`z
- If you are requesting for audio support,this must be done before starting in accessory mode
- If manufacturer,model identity strings are not supplied device goes to audio only mode

bmRequestType	**TYPE_VENDOR| DIR_HOST_TO_DEVICE**
bRequest	**53**
wValue	**1**
wIndex	**0**
Data	**NULL**
wLength	**0**

HID Support

- ADK can acts as HID event source or HID proxy to your android device
- The following request codes are used for HID support

Register HID	54
Unregister HID	55
Set HID Report	56
Send HID Event	57

Configuration

- Upon initialization android device switches to accessory mode from MTP/PTP mode and re-enumerated by ADK Host with one of the following vendor id,product id combinations and interface 0 with two bulk end points for communication

0x18D1:0x2D00	**Accessory only**
0x18D1:0x2D01	**Accessory+ADB**
0x18D1:0x2D02	**Audio only**
0x18D1:0x2D03	**Audio + ADB**
0x18D1:0x2D04	**Accessory + Audio**
0x18D1:0x2D05	**Accessory + Audio + ADB**

Using driver skeleton

- Hosted on github.com/rajeshsola/adk-driver-skeleton

  based on Gary Bisson's libusb based linux-adk code

- Load the driver,optionally with module parameters

  my_dev_vid, my_dev_pid

- Initialize using simple shell commands

```
cat /sys/kernel/adk_linux/aoa_init/version
echo “Manu=OSI Days” > /sys/kernel/adk_linux/aoa_init/identity
echo “Model=uisample” > /sys/kernel/adk_linux/aoa_init/identity
echo “version=1.0” > /sys/kernel/adk_linux/aoa_init/identity
echo “accessory” > /sys/kernel/adk_linux/aoa_init/start
```

- Alternatives for initialization

  ioctl code

  module parametet init_on_probe=1

Communication – ADK side

- Simple userspace file operations on linux(ADK) side to talk to the app
- Start the communication

  fd=open("/dev/aoa-skel0",O_RDWR);
- For output peripherals(Android to ADK), eg:- LEDs,Volume

  read(fd,buf,len);

  //process the buffer and control peripherals
- For input peripherals(ADK to Android), eg:- Sensors, RTC

  //fill the buffer with state of peripherals

  write(fd,str,len);
- Stop the communication

  close(fd);
- can go for threaded design for Asynchronous transfer

Sample Apps

- From Google Playstore
  - ADK 2012 from Google
  - Basic Accessory Demo from Microchip
- From other repositories with source code
  - SimpleAccessory Demo
  - Examples for AOA Kit from Embedded Artists
  - ADK Demo from rowboat project for beagle bone
  - TinyAccessory Demo
  - FTDI Android examples

Developing Your own Apps

- intent-filter section and other changes in Manifest file
- Getting the reference of USB service manager
- Getting the reference for connected accessory
- Checking/Obtaining permissions
- Getting references of file streams
- Steps to take care during Activity life cycle
  - OnCreate
  - OnPause
  - OnResume
  - OnDestroy
- A walk through of skeleton code ADKControl class

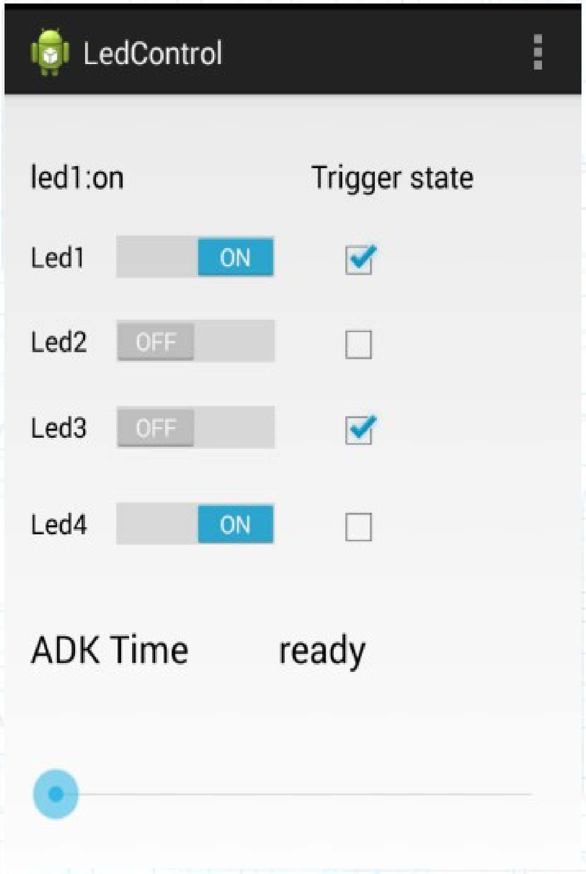
Communication – Android App side

- Add the skeleton class ADKControl to your project
- Create an instance of ADKControl in MainActivity by passing the reference of same,if necessary pass the reference of ADKControl to other activities or fragments or retrieve through getInstance(null)

```
ADKControl adk=ADKControl.getInstance(this)
```

- Use sendCommand method for output activity,in appropriate listeners
- Use readData method for input activity, preferably in a different thread for asynchronous read
- Can obtain references for Input,Output references using methods getInputStream,getOutputStream

An Example – App to ADK



led1:on Trigger state

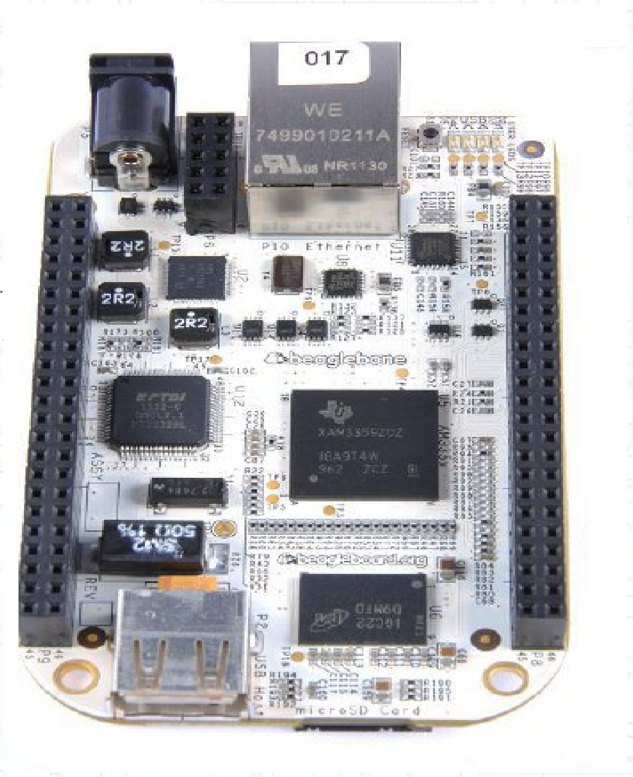
Led1 ON ✓

Led2 OFF

Led3 OFF ✓

Led4 ON

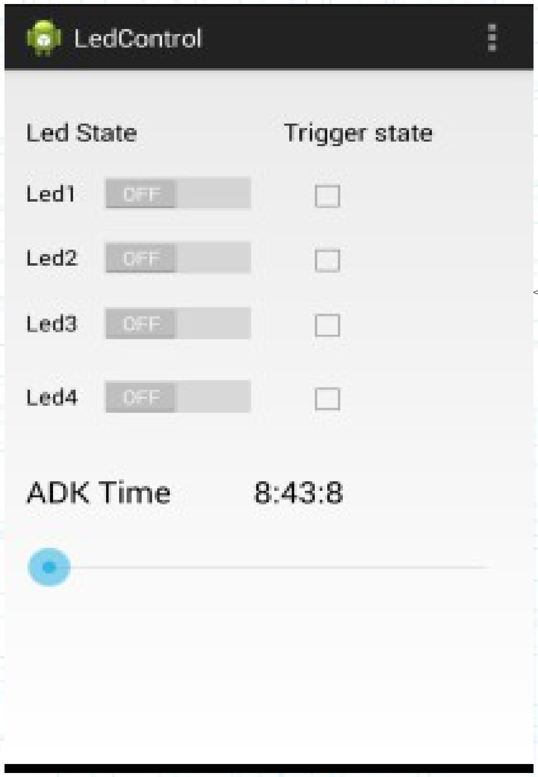
ADK Time ready



Byte buffer

Peripheral Type	Peripheral Id	Peripheral State/Data	Peripheral Data	Peripheral Data	--------------->
0	1	2	3	4	

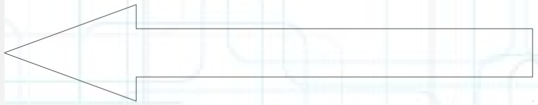
An Example – ADK to App

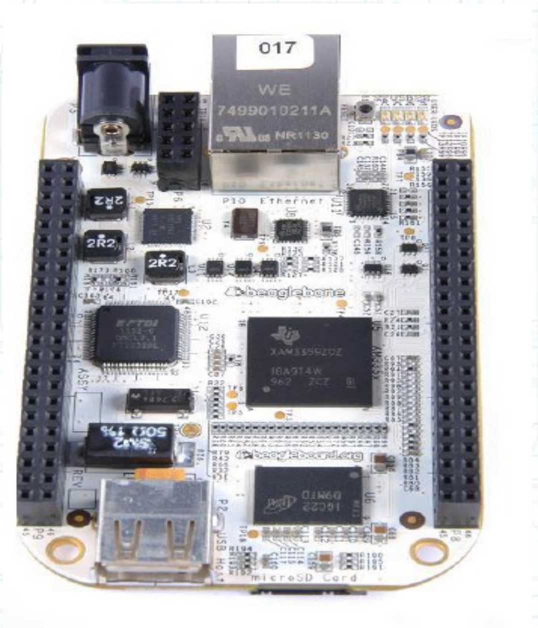


LedControl

Led State		Trigger state
Led1	OFF	☐
Led2	OFF	☐
Led3	OFF	☐
Led4	OFF	☐

ADK Time 8:43:8





Input Type	Data	Data	Data	Data	---------------->

0x46	hours	mins	secs

Thank You

http://github.com/rajeshsola/adk-driver-skeleton

rajeshsola@gmail.com