An Approach to interface CPU with USB OTG controller

Richa Bansal
Computer Science and Engineering
CET-IILM-AHL,
Greater Noida, India
bansalricha14@gmail.com

Akash Goel
Automotive Product Group
ST Microelectronics
Greater Noida, India
akashknmiet@gmail.com

***Abstract*— With the wide application of ARM technology, design an interface between ARM and USB-OTG controller which can be used in mobile communication field, PDA, mobile phone and other mobile devices that can exchange data with USB peripherals directly with or without an interaction of PC. Interfacing requires more of AHB protocol defining Bus interconnection, control signals, address decoding. Primarily this paper is focused on discussing methodology and approach to interface ARM with OTG controller and to understand the communication between OTG controller and a number of devices having dual role capability. Secondly paper also covers Advanced Microcontroller Bus Architecture and USB architecture with software programming of USB in both A (Host) and B (Device) mode. Finally, an ARM based interface with USB controller having software capabilities can be used to communicate with wide range of devices like Mobile, Printer etc without PC interaction.**

***Keywords*— AMBA, ARM, USB-OTG, AHB**

I. INTRODUCTION

Figure 1 describes the ARM with USB OTG interface, USB transceiver with PHY connector. ARM is used to program USBOTG register so ARM Master Interface is connected with the slave interface of the USB OTG.FIFO of the USB controller can be implemented on the RAM. All the USB IN and OUT transaction is through the FIFO.

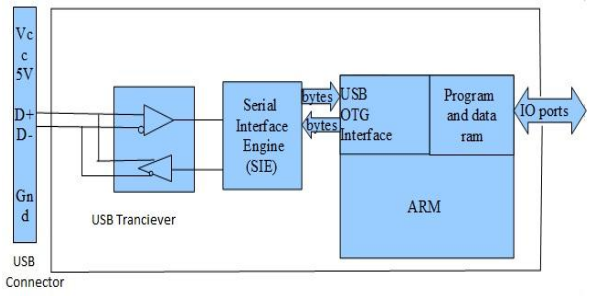

Figure 1. Integrated USB transceiver connects to the USB

IN and OUT direction are always taken in respect of host[3], i.e. if USB controller is sending data to the device it means OUT transaction and if data is going to the host it is IN transaction. Once a device is connected to an OTG device, status on ID pin decides if the connected Device is an Host or an Device. Depending on which kind of device is connected processor will program the USB controller register to configure an OTG enable USB as HOST or DEVICE.

Register Programming of USB controller is done through ARM processor using the AHB bus, Address decoder that find the location of the register to be programmed and Arbiter is used if more peripherals are connected to the bus.

An integrated transceiver is connected with the USB bus pins D+ and D- lines. A serial Interface Engine (SIE) encodes and decodes the serial data and performs error detection, bit stuffing and other signaling checks required by the USB and then transfers data bytes to and from the USB interface.

Besides USB OTG control module, OTG protocol also defines a new interface (named Micro-AB) in OTG hardware system. Here Micro A and MICRO B are fit for MICRO AB interface. When two OTG devices are connected then detection of the device that which is behaving as host and which is behaving as device, it can detect by MICRO A and MICRO B plug. If a device is connected with MICRO A plug then it is a host and if by MICRO B plug then it is a slave. USB Interface, Micro A and Micro B plug also added an “ID” pin. In Micro-A plug, the "ID" pin is connected with ground and in Micro-B plug ID pin is in high resistance.

Advanced Bus Microcontroller Architecture[1]

Three distinct buses are defined within the AMBA specification:

- The Advanced High-performance Bus (AHB)
- The Advanced System Bus (ASB)
- The Advanced Peripheral Bus (APB).

As for high performance, AHB is the most suitable protocol, so in this paper AHB will be targeted to interface USBOTG

II. OTG PROTOCOL

OTG is the development and supplement of USB technology. OTG function is to exchange data between OTG devices under the condition of no-PC. OTG protocol prescribes MICRO-AB and two Communication protocols. The two protocols are HNP and SRP [4]. In OTG protocol, initialization mode of device is decided by hardware and host-slave is switched by software.
USB devices can either run in host mode or in slave mode before OTG protocol occurs. The dual-role OTG device can

run in host mode and slave mode. Two new USB OTG devices are defined in the OTG protocol.

A. *SESSION REQUEST PROTOCOL*

The OTG supplement defines a Session Request Protocol (SRP), which allows a B-device to request the A-device to turn on VBUS and start a session. This protocol allows the A-device[2], which may be battery powered, to conserve power by turning VBUS off when there is no bus activity while still providing a means for the B-device to initiate bus activity.

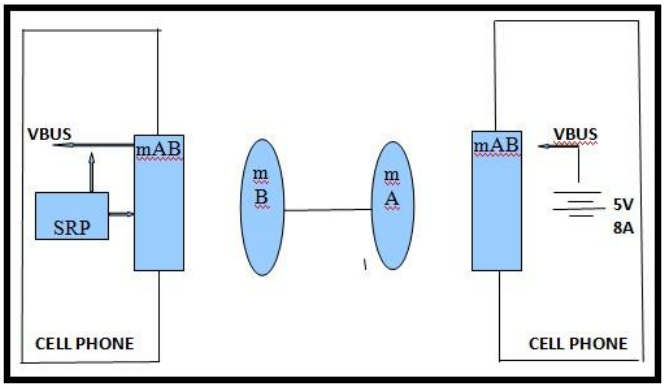

Figure 2. Example of Session Request

Any A-device, including a PC or laptop, is allowed to respond to SRP. Any B-device, including a standard USB peripheral, is allowed to initiate SRP. A dual-role device is required to be able to initiate and respond to SRP. The B-device may not attempt to start a new session until it has determined that the A-device should have detected the end of the previous session. The A-device detects the end of a session by sensing that VBUS has dropped below its session valid threshold. Since the A-device Session Valid threshold may be as low as 0.8 V (VA_SESS_VLDmin), the B-device must insure that VBUS is below this level before requesting a new session. The B-device may ensure that VBUS is below the B-device Session End threshold either by direct measurement of VBUS or by timing the discharge.

B. *HOST NEGOTIATION PROTOCOL*

Since dual-role devices have a Mini-AB receptacle, a dual-role device can default to being either Host or Peripheral, depending up which type of plug (Mini-A plug for Host, Mini-B plug for Peripheral) is inserted. By utilizing the Host Negotiation Protocol (HNP), a dual-role B-device, which is the default Peripheral, may make a request to be Host. The process for this exchange of the role of Host is described in this section.

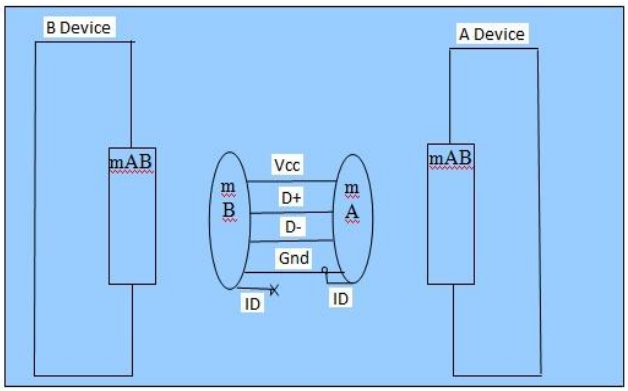

Figure 3. Example of Host Negotiation

This protocol eliminates the need for the consumer to swap the cable connection in order to change the roles of the connected devices. HNP is used to transfer control of a connection from the default Host (A-device) to the default Peripheral (B-device). This is accomplished by having the A-device condition the B-device to be able to take control of the bus, and then having the A-device present an opportunity for the B-device to take control. The B-device is conditioned when the A-device sends a SetFeature (b_hnp_enable) command. After sending this command, the A-device may suspend the bus to signal the B-device that it may now take control of the bus. If the B-device wants to use the bus at that time, it signals a disconnect to the A-device. If the A-device has enabled the B-device to become Host, then the A-device will interpret this disconnect during suspend as a request from the B-device to become Host. The A-device will complete the handoff by turning on its pull-up resistor on D+.

When the B-device has finished using the bus, it starts the process of returning control to the A-device simply by stopping all bus activity and turning on its D+ pull-up resistor. The A-device will detect this lack of activity and turn off its pull-up resistor. When the A-device detects the connection from the B-device, it will resume bus operation as Host.

III. SOFTWARE DESIGN

A. *OTG Software System Structure*

USB OTG host software system includes initialization mode judgment layer (IMJL), OTG driver layer (OTGD) and user application layer (UAL). The OTGD includes host software packet, slave software packet and host-slave switch protocol software packet. Software structure of the system is shown in Fig. 4. The IMJL detects the type of micro plug and accordingly transmit the role to OTGD. After receiving the role switch signal provided by the ISD, theOTGD executes the corresponding software packet (Host software packet or slave software packet) to implement host or slave function. The OTGD also includes role switch protocol software. The UAL receives the user's command and then sends it to the OTGD.

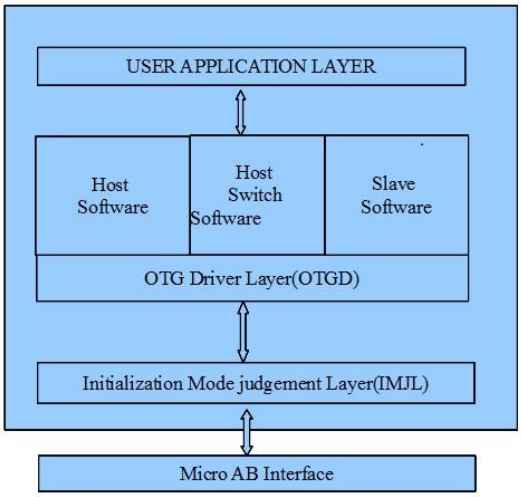

Figure 4.

Finally, data is transmitted on USB bus. The software design in the paper is mainly the selection and implementation of host-slave software packet.

B. HOST/SLAVE SOFTWARE PACKET

OTG device is different from traditional USB device that previously run only in single mode i.e either host or slave mode but in OTG bost host and device can switch the roles whenever needed. Thus, enumeration needs to be completed rightly whatever OTG devices communicate or USB devices communicate. When we ON our computer and plug in a USB device, and magically, device is connected and its driver is loaded. Inside every USB device is a table of 'descriptors' that are the sum total of the device's requirements and capabilities. When we plug into USB, the host goes through a 'sign-on' sequence:

1. The host sends a "Get_Descriptor/Device" request to address zero (devices must respond to address zero when first attached).
2. The device dutifully responds to this request by sending ID data back to the hosttelling what it is.
3. The host sends the device a "Set_Address" request, which gives it a unique addressto distinguish it from the other devices connected to the bus.
4. The host sends more "Get Descriptor" requests, asking more device information. From this, it learns everything else about the device, like how many endpoints thedevice has, its power requirements, what bus bandwidth it requires, and what driver to load.

- Slave Enumeration

In USB communication system, the slave is controlled by host. When slave receives command packet sent by host, a USB interrupt is generated in slave. And then the slave sends device information to host by received command request to do device enumeration. Fig. 5 is the slave enumeration flow chart.

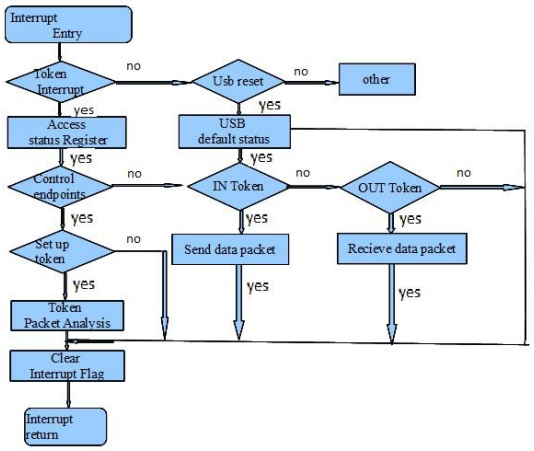

Figure 5. Slave Enumeration

- Host Enumeration

USB host is the control core of communication system. It sends command in order to get the device information from slave. Fig. 6 is the host enumeration flow chart.

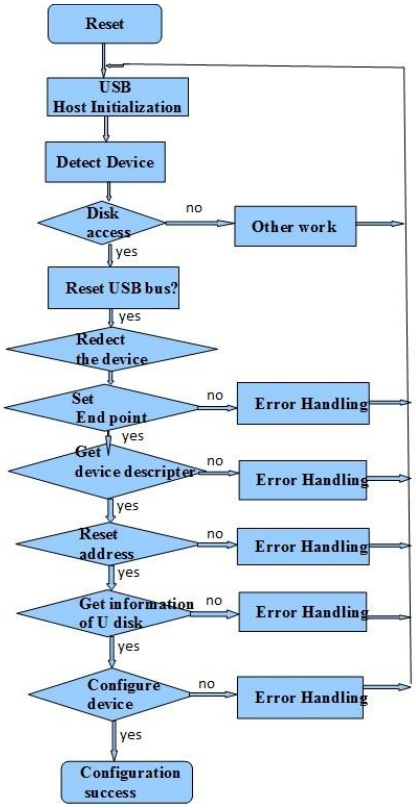

Figure 6. Example of Host Enumeration

C. Initialization mode judgment

An OTG device needs to be appointed to the host and the other is appointed to slave in OTG device communication system. Fig. 7 is initialization mode judgment flow chart.

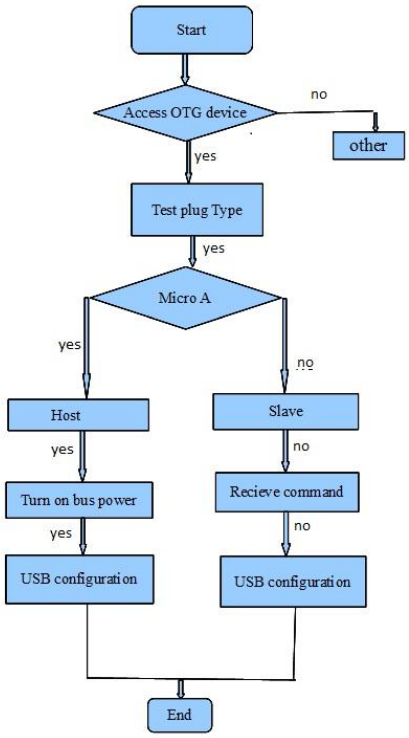

Figure 7.

D. OTG Host/Slave switch protocol software

In software system of an OTG device, the OTGD includes host-slave switch protocol software. An advantage of the OTG device can switch a host to a slave or a slave to a host by switch protocol software. The protocol software is compiled by the HNP. Host-slave switch process is shown in Fig. 8. First, a host sends "Set Feature"[5] command to a slave to judge whether the slave supports HNP. If it supports HNP,the host suspends the bus to inform the slave that it may now take control of the bus. If B-device wants to use the bus at that time, it turns on its pull-down resistor on D+ to disconnect to host. If the host has enabled the slave to become a host, the A-device will interpret this disconnect during suspend as a request from the B-device to become a host. The A-device will complete the handoff by turning on its pull-up resistor on D+. The USB configuration process is secondly completed. Finally, host-slave switch is achieved.

E. User applications software

Because USB OTG device can run in host or slave mode, the user application software also is consisted of the host application software and slave application software. The user application software is designed according to actual needs.

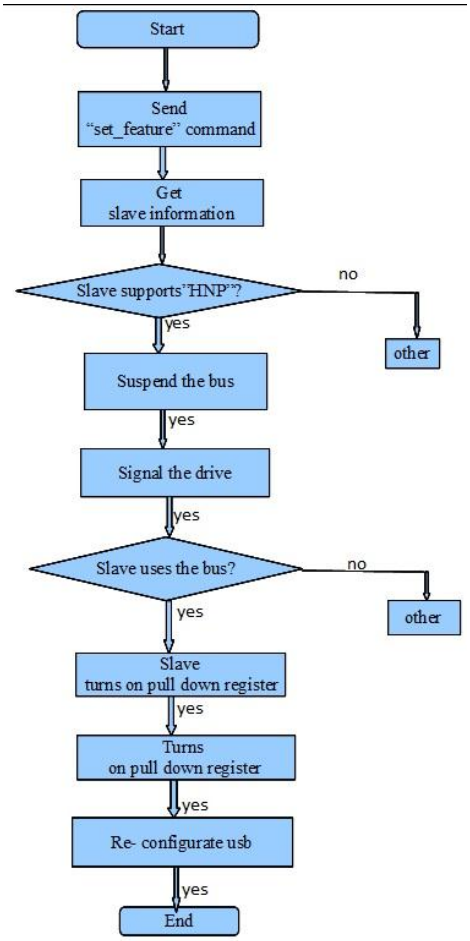

Figure 8. Host/Slave Switch

Its main function is that UAL receives and deals with the command from so UAL and control the system to execute the command. In the design the software design focus on initialization mode judgment software and role switch protocol software.

IV. DESIGN CONSIDERATIONS AND HARDWARE IMPLEMENTATION

Complex digital systems require high-performance external interfaces. Host Controller IP cores interconnects the on-chip AMBA bus and external USB transceiver (PHY).

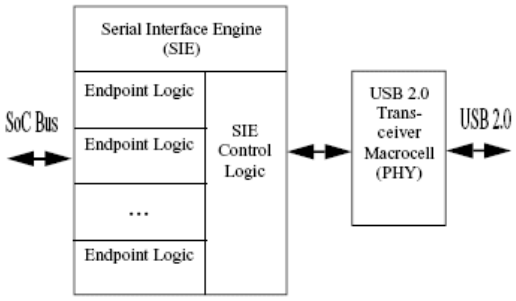

Figure 9. Peripheral USB Controller and PHY

The USB OTG Controller core provides a link between the AMBA on-chip bus and Universal Serial Bus(USB). The host controller supports High-, Full- and Low-Speed USB traffic. The architecture is centered on the AMBA Advanced High-speed Bus (AHB), to which the USB Host Controller IP core and other high-bandwidth units are connected.

The transceiver (PHY) is connected to USB controller using the UTMI (USB transceiver macrocell interface) or ULPI (UTMI+ low pin interface) interfaces. And interact with external devices connected on USB physical interface using D+/ D- lines [6].

Design can be implemented for the following modes:

1) *Slave mode:*
Slave mode is typically selected when area is a concern and you have enough CPU bandwidth to execute the USB driver and to move the data between memory and DWC_otg.

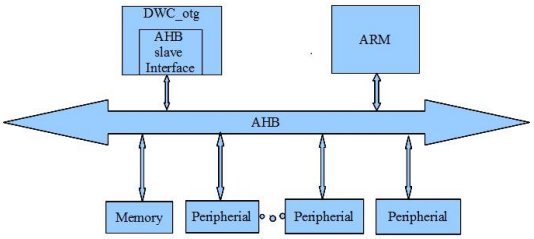

Figure 10. Slave Mode configuration

2) *DMA mode:*
DMA mode is typically selected when the CPU bandwidth to process the USB transfer is limited and you would like an DMA controller to take care of the data transfers between the memory and OTG.

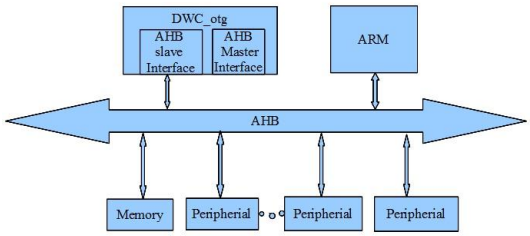

Figure 11. DMA Mode configuration

As shown in figure 11, an AHB master interface is required to interface DMA with OTG.

V. CONCLUSION

This work represents the frame work required to integrate an USB controller in System on Chip. This paper outlined some of the concerns that the system designer may have while designing the interface between an otg and ARM processor.

VI. FUTURE SCOPE

Currently many of the devices need PC only because they can act only in device mode but with OTG a device can directly send command to other device as an example a camera can directly send a command to the printer to print with no PC.

REFERENCES

[1] AMBA specification, Technical Manual, Rev 2.0, May 13 ,1999

[2] MA Wei and SHAO Beibei, "On-To-Go Opens USB's New Application," Electronic & Computer Design World. vol.6, 2002, pp.68-69.

[3] Universal Serial Bus Specification Revision 3.0 : 3.1. 12 November 2008. p. 41 (3-1)

[4] MO Yimin and LIU Qing, "The Technology of USB OTG and Its Application," Mechanical & Electrical Engineering Technology, vol. 35, no. 11, 2006, pp. 95-97.

[5] MA Wei and SHAO Beibei, "On-To-Go Opens USB's New Application," Electronic & Computer Design World. vol.6, 2002, pp.68-69.

[6] Don Andserson, " Universal Serial Bus System Architecture," PC System Archicture Series,Revision 3.0,January '1997