
USB OTG Like Connector Implementation

AT91SAM ARM-based Microcontrollers

1. Scope

The USB OTG (On-The-Go) is an interface to communicate with portable USB devices.

In the SAM9x5 Series (SAM9G15/G25/G35/X25/X35) and SAMA5D3x products, the USB host and device share one transceiver. The shared USB transceiver can be controlled either by the USB Host Port Controller or by the USB Device Controller. This configuration allows to emulate a "USB OTG like" connector. The purpose of this document is to help the developers to implement this kind of USB OTG.

2. References

- [1] Universal Serial Bus Specification, Rev 2.0, April 27, 2000
- [2] On-The-Go Supplement to the USB 2.0 Specification, Rev 1.0a, June 24, 2003
- [3] http://www.at91.com/linux4sam/bin/view/Linux4SAM/GettingStarted_9x5

3. USB OTG

3.1 USB OTG Purpose

The USB OTG is a USB controller that can be auto-configured to host or device according to ID pin status. It means that one mini footprint receptacle can be configured to host or device.

3.2 USB OTG Features

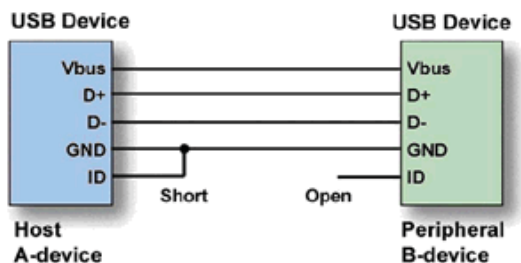
An On-The-Go device must include the following features and characteristics:

- A limited Host capability
- Full-speed operation as a peripheral (high-speed optional)
- Full-speed support as a host (low-speed and high-speed optional)
- Targeted Peripheral List
- Session Request Protocol
- Host Negotiation Protocol
- One, and only one Mini-AB receptacle
- Minimum 8 mA output on V_{BUS}
- Means for communicating messages to the user

3.3 USB OTG Plug Connection

The ID pin on a Mini-A plug shall be connected to the GND pin; the ID pin on a Mini-B plug is not connected (open drain) or is connected to ground by a resistance greater than 100 K.

Figure 3-1. USB OTG Plug Connection

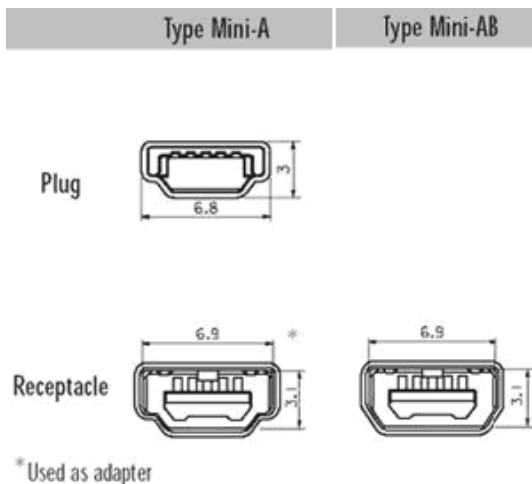


3.4 USB OTG Connectors

The OTG supplement defines the following additional connectors:

- Mini-A plug and receptacle
- Mini-AB receptacle

Figure 3-2. USB OTG Connectors

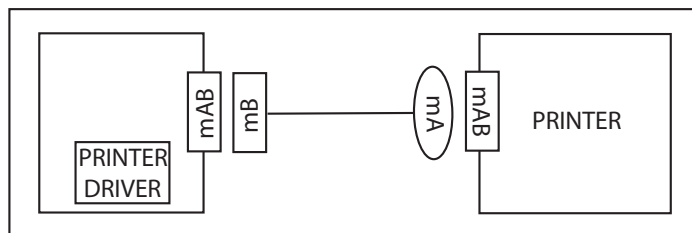


3.5 USB OTG Host Negotiation Protocol

The Host Negotiation Protocol (HNP) allows the host function to be transferred between two directly connected OTG devices and eliminates the need for a user to switch the cable connections in order to allow a change in control of communications between the devices. HNP will typically be initiated in response to an input from the user or an application on the OTG B-device. HNP may only be implemented through the Micro-AB receptacle on a device. The A-device is always responsible for powering the USB interface regardless of whether it is acting in host or peripheral role.

To understand the need for the HNP and host/peripheral role reversal, the example in [Figure 3-3](#) shows two OTG devices, a PDA and a printer. The PDA has a printer driver inside. The two devices are connected with the new OTG cable as shown, making the printer the default host (A-Device) and the PDA the default peripheral (B-Device). But this setup is backwards. The PDA, which has the printer driver, needs to act as USB host to the printer, which contains no driver. Rather than bothering the user to reverse the cable, HNP allows the devices' roles to reverse automatically and silently.

Figure 3-3. USB OTG Connection

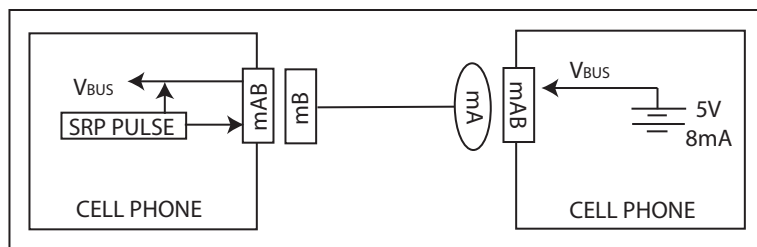


3.6 USB OTG Session Request Protocol

In order to conserve power, the OTG supplement allows an A-device to leave V_{BUS} turned off when the bus is not being used. If the B-device wants to use the bus when V_{BUS} is turned off, then it requires the A-device to supply power on V_{BUS} . For this reason, the OTG supplement defines the Session Request Protocol (SRP).

An OTG session is defined as the time that the A-Device is furnishing V_{BUS} power (Note: the A-Device always supplies V_{BUS} power, even if it is functioning as a peripheral due to HNP). The A-Device can end a session by turning off V_{BUS} to conserve power, a very important requirement in a battery-powered device such as a cell phone.

Figure 3-4. USB OTG Session Request Protocol (SRP)



3.7 USB OTG Electrical Characteristics

Table 3-1. USB OTG Electrical Characteristics

Parameter	Conditions	Min	Max	Units
A-device Output Voltage	$0 \leq V_{BUS_A_VBUS_OUT} < 100 \text{ mA}$	4.4	5.25	V
V_{BUS} Rise Time	$C_{LOAD} = 10 \mu\text{F}$ $I_{VBUS} = \text{rated load}$ $0V < V_{BUS} < 4.4V$		100	ms
A-device Leakage Voltage			0.2	V
A-device Output Current	$4.4V \leq V_{BUS} \leq 5.25V$	8		mA
B-device (SRP capable) to OTG Device Output Voltage		2.1	5.25	V
B-device (SRP capable) to Host Output Voltage			2.0	V
B-device (On-The-Go) Unconfigured Average Current	$0V \leq V_{BUS} \leq 5.25V$ $T_{AVG} = 1 \text{ ms}$		150	μA
B-device (SRP capable, peripheral-only) Unconfigured Average Current	$0V \leq V_{BUS} \leq 5.25V$ $T_{AVG} = 1 \text{ ms}$		8	mA
OTG Device Leakage			0.342	V
Resistance to Ground on Mini-B Plug		100		$k\Omega$
Resistance to Ground on Mini-A Plug			10	Ω

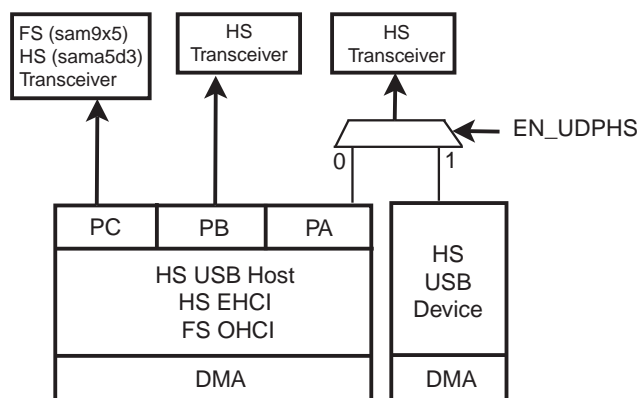
4. USB in SAM9x5 Series and SAMA5D3x Products

4.1 USB Selection

For SAM9x5 series and SAMA5D3x products, there are possible use cases as below:

- Three Hosts:
 - Two HS Hosts and one FS Host (SAM9x5 series)
 - Three HS Hosts (SAMA5D3x)
- Two Hosts and one Device:
 - One HS Host, one FS Host (SAM9x5 series)
 - Two FS Hosts (SAMA5D3x)
 - One HS Device

Figure 4-1. USB Selection



4.2 USB Characteristics

- Compliant with Enhanced HCI Rev 1.0 Specification
 - Compliant with USB V2.0 High-speed
 - Supports High-speed 480 Mbps
- Compliant with Open HCI Rev 1.0 Specification
 - Compliant with USB V2.0 Full-speed and Low-speed Specification
 - Supports both Low-speed 1.5 Mbps and Full-speed 12 Mbps USB devices
- Root Hub Integrated with X Downstream USB Ports
- Shared Embedded USB Transceivers
- Supports Power Management
- 2 Hosts (A and B) High Speed (EHCI) and Full Speed (OHCI), 1 Host (C) Full Speed only (OHCI) (SAM9x5 series)
- 3 Hosts (A, B and C) High Speed (EHCI) and Full Speed (OHCI) (SAMA5D3x)

4.3 USB Dedicated Function and DEMUX

Both Port C and Port B have dedicated functions. For SAM9x5 series, Port C supports Full speed OHCI (Open Host Controller Interface) and Port B supports HS USB Host. For SAMA5D3x, Port C and Port B both support HS USB Host. HS USB Host port A and HS USB Device share one transceiver. It is controlled by EN_UDPHS bit located in the USB Device user interface.

Set to 0 → Port A USB Host selected

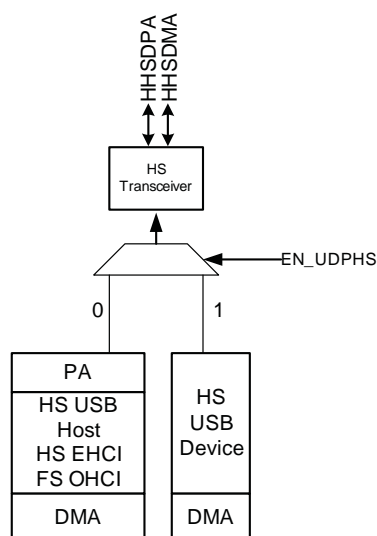
Set to 1 → USB Device selected

5. USB OTG Like Connector Implementation

5.1 Manual Operation

HHSDPA and HHSDMA bonding out pads are connected with receptacle, such as Mini-A receptacle. One receptacle can be configured to host or device according to the customer requirements, just like USB OTG.

Figure 5-1. USB Selection for USB OTG Like Connector Implementation



- Configured to Host

When USB gadget is not enabled in the code, set `EN_UDPHS` bit to 0 (this register is located in the `UDPHS_CTRL` control register). The USB host is selected and programmed; use the port as host, which is able to detect USB devices supported by the system, such as U-disk, Key board, Mouse, etc. (when connected to USB host, no functionalities).

- Configured to Device

When USB gadget is enabled in the code, set `EN_UDPHS` bit to 1 (this register is located in the `UDPHS_CTRL` control register). The USB device is selected and programmed; use the port as slave. When connected to USB host, it runs the predefined functionalities (when connected to USB slave device, no functionalities).

5.2 Auto Detection

Two GPIOs are used: GPIO1 is for ID status detection and GPIO2 is for V_{BUS} output control.

By default, the USB is configured to device and the GPIO2 disables the EK board as the USB V_{BUS} power supply, waiting for the external USB host power supply. When an external USB peripheral plug is detected, check the GPIO1 ID status. If `GPIO1=1`, keep the same configuration. After getting the power supply from external host of V_{BUS} , the device pulls up the bus line and handshakes with the host.

When GPIO1 detects ID status, if `GPIO1=0`, which means ID Pin is connected to ground, the GPIO generates an interrupt and then the CPU configures the USB as the host. Then GPIO2 enables V_{BUS} to power supply the external USB peripheral. The host pulls down the bus line, waits for the bus line pull-up signal from external USB peripheral, and then handshakes with the external peripheral.

5.3 USB OTG Like Connector Implementation Example Based on SAM9x5-EK

The implementation takes SAM9x5 series as an example. The case is similar for SAMA5D3x.

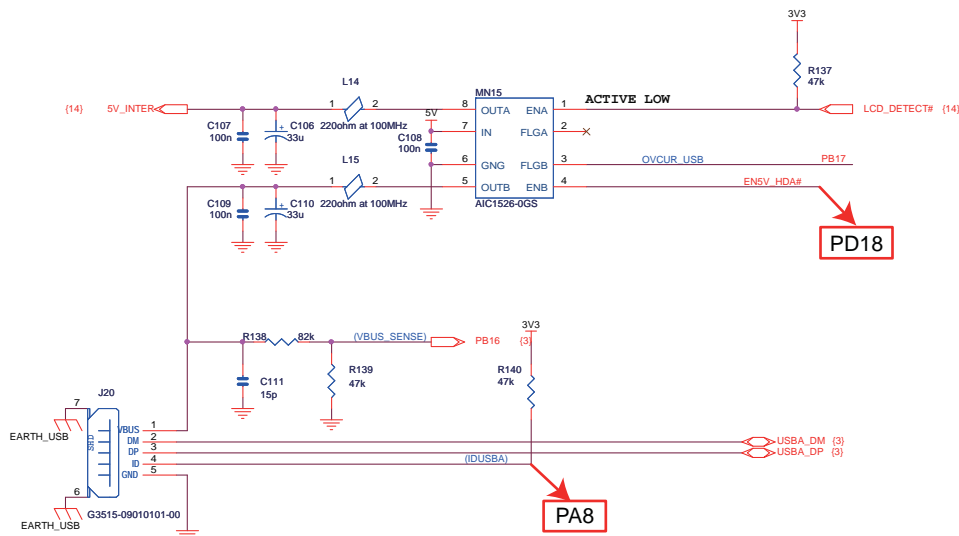
5.3.1 Hardware Implementation

Make some modifications on SAM9x5-EK board to realize the hardware implementation. Use two GPIOs (defined by users) for ID status detection and V_{BUS} output control respectively.

SAM9X35-EK is used in this example. On SAM9X35-EK board, add a sky-wire from Pin 4 (ID pin) of USB-A receptacle (J20) to PA8 as GPIO1 for ID status detection, and use PD18 as GPIO2 for V_{BUS} output control.

The schematic is shown as below:

Figure 5-2. Schematic for USB OTG Like Connector Implementation



5.3.2 Software Implementation

Make some modifications on the released Linux kernel source code for SAM9x5-EK to realize the software implementation.

5.3.2.1 Software Environment

- Linux OS
- SAM-BA 2.11 or later version
- Serial terminal emulator (HyperTerminal, minicom, Tera Term, etc)

5.3.2.2 Software Implementation

1. Get the Linux kernel source code
2. Modify the Linux kernel source code
 - Automatically modify the Linux kernel source code by applying a patch

A USB OTG like connector patch targeted at this example is provided. Find the *sam9x5ek_usb_otg_like_connector.patch* in the attachment of this app note.

```
<get> sam9x5ek_usb_otg_like_connector.patch
```

Note: Copy this file into linux-2.6.39 directory.

```
patch -p1 < sam9x5ek_usb_otg_like_connector.patch
```

- Manually modify the Linux kernel source code

For details on the modification of Linux kernel source code, please refer to “[Example Codes for Linux Kernel Modification](#)”.

3. Configure and build the new Linux kernel

For the detailed execution of the above steps, please refer to:

http://www.at91.com/linux4sam/bin/view/Linux4SAM/SAM9x5Page#Linux4SAM_AT91SAM9x5_Experimenta

5.3.2.3 Example Codes for Linux Kernel Modification

The code modification mainly occurs in three files: `atmel_usba_udc.c`, `atmel_usba_udc.h`, `ohci-at91.c`.

- Configure USB to Host/Device by identifying the status change of ID pin and VBUS pin

Location: `drivers/usb/gadget/atmel_usba_udc.h`

```
/* Add the definition of udc->id_prev as the previous ID pin status for
Host/Device status change identification */
struct usba_udc {
    ...
    u16 test_mode;
    int vbus_prev;
    int id_prev;
    ...
}
```

Location: `drivers/usb/gadget/atmel_usba_udc.c`

Target Function: `static irqreturn_t usba_vbus_irq(int irq, void *devid)`

```
/* If ID pin is pulled down, configure USB to Host */
if (!gpio_get_value(AT91_PIN_PA8)) {
    udc->id_prev = 0;
    usba_writel(udc, CTRL, USBA_DISABLE_MASK);
    return IRQ_HANDLED;
}

if (udc->id_prev != gpio_get_value(AT91_PIN_PA8)) {
    udc->id_prev = 1;
    return IRQ_HANDLED;
}
```

Location: `drivers/usb/gadget/atmel_usba_udc.c`

Target Function: `int usb_gadget_probe_driver(struct usb_gadget_driver *driver,`

```
    int (*bind)(struct usb_gadget *))
/* Only when VBUS is powered on and ID pin is float, configure USB to
Device */
if (vbus_is_present(udc) && udc->vbus_prev == 0 &&
    gpio_get_value(AT91_PIN_PA8)) {
    toggle_bias(1);
    usba_writel(udc, CTRL, USBA_ENABLE_MASK);
    usba_writel(udc, INT_ENB, USBA_END_OF_RESET);
}
```


- Control V_{BUS} power for USB Host/Device

Location: drivers/usb/host/ohci-at91.c

Target Function: static irqreturn_t ohci_at91_otg_irq(int irq, void *data)

```
/* debounce */
mdelay(10);

if (gpio_get_value(AT91_PIN_PA8)) {
/* If ID pin is float, power off  $V_{BUS}$  */
at91_set_gpio_output(AT91_PIN_PD18, 1);
} else {
/* If ID pin is pulled down, power on  $V_{BUS}$  */
at91_set_gpio_output(AT91_PIN_PD18, 0);
}

return IRQ_HANDLED;
}
```

- Register an interrupt for ID pin status

Location: drivers/usb/host/ohci-at91.c

Target Function: static int ohci_hcd_at91_drv_probe(struct platform_device *pdev)

```
/* Register an interrupt to identify the ID pin status */
int ret;

ret = request_irq(AT91_PIN_PA8, ohci_at91_otg_irq, 0, "otg_irq", pdev);
if (ret) {
    dev_err(&pdev->dev, "request irq failed\n");
}

if (!gpio_get_value(AT91_PIN_PA8))
    at91_set_gpio_output(AT91_PIN_PD18, 0);
```

5.3.3 USB OTG Like Connector Verification

After the modified Hardware and Software are both ready for use, verify the USB OTG like connector.

1. Download the modified Linux Binaries into SAM9X35-EK board
2. Open a Terminal for message printout
3. Verify the USB OTG like connector
 - Connect a Host (PC, etc.) with the micro USB receptacle on EK board:

The EK board will be detected on the Host as a device.

- Connect a Device (U Disk, portable devices, etc) with the micro USB receptacle on EK board:

The corresponding information will be displayed on the Terminal when plugging/unplugging the Device.

The USB OTG like connector is successfully implemented on SAM9X35-EK board.

6. Differences from USB OTG

On Hardware side, the Mini receptacle can be configured to host or device, just like USB OTG, and with the auto detect solution, two GPIOs can be configured to support the features. GPIO1 detects ID pin status and automatically generates interrupt of the status; GPIO2 handles V_{BUS} output control.

However, the solution is still not the same with the real USB OTG that doesn't need external detection circuit. The differences between them are as below:

- V_{BUS} output voltage range:
 - USB 2.0: 4.75V to 5.25V
 - USB OTG: 4.4V to 5.25V
- V_{BUS} output current range:
 - USB 2.0: 100 mA to 500 mA
 - USB OTG: from 8 mA
- HNP, SRP are not supported on USB in SAM9x5 series and SAMA5D3x products.

7. Revision History

Doc. Rev.	Comments	Change Request Ref.
11201A	First issue.	



Enabling Unlimited Possibilities®

Atmel Corporation

1600 Technology Drive
San Jose, CA 95110
USA

Tel: (+1) (408) 441-0311

Fax: (+1) (408) 487-2600

www.atmel.com

Atmel Asia Limited

Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG

Tel: (+852) 2245-6100

Fax: (+852) 2722-1369

Atmel Munich GmbH

Business Campus
Parking 4
D-85748 Garching b. Munich
GERMANY

Tel: (+49) 89-31970-0

Fax: (+49) 89-3194621

Atmel Japan G.K.

16F Shin-Osaki Kangyo Bldg
1-6-4 Osaki, Shinagawa-ku
Tokyo 141-0032
JAPAN

Tel: (+81) (3) 6417-0300

Fax: (+81) (3) 6417-0370

© 2013 Atmel Corporation. All rights reserved. / Rev.: 11201A--ATARM--30-Jan-13



Atmel®, Atmel logo and combinations thereof, Enabling Unlimited Possibilities®, and others are registered trademarks or trademarks of Atmel Corporation or its subsidiaries. ARM®, ARM®Powered logo and others are registered trademarks or trademarks of ARM Ltd. Other terms and product names may be trademarks of others.

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN THE ATMEL TERMS AND CONDITIONS OF SALES LOCATED ON THE ATMEL WEBSITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS AND PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and products descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.