Multifunction Composite Gadget

Overview

The Multifunction Composite Gadget (or g_multi) is a composite gadget that makes extensive use of the composite framework to provide a... multifunction gadget.

In it's standard configuration it provides a single USB configuration with RNDIS[1] (that is Ethernet), USB CDC[2] ACM (that is serial) and USB Mass Storage functions.

A CDC ECM (Ethernet) function may be turned on via a Kconfig option and RNDIS can be turned off. If they are both enabled the gadget will have two configurations -- one with RNDIS and another with CDC ECM[3].

Please note that if you use non-standard configuration (that is enable CDC ECM) you may need to change vendor and/or product ID.

Host drivers

To make use of the gadget one needs to make it work on host side -- without that there's no hope of achieving anything with the gadget. As one might expect, things one need to do very from system to system.

Linux host drivers

Since the gadget uses standard composite framework and appears as such to Linux host it does not need any additional drivers on Linux host side. All the functions are handled by respective drivers developed for them.

This is also true for two configuration set-up with RNDIS configuration being the first one. Linux host will use the second configuration with CDC ECM which should work better under Linux.

Windows host drivers

For the gadget to work under Windows two conditions have to be met:

Detecting as composite gadget

First of all, Windows need to detect the gadget as an USB composite gadget which on its own have some conditions[4]. If they are met, Windows lets USB Generic Parent Driver[5] handle the device which then tries to match drivers for each individual interface (sort of, don't get into too many details).

The good news is: you do not have to worry about most of the conditions!

The only thing to worry is that the gadget has to have a single configuration so a dual RNDIS and CDC ECM gadget won't work unless you create a proper INF -- and of course, if you do submit it!

Installing drivers for each function

The other, trickier thing is making Windows install drivers for each individual function.

For mass storage it is trivial since Windows detect it's an interface implementing USB Mass Storage class and selects appropriate driver.

Things are harder with RDNIS and CDC ACM.

RNDIS

To make Windows select RNDIS drivers for the first function in the gadget, one needs to use the [[file:linux.inf]] file provided with this document. It "attaches" Window's RNDIS driver to the first interface of the gadget.

Please note, that while testing we encountered some issues[6] when RNDIS was not the first interface. You do not need to worry abut it unless you are trying to develop your own gadget in which case watch out for this bug.

CDC ACM

Similarly, [[file:linux-cdc-acm.inf]] is provided for CDC ACM.

Customising the gadget

If you intend to hack the g_multi gadget be advised that rearranging functions will obviously change interface numbers for each of the functionality. As an effect provided INFs won't work since they have interface numbers hard-coded in them (it's not hard to change those though[7]).

This also means, that after experimenting with g_multi and changing provided functions one should change gadget's vendor and/or product ID so there will be no collision with other customised gadgets or the original gadget.

Failing to comply may cause brain damage after wondering for hours why things don't work as intended before realising Windows have cached some drivers information (changing USB port may sometimes help plus you might try using USBDeview[8] to remove the phantom device).

INF testing

Provided INF files have been tested on Windows XP SP3, Windows Vista and Windows 7, all 32-bit versions. It should work on 64-bit versions as well. It most likely won't work on Windows prior to Windows XP SP2.

Other systems

At this moment, drivers for any other systems have not been tested. Knowing how MacOS is based on BSD and BSD is an Open Source it is believed that it should (read: "I have no idea whether it will") work out-of-the-box.

For more exotic systems I have even less to say...

Any testing and drivers *are welcome*!

Authors

This document has been written by Michal Nazarewicz ([[mailto:mina86@mina86.com]]). INF files have been hacked with support of Marek Szyprowski ([[mailto:m.szyprowski@samsung.com]]) and Xiaofan Chen ([[mailto:xiaofanc@gmail.com]]) basing on the MS RNDIS template[9], Microchip's CDC ACM INF file and David Brownell's ([[mailto:dbrownell@users.sourceforge.net]]) original INF files.

Footnotes

[1] Remote Network Driver Interface Specification, [[https://msdn.microsoft.com/en-us/library/ee484414.aspx]].

[2] Communications Device Class Abstract Control Model, spec for this and other USB classes can be found at [[http://www.usb.org/developers/devclass_docs/]].

[3] CDC Ethernet Control Model.

[4] [[https://msdn.microsoft.com/en-us/library/ff537109(v=VS.85).aspx]]

[5] [[https://msdn.microsoft.com/en-us/library/ff539234(v=VS.85).aspx]]

[6] To put it in some other nice words, Windows failed to respond to any user input.

[7] You may find [[http://www.cygnal.org/ubb/Forum9/HTML/001050.html]] useful.

[8] https://www.nirsoft.net/utils/usb_devices_view.html

[9] [[https://msdn.microsoft.com/en-us/library/ff570620.aspx]]