2 December 1998

Application Notes

USB Keyboard mouse using PDIUSBD11

Application Notes: USB Keyboard Mouse using PDIUSBD11

Disclaimers:

Life Support – These products are not designed for use in life support appliances, devices or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips Semiconductors customers using or selling thes products for use in such applications do so at their own risk and agree to fully indemnify Philips Semiconductors for any damages resulting from such application.

Right to make change – Philips Semiconductors reserves the right to make changes, without notice, in the products, including circuits, standard cells, and/or software, described or contained herein in order to improve design and/or performance. Philips Semiconductors assumes no responsibility or liability for the use of any of these products, conveys no licence or title under any patent, copyright, or mask work right to these products and makes no representations or warranties that these products are free from patent, copyright or mask work right infringement, unless otherwise specified.

Application information – Applications that are described herein for any of these products are for illustrative purposes only. Philips Semiconductors make no representation or warranty that such applications will be suitable for the specified use without further test or modification

LICENSES

Purchase of Philips I^2C components

Purchase of Philips I^2C components conveys a license under the Philips I^2C patent to use the components in the I^2C system provided the system conforms to the I^2C specification defined by Philips. This specification can be ordered using the code 9398 393 40011.

Application Notes: USB Keyboard Mouse using PDIUSBD11

Introduction

The PDIUSBD11 (D11) is a high speed USB interface device. It can be interfaced to a micro-controller using a minimum of 3 I/O lines. These are a pair of I^2C lines and an interrupt line.

The D11 can be configured in two modes:

Mode 1 - as a single function.
Mode 2 - as three embedded functions.

Mode 1: Single Function
There are a total of 4 endpoints for a single function configuration. Endpoint 0 (the default Control endpoint) and endpoint 1-3 are generic. They can be used as either interrupt, bulk or control endpoints. Each endpoint has a buffer size of 8 bytes.

Mode 2: 3 Embedded Devices
In the 3 embedded function configuration, each of the 3 functions consists of 2 endpoints: a default Control endpoint 0, and a generic endpoint 1.

The D11 can be used as a HID device (although it is not restricted to just HID use). These application notes show how a keyboard with a PS2 mouse can be used.

HARDWARE DESCRIPTION

The USB Keyboard mouse makes use of Philips D11 and 8051 Micro-Controller (MCU). The keyboard matrix implements 16 output and 8 input MCU lines, as follows:

- I^2C communicates with H11A, using 3 I/O MCU lines.
- PS/2 communicates with the PC 2 I/O MCU lines.
- Num Lock, Caps Lock and Scroll Lock LEDs use 3 I/O MCU lines.
- PS/2 mouse uses 2 I/O MCU lines to generate PS/CLK and PS/DATA.

Firmware description

Files description:

1. **Kbms1_7.asm** is the source file, containing all the routines.
2. **transctn.typ** contains constant values for various transaction codes, between D11 and the Host.
3. **H11.cmd**. contains the I2C commands of the D11A (this file is a subset of H11/H11A commands, and is functionally equivalent).

Application Notes: USB Keyboard Mouse using PDIUSBD11

USB Class description

The firmware makes use of an HID-compliant composite device. This is a USB device with a single configuration but with two different interfaces. Each interface defines it's own interrupt endpoint and report descriptor.

Interface 0 defines the USB keyboard with endpoint 1 used as the interrupt endpoint for key-pressed data.

Interface 1 links with the USB mouse, using endpoint 2 as the Interrupt endpoint for tracking the movements of the mouse.

Flow Chart

The main loop polls the USB interrupt from the D11; it also runs a routine to detect key-press events.

a) Main loop

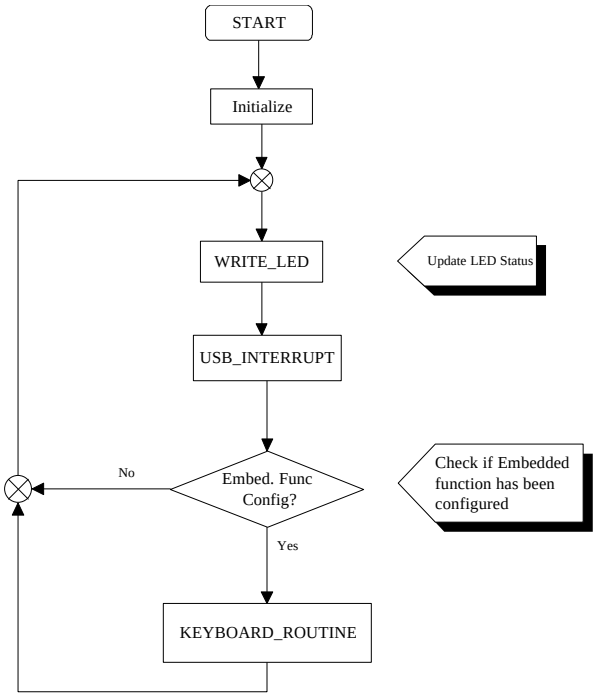


Fig. 1: MAIN ROUTINE

This routine initializes D11 and loops for D11 Interrupt pin to go LOW. The routine branches to USB_INTERRUPT on a HIGH-to-LOW transition on the Interrupt pin.

Application Notes: USB Keyboard Mouse using PDIUSBD11

b) USB Interrupt

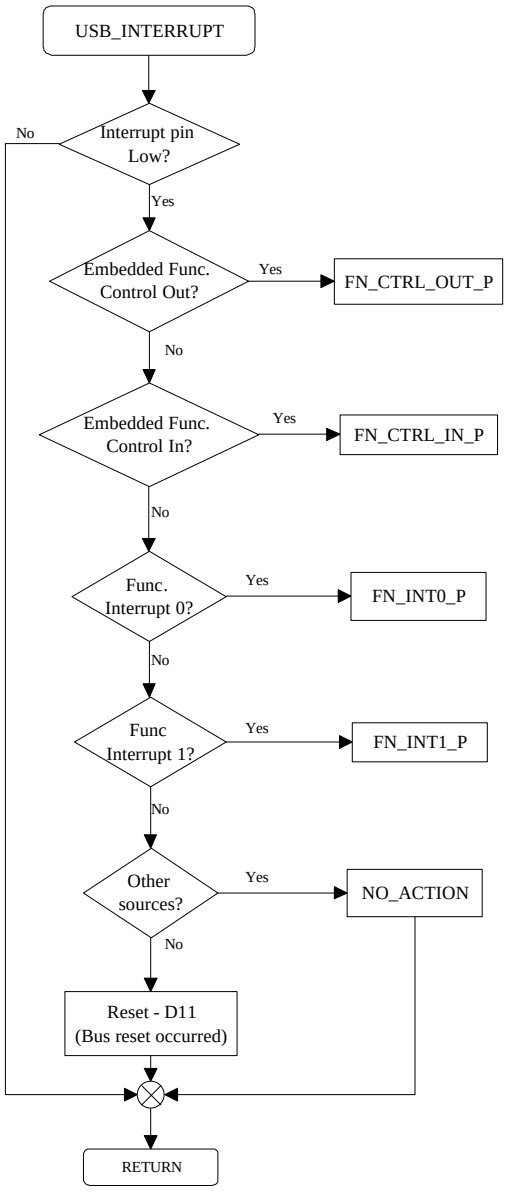


FIG. 2: USB INTERRUPT

This routine checks for the source causing the Interrupt pin to go LOW and branches to the respective routines. The source of the Interrupt can be from:
Embedded_Control_Output_Endpoint
Embedded_Control_Input_Endpoint
Embedded_Interrupt_Endpoint

c) Function Control OUT routine

The deciphered SETUP token is kept as a transaction code which is defined in the file transctn.typ. The actual data transfer is performed by the Function Control IN routine.

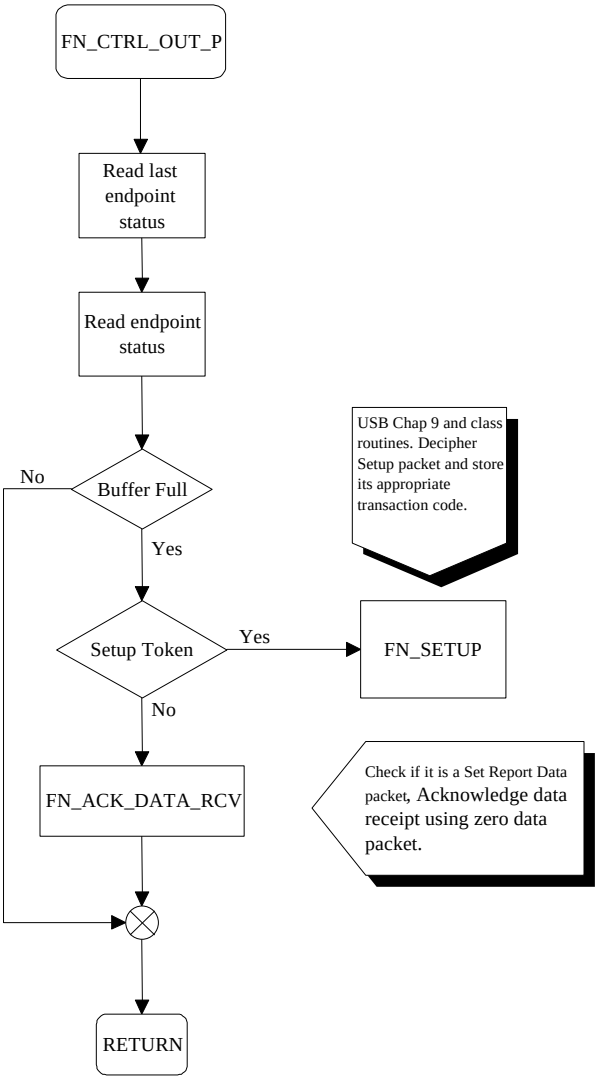


FIG 4: FUNCTION CONTROL OUT

This routine is executed when D11A receives a Setup packet on the upstream. The packet is rejected if all 8 bytes are not received, or if the received packet is not a Setup packet. When a Setup packet is received, the firmware branches to READ_SETUP_DATA to decipher the Setup packet and to execute respective routines.

Application Notes: USB Keyboard Mouse using PDIUSBD11

d) Function Control IN

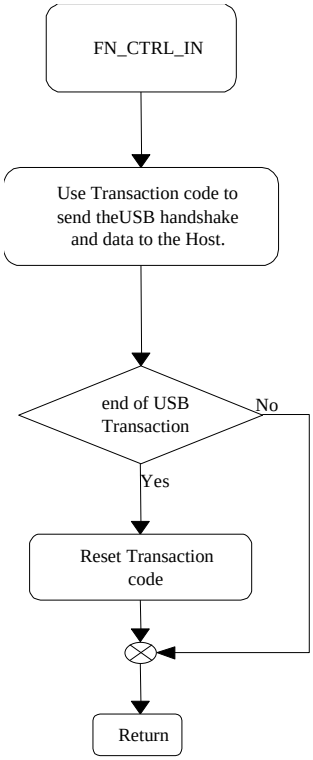


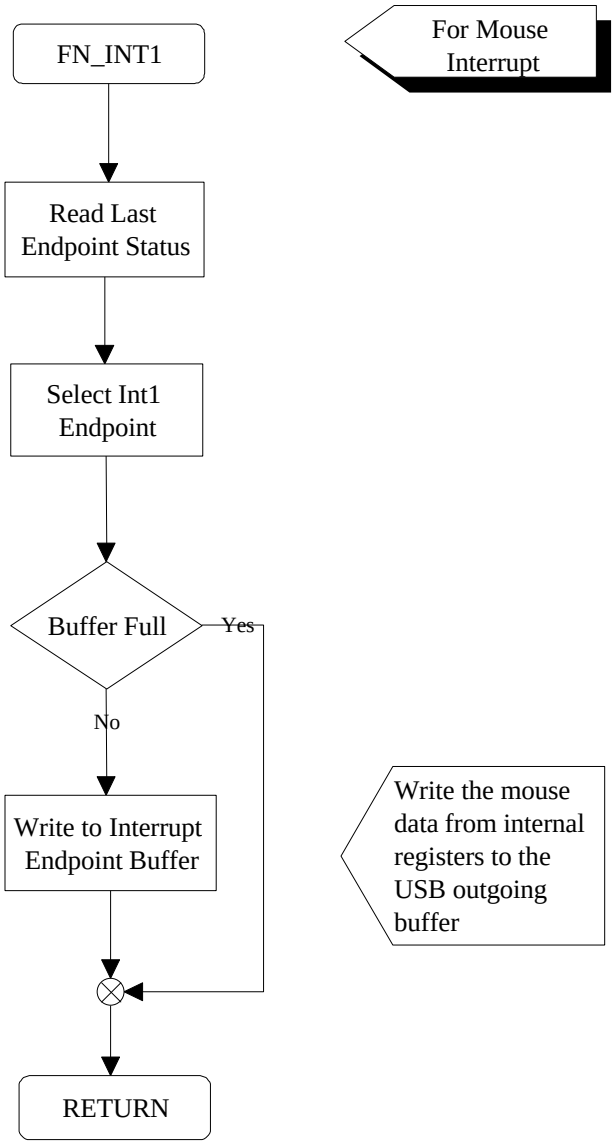
FIG 5: FUNCTION CONTROL IN

This routine is executed when D11 receives an IN token on the upstream. The packet is rejected if all 8 bytes are not received. The firmware branches to MORE_MESSAGES if any data remains from the previous IN token. If the last transaction was SET_ADDRESS, the Hub address is changed to the new address during this transaction.

Application Notes: USB Keyboard Mouse using PDIUSBD11

e) Function Interrupt 1.

The mouse interrupt routine is called when each IN token arrives on endpoint 2.

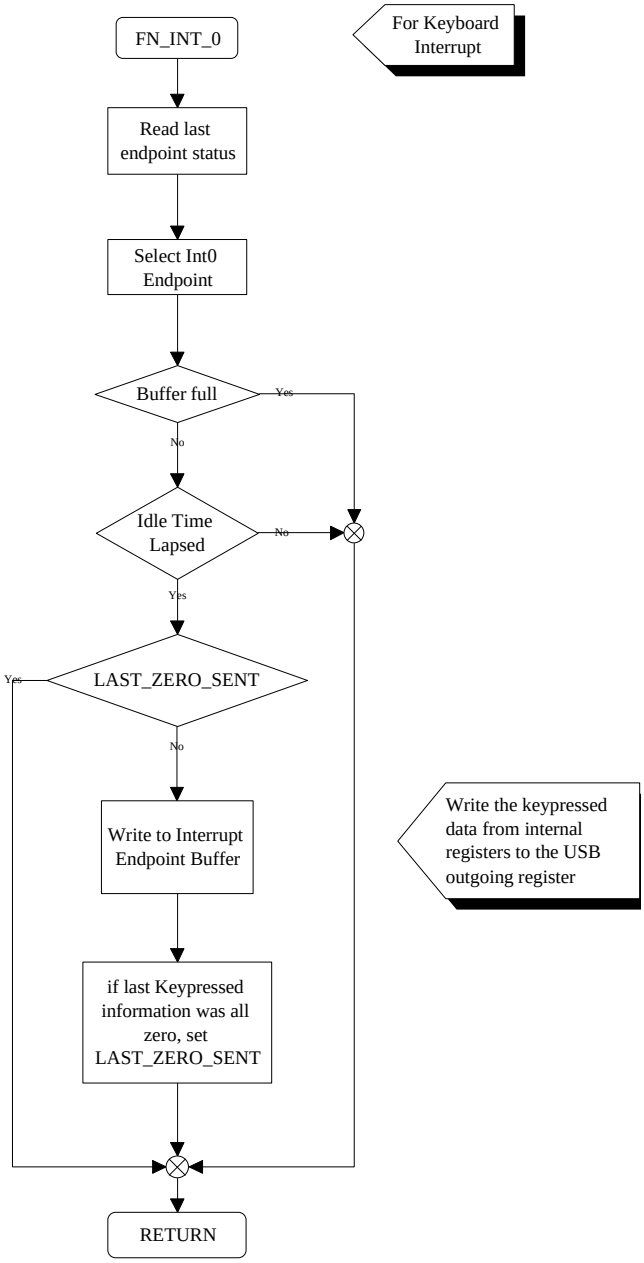


Application Notes: USB Keyboard Mouse using PDIUSBD11

f) Function-Interrupt 0.

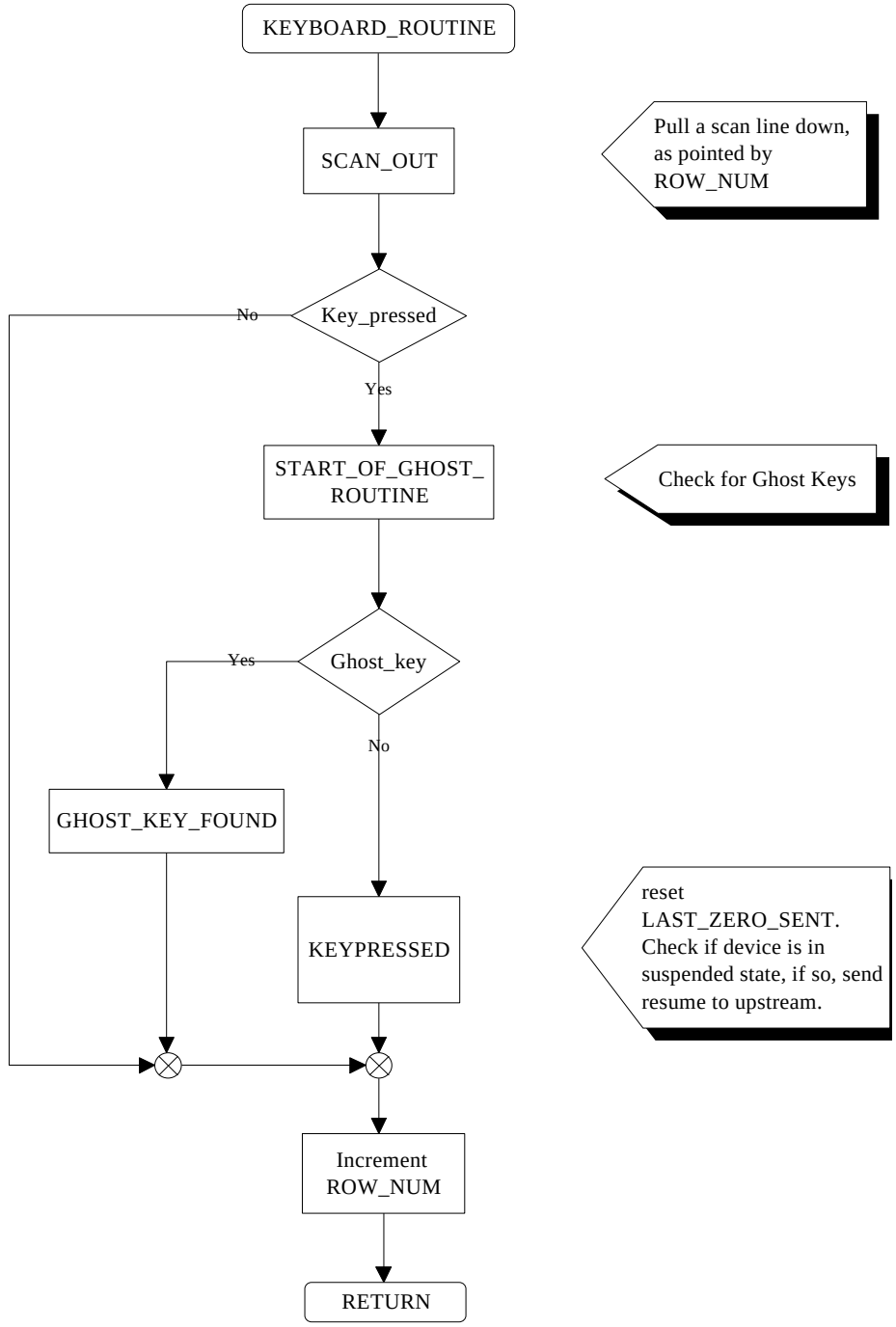
D11A should be configured to operate in Debug mode, to enable polling by the Host on a specific endpoint.

In the USB Keyboard Mouse, one of the endpoints is configured as an Input with an Interrupt attribute. This means that D11A generates an Interrupt for every Input token received from the Host on this endpoint.



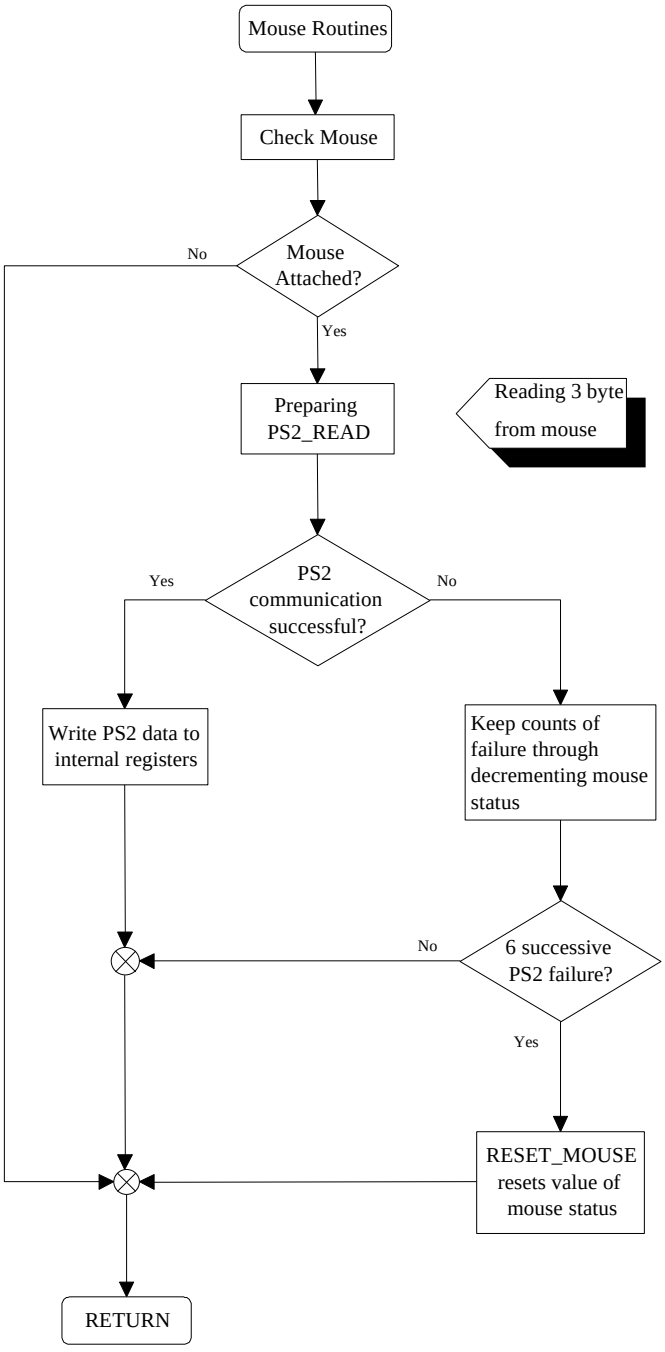
Application Notes: USB Keyboard Mouse using PDIUSBD11

g) Keyboard routine:



Application Notes: USB Keyboard Mouse using PDIUSBD11

h) Mouse routine:



Application Notes: USB Keyboard Mouse using PDIUSBD11

Notes on Compatibility

The D11 firmware can be used with an D11 because both are software -compatible. This means that the main Interrupt routine also points to a Hub Interrupt. However, the Interrupt is never called.

If the Interrupt pin is activated without a source from byte 1 of the Interrupt register, this indicates that a BUS reset condition has occurred, since endpoints 8 and 9 are not used in the current configuration. Bus-resets can also be determined by reading the 6^{th} bit of the 2^{nd} byte of the Interrupt register.