# Computer Labs: The PS/2 Mouse

2º MIEIC

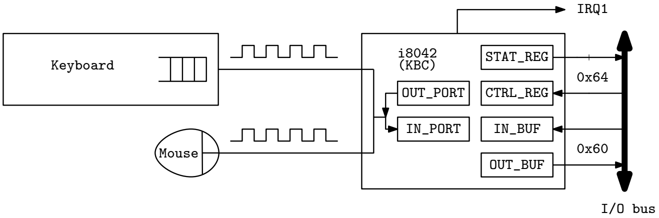
Pedro F. Souto (pfs@fe.up.pt)

October 8, 2011

## PS/2 Mouse Operation

- The mouse has its own controller chip
- It keeps the state of its buttons, i.e. whether or not they are pressed down
- It has two 9-bit 2's complement counters to keep track of the mouse's movement in the plane (one in each direction)
  - They measure a relative movement, i.e. they are reset every time the mouse reports their value
    - The default **resolution** is 4 counts/mm
  - If either of these counters overflows, the controller sets a corresponding overflow flag
- The controller sends this information to the PC via a serial line in 3-byte data packet
  - The protocol used for communication is the protocol used for communication with keyboard
  - On the PC side communication is handled by the KBC

# PS/2 Mouse



| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Byte 1 | Y Ovfl | X Ovfl | Y Sign | X Sign | 1 | M.B. | R.B. | L.B. |
| Byte 2 | X delta | | | | | | | |
| Byte 3 | Y delta | | | | | | | |

- A **scaling** parameter in the mouse controller affects the value of the counters reported by the mouse. There are 2 values for this parameter:
  - 1:1 In this case, the values reported are the counters values
  - 2:1 In this case, the values reported are a function of the counters values as determined by a table

## PS/2 Mouse Operating Modes

**Stream Mode** The mouse sends the data packet at a (programmable) maximum fixed rate to the KBC, as determined by “mouse events”, i.e. mouse movements and changes in buttons state

**Remote Mode** The mouse sends data packets only upon request of the KBC

- In either case, each of the bytes of the mouse data packet are put in the KBC’s output buffer, and
- The KBC raises IRQ12 (i.e. IRQ4 of PIC 2)
  - Once for each byte
- The mouse IH should read one byte per interrupt

## PS/2 Mouse-Related KBC Commands

- These commands are for the KBC and must be written to the `CMD_REG`
  - Arguments and return values are passed via the `DATA_REG`

| Command | Meaning | Args (A)/ Return (R) |
|---|---|---|
| `0x20` | Read Command Byte | Command byte (R) |
| `0x60` | Write Command Byte | Command byte (A) |
| `0xA7` | Disable Mouse | |
| `0xA8` | Enable Mouse | |
| `0xA9` | Check Mouse Interface | Returns 0, if OK |
| `0xD4` | Write Byte to Mouse | Byte (A) |

- `0xD4` commands the KBC to forward its argument to the mouse without any interpretation

## (KBC “Command Byte”)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | DIS2 | DIS | – | – | INT2 | INT |

`DIS2` 1: disable mouse

`DIS` 1: disable keyboard

`INT2` 1: enable interrupt on OBF, from mouse;

`INT` 1: enable interrupt on OBF, from keyboard

`-` : Either not used or not relevant

Read Use KBC command 0x20, which must be written to `CMD_REG`

Write Use KBC command 0x60, which must be written to `CMD_REG`

# PS/2 Mouse Commands (1/3)

## Commands passed as arguments of command `0xD4`

| Command | Function | Description/Comments |
|---|---|---|
| `0xFF` | Reset | Mouse reset |
| `0xFE` | Resend | For serial communications errors |
| `0xF6` | Set Defaults | Set default values |
| `0xF5` | Disable Stream Mode | In stream mode, should be sent before any other command |
| `0xF4` | Enable Sending Data Packets | In stream mode only |
| `0xF3` | Set Sample Rate | Sets state sampling rate |
| `0xF0` | Set Remote mode | Send data on request only |
| `0xEB` | Read Data | Send data packet request |
| `0xEA` | Set Stream Mode | Send data on events |
| `0xE9` | Status Request | Get mouse configuration (3 bytes) |
| `0xE8` | Set Resolution | |
| `0xE7` | Set Scaling 2:1 | *Acceleration* mode |
| `0xE6` | Set Scaling 1:1 | Linear mode |

Note 1 Arguments of these commands, if any, must also be passed as arguments of command `0xD4`

Note Responses to these commands, if any, are put in the KBC's `OUT_BUF` and should be read via the `DATA_REG`

# PS/2 Mouse Commands (2/3)

- Each of these commands is sent to the mouse, it is not interpreted by the KBC
  - The command is passed as argument of command `0xD4`
  - Arguments, if any, of a command must also be passed as arguments of command `0xD4` of the KBC
    - Command `0xD4` is: “Write **Byte** to Mouse”
- In response to all bytes it receives

  either commands (except for the resend command, `0xFE`) or their arguments

  the mouse controller sends an acknowledgment byte:

  ACK `0xFA` if everything OK

  NACK `0xFE` if invalid byte (may be because of a serial communication error)

  ERROR `0xFC` second consecutive invalid byte

## PS/2 Mouse Commands (3/3)

- The acknowledgment byte for each byte written as argument of command `0xD4` is put in the KBC's `OUT_BUF` and should be read via the `DATA_REG`
- Note that:

> *"When the host gets an `0xFE` response, it should retry the offending command. If an argument byte elicits an `0xFE` response, the host should retransmit the entire command, not just the argument byte."*
>
> Synaptics TouhcPad Interfacing Guide, pg. 31

IMPORTANT The acknowledgment byte is **not** the response to the command.

- For commands that elicit one response, the mouse controller will send it after the acknowledgment to the last byte of the command (including the args, if any).

## KBC Initialization: `kbc_init()`

- In Lab5, we will use Minix 3, which already initializes the mouse, even though it does not use it

1. Enable mouse interface
2. Reset mouse
   - Check for ACK (0xFA) response
3. Wait for 300-500 ms
   - Read OK (0xAA) , Read devide ID (0x00)
4. If OK response

   4.1 Disable stream mode

   4.2 Set sample rate (samples/s)

   4.3 Set mouse resolution (counts/mm)

   4.4 Set mouse scaling to 1:1

   4.5 Read KBC command byte
   - If timeout, enable KBD interface, mouse and IRQ1 and IRQ12 generation

   4.6 Write it back, enabling IRQ12 generation

   4.7 Enable stream mode

## kbc_init() Diagnosis

### Mouse reset

```
write_kbc: add=64 data=d4 // write Mouse CMD to CMD_REG
write_kbc: add=60 data=ff // write Reset to DATA_REG
read_kbc: data=fa         // read ACK from DATA_REG
read_kbc: data=aa         // read OK from DATA_REG
read_kbc: data=0          // read DeviceID from DATA_REG
```

### Disable stream mode

```
write_kbc: add=64 data=d4 // write Mouse CMD to CMD_REG
write_kbc: add=60 data=f5 // write Disable Stream to DATA_RE
read_kbc: data=fa         // read ACK as response
```

### Disable mouse

```
write kbc: add=64 data=a7 // write to CMD_REG
```

## KBC: Some Success Hints

- Use `kbc_init()` to initialize the KBC
  - Disable KBD and mouse interrupts before, enable them afterwards
- In the IH, read only one byte from the KBC
  - No need to ckeck the `STAT_REG`
  - The KBC uses different IRQ lines for the keyboard and the mouse
- Variables to update in the IRQ
  - A 3-byte array for the mouse packet
  - The index of the current position of the array (use an int)
- Make sure that when you display the 3-bytes, they all belong to the same packet.
- Finally,

  > If the device is in Stream mode (the default) and has been enabled with an Enable (`0xF4`) command, then the host should disable the device with a Disable (`0xF5`) command before sending any other command.
  >
  > Synaptics TouchPad Interfacing Guide, pg. 33

## Further Reading

- Synaptics Synaptics TouchPad Interfacing Guide, 2nd Ed. (Read only Subsections 3.2.3 thru 3.7.1, except Section 3.5 and Subsection 3.6.2.)
- Andries Brouwer's The PS/2 Mouse, Ch. 13 of Keyboard scancodes
- Adam Chapweske's The PS/2 Mouse Interface