Computer Labs: The PS/2 Mouse

2º MIEIC

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

October 27, 2017

Lab4: The PS/2's Mouse

- Write functions:

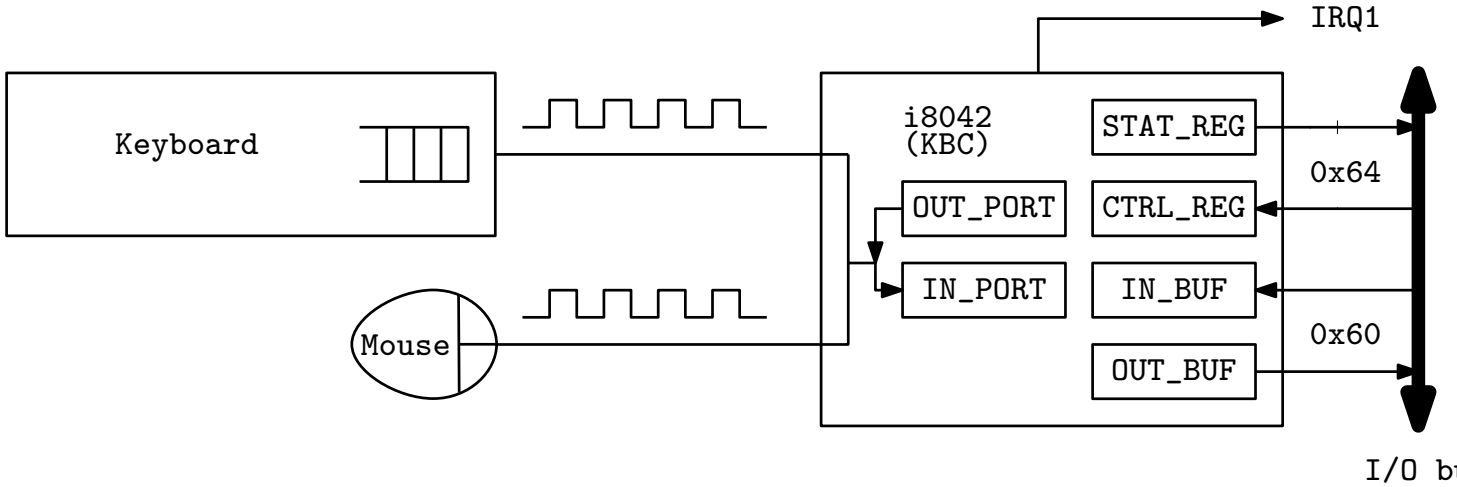
```
int mouse_test_packet(unsigned short cnt);
int mouse_test_async(void);
int mouse_test_remote(unsigned short cnt);
int mouse_test_gesture(short length, unsigned short toleran
```

  that require interfacing with the mouse, via the PC's keyboard controller
- These functions are not the kind of functions that you can reuse later in your project
  - The idea is that you design the lower level functions (with the final project in mind).
- What's new?
  - Use the KBC controller (i8042) to interface with the mouse
  - Process mouse interrupts
  - (Handling multiple asynchronous interrupts)
  - Use state machines

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

Lab4: `mouse_test_packet` (1/2)

What Print the packets received from the mouse

Details Should:

- Terminate after processing the given number of packets
- Display the packets contents in a human friendly way (see Figure 1 of lab handout)

How Need to subscribe the mouse interrupts

- Upon an interrupt, read the byte from the `OUT_BUF`

Note There is no need to configure the mouse

- It is already initialized by Minix

But Need to enable stream mode (see PS/2 Mouse commands)

- Minix disables stream mode in text mode

Issue Minix already has an IH installed

- Disable it by subscribing the mouse interrupt with `IRQ_EXCLUSIVE` policy

Lab4: `mouse_test_packet` (2/2)

KBC interrupt subscription in exclusive mode;

`driver_receive()` loop (similar to that of labs 2 and 3)

Interrupt handler reads the bytes from the KBC's `OUT_BUF`

- Should read only one byte per interrupt
  - The communication between the mouse and the KBC is too slow
- Should use:

  `packet[]` to store the packet bytes

  `counter` to keep track of byte number

Synchronization Issues All 3 bytes must belong to the same packet

Challenge The bytes in a packet have no id

Hint Bit 3 of first byte of a packet is always set

- But this bit may also be set in other bytes of a packet

PS/2 Mouse-Related KBC Commands

- These commands are for the KBC and must be written to port `0x64`
  - Arguments and return values are passed via port `0x60`
  - Do not forget to check the IBF bit in the `STATUS_REG`, before writing to either port

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 port `0x64`

Write Use KBC command 0x60, which must be written to port `0x64`

Status Register

- Input from/output to KBC requires reading the status register

Bit	Name	Meaning (if set)
7	Parity	Parity error - invalid data
6	Timeout	Timeout error - invalid data
5	Aux	Mouse data
4	INH	Inhibit flag: 0 if keyboard is inhibited
3	A2	A2 input line: 0 data byte 1 command byte
2	SYS	System flag: 0 if system in power-on reset, 1 if system already initialized
1	IBF	Input buffer full don't write commands or arguments
0	OBF	Output buffer full - data available for reading

- Bits 5, `Aux`, indicates whether the data in the `OUT_BUF` is coming from the Mouse (auxiliary device) or the keyboard
- Do not write to the `IN_BUF (0x60)` or the `CTRL_REG (0x64)`, if bit 1, i.e. the `IBF`, is set.

PS/2 Mouse Commands (1/4)

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 (Data Reporting)	In stream mode, should be sent before any other command
`0xF4`	Enable (Data Reporting)	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 port `0x60`

PS/2 Mouse Commands (2/4)

- 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

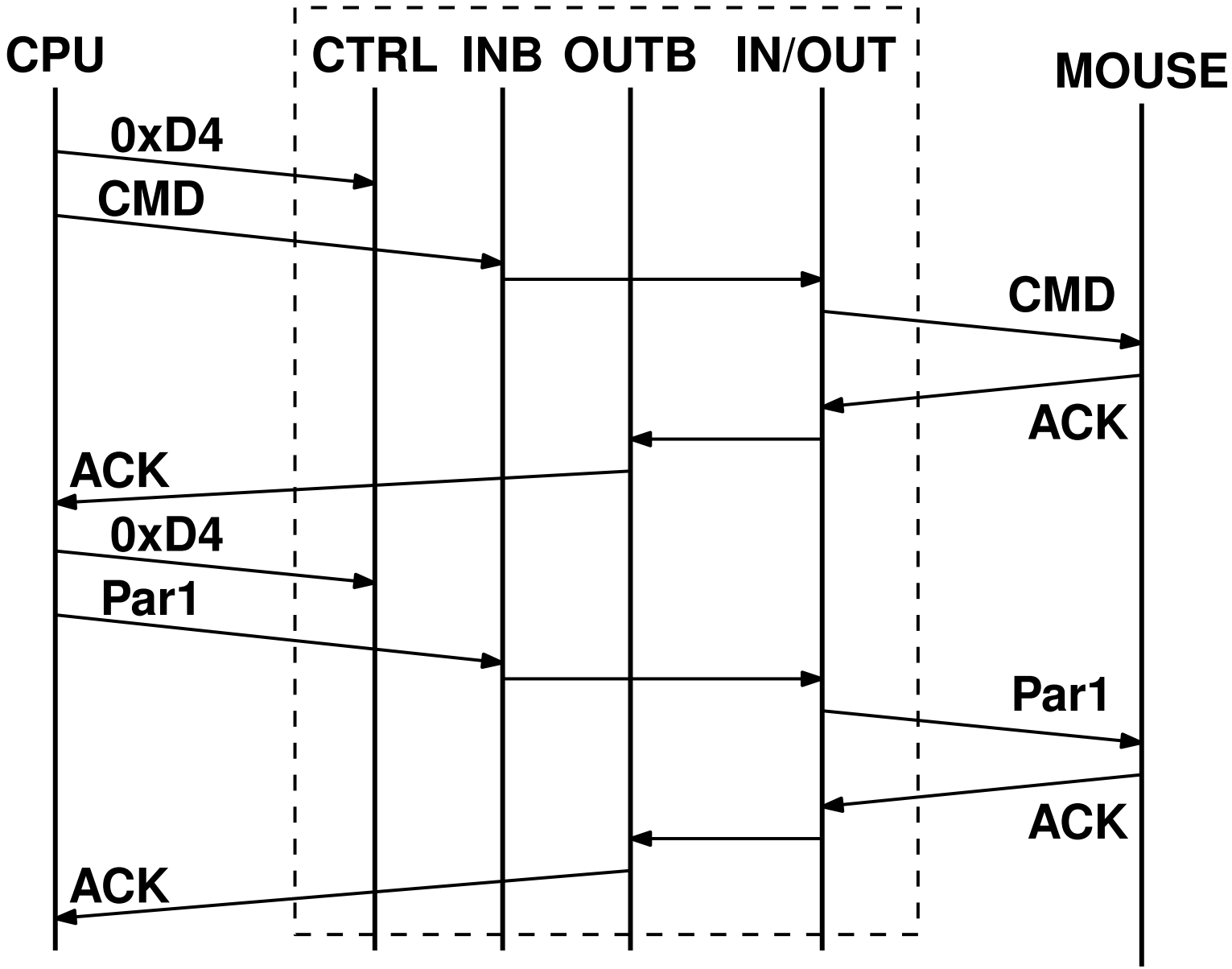
  the mouse controller sends an acknowledgment byte:

  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 (3/4)



Not representing polling of `STATUS_REG` for `IBF/OBF`

PS/2 Mouse Commands (4/4)

- 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 port `0x60`
- 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).

Lab 4: Other functions

`test_async()`

- Must subscribe also the Timer 0 interrupts
- Similar to `kbd_test_timed_scan()`, of Lab 3

`mouse_test_remote()`

- Use **remote mode** rather than stream mode
  - Must give appropriate mouse commands: Disable (`0xF5`) followed by Set Remote mode (`0xF0`)
- Set **stream mode** before exiting
  - ... but ensure data reporting is disabled
- Mouse packets must be read (using Read Data (`0xEB`)) and displayed periodically
  - You can measure the period with `tickdelay()`
  - You can use either polling or interrupts to read the packet bytes
    - If you use polling, you need to disable Minix's IH
    - Either by subscribing the mouse interrupts, but not checking for notifications
    - Or by writing to the KBC's command byte
  - On the platforms I've tested, no displacement reported

Mouse: Some Success Hints

- 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 IH
  - 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.
- Do not forget:

  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
- Finally:
  - If a byte is left in the `OUT_BUF`, the KBC will not generate further interrupts

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