Computer Labs: The PS/2 Mouse

2º MIEIC

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

March 13, 2025

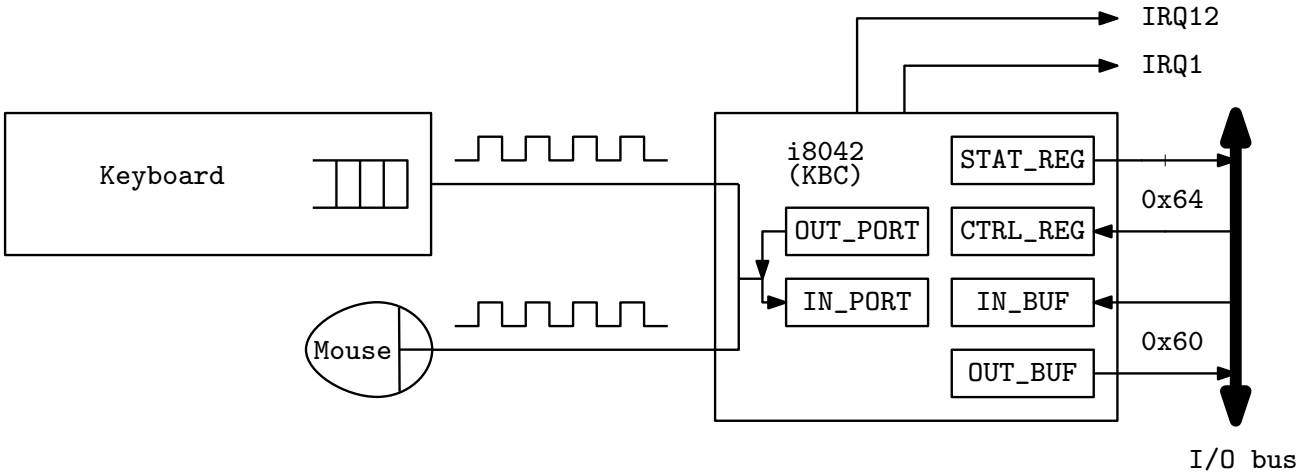
Lab4: The PS/2's Mouse

- Write functions:

```
int mouse_test_packet(uint32_t cnt);
int mouse_test_async(uint8_t idle_time)
int mouse_test_gesture(?????);
```

  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, like the keyboard. It:

  Detects events on the mouse

  Pressing/releasing of the mouse buttons
  Displacement of the mouse on the plane.

  - It uses two 9-bit **2's complement** counters, one per direction
  - That are reset every time the controller reports their value
  - Some touchpads can be configured to report their absolute position

  Reports these events to the KBC, by sending a 3-byte packet via a serial line

  - This protocol is the one also used in the communication between the keyboard and the KBC

PS/2 Mouse Data Packet

	7	6	5	4	3	2	1	0
Byte 1	Y Ovfl	X Ovfl	MSB Y Delta	MSB X Delta	1	M.B.	R.B.	L.B.
Byte 2	X delta							
Byte 3	Y delta							

`X MSB/Y MSB` Relative displacement MSBit X/Y axis

`X delta/Y delta` Relative displacement 8 LS bits in the X/Y axis since the previous packet

- `X MSB` and `X delta` form a 9-bit 2's complement integer

`X Ovfl/Y Ovfl` Flag that the mouse displacement is too large to be represented as a 9-bit 2-complement integer

`M.B, R.B, L.B` State of the middle, right and left buttons: `1` if pressed.

- A **scaling** parameter in the mouse controller affects the value of the counters reported by the mouse. There are 2 values:

  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
  - This can be enabled/disabled by writing in the command byte
- The mouse IH should read one byte per interrupt
  - In remote mode it is easier not to use interrupts

Lab4: `mouse_test_packet` (1/2)

What Print the packets received from the mouse in **stream mode**

Details Should:

- Terminate after processing the given number of packets
- Display the packets contents in a human friendly way, by LCF function (**we** provide it): `mouse_print_packet()`

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
- **Initially**, can use `mouse_enable_data_reporting()` of the LCF (i.e., provided by us)

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
  - Communication between the mouse and the KBC is too slow
- Must not call `mouse_print_packet()`

Packet Assembly Can 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

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
- 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

(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 2 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

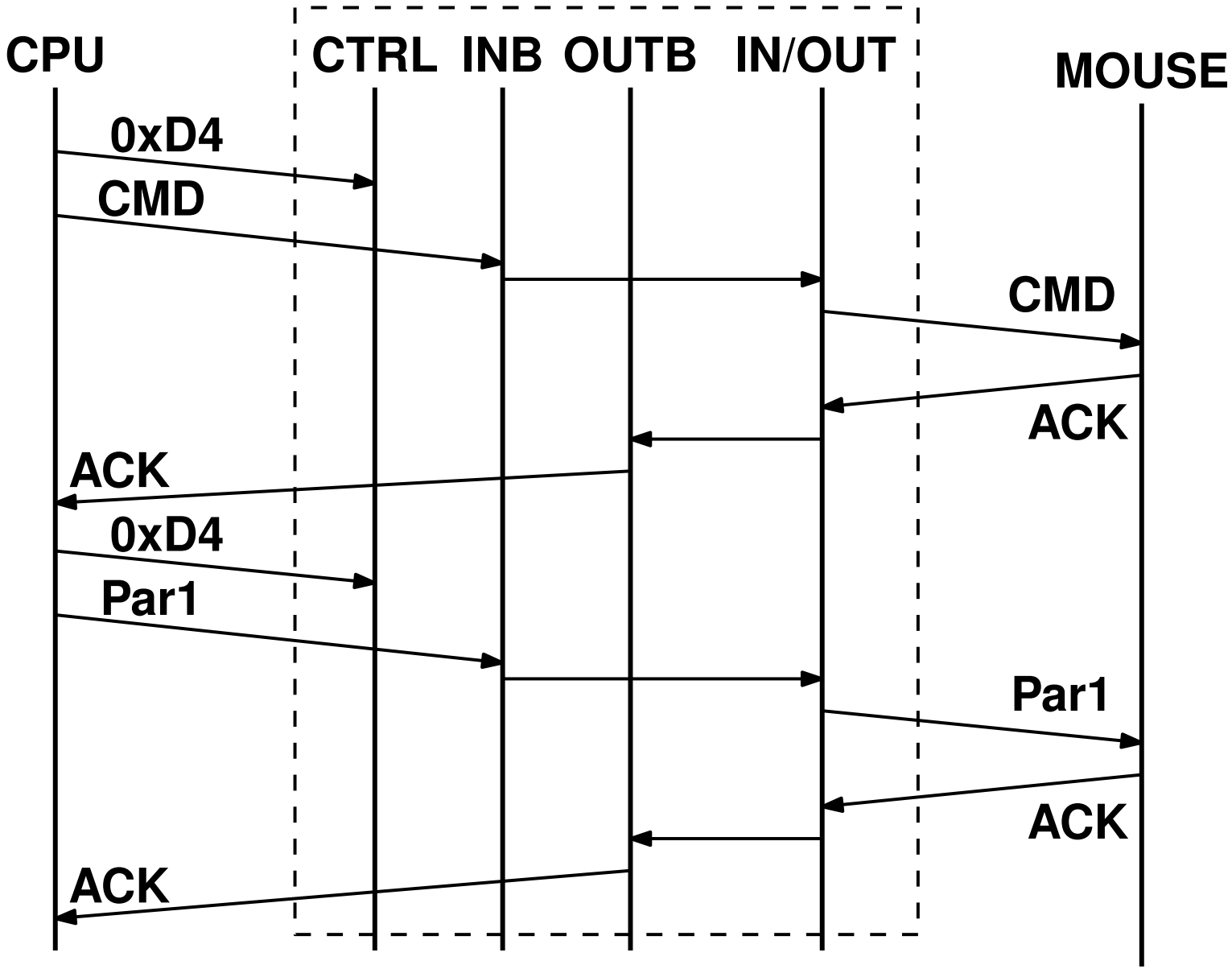
  the mouse controller sends an acknowledgment byte:

  ACK (`0xFA`) if everything OK

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

  ERROR (`0xFC`) second consecutive invalid byte
- 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`

PS/2 Mouse Commands (3/4)



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

PS/2 Mouse Commands (4/4)

- 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: `mouse_test_async()`

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

Mouse: Some Success Hints

- In the IH, read only one byte from the KBC
  - No need to ckeck the `OBF` or the `AUX` bits
  - The KBC uses different IRQ lines for the keyboard and the mouse
- Assemble packets using:
  - A 3-byte array for the mouse packet
  - The index of the current position of the array
- 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