Next Previous Contents

11. The AT keyboard controller

A user program can talk to the keyboard controller on the motherboard. The keyboard controller can again talk to the keyboard.

When a key is pressed the keyboard sends the corresponding keyboard scancode to the keyboard controller, and the keyboard controller translates that and interrupts the CPU, allowing the CPU to read the result.

More detailed: when a key is pressed, the keyboard sends a start bit (low), followed by 8 data bits for the keyboard scancode of the key (least significant first), followed by an odd parity bit, followed by a stop bit (high). The keyboard controller reads the data and checks the parity. If incorrect, retransmission is requested. If incorrect again a parity error is reported. If the time between request to send and start of transmission is greater than 15 ms, or if the eleven bits are not received within 2ms, a timeout is reported. In both cases (parity error or timeout), the data byte is set to 0xff.

The keyboard controller has three 8-bit registers involved in communication with the CPU: its input buffer, that can be written by the CPU by writing port 0x60 or port 0x64; its output buffer, that can be read by the CPU by reading from port 0x60; and the status register, that can be read by the CPU by reading from port 0x64.

If the CPU writes to port 0x64, the byte is interpreted as a command byte. If the CPU writes to port 0x60, the byte is interpreted as a data byte.

The keyboard controller has two 8-bit I/O ports involved in communication with the keyboard: the input port P1 (receiving input from the keyboard) and the output port P2 (for sending output to the keyboard).

11.1 The keyboard controller status register

The keyboard controller has an 8-bit status register. It can be inspected by the CPU by reading port 0x64.

(Typically, it has the value 0x14: keyboard not locked, self-test completed.)

PARE	TIM	AUXB	KEYL	C/D	SYSF	INPB	OUTB

Bit 7: Parity error

0: OK. 1: Parity error with last byte.

Bit 6: Timeout

0: OK. 1: Timeout. On PS/2 systems: General timeout. On AT systems: Timeout on transmission from keyboard to keyboard controller. Possibly parity error (in which case both bits 6 and 7 are set).

Bit 5: Auxiliary output buffer full

On PS/2 systems: Bit 0 tells whether a read from port 0x60 will be valid. If it is valid, this bit 5 tells what data will be read from port 0x60. 0: Keyboard data. 1: Mouse data.

On AT systems: 0: OK. 1: Timeout on transmission from keyboard controller to keyboard. This may indicate that no keyboard is present.

Bit 4: Keyboard lock

0: Locked. 1: Not locked.

Bit 3: Command/Data

0: Last write to input buffer was data (written via port 0x60). 1: Last write to input buffer was a command (written via port 0x64). (This bit is also referred to as Address Line A2.)

Bit 2: System flag

Set to 0 after power on reset. Set to 1 after successful completion of the keyboard controller self-test (Basic Assurance Test, BAT). Can also be set by command (see below).

Bit 1: Input buffer status

0: Input buffer empty, can be written. 1: Input buffer full, don't write yet.

Bit 0: Output buffer status

0: Output buffer empty, don't read yet. 1: Output buffer full, can be read. (In the PS/2 situation bit 5 tells whether the available data is from keyboard or mouse.) This bit is cleared when port 0x60 is read.

11.2 The keyboard controller command byte

The keyboard controller is provided with some RAM, for example 32 bytes, that can be accessed by the CPU. The most important part of this RAM is byte 0, the Controller Command Byte (CCB). It can be read/written by writing 0x20/0x60 to port 0x64 and then reading/writing a data byte from/to port 0x60.

This byte has the following layout.

Bit 7: Unused

Always 0.

Bit 6: Translate

0: No translation. 1: Translate keyboard scancodes, using the translation table given above. MCA type 2 controllers cannot set this bit to 1. In this case scan code conversion is set using keyboard command 0xf0 to port 0x60.

Bit 5: Mouse enable

On an EISA or PS/2 system: 0: Enable mouse. 1: Disable mouse by driving the clock line low. On an ISA system: "PC Mode": 0: use 11-bit codes, check parity and do scan conversion. 1: use 8086 codes, don't check parity and don't do scan conversion.

Bit 4: Keyboard enable

0: Enable keyboard. 1: Disable keyboard by driving the clock line low.

Bit 3: Ignore keyboard lock

For PS/2: Unused, always 0. For AT: 0: No action. 1: Force bit 4 of the status register to 1, "not locked". This is used for keyboard testing after power on. Maybe only on older motherboards.

Bit 2: System flag

This bit is shown in bit 2 of the status register. A "cold reboot" is one with this bit set to zero. A "warm reboot" is one with this bit set to one (BAT already completed). This will influence the tests and initializations done by the POST.

Bit 1: Mouse interrupt enable

On an ISA system: unused, always 0. On an EISA or PS/2 system: 0: Do not use mouse interrupts. 1: Send interrupt request IRQ12 when the mouse output buffer is full.

Bit 0: Keyboard interrupt enable

0: Do not use keyboard interrupts. 1: Send interrupt request IRQ1 when the keyboard output buffer is full.

When no interrupts are used, the CPU has to poll bits 0 (and 5) of the status register.

11.3 Keyboard controller commands

The CPU can command the keyboard controller by writing port 0x64. Useful, generally available, keyboard commands are:

20	Read keyboard controller command byte
60	Write keyboard controller command byte
aa	Self test
ab	Interface test
ad	Disable keyboard
ae	Enable keyboard
c0	Read input port
d0	Read output port
d1	Write output port
e0	Read test inputs
fe	System reset

Useful, generally available, mouse commands are:

a7	Disable mouse port
a8	Enable mouse port
a9	Test mouse port
d4	Write to mouse

Obscure, probably obsolete, commands:

00-1f	Read keyboard controller RAM
20-3f	Read keyboard controller RAM
40-5f	Write keyboard controller RAM
60-7f	Write keyboard controller RAM
90-93	Synaptics multiplexer prefix
90-9f	Write Port13-Port10
a0	Read copyright
a1	Read firmware version
a2	Switch speed
a3	Switch speed
a4	Check if password installed
a5	Load password
a6	Check password
ac	Diagnostic dump
af	Read keyboard version
b0-b5	Reset keyboard controller line
b8-bd	Set keyboard controller line
c1	Continuous input port poll, low
c2	Continuous input port poll, high
c8	Unblock lines P22 and P23
c9	Block lines P22 and P23
ca	Read keyboard controller mode
cb	Write keyboard controller mode
d2	Write keyboard output buffer
d3	Write mouse output buffer
dd	Disable A20 address line
df	Enable A20 address line
f0-ff	Pulse output bit

Command 0x00-0x1f: Read keyboard controller RAM

(AMIBIOS only) Aliases for 0x20-0x3f.

Command 0x20-0x3f: Read keyboard controller RAM

The last six bits of the command specify the RAM address to read. The read data is placed into the output buffer, and can be read by reading port 0x60. On MCA systems, type 1 controllers can access all 32 locations; type 2 controllers can only access locations 0, 0x13-0x17, 0x1d, 0x1f.

Location 0 is the Command byte, see above.

Location 0x13 (on MCA) is nonzero when a password is enabled.

Location 0x14 (on MCA) is nonzero when the password was matched.

Locations 0x16-0x17 (on MCA) give two make codes to be discarded during password matching.

Command 0x40-0x5f: Write keyboard controller RAM

(AMIBIOS only) Aliases for 0x40-0x5f.

Command 0x60-0x7f: Write keyboard controller RAM

Command 0x90-0x93: Synaptics routing prefixes

Prefix a PS/2 mouse command with one of these to talk to one of at most four multiplexed devices. See also the multiplexing handshake below.

Unfortunately, VIA also uses this command:

Command 0x90-0x9f: Write Port13-Port10

(VIA VT82C42) Write low nibble to Port13-Port10.

Command 0xa0: Read copyright

On some keyboard controllers: an ASCIZ copyright string (possibly just NUL) is made available for reading via port 0x60. On other systems: no effect, the command is ignored.

Command 0xa1: Read controller firmware version

On some keyboard controllers: a single ASCII byte is made available for reading via port 0x60. On other systems: no effect, the command is ignored.

Command 0xa2: Switch speed

(On ISA/EISA systems with AMI BIOS) Reset keyboard controller lines P22 and P23 low. These lines can be used for speed switching via the keyboard controller. When done, the keyboard controller sends one garbage byte to the system.

Command 0xa3: Switch speed

(On ISA/EISA systems with AMI BIOS) Set keyboard controller lines P22 and P23 high. These lines can be used for speed switching via the keyboard controller. When done, the keyboard controller sends one garbage byte to the system.

(Compaq BIOS: Enable system speed control.)

Command 0xa4: Check if password installed

On MCA systems: Return 0xf1 (via port 0x60) when no password is installed, return 0xfa when a password has been installed. Some systems without password facility always return 0xf1.

(On ISA/EISA systems with AMI BIOS) Write Clock = Low.

(Compaq BIOS: toggle speed.)

Command 0xa5: Load password

On MCA systems: Load a password by writing a NUL-terminated string to port 0x60. The string is in scancode format.

(On ISA/EISA systems with AMI BIOS) Write Clock = High.

(Compaq BIOS: special read of P2, with bits 4 and 5 replaced: Bit 5: 0: 9-bit keyboard, 1: 11-bit keyboard. Bit 4: 0: outp-buff-full interrupt disabled, 1: enabled.)

Command 0xa6: Check password

On MCA systems: When a password is installed: Check password by matching keystrokes with the stored password. Enable keyboard upon successful match.

(On ISA/EISA systems with AMI BIOS) Read Clock. 0: Low. 1: High.

Command 0xa7: Disable mouse port

On MCA systems: disable the mouse (auxiliary device) by setting its clock line low, and set bit 5 of the Command byte. Now P23 = 1.

(On ISA/EISA systems with AMI BIOS) Write Cache Bad.

Command 0xa8: Enable mouse port

On MCA systems: enable the mouse (auxiliary device), clear bit 5 of the Command byte. Now P23 = 0.

(On ISA/EISA systems with AMI BIOS) Write Cache Good.

Command 0xa9: Test mouse port

On MCA and other systems: test the serial link between keyboard controller and mouse. The result can be read from port 0x60. 0: OK. 1: Mouse clock line stuck low. 2: Mouse clock line stuck high. 3: Mouse data line stuck low. 4: Mouse data line stuck high. 0xff: No mouse.

(On ISA/EISA systems with AMI BIOS) Read Cache Bad or Good. 0: Bad. 1: Good.

Command 0xaa: Self test

Perform self-test. Return 0x55 if OK, 0xfc if NOK.

Command 0xab: Interface test

Test the serial link between keyboard controller and keyboard. The result can be read from port 0x60. 0: OK. 1: Keyboard clock line stuck low. 2: Keyboard clock line stuck high. 3: Keyboard data line stuck low. 4: Keyboard data line stuck high. 0xff: General error.

Command 0xac: Diagnostic dump

(On some systems) Read from port 0x60 sixteen bytes of keyboard controller RAM, and the output and input ports and the controller's program status word.

Command 0xad: Disable keyboard

Disable the keyboard clock line and set bit 4 of the Command byte. Any keyboard command enables the keyboard again.

Command 0xae: Enable keyboard

Enable the keyboard clock line and clear bit 4 of the Command byte.

Command 0xaf: Read keyboard version

(Award BIOS, VIA)

Command 0xb0-0xb5,0xb8-0xbd: Reset/set keyboard controller line

AMI BIOS: Commands 0xb0-0xb5 reset a keyboard controller line low. Commands 0xb8-0xbd set the corresponding keyboard controller line high. The lines are P10, P11, P12, P13, P22 and P23, respectively. (In the case of the lines P10, P11, P22, P23 this is on ISA/EISA systems only.) When done, the keyboard controller sends one garbage byte to the system.

VIA BIOS: Commands 0xb0-0xb7 write 0 to lines P10, P11, P12, P13, P22, P23, P14, P15. Commands 0xb8-0xbf write 1 to lines P10, P11, P12, P13, P22, P23, P14, P15.

Command 0xc0: Read input port

Read the input port (P1), and make the resulting byte available to be read from port 0x60.

Command 0xc1: Continuous input port poll, low

(MCA systems with type 1 controller only) Continuously copy bits 3-0 of the input port to be read from bits 7-4 of port 0x64, until another keyboard controller command is received.

Command 0xc2: Continuous input port poll, high

(MCA systems with type 1 controller only) Continuously copy bits 7-4 of the input port to be read from bits 7-4 of port 0x64, until another keyboard controller command is received.

Command 0xc8: Unblock keyboard controller lines P22 and P23

(On ISA/EISA systems with AMI BIOS) After this command, the system can make lines P22 and P23 low/high using command 0xd1.

Command 0xc9: Block keyboard controller lines P22 and P23

(On ISA/EISA systems with AMI BIOS) After this command, the system cannot make lines P22 and P23 low/high using command 0xd1.

Command 0xca: Read keyboard controller mode

(AMI BIOS, VIA) Read keyboard controller mode to bit 0 of port 0x60. 0: ISA (AT) interface. 1: PS/2 (MCA)interface.

Command 0xcb: Write keyboard controller mode

(AMI BIOS) Write keyboard controller mode to bit 0 of port 0x60. 0: ISA (AT) interface. 1: PS/2 (MCA)interface. (First read the mode using command 0xca, then modify only the last bit, then write the mode using this command.)

Command 0xd0: Read output port

Read the output port (P2) and place the result in the output buffer. Use only when output buffer is empty.

Command 0xd1: Write output port

Write the output port (P2). Note that writing a 0 in bit 0 will cause a hardware reset.

(Compaq: the system speed bits are not set. Use commands 0xa1-0xa6 for that.)

Command 0xd2: Write keyboard output buffer

(MCA) Write the keyboard controllers output buffer with the byte next written to port 0x60, and act as if this was keyboard data. (In particular, raise IRQ1 when bit 0 of the Command byte says

so.)

Command 0xd3: Write mouse output buffer

(MCA) Write the keyboard controllers output buffer with the byte next written to port 0x60, and act as if this was mouse data. (In particular, raise IRQ12 when bit 1 of the Command byte says so.)

Not all systems support this.

Synaptics multiplexing On the other hand, Synaptics (see ps2-mux.PDF) uses this command as a handshake between driver and controller: if the driver gives this command three times, with data bytes 0xf0, 0x56, 0xa4 respectively, and reads 0xf0, 0x56, but not 0xa4 back from the mouse output buffer, then the driver knows that the controller supports Synaptics AUX port multiplexing, and the controller knows that it does not have to do the usual data faking and goes into multiplexed mode. The third byte read is the version of the Synaptics standard.

There is a corresponding deactivation sequence, namely 0xf0, 0x56, 0xa5. (And again the last byte is changed to the version number of the standard supported.) This latter sequence works both in multiplexed mode and in legacy mode and can thus be used to determine whether this feature is present without activating it.

See also the multiplexer commands 0x90-0x93.

For some laptops it has been reported that bit 3 of every third mouse byte is forced to 1 (as it would be with the standard 3-byte mouse packets). This may turn 0xf0, 0x56, 0xa4 into 0xf0, 0x56, 0xac and cause misdetection of Synaptics multiplexing (for version 10.12).

Command 0xd4: Write to mouse

(MCA) The byte next written to port 0x60 is transmitted to the mouse.

Command 0xdd: Disable A20 address line

(HP Vectra)

Command 0xdf: Enable A20 address line

(HP Vectra)

Command 0xe0: Read test inputs

This command makes the status of the Test inputs T0 and T1 available to be read via port 0x60 in bits 0 and 1, respectively. Use only when the output port is empty.

Command 0xf0-0xff: Pulse output bit

Bits 3-0 of the output port P2 of the keyboard controller may be pulsed low for approximately 6 l'seconds. Bits 3-0 of this command specify the output port bits to be pulsed. 0: Bit should be pulsed. 1: Bit should not be modified. The only useful version of this command is Command 0xfe. (For MCA, replace 3-0 by 1-0 in the above.)

Command 0xfe: System reset

Pulse bit 0 of the output port P2 of the keyboard controller. This will reset the CPU.

11.4 The input port P1

This has the following layout.

bit 7	Keyboard lock	0: locked, 1: not locked
bit 6	Display	0: CGA, 1: MDA
bit 5	Manufacturing jumper	0: installed, 1: not installed
		with jumper the BIOS runs an infinite diagnostic loop
bit 4	RAM on motherboard	0: 512 KB, 1: 256 KB
bit 3		Unused in ISA, EISA, PS/2 systems
		Can be configured for clock switching
bit 2		Unused in ISA, EISA, PS/2 systems
		Can be configured for clock switching
	Keyboard power	PS/2 MCA: 0: keyboard power normal, 1: no power
bit 1	Mouse data in	Unused in ISA
bit 0	Keyboard data in	Unused in ISA

Clearly only bits 1-0 are input bits. Of the above, the original IBM AT used bits 7-4, while PS/2 MCA systems use only bits 2-0.

Where in the above lines P10, P11, etc are used, these refer to the pins corresponding to bit 0, bit 1, etc of port P1.

11.5 The output port P2

This has the following layout.

bit 7	Keyboard data	data to keyboard
bit 6	Keyboard clock	
bit 5	IRQ12	0: IRQ12 not active, 1: active
bit 4	IRQ1	0: IRQ1 not active, 1: active
bit 3	Mouse clock	Unused in ISA
bit 2	Mouse data	Unused in ISA. Data to mouse
bit 1	A20	0: A20 line is forced 0, 1: A20 enabled
bit 0	Reset	0: reset CPU, 1: normal

Where in the above lines P20, P21, etc are used, these refer to the pins corresponding to bit 0, bit 1, etc of port P2.

11.6 The test port T

bit 0

Keyboard clock (input).

bit 1

(AT) Keyboard data (input). (PS/2) Mouse clock (input).

Next Previous Contents