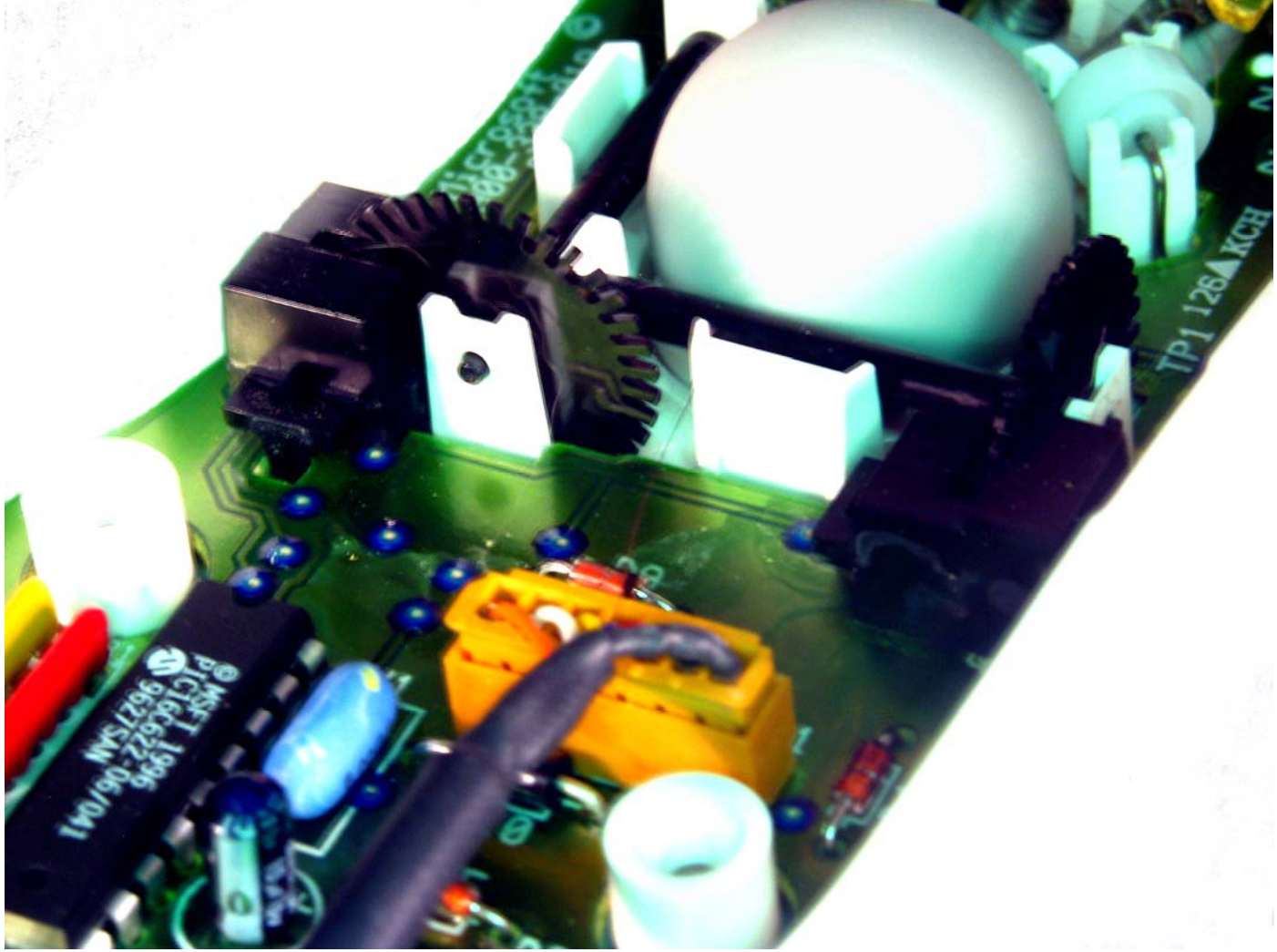


A PS/2 mouse is shown above with the cover removed. The ball (upper right) rolls two plastic X and Y axles with a slotted wheel at one end. The slotted wheel passes through a square slotted case containing an IR emitter and detector pair. When the wheel rotates it generates pulses by interrupting the IR light beam. A microcontroller (lower left) counts the pulses and sends data packets containing mouse movement and button data to the PC.

Table 11.1 PS/2 Keyboard Commands and Messages.

Commands Sent to Keyboard	Hex Value
Reset Keyboard Keyboard returns AA, 00 after self-test	FF
Resend Message	FE
Set key typematic (autorepeat) XX is scan code for key	FB, XX
Set key make and break	FC, XX
Set key make	FD, XX
Set all key typematic, make and break	FA
Set all keys make	F9
Set all keys make and break	F8
Make all keys typematic (autorepeat)	F7
Set to Default Values	F6
Clear Buffers and start scanning keys	F4
Set typematic (autorepeat) rate and delay Set typematic (autorepeat) rate and delay Bits 6 and 5 are delay (250ms to 1 sec) Bits 4 to 0 are rate (all 0's-30x/sec to all 1's 2x/sec)	F3, XX
Read keyboard ID Keyboard sends FA, 83, AB	F2
Set scan code set XX is 01, 02, or 03	F0, XX
Echo	EE
Set Keyboard LEDs XX is 00000 Scroll, Num, and Caps Lock bits 1 is LED on and 0 is LED off	ED, XX

Table 11.2 PS/2 Commands and messages sent by keyboard.

Messages Sent by Keyboard	Hex Value
Resend Message	FE
Two bad messages in a row	FC
Keyboard Acknowledge Command Sent by Keyboard after each command byte	FA
Response to Echo command	EE
Keyboard passed self-test	AA
Keyboard buffer overflow	00

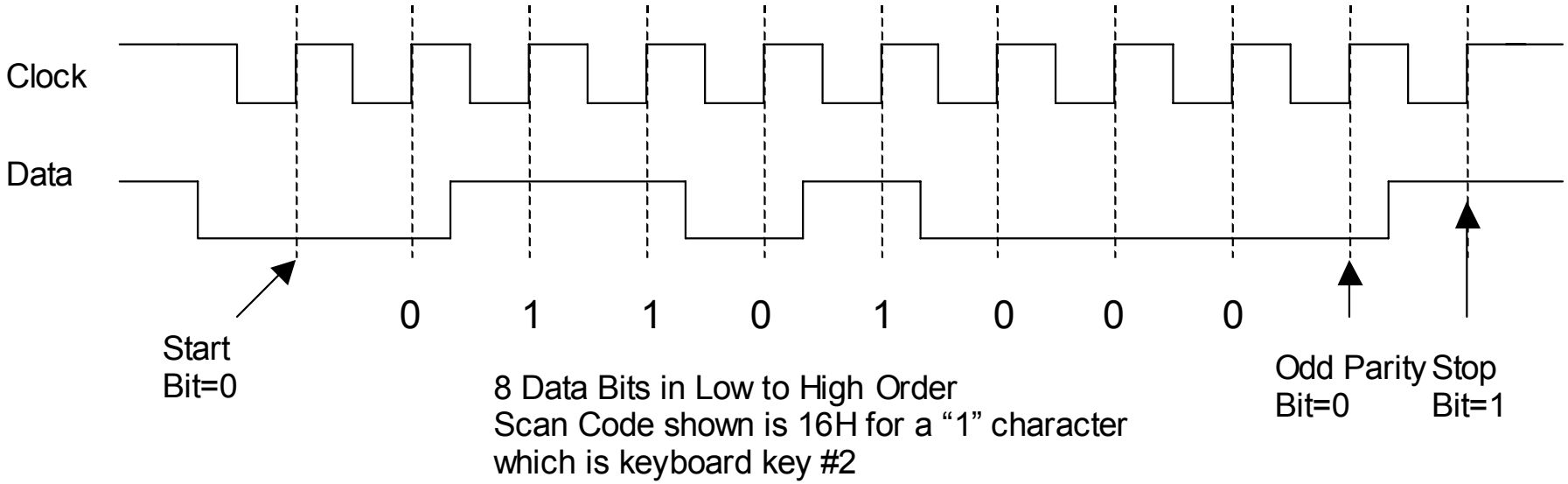


Figure 11.1 Keyboard Transmission of a Scan Code.

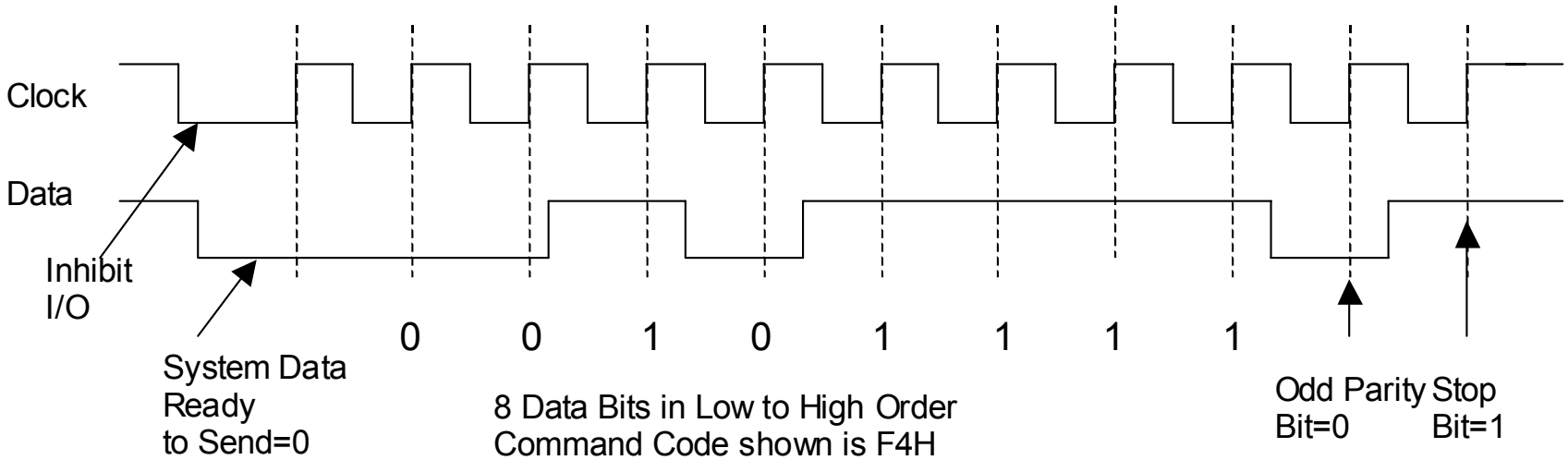


Figure 11.2 System Transmission of a Command to PS/2 Device.

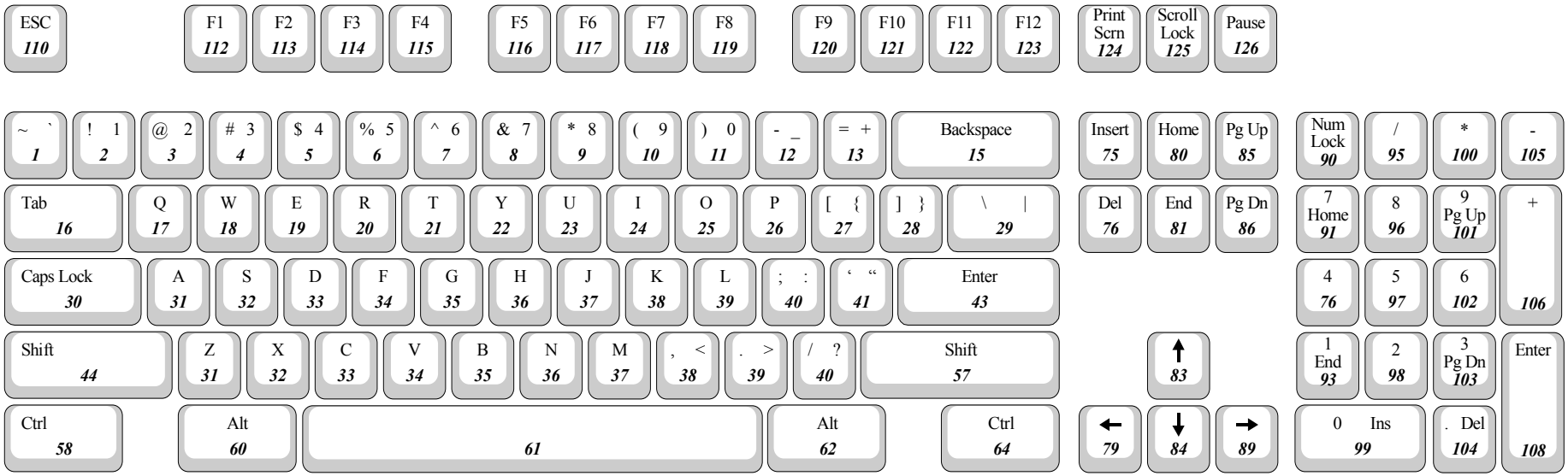


Figure 11.3 Key Numbers for Scan Code.

Table 11.3 Scan Codes for PS/2 Keyboard.

Key#	Make Code	Break Code	Key#	Make Code	Break Code	Key#	Make Code	Break Code
1	0E	F0 0E	31	1C	F0 1C	90	77	F0 77
2	16	F0 16	32	1B	F0 1B	91	6C	F0 6C
3	1E	F0 1E	33	23	F0 23	92	6B	F0 6B
4	26	F0 26	34	2B	F0 2B	93	69	F0 69
5	25	F0 25	35	34	F0 34	96	75	F0 75
6	2E	F0 2E	36	33	F0 33	97	73	F0 73
7	36	F0 36	37	3B	F0 3B	98	72	F0 72
8	3D	F0 3D	38	42	F0 42	99	70	F0 70
9	3E	F0 3E	39	4B	F0 4B	100	7C	F0 7C
10	46	F0 46	40	4C	F0 4C	101	7D	F0 7D
11	45	F0 45	41	52	F0 52	102	74	F0 74
12	4E	F0 4E	43	5A	F0 5A	103	7A	F0 7A
13	55	F0 55	44	12	F0 12	104	71	F0 71
15	66	F0 66	46	1A	F0 1A	105	7B	F0 7B
16	0D	F0 0D	47	22	F0 22	106	79	F0 79
17	15	F0 15	48	21	F0 21	110	76	F0 76
18	1D	F0 1D	49	2A	F0 2A	112	05	F0 05
19	24	F0 24	50	32	F0 32	113	06	F0 06
20	2D	F0 2P	51	31	F0 31	114	04	F0 04
21	2C	F0 2C	52	3A	F0 3A	115	0c	F0 0C
22	35	F0 35	53	41	F0 41	116	03	F0 03
23	3C	F0 3C	54	49	F0 49	117	0B	F0 0B
24	43	F0 43	55	4A	F0 4A	118	83	F0 83
25	44	F0 44	57	59	F0 59	119	0A	F0 0A
26	4D	F0 4D	58	14	F0 14	120	01	F0 01
27	54	F0 54	60	11	F0 11	121	09	F0 09
28	5B	F0 5B	61	29	F0 29	122	78	F0 78
29	5D	F0 5D	62	E0 11	E0 F0 11	123	07	F0 07
The remaining key codes are a function of the shift, control, alt, or num-lock keys.								

Table 11.3 (Continued) - Scan Codes for PS/2 Keyboard.

Key	No Shift or Num Lock		Shift*		Num Lock On	
#	Make	Break	Make	Break	Make	Break
76	E0 70	E0 F0 70	E0 F0 12 E0 70	E0 F0 70 E0 12	E0 12 E0 70	E0 F0 70 E0 F0 12
76	E0 71	E0 F0 71	E0 F0 12 E0 71	E0 F0 71 E0 12	E0 12 E0 71	E0 F0 71 E0 T0 12
79	E0 6B	E0 F0 6B	E0 F0 12 E0 6B	E0 F0 6B E0 12	E0 12 E0 6B	E0 F0 6B E0 F0 12
80	E0 6C	E0 F0 6C	E0 F0 12 E0 6C	E0 F0 6C E0 12	E0 12 E0 6C	E0 F0 6C E0 F0 12
81	E0 69	E0 F0 69	E0 F0 12 E0 69	E0 F0 69 E0 12	E0 12 E0 69	E0 F0 69 E0 F0 12
83	E0 75	E0 F0 75	E0 F0 12 E0 75	E0 F0 75 E0 12	E0 12 E0 75	E0 F0 75 E0 F0 12
84	E0 72	E0 F0 72	E0 F0 12 E0 72	E0 F0 72 E0 12	E0 12 E0 72	E0 F0 72 E0 F0 12
85	E0 7D	E0 F0 7D	E0 F0 12 E0 7D	E0 F0 7D E0 12	E0 12 E0 7D	E0 F0 7D E0 F0 12
86	E0 7A	E0 F0 7A	E0 F0 12 E0 7A	E0 F0 7A E0 12	E0 12 E0 7A	E0 F0 7A E0 F0 12
89	E0 74	E0 F0 74	E0 F0 12 E0 74	E0 F0 74 E0 12	E0 12 E0 74	E0 F0 74 E0 F0 12

* When the left Shift Key is held down, the 12 - FO 12 shift make and break is sent with the other scan codes. When the right Shift Key is held down, 59 – FO 59 is sent.

Key	Scan Code		Shift Case *	
#	Make	Break	Make	Break
95	E0 4A	E0 F0 4A	E0 F0 12 E0 4A	E0 12 F0 4A

* When the left Shift Key is held down, the 12 - FO 12 shift make and break is sent with the other scan codes. When the right Shift Key is held down, 59 - FO 59 is sent. When both Shift Keys are down, both sets of codes are sent with the other scan codes.

Key	Scan Code		Control Case, Shift Case		Alt Case	
#	Make	Break	Make	Break	Make	Break
124	E0 12 E0 7C	E0 F0 7C E0 F0 I2	E0 7C	E0 F0 7C	84	F0 84

Key #	Make Code	Control Key Pressed
126 *	El 14 77 El F0 14 F0 77	E0 7E E0 F0 7E

* This key does not repeat

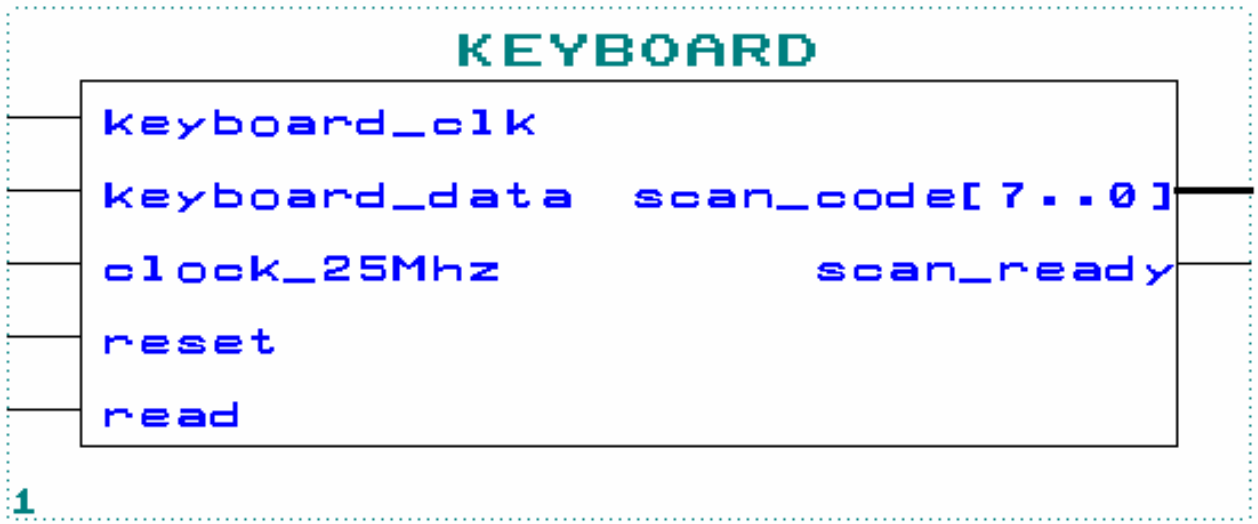


Figure 11.4 Keyboard UP1core

```
LIBRARY IEEE;
USE  IEEE.STD_LOGIC_1164.ALL;
USE  IEEE.STD_LOGIC_ARITH.ALL;
USE  IEEE.STD_LOGIC_UNSIGNED.ALL;

ENTITY keyboard IS
    PORT(   keyboard_clk, keyboard_data, clock_25Mhz ,
            reset, read         : IN    STD_LOGIC;
            scan_code           : OUT   STD_LOGIC_VECTOR( 7 DOWNTO 0 );
            scan_ready          : OUT   STD_LOGIC);
END keyboard;

ARCHITECTURE a OF keyboard IS
    SIGNAL INCNT                : STD_LOGIC_VECTOR( 3 DOWNTO 0 );
    SIGNAL SHIFTIN              : STD_LOGIC_VECTOR( 8 DOWNTO 0 );
    SIGNAL READ_CHAR            : STD_LOGIC;
    SIGNAL INFLAG, ready_set    : STD_LOGIC;
    SIGNAL keyboard_clk_filtered : STD_LOGIC;
    SIGNAL filter               : STD_LOGIC_VECTOR( 7 DOWNTO 0 );

BEGIN
    PROCESS ( read, ready_set )
    BEGIN
        IF read = '1' THEN
           scan_ready <= '0';
        ELSIF ready_set'EVENT AND ready_set = '1' THEN
           scan_ready <= '1';
        END IF;
    END PROCESS;
                        --This process filters the raw clock signal coming from the
                        -- keyboard using a shift register and two AND gates
Clock_filter:
    PROCESS
        BEGIN
        WAIT UNTIL clock_25Mhz'EVENT AND clock_25Mhz = '1';
        filter ( 6 DOWNTO 0 ) <= filter( 7 DOWNTO 1 ) ;
        filter( 7 ) <= keyboard_clk;
        IF filter = "11111111" THEN
           keyboard_clk_filtered <= '1';
        ELSIF filter = "00000000" THEN
           keyboard_clk_filtered <= '0';
        END IF;
    END PROCESS Clock_filter;
```

```
--This process reads in serial scan code data coming from the keyboard
    PROCESS
    BEGIN
      WAIT UNTIL (KEYBOARD_CLK_filtered'EVENT AND KEYBOARD_CLK_filtered = '1');
      IF RESET = '1' THEN
          INCNT  <= "0000";
          READ_CHAR    <= '0';
      ELSE
          IF KEYBOARD_DATA = '0' AND READ_CHAR = '0' THEN
              READ_CHAR    <= '1';
              ready_set    <= '0';
          ELSE
                          -- Shift in next 8 data bits to assemble a scan code
              IF READ_CHAR = '1' THEN
                  IF INCNT < "1001" THEN
                      INCNT   <= INCNT + 1;
                      SHIFTIN( 7 DOWNTO 0 ) <= SHIFTIN( 8 DOWNTO 1 );
                      SHIFTIN( 8 )      <= KEYBOARD_DATA;
                      ready_set    <= '0';
                          -- End of scan code character, so set flags and exit loop
                  ELSE
                      scan_code  <= SHIFTIN( 7 DOWNTO 0 );
                      READ_CHAR    <='0';
                      ready_set    <= '1';
                      INCNT  <= "0000";
                  END IF;
              END IF;
          END IF;
      END IF;
    END PROCESS;
END a;
```

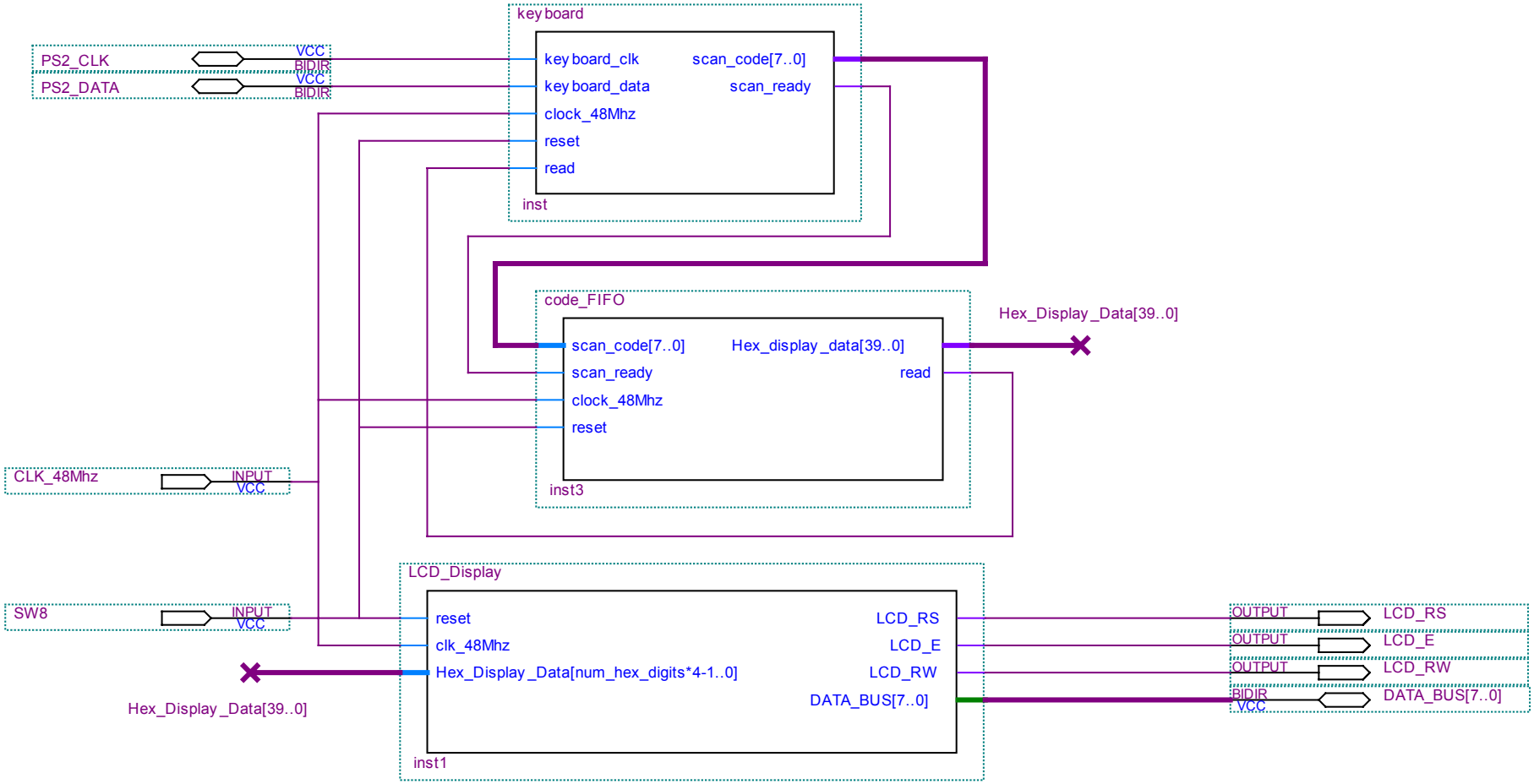


Figure 11.5 Example design using the Keyboard UP3core.

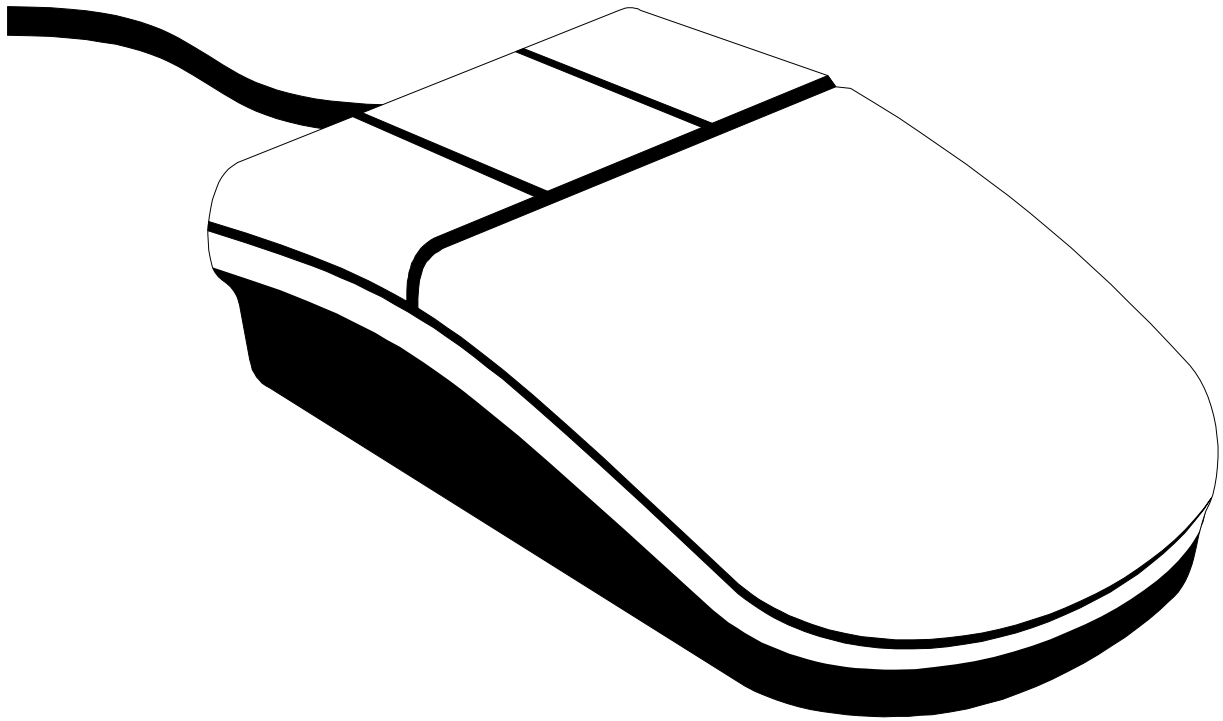


Table 11.4 PS/2 Mouse Commands.

Commands Sent to Mouse	Hex Value
Reset Mouse	FF
Mouse returns AA, 00 after self-test	
Resend Message	FE
Set to Default Values	F6
Enable Streaming Mode Mouse starts sending data packets at default rate	F4
Disable Streaming Mode	F5
Set sampling rate XX is number of packets per second	F3, XX
Read Device Type	F2
Set Remote Mode	EE
Set Wrap Mode Mouse returns data sent by system	EC
Read Remote Data Mouse sends 1 data packet	EB
Set Stream Mode	EA
Status Request Mouse returns 3-bytes with current settings	E9
Set Resolution XX is 0, 1, 2, 3	E8, XX
Set Scaling 2 to 1	E7
Reset Scaling	E6

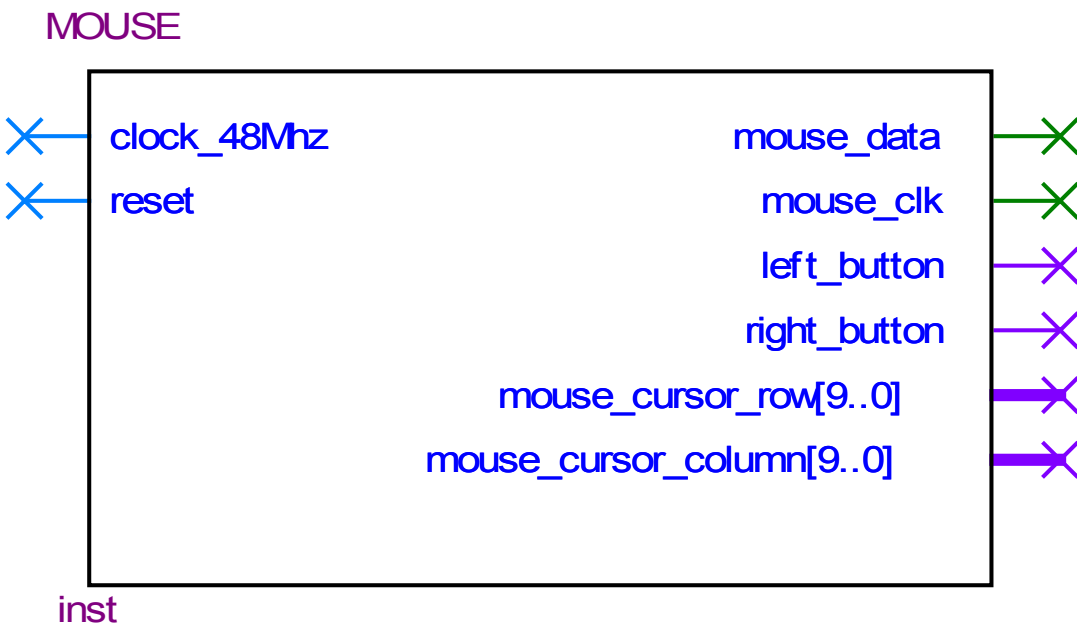
Table 11.5 PS/2 Mouse Messages.

Messages Sent by Mouse	Hex Value
Resend Message	FE
Two bad messages in a row	FC
Mouse Acknowledge Command Sent by Mouse after each command byte	FA
Mouse passed self-test	AA

Table 11.6 PS/2 Mouse Data Packet Format.

	MSB							**LSB**
Bit	7	6	5	4	3	2	1	0
Byte 1	*Yo*	*Xo*	*Ys*	*Xs*	1	M	R	L
Byte 2	X7	X6	X5	X4	X3	X2	X1	X0
Byte 3	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0

L = Left Key Status bit (For buttons 1 = Pressed and 0 = Released)
M = Middle Key Status bit (This bit is reserved in the standard PS/2 mouse protocol, but some three button mice use the bit for middle button status.)
R = Right Key Status bit
X7 – X0 = Moving distance of X in two's complement
(Moving Left = Negative; Moving Right = Positive)
Y7 – Y0 = Moving distance of Y in two's complement
(Moving Up = Positive; Moving Down = Negative)
Xo = X Data Overflow bit (1 = Overflow)
Yo = Y Data Overflow bit (1 = Overflow)
Xs = X Data sign bit (1 = Negative)
Ys = Y Data sign bit (1 = Negative)



The Mouse UP3core

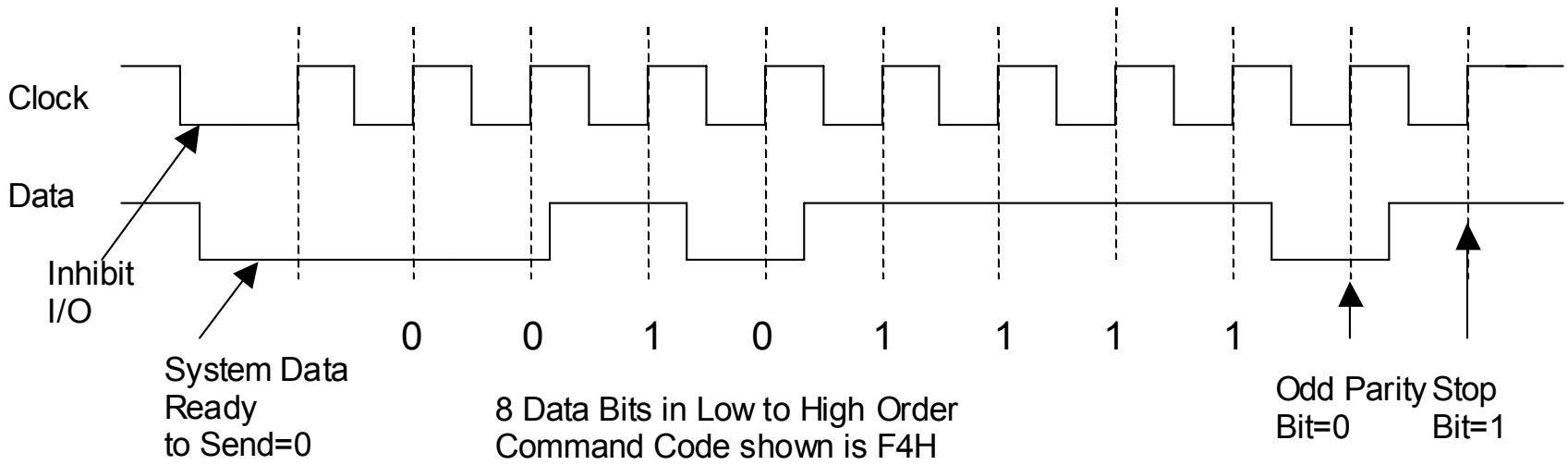


Figure 11.6 Transmission of Mouse Initialization Command.

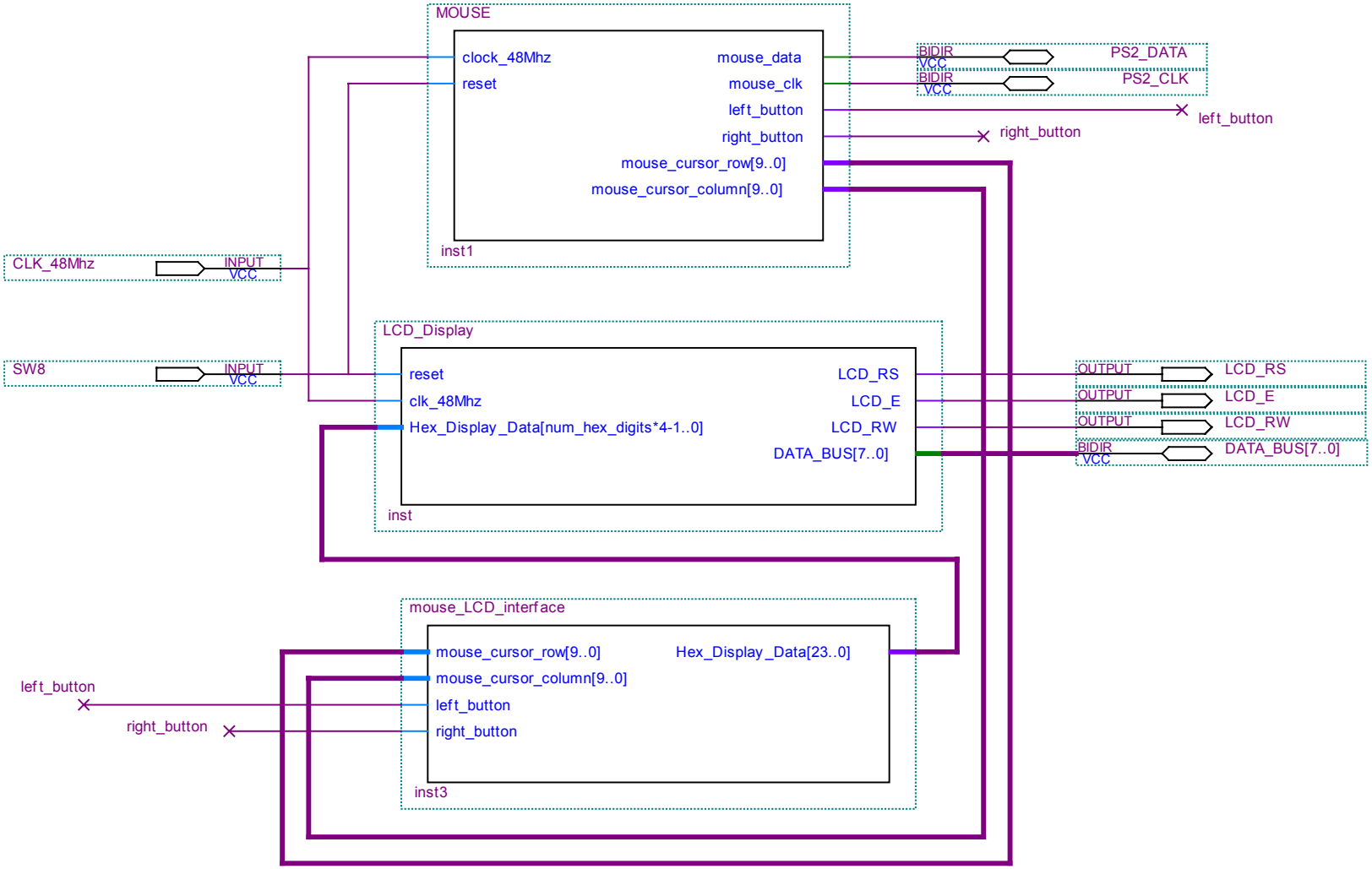


Figure 11.7 Example Design Using Mouse UP3core