High Data Rate
Wireless USB
Optical Mouse
Solution Using the
MC68HC908QY4
and MC68HC908JB12
Designer Reference
Manual

M68HC08
Microcontrollers

For More Information On This Product,
Go to: www.freescale.com
Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com
High Data Rate Wireless USB Optical Mouse Solution Using the MC68HC908QY4 and MC68HC908JB12
Designer Reference Manual

by: Dennis Lui
   Ernest Chan
   Applications Engineering
   Motorola Microcontroller Division
   Hong Kong

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to: http://motorola.com/semiconductors

Motorola and the Stylized M Logo are registered trademarks of Motorola, Inc.
DigitalDNA is a trademark of Motorola, Inc.
This product incorporates SuperFlash® technology licensed from SST.
All brand names and product names appearing in this document are registered trademarks or trademarks of their respective holders.

© Motorola, Inc., 2004

High Data Rate Wireless USB Optical Mouse Solution

For More Information On This Product, Go to: www.freescale.com
### Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision Level</th>
<th>Description</th>
<th>Page Number(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>January, 2004</td>
<td>N/A</td>
<td>Initial release</td>
<td>N/A</td>
</tr>
</tbody>
</table>

For More Information On This Product, Go to: [www.freescale.com](http://www.freescale.com)
Table of Contents

Section 1. System Overview
1.1 Introduction ............................................. 9
1.2 Features .................................................. 10
1.3 System Overview .......................................... 10
1.4 Transmitter and Receiver ................................. 11

Section 2. RF Front End
2.1 Introduction ............................................. 13
2.2 Functional Description ................................... 13
2.3 RF Transmitter ........................................... 13
2.4 RF Receiver .............................................. 14
2.5 PCB Layout Guidelines ................................. 14

Section 3. Universal Serial Bus (USB) Overview
3.1 Introduction ............................................. 15
3.2 JB12 USB Module ...................................... 15
3.2.1 USB Module Description ............................ 16
3.2.2 USB Connection ..................................... 17
3.2.3 USB Protocol ......................................... 17

Section 4. Optical Mouse Transmitter
4.1 Introduction ............................................. 19
4.2 System Overview ........................................ 19
4.2.1 QY4 Microcontroller ............................... 20
4.2.2 Optical Mouse Sensor ............................... 20

Section 5. Mouse Transmitter Firmware
5.1 Firmware Structure ..................................... 21
5.2 Power Management .................................... 22
5.3 Data Packet Format .................................... 23
5.3.1 Coding Method ..................................... 23
5.3.2 Packet Types ...................................... 23
5.3.3 Button Status and Z Displacement Packet ....... 24
5.3.4 X-Y Displacements Packet ......................... 25
## Table of Contents

**Section 6. Universal Serial Bus (USB) Receiver**

6.1 Introduction ................................................. 27  
6.2 System Overview .......................................... 28

**Section 7. Receiver Firmware**

7.1 Firmware Structure ........................................... 29  
7.2 USB Report .................................................. 35 
7.3 Remote Wakeup .............................................. 35 
7.3.1 Wakeup Detection Mechanism ............................... 35 
7.3.2 Power Consideration ...................................... 37 
7.4 ID Updating Process ......................................... 37

**Section 8. Testing and Customization**

8.1 Testing ......................................................... 39  
8.2 Hardware Customization ...................................... 39 
8.2.1 Optical Mouse Transmitter ................................. 39 
8.2.2 RF Circuitry ................................................ 41 
8.3 Firmware Customization ...................................... 42 
8.3.1 Mouse Transmitter ......................................... 42 
8.3.2 Receiver .................................................... 42
### List of Figures and Tables

<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>System Block Diagram</td>
<td>11</td>
</tr>
<tr>
<td>3-1</td>
<td>USB Block Diagram</td>
<td>16</td>
</tr>
<tr>
<td>3-2</td>
<td>USB Connection</td>
<td>17</td>
</tr>
<tr>
<td>3-3</td>
<td>USB Packet Types</td>
<td>18</td>
</tr>
<tr>
<td>4-1</td>
<td>Transmitter Block Diagram</td>
<td>20</td>
</tr>
<tr>
<td>5-1</td>
<td>Firmware Structure</td>
<td>21</td>
</tr>
<tr>
<td>5-2</td>
<td>Power Management</td>
<td>22</td>
</tr>
<tr>
<td>5-3</td>
<td>Manchester Coding</td>
<td>23</td>
</tr>
<tr>
<td>5-4</td>
<td>Packet Format for Button and Z Displacement</td>
<td>24</td>
</tr>
<tr>
<td>5-5</td>
<td>Start Field</td>
<td>24</td>
</tr>
<tr>
<td>5-6</td>
<td>Button Status Byte</td>
<td>24</td>
</tr>
<tr>
<td>5-7</td>
<td>Z Displacement or ID Byte</td>
<td>24</td>
</tr>
<tr>
<td>5-8</td>
<td>Checksum Byte</td>
<td>24</td>
</tr>
<tr>
<td>5-9</td>
<td>Packet Format for X-Y Displacements</td>
<td>25</td>
</tr>
<tr>
<td>5-10</td>
<td>Start Field for X-Y Displacement Packet</td>
<td>25</td>
</tr>
<tr>
<td>6-1</td>
<td>USB Receiver</td>
<td>27</td>
</tr>
<tr>
<td>7-1</td>
<td>Firmware Flow</td>
<td>30</td>
</tr>
<tr>
<td>7-2</td>
<td>USB Interrupt Routine</td>
<td>31</td>
</tr>
<tr>
<td>7-3</td>
<td>Setup Routine</td>
<td>32</td>
</tr>
<tr>
<td>7-4</td>
<td>OUT EP0 Handler</td>
<td>32</td>
</tr>
<tr>
<td>7-5</td>
<td>IN EP0 Handler</td>
<td>33</td>
</tr>
<tr>
<td>7-6</td>
<td>Timer Capture Interrupt For Receiver</td>
<td>34</td>
</tr>
<tr>
<td>7-7</td>
<td>Timing Parameters for Packet Detection</td>
<td>36</td>
</tr>
<tr>
<td>7-8</td>
<td>Detection of Valid Wakeup Packets</td>
<td>36</td>
</tr>
<tr>
<td>7-9</td>
<td>JB12 Rejecting Packet-Like Noise to Prevent False Wakeup</td>
<td>37</td>
</tr>
<tr>
<td>8-1</td>
<td>USB Report for JB12 to Host for the Test Setup</td>
<td>40</td>
</tr>
<tr>
<td>8-2</td>
<td>Loop Antenna Impedance</td>
<td>41</td>
</tr>
</tbody>
</table>
## List of Figures and Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7-1</td>
<td>Interface 1 Mouse Input Report</td>
<td>35</td>
</tr>
<tr>
<td>8-1</td>
<td>Tx Matching Examples</td>
<td>41</td>
</tr>
<tr>
<td>8-2</td>
<td>Rx Matching Examples</td>
<td>41</td>
</tr>
</tbody>
</table>
Section 1. System Overview

1.1 Introduction

This manual describes a reference design for a high data rate wireless universal serial bus (USB) optical mouse solution by using the MC68HC908QY4 and the MC68HC908JB12.

Because of the lower cost, adoption of wireless PC peripherals has been fast paced. Consumers are quickly recognizing the advantages of using wireless PC peripherals, which include portability and placement flexibility. However, there are two major performance issues in general 27-MHz wireless platforms. These issues affect wireless mouse applications in particular.

1. The mouse cursor motion tracking speed is slow
2. There is a long legacy time in response to an end-user action

These issues can be resolved by implementation of a new high-speed transmission link. The tracking speed of the high data rate wireless mouse is almost doubled and the response time is reduced by half. The performance is even compatible to a cabled mouse in terms of data report rates uploaded to the PC host from the peripheral device. From the end users point of view, the overall movement of the mouse can be fully cached up and reflected on the computer screen without any noticeable delay in cursor response.

The following reference materials are available at:
http://motorola.com/semiconductors

- Schematic diagram
- PCB layout
- PCB Gerber
- Bill of materials
- Source code
1.2 Features

Main features of the mouse include:
- A high data rate 27-MHz RF link
- Windows® 98, Windows 2000, and Windows XP compatibility
- User identity code to avoid conflict with other devices
- USB 2.0 low-speed compliance
- 4.8 kbps transmission data rate
- 2 meter communication distance
- 3361 compatible device for RF receiver design

1.3 System Overview

The system consists of:
- A high data rate wireless optical mouse using the MC68HC908QY4 (hereafter referred as QY4)
- A USB receiver using the MC68HC908JB12 (hereafter referred as JB12).

The QY4 was chosen as the mouse transmitter because it includes:
- An internal oscillator circuit
- An auto wakeup function

The JB12 is suitable for this high data rate wireless optical mouse application because it has:
- 12-K FLASH memory
- 6-MHz bus processing power
- USB function
- An enhanced timer capture module

**NOTE:** The traditional track-ball type X-Y detection method was replaced by the advance optical navigation technology that serves as a non-mechanical motion estimation.

A block diagram of the system is shown in [Figure 1-1](#).
1.4 Transmitter and Receiver

On the transmitter side:
The data generated from a displacement detection / button status in the mouse application is encoded with a pre-defined serial type protocol handled by firmware in the MCU. In the RF stage, the encoded data is used for FSK modulation.

On the receiver side:
The captured data from the RF receiver stage is decoded with a corresponding packet format used for mouse applications. The final data is sent to the host through the USB interface.
Section 2. RF Front End

2.1 Introduction

A high data rate 27-MHz RF link is designed as the wireless communication media for this application. The RF frequency is determined by the crystal frequency used at the oscillator circuit stage and there are two selectable channels for transmission. The transmission data rate is 4.8 kbps.

2.2 Functional Description

The high-frequency carrier signal on the transmitter side is modulated by the digital encoded data from the QY4 using a FSK modulation scheme. The modulated RF signal is propagated through free-air space and received by an integrated chip, the 3361, on the receiver side which includes all mixer, local oscillator, and demodulator circuits. The demodulated data output is received by the JB12 for decoding and processing. The data will then be converted to the USB mouse report format and sent to the host.

2.3 RF Transmitter

The RF transmitter consists of three parts:

- The crystal type oscillator
- The FSK modulation switching circuit
- The RF amplifier

The crystal oscillator works with a crystal frequency at half of the target channel frequency and the second harmonic frequency is filtered out by the RF amplifier together with a high Q-factor antenna. For example, a 13.5225-MHz crystal is used for a frequency channel at 27.045-MHz.

The FSK modulation is achieved by changing the loading capacitance at the crystal with a transistor switching circuit controlled by the encoding data generated from the MCU. The maximum data rate for a particular FSK transmission is limited by the RF bandwidth of the system and controlled by the frequency deviation which represents the logic “0” and logic “1” data. In general, the frequency deviation should be adjusted proportionally to the change on the required transmission data rate. Higher data rates require more bandwidth / frequency deviation. The frequency deviation for this 4.8-kbps data
rate wireless mouse application is increased from ±2.5 kHz to ±4.5 kHz which is controlled by the crystal characteristics.

The gain of the RF amplifier in the final stage should be adjusted to compensate the gain loss at the oscillator stage with extend bandwidth operation. Two stages of RF amplifier are used in this reference design to maintain the performance in communication distance.

2.4 RF Receiver

The RF receiver is implemented by using a single-chip solution (3361 compatible part) which includes:

- A frequency downward conversion mixer
- A local oscillator circuit
- A baseband FSK quadrature demodulation unit

The RF input signal from the antenna, is frequency down converted into an IF signal at 455 kHz by the mixer and oscillator circuits. The IF frequency value is equal to the RF input frequency plus or minus the LO input frequency. The higher frequency components should be filtered out by using a passive IF filter. The bandwidth should be increased from 15 kHz to 20 kHz to match the data rate change on the transmitter side.

However, the image frequency component would not be filtered out by the IF filter. This should be considered in the PCB layout in order to prevent any noise component at image frequency to be injected into the mixer input. An example of this would be a noise pattern generated from the MCU.

The frequency used for a 3361 mixer local oscillator is selected by the crystal connected at the oscillator base input pin. There are two frequency channels that can be selected to match the transmitter channel.

2.5 PCB Layout Guidelines

Care should be taken in PCB layout in order to avoid any noise generated from MCU coupling into the RF stage. For example:

- The power supply traces used for digital and analog circuit blocks should be separated.
- The location of decoupling capacitors should be as close as possible to device’s supply input pins (V_{DD}/V_{SS} or V_{CC}/GND).
- The V_{DD} to V_{SS} ground loop area should be reduced to minimize the magnetic coupling effect.
- The PCB trace loop formed by any input/output (I/O) signal pin should be kept to a minimum.
- The RF receiver uses a loop antenna formed by using a PCB trace line.
Section 3. Universal Serial Bus (USB) Overview

3.1 Introduction

The universal serial bus (USB) is an industry-standard extension to PC architecture providing a low-cost plug-and-play solution for PC peripheral devices. It is a serial data link with a high data-transfer rate and device-control capability. Peripheral USB devices can be configured automatically when connected to a host because the USB software driver is mapped and loaded in the operating system (OS) according to the peripheral device class.

The USB driver knows how to communicate with the devices and the USB devices will report their attributes (using a specific report format called descriptor) to the host during device configuration. The descriptor is a data structure with a defined format that describes the device's capabilities and resource requirements.

For more detailed information regarding the descriptor format, please refer to the Universal Serial Bus Specification Revision 2.0 at:
http://www.usb.org/developers/docs

3.2 JB12 USB Module

Features of the JB12 USB module include:

- Universal Serial Bus Specification 2.0 low-speed functions
- 1.5-Mbps data rate
- On-chip 3.3-V regulator
- Endpoint 0 with 8-byte transmit buffer and 8-byte receive buffer
- Endpoint 1 with 8-byte transmit buffer
- Endpoint 2 with 8-byte transmit buffer and 8-byte receive buffer
- USB data control logic
- USB reset options
- Suspend and resume operations with remote wakeup support
- USB-generated interrupts
- STALL, NAK, and ACK handshake generation
3.2.1 USB Module Description

Figure 3-1 shows a block diagram of the USB module. The USB module manages communications between the host and the USB function. The module is partitioned into three functional blocks. These blocks consist of:

- USB transceiver
- USB control logic
- USB registers

![USB Block Diagram](image)

The USB transceiver provides the physical interface to the USB D+ and D− data lines. The USB transceiver uses a differential output driver to drive the USB data signal onto the USB cable. The output swings between the differential high and low state are well balanced to minimize signal skew.

The USB control logic manages data movement between the CPU and the transceiver. The control logic handles both transmit and receive operations on the USB. It contains the logic used to manipulate the transceiver and the endpoint registers.

The MCU program controls and monitors the USB operation and data transfer status through the corresponding USB registers.

Please refer to the *MC68HC908JB16 Technical Data* (Motorola document order number MC68HC908JB16/D) and to its Addendum (Motorola document order number HC908JB16AD/D) for a more detailed description of each block.
3.2.2 USB Connection

The USB devices are connected to the host by a 4-wire cable. The USB signals are transmitted over two wires (D+ and D–) on each point-to-point segment. The cable also carries VBUS and GND wires (+5 V and ground) to deliver power to devices. The USB data lines are required by the USB specification to have an output voltage between 2.8 V and 3.6 V. The data lines also are required to have an external 1.5-kΩ pullup resistor connected between a data line and a voltage source between 3.0 V and 3.6 V. Figure 3-2 shows the worst-case electrical connection for the voltage regulator.

![Figure 3-2. USB Connection](image)

For detailed electrical specifications, please refer to *Universal Serial Bus Specification Revision 2.0* at: [http://www.usb.org/developers/docs](http://www.usb.org/developers/docs)

3.2.3 USB Protocol

The USB is a polled bus and the physical layers are configured in a tiered star topology with point-to-point connections between the host and a hub or a device. All data transfers are initiated and controlled by the host on a scheduled basis. Each transaction is comprised of up to three packets. Figure 3-3 shows the USB packet types.

The host will send out a token packet describing the type and direction of transaction, device address, and endpoint number. The target device will respond to the token if the address field decoding is a match. The transaction direction is indicated in the token packet. The source of the transaction sends a data packet, or indicates there is no information to transfer, and the destination will respond with a handshake packet indicating if the transaction is correct or not.
## Universal Serial Bus (USB) Overview

### Token Packet:

```
<table>
<thead>
<tr>
<th></th>
<th>SYN</th>
<th>PID</th>
<th>PID</th>
<th>ADDR</th>
<th>ENDP</th>
<th>CRC5</th>
<th>EOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SETUP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

### Data Packet:

```
<table>
<thead>
<tr>
<th></th>
<th>SYN</th>
<th>PID</th>
<th>PID</th>
<th>DATA</th>
<th>CRC16</th>
<th>EOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATA1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

### Handshake Packet:

```
<table>
<thead>
<tr>
<th></th>
<th>SYN</th>
<th>PID</th>
<th>PID</th>
<th>EOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAK</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STALL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

**Figure 3-3. USB Packet Types**

Please refer to the *MC68HC908JB16 Technical Data* (Motorola document order number MC68HC908JB16/D) for a detailed description of each packet format.
Section 4. Optical Mouse Transmitter

4.1 Introduction

The features of the high data rate wireless USB optical mouse solution include an internal oscillator circuit which can generate a clock of 12.8 MHz with no external components needed. The auto wakeup module generates a periodic interrupt during stop mode to wake the part up without requiring an external signal. These features make the QY4 MCU suited for wireless optical mouse applications.

The main features of the reference design include:
- 27-MHz RF transmitter
- 4.8-kbps transmission data rate
- 800-DPI resolution
- Smart power management

4.2 System Overview

The mouse transmitter consists of:
- The QY4
- The Agilent optical mouse sensor (ADNS-2030)
- The RF front end

Refer to Figure 4-1 for a block diagram of the system.

**NOTE:** RF data is transmitted by means of setting and clearing the RF_Data and the RF_Off pin.
4.2.1 QY4 Microcontroller

The functions of the QY4 are:
- To get the XY displacement from the sensor
- Detect the Z displacement
- Check button status
- Control the RF circuitry to send out data
- Perform the overall power management

Three standard left, middle, and right buttons together with one button for the identity device (ID) code are implemented. The ID code can either be stored in the RAM or in the FLASH of the QY4. When the ID button in the transmitter and the one in the receiver are pressed, a random ID code is generated at the transmitter and sent to the receiver. After receiving the new ID code, the receiver stores it in the FLASH of the receiver MCU.

4.2.2 Optical Mouse Sensor

The ADNS-2030 is a 3-V supply sensor is specially design for wireless optical mouse applications. The communication between the sensor and the QY4 is through a serial peripheral interface (SPI) with clock input at the SCLK pin and bidirectional data interface at the SDIO pin. The power down (PD) pin is used to power down the sensor when not in use.
Section 5. Mouse Transmitter Firmware

5.1 Firmware Structure

The firmware structure consists of two main parts:

- Main routine
- Timer interrupt routine

Figure 5-1 shows the flow of the main program and the timer interrupt routine. It also indicates the main functions that the QY4 are to perform. The main challenge in wireless optical mouse design is the power management to minimize the power consumption and maximize the performance.

![Figure 5-1. Firmware Structure](image)

The main program continually checks one of the sensor registers to see if any XY movement has happened. If any XY movement is detected, it gets the X and Y displacements from the sensor registers, puts them in the FIFO buffer and sets the corresponding flags. For every millisecond timer tick, it checks the Z movement and the buttons status.
A timer interrupt is set for every 104 µs which is the base time for the 4.8-kHz data rate transmission. By configuring the timer to output compare mode, the RF_Data output pin can be set, cleared, or toggled for every 104 µs. The timer interrupt routine determines whether to set or clear the RF_Data pin at the next interrupt time. It also determines what the current RF_Off pin status should be.

### 5.2 Power Management

Power management plays a very important role in the wireless optical mouse solution.

**Figure 5-2. Power Management**

Figure 5-2 shows the power management flow. There are three defined stages:

- Power saving
- High current
- Sleep

After power up, the mouse is put in the power saving stage. In this stage, the sensor is only turned on every 20 milliseconds to see if any XY movement happened. The Z movement and buttons are sensed every millisecond.

If no activity happens for 90 seconds, it enters sleep stage. In this stage, the QY4 is put in stop mode and will wake up every 200 milliseconds to monitor any
activity. If still no activity happens for 10 minutes, the activities are monitored every 500 milliseconds.

Any XY movement will cause the mouse to enter high current stage. At this stage, the sensor is powered on and XY movement is continually monitored. The Z movement and button activities are still monitored every millisecond. If no XY movement happens for 5 seconds, it then enters power saving stage.

**NOTE:** Except for the 1 millisecond timer tick, all of the above mentioned timings can be configured by changing their constant values.

### 5.3 Data Packet Format

A data packet consists of a start field, a data field, and a checksum field.

#### 5.3.1 Coding Method

Except for the start field, both the data and the checksum fields are encoded by using the Manchester Coding method. That is, a logic '0' is represented by two equal times 'T' of a logic high or a logic low and vice verse for a logic '1'. An additional '0' is added to the end of each byte of these two fields as stop bits. Refer to **Figure 5-3**.

![Manchester Coding](image)

**Figure 5-3. Manchester Coding**

The order of transmission will be from least significant bit (LSB) to most significant bit (MSB). For example, bit 0 will be transmitted first.

The basic time of each logic level toggle is 104 $\mu$s (T). Therefore, each bit in the data and checksum fields will be a 2T period according to the Manchester Coding.

#### 5.3.2 Packet Types

There are two types of packets:
- X-Y displacements packet
- Button status and Z displacement packet

As the header patterns between these two packet types are different, the receiver will not interpret a X-Y displacement packet as a Z and button status packet or vice versa.
5.3.3 Button Status and Z Displacement Packet

Figure 5-4 shows the packet format for button and Z displacement.

<table>
<thead>
<tr>
<th>Start</th>
<th>Button Status</th>
<th>Z or ID</th>
<th>Checksum</th>
</tr>
</thead>
</table>

**Figure 5-4. Packet Format for Button and Z Displacement**

The start field consists of the SYNC pattern, preamble, and a header as shown in Figure 5-5.

**Figure 5-5. Start Field**

The button status byte (Figure 5-6) represents the status of the four buttons. It shows which buttons are pressed or released. The bit value of ‘1’ means the button was pressed and ‘0’ means the button was released.

**Figure 5-6. Button Status Byte**

This byte represents either the Z displacement or the new ID code (Figure 5-7). The Z displacement byte represents the Z displacement in 2's complement if the ID bit in the button status equals '0'. If the ID bit equals '1', it represents the new ID code.

**Figure 5-7. Z Displacement or ID Byte**

The checksum is the sum of the button status byte, the Z Displacement byte, and the stored ID byte. See Figure 5-8.

**Figure 5-8. Checksum Byte**
5.3.4 X-Y Displacements Packet

Figure 5-9 shows the packet format for X-Y displacements with the start field illustrations in Figure 5-10.

<table>
<thead>
<tr>
<th>Start</th>
<th>X Displacement</th>
<th>Y Displacement</th>
<th>Checksum</th>
</tr>
</thead>
</table>

![Figure 5-9. Packet Format for X-Y Displacements](image)

The X or Y displacement is represented in 2's complement and the checksum byte is the sum of the X displacement, the Y displacement, and the stored ID byte.
6.1 Introduction

The features of the JB12 include a configurable universal serial bus (USB) and PS/2 interface, which makes this MCU suited for personal computer human interface devices (HID) applications, such as mice. The enhanced timer function also allows it to capture and decode data easily, especially for high data rate wireless HID applications. Refer to Figure 6-1.

The main features of the receiver include:
- 27-MHz RF receiver
- Fully USB specification 2.0 low-speed compliant
- Windows 98, 2000, and XP compatible

![Figure 6-1. USB Receiver](image-url)
Universal Serial Bus (USB) Receiver

6.2 System Overview

The receiver consists of:

- The JB12
- A button for identity device (ID) code setting
- A in-circuit programming (ICP) connector
- RF front-end circuitry

The functions of the JB12 are to handle the USB transactions, capture and process data from RF receiver front end. The processed data is converted into USB report format and sent to the host.

The ID button is used to interrupt the MCU for a new ID code detection. The MCU will search if there is any ID code embedded in the RF data packet and store the new ID in FLASH memory.

There is a transistor wakeup circuit connected at the IRQ pin and is controlled by a I/O pin. The MCU will configure the wakeup circuit before entering stop mode for power saving. The RC components connected at the transistor base input will be charged up, and eventually turn on the transistor and pull low the IRQ to wakeup the MCU.

The ICP interface is connected at port A. The ICP is used for future firmware updating through the Cyclone Programming Tools.
7.1 Firmware Structure

The firmware consists of three main parts:

- Main routine
- Timer interrupt routine to capture and decode mouse data
- USB interrupt routine

The USB routine includes reference codes for both universal serial bus (USB) mouse and keyboard. Thus, it can be a reference on how to implement a composite USB keyboard-mouse device using the JB12.

Figure 7-1 shows the flow of the main program for the JB12 receiver. The main routine continually checks to see if there is any valid mouse data in the receiver buffer queue. If there is new data received, the data will be converted to USB report format and sent to the host via the USB endpoints.

Figure 7-2 shows the USB interrupt routines. The USB engine automatically responds to a valid USB token with either ACK, NAK, or STALL, depending on the register settings, and ignores it if it’s invalid. The firmware has to set the registers for the USB engine to give correct response to the token in different stages. The USB interrupt will be executed whenever there is an EOP, resume signal from host, valid data received, or data transmitted. The USB interrupt routine also makes preparation for the next USB transaction and handles any valid command or data received.

Figure 7-3, Figure 7-4, and Figure 7-5 show the routines handling control transfers. Control transfers have two or three transaction stages: setup, data (optional), and status as shown below:

- Control write: SETUP, OUT, OUT, OUT... IN
- Control read: SETUP, IN, IN, IN... OUT
- No Data control: SETUP, IN

The firmware first distinguishes the kind of control transfer and then does the corresponding preparation for the next stage.
Figure 7-1. Firmware Flow

1. **DEVICE CONFIGURED?**
   - NO: Cell A, YES: Cell B

2. **NEW RECEIVED MOUSE DATA IN BUFFER?**
   - NO: Cell C, YES: Cell D

3. **CONVERT DATA TO MOUSE REPORT**

4. **NEW ENDPOINT 1 REPORT?**
   - YES: Cell E, NO: Cell F

5. **EP1 TX BUFFER EMPTY?**
   - YES: Cell G, NO: Cell H

6. **TX EP1 IN REPORT**
   - YES: Cell I, NO: Cell J

7. **NEW ENDPOINT 2 REPORT?**
   - YES: Cell K, NO: Cell L

8. **EP2 TX BUFFER EMPTY?**
   - YES: Cell M, NO: Cell N

9. **TX EP2 IN REPORT**
   - YES: Cell O, NO: Cell P

10. **USB IDLE FOR 3 ms?**
    - YES: Cell Q, NO: Cell R

11. **SUSPEND DEVICE**
    - YES: Cell S, NO: Cell T

12. **RF PACKET DETECTED OR RESUME FROM HOST?**
    - YES: Cell U, NO: Cell V
Figure 7-2. USB Interrupt Routine
Figure 7-3. Setup Routine

- Setup Handler
  1. Uninstall EP0 IN and OUT
  2. Copy 8-byte setup data to RAM buffer
  3. Clear EP0 RX flag
  4. Set NAK to IN EP0

- Standard Device Request?
  - Yes: Handle Standard Device Request
  - No: Go to HID Class Request

- HID Class Request?
  - Yes: Handle HID Class Request
  - No: Return Stall

- Return Stall

- Return

Figure 7-4. OUT EP0 Handler

- Out EP0 Handler
  1. Set NAK to EP0 IN
  2. Set STALL to EP0 OUT (CONTROL TRANSFER COMPLETED)

- Status Stage?
  - Yes: 1. Copy data to buffer
  - No: 2. Process OUT data

- Valid Data?
  - Yes: 1. Copy data to buffer
  - No: 2. Process OUT data

- Return Stall

- Return
One timer interrupt routine is used to capture the RF mouse data. The firmware flow is shown Figure 7-6. This timer is set to input capture mode. When an interrupt happens on the falling and rising edge, it can calculate the pulse width of the RF data.

The routine will proceed to detect data only if the pulse width matches the data structure of the RF packet. The detected data bit will be put into a temporary buffer. The timer interrupt events will continue to occur until one complete RF packet has been received. All the received data will then be put into a receiver buffer queue for the main program to process. In the case where an USB bus is being suspended, a received mouse packet means the USB bus and the host computer needed to wake up. The routine will then signal the main routine to wakeup the host instead of update the receiver buffer queue.

For more detailed information, refer to 7.3.1 Wakeup Detection Mechanism.
Figure 7-6. Timer Capture Interrupt For Receiver
7.2 USB Report

As previously mentioned, the USB routine in the JB12 is a composite device of keyboard and mouse. It implements two HID interfaces on endpoint 1 and endpoint 2. HID interface 0 (endpoint 1) implements a standard HID keyboard with identical report and boot protocols. HID interface 1 (endpoint 2) implements multimedia, power management keys, and mouse data.

The mouse report uses report ID number 3 in the HID interface 1. The report structure is shown in Table 7-1.

Table 7-1. Interface 1 Mouse Input Report

<table>
<thead>
<tr>
<th>Byte</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Report ID = 3</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Middle button</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Right button</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Left button</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X displacement</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y displacement</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Z displacement</td>
</tr>
</tbody>
</table>

Reports with ID numbers 1 and 2 in interface 1 have been used for keyboard power management keys and multimedia keys for the keyboard protocol. If you are interested in the composite keyboard mouse device report format in detail, please refer to the reference design entitled USB Wireless Optical Mouse and Multimedia Keyboard Solution (Motorola document order number, DRM042/D).

7.3 Remote Wakeup

The JB12 receiver supports remote wakeup functions that can wake up the host computer during USB suspend.

During suspend, the MCU will be periodically awakened by the IRQ interrupt. The MCU then turns on the RF front end and detects whether valid mouse RF packets have arrived to wake up the host. This periodic IRQ interrupt signal is generated through the external RC charging and discharging circuit. The MCU initializes this charging and discharging cycle before it enters power saving mode.

7.3.1 Wakeup Detection Mechanism

After suspend, the MCU will wakeup for a short period of time for each IRQ interrupt. This period is shorter than one complete RF packet. In the case that an RF packet has arrived, the MCU can only determine that a portion of the packet is being received.
For this short detection period, there is the possibility that the noise hit into the RF front end would have a pattern like a packet portion. Therefore, if a packet portion is detected, the MCU will turn on for a longer duty cycle. This duty cycle is for receiving the next complete RF packet that can wake up the host.

The mechanism and the timing parameters for detecting a wakeup packet are shown in Figure 7-7, Figure 7-8, and Figure 7-9.

**Figure 7-7. Timing Parameters for Packet Detection**

- $t_{PKT}$: Length of one complete RF packet
- $t_{IRQ}$: Time interval of successive IRQ wakeup periods during device suspend.
- $t_{ON1}$: Duration of MCU being turned on in each IRQ wakeup period. MCU is turned on in this period to detect a (portion of) RF packet has been received.

**Figure 7-8. Detection of Valid Wakeup Packets**

If during a $t_{ON1}$ period a portion of a RF packet is detected, the RF receiver will turn on ($t_{ON2}$) for a longer time to detect a complete RF packet.

If one complete RF packet is received, the MCU will wake up the host.
7.3.2 Power Consideration

The average power consumption during suspend is given by the following equation:

\[ P_{\text{Average}} = \frac{P_{\text{ON}} \times t_{\text{ON}} + P_{\text{Sleep}} \times (t_{\text{IRQ}} - t_{\text{ON}})}{t_{\text{IRQ}}} \]

Power consumption can be decreased by increasing \( t_{\text{IRQ}} \). But, if \( t_{\text{IRQ}} \) is larger than \( t_{\text{PKT}} \) the receiver may not catch the incoming RF packet. Therefore, choosing \( t_{\text{IRQ}} \) equal to \( t_{\text{PKT}} \) could be an optimizing value for \( t_{\text{IRQ}} \).

Another way for reducing power consumption is to shorten \( t_{\text{ON}} \). But, \( t_{\text{ON}} \) cannot be too small in practice. Otherwise, the false wakeup for 2nd stage complete packet detection can occur more frequently resulting in more power being actually consumed (refer to Figure 7-9). Note that this parameter can be fine-tuned in the actual application system.

7.4 ID Updating Process

The user must follow these steps to update the mouse ID:
1. Press and hold the “PTA0” button in the receiver
2. Press the "CONNECT" button once in the mouse
3. Release the “PTA0” button in the receiver

Once these steps are complete, the new mouse ID will be updated in the receiver.
Receiver Firmware
8.1 Testing

This solution was tested under different Windows Operating Systems on several different PCs for USB compatibility. These tests included:

- USB compliance test using Command Verifier, Version 1.2
- Compatibility tests under Windows 98SE, 2000, and XP
- Compatibility tests under AMD 750, Intel 810 and 845 chip set desktops, and IBM Thinkpad T23, DELL Latitude C640

The integrity of the RF data link between the mouse and receiver had been verified by recording the actual USB report from the receiver to host.

A test had been setup where the mouse transmitter was sending some pre-defined RF data continuously. At the receiver side, an USB analyzer had been connected between the JB12 and the host computer. The analyzer captured the USB data from the JB12 to host PC. The captured data was then compared with pre-defined data from the mouse.

These tests showed that the JB12 solution can achieve an average valid USB data report rate of about 10 ms without loss of any data from the transmitter side. Refer to Figure 8-1.

8.2 Hardware Customization

8.2.1 Optical Mouse Transmitter

The step-up DC-to-DC converter (NCP1400ASN33T1) is for reference only, customers can choose any regulator they prefer. The LEDs for the sensor and ZLED can be connected to the regulator output or connected to the batteries output. The advantage of connecting to the regulator output is that the system can work in a lower voltage, but the drawback is higher current consumption. The advantage of connecting to the batteries output is the lower power consumption but the system will not work properly if battery voltage is below 2.5 V.
### USB Report for JB12 to Host for the Test Setup

<table>
<thead>
<tr>
<th>Field</th>
<th>IN</th>
<th>ADDR</th>
<th>ENDP</th>
<th>CRC5</th>
<th>EOP</th>
<th>TIME STAMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000001</td>
<td>0x96</td>
<td>2</td>
<td>2</td>
<td>0x01</td>
<td>1.867 µs</td>
<td>00003.7135 0035</td>
</tr>
<tr>
<td>00000001</td>
<td>0xD2</td>
<td>03</td>
<td>01</td>
<td>01</td>
<td>01</td>
<td>0xF3F5</td>
</tr>
<tr>
<td>00000001</td>
<td>0xC3</td>
<td>03</td>
<td>02</td>
<td>02</td>
<td>02</td>
<td>0xFCFA</td>
</tr>
<tr>
<td>00000001</td>
<td>0x4B</td>
<td>0x4B</td>
<td>1.867 µs</td>
<td>00003.7199 4803</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000001</td>
<td>0x96</td>
<td>2</td>
<td>2</td>
<td>0x01</td>
<td>1.867 µs</td>
<td>00003.7198 7391</td>
</tr>
<tr>
<td>00000001</td>
<td>0x96</td>
<td>0xD2</td>
<td>03</td>
<td>03</td>
<td>03</td>
<td>0xF6F0</td>
</tr>
<tr>
<td>00000001</td>
<td>0x96</td>
<td>0xC3</td>
<td>03</td>
<td>02</td>
<td>02</td>
<td>0xFCFA</td>
</tr>
</tbody>
</table>

**Figure 8-1. USB Report for JB12 to Host for the Test Setup**
8.2.2 RF Circuitry

If the PCB loop antenna is changed in terms of trace length or width, the corresponding loop antenna inductance, L, would be different and the matching network should be adjusted to maintain the maximum signal transfer condition.

![Figure 8-2. Loop Antenna Impedance](image)

On the transmitter side, the value for matching component C can be easily calculated by the following equation:

\[
2\pi f = \frac{1}{\sqrt{LC}}
\]

Where:

- \( f = 27 \text{ MHz} \)
- \( L = \text{Loop antenna inductance} \)

**Table 8-1. Tx Matching Examples**

<table>
<thead>
<tr>
<th>Case</th>
<th>L1</th>
<th>C = C14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smaller inductance</td>
<td>200</td>
<td>173 pF (~180 pF)</td>
</tr>
<tr>
<td>Original reference design</td>
<td>240</td>
<td>145 pF (~150 pF)</td>
</tr>
<tr>
<td>Larger inductance</td>
<td>300</td>
<td>116 pF (~120 pF)</td>
</tr>
</tbody>
</table>

On the receiver side, one of the simple methods to keep the matching condition for different inductance values is to adjust the capacitance value C such that the sum of L and C impedance \([XL + (-XC)]\) at 27 MHz is equal to a pre-matching impedance value \((Z4 = 26.3 \text{ j})\).

\[
Z4 = j\omega L + \frac{1}{j\omega C} \at 26.3 \text{j}
\]

**Table 8-2. Rx Matching Examples**

<table>
<thead>
<tr>
<th>Case</th>
<th>L2</th>
<th>C = C45 + C46</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smaller inductance</td>
<td>200 nH (34j)</td>
<td>765.5 pF (~7.7j)</td>
</tr>
<tr>
<td>Original reference design</td>
<td>236 nH (40j)</td>
<td>430 pF (~13.7j)</td>
</tr>
<tr>
<td>Larger inductance</td>
<td>300 nH (51j)</td>
<td>238.6 pF (~24.7j)</td>
</tr>
</tbody>
</table>
8.3 Firmware Customization

8.3.1 Mouse Transmitter

To customize the mouse transmitter:
- Set compiler option to store the ID code in RAM or in FLASH
- Set the timing parameters for power management

8.3.2 Receiver

To customize the receiver:
- Change vendor ID, product ID, and product revision number in the device descriptor table in "KBD-MSE.H"
- Change the report descriptor in "KBD-MSE.H" if necessary