# *IrDA Control Host Controller with USB Interface*

## *P/N: LZ85202*

*For*
*IrDA Control Host Device Implementation*

## Users' Manual

Version 1.00

March 23, 1999

SHARP CORPORATION

**(INFORMATION DISCLAIMER)**

This document is a “Users’ Manual” for the Sharp IrDA Control Host Controller with USB I/F (HC, P/N: LZ85202). Copying the whole content or any part of content of this document is strictly prohibited unless otherwise approved by written permission of Sharp Corporation. All information in this document is applicable for design reference purposes only, and is not regarded as device specifications.
Sharp holds the right to change or alter this information or prices of specified components at any time without notice.

In the absence of confirmation by device Specification Sheets, SHARP takes no responsibility for any defects that occur in equipment using any of SHARP’s devices, shown in catalogues, data books, Preliminary Information, etc. Contact SHARP, or SHARP local representatives in order to obtain the latest version of the device Specification Sheets before using any SHARP’s devices.

**LIFE SUPPORT POLICY**

SHARP components should not be used in medical devices with life support functions or in safety equipment (or similar applications where component failure may result in loss of life or physical harm) without the written approval of an officer of the SHARP Corporation.

SHARP reserves the right to make changes in specifications at any time and without notice. SHARP does not assume any responsibility for the use of any circuitry described; no circuit patent licenses are implied.

**Table Of Contents**

**Figures and Tables**

**Record Of Modification**

| Version | Date of Issue | Comments |
|---|---|---|
| 0.1 Preliminary | December 15, 1998 | First Edition (for Sharp Internal Evaluation Purpose ONLY) |
| 0.22 Preliminary | December 20, 1998 | First Edition (*"Preliminary"*, for Limited Public Release) |
| 0.26 Preliminary | January 20, 1999 | Following items added;<br>- Additional explanation of the Dongle Driver<br>- Host address<br>- DC bias mode |
| 0.90 Preliminary | February 23, 1999 | **First Release (for Public Distribution)** |
| 1.00 | March 23, 1999 | **Official Release (As Official Documentation)** |

# 1. IrDA Control Host Controller with USB I/F General Description

The Sharp IrDA Control Host Controller with USB I/F*1 (HC, Sharp P/N: LZ85202) is a communication controller with USB interface that fully supports IrDA Control communication services (reference document [1]). Using HC in conjunction with the Sharp IrDA Control Infrared Transceiver (P/N: GP2W2001YK), USB transceiver LSI, and serial EEPROM will implements IrDA Control Host device, to which up to 8 wireless peripherals, such as mice and keyboards in PC environment, can be connected for comfortable wireless operation environment.

*1 IrDA Control Host Controller with USB I/F … Represents "USB dongle *2 Controller for IrDA Control Host Implementation".

*2 USB dongle … An external adapter (hardware) can be connected to the external USB port of specific products, with which a new interface (IrDA interface) can be provided.

**HC Features:**

- Equipped with USB I/F to interface with PC
- All required functions for IrDA Control communication, Media Access Control (MAC) layer, Human Input Device (HID LLC) layer, IrDA Control to USB Bridge layer, and USB Controller are embedded in the HC.
- Optimized direct interface to Sharp IrDA Control Infrared Transceiver (Sharp P/N: GP2W2001YK).
- Optimized interface to USB transceiver LSI (P/N: PDIUSBP11A by PHILIPS, tolerant shifter required).
- Conforms to EP0, EP1, EP2 (for future extension), and EP3 as an IrDA Control Endpoint. A single HC supports maximum of 8 wireless links with IrDA Control Peripherals.

Figure 1.1 shows the HC interface signals. The HC signal lines are generally categorized to 2 types; interface signals with IrDA Control infrared transceiver, and USB interface signals for USB transceiver.

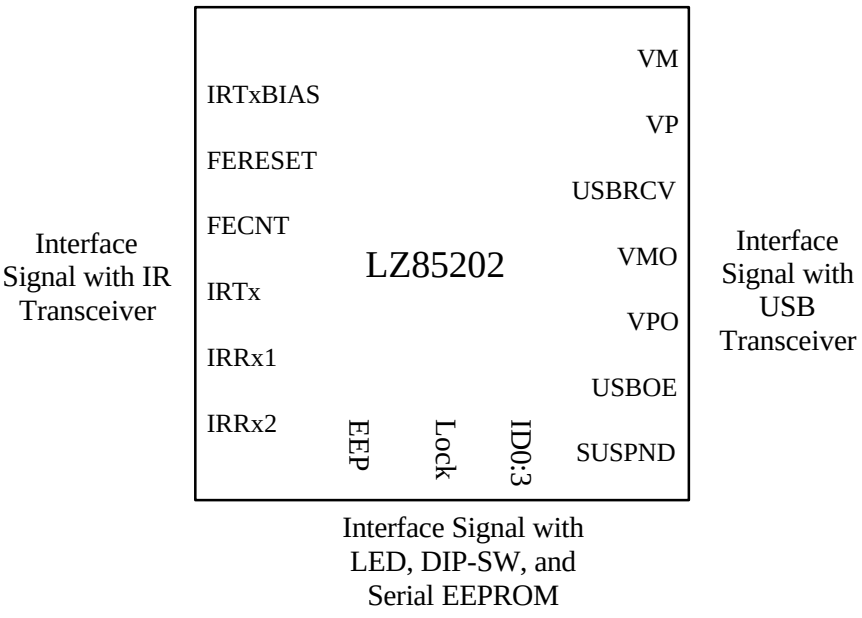



Figure 1.1. HC Interface Signals

# 2. IrDA Control System

## 2.1. General Description

The IrDA Control system consists of one (1) Host device and several Peripherals (up to 8 devices). Infrared data communication will occur between the Host device and the Peripherals. In the usual operating environment, the PC will perform as the Host device, and any input devices connected to the PC, such as a mouse and keyboard, gamepads or joysticks will act as Peripherals (Figure 2.1.). Other input devices can be developed within this model.

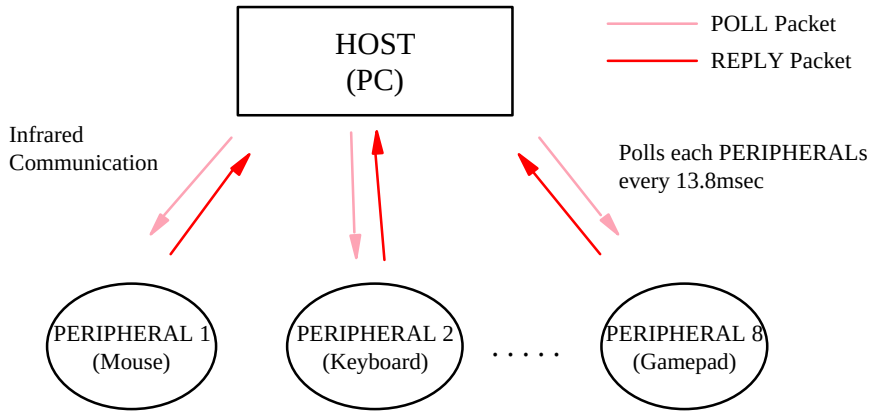



Figure 2.1. IrDA Control System

Figure 2.2 shows the IrDA Control system example, where the USB Dongle and IrDA Control Peripheral are required. In the IrDA Control physical layer (IrDA Control-PHY), the infrared link must provide a data communication path at a data rate of 75 kbps using a 16 PSM (Pulse Sequence Modulation Scheme) with 1.5 MHz subcarrier frequency.

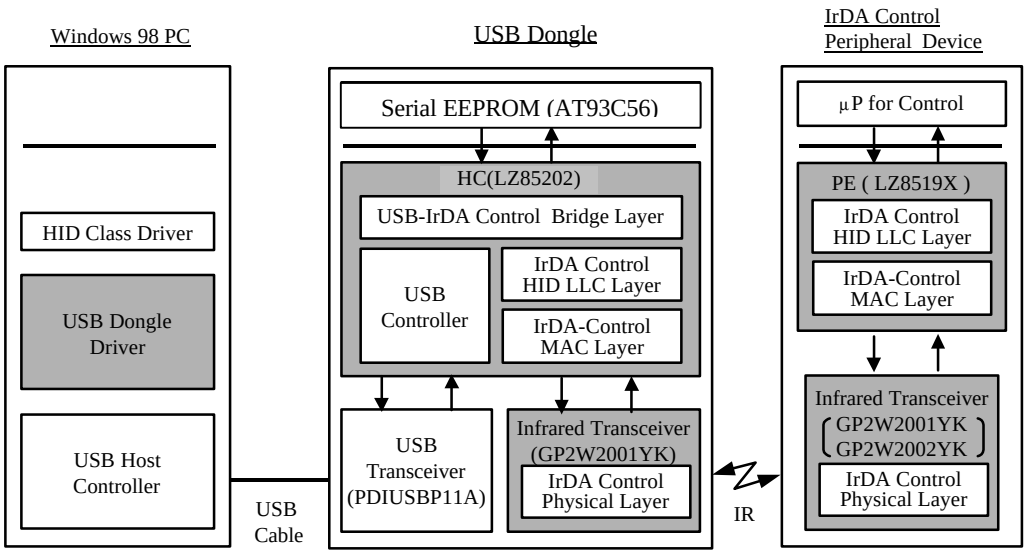



Figure 2.2. Required USB Dongle Functions for IrDA Control Subsystem

**USB Dongle**: The IrDA Control-PHY layer on the USB Dongle side can be implemented by the Sharp IrDA Control infrared transceiver (Sharp P/N: GP2W2001YK), as it is designed to provide full support for IrDA Control communication for the IrDA Control-PHY in the USB Dongle. Note that two assist LEDs are needed to ensure the FOV (Field Of View or Field Of Visibility) for the transmitter side required by the Host device, while the receiver of the transceiver meets the specification by itself.
The IrDA Control MAC layer (IrDA Control-MAC) manages the IrDA Control infrared wireless link between the Host device and the Peripherals. Actual activity of MAC layer is shown below.

1. The Host device will poll "Bound" Peripherals in a certain order. The Host device will insert the communication data (the information forwarded from Host HID LLC Layer) into the polling packet for each Peripheral.
2. The Host device will then wait for the corresponding response packets from the Peripherals. The received corresponding packets include the communication data from the Peripherals. The Host device will extract these data packets and forward them to its HID LLC layer.
3. In the event that the corresponding packets from the Peripheral are missing for a certain time period, the Host device will stop polling them. (This state will be seen when the user stops moving the Peripherals.) This state is expressed as "The Peripheral is in an **Unbound state**." On the other hand, while the Peripheral is polled, this state is expressed as "The Peripheral is in the **Bound state**." The Host device will immediately re-start polling the Peripheral when the Host device receives the IrDA Control infrared signal transmitted from the Unbound Peripherals. At this time it will again be **bound**. This step requires that the Peripheral was previously enumerated.

The HID LLC layer provides error correction, data re-sending, and flow management services between the USB Dongle and peripherals. The HID LLC layer formats information sent by the Peripherals to the Host device. The USB-IrDA Control bridge layer provides a data Communication Bridge between the USB data and the Peripheral data. It ensures interoperability between the USB data packets and the data packet sent by the IrDA Control peripherals. The USB controller enables the USB interface for communication with the PC.

As described above, 5 functions, which are PHY layer, MAC layer, HID LLC layer, USB-IrDA Control bridge layer, and USB controller, are required for the USB Dongle implementation.
The HC is embedded with all these 5 functions (the MAC layer and HID LLC layer, USB-IrDA Control bridge layer, and USB controller), enabling a simple single-chip implementation for the USB Dongle. Supporting devices are the Sharp IrDA Control Infrared transceiver, USB transceiver and a serial EEPROM. (See section 3 for details.) Figure 2.3 shows an example of IrDA Control Host device implementation.

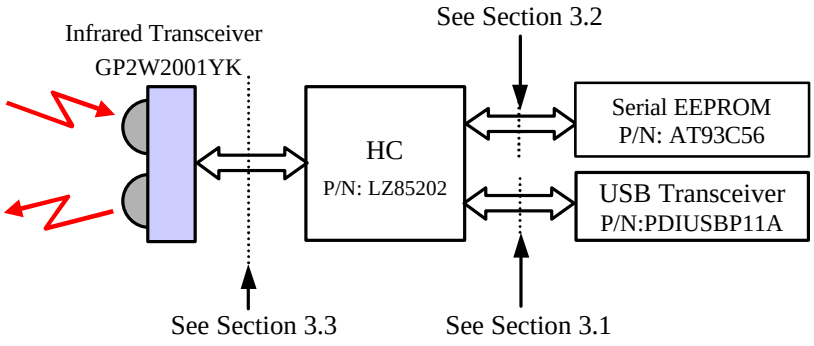



Figure 2.3. IrDA Control Host Dongle Implementation Example

**IrDA Control Peripherals**: The functions of the IrDA Control-PHY, MAC and LLC layers on the peripherals are similar to those of the USB Dongle. Infrared link communication with the USB Dongle can be made available by connecting infrared transceivers. (GP2W2001YK, GP2W2002YK: either model will implement the peripherals depending on the peripheral type) The MAC layer functions on the peripherals are described below.

1. The IrDA Control Peripheral would first send its identification data, such as PFID, Peripheral Info. etc., to a Host device to register itself. This registration process is called "**Enumeration**".
2. Once Enumeration is completed, the Host device will allocate the address (PADD) to the enumerated Peripherals and will start polling accordingly (The procedure to start polling is called "**Binding**.") The polling packet from the Host device to the Peripheral contains the data for the designated Peripheral. The IrDA Control MAC Layer in the Peripheral will extract this data and forward it to its upper layer, the HID LLC Layer. The Peripherals will reply with a response packet that includes the communication data from its HID LLC Layer. The Peripherals must start to respond within 213 us from the receipt of the polling packet, if the Peripherals have any data for the Host device. The Peripheral Engine IC controls this response time.

The HID LLC layer provides error correction, data re-sending, and flow management services between the USB Dongle and the peripheral. IrDA Control Peripherals send information to the Host device by using the HID LLC layer.
For the IrDA Control Peripheral implementation, Sharp provides an embedded communication controller called PE (Peripheral Engine, Sharp P/N: LZ85194). This communication LSI performs all functions of both MAC Layer and HID LLC Layer, and simplifies the peripheral implementation. With this LSI, the IrDA Control wireless peripherals can be implemented by just adding this PE, microcomputer for system control and the infrared transceiver.

In order to implement the whole IrDA Control system, HC (LZ85202), USB Transceiver (PDIUSBP11A), serial EEPROM (AT93C56) and infrared transceiver (GP2W2001YK, GP2W2002YK) are required for the USB Dongle. For the peripherals, PE (LZ85194), a Microcontroller for system control, infrared transceiver (GP2W2001YK, GP2W2002YK) are required.

## 2.2. Protocol Stacks

In an IrDA Control subsystem, certain procedures are required to make the infrared wireless communication available between a Host device and Peripherals. These required procedures are described below.

### 2.2.1. Enumeration Sequence

Enumeration is a registration procedure where the Peripherals will be acknowledged and registered to the Host device. The Host device does not know what kind of and/or how many Peripherals exist at the beginning. Thus, upon receipt of an Enumeration request from the Peripheral, the Host device will send its specific information to the Peripheral and register it, so that infrared communication can be initiated. Figure 2.4 shows an actual Enumeration sequence between the Host device and the Peripheral.

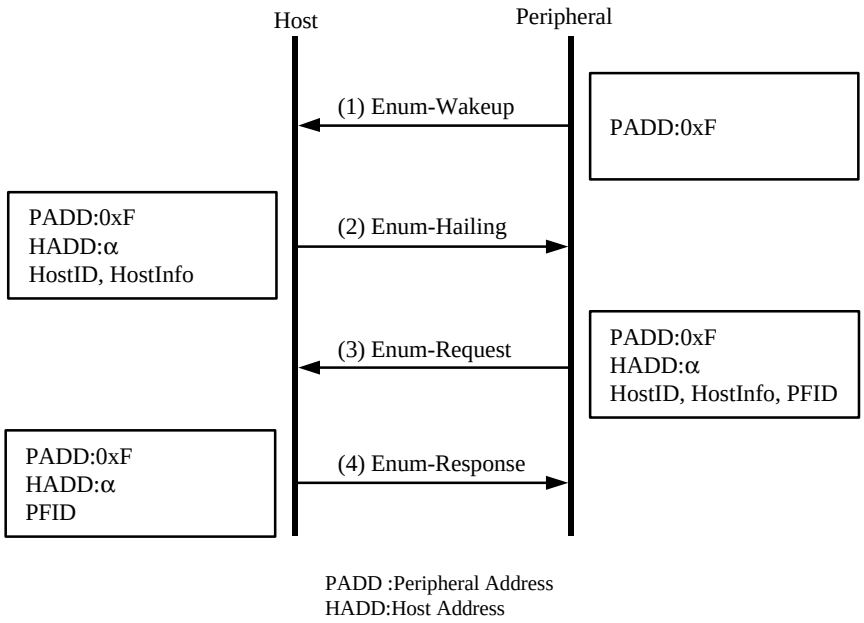



Figure 2.4. Enumeration Sequence

In Figure 2.4, (1) to (4) illustrates the types of data packets for the Host and the Peripheral. Explanation for the Enumeration sequence is as described below.

(1) Peripheral will always send the **Enum-Wakeup** packet to the Host device to request the polling.
(2) The Host device that received the above request from the Peripheral will then send **Enum-Hailing** packet so that it can receive the information from the Peripheral. (At this time, the Host device will add Host Address, HostID, and HostInfo, which identify the Host device itself.)
(3) The Peripheral obtains the Host information (HostID, HostInfo) out of the **Enum-Hailing** packet from the Host device, and then request Enumeration with this Host device by **Enum-Request**.
(4) The Host device will then acknowledge the Enumeration completion to the Peripheral, which requested the Enumeration with this Host device.

### 2.2.2. Binding Sequence

Binding is a communication initiation procedure where the Host device receives specific information from the Peripheral and initiates data communication by allocating the Peripheral address (PADD). While bound, the Peripheral can be polled by the Host device and can provide data to the Host device by responding to the poll. Figure 2.5 shows an actual Bind sequence between the Host device and the Peripheral.

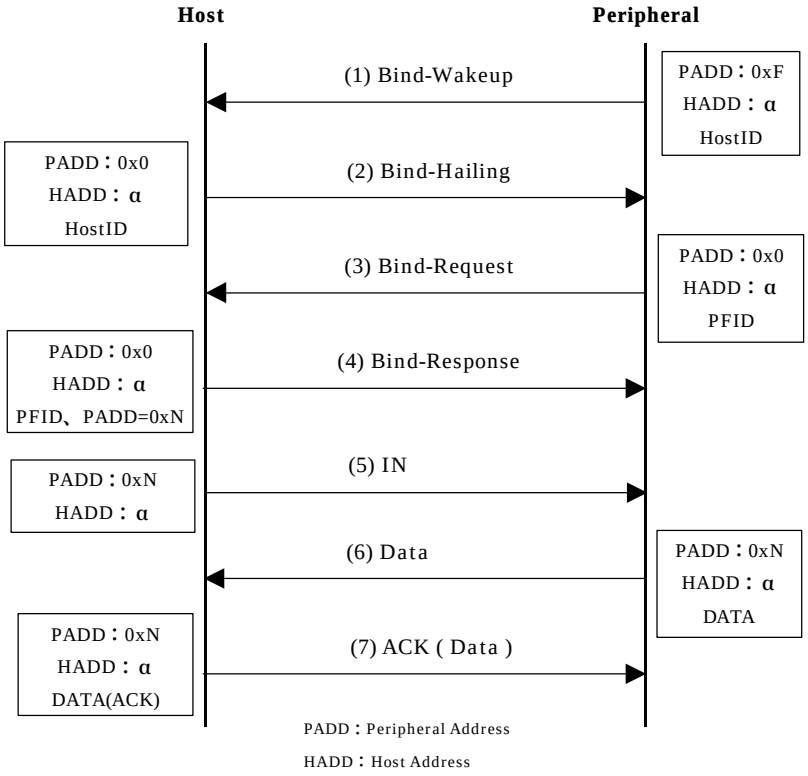



Figure 2.5. Binding Sequence

In Figure 2.5, (1) to (7) illustrates the types of data packets for the Host and the Peripheral. Explanation for the binding sequence is as described below.

(1) Peripheral will send the **Bind-Wakeup** packet to the Host device to request the polling, if the Host device does not give **Bind-Hailing** for a certain period of time (69 ms).
(2) The Host device that received above request from the Peripheral will then send **Bind-Hailing** packet so that it can communicate (or bind) with the Peripheral. (At this time, the Host device will tentatively assign Peripheral Address 0.)
(3) The Peripheral will then send **Bind-Request** to this Host device as a reply to the **Bind-Hailing**.
(4) The Host device will then acknowledge the Peripheral of Binding completion, with which the Host device officially assigns the Peripheral Address (PADD).
(5) Host device will start polling (IN).
(6) The Peripheral will reply to this poll from the Host device with the data.
(7) The Host device will return ACK as a confirmation of data receipt.

For the data communication, procedure (5), (6), and (7) will be repeated.

# 3. IrDA Control Host Controller Interface

## 3.1. Interface with USB Transceiver LSI

The HC and the USB transceiver (recommended LSI P/N: PDIUSBP11A by Philips) must be interfaced as described in Figure 3.1. (Signal lines: USBRCV, VP, VM, VPO, VMO, OEB, and SUSPND)
Because the recommended USB transceiver LSI is 3.3 V operation while the HC is 5.0 V operation, tolerant logic (3.3 V to 5.0 V) needs to be added on the HC's input signal lines (which are USBRCV, VP, and VM). For the same reason, tolerant logic (5.0V to 3.3V) is also recommended on the HC's output signal lines (VPO, VMO, USBOEB, and SUSPND) though the recommended USB transceiver can accept 5.0 V inputs.

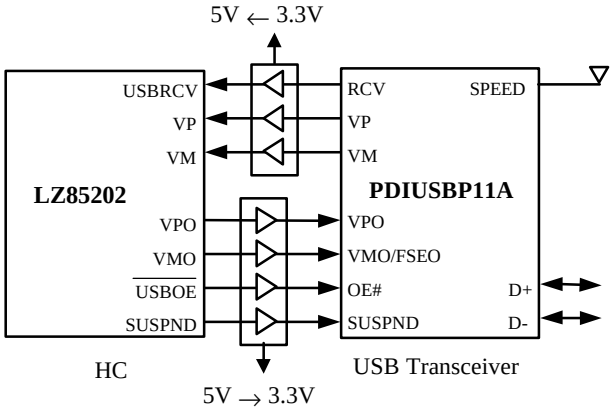



Figure 3.1. Interface between HC and USB Transceiver LSI

## 3.2. Interface with Serial EEPROM

For the USB dongle implementation using HC, an external serial EEPROM needs to be connected to HC. This is used to store the enumerated Peripherals information. (Recommended serial EEPROM P/N: AT93C56, 2 Kbit [128 x 16] by ATMEL) These LSIs must interface as described in Figure 3.2. (Signal lines: EEPCS, EEPCK, and EEPDATA) The HC and the serial EEPROM have serial connection. These two LSIs can be operated at 5.0V. 10 kΩ resistor is recommended between EEPDATA of HC and DO of serial EEPROM for HC device protection. Please refer to the AT93C56 data sheet for the timings of EEPROM.

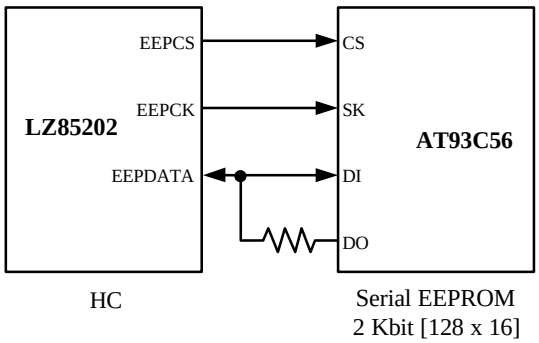



Figure 3.2. Interface between HC and Serial EEPROM

In case of using EEPROM other than the recommended P/N, it must completely meet the all operation timings shown in AT93C56 (ATMEL) specifications.

## 3.3. Interface with Infrared Transceiver (Front-end, FE)

The HC and the infrared transceiver (Sharp P/N: GP2W2001YK) must be interfaced as described in Figure 3.3. (Signal lines: FERESET, FECNT, IRRx1, and IRTx).
IRTx is a signal line for data transmission, IRRx1 for data receipt. FEREST is provided to reset the infrared transceiver's receiver sensitivity so that the transceiver can receive a very weak optical signal even after receiving a strong optical signal. FECNT is a control pinout to shut down the transceiver, so that the power consumption of the overall subsystem can be minimized while not in use. For the transmission signal line, (IRTx - TXD), 1.5 kΩ resistor is recommended for the current level adjustment, and 330 pF capacitance in parallel as a speed up capacitor. Note that 56 Ω resister is required on the AVcc pinout of the transceiver for the current level adjustment.
Please also refer to the transceiver data sheets as well as its appendix document for further instructions. (These documents are available from Sharp web site. http://www.sharp.co.jp/ecg/irda/irda-e.html)
For the detailed interface information, please refer to the circuit diagram in Appendix. When DC BIAS mode (See 4.3.3) is NOT supported, see " A-1" if the DC BIAS mode is supported, see "A-2".

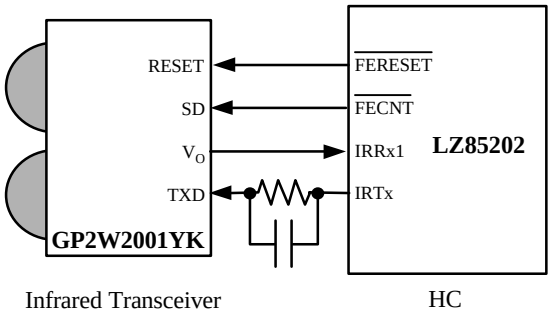



Figure 3.3. Interface between HC and Infrared Transceiver

# 4. USB Dongle Functions (USB Function) utilizing HC

In order for the plural peripherals (up to 8 devices) to simultaneously communicate within the IrDA Control system, the USB Dongle is required in the system. This USB Dongle is the adapter connectable to USB I/F and it provides infrared communication link with the task given below.

1. Connect the USB Dongle to the Host device via USB I/F.
2. Install the driver software (Dongle Driver software: see 4.2.4) into Windows98 OS

After the task 2, driver software installed in Windows 98 OS (HID driver) can be made available, by which the Host device can control maximum of 8 peripherals over infrared communication. Other functions of the IrDA Control Host Dongle are listed below:

- With a combination of BIOS that supports USB keyboards in market place, the dongle will be recognized as a keyboard with pointing device capability, and then be recognized as USB dongle once all drivers are loaded to the Windows 98 Operating System. (see section 4.2.3.)
- Full Speed data transfer (12 Mbps) service is available as USB function.

## 4.1. Module Structure

The HC internal modules that control the protocols of the USB Dongle on an IrDA Control subsystem are structured as illustrated in Figure 4.1;

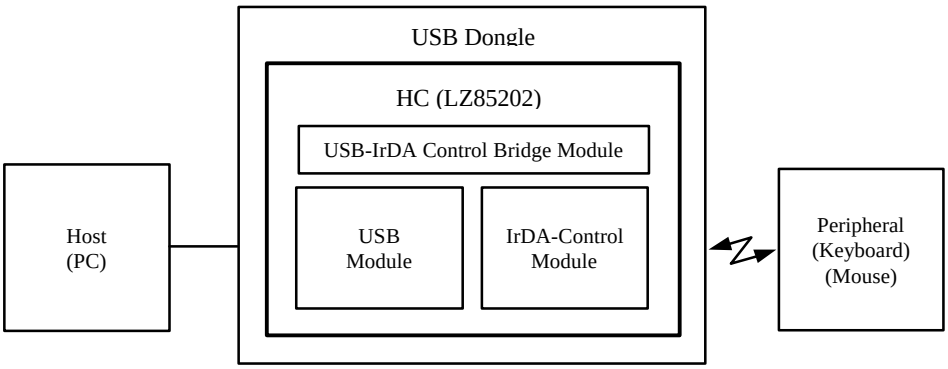



Figure 4.1. Module Structure

The HC is a LSI that manages the data transactions of the IrDA Control system via the USB I/F. It has an internal module, which is structured by the protocol control modules for both USB side (Host side) and IrDA Control side (Peripheral side). It also provides a protocol transforming bridge for the data transaction between these modules.

## 4.2. USB Protocol Process (USB Module)

In general, data exchanges between the Host device and the USB peripheral uses three different types of data, which are: TALKN packet (IN / OUT packet), data packet (packet of DATA0 & DATA1), and handshake packet (ACK, NAK and STALL).
In the IrDA Control data communication system, the HC's internal USB module controls the protocols for the above 3 different types of data packets, and takes the role of data transaction bridge to the Dongle Driver (see 4.2.4). The following section describes the USB protocol process by the HC.

### 4.2.1. Endpoint

The IrDA Control Endpoints listed in Table 4.1 are available on the HC.
The Endpoint is a concept in USB, which represents both ends of the communication flow between the USB Host and the devices, recognized as one communication path and/or channel.

Pipe is originally a concept that represents the logical communication channel itself. The types of the pipe are based on the Endpoints in Table 4.1, EP0 is the control pipe, EP1 and EP3 are data pipes (IN).

| Endpoint | Transfer Method | Description |
|---|---|---|
| EP0 | Control (OUT) | Used for USB-Enumeration information exchange in IrDA Control protocol |
| | Control (IN) | |
| EP1 | Interrupt (IN) | Send data from the Peripheral to the Host |
| EP3 | Interrupt (IN) | Send data from the Peripheral to the Host |

Table 4.1. Endpoint

### 4.2.2. Suspend mode and Remote wake-up

Suspend and Resume are the functions supported by the regular USB device. The current HC can support the remote wake-up function if it meets the conditions given below;
If the Host is in stand-by mode, and the USB Dongle is externally power supplied, the remote wake-up function will be supported. The USB Dongle has a function to make the Host resume when it receives a data from the peripheral during its suspend status.

### 4.2.3. Operation Mode (USB Configuration)

In an IrDA Control communication system, the USB Dongle implemented with the HC has two operation modes. One is "Emulation Mode", the other is "Bridge Mode." When booting the Host PC, the configuration value for the USB Dongle is first set to "1", which represents the Emulation Mode. After Windows 98 operating system is booted, and the "Dongle Driver" (see section 4.2.4) is loaded to the operating system, the configuration value of the USB Dongle will then set to "2", which represents the Bridge Mode. Description of these two operation modes is given below.

| Operation Modes | Configuration Values | Operation Mode Description |
|---|---|---|
| Emulation Mode | 1 | The dongle can emulate itself as a general wireless keyboard with pointing device capability (e.g. mouse) under this mode. Endpoint 1 is used for the keyboard data transmission, Endpoint 3 for mouse data transmission. |
| Bridge Mode | 2 | In this mode, the dongle will perform as the USB Dongle. IrDA Control keyboards and mice will be recognized as USB keyboard or mice in Host PC HID class. In this case, HC will allow several Peripherals to share a single Endpoint, and virtually perform as if it has multiple Endpoints for the Host PC. This mechanism will allow multiple peripherals to be connected to a single dongle, but the "Dongle Driver" needs to be installed to support USB Dongle function. |

### 4.2.4. Dongle Driver

When initiating the IrDA Control peripheral to communicate with the Host device with Windows 98 boot, the Dongle Driver must be properly installed into the Windows 98 operating system prior to the actual communication. Once this driver is installed to the PC, the Dongle Driver will be loaded on the Host when the USB Dongle is connected, which will be configured to the bridge mode (Configuration value: 2).
The Dongle Driver recognizes the connected IrDA Control peripherals (Keyboard, Mouse) as USB peripherals (USB keyboard, USB mouse), for which the HID drivers already provided on the Windows 98 operating system can be utilized. Installing this "Dongle Driver" on the Host (PC) will allow end users to utilize new IrDA Control peripherals without installing drivers unique to the peripherals.

### 4.2.5. Protocols between Host and USB Dongle

Figure 4.2 illustrates the protocols between the Host and USB Dongle implemented with HC. Following sections describes the items related to the protocol process between the Dongle Driver and the Host device.

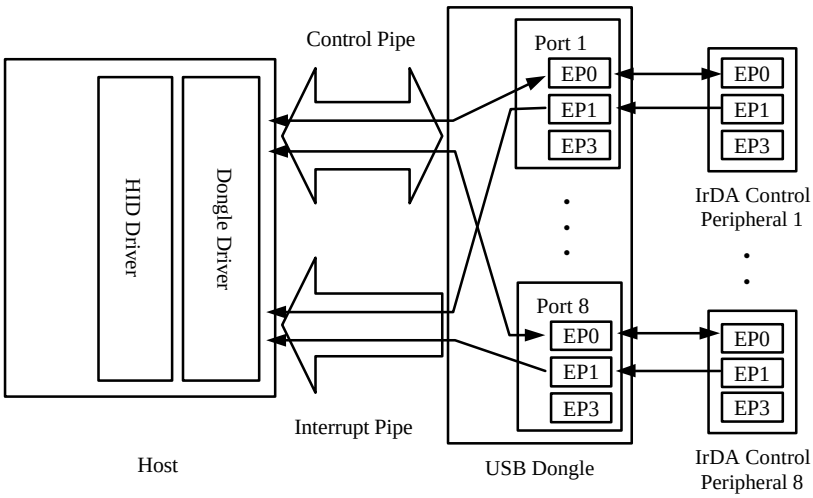



Figure 4.2. Protocols between Host and USB Dongle

The IrDA Control peripherals will send the data to the assigned port within the USB Dongle (Control information to Endpoint 0, and data to Endpoint 1). The USB Dongle will then send the data received from the IrDA Control peripherals to its Host. Control pipe is used for the control information on Endpoint 0, while the interrupt pipe is used for the data on Endpoint 1.

### 4.2.6. USB Requests

In general, USB devices respond to the USB request from the Host device through the control pipe, where the control transfer is applied for data transaction.
In the IrDA Control system, the USB Dongle starts data transmission with the setup packet. HC supports the control transfer and the interrupt transfer.
The USB Dongle implemented by the Sharp HC supports Standard Requests and HID Class Requests. Table 4.2 and Table 4.3 provides descriptions to these Requests;

(1) Standard Requests
Standard Requests that HC supports are listed in Table 4.2 below;

| Requests | Recipient | Function | | Description |
|---|---|---|---|---|
| Clear Feature | Device | Remote_Wake-up | Y | Disable Remote_Wake-up function. |
| | Endpoint | Endpoint_STALL | Y | Clear the STALL status of Endpoint. |
| Get Configuration | | | Y | Reply the current configuration value. (Configuration value is 1or 2) |
| Get Descriptor | | Device | Y | Reply the Device Descriptor. |
| | | Configuration | Y | Reply the Descriptor in following order; Configuration 1→ Interface (Keyboard) → HID (Keyboard) → Endpoint (Keyboard) → Interface (Mouse) →HID (Mouse) →Endpoint (Mouse) → Configuration 2 →Interface (Dongle) →Endpoint (Dongle) |
| | | String | N/A | (String is not used ) |
| | | Report | Y | Reply Report Descriptor. |
| Get Interface | Interface | | Y | Reply “0”. |
| Get Status | Device | | Y | Reply current condition. Self power is fixed to “1”. |
| | Interface | | Y | Reply “0”. |
| | Endpoint | | Y | Endpoint 0 & Endpoint 1 status is STALL. For other Endpoints, reply STALL. |
| Set Address | | | Y | Set the USB address of the USB Dongle. |
| Set Configuration | | | Y | Enters into the operation mode of USB Configuration depends on the value. |
| Set Descriptor | | | N/A | |
| Set Feature | Device | | Y | Enable Remote_Wake-up. |
| | Interface | | Y | Reply STALL. |
| | Endpoint | | Y | Set to the STALL status. |
| Set Interface | | | Y | |
| Synch Frame | | | N/A | |

Table 4.2. Standard Request

(2) HID Class Requests

HID Class Requests that HC supports are listed in Table 4.3 below;

| Requests | Recipient | | Description |
|---|---|---|---|
| Get_Idle | Interface | Y | Idle configuration of USB devices. Reply the value of "Set_Idle" because IrDA Contol peripherals can not configure the Idle. |
| Set_Idle | Interface | Y | Receives command, but not reflected to the operation. |
| Get_Report | Interface | Y | Send the Report data from the USB Device to the USB Host. |
| Set_Report | Interface | Y | Send the Report data from the USB Host to the USB Device |
| Get_Protocol | Interface | Y | Reply the protocol status of USB device. Reply the value of "Set_Protocol" because IrDA Control protocols are always Report Protocol. |
| Set_Protocol | Interface | Y | Receives command, but not reflected to the operation. |

Table 4.3. HID Class Request

### 4.2.7. Descriptor

Descriptors are exchanged between the USB Host and the USB Dongle during the USB-Enumeration through Endpoint 0. Types of the Descriptors are listed in the Table 4.4, 4.5, 4.6, 4.7, 4.8, and 4.9 categorizing their objective;

| Field | Value (HEX) | Description |
|---|---|---|
| bLength | 12 | Number of Descriptor bytes |
| bDescriptorType | 01 | Device Descriptor type |
| bcdUSB | 0100 | Release No. of the USB Specifications expressed by Binary-Coded Decimal. (e.g. 2.10 = 0x210) |
| bDeviceClass | 00 | Class code (by USB assignment) |
| bDeviceSubClass | 00 | Sub class code (by USB assignment) |
| bDeviceProtocol | 00 | Protocol code (by USB assignment) |
| bMaxPacketSize0 | 08 | Max. packet size of Endpoint0 (applicable size : 8, 16, 32, 64 ) |
| idVendor | 04DD | Vender ID (by USB assignment) [ SHARP ] |
| idProduct | 2000 | Product ID (by manufacturer assignment) |
| bcdDevice | 0100 | Release No. of devices expressed by Binary-Coded Decimal. |
| imanufacturer | 00 | String Descriptor Index described by manufacturers. |
| iProduct | 00 | String Descriptor Index described to the product. |
| iSerialNumber | 00 | String Descriptor Index that describes serial No. of devices |
| bNumConfigurations | 02 | Available number of Configuration<br>(1 : at Boot 2 : In using Dongle Driver) |

Table 4.4. Device Descriptor

| Field | Value (HEX) | | Description |
|---|---|---|---|
| | K & M | IrDA-C | |
| bLength | 09 | 09 | Number of Descriptor bytes |
| bDescriptorType | 02 | 02 | Configuration Descriptor type |
| wTotalLength | 003b | 0019 | Data length of all Descriptors replied as Configuration (Configuration, Interface, Endpoint, class or Vendor) |
| bNumInterfaces | 02 | 01 | The number of interfaces supported by Configuration |
| bConfigurationValue | 01 | 02 | Number of arguments used for Configuration selection by Set Configuration Request. |
| iConfiguration | 00 | 00 | String Descriptor Index that describes this Configuration |
| bmAttributes | E0 | E0 | Configuration Characteristics |
| MaxPower | 01 | 01 | Max. power consumption of USB device in this Configuration |

K & M: Emulation Mode
IrDA-C: Bridge Mode

Table 4.5. Configuration Descriptor

| Field | Value (HEX) | | | Description |
|---|---|---|---|---|
| | K & M | | IrDA-C | |
| | Key | Mouse | | |
| bLength | 09 | 09 | 09 | Number of Descriptor bytes |
| bDescriptorType | 04 | 04 | 04 | Interface Descriptor type |
| bInterfaceNumber | 00 | 01 | 00 | Interface number (Index of interface arrangement supported simultaneously.) |
| bAlternateSetting | 00 | 00 | 00 | Number used in order to select alternative configuration of interface specified by bInterfaceNumber. |
| bNumEndpoints | 01 | 01 | 01 | The number of Endpoints used in this Interface. (Exception: Endpoint “0”) |
| bInterfaceClass | 03 | 03 | 00 | Class code (by USB assignment) |
| bInterfaceSubClass | 01 | 01 | 00 | Sub class code (by USB assignment) |
| bInterfaceProtocol | 01 | 02 | 00 | Protocol code (by USB assignment) |
| iInterface | 00 | 00 | 00 | String Descriptor Index that describes this Interface |

Table 4.6. Interface Descriptor

| Field | Value (HEX) | | Description |
|---|---|---|---|
| | Key | Mouse | |
| bLength | 09 | 09 | Number of Descriptor bytes |
| bDescriptorType | 21 | 21 | HID Descriptor type |
| bcdHID | 0100 | 0100 | HID release No. expressed by Binary-Coded Decimal. |
| Country Code | 00 | 00 | |
| bNumEndpoint | 01 | 01 | Number of Endpoint used in this Interface. (Exception: Endpoint “0”) |
| DescriptorType | 22 | 22 | |
| DescriptorLength | 003F | 0032 | |

Table 4.7. HID Descriptor

| Field | Value (HEX) | | | Description |
|---|---|---|---|---|
| | Key | Mouse | IrDA-C | |
| bLength | 07 | 07 | 07 | Number of Descriptor bytes |
| bDescriptorType | 05 | 05 | 05 | Endpoint Descriptor type |
| bEndpointAddress | 81 | 83 | 81 | Address of the USB device as Endpoint |
| bmAttributes | 03 | 03 | 03 | Describe attributes of the Endpoint used with bConfiguration Value. |
| wMaxPacketSize | 0008 | 0003 | 0032 | Max. packet size for transmission and/or receive by the Endpoint |
| bInterval | 0A | 0A | 0A | Polling interval of Endpoint for data transfer |

Table 4.8. Endpoint Descriptor

| Value (HEX) (Keyboad) | Field (Keyboard) | Value (HEX) (Mouse) | Field (Mouse) |
|---|---|---|---|
| 0501 | Usage Page(Generic Desktop) | 0501 | Usage Page(Generic Desktop) |
| 0906 | Usage (Keyboard ) | 0902 | Usage (Mouse) |
| A101 | Collection (application) | A101 | Collection (application) |
| 0507 | Usage Page (Key Codes) | 0901 | Usage Page (Pointer) |
| 19E0 | Usage Minimum (224) | A100 | Collection (Physical) |
| 29E7 | Usage Maximum (231) | 0509 | Usage Page (Buttons) |
| 1500 | Logical Minimum (0) | 1901 | Usage Minimum (01) |
| 2501 | Logical Maximum (1) | 2903 | Usage Maximum (03) |
| 9508 | Report Count (8) | 1500 | Logical Minimum (0) |
| 7501 | Report Size (1) | 2501 | Logical Maximum (1) |
| 8102 | Input (Data, Variable, Absolute) | 9503 | Report Count (3) |
| 9501 | Report Count (1) | 7501 | Report Size (1) |
| 7508 | Report Size (8) | 8102 | Input (Data, Variable, Absolute) |
| 8101 | Input (Constant) | 9501 | Report Count (1) |
| 9505 | Report Count (5) | 7505 | Report Size (5) |
| 7501 | Report Size (1) | 8101 | Input (Constant) |
| 0508 | Usage Page (LEDs) | 0501 | Usage Page (Generic Desktop) |
| 1901 | Usage Minimum (1) | 0930 | Usage (X) |
| 2905 | Usage Maximum (5) | 0931 | Usage (Y) |
| 9102 | Output (Data, Variable, Absolute) | 1581 | Logical Minimum (-127) |
| 9501 | Report Count (1) | 257F | Logical Maximum (127) |
| 7503 | Report Size (3) | 7508 | Report Size (8) |
| 9101 | Output (Constant) | 9502 | Report Count (2) |
| 9506 | Report Count (6) | 8106 | Input (Data, Variable, Absolute) |
| 7508 | Report Size (8) | C0 | End Collection |
| 1500 | Logical Minimum (0) | C0 | End Collection |
| 2565 | Logical Maximum (101) | | |
| 0507 | Usage Page (Key Codes) | | |
| 1900 | Usage Minimum (0) | | |
| 2965 | Usage Maximum (101) | | |
| 8100 | Input (Datum Array) | | |
| C0 | End Collection | | |

Table 4.9. Report Descriptor (Keyboard & Mouse Emulation Mode)

## 4.3. IrDA Control Protocol Process (IrDA Control Module)

There are three communication statuses for the Peripherals in the practical data transaction on the IrDA Control subsystem;

*(1) Enumeration Status*
The communication status where dongle starts communication, complete Enumeration process, and then goes into the Sleep status by the Wakeup packet from the Peripheral.

*(2) Communication Status*
The communication status where the dongle polls the enumerated IrDA Control peripherals after completion of Bind process. The Host device will also executes Hailing for Enumeration as well as Binding periodically.

**Hailing for Enumeration and Bind:**
When the Host device is in the condition in which it can further bind the other IrDA Control Peripherals (see table 4.10), the Host device will execute Hailing for the Enumeration as well as Binding periodically. However, if 7 NCL (Non-Critical Latency) devices have already been bound, the Host will NOT execute Enum-Hailing.

| | Non-Critical Latency (NCL*1) Peripherals | Critical Latency (CL*2) Peripherals |
|---|---|---|
| Pattern 1 | *8 | 0 |
| Pattern 2 | *7 | 1 |
| Pattern 3 | *6 | 2 |
| Pattern 4 | *5 | 3 |
| Pattern 5 | *1 | 4 |

Table 4.10. Bind Management

*NOTE)
Number of NCL Peripherals allowed to be bound with CL Peripherals is referring to the case when NCL Peripherals will only use short packet.

*1 NCL : Non-Critical Latency devices that do not require an immediate response to the transmitted data. Mice and keyboards should be categorized as NCL devices. In IrDA Control system, NCL devices remain bound for 5 seconds after the last data transmission.

*2 CL : Critical Latency devices that require an immediate response to the transmitted data. Gamepads or Joysticks should be categorized as CL devices. In IrDA Control system, CL devices remain bound for 30 seconds after the last data transmission.

*(3) Sleep Status*
The communication status where no IrDA Control peripherals are bound, thus no communication occurs between the Host and the Peripherals.

With the process (1) and (2), the IrDA Control Peripherals can communicate with the Host device over the IrDA Control subsystem. As described in Table 4.10, the IrDA Control subsystem allows maximum of 8 NCL Peripherals or maximum of 4 CL Peripherals to be bound simultaneously.

Following is an example of the IrDA Control protocol processing, assuming that a PC performs is the USB Root Hub for the Host device.
While the configuration of the USB Dongle operation mode is set to "Emulation Mode" (see Section 4.2.3.), a PC input wireless peripheral that has keyboard and mouse capabilities can be implemented by using EP1 and EP3 of HC. (See Figure 4.3.) The HC will receive the keyboard data from EP1 and the mouse data from EP3. HC will also be requested the descriptor of the USB protocol by the Dongle Driver, for which EP0 is used to exchange the descriptor with the IrDA Control peripherals and respond to.

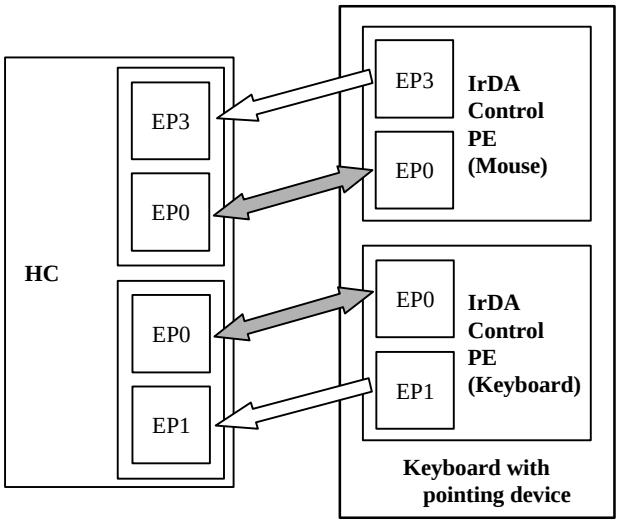



Figure 4.3. Example of Using EP1 and EP3 Simultaneously (Keyboard with Pointing Device)

### 4.3.1. Memorizing Enumeration Information

An external serial EEPROM can be used to memorize the Enumeration information and/or address even the main power supply of the Host device hardware is turned off (see Section 3.2. and 6.6.1).
This function allows the use of the IrDA Control Peripherals, which are already enumerated to the Host device, under the same user environment next time the users turn on the Host device. Following are the items to be memorized to the serial EEPROM;

- Host Address
- Host ID
- PFID, Peripheral Information

Above memorized Enumeration information allows users to use the same IrDA Control peripherals without enumeration process. The old data on the serial EEPROM is renewed as data exchange fashion when writing the new data to it is completed. For this data exchange, error check code is added to ensure the data renewal.
To memorize above Enumeration information to the serial EEPROM of the USB Dongle, HC pinouts of EEPCS, EEPCK, and EEPDATA must be correctly connected to the serial EEPROM pinouts of CS, SK, and DI.

### 4.3.2. Support for the Peripherals

The USB Dongle will allocate the registration numbers to the Peripherals once they are enumerated.
Allocation of the registration numbers has a certain priority. Based on this priority, the Peripherals will have registration numbers allocated by the Host device. Table 4.11 shows the registration numbers as well as its priority.

| Priority | Keyboard (Keyboard & Mouse Composite) | Mouse | Other Peripherals |
|---|---|---|---|
| 1 | 0 | 1 | 2 ~ 7 |
| 2 | 2 ~ 7 | 2 ~ 7 | 1 |
| 3 | 1 | 0 | 0 |

Table 4.11. Registration Numbering of Enumeration (Priority)

As long as there is a vacancy in the registration numbers, IrDA Control Peripherals will have their registration number based on the Table 4.10. However, there could be the case that a new Peripheral wants to be enumerated when no vacancy on the registration numbers (meaning that 8 Peripherals are already enumerated, and 9th Peripheral wants to be enumerated).
In such cases, the USB Dongle tries to delete the port and enumerate the new Peripheral by following the priorities described below.

1. List up the deletion candidates from the Peripherals that have registration number "2" or larger.
2. Remove the Peripherals currently bound from the list of deletion candidates.
3. If the Peripheral that is currently trying to be enumerated is NOT a keyboard, and the deletion candidate is the only enumerated keyboard, the keyboard will be removed from the list of deletion candidates, and will not be dropped.
4. If the Peripheral that is currently trying to be enumerated is NOT a mouse, and the deletion candidate is the only enumerated mouse, the mouse will be removed from the list of deletion candidates, and will not be dropped.

By applying above criteria, when there is any registration number that can be deleted, the USB Dongle will accept the request from the 9th Peripheral and enumerate. If not, the USB Dongle will reject the Enumeration request.

### 4.3.3. DC BIAS mode

IR Data Transmission is commonly used in the remote controller for consumer electronic products such as TV, video player, and air conditioners. When using an IrDA Control System in such environment, the DC BIAS mode made available on the HC should be used to not interfere with consumer remote control data. HC can be configured to support the DC BIAS by applying an input to pin #93. To activate the DC BIAS mode, set the signal level of #93 pin to High.
Detailed interface between the HC and the Infrared Transceiver should be designed by referring to the circuit drawn in Appendix A-2.

**4.3.4. Setting the Host Address (Using DIP Switch)**

The Host Address is determined after the IrDA Control Host device receives the wakeup packet from the Peripherals. At the same time, the Host ID is also determined. The HC sets the communication controller timer when the wakeup packet is received, and sets the Host Address by random numbers.

In the IrDA Control system, there is a possibility that multiple Host devices have the same Host Address in the reached range because of Host Address generated by random numbers. Manual configuration of the Host Address by the use of DIP switch may minimize this possibility. If the DIP switch is used, the upper nibble of the Host Address will be set with the 4 bits information of the DIP switch, and the lower nibble will be determined by the random numbers. Using DIP switch, HC configures the Host Address by the following instruction;

1. Observe the DIP switch value first.
2. Attempt to restore the Host Address

When HC is to restore the Host Address, Host Address configuration varies whether the restoration succeeded or failed;

*(Case A) Restoration Failed*

(A-1) DIP switch value is NOT "0";
➔ Upper nibble by DIP switch, lower nibble by random number.
(A-2) DIP switch value is "0";
➔ All by random number.

*(Case B) Restoration Succeeded*

(A-1) DIP switch value is NOT "0", and the restored Host Address differs from that of EEPROM;
➔ Upper nibble by DIP switch, lower nibble by random number.
(A-2) DIP switch value is "0";
➔ Use restored Host Address.

For other cases when the DIP switch value is "0", Host Address may be changed by the USB request. In changing the Host Address by the USB request, the upper nibble is configured by the USB request, and the lower nibble by the random number.

After all, if the Host Address is changed, HC performs the following configuration;

3. Re-generate the HostID by random number, and initialize the Enumeration Table.

## 4.4. USB-IrDA Control Bridge Process (USB-IrDA Bridge Module)

The protocol interface for USB and that of IrDA Control deal with different data forms, and the data cannot be directly exchanged between the USB protocol process module and the IrDA Control protocol process module.
As explained, the USB-to-IrDA Control bridge module is required to translate the data processed by the USB module (or IrDA Control module) into the data that can be handled by the IrDA Control module (or USB module).

### 4.4.1. USB-IrDA Control Mapping

Table 4.12 shows the mapping for USB requests and IrDA Control protocols;

| USB Requests | IrDA-C Protocol | Parameter |
|---|---|---|
| Get Descriptor | Get Descriptor | 1ST byte : Descriptor ID<br>2ND byte : Index<br>3RD byte : Language ID<br>4TH byte : Language ID |
| Get Descriptor (Report) | Get Descriptor | 1ST byte : Descriptor ID<br>2ND byte : Endpoint |
| Get Descriptor (Others) | Get Descriptor | 1ST byte : Descriptor ID |
| Get Report | Get Status | 1ST byte : Report ID<br>2ND byte : Report Type |
| Set Report | Set Mode | 1ST byte : Report ID<br>2ND byte : Report Type<br>3RD byte / beyond : Data |

Table 4.12. USB Request - IrDA Control Mapping

### 4.4.2. Status Indication LEDs

The Sharp HC has the output pins to illuminate the LEDs in order to indicate the internal status of the USB Dongle. These pinouts are all active HIGH. Following are the indicators that represent the internal status of the USB Dongle;

*(1) USB Configuration Indicator*

Because the HC supports two operation modes (Emulation Mode, Bridge Mode), the HC will illuminate the LED to indicate the USB Dongle operation mode. (This LED also indicates that the USB interface on the USB Dongle is correctly connected to the Host.)

*(2) Keyboard Indicators*

There are some indicators on a keyboard to show the key configurations. (NumLock, CapsLock, and ScrollLock). IrDA Control keyboards corresponds to these status indication LED by Set Report of the USB request. USB Dongle, therefore, illuminates these status indication LEDs with the Set Report.

*(3) Power Supply Alarm Indicator*

In order to secure the data communication with the IrDA Control Peripherals, the Host device in IrDA Control subsystem periodically inquires to the Peripherals of their power supply condition.
The HC has the capability to illuminate the alarm LED on the USB Dongle, when it receives the alarm from the Peripherals.

### 4.4.3. Peripheral Power Supply Information

**<FUNCTION>**

In the IrDA Control subsystem, the IrDA Control Peripherals need to have their own power supply to communicate with the Host device over IR. In order to ensure the proper communication, the USB-IrDA Control Bridge in the HC inquires about the power supply condition of the IrDA Control Peripherals every time they are bound to the Host device. The result will be forwarded to the USB Host via USB.

1) In the event that the Peripheral does not appropriately reply to the inquiry, the HC will regard that the device does not support the Peripheral Power Supply Information function, and will not inquire about the power supply conditions from the next Binding.
2) During the control transfer (USB-Enumeration), HC will inquire about the power supply condition when the current control transfer is completed.
3) If the USB Host requests the HC for the control transfer during the power supply condition inquiry, the HC will respond to the USB request by STALL.

**<SPECIFICATIONS>**

1) The HC will individually record the data transaction start time after Binding for each Peripheral. The Host device will then issue "Get Status", after more than 1 second of time from the recorded data transaction start time.
2) In the parameter of "Get Status" issued by the Host device, 2 bytes of data, which are "0x00 (Report ID)" and "0xF0 (Report type)" are enclosed. The Peripheral Engine in the Peripheral (PE, Sharp P/N: LZ85194) extracts these 2 bytes of data, and report to the peripheral system μP.
3) The Peripheral recognizes above 2 bytes of data as "power supply condition inquiry", and respond to the Host device by 1 byte of data, which are either "0x00 (normal)", "0x01 (alarm)", or "0x02 (caution)".
4) In case that the USB Dongle fails to receive the power supply condition information from the Peripheral, or exceeds the retry attempts for the "Get Status" of the IrDA Control HID LLC layer, the HC will not report the power supply conditions to the USB Host. The USB Dongle will then disable this power supply condition inquiry function and will not inquire the condition at next Binding.

### 4.4.4. Auto Repeat Cancellation for Keyboard Error

In general, keyboards make the desired character appear on the display as long as it sends the “key stroke” data to the Host device. Or, the typed character keeps appearing on the display until the keyboard sends the “key release” code to the Host device. When using a wireless keyboard, the “key release” code thus needs to be correctly received for the desired operation. Such a case may happen, such as an unexpected disconnection of the keyboard

The “Auto Repeat Cancellation for Keyboard Error” is a unique function of the HC, which is to prevent the undesired phenomena described above. In the event that the USB Dongle does not receive any data from the IrDA Control keyboard over 220 ms after receiving “key stroke”, the USB Dongle automatically sends “key release” code to the Host device.
In another words, the IrDA Control keyboard must periodically send the data to the USB Dongle every 220ms. Figure 4.4 shows the Auto Repeat Cancellation Function for Key board.

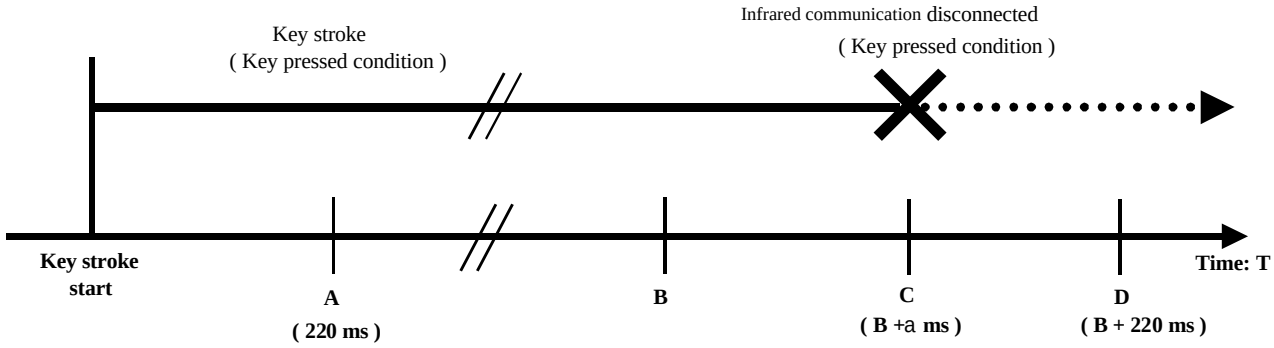



Figure 4.4. Sequence of Auto Repeat Cancellation for Keyboard

<Status Explanation "A" to "D" in Figure 4.4>

A: The USB Dongle tries to send "key release" code to the Host device, as 220 ms has already passed after receipt of the "key stroke" without receiving "key release" code. But the IrDA Control keyboard continues sending another "key stroke" code to the USB Dongle before reaching status A. In this case, the USB Dongle will not send the "key release" code to the Host devices, and the system enters to the "Auto Repeat" mode. (Showing the same character, or keep performing the same command.)

B: The IrDA Control keyboard is in the condition where a certain key is kept pressed. In the status of A and B, the IrDA Control keyboard periodically sends "key stroke" data to the USB Dongle.

C: Assume IR link is intercepted between the IrDA Control keyboard and the USB Dongle. In this condition, the USB Dongle cannot receive the "key stroke " data from the IrDA Control keyboard.

D: The USB Dongle should receive the "key stroke" data from the IrDA Control keyboard, but prevented because of losing the IR link in status C. The USB Dongle cannot receive the "key stroke" data. At this point, the USB Dongle recognizes the situation as the IrDA Control keyboard is disconnected, and sends the "key release" data to the Host devices to cancel the Auto Repeat mode for the IrDA Control keyboard.

**<SPECIFICATIONS>**

Peripherals are distinguished with the keyboard bit of PFID. The report mechanism for the keyboard keys are assumed to conform to the boot device format defined in the USB specifications.

1) All "0"s in the report data represents the key release. Thus, the key release code generated by the HC also conforms to this definition.
2) In case of keyboard with pointing device capability, HC will recognize Endpoint 1 as keyboard.

## 4.5. USB Dongle Operation

### 4.5.1. The Communication between USB Dongle and Host (Windows 98)

The data between the Root Host and the USB Dongle is communicated through the two pipes, the Control Pipe (Endpoint 0) and the Interrupt Pipe (Endpoint 1).

By using the Control Pipe, the USB and the HID request are transferred from the Host to the USB Dongle, and the data corresponding to this request (CData) is IN / OUT transferred.

The Interrupt IN polling requests between the HID and the Dongle Driver, and between the Dongle Driver and the USB Dongle are transferred through the Interrupt Pipe. The USB Dongle responds with the data (SData) corresponding to the IN polling request. Figure 4.5 shows the transferring path for these data.

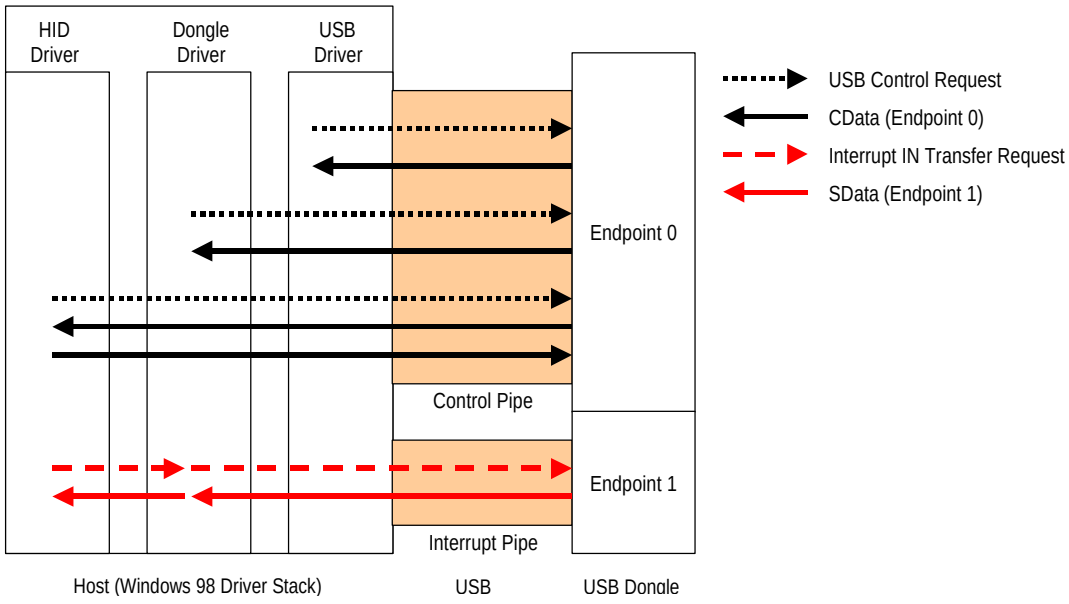



Figure 4.5. Requests and Data Stream between Host and USB Dongle

### 4.5.2. USB-Enumeration on the USB Dongle

The following sequence represents the request and the relative data sequence, where the USB Dongle itself is enumerated to the Host device (Windows 98 Operating System) via USB.

| No. | Host Request ( Issued by USB Driver & Dongle Driver ) | Reply Data from USB Dongle |
|---|---|---|
| Connect the dongle to USB; USB driver to issue following USB requests to dongle to load the Dongle Driver; | | |
| 1 | Get Descriptor (Device) | |
| 2 | | CData (Device Descriptor) Top 8 bytes |
| 3 | Set Address | |
| 4 | Get Descriptor (Device) | |
| 5 | | CData (Device Descriptor) |
| 6 | Get Descriptor (Configuration) | |
| 7 | | CData (Configuration Descriptor[1]) |
| 8 | | CData (Interface Descriptor) |
| 9 | | CData (HID Descriptor) |
| 10 | | CData (Endpoint Descriptor) ♦1 |
| 11 | | CData (Interface Descriptor) |
| 12 | | CData (HID Descriptor) |
| 13 | | CData (Endpoint Descriptor) |
| 14 | | CData (Configuration Descriptor[2]) |
| 15 | | CData ( Interface Descriptor) ♦2 |
| 16 | | CData (Endpoint Descriptor) |
| Dongle Driver Loading & USB Request is issued by Dongle Driver | | |
| 17 | Get Descriptor (Device) | |
| 18 | | CData (Device Descriptor) |
| 19 | Get Descriptor (Configuration) | |
| 20 | | CData (Configuration Descriptor[1]) |
| 21 | | CData (Interface Descriptor) |
| 22 | | CData (HID Descriptor) |
| 23 | | CData (Endpoint Descriptor) ♦1 |
| 24 | | CData (Interface Descriptor) |
| 25 | | CData (HID Descriptor) |
| 26 | | CData (Endpoint Descriptor) |
| 27 | | CData (Configuration Descriptor[2]) |
| 28 | | CData ( Interface Descriptor) ♦2 |
| 29 | | CData (Endpoint Descriptor) |
| 30 | Set Configuration (2) | |
| Interrupt Pipe is open; Dongle Driver to issue Interrupt IN transfer request to the USB Dongle | | |
| 31 | Interrupt IN | |

♦1: Configuration for Emulation mode
♦2: Configuration for Bridge mode

Table 4.13. USB-Enumeration of USB Dongle

### 4.5.3 USB-Enumeration for Peripherals

Table 4.14 represents the request and the relative data sequence between the Host device (Windows 98 Operating System) and the USB Dongle, where the USB Dongle is USB-Enumerated to the Host, and the IrDA Control Peripheral is newly bound under the condition that USB Dongle is operating at Configuration [2] (Bridge Mode) .

| No. | Request issued by HID Driver | Request issued by Dongle Driver and Reply Data | Reply Data from USB Dongle |
|---|---|---|---|
| | The new peripheral is bound. | | |
| 1 | | Interrupt IN | |
| 2 | | | SData<br>( Trigger of USB-Enumeration) |
| 3 | | Get Descriptor (Device) | |
| 4 | | | CData (Device Descriptor) |
| 5 | | Get Descriptor (Configuration) | |
| 6 | | | CData (Configuration Descriptor) |
| 7 | | Get Descriptor (IrDA Control) | |
| 8 | | | CData (IrDA-C Descriptor) |
| | Loading the HID Driver (HID Class driver, etc.) | | |
| 9 | Get Descriptor (Device) | | |
| | | CData (Device) | |
| 10 | Get Descriptor (Configuration) | | |
| | | CData (Configuration) | |
| 11 | Get Descriptor (Configuration) | | |
| | | CData (Configuration) | |
| 12 | Set Configuration [1] | | |
| 13 | Set Idle | | |
| 14 | | Set Idle | |
| 15 | Get Descriptor (Report) | | |
| 16 | | Get Descriptor (Report) | |
| | | | CData (Report Descriptor) |
| 18 | Interrupt IN | | |
| 19 | | Set Feature (Endpoint Enable) | |
| | After the Set Future, Polling for IrDA Control Peripheral starts. | | |
| 20 | | Interrupt IN | |
| 21 | | | SData<br>( Data from the Peripheral) |

Table 4.14. USB Enumeration of IrDA Control Peripherals

### 4.5.4. Dongle Driver Request

There are two types of USB requests issued by the Dongle Driver to the dongle in the Enumeration for USB Dongle and IrDA Control Peripheral. They are:

1. Get Descriptor (Device, Configuration, IrDA Control, Report, String)
2. Set Feature

These are Vendor Specific requests. The Peripheral enumeration recognition information is set to the wValue and wIndex of the setup packet.

### 4.5.5. HID Driver Request

There are 5 types of requests issued by the HID driver to the Dongle Driver (dongle) in the Enumeration for USB Dongle and IrDA Control Peripheral. They are:

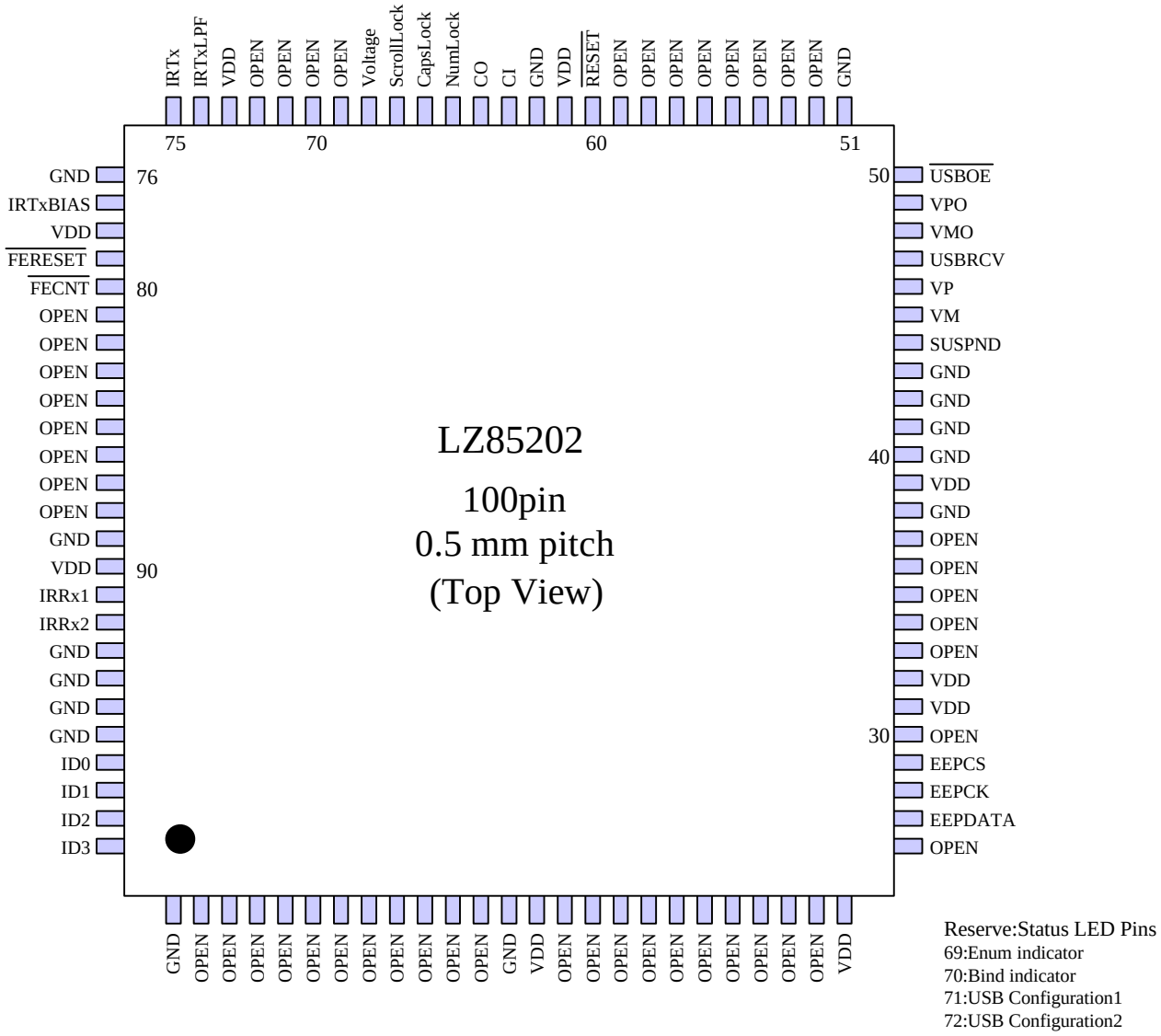
1. Get Descriptor (Device, Configuration)
2. Get Descriptor (Report)
3. Set Configuration
4. Set Idle
5. Set Report (depends on the IrDA Control Peripheral. There is no information in Table 4.14 )

The above item 1 is a USB standard request. The Dongle Driver corresponding to the HID request provides the data. This is because the Dongle Driver memorizes the data, such as Device, Configuration and IrDA Control Descriptor, which is exchanged just after a IrDA Control Peripheral is bound.

*NOTE) For the details of each requests, see the USB and HID class specifications.

# 5. Pinouts

## 5.1. Pinout Assignment




Reserve:Status LED Pins
69:Enum indicator
70:Bind indicator
71:USB Configuration1
72:USB Configuration2

*OPEN: NO CONNECTION. However, for those willing to use status LEDs, note that pins # 69 to 72 can be used for status LED illumination.

Figure 5.1. HC Pin Assignment

## 5.2. Pinout Functions and Descriptions

| Symbol | I/O | Description |
|---|---|---|
| CI<br>CO | INPUT<br>OUTPUT | Connected to the crystal oscillator. (See the recommended circuit diagram : Figure 6.2) |
| RESET | INPUT | ACTIVE LOW / Rest signal input. |
| IRTx | OUTPUT | Data signal output for infrared transmission. It must be connected to the TXD pin of the infrared transceiver (GP2W2001YK) |
| IRTxBIAS | OUTPUT | BIAS signal output for infrared communication. It is used to add DC BIAS on the IR transmission. |
| IRTxLPFC | OUTPUT | LPFC signal output for infrared transmission. This function is for future expansion. In practical implementation, it is regarded as OPEN. |
| IRRx1 | Schmitt INPUT | Data signal input for infrared receipt. It must be connected to the Vo pin of infrared transceiver (GPW2001YK). |
| IRRx2 | Schmitt INPUT | Data signal input for infrared receiver. This functionally is for future expansion. In practical implementation, it must be connected to GND. |
| FECNT | OUTPUT | ACTIVE LOW / Control signal output to shutdown the infrared transceiver. It must be connected to the SD pin of the infrared transceiver (GPW2001YK). |
| FERESET | OUTPUT | ACTIVE LOW / Reset signal output for infrared transceiver. It must be connected to the REST pin of infrared transceiver (GPW2001YK). |
| USBRCV | INPUT | Receive signal input for USB transceiver. It must be connected to the RCV pinout of USB transceiver (PDIUSBP11). |
| VP | INPUT | D+ signal input for USB transceiver. It must be connected to the VP pinout of USB transceiver (PDIUSBP11). |
| VM | INPUT | D- signal input for USB transceiver. It must be connected to the VM pinout of USB transceiver (PDIUSBP11). |
| VPO | OUTPUT | D+ signal output for USB transceiver. It must be connected to the VPO pinout of USB transceiver (PDIUSBP11). |
| VMO | OUTPUT | D- signal output for USB transceiver. It must be connected to the VMO pinout of USB transceiver (PDIUSBP11). |
| USBOE | OUTPUT | ACTIVE LOW / Data output enable signal output for USB transceiver. It must be connected to the OE pin of the USB transceiver (PDIUSBP11) |
| SUSPEND | OUTPUT | It must be connected to the SUSPEND pinout of USB transceiver. |
| EEPCS | Tri-state w/ pull down | Chip select signal output for serial EEPROM. It is also possible to use this pin for a general-purpose output pinout. Hi-Z is output during RESET period. |
| EEPCK | Tri-state w/ pull down | Clock signal output for serial EEPROM. It is also possible to use this pin for a general-purpose output pinout. Hi-Z is output during RESET period. |
| EEPDATA | INPUT<br>OUTPUT | Data signal for serial EEPROM. It is also possible to use this pin for a general-purpose output pinout. Hi-Z is output during RESET period. |
| ID3:0 | INPUT | ID recognition signal input pinouts for IrDA Control Communication. The least 4 bit data in 00 to FFh can be configured with these 4 input terminals. |
| NumLock | OUTPUT | Signal output to illuminate NumLock LED. |
| CapsLock | OUTPUT | Signal output to illuminate CapsLock LED. |
| ScrollLock | OUTPUT | Signal output to illuminate ScrollLock LED. |
| Voltage | OUTPUT | Signal output to illuminate LED for power supply condition information of the Peripherals. |
| VDD | | Power supply |
| GND | | GND |

Table 5.1. Pinout Function (LZ85202)

## 6. Block Diagram

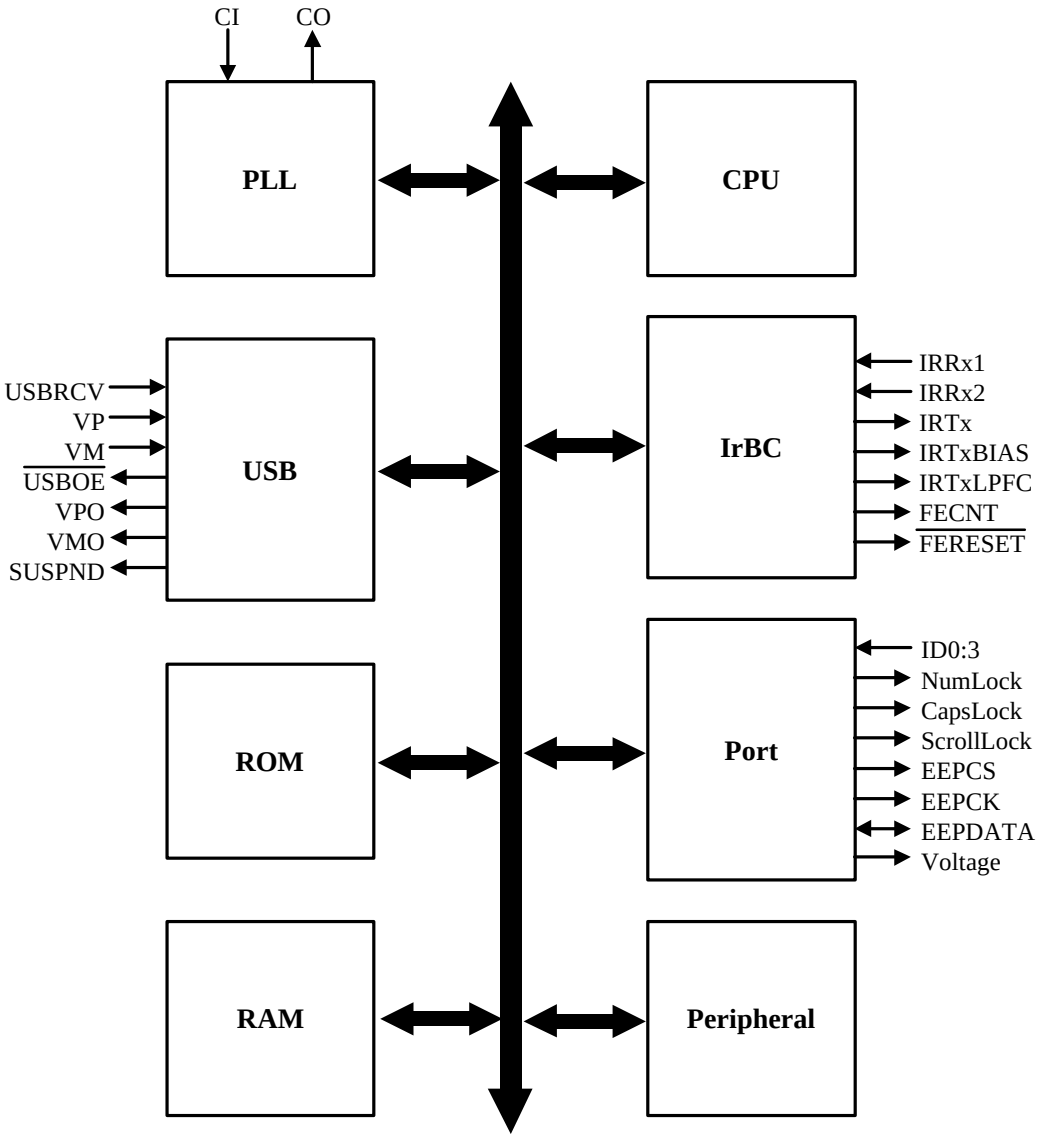



Figure 6.1. HC Block Diagram

### 6.1. CPU

This CPU controls all HC functionality.

## 6.2. PLL (Phase Lock Loop) and CG (Clock Generator)

The internal PLL generates the master oscillation for the HC from the external 12 MHz crystal connected to the CI pin and CO pin. The generated signal supplies the each clock signals required for the internal functions of the HC.

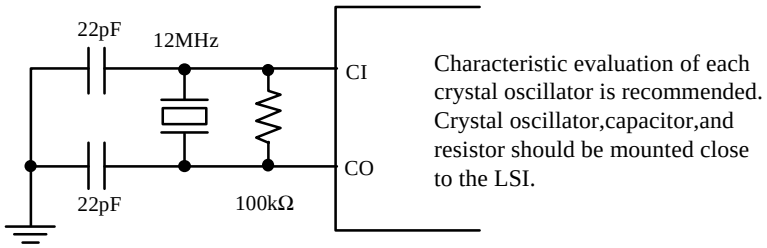



Figure 6.2. HC Recommended Oscillation Circuit

In addition, when the RESET signal is asserted to the HC, clock signals will be supplied to the internal circuit after the PLL counts 65535 clock cycles.

## 6.3. USB (Universal Serial Bus)

The USB block is a USB device controller for devices that support the full speed of 12 Mbps. Endpoints and the maximum packet size (FIFO size) are described as below:

| | | | |
|---|---|---|---|
| Endpoint0 | : Control | OUT | 8 bytes |
| | | IN | 8 bytes |
| Endpoint1 | : Bulk / Interrupt | IN | 64 bytes |
| Endpoint2 | : Bulk / Interrupt | OUT | 64 bytes |
| Endpoint3 | : Bulk / Interrupt | IN | 8 bytes |

## 6.4. IrBC

Computing capability for CRC-8 and CRC-16, and has built-in 2 channels of 8 bit timer available for the protocols, and 10- byte size FIFO for both transmission and receive.

## 6.5. RAM (Random Access Memory) and ROM (Read Only Memory)

Memories for HC functions. RAM is used for the work area, and the ROM contains the protocol as well as application firmware of the HC.

## 6.6. Port

### 6.6.1. Serial EEPROM

As described in separate section, serial EEPROM is used to memorize variety of data such as Host Address, Host ID, PFID, and Peripheral Specific Information. The ports on the serial EEPROM are configured as illustrated in Figure 6.3.

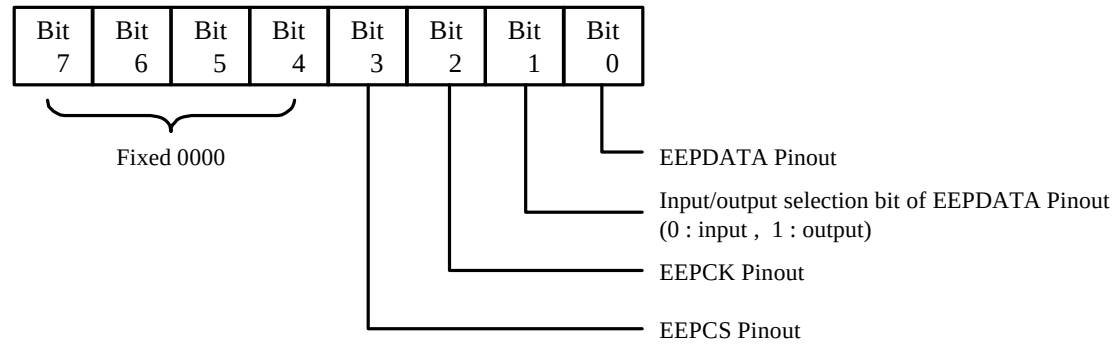



Figure 6.3. Serial EEPROM Bit Assignment

### 6.6.2. Status Indicator LED

As described in separate section, LEDs are used to indicate the internal status of the USB Dongle. Bits are configured as described in Figure 6.4. (Current HC firmware assigns status LEDs from #65 to #72 pinouts.)

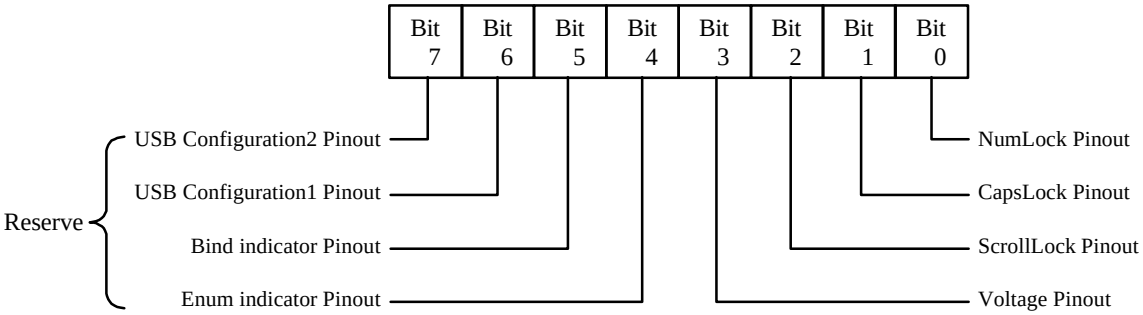



Figure 6.4. Status LED Bit Assignment

### 6.3.3. DIP Switch

In case of manual setting for the Host Address, or the setting of DC BIAS mode, DIP switch implementation is recommended for HC. HC's #93 pinout through #100 pinout are assigned for the use of DIP switch. #93 pinout is assigned for the DC BIAS mode (Recommended DIP switch: No.1).
Pins #94 through #96 are reserved for future expansion. Pins #97 through #100 are assigned for manual setting for upper 4 bits of the Host Address. (No.5 through No.8 of the recommended DIP switch)

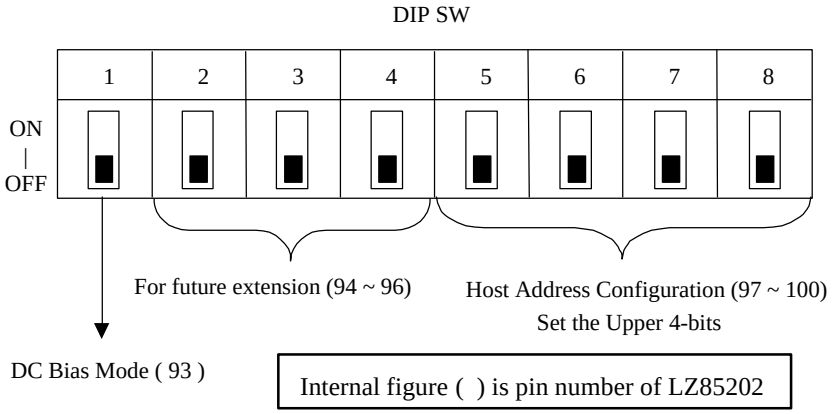



Figure 6.5. DIP SW Assignment

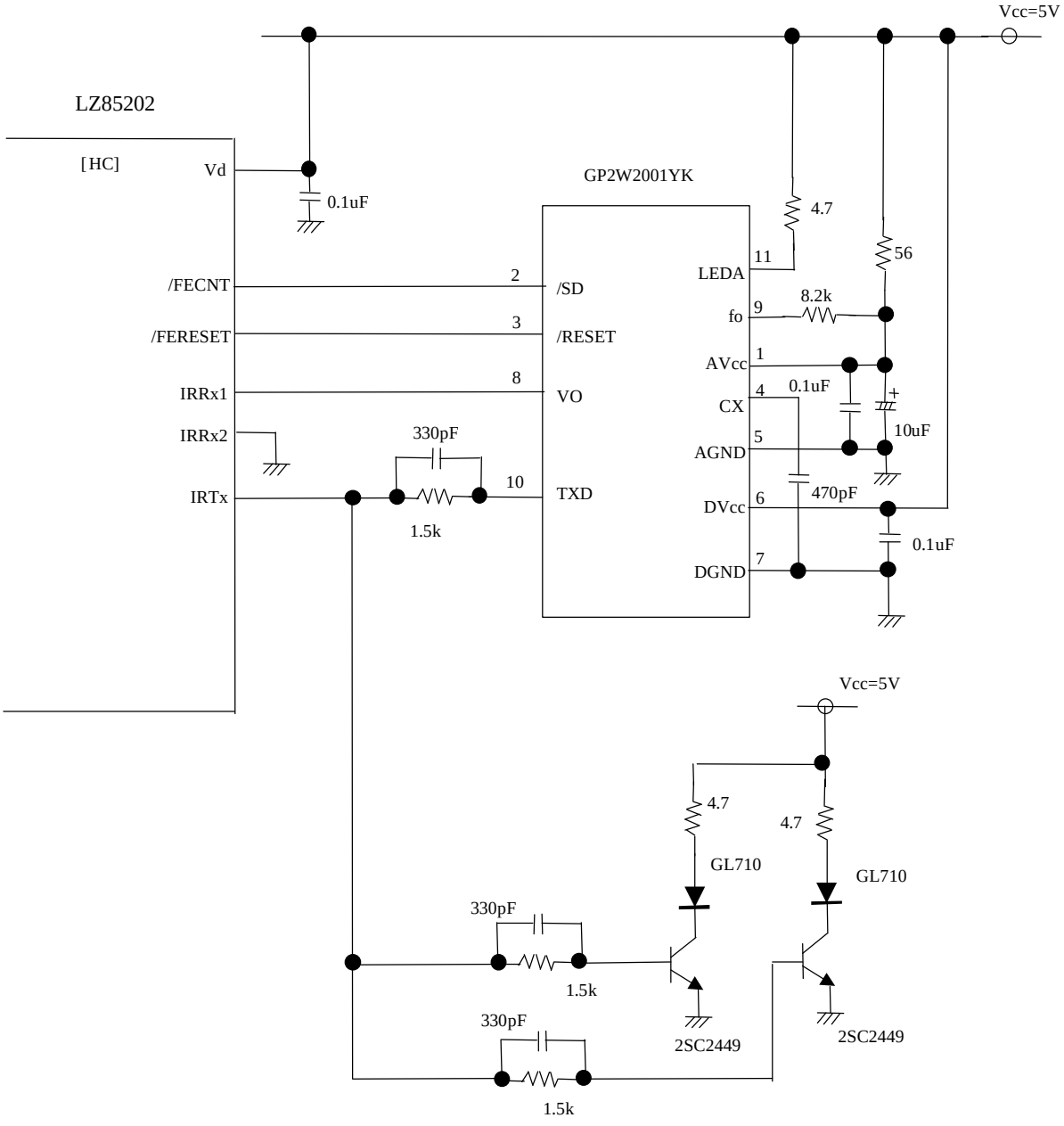
## 7. Reference Document

1) IrDA Control Specification (Formerly IrBus) IrDA CIR (Control IR) Standard Final Specification, Final Revision 1.0, June 30, 1998 (http://www.irda.org)
2) Universal Serial Bus Specification, Revision 1.0, January 15, 1996 (http://www.usb.org)
3) Universal Serial Bus (USB) Device Class Definition for Human Interface devices (HID) Firmware Specification Version 1.0 - Final

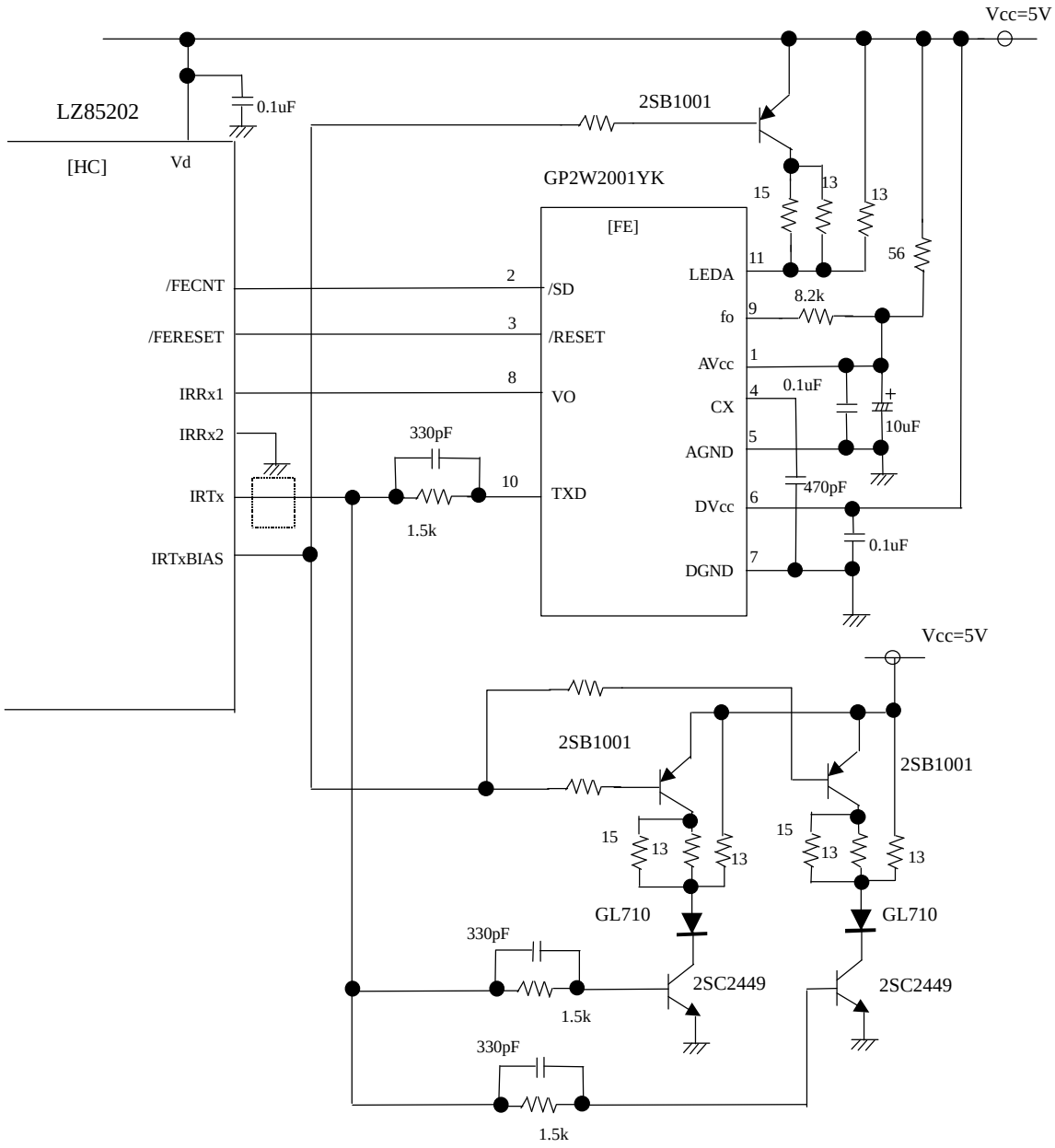
# 8. Appendix

A-1 and A-2 show the referenced circuit diagram between the infrared transceiver and HC. A-1 is for NO support for DC BIAS mode, and A-2 is WITH support of the DC BIAS mode.

## 8.1. Connection Between HC and FE without DC BIAS (Appendix. A-1)



## 8.2. Connection Between HC and FE with DC Bias (Appendix. A-2)




(IMPORTANT NOTICE)
The application circuits provided in Appendix A-1 and A-2 are based on the optimized passive components value as of February 23, 1999, and are for design reference purpose only. Passives and their values are subject to change without notice along with Sharp's internal evaluation purpose.

## 8.3. USB Dongle Reference Circuit Diagram / Parts List

The USB Dongle reference circuit schematic and its parts list are also provided with this document. Parts list is provided below. Please refer to the separate sheet for the circuit schematics.

| No. | Symbol | Device | Ratings | Part No. | Vendor | Qty | Remarks |
|---|---|---|---|---|---|---|---|
| 1 | U3 | HC | | LZ85202 | SHARP | 1 | HC |
| 2 | U5 | IrDA-C Transceiver | | GP2W2001YK | SHARP | 1 | IR transceiver |
| 3 | U2 | Voltage Regulator | | LT1117CST | LT | 1 | SOT-223 |
| 4 | U1 | Serial Bus Transceiver | | PDIUSBP-11D | PHILIPS | 1 | SOP14 |
| 5 | U4 | Serial EEPROM | | AT93C56-10PC-8P3 | ATMEL | 1 | DIP8 |
| 6 | U6 | C-MOS | 74VHCT14A | TC74VHCT14A | TOSHIBA | 1 | |
| 7 | U7 | RESET Control | 4.0V CMOS-TP | RN5VD40CA | RICOH | 1 | |
| 8 | Q1, Q2 | Transistor | | 2SC2449 | FUJI Electric | 2 | |
| 9 | Q3 ~ Q5 | Transistor | | 2SB1001 | HITACHI | 3 | |
| 10 | Y1 | Crystal Oscillator | 12 MHz | AT-49 12 MHz<br>HC-49/U-S 12 MHz | KDS<br>KSS | 1 | |
| 11 | D1 ~ D7, D9 | LED | Green | GL-3PG-8<br>GL-3KG-8 | SHARP | 8 | |
| 12 | D8 | LED | Red | GL-3PR-8 | SHARP | 1 | |
| 13 | D10, D11 | LED | | GL710 | SHARP | 2 | Assist LED |
| 14 | S1 | DIP SW | 4-circuits | DSS-104<br>DAS-4H | FUJISOKU<br>MATSUKYU | 1 | |
| 15 | C8 ~ C10 | Electrolytic Capacitor | 100 uF, 16 WV | 16MV 100HW<br>UVR1C 101MDA | SANYO<br>NICHICON | 3 | |
| 16 | C12 | Electrolytic Capacitor | 4.7 uF, 50 WV | 50MV 4.7HW<br>UVR1H 4R7MDA | SANYO<br>NICHICON | 1 | |
| 17 | C16 | Electrolytic Capacitor | 10 uF, 35 WV | DT1V106M | NEC | 1 | |
| 18 | C6, C7 | Ceramic Capacitor | 10 pF, 50 V | HE40SJSL 100D | KCK | 2 | |
| 19 | C4,C17,C18 | Ceramic Capacitor | 330 pF, 50 V | HE40SJYB 331K | KCK | 3 | |
| 20 | C2 | Ceramic Capacitor | 470 pF, 50 V | HE40SJYB 471K | KCK | 1 | |
| 21 | C1, C3, C5, C11, C13, C14, C15, C19 | Layered Ceramic Capacitor | 0.1 uF, 50 V | RPE132-901F104Z50 | MURATA | 8 | |
| 22 | R5 ~ R13 | Carbon Resistor | 0.2 W, 470 Ω, 5% | R20 T-24J 471 | ROHM | 9 | |
| 23 | R15 | Carbon Resistor | 0.2 W, 4.7k Ω, 5% | R20 T-24J 472 | ROHM | 1 | |
| 24 | R1, R23, R24, R25 | Carbon Resistor | 0.2 W, 1.5 kΩ, 5% | R20 T-24J 152 | ROHM | 4 | |
| 25 | R20 | Carbon Resistor | 0.2 W, 8.2 kΩ, 5% | R20 T-24J 822 | ROHM | 1 | |
| 26 | R2, R14 | Carbon Resistor | 0.2 W, 10 kΩ, 5% | R20 T-24J 103 | ROHM | 2 | |
| 27 | R16 | Carbon Resistor | 0.2 W, 1 MΩ, 5% | R20 T-24J 105 | ROHM | 1 | |
| 28 | R17 | Carbon Resistor | 0 Ω | R20 T-24J 000 | ROHM | 1 | |
| 29 | R4 | Chip Carbon Resistor | 0.1 W, 124 Ω, 1% | MCR10 EZH F1240<br>RK73H2A TD 124ΩF | ROHM<br>KOA | 1 | |
| 30 | R3 | Chip Carbon Resistor | 0.1 W, 205 Ω, 1% | MCR10 EZH F2050<br>RK73H2A TD 205ΩF | ROHM<br>KOA | 1 | |
| 31 | R26 ~ R29, R33, R34 | Carbon Resistor | 0.2 W, 13 Ω, 5% | R20 T-24J 130 | ROHM | 6 | |
| 32 | R30, R31, R35 | Carbon Resistor | 0.2 W, 15 Ω, 5% | R20 T-24J 150 | ROHM | 3 | |
| 33 | R22 | Carbon Resistor | 0.2 W, 56 Ω, 5% | R20 T-24J 560 | ROHM | 1 | |
| 34 | R18, R19 | Carbon Resistor | 0.2 W, 22 Ω, 5% | R20 T-24J 220 | ROHM | 2 | |
| 35 | R32, R36, R37 | Carbon Resistor | 0.2 W, 56 kΩ, 5% | R20 T-24J 563 | ROHM | 3 | |
| 36 | RA1 | Module Resister | 8-elements, 1 MΩ | RGLD 8X 105J<br>RKC1/8B8 1MΩJ<br>RMLS 8J 105 | MURATA<br>KOA<br>ROHM | 1 | |
| 37 | JP3 | Connector | 1 x 2P | PS-2PF-D4T1-PLK1 | JAE | 1 | |
| 38 | | Jumper Pin | | PS-2SH4-1 | JAE | 1 | |
| 39 | JP1 | Connector | 2P | 171826-2 | AMP | 1 | |
| 40 | JP2 | Connector | 4P | UBB-4R-D10T-1 | JST | 1 | |

SHARP

**SHARP CORPORATION Japan**
INTERNATIONAL SALES & MARKETING GROUP
-IC/ELECTRONIC COMPONENTS
22-22 Nagaike-cho, Abeno-ku, OSAKA, 545-8522, JAPAN
PHONE: (81) 6-621-1221
FAX: (81) 6-624-0163

**NORTH AMERICA:**
**SHARP MICROELECTRONICS OF THE AMERICAS**
Microelectronics Group
5700 Northwest, Pacific Rim Boulevard#20
Camas, WA, 98607, U.S.A.
PHONE: (1) 360-834-2500
FAX: (1) 360-834-8903

**EUROPE:**
**SHARP ELECTRONICS (EUROPE) GmbH**
Microelectronics Division
Sonninstrasse 3, 20097 Hamburg, Germany
PHONE: (49) 40-2376-2286
FAX: (49) 40-2376-2232

**TAIWAN:**
**SHARP ELECTRONIC COMPONENTS (TAIWAN) CORPORATION**
8 Fl., No. 16, Sec 4, Nanking E Rd.,
Taipei, Taiwan, Republic of China
PHONE: (886) 2-577-7341
FAX: (886) 2-577-7326 / 2-577-7328

**HONG KONG**
**SHARP - ROXY (HONG KONG) LTD.**
3rd Business Division,
Room 1701 - 1711, Admiralty Centre Tower 1,
Harcourt Road, Hong Kong
PHONE: (852) 28229311
FAX: (852) 28660779

**SINGAPORE:**
**SHARP ELECTRONICS (SINGAPORE) PTE., LTD.**
438A, Alexandra Road, #05-01/02,
Alexandra Technopark, Singapore 119967
PHONE: (65) 271-3566
FAX: (65) 271-3855

**KOREA:**
**SHARP ELECTRONIC COMPONENTS (KOREA) CORPORATION**
RM 501 Geosung B/D.
541, Dohwa-dong, Mapo-ku, Seoul, Korea
PHONE: (82) 2-711-5813 ~ 8
FAX: (82) 2-711-5819