DATA SHEET

O K I A S I C P R O D U C T S

# W712
# Universal Serial Bus Controller
# 0.25µm, 0.35µm, 0.5µm Technology
# Mega Macrofunction

**July 2001**

Oki Semiconductor

**CONTENTS**

# Oki Semiconductor

# W712 USB Device Controller

**Mega Macrofunction**

## DESCRIPTION

The Universal Serial Bus (USB) Device Controller mega macrofunction is a featured element in Oki's Sea of Gates (SOG), Customer Structured Array (CSA) families, and Standard Cell (SC) families. Oki's USB mega macrofunction provides a USB interface, control/status block, first-in, first-out (FIFO) control, and application interface in two highly integrated submodules for system design interfaces based on the USB protocol. The submodule partitioning allows custom configurations to be easily developed. The USB mega macrofunction connects an industry standard USB interface with a microprocessor-style parallel application interface. This straightforward interface permits easy integration of the USB mega macrofunction to the target application, allowing designers to reduce development time, risk, and time-to-market. Oki's W712 USB Device Controller mega macrofunction provides a complete USB device interface solution and is fully compliant with the Universal Serial Bus 1.1 specification. (For more details on the Universal Serial Bus 1.1 specification, refer to www.usb.org.)

## FEATURES

- USB 1.1 compliant
- Full-speed (12 Mbps) and low-speed (1.5 Mbps) support
- Microprocessor-style parallel application interface
- Supports isochronous, control, interrupt, and bulk transfers
- Supports four transmit FIFOs
  - Three 64 byte
  - One 2 Kbyte (2-level)
- Supports four receive FIFOs
  - Three 64 byte
  - One 2 Kbyte (2-level)
- Supports one control endpoint and six additional endpoint addresses
- Expandable up to 32 endpoint addresses
- Customizable to specific application requirements

### Supported ASIC Families

| Family Name | Family Type | Process |
|---|---|---|
| MSM13R/12R | SOG | 0.5μm DLM/TLM |
| MSM98R | CSA | 0.5μm TLM |
| MSM30R/32R | SOG | 0.5μm DLM/TLM |
| MSM92R | CSA | 0.5μm TLM |
| MSM13Q/14Q | CBA | 0.35μm TLM/QLM |
| MSM98Q/99Q | CSA | 0.35μm TLM/QLM |
| MG73Q/74Q | CSA | 0.35μm TLM/QLM |
| MG113P/114P/115P | SOG | 0.25μm TLM/QLM/FLM |
| MG73P/74P/75P | CSA | 0.25μm TLM/QLM/FLM |
| MG87P3/87P4/87P5 | SC | 0.25μm TLM/QLM/FLM |

## Recommended Operating Conditions ($V_{SS}$ = 0 V)

| Parameter | Process | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Power supply voltage | 0.5μm, 0.35μm | $V_{DD}$ Core | 3.0 | 3.3 | 3.6 | V |
| | 0.25 μm | $V_{DD}$ Core | 2.25 | 2.5 | 2.75 | V |
| Operating temperature | 0.5μm, 0.35μm, 0.25 μm | $T_j$ | -40 | +25 | +85 | °C |

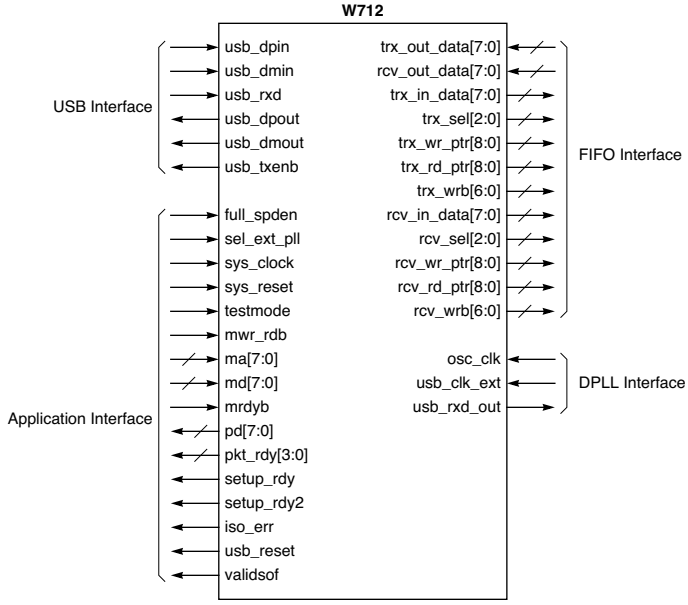



**Figure 1. Logic Symbol**

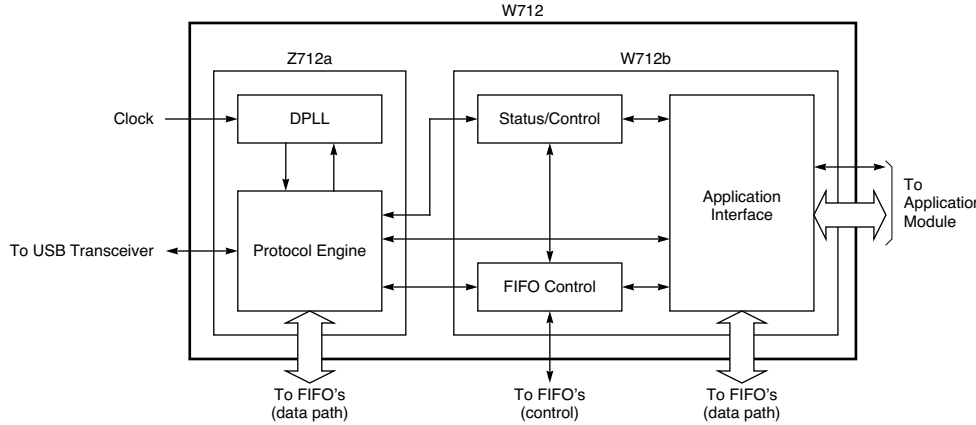



**Figure 2. W712 Block Diagram**

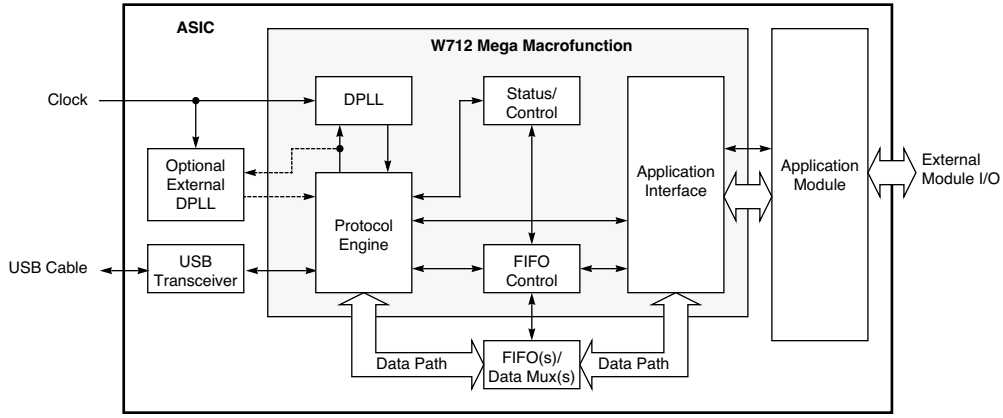



**Figure 3. Example USB Mega Macrofunction Application**

## SIGNAL DESCRIPTIONS

### USB Interface

<table>
<tr><th>Signal</th><th>Type</th><th>Assertion</th><th>Description</th></tr>
<tr><td>usb_dpin</td><td>Input</td><td>—</td><td>USB Data Plus In. This input and the usb_dmin input are the received single ended data from the USB transceiver. The table below shows values and results for these signals.
<table>
<tr><th>usb_dpin</th><th>usb_dmin</th><th>Result</th></tr>
<tr><td>0</td><td>0</td><td>SE0</td></tr>
<tr><td>0</td><td>1</td><td>Logic “0”</td></tr>
<tr><td>1</td><td>0</td><td>Logic “1”</td></tr>
<tr><td>1</td><td>1</td><td>Undefined</td></tr>
</table></td></tr>
<tr><td>usb_dmin</td><td>Input</td><td>—</td><td>USB Data Minus In. This input and the usb_dpin input are the received single ended data from the USB transceiver. See the table above for more information.</td></tr>
<tr><td>usb_dpout</td><td>Output</td><td>—</td><td>USB Data Plus Out. This output and the usb_dmout signal come from the USB transmit engine and drive the differential output buffers. The table below shows values and results for these signals.
<table>
<tr><th>usb_dpout</th><th>usb_dmout</th><th>Result</th></tr>
<tr><td>0</td><td>0</td><td>SE0</td></tr>
<tr><td>0</td><td>1</td><td>Logic “0”</td></tr>
<tr><td>1</td><td>0</td><td>Logic “1”</td></tr>
<tr><td>1</td><td>1</td><td>Undefined</td></tr>
</table></td></tr>
<tr><td>usb_dmout</td><td>Output</td><td>—</td><td>USB Data Minus Out. This output and the usb_dpout signal come from the USB transmit engine and drive the differential output buffers. See the table above for more information.</td></tr>
<tr><td>usb_rxd</td><td>Input</td><td>—</td><td>USB Differential Received Data. This input comes from the USB differential receiver, and connects to the W712 mega macrofunction.</td></tr>
<tr><td>$\overline{\text{usb\_txenb}}$</td><td>Output</td><td>LOW</td><td>USB 3-State Output Enable. This signal connects to the transceiver EB input through an inverter gate. When the W712 mega macrofunction asserts this signal, the transceiver transmits data on the USB bus. See Appendix for the USB transceiver Data Sheets.</td></tr>
</table>

## Application Interface

| Signal | Type | Assertion | Description |
|---|---|---|---|
| sys_clock | Input | — | **Clock**. Attach a 12-MHz clock signal to this input for full-speed operation and 1.5 MHz for low-speed operation. |
| sys_reset | Input | HIGH | **W712 Reset.** Asserting this signal HIGH resets the W712 mega macrofunction. The application module is required to assert this signal at power-on. |
| mwr_rdb | Input | — | **Write/Read Select.** When external application logic drives this signal HIGH, the application is in WRITE mode. When driven LOW, the application is in READ mode. External application logic asserts this signal HIGH when writing data to the transmit FIFOs or to the register files. External application logic asserts this signal LOW when reading data from the receiving FIFOs or from the register files. The register files contain information describing the function and transaction status. |
| usb_reset | Output | HIGH | **USB Reset**. This is the reset signal from the USB device controller. |
| ma[7:0] | Input | — | **Address Bus.** These eight inputs receive the address of the register files in the USB device controller. |
| md[7:0] | Input | — | **Input Data Bus.** These eight inputs receive the data to be stored in the register files or transmit FIFOs. |
| $\overline{\text{mrdyb}}$ | Input | LOW | **Data Strobe.** When driven LOW and in WRITE mode, the data on the md[7:0] signal lines are valid for writing. When driven LOW and in READ mode, the data on the pd[7:0] signals are valid for reading. |
| pd[7:0] | Output | — | **Output Data Bus**. These eight outputs transmit data received from either the register files or the receive FIFOs. |
| pkt_rdy[3:0] | Output | HIGH | **Packet Ready**. When the W712 asserts this signal, it indicates that one of the four receive FIFOs contains valid data. The application reads the data through the pd[7:0] bus. |
| full_spden | Input | — | **USB Full Speed Enable.** The application module asserts this pin to select full-speed operation and deasserts the pin to select low-speed operation. |
| setup_rdy | Output | HIGH | **Setup Ready.** Asserting this signal HIGH indicates that an 8-byte SETUP data has been received from the USB bus. |
| iso_err | Output | HIGH | **Isochronous Error.** Used for loopback testing or to indicate isochronous data has been received with DATA1 PID. |
| validsof | Output | HIGH | **Valid SOF**. This signal is asserted for two bit times, asynchronous to sys_clock, and indicates a valid SOF token is received when asserted HIGH. |
| sel_ext_pll | Input | HIGH | **Select External PLL**. Assertion of this signal selects the external PLL option. |
| setup_rdy2 | Output | HIGH | **Second Setup Ready**. Assertion of this signal indicates that a new 8-byte SETUP DATA has been received, while internally the device controller still sees the “setup_rdy” signal asserted. This signal is asserted for two bit times, asynchronous to sys_clock. |
| testmode | Input | HIGH | **Testmode**. Asserting this signal invokes a loopback test mode. |

## FIFO Interface

| Signal | Type | Assertion | Description |
|---|---|---|---|
| trx_out_data[7:0] | Input | — | **Transmit FIFO(s} data output**. Output data from the transmission RAM selected for reading. |
| rcv_out_data[7:0] | Input | — | **Receive FIFO(s) data output**. Output data from the receiving RAM selected for reading. |
| trx_in_data[7:0] | Output | — | **Transmit FIFO(s) data input**. Input data to all transmission RAMs. |
| trx_sel[2:0] | Output | HIGH | **Transmit FIFO(s) select**. Selects one of the seven transmission RAMs for reading. |
| trx_wr_ptr[8:0] | Output | — | **Transmit FIFO(s) write pointer**. Write address to all transmission RAMs. |
| trx_rd_ptr[8:0] | Output | — | **Transmit FIFO(s) read pointer**. Read address to all transmission RAMs. |
| trx_wrb[6:0] | Output | LOW | **Transmit FIFO(s) write strobe**. Write enable. One bit per transmission RAM. |
| rcv_in_data[7:0] | Output | — | **Receive FIFO(s) data input**. Input data to all receiving RAMs. |
| rcv_sel[2:0] | Output | HIGH | **Receive FIFO(s) select**. Selects one of the seven receiving RAMs for reading. |
| rcv_wr_ptr[8:0] | Output | — | **Receive FIFO(s) write pointer**. Write address to all receiving RAMs. |
| rcv_rd_ptr[8:0] | Output | — | **Receive FIFO(s) read pointer**. Read address to all receiving RAMs. |
| rcv_wrb[6:0] | Output | LOW | **Receive FIFO(s) write strobe**. Write enable. One bit per receiving RAM. |

## DPLL Interface

| Signal | Type | Assertion | Description |
|---|---|---|---|
| osc_clk | Input | — | **Oscillator Clock**. Attach a 48 MHz clock signal for full-speed operation or a 6 MHz clock signal for low-speed operation. |
| usb_clk_ext | Input | — | **USB Clock External**. This is the output clock signal from an external DPLL. This clock should run at 12 MHz for full-speed operation or 1.5 MHz for low-speed operation. If an external DPLL is not used, this pin should be connected to VDD or GND. |
| usb_rxd_out | Output | — | **Synchronized USB Differential Received Data**. This signal comes from the USB differential receiver and is synchronized with the oscillator input. This signal connects to the external DPLL if it is used. |

## FUNCTIONAL DESCRIPTION

The W712 controller consists of two submodules: the Z712a hard macro and the W712b soft macro, each containing multiple function blocks. The Z712a includes the Protocol Engine, Digital Phase-Locked Loop (DPLL), and Timer Blocks. The W712b includes the Status/Control, FIFO Control, Application Interface, Frame Timer Synthesizer, and remote wakeup blocks.

### Protocol Engine

The Protocol Engine handles the USB communication protocol. It performs packet sequencing, signal generation/detection, cyclic redundancy check (CRC) generation/checking, non-return-to-zero-invert (NRZI) data encoding, bit-stuffing and packet ID (PID) generation/decoding.

### Digital Phase-Locked Loop

The DPLL extracts the clock and data from the USB differential received data.

### Timer

The Timer block monitors idle time on the USB bus.

### Status/Control

The Status/Control block uses transfer type and FIFO state information to manage the reception and transmission of USB data. This block monitors the transaction status and communicates control events to the application via the Application Interface.

### FIFO Control

The FIFO control block manages all FIFO operations for transmitting and receiving USB data sets. The W712 supports eight FIFOs (four transmit and four receive). They can be configured as described in the table below.

### FIFO Configuration

| FIFO Type | Endpoint Address | Programmable | Function |
|---|---|---|---|
| Transmit | 0 | 64 bytes | Control Transfers |
| Transmit | 5 | 64 bytes | Interrupt and Bulk Transfers |
| Transmit | 6 | 64 bytes | Interrupt and Bulk Transfers |
| Transmit | 7 | 2 Kbytes | Isochronous, Interrupt, or Bulk Transfers |
| Receive | 0 | 64 bytes | Control Transfers |
| Receive | 1 | 64 bytes | Bulk Transfers |
| Receive | 2 | 64 bytes | Bulk Transfers |
| Receive | 3 | 2 Kbytes | Isochronous or Bulk Transfers |

Endpoints 3 and 7 are 2-level FIFOs which support up to two separate data sets of variable sizes. All FIFOs have flags that detect a full or empty FIFO and have the capability of re-transmitting or re-receiving the current data set.

### Application Interface

The Application Interface uses a microprocessor-style parallel interface between the customer's module and the W712. The integration of the W712 into a customer-designed module is limited only by the available gates and I/O pins in the array. The Customer Application Module may have its own external I/O, which does not interface with the W712. All application interface signals are unidirectional and are either inputs or outputs of the W712.

### Frame Timer Synthesizer

This block synthesizes the SOF signal in the event of a SOF token is lost.

### Remote Wakeup

This block provides support for the remote wakeup function.

### USB Transfers

The W712 supports all four transfer types defined by the USB specification. These are:

- **Control**—transfers must be supported by every peripheral for configuration, command and status information flow between the host and peripheral.
- **Isochronous**—transfers provide guaranteed bus access and constant data rates for USB devices.
- **Interrupt**—transfers support human input devices that need to communicate small amounts of data infrequently.
- **Bulk**—transfers enable devices to transfer large amounts of data as bus bandwidth becomes available.

### USB Interface

The W712 connects to the USB via Oki's universal USB transceivers. The USB specific I/O converts the W712's internal unidirectional signals into compliant USB signals. The USB transceiver allows the designers' application module to interface with the physical layer of the USB to transmits and receive serial data at both full-speed (12 Mbps) and low-speed (1.5 Mbps) data rates. (See Appendix for Oki's USB transceiver Data Sheets.)

### Registers

Oki's W712 provides status/control register for the application to store its current configuration information as well as to read/write the status flags for the USB flow.

Reading and writing the registers are achieved via the MWR_RDB, MRDYB, MA[7:0], MD[7:0], and PD[7:0] signals.

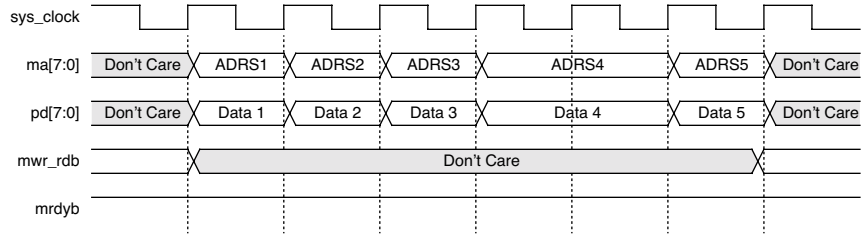



**Figure 4. USB Status/Control Register Read**

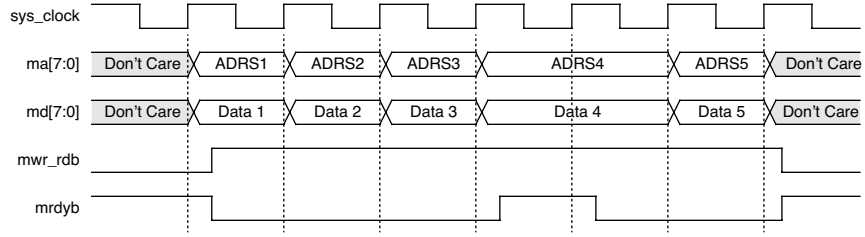



**Figure 5. USB Status/Control Register Write**

**Device Address Register**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| reserved | ADR6 | ADR5 | ADR4 | ADR3 | ADR2 | ADR1 | ADR0 | Device Address Register |
| N/A | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Read/Write Map |

The USB Device Address is set to zero (0) during a reset. Bit 7 of the register is reserved since USB uses a 7-bit address to select up to 127 physical devices. ADR6 is the MSB bit and ADR0 is the LSB bit.

| | |
|---|---|
| Application Write | MA[7:0] = 40H (01000000B)<br>MD[7:0] = Device Address<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read | MA[7:0] = COH (11000000B)<br>PD[7:0] = Device Address<br>MRDYB = 1 |

### Device State Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| reserved | reserved | reserved | REWP | SPD | CFG | ADDR | DEF | Device State Register |
| N/A | N/A | N/A | R/W | R | R/W | R/W | R/W | Read/Write Map |

DEF: Default State bit. This bit is set to 1 during a reset and the USB device is in Default State. Application reset this bit to 0 if the device is not in Default State.

ADDR: Address State bit. Application sets it to 1 if the USB device is in Address State. Application reset it to 0 if the USB device is not in Address State. This bit is reset to 0 during a reset.

CFG: Configure State bit. Application sets it to 1 if the USB device is in Configured State. Application reset it to 0 if the USB device is not in Configured State. This bit is reset to 0 during a reset.

SPD: Suspend State bit. USB Device Controller sets it to 1 if the USB device is in Suspended State. USB Device Controller resets it to 0 if the USB device is out of the Suspended State. This bit is reset to 0 during a reset.

REWP: Remote Wakeup bit. Application sets it to 1 if the application wants to initiate a remote wakeup. The application needs to confirm its remote wakeup feature is enabled before setting this bit. USB Device Controller resets it to 0 after the remote wakeup is completed. This bit is reset to 0 during a reset.

| | |
|---|---|
| Application Write | MA[7:0] = 41H (01000001B)<br>MD[7:0] = Desired value<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read | MA[7:0] = C1H (11000001B)<br>PD[7:0] = Device State<br>MRDYB = 1 |

### Packet Ready Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| TR7P | TR6P | TR5P | TR0P | RC3P | RC2P | RC1P | RC0P | Packet Ready Register |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Read/Write Map |

RC0P: RECEIVE PACKET READY 0 bit. USB Device Controller sets this bit to 1 if a valid data packet is received in FIFO #0 for Endpoint 0. Application resets this bit to 0 when finishes reading the data packet.

RC1P: RECEIVE PACKET READY 1 bit. USB Device Controller sets this bit to 1 if a valid data packet is received in FIFO #1 for Endpoint 1. Application resets this bit to 0 when finishes reading the data packet.

RC2P: RECEIVE PACKET READY 2 bit. USB Device Controller sets this bit to 1 if a valid data packet is received in FIFO #2 for Endpoint 2. Application resets this bit to 0 when finishes reading the data packet.

RC3P: RECEIVE PACKET READY 3 bit. USB Device Controller sets this bit to 1 if a valid data packet is received in FIFO #3 for Endpoint 3. Application resets this bit to 0 when finishes reading the data packet.

TR0P: TRANSMIT PACKET READY 0 bit. Application sets this bit to 1 if it finishes writing data into FIFO #4 for Endpoint 0 for transmission. USB Device Controller resets this bit to 0 if it transmitted the data packet upstream and received an ACK from the HOST.

TR5P: TRANSMIT PACKET READY 5 bit. Application sets this bit to 1 if it finishes writing data into FIFO #5 for Endpoint 5 for transmission. USB Device Controller resets this bit to 0 if it transmitted the data packet upstream and received an ACK from the HOST.

TR6P: TRANSMIT PACKET READY 6 bit. Application sets this bit to 1 if it finishes writing data into FIFO #6 for Endpoint 6 for transmission. USB Device Controller resets this bit to 0 if it transmitted the data packet upstream and received an ACK from the HOST.

TR7P: TRANSMIT PACKET READY 7 bit. Application sets this bit to 1 if it finishes writing data into FIFO #7 for Endpoint 7 for transmission. USB Device Controller resets this bit to 0 if it transmitted the data packet upstream and received an ACK from the HOST.

| **Application Write to RC0P, RC1P, RC2P, RC3P** | | | | |
|---|---|---|---|---|
| **MA[7:0]** | **MD[7:0]** | **MWR_RDB** | **MRDYB** | **BIT** |
| 48H (01001000B) | 01H (00000001B) | 1 | 0 | RC0P |
| 48H (01001000B) | 02H (00000010B) | 1 | 0 | RC1P |
| 48H (01001000B) | 04H (00000100B) | 1 | 0 | RC2P |
| 48H (01001000B) | 08H (00001000B) | 1 | 0 | RC3P |

| **Application Write to TR0P, TR5P, TR6P, TR7P** | | | | |
|---|---|---|---|---|
| **MA[7:0]** | **MD[7:0]** | **MWR_RDB** | **MRDYB** | **BIT** |
| 48H (01001000B) | 10H (00010000B) | 1 | 0 | TR0P |
| 48H (01001000B) | 20H (00100000B) | 1 | 0 | TR5P |
| 48H (01001000B) | 40H (01000000B) | 1 | 0 | TR6P |
| 48H (01001000B) | 80H (10000000B) | 1 | 0 | TR7P |

| | |
|---|---|
| Application Read | MA[7:0] = C8H (11001000B)<br>PD[7:0] = Packet Ready Flag<br>MRDYB = 1 |

**Endpoint 0 Receive Byte Count Register**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| reserved | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Endpoint 0 Rx Byte Count Register |
| N/A | R | R | R | R | R | R | R | Read/Write Map |

This register reports the number of bytes stored in RECEIVE FIFO #0 associated with Endpoint 0. The value represents the amount of data being received from the USB HOST in a particular DATA0/DATA1 data packet.

| | |
|---|---|
| Application Read: | MA[7:0] = C9H (11001001B)<br>PD[7:0] = EP0 Byte count<br>MRDYB = 1 |

**Endpoint 1 Receive Byte Count Register**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| reserved | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Endpoint 1 Rx Byte Count Register |
| N/A | R | R | R | R | R | R | R | Read/Write Map |

This register reports the number of bytes stored in RECEIVE FIFO #1 associated with Endpoint 1. The value represents the amount of data being received from the USB HOST in a particular DATA0/DATA1 data packet.

| | |
|---|---|
| Application Read: | MA[7:0] = CAH (11001010B)<br>PD[7:0] = EP1 Byte count<br>MRDYB = 1 |

**Endpoint 2 Receive Byte Count Register**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| reserved | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Endpoint 2 Rx Byte Count Register |
| N/A | R | R | R | R | R | R | R | Read/Write Map |

This register reports the number of bytes stored in RECEIVE FIFO #2 associated with Endpoint 2. The value represents the amount of data being received from the USB HOST in a particular DATA0/DATA1 data packet.

| | |
|---|---|
| Application Read: | MA[7:0] = CBH (11001011B)<br>PD[7:0] = EP2 Byte count<br>MRDYB = 1 |

### Endpoint 3 Receive Byte Count (LSB) Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| reserved | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Endpoint 3 Rx Byte Count (LSB) Register |
| N/A | R | R | R | R | R | R | R | Read/Write Map |

This register reports the LSB value of the number of bytes stored in RECEIVE FIFO #3 associated with Endpoint 3.

| | |
|---|---|
| Application Read: | MA[7:0] = CCH (11001100B)<br>PD[7:0] = EP3 LSB Byte count<br>MRDYB = 1 |

### Endpoint 3 Receive Byte Count (MSB) Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| reserved | reserved | reserved | reserved | reserved | Bit 2 | Bit 1 | Bit 0 | Endpoint 3 Rx Byte Count (MSB) Register |
| N/A | N/A | N/A | N/A | N/A | R | R | R | Read/Write Map |

This register reports the MSB value of the number of bytes stored in RECEIVE FIFO #3 associated with Endpoint 3.

| | |
|---|---|
| Application Read: | MA[7:0] = CDH (11001101B)<br>PD[7:0] = EP3 MSB Byte count<br>MRDYB = 1 |

### BmRequestType Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | BmRequestType Register |
| R | R | R | R | R | R | R | R | Read/Write Map |

This register stores the BmRequestType field of the eight bytes setup packet. The detail of this field can be found from the USB specification V1.0 in section 9.3.

Bit[7]: Data Transfer Direction.

Bit[6:5]: Type.

Bit[4:0]: Recipient.

| | |
|---|---|
| Application Read: | MA[7:0] = D0H (11010000B)<br>PD[7:0] = BmRequestType Field<br>MRDYB = 1 |

### BRequest Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | BRequest Register |
| R | R | R | R | R | R | R | R | Read/Write Map |

This register stores the BRequest field of the eight bytes setup packet. The detail of this field can be found from the USB specification V1.0 in section 9.3.

| | |
|---|---|
| Application Read: | MA[7:0] = D1H (11010001B)<br>PD[7:0] = BRequest Field<br>MRDYB = 1 |

### WValueLSB Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | WValueLSB Register |
| R | R | R | R | R | R | R | R | Read/Write Map |

This register stores the WValueLSB field of the eight bytes setup packet. The detail of this field can be found from the USB specification V1.0 in section 9.3.

Bit[7:0]: The least significant byte of the word-sized field.

| | |
|---|---|
| Application Read: | MA[7:0] = D2H (11010010B)<br>PD[7:0] = WValueLSB Field<br>MRDYB = 1 |

### WValueMSB Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | WValueMSB Register |
| R | R | R | R | R | R | R | R | Read/Write Map |

This register stores the WValueMSB field of the eight bytes setup packet. The detail of this field can be found from the USB specification V1.0 in section 9.3.

Bit[7:0]: The most significant byte of the word-sized field.

| | |
|---|---|
| Application Read: | MA[7:0] = D3H (11010011B)<br>PD[7:0] = WValueMSB Field<br>MRDYB = 1 |

### WIndexLSB Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | WIndexLSB Register |
| R | R | R | R | R | R | R | R | Read/Write Map |

This register stores the WIndexLSB field of the eight bytes setup packet. The detail of this field can be found from the USB specification V1.0 in section 9.3.

Bit[7:0]: The least significant byte of the Index field.

| | |
|---|---|
| Application Read: | MA[7:0] = D4H (11010100B)<br>PD[7:0] = WIndexLSB Field<br>MRDYB = 1 |

### WIndexMSB Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | WIndexMSB Register |
| R | R | R | R | R | R | R | R | Read/Write Map |

This register stores the WIndexMSB field of the eight bytes setup packet. The detail of this field can be found from the USB specification V1.0 in section 9.3.

Bit[7:0]: The most significant byte of the Index field.

| | |
|---|---|
| Application Read: | MA[7:0] = D5H (11010101B)<br>PD[7:0] = WIndexMSB Field<br>MRDYB = 1 |

### WLengthLSB Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | WLengthLSB Register |
| R | R | R | R | R | R | R | R | Read/Write Map |

This register stores the WLengthLSB field of the eight bytes setup packet. The detail of this field can be found from the USB specification V1.0 in section 9.3.

Bit[7:0]: The least significant byte of the WLength field.

| | |
|---|---|
| Application Read: | MA[7:0] = D6H (11010110B)<br>PD[7:0] = WLengthLSB Field<br>MRDYB = 1 |

### WLengthMSB Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | WLengthMSB Register |
| R | R | R | R | R | R | R | R | Read/Write Map |

This register stores the WLengthMSB field of the eight bytes setup packet. The detail of this field can be found from the USB specification V1.0 in section 9.3.

Bit[7:0]: The most significant byte of the WLength field.

| | |
|---|---|
| Application Read: | MA[7:0] = D7H (11010111B)<br>PD[7:0] = WLengthMSB Field<br>MRDYB = 1 |

### Receive EndPoint 0 Payload Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | Receive EndPoint 0 Payload Register |
| R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the MaxPacketSize information for Receive Endpoint 0.

PYLD[6:0]: MaxPacketSize bits. The size of receive EndPoint 0 can be up to 64 bytes.

RESV[7]: Reserved bits. This bits is assigned the value 0.

| | |
|---|---|
| Application Write: | MA[7:0] = 62H (01100010B)<br>MD[7:0] = MaxPacketSize<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = E2H (11100010B)<br>PD[7:0] = MaxPacketSize<br>MRDYB = 1 |

### EndPoint 1 Configuration Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| OVFL | FIFO | FIFO | FIFO | XFER | XFER | STAL | CFG | EndPoint 1 Configuration Register |
| R | R | R | R | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the configuration information for Endpoint 1.

CFG: Configuration Bit of EndPoint 1. System/USB Reset resets this bit to 0. Application sets this bit to 1 before the device can begin communication with the HOST through EndPoint 1.

STAL: STALL bit. System/USB Reset resets this bit to 0. Application sets this bit to 1 if EndPoint 1 is STALLed. Application resets this bit to 0 if the STALL condition of EndPoint 1 is cleared.

XFER[3:2]: Transfer type bits. Application decides the type.
XFER[3:2] = 00 = Control transfer
XFER[3:2] = 10 = Bulk transfer

FIFO[6:4]: FIFO Number bits.
FIFO[6:4] = 001
EndPoint 1 is assigned to FIFO #1

OVFL: OverFlow bits. USB Device Controller sets this bit to 1 when the received data bytes in FIFO #1 exceed the MaxPacketSize specified for EndPoint 1. Resetting the FIFO #1 resets this bit to 0.

| | |
|---|---|
| Application Write: | MA[7:0] = 64H (01100100B)<br>MD[7:0] = EndPoint 1 information<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = E4H (11100100B)<br>PD[7:0] = EndPoint 1 information<br>MRDYB = 1 |

### EndPoint 1 Toggling Bit Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | RESV | RESV | RESV | RESV | RESV | RESV | TOGL | EndPoint 1 Toggling Bit Register |
| R | R | R | R | R | R | R | R/W | Read/Write Map |

This register stores the toggling bit information for Endpoint 1.

TOGL: Toggling Bit of EndPoint 1.

RESV[7:1]: Reserved bits. These bits are assigned the value 0000000.

| | |
|---|---|
| Application Write to reset: | MA[7:0] = 65H (01100101B)<br>MD[7:0] = 01H (00000001B)<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = E5H (11100101B)<br>PD[7:0] = Toggling bit<br>MRDYB = 1 |

### EndPoint 1 Payload Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | EndPoint 1 Payload Register |
| R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the MaxPacketSize information for Endpoint 1.

PYLD[6:0] MaxPacketSize bits. The size of EndPoint 1 can be up to 64 bytes.
:

RESV[7]: Reserved bits. This bits is assigned the value 0.

| | |
|---|---|
| Application Write: | MA[7:0] = 66H (01100110B)<br>MD[7:0] = MaxPacketSize<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = E6H (11100110B)<br>PD[7:0] = MaxPacketSize<br>MRDYB = 1 |

### EndPoint 2 Configuration Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| OVFL | FIFO | FIFO | FIFO | XFER | XFER | STAL | CFG | EndPoint 2 Configuration Register |
| R | R | R | R | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the configuration information for Endpoint 2.

CFG: Configuration Bit of EndPoint 2. System/USB Reset resets this bit to 0. Application sets this bit to 1 before the device can begin communication with the HOST through EndPoint 2.

STAL: STALL bit. System/USB Reset resets this bit to 0. Application sets this bit to 1 if EndPoint 2 is STALLed. Application resets this bit to 0 if the STALL condition of EndPoint 2 is cleared.

XFER[3:2]: Transfer type bits. Application decides the type.
XFER[3:2] = 00 = Control transfer
XFER[3:2] = 10 = Bulk transfer

FIFO[6:4]: FIFO Number bits.
FIFO[6:4] = 010
EndPoint 2 is assigned to FIFO #2

OVFL: OverFlow bits. USB Device Controller sets this bit to 1 when the received data bytes in FIFO #2 exceed the MaxPacketSize specified for EndPoint 2. Resetting the FIFO #2 resets this bit to 0.

| | |
|---|---|
| Application Write: | MA[7:0] = 68H (01101000B)<br>MD[7:0] = EndPoint 2 information<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = E8H (11101000B)<br>PD[7:0] = EndPoint 2 information<br>MRDYB = 1<br>EndPoint 2 Toggling bit register |

### EndPoint 2 Toggling Bit Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | RESV | RESV | RESV | RESV | RESV | RESV | TOGL | EndPoint 2 Configuration Register |
| R | R | R | R | R | R | R | R/W | Read/Write Map |

This register stores the toggling bit information for Endpoint 2.

TOGL: Toggling Bit of EndPoint 2.

RESV[7:1] : Reserved bits. These bits are assigned the value 0000000.

| | |
|---|---|
| Application Write to reset: | MA[7:0] = 69H (01101001B)<br>MD[7:0] = 01H (00000001B)<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = E9H (11101001B)<br>PD[7:0] = Toggling bit<br>MRDYB = 1 |

### EndPoint 2 Payload Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | EndPoint 2 Payload Register |
| R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the MaxPacketSize information for Endpoint 2.

PYLD[6:0]: MaxPacketSize bits. The size of EndPoint 2 can be up to 64 bytes.

RESV[7]: Reserved bits. This bits is assigned the value 0.

| | |
|---|---|
| Application Write: | MA[7:0] = 6AH (01101010B)<br>MD[7:0] = MaxPacketSize<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = EAH (11101010B)<br>PD[7:0] = MaxPacketSize<br>MRDYB = 1 |

### EndPoint 3 Configuration Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| OVFL | FIFO | FIFO | FIFO | XFER | XFER | STAL | CFG | EndPoint 3 Configuration Register |
| R | R | R | R | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the configuration information for Endpoint 3.

CFG: Configuration Bit of EndPoint 3. System/USB Reset resets this bit to 0. Application sets this bit to 1 before the device can begin communication with the HOST through EndPoint 3.

STAL: STALL bit. System/USB Reset resets this bit to 0. If EndPoint 3 is for Isochronous transfer, application should set this bit to 0. For other types of transfer, application sets this bit to 1 if EndPoint 3 is STALLed, and resets this bit to 0 if the STALL condition of EndPoint 3 is cleared.

XFER[3:2]: Transfer type bits. Application decides the type.
XFER[3:2] = 00 = Control transfer
XFER[3:2] = 01 = Isochronous transfer
XFER[3:2] = 10 = Bulk transfer

FIFO[6:4]: FIFO Number bits.
FIFO[6:4] = 011
EndPoint 3 is assigned to FIFO #3

OVFL: OverFlow bits. USB Device Controller sets this bit to 1 when the received data bytes in FIFO #3 exceed the MaxPacketSize specified for EndPoint 3. Resetting the FIFO #3 resets this bit to 0.

| | |
|---|---|
| Application Write: | MA[7:0] = 6CH (01101100B)<br>MD[7:0] = EndPoint 3 information<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = ECH (11101100B)<br>PD[7:0] = EndPoint 3 information<br>MRDYB = 1 |

### EndPoint 3 Toggling Bit Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | RESV | RESV | RESV | RESV | RESV | RESV | TOGL | EndPoint 3 Toggling Bit Register |
| R | R | R | R | R | R | R | R/W | Read/Write Map |

This register stores the toggling bit information for Endpoint 3.

TOGL: Toggling Bit of EndPoint 3. If EndPoint 3 is for Isochronous transfer, this bit is reserved.

RESV[7:1]: Reserved bits. These bits are assigned the value 0000000.

| | |
|---|---|
| Application Write to reset: | MA[7:0] = 6DH (01101101B)<br>MD[7:0] = 01H (00000001B)<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = EDH (11101101B)<br>PD[7:0] = Toggling bit<br>MRDYB = 1 |

### EndPoint 3 Payload LSB Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | EndPoint 3 Payload LSB Register |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the LSB MaxPacketSize information for Endpoint 3. The size of EndPoint 3 can be up to 1023 bytes.

PYLD[7:0] LSB MaxPacketSize bits. PYLD[7] is reserved bit if EndPoint 3 is not for Isochronous transfer.
:

| | |
|---|---|
| Application Write: | MA[7:0] = 6EH (01101110B)<br>MD[7:0] = LSB MaxPacketSize<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = EEH (11101110B)<br>PD[7:0] = LSB MaxPacketSize<br>MRDYB = 1 |

### EndPoint 3 Payload MSB Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | RESV | RESV | RESV | RESV | PYLD | PYLD | PYLD | EndPoint 3 Payload MSB Register |
| R | R | R | R | R | R/W | R/W | R/W | Read/Write Map |

This register stores the MSB MaxPacketSize information for Endpoint 3.

PYLD[2:0]: MSB MaxPacketSize bits. These bits are reserved if EndPoint 3 is not for Isochronous transfer.

RESV[7:3]: Reserved bits. This bits are assigned the value 00000.

| | |
|---|---|
| Application Write: | MA[7:0] = 6FH (01101111B)<br>MD[7:0] = MSB MaxPacketSize<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = EFH (11101111B)<br>PD[7:0] = MSB MaxPacketSize<br>MRDYB = 1 |

### Transmit EndPoint 0 Payload Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | Transmit EndPoint 0 Payload Register |
| R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the MaxPacketSize information for Transmit Endpoint 0.

PYLD[6:0]: MaxPacketSize bits. The size of Transmit EndPoint 0 can be up to 64 bytes.

RESV[7]: Reserved bits. This bits is assigned the value 0.

| | |
|---|---|
| Application Write: | MA[7:0] = 72H (01110010B)<br>MD[7:0] = MaxPacketSize<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = F2H (11110010B)<br>PD[7:0] = MaxPacketSize<br>MRDYB = 1 |

### EndPoint 0 Status Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | RESV | RESV | EPST | EPST | STAL | RESV | SURD | EndPoint 0 Status Register |
| R | R | R | R | R | R/W | R | R/W | Read/Write Map |

This register stores the status information of EndPoint 0.

SURD: SetUpRdy bit.
USB Device Controller sets this bit when it received a valid setup packet from the HOST. Application resets this bit after it finishes
processing the 8-byte setup data.

RESV[1]: Reserved bit.
This bit is assigned the value 0.

STAL: STALL bit. System/USB Reset resets this bit to 0.
Application sets this bit to 1 if EndPoint 0 is STALLed, and resets this bit to 0 if the STALL condition of EndPoint 0 is cleared.

EPST[4:3]: SetUp Stage bits.
These two bits reports the current phase of a CONTROL transfer.
EPST[4:3] = 00 = SETUP phase
EPST[4:3] = 01 = DATA phase
EPST[4:3] = 10 = STATUS phase

RESV[7:5]: Reserved bits. These bits are assigned the value 000.

| **Application Write:** | | | | |
|---|---|---|---|---|
| **MA[7:0]** | **MD[7:0]** | **MWR_RDB** | **MRDYB** | **BIT** |
| 73H (01110011B) | 01H (00000001B) | 1 | 0 | SURD (Reset) |
| 73H (01110011B) | 04H (00000100B) | 1 | 0 | STAL (Set) |
| 73H (01110011B) | 00H (00000000B) | 1 | 0 | STAL (Reset) |

| | |
|---|---|
| Application Read: | MA[7:0] = F3H (11110011B)<br>PD[7:0] = Status condition<br>MRDYB = 1 |

### EndPoint 5 Configuration Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| INTR | FIFO | FIFO | FIFO | XFER | XFER | STAL | CGF | EndPoint 5 Configuration Register |
| R/W | R | R | R | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the configuration information for Endpoint 5.

CFG: Configuration Bit of EndPoint 5. System/USB Reset resets this bit to 0. Application sets this bit to 1 before the device can begin communication with the HOST through EndPoint 5.

STAL: STALL bit. System/USB Reset resets this bit to 0.
Application sets this bit to 1 if EndPoint 5 is STALLed.
Application resets this bit to 0 if the STALL condition of EndPoint 5 is cleared.

XFER[3:2]: Transfer type bits. Application decides the type.
XFER[3:2] = 00 = Control transfer
XFER[3:2] = 10 = Bulk transfer
XFER[3:2] = 11 = Interrupt transfer

FIFO[6:4]: FIFO Number bits.
FIFO[6:4] = 101
EndPoint 5 is assigned to FIFO #5

INTR: Interrupt Rate Feedback bit. System/USB Reset resets this bit to 0. Application sets this bit to 1 if Interrupt Rate Feedback information is needed.

| | |
|---|---|
| Application Write: | MA[7:0] = 74H (01110100B)<br>MD[7:0] = EndPoint 5 information<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = F4H (11110100B)<br>PD[7:0] = EndPoint 5 information<br>MRDYB = 1 |

### EndPoint 5 Toggling Bit Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | RESV | RESV | RESV | RESV | RESV | RESV | TOGL | EndPoint 5 Toggling Register |
| R | R | R | R | R | R | R | R/W | Read/Write Map |

This register stores the toggling bit information for Endpoint 5.

TOGL: Toggling Bit of EndPoint 5.

RESV[7:1]: Reserved bits. These bits are assigned the value 0000000.

| | |
|---|---|
| Application Write to reset: | MA[7:0] = 75H (01110101B)<br>MD[7:0] = 01H (00000001B)<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = F5H (11110101B)<br>PD[7:0] = Toggling bit<br>MRDYB = 1 |

### EndPoint 5 Payload Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | EndPoint 5 Payload Register |
| R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the MaxPacketSize information for Endpoint 5.

PYLD[6:0]: MaxPacketSize bits. The size of EndPoint 5 can be up to 64 bytes.

RESV[7]: Reserved bits. This bits is assigned the value 0.

| | |
|---|---|
| Application Write: | MA[7:0] = 76H (01110110B)<br>MD[7:0] = MaxPacketSize<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = F6H (11110110B)<br>PD[7:0] = MaxPacketSize<br>MRDYB = 1 |

### EndPoint 6 Configuration Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| INTR | FIFO | FIFO | FIFO | XFER | XFER | STAL | CFG | EndPoint 6 Configuration Register |
| R/W | R | R | R | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the configuration information for Endpoint 6.

CFG: Configuration Bit of EndPoint 6. “System/USB Reset” resets this bit to 0. Application sets this bit to 1 before the device can begin communication with the HOST through EndPoint 6.

STAL: STALL bit. “System/USB Reset” resets this bit to 0. Application sets this bit to 1 if EndPoint 6 is STALLed. Application resets this bit to 0 if the STALL condition of EndPoint 6 is cleared.

XFER[3:2]: Transfer type bits. Application decides the type.
00 = Control transfer
10 = Bulk transfer
11 = Interrupt transfer

FIFO[6:4]: FIFO Number bits.
FIFO[6:4] = 110
EndPoint 6 is assigned to FIFO #6

INTR: Interrupt Rate Feedback bit. “System/USB Reset” resets this bit to 0. Application sets this bit to 1 if Interrupt Rate Feedback information is needed.

| | |
|---|---|
| Application Write: | MA[7:0] = 78H (01111000B)<br>MD[7:0] = EndPoint 6 information<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = F8H (11111000B)<br>PD[7:0] = EndPoint 6 information<br>MRDYB = 1 |

### EndPoint 6 Toggling Bit Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | RESV | RESV | RESV | RESV | RESV | RESV | TOGL | EndPoint 6 Toggling Bit Register |
| R | R | R | R | R | R | R | R/W | Read/Write Map |

This register stores the toggling bit information for Endpoint 6.

TOGL: Toggling Bit of EndPoint 6.

RESV[7:1]: Reserved bits. These bits are assigned the value 0000000.

| | |
|---|---|
| Application Write to reset: | MA[7:0] = 79H (01111001B)<br>MD[7:0] = 01H (00000001B)<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = F9H (11111001B)<br>PD[7:0] = Toggling bit<br>MRDYB = 1 |

### EndPoint 6 Payload Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | EndPoint 6 Payload Register |
| R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the MaxPacketSize information for Endpoint 6.

PYLD[6:0]: MaxPacketSize bits. The size of EndPoint 6 can be up to 64 bytes.

RESV[7]: Reserved bits. This bits is assigned the value 0.

| | |
|---|---|
| Application Write: | MA[7:0] = 7AH (01111010B)<br>MD[7:0] = MaxPacketSize<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = FAH (11111010B)<br>PD[7:0] = MaxPacketSize<br>MRDYB = 1 |

### EndPoint 7 Configuration Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| INTR | FIFO | FIFO | FIFO | XFER | XFER | STAL | CFG | EndPoint 7 Configuration Register |
| R/W | R | R | R | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the configuration information for Endpoint 7.

CFG: Configuration Bit of EndPoint 7. “System/USB Reset” resets this bit to 0. Application sets this bit to 1 before the device can begin communication with the HOST through EndPoint 7.

STAL: STALL bit. “System/USB Reset” resets this bit to 0.
If EndPoint 7 is for Isochronous transfer, application should set this bit to 0. For other types of transfer, application sets this bit to 1 if EndPoint 7 is STALLed, and resets this bit to 0 if the STALL condition of EndPoint 7 is cleared.

XFER[3:2]: Transfer type bits. Application decides the type.
XFER[3:2] = 00 = Control transfer
XFER[3:2] = 01 = Isochronous transfer
XFER[3:2] = 10 = Bulk transfer

FIFO[6:4]: FIFO Number bits.
FIFO[6:4] = 111
EndPoint 7 is assigned to FIFO #7

INTR: Interrupt Rate Feedback bit. “System/USB Reset” resets this bit to 0. Application sets this bit to 1 if Interrupt Rate Feedback information is needed.

| | |
|---|---|
| Application Write: | MA[7:0] = 7CH (01111100B)<br>MD[7:0] = EndPoint 7 information<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = FCH (11111100B)<br>PD[7:0] = EndPoint 7 information<br>MRDYB = 1 |

### EndPoint 7 Toggling Bit Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | RESV | RESV | RESV | RESV | RESV | RESV | TOGL | EndPoint 7 Toggling Bit Register |
| R | R | R | R | R | R | R | R/W | Read/Write Map |

This register stores the toggling bit information for Endpoint 7.

TOGL: Toggling Bit of EndPoint 7. If EndPoint 7 is for Isochronous transfer, this bit is reserved.

RESV[7:1]: Reserved bits. These bits are assigned the value 0000000.

| | |
|---|---|
| Application Write to reset: | MA[7:0] = 7DH (01111101B)<br>MD[7:0] = 01H (00000001B)<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = FDH (11111101B)<br>PD[7:0] = Toggling bit<br>MRDYB = 1 |

### EndPoint 7 Payload LSB Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | PYLD | EndPoint 7 Payload LSB Register |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Read/Write Map |

This register stores the LSB MaxPacketSize information for Endpoint 7. The size of EndPoint 7 can be up to 1023 bytes.

PYLD[7:0]: LSB MaxPacketSize bits. PYLD[7] is reserved bit if EndPoint 7 is not for Isochronous transfer.

| | |
|---|---|
| Application Write: | MA[7:0] = 7EH (01111110B)<br>MD[7:0] = LSB MaxPacketSize<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = FEH (11111110B)<br>PD[7:0] = LSB MaxPacketSize<br>MRDYB = 1 |

### EndPoint 7 Payload MSB Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RESV | RESV | RESV | RESV | RESV | PYLD | PYLD | PYLD | EndPoint 7 Payload MSB Register |
| R | R | R | R | R | R/W | R/W | R/W | Read/Write Map |

This register stores the MSB MaxPacketSize information for Endpoint 7.

PYLD[2:0]: MSB MaxPacketSize bits. These bits are reserved if EndPoint 7 is not for Isochronous transfer.

RESV[7:3]: Reserved bits. These bits are assigned the value 00000.

| | |
|---|---|
| Application Write: | MA[7:0] = 7FH( 01111111B)<br>MD[7:0] = MSB MaxPacketSize<br>MWR_RDB = 1<br>MRDYB = 0 |
| Application Read: | MA[7:0] = FFH (11111111B)<br>PD[7:0] = MSB MaxPacketSize<br>MRDYB = 1 |

### FIFO Interface

The W712 provides four Receive FIFOs and four Transmit FIFOs as data buffers between the HOST and the applications. Each FIFO is associated with an EndPoint address as listed below.

| EndPoint Address | FIFO Number | Function |
|---|---|---|
| EP0 (receive) | FIFO #0 | Control Transfer |
| EP1 (receive) | FIFO #1 | Bulk Transfer |
| EP2 (receive) | FIFO #2 | Bulk Transfer |
| EP3 (receive) | FIFO #3 | Isochronous or Bulk Transfer |
| EP0 (transmit) | FIFO #4 | Control Transfer |
| EP5 (transmit) | FIFO #5 | Bulk or Interrupt Transfer |
| EP6 (transmit) | FIFO #6 | Bulk or Interrupt Transfer |
| EP7 (transmit) | FIFO #7 | Isochronous, Interrupt, or Bulk Transfer |

Reading and writing the FIFO are achieved via the MWR_RDB, MRDYB, MA[7:0], MD[7:0], and PD[7:0] signals.

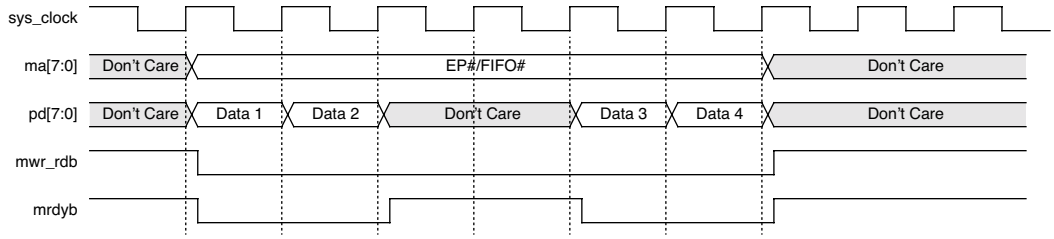



**Figure 6. USB Receive FIFO Read**

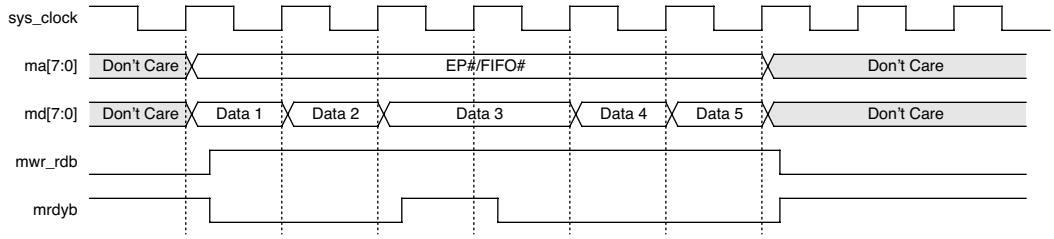



**Figure 7. USB Transmit FIFO Write**

| Application Write to the FIFOs [1] | | | | | | |
|---|---|---|---|---|---|---|
| **FIFO#** | **MA[7]** | **MA[1]** | **MA[0]** | **MWR_RDB** | **MRDYB** | **DATA IN** |
| 7 | 1 | 1 | 1 | 1 | 0 | MD[7:0] |
| 6 | 1 | 1 | 0 | 1 | 0 | MD[7:0] |
| 5 | 1 | 0 | 1 | 1 | 0 | MD[7:0] |
| 4 | 1 | 0 | 0 | 1 | 0 | MD[7:0] |

1. The WRITE pointer is incremented at the rising edge of the sys_clock with stable MA[7:0], MWR_RDB, and MRDYB signals. The WRITE pointer always start at zero.

| Application Read from the FIFOs [1] | | | | | | |
|---|---|---|---|---|---|---|
| **FIFO#** | **MA[7]** | **MA[1]** | **MA[0]** | **MWR_RDB** | **MRDYB** | **DATA OUT** |
| 3 | 0 | 1 | 1 | 0 | 0 | PD[7:0] |
| 2 | 0 | 1 | 0 | 0 | 0 | PD[7:0] |
| 1 | 0 | 0 | 1 | 0 | 0 | PD[7:0] |
| 0 | 0 | 0 | 0 | 0 | 0 | PD[7:0] |

1. The READ pointer is incremented at the rising edge of the sys_clock with stable MA[7:0], MWR_RDB, and MRDYB signals. The READ pointer always start at zero.

The W712 allows application to flush the Transmit FIFOs and reset the corresponding packet ready signal when needed.

| Application to reset the FIFOs and Packet Ready signals | | |
|---|---|---|
| **Address (MA)** | **Data (MD)** | **Function** |
| 4EH (01001110B) | 20H (00100000B) | Flush FIFO #5 and reset PktRdy[5] |
| 4EH (01001110B) | 40H (01000000B) | Flush FIFO #6 and reset PktRdy[6] |
| 4EH (01001110B) | 80H (10000000B) | Flush FIFO #7 and reset PktRdy[7] |

## OPERATIONAL OVERVIEW

### FIFO

The FIFO in the USB device controller serves two purposes by providing:

- A temporary place to store the data packet for each transaction so that both the hardware and the software can packetize the transmitted data within the FIFO.
- Data synchronization between the two frequency domains (the W712 and the application logic).

All the FIFOs are packet-based and they always start at address 0.

Each endpoint address has its own FIFO. For the Control, Interrupt, and Bulk endpoints, the FIFO size is the MaxPacketSize of the corresponding endpoint. For the Isochronous endpoint address, the W712 has to guarantee the availability of the FIFO space for the receiving or transmitting of a data packet at the start of each frame regardless of the existence of the data packet. Hence, each Isochronous endpoint address requires two FIFOs with each FIFO being the size of the defined MaxPacketSize. One FIFO is for the current frame, and the other one stores the data packet from the previous frame.

*Figure 8* is an example of how the receiving FIFOs are used. At the start of a frame (Frame N), FIFO A is ready to accept data from the USB bus while FIFO B stores the data packet from the previous frame (Frame N-1). At the next start of frame (Frame N+1), FIFO B is ready to accept data from the USB bus and FIFO A will contain data from the previous frame (Frame N). Likewise, the transmission FIFOs operate in a similar manner as shown in *Figure 9*.

The double FIFO operation requires the application to finish reading from the FIFO or writing to the FIFO before the end of the current frame.

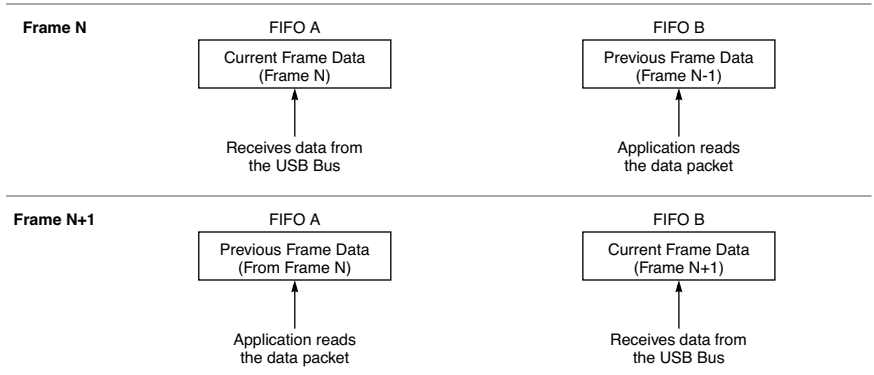



**Figure 8. Receive Double FIFOs**

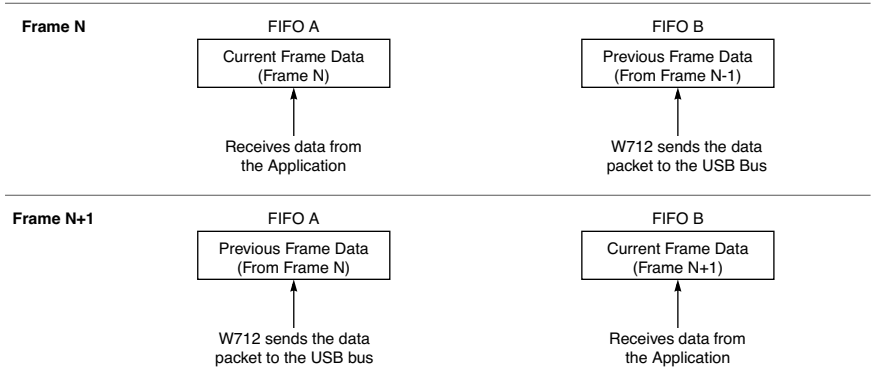



**Figure 9. Transmission Double FIFOs**

### Data Packet Availability and Transfer

The data packet transfer between the HOST and the application is simplified by using the Packet Ready signals.

### Transmitting Data

When the application transmits data to the USB bus, it puts data into an appropriate transmit FIFO and then sets the corresponding Pkt_Rdy bit. The W712 sends the data packet to the USB bus in response to an IN token from the HOST that requests data from an endpoint address associated with this FIFO. If a valid ACK handshake is received from the USB bus, the W712 resets the Pkt_Rdy bit indicating the transmission is completed successfully. If the W712 does not receive an ACK handshake, the W712 re-transmits the same data packet to the USB bus on the next received IN token to that same endpoint address. The application should never write to a FIFO when its Pkt_Rdy bit is set.

There is no data transmission to the USB bus if the Pkt_Rdy bit is not set. In response to an IN token from the HOST, the W712 send a NAK handshake to the USB bus if the Pkt_Rdy bit is not set regardless of whether the FIFO is filled with valid data packet.

If the application would like to transmit a zero length data packet, it simply asserts the Pkt_Rdy bit of the associated FIFO.

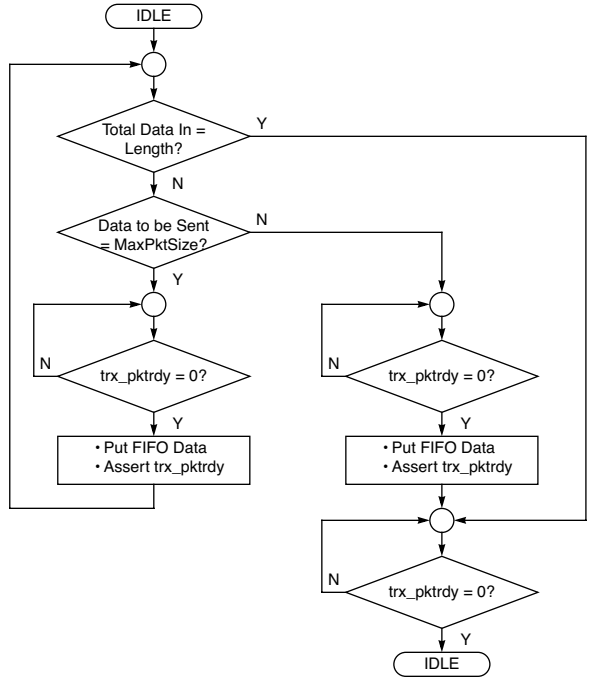



**Figure 10. Interrupt Transfer or Bulk Transfer**

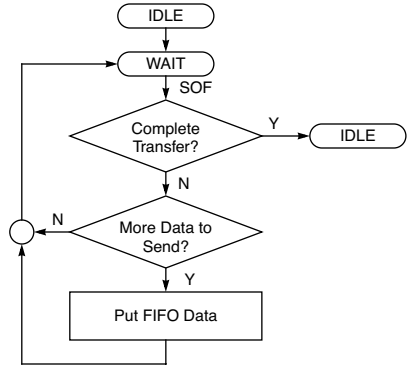



**Figure 11. ISO In Transfer**

### Receiving Data

When the application expects data from the USB bus, it has to observe the Pkt_Rdy bit associated with the receiving FIFO. The Pkt_Rdy bit is asserted by the W712 after it receives a valid data packet. After-

wards, the application reads the byte count, the data packet, and resets the Pkt_Rdy bit when it finishes. The W712 sends a NAK handshake to the USB bus in response to a DATA0/DATA1 data packet if the Pkt_Rdy bit is set.

If the application receives a zero length data packet, it simply resets the Pkt_Rdy bit.

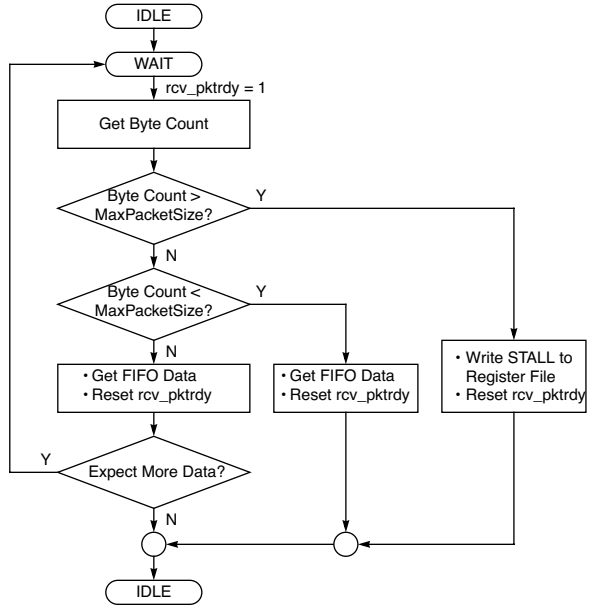



**Figure 12. Bulk Out Transfer**

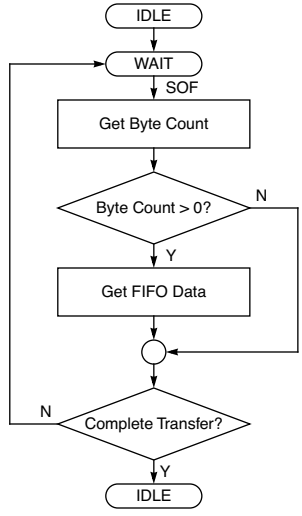
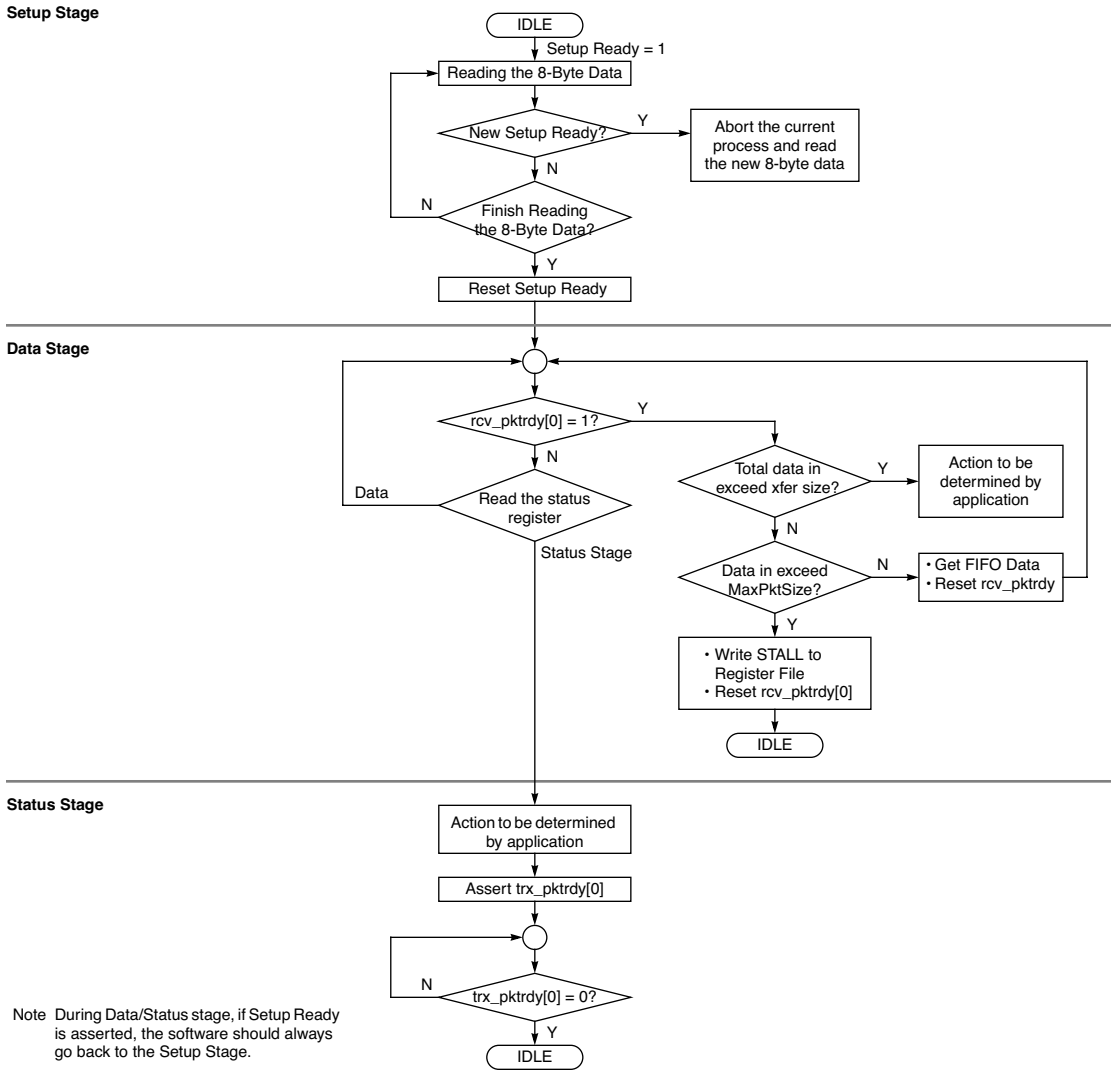



**Figure 13. ISO Out Transfer**

### Control Transfer Handling

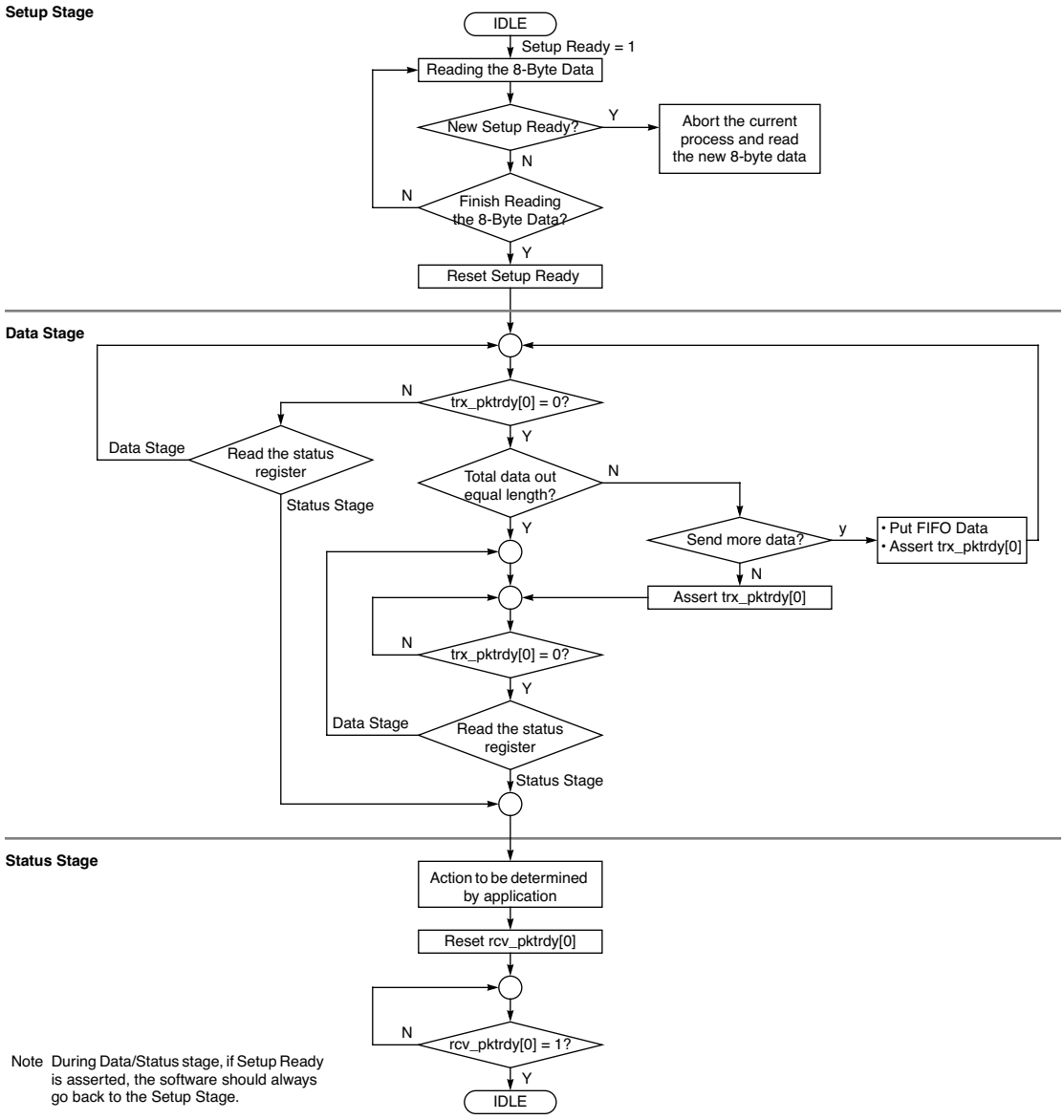
Control transfers have stages. A SETUP stage, an optional DATA stage, and a STATUS stage. During each of these stages, one or more sequences of Token, Data, Handshake occur. A SETUP stage may be received at any time and implies that any previous control transfer has been aborted. The W712 incorporates additional hardware to track the current stage of a control transfer. The application need only monitor the stage during a control transfer.

After receiving a SETUP 8-byte data packet, the W712 asserts the setup_rdy bit. The application should process the 8-byte data packet and then reset the setup_rdy bit. In case a second SETUP stage is received before the application resets the setup_rdy bit, the W712 asserts the setup_rdy2 bit which lasts for only 2 USB bit times while the setup_rdy bit remains asserted. The application should abort the previous SETUP stage and process the new 8-byte data packet.




Note During Data/Status stage, if Setup Ready is asserted, the software should always go back to the Setup Stage.

**Figure 14. Control Write Transfer**




Note During Data/Status stage, if Setup Ready is asserted, the software should always go back to the Setup Stage.

**Figure 15. Control Read Transfer**

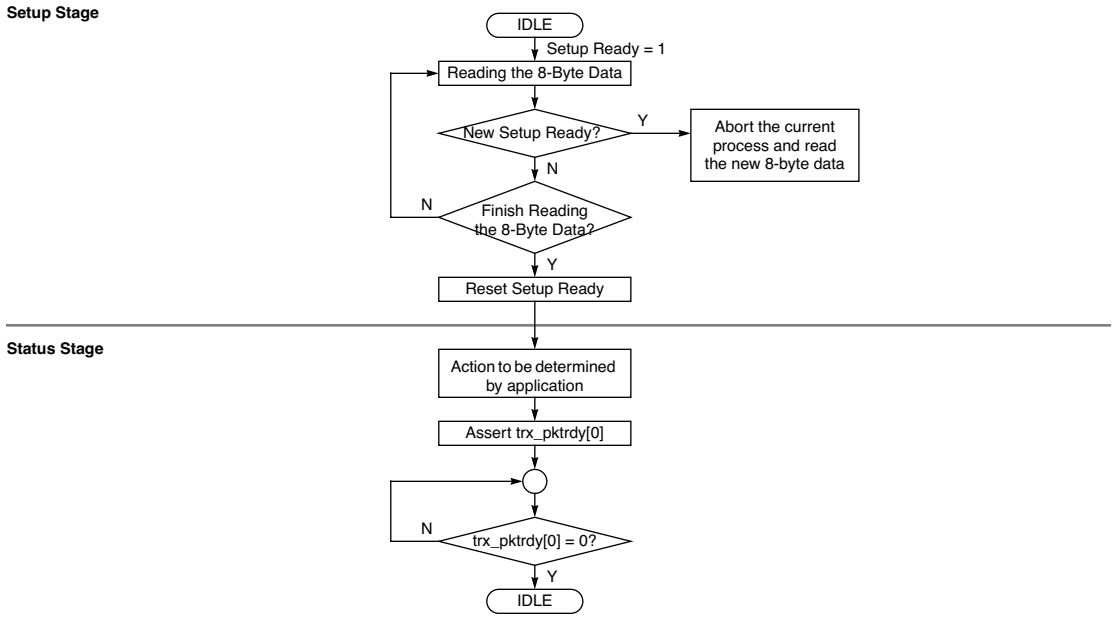



**Figure 16. Control No-Data Transfer**

## GLOSSARY

| Term | Explanation |
|---|---|
| Bandwidth | The amount of data transmitted per unit of time, typically bits per second (bps) or bytes per second (Bps). |
| Bit | A unit of information used by digital computers. Represents the smallest piece of addressable memory within a computer. A bit expresses the choice between two possibilities and is typically represented by a logical one (1) or zero (0). |
| Bit Stuffing | Insertion of a “0” bit into a data stream to cause an electrical transition on the data wires allowing a PLL to remain locked. |
| Bulk Transfer | Nonperiodic, large burst communication typically used for a transfer that can use any available bandwidth and also be delayed until bandwidth is available. |
| Control Transfer | One of four USB Transfer Types. Control transfers support configuration/command/status type communications between client and function. |
| CRC | See Cyclic Redundancy Check. |
| Cyclic Redundancy Check | A check performed on data to see if an error has occurred in transmitting, reading, or writing the data. The result of a CRC is typically stored or transmitted with the checked data. The stored or transmitted result is compared to a CRC calculated for the data to determine if an error has occurred. |
| Device Endpoint | A uniquely identifiable portion of a Universal Serial Bus device that is the source or sink of information in a communication flow between the host and device. |
| Endpoint | See Device Endpoint. |
| Interrupt Transfer | One of four USB Transfer Types. Interrupt transfers characteristics are small data, non-periodic, low frequency, bounded latency, device initiated communication typically used to notify the host of device service needs. |
| Isochronous Transfer | One of four Universal Serial Bus Transfer Types. Isochronous transfers are used when working with isochronous data. Isochronous transfers provide periodic, continuous communication between host and device. |
| Non-Return-to-Zero-Invert | NRZI, a method of encoding serial data in which ones and zeroes are represented by opposite and alternating high and low voltages where there is no return to zero (reference) voltage between encoded bits. Eliminates the need for clock pulses. |
| NRZI | See Non-Return-to-Zero-Invert. |
| PLL | Phase Locked Loop. A circuit that acts as a phase detector to keep an oscillator in phase with an incoming frequency. |
| Protocol | A specific set of rules, procedures, or conventions relating to format and timing of data transmission between two devices. |
| Transaction | The delivery of service to an endpoint. Consists of a token packet, optional data packet, and optional handshake packet. Specific packets are allowed/required based on the transaction type. |
| Transfer | One or more bus transactions to move information between a software client and its function. |
| Transfer Type | Determines the characteristics of the data flow between a software client and its function. Four Transfer types are defined: control, interrupt, bulk, and isochronous. |
| Universal Serial Bus | A collection of USB devices and the software and hardware that allow them to connect the capabilities provided by functions to the host. |
| Universal Serial Bus Interface | The hardware interface between the USB cable and a Universal Serial Bus device. This includes the protocol engine required for all USB devices to be able to receive and send packets. |
| USB | See Universal Serial Bus. |

NOTES

The information contained herein can change without notice owing to product and/or technical improvements.

Please make sure before using the product that the information you are referring to is up-to-date.

The outline of action and examples of application circuits described herein have been chosen as an explanation of the standard action and performance of the product. When you actually plan to use the product, please ensure that the outside conditions are reflected in the actual circuit and assembly designs.

Oki assumes no responsibility or liability whatsoever for any failure or unusual or unexpected operation resulting from misuse, neglect, improper installation, repair, alteration or accident, improper handling, or unusual physical or electrical stress including, but not limited to, exposure to parameters outside the specified maximum ratings or operation outside the specified operating range.

Neither indemnity against nor license of a third party's industrial and intellectual property right,etc.is granted by us in connection with the use of product and/or the information and drawings contained herein. No responsibility is assumed by us for any infringement of a third party's right which may result from the use thereof.

When designing your product, please use our product below the specified maximum ratings and within the specified operating ranges, including but not limited to operating voltage, power dissipation, and operating temperature.

The products listed in this document are intended for use in general electronics equipment for commercial applications (e.g.,office automation, communication equipment, measurement equipment, consumer electronics, etc.).These products are not authorized for use in any system or application that requires special or enhanced quality and reliability characteristics nor in any system or application where the failure of such system or application may result in the loss or damage of property or death or injury to humans. Such applications include, but are not limited to: traffic control, automotive, safety, aerospace, nuclear power control, and medical, including life support and maintenance.

Certain parts in this document may need governmental approval before they can be exported to certain countries. The purchaser assumes the responsibility of determining the legality of export of these parts and will take appropriate and necessary steps, at their own expense, for export to another country.



Oki Semiconductor reserves the right to make changes in specifications at anytime and without notice. This information furnished by Oki Semiconductor in this publication is believed to be accurate and reliable. However, no responsibility is assumed by Oki Semiconductor for its use; nor for any infringements of patents or other rights of third parties resulting from its use. No license is granted under any patents or patent rights of Oki.

# Oki Regional Sales Offices

## Silicon Solutions


**Northwest Area**
785 N. Mary Avenue
Sunnyvale, CA 94085
Tel: 408/720-8940
Fax: 408/720-8965

**Southwest Area**
San Diego, CA
Tel: 760/214-6512
760/214-6414
Fax: 408/737-6568
408/737-6567

**Northeast Area**
Shattuck Office Center
138 River Road
Andover, MA 01810
Tel: 978/688-8687
Fax: 978/688-8896

**South Central Area**
Park Creek II
2007 N. Collins Blvd., Suite 305
Richardson, TX 75080
Tel: 972/238-5450
Fax: 972/238-0268


**Oki Web Site:**
*http://www.okisemi.com/us*


Oki Stock No: 320037-002


# Oki Semiconductor


**Corporate Headquarters**
785 N. Mary Avenue
Sunnyvale, CA 94085-2909
Tel: 408/720-1900
Fax: 408/720-1918