NetChip
Technology, Inc.
335 Pioneer Way
Mt View, California 94041
(650) 526-1490 Fax (650) 526-1494
e-mail: sales@netchip.com
Internet: www.netchip.com

NET2890 USB Interface Controller Preliminary Specification

For Revision 2 IC

Doc #:	**605-2890-0208**
Revision:	**2.0 Draft 8**
Date:	**2/1/99**

Revision History

Revision	Issue Date	Comments
1.0 Draft 1	June 30, 1997	Initial Spec. Based on NET2888 Rev 2 Draft 3
1.0 Draft 2	July 24, 1997	Modified register architecture
1.0 Draft 3	October 4, 1997	More Register Modifications
1.0 Draft 4	October 6, 1997	More Register Modifications (MAINCTL, IRQSTAT1, IRQSTAT3, IRQENB1, IRQENB3, FIFOSTAT, FIFOINTENB
1.0 Draft 5	November 19, 1997	Clarify some register descriptions
2.0 Draft 1	January 17, 1998	Initial Specification for NET2890 Revision 2.
2.0 Draft 2	February 13, 1998	Update some registers
2.0 Draft 3	March 2, 1998	Update some registers; Change ‘mapped’ to ‘paged’ register nomenclature
2.0 Draft 4	March 12, 1998	Added some new register bits. Clarified sections of documentation.
2.0 Draft 5	March 23, 1998	Finalize changes to Revision 2.
2.0 Draft 6	April 3, 1998	Clarify some sections.
2.0 Draft 7	Aug 24, 1998	Clarify **FIFO Valid** and FIFO processing
2.0 Draft 8	Feb 1, 1999	Update AC timing, and DC specifications

NET2890 USB Interface Controller

1. Highlights

1.1 Features

- USB Specification Version 1.0 Compliant
- Bridges between a Processor-Independent local bus and a USB bus
- USB device bandwidth of up to 12Mb/sec
- Four Configurable Endpoints, in addition to Endpoint 0
- Each endpoint can be Isochronous, Bulk, or Interrupt, as well as In or Out
- One 128 byte FIFO attached to each endpoint allows concurrency
- Separate 16 byte FIFO for Control Endpoint 0
- Supports local CPU or DMA data transfers
- Automatic retry of failed packets
- Diagnostic register allows forced USB errors
- Atomic operation to set and clear status bits simplifies software
- Low power CMOS in 48 Pin Plastic QFP Package
- 3.3V operating voltage, 3.3V / 5.0 V dual-voltage I/O interface
- Pin compatible with NET2888

1.2 Overview

The NET2890 USB Interface Controller allows control, isochronous, bulk and interrupt transfers between a generic local bus and a Universal Serial Bus (USB). The NET2890 supports the connection between a host computer and an intelligent peripheral such as a digital camera or scanner.

The five main components of the NET2890 are the USB Bus Interface, the Local Bus Interface, the Configuration Registers, the five endpoint FIFOs, and the Serial Interface Engine.

The USB Interface is responsible for the following functions:

- Host to device Communication.
- USB bit level protocol (Serial Interface Engine).
- Automatic retry of failed packets.
- Up to four Isochronous, Bulk, or Interrupt endpoints, each with a 128 byte FIFO.
- Configurable Control Endpoint 0 with a 16 byte FIFO.
- Interface to FIFOs and Local bus controller.
- Simulated disconnect signaling

The Local Bus Interface is responsible for the following functions:

- FIFO Control.
- Local CPU interface.
- Local DMA controller interface.
- Interrupts.

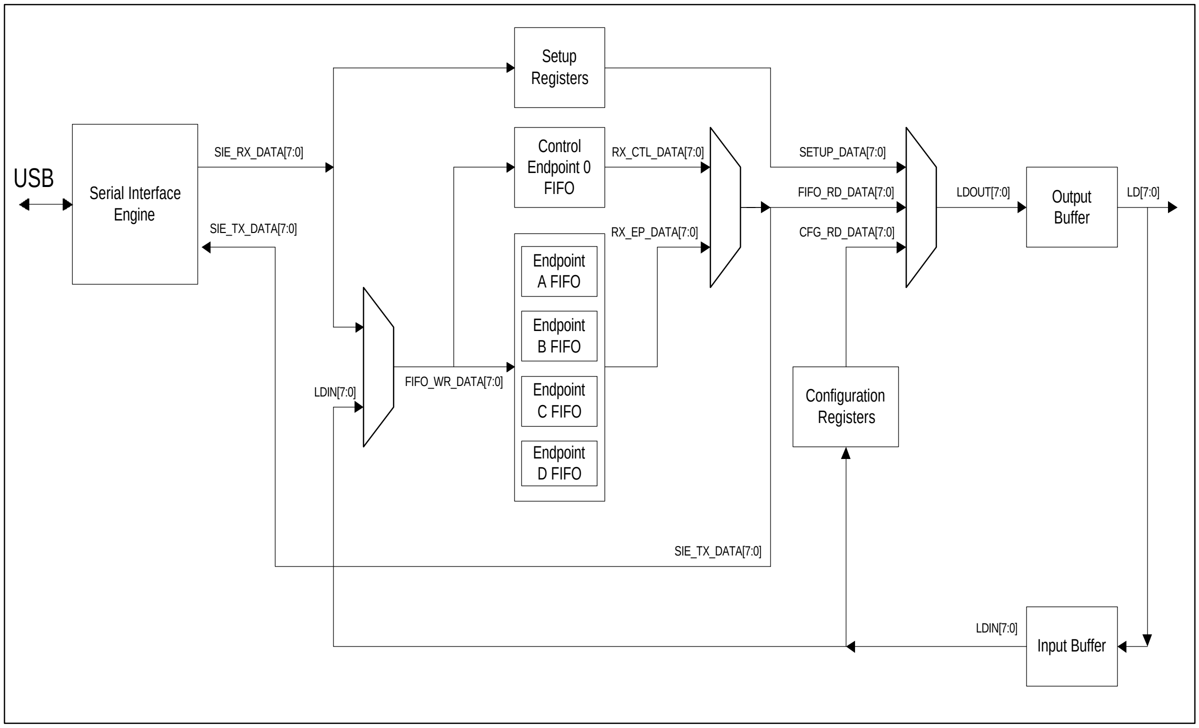
The Configuration Registers provide the following features:

- Directly accessible base registers for common functions.
- Paged registers for each of the 5 endpoints.
- Paged registers for Endpoint 0’s Setup Packet.
- Indexed registers for infrequently accessed functions.
- Registers are accessible only from the local bus.

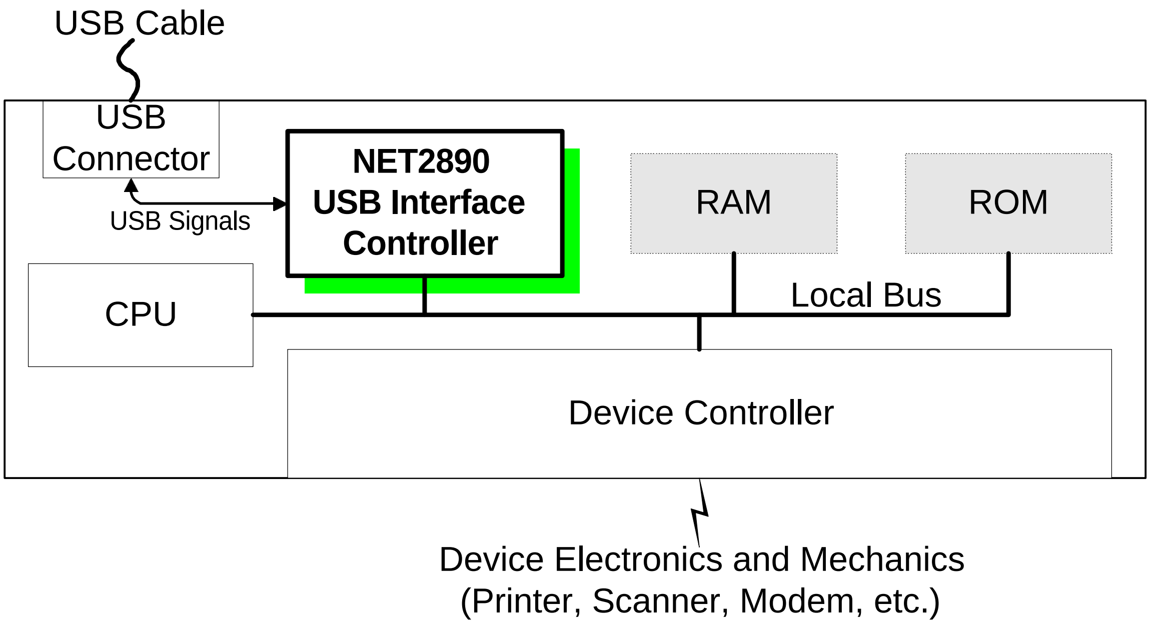
The Serial Interface Engine (SIE) provides the following features:

- Serial data transmitter and receiver.
- Digital phase lock loop and clock recovery.
- CRC generator and checker.
- NRZI encoder/decoder.
- Bit stuffer/unstuffer.
- Packet Identifier (PID) decoder.
- Sync detector.
- Forced error conditions.

1.3 NET2890 Block Diagram



1.4 NET2890 Typical System Block Diagram


USB Cable
USB Connector
USB Signals
NET2890 USB Interface Controller
RAM
ROM
CPU
Local Bus
Device Controller
Device Electronics and Mechanics
(Printer, Scanner, Modem, etc.)

NET2888/NET2890 Differences

The NET2890 was derived from the NET2888. Following is a list of differences between the devices.

1.4.1 Pin Changes

- Pin 2 becomes SOF# output
- ISO# (pin 29) becomes No-Connect

1.4.2 Register Changes

- Completely re-defined register set.

1.4.3 Functional Changes

- The USB protocol, including Endpoint 0, is now managed by the local CPU.
- Endpoints are now configurable.
- Automatic retry of failed packets.

1.5 Changes From Rev 1 to Rev 2

The following changes have been made to the NET2890 from Revision 1 to Revision 2.

1.5.1 Pin Changes

None.

1.5.2 Register Changes

MAPSEL

- Renamed PAGESEL.

MAINCTL

- Bits re-arranged to simplify software:
- The "Bus-Powered" bit has inverted polarity to "Self-Powered". See the description in Section 5, Registers.
- Retry Enable added to bit 4.

DMACTL

- Bits 2:0 select Endpoints 0,A-D instead of FIFO.

IRQSTAT1

- Moved **SOF interrupt** to bit 7 (from old IRQSTAT3 bit 0).
- Removed **Interrupt Status 3** (bit 6).

IRQSTAT2

- FIFO interrupts moved into EPIRQSTAT register.
- New interrupt status bits incorporated.

IDXDATA2

- Removed

MAPDATA1

- Removed.

MAPDATA2

- Removed

PKTLENLSB

- Moved from Index register 5 to Base register 0Bh.

PKTLENMSB

- Moved from Index register 6 to Base register 0Ch.

FIFOBASE

- Register is obsolete.

FIFOSIZE

- Register is obsolete. FIFOs are all fixed sizes.

FIFOCTL

- Moved to Endpoint paged registers address 1Ch.
- “Fifo Enable” bit, “Endpoint Select”bits are obsolete.
- FIFO Valid Bit removed, moved to FIFOSTAT.

FIFOINTENB

- Removed. Interrupt enables moved to EPIRQENB register.

FIFOSTAT

- Moved to Endpoint registers address 1Eh.
- “FIFO Valid” bit added, changed to Yes/SET with Auto-Clear feature.
- FIFO Almost Full and Almost Empty interrupts moved to EPIRQSTAT register.

FIFOCNT

- Moved to Endpoint paged registers address 1Ah.

EPRSPSET

- “Zero-Length Packet Response” bit is obsolete.
- “Endpoint Stall” and “Endpoint Toggle” bits have been swapped. See the description in Section 5, Registers.

EPRSPCLR

- “Zero-Length Packet Response” bit is obsolete.
- “Endpoint Stall” and “Endpoint Toggle” bits have been swapped. See the description in Section 5, Registers.

EPIRQENB

- Bits have been re-arranged to simplify software. See Section 5, Registers.

EPUSBSTAT

- Transmit Register Valid, Receive Register Valid bits are obsolete.
- “Timeout” status bit and Short Packet Transferred added.

IRQENB2

- FIFO interrupt enables moved to EPIRQENB register.

INDEX2

- Removed.

MINDEX1

- Removed.

MINDEX2

- Removed.

DIAG

- Added new index register to control diagnostic features.

EPnPKTSIZE

- Added new maximum packet size registers (Index 10h to 19h).

FIFOETHRS

- Renamed Fn_AETH. The FIFO almost empty threshold registers have been moved from the Endpoint/FIFO registers to the index registers.

FIFOFTHRS

- Renamed Fn_AFTH. The FIFO almost full threshold registers have been moved from the Endpoint/FIFO registers to the index registers.

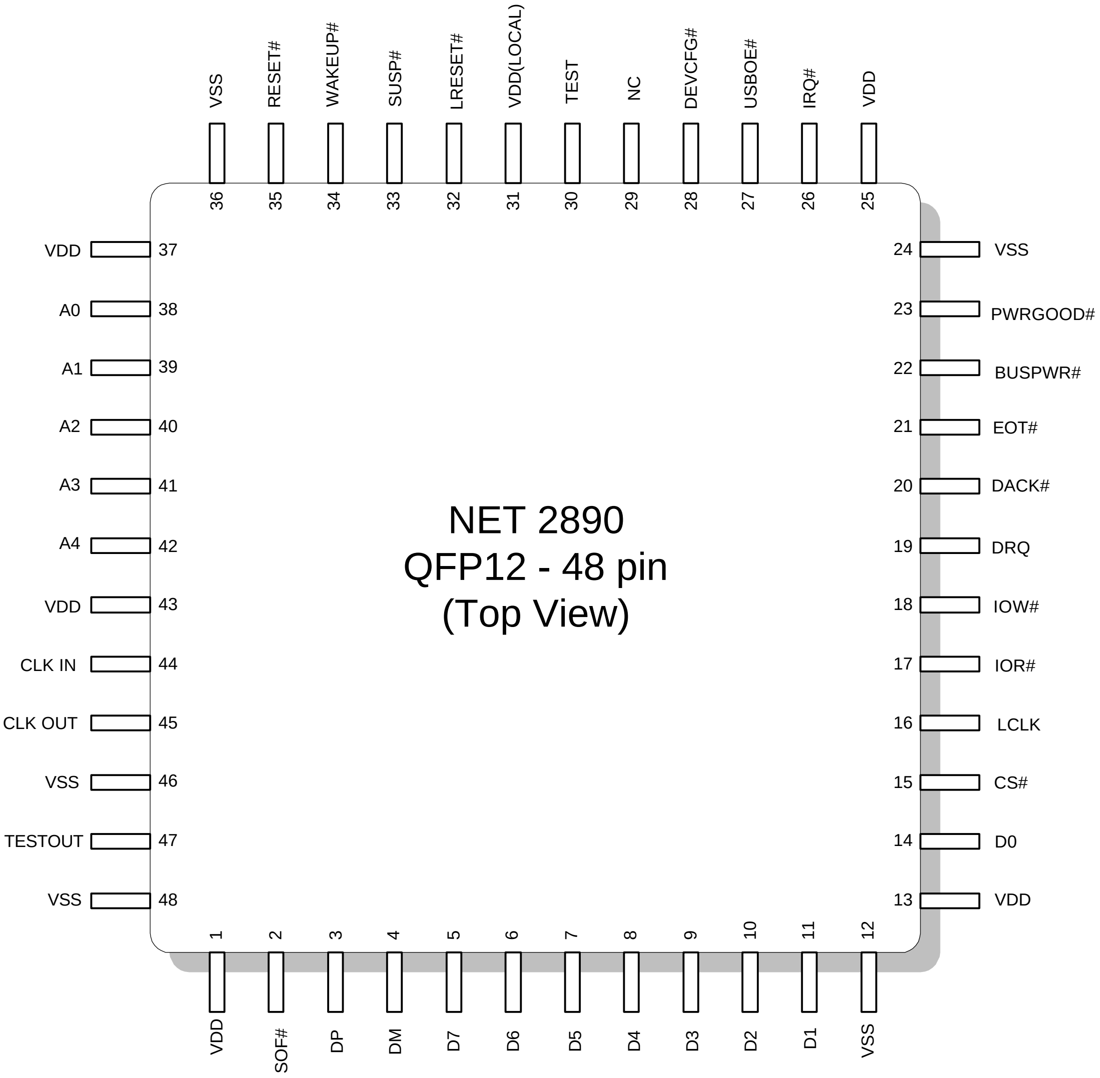
1.5.3 Functional Changes

- Each of the four data endpoints (A-D) has a 128 byte FIFO permanently associated with it.
- FIFOs 5-7 are obsolete: each endpoint has a FIFO attached to it (FIFO A - D), and each endpoint may be configured as an IN or an OUT endpoint.
- There is a separate 16 byte FIFO (FIFO 0) for Endpoint 0 (the default control pipe).
- There is no second bank of SETUP registers.
- “Packet Transmitted” interrupt (in EPIRQSTAT) only occurs when data is transmitted to the host There is no interrupt when the NET2890 responds with a NAK or STALL to the host.
- FIFO registers are grouped into the same page as the corresponding endpoint, instead of separate pages.
- FIFOs A-D increased to 128 bytes. Corresponding threshold and count registers increased to 8 bits.
- Failed packets are automatically retried. If retries are enabled and a packet transfer fails, no interrupt or status information in the NET2890 changes.
- FIFO_VALID_MODE and FIFO_VALID can be used to generate zero-length-packets when required.
- An interrupt (Control Status Phase) can be generated when the host begins a control status phase.
- The new DIAG register can be used to artificially generate USB error conditions.

1.5.4 Documentation Changes

- Instead of referring to “Physical” endpoints 1-4 and “Logical” endpoints 1-4, the local side will access endpoints as Endpoints A-D. Endpoints A-D may each be mapped to any available endpoint address (1-15), in any direction.
- References to “Mapped” registers are all changed to “Paged”.

2. Pin Connection Diagram



3. Pin Description

Pin Type	Description
I	Input
O	Output
I/O	Bi-Directional
S	Schmitt Trigger
TS	Tri-State
TP	Totem-Pole
OD	Open-Drain
PD	50K Pull-Down
PU	50K Pull-Up
#	Active low

NOTE: Input pins that do not have an internal pull-up or pull-down resistor (designated by PU or PD in the ”Type” column below) must be driven externally when the NET2890 is in the suspended state.

Signal Name	Pin	Type	Description
CLKIN	44	I	**48 MHz Oscillator input.** Connect to 48 MHz crystal.
CLKOUT	45	O	**48 MHz Oscillator output.** Connect to 48 MHz crystal. The oscillator stops when the device is suspended by the USB Host.
RESET#	35	I, S, PU	**Reset.** External reset. Connect to local or power-on reset. To reset when oscillator is stopped (initial power-up or in suspend state), assert for at least two ms. When oscillator is running, assert for at least five 48-MHz clock periods.
DP,DM	3,4	I/O	**USB Data Port** DP and DM are the differential data signals of the USB data port NOTE: An external 1.5 KΩ resistor must be connected from DP to 3.3V. This pull-up resistor indicates to the host or upstream hub that a full-speed device is connected to the USB.
D[7:0]	5-11, 14	I/O, 12mA, TS	**Data Bus.** The bi-directional 8-bit data bus is used by devices on the local bus to read or write registers or FIFOs in the NET2890. D7 is the most-significant bit.
A[4:0]	42-38	I	**Address Bus.** The local address bus is used by devices on the local bus to select registers within the NET2890. A4 is the most-significant bit.
CS#	15	I, PU	**Chip Select.** The chip select is used by devices on the local bus to enable access to registers within the NET2890. This signal is ignored if DACK# is asserted.
IOR#	17	I, PU	**I/O Read.** The I/O read strobe is asserted along with CS# and A[4:0] when a device on the local bus reads from an internal register or FIFO. It also allows a FIFO to be read during DMA transfers when DACK# is asserted.

IOW#	18	I, PU	**I/O Write.** The I/O write strobe is asserted along with CS# and A[4:0] when a device on the local bus writes to an internal register or FIFO. It also allows a FIFO to be written during DMA transfers when DACK# is asserted.
DRQ	19	O, 12mA, TP	**DMA Request.** This signal indicates to an external DMA controller that a byte should be transferred to/from the FIFO. During a transfer, DRQ remains asserted until the DACK# input goes active. This output floats when the device is suspended by the USB Host.
DACK#	20	I, PU	**DMA Acknowledge.** This signal from an external DMA controller is used to transfer data to/from the FIFO in response to DRQ. IOR# and IOW# determine the direction of the DMA transfer.
EOT#	21	I, PU	**End of Transfer.** This signal from an external DMA controller is used to terminate a DMA transfer. If it is asserted during a DMA cycle, the current byte will be transferred, but no additional bytes will be requested. EOT# can be programmed to cause a USB interrupt.
IRQ#	26	O, 12mA, OD	**Interrupt Request Output.** The interrupt request output is used to interrupt a processor on the local bus. There are several sources of this interrupt which are described in the Register Description Section.
USBOE#	27	I/O, 12mA, S, TS, PU	**USB Port Output Enable.** When RESET# is not asserted, this is an active low output that is asserted when the NET2890 is driving the USB port data lines. It is intended for debugging purposes only. This signal is not driven while the device is suspended, but will be pulled high by the internal pull-up resistor. When RESET# is asserted, this pin is an input. Driving a LOW level during RESET# holds the NET2890 in a low-power mode by disabling the internal oscillator.
DEVCFG#	28	O, 12mA, TP, PD	**Device Configured.** This active low output is true when the NET2890 has been configured by the USB host. This bit is initialized to inactive (high) during reset and is controlled by the local CPU through the MAINCTL configuration register. This signal is not driven while the device is suspended, but will be pulled low by the internal pull-down resistor. If DEVCFG# is not needed locally, this signal can be used as a general output pin.
SUSP#	33	O, 12mA, TP, PD	**Device Suspended.** This active low output is true when the NET2890 has been suspended by the USB host. This signal is not driven while the device is suspended, but will be pulled low by the internal pull-down resistor.
LCLK	16	O, 12mA, TP, PD	**Local Clock.** This pin is a buffered clock output from either the internal 48 MHz oscillator or the derived USB clock, depending on the state of the "Local Clock Output" bits in the LOCALCTL configuration register. This signal is not driven while the device is suspended, but will be pulled low by the internal pull-down resistor. When the internal oscillator is started, LCLK is prevented from being driven for 2 msec.

LRESET#	32	O, 12mA, TP, PU	**Local Reset.** This active low output is asserted when either the RESET# pin is asserted, or a USB upstream port reset is detected. This signal is not driven while the device is suspended, but will be pulled high by the internal pull-up resistor.
WAKEUP#	34	I, PU	**Wakeup.** This active low input causes the NET2890 to request a USB remote wakeup if device remote wakeup is enabled.
BUSPWR#	22	I, S	**Bus Powered.** This active low input indicates that the logic external to the NET2890 is powered by the USB bus. If this input is high, then the external logic is self-powered. This bit is readable by the local CPU through the MAINCTL configuration register. Alternatively, this bit can be used as a general input signal.
PWRGOOD#	23	I, S	**Power Good.** This active low input indicates that an external power supply used for self-powered mode is operational. This bit is readable by the local CPU through the MAINCTL configuration register. Alternatively, this bit can be used as a general input signal.
SOF#	2	O, 12 mA, TP, PU	**Start of Frame.** This signal pulses low for 8 USB full speed clocks when the Frame Timer is locked and an SOF token or synthesized SOF token is detected.
TEST	30	I, PD	**Test.** For normal operation, connect this pin to ground.
TESTOUT	47	O, 12 mA	**Test Output.** Used for manufacturing test.
NC	29	--	**No connect.**
VDD (USB, Core)	1, 13, 25, 37,43	Power	**Core Supply Voltage.** Connect this pin to the +3.3V supply voltage to supply the core and the USB interface.
VDD (Local)	31	Power	**Local Supply Voltage.** Connect this pin to the +3.3V or +5.0V supply voltage to supply the local interface.
VSS	12, 24, 36, 46, 48	GND	**Device Ground.** Connect this pin to ground.

4. Functional Description

4.1 USB Interface

The NET2890 is a USB function device, and as a result is always a slave to the USB host. The bit and packet level protocols, as well as the electrical interface of the NET2890, conform to USB Specification Version 1.0. All USB data transfers to and from the NET2890 USB port are initiated by the USB host. The NET2890 can be configured for up to 4 endpoints, in addition to Endpoint 0. Each endpoint can be an isochronous, bulk or interrupt type. The configuration registers are used to program the characteristics of each endpoint.

4.2 USB Protocol

The packet protocol of the USB bus consists of tokens, packets, transactions, and transfers.

4.2.1 Tokens

Tokens are a type of Packet Identifier (PID), and follow the sync byte at the beginning of a token packet. The four types of tokens are OUT, IN, SOF, and SETUP.

4.2.2 Packets

There are four types of packets: start-of-frame (SOF), token, data, and handshake. Each packet begins with a sync byte and a Packet Identifier (PID). The other fields vary depending on the type of packet.

An SOF packet consists of the following fields:

- Sync byte (8-bits)
- Packet Identifier (8-bits)
- Frame Number (11-bits)
- CRC (5-bits)

A token packet consists of the following fields:

- Sync byte (8-bits)
- Packet Identifier (8-bits)
- Address (7-bits)
- Endpoint (4-bits)
- CRC (5-bits)

A data packet consists of the following fields: Data packets are always preceded by a token packet.

- Sync byte (8-bits)
- Packet Identifier (8-bits)
- Data (n bytes)
- CRC (16-bits)

A handshake packet consists of the following fields:

- Sync byte (8-bits)
- Packet Identifier (8-bits)

4.2.3 Transaction

A transaction consists of a token packet, optional data packet(s), and a handshake packet.

4.2.4 Transfer

A transfer consists of one or more transactions. Control transfers consist of a setup transaction, optional data transactions, and a handshake (status) transaction.

4.3 Automatic Retries

4.3.1 Out Transactions

If an error occurs during an OUT transaction, the NET2890 reloads its local side FIFO read pointer back to the beginning of the failed packet. The host then sends another OUT token and re-transmits the packet. Once the packet has been successfully received by the NET2890, the Packet Received interrupt is set. The NET2890 can handle any number of back-to-back retries, but the host determines how many times a packet is retried. Windows 98 currently performs three retries before giving up on the device.

4.3.2 In Transactions

If an error occurs during an IN transaction, the NET2890 reloads its USB side FIFO read pointer back to the beginning of the failed packet. The host then sends another IN token and the NET2890 re-transmits the packet. Once the packet has been successfully received by the host, the Packet Transmitted interrupt is set. If additional data is written to the FIFO after the failed packet, and the failed packet was less than the maximum packet size, then the re-transmitted packet may be larger than the failed packet.

4.4 Packet Lengths

The maximum packet length of an endpoint is determined by the corresponding ‘Max Packet Size’ register. For IN transactions, the NET2890 will return a maximum size packet to the host if there are at least ‘max packet’ bytes in the FIFO. A packet of size less than ‘Max Packet Size’ is returned to the host in response to an IN token if either of the following are true

- FIFO Valid Mode (FIFOCTL[6]) and FIFO Valid (FIFOSTAT[7]) are both true
- FIFO Valid Mode (FIFOCTL[6]) is false

4.5 USB Endpoints

The NET2890 supports Control, Isochronous, Bulk, and Interrupt endpoints. All endpoints are unidirectional except for Control endpoints. Bi-directional bulk, isochronous, or interrupt traffic requires two endpoints.

4.5.1 Control Endpoint - Endpoint 0

The control endpoint, Endpoint 0, is a reserved endpoint. The host uses this endpoint to configure and gain information about the device, its configurations, interfaces and other endpoints. Control endpoints are bi-directional, and data delivery is guaranteed.

The host sends 8-byte setup packets to Endpoint 0 to which the device interprets and responds. The NET2890 has a set of registers dedicated to storing the setup packet, as well as a dedicated 16-byte bi-directional FIFO for Control data. For Control writes, data flows through the FIFO from the USB bus to the local bus. For Control reads, data flows through the FIFO from the local bus to the USB bus.

When Endpoint 0 detects a setup packet, the NET2890 sets status bits and interrupts the local CPU. The CPU reads the setup packet from NET2890 registers, and responds based on the contents. Any data returned to the host, including status and descriptors, is provided by the local CPU. Refer to the Chapter 6, Standard Device Requests, for a description of the data which must be returned for each USB request.

4.5.1.1 Control Write Transfer

A successful control write transfer to Control Endpoint 0 consists of the following:

Transaction	Stage	Packet Contents	# of bytes	Source
Setup	Setup Token	SETUP PID, address, endpoint, and CRC5	3	Host
	Data	DATA0 PID, 8 data bytes, and CRC16	11	Host
	Status	ACK	1	NET2890
Data (zero, one or more packets)	OUT Token	OUT PID, address, endpoint, and CRC5	3	Host
	Data (1/0)	DATA PID, N data bytes, and CRC16	N+3	Host
	Status	ACK	1	NET2890
Status	IN Token	IN PID, address, endpoint, and CRC5	3	Host
	Data	DATA1 PID, zero length packet, and CRC16	3	NET2890
	Status	ACK	1	Host

During the Setup transaction, the NET2890 stores the data stage packet in its **Setup Registers**. The NET2890 returns an ACK handshake to the host after all 8 bytes have been received. A **Setup Packet Interrupt** status bit is set to notify the local CPU that a setup packet has been received. The 8-byte data packet is then read and interpreted by the local CPU. A Setup transaction cannot be stalled or NAKed, but if the data is corrupted, then the NET2890 will not return an ACK to the host.

During the Data transaction, zero, one or more data packets are written into the Endpoint 0 FIFO. For each packet:

- Interrupt status bits are set and can interrupt the local CPU
- The local CPU reads the FIFO
- The NET2890 returns an ACK if no error has occurred.

For a successful Status transaction, the NET2890 returns a zero length data packet. A NAK or STALL can be returned if an error occurred.

4.5.2 Control Write Transfer Details

For control write transfers, the host first sends eight bytes of setup information. The setup bytes are stored into an 8-byte register bank that can be accessed from the local CPU. After the eight bytes have been stored into the **Setup Registers**, the **Setup Packet Interrupt** status bit is set. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. The NET2890 will not respond to the host if the **Setup Packet Interrupt** status bit is still set from a previous setup packet, and a new setup packet arrives from the host. This prevents the eight bytes of the previous setup packet from being over-written before the local CPU has read them.

The local CPU then reads the 8-byte setup packet and prepares to respond to the optional Data transactions. The number of bytes to be transferred in the Data transactions is specified in the setup packet. When the setup packet is received, the **Control Status Phase Handshake** bit is automatically set in anticipation of the control status phase. While this bit is set, the control status phase will be acknowledged with a NAK, allowing the local CPU to prepare its handshake response (ACK or STALL). Once the **Control Status Phase Handshake** bit cleared, the ACK or STALL handshake will be returned to the host.

During a control write operation, optional Data transactions can follow the Setup transaction. The **Data Out Token Interrupt** status bit is set at the beginning of each Data transaction. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. The bytes corresponding to the Data transaction are stored into the Endpoint 0 FIFO. If the FIFO fills up and another byte is transferred from the host, the NET2890 will return a NAK handshake to the host, signaling that the data could not be accepted. After each transaction, the local CPU should check the **USB OUT ACK Sent, USB OUT NAK Sent,** and **Timeout** status bits to determine if the packet was successfully received.

- If a packet is not successfully received (NAK or Timeout status) and retries are disabled, the **Data Packet Received Interrupt** status bit will be set. The packet data which is in the FIFO or has already been read by the CPU should be discarded. The host will resend the same packet again.

- If a packet is not successfully received (NAK or Timeout status) and retries are enabled, the **Data Packet Received Interrupt** status bit will not be set, and the data will be automatically flushed from the FIFO. The host will resend the same packet again. This process is transparent to the local CPU.

- If the local CPU has stalled this endpoint by setting the **Endpoint Stall** bit, the NET2890 will not store any data into the FIFO, and will respond with a STALL acknowledge to the host. There will not be a Status transaction in this case.

The local CPU can either start polling for valid data immediately after receiving the setup packet, or can wait for the **Data Packet Received Interrupt** status bit to be set. As the FIFO is filling up from the USB side, the local CPU can poll the FIFO status register to determine when a byte is available. Otherwise it can either poll the **Data Packet Received Interrupt** status bit, or enable it as an interrupt, and then read the entire packet from the FIFO at once. If the host tries to write more data than was indicated in the setup packet, then the local CPU should set the STALL bit for Endpoint 0. In this case there will not be a status stage from the host.

After all of the optional Data transaction packets have been received, the host will send an IN token, signifying the Status transaction. The **Control Status Interrupt** status bit is set after the IN token of the Status transaction has been received. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. Until the **Control Status Phase Handshake** bit is cleared by the local CPU, the NET2890 will respond to the Status transaction with NAKs, indicating that the device is still processing the setup command. When the **Control Status Phase Handshake** bit has been cleared by the local CPU, the NET2890 will respond with a zero length data packet (transfer OK) or STALL (device had an error).

4.5.2.1 Control Read Transfer

A successful control read transfer from Control Endpoint 0 consists of the following:

Transaction	Stage	Packet Contents	# of bytes	Source
Setup	Setup Token	SETUP PID, address, endpoint, and CRC5	3	Host
	Data	DATA0 PID, 8 data bytes, and CRC16	11	Host
	Status	ACK	1	NET2890
Data (zero, one, or more packets)	IN Token	IN PID, address, endpoint, and CRC5	3	Host
	Data (1/0)	DATA PID, N data bytes, and CRC16	N+3	NET2890
	Status	ACK	1	Host
Status	OUT Token	OUT PID, address, endpoint, and CRC5	3	Host
	Data	DATA1 PID, zero length packet, and CRC5	3	Host
	Status	ACK	1	NET2890

The Setup transaction is processed in the same way as for control write transfers.

During the Data transaction, zero, one or more data packets are read from the Endpoint 0 FIFO. For each packet:

- Interrupt status bits are set and can interrupt the local CPU
- The local CPU writes data to the FIFO
- If there is no data in the FIFO, a NAK or zero length packet is returned to the host
- The Host returns an ACK to the NET2890 if no error has occurred.

For a successful Status transaction, the Host sends a zero length data packet, and the NET2890 responds with an ACK. A NAK or STALL can be returned if an error occurred.

4.5.2.2 Control Read Transfer Details

For control read transfers, the host first sends eight bytes of setup information. The setup bytes are stored into an 8-byte register bank that can be accessed from the local CPU. After the eight bytes have been stored into the **Setup Registers**, the **Setup Packet Interrupt** status bit is set. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. The NET2890 will not respond to the host if the **Setup Packet Interrupt** status bit is still set from a previous setup packet, and a new setup packet arrives from the host. This prevents the eight bytes of the previous setup packet from being over-written before the local CPU has read them.

The local CPU then reads the 8-byte setup packet and prepares to respond to the optional Data transactions. The number of bytes to be transferred in the Data transactions is specified in the setup packet. When the setup packet is received, the **Control Status Phase Handshake** bit is automatically set. While this bit is set, the control status phase will be acknowledged with a NAK, allowing the local CPU to prepare its handshake response (ACK or STALL). Once the **Control Status Phase Handshake** bit is cleared, the ACK or STALL handshake will be returned to the host.

During a control read operation, optional Data transactions can follow the Setup transaction. After the Setup transaction, the local CPU can start writing the first byte of packet data into the Endpoint 0 FIFO in anticipation of the Data transaction. The **Data In Token Interrupt** status bit is set at the beginning of each Data transaction. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. If there is data in the Endpoint 0 FIFO, it is returned to the host. If Endpoint 0 has no data to return, it returns either a zero length packet (signaling that there is no more data available) or a NAK handshake (the data is not available yet), depending on the **FIFO Valid** and **FIFO Valid Mode** bits.

FIFO Valid Mode	FIFO Valid Bit	End of Transfer Response Bit	Size of Previous Packet	Amount of Data in FIFO	Action
0	0	0	X	empty	NAK to host
0	0	1	Maximum	empty	Zero length packet to host
0	0	1	< Maximum	empty	NAK to host
0	1	X	X	empty	Zero length packet to host
0	X	X	X	>0	Return data to host
1	0	X	X	< Max Packet Length	NAK to host
1	X	X	X	>= Max Packet Length	Return data to host
1	1	X	X	empty	Zero length packet to host
1	1	X	X	>0	Return data to host

After each packet has been sent to the host, the **Data Packet Transmitted Interrupt** status bit is set. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. If retries are disabled, the local CPU should check the **USB IN ACK Sent, USB IN NAK Sent,** and **Timeout** status bits to determine if the packet was successfully transmitted.

- If a packet is not successfully transmitted (**Timeout** status bit set) and retries are *enabled*, the **Data Packet Transmitted Interrupt** status bit will not be set, and the same packet is sent to the host when another IN token is received. The retry operation is transparent to the local CPU.

- If a packet is not successfully transmitted (**Timeout** status bit set) and retries are *disabled*, the **Data Packet Transmitted Interrupt** status bit will be set. The local CPU needs to flush the FIFO and reload the packet for the next IN token.

- If the host tries to read more data than was requested in the setup packet, the local CPU should set the STALL bit for the endpoint.

After all of the optional Data transaction packets have been transmitted, the host will send an OUT token, followed by a zero length data packet, signifying the Status transaction. The **Control Status Interrupt** status bit is set after the OUT token of the Status transaction has been received. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. Until the **Control Status Phase Handshake** bit is cleared by the local CPU, the NET2890 will respond to the Status transaction with NAKs, indicating that the device is still processing the command specified by the Setup transaction. When the **Control Status Phase Handshake** bit has been cleared by the local CPU, the NET2890 will respond with an ACK (transfer OK) or STALL (Endpoint 0 is stalled).

4.5.3 Isochronous Endpoints

Isochronous endpoints are used for the transfer of time critical data. Isochronous transfers do not support any handshaking or error checking protocol, and are guaranteed a certain amount of bandwidth during each frame. The Serial Interface Engine in the NET2890 ignores CRC and bit stuffing errors during isochronous transfers, but sets the **EPUSBSTAT** handshaking bits the same as for non-isochronous packets so that the local CPU can detect errors. Isochronous endpoints are unidirectional, with the direction defined by the endpoint configuration registers.

The maximum packet size of an Isochronous endpoint can be larger than the 128 byte FIFOs in the NET2890. For an Isochronous OUT endpoint, the local CPU or DMA reads data from the FIFO at the same time that data is being received from the USB. The local CPU or DMA must be able to read data fast enough from the FIFO to prevent an overflow. The FIFO Almost Full Threshold can also be used to prevent overflows by interrupting the CPU when the FIFO is almost full.

For an Isochronous IN endpoint, the local CPU or DMA writes data to the FIFO at the same time that data is being transmitted to the USB. The local CPU or DMA must be able to write data fast enough to the FIFO to prevent an underflow. The FIFO Almost Empty Threshold can also be used to prevent underflows by interrupting the CPU when the FIFO is almost empty.

4.5.3.1 Isochronous Out Transactions

Isochronous Out endpoints are used to transfer data from a USB host to the NET2890 local bus. An Isochronous OUT transaction consists of the following:

Stage	**Packet Contents**	**# of bytes**	**Source**
OUT Token	OUT PID, address, endpoint, and CRC5	3	Host
Data	DATA0 PID, N data bytes, and CRC16	N+3	Host

The USB host initiates an Isochronous OUT transaction by sending an OUT token to an Isochronous OUT endpoint. The **Data OUT Token Interrupt** status bit is set when the OUT token is recognized. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. The bytes corresponding to the Data stage are stored into the endpoint’s FIFO. If the FIFO is full when another byte is transferred from the host, the byte will be discarded and the **USB OUT NAK Sent** status bit will be set, even though a NAK is not actually sent to the host for isochronous packets. No handshake packets are returned to the host, but the **USB OUT ACK Sent**, **USB OUT NAK Sent**, and **Timeout** status bits are still set to indicate the status of the transaction. After every data packet is received, the local CPU should sample these status to determine if the packet was successfully received by the host.

By definition, isochronous endpoints do not utilize handshaking with the host. Since there is no way to return a stall acknowledge from an isochronous endpoint to the host, data which is sent to a stalled isochronous endpoint will be received normally.

The local CPU can either start polling for valid data immediately after receiving the OUT token, or can wait for the **Data Packet Received Interrupt** status bit to be set. If it waits for the interrupt, then the Maximum Packet Size must be less than the FIFO size. As the FIFO is filling up from the USB side, the local CPU can poll the **FIFO Empty** status bit to determine when a byte is available. Otherwise it can either poll the **Data Packet Received Interrupt** status bit, or enable it as an interrupt, and then read the entire packet from the FIFO at once. The FIFO almost full interrupt can also be used by the local CPU to determine when to start reading Isochronous OUT data from the FIFO. The local CPU can either poll the **FIFO Almost Full Interrupt** status bit, or enable it to generate a local interrupt by asserting IRQ#.

4.5.3.2 Isochronous In Transactions

Isochronous In endpoints are used to transfer data from the NET2890 local bus to a USB host. An isochronous IN transaction consists of the following:

Stage	Packet Contents	# of bytes	Source
IN Token	IN PID, address, endpoint, and CRC5	3	Host
Data	DATA0 PID, N data bytes, and CRC16	N+3	NET2890

The USB host initiates an Isochronous IN transaction by sending an IN token to an Isochronous IN endpoint. The **Data IN Token Interrupt** status bit is set when the IN token is recognized. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. If there is data in the endpoint's FIFO, it is returned to the host. If the endpoint has no data to return, a zero length packet is returned to the host. The NET2890 responds to the IN token according to the following table.

FIFO Valid Mode	FIFO Valid Bit	End of Transfer Response Bit	Size of Previous Packet	Amount of Data in FIFO	Action
0	0	0	X	empty	Zero length packet to host; **USB IN NAK Sent** status bit set
0	0	1	Maximum	empty	Zero length packet to host
0	0	1	< Maximum	empty	Zero length packet to host; **USB IN NAK Sent** status bit set
0	1	X	X	empty	Zero length packet to host
0	X	X	X	>0	Return data to host
1	0	X	X	< Max Packet Length	Zero length packet to host; **USB IN NAK Sent** status bit set
1	X	X	X	>= Max Packet Length	Return data to host
1	1	X	X	empty	Zero length packet to host
1	1	X	X	>0	Return data to host

After the packet has been sent to the host, the **Data Packet Transmitted Interrupt** status bit is set. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. No handshake packets are returned to the host, but the **USB IN ACK Sent**, **USB IN NAK Sent**, and **Timeout** status bits are still set to indicate the status of the transaction. After every data packet is transmitted, the local CPU should sample these status bits to determine if the packet was successfully transmitted to the host.

By definition, isochronous endpoints do not utilize handshaking with the host. Since there is no way to return a stall acknowledge from an isochronous endpoint to the host, data which is sent to a stalled isochronous endpoint will be received normally.

4.5.4 Bulk Endpoints

Bulk endpoints are used for guaranteed error-free delivery of large amounts of data between a host and device. Bulk endpoints are unidirectional, with the direction defined by the endpoint configuration registers.

4.5.4.1 Bulk Out Transactions

Bulk Out endpoints are used to transfer data from a USB host to the NET2890 local bus. A bulk write transaction to a Bulk Out endpoint consists of the following:

Stage	Packet Contents	# of bytes	Source
OUT Token	OUT PID, address, endpoint, and CRC5	3	Host
Data (1/0)	DATA PID, N data bytes, and CRC16	N+3	Host
Status	ACK, NAK, or STALL	1	NET2890

The USB host initiates a Bulk OUT transaction by sending an OUT token to a Bulk OUT endpoint. The **Data OUT Token Interrupt** status bit is set when the OUT token is recognized. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. The bytes corresponding to the Data stage are stored into the endpoint's FIFO. If the FIFO is full when another byte is transferred from the host, the byte will be discarded and the **USB OUT NAK Sent** status bit will be set. At the completion of the packet, a NAK handshake will be returned to the host, indicating that the packet could not be accepted.

All USB data passes through the endpoint's FIFO to the local bus. The local CPU can poll the **FIFO Empty** status bit or the **FIFOCNT** register to determine when valid data is available. Also, the endpoint **FIFO Almost Full Threshold Register** can be programmed to generate an interrupt when a selected number of bytes have been received. If the local CPU knows that an incoming packet will fit entirely into the FIFO, it can wait until **Data Packet Received Interrupt** occurs before reading the data from the FIFO.

The local CPU can use the following methods to read the OUT data from the endpoint's FIFO:

- Poll the **FIFO Empty** status bit to determine when a valid data is available.

- Poll the **Data Packet Received Interrupt** status bit, or enable to it generate a local interrupt. The Maximum Packet Size must be less than the FIFO size.

- Poll the **FIFO Almost Full Interrupt** status bit, or enable it to generate a local interrupt.

After each transaction, the local CPU should check the **USB OUT ACK Sent, USB OUT NAK Sent,** and **Timeout** status bits to determine if the packet was successfully received.

- If a packet is not successfully received (**USB OUT NAK Sent** or **Timeout** status bits set) and retries are disabled, the **Data Packet Received Interrupt** status bit will be set. The packet data which is in the FIFO or has already been read by the CPU should be discarded. The host will resend the same packet again.

- If a packet is not successfully received (**USB OUT NAK Sent** or **Timeout** status bits set) and retries are enabled, the **Data Packet Received Interrupt** status bit will not be set, and the data will be automatically flushed from the FIFO. The host will resend the same packet again. This process is transparent to the local CPU.

- If the local CPU has stalled this endpoint by setting the **Endpoint Stall** bit, the NET2890 will not store any data into the FIFO, and will respond with a STALL acknowledge to the host.

4.5.4.2 Bulk In Endpoints

Bulk IN Endpoints are used to transfer data from the NET2890 local bus to a USB host. A bulk read transaction from a Bulk IN Endpoint consists of the following:

Stage	**Packet Contents**	**# of bytes**	**Source**
IN Token	IN PID, address, endpoint, and CRC5	3	Host
Data (1/0)	DATA PID, N data bytes, and CRC16, or NAK or STALL	N+3	NET2890
Status	ACK	1	Host

The USB host initiates a Bulk IN transaction by sending an IN token to a Bulk IN endpoint. The **Data IN Token Interrupt** status bit is set when the IN token is recognized. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. If there is data in the endpoint's FIFO, it is returned to the host. If the endpoint has no data to return, it returns either a zero length packet (signaling that there is no more data available) or a NAK handshake (the data is not available yet), depending on the **FIFO Valid** and **FIFO Valid Mode** bits.

FIFO Valid Mode	**FIFO Valid Bit**	**End of Transfer Response Bit**	**Size of Previous Packet**	**Amount of Data in FIFO**	**Action**
0	0	0	X	empty	NAK to host
0	0	1	Maximum	empty	Zero length packet to host
0	0	1	< Maximum	empty	NAK to host
0	1	X	X	empty	Zero length packet to host
0	X	X	X	>0	Return data to host
1	0	X	X	< Max Packet Length	NAK to host
1	X	X	X	>= Max Packet Length	Return data to host
1	1	X	X	empty	Zero length packet to host
1	1	X	X	>0	Return data to host

After the packet has been sent to the host, the **Data Packet Transmitted Interrupt** status bit is set. If this interrupt is enabled, the local interrupt IRQ# pin is asserted. If retries are disabled, the local CPU should check the **USB IN ACK Sent, USB IN NAK Sent,** and **Timeout** status bits to determine if the packet was successfully transmitted.

- If a packet is not successfully transmitted (**Timeout** status bit set) and retries are *enabled*, the **Data Packet Transmitted Interrupt** status bit will not be set, and the same packet is sent to the host when another IN token is received. The retry operation is transparent to the local CPU.

- If a packet is not successfully transmitted (**Timeout** status bit set) and retries are *disabled*, the **Data Packet Transmitted Interrupt** status bit will be set. The local CPU needs to flush the FIFO and reload the packet for the next IN token.

- If the host tries to read more data than was requested in the setup packet, the local CPU should set the STALL bit for the endpoint.

4.5.5 Interrupt Endpoints

Interrupt endpoints are used for returning small amounts of status information to the host with a bounded service period. The USB Specification Version 1.0 only defines interrupt endpoint data transfers from the device to the host (IN endpoints). In the NET2890, both directions of interrupt data transfers are supported.

4.5.5.1 Interrupt Out Transactions

Interrupt Out endpoints are used to transfer data from a USB host to the NET2890 local bus. An interrupt OUT transaction to an Interrupt OUT endpoint consists of the following:

Stage	**Packet Contents**	**# of bytes**	**Source**
OUT Token	OUT PID, address, endpoint, and CRC5	3	Host
Data (1/0)	DATA PID, N data bytes, and CRC16	N+3	Host
Status	ACK, NAK, or STALL	1	NET2890

The behavior of an Interrupt OUT endpoint is the same as a Bulk OUT endpoint, except for the toggle bit. If the **Interrupt Mode** bit is cleared, the toggle bit of the Interrupt OUT endpoint is initialized to 0 (DATA0 PID), and behaves the same as a Bulk OUT endpoint. If the **Interrupt Mode** bit is set, the toggle bit of the Interrupt OUT endpoint changes after each data packet is received from the host, without regard to the Status stage.

Note: Interrupt write transactions are not supported in the Rev 1.0 USB Specification.

4.5.5.2 Interrupt In Endpoints

An Interrupt IN endpoint is polled at a rate which is specified in the endpoint descriptor. An interrupt transaction from an Interrupt IN endpoint consists of the following:

Stage	**Packet Contents**	**# of bytes**	**Source**
IN Token	IN PID, address, endpoint, and CRC5	3	Host
Data (1/0)	DATA PID, N data bytes, and CRC16	N+3	NET2890
Status	ACK	1	Host

The behavior of an Interrupt IN endpoint is the same as a Bulk IN endpoint, except for the toggle bit. If the **Interrupt Mode** bit is cleared, the toggle bit of the Interrupt IN endpoint is initialized to 0 (DATA0 PID), and behaves the same as a Bulk IN endpoint. An interrupt endpoint may be used to communicate rate feedback information for certain types of isochronous functions. To support this mode, the **Interrupt Mode** bit is set, and the toggle bit of the Interrupt IN endpoint changes after each data packet is sent to the host, without regard to the Status stage.

4.6 FIFOs

The NET2890 contains four 128-byte FIFOs, one associated with each data endpoint A-D. The direction of the FIFO is determined by the **Endpoint Direction** bit (IN, OUT) of the endpoint. In addition, there is a 16 bytes bi-directional FIFO dedicated for Control transfers, which is associated with Endpoint 0. The direction of FIFO 0 is determined by the most significant bit of the first setup byte.

Each of the FIFOs has programmable threshold registers. Interrupts can be generated for the **FIFO Almost Empty Interrupt** or **FIFO Almost Full Interrupt** conditions. If a FIFO becomes full, write cycles are ignored until space becomes available again. Reads from an empty FIFO produce undefined data.

There is space in the 128-byte FIFOs for multiple BULK or INTERRUPT packets. Each endpoint has a set of maximum packet size registers (**EPnPKTSIZLSB, EPnPKTSIZMSB**) which should be programmed with the same value that is returned to the host in the Endpoint Descriptor (bytes 4 and 5).

4.6.1 IN FIFOs

The **FIFO Valid** bit determines the response (data or NAK) to an IN token when the **FIFO Valid Mode** bit is set. The **FIFO Valid** bit is set by either the CPU or automatically at the end of a DMA. When there are at least EPnPKTSIZ bytes in the FIFO, the data is automatically validated, and will be sent in response to the next IN token. The NET2890 will not send more than EPnPKTSIZ bytes per packet. The local CPU can continue loading data for the next packet until the FIFO is full, and the NET2890 will automatically divide the data flow into EPnPKTSIZ packets. This allows USB transfers to overlap with loading of data from the local bus.

The local CPU should only set **FIFO Valid** when it wants to send a short or zero-length packet (indicating end of transfer). If more data is loaded after **FIFO Valid** is set but before the next IN token occurs, that data will be included in response to the IN token. If enough data is added to make EPnPKTSIZ bytes or more, EPnPKTSIZ bytes will be sent in the next packet and **FIFO Valid** will not be cleared. **FIFO Valid** is cleared only when a short or zero-length packet is successfully sent to the host. It is also cleared by a local reset, USB reset, FIFO flush, or if the endpoint type is changed from IN to OUT.

End of transfers are indicated by short (less than EPnPKTSIZ) or zero-length packets. If **FIFO Valid Mode** is true, the NET2890 will send a short or zero length packet only if the local CPU has set the **FIFO Valid** bit.

If a short packet has been loaded into the FIFO and the **FIFO Valid** bit has been set, additional data written to the FIFO before the completion of the USB data packet is included in the current packet. If the additional data results in a packet size of EPnPKTSIZE or greater, then the original short packet will have been converted to a maximum size packet, and the **FIFO Valid** bit will not be cleared until a short or zero length packet is transferred.

During DMA transfers, if the number of bytes transferred is an integer multiple of the maximum packet size, then a zero length packet will always be sent following the DMA data. This happens because the **FIFO Valid** bit is always set at the end of a DMA transfer, and the FIFO will be empty in this case.

If **FIFO Valid Mode** is false, and there are fewer than EPnPKTSIZ bytes in the FIFO, the NET2890 will send whatever is currently in the FIFO in response to the next IN token even if **FIFO Valid** has not been set. Zero length packets can be sent by setting the **FIFO Valid** bit when the FIFO is empty.

4.6.2 OUT FIFOs

When receiving data, the NET2890 will NAK the host (indicating that it cannot accept the data) if either the FIFO runs out of room, or the **Data Packet Received Interrupt** status bit is set. USB OUT transfers can overlap with the local CPU unloading the data using the following sequence:

- When the local CPU gets the **Data Packet Received Interrupt**, it reads the **FIFOCNT** register so it knows how many bytes are in the current packet.
- Then the local CPU clears the **Data Packet Received Interrupt** which allows the next packet to be received.
- Now the local CPU can unload data from the FIFO while the next USB OUT transaction is occurring.

4.7 Interrupt and Status Register Operation

4.7.1 Interrupt Status Register 1 (IRQSTAT1)

Bits 4:0 of this register indicate whether one of the endpoints has an interrupt pending. These bits cannot be written, and can cause a local interrupt if the corresponding interrupt enable bits are set in the **IRQENB1** register. Bit 7 is automatically set when a start-of-frame (SOF) token is received, and is cleared by writing a 1. This bit can cause a local interrupt if the corresponding interrupt enable bit is set in the **IRQENB1** register. Note that the interrupt status bits can be set without the corresponding interrupt enable bit being set. This allows the local CPU to operate in a polled, as well as an interrupt driven environment.

4.7.2 Interrupt Status Register 2 (IRQSTAT2)

Each of the bits of this register is set when a particular event occurs in the NET2890, and are cleared by writing a 1 to the corresponding bit. These bits can cause a local interrupt if the corresponding interrupt enable bits are set in the **IRQENB2** register.

4.7.3 Endpoint Response Registers (EPRSPSET, EPRSTCLR)

Each endpoint has a set of **Endpoint Response Registers**. The bits in these registers determine how the NET2890 will respond to various situations during a USB transaction. Writing a 1 to any of the bits in the **EPRSPSET** register will set the corresponding bits. Writing a 1 to any of the bits in the **EPRSPCLR** register will clear the corresponding bits. Reading either register returns the current state of the bits.

4.7.4 Endpoint Interrupt Status Register (EPIRQSTAT)

Each endpoint has an **Endpoint Interrupt Status Register.** Each of the bits of this register is set when a particular endpoint event occurs, and are cleared by writing a 1 to the corresponding bit. These bits can cause a local interrupt if the corresponding interrupt enable bits are set in the **EPIRQENB** register. Reading the **EPRIQSTAT** register returns the current state of the bits.

4.8 Local Bus

Data passes between the local bus and the USB through the appropriate endpoint's FIFO. The local bus is an 8-bit non-multiplexed bus, and is controlled with a simple set of interface signals. The NET2890 acts only as a slave on the local bus. When the NET2890 is acting as a local bus slave, a CPU on the local bus can access all of the internal configuration registers, as well as the FIFOs.

4.8.1 Maximum Throughput

Note that maximum USB throughput is 1.5 Mbytes/sec, so the local bus utilization for accessing the NET2890 is very low. This allows a fast local CPU to perform many other operations during USB data transfers.

4.8.1.1 CPU writes to FIFO

- T1 (address setup) : 5
- T3 (write strobe width) : 10
- T7 (I/O recovery) : 42
- Total = 57 ns →17.5 Mbytes/sec
- If the address setup can be overlapped with the I/O recovery, then
  Total = 52 ns → 19.2 Mbytes/sec

4.8.1.2 CPU reads from FIFO

- T1 (address setup) : 5
- T4 (read access time) : 29
- Target setup time: (assume 5ns)
- T7 (I/O recovery) : 42
- Total = 81 ns → 12.3 Mbytes/sec
- If the address setup can be overlapped with the I/O recovery, then
  Total = 76 ns → 13.2 Mbytes/sec

4.8.1.3 DMA writes to FIFO

- The maximum rate of DMA write transfers is determined by how quickly DACK# and IOW# are asserted after DRQ. An internal state machine asserts DRQ on the rising edge of a 48MHz clock. With a clock period of 20.8ns, a clock to output delay of 5ns on DRQ, and a 5ns setup time on DACK# and IOW#, both DACK# and IOW# would have to be asserted within 10.8ns of DRQ in order to be sampled on the next rising edge of the clock. Then DACK# and IOW# would both need to be negated 25 nsec later to minimize the length of the cycle. If this could be accomplished, the minimum cycle time would be 125ns yielding a burst rate of 8 Mbytes/sec.

4.8.1.4 DMA reads from FIFO

- The maximum rate of DMA read transfers is the same as the maximum rate of DMA write transfers.

4.8.2 DMA Transfers from NET2890 to Local Bus

A Direct Memory Access (DMA) controller may be used on the local bus to transfer data to and from the NET2890. For host to device transfers, the local and host CPUs first arrange to transfer a block of data from host memory to local shared memory. The local CPU then programs the DMA controller for fly-by demand mode transfers. In this mode, transfers occur only when the NET2890 requests them, and the data is read from one of the NET2890 endpoint FIFOs and written into local memory during the same bus transaction. The DMA address counter is programmed to point to the destination memory block in local shared memory, and the byte count to the number of bytes in the block to be transferred.

After the DMA controller has been programmed, the **DMA Request Enable** bit is set in the NET2890. The USB host performs OUT data transactions over the USB bus to an endpoint's FIFO in the NET2890.

As long as there is data available in the selected endpoint's FIFO, the NET2890 will request local DMA transfers by asserting DRQ. The DMA controller then requests the local bus from the local CPU. After the DMA controller has been granted the bus, it drives a valid memory address and asserts DACK#, IOR#, and MEMW#, thus transferring a byte from an endpoint's FIFO to local memory. The DMA transfers continue until the DMA byte count reaches zero. If the EOT# pin is asserted during the last DMA transfer, the **EOT Interrupt** status bit will be set. If this interrupt is enabled, the local interrupt IRQ# pin is asserted.

4.8.3 DMA Transfers from Local Bus to NET2890

For device to host transfers, the local and host CPUs first arrange to transfer a block of data from local memory to host memory. The local CPU then programs the DMA controller for fly-by demand mode transfers. In this mode, transfers occur only when the NET2890 requests them, and the data is read from local memory and written into the selected endpoint's FIFO during the same bus transaction. The DMA address counter is programmed to point to the source memory block in local memory, and the byte count to the number of bytes in the block to be transferred. After the DMA controller has been programmed, the **DMA Request Enable** bit is set in the NET2890. As long as there is space available in the selected endpoint's FIFO, and the byte count is non-zero, the NET2890 will request DMA transfers by asserting DRQ. The DMA controller then requests the local bus from the local CPU. After the DMA controller has been granted the bus, it drives a valid memory address and asserts DACK#, MEMR#, and IOW#, thus transferring a byte from memory to the endpoint's FIFO.

The USB host sends an IN token to the NET2890 and starts an IN data transaction from the selected endpoint's FIFO. The DMA transfers continue until the DMA byte count reaches zero. If the EOT# pin is asserted during the last DMA transfer, the **EOT Interrupt** status bit will be set. If this interrupt is enabled, the local interrupt IRQ# pin is asserted.

A single DMA setup can be used to transfer multiple packets. The DMA Request signal (DRQ) is asserted anytime there is space available in the FIFO. The endpoint's maximum packet size registers control the maximum number of bytes transmitted to the host in the packet.

4.8.4 Terminating DMA Transfers

The EOT# signal is used to halt a DMA transfer, and is typically provided by an external DMA controller. It should be asserted while DACK# and IOR# or IOW# are simultaneously active to indicate that DMA activity has stopped. Although an EOT# signal indicates that DMA has terminated, the USB transfer is not complete until the last byte has been transferred from the endpoint's FIFO to the USB. The EOT# input resets the NET2890 **DMA Request Enable** bit. When EOT# is detected, the **FIFO VALID** bit is automatically set, causing the remaining data to be sent to the host. If there is no data in the FIFO, then a zero length packet will be returned in response to the next IN token.

If no EOT# signal is provided by the DMA controller, the local CPU can terminate the DMA transfer at any time by resetting the NET2890 **DMA Request Enable** bit. If the NET2890 **DMA Request Enable** bit is cleared during the middle of a DMA cycle, the current cycle will complete before DMA requests are terminated. The FIFO Valid (FIFOSTAT[7]) bit is not automatically set when the DMA Request Enable (DMACTL[3]) is cleared. In this case, the CPU needs to explicitly set this bit if there is a short packet in the FIFO.

If a packet has an odd number of bytes, it is not allowable to pad the packet with one extra byte. If a DMA controller can only transfer an even number of bytes, then the local CPU can be used to write the last odd byte into the FIFO. This approach works as long as the DMA EOT# pin is not used. If the EOT# pin is asserted at the end of a DMA, the FIFO Valid (FIFOSTAT[7]) bit is automatically set. This could be a problem since the last odd byte has not been written to the FIFO by the local CPU. There are no other limitations to intermixing CPU and DMA transfers, as long as DACK# is not asserted during CPU transfers.

4.9 Suspend Mode

When there is a 3 msec period of inactivity on the USB, the USB specification requires a device to enter into a low-power suspended state. The device may not draw more than 500 µA of current while in this state. To facilitate this, the NET2890 provides a **Suspend Request Interrupt Status** bit and a **Suspend Control** bit. Additionally, the NET2890 allows local bus hardware to initiate a “device remote wake-up” to the USB.

4.9.1 The Suspend Sequence

The typical sequence of a suspend operation is as follows:

- During device configuration, the local CPU enables the **Suspend Request Interrupt Status** bit to generate a local interrupt.
- When the USB is idle for three milliseconds, the NET2890 sets the **Suspend Request Interrupt Status** bit, generating an interrupt to the local CPU. This interrupt can also occur if the NET2890 is not connected to a host, and the USB data lines are pulled to the idle state (DP high, DM low).
- The local CPU accepts this interrupt by clearing the **Suspend Request Interrupt Status** bit, and performs the tasks required to ensure that not more than 500 µA of current is drawn from the USB power bus.
- The local CPU writes a 1 to the **Suspend Control Bit** to initiate the suspend.

In suspend mode, the NET2890’s oscillator shuts down, and most output pins are tri-stated to conserve power (see section 3, **Pin Description**). As the NET2890 enters the suspend state, the SUSP# output pin will be driven low for 20 nsec and then floated. It has an internal pull-down resistor to keep it low during suspend. Note that input pins on the NET2890 which do not have an internal pull-up or pull-down resistors should not be allowed to float during suspend mode. The NET2890 will leave suspend mode by detecting a host initiated wake-up or by a device remote wake-up.

If a device is self-powered, it may ignore the USB suspend request and never set the Suspend Control (IRQSTAT2[4]) bit.

4.9.2 Host-Initiated Wake-Up

The host may wake up the NET2890 by driving any non-idle state on the USB. The NET2890 will detect the host’s wake-up request, and re-starts its internal oscillator. Two milliseconds later, the SUSP# output signal is driven high to indicate that the NET2890 has completed its wake-up.

4.9.3 Device-Remote Wake-Up

The device hardware signals a device remote wake-up by driving the WAKEUP# input pin low. If the **Device Remote Wake-up Enable** bit is set, the NET2890 will send a 10-ms wake-up signal to the USB host, and concurrently re-start its local oscillator. Two milliseconds after the WAKEUP# pin is asserted, the SUSP# line is driven high to indicate that the NET2890 has completed its wake-up.

4.9.4 Resume Interrupt

When the NET2890 begins either a Device-Remote Wake-Up or Host-Initiated Wake-Up, it may be programmed to generate a resume interrupt. The **Resume Interrupt Status** bit is set when a resume is detected, and can be enabled to generate an interrupt with the **Resume Interrupt Enable** bit.

4.10 Root Port Reset

If the SIE in the NET2890 detects a single-ended zero on the root port for greater than 2.5 microseconds, it is interpreted as a root port reset. The LRESET# output pin is asserted, and the following resources are reset:

- SIE
- USB state machines
- Local state machines
- **Our Address Register**
- **Device Configured** bit of **Main Control Register**.
- FIFOs and **FIFO Valid** bits

The remainder of the configuration registers are not affected by the root port reset. The **Root Port Reset Interrupt Status** bit is set when a root port reset has been detected, and if enabled, a local interrupt is generated. The local CPU should take appropriate action when this interrupt occurs.

According the USB Specification, the width of the USB reset is minimally 10ms and may be longer depending on the upstream host or hub. There is no specified maximum width for the USB reset. The LRESET# pin is asserted while the USB reset is active, and is negated a few 48 MHz clocks after the USB reset condition is removed by the host.

4.11 NET2890 Power Configuration

The USB specification defines both bus-powered and self-powered devices. A *bus-powered* device is a peripheral which derives all of its power from the upstream USB connector, while a *self-powered* device has an external power supply. The NET2890 is well suited for both types of applications.

The most significant consideration when deciding whether to build a bus-powered or a self-powered device is power consumption. The USB specification lays out the following requirements for maximum current draw:

- A peripheral not configured by the host can draw only 100 mA from the USB power pins.
- A device may not draw more than 500 mA from the USB connector's power pins.
- In suspend mode, the peripheral may not draw more than 500 µA from the USB connector's power pins

If these power considerations can be met without the use of an external power supply, the peripheral can be bus-powered; otherwise a self-powered design should be implemented.

4.11.1 Bus-Powered Device

If the local bus is powered at 3.3 Volts, the V_{DD} and V_{DDL} pins of the device are connected to a 3.3 Volt regulated source derived from the USB 5.0 volt power pin. For a 5.0 volt local bus, the V_{DDL} pin may be connected directly to the USB 5.0 Volt power pin, while the V_{DD} pins must still be connected to 3.3 Volts through a regulator. The rest of the local-side circuitry is also connected to the USB power pin, either through a regulator or directly. Therefore, the peripheral's local circuitry and the NET2890 will all power up simultaneously, and initialization can occur normally with a power-on reset.

4.11.2 Self-Powered Device

Generally, a peripheral with higher power requirements will be self-powered. In a self-powered device, the NET2890 V_{DD} and V_{DDL} pins of the NET2890 are powered by the local power supply. This allows the local bus to continue accessing the NET2890, even when the device is not connected to the USB bus. The USB connector's power pin is left unconnected.

While the peripheral is connected to the USB, the NET2890 will automatically request suspend mode when appropriate, as described in section 4.9.
The NET2890 should not be powered-down when its local bus is still connected to a powered-up device.

There are ESD protection circuits in the NET2890 which will short V_{CC} pins to ground. If the V_{CC} pins are not powered, they will sink too much current from the board.

4.11.3 Low-Power Modes

4.11.3.1 USB Suspend (Unplugged from USB)

The NET2890 may draw a small amount of power when un-plugged from the USB. In power-sensitive applications, the local CPU can force the NET2890 to enter low-power suspend mode when unplugged from the USB by setting the **Suspend Control Bit**. The NET2890 will automatically wake-up when the peripheral is re-connected to the USB. *Do not force suspend mode unless the peripheral is unplugged from the USB. When the NET2890 is connected to the USB, it is a violation of the USB specification to enter the suspend state unless the upstream port has been idle for at least 3 milliseconds.*

This is the preferred method of suspending the NET2890, since a USB plug-in will automatically cause the NET2890 to wake-up and set the **Resume Interrupt Status** bit.

4.11.3.2 Power-On Standby

The peripheral can prevent the NET2890 from starting its oscillator on power-up by driving a LOW into the USBOE# pin from an external open-drain source while RESET# is asserted (LOW). In this state the NET2890 requires only a small quiescent standby current.

When the peripheral wishes to start the oscillator, it releases the USBOE# line and continues to assert RESET# for a minimum of 2 milliseconds. Note that while the oscillator is stopped, the NET2890 cannot respond to USB requests, so the oscillator must be allowed to start when the peripheral detects a USB plug-in event. Note also that USBOE# should only be driven from an external source while RESET# is asserted.

The NET2890 will not detect a USB plug-in event while in this standby state. The device is responsible for detecting the plug-in, and ending the standby condition. This standby technique is appropriate when the device's power budget does not allow the NET2890 to be active long enough to shut it down by setting the **Suspend Control Bit**.

5. Configuration Registers

5.1 Register Description

The NET2890 occupies a 32 byte local register space which can be accessed by a CPU on the local bus. The base registers are accessed directly, while the paged registers are paged in using the PAGESEL register. There are four sets of index registers which allow selected paged and indexed registers to be accessed directly from the Base Register range.

After the NET2890 is powered-up or reset, the registers are set to their default values. Writes to unused registers are ignored, and reads from unused registers return a value of 0.

For compatibility with future revisions, **reserved** bits within a register should always be written with a zero.

5.2 Register Summary

5.2.1 Base registers (unpaged)

Address	Register Name	Register Description	Page
00h	PAGESEL	Paged Register Select	39
01h	MAINCTL	Main Control	39
02h	DMACTL	DMA Control	39
03h	IRQSTAT1	Interrupt Request Status 1	40
04h	IRQSTAT2	Interrupt Request Status 2	40
05h		Reserved	
06h	IDXADDR	Indexed Register Address	41
07h	IDXDATA	Indexed Register Data	41
08h		Reserved	
09h		Reserved	
0Ah		Reserved	
0Bh	PKTLENLSB	Packet Length (LSB)	41
0Ch	PKTLENMSB	Packet Length (MSB)	41
0Dh-0Fh		Reserved	

5.2.2 Endpoint / FIFO registers (paged)

Address	Register Name	Register Description	Page
10h	EPCFG	Endpoint Configuration	42
11h	EPRSPSET	Endpoint Response Set	43
12h	EPRSPCLR	Endpoint Response Clear	44
13h	EPIRQENB	Endpoint Interrupt Enable	45
14h	EPIRQSTAT	Endpoint Interrupt Status	45
15h	EPUSBSTAT	Endpoint USB Status	46
16h	EPDOUT	Endpoint DATA OUT (NET2890 Receive)	46
17h	EPDIN	Endpoint DATA IN (NET2890 Transmit)	46
18h		Reserved	
19h		Reserved	
1Ah	FIFOCNT	FIFO Count	47
1Bh		Reserved	
1Ch	FIFOCTL	FIFO Control	47
1Dh		Reserved	
1Eh	FIFOSTAT	FIFO Status	47
1Fh		Reserved	

5.2.3 Setup Packet registers (paged)

Address	Register Name	Register Description	Page
10h	SETUP0	Setup byte 0	48
11h	SETUP1	Setup byte 1	48
12h	SETUP2	Setup byte 2	48
13h	SETUP3	Setup byte 3	48
14h	SETUP4	Setup byte 4	49
15h	SETUP5	Setup byte 5	49
16h	SETUP6	Setup byte 6	49
17h	SETUP7	Setup byte 7	49
18h-1Fh		Reserved	

5.2.4 Indexed registers

Index	Register Name	Register Description	Page
00h	LOCALCTL	Local Bus Control	50
01h	OURADDR	Our USB Address	50
02h	IRQENB1	Interrupt Request Enable 1	50
03h	IRQENB2	Interrupt Request Enable 2	51
04-06h		Reserved	
07h	FRAMELSB	Frame Count (LSB)	51
08h	FRAMEMSB	Frame Count (MSB)	51
09h		Reserved	
0Ah		Reserved	
0Bh		Reserved	
0Ch	DIAG	Diagnostic Controls	52
0Dh-Fh		Reserved	
10h	EP0PKTSIZLSB	EP0 maximum packet size (LSB)	52
11h			
12h	EPAPKTSIZLSB	EPA maximum packet size (LSB)	52
13h	EPAPKTSIZMSB	EPA maximum packet size (MSB)	52
14h	EPBPKTSIZLSB	EPB maximum packet size (LSB)	52
15h	EPBPKTSIZMSB	EPB maximum packet size (MSB)	53
16h	EPCPKTSIZLSB	EPC maximum packet size (LSB)	53
17h	EPCPKTSIZMSB	EPC maximum packet size (MSB)	53
18h	EPDPKTSIZLSB	EPD maximum packet size (LSB)	53
19h	EPDPKTSIZMSB	EPD maximum packet size (MSB)	53
1Ah-1Fh		Reserved	
20h	F0_AETH	FIFO 0 Almost Empty Threshold	53
21h		Reserved	
22h	F0_AFTH	FIFO 0 Almost Full Threshold	54
23h		Reserved	
24h	FA_AETH	FIFO A Almost Empty Threshold	54
25h		Reserved	
26h	FA_AFTH	FIFO A Almost Full Threshold	54
27h		Reserved	
28h	FB_AETH	FIFO B Almost Empty Threshold	54
29h		Reserved	
2Ah	FB_AFTH	FIFO B Almost Full Threshold	54
2Bh		Reserved	
2Ch	FC_AETH	FIFO C Almost Empty Threshold	54
2Dh		Reserved	
2Eh	FC_AFTH	FIFO C Almost Full Threshold	54
2Fh		Reserved	
30h	FD_AETH	FIFO D Almost Empty Threshold	55
31h		Reserved	
32h	FD_AFTH	FIFO D Almost Full Threshold	55
33h		Reserved	
34h-FEh		Reserved	
FFh	CHIPREV	Chip Revision	55

5.3 Base Registers

5.3.1 (Address 00h; PAGESEL) Paged Register Select

Bits	Description	Read	Write	Default Value
7:5	**Paged Register Set select.** These bits select one of the paged sets of configuration registers. The select value pages one of the sets of registers into the address range 10-1F. All values not listed below are Reserved. **Value** **Paged Register Set** 0h Endpoint 0 (Control) 1h Endpoint A 2h Endpoint B 3h Endpoint C 4h Endpoint D 7h Setup Packet Registers	Yes	Yes	0
4:0	**Reserved**	Yes	No	0

5.3.2 (Address 01h; MAINCTL) Main Control Register

Bits	Description	Read	Write	Default Value
7:5	**Reserved**	Yes	No	0
4	**Retry Enable**. If set, this bit enables the automatic retry feature. If an error occurs during an IN packet, the same packet data is transmitted when the next IN token is received. If an error occurs during an OUT packet, the corresponding FIFO is flushed. No interrupts or status bits will change as a result of failed packets. Note that isochronous endpoints are never retried.	Yes	Yes	1
3	**Device Remote Wake-up Enable**. If set, this bit enables the WAKEUP# pin to cause a device remote wake-up.	Yes	Yes	0
2	**Device Configured**. When the NET2890 has been successfully configured by the host, the local CPU can set this bit which causes the DEVCFG# output pin to be asserted. This bit is cleared when a root port reset is detected.	Yes	Yes	0
1	**Power Good.** When the PWRGOOD# pin is asserted, this bit is read as a one, indicating that the local power supply is operational.	Yes	No	0
0	**Self Powered.** When the BUSPWR# pin is negated (high), this bit is read as a one, indicating that the NET2890 is operating in the self powered mode.	Yes	No	0

5.3.3 (Address 02h; DMACTL) DMA Control Register

Bits	Description	Read	Write	Default Value
7:5	**Reserved**	Yes	No	0
4	**DMA Request.** This status bit reflects the state of the DRQ output pin, and allows a CPU on the local bus to monitor DMA transfers.	Yes	No	0
3	**DMA Request Enable.** Writing a 1 to this bit causes the NET2890 to start requesting DMA cycles from a DMA controller on the local bus. If the EOT# input is asserted, this bit is automatically reset. A CPU on the local bus may also explicitly reset this bit to terminate a DMA transfer. If the CPU sets the **FIFO Valid** bit of the endpoint selected by field 2:0, this bit is cleared. This bit can be read to determine whether a DMA transfer is still in progress.	Yes	Yes	0
2:0	**DMA Endpoint Select.** This field determines which Endpoint is being accessed during a DMA transfer. 000 = Endpoint 0, 001 = A, 010 = B, 011 = C, 100 = D.	Yes	Yes	0

5.3.4 (Address 03h; IRQSTAT1) Interrupt Status Register 1

Bits	Description	Read	Write	Default Value
7	**SOF Interrupt Status.** This bit indicates when a start-of-frame packet has been received by the NET2890. This status bit is cleared by writing a 1.	Yes	Yes/CLR	0
6	**Reserved.**	Yes	No	0
5	**Interrupt Status 2.** This bit indicates when one of the IRQSTAT2 interrupt bits is active and the corresponding interrupt is enabled. Note that this bit is not set as a result of “Suspend Control” being active.	Yes	No	0
4	**Endpoint D Interrupt Status.** This bit conveys the interrupt status for Endpoint D. If set, Endpoint D’s interrupt status register should be read to determine the cause of the interrupt. This bit is set independently of the interrupt enable bit.	Yes	No	0
3	**Endpoint C Interrupt Status.** This bit conveys the interrupt status for Endpoint C. If set, Endpoint C’s interrupt status register should be read to determine the cause of the interrupt. This bit is set independently of the interrupt enable bit.	Yes	No	0
2	**Endpoint B Interrupt Status.** This bit conveys the interrupt status for Endpoint B. If set, Endpoint B’s interrupt status register should be read to determine the cause of the interrupt. This bit is set independently of the interrupt enable bit.	Yes	No	0
1	**Endpoint A Interrupt Status.** This bit conveys the interrupt status for Endpoint A. If set, Endpoint A’s interrupt status register should be read to determine the cause of the interrupt. This bit is set independently of the interrupt enable bit.	Yes	No	0
0	**Endpoint 0 Interrupt Status.** This bit conveys the interrupt status for Endpoint 0. If set, Endpoint 0’s interrupt status register should be read to determine the cause of the interrupt. This bit is set independently of the interrupt enable bit.	Yes	No	0

5.3.5 (Address 04h; IRQSTAT2) Interrupt Status Register 2

NOTE: These status bits are set independently of the corresponding interrupt enable bits. Writing a 0 to these bits has no effect.

Bits	Description	Read	Write	Default Value
7	**Control Status Interrupt.** This bit is set when an IN or OUT token indicating Control Status has been received. This status bit is cleared by writing a 1.	Yes	Yes/CLR	0
6	**Setup Packet Interrupt.** This bit is set when a setup packet has been received from the host. This bit must be cleared (by writing a 1) before the next setup packet can be received. If the bit is not cleared, successive setup packets will not be acknowledged.	Yes	Yes/CLR	0
5	**Input Pin Change Interrupt Status.** If set, this bit indicates that a change occurred in the BUSPWR# or PWRGOOD# input pins. Read the MAINCTL register for the current state of these pins. This status bit is cleared by writing a 1.	Yes	Yes/CLR	0
4	**Suspend Control**. If set, this bit indicates that there is a pending suspend request from the host. Writing a 1 clears this bit and causes the NET2890 to enter the suspended state.	Yes	Yes/CLR	0
3	**EOT Interrupt Status.** This bit indicates when the EOT# input has been asserted simultaneously with DACK# and either IOR# or IOW#, indicating the completion of a DMA transfer. This status bit is cleared by writing a 1. This bit is set independently of the corresponding interrupt enable bit.	Yes	Yes/CLR	0
2	**Root Port Reset Interrupt Status.** This bit indicates when a root port reset has been received by the NET2890. This status bit is cleared by writing a 1.	Yes	Yes/CLR	0
1	**Suspend Request Interrupt Status.** This bit indicates when a suspend-request has been received by the NET2890. This status bit is cleared by writing a 1.	Yes	Yes/CLR	0
0	**Resume Interrupt Status.** If set, this bit indicates that a device resume has occurred. This status bit is cleared by writing a 1.	Yes	Yes/CLR	0

5.3.6 (Address 06h; IDXADDR) Index Register Address

Bits	Description	Read	Write	Default Value
7:0	**Index Register Address.** This register selects which indexed register is accessed when the IDXDATA port is read or written.	Yes	Yes	0

5.3.7 (Address 07h; IDXDATA) Indexed Register Data

Bits	Description	Read	Write	Default Value
7:0	**Indexed Register Data.** This port provides access to the indexed data register selected by IDXADDR.	See Below	See Below	See Below

5.3.8 (Address 0Bh; PKTLENLSB) Packet Length (LSB)

Bits	Description	Read	Write	Default Value
7:0	**Packet Length LSB.** This field provides the least significant bits of the length of the last packet transferred. This field is **not** updated when setup packets are received, because they have a fixed length of 8.	Yes	No	0

5.3.9 (Address 0Ch; PKTLENMSB) Packet Length (MSB)

Bits	Description	Read	Write	Default Value
7:2	**Reserved.**	Yes	No	0
1:0	**Packet Length MSB.** This field provides the most significant bits of the length of the last packet transferred. This field is **not** updated when setup packets are received, because they have a fixed length of 8.	Yes	No	0

5.4 *Endpoint / FIFO Registers*

There are 5 sets of endpoint / FIFO registers, one for each endpoint.

5.4.1 (Address 10h; EPCFG) Endpoint Configuration Register (one per Endpoint)

NOTE: For Endpoint 0, all fields in this register are assigned to fixed values, and are **RESERVED**.

Bits	Description	Read	Write	Default Value
7:4	**Endpoint Number.** This field selects the number of the endpoint. Valid numbers are 0 to 15. This field has no effect on Endpoint 0, which always has an endpoint number of 0.	Yes	Yes	0
3	**Endpoint Direction.** This bit selects the direction of this endpoint, if it is unidirectional. 0 = OUT, 1 = IN. Endpoint 0 is bi-directional, and ignores this bit. The direction is with respect to the USB host point of view. Note that a maximum of one OUT and IN endpoint is allowed for each endpoint number.	Yes	Yes	0
2	**Endpoint Enable.** When set, this bit enables this endpoint. This bit has no effect on Endpoint 0, which is always enabled.	Yes	Yes	0
1:0	**Endpoint Type.** This field selects the type of this endpoint. Endpoint 0 is forced to a Control type. Value Description 0 **Reserved** 1 Isochronous 2 Bulk 3 Interrupt	Yes	Yes	0

5.4.2 (Address 11h; EPRSPSET) Endpoint Response Set Register (one per Endpoint)

NOTE: Writing a 1 to a bit position sets that bit and writing a 0 has no effect.

Bits	Description	Read	Write	Default Value
7	**Reserved.**	Yes	No	0
6	**End of Transfer Response.** This bit is only used for IN endpoints. If FIFO Valid Mode is false, setting this bit causes the endpoint to automatically terminate USB transfers with a zero length packet when necessary. If the last packet sent was full (Short Packet Transferred is false), and an IN token arrives when the FIFO is empty, the endpoint returns a zero length packet (instead of NAK) to indicate end of transfer to the host.	Yes	Yes/Set	1
5	**Interrupt Mode.** This bit is only used for INTERRUPT endpoints. For normal interrupt data, this bit should be set to zero and standard data toggle protocol is followed. When this interrupt endpoint is used for isochronous rate feedback information, this bit should be set high. In this mode the data toggle bit is changed after each packet is sent to the host without regard to handshaking. No packet retries are performed in the rate feedback mode.	Yes	Yes/Set	0
4	**Control Status Phase Handshake.** This bit is only used for endpoint 0. This bit is automatically set when a setup packet is detected. While the bit is set, a control status phase will be acknowledged with a NAK. Once cleared, the proper response will be returned to the host (ACK for Control Reads and zero-length packets for Control Writes).	Yes	Yes/Set	0
3	**Receive Handshake Mode.** This bit is only used for OUT endpoints. This bit selects the response if the previous receive packet has not been processed by the local CPU. If this bit is low, the NET2890 will accept the OUT packet if there is space available in the endpoint's FIFO. If this bit is high, the NET2890 will respond to the host with a NAK if an OUT packet is received and the Data Packet Received Interrupt status bit is still set from the previous packet.	Yes	Yes/Set	1
2	**Reserved.**	Yes	No	0
1	**Endpoint Toggle.** This bit is used to set the endpoint data toggle bit. Reading this bit returns the current state of the endpoint data toggle bit.	Yes	Yes/Set	0
0	**Endpoint Stall.** This bit is used to set the endpoint stall bit. When an Endpoint Set Feature Standard Request to the stall bit is detected by the local CPU, it must write a 1 to this bit. Reading this bit returns the current state of the endpoint stall bit.	Yes	Yes/Set	0

5.4.3 (Address 12h; EPRSPCLR) Endpoint Response Clear Register (one per Endpoint)

Note: Writing a 1 to a bit position clears that bit and writing a 0 has no effect.

Bits	Description	Read	Write	Default Value
7	**Reserved.**	Yes	No	0
6	**End of Transfer Response.** This bit is only used for IN endpoints. If FIFO Valid Mode is false, setting this bit causes the endpoint to automatically terminate USB transfers with a zero length packet when necessary. If the last packet sent was full (Short Packet Transferred is false), and an IN token arrives when the FIFO is empty, the endpoint returns a zero length packet (instead of NAK) to indicate end of transfer to the host.	Yes	Yes/Clr	1
5	**Interrupt Mode.** This bit is only used for INTERRUPT endpoints. For normal interrupt data, this bit should be set to zero and standard data toggle protocol is followed. When this interrupt endpoint is used for isochronous rate feedback information, this bit should be set high. In this mode the data toggle bit is changed after each packet is sent to the host without regard to handshaking. No packet retries are performed in the rate feedback mode.	Yes	Yes/Clr	0
4	**Control Status Phase Handshake.** This bit is only used for endpoint 0. This bit is automatically set when a setup packet is detected. While the bit is set, a control status phase will be acknowledged with a NAK. Once cleared, the proper response will be returned to the host (ACK for Control Reads and zero-length packets for Control Writes).	Yes	Yes/Clr	0
3	**Receive Handshake Mode.** This bit is only used for OUT endpoints. This bit selects the response if the previous receive packet has not been processed by the local CPU. If this bit is low, the NET2890 will accept the OUT packet if there is space available in the endpoint's FIFO. If this bit is high, the NET2890 will respond to the host with a NAK if an OUT packet is received and the Data Packet Received Interrupt status bit is still set from the previous packet.	Yes	Yes/Clr	1
2	**Reserved.**	Yes	No	0
1	**Endpoint Toggle.** This bit is used to clear the endpoint data toggle bit. When an Endpoint Clear Feature Standard Request to clear a stall bit is detected by the local CPU, it must write a 1 to this bit, thus resetting the toggle. Reading this bit returns the current state of the endpoint data toggle bit.	Yes	Yes/Clr	0
0	**Endpoint Stall.** This bit is used to clear the endpoint stall bit. When an Endpoint Clear Feature Standard Request to the stall bit is detected by the local CPU, it must write a 1 to this bit. Reading this bit returns the current state of the endpoint stall bit.	Yes	Yes/Clr	0

5.4.4 (Address 13h; EPIRQENB) Endpoint Interrupt Enable Register (one per Endpoint)

Bits	Description	Read	Write	Default Value
7:6	**Reserved.**	Yes	No	0
5	**Data Packet Received Interrupt Enable.** When set, this bit enables a local interrupt to be set when a data packet has been received from the host.	Yes	Yes	0
4	**Data Packet Transmitted Interrupt Enable.** When set, this bit enables a local interrupt to be set when a data packet has been transmitted to the host.	Yes	Yes	0
3	**Data OUT Token Interrupt Enable.** When set, this bit enables a local interrupt to be set when a Data OUT token has been received from the host.	Yes	Yes	0
2	**Data IN Token Interrupt Enable.** When set, this bit enables a local interrupt to be set when a Data IN token has been received from the host.	Yes	Yes	0
1	**FIFO Almost Full Interrupt Enable.** When set, this bit enables a local interrupt to be set when the number of bytes in the FIFO becomes equal to or greater than the FIFO Almost Full Threshold.	Yes	Yes	0
0	**FIFO Almost Empty Interrupt Enable.** When set, this bit enables a local interrupt to be set when the number of bytes in the FIFO becomes equal to or less than the FIFO Almost Empty Threshold.	Yes	Yes	0

5.4.5 (Address 14h; EPIRQSTAT) Endpoint Interrupt Status Register (one per Endpoint)

Bits	Description	Read	Write	Default Value
7:6	**Reserved.**	Yes	No	0
5	**Data Packet Received Interrupt.** This bit is set when a data packet is received from the host by this endpoint. This status bit is cleared by writing a 1. If this bit is set, the **Receive Handshake Mode** bit is set, and another IN token is received, a NAK is returned to the host.	Yes	Yes/CLR	0
4	**Data Packet Transmitted Interrupt.** This bit is set when a data packet is transmitted from the endpoint to the host. This status bit is cleared by writing a 1.	Yes	Yes/CLR	0
3	**Data OUT Token Interrupt.** This bit is set when a Data OUT token has been received from the host. This status bit is cleared by writing a 1.	Yes	Yes/CLR	0
2	**Data IN Token Interrupt.** This bit is set when a Data IN token has been received from the host. This status bit is cleared by writing a 1.	Yes	Yes/CLR	0
1	**FIFO Almost Full Interrupt.** This bit is set when the number of bytes in the FIFO becomes equal to or greater than the FIFO Almost Full Threshold.	Yes	Yes/CLR	0
0	**FIFO Almost Empty Interrupt.** This bit is set when the number of bytes in the FIFO becomes equal to or less than the FIFO Almost Empty Threshold.	Yes	Yes/CLR	0

5.4.6 (Address 15h; EPUSBSTAT) Endpoint USB Status Register (one per Endpoint)

Bits	Description	Read	Write	Default Value
7	**Timeout.** For an IN endpoint, the last USB packet transmitted was not acknowledged by the Host PC, indicating a bus error. The Host PC will expect the same packet to be retransmitted in response to the next IN token. For an OUT endpoint, the last USB packet received had a CRC or bit-stuffing error, and was not acknowledged by the NET2890. The packet will be retransmitted by the Host PC. Writing a 1 clears this bit.	Yes	Yes/CLR	0
6	**USB STALL Sent.** The last USB packet could not be accepted or provided because the endpoint was stalled, and was acknowledged with a STALL. Writing a 1 clears this bit.	Yes	Yes/CLR	0
5	**USB IN NAK Sent.** The last USB IN packet could not be provided, and was acknowledged with a NAK. Writing a 1 clears this bit.	Yes	Yes/CLR	0
4	**USB IN ACK Rcvd.** The last USB IN data packet transferred was successfully acknowledged with an ACK from the host. Writing a 1 clears this bit.	Yes	Yes/CLR	0
3	**USB OUT NAK Sent.** The last USB OUT data packet could not be accepted, and was acknowledged with a NAK to the host. If retries are disabled, the FIFO data may be corrupted and the local CPU should flush the FIFO. Writing a 1 clears this bit.	Yes	Yes/CLR	0
2	**USB OUT ACK Sent.** The last USB OUT data packet transferred was successfully acknowledged with an ACK to the host. Writing a 1 clears this bit.	Yes	Yes/CLR	0
1	**Short Packet Transferred.** The length of the last packet was less than the Maximum Packet Size (EPnPKTSIZ). Writing a 1 clears this bit.	Yes	Yes/CLR	0
0	**Reserved.**	Yes	No	0

5.4.7 (Address 16h; EPDOUT) Endpoint Data Out Register (one per Endpoint)

Bits	Description	Read	Write	Default Value
7:0	**Endpoint Data Out Register.** For an OUT endpoint, this register is used by the local CPU to read data from the endpoint’s FIFO. This register is unused for an IN endpoint.	Yes	No	0

5.4.8 (Address 17h; EPDIN) Endpoint Data In Register (one per Endpoint)

Bits	Description	Read	Write	Default Value
7:0	**Endpoint Data In Register.** For an IN endpoint, this register is used by the local CPU to write data to the endpoint’s FIFO. This register is unused for an OUT endpoint.	No	Yes	0

5.4.9 (Address 1Ah; FIFOCNT) FIFO Count Register (one per Endpoint)

Bits	Description	Read	Write	Default Value
7:0	**FIFO Count.** This register returns the number of FIFO entries containing valid data. Values range from 0 (empty) to 128 (full) for Endpoints A through D, and from 0 (empty) to 16 (full) for Endpoint 0. **Note:** If retries are disabled, and the endpoint is actively involved in a USB packet transfer, the value read from this register may not be valid.	Yes	No	0

5.4.10 (Address 1Ch;FIFOCTL) FIFO Control Register (one per Endpoint)

Bits	Description	Read	Write	Default Value
7	**Reserved**	Yes	No	0
6	**FIFO Valid Mode.** This bit is only used for IN endpoints. Setting this bit causes the NET2890 to respond to an IN token with a NAK handshake unless there are more than EPnPKTSIZ bytes in the FIFO or the FIFO Valid bit is set. When this bit is cleared, any data waiting in the endpoint's FIFO will be sent in response to a n IN token, and the FIFO Valid bit is ignored (except for zero-length packets).	Yes	Yes	0
5	**FIFO Flush.** Writing a 1 to this bit causes the FIFO to be flushed and the corresponding **FIFO Count** register and **FIFO Valid** bit to be cleared. This bit clears itself, and writing a 0 has no effect. Reading this bit always returns a 0.	No	Yes	0
4:0	**Reserved**	Yes	No	0

5.4.11 (Address 1Eh;FIFOSTAT) FIFO Status Register (one per Endpoint)

Bits	Description	Read	Write	Default Value
7	**FIFO Valid.** This bit is only used for IN endpoints. If the FIFO Valid Mode bit is set and there are fewer than EPnPKTSIZ bytes in the FIFO, the NET2890 will respond to IN tokens with NAK handshake unless this bit is set. Setting this bit causes the data in the FIFO to be returned to the host as a short packet. Setting this bit when the FIFO is empty causes a zero-length packet to be returned. This bit is automatically cleared when a short (fewer than EPnPKTSIZ) packet is transmitted to the Host PC. (See EPnPKTSIZ registers).	Yes	Yes/SET	0
6	**Reserved**	Yes	No	0
5	**FIFO Overflow.** If set, this bit indicates that an attempt was made to write to the FIFO when the FIFO was full. Writing a 1 clears this bit.	Yes	Yes/CLR	0
4	**FIFO Underflow.** If set, this bit indicates that an attempt was made to read the FIFO when the FIFO was empty. Writing a 1 clears this bit.	Yes	Yes/CLR	0
3	**FIFO Full.** If set, this bit indicates that the FIFO is full.	Yes	No	0
2	**FIFO Empty.** If set, this bit indicates that the FIFO is empty.	Yes	No	1
1:0	**Reserved**	Yes	No	0

5.5 Setup Registers

The Setup registers are dedicated to storing an 8-byte SETUP packet received by Endpoint 0.

5.5.1 (Address 10h; SETUP0) Setup Byte 0

Bits	Description	Read	Write	Default Value
7:0	**Setup Byte 0.** This port provides byte 0 of the last setup packet received. For a Standard Device Request, the following bmRequestType information is returned: Bit Description 7 Direction: 0 = host to device; 1 = device to host 6:5 Type: 0 = Standard, 1 = Class, 2 = Vendor, 3 = Reserved 4:0 Recipient: 0 = Device, 1 = Interface, 2 = Endpoint, 3 = Other, 4-31 = Reserved	Yes	No	0

5.5.2 (Address 11h; SETUP1) Setup Byte 1

Bits	Description	Read	Write	Default Value
7:0	**Setup Byte 1.** This port provides byte 1 of the last setup packet received. For a Standard Device Request, the following bRequest Code information is returned: Code Description 00h Get Status 01h Clear Feature 02h Reserved 03h Set Feature 04h Reserved 05h Set Address 06h Get Descriptor 07h Set Descriptor 08h Get Configuration 09h Set Configuration 0Ah Get Interface 0Bh Set Interface 0Ch Synch Frame	Yes	No	0

5.5.3 (Address 12h; SETUP2) Setup Byte 2

Bits	Description	Read	Write	Default Value
7:0	**Setup Byte 2.** This port provides byte 2 of the last setup packet received. For a Standard Device Request, the least significant byte of the wValue field is returned.	Yes	No	0

5.5.4 (Address 13h; SETUP3) Setup Byte 3

Bits	Description	Read	Write	Default Value
7:0	**Setup Byte 3.** This port provides byte 3 of the last setup packet received. For a Standard Device Request, the most significant byte of the wValue field is returned.	Yes	No	0

5.5.5 (Address 14h; SETUP4) Setup Byte 4

Bits	Description	Read	Write	Default Value
7:0	**Setup Byte 4.** This port provides byte 4 of the last setup packet received. For a Standard Device Request, the least significant byte of the wIndex field is returned.	Yes	No	0

5.5.6 (Address 15h; SETUP5) Setup Byte 5

Bits	Description	Read	Write	Default Value
7:0	**Setup Byte 5.** This port provides byte 5 of the last setup packet received. For a Standard Device Request, the most significant byte of the wIndex field is returned.	Yes	No	0

5.5.7 (Address 16h; SETUP6) Setup Byte 6

Bits	Description	Read	Write	Default Value
7:0	**Setup Byte 6.** This port provides byte 6 of the last setup packet received. For a Standard Device Request, the least significant byte of the wLength field is returned.	Yes	No	0

5.5.8 (Address 17h; SETUP7) Setup Byte 7

Bits	Description	Read	Write	Default Value
7:0	**Setup Byte 7.** This port provides byte 7 of the last setup packet received. For a Standard Device Request, the most significant byte of the wLength field is returned.	Yes	No	0

5.6 Indexed Registers

5.6.1 (Index 00h; LOCALCTL) Local Bus Control

Bits	Description	Read	Write	Default Value
7:2	**Reserved.**	Yes	No	0
1:0	**Local Clock Output.** This bit controls the output of the LCLK pin. 0 = 48 MHz clock derived from the internal oscillator 1 = 24 MHz clock derived from the internal oscillator 2 = 12 MHz clock derived from the internal oscillator 3 = stopped	Yes	Yes	0

5.6.2 (Index 01h; OURADDR) Our USB Address Register

Bits	Description	Read	Write	Default Value
7	**Reserved.**	Yes	No	0
6:0	**Our USB Address.** After the local CPU has successfully completed the status phase of a SET Address request from the host, it writes the new device address into this register. This register is cleared when a root port reset is detected	Yes	Yes	0

Note: When a Set Address command is sent from the host, the local CPU must write the new device address into the OURADDR configuration register.

5.6.3 (Index 02h; IRQENB1) Interrupt Enable Register 1

Bits	Description	Read	Write	Default Value
7	**SOF Interrupt Enable.** When set, this bit enables a local interrupt to be generated when a start-of-frame packet is received by the NET2890.	Yes	Yes	0
6:5	**Reserved**	Yes	No	0
4	**Endpoint D Interrupt Enable.** When set, this bit enables a local interrupt to be set when an interrupt is active on this endpoint.	Yes	Yes	0
3	**Endpoint C Interrupt Enable.** When set, this bit enables a local interrupt to be set when an interrupt is active on this endpoint.	Yes	Yes	0
2	**Endpoint B Interrupt Enable.** When set, this bit enables a local interrupt to be set when an interrupt is active on this endpoint.	Yes	Yes	0
1	**Endpoint A Interrupt Enable.** When set, this bit enables a local interrupt to be set when an interrupt is active on this endpoint.	Yes	Yes	0
0	**Endpoint 0 Interrupt Enable.** When set, this bit enables a local interrupt to be set when an interrupt is active on this endpoint.	Yes	Yes	0

5.6.4 (Index 03h; IRQENB2) Interrupt Enable Register 2

Bits	Description	Read	Write	Default Value
7	**Control Status Interrupt Enable.** When set, this bit enables a local interrupt to be generated when an IN or OUT token indicating Control Status has been received.	Yes	Yes	0
6	**Setup Packet Interrupt Enable.** When set, this bit enables a local interrupt to be generated when a setup packet has been received from the host.	Yes	Yes	0
5	**Input Pin Change Interrupt Enable.** When set, this bit enables a local interrupt to be generated when a change has been detected on either the BUSPWR# or PWRGOOD# pins.	Yes	Yes	0
4	**Reserved.**	Yes	No	0
3	**EOT Interrupt Enable.** When set, this bit enables a local interrupt to be generated when an EOT# signal is received from the DMA controller.	Yes	Yes	0
2	**Root Port Reset Interrupt Enable.** When set, this bit enables a local interrupt to be generated when a root port reset is detected.	Yes	Yes	0
1	**Suspend Request Interrupt Enable.** When set, this bit enables a local interrupt to be generated when the USB host is requesting the NET2890 to enter suspend mode.	Yes	Yes	0
0	**Resume Interrupt Enable.** When set, this bit enables a local interrupt to be generated when a device resume has been detected.	Yes	Yes	0

5.6.5 (Index 07h; FRAMELSB) Frame Counter (LSB)

Bits	Description	Read	Write	Default Value
7:0	**Frame Counter LSB.** This field contains the least-significant bits of the frame counter from the most recent start-of-frame packet.	Yes	No	0

5.6.6 (Index 08h; FRAMEMSB) Frame Counter (MSB)

Bits	Description	Read	Write	Default Value
7:4	**Reserved.**	Yes	No	0
3	**Fast Times**. When this bit is set, the frame counter operates at a fast speed for testing purposes only.	Yes	Yes	0
2:0	**Frame Counter MSB.** This field contains the most-significant bits of the frame counter from the most recent start-of-frame packet.	Yes	No	0

5.6.7 (Index 0Ch; DIAG) Diagnostic Register

Bits	Description	Read	Write	Default Value
7:4	**Reserved**	Yes	No	0
3	**Simulated Disconnect.** When this bit is set, the NET2890 will drive the USB data lines to the "disconnected" state. The local CPU should ensure that this bit is set for at least 3 milliseconds to simulate a proper disconnect event.	Yes	Yes	0
2	**Force Receive Error.** When this bit is set, an error is forced on the next received data packet. As a result, the packet will not be acknowledged. This bit is automatically cleared at the end of the next packet.	Yes	Yes/Set	0
1	**Prevent Transmit Bit-Stuff.** When this bit is set, normal bit-stuffing is suppressed during the next transmitted data packet. This will cause a bit-stuffing error when six or more consecutive bits of '1' are in the data stream. This bit is automatically cleared at the end of the next packet.	Yes	Yes/Set	0
0	**Force Transmit CRC Error.** When this bit is set, a CRC error is forced on the next transmitted data packet. The CRC error is generated by inverting the most significant bit of the calculated CRC. This bit is automatically cleared at the end of the next packet.	Yes	Yes/Set	0

5.6.8 (Index 10h; EP0PKTSIZLSB) EP0 Max Packet Size (LSB)

Bits	Description	Read	Write	Default Value
7:0	**EP0 Max Packet Size LSB.** This field provides the least significant bits of the Endpoint 0 Maximum Packet Size.	Yes	Yes	8

5.6.9 (Index 12h; EPAPKTSIZLSB) EPA Max Packet Size (LSB)

Bits	Description	Read	Write	Default Value
7:0	**EPA Max Packet Size LSB.** This field provides the least significant bits of the Endpoint A Maximum Packet Size.	Yes	Yes	8

5.6.10 (Index 13h; EPAPKTSIZMSB) EPA Max Packet Size (MSB)

Bits	Description	Read	Write	Default Value
7:2	**Reserved**	Yes	No	0
1:0	**EPA Max Packet Size MSB.** This field provides the most significant bits of the Endpoint A Maximum Packet Size.	Yes	Yes	0

5.6.11 (Index 14h; EPBPKTSIZLSB) EPB Max Packet Size (LSB)

Bits	Description	Read	Write	Default Value
7:0	**EPB Max Packet Size LSB.** This field provides the least significant bits of the Endpoint B Maximum Packet Size.	Yes	Yes	8

5.6.12 (Index 15h; EPBPKTSIZMSB) EPB Max Packet Size (MSB)

Bits	Description	Read	Write	Default Value
7:2	**Reserved**	Yes	No	0
1:0	**EPB Max Packet Size MSB.** This field provides the most significant bits of the Endpoint B Maximum Packet Size.	Yes	Yes	0

5.6.13 (Index 16h; EPCPKTSIZLSB) EPC Max Packet Size (LSB)

Bits	Description	Read	Write	Default Value
7:0	**EPC Max Packet Size LSB.** This field provides the least significant bits of the Endpoint C Maximum Packet Size.	Yes	Yes	8

5.6.14 (Index 17h; EPCPKTSIZMSB) EPC Max Packet Size (MSB)

Bits	Description	Read	Write	Default Value
7:2	**Reserved**	Yes	No	0
1:0	**EPC Max Packet Size MSB.** This field provides the most significant bits of the Endpoint C Maximum Packet Size.	Yes	Yes	0

5.6.15 (Index 18h; EPDPKTSIZLSB) EPD Max Packet Size (LSB)

Bits	Description	Read	Write	Default Value
7:0	**EPD Max Packet Size LSB.** This field provides the least significant bits of the Endpoint D Maximum Packet Size.	Yes	Yes	8

5.6.16 (Index 19h; EPDPKTSIZMSB) EPD Max Packet Size (MSB)

Bits	Description	Read	Write	Default Value
7:2	**Reserved**	Yes	No	0
1:0	**EPD Max Packet Size MSB.** This field provides the most significant bits of the Endpoint D Maximum Packet Size.	Yes	Yes	0

5.6.17 (Index 20h; F0_AETH) FIFO 0 Almost Empty Threshold Register

Bits	Description	Read	Write	Default Value
7:5	**Reserved**	Yes	No	0
4:0	**FIFO Almost Empty Threshold.** This register determines the threshold at which the FIFO almost empty status bit is set. The threshold is in terms of bytes. For Endpoint 0, this value can range from 0 to 16.	Yes	Yes	0

5.6.18 (Index 22h; F0_AFTH) FIFO 0 Almost Full Threshold Register

Bits	Description	Read	Write	Default Value
7:5	**Reserved**	Yes	No	0
4:0	**FIFO Almost Full Threshold.** This register determines the threshold at which the FIFO almost full status bit is set. The threshold is in terms of bytes. For Endpoint 0, this value can range from 0 to 16.	Yes	Yes	0

5.6.19 (Index 24h; FA_AETH) FIFO A Almost Empty Threshold Register

Bits	Description	Read	Write	Default Value
7:0	**FIFO Almost Empty Threshold.** This register determines the threshold at which the FIFO almost empty status bit is set. The threshold is in terms of bytes	Yes	Yes	0

5.6.20 (Index 26h; FA_AFTH) FIFO A Almost Full Threshold Register

Bits	Description	Read	Write	Default Value
7:0	**FIFO Almost Full Threshold.** This register determines the threshold at which the FIFO almost full status bit is set. The threshold is in terms of bytes.	Yes	Yes	0

5.6.21 (Index 28h; FB_AETH) FIFO B Almost Empty Threshold Register

Bits	Description	Read	Write	Default Value
7:0	**FIFO Almost Empty Threshold.** This register determines the threshold at which the FIFO almost empty status bit is set. The threshold is in terms of bytes	Yes	Yes	0

5.6.22 (Index 2Ah; FB_AFTH) FIFO B Almost Full Threshold Register

Bits	Description	Read	Write	Default Value
7:0	**FIFO Almost Full Threshold.** This register determines the threshold at which the FIFO almost full status bit is set. The threshold is in terms of bytes.	Yes	Yes	0

5.6.23 (Index 2Ch; FC_AETH) FIFO C Almost Empty Threshold Register

Bits	Description	Read	Write	Default Value
7:0	**FIFO Almost Empty Threshold.** This register determines the threshold at which the FIFO almost empty status bit is set. The threshold is in terms of bytes	Yes	Yes	0

5.6.24 (Index 2Eh; FC_AFTH) FIFO C Almost Full Threshold Register

Bits	Description	Read	Write	Default Value
7:0	**FIFO Almost Full Threshold.** This register determines the threshold at which the FIFO almost full status bit is set. The threshold is in terms of bytes.	Yes	Yes	0

5.6.25 (Index 30h; FD_AETH) FIFO D Almost Empty Threshold Register

Bits	Description	Read	Write	Default Value
7:0	**FIFO Almost Empty Threshold.** This register determines the threshold at which the FIFO almost empty status bit is set. The threshold is in terms of bytes	Yes	Yes	0

5.6.26 (Index 32h; FD_AFTH) FIFO D Almost Full Threshold Register

Bits	Description	Read	Write	Default Value
7:0	**FIFO Almost Full Threshold.** This register determines the threshold at which the FIFO almost full status bit is set. The threshold is in terms of bytes.	Yes	Yes	0

5.6.27 (Index FFh; REVISION) Revision Register

Bits	Description	Read	Write	Default Value
7:0	**Chip Revision.** This register returns the current silicon revision number of the NET2890.	Yes	No	Current Revision

6. Standard Device Requests

Standard device requests must be supported by Endpoint 0. See also chapter 9, USB specification. The local bus CPU decodes the setup packets for Endpoint 0 and generates a response based on the following tables.

Table 6-1: Standard Request Codes

bRequest	Value
Get_Status	0
Clear_Feature	1
Reserved	2
Set_Feature	3
Reserved	4
Set_Address	5
Get_Descriptor	6
Set_Descriptor	7
Get_Configuration	8
Set_Configuration	9
Get_Interface	Ah
Set_Interface	Bh
Synch_Frame	Ch

Table 6-2. Descriptor Types

Descriptor Types	Value
Device	1
Configuration	2
String	3
Interface	4
Endpoint	5

6.1 Control 'Read' Transfers

6.1.1 Get Device Status

Offset	Number of Bytes	Description	Suggested Value
0	2	bits 15:2 = Reserved bit 1 = Device Remote Wakeup enabled bit 0 = Device is operating in Self-Powered mode.	

6.1.2 Get Interface Status

Offset	Number of Bytes	Description	Suggested Value
0	2	bits 15:0 = Reserved	0000h

6.1.3 Get Endpoint Status

Offset	Number of Bytes	Description	Suggested Value
0	2	bits 15:1 = Reserved bit 0 = Endpoint is stalled	0000h

6.1.4 Get Device Descriptor (18 Bytes)

Offset	Number of Bytes	Description	Suggested Value
0	1	Length	12h
1	1	Type (device)	
2	2	USB Specification Release Number	0100h
4	1	Class Code	
5	1	Sub Class Code	
6	1	Protocol	
7	1	Maximum Endpoint 0 Packet Size	10h
8	2	Vendor ID	0525h
10	2	Product ID	2890h
12	2	Device Release Number	
14	1	Index of string descriptor describing manufacturer	01h
15	1	Index of string descriptor describing product	02h
16	1	Index of string descriptor describing serial number	00h
17	1	Number of configurations	

6.1.5 Get Configuration Descriptor (55 bytes)

The NET2890 can support a variety of configurations, interfaces, and endpoints, each of which is defined by the descriptor data returned to the host. The local CPU has the responsibility of providing this data to the NET2890 when the host requests it.

This example has one configuration, and two interfaces, each with two endpoints. The first interface defines one Bulk OUT endpoint at address 1 with maximum packet size of 8 and one Interrupt IN endpoint at address 82h (endpoint number = 2) with a maximum packet size of 8. The second interface defines one Bulk OUT endpoint at address 3 with maximum packet size of 64 and one Bulk IN endpoint at address 84h (endpoint number = 4) with a maximum packet size of 64.

Note that all interface and endpoint descriptors are returned in response to a Get Configuration Descriptor request.

Offset	Number of Bytes	Description	Suggested Value
Configuration Descriptor			
0	1	Length	09h
1	1	Type (configuration)	02h
2	2	Total length returned for this configuration	0037h
4	1	Number of Interfaces	02h
5	1	Number of this configuration	01h
6	1	Index of string descriptor describing this configuration	00h
7	1	Attributes bit 7 = Bus Powered bit 6 = Self-Powered bit 5 = Remote-Wakeup bits 4:0 = Reserved	
8	1	Maximum USB power required (in 2 mA units)	
Interface 0 Descriptor			
0	1	Size of this descriptor in bytes	09h
1	1	Type (interface)	04h
2	1	Number of this interface	00h
3	1	Alternate Interface	00h
4	1	Number of endpoints used by this interface (excluding Endpoint 0)	02h
5	1	Class Code	00h
6	1	Sub Class Code	00h
7	1	Device Protocol	00h
8	1	Index of string descriptor describing this interface	00h

Get Configuration Descriptor (continued)

Offset	Number of Bytes	Description	Suggested Value
OUT Endpoint 1 Descriptor			
0	1	Size of this descriptor	07h
1	1	Descriptor Type (endpoint)	05h
2	1	Endpoint Address bit 7 = direction (1 = IN, 0 = OUT) bits 6:4 = reserved bits 3:0 = endpoint number	01h
3	1	Endpoint Attributes bits 7:2 = reserved bits 1:0 00 = Control 01 = Isochronous 10 = Bulk 11 = Interrupt	02h
4	2	Maximum packet size of this endpoint	0008h
6	1	Interval for polling endpoint (not used)	00h
IN Endpoint 2 Descriptor			
0	1	Size of this descriptor	07h
1	1	Descriptor Type (endpoint)	05h
2	1	Endpoint Address bit 7 = direction (1 = IN, 0 = OUT) bits 6:4 = reserved bits 3:0 = endpoint number	82h
3	1	Endpoint Attributes bits 7:2 = reserved bits 1:0 00 = Control 01 = Isochronous 10 = Bulk 11 = Interrupt	03h
4	2	Maximum packet size of this endpoint	0008h
6	1	Interval for polling endpoint	
Interface 1 Descriptor			
0	1	Size of this descriptor in bytes	09h
1	1	Type (interface)	04h
2	1	Number of this interface	01h
3	1	Alternate Interface	00h
4	1	Number of endpoints used by this interface (excluding Endpoint 0)	02h
5	1	Class Code	00h
6	1	Sub Class Code	00h
7	1	Device Protocol	00h
8	1	Index of string descriptor describing this interface	00h

Get Configuration Descriptor (continued)

Offse t	Number of Bytes	Description	Suggested Value
OUT Endpoint 3 Descriptor			
0	1	Size of this descriptor	07h
1	1	Descriptor Type (endpoint)	05h
2	1	Endpoint Address bit 7 = direction (1 = IN, 0 = OUT) bits 6:4 = reserved bits 3:0 = endpoint number	03h
3	1	Endpoint Attributes bits 7:2 = reserved bits 1:0 00 = Control 01 = Isochronous 10 = Bulk 11 = Interrupt	02h
4	2	Maximum packet size of this endpoint for bulk mode	0040h
6	1	Interval for polling endpoint	00h
IN Endpoint 4 Descriptor			
0	1	Size of this descriptor	07h
1	1	Descriptor Type (endpoint)	05h
2	1	Endpoint Address bit 7 = direction (1 = IN, 0 = OUT) bits 6:4 = reserved bits 3:0 = endpoint number	84h
3	1	Endpoint Attributes bits 7:2 = reserved bits 1:0 00 = Control 01 = Isochronous 10 = Bulk 11 = Interrupt	02h
4	2	Maximum packet size of this endpoint for bulk mode	0040h
6	1	Interval for polling endpoint	00h

6.1.6 Get String Descriptor 0

Offse t	Number of Bytes	Description	Suggested Value
0	4	Language ID (English = 09, U.S. = 04)	04h, 03h, 0409h

6.1.7 Get String Descriptor 1

Offse t	Number of Bytes	Description	Suggested Value
0	38	Manufacturer Descriptor. The text string is encoded in UNICODE.	26h, 03h, “NetChip Technology”

6.1.8 Get String Descriptor 2

Note: Strings are encoded using UNICODE.

Offset	Number of Bytes	Description	Suggested Value
0	66	Product Descriptor. The text string is encoded in UNICODE.	42h, 03h, "NET2890 USB Interface Controller"

6.1.9 Get Configuration

Offset	Number of Bytes	Description	Suggested Value
0	1	Returns current device configuration	00h or currently selected configuration.

6.1.10 Get Interface

Offset	Number of Bytes	Description	Suggested Value
0	1	Returns current alternate setting for the specified interface	00h or currently selected interface.

6.2 Control 'Write' Transfers

6.2.1 Set Address

Offset	Number of Bytes	Description	Suggested Value
--	0	Sets USB address of device Value = device address, Index = 0	--

Note: When a Set Address command is sent from the host, the local CPU must write the new device address into the OURADDR configuration register.

6.2.2 Set Configuration

Offset	Number of Bytes	Description	Suggested Value
--	0	Sets the device configuration Value = Configuration value	--

6.2.3 Set Interface

Offset	Number of Bytes	Description	Suggested Value
--	0	Selects alternate setting for specified interface Value = Alternate setting, Index = specified interface	--

6.2.4 Device Clear Feature

Offset	Number of Bytes	Description	Suggested Value

--	0	Clear the selected device feature
		Value = feature selector
		FS = 1 --> Device Remote Wakeup (disable)
			--

6.2.5 Device Set Feature

Offset	Number of Bytes	Description	Suggested Value

--	0	Set the selected device feature
		Value = feature selector
		FS = 1 --> Device Remote Wakeup (enable)
			--

6.2.6 Endpoint Clear Feature

Offset	Number of Bytes	Description	Suggested Value

--	0	Clear the selected endpoint feature
		Value = feature selector, Index = endpoint number
		FS = 0 --> Endpoint stall (clears stall bit)
			--

6.2.7 Endpoint Set Feature

Offset	Number of Bytes	Description	Suggested Value

--	0	Set the selected endpoint feature
		Value = feature selector, Index = endpoint number
		FS = 0 --> Endpoint stall (sets stall bit)
			--

7. Electrical Specifications

7.1 Absolute Maximum Ratings

Conditions that exceed the Absolute Maximum limits may destroy the device.

Symbol	Parameter	Conditions	Min	Max	Unit
V_{DDC}	Core/USB Supply Voltage	With Respect to Ground	-0.5	4.6	V
V_{DDL}	Local Supply Voltage (+5V)	With Respect to Ground	-0.5	7.0	V
V_I	DC input voltage	With Respect to Ground	-0.5	V_{DD}+0.5	V
I_{OUT}	DC Output Current, per pin		-25	25	mA
T_{STG}	Storage Temperature	No bias	-65	150	° C
T_{AMB}	Ambient temperature	Under bias	-65	150	° C
P_D	Power Dissipation	Under bias			mW
V_{ESD}	ESD Rating	R = 1.5K, C = 100pF		2	KV

7.2 Recommended Operating Conditions

Conditions that exceed the Operating limits may cause the device to function incorrectly.

Symbol	Parameter	Conditions	Min	Max	Unit
V_{DDC}	Core/USB Supply Voltage		3.0	3.6	V
V_{DDL}	Local Bus Supply Voltage	5.0V operation	4.75	5.25	V
V_{DDL}	Local Bus Supply Voltage	3.3V operation	3.0	3.6	V
V_{IH}	High Level Input Voltage		2		V
V_{IL}	Low Level Input Voltage			0.8	V
V_I	Input Voltage		0	V_{DDL}	V
V_O	Output Voltage		0	V_{DDL}	V
I_{OH}	High Level Output Current			-8	mA
I_{OL}	Low Level Output Current			8	mA
T_A	Operating Temperature		0	70	° C
t_R	Input rise time		1	10	ns/V
t_F	Input fall time		1	10	ns/V

7.3 DC Specifications

7.3.1 Core DC Specifications

Operating Conditions: V_{DDC}: 3.3V ± 5%, T_A = 0°C to 70°C
All typical values are at V_{DDC} = 3.3V and T_A = 25°C
Operating Conditions: Notes 1, 2, and 9.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
I_{DDC}	V_{DDC} Supply Current	V_{DDC} = 3.3V		45	60	mA
I_{DDCS}	V_{DDC} Supply Current (Suspend)	Device suspended		1	10	µA
T_J	Junction temperature	Under bias				° C

7.3.2 USB Port DC Specifications

Operating Conditions: V_{DDC}: 3.3V ± 5%, T_A = 0°C to 70°C
All typical values are at V_{DDC} = 3.3V and T_A = 25°C
Operating Conditions: Notes 1,2.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{DI}	Differential Input Sensitivity	| (D+) - (D-) |	0.2			V
V_{CM}	Differential Common Mode Range	Includes VDI range	0.8		2.5	V
V_{SE}	Single Ended Receiver Threshold		0.8		2.0	V
V_{OH}	Static Output High	RL of 15 KΩ to GND	2.8		3.6	V
V_{OL}	Static Output Low	RL of 1.5 KΩ to 3.6V			0.3	V
I_{LO}	Hi-Z State Data Line Leakage	0V < V_{IN} < 3.3V	-10		+10	µA
C_{IO}	I/O Capacitance	Pin to GND			20	pF

7.3.3 Local Bus (+3.3V) DC Specifications

Operating Conditions: V_{DDL}: 3.3V ± 5%, T_A = 0°C to 70°C
All typical values are at V_{DDL} = 3.3V and T_A = 25°C
Operating Conditions: Note 9

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{OH}	Output High Voltage	I_{OH} = -8mA	2.0			V
V_{OL}	Output Low Voltage	I_{OL} = 8mA			0.4	V
I_{IH}	Input High Leakage	V_{IH} = 3.3V			10	µA
I_{IL}	Input Low Leakage	V_{IL} = 0V			10	µA
I_{OZ}	Hi-Z State Data Line Leakage	0V < V_{IN} < 3.3V	-10		+10	µA
I_{DDLS}	V_{DDL} Supply Current (Suspend)	Device suspended		1	10	µA
I_{DDL}	V_{DDL} Supply Current	V_{DDC} = 3.3V		6	10	mA
C_{IO}	I/O Capacitance	Pin to GND			5	pF
C_{IN}	Input Capacitance	Pin to GND			10	pF

7.3.4 Local Bus (+5.0V) DC Specifications

Operating Conditions: V_{DDL}: 5.0V ± 5%, T_A = 0°C to 70°C
All typical values are at V_{DDL} = 5.0V and T_A = 25°C
Operating Conditions: Note 9

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{OH}	Output High Voltage	I_{OH} = -8mA	4.0			V
V_{OL}	Output Low Voltage	I_{OL} = 8mA			0.4	V
I_{IH}	Input High Leakage	V_{IH} = 5.0V			10	µA
I_{IL}	Input Low Leakage	V_{IL} = 0V			10	µA
I_{OZ}	Hi-Z State Data Line Leakage	0V < V_{IN} < 5.0V	-10		+10	µA
I_{DDLS}	V_{DDL} Supply Current (Suspend)	Device suspended		1	10	µA
I_{DDL}	V_{DDL} Supply Current	V_{DDC} = 3.3V		8	15	mA
C_{IO}	I/O Capacitance	Pin to GND			5	pF
C_{IN}	Input Capacitance	Pin to GND			10	pF

7.4 AC Specifications

7.4.1 USB Port AC Specifications

Operating Conditions: V_{DD}: 3.3V ± 5%, T_A = 0°C to 70°C
All typical values are at V_{DD} = 3.3V and T_A = 25°C
Operating Conditions: Notes 1,2,3.

Symbol	Parameter	Conditions	Waveform	Min	Typ	Max	Unit
T_R	Rise & Fall Times	C_L = 50 pF	Figure 8-1	4		20	ns
T_F		Notes 4,5		4		20	
T_{RFM}	Rise/Fall time matching	($T_{R/}$ T_F)	Figure 8-1	90		110	%
V_{CRS}	Output Signal Crossover Voltage			1.3		2.0	V
Z_{DRV}	Driver Output Resistance	Steady State Drive		28		43	Ω
T_{DRATE}	Data Rate			11.97	12	12.03	Mbs
T_{DDJ1}	Source Differential Driver Jitter to Next Transition	Notes 6,7	Figure 8-2	-3.5	0	3.5	ns
T_{DDJ2}	Source Differential Driver Jitter for Paired Transitions	Notes 6,7	Figure 8-2	-4.0	0	4.0	ns
T_{DEOP}	Differential to EOP Transition Skew	Note 7	Figure 8-3	-2	0	5	ns
T_{EOPT}	Source EOP Width	Note 7	Figure 8-3	160	167	175	ns
T_{JR1}	Receiver Data Jitter Tolerance to Next Transition	Note 7	Figure 8-4	-18.5	0	18.5	ns
T_{JR2}	Receiver Data Jitter Tolerance for Paired Transitions	Note 7	Figure 8-4	-9	0	9	ns
T_{EOPR1}	EOP Width at Receiver; Must reject as EOP	Note 7	Figure 8-3	40			ns
T_{EOPR2}	EOP Width at Receiver; Must accept as EOP	Note 7	Figure 8-3	80			ns

7.4.2 USB Port AC/DC Specification Notes

1. All voltages measured from the local ground potential, unless otherwise specified.
2. All timings use a capacitive load (C_L) to ground of 50 pF, unless otherwise specified.
3. Full Speed timings have a 1.5 kΩ pull-up to 3.3 V on the D+ data line.
4. Measured from 10% to 90% of the data signal.
5. The rising and falling edges should be smoothly transitioning (monotonic).
6. Timing difference between the differential data signals.
7. Measured at crossover point of differential data signals.
8. The maximum load specification is the maximum effective capacitive load allowed that meets the target hub V_{BUS} droop of 330 mV.
9. V_{DDC} and I_{DDC} refer to core power supply (pins designated V_{DD}). V_{DDL} and I_{DDL} refer to local bus power supply (pins designated $V_{DDL(LOCAL)}$).

7.4.3 USB Port AC Waveforms

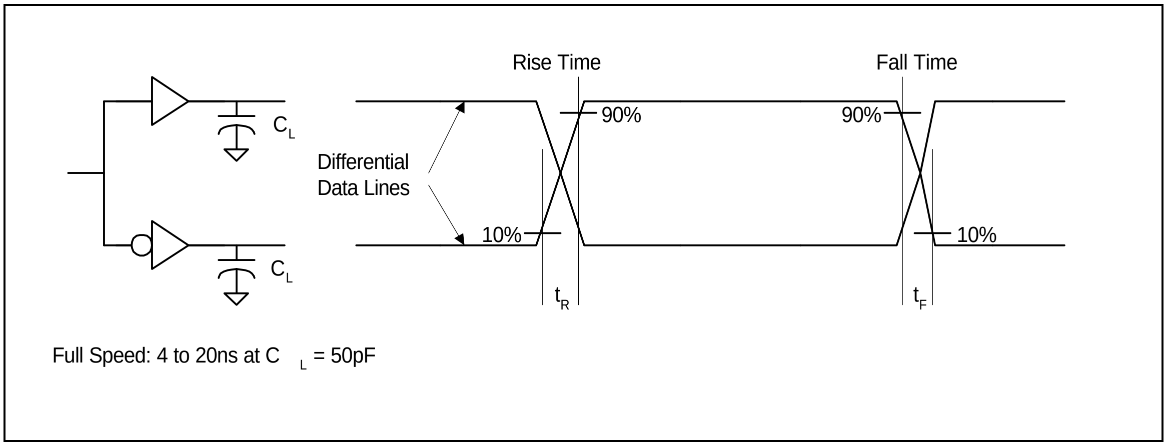


Figure 7-1. Data Signal Rise and Fall Time

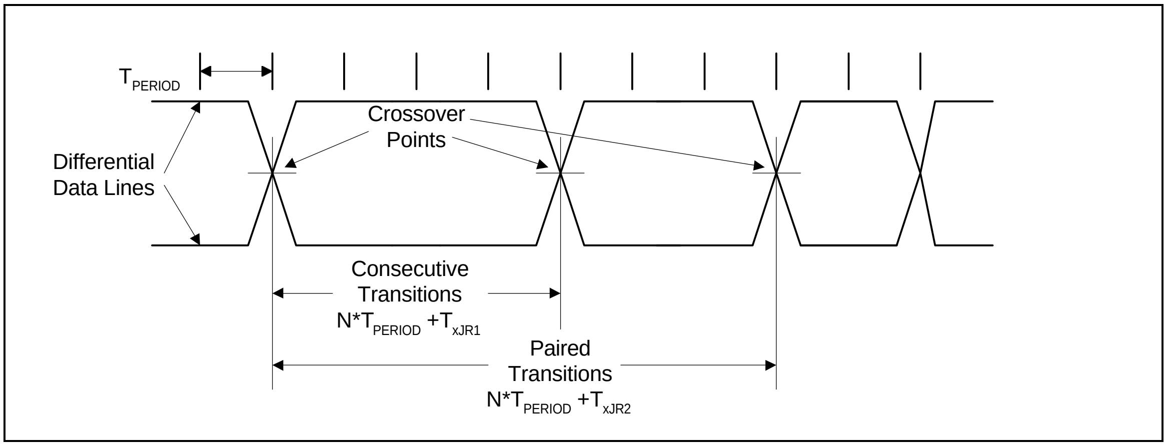


Figure 7-2. Differential Data Jitter

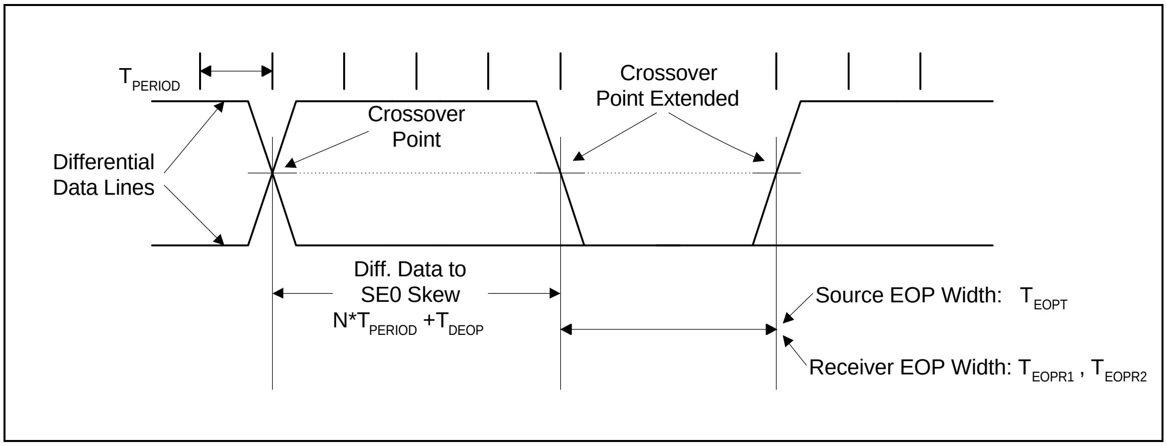

T_{PERIOD}
Crossover Point
Crossover Point Extended
Differential Data Lines
Diff. Data to SE0 Skew
$N*T_{PERIOD} + T_{DEOP}$
Source EOP Width: T_{EOPT}
Receiver EOP Width: T_{EOPR1} , T_{EOPR2}

Figure 7-3. Differential to EOP Transition Skew and EOP Width

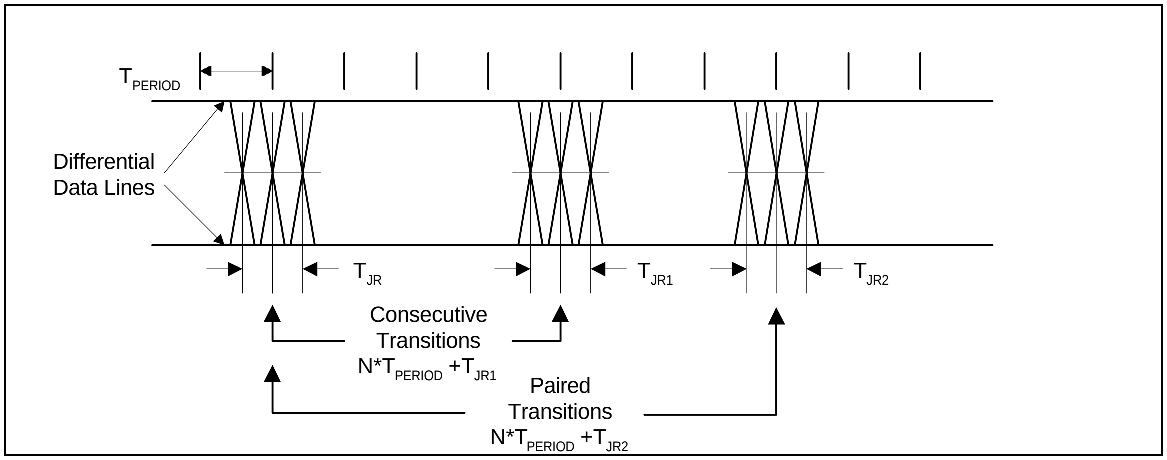
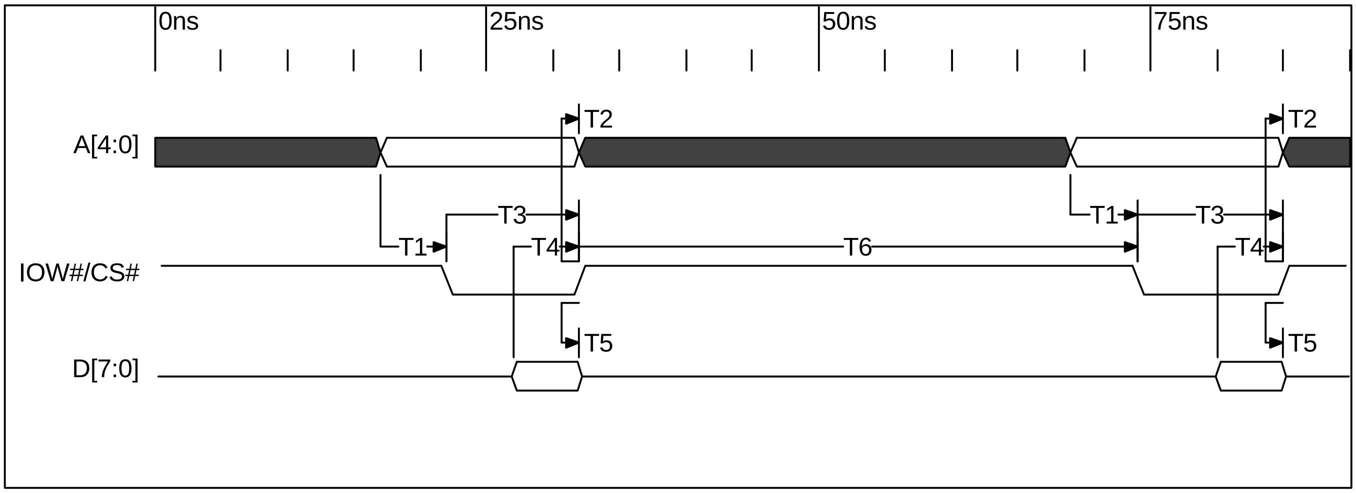


Figure 7-4. Receiver Jitter Tolerance

7.4.4 Local Bus Write to Register

NAME	DESCRIPTION	MIN	MAX	UNIT
T1	Address setup to write enable*	5		ns
T2	Address hold from end of write enable*	0		ns
T3	Write enable width*	10		ns
T4	Data setup to end of write enable*	5		ns
T5	Data hold time from end of IOW#	0		ns
T6	I/O Recovery Time	42		ns

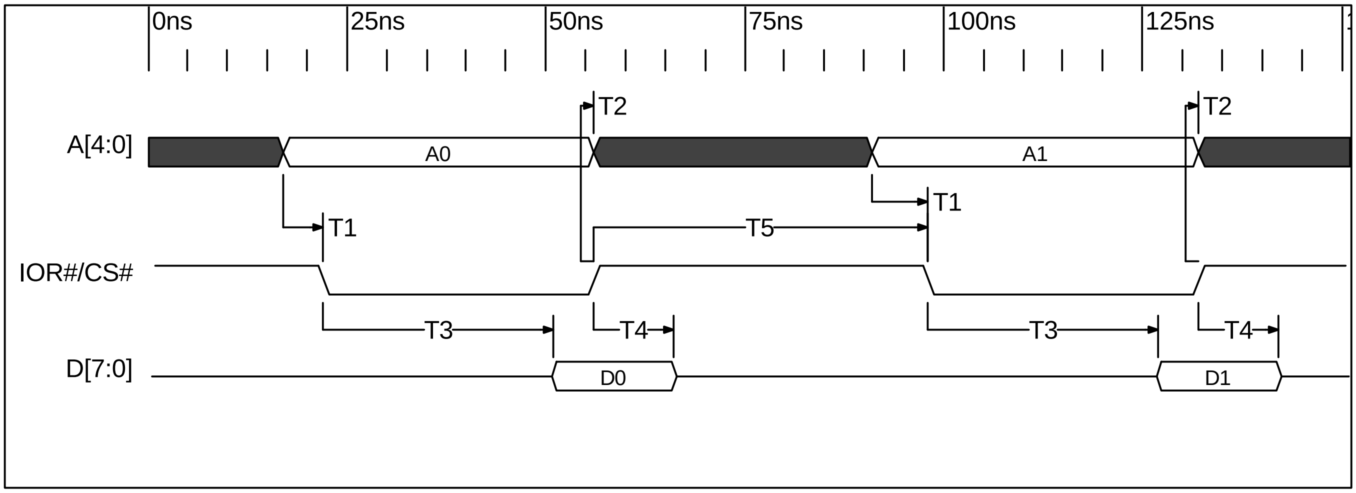
* Write enable is the occurrence of both IOW# and CS#.



7.4.5 Local Bus Read from Register

NAME	DESCRIPTION	MIN	MAX	UNIT
T1	Address setup to read enable*	5		ns
T2	Address hold from end of read enable*	0		ns
T3	Data access time from read enable*		29	ns
T4	Data tri-state time from end of IOR#	0	10	ns
T5	I/O Recovery Time	42		ns

* Read enable is the occurrence of both IOR# and CS#

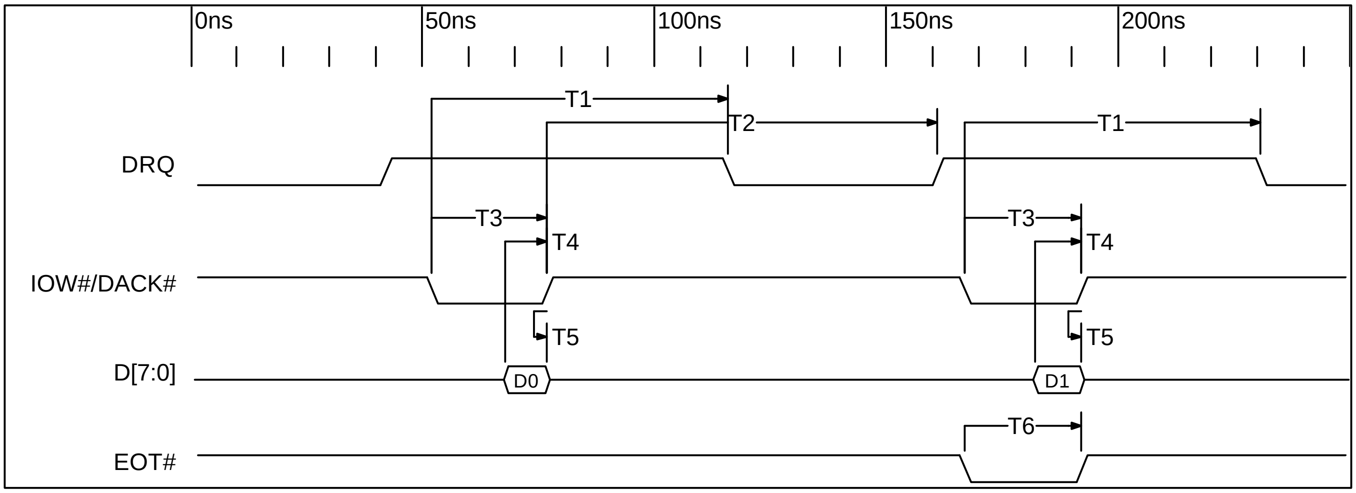


7.4.6 DMA Write to FIFO

NAME	DESCRIPTION	MIN	MAX	UNIT
T1	DRQ false from write enable true	42	64	ns
T2	Write enable false to DRQ true	84		ns
T3	Write enable width	25		ns
T4	Data setup to end of write enable	5		ns
T5	Data hold time from end of write enable	0		ns
T6	Width of EOT# pulse (see note)	25		ns

* Write enable is the occurrence of both IOW# and DACK#.

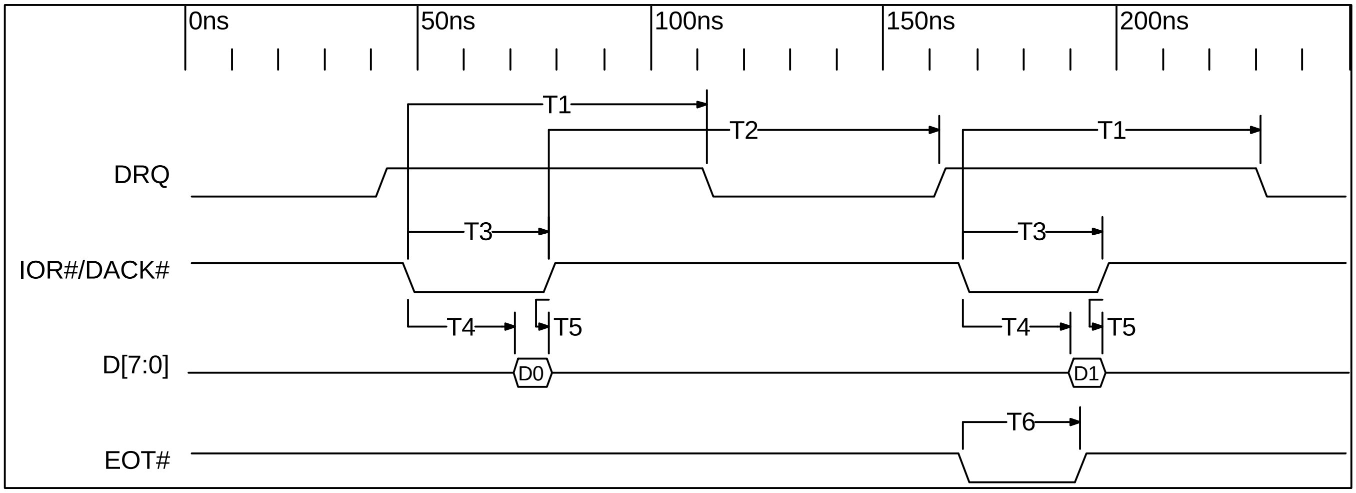
Note: EOT#, IOW#, and DACK# must be concurrently true for at least T6 for proper recognition of the EOT# pulse.



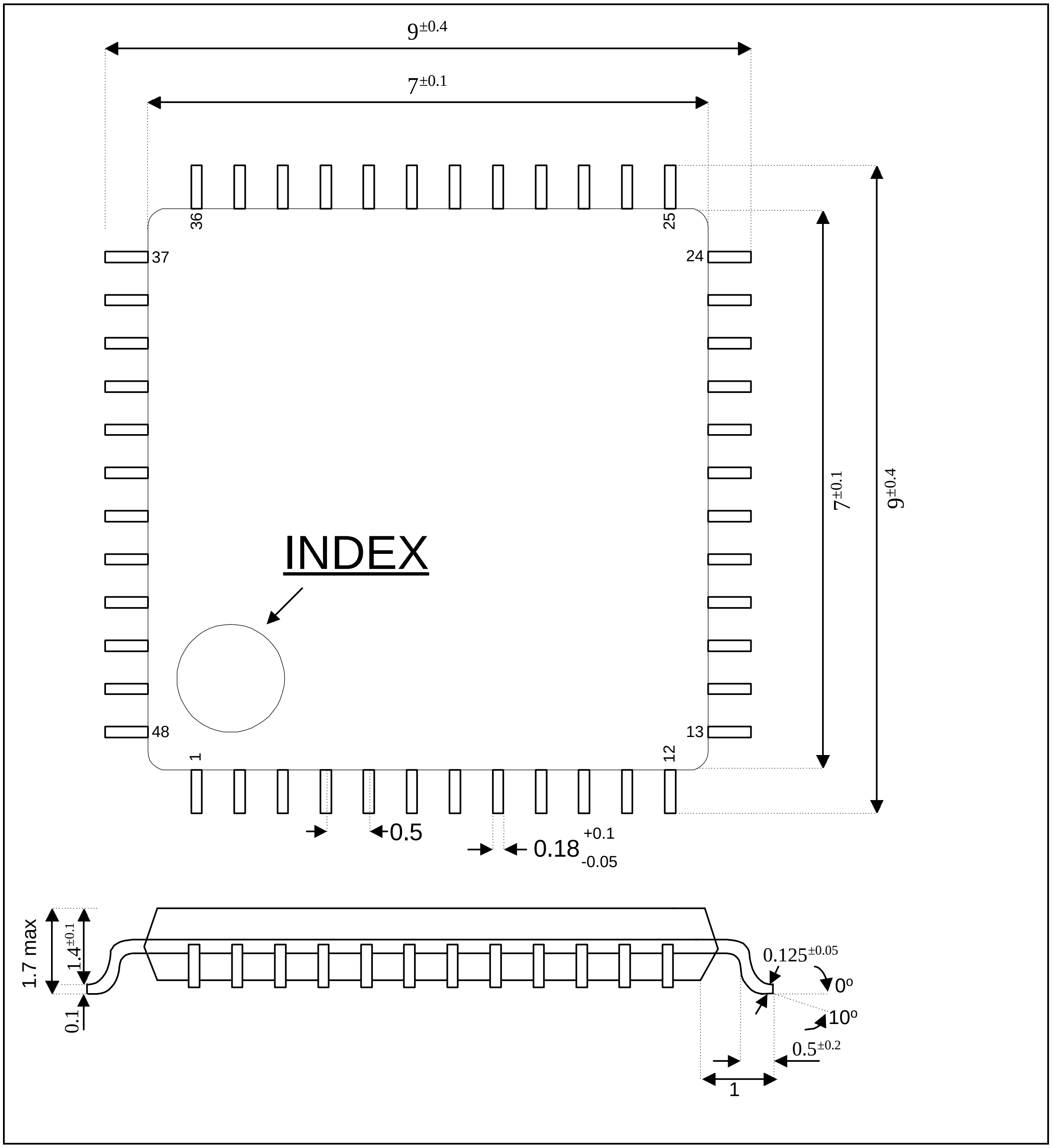
7.4.7 DMA Read from FIFO

NAME	DESCRIPTION	MIN	MAX	UNIT
T1	DRQ false from read enable true	42	64	ns
T2	Read enable false to DRQ true	30		ns
T3	Read enable width	25		ns
T4	Data access time from read enable		23	ns
T5	Data hold time from end of read enable	0	10	ns
T6	Width of EOT# pulse (see note)	25		ns

* Read enable is the occurrence of both IOR# and DACK#.
Note: EOT#, IOR#, and DACK# must be concurrently true for at least T6 for proper recognition of the EOT# pulse.



8. Mechanical Drawing



All dimensions in millimeters.