

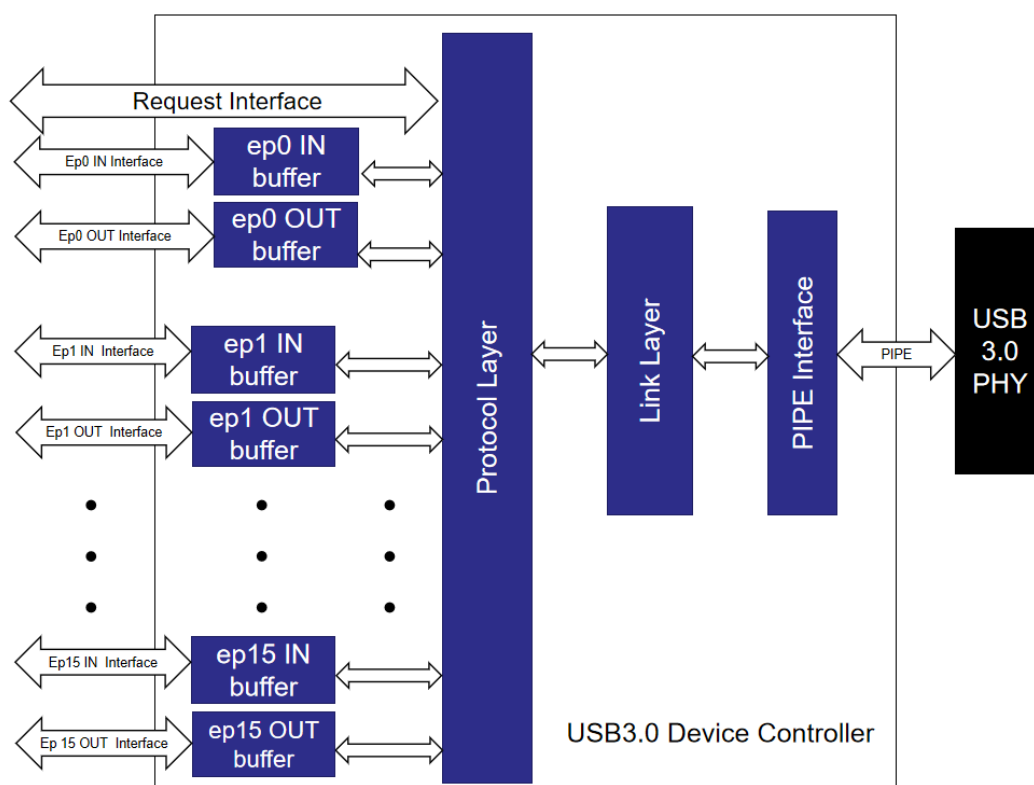
# Gowin USB3.0 Device Controller IP Application Note

## Functional Description

### USB3.0 Device Controller

USB3.0 Device Controller is located between User Design and PHY. USB3.0 Controller connects user design with PHY in series, receives commands from USB and realizes the data transaction between user design and USB. The functional block diagram of USB device controller is as shown below.

Figure 1 USB3.0 Device Controller Block Diagram



## USB3.0 Device Controller IN Endpoint User Interface

Table 1 Device Controller IN Endpoint User Interface

Name	I/O	Bit Width	Description
epX_in_buf_data_i	I	32	Buffer write data Bit            Description [31:24]      Byte 0 [23:16]      Byte 1 [15:8]        Byte 2 [7:0]         Byte 3
epX_in_buf_wren_i	I	1	Buffer write enable signal If it is set to 1, epX_in_buf_data_i will be written to the IN buffer on the next clock edge.
epX_in_buf_eob_i	I	1	Data packet end-of-burst signal When the data packet write completion signal is active, the Device Controller writes this signal value to the EOB/LPF field of the data packet.
epX_in_buf_data_commit_i	I	1	Data packet write completion signal After each complete data packet is written, this signal needs to be asserted high for one clock cycle. The Device Controller will send the packet data to the host during the next polling cycle.
epX_in_buf_data_commit_len_i	I	11	This data packet length ranges from 0 to 1024. After the write completion signal is set to 1, the Device Controller latches the length of this data packet in the next clock cycle.
epX_in_buf_ready_o	O	1	Buffer idle signal If set to 1, it indicates that writing a data packet is allowed; otherwise, writing is not allowed.

**Note!**

X indicates a specific endpoint.

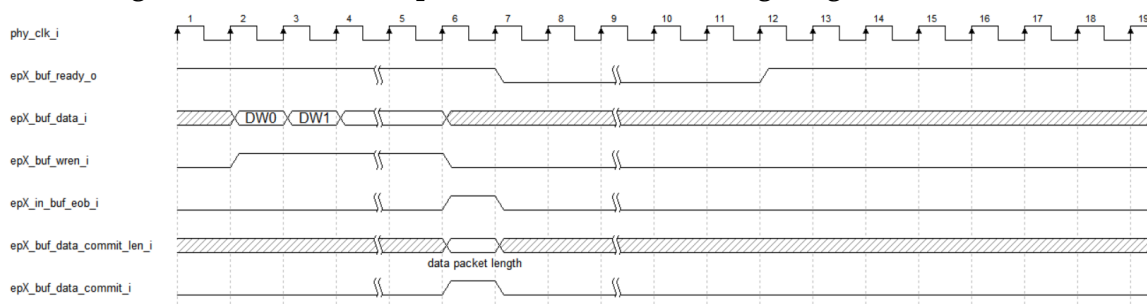
The Device Controller allocates IN buffer internally for the IN endpoint to store data packets awaiting transmission. The buffer can store up to N data packets, where N is equal to the maximum burst size for the corresponding endpoint. If epX\_in\_buf\_ready\_o is set to 1, it indicates that the IN buffer is idle and a data packet can be stored. Otherwise, data packets cannot be written.

The user writes data by asserting `epX_in_buf_wren_i` high. After the user has completely written a data packet, he needs to assert `epX_in_buf_data_commit_i` high for one clock cycle, sets `epX_in_buf_data_commit_len_i` to the length of the data packet, and properly configures `epX_in_buf_eob_i`. The Device Controller will read this data packet and send it to the host during the next polling of the endpoint.

During host polling, if there is no complete data packet in the IN buffer for the polled endpoint, the Device Controller will automatically respond to the host based on the endpoint type. For isochronous endpoints, the Device Controller will not respond; otherwise, it will reply with an NRDY.

The below is the IN Endpoint user interface timing diagram, illustrating the process of writing a data packet to endpoint X. The data packet shown is the last packet in a burst transfer.

**Figure 2 USB 3.0 IN Endpoint User Interface Timing Diagram**



## USB3.0 Device Controller OUT Endpoint User Interface

**Table 2 Device Controller OUT Endpoint User Interface**

Name	I/O	Bit Width	Description										
epX_out_buf_data_o	O	32	Buffer read data <table><tr><th>Bit</th><th>Description</th></tr><tr><td>[31:24]</td><td>Byte 0</td></tr><tr><td>[23:16]</td><td>Byte 1</td></tr><tr><td>[15:8]</td><td>Byte 2</td></tr><tr><td>[7:0]</td><td>Byte 3</td></tr></table>	Bit	Description	[31:24]	Byte 0	[23:16]	Byte 1	[15:8]	Byte 2	[7:0]	Byte 3
Bit	Description												
[31:24]	Byte 0												
[23:16]	Byte 1												
[15:8]	Byte 2												
[7:0]	Byte 3												
epX_out_buf_rden_i	I	1	Buffer read enable When asserted high, the Device Controller will send the data to epX_buf_data_o in the next clock cycle.										
epX_out_buf_len_o	O	11	Packet length										
epX_out_buf_has_data_o	O	1	Indicator for Buffer containing packet										

Name	I/O	Bit Width	Description
			If the signal is 1, it indicates that there is a packet of data in the buffer, and the length of this packet is indicated by epX_buf_len_o. Otherwise, it indicates that the buffer does not contain any packets.
epX_out_buf_data_ack_i	I	1	Packet read completion signal After the user has completed reading a packet of data, this signal is asserted high for one clock cycle.

**Note!**

X indicates a specific endpoint.

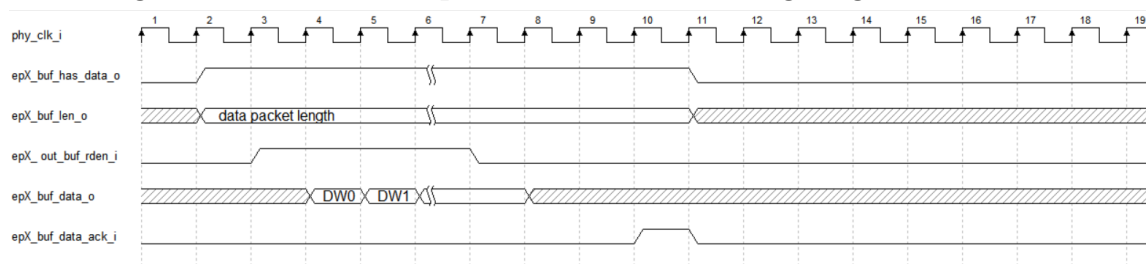
The Device Controller internally allocates OUT buffer for the OUT endpoint to store received packets. The OUT buffer can store up to N packets, where N is equal to the maximum burst packet size for the corresponding endpoint. When the Device Controller receives a packet sent by the host, the epX\_out\_buf\_has\_data\_o will be set to 1 and epX\_out\_buf\_len\_o will be set to the length of the packet.

When epX\_out\_buf\_has\_data\_o is set to 1, the user can assert epX\_out\_buf\_rden\_i high to read the packet. After the user completes reading a packet, epX\_out\_buf\_data\_ack\_i needs to be asserted high for one clock cycle.

If the Device Controller receives a packet from the host and there is no available space in the OUT buffer of the polled endpoint, the Device Controller will automatically respond to the host based on the endpoint type. For isochronous endpoints, the Device Controller will not respond; otherwise, it will reply with an NRDY.

The following diagram shows the timing of OUT Endpoint user interface, illustrating the process of the user reading a packet of data from the endpoint X.

**Figure 3 USB3.0 OUT Endpoint User Interface Timing Diagram**



## USB3.0 Device Controller Request User Interface

This user interface is specifically used for control transfers, i.e., data transfers on endpoint 0.

### Request User Interface

Table 3 Ep0 SETUP User Interface

Name	I/O	Bit Width	Description
request_active_o	O	1	RX request valid signal, active-high. When active, it indicates that the device has received a request from the host.
bmRequestType_o	O	8	When request_active_o is active, it indicates that the bmRequestType field value of the received Setup packet will remain unchanged until a new request is received.
bRequest_o	O	8	When request_active_o is active, it indicates that the bRequest field value of the received Setup packet will remain unchanged until a new request is received.
wValue_o	O	16	When request_active_o is active, it indicates that the wValue field value of the received Setup packet will remain unchanged until a new request is received.
wIndex_o	O	16	When request_active_o is active, it indicates that the wIndex field value of the received Setup packet will remain unchanged until a new request is received.
wLength_o	O	16	When request_active_o is active, it indicates that the wLength field value of the received Setup packet will remain unchanged until a new request is received.

USB 3.0 control transfers consist of three stages: Setup stage, the Data stage, and Status stage. The Data stage may not exist.

During the Setup stage, after the Device Controller receives a Setup packet sent by the Host, it automatically parses the packet. If the request is SET ADDRESS, the Device Controller will receive the assigned address and use it for communication with the host. If the request is not SET ADDRESS, the request will be sent to the user through this interface. If the request includes a Data stage, the Device Controller will automatically transition to the Data stage; otherwise, it will move to the Status stage.

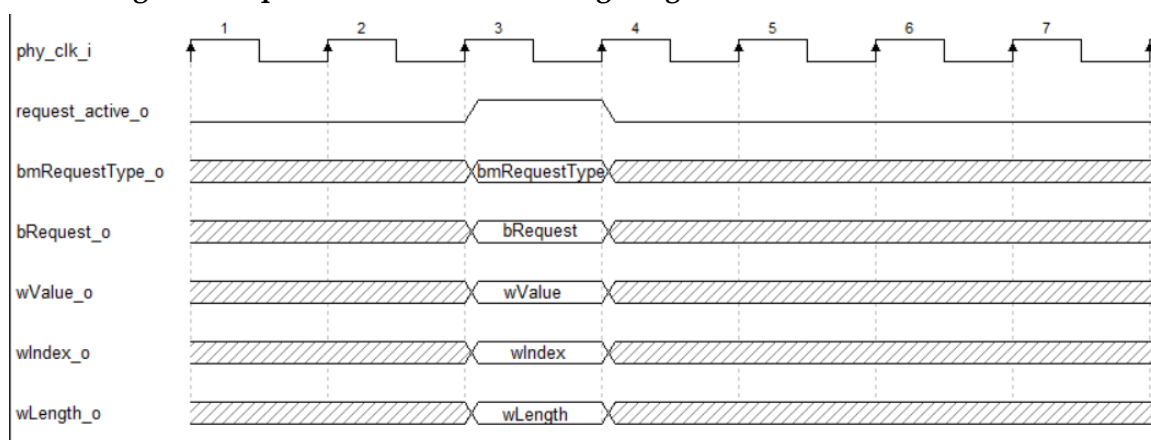
During the Data stage, if the host needs to send data to the device, the

Device Controller will send the received data to the user through the Ep0 OUT user interface. If the host needs to read data from the device, the user needs to write the data through the Ep0 IN user interface, and the Device Controller will read and send it to the host.

During the Status stage, the Device Controller handles the process automatically, so the user does not need to manage this stage.

The diagram below illustrates the timing of the Request user interface during the Setup stage, showing the process of the Device Controller sending the Setup packet to the user.

**Figure 4 Request User Interface Timing Diagram**



### Ep0 IN User Interface

You can refer to USB3.0 Device Controller IN Endpoint User Interface.

### Ep0 OUT User Interface

You can refer to USB3.0 Device Controller OUT Endpoint User Interface.

## USB3.0 Device Controller PIPE Interface

**Table 4 Device Controller PIPE Interface**

Name	I/O	Bit Width	Description
phy_clk_i	I	1	Connects to the PCLK of PIPE interface
phy_pipe_rx_data_i	I	32	Connects to the RxData[31:0] of PIPE interface
phy_pipe_rx_datak_i	I	4	Connects to the RxDataK[3:0] of PIPE interface
phy_pipe_rx_valid_i	I	1	Connects to the RxValid of PIPE interface
phy_pipe_tx_data_o	O	32	Connects to the TxData[31:0] of PIPE interface
phy_pipe_tx_datak_o	O	4	Connects to the TxDataK[3:0] of PIPE interface
phy_reset_n_o	O	1	Connects to the Reset# of PIPE interface
phy_tx_detrx_lpbk_o	O	1	Connects to the TxDetectRx/Loopback of PIPE interface
phy_tx_elecidle_o	O	1	Connects to the TxElecIdle of PIPE interface
phy_rx_elecidle_i	I	1	Connects to the RxElecIdle of PIPE interface
phy_rx_status_i	I	3	Connects to the RxStatus[2:0] of PIPE interface
phy_power_down_o	O	2	Connects to the PowerDown[1:0] of PIPE interface
phy_phy_status_i	I	1	Connects to the PhyStatus of PIPE interface
phy_pwrpresent_i	I	1	Connects to the PowerPresent of PIPE interface
phy_tx_oneszeros_o	I	1	Connects to the TxOnesZeros of PIPE interface
phy_tx_deemph_o	O	2	Connects to the TxDeemph[1:0] of PIPE interface
phy_tx_margin_o	O	3	Connects to the TxMargin[2:0] of PIPE interface
phy_tx_swing_o	O	1	Connects to the TxSwing of PIPE interface
phy_rx_polarity_o	O	1	Connects to the RxPolarity of PIPE interface
phy_rx_termination_o	O	1	Connects to the RX Termination of PIPE interface
phy_rate_o	O	1	Connects to the Rate of PIPE interface
phy_elas_buf_mode_o	O	1	Connects to the Elasticity Buffer Mode of PIPE interface

This interface can connect with any USB 3.0 PHY that supports the PIPE interface. The PCLK in the USB 3.0 PHY should be 125 MHz, and the data bus width should be 32 bits.

## Signal Definition

The signal definition of Gowin USB3.0 Device Controller IP is as shown below.

**Table 5 Signal Definition**

No.	Signal Name	I/O	Data Width	Description
1	phy_clk_i	I	1	Connects to the PCLK of PIPE interface
2	phy_pipe_rx_data_i	I	32	Connects to the RxData[31:0] of PIPE interface
3	phy_pipe_rx_datak_i	I	4	Connects to the RxDataK[3:0] of PIPE interface
4	phy_pipe_rx_valid_i	I	1	Connects to the RxValid of PIPE interface
5	phy_pipe_tx_data_o	O	32	Connects to the TxData[31:0] of PIPE interface
6	phy_pipe_tx_datak_o	O	4	Connects to the TxDataK[3:0] of PIPE interface
7	phy_reset_n_o	O	1	Connects to the Reset# of PIPE interface
8	phy_tx_detrx_lpbk_o	O	1	Connects to the TxDetectRx/Loopback of PIPE interface
9	phy_tx_elecidle_o	O	1	Connects to the TxElecIdle of PIPE interface
10	phy_rx_elecidle_i	I	1	Connects to the RxElecIdle of PIPE interface
11	phy_rx_status_i	I	3	Connects to the RxStatus[2:0] of PIPE interface
12	phy_power_down_o	O	2	Connects to the PowerDown[1:0] of PIPE interface
13	phy_phy_status_i	I	1	Connects to the PhyStatus of PIPE interface
14	phy_pwrpresent_i	I	1	Connects to the PowerPresent of PIPE interface
15	phy_tx_oneszeros_o	I	1	Connects to the TxOnesZeros of PIPE interface
16	phy_tx_deemph_o	O	2	Connects to the TxDeemph[1:0] of PIPE interface
17	phy_tx_margin_o	O	3	Connects to the TxMargin[2:0] of PIPE



No.	Signal Name	I/O	Data Width	Description
				interface
18	phy_tx_swing_o	O	1	Connects to the TxSwing of PIPE interface
19	phy_rx_polarity_o	O	1	Connects to the RxPolarity of PIPE interface
20	phy_rx_termination_o	O	1	Connects to the RX Termination of PIPE interface
21	phy_rate_o	O	1	Connects to the Rate of PIPE interface
22	phy_elas_buf_mode_o	O	1	Connects to the Elasticity Buffer Mode of PIPE interface
23	request_active_o	O	1	RX request valid signal, active-high. When active, it indicates that the device has received a request from the host.
24	bmRequestType_o	O	8	When request_active_o is active, the value of bmRequestType_o equals the bmRequestType field value of the received Setup packet.
25	bRequest_o	O	8	When request_active_o is active, the value of bRequest_o equals the bRequest field value of the received Setup packet.
26	wValue_o	O	16	When request_active_o is active, the value of wValue_o equals the wValue field value of the received Setup packet.
27	wIndex_o	O	16	When request_active_o is active, the value of wIndex_o equals the wIndex field value of the received Setup packet.
28	wLength_o	O	16	When request_active_o is active, the value of wLength_o equals the wLength field value of the received Setup packet.
29	epX_in_buf_data_i	I	32	Buffer write data <div> <div>Bit</div> <div>Description</div> <div>[31:24]</div> <div>Byte 0</div> <div>[23:16]</div> <div>Byte 1</div> <div>[15:8]</div> <div>Byte 2</div> <div>[7:0]</div> <div>Byte 3</div> </div>
30	epX_in_buf_wren_i	I	1	Buffer write enable signal

No.	Signal Name	I/O	Data Width	Description										
				If it is set to 1, epX_in_buf_data_i will be written to the IN buffer on the next clock edge.										
31	epX_in_buf_eob_i	I	1	Data packet end-of-burst signal When the data packet write completion signal is active, the Device Controller writes this signal value to the EOB/LPF field of the data packet.										
32	epX_in_buf_data_commit_i	I	1	Data packet write completion signal After each complete data packet is written, this signal needs to be asserted high for one clock cycle. The Device Controller will send the packet data to the host during the next polling cycle.										
33	epX_in_buf_data_commit_len_i	I	11	This data packet length ranges from 0 to 1024. After the write completion signal is set to 1, the Device Controller latches the length of this data packet in the next clock cycle.										
34	epX_in_buf_ready_o	O	1	Buffer idle signal If it is set to 1, it indicates that writing a data packet is allowed; otherwise, writing is not allowed.										
35	epX_out_buf_data_o	O	32	Buffer read data <table><thead><tr><th>Bit</th><th>Description</th></tr></thead><tbody><tr><td>[31:24]</td><td>The i-th byte of the packet</td></tr><tr><td>[23:16]</td><td>The (i+1)-th byte of the packet</td></tr><tr><td>[15:8]</td><td>The (i+2)-th byte of the packet</td></tr><tr><td>[7:0]</td><td>The (i+3)-th byte of the packet</td></tr></tbody></table>	Bit	Description	[31:24]	The i-th byte of the packet	[23:16]	The (i+1)-th byte of the packet	[15:8]	The (i+2)-th byte of the packet	[7:0]	The (i+3)-th byte of the packet
Bit	Description													
[31:24]	The i-th byte of the packet													
[23:16]	The (i+1)-th byte of the packet													
[15:8]	The (i+2)-th byte of the packet													
[7:0]	The (i+3)-th byte of the packet													
36	epX_out_buf_rden_i	I	1	Buffer read enable When asserted high, the Device Controller will send the data to epX_buf_data_o in the next clock cycle.										
37	epX_out_buf_len_o	O	11	Packet length										
38	epX_out_buf_has_data_o	O	1	Indicator for Buffer containing packet If the signal is 1, it indicates that there is a										

No.	Signal Name	I/O	Data Width	Description
				packet of data in the buffer, and the length of this packet is indicated by epX_buf_len_o. Otherwise, it indicates that the buffer does not contain any packets.
39	epX_out_buf_data_ack_i	I	1	Packet read completion signal After the user has completed reading a packet of data, this signal is asserted high for one clock cycle.
40	warm_or_hot_reset_o	O	1	Host reset signal, active-high When active, it indicates that the host sends warm reset or hot reset.
41	host_requests_data_from_endpt_o	O	1	Host data request signal, active-high. When active, it indicates that the host is polling the IN endpoint. If the IN buffer of Device Controller contains a packet of data, the Device Controller will send this packet to the host. Otherwise, the Device Controller will respond with NRDY.
42	host_requests_endpt_num_o	O	4	Host polling data endpoint address When host_requests_data_from_endpt_o is active, this signal indicates the specific IN endpoint address that the host is polling.

**Note!**

X indicates a specific endpoint.

## Parameter Configuration Option

The file usb3\_macro\_define.v contains device endpoint numbers and endpoint parameters.

**Table 6 Configuration Option Description**

Parameter	Description										
ENDPTX_IN	If this parameter is defined, IN endpoint X is enabled.										
ENDPTX_IN_TYPE	<p>If IN endpoint X is enabled, this parameter defines the transfer mode for IN endpoint X as shown below:</p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>01b</td><td>Synchronous transfer</td></tr> <tr> <td>10b</td><td>Bulk transfer</td></tr> <tr> <td>11b</td><td>Interrupt transfer</td></tr> <tr> <td>00b</td><td>Undefined</td></tr> </table>	Value	Description	01b	Synchronous transfer	10b	Bulk transfer	11b	Interrupt transfer	00b	Undefined
Value	Description										
01b	Synchronous transfer										
10b	Bulk transfer										
11b	Interrupt transfer										
00b	Undefined										
ENDPTX_IN_BURST	If IN endpoint X is enabled, this parameter defines the maximum burst count for IN endpoint X, ranging from 1 to 16.										
ENDPTX_IN_MAX	If IN endpoint X is enabled, this parameter defines the maximum packet length for IN endpoint X, ranging from 1 to 1024.										
ENDPTX_OUT	If this parameter is defined, OUT endpoint X is enabled.										
ENDPTX_OUT_TYPE	<p>If OUT endpoint X is enabled, this parameter defines the transfer mode for OUT endpoint X as shown below:</p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>01b</td><td>Synchronous transfer</td></tr> <tr> <td>10b</td><td>Bulk transfer</td></tr> <tr> <td>11b</td><td>Interrupt transfer</td></tr> <tr> <td>00b</td><td>Undefined</td></tr> </table>	Value	Description	01b	Synchronous transfer	10b	Bulk transfer	11b	Interrupt transfer	00b	Undefined
Value	Description										
01b	Synchronous transfer										
10b	Bulk transfer										
11b	Interrupt transfer										
00b	Undefined										
ENDPTX_OUT_BURST	If OUT endpoint X is enabled, this parameter defines the maximum burst count for OUT endpoint X, ranging from 1 to 16.										
ENDPTX_OUT_MAX	If OUT endpoint X is enabled, this parameter defines the maximum packet length for OUT endpoint X, ranging from 1 to 1024.										

**Note!**

The X in ENDPTX means the endpoint number, ranging from 1 to 15.

## Support and Feedback

Gowin Semiconductor provides customers with comprehensive technical support. If you have any questions, comments, or suggestions, please feel free to contact us directly using the information provided below.


Website: [www.gowinsemi.com](http://www.gowinsemi.com)

E-mail: [support@gowinsemi.com](mailto:support@gowinsemi.com)

## Revision History

Date	Version	Description
08/13/2024	1.0E	Initial version published.

**Copyright © 2024 Guangdong Gowin Semiconductor Corporation. All Rights Reserved.**

**GOWIN**  is a trademark of Guangdong Gowin Semiconductor Corporation and is registered in China, the U.S. Patent and Trademark Office, and other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders. No part of this document may be reproduced or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior written consent of GOWINSEMI.

#### **Disclaimer**

GOWINSEMI assumes no liability and provides no warranty (either expressed or implied) and is not responsible for any damage incurred to your hardware, software, data, or property resulting from usage of the materials or intellectual property except as outlined in the GOWINSEMI Terms and Conditions of Sale. GOWINSEMI may make changes to this document at any time without prior notice. Anyone relying on this documentation should contact GOWINSEMI for the current documentation and errata.