DS\_BASIC IP SPEC

Table of Contents

[Introduction 2](#_Toc116983014)

[Feature（silergy internal） 2](#_Toc116983015)

[Register Definition 2](#_Toc116983016)

[Register Map 2](#_Toc116983017)

[Functional Details 3](#_Toc116983018)

[Block Diagram 3](#_Toc116983019)

[Module input/output list 3](#_Toc116983020)

[DS\_BASIC function description 6](#_Toc116983021)

## Introduction

The DS\_BASIC module is used to convert input dual daisy chain signals to byte data rx\_data[8:0], and convert byte data tx\_data[8:0] to dual daisy chain outputs.

## Feature

Key features of the DS\_BASIC module are:

• Analysis received daisy chain data

• Send daisy chain data

• Direction control

## Register Definition

### Register Map

Table 1 DS\_BASIC Register Map

| **Name** | **Add** | **D7** | **D6** | **D5** | **D4** | **D3** | **D2** | **D1** | **D0** | **Default** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| COMM\_CONF2 | 0x0003 | COMN\_TX\_DIS | COMS\_TX\_DIS | STACK\_RESPONSE<5:0> | | | | | | 00 |
| CTRL2 | 0x2003 |  |  |  |  |  | CMP\_BIST\_GO | ADD\_W\_EN | SPI\_DIR |  |

## Functional Details

### Block Diagram

The following diagram shows the DS\_BASIC architecture and internal modules and connections.



Figure1 DS\_BASIC diagram

### Module input/output list

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Dir | Width | Discirption | duration |
| rev\_dsy\_data | O | 9 | received daisy chain data | Level(32M domain) |
| neg\_rx\_en\_dsy | O | 1 | negedge of rx\_en\_dsy | 4 CLK\_32M |
| neg\_rx\_en\_dsy\_8M | O | 1 | negedge of rx\_en\_s\_dsy or rx\_en\_n\_dsy | 1 CLK\_REG |
| neg\_rx\_en\_s\_dsy | O | 1 | negedge of rx\_en\_s\_dsy | 4 CLK\_32M |
| neg\_rx\_en\_n\_dsy | O | 1 | negedge of rx\_en\_n\_dsy | 4 CLK\_32M |
| MISS | O | 1 |  |  |
| ORDER | O | 1 |  |  |
| SYNCT | O | 1 |  |  |
| SYNCD | O | 1 |  |  |
| BIT | O | 1 |  |  |
| WR\_COM\_PLUS | O | 1 | daisy chain output data in positive phase | 8 CLK\_32M |
| WR\_COM\_MINUS | O | 1 | daisy chain output data in negtive phase | 8 CLK\_32M |
| TX\_EN\_S | O | 1 | enable daisy chain transmitting on S port |  |
| TX\_EN\_N | O | 1 | enable daisy chain transmitting on N port |  |
| D2A\_RX\_EN\_S | O | 1 | enable daisy chain receiving on S port |  |
| D2A\_RX\_EN\_N | O | 1 | enable daisy chain receiving on N port |  |
| send\_char\_end\_pos | O | 1 | mark byte transmitting end time | 4 CLK\_32M |
| tx\_crc | O | 16 | crc16 result of tx\_one |  |
| rx\_en\_n | O | 1 | daisy chai signal is being received on N port |  |
| rx\_en\_s | O | 1 | daisy chai signal is being received on S port |  |
| rx\_en | O | 1 | daisy chai signal is being received |  |
| tx\_en\_32M | O | 1 | sync send\_start with CLK\_32M | 2 CLK\_32M |
| neg\_TX\_EN\_S | O | 1 | negedge of TX\_EN\_S | 1 CLK\_32M |
| neg\_TX\_EN\_N | O | 1 | negedge of TX\_EN\_N | 1 CLK\_32M |
| clr\_crc\_dsy | O | 1 | crc clear | 3~4 CLK\_32M |
| TX\_timeout | O | 1 | no data to tranmit for a timeout time when TX\_EN\_X high |  |
| pos\_TBYTE\_FAST | O | 1 | fault flag: receiving data is too fast | 4 CLK\_32M |
| pos\_TBYTE\_TO | O | 1 | fault flag: receiving data is too slow | 4 CLK\_32M |
| CLK\_32M\_SC | I | 1 | CLK\_32M after scan mux |  |
| resetb\_CLK | I | 1 | Asynchronous reset signal(synchronously released) |  |
| rstb\_32M\_ok\_and\_sr | I | 1 | CLK\_32M\_OK low or soft reset |  |
| SOFT\_RSTB\_32M | I | 1 | Soft reset |  |
| rst\_spi | I | 1 | When SPI\_EN, reset spi | 4 CLK\_32M |
| CLK\_REG\_SC | I | 1 | Scan-mux result of 8MHz clock from CLK\_32M | 8MHz |
| SLEEP\_MODE | I | 1 | Synchronous result of A2D\_SLEEP\_1P8 by CLK\_256K\_SC | Level |
| A2D\_COM\_PLUS\_S | I | 1 | Positive input comparator in S port | async |
| A2D\_COM\_MINUS\_S | I | 1 | Negative input comparator in S port | async |
| A2D\_COM\_PLUS\_N | I | 1 | Positive input comparator in N port | async |
| A2D\_COM\_MINUS\_N | I | 1 | Negative input comparator in N port | async |
| dev\_addr\_dlv | I | 1 | Device address identify delivery | Level(8M domain) |
| dev\_addr\_dlv\_spi | I | 1 | Device address identify delivery when SPI\_EN high | Level(8M domain) |
| tx\_data | I | 9 | Data to be transmit | CLK\_REG domain |
| state\_tx\_init | I | 1 | tx\_state is STATE\_INIT | 1 CLK\_REG |
| state\_tx\_pec | I | 1 | tx\_state is STATE\_PEC | 1 CLK\_REG |
| state\_rx\_init | I | 1 | state is STATE\_INIT | 1 CLK\_REG |
| state\_rx\_bps | I | 1 | state is STATE\_BYPASS | 1 CLK\_REG |
| response | I | 1 | Current device response | Level(8M domain) |
| pos\_response | I | 1 | Positive edge of response | 1 CLK\_REG |
| neg\_response | I | 1 | Negative edge of response | 1 CLK\_REG |
| pos\_next\_rps | I | 1 | Current device is the next to response | 1 CLK\_32M |
| bypass\_end | I | 1 | Mark the ending time of a bypass byte | 1 CLK\_REG |
| rx\_dev\_addr | I | 1 | Receive 9’h1C0 when state is STATE\_INT or STATE\_BYPASS | 4 CLK\_32M |
| cnt\_rx\_byte\_num | I | 8 | Rx byte numer | Level(8M domain) |
| rd | I | 1 | Current device in read station | Level(8M domain) |
| D2A\_TOP\_DEV | I | 1 | Current device is fastest from bridge | Level(8M domain) |
| stack | I | 1 | Stack operation | Level(8M domain) |
| COMN\_TX\_DIS | I | 1 | N port transmit disable | Level(8M domain) |
| COMS\_TX\_DIS | I | 1 | S port transmit disable | Level(8M domain) |
| wait\_re\_clocking | I | 14 | Wait time before transmitting | CLK\_REG domain |
| tx\_start | I | 1 | Transmitting start | 1 CLK\_REG |
| adr\_idty\_done | I | 1 | Address identify done | Level(8M domain) |
| tail\_blanking | I | 1 | Tail blanking time | Level(8M domain) |
| SPI\_EN | I | 1 | SPI enable | async |
| SPI\_DIR | I | 1 | Direction configured by i2c\_master | Level(8M domain) |
| SPI\_RX\_EN | I | 1 | A byte is received by SPI interface | 4 CLK\_32M |
| clr\_crc\_spi | I | 1 | At SPI\_CSB negedge, clr\_crc\_spi generate one pulse to set CRC result to default FFFF when SPI\_EN high. | 4 CLK\_32M |
| RESP | I | 1 | Response by bridge device | Level(8M domain) |
| spi\_rx\_pro | I | 1 | Spi rx process | 4 CLK\_32M |
| neg\_rx\_en | I | 1 | Negedge of rx\_en | 1 CLK\_REG |
| next\_rps | I | 1 | Current device is the next to response | Level(8M domain) |
| rx\_data | I | 9 | Received data | Level(32M domain) |
| TONE\_TRANS\_EN\_N | I | 1 | N port tone transmission enable | Level(CLK\_OUT domain) |
| TONE\_TRANS\_EN\_S | I | 1 | S port tone transmission enable | Level(CLK\_OUT domain) |
| rx\_en\_256K | I | 1 | Daisy chain or spi rx\_en | Level(CLK\_256K domain) |
| neg\_tx\_init | I | 1 | Pulse after tx\_state jump to STATE\_INIT from STATE\_PEC | 1 CLK\_REG |
| STACK\_RESPONSE | I | 6 | Internal time between response bytes | Level(8M domain) |
| FRAME\_DONE | I | 9 | Received frame done | 1 CLK\_REG |
| FR\_CRC\_FLT | I | 1 | Frame CRC fault | 1 CLK\_REG |
| SCAN\_MODE | I | 1 | Scan mode | level |

#### Clock Domain

The clock for DS\_BASIC is CLK\_32M\_SC.

### DS\_BASIC function description

#### 1 D2A\_RX\_EN\_X and TX\_EN\_X

Requirements for D2A\_RX\_EN\_X and TX\_EN\_X by analog part:



Figure2 D2A\_RX\_EN\_X and TX\_EN\_X timing requirement by analog part

1.1 D2A\_RX\_EN\_x:

D2A\_RX\_EN\_S and D2A\_RX\_EN\_N are both initial high(HWR005\_DS\_BASIC). When D2A\_RX\_EN\_S high, RX in S port is enabled. When D2A\_RX\_EN\_N is high, RX in N port is enabled.

Besides daisy chain receive control, D2A\_RX\_EN\_x also enable SPI receiving. When SPI\_EN high, BM20A is used as bridge. If SPI\_DIR is high, SPI replace the N port, so SPI\_MOSI can be received when D2A\_RX\_EN\_N high; If SPI\_DIR is low, SPI replace the S port, so SPI\_MOSI can be received when D2A\_RX\_EN\_S high. This feature is delivered by COPY\_NXT, which is an output from COMM\_CTRL, and is described in IP\_SPEC\_COMM\_CTRL.docx in detail. (HWR006\_DS\_BASIC, HWR010\_DS\_BASIC)



Figure3 D2A\_RX\_EN\_X and D2A\_TX\_EN\_X in normal copy case

1.1.1 D2A\_RX\_EN\_N:

When SPI\_EN high and SPI\_CLR\_DET high, or when SOFT\_RSTB low, D2A\_RX\_EN\_N reset to 1.

When no more data is being transmitting, TX\_timeout is generated, D2A\_RX\_EN\_N reset to 1.

When data comes from S port(A2D\_COM\_PLUS\_S and A2D\_COM\_MINUS\_S), D2A\_RX\_EN\_N shall change to 0(HWR007\_DS\_BASIC). When bit receiving timeout, D2A\_RX\_EN\_N shall change to 1.

When SPI replacing S port starts to receive data, which means ①SPI replace S port (SPI\_EN and !SPI\_DIR), ②SPI start to receive(clr\_crc\_spi) and ③not in response procedure(RESP low), D2A\_RX\_EN\_N shall change to 0.

When BM20A is responsing data in S port, which means ①posedge of response ②D2A\_TOP\_DEV or response to single read command(!stack), and ③TX\_EN\_S high, D2A\_RX\_EN\_N shall change to 0. (HWR005\_DS\_BASIC)

When 500ns after transmitting ends(T2R high), D2A\_RX\_EN\_N changes to !tx\_n. (tx\_n means response on N port).

When received bit length overtime, D2A\_RX\_EN\_N reset to 1.

When sending TONE via N port(TONE\_TRANS\_EN\_N\_32M\_sync[1] high), D2A\_RX\_EN\_N change to 0(HWR005\_DS\_BASIC); When sending TONE via N port ends((|TONE\_TRANS\_EN\_N\_32M\_sync[4:0]) high and TONE\_TRANS\_EN\_N\_32M\_sync[1] low), D2A\_RX\_EN\_N change to !TONE\_TRANS\_EN\_N\_32M\_sync[3] & (state\_rx\_init | state\_rx\_bps).

1.1.2 D2A\_RX\_EN\_S:

When SPI\_EN high and SPI\_CLR\_DET high, or when SOFT\_RSTB low, D2A\_RX\_EN\_S reset to 1.

When no more data is being transmitting, TX\_timeout is generated, D2A\_RX\_EN\_S reset to 1.

When data comes from N port(A2D\_COM\_PLUS\_N and A2D\_COM\_MINUS\_N), D2A\_RX\_EN\_S shall change to 0(HWR007\_DS\_BASIC). When bit receiving timeout, D2A\_RX\_EN\_S shall change to 1.

When SPI replacing N port starts to receive data, which means ①SPI replace N port (SPI\_EN and SPI\_DIR), ②SPI start to receive(clr\_crc\_spi, high pulse when SPI\_CSB negedge) and ③not in response procedure(RESP low), D2A\_RX\_EN\_S shall change to 0.

When responsing data in N port, which means ①posedge of response ②D2A\_TOP\_DEV or response to single read command(!stack), and ③TX\_EN\_N high, D2A\_RX\_EN\_S shall change to 0. (HWR005\_DS\_BASIC)

When 500us after transmitting ends (T2R high), D2A\_RX\_EN\_S changes to !tx\_s. (tx\_s means response on S port).

When received bit length overtime, D2A\_RX\_EN\_S reset to 1.

When sending TONE via S port (TONE\_TRANS\_EN\_S\_32M\_sync[1] high), D2A\_RX\_EN\_S change to 0(HWR005\_DS\_BASIC); When sending TONE via S port ends ((|TONE\_TRANS\_EN\_S\_32M\_sync[4:0]) high and TONE\_TRANS\_EN\_S\_32M\_sync[1] low), D2A\_RX\_EN\_S change to !TONE\_TRANS\_EN\_S\_32M\_sync[3] & (state\_rx\_init | state\_rx\_bps).

1.2 TX\_EN\_x:

TX\_EN\_S and TX\_EN\_N are both initial low. (HWR005\_DS\_BASIC)

1.2.1 TX\_EN\_N:

When SPI replacing N port(SPI\_EN & SPI\_DIR) and when response to spi\_master(RESP) and receiving data from last device(D2A\_RX\_EN\_S), D2A\_TX\_EN\_N is 0. (HWR001\_BASIC\_CTRL)

When SPI\_EN high and SPI\_CLR\_DET high, or when SOFT\_RSTB low, TX\_EN\_N reset to 0.

When no more data is being transmitting, TX\_timeout is generated, TX\_EN\_N reset to 0.

When receiving data from N port and cnt\_bit>=5, TX\_EN\_N is 0.

For coping to opposite port(HWR002\_DS\_BASIC):

When data comes from S port (A2D\_COM\_PLUS\_S and A2D\_COM\_MINUS\_S), after Tr2t, TX\_EN\_N shall change to 1(can be disabled by COMN\_TX\_DIS) (HWR009\_DS\_BASIC). TX\_EN\_N shall change to 0 when coping ends. Tr2t is half byte time in normal copies; Tr2t is 75us after 1st address identify frame byte is received.

When bypass\_end, TX\_EN\_N is 0.

For response:

If Address Identify Command comes from N port, after the frame has been copied to opposite S port, Response shall be in “1st device, 2nd device … top device” order. TX\_EN\_N shall be high(can be disabled by COMN\_TX\_DIS) after Treponse (HWR009\_DS\_BASIC). TX\_EN\_N shall change to 0 when response ends.

If Stack Read Command comes from N port, after the frame has been copied to opposite S port, Response shall be in “top device, (top-1)st device … 1st device” order. All response frames by other devices shall be copied via N port. After coping, TX\_EN\_N shall be high(can be disabled by COMN\_TX\_DIS) after Treponse (HWR009\_DS\_BASIC). TX\_EN\_N shall change to 0 when response ends.

If Single Read Command comes from N port, after the frame has been copied to opposite S port, if current device is chosen to response, after coping, TX\_EN\_N shall be high(can be disabled by COMN\_TX\_DIS) after Treponse (HWR009\_DS\_BASIC). TX\_EN\_N shall change to 0 when response ends. If current device is not chosen to response, TX\_EN\_N follow copy logic.

For Address identify command, Tresponse shall be typically 24us. For other commands, Tresponse shall be typically 30us.

1.2.2 TX\_EN\_S :

When SPI replacing S port(SPI\_EN & !SPI\_DIR) and when response to spi\_master(RESP) and receiving data from last device(D2A\_RX\_EN\_S), D2A\_TX\_EN\_S is 0. (HWR001\_BASIC\_CTRL)

When SPI\_EN high and SPI\_CLR\_DET high, or when SOFT\_RSTB low, TX\_EN\_S reset to 0.

When no more data is being transmitting, TX\_timeout is generated, TX\_EN\_S reset to 0.

When receiving data from S port and cnt\_bit>=5, TX\_EN\_S is 0.

For coping to opposite port(HWR002\_DS\_BASIC):

When data comes from N port (A2D\_COM\_PLUS\_N and A2D\_COM\_MINUS\_N), after Tr2t, TX\_EN\_S shall change to 1(can be disabled by COMS\_TX\_DIS) (HWR009\_DS\_BASIC). TX\_EN\_S shall change to 0 when coping ends. Tr2t is half byte time in normal copies; Tr2t is 75us after 1st address identify frame byte is received.

When bypass\_end, TX\_EN\_S is 0.

If Address identify Command comes from S port, after the frame has been copied to opposite N port, Response shall be in “1st device, 2nd device … top device” order. TX\_EN\_S shall be high(can be disabled by COMS\_TX\_DIS) after Treponse (HWR009\_DS\_BASIC). TX\_EN\_S shall change to 0 when response ends.

If Stack Read Command comes from S port, after the frame has been copied to opposite N port, Response shall be in “top device, (top-1)st device … 1st device” order. All response frames by other devices shall be copied via S port. After coping, TX\_EN\_S shall be high(can be disabled by COMS\_TX\_DIS) after Treponse (HWR009\_DS\_BASIC). TX\_EN\_S shall change to 0 when response ends.

If Single Read Command comes from S port, after the frame has been copied to opposite N port, if current device is chosen to response, after coping, TX\_EN\_S shall be high(can be disabled by COMS\_TX\_DIS) after Treponse (HWR009\_DS\_BASIC). TX\_EN\_S shall change to 0 when response ends. If current device is not chosen to response, TX\_EN\_S follow copy logic.

For Address identify command, Tresponse shall be typically 24us. For other commands, Tresponse shall be typically 30us.

 Figure4 D2A\_RX\_EN\_X and D2A\_TX\_EN\_X in address identify case

~~~~

Figure5 D2A\_RX\_EN\_X and D2A\_TX\_EN\_X in stack read case



Figure6 D2A\_RX\_EN\_X and D2A\_TX\_EN\_X in single read case

#### 2 Receiving

Ideally, daisy chain signal is shaped like this:



Figure3 ideal daisy chain signal

In fact, daisy chain signal may shaped like the following figure, the middle of each bit is shown in blue dummy lines, it is named “middle line” in the following description.



Figure4 real daisy chain signal

Capture received data(HWR001\_DS\_BASIC, HWR006\_COMM\_CTRL):

* 1. Filter input daisy chain signals:

Input filter capture A2D\_COM\_PLUS\_X and A2D\_COM\_MINUS\_X every clock(in this module, clock is CLK\_32M\_SC). Only when 2 success values are the same, the value is regarded as real, and will go on to next operations.



Figure5 input filter

2.2 Bit\_length[4:0]

At beginning, bit\_length[4:0] shall be updated every byte. As the preamble half bit may disappear, it shall not be calculated.



Figure6 bit\_length generation

Bit\_length[4:0] is the average value of cnt\_1bit\_pos[4:0] and cnt\_1bit\_neg[4:0].

* 1. cnt\_pre\_edge[4:0], cnt\_gap\_edge[3:0], gap\_edge\_updated and gap\_edge[3:0]

These 4 signals are generated to mark the middle line of each bit. (HWR003\_DS\_BASIC)

Cnt\_pre\_edge[4:0] counts the interval time between falling edges of p\_dly\_x[3] or m\_dly\_x[3] at middle line time range. Middle line time range is defined as when cnt\_pre\_edge equals to BIT\_LENGTH\*(0.75, 1.25). cnt\_pre\_edge[4:0] start counting at the 2nd falling edge of m\_dly\_x[3].



Figure7 cnt\_pre\_edge start to count

Cnt\_gap\_edge[3:0] counts the gap time between 2nd half and 1st half of each bit. Cnt\_gap\_edge[3:0] starts counting at the 1st time cnt\_pre\_edge[4:0] clears.

Gap\_edge\_updated is defined to ensure gap\_edge update only once per bit. Gap\_edge\_updated is high when cnt\_gap\_edge[3:0] reset(2nd half of each bit comes), and is low after bit grap time.

Gap\_edge[3:0] keep the maximum cnt\_gap\_edge[3:0] until next time cnt\_pad\_edge[3:0] reset to 0.



Figure8 cnt\_pre\_edge, cnt\_gap\_edge, gap\_edge\_updated and gap\_edge

2.4 buf\_h\_lst\_half[7:0], buf\_h\_2nd\_half[7:0], buf\_l\_lst\_half[7:0], buf\_l\_2nd\_half[7:0]



Figure9.1 buf\_p[15:0] and buf\_m[15:0](first half bit)



Figure9.2 buf\_p[15:0] and buf\_m[15:0](second half bit)

These 4 signals are generated to vote for the received bit(HWR004\_DS\_BASIC):

2.4.1 buf\_p[15:0] and buf\_m[15:0]

During a byte receiving time(rx\_en\_s\_dsy or rx\_en\_n\_dsy), buf\_p[15:0] and buf\_m[15:0] are updated every CLK\_32M\_SC.

In detail, when rx\_en\_s\_dsy, buf\_p[15:0] is updated every CLK\_32M\_SC by p\_dly\_s[3],

buf\_m[15:0] is updated every CLK\_32M\_SC by m\_dly\_s[3];

when rx\_en\_n\_dsy, buf\_p[15:0] is updated every CLK\_32M\_SC by p\_dly\_n[3],

buf\_m[15:0] is updated every CLK\_32M\_SC by m\_dly\_n[3].

2.4.2 buf\_diff[15:0]

buf\_diff[15:0] is the xor-ed result of buf\_p[15:0] and buf\_m[15:0]. Only complementary values can be used.

2.4.3 buf\_h\_lst\_half[7:0] and buf\_l\_lst\_half[7:0]

As the first half bit is distinguished (cnt\_gap\_edge[3:0]/2) clocks later than real middle bit time, buf\_h\_lst\_half[7:0] is the higher cnt\_gap\_edge[3:1] bits of complementary value of buf\_p[15:0]; buf\_l\_lst\_half[7:0] is the higher cnt\_gap\_edge[3:1] bits of complementary value of ~buf\_p[15:0];

2.4.5 buf\_h\_2nd\_half[7:0] and buf\_l\_2nd\_half[7:0]

Second half bit is distinguished ((BIT\_LENGTH[4:0]-GAP\_EDGE[3:0])/2) clocks later than the beginning time of 2nd half bit.

buf\_h\_2nd\_half[7:0] is complementary value of buf\_p[7:0], and buf\_l\_2nd\_half[7:0] is complementary value of ~buf\_p[7:0].

2.5 Cnt\_hh[3:0] and cnt\_ll[3:0]

Cnt\_hh[3:0] count the A2D\_COM\_PLUS\_x high and A2D\_COM\_MINUS\_x low time during the 1st half bit time, and count the A2D\_COM\_MINUS\_x high and A2D\_COM\_PLUS\_x low time during the 2nd half bit time.

Cnt\_ll[3:0] count the A2D\_COM\_MINUS\_x high and A2D\_COM\_PLUS\_x low time during the 1st half bit time, and count the A2D\_COM\_PLUS\_x high and A2D\_COM\_MINUS\_x low time during the 2nd half bit time.

* 1. Grap\_moment[2:0]

Grap\_moment[2:0] is defined to mark the time to grap bit value. Grap\_moment[0] is high when gap of cnt\_pre\_edge and gap\_edge equals to (bit\_length – gap\_edge)/2. High pulse is delivered to grap\_moment[1], grap\_moment[2] one by one.

Grap\_moment[1] is the real grap time for p\_bit. So it’s also assigned as grap\_pulse.

* 1. P\_bit

P\_bit is the received bit data.

When grap\_moment[1] is high, if cnt\_hh[3:0] > cnt\_ll[3:0], p\_bit is high;

if cnt\_h[3:0] <= cnt\_l[3:0], p\_bit is low.

* 1. Cnt\_bit[4:0]

Cnt\_bit[4:0] counts the number of bits in each byte.

* 1. Rev\_dsy\_data[8:0]

When (cnt\_bit[4:0] == 5’h3) and grap\_pulse, p\_bit means sof bit. Except for sof bit, following bits are in LSB-first order. Rev\_dsy\_data[8] is sof bit, rev\_dsy\_data[7:0] are MSB-first received data.

* 1. Rx\_en\_x\_dsy\_m, rx\_en\_x\_dsy, rx\_en\_x, rx\_en and neg\_rx\_en\_dsy

Rx\_en\_x\_dsy\_m and rx\_en\_x\_dsy are defined to differ daisy chain signals from tone.

Rx\_en\_x\_dsy\_m is high only between the 1st and 2nd rising edge of m\_dly\_x[3]. If no 2nd rising edge is detected, rx\_en\_x\_dsy\_m will clear to 0 when timeout (no new m\_dly\_x[3]’s rising edge is detected for 63 CLK\_32M clocks).

Rx\_en\_x\_dsy is high when p\_dly\_x[3]’s rising edge when rx\_en\_x\_dsy\_m high. Normally, rx\_en\_x\_dsy is low when a byte is completely received. If tone signals are received, rx\_en\_x\_dsy will clear to 0 when timeout(no new p\_dly\_x[3]’s posedge is detected for 32 CLK\_32M clocks).

Rx\_en\_x considers the SPI ports and rx\_en\_x\_dsy.

If SPI replace S port(SPI\_EN & !SPI\_DIR), rx\_en\_s is 0, else rx\_en\_s is rx\_en\_s\_dsy.

If SPI replace N port(SPI\_EN & SPI\_DIR), rx\_en\_n is 0, else rx\_en\_n is rx\_en\_n\_dsy.

Rx\_en is the or-ed result of rx\_en\_s\_dsy and rx\_en\_n\_dsy. All data are only captured when rx\_en high(HWR012\_DS\_BASIC).

Neg\_rx\_en\_dsy is the negedge of rx\_en.



Figure10.1 rx\_en\_x\_dsy\_m, rx\_en\_x\_dsy, rx\_en\_x and rx\_en

 

Figure10.2 positive tone disturb Figure10.3 negative tone disturb

* 1. Tail\_blanking and idle\_clear(HWR011\_DS\_BASIC)

Tail\_blanking is a mask signal to avoid receiving. It is defined to ignore tail from analog part. After a byte is completely received, within (500+8\*STACK\_RESPONSE\*TCLK\_32M)ns, no edges of p\_dly\_x[3] and m\_dly\_x[3] shall be used for receiving.

Idle\_clear is a clear signal for cnt\_m\_dly\_neg[1:0], which counts the number of m\_dly\_x[3]’s falling edge. When cnt\_m\_dly\_neg[1:0] gets clear, all information about the last byte is totally removed. Idle\_clear is high if p\_dly\_x[3] and m\_dly\_x[3] and tail\_blanking are all 0 for 500ns. Idle\_clear is low when p\_dly\_x[3] or m\_dly\_x[3] is high.

****

Figure11 tail\_blanking and idle\_clear

#### 3 Transmitting

After tx\_start(from instance u\_COMM\_CTRL) high, u\_DS\_BASIC starts to transmit data.

* 1. Send\_start and tx\_en[1:0]

Rising edge of tx\_start lead to signals send\_start and tx\_en[1:0].

* 1. Cnt\_tx\_bit[4:0], cnt\_tx\_bit\_num[3:0] and tx\_character\_ends(HWR006\_COMM\_CTRL)

When transmitting starts, tx\_en[0] high. cnt\_tx\_bit[4:0] starts to count the main clock number per bit. Cnt\_tx\_bit\_num[3:0] starts to count the number of bits per byte.

When (cnt\_tx\_bit\_num[3:0]>=4’hD), a byte is completely transmitted, tx\_character\_ends is high

* 1. Tx\_one(HWR006\_COMM\_CTRL)

Tx\_one captures data one bit by one bit(according to cnt\_tx\_bit[4:0] and cnt\_tx\_bit\_num[3:0]) from tx\_data[8:0](from instance u\_COMM\_CTRL). Tx\_data[8] is the first bit, others are in LSB-first order. When in 1st half bit(cnt\_tx\_bit[4:0]<8), tx\_one is the inverted value of last bit; when in 2nd half bit, tx\_one is the value of current bit.

* 1. WR\_COM\_PLUS and WR\_COM\_MINUS(HWR006\_COMM\_CTRL)

WR\_COM\_PLUS and WR\_COM\_MINUS influence the final output D2A\_WR\_COM\_PLUS and D2A\_ WR\_COM\_MINUS. WR\_COM\_PLUS and WR\_COM\_MINUS are complementary signals which output tx\_one bit by bit.

* 1. Tx\_crc[15:0]

Tx\_crc[15:0] is result of CRC16 algorithm with polynomial 8005(X16+X15+X2+1), initial CRC 16’hFFFF, using tx\_one as input bit stream.