Y145 (Z80-SIO)

Core Specification

Disclaimer

Every effort has been made to ensure the accuracy of the information contain herein. If you find errors or inconsistencies please bring them to our attention. In all cases, however, the Verilog HDL source code for the Y145 design defines “proper operation”.

Table of Contents

This page intentionally left blank.

Overview

The Y145 core is an exact copy of the Zilog Z80-SIO, having been created using the orginal Zilog schematics under contract for Zilog. That contract grants Systemyde the right to license the design. Because this core is an exact copy of the Zilog design, it is best to refer to the original Zilog documentation for the Z80-SIO for details about the operation of the device.

General Features:

- Technology-independent Verilog HDL implementation.

- 8-bit CPU interface.

- Two independent full-duplex channels.

- Receivers are quadruply buffered; transnmitters are doubly-buffered.

- Z80-CPU interrupt structure.

- Optional 32-bit CRC.

- Optional direct Interrupt Acknowledge and Return-from-Interrupt inputs.

This page intentionally left blank.

Signal Descriptions

As a direct copy of the Z80-SIO, the Y145 top-level interface includes bidirectional and open-drain signals. Since these types of signals are not appropriate for use in an SOC environment, the design is structured as a core (called **sio_mega.v**) instantiated in the top-level pad ring (called **sio_top.v**) where the bidirectional and open-drain signals are created. In a similar fashion, the data bus is made bidirectional at the **sio_mega.v** level of the hierarchy.

The original Z80-SIO is available in three different pinouts, that were created by different bonding options using a single die. The Y145 implements the logic of the original die, so if your design requires a specific pinout, refer to the Zilog documentation for how the base signals are connected to create the different package options.

Bidirectional and open-drain signals:

DBUS (bidirectional, 3-state). The *System Data Bus* signals carry data and status to and from the device.

INT_ (output, open-drain, active-Low). The *Interrupt Request* output is driven Low to request an interrupt.

W_RDYA_, W_RDYB_ (outputs, open-drain or driven). The *Wait/Ready A* and *Wait/Ready B* outputs can be programmed to operate as either open-drain Wait signals for the CPU or as Ready signals for a DMA controller. The reset state is open-drain.

SYNCA_, SYNCB_ (inputs/outputs, active-Low). The *Sync A* and *Sync B* signals are either inputs (in Async mode or Extrernal Sync mode) or outputs (in the synchronous modes.)

Core signals:

B_A_ (input, High selects Channel B). The *Channel A or Channel B Select* signal controls which channel is accessed duirng a bus transfer.

C_D_ (input, High selects Control). The *Control or Data Select* signal controls whether the bus transfer is for control information or data.

CE_ (input, active-Low). The *Chip Enable* signal enables bus transfers.

CLK (input). The *System Clock* signal is the master clock for the device.

CTSA_, CTSB_ (inputs, active-Low). The *Clear To Send A* and *Clear To Send B* signals can be used to enable their respective transmitters. Interrupts are generated on both transitions of these signals.

DCDA_, DCDB_ (inputs, active-Low). The *Data Carrier Detect A* and *Data Carrier Detect B* signals can be used to enable their respective receivers. Interrupts are generated on both transitions of these signals.

DTRA_, DTRB_ (outputs, active-Low). The *Data Terminal Ready A* and *Data Terminal Ready B* signals reflect the state of the corresponding control bits.

IEI (input, active-High). The *Interrupt Enable In* signal is part of the standard Z80 interrupt daisy-chain.

IEO (output, active-High). The *Interrupt Enable Out* signal is part of the standard Z80 interrupt daisy-chain.

INT_OUT_ (output, active-Low). The *Interrupt Requet Output* signal is the driven version of the interrupt request.

IORQ_ (input, active-Low). The *Input/Output Request* signal is used, with other bus control signals, to identify the type of bus transaction in progress.

M1_ (input, active-Low). The *Machine Cycle One* signal is used, with other bus control signals, to identify the type of bus transaction in progress.

RD_ (input, active-Low). The *Read Cycle Status* signal is used, with other bus control signals, to identify the type of bus transaction in progress.

RESET_ (input, active-Low). The *Reset* signal initializes the core, forcing everything to the inactive state.

RTSA_, RTSB_ (outputs, active-Low). The *Request To Send A* and *Request To Send B* signals reflect the state of the corresponding control bits, with some additional functionality in the Async mode.

RXCA_, RXCB_ (inputs). The *Receive Clock A* and *Receive Clock B* signals sample receive data and clock the respective receivers.

RXDA_, RXDB_ (inputs, active-High). The *Receive Data A* and *Receive Data B* signals are the inputs to the respective receivers.

SYNCA_IN_, SYNCB_IN_ (inputs, active-Low). The *Synchronization Input A* and *Synchronization Input B* signals are the input path for the original Z80-SIO SYNCA_ and SYNCB_ signals.

SYNCA_OUT_, SYNCB_OUT_ (outputs, active-Low). The *Synchronization Output A* and *Synchronization Output B* signals are the output path for the original Z80-SIO SYNCA_ and SYNCB_ signals.

SYNCA_TRI_, SYNCB_TRI_ (outputs, active-High). The *Synchronization Enable A* and *Synchronization Enable B* signals are the direction control signals for the original Z80-SIO SYNCA_ and SYNCB_ signals. High enables output.

TXCA_, TXCB_ (inputs). The *Transmit Clock A* and *Transmit Clock B* signals clock the transmit data and clock the respective transmitters.

TXDA_, TXDB_ (outputs, active-High). The *Transmit Data A* and *Transmit Data B* signals are the outputs from the respective transmitters.

W_RDYA_OUT_, W_RDYB_OUT_ (outputs, active-Low). The *Wait/Ready Output A* and *Wait/Ready Output B* signals are the output path for the original Z80-SIO W/RDYA_ and W/RDYB_ signals.

W_RDYA_TRI_, W_RDYB_TRI_ (outputs, active-High). The *Wait/Ready Enable A* and *Wait/Ready Enable B* signals are the driver enable signals for the original Z80-SIO W/RDYA_ and W/RDYB_ signals. High enables output.

Optional features signals:

cha_crc32, chb_crc32 (inputs, active High). The *CRC 32-bit Enable A* and *CRC 32-bit Enable B* signals enable the optional 32-bit CRC polynomial in SDLC mode. These signals should be tied Low for normal Z80-SIO operation.

iack (input, active High). The *Interrupt Acknowledge* signal can be used in systems that do not generate the special Interrupt acknowledge bus timing of a Z80-CPU to set the highest-priority Interrupt-Under-Service (IUS) bit at the start of an interrupt service routine. A one clock cycle pulse in this input takes the place of the Z80 Interrupt Acknowledge bus cycle. This input should be tied Low for normal Z80-SIO operation.

rti (input, active High). The *Return from Interrupt* signal can be used in systems that do not fetch the ED-4D opcode of the Z80 RETI instruction to clear the highest-priority IUS bit at the end of an interrupt service routine. A one clock cycle pulse in this input takes the place of the Z80 RETI instruction fetch. This input should be tied Low for normal Z80-SIO operation.

Verilog

Shown below are the **sio_top.v** Verilog connections:

```
module sio_top  (DBUS, SYNCA_, SYNCB_, DTRA_, DTRB_, IEO, INT_, RTSA_, RTSB_, TXDA, TXDB,
                 W_RDYA_, W_RDYB_, B_A_, C_D_, CE_, CLK, CTSA_, CTSB_, DCDA_, DCDB_, IEI,
                 IORQ_, M1_, RD_, RESET_, RXCA_, RXCB_, RXDA, RXDB, TXCA_, TXCB_,
                 cha_crc32, chb_crc32, iack, rti);

  input      B_A_;          /* b channel/a channel                                          */
  input      C_D_;          /* control/data                                                 */
  input      CE_;           /* chip enable                                                  */
  input      CLK;           /* system clock                                                 */
  input      CTSA_;         /* clear to send a                                              */
  input      CTSB_;         /* clear to send b                                              */
  input      DCDA_;         /* data carrier detect a                                        */
  input      DCDB_;         /* data carrier detect b                                        */
  input      IEI;           /* interrupt enable input                                       */
  input      IORQ_;         /* io request                                                   */
  input      M1_;           /* machine cycle one                                            */
  input      RD_;           /* read operation                                               */
  input      RESET_;        /* chip reset                                                   */
  input      RXCA_;         /* receive clock a                                              */
  input      RXCB_;         /* receive clock b                                              */
  input      RXDA;          /* receive data a                                               */
  input      RXDB;          /* receive data b                                               */
  input      TXCA_;         /* transmit clock a                                             */
  input      TXCB_;         /* transmit clock b                                             */
  input      cha_crc32;     /* 32-bit crc enable a                                          */
  input      chb_crc32;     /* 32-bit crc enable b                                          */
  input      iack;          /* force interrupt acknowledge                                  */
  input      rti;           /* force return from interrupt                                  */
  output     DTRA_;         /* data terminal request a                                      */
  output     DTRB_;         /* data terminal request b                                      */
  output     IEO;           /* interrupt enable out                                         */
  output     INT_;          /* interrupt request                                            */
  output     RTSA_;         /* request to send a                                            */
  output     RTSB_;         /* request to send b                                            */
  output     TXDA;          /* transmit data a                                              */
  output     TXDB;          /* transmit data b                                              */
  output     W_RDYA_;       /* wait/ready a                                                 */
  output     W_RDYB_;       /* wait/ready b                                                 */
  inout      SYNCA_;        /* sync a                                                       */
  inout      SYNCB_;        /* sync b                                                       */
  inout [7:0] DBUS;         /* data bus                                                     */
```

Shown below are the **sio_mega.v** Verilog connections:

```
module sio_mega (DBUS, DTRA_, DTRB_, IEO, INT_OUT_, RTSA_, RTSB_, SYNCA_OUT_, SYNCB_OUT_,
                 SYNCA_TRI_, SYNCB_TRI_, TXDA, TXDB, W_RDYA_OUT_, W_RDYB_OUT_,
                 W_RDYA_TRI_, W_RDYB_TRI_, B_A_, C_D_, CE_, CLK, CTSA_, CTSB_, DCDA_,
                 DCDB_, IEI, IORQ_, M1_, RD_, RESET_, RXCA_, RXCB_, RXDA, RXDB, SYNCA_IN_,
                 SYNCB_IN_, TXCA_, TXCB_, cha_crc32, chb_crc32, iack, rti);

  input      B_A_;          /* b channel/a channel                                      */
  input      C_D_;          /* control/data                                             */
  input      CE_;           /* chip enable                                              */
  input      CLK;           /* system clock                                             */
  input      CTSA_;         /* clear to send a                                          */
  input      CTSB_;         /* clear to send b                                          */
  input      DCDA_;         /* data carrier detect a                                    */
  input      DCDB_;         /* data carrier detect b                                    */
  input      IEI;           /* interrupt enable input                                   */
  input      IORQ_;         /* io request                                               */
  input      M1_;           /* machine cycle one                                        */
  input      RD_;           /* read operation                                           */
  input      RESET_;        /* chip reset                                               */
  input      RXCA_;         /* receive clock a                                          */
  input      RXCB_;         /* receive clock b                                          */
  input      RXDA;          /* receive data a                                           */
  input      RXDB;          /* receive data b                                           */
  input      SYNCA_IN_;     /* sync a input                                             */
  input      SYNCB_IN_;     /* sync b input                                             */
  input      TXCA_;         /* transmit clock a                                         */
  input      TXCB_;         /* transmit clock b                                         */
  input      cha_crc32;     /* 32-bit crc enable a                                      */
  input      chb_crc32;     /* 32-bit crc enable b                                      */
  input      iack;          /* force interrupt acknowledge                              */
  input      rti;           /* force return from interrupt                              */
  output     DTRA_;         /* data terminal request a                                  */
  output     DTRB_;         /* data terminal request b                                  */
  output     IEO;           /* interrupt enable out                                     */
  output     INT_OUT_;      /* interrupt request                                        */
  output     RTSA_;         /* request to send a                                        */
  output     RTSB_;         /* request to send b                                        */
  output     SYNCA_OUT_;    /* sync a output                                            */
  output     SYNCB_OUT_;    /* sync b output                                            */
  output     SYNCA_TRI_;    /* sync a tristate enable                                   */
  output     SYNCB_TRI_;    /* sync b tristate enable                                   */
  output     TXDA;          /* transmit data a                                          */
  output     TXDB;          /* transmit data b                                          */
  output     W_RDYA_OUT_;   /* wait/ready a output                                      */
  output     W_RDYB_OUT_;   /* wait/ready b output                                      */
  output     W_RDYA_TRI_;   /* wait/ready a tristate enable                             */
  output     W_RDYB_TRI_;   /* wait/ready b tristate enable                             */
  inout [7:0] DBUS;         /* data bus                                                 */
```