

**Communication System**

The system consists three parts: the Nios II Microprocessor, the transmitting part, and the receiving part. The transmit and receive process are independent from each other. The microprocessor takes 8-bit Parallel Data Bus from the receiving port, and outputs data as 8-bit Parallel Data bus to the transmitting part. The handling of the parallel data is done in the Nios II Microprocessor.

Data comes in from or transfers out to outside connections such as an General Purpose Input/Output(GPIO) port, or bluetooth connections one bit at a time. Our system performs asynchronous serial communication, in which serial data is grouped into frames. These serial data are clocked by two 4-bit counters - *bit sample count (bsc)*, and *bit identification count (bic)*.

**Framing**

A frame starts with a *start bit*, consists 8 bits of data in the middle, and ends with a  *stop bit.* The *start* *bit* enables receiving device to temporarily synchronize with the transmitting device. The *Stop bit* indicates the end of a frame and gets the receiving device ready for the next frame. Both the receiving device and the transmitting device must share the same framing setup to ensure successful temporal synchronization. In our system, the *start bit* will be 0 and the *end bit* will be 1.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 8-bit data | | | | | | | | 1 |

Figure1.1 A frame with start bit and end bit.

**Timing**

Two 4-bit counters are used to clock data in both directions. *Bit Sample Count(bsc)* is triggered when the *start bit* is detected. *bsc* indicates whether we are looking at the start(0000), middle, or end (1111) of a bit. We choose to sample the 1-bit data at the halfway point, in other words, when *bsc* is 1000. *Bit Identification Count (bic)* is triggered when the first bit is received/sent, in other words, when *bsc* is 1000. There are 10 bits in each frame, the *bic* starts from 0000, 0001, 0010, …, and starts again at 1111. When *bic* reaches 1010, a flag will be set up to tell the microprocessor that all 10 bits of data are received/sent.

**Receive Data**

**Start Bit Detect - Transmit Enable**

***input*** - *reset, clock, serial data input, character received.*

***output*** -  *receive start.*

This logic block identifies the start of an input data frame. When it detects a ‘0’ from the serial data input, it will generate a *receive start* signal to the control logic which starts counting the input data bits. It will stop the *receive start* signal once the 10 bits of the frame have been received, or when the whole system is reset. In other cases, the *receive start* signal will remain at its current state.

**Bit Sample Count (bsc) - srClock**

***input*** *- reset, clock, enable.*

***output*** *- source clock.*

***register*** *- (4 bit) bit sample count.*

As noted before, bsc is used to keep track of whether we are looking at the start or end of the incoming bit. *bit sample count* is a 4-bit counter which starts from 0000 and ends at 1111. It will start counting when it receives the *enable* (receive start) flag. The *source clock* will be set to 1 every time when *bit sample count* counts to 1000, and it will be 0 in all other cases.

**Bit Identification Count Receive (bic) - charReceived**

***input*** *- reset, clock, source clock, receive enable.*

***output*** *- character received.*

***register*** *- (4 bit) bit identification count.*

As noted before, bic is used to keep track of which bit it is present at the input frame. It will start counting at the *receive enable* signal. Starting from 0000, the *bit identification count* counter will increment at each *source clock* while *receive enable* is true. The value of the *bic*  counter represents how many bits have been received by the microprocessor. When *bit identification count* counts to 1010 (which is 10 in decimal), the *character received* flag will be turned to 1, it will remain 0 in all other cases.

**Serial to Parallel Buffer and Shift Register**

***input*** *- reset, source cllock, serial data input.*

***output*** *- (8 bit) parallel data.*

***register*** *- (10 bit) buffer.*

This serial to parallel shift register propagates the *serial data input* value to the left at each *source clock*. A 10-bit *buffer* is used to keep track of the *parallel data input*. The *start bit* and *end bit* of the parallel data bus is neglected because they do not contain useful information. The 8 bits in the middle are passed to the microprocessor.

**Nios II Processor**

***input*** *- (8 bit) parallel data bus, character received.*

The microprocessor reads the parallel data bus when the character received flag is set to 1. This 8-bit parallel data bus is the binary Ascii code for one character. The c ‘putchar()’ function can print the data bus out as characters. For example, the ascii code for A is 097, which is 01100001 in binary.

**Transmit Data**

**Nios II Processor**

***input -***  *character sent.*

***output*** *- (8 bit) parallel data bus, load, transmit enable.*

When user types characters in the Nios II console, the c function ‘getchar()’ reads the character and sends it in binary Ascii format. At the moment the frame of bits is sent, the processor also generates a *load* signal, which notifies the buffer to load the new data. Similarly, the *Transmit Enable* signal is sent to the *bsc* block and the  *bic transfer*  block to indicate the start of the transmission.

**Bit Sample Count (bsc) - srClock**

***input*** *- reset, clock, enable.*

***output*** *- source clock.*

***register*** *- (4 bit) bit sample count.*

The bsc module used for transmission is the same as the one used for receive. bsc is used to keep track of whether we are looking at the start or end of the incoming bit. *bit sample count* is a 4-bit counter which starts from 0000 and ends at 1111. It will start counting when it receives the *enable* (receive start) flag. The *source clock* will be set to 1 every time when *bit sample count* counts to 1000, and it will be 0 in all other cases.

**Bit Identification Count Transfer (bic) - charSent**

***input*** *- reset, clock, source clock, transfer enable, load.*

***output*** *- character sent.*

***register*** *- (4 bit) bit identification count.*

bic is used to keep track of which bit it is present at the transmit frame. It will start counting at the positive *transfer enable* signal. Starting from 0000, the *bit identification count* counter will increment at each *source clock* while 1) *transfer enable* is true and 2) the shift register is not loading (negative *load*). The value of the *bic*  counter represents how many bits have been sent by the microprocessor. When *bit identification count* counts to 1010 (which is 10 in decimal), the *character sent* flag will be turned to 1, it will remain 0 in all other cases.

**Parallel to Serial Buffer and Shift Register**

**input** - reset, source clock, load, transfer enable, clock, character sent, [7:0] data.

**output** - serial data output.

**register** - (10 bit) buffer.

When *load* is positive and *character sent*  is false, the 8-bit parallel data is loaded into a 10-bit buffer, with a *start bit* - 0, and an *end bit* - 1. This parallel to serial shift register pushes the parallel data out from the left one bit at each positive source clock edge. If the *character sent* flag is positive, the buffer will be set to its default, which is 10 bits of 1.