IPv4 Receiver Module

Software Test-bench IDD

# Register Summary:

The following registers are addressable by the AXI4 bus from the Zynq PS. Only a subset of the registers are writeable where-as all registers are readable by the Zynq PS. The Zynq PS interface is 32 bit, all registers are 32 bits wide. The address bus is 11 bits wide.

|  |  |  |
| --- | --- | --- |
| **Register Address** | **Register Name** | **Register Mode** |
| 0x000 | CONFIG REGISTER | W/R |
| 0x001 | COUNT SET REGISTER | W/R |
| 0x002 | TEST BENCH IN COUNT REGISTER | R |
| 0x003 | TEST BENCH OUT COUNT REGISTER | R |
| 0x004 | TEST BENCH STATUS REGISTER | R |
| 0x005 – 0x00D | INPUT PACKET HEADER REGISTERS | W/R |
| 0x00D – 0x204 | INPUT PACKET DATA REGISTERS | W/R |
| 0x205 – 0x404 | TEST BENCH OUTPUT | R |
| 0x405 – 0x604 | TEST BENCH OUTPUT VALID | R |

# Register Details:

## CONFIG REGISTER 0x000:

|  |  |  |  |
| --- | --- | --- | --- |
|  | DATA SRC | MODE | START |
| Bit 31 Bit 3 |  |  | Bit 0 |

START – Test-bench start bit. Set this bit to (0) to ensure the test-bench does not run while configuring the rest of the test bench. Set this bit to (1) to execute the test-bench. When the test-bench reaches a state of completion based on the current configuration the START bit must be set to (0) in order to allow for the test-bench to be run again.

MODE – Test-bench mode bit. This bit controls whether the test-bench executes once or multiple times based on the COUNT SET REGISTER. Set this bit to (0) to configure the test-bench to run once ignoring the COUNT SET REGISTER. Set this bit to (1) to configure the test-bench to run multiple times. When set to (1) the number of times the test-bench will complete is controlled by the COUNT SET REGISTER.

DATA SRC – Test-bench packet data source bit. This bit controls where the packet data that is placed into the inputs of the DUT comes from. To run the test-bench with the packet placed in the INPUT PACKET registers as is set this bit to (0). To run the test-bench with redundant payload for length testing set this bit to (1). When set to (1) the payload data transferred to the DUT comes from register 0x00D only, the number of inputs are based on the header registers length. When set to (0) the payload data transferred to the DUT starts at register 0x00D and ends based on the packet length up to register 0x204.

## COUNT SET REGISTER 0x001:

|  |  |
| --- | --- |
|  |  |
| Bit 31 | Bit 0 |

32 bit register for the number of times to run the test bench. Set this register along with the MODE bit of the CONFIG REGISTER in order to have the test-bench execute multiple times in a row without having to toggle the START bit of the CONFIG REGISTER.

## TEST BENCH IN COUNT REGISTER 0x002:

|  |  |
| --- | --- |
|  |  |
| Bit 31 | Bit 0 |

32 bit register for the number of times the test-bench has run the input cycle on the DUT. This register is read only from the Zynq PS. This register should be the same as the TEST BENCH OUT COUNT REGISTER. If there is a difference then there has been an error in the DUT. When executing under multiple execution mode the test-bench stops executing when this register matches the COUNT SET REGISTER.

## TEST BENCH OUT COUNT REGISTER 0x003:

|  |  |
| --- | --- |
|  |  |
| Bit 31 | Bit 0 |

32 bit register for the number of times the DUT has communicated a full output cycle to the test-bench. This register is read only from the Zynq PS. This register should match the TEST BENCH IN COUNT REGISTER. If the registers do not match there has been an error in the DUT such as a double execution or a missed execution.

## TEST BENCH STATUS REGISTER 0x004:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | OUTPUT LENGTH | END | START |
| Bit 31 | 10 2 | |  | Bit 0 |

This register is read only from the Zynq PS.

START – This bit is set to (0) when the DUT is not currently in the process of an output cycle. This bit is set to (1) when the DUT starts an output cycle. The bit is latched until the DUT indicates it is finishing an output cycle.

END – This bit is set to (0) when the DUT has not yet executed an output cycle. This bit is set to (1) when the DUT indicates it is finishing an output cycle. If the DUT is not currently in the process of an output cycle and has previously finished an output cycle this bit is latched to (1).

OUTPUT LENGTH – This 9 bit field holds the number of 32 bit half words from the last DUT output cycle stored in the TEST BENCH OUTPUT registers. This field is reset when the START bit of the CONFIG REGISTER is set to (0), proper timing is to get the output prior to setting the START bit of the CONFIG REGISTER to (0) to initiate another execution of the test-bench.

## INPUT PACKET HEADER REGISTERS 0x005 – 0x00D:

## 0x005:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| VERSION | IHL | DSCP | ECN | TOTAL LENGTH |
| Bit 31 | 27 | 23 | 17 | 15 Bit 0 |

TOTAL LENGTH - This 16-bit field defines the entire packet size in bytes, including header and data. The minimum size is 20 bytes (header without data) and the maximum is 65,535 bytes. All hosts are required to be able to reassemble datagrams of size up to 576 bytes, but most modern hosts handle much larger packets. Sometimes links impose further restrictions on the packet size, in which case datagrams must be fragmented. Fragmentation in IPv4 is handled in either the host or in routers.

ECN - Explicit congestion notification: This field is defined in RFC 3168 and allows end-to-end notification of network congestion without dropping packets. ECN is an optional feature that is only used when both endpoints support it and are willing to use it. It is only effective when supported by the underlying network.

DSCP - Differentiated services code point: Originally defined as the Type of service (ToS) field. This field is now defined by RFC 2474 (updated by RFC 3168 and RFC 3260) for Differentiated services (DiffServ). New technologies are emerging that require real-time data streaming and therefore make use of the DSCP field. An example is Voice over IP (VoIP), which is used for interactive data voice exchange.

IHL - Internet header length: The Internet Header Length (IHL) field has 4 bits, which is the number of 32-bit words. Since an IPv4 header may contain a variable number of options, this field specifies the size of the header (this also coincides with the offset to the data). The minimum value for this field is 5, which indicates a length of 5 × 32 bits = 160 bits = 20 bytes. As a 4-bit field, the maximum value is 15 words (15 × 32 bits, or 480 bits = 60 bytes).

VERSION - The first header field in an IP packet is the four-bit version field. For IPv4, this is always equal to 4.

## 0x006:

|  |  |  |
| --- | --- | --- |
| IDENTIFICATION | FLAGS | FRAGMENT OFFSET |
| 31 | 15 | 12 Bit 0 |

FRAGMENT OFFSET - The fragment offset field is measured in units of eight-byte blocks. It is 13 bits long and specifies the offset of a particular fragment relative to the beginning of the original unfragmented IP datagram. The first fragment has an offset of zero. This allows a maximum offset of (213 – 1) × 8 = 65,528 bytes, which would exceed the maximum IP packet length of 65,535 bytes with the header length included (65,528 + 20 = 65,548 bytes).

FLAGS - A three-bit field follows and is used to control or identify fragments. They are (in order, from most significant to least significant):

bit 0: Reserved; must be zero.

bit 1: Don't Fragment (DF)

bit 2: More Fragments (MF)

If the DF flag is set, and fragmentation is required to route the packet, then the packet is dropped. This can be used when sending packets to a host that does not have sufficient resources to handle fragmentation. It can also be used for Path MTU Discovery, either automatically by the host IP software, or manually using diagnostic tools such as ping or traceroute. For unfragmented packets, the MF flag is cleared. For fragmented packets, all fragments except the last have the MF flag set. The last fragment has a non-zero Fragment Offset field, differentiating it from an unfragmented packet.

IDENTIFICATION - This field is an identification field and is primarily used for uniquely identifying the group of fragments of a single IP datagram. Some experimental work has suggested using the ID field for other purposes, such as for adding packet-tracing information to help trace datagrams with spoofed source addresses, but RFC 6864 now prohibits any such use.

## 0x007:

|  |  |  |
| --- | --- | --- |
| TIME TO LIVE | PROTOCOL | HEADER CHECKSUM |
| Bit 31 | 23 | 15 Bit 0 |

HEADER CHECKSUM - The 16-bit checksum field is used for error-checking of the header. When a packet arrives at a router, the router calculates the checksum of the header and compares it to the checksum field. If the values do not match, the router discards the packet. Errors in the data field must be handled by the encapsulated protocol. Both UDP and TCP have checksum fields.

When a packet arrives at a router, the router decreases the TTL field. Consequently, the router must calculate a new checksum. RFC 791 defines the checksum calculation:

The checksum field is the 16-bit one's complement of the one's complement sum of all 16-bit words in the header. For purposes of computing the checksum, the value of the checksum field is zero.

PROTOCOL - This field defines the protocol used in the data portion of the IP datagram. The Internet Assigned Numbers Authority maintains a list of IP protocol numbers which was originally defined in RFC 790.

TIME TO LIVE - An eight-bit time to live field helps prevent datagrams from persisting (e.g. going in circles) on an internet. This field limits a datagram's lifetime. It is specified in seconds, but time intervals less than 1 second are rounded up to 1. In practice, the field has become a hop count—when the datagram arrives at a router, the router decrements the TTL field by one. When the TTL field hits zero, the router discards the packet and typically sends an ICMP Time Exceeded message to the sender. The program traceroute uses these ICMP Time Exceeded messages to print the routers used by packets to go from the source to the destination.

## 0x008:

|  |  |
| --- | --- |
|  |  |
| Bit 31 | Bit 0 |

Source address: This field is the IPv4 address of the sender of the packet. Note that this address may be changed in transit by a network address translation device.

## 0x009:

|  |  |
| --- | --- |
|  |  |
| Bit 31 | Bit 0 |

Destination address: This field is the IPv4 address of the receiver of the packet. As with the source address, this may be changed in transit by a network address translation device.

## 0x00A - 0x00D:

|  |  |
| --- | --- |
|  |  |
| Bit 31 | Bit 0 |

Options fields defined by RFC 791

## INPUT PACKET DATA REGISTERS 0x00E – 0x204:

|  |  |
| --- | --- |
|  |  |
| Bit 31 | Bit 0 |

Payload registers for the IPv4 packet. These registers are read and write from the Zynq PS. These registers are 32 bits wide, and represent half words for the DUT which uses 64 bit words. When not in redundant source mode these registers are read into the DUT in sequence according to the packet length established in the header subtracting the header length.

## TEST-BENCH OUTPUT 0x205 – 0x404:

|  |  |
| --- | --- |
|  |  |
| Bit 31 | Bit 0 |

These 32 bit wide registers are read only from the Zynq PS. These registers store the data output from the DUT.

## TEST-BENCH OUTPUT VALID 0x405 – 0x604:

|  |  |
| --- | --- |
|  | VALID |
| Bit 31 | 3 Bit 0 |

VALID - This 4 bit field represents the valid output signals from the DUT WRT the TEST-BENCH OUTPUT registers. Bit 3 is set to (1) when bits (31 to 24) of the corresponding output register are valid. Bit 2 is set to (1) when bits (23 to 16) of the corresponding output register are valid. Bit 1 is set to (1) when bits (15 to 8) of the corresponding output register are valid. Bit 0 is set to (1) when bits (7 to 0) of the corresponding output register are valid.