NES-IPCORE-ARINC429

Users-Manual

INDEX

1. Users Information

Nolam Embedded Systems NES**-IPCORE-ARINC429** thank you for your purchase!

We at Nolam Embedded Systems have engineered and produced a high quality product that combines reliability with performance. Your organization will see the benefits of your Nolam Embedded Systems purchase for years to come as we provide a total solution through quality products and continuing customer support.

If you have requested this document and are reading it prior to purchase, we appreciate your interest and look forward to having you join the growing number of satisfied Nolam Embedded Systems customers.

2. Copyright

3. Purpose & Scope

This document contains all information needed for using the NES-ARINC429-IP intellectual property.

The IP implements an interface of ARINC 429 defined in the document “ARINC 429 specification”. The IP could interface with up to 16 Tx channels and 16 Rx channels.

The IP is only supporting full-duplex operation, since this is the only operation mode allowed by the standard.

Acronyms and abbreviations

ARINC	Aeronautical Radio Incorporated
ASIC	Application Specific Integrated Circuit
ASSP	Application Specific Standard Product
DMA	Direct Memory Access
ID	Identifer
MIB	Management Information Base
IP	Intellectual Property
VL	Virtual Link
VHDL	VHSIC Hardware Description Language,
VHSIC	Very High Speed Integrated Circuit

4. Overview

This IP core implements an ARINC429 protocol interface communication.

Product Summary

Intended Use

- ARINC429 Transmitter (Tx)
- ARINC429 Receiver (Rx)

Key Features :

- Supports ARINC specification 429
- Up to 16 Rx and 16 Tx Channels
- Programmable FIFO Depth
- Selectable clock speed
  - 4, 8, 12, 16, 20, 24, 48, 50 or 52 MHz
- Selectable data rate for each channel
  - 12.5 kbps
  - 100 kbps
- Configurable Label Memory Size, up to 256 words
- Loopback funcion for testing
- ARINC 429 bus interface
  - supports standard line drivers and receivers
- Memory interface
  - provides a decoder module to access to memory

Development System

- Complete ARINC 429 Rx/Tx
- Implementation in NES-VESTA board
  - implemented in a cycloneIII device
  - controlled via an USB port
- Line Driver and Receiver components (to be added)

Synthesis and simulation support

- Xilinx ISE

4.1. General description

This IP core ARINC429 provides a complete transmitter (Tx) and receiver (Rx). The total system implemented with our communication protocol is shown in Figure 1.

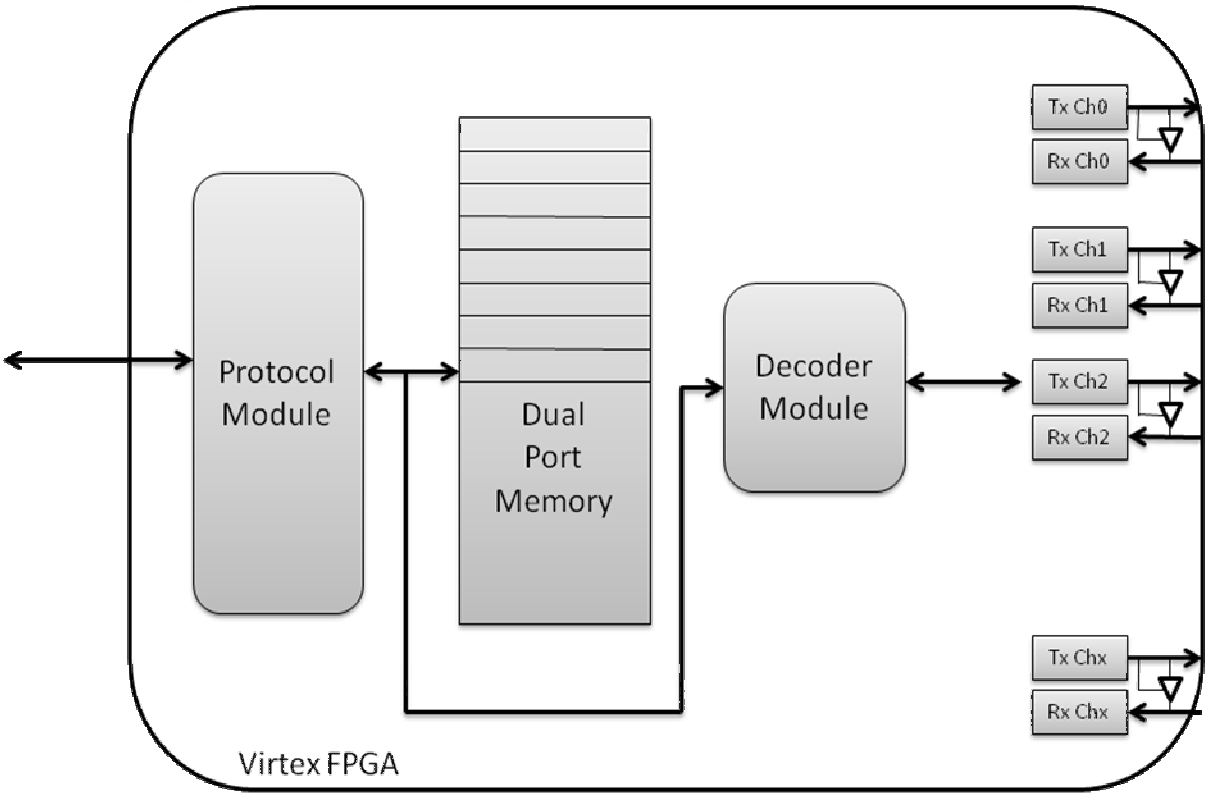


Figure 1 Block Diagram of total system in NES-VESTA

The core consists of two main blocks: Transmit, Receive. ARINC429 Core requires control signals generated by decoder module that you should create according to your system, in this case a USB protocol module. The decoder module generates chip select and registers select signals, reads and writes memory, sends configuration data and Tx data to ARINC429 core, and receives Rx data and status data. The Rx and Tx cores interface to the ARINC429 bus through an external ARINC 429 line driver and line receiver.

External Components
There are two external components required for proper operation of ARINC429 core:

- Standard ARINC 429 Line Driver
- Standard ARINC 429 Line Receiver

4.2. ARINC429 Overview

ARINC429 is a two-wire, point-to-point data bus that is application-specific for commercial and transport aircraft. The connection wires are twisted pairs. Words are 32 bits in length and most messages consist of a single data word. The specification defines the electrical standard and data characteristics and protocols.

ARINC429 uses a unidirectional data bus standard (Tx and Rx are on seperate ports) known as the Mark 33 Digital Information Transfer System (DITS). Messages are transmitted at 12.5, 50(optional), or 100kbps to other system elements that are monitoring the bus messages. There are always either 32-bit data words or the Null state on the bus. Data of

ARINC standard is transmitted in a bipolar, Return-to-Zero format consisting of High, Low, and Null states. A minimum of four Null bits should be transmitted between two words.

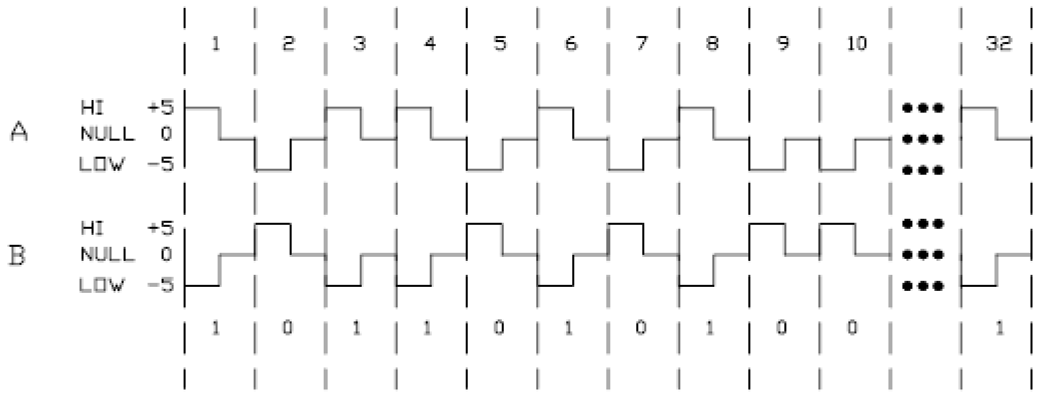


Figure 2 Data format of ARINC 429 standard

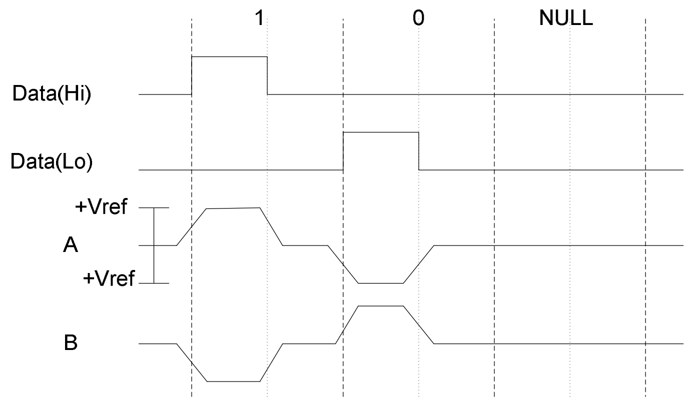
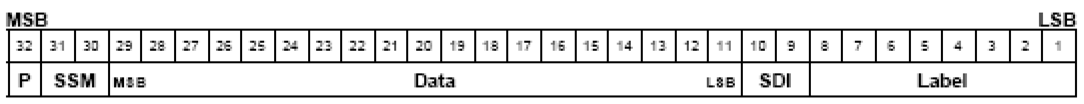


Figure 3: ARINC429 waveform of line driver and receiver

ARINC 429 data words are 32 bit words made up of five primary fields:

- Parity - 1 bit
- Sign/Status Matrix (SSM) -2 bits
- Data -19bits
- Source/Destination Identifier(SDI) -2 bits
- Label -8 bits



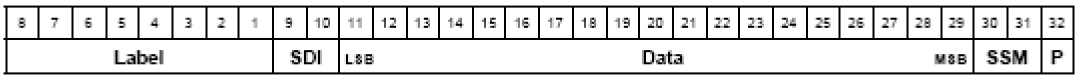
MSB																															LSB
32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
P	SSM		MSB Data																		LSB	SDI		Label							

Figure 3: ARINC 429 32 bit word format

The parity bit is bit 32 (the MSB). SSM is the Sign/Status Matrix and is included as bits 30 and 31. Bits 11 to 29 contain the data. Binary Coded Decimal (BCD) and binary encoding (BNR) are common ARINC data formats. Data formats can also be mixed. Bits 9 and 10 are Source/Destination Identifiers (SDI) and indicate for which receiver the data is intended. Bits 1 to 8 contain a label (label words) identifying the data type.

Label words are quite specific in ARINC 429. Each aircraft may be equipped with different electronic eqipment and systems needing interconnection. A large amount of equipments may be involved, depending on the aircraft. The ARINC specification identifies the equipment

ID, a series of digital identification numbers. Examples of equipment are Flight Management Computers, Inertial Reference Systems, Fuel Tanks, Tire Pressure Monitoring Systems, and GPS Sensors.
The only two fields definitively required are the lable and the parity bit. Any unused bits are padded with zeros.
When transmitting data words on the ARINC bus, the Label is transmitted first, MSB first, followed by the rest of the bit field, LSB first. Bit transmission order looks like this:



8	7	6	5	4	3	2	1	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Label								SDI		LSB								Data										MSB	SSM		P

Figure 4: ARINC 429 word transfer order

5. ARINC 429 Core

5.1 ARINC 429 Core interface

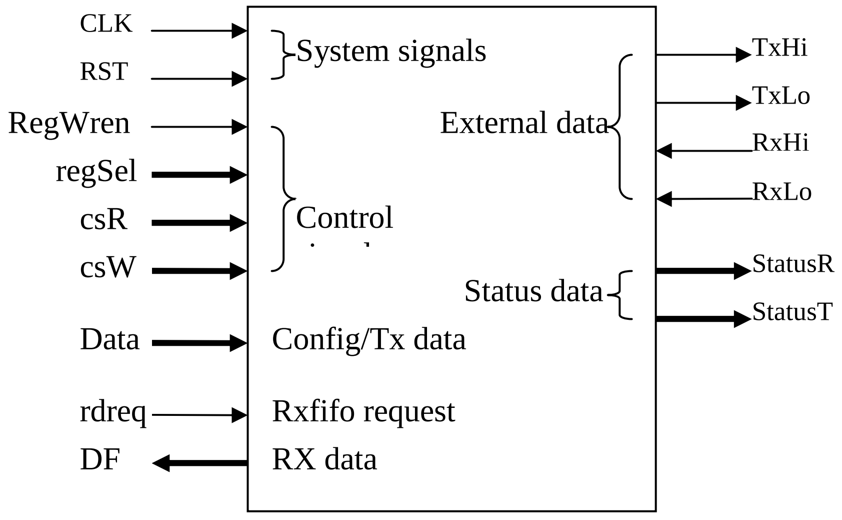


Figure 5: IP external representation

Signals description:

- System signals
  - *CLK* : selectable clock frequency, generic parameter clkSel needs to be configured accordingly to generate a proper sample clock for Tx and Rx(refer to)
  - *RST*: synchronous reset signal
- Control signals
  - *RegWren*: a pulse of one system clock period on this signal indicates a requirement to write a register and the new value is avaialble on Data port
  - *regSel*(1..0) : register select signal indicates which register's value needs to be loaded:
    "00": RX control register
    "01": Label memory register
    "10": TXD register
    "11": TX control register
  - *csR*(3..0): select signal for channels(0 to 15) whose register 's value is avaiable

The RegWren, regSel and csR registers are used together to read in registers' values. During high level of RegWren, the required register select value, channel select value should be valid on ports RegSel and csR, and the value of register to be written should be also valid at the same time on bus Data.

- *csW*(3..0): select signal for Rx channels(0 to 15) to which rdreq signal is active.

An example of usage of the control signals is shown in Figure 6.

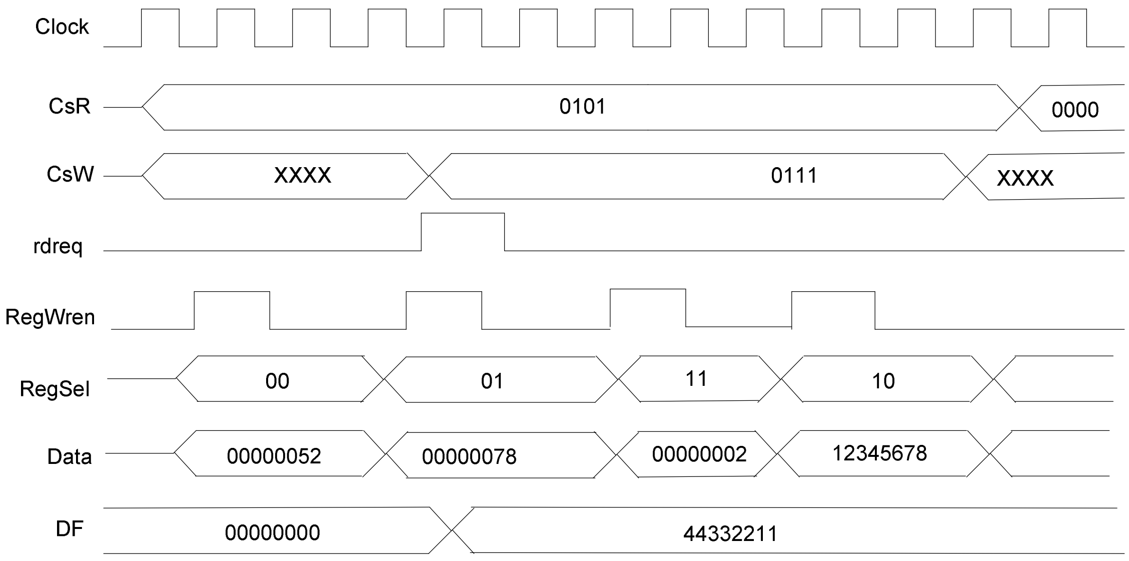


Figure 6: An example of access registers

In this example, we enable the loopback function of channel 5, so RX channel 5 will not accept input from external ports RxHi and RxLo, but data sent by Tx Channel 5 will be received internally by the Rx Channel5. We first write Rx control register to 0x52 indicating that data rate is selected to 100kbps, label comparison is enabled, SDI comparison is enable, and the SDI bits are "10" (please refer to Section 5.5 of the ARINC429 Report for more details on the use of Registers and their description).

Secondly, we write a label 0x78 into label memory. At the same time we try to read RX FIFO of RX channel 7, after one cycle, we can get output from DF port (e.g. 0x44332211). Then we set Tx control register of channel 5 to 0x02 which means a data rate of 100kbps, loopback enabled. At then, we write 0x12345678 to the Tx FIFO using the TXD register. Since label and SDI bits are matched with the corresponding value, the word 0x12345678 will be received by Rx channel with no parity error.

- *Data* (31... 0): the 32bit data is the content of register selected by regSel and csR. For control registers and label memory register, only the less significant byte is valid data, the rest three bytes are ignored.
- *Rdreq*: read request of Rx FIFO,
- *DF*(31..0) : the 32bit RX data read from Rx FIFO; available one cycle after Rdreq
- *External data*: interface with ARINC 429 line driver and receiver
- Status Data(7..0): content of status register of Rx and Tx

5.2. Rx block

The Rx block is responsibe for generating sample clock, performing serial-to-parallel conversion and performing parity / label / SDI check on the incoming data. The incoming serial data RxHi and RxLo is strobed by sample clock which is 8 times of data rate, and rearranged to the normal order from LSB to MSB. It is then checked with the parity bit using the value calculated while receiving data, and compared with programmed labels in the label memory, then compared with the SDI bits in control register. The two comparisons could be disabled by writing a corresponding control word in the Rx control register.
If the label-compare bit in the Rx control register is enabled, then the data which matches a label with the stored labels will be stored in the Rx FIFO. If the label-compare bit in the receive control register is disabled, then the incoming data will be stored in the FIFO without comparing against the labels in label memory.

To program label memory, you should set bit 7 of RX control register. When this bit is set to one, the read and the write pointers of label memory are initialized to zero. The old label content still exists, but the core keeps track only of the new labels and does not use the old lables during label compare. To load new labels into label memory, the bit7 of RX control register has to be set to zero again before starting to write any new value to the label memory register. And during reception, this bit needs to be zero to garantee a correct reception.

The Rx module contains 3 registers: control register, status register and label memory register. The control register is used to configure functions, and status register is used to show the status of RX FIFO and parity error. Refer to Table 2 and Table 3 for detailed descriptions of the control and status register bits. Access to control register and label memory register is decided by signals RegSel and RegWren. The latest value of status regsiter is outputted by port statusR.

The status regsiter is mainly used to decide when data is received by three FIFO status bits:

- rx_fifo_empty: FIFO is empty;
- rx_fifo_half_full: FIFO is filled up to the programmed RX_FIFO_LEVEL;
- rx_fifo_full: FIFO is full.

5.3. Tx block

The TX block converts the 32 bits parallel data from TX FIFO to serial data according to the order described above, and shapes data for ARINC429 line drivers. Before sending to TX block, the 32bit data has already contains all the information in the ARINC 429 format. The transmission starts as soon as one complete ARINC word has been written in TX FIFO. The Tx block contains two 8bits registers (refer to Table 4 and Table 5) and also a 32bits register TXD: TX control register and status register. Control register is used to configure functions and status register shows 3 FIFO statuses which are the same as those of RX FIFO.

TXD register contains data to be transmitted. To write a word in TX FIFO, you need to write in TXD register with correct RegSel, RegWren signals and a valid data on bus Data, then data in TXD will be written directly into TX FIFO.

The loopback function of Tx block allows a wrap back of transmitted data of TX channel x to RX channel x. If Loopback is enabled, the related RX channel will not receive frames from external ports. The frame sent by TX channel will be looped back internally.

For both RX and TX FIFOs, are 32bits wide and has a depth of 64 words. Depending on the FIFO status, the host will either read the FIFO before it overflows, or not attempt to read the FIFO if it is empty.
Note: The FIFOs used for both RX block and TX block are configured to ignore write access when FIFO is already full and read access when FIFO is still empty. That's to say, writing to a full FIFO has no effect, and reading an empty FIFO will return zeros.

5.4. Parameters

Parameter Name	Description	Value Range	Unit	Default
clkSel_p	Selection of input clock for sample clock generator. Adapt its value according to the clock frequency you use.	"000": 48 "001": 4 "010": 12 "011": 16 "100": 20 "101": 24 "110": 52	MHz	"000": 48MHz
cs_p	Channel select	"0000"~"1111"		"1000"
lab_size_p	label memory size	1~256	byte	64
fifo_depth_p	FIFO depth, a power of two	depends on memory space	word	32
fifo_level_p	when number of filled fifo is greater than or equal to this value, the fifo_half_full signal will be asserted	1~fifo_depth_p-1	word	16
fifo_widthu_p	Width of fifo pointer. The recommended value for this parameter is ceil.	1~log2(fifo_depth_p)	bit	5(log2(32))

Table 1: Parameters of ARINC 429 Core

5.5. Register Description

Rx Control register

Bit	Function	Reset Value	Description
0	data rate selection	0	0: 100kbps, 1: 12,5kbps
1	label comparison enable	0	0: disable, 1: enable
2	clear parity error in status register. Writing 1 to this bit once will clear PE only once.	0	0: disable, 1: enable
3	Parity odd/even	0	0: odd, 1: even
4	SDI bits comparison enable	0	0: disable, 1: enable

5	match bit9 of the word	0	if bit4 is set, then this bit should match the bit9
6	match bit10 of the word	0	if bit4 is set, then this bit should match the bit10
7	reload label memory	0	1: label memory pointer are initialized to zero

Table 2: Rx control register description

Bit	Function	Reset Value	Description
0	FIFO empty	1	0: not empty, 1: empty
1	FIFO reaches programmed level	0	0: less than programmed level, 1: greater than programmed level
2	FIFO full	0	0: not full, 1: full
3	Parity Error	0	0: no parity error, 1: error detected

Table 3: Rx status registers description

Bit	Function	Reset Value	Description
0	data rate selection	0	0: 100kbps, 1: 12,5kbps
1	Loopback enable	0	0: disable, 1: enable
2	Reserved	0	reserved bit
3	parity odd/even	0	0: odd, 1: even

Table 4: Tx control register description

Bit	Function	Reset Value	Description
0	FIFO empty	1	0: not empty, 1: empty
1	FIFO reaches programmed level	0	0: less than programmed level, 1: greater than programmed level
2	FIFO full	0	0: not full, 1: full

Table 5: Tx status register description