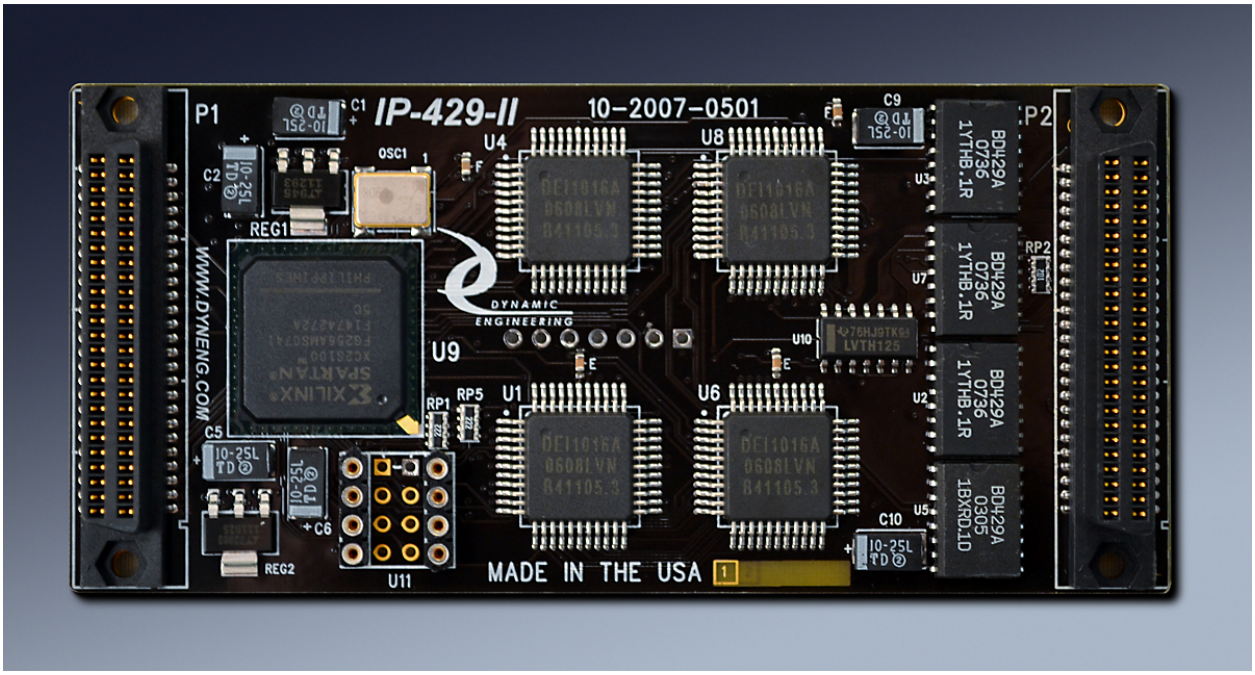
DYNAMIC ENGINEERING

150 DuBois St Suite C, Santa Cruz, CA 95060
831-457-8891 **Fax** 831-457-4793
Web Page http://www.dyneng.com
E-Mail sales@dyneng.com
Est. 1988

User Manual

IP-429-II

ARINC 429 Interface
IP Module



Revision A3 7/25/16
Corresponding Hardware: 10-2007-0502

IP429-II
ARINC 429 Interface
IP Module

Dynamic Engineering
150 DuBois St Suite C
Santa Cruz, CA 95060
831- 457-8891
831-457-4793 FAX

Table of Contents

List of Figures

Product Description

ARINC 429 is a versatile standardized bus interface. ARINC 429 is robust featuring Manchester encoding and enough voltage swing to provide excellent noise immunity and reasonable throughput rates. ARINC 429 is the interface of choice for critical applications; for example aircraft instrumentation and control. There are many devices supporting the 429 bus - printers, instrumentation, sensors and more.

IP-429-II makes it easy to gain access to the ARINC 429 bus. Just connect; program a few registers and use like an IO device. Reference software, schematics and debugging aides are available in the Engineering Kit.

IP-429-II is an upgrade to the original IP-429 design. Dynamic Engineering designed the original IP-429 in 1998 and both manufactured and licensed the design. GreenSpring Computers, and later SBS manufactured the board under their licensing agreements.

The base version of the upgraded design uses the same address and bit maps. The design is enhanced with the Time Trigger / Time Stamp capability now being included on all versions and moved into the IO space. The IO connector has the same signal definitions. The revision 2 and later boards have FLASH memory instead of the PROM and the JTAG signals available on the IO connector.

Please refer to the web page for updated versions with enhanced features. The new versions will have different IDPROM definitions to allow differentiation in your system. Previously purchased IP-429-II rev B and later boards can be updated to make use of the new features.

IP-429-II is capable of providing up to 8 Receive and 4 Transmit channels of ARINC 429 compatible IO in one IP position. Each of the eight (8) receive channels has a 32 bit Time Tag register associated with it. The Time Tag generator operates with a 1 uS resolution. When each new message is received the Data Ready signal from the 429 decoder device is used to trigger the time stamp for that channel.

In addition to the ARINC 429 capability, a parallel IO port is provided to allow custom formats and specialized status to be implemented. The hardware is capable of other formats including 571, 575, and 706 with software set-up changes.

IP-429 supports both the High and Low speed standards for data transmission. A programmable register allows each TX channel to operate at either rate. Each transmitter is supported by an 8 deep 32 bit wide FIFO.

Transmission and Reception can be done on an interrupt or polled basis. The interrupts are individually maskable. The vector is user programmable by a read/write register. The interrupt occurs on IntReq0.

The IP interface is 8 and 32 MHz. capable for efficient programming and data transfer.

All configuration registers support read and write operations for maximum software convenience. Word and byte operations are supported (please refer to the memory map).

IP-429-II conforms to the VITA standard. This guarantees compatibility with multiple IP Carrier boards. Because the IP may be mounted on different form factors, while maintaining plug and software compatibility, system prototyping may be done on one IP Carrier board, with final system implementation on a different one. Dynamic Engineering carrier boards have Drivers available for Windows® and Linux. IP-429-II is supported with a Windows driver included with the purchase of the card. This manual contains enough data to write your own driver should you prefer to take that approach.

Using IP-429 and a PCI3IP or PCI5IP creates a PCI-429 capability. For example using an IP-429-4 and a PCI3IP makes a PCI slot compatible ARINC 429 solution with 8 receivers and 4 transmitters. Adding more IP's can add more channels up to 40 receivers and 20 transmitters in the PCI slot [on a PCI5IP]. Other options are available for PCIe, cPCI [3U and 6U] and PC104p. Coming soon VPX, cPCIe, and PCIe104 carriers.

Theory of Operation

IP-429-II is designed for the purpose of transferring data from one point to another with the ARINC-429 protocol.

IP-429-II features a Xilinx FPGA and industry standard 3282 compatible ARINC 429 receiver / transmitter devices. The FPGA contains the IP interface and control required for the 429 devices. Each 3282 provides two Receiver channels and one partial Transmit channel. A separate device provides the “drive” circuitry for the transmit channel. The ‘3282 data sheet is available on-line from the IP-429 webpage.

IP-429-II is a part of the IP Module family of I/O products. It meets the IP Module VITA Standard. Contact VITA for a copy of this specification. It is assumed that the reader is at least casually familiar with this document and logic design. In standard configuration it is a Type II mechanical with passive low profile components on the back of the board and one slot wide.

The bus interface to the host CPU is controlled by a logic block within the Xilinx device that contains the decoding and timing elements required to interface to the IP bus interface. The timing is referenced to the 8 or 32 MHz IP logic clock. The IP responds to the ID, INT, MEM and IO selects. The FPGA design requires wait states for read or write cycles to any address. Hold cycles are supported as required by the host processor. Data remains enabled during a read until the host removes the SEL line. Local timing terminates a write cycle prior to SEL being de-asserted.

The interface to the 429 encoder/decoder modules is operated at 100 MHz. to allow for precise timing with as little wasted time as possible. The IO space decodes for the write and read functions are set based on the IP clock rate. The state-machine detects the request and handles the data transfer to/from the encoder/decoder. The number of wait-states will depend on the IP clock rate in use. At 8 Mhz the timing of the encoder/decoder is similar which will lead to fewer wait states. At 32 Mhz the IP clock is faster than the read or write timing to the encoder/decoder leading to more wait-states. The overall time will be shorter with the 32 MHz clock. If available it [32 MHz] is recommended for use.

Before transmitting or receiving data IP-429-II requires register programming to select data rates and formats as well as system clock speed, interrupt masking etc. Each of the programmable registers is described in the programming section. Once IP-429-II has been set-up for operation in your environment, data can be transferred.

To transmit, data is loaded into the transmitter FIFO within the channel of interest and

the transmission enabled. If programmed, an interrupt will alert the host that the data has been transferred. If preferred, the TX status can be polled and the transmitter status determined. Each transmitter channel is independent of the other transmitter channels. *Please note: The 3282 FIFO will store up to 8 values to transmit if the transmitter enable is disabled while loading. If the transmitter is enabled while loading the FIFO, only 1 data word out of the 8 loaded will be transmitted [most likely based on system write timing]. Please refer to the 3282 data sheet for more information.*

To receive data another transmitter in the system sends data on the bus, which is connected, to the IP-429. There are up to eight receive channels per IP-429-II and each channel can be connected to a different ARINC 429 bus. The Receiver channels are controlled in pairs for clock speed. When data is received, an interrupt can be generated to the host. The interrupt to the host is the 'OR' of the interrupt requests from all of the channels. Each channel has an interrupt mask. There is a status register to allow the host to determine the cause of the interrupt and to set priorities for responding to the interrupt(s). The interface can be operated in a polled mode by reading the status register.

If the system needs to know when data was received, Time Tagging can be useful. IP-429-II supports Time Tagging by providing a 32 bit counter, which operates at 1 MHz to provide a 1 uS Time Tag count. When the interrupt from any of the receive channels is asserted the current "time" is registered into one of eight Time Tag registers. The software can access the registers to read the count. The time should be read before another interrupt is generated on that channel or the time will be updated for the new interrupt. The counter is resettable to allow synchronization with a system timer. With 32 bits, the counter will roll over after ~71.58 minutes.

The Time Tag registers are internal to the Xilinx and are loaded automatically. The registers are aligned to allow reading as longwords – using the automatic double reads built into Dynamic Engineering carriers.

Address Map

IO Space

Function	Offset	Width	Type
// IP 429 relative addresses //			
#define IP429II_BASE_REG0	0x00		// byte on word boundary
#define IP429II_BASE_REG1	0x02		// byte on word boundary
#define IP429II_BASE_REG2	0x04		// byte on word boundary
#define IP429II_BASE_REG3	0x06		// byte on word boundary
#define IP428II_VECTOR	0x08		// byte on word boundary
#define IP429II_STATUS0	0x0a		// byte on word boundary
#define IP429II_STATUS1	0x0c		// byte on word boundary
#define IP429II_PARALLEL	0x0e		// byte on word boundary
#define IP429II_INFO	0x10		// Driver info register word
#define IP429II_TS_DEV1_CH1_LWR	0x14		Time Tag Device 1 receiver 1 D15..D0 read only
#define IP429II_TS_DEV1_CH1_UPR	0x16		Time Tag Device 1 receiver 1 D31..D16 read only
#define IP429II_TS_DEV1_CH2_LWR	0x18		Time Tag Device 1 receiver 2 D15..D0 read only
#define IP429II_TS_DEV1_CH2_UPR	0x1A		Time Tag Device 1 receiver 2 D31..D16 read only
#define IP429II_TS_DEV2_CH1_LWR	0x1C		Time Tag Device 2 receiver 1 D15..D0 read only
#define IP429II_TS_DEV2_CH1_UPR	0x1E		Time Tag Device 2 receiver 1 D31..D16 read only
#define IP429II_TS_DEV2_CH2_LWR	0x20		Time Tag Device 2 receiver 2 D15..D0 read only
#define IP429II_TS_DEV2_CH2_UPR	0x22		Time Tag Device 2 receiver 2 D31..D16 read only
#define IP429II_TS_DEV3_CH1_LWR	0x24		Time Tag Device 3 receiver 1 D15..D0 read only
#define IP429II_TS_DEV3_CH1_UPR	0x26		Time Tag Device 3 receiver 1 D31..D16 read only
#define IP429II_TS_DEV3_CH2_LWR	0x28		Time Tag Device 3 receiver 2 D15..D0 read only
#define IP429II_TS_DEV3_CH2_UPR	0x2A		Time Tag Device 3 receiver 2 D31..D16 read only
#define IP429II_TS_DEV4_CH1_LWR	0x2C		Time Tag Device 4 receiver 1 D15..D0 read only
#define IP429II_TS_DEV4_CH1_UPR	0x2E		Time Tag Device 4 receiver 1 D31..D16 read only
#define IP429II_TS_DEV4_CH2_LWR	0x30		Time Tag Device 4 receiver 2 D15..D0 read only
#define IP429II_TS_DEV4_CH2_UPR	0x32		Time Tag Device 4 receiver 2 D31..D16 read only
#define IP429II_OE0_DEV1_L	0x40		// read from Device 1 port 1 lower half
#define IP429II_OE0_DEV1_U	0x42		// read from Device 1 port 1 upper half
#define IP429II_OE1_DEV1_L	0x44		// read from Device 1 port 2 lower half
#define IP429II_OE1_DEV1_U	0x46		// read from Device 1 port 2 upper half
#define IP429II_LD1_DEV1	0x48		// write to Device 1 TX port lower half
#define IP429II_LD2_DEV1	0x4a		// write to Device 1 TX port upper half
#define IP429II_CNTL_DEV1	0x4c		// write to Device 1 control word
#define IP429II_OE0_DEV2_L	0x50		// read from Device 2 port 1 lower half
#define IP429II_OE0_DEV2_U	0x52		// read from Device 2 port 1 upper half
#define IP429II_OE1_DEV2_L	0x54		// read from Device 2 port 2 lower half

```
#define IP429II_OE1_DEV2_U      0x56    // read from Device 2 port 2 upper half
#define IP429II_LD1_DEV2        0x58    // write to Device 2 TX port lower half
#define IP429II_LD2_DEV2        0x5a    // write to Device 2 TX port upper half
#define IP429II_CNTL_DEV2       0x5c    // write to Device 2 control word

#define IP429II_OE0_DEV3_L      0x60    // read from Device 3 port 1 lower half
#define IP429II_OE0_DEV3_U      0x62    // read from Device 3 port 1 upper half
#define IP429II_OE1_DEV3_L      0x64    // read from Device 3 port 2 lower half
#define IP429II_OE1_DEV3_U      0x66    // read from Device 3 port 2 upper half
#define IP429II_LD1_DEV3        0x68    // write to Device 3 TX port lower half
#define IP429II_LD2_DEV3        0x6a    // write to Device 3 TX port upper half
#define IP429II_CNTL_DEV3       0x6c    // write to Device 3 control word

#define IP429II_OE0_DEV4_L      0x70    // read from Device 4 port 1 lower half
#define IP429II_OE0_DEV4_U      0x72    // read from Device 4 port 1 upper half
#define IP429II_OE1_DEV4_L      0x74    // read from Device 4 port 2 lower half
#define IP429II_OE1_DEV4_U      0x76    // read from Device 4 port 2 upper half
#define IP429II_LD1_DEV4        0x78    // write to Device 4 TX port lower half
#define IP429II_LD2_DEV4        0x7a    // write to Device 4 TX port upper half
#define IP429II_CNTL_DEV4       0x7c    // write to Device  4 control word
```

FIGURE 1 IP-429 INTERNAL ADDRESS MAP

The address map provided is for the local decoding performed within the IP-429-II. The addresses are all offsets from a base address. The carrier board where the IP is installed provides the base address and controls the “naming of the bytes”. We refer to the bytes with upper D15-D8 and lower is D7-D0. When byte wide data is located on the lower byte then an odd address results or the use of a word access using only the lower byte of data. We prefer the word oriented approach because it is more consistent across platforms.

Please note the double wide registers for TX and RX data functions are located on LW boundaries allowing for 32 bit read or write actions with carriers that support automatic 32 ⇔ 16 conversion. All Dynamic Engineering carriers have this feature.

IP-429-II can have up to 4 of the encoder/decoder “chips” installed. The –1 version has device 1 installed , -2 has both device 0 and device 1 and so forth. The names for the decodes above have the chip number followed by the port within the chip and the function. For example IP429II_OE1_DEV3_U is the upper half data read from chip 3 port 1. The OE, LD, CNTL etc. can be used as a guide to the IC function name when comparing with the schematic or datasheet.

Programming

Programming IP-429 requires only the ability to read and write data in the host's IP I/O and Memory spaces. The base address is determined by the IP Carrier board. This documentation refers to the address where the IO space for the slot that the IP is installed in as the base address.

A typical sequence would be to first write to the vector register with the desired interrupt vector. Please note that most carrier boards do not use the interrupt vector. *VME is about the only system still using vectored interrupts.* The interrupt service routine should be loaded and the mask should be set. For auto-vectored systems – PCI, PCIe, VPX etc. this step can be skipped.

Refer to the Theory of Operation section above and the Interrupts section below for more information regarding the exact sequencing and interrupt definitions.

A local oscillator provides a 50 MHz reference. The reference is used to generate a 1 MHz. clock used for the time-stamp and encoder/decoder functions. The clock is doubled [100 MHz.] to use as a reference for the IO state-machine. No programming required for this functionality.

Dynamic Engineering carriers come with Windows and other drivers. The drivers are structured to detect the carrier and the installed IP modules. The carrier driver will automatically load the appropriate IP module driver if available and the generic driver for that IP position if the installed IP is not in the list of supported devices. The carrier has a DIP switch which can be used to differentiate between carrier boards when more than one are in the same system. The IP's are also named to allow for complete control over the addressing path in systems where the address is not known until run-time [PCI, PCIe etc].

Dynamic Engineering drivers take care of the system level interaction and provide an easy to use platform to write your user level software. The driver comes with "Userap" which is an open source collection of tests using the driver to test the card in question. Userap is a great reference for loop-back testing and other quick start code writing.

The use of Dynamic Engineering drivers is highly recommended. The details that follow are for those who just need to know or need to implement a driver for another carrier or unsupported OS.

http://www.dyneng.com/drivers.html has a more complete summary of driver operation.

Register Definitions

IP429II_BASE_REG0

[$00 429 Control Register Port read/write

CONTROL REGISTER 0	
DATA BIT	**DESCRIPTION**
7	Send4 0 = tx disabled, 1 = enabled ch3
6	Send3 0 = tx disabled, 1 = enabled ch2
5	Send2 0 = tx disabled, 1 = enabled ch1
4	Send1 0 = tx disabled, 1 = enabled ch0
3	spare
2	CLR_CNT 0 = run, 1 = clear
1	INT_SET 0 = no interrupt, 1 = force interrupt
0	Clock Speed 0 = 8Mhz, 1 = 32 MHz.

FIGURE 2 IP-429 CONTROL REGISTER 0 BIT MAP

The Clock Speed bit is used to select either the 8 MHz or 32 MHz. IP Clock rate. The Clock Speed selection defaults to 8 MHz. = 0. The selection should match the IP reference clock rate to insure proper operation. This is a legacy bit retained for software commonality but has no effect on operation.

INT_SET is used to force an interrupt condition to occur. This control bit is useful for SW development and HW testing. Default to no interrupt = 0, force interrupt with 1.

CLR_CNT is used to reset the 32 bit Time Stamp counter. The clear bit is written with a 1 and then released with a 0 to set the counter to a 0 value and restore operation. Please refer to the Time Stamp registers for more information.

The Send bits are used to enable the transmitter sections to send when ready. The transmitters are ready to send when there is data in the output FIFO. The Sendx bit should be held in the disabled state if more than one word is to be placed into the output FIFO. If only one word is to be sent each time the Sendx bit can be left enabled and the data will automatically be transferred when LD2 is accessed with the second write.

IP429II_BASE_REG1

$02 429 Control Register Port read/write

CONTROL REGISTER 1	
DATA BIT	**DESCRIPTION**
7	/RESET4 reset, 1 = enabled
6	/RESET3
5	/RESET2
4	/RESET1
3	/DBCEN4 1 = force parity, 0 = normal
2	/DBCEN3
1	/DBCEN2
0	/DBCEN1

FIGURE 3 IP-429 CONTROL REGISTER 1 BIT MAP

/DBCENx is used to force Parity to be inserted into the data stream. This bit is normally left programmed to 0. There is another control bit within the -3282 that also controls Parity along with even or odd sense. It is recommended to use the 3282 control bit. Default is 0.

/RESETx is used to reset the 3282 associated with each channel. The channel should be reset when the 3282 control register is written to. [write only port]. Default is reset [0]. 1 = normal operation. Reset should be asserted for 200 nS minimum. This bit is named based on the 3282 definition. With positive logic this bit acts as an enable. '1' is enabled.

The TX FIFO, bit counters, word counter, gap timers, /DRx, and TXR are affected by reset assertion. The Control register is not.

IP429II_BASE_REG2

$04 429 Control Register Port read/write

CONTROL REGISTER 2	
DATA BIT	**DESCRIPTION**
7	spare
6	spare
5	spare
4	spare
3	Hi/_LO_4 1 = 100Khz, 0 = 12.5 KHz.
2	Hi/_LO_3
1	Hi/_LO_2
0	Hi/_LO_1

FIGURE 4 IP-429 CONTROL REGISTER 2 BIT MAP

The transmit speed port is used to select the transmit speed for each of the ports installed. 1 = 100 KHz. and 0 = 12-14.5Khz. operation. The spare bits are read-write. Please note that the 3282 programming needs to match the CNTL2 selections for speed. This is a legacy register retained for software commonality but has no effect on operation. The new transmitter does not require this input.

IP429II_BASE_REG3

$06 429 Control Register Port read/write

CONTROL REGISTER 3	
DATA BIT	**DESCRIPTION**
7	Tx Int En 4 0 =disabled 1 = enabled
6	Tx Int En 3 0 =disabled 1 = enabled
5	Tx Int En 2 0 =disabled 1 = enabled
4	Tx Int En 1 0 =disabled 1 = enabled
3	Rx Int En 4 0 = disabled 1 = enabled
2	Rx Int En 3 0 = disabled 1 = enabled
1	Rx Int En 2 0 = disabled 1 = enabled
0	Rx Int En 1 0 = disabled 1 = enabled

FIGURE 5 IP-429 CONTROL REGISTER 3 BIT MAP

Configuration	Devices
-1	1
-2	1,2
-3	1,2,3
-4	1,2,3,4

Tx Int En X is used to enable or disable the interrupt associated with the transmit device installed.

Devices not populated on the board should be masked off. 1 = enabled and 0 = disabled for the interrupt. Default = 0. The leading edge of the TXRx status line from the “3282” is used to generate the interrupt. After a transmission the TX FIFO is empty causing the TXRx line to transition high. The line will remain high until data is written into the FIFO for that channel. The interrupt is edge triggered so no further interrupts will occur until another transition occurs. Clear the interrupt request by disabling the interrupt enable for that channel and then [if desired] re-enable. The TXRx bits are available in STATUS1.

Rx Int En X is used to enable or disable the interrupt associated with the receive device. Each device has two receivers. The interrupts for the two receivers are masked together by the chip interrupt mask. The individual receiver within each device is identifiable with STATUS0.

Devices not populated on the board should be masked off. 1 = enabled and 0 = disabled for the interrupt. Default = 0. The level from the DRx status lines from the “3282” are used to generate the interrupt. After a reception the receive register has

data available is causing the DRx line to transition low. The line will remain low until data is read for that channel.

DRx lines transition off once the data is read no further interrupts will occur until another reception occurs. Clear the INT request by reading the data or setting the mask to 0.

IP429II_VECTOR

$08 429 Interrupt Vector Port
The Interrupt vector for the 429 is stored in this byte wide register. This read/write register is initialized to 'xxFF' upon power-on reset or software reset. The vector is stored in the odd byte location [D7..0]. The vector should be initialized before the interrupt is enabled or the mask is lowered. The interrupt is not cleared when the CPU reads the vector. Please refer to the REG3 register description. For auto-vectored systems this register can be ignored.

IP429II_STATUS0

[$0A 429 Control Register Port read only

STATUS REGISTER 0

DATA BIT	DESCRIPTION
15	Interrupt Request Active
14-8	spare, set to '0'
7	DR2 device 4 1 = data ready, 0 = no data
6	DR1 device 4
5	DR2 device 3
4	DR1 device 3
3	DR2 device 2
2	DR1 device 2
1	DR2 device 1
0	DR1 device 1

FIGURE 6 IP-429 STATUS REGISTER 0 BIT MAP

The DRx bits are the Data Ready status outputs from the encoder/decoder devices. The signals are buffered and inverted to create active high data ready signals. DR1 is receiver 1 of device N. DR2 is receiver 2 of device N. Each of the 4 channels has 2 receivers. There are four devices and your hardware may not have all four installed. Uninstalled devices should be masked off when reading this status register. These bits can be used for polled operation of the receivers. The signals return to zero when the data is read. The _oe1 addresses should be accessed for the receiver 1 data and the _oe2 addresses for the receiver 2 data within each channel.

IP429II_STATUS1

[$0C 429 Control Register Port read only

STATUS REGISTER 1	
DATA BIT	**DESCRIPTION**
15-12	Flash Revision Major
11-8	Flash Revision Minor
7-4	Version
3	TXR4
2	TXR3
1	TXR2
0	TXR1

FIGURE 7 IP-429 STATUS REGISTER 1 BIT MAP

The TXRx bits are the Transmitter Ready bits driven from the “3282”. When high the bits indicate that the transmitter is ready to receive data. The rising edge of the signal is used to create the TX completed interrupt as that edge indicates that the transmitter is ready to receive more data. When the signal is low it indicates that the transmitter is ready to send. If the SENDx bit is enabled then the transmission will begin. This bit can be used for polled operation of the transmitter.

With Revision A1 of the FLASH corresponding to 02 and later revision fabs the Version and Revision status fields have been added.

Version is a nibble wide field showing the number of installed 429 channels 1-4.
Flash Revision Major and Minor reflect the revision currently programmed into the card.
A1 is the current revision.

IP429II_Parallel

$0E 429 Control Register Port read/write

CONTROL REGISTER Parallel Port	
DATA BIT	**DESCRIPTION**
15-8	read-back of lower 7 bits
7	PI_7 – read-back “cable side”
6	PI_6
5	PI_5
4	PI_4
3	PIO_3
2	PIO_2
1	PIO_1
0	PIO_0

FIGURE 8 IP-429 CONTROL REGISTER 0 BIT MAP

The PIO port consists of read only and read-writeable bits. The lower 4 bits [0-3] are read-writeable. The bits are used to control open drain high sink drivers. The read-back comes from the bus side of the drivers. If high then a low on the driver indicates another device is driving the line low. The lines are pulled up with 1K pull-up resistors and are terminated with 33Ω series resistors. The upper 4 bits are read only and are terminated with 33Ω series terminators.

The port is configured with an 8 bit register. The lower 4 bits are used to control the open drain drivers as described above. The lower 8 bits of the port are the read-back from the cable side of the interface. The upper 8 bits are the registered bits read-back – not affected by the cable.

IP429_Encoder/Decoder

#define IP429II_OE0_DEV1_L	0x40	// read from Device 1 port 1 lower half
#define IP429II_OE0_DEV1_U	0x42	// read from Device 1 port 1 upper half
#define IP429II_OE1_DEV1_L	0x44	// read from Device 1 port 2 lower half
#define IP429II_OE1_DEV1_U	0x46	// read from Device 1 port 2 upper half
#define IP429II_LD1_DEV1	0x48	// write to Device 1 TX port lower half
#define IP429II_LD2_DEV1	0x4a	// write to Device 1 TX port upper half
#define IP429II_CNTL_DEV1	0x4c	// write to Device 1 control word

There are up to 4 “3282” devices installed per board. Each “3282” has two receivers and 1 transmitter port. To support the control of the part several addresses are defined per channel and replicated for each channel. Each channel is independent of the other channels. Some settings within each “channel” [device installed] are binding on the assets of the channel. For example the receive rate can be programmed high or low per channel and affects both receivers within the channel.

IP429II_OE0_DEV1_L
IP429II_OE0_DEV1_U
IP429II_OE1_DEV1_L
IP429II_OE1_DEV1_U

These four addresses are for the reading of data from receiver 1 or 2 of channel 0. One should access Lower then Upper. Both addresses actually activate the same control signal to the 3282, but provide a different sense to the SEL line to tell the 3282 which 16 bit word to enable onto the bus at that time. When “U” is accessed the DRx flag is reset.

“L” corresponds to the lower [D15..0] 16 bits of the 32 bit ARINC 429 data word.
“U” corresponds to the upper [D31..16] 16 bits of the 32 bit ARINC 429 data word.

IP429II_LD1_DEV1
IP429II_LD2_DEV1

The load addresses are for loading the upper and lower words for transmission. Word 1 should be loaded before word 2. Word 1 corresponds to the lower word of data [D15..0], and word 2 corresponds to the upper word [D31..16]

IP429_CHx_CNTL

CONTROL REGISTER 0	
DATA BIT	**DESCRIPTION**
15	WLSEL
14	RCVSEL
13	TXSEL
12	PARCK
11	Y2
10	X2
9	SDEN2
8	Y1
7	X1
6	SDEN1
5	/SLFTST
4	PAREN
3	UNUSED
2	UNUSED
1	UNUSED
0	UNUSED

FIGURE 9 IP-429 3282 CONTROL REGISTER BIT MAP

The details of programming the 3282 can be found in the DDC-ILC DATA Device Corporation Data Book. A copy of the 3282 data sheet is available on our web site. The file is available on the IP-429 page. http://www.dyneng.com/ip429.html

A summary of the control bits follows.

PAREN - Transmitter Parity Enable. 0 = data 1 = parity if DBCEN is set low. If DBCEN is programmed high then parity is always inserted independent of PAREN.

/SLFTST - Self Test Enable. Logic 0 enables self-test and 1 places the device in normal mode. In self test, the data is transmitted to the receiver channels. Receiver 1 receives the normal data and receiver 2 receives the complement.

When in self test mode some confusion can arise from the treatment of the parity bit. The '3282 allows you to turn off parity on the transmitter, but parity is always enabled for the receiver. The receiver will calculate parity based on the entire 32-bit word. The receiver will insert a logic zero for odd parity (no error) and a logic one for even parity (error). In the case of writing 0x4321 and 0x8765, parity is odd so the receiver inserts a logic zero into bit 8 of word 1. This explains why it is received as 0x4221. If you were to write 0x5321 and 0x8765, you would receive 0x5321 and 0x8765. This would be an

error condition because parity is even and thus, bit 8 of word 1 is asserted high. This is very briefly mentioned on page 6 of the DDC-03282 data sheet.

SDEN1,2 X1,2 Y1,2 S/D code check enable and check bits. If SDENx is set to 1 then the corresponding receiver checks for the X and Y bits to match. If set to 0 then all properly formatted data is received.

PARCK Parity Check Enable A 1 causes even parity and a 0 selects odd parity for the transmitter if PAREN is set.

TXSEL, RCVSEL 0 = 100K transmit or receive speed and 1 selects 12.5K. Transmitter needs to match the receiver of the data stream not necessarily the receiver within the same device. Both receiver channels are affected by the selection within the same channel.

WLSEL Word length select. 0 = 32 bits, 1 = 25 bits in length.

In general, the control word options should only be changed when the device is placed in the reset state with control register 1.

IP429_CHXX_TS

The Time Stamp registers are now accessed in the IO space. Be sure to use the correct offset. There are eight (8) registers each used to store one 32 bit Time Tag. The receiver number is the number assigned to each receiver on the board. Please refer to the chart below. The U/L number indicates which part of the Time Tag/Stamp is available at that address. The Time Stamp is a 32 bit number and the port size available is 16 bits. It takes two reads and 1 shift to get the full 32 bit value. The registers are organized to allow auto-incremented addresses. For example with a PCI3IP you can do a 32 bit read from the L address to read both the L and U as a packed 32 bit word.

The base counter is 32 bits wide and “counts up” at a rate of 1 MHz. When the Receive interrupt for a particular channel is detected, the current count is stored into the Time Tag register for that channel. The interrupt event used is electronically prior to the mask and operates even if the interrupt for a particular channel is masked off. The Time Tag can be used in polled mode.

Channel	Receivers	3282 definition
channel 1	CH1,CH2	1,2
channel 2	CH1,CH2	1,2
channel 3	CH1,CH2	1,2
channel 4	CH1,CH2	1,2

IP429II_TS_DEV1_(CH1,CH2)_(UPR,LWR) - The Device differentiates between the installed devices per the dash number. The CH1 or CH2 selects which receiver is being referenced. The UPR or LWR selects the part of the Time Stamp being retrieved. Use of 32 bit accesses for the Time Stamp, Load and Read functions is recommended.

Interrupts

All IP Module interrupts are vectored. The vector from the IP-429 comes from a vector register loaded as part of the initialization process. The vector register can be programmed to any 8 bit value. The default value is $FF which is sometimes not a valid user vector. Software is responsible for choosing a valid user vector.

Although the IP specification calls for vectored interrupts many modern systems are auto-vectored and do not require a vector to be read. This will be an automatic part of your system – with vectored interrupts the INT space is accessed to read the vector and in an auto-vectored system the status register is read to determine what the cause or causes are.

The IP-429 interface logic will generate an interrupt request when a transmission or reception is complete and the INTEN bits in the control registers are set. The interrupt is mapped to interrupt request 0. The CPU will respond by asserting INT. The hardware will automatically supply the appropriate interrupt vector and clear the request when accessed by the CPU. The source of the interrupt is obtained by reading the status registers. The status remains valid until the interrupt is cleared. The interrupt is cleared by accessing the individual 429 interface devices for Receiver interrupts and toggling the enable for Transmitter interrupts.

The interrupt level seen by the CPU is determined by the IP Carrier board being used.

If operating in a polled mode and making use of the interrupts for status the interrupt enables should be disabled and the status register polled to determine which interrupt is active.

Power on initialization will provide a cleared interrupt request, interrupts disabled, and interrupt vector of $FF.

ID PROM

Every IP contains an ID PROM, whose size is at least 32 x 8 bits. The ID PROM aids in software auto configuration and configuration management. The user's software, or a supplied driver, may verify that the device it expects is actually installed at the location it expects, and is nominally functional. The ID PROM contains the manufacturing revision level of the IP. If a driver requires that a particular revision is present; it may check for it directly.

The location of the ID PROM in the host's address space is dependent on which carrier is used. Normally the ID PROM space is directly above the IP's I/O space, or at IP-base + $80. Dynamic Engineering parent drivers use the ID PROM automatically to instantiate the correct child driver for a particular slot.

Standard data in the ID PROM on the IP-429-II is shown in the figure below. For more information on IP ID PROM's refer to the IP Module Logic Interface Specification, available from Dynamic Engineering.

Each of the four dash versions of the IP-429 has unique IDPROM information. The Dash number is reflected in the Driver ID low byte and that change is reflected in the CRC value.

Address	Data-1	-2	-3	-4
01	ASCII "I" ($49)			
03	ASCII "P" ($50)			
05	ASCII "A" ($41)			
07	ASCII "H" ($48)			
09	Manufacturer ID ($1E)			
0B	Model Number ($02)			
0D	Revision ($A1)			
0F	reserved (00)			
11	Driver ID, low byte (01)	02	03	04
13	Driver ID, high byte (00)			
15	No of extra bytes used (0C)			
17	CRC (F8)	2A	7B	AF

FIGURE 10 IP-429 ID PROM STANDARD

LOOP-BACK

With the use of an IP-Debug-IO plus the software contained in the UserAp for the IP-429-II Loop-Back can be performed. The following table shows the connections used for the Dynamic Engineering test software. There are 4 possible devices and all are shown. The first set is for Device 1. The first entry is for the A signal and the second for the B.

The second table is for the Parallel Port. The lower signals are tied to the upper.

TX	CH1	CH2
7	1	4
8	2	5
16	10	13
17	11	14
26	19	22
27	20	23
35	29	32
36	30	33

PIO	
41	45
42	46
43	47
44	48

IP Module Logic Interface Pin Assignment

The figure below gives the pin assignments for the IP Module Logic Interface on the IP-429. Pins marked n/c below are defined by the specification, but not used on the IP-429. Also see the User Manual for your carrier board for more information.

GND		GND		1		26	
	CLK		+5V		2		27
Reset*		R/W*		3		28	
	D0		IDSEL*		4		29
D1		n/c		5		30	
	D2		MEMSEL*		6		31
D3		n/c		7		32	
	D4		IntSel*		8		33
D5		n/c		9		34	
	D6		IOSel*		10		35
D7		n/c		11		36	
	D8		A1		12		37
D9		n/c		13		38	
	D10		A2		14		39
D11		n/c		15		40	
	D12		A3		16		41
D13		IntReq0*		17		42	
	D14		A4		18		43
D15		n/c		19		44	
	BS0*		A5		20		45
BS1*		n/c		21		46	
	-12V		A6		22		47
+12V		Ack*		23		48	
	+5V		n/c		24		49
GND		GND		25		50	

NOTE 1: The no-connect signals above are defined by the IP Module Logic Interface Specification, but not used by this IP. See the Specification for more information.

NOTE 2: The layout of the pin numbers in this table corresponds to the physical placement of pins on the IP connector. Thus this table may be used to easily locate the physical pin corresponding to a desired signal. Pin 1 is marked with a square pad on the IP Module.

FIGURE 11 IP-429 LOGIC INTERFACE

IP Module IO Interface Pin Assignment

The figure below gives the pin assignments for the IP Module IO Interface on the IP-429. Also see the User Manual for your carrier board for more information.

Dev1Ch1_RXA	Dev3_TXA	1		26	
Dev1Ch1_RXB	Dev3_TXB		2		27
GND	GND	3		28	
Dev1Ch2_RXA	Dev4Ch1_RXA		4		29
Dev1CH2_RXB	Dev4Ch1_RXB	5		30	
GND	GND		6		31
Dev1_TXA	Dev4Ch2_RXA	7		32	
Dev1_TXB	Dev4Ch2_RXB		8		33
GND	GND	9		34	
Dev2Ch1_RXA	Dev4_TXA		10		35
Dev2Ch1_RXB	Dev4_TXB	11		36	
GND	TDI		12		37
Dev2Ch2_RXA	TMS	13		38	
Dev2Ch2_RXB	TCK		14		39
GND	TDO	15		40	
Dev2_TXA	PIO_0		16		41
Dev2_TXB	PIO_1	17		42	
GND	PIO_2		18		43
Dev3Ch1_RXA	PIO_3	19		44	
Dev3Ch1_RXB	PI_4		20		45
GND	PI_5	21		46	
Dev3Ch2_RXA	PI_6		22		47
Dev3Ch2_RXB	PI_7	23		48	
GND	Fused 3.3V		24		49
GND	GND	25		50	

NOTE 1: The layout of the pin numbers in this table corresponds to the physical placement of pins on the IP connector. Thus this table may be used to easily locate the physical pin corresponding to a desired signal.

FIGURE 12 IP-429 IO INTERFACE

Please note: the IO assignments reference all of the possible TX and RX channels. Your board may not have all of the IO installed depending on what was purchased.

Dev1CH1 and Dev1CH2 are part of channel 1, which is installed in all configurations.
Dev2CH1 and Dev2CH2 are part of channel 2, which is installed, in -2, -3, -4 models.
Dev3CH1 and Dev3CH2 are part of channel 3, which is installed, in -3, -4 models.
Dev4CH1 and Dev4CH2 are part of channel 4, which is only installed, in the -4 model.

Dev1, 2, 3, 4 TX refer to the transmitters that can be installed. Each channel has one transmitter.

Applications Guide

Interfacing

Some general interfacing guidelines are presented below. Do not hesitate to contact the factory if you need more assistance.

Watch the system grounds. All electrically connected equipment should have a fail-safe common ground that is large enough to handle all current loads without affecting noise immunity. Power supplies and power-consuming loads should all have their own ground wires back to a common point.

Keep cables short. Flat cables, even with alternate ground lines, are not suitable for long distances. Proper ARINC429 approved cabling is recommended. For lab environments standard twisted pair cabling will work.

Dynamic Engineering can manufacture custom cables to your specifications. Please contact sales@dyneng.com with your requirements.

We provide the components. You provide the system. Safety and reliability can be achieved only by careful planning and practice. Inputs can be damaged by static discharge, and by applying voltages exceeding the device specifications.

Terminal Block. We offer a high quality 50 screw terminal block that directly connects to the flat cable. The terminal block mounts on standard DIN rails. [http://www.dyneng.com/HDRterm50.html]

Carriers – Dynamic Engineering has carriers for PCIe, PCI, cPCI and PC104p. Check for VPX. User the Java menu [hover over Embedded Products] to navigate to the format [cPCI etc] and then the function on our website.

Drivers and reference software. Drivers are included with Dynamic Engineering products. Any released driver can be requested for the hardware. In many cases Linux, Windows, and VxWorks are supported and in some cases additional custom drivers for alternate OS are also available. Please check the DDS [Dynamic Data Sheet] online to see what is currently available. Each driver comes with open source reference software that uses the driver to operate the design.

Construction and Reliability

IP Modules were conceived and engineered for rugged industrial environments. IP-429-II is constructed out of 0.062 inch thick high temp FR4 material.

The IP Module connectors are keyed and shrouded with Gold plated pins on both plugs and receptacles. They are rated at 1 Amp per pin, 200 insertion cycles minimum. These connectors make consistent, correct insertion easy and reliable.

The IP is secured against the carrier with four metric M2 stainless steel screws. The heads of the screws are countersunk into the IP. The four screws provide significant protection against shock, vibration, and incomplete insertion. For most applications they are not required.

The IP Module provides a low temperature coefficient of 0.89 W/^{o}C for uniform heat. This is based upon the temperature coefficient of the base FR4 material of 0.31 W/m-^{o}C, and taking into account the thickness and area of the IP. The coefficient means that if 0.89 Watts are applied uniformly on the component side, then the temperature difference between the component side and solder side is one degree Celsius.

Thermal Considerations

The IP-429 design consists of CMOS circuits. Power dissipation due to internal circuitry is very low. It is possible to create higher dissipation with the externally connected logic. If more than one a Watt is required to be dissipated due to external loading, forced air cooling is recommended. With the one degree differential temperature to the solder side of the board, external cooling is easily accomplished.

Warranty and Repair

Please refer to the warranty page for the current warranty offered and options.
http://www.dyneng.com/warranty.html

Service Policy

Before returning a product for repair, verify as well as possible that the suspected unit is at fault. Then call the Customer Service Department for a RETURN MATERIAL AUTHORIZATION (RMA) number. Carefully package the unit, in the original shipping carton if this is available, and ship prepaid and insured with the RMA number clearly written on the outside of the package. Include a return address and the telephone number of a technical contact. For out-of-warranty repairs, a purchase order for repair charges must accompany the return. Dynamic Engineering will not be responsible for damages due to improper packaging of returned items. For service on Dynamic Engineering Products not purchased directly from Dynamic Engineering contact your reseller. Products returned to Dynamic Engineering for repair by other than the original customer will be treated as out-of-warranty.

Out of Warranty Repairs

Out of warranty repairs will be billed on a material and labor basis. Customer approval will be obtained before repairing any item if the repair charges will exceed one half of the quantity one list price for that unit. Return transportation and insurance will be billed as part of the repair and is in addition to the minimum charge.

For Service Contact:

Customer Service Department
Dynamic Engineering
150 DuBois St Suite C
Santa Cruz, CA 95060
831-457-8891
831-457-4793 fax
Internet Address support@dyneng.com

Specifications

Logic Interface:	IP Module Logic Interface
ARINC Interface:	429, other interfaces possible
CLK rates supported:	100, 12-14.5 KHz. Others by special request
Software Interface:	Control Register, ID PROM, Vector Register, and Status Port
Initialization:	Hardware Reset forces all registers to 0. Software Reset Command resets the control register, and FIFO's.
Access Modes:	Word or Byte in I/O Word in ID Space Vectored interrupt
Access Time:	back-to-back cycles in 500ns (8Mhz.) or 125 nS (32 MHz.) control registers. The ARINC 429 data interface has longer access times required.
Wait States:	variable depending on clock rate and what is being accessed. 1-7.
Interrupt:	programmable
DMA:	No Logic Interface DMA Support implemented at this time
Onboard Options:	All Options are Software Programmable
Interface Options:	50 pin flat cable 50 screw terminal block interface User cable
Dimensions:	Standard Single IP Module. 1.8 x 3.9 x 0.344 (max.) inches Type II with passive components on rear of board.
Construction:	FR4 Multi-Layer Printed Circuit, SMT devices, FLASH program storage
Temperature Coefficient:	0.89 W/^{O}C for uniform heat across IP
Power:	Max. 440 mA @ 5V ±12 nominal current required

Order Information

standard temperature range -40 - 85ºC

IP-429-1	IP Module with 1 Tx and 2 Rx ARINC 429 channels
IP-429-2	IP Module with 2 Tx and 4 Rx ARINC 429 channels
IP-429-3	IP Module with 3 Tx and 6 Rx ARINC 429 channels
IP-429-4	IP Module with 4 Tx and 8 Rx ARINC 429 channels
-CC	Boards can be ordered with conformal coating for humid environments.
-CC-T	Extended temperature testing after conformal coating
-ROHS	Standard solder processing is used unless ROHS is requested.
-CableGnd	After programming remove and add components to change JTAG and 3.3 to original GND definitions. Not recommended for new designs, provided for backward compatibility.
Tools for IP-429-x	IP-Debug-Bus - IP Bus interface extender http://www.dyneng.com/ipdbgbus.html IP--Debug-IO - IO connector breakout http://www.dyneng.com/ipdbgio.html HDRterm50 50 position terminal block breakout from ribbon cable http://www.dyneng.com/HDRterm50.html