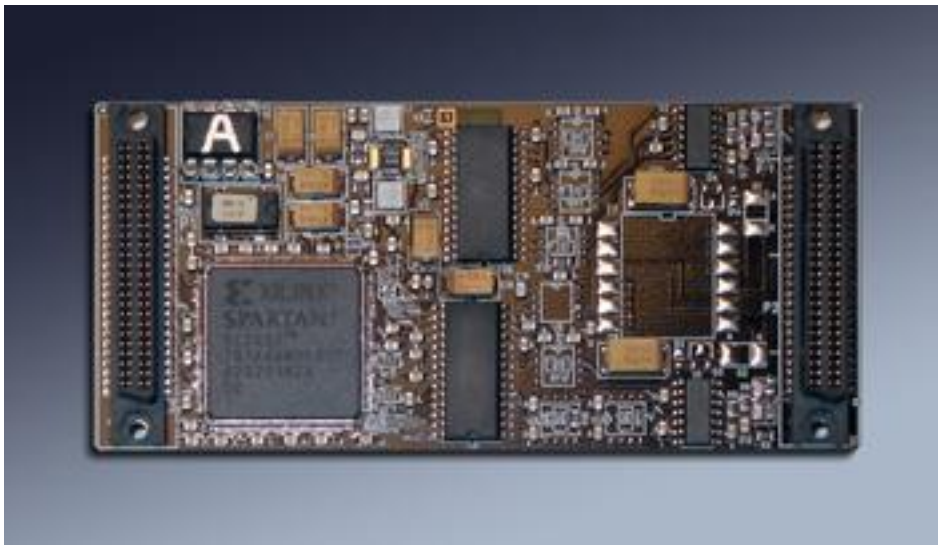
DYNAMIC ENGINEERING
150 DuBois St. Suite B&C, Santa Cruz, Ca. 95060
831-457-8891
sales@dyneng.com
www.dyneng.com
Est. 1988

User Manual

IP-CAN

Controller Area Network Interface
2 independent channels
direct or transformer coupled

IndustryPack® Module



Revision B
Corresponding Hardware: Revision 3, 4
PROM/FLASH revision 3,4
10-2006-1103/4

IP-CAN
Controller Area Network Interface
IndustryPack® Module

Dynamic Engineering has made every effort to ensure that this manual is accurate and complete. Still, the company reserves the right to make improvements or changes in the product described in this document at any time and without notice. Furthermore, Dynamic Engineering assumes no liability arising out of the application or use of the device described herein.

The electronic equipment described herein generates, uses, and can radiate radio frequency energy. Operation of this equipment in a residential area is likely to cause radio interference, in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

Dynamic Engineering's products are not authorized for use as critical components in life support devices or systems without the express written approval of the president of Dynamic Engineering.

This product has been designed to operate with IP Module carriers and compatible user-provided equipment. Connection of incompatible hardware is likely to cause serious damage.

Revised June 14, 2024.

Table of Contents

List of Figures

Product Description

IP-CAN is part of the IndustryPack® “IP” Module family of modular I/O components by Dynamic Engineering. The IP-CAN provides two independent CAN channels. The SJA1000 CAN interface provides the protocol processing and the SJA1041 provides the network physical interface. IP-CAN can be selectively built with direct connect or galvanic coupled bus interface. Please add the correct suffix to select the model you require.

The SJA1000 supports both BasicCAN® and PeliCAN® operation. To support the SJA1000 the IP-CAN has an FPGA which provides the IP interface, clocking to the CAN controller, interrupt processing, and other support functions.

Some of the base features of the SJA1000 are:

· PCA82C200 mode (BasicCAN mode is default)
· Extended receive buffer (64-byte FIFO)
· CAN 2.0B protocol compatibility (extended frame passive in PCA82C200 compatibility mode)
· Supports 11-bit identifier as well as 29-bit identifier
· Bit rates up to 1 Mbits/s
· PeliCAN mode extensions:
– Error counters with read/write access
– Programmable error warning limit
– Last error code register
– Error interrupt for each CAN-bus error
– Arbitration lost interrupt with detailed bit position
– Single-shot transmission (no re-transmission)
– Listen only mode (no acknowledge, no active error flags)
– Hot plugging support (software driven bit rate detection)
– Acceptance filter extension (4-byte code, 4-byte mask)
– Reception of ‘own’ messages (self reception request)
· 24 MHz clock frequency
· Programmable CAN output driver configuration

The internal decoding performed within the FPGA is synchronized to the CAN clock. A frequency doubler is used to allow tighter timing control over the asynchronous interface that the SJA1000 employs. With the tighter timing fewer wait-states are required for an access from the IP bus.

The IP interface is 8 and 32 MHz compatible. When operating in 8 or 32 MHz the

software interface selection bit should be used to optimize the hardware interface. Writes to the SJA1000 are done as pass through to allow faster release on the IP bus. Reads are done with the minimized interface to reduce wait-states.

Each channel has a control register within the FPGA to set the channel specific options. Each channel has a portion of the IP Memory space allocated to allow direct offsets based on the SJA addresses to be utilized. The SJA1000 is an 8 bit device and is accessed with data on D7-0. All addresses are on word boundaries.

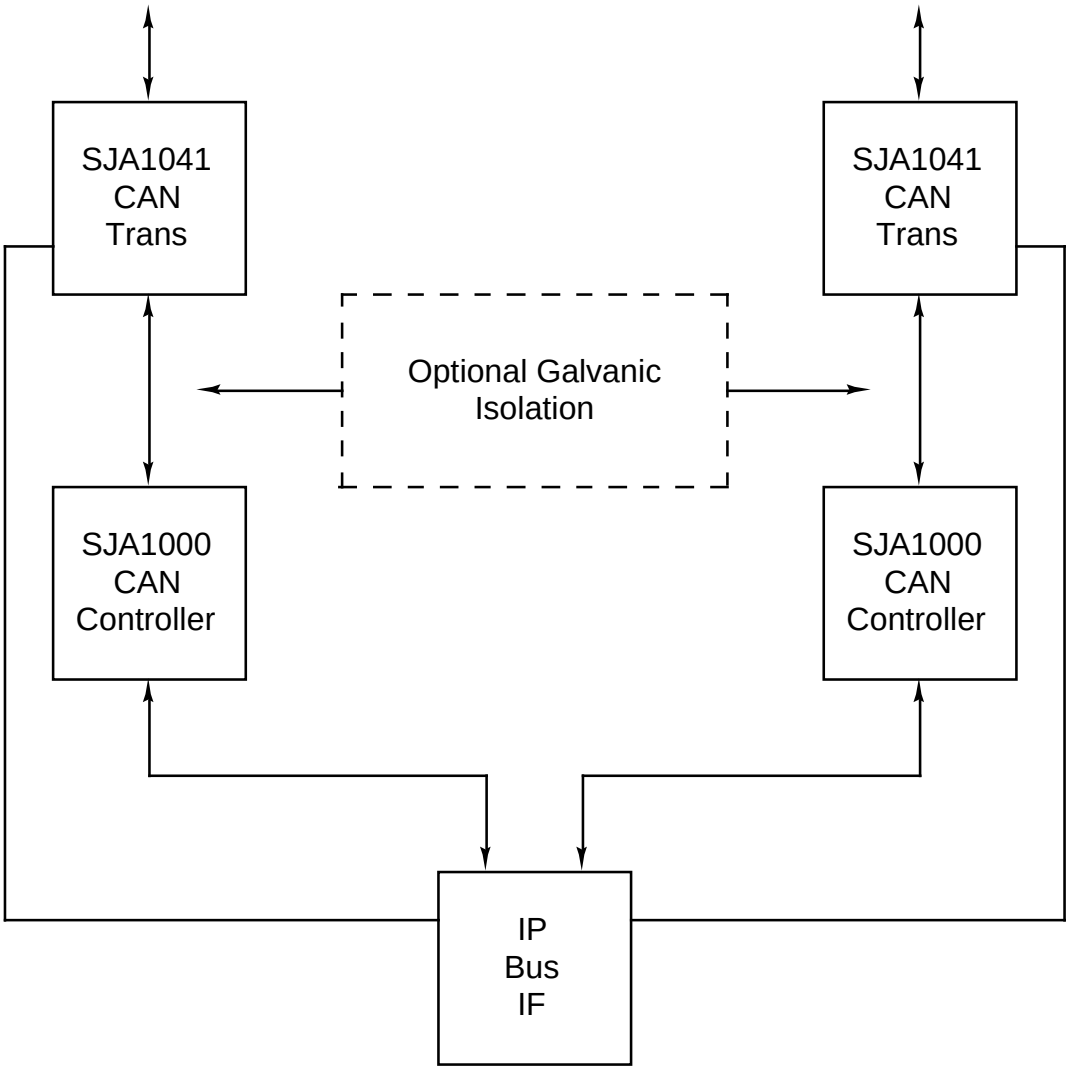


FIGURE 1 IP-CAN BLOCK DIAGRAM

All FPGA configuration registers support read and write operations for maximum software convenience. The SJA1000 has register dependent R/W capability.

IP-CAN conforms to the IndustryPack® standard. This guarantees compatibility with compliant carrier boards. Because the IP-CAN may be mounted on different form factors, while maintaining plug and software compatibility, system prototyping may be done on one Carrier board, with final system implementation on a different one. For example the PCI3IP/PCIe3IP – PCI/PCIe carrier for IP Modules can be used for development in a conventional PC. Later the hardware and software can be ported to the target. https://www.dyneng.com/PCIe3IP.html

With the Dynamic Engineering Windows driver collection for IP and carrier modules a Parent – Child architecture is employed. The IP portion of the driver is directly portable between the various Dynamic Engineering IP carriers [PCI3IP, PCI5IP, PC104pIP, PC104p4IP, cPCI2IP, cPCI4IP, PCIe3IP, PCIe5IP etc]. The parent portion of the driver contains the carrier specific design information. This means that software developed for the IP-CAN on one platform can be directly ported to another. PCI to cPCI for example.

Designers can even make use of the Dynamic Engineering carrier driver for non Dynamic Engineering IP modules using the Generic IP capability built into the parent portion of the driver. IP modules that the carrier driver does not recognize are installed as “generic” and accessed with a address, data interface model. Software developed for the Generic mode can also be ported between modules.

IP-CAN is tested with a combination of internal and external tests. The registers can be tested with R/W tests, the channels can be tested with self-tests, and the channels can be tested with an inter-channel loop test. The ATP for IP-CAN includes internal, same channel and cross channel tests.

Interrupts are supported by IP-CAN. The SJA-1000 has several programmable interrupts for RX, TX and error conditions. Each channel has a separate interrupt and enable. All interrupts are individually maskable and a master interrupt enable is also provided to disable all interrupts simultaneously. In addition a board level SW interrupt is provided for test and software development purposes.

IP-CAN is implemented with the idea of offloading the CPU as much as possible. The SJA1000’s have internal FIFOs to cut down on the number of interrupts to deal with. With address masking only the messages of interest can be received to further reduce CPU traffic. Most registers retain their information between messages allowing for quick reload and send once the initial programming has been implemented.

The SJA1041 is capable of 1MHz operation. The SJA1000 is referenced to a 24 MHz clock to allow full speed operation. Alternate reference frequencies can be arranged by changing the reference oscillator. The SJA1000 supports multiple IO speeds with internal clock division circuitry.

The CANL and CANH signals are tied through transguard devices to the local ground plane to provide over voltage protection.

The transceiver and all signals between the control and FPGA to the transceiver can be isolated with an optional galvanic coupling. A separate power supply is installed with the input isolated from the output to power the transceivers. Each channel is completely separated from the other channel.

GNDA* and GNDB* are reference pins which can be used on the IO connector.

When using the isolated version, 0Ω resistors are installed and the system provides the reference ground. In isolated mode the GNDA and GNDB connections are required.

In direct mode the grounds are tied to the IP ground and the resistors can be substituted with capacitors for AC coupling if desired. The GNDA and GNDB connections are optional with direct mode.

When the isolation is not required the galvanic isolation is bypassed with resistor jumpers.

The CANL and CANH signals are terminated with the standard split parallel termination configuration. The resistor value is 1K with a 63.4 ohm user selectable resistor in parallel. The effective resistance is 60.4Ω when selected. If you're application will terminate the cables separately from the on-board channels leave the optional resistor not enabled. If terminating the cable IP-CAN enable the second resistor.

Address Maps

Address Map Internal

Offset in IO Space

IP_CAN_BASE	0x0000	// 0 base control register offset
IP_CAN_VECTOR	0x0002	// 1 Vector Register offset
IP_CAN_STATUS	0x0004	// 2 Base Status register
IP_CAN_ControlReg0	0x0014	// 10 Channel 0 control register
IP_CAN_ControlReg1	0x0028	// 20 Channel 1 control register

FIGURE 2 IP-CAN INTERNAL ADDRESS MAP

// IP-CAN relative addresses //
// CAN Controller internal accesses are bytes on word boundaries in **Memory space**
//   byte address shown after "//"
// Channel 0/1 is offset based on address 9
/* Duplicate address definitions determined by mode BasiCAN, PeliCAN, Selftest etc.*/

Address Map Channel 0

Offset in Memory Space

```
ModeControlreg0   0x0000   //0 SJA1000 Mode Control Register Address channel 0
commandreg_0      0x0002   //1 Command Register
SR_0              0x0004   //2 Status register
IR_0              0x0006   //3 Interrupt register
IER_0             0x0008   //4 Interrupt Enable
BTR0_0            0x000C   //6 SJA1000 Bus Timing Register 0 channel 0
BTR1_0            0x000E   //7 SJA1000 Bus Timing Register 1 channel 0
OCR_0             0x0010   //8 Output Control register
ALC_0             0x0016   //11 Arbitration Lost Capture
ECC_0             0x0018 //12 Error Code Capture
EWLR_0            0x001a   //13 Error Warning Limit
RXERR_0           0x001c   //14 RX Error Counter
TXERR_0           0x001e   //15 TX Error Counter
FRM_0             0x0020   //16 Tx Buffer (write) Rx Buffer (read) Frame Information
ACR0_0            0x0020   //16 Acceptance Code 0 in RESET mode
ID0_0             0x0022   //17 RX ID, TX ID, or acceptance code 1 in RESET mode
ACR1_0            0x0022   //17 Acceptance Code 1  in RESET mode
ID1_0             0x0024   //18 ID bytes
ACR2_0            0x0024   //18 Acceptance Code 2 in RESET mode
ID2_0             0x0026   //19
ACR3_0            0x0026   //19 Acceptance Code 3 in RESET mode
DATS_0            0x0026   //19 Data start standard frame
ID3_0             0x0028   //20 ID cont. for extended frames
AMR0_0            0x0028   //20 Acceptance Mask 0 in RESET mode
DATE_0            0x002a   //21 Data start extended frame
AMR1_0            0x002a   //21 Acceptance Mask 1 in RESET mode
AMR2_0            0x002c   //22 Acceptance Mask 2 in RESET mode
AMR3_0            0x002e   //23 Acceptance Mask 3 in RESET mode
RMC_0             0x003a   //29 Rx Message Counter (number of msgs. in RX FIFO
RBSA_0            0x003c   //30 Rx Buffer Start Addr. (address of current MSG)
CDR_0             0x003e   //31 Clock Divider
```

```
txfrm1_0          0x0022  //17 TX FRAME identifier 1 SFF
txfrm2_0          0x0024  //18 TX FRAME identifier 2 SFF
txbuffer1_0       0x0026  //19 TX data 1
txbuffer2_0       0x0028  //20 TX data 2
txbuffer3_0       0x002a  //21 TX data 3
txbuffer4_0       0x002c  //22 TX data 4
txbuffer5_0       0x002e  //23 TX data 5
txbuffer6_0       0x0030  //24 TX data 6
txbuffer7_0       0x0032  //25 TX data 7
txbuffer8_0       0x0034  //26 TX data 8
txbuffer11_0      0x0036  //27
txbuffer12_0      0x0038  //28

txframeinford_0   0x00c0  //96
txbufferfrm1_0    0x00c2  //97
txbufferfrm2_0    0x00c4  //98
txbufferrd1_0     0x00c6  //99
txbufferrd2_0     0x00c8  //100
txbufferrd3_0     0x00ca  //101
txbufferrd4_0     0x00cc  //102
txbufferrd5_0     0x00ce  //103
txbufferrd6_0     0x00d0  //104
txbufferrd7_0     0x00d2  //105
txbufferrd8_0     0x00d4  //106
txbufferrd11_0    0x00d6  //107
txbufferrd12_0    0x00d8  //108

rxfrm1_0          0x0022  //17 RX FRAME identifier 1 SFF
rxfrm2_0          0x0024  //18 RX FRAME identifier 2 SFF
rxbuffer1_0       0x0026  //19 RX data 1
rxbuffer2_0       0x0028  //20 RX data 2
rxbuffer3_0       0x002a  //21 RX data 3
rxbuffer4_0       0x002c  //22 RX data 4
rxbuffer5_0       0x002e  //23 RX data 5
rxbuffer6_0       0x0030  //24 RX data 6
rxbuffer7_0       0x0032  //25 RX data 7
rxbuffer8_0       0x0034  //26 RX data 8
rxbuffer11_0      0x0036  //27
xbuffer12_0       0x0038  //28
```

FIGURE 3 IP-CAN CHANNEL 0 ADDRESS MAP

Address Map Channel 1

Offset in Memory Space

ModeControlreg1	0x0200	// SJA1000 Mode Control Register Address channel 1
commandreg_1	0x0202	//1 Command Register
SR_1	0x0204	//2 Status register
IR_1	0x0206	//3 Interrupt register
IER_1	0x0208	//4 Interrupt Enable
BTR0_1	0x020c	//6 SJA1000 Bus Timing Register 0 channel 1
BTR1_1	0x020e	//7 SJA1000 Bus Timing Register 1 channel 1
OCR_1	0x0210	//8 Output Control register
ALC_1	0x0216	//11 Arbitration Lost Capture
ECC_1	0x0218	//12 Error Code Capture
EWLR_1	0x021a	//13 Error Warning Limit
RXERR_1	0x021c	//14 RX Error Counter
TXERR_1	0x021e	//15 TX Error Counter
FRM_1	0x0220	//16 Tx Buffer (write) Rx Buffer (read) Frame Information
ACR0_1	0x0220	//16 Acceptance Code 0 in RESET mode
ID0_1	0x0222	//17 RX ID, TX ID, or acceptance code 1 in RESET mode
ACR1_1	0x0222	//17 Acceptance Code 1 in RESET mode
ID1_1	0x0224	//18 ID bytes
ACR2_1	0x0224	//18 Acceptance Code 2 in RESET mode
ID2_1	0x0226	//19
ACR3_1	0x0226	//19 Acceptance Code 3 in RESET mode
DATS_1	0x0226	//19 Data start standard frame
ID3_1	0x0228	//20 ID cont. for extended frames
AMR0_1	0x0228	//20 Acceptance Mask 0 in RESET mode
DATE_1	0x022a	//21 Data start extended frame
AMR1_1	0x022a	//21 Acceptance Mask 1 in RESET mode
AMR2_1	0x022c	//22 Acceptance Mask 2 in RESET mode
AMR3_1	0x022e	//23 Acceptance Mask 3 in RESET mode
RMC_1	0x023a	//29 Rx Message Counter (number of msgs. in RX FIFO
RBSA_1	0x023c	//30 Rx Buffer Start Addr. (address of current MSG)
CDR_1	0x023e	//31 Clock Divider

txfrm1_1	0x0222	//17 TX FRAME identifier 1 SFF
txfrm2_1	0x0224	//18 TX FRAME identifier 2 SFF
txbuffer1_1	0x0226	//19 TX data 1
txbuffer2_1	0x0228	//20 TX data 2
txbuffer3_1	0x022a	//21 TX data 3
txbuffer4_1	0x022c	//22 TX data 4
txbuffer5_1	0x022e	//23 TX data 5

```
txbuffer6_1          0x0230  //24 TX data 6
txbuffer7_1          0x0232  //25 TX data 7
txbuffer8_1          0x0234  //26 TX data 8

txframeinford_1      0x02c0  //96
txbufferfrm1_1       0x02c2  //97
txbufferfrm2_1       0x02c4  //98
txbufferrd1_1        0x02c6  //99
txbufferrd2_1        0x02c8  //100
txbufferrd3_1        0x02ca  //101
txbufferrd4_1        0x02cc  //102
txbufferrd5_1        0x02ce  //103
txbufferrd6_1        0x02d0  //104
txbufferrd7_1        0x02d2  //105
txbufferrd8_1        0x02d4  //106

rxfrm1_1             0x0222  //17 RX FRAME identifier 1 SFF
rxfrm2_1             0x0224  //18 RX FRAME identifier 2 SFF
rxbuffer1_1          0x0226  //19 RX data 1
rxbuffer2_1          0x0228  //20 RX data 2
rxbuffer3_1          0x022a  //21 RX data 3
rxbuffer4_1          0x022c  //22 RX data 4
rxbuffer5_1          0x022e  //23 RX data 5
rxbuffer6_1          0x0230  //24 RX data 6
rxbuffer7_1          0x0232  //25 RX data 7
rxbuffer8_1          0x0234  //26 RX data 8
```

FIGURE 4 IP-CAN CHANNEL 1 ADDRESS MAP

Programming

Programming IP-CAN requires only the ability to read and write data from the host. The base address refers to the first user address for the slot in which the IP is installed.

Depending on the software environment it may be necessary to set-up the system software with IP-CAN "registration" data. The Dynamic Engineering Driver operates in a “plug and play” mode with a parent ⇔ child architecture.

In order to receive and or transmit data the software is required to enable the controller for the channel(s) of interest. The initialization procedure is a multi-step process determined by the SJA1000, and the desired mode of operation. It is recommended that the user who is writing their own driver refer to the Phillips Application notes for the SJA1000.

Interrupts are used to help manage the data transfer process. When a programmed transfer is completed the interrupt can be generated to alert the host to program a new transfer. The transfers are independent for each channel allowing the CPU interaction to be minimized.

The Dynamic Engineering IP-CAN driver for Windows manages the interaction and can set-up the transfer for you. The driver is easily integrated into a Visual Studio C programming environment. Please refer to the driver manual for more information.

Alignment: With Dynamic Engineering IP Modules; registers are based on 16 bit words whether there are 16 bits present or not. D0 in the register is always aligned with D0 on the IP connector. On Dynamic Engineering carriers D0 on the IP connector also aligns with D0 on the bus when LW aligned, and with D16 on the bus when the D31-D16 portion of the bus is active. DE carriers provide byte and word swapping to move D0 to be aligned based on whatever is the protocol being used on the main bus. Your carrier and or processor may introduce alternate alignments which will require byte and or word swapping to compensate.

Dynamic Engineering PCI based carriers provide byte, word and LW interaction with the installed IP. LW are turned into 2 – 16 bit accesses with improved performance compared to two separate accesses. Dynamic Engineering PCIe carriers add 64 bit access capability transferring 4 – 16 bit words per access.

Register Definitions

IP_CAN_BASE

IP_CAN_BASE 0x0000 // 0 base control register offset

BASE Control Register

DATA BIT	DESCRIPTION
15-5	Spare
4	CLK8_32
3-0	Spare

FIGURE 5 IP-CAN BASE CONTROL REGISTER BIT MAP

CLK8_32 when ‘0’ selects the 32 MHz clock rate and when ‘1’ selects 8 MHz. The time to access the CAN interface is fixed and requires fewer clocks at 8 MHz than when running at 32 MHz. The default is 32 MHz operation for the added wait states. Setting this bit when running at 8 Mhz will improve performance. Clear this bit when running at 32 MHz for proper operation. The IP reference clock rate is set on the carrier. All Dynamic Engineering carriers have slot by slot clock selection. If 32 MHz is available it is the recommended clock rate to use. Even with the added wait-states 32 MHz will provide better performance overall.

IP_CAN_VECTOR

IP_CAN_Vector 0x0002 // 1 IP vector port

Vector Port

DATA BIT	DESCRIPTION
15-8	Spare
7-0	switch in

FIGURE 6 IP-CAN VECTOR BIT MAP

If the system uses a vectored interrupt approach then the vector port should be initialized to the vector value assigned to this device. IP-CAN is designed to be used as vectored or auto-vectored. In auto-vectored situations this port is unused. The Status port can be used to determine the source of any pending interrupts from IP-CAN.

IP_CAN_STATUS

IP_CAN_STATUS 0x0004 // 2 base Status register

CONTROL RX

DATA BIT	DESCRIPTION
15-8	Spare
7	CH1_ERR
6	CH1_INT
5	CH0_ERR
4	CH0_INT
3-1	Spare
0	LOC_INT

FIGURE 7 IP-CAN INTERRUPT STATUS BIT MAP

LOC_INT is set when INTFORCE, CH0_INT or CH1_INT are set and the corresponding mask is enabled. This bit is cleared by dealing with the interrupt source.

CH0_INT and CH1_INT are the interrupts from channel 0 and channel 1 CAN interface devices. The bits are before the interrupt mask and will be set even if the channel interrupt mask is not set to allow polled operation.

The interrupt signals are inverted in hardware to provide and active high status. Each interrupt can be enabled to be set from a variety of sources. Please refer to the IER, and IR registers for channel based interrupt control.

CH0_ERR and CH1_ERR are status bits from the transceivers. Please refer to the SJA1041 documentation for more information about the use of this bit. The ERR bits are not inverted between the 1041 and the FPGA status register.

IP_CAN_ControlReg0, IP_CAN_ControlReg1

IP_CAN_ControlReg0,1 0x0014,0x0028 // 10,20 direction and termination register

CONTROL DIR_TERM REGISTER

DATA BIT	DESCRIPTION
15	Interrupt Mask
14	1041 Stand-by
13	1041 Enable
12	1000 Enable
11	Termination Control
10	Force Interrupt
9-0	Spare

FIGURE 8 IP-CAN LOCAL CONTROL BIT MAP

1000 Enable is tied to the /RST line on the SJA1000. When '0' the SJA1000 is held in reset. When '1' the SJA1000 can operate normally. Please note that there is a "Reset" mode defined by the SJA1000 which is not controlled by this bit. Refer to the ModeControlreg within the SJA1000 for more mode information.

1041 Enable must be set high for the transceiver to operate. 1041 Enable is directly tied to the EN pin on the channel transceiver.

1041 Stand-by is tied to /STB. Stand-by when low will cause the 1041 to go into one of the lower power states. 1041 Stand-by must be set to '1' for normal operation.

Interrupt Mask when set '1' enables the channel interrupt to cause an IP level interrupt. When '0' the interrupt from the channel can still be used as a status bit, and the interrupt will not cause an IP level interrupt.

Termination Control when set '1' enables the second termination resistors to provide a parallel value of 60.4 ohms. When '0' the second termination resistors are "disabled" resulting in 1K ohm termination values.

Force Interrupt when set causes an interrupt request to the host. The Interrupt Mask must be enabled for the interrupt to be asserted. This bit is normally used for software development and hardware test. Normally cleared to '0'.

Recommended Initialization sequence.

```
*pcontrolreg0  = 0x6000; // pointer to IP-CAN register ch 0, reset, Enabled, not Standby
*pcontrolreg0  = 0x7000; // pointer to IP-CAN register ch 0, not reset, Enabled, not Standby
*pcontrolreg1  = 0x6000; // pointer to IP-CAN register ch 0, reset, Enabled, not Standby
*pcontrolreg1  = 0x7000; // pointer to IP-CAN register ch 0, not reset, Enabled, not Standby
```

Interrupts

IP-CAN interrupts are treated as auto-vectored. When the software enters into an exception handler to deal with an IP-CAN interrupt the software must read the status register(s) to determine the cause(s) of the interrupt, clear the interrupt request(s) and process accordingly. Power on initialization will provide a cleared interrupt request and interrupts disabled.

The interrupt is mapped to INT0 on the IP connector, which is mapped to a system interrupt via the host [carrier] device. The source of the interrupt is obtained by reading the Interrupt Status register. The status remains valid until that bit in the status register is cleared.

When an interrupt occurs, the Master channel interrupt enable should be cleared and the status register read to determine the cause of the interrupt. Next perform any processing needed to remove the interrupting condition, clear the status and enable the channel interrupt again.

The individual enables operate after the interrupt holding latches, which store the interrupt conditions for the CPU. This allows for operating in polled mode simply by monitoring the Interrupt Status register.

Loop-back

The Engineering kit has reference software, which includes an external loop-back test. The test requires an external cable with the following pins connected. Channel 0 to Channel 1.

Rev 1-3			Rev 4		
SIGNALs	0	1	SIGNALs	0	1
CANL	1	24		1	23
CANH	2	25		2	24

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 be 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 IPs I/O space, or at IP-base + $80. Macintosh drivers use the ID PROM automatically.

Standard data in the ID PROM on the IP-CAN is shown in the figure below. For more information on IP ID PROMs refer to the IP Module Logic Interface Specification.

Address	Data	
01	ASCII "I"	($49)
03	ASCII "P"	($50)
05	ASCII "A"	($41)
07	ASCII "H"	($48)
09	Manufacturer ID	($1E)
0B	Model Number	($0A) IP-CAN
0D	Revision	($C0)
0F	reserved	($00) Customer Number
11	Driver ID, low byte	($00) Design Number
13	Driver ID, high byte	($00)
15	No of extra bytes used	($0C)
17	CRC	($BC)

FIGURE 9 IP-CAN ID PROM

IP-CAN Logic Interface Pin Assignment

The figure below gives the pin assignments for the IP Module Logic Interface on the IP-CAN. Pins marked n/c below are defined by the specification, but not used on the IP-CAN. 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		INTREG0*		17		42	
	D14		A4		18		43
D15		n/c		19		44	
	BS0*		A5		20		45
BS1*		n/c		21		46	
	n/c		A6		22		47
n/c		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 10 IP-CAN LOGIC INTERFACE

IP-CAN IO Pin Assignment Rev 1-3

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

CH0_CANL	GNDA*	1		26		
	CH0_CANH	GNDA*		2		27
			3		28	
				4		29
			5		30	
				6		31
			7		32	
				8		33
			9		34	
				10		35
			11		36	
				12		37
			13		38	
				14		39
			15		40	
				16		41
			17		42	
				18		43
			19		44	
				20		45
			21		46	
				22		47
			23		48	
	CH1_CANL	GNDB*		24		49
CH1_CANH	GNDB*		25		50	

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

FIGURE 11 IP-CAN CONNECTOR PINOUT REV 1-3

IP-CAN IO Pin Assignment Rev 4

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

CH0_CANL	GNDA*	1	26
CH0_CANH	GNDA*	2	27
GNDA*		3	28
GNDA*		4	29
		5	30
		6	31
		7	32
		8	33
		9	34
		10	35
		11	36
		12	37
		13	38
		14	39
		15	40
		16	41
		17	42
		18	43
		19	44
		20	45
		21	46
GNDB*		22	47
CH1_CANL	GNDB*	23	48
CH1_CANH	GNDB*	24	49
GNDB*	GNDB*	25	50

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

FIGURE 12 IP-CAN CONNECTOR PINOUT REV 4

Applications Guide

Interfacing

The pin-out tables are displayed with the pins in the same relative order as the actual connectors. The pin definitions are defined with noise immunity in mind. The pairs should be connected with standard CAN cabling or twisted pair wiring for best results.

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.

Power all system power supplies from one switch. Connecting external voltage to the IP-CAN when it is not powered can damage it, as well as the rest of the host system. This problem may be avoided by turning all power supplies on and off at the same time. Alternatively, use the isolated version.

Keep cables short. Flat cables, even with alternate ground lines, are not suitable for long distances. IP-CAN does not contain special input protection. The connector is pinned out for a standard Header cable to be used. The twisted pairs are defined to match up with the IP-CAN pin definitions. It is suggested that this standard cable be used for most of the cable run.

Custom cables can be manufactured with discrete wire header and direct connection to your mating equipment.

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

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, or by applying voltage outside of the CAN devices rated voltages.

Construction and Reliability

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

Through hole and surface mounting of components are used. IC sockets use screw machine pins. High insertion and removal forces are required, which assists in the retention of components. If the application requires unusually high reliability or is in an environment subject to high vibration, the user may solder the corner pins of each socketed IC into the socket, using a grounded soldering iron.

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 .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 .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-CAN design consists of CMOS circuits. The power dissipation due to internal circuitry is very low. It is possible to create a higher power dissipation with the externally connected logic. If more than one 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 on our website 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 B&C
Santa Cruz, CA 95060
831-457-8891
support@dyneng.com

Specifications

Host Interface:	IP Module 8 and 32 MHz capable
CAN Interface:	2 Channels with full protocol support
Tx Data rates generated:	Programmable. 24 MHz reference to SJA1000
Software Interface:	Control Registers, Status Ports
Initialization:	Hardware Reset forces all registers to 0.
Access Modes:	IO, Memory, ID, INT spaces (see memory map)
Wait States:	minimized based on programmed clock rate
Interrupt:	Channel interrupt for each CAN channel Software interrupt
Onboard Options:	Most Options are Software Programmable. Build options for isolated IO and reference grounds.
Interface Options:	IP IO connector routed through IP Carrier. Twisted pair cable recommended
Dimensions:	Type 1 IP Module.
Construction:	High temp FR4 Multi-Layer Printed Circuit, Through Hole and Surface Mount Components. Programmable parts are socketed.
Temperature Coefficient:	89 W/^{o}C for uniform heat across IP
Power:	Typical **86** mA @ 5V in Direct Coupled Mode running channel to channel test

Order Information

IP-CAN	IP Module with 2 CAN channels SJA1000 / SJA1041A per channel
-ISO	Add for Isolation option
Tools for IP-CAN	IP-Debug-Bus - IP Bus interface extender https://www.dyneng.com/ipdbgbus.html IP--Debug-IO - IO connector breakout https://www.dyneng.com/ipdbgio.html
Eng Kit–IP-CAN	IP-Debug-IO - IO connector breakout IP-Debug-Bus - IP Bus interface extender Technical Documentation, IP-CAN Reference test software Data sheet reprints are available from the manufacturer's web site

***Note**: The Engineering Kit is strongly recommended for first time* IP-CAN *purchases.*