DDC SDC-36015

Multi-Channel Converter Card (Base Card Only)

ILC DATA DEVICE CORPORATION®

CERTIFIED AND QUALIFIED TO MIL-STD-1772

LEADERS IN SYNCHRO AND DATA CONVERSION PRODUCTS

SDC-36015
Six Channel
Synchro/Resolver-to-Digital
Converter Card
User's Guide

ILC DATA DEVICE CORPORATION®

105 Wilbur Place, Bohemia, New York 11716

For technical support: 1-800-DDC-1772, ext. 7383 (outside NY)
1-800-245-3413, ext. 7383 (in Canada)

Headquarters - Tel: (516) 567-5600, ext. 7383
Fax: (516) 567-7358 or (516) 563-4331

West Coast - Tel: (714) 895-9777, Fax: (714) 895-4988

Europe - Tel: 44 (1635) 40158, Fax: 44 (1635) 32264

Asia/Pacific - Tel: 81 (33) 814-7688, Fax: 81 (33) 814-7689

REV. A FEBRUARY 1995

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

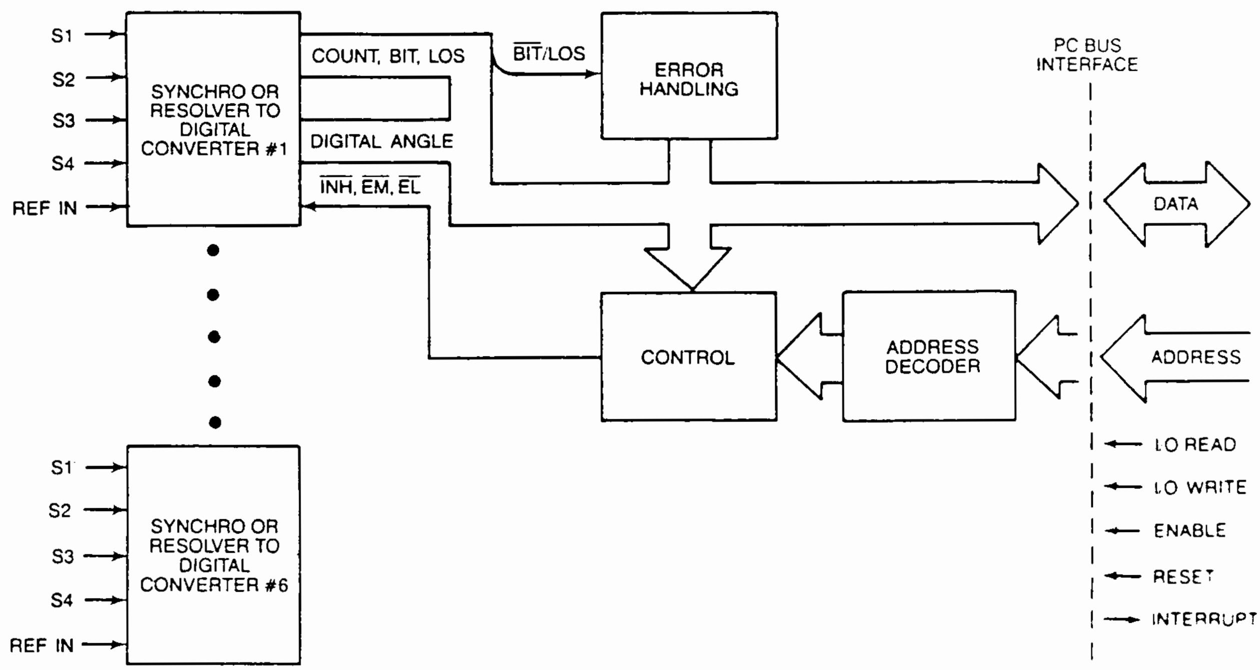


FIGURE 1. SDC-36015 BLOCK DIAGRAM

Description

Thank you for purchasing ILC Data Device Corporation's SDC-36015 Synchro/Resolver-to-Digital IBM PC® Card. This guide contains the following: a card description including converter characteristics, configuring the hybrid converters, reference requirements, configuring the SDC-36015 card, output formats and jumper configurations; interface requirements both external and internal; installation and use of the supplied software to develop custom applications.

Hardware

The SDC-36015, is a register based, full size IBM PC card designed for one to six channels of resolver-to-digital or synchro-to-digital conversion. The SDC-36015 uses either the RDC-19200 Series (for resolver inputs) or SDC-19204 Series (for synchro inputs) hybrid converters.

The converters feature jumper programmable resolution (10-,12-,14-,or 16-bits), programmable high or low bandwidth, and a high quality velocity output voltage (VEL). Output angle information is provided in two 8-bit bytes to the computer.

All information generated by the SDC-36015 is I/O mapped in four ram locations and is available to the computer with a read command. This includes digital angle, 4-bits of turns counting, Built-in-Test (BIT) and Loss-of-Signal (LOS) information for each channel.

For reference we have included a detailed Functional Block Diagram of the SDC-36015 in Appendix D of this guide.

Software

The SDC-36015 is a port I/O addressed PC card. Software communication with the card is done by reading and writing to 4, 8-Bit registers. Any operating environment which implements the CPU port input and port output instruction operations can access the SDC-36015.

The software included with the SDC-36015 card simplifies the communication by doing the necessary bit manipulation and register mapping. This higher level abstraction allows for the user's concentration on the application software, not on interface level code.

This library was created for use under both C and Pascal compilers. A complete description of the routines and techniques used for creating and compiling your application under "C" and Pascal are included in this document.

This library was also designed to be a *National Instruments LabWindows INSTRUMENT MODULE*. This module simplifies creation of LabWindows applications in National Instrument's environment.

Included on the distribution disk in a complete menu driven software package to use the SDC-36015 "out of the box." A description on the use of this software is given in the "Menu Software" section of this guide.

Refer to Appendix A for detailed instructions on how to use the routines. Source code listings which compile under Borland "C," Turbo Pascal, and *LabWindows* are included in this guide.

SDC-36015 DESIGN

Theory of Operation

The SDC-36015 is a full sized 8 bit EISA/ISA interface card for use in the IBM PC/AT/x86 compatible computer. The SDC-36015 card can accommodate the RDC-19200, RDC-19201, RDC-19202, RDC-19203, SDC-19204, SDC-19206, and SDC-14560 hybrid converters.

Type II Converters

A discussion of Type II converters follows. (An in-depth explanation of the synchro/resolver conversion process is available in the **SYNCHRO/RESOLVER CONVERSION HANDBOOK** published by ILC Data Device Corporation.)

The up-down counter portion of the synchro converter like any counter is functionally an incremental integrator. Therefore, the tracking converter constitutes in itself a closed-loop servomechanism (continuously attempting to null the error to zero) with two lags...two integrators in series. This is a "Type II" servo loop, which has decided advantages over Type I or Type 0 loops, as we shall see.

In order to appreciate the value of the Type II servo behavior of this tracking converter, consider first that the shaft of the synchro is not moving. Ignoring inaccuracies, drifts, and the inevitable quantizing error (e.g., $\pm 1/2$ LSB), the error should be zero, and the digital output represent the true shaft angle of the synchro.

Now, start the synchro shaft moving, and allow it to accelerate uniformly. During the acceleration, an error will develop, because the converter cannot instantaneously respond to the change in angular velocity. However, since the VCO (Voltage Controlled Oscillator) in the converter is controlled by an integrator, whose output is the integral of the error, the greater the lag (between angle θ - the angle to be digitized - and angle Φ - the angle stored in digital form in the up-down counter), the faster will the counter be called upon to "catch up." When this "catch up" velocity becomes constant at V, the VCO will have settled to a ratio of counting that exactly corresponds to the rate of change in θ per unit time and instantaneously θ-Φ, in other words, $d\Phi/dt$ will always equal ("track") $d\theta/dt$ without a velocity or position error. The only errors, therefore, will be momentary (transient) errors, during acceleration or deceleration. Furthermore, the information produced by the tracking converter is always "fresh," being continually updated, and always available at the output of the counter. Since $d\theta/dt$ tracks the input velocity it can be brought out as a velocity (VEL) or tracking rate signal which is of sufficient linearity in modern converters to eliminate the need for a tachometer in many systems. Suitably scaled it can be used as the velocity feedback signal to stabilize the servo system or motor.

Hybrid Converter Summary

TABLE 1 summarizes the differences between the converter options available for the SDC-36015 card manufactured by DDC. It is provided for information only to the user to clarify the converter capabilities on the DDC Resolver-to-Digital or Synchro-to-Digital card.

TABLE 1. HYBRID CONVERTER OPTIONS

FEATURE	SDC-14560	RDC/SDC-1920X
Selected Channel Angle	X	X
Programmable Resolution	X	X
Programmable Bandwidth		X
Velocity	X	X
Turns Count		X
LOS (Loss-of- Signal) Indicator		X
BIT (Built-In-Test)	X	X

Computer Interface Circuitry

The computer interface circuitry consists of the Status/Error Map Byte, Status/Count Byte, LSB byte, MSB byte, read registers and Control write register.

The Interface Circuit is the physical interface to the computer backplane and consists of the data bus transceivers along with the address and control line receivers. These components take care of exchanging information between the SDC-36015 and the backplane of the computer.

The Base Address Switches are the physical switches used to set the Base Address of the SDC-36015. Setting the Base Address Switches establishes the I/O address space where the computer can find the SDC-36015. These switches are preset at the factory to the base address of 300h (HEX 300), but they can be set to any base address from 0h to 3FCh. Before installing the card in the computer, be careful how you configure the Base Address Switches; this will ensure that no other devices share the same base address because unexpected computer operation will occur if there is a conflict. Leave selection of the base address for the card to someone skilled in the operation, and configuration, of the specific computer into which you are installing the SDC-36015. In general it is not a good idea to set the base address to values less than 300h or more than 390h because the computer uses this I/O address space internally to control devices like the hard disk drive, the DMA controller, the monitor, and other interfaces that are best left alone.

The Address Decoder compares the base address of the card to the I/O requests from the computer. When the address of the computer's request matches the base address of the card then the following occurs: The card interface circuitry is enabled; the LSBs of the I/O address are decoded; and the card generates the appropriate register read/write signals that cause data transfers to take place.

When a valid address is present in the 4 byte address space, and the I/O write strobe is active, the data bus is latched into the command register. At the end of the write cycle, the latched command is enabled signalling the SDC-36015 converters. This include inhibit control, counter reset, irq reset, and active converter selection. The inhibit control line of the latch drives all the converters. The counter reset line is connected to the CLEAR on all latch connected to the Up/Down counter, BIT, and LOS lines. The converter select lines is multiplexed to enable the LSB, MSB, or TURNS/ERROR latch of a specific converter for reading.

When a valid address is present and the I/O Read line is active, the corresponding latch/converter is enabled to drive the 8 bit data bus. When the read cycle is complete, the latch is tri-stated.

CONFIGURING THE SDC-36015

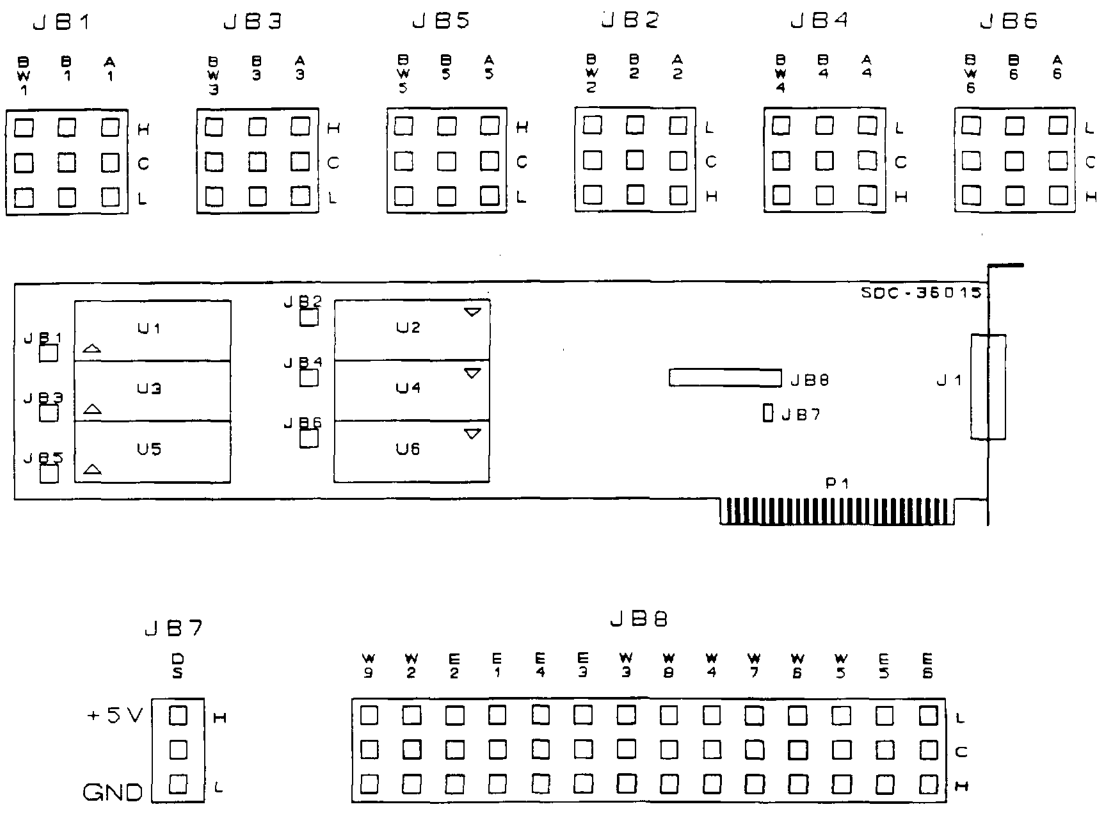


FIGURE 2. SDC-36015 MECHANICAL ASSEMBLY OUTLINE

Installing the Hybrids

Since the SDC-36015 card can accommodate the RDC-19200, RDC-19201, RDC-19202, RDC-19203, SDC-19204, SDC-19206, and SDC-14560 hybrid converters, there are two sockets per channel. Use the 36 pin sockets for the SDC-1456X series of converters and the 40 pin sockets for the RDC/SDC-1920X series of converters.

WARNING: The hybrid converters and the SDC-35015 PC board are static sensitive; ESD (Electro-Static-Discharge) procedures must be followed at all times to prevent damage to the hybrids and/or the card. Warranties do not cover improperly handled parts.

Please note that the hybrid pin 1 is labeled for each channel on the PC board; the hybrid pin 1 is indicated by a triangle on the package. Be very careful installing each hybrid since an incorrectly installed hybrid can be damaged. Converters U1, U3, and U5 have pin 1 in the lower lefts corner when installed properly. U2, U4, and U6 have pin 1 to the upper right corner when they are installed correctly. See FIGURE 2 for clarification; the triangle shown on U1 through U6 indicates pin 1 on the outline drawing.

Setting the Resolution

The RDC-1920X series of converters and the SDC-14560 have pin programmable resolutions depending on the logic input levels of pins 7 and 8 (pins A and B respectively) of the hybrid. To insure that no race conditions exist between counting and changing the resolution, inputs A and B are latched internally on the trailing edge of CB (converter busy).

Important: A logic "1" for the resolution/bandwidth jumpers for channels 1, 3, and 5 is towards the top of the card. A logic "1" for the resolution/bandwidth jumpers for channels 2, 4, and 6 is towards the bottom of the card. A logic "1" requires a jumper between points C-H, and a logic "0" requires the jumper between points C-L.(See Mechanical Outline.)

Jumper blocks 1-6 shown on the SDC-36015 Mechanical Outline are used to program the resolution and bandwidth of the hybrid converters.

Note: Remove all power to the SDC-36015 card before removing or installing any jumpers.

Set the resolution jumper switches A and B for each channel being used prior to installing the card in the PC. The resolution jumper settings for 10-, 12-, 14-, and 16-bit accuracy are shown in TABLE 2.

TABLE 2. RESOLUTION JUMPER SETTINGS

JUMPER AX*	JUMPER BX*	RESOLUTION
0	0	10 BIT
1	0	12 BIT
0	1	14 BIT
1	1	16 BIT

0 = Jumper disconnected.
1 = Jumper connected.

* **X** = channel (i.e. A1 = jumper A for channel 1)

Setting the Bandwidth

The bandwidth can only be set on the RDC/SDC-1920X series of converters. Each channel can be programmed for High or Low bandwidth. The bandwidth jumper for each channel is part of the jumper blocks listed in TABLE 3. Each channel jumper is labeled BW1, BW2, etc. depending on the particular channel it controls. (See the Mechanical Outline) Set the bandwidth jumper(s) to a logic "1" for high bandwidths (up to 530 Hz) or a logic "0" for low bandwidths (up to 130 Hz).

The bandwidth options for the RDC/SDC-1920X converters are listed in TABLE 3.

TABLE 3. BANDWIDTH OPTIONS

PRODUCT	HIGH BW	LOW BW
RDC-19200	530 Hz	130 Hz
RDC-19201	530 Hz	130 Hz
RDC-19202	530 Hz	130 Hz
RDC-19203	530 Hz	130 Hz
SDC-19204	53 Hz	13 Hz
SDC-19206	53 Hz	13 Hz

Setting the Drive Select

In addition to setting the resolution and bandwidth jumpers for the hybrid converters, it is also necessary to set the Drive Select (DS) jumper (Jumper Block 7 on the Mechanical Outline) for the appropriate converter type before applying power to the SDC-36015 card. If the converters are the RDC/SDC-1920X series, the Drive Select (DS) jumper must be set to the +5 V position. If the SDC-1456X series of converters are being used, the DS jumper must be set to the GND position.

TABLE 4. DRIVE SELECT JUMPER

CONVERTER	(DS) JUMPER
RDC/SDC-1920X	+5 V
SDC-1456X	GND

Port Address Selection

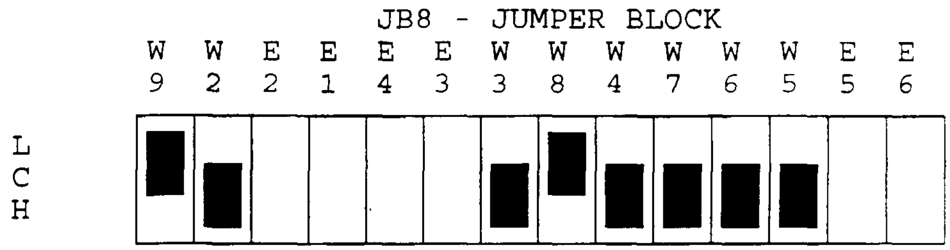
Communication with the SDC-36015 card is done via the PC EISA/ISA backplane through the P1 connector. All I/O is done using four Read/Write Port registers mapped in a Base Address + Offset fashion. The Base address is setting jumper connections on the JB8 jumper block (See mechanical outline for location information). The SDC-36015 is preset at the factory to HEX address 300. In this configuration, the SDC-36015 will occupy the address space of 300h to 303h. The Base Address can be changed to any address from 000h to 3FCh in increments of 4 using TABLE 5 as a reference.

Be careful how you configure the base address jumpers; this will ensure that no other devices share the same base address because unexpected computer operation will occur if there is an address conflict. Leave selection of the base address for the card to someone skilled in the operation, and configuration, of the specific computer into which you are installing the SDC-36015. In general, it is not a good idea to set the base address to values less than 300h or more than 390h because the computer uses this I/O address space internally to control devices like the hard disk drive, the DMA controller, the monitor, and other interfaces that are best left alone.

TABLE 5. BASE ADDRESS SELECTION

JUMPER	ADDRESS	ON (CL)	OFF (CH)	VALUE
W9	A9	X		HEX 3
W8	A8	X		
W7	A7		X	HEX 8
W6	A6		X	
W5	A5		X	
W4	A4		X	
W3	A3		X	HEX 0
W2	A2		x	
-	A1	-	-	
-	A0	-	-	

The example below sets the SDC-36015 PC card to the Port Address of 300h. Please note the W9-W2 jumper positions.



Operating With Less Than Six Channels

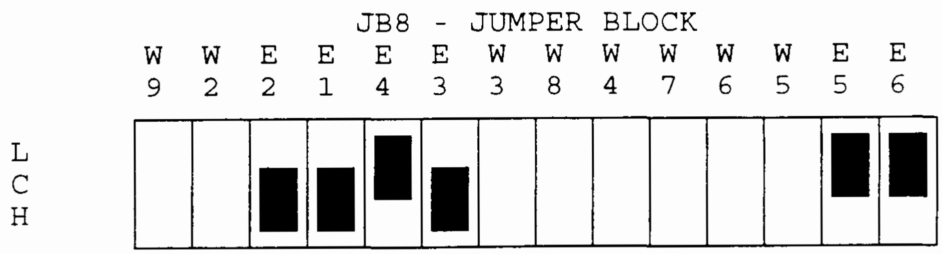
If the SDC-36015 is to be used with less than all six channels, or, if it is required that a specific channel does not set the INTERRUPT line, then that/those channel(s) should have the interrupt function disabled.

To disable the INTERRUPT, use jumper switches E1 to E6 (for channels 1 through 6). The interrupt is disabled when the jumper is in the CL (towards the top of the board). The BIT and LOS information going to the data bus and the STATUS ERROR MAP is not affected by the disable function.

TABLE 6. INTERRUPT ENABLE JUMPERS

JUMPER	CHANNEL	ENABLE	DISABLE
E1	1	CH	CL
E2	2	CH	CL
E3	3	CH	CL
E4	4	CH	CL
E5	5	CH	CL
E6	6	CH	CL

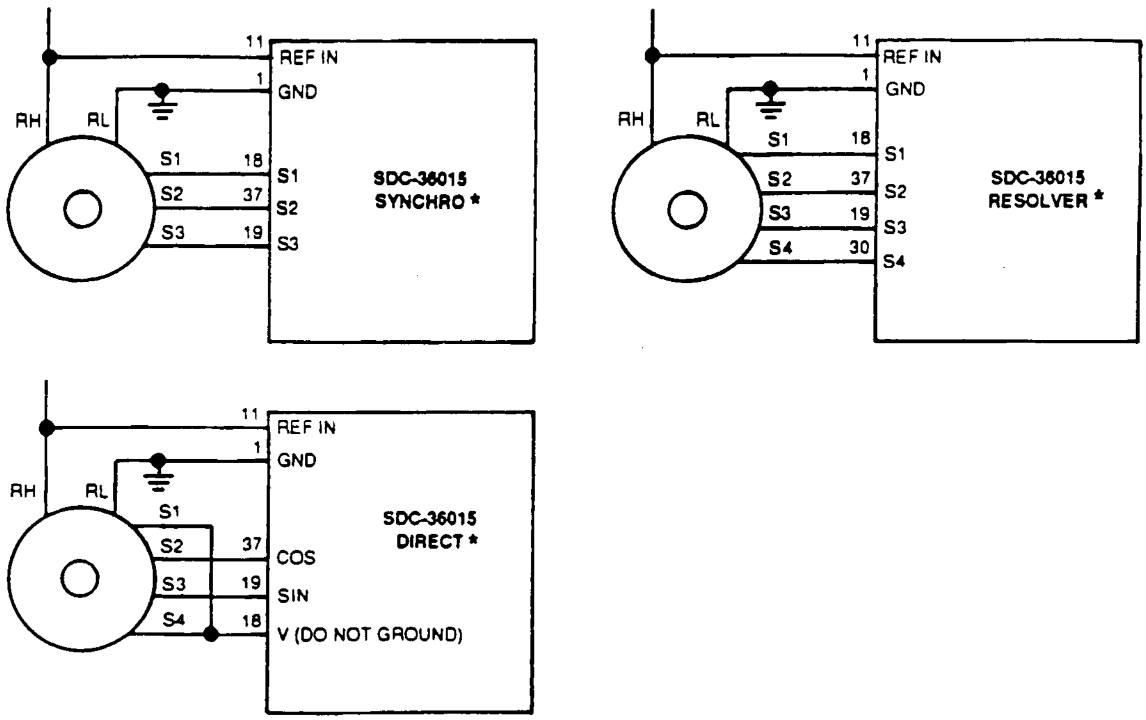
The example below sets the SDC-36015 PC card to enable interrupt for channels one through three. Please note the E1-E6 jumper positions.



INTERFACING TO THE SDC-36015

External Connections to the SDC-36015

FIGURE 3 shows the correct wiring configurations for the three SDC-36015 converter card formats.



*Transformer isolation is recommended.

FIGURE 3. SDC-36015 WIRING CONFIGURATIONS

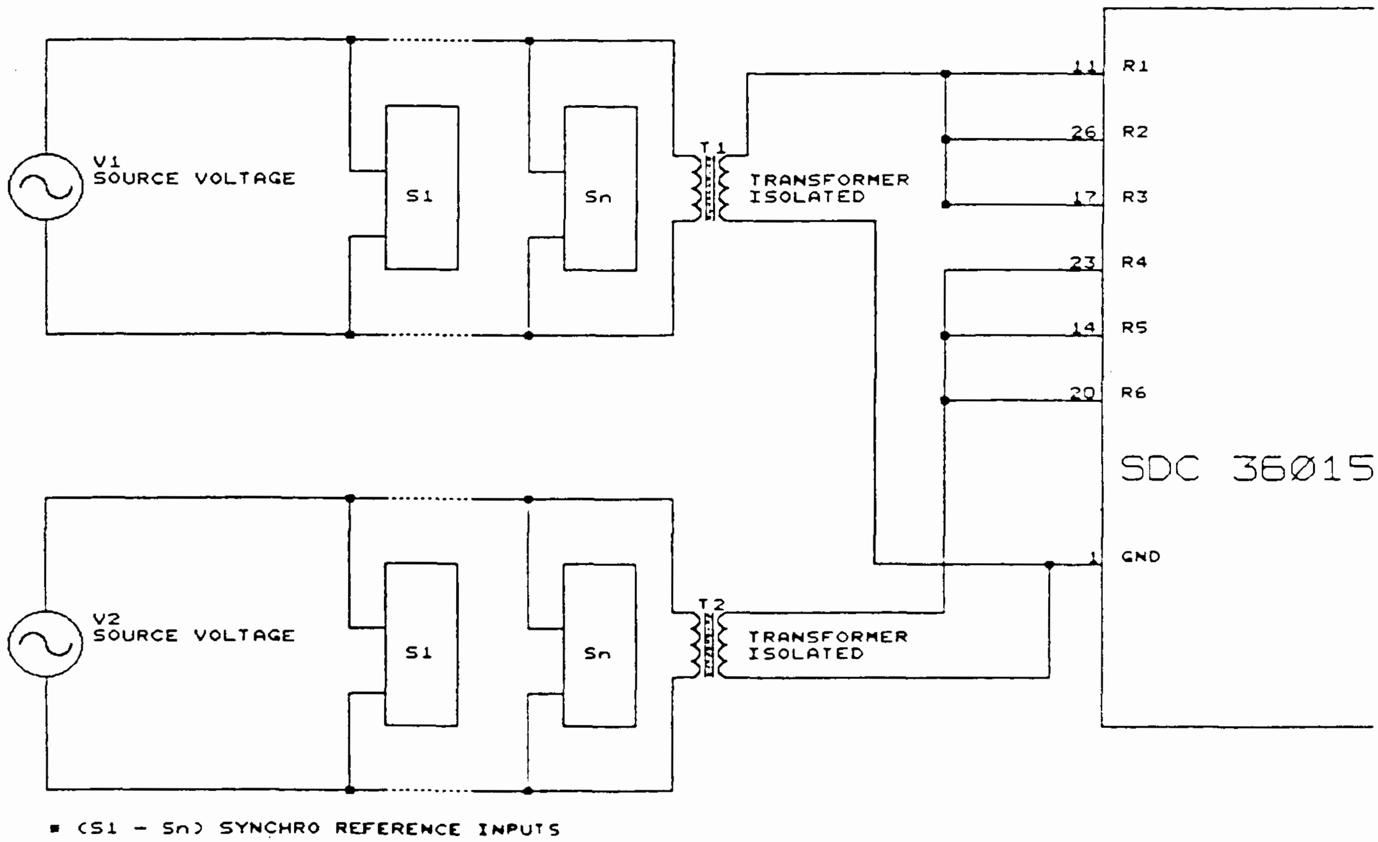


FIGURE 4. SDC-36015 ISOLATED REFERENCE SCHEMATIC

Velocity Scaling

The analog voltage generated by the converters on the SDC-36015 is proportional to the rate of change of the shaft angle. This voltage is available at the edge connector of the SDC-36015 as the Velocity (VEL) signal. This feature is possible because the synchro/resolver converters used on the SDC-36015 are "Type II" servo loop converters.

The velocity output scale factor can be increased by connecting external resistors R1, R8, R11, R18, R19 and R26 for channels 1 to 6 respectively as long as the RDC- or SDC-1920X series of converters are being used. This resistance value can be determined by the following formula:

$$R = \frac{10 \times B/A}{1 - B/A}$$

where:

R = external resistor in K Ohms
A = specified voltage scaling (RPS/VOLT)
B = desired voltage scaling (RPS/VOLT)

To determine "A" refer to TABLE 7.

TABLE 7. VELOCITY OUTPUT VOLTAGE SCALING (RPS/VOLT)

BW	10 BIT	12 BIT	14 BIT	16 BIT
HIGH	80	20	5	1.25
LOW	20	5	1.25	0.32

Signal Input Requirements

The reference requirement for the SDC-36015 card is entirely dependent on the hybrid model installed on the card. Verify that the reference frequency and voltage level along with the input signal voltage levels are set properly before installing the SDC-36015 card in your computer. TABLE 8 illustrates the input requirements for the RDC-1920X series of resolver-to-digital converters.

TABLE 8. RDC-1920X SERIES INPUT REQUIREMENTS

MODEL NUMBER	CARRIER FREQUENCY[1]	VOLT. RANGE-REF.	INPUT FORMAT	INPUT VOLTAGE[2]
RDC-19200	360 Hz to 6 KHz	4 - 50 Vrms	Resolver	11.8 V L-L
RDC-19201	360 Hz to 6 KHz	4 - 50 Vrms	Resolver	11.8 V L-L
RDC-19202	360 Hz to 6 KHz	4 - 50 Vrms	Resolver	2 V L-L
RDC-19203	360 Hz to 6 KHz	4 - 50 Vrms	Resolver	2 V L-L

[1] ±10% harmonic distortion.

[2] ±10% signal amplitude variation.

TABLE 9 illustrates the input requirements for the SDC-1920X series of synchro-to-digital converters.

TABLE 9. SDC-1920X SERIES INPUT REQUIREMENTS				
MODEL NUMBER	**CARRIER FREQUENCY[3]**	**VOLT. RANGE-REF.**	**INPUT FORMAT**	**INPUT VOLTAGE[4]**
SDC-19204	47 Hz to 1 KHz	4 - 130 Vrms	Synchro	11.8 V L-L
SDC-19206	47 Hz to 1 KHz	4 - 130 Vrms	Synchro	90 V L-L

[3] ±10% harmonic distortion.

[4] ±10% signal amplitude variation.

TABLE 10 illustrates the input requirements for the SDC-1456X series of converters.

TABLE 10. SDC-1456X SERIES INPUT REQUIREMENTS

MODEL NUMBER	CARRIER FREQUENCY[5]	VOLT. RANGE-REF.	INPUT FORMAT	INPUT VOLTAGE[6]
SDC-14560	360 Hz to 1 KHz	4 - 130 Vrms	Synchro	11.8 V L-L
SDC-14561	360 Hz to 1 KHz	4 - 130 Vrms	Synchro	90 V L-L
SDC-14562	47 Hz to 1 KHz	4 - 130 Vrms	Synchro	90 V L-L
SDC-14563	360 Hz to 1 KHz	4 - 130 Vrms	Synchro	11.8 V L-L
SDC-14564	360 Hz to 1 KHz	4 - 130 Vrms	Resolver	26 V L-L
SDC-14565	360 Hz to 1 KHz	4 - 130 Vrms	Resolver	11.8 V L-L
SDC-14566	360 Hz to 1 KHz	4 - 130 Vrms	Resolver	11.8 V L-L
SDC-14567	360 Hz to 1 KHz	4 - 130 Vrms	Resolver	1 V Direct
SDC-14568	47 Hz to 1 KHz	4 - 130 Vrms	Resolver	1 V Direct
SDC-14569	360 Hz to 1 KHz	4 - 130 Vrms	Resolver	1 V Direct

[5] ±10% harmonic distortion.

[6] ±10% signal amplitude variation.

Computer Timing Information

Data to and from the SDC-36015 card is controlled by the computer using standard I/O Read/Write cycles. All of the interface logic uses high-speed components allowing the card to be addressed in zero wait-state time cycles, however, the following precautions must be taken:

1. If a step input is given to a converter allow at least 50 ms for the output to stabilize.

2. The converter should be in an inhibit state (set with the control register) before the LSB and MSB data read. This ensures a consistant angle during the two read cycles. The converter must then be set to continue tracking by disabling the inhibit bit in the control register.

The J1 Signal I/O Connector

The SDC-36015's I/O signals are brought off the circuit card through a 37 pin male "D" type connector mounted on the rear of the circuit card. The signal connections are listed in TABLE 11.

TABLE 11. J1 SIGNAL I/O CONNECTOR

Name	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6
S1	18	9	15	24	12	21
S2	37	27	34	6	31	3
S3	19	28	16	25	13	22
S4	30	10	35	7	32	4
VEL	29	8	36	5	33	2
REF IN	11	26	17	23	14	20
GND	1	1	1	1	1	1

INSTALLING THE SDC-36015 CARD

Hardware Installation

After the card is configured to the necessary application, the board can be placed in any 8 bit slot of an IBM PC/XT/AT/286/386/486 or compatible. The following steps should be followed.

1. Turn off the power to the PC.

2. Wear an anti-static device during handling of the card.

3. Remove the cover to the PC and find an empty 8/16-bit EISA/ISA slot.

4. Remove the back slot cover from the PC.

5. Install the SDC-36015 firmly in the expansion slot.

6. Secure the SDC-36015 in place.

7. Return the cover to the PC.

8. Power up the PC.

Software Installation

The SDC-36015 comes with a software support disk. This disk contains "C" , Pascal. and LabWindows software modules, demonstration software, and supporting files. This information on this disk can be installed to a hard drive or network drive using the "INSTALL.EXE" program located in the root directory of the supplied diskette. After installing the following directory structure will be created.

```
\SDC36015
    |
    |------C
    |------PASCAL
    |------EXAMPLE
    |------DEMO
    |------MENU
    |------LW
```

SOFTWARE CONTROL OF THE SDC-36015 PC Card

Register Addressing

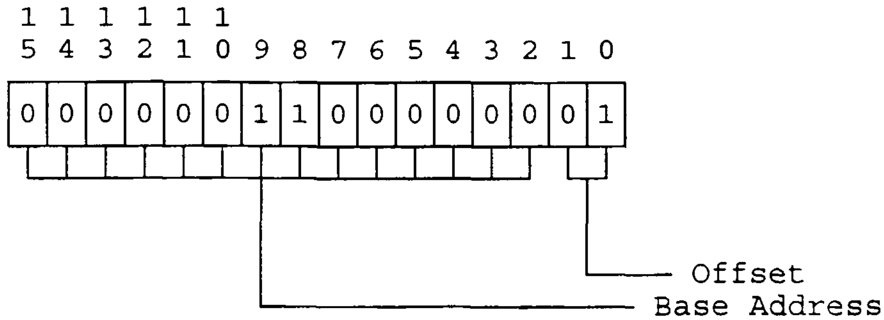
The SDC-36015 is I/O addressed in typical Base Address and Offset fashion. There are four addressable registers on the card which are decoded using the two LSBs of the I/O address. The Base Address of the card utilizes the remaining eight I/O address lines and the base address is set via the Address Jumpers W2 thru W9 located on the circuit card assembly in Jumper Block 8 as shown in the SDC-36015's Mechanical Outline. The SDC-36015 Base Address is preset at the factory to 300h (300 HEX). In this configuration the card will occupy the I/O address space 300h thru 303h. TABLE 5 shows the Address jumper setting as the card leaves the factory. The Base Address can be set to any address from 000h to 3FCh in increments of 4 using TABLE 5 as a reference. TABLE 12 contains an I/O Memory Map of the SDC-36015. Note that the Control Word is not fully address decoded and is redundant at all four I/O Write locations.

TABLE 12. I/O MEMORY MAP

A1	A2	I/O Write	I/O Read
0	0	Control Word	8 LSBs of Binary Angle
0	1	Control Word	8 MSBs of Binary Angle
1	0	Control Word	Status/Count Information
1	1	Control Word	Status/Error Map

All information generated by the SDC-36015 is memory mapped as four I/O memory locations and is available to the computer via standard I/O read cycles. This data consists of the selected channel output angle data in two bytes, and Built-In-Test information for each channel. In addition, if the RDC/SDC-1920X series of converters are utilized, 4-bits of turn count information and a Loss-Of-Signal (LOS) indicator are available for the selected channel.

An example of the creation of a Port address is explained below. This address would be 301h (Angle MSB).



Register Mapping

<u>Control Register</u>

The Command Word is in the following format:

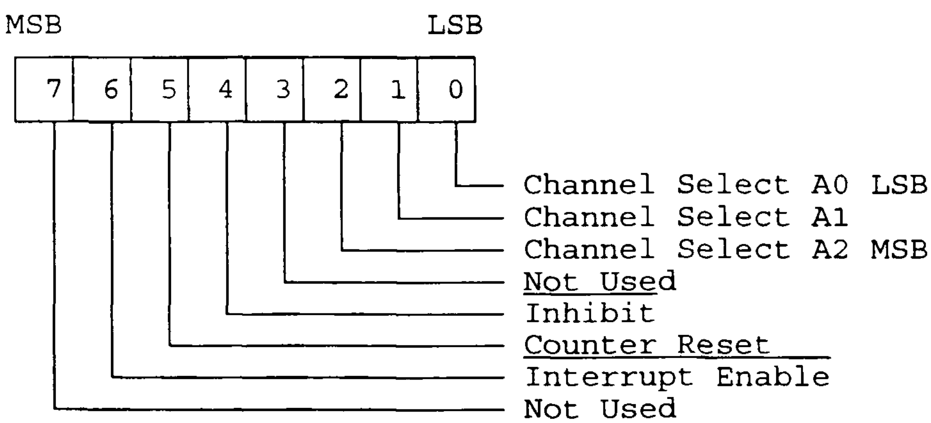


FIGURE 5. CONTROL REGISTER BITMAP

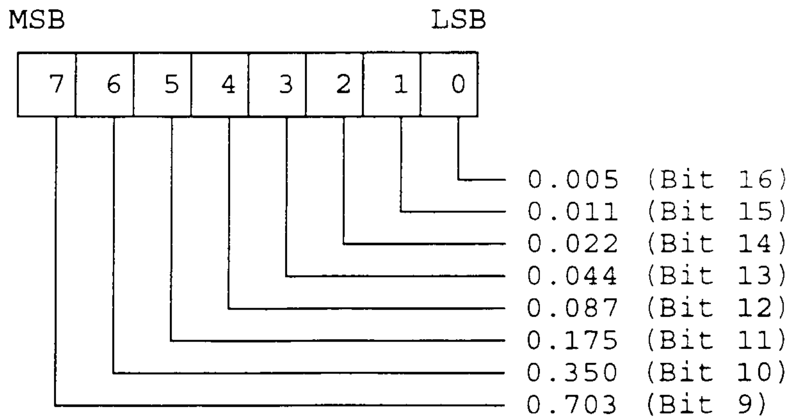
The bitmap for the Control Word is shown in TABLE 13.

TABLE 13. COMMAND WORD BITMAP

FUNCTION	D7	D6	D5	D4	D3	D2	D1	D0
CHANNEL 1 ENABLED	X	X	X	X	X	0	0	0
CHANNEL 2 ENABLED	X	X	X	X	X	0	0	1
CHANNEL 3 ENABLED	X	X	X	X	X	0	1	0
CHANNEL 4 ENABLED	X	X	X	X	X	0	1	1
CHANNEL 5 ENABLED	X	X	X	X	X	1	0	0
CHANNEL 6 ENABLED	X	X	X	X	X	1	0	1
INHIBIT ON	X	X	X	0	X	X	X	X
COUNTER RESET (TURNS)	X	X	1	X	X	X	X	X
INTERRUPT ENABLE	X	0	X	X	X	X	X	X
X = don't care								

LSB and MSB Registers

LSB Register



MSB Register

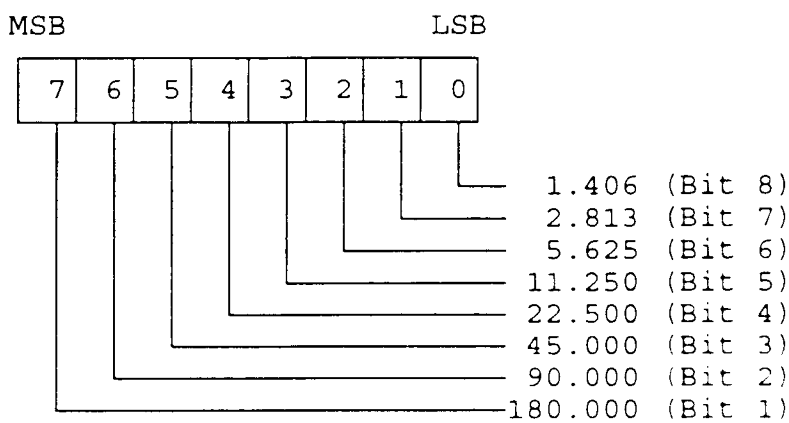


FIGURE 6. LSB AND MSB REGISTER BITMAP

Decoding the Output Angle

The Binary Angular data for the selected channel is available in two contiguous bytes of data as shown in TABLE 12. One byte contains the 8 MSBs of the angle and the other byte contains the remaining 8 LSBs. The data in both bytes is left justified. The composite angular position must be computed from the two bytes of data using the following formula:

Shaft Angle = 360 * (MSB * 256 + LSB)/65536

where:

Shaft Angle is in decimal degrees. MSB and LSB are the decimal equivalent of the two eight bit bytes representing the angle.

For example:

If the following digital angle data was output:

MSB = 01011110
LSB = 00100110

The decimal equivalent of the MSB is 94.
The decimal equivalent ot the LSB is 38.

Inserted into the equation above equals:

Shaft angle = 360 * (94 * 256 + 38)/65536

= 132.396

Status/Count Register

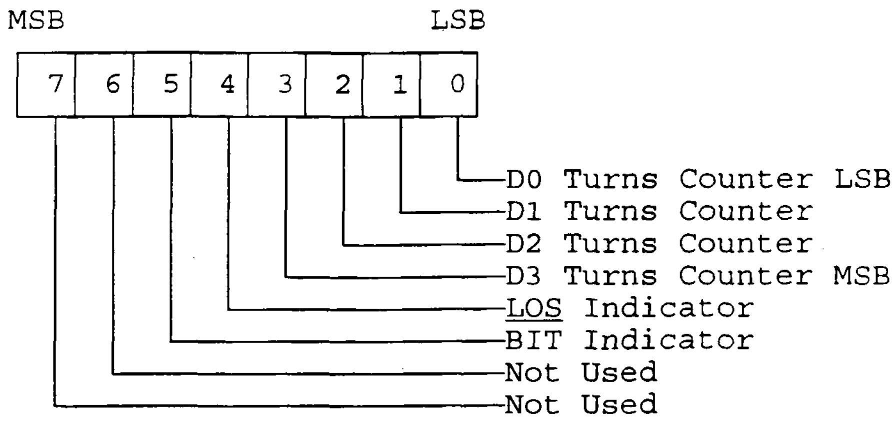


FIGURE 7. STATUS/COUNT REGISTER BITMAP

When using any of the converters available for the SDC-36015, BIT (Built-In-Test) information is available as an output from the SDC-36015 as part of the 8-bit Status/Count register.If the RDC/SDC-1920X series of converters are used, LOS (Loss-of-Signal) is available as bit 4 of the same Status/Count byte and turns count information is available as the 4 LSBs of this same byte.

LOS : During normal operation this bit will be in the logic 0 state. A logic 1 condition for this bit indicates that the Signal input to the converter is missing (Loss-of-Signal) or too low.

BIT : During normal operation this bit will be in the logic 1 state. A logic 0 condition for this bit indicates that the converter is not tracking the input synchro/resolver signal. This bit will momentarily go to a logic 0 during a large angle step.

D0-D3 Turns Counter : This value counts major carry transistions of the digital angle.

Status/Error Information

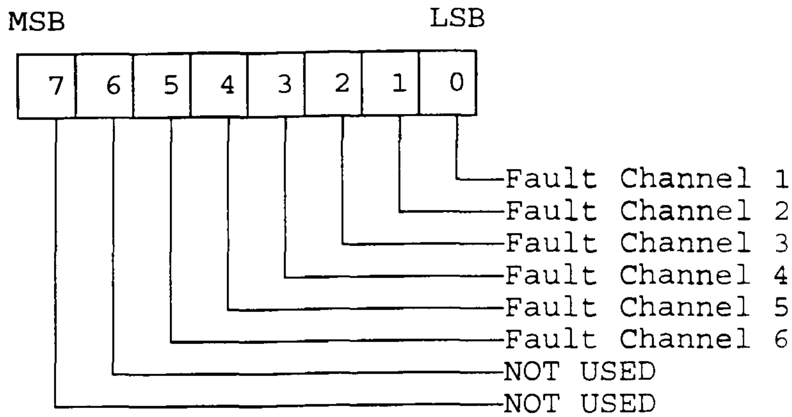


FIGURE 8. STATUS/ERROR INFORMATION BITMAP

If a BIT and/or LOS error is indicated by any of the channels of the SDC-36015, the error is shown as a logic 1 in the associated bit of this register. If the Irq is enabled, this data is latched at the same time the Irq3 line goes high indicating a Irq condition to the computer. Note that the interrupt handling routine should reset the Irq3 line and the data latched in this register by disabling and then enabling the Irq (see control register).

USING THE "C" LIBRARY

The subdirectory named C contains "SDC36015.C" which is the source code for the library. The accompanying include file "SDC36015.H" is also in this subdirectory. The compiler must be informed that \SDC36015\C contains this additional library and include file.

The top of your source code should contain the following statement:

#include "\SDC36015\C\SDC36015.H"

After the above line, the program will be able to make calls to the library routines and reference any predefined constants used with the library.

Compiling under Borland "C"

In the **\SDC36015\EXAMPLE** subdirectory, the necessary batch files exist for compiling and linking the "C" example programs. These batch files should be edited to reflect the directory setup of the computer in use. The Borland "C," BCC compiler is used with the -c option. This means to compile only. The next option compiles under the large memory model. Next the location of your BORLAND "C" INCLUDE directory is added with the -I option. In this case, Borland "C" is on drive **C:** in the **\BCPP20** subdirectory. Following the compiles of **example1.c** and the library **SDC36015.C**, Turbo Link, TLINK is executed. The /L option tells the linker where the Borland "C" LIBRARY files are. Please note that the BORLAND "C" directory must be in your DOS path in order to use BCC and TLINK.

MAKE1_BC.BAT

```
@echo creating EXAMPLE1.EXE
bcc -c -ml -IC:\BCPP20\INCLUDE example1.c
bcc -c -ml -IC:\BCPP20\INCLUDE ..\C\sdc36015.c
tlink /LC:\BCPP20\LIB @make1.lnk
```

MAKE1_BC.LNK

```
c0l.obj+
example1.obj+
sdc36015.obj
example1.exe
NUL.MAP -c
mathl.lib+
cl.lib+
emu.lib
```

"C" Software Routines

Initializing the Card

The first routine that should be called is **SDC36015_Initialize.** This clears all the configuration and error checking information private to the libraries. This routine can be called multiple times and will simply 'clear out' all the configurations that were done before it was called.

The SDC-36015 library was designed to use the same routines with multiple cards in the same system. The library keeps track of each card's address so subsequent access to the card using the various routines can be done without passing in an address. The **SDC36015_Initialize** routine must be called before the card can be accessed. The following routine configures card 0 to the hexadecimal port address **0x300**. After this call, subsequent reference to card 0 will be done with port address **0x300**.

SDC36015_Initialize (0, 0x300);

The power turn-on PC RESET signal generated by the PC resets the IRQ line 3 to a logic 0. The command register is programmed to the following settings after the Initialize routine is called...

- Disable IRQ
- Track
- Channel 1 Selected

Programming the SDC-36015

The PC interface to the SDC-36015 controls the digital lines configuring the card for different operations. The control register can be accessed by using the **SDC36015_Program** routine. When 'SDC36015.H" is included in the application source code file, the available control options are defined as constants. This simplifies the controlling of the card by allowing direct commands to change the control words. The necessary bit manipulation is done. Refer to the following list of constants.

Command	Description
SDC36015_IRQENA	Enables the Irq 3 output capability
SDC36015_IRQDIS	Disables the Irq 3 output capability
SDC36015_INHIBIT	Inhibits all Converters
SDC36015_TRACK	Tracks on all Converters
SDC36015_CRESET	Resets all Turns Counters
SDC36015_C1	Selects Channel 1 as the Active Channel
SDC36015_C2	Selects Channel 2 as the Active Channel
SDC36015_C3	Selects Channel 3 as the Active Channel
SDC36015_C4	Selects Channel 4 as the Active Channel
SDC36015_C5	Selects Channel 5 as the Active Channel
SDC36015_C6	Selects Channel 6 as the Active Channel

The constants are passed into the **SDC36015_Program** routine and the desired bits are toggled in the Control Word.

SDC36015_Program (0 , SDC36015_CRESET); /* Card 0 */

This routine is used for selecting the active channel. The active channel selects the converter to read LOS, BIT, Turns Count, and Angle Data.

Reading the BIT Flag

Both the RDC/SDC-1920X and SDC-1456X support a BIT Flag for error detection. If a fault condition exists in the active converter the flag is active. This fault may show momentarily if a large angular change is made.

The **SDC36015_Read_Bite** routine is used to read the error condition of the card. The bit is normally in a logic 1 state. The bit goes to a logic 0 if the error condition is present.

int BiteState;

BiteState = SDC36015_Read_Bite (0);

Reading the LOS Flag

The RDC/SDC-1920X converters support a LOS (Loss-of-Signal) Flag for error detection. The **SDC36015_Read_Los** routine returns the status of the Los flag. This bit is normally in a logic 0 state. The bit goes to a logic 1 if the synchro/resolver input signal is either too low or missing.

int LosState;

LosState = SDC36015_Read_Los (0);

Reading the Angle

The routine **SDC36015_Read_Angle** allows the reading of angle information in 16 bit binary form. The following call would read card number **1**, channel **0** into the variable called **Angle**. This variable is of a defined type WORD. This is defined in SDC36016.H as a 16 bit unsigned integer. Note that the converter should be in an inhibit state during the read to ensure the consistency of the LSB and MSB during the PC read cycles. This is controlled by the third parameter. If this parameter is passed the constant MODE_CONTINOUS the angle is not latched with the **Read_Angle** routine. If this parameter is passed the constant MODE_LATCH, the converter is inhibited, the data is read, and the converter is set to continue tracking the input angle.

WORD Angle;

Angle = SDC36015_Read_Angle (1, MODE_LATCH);

Reading the Turns Counter

The routine **SDC36015_Read_Count** allows the reading of the 4-bit turns counter associated with the active channel. This passed a card number and returns and integer. Note that only the last 4 bits are used reflecting the 4-bit turns counter.

int Count;

Count = SDC36015_Read_Count (0); /* Card 0 */

Reading the Status

If a fault condition BIT or LOS occurs on a channel, the associated status bit in the status register is set active. The value of this register changes constantly with the error conditions of the converters. If an interrupt condition occurs, however, the data is latched until the irq is cleared using the command register. Note the six channels are represented with channel 1 representing the LSB and channel 6 representing bit 6 of the returned value.

int Status;

Status = SDC36015_Read_Status (0);

Closing the SDC-36015 Card

During an application it is good programming practice to INITIALIZE the SDC-36015 card at the beginning of the access to the card, and to CLOSE the card when access to the card is finished. This returns the card to the following command state.

- Disable IRQ
- Inhibit
- Channel 1 Selected

```
SDC36015_Close( 0 ); /* Close Card 0 */
```

"C" Programming Algorithms

Converting to Decimal

The SDC36015_Read_Angle routine returns a 16-bit binary coded angle. This can be converted to a standard floating point double precision value with the following "C" routine.

```
WORD   AngleBinary;
double AngleDecimal;

AngleBinary = SDC36015_Read_Angle ( 0 , MODE_LATCH );
AngleDecimal = ( AngleBinary * 360.0 ) / 65536.0 );
```

Reading Multiple Channels

Since only one channel can be active on the SDC-36015 card, the channel must be selected before reading data.

```
WORD angle[6];
int  bite[6];
int  X;

for (X=0;X<6;X++)
     {
      SDC36015_Program(0,(0x1000*X)+SDC36015_C1);
      angle[X]=SDC36015_Read_Angle(0,MODE_LATCH);
      bite[X]=SDC36015_Read_Bite(0);
     }
```

USING THE PASCAL UNIT

The subdirectory named PASCAL contains 'SDC36015.PAS' which is the source code for the Pascal Unit. The also in this directory is the compiled Turbo Pascal unit file 'SDC36015.TPU'. This was compiled with Turbo Pascal version 7.0. The compiler must be informed that \SDC36015\PASCAL contains additional unit files.

The top of your source code should contain the following statement:

Uses SDC36015;

After the above line, the program will be able to make calls to the software routines and reference any predefined constants used with the module.

Pascal Software Routines

Initializing the card

The first routine that should be called is **SDC36015_Initialize.** This clears all the configuration and error checking information private to the libraries. This routine can be called multiple times and will simply "clear out" all the configurations that were done before it was called.

The SDC-36015 unit was designed to use the same routines with multiple cards in the same system. The unit keeps track of each card's address so subsequent access to the card using the various routines can be done without passing in an address. The **SDC36015_Initialize** routine must be called before the card can be accessed. The following routine configures card 0 to the hexadecimal port address **$300**. After this call, subsequent reference to card 0 will be done with port address **$300**.

SDC36015_Initialize (0, $300);

The power turn-on PC RESET signal generated by the PC resets the IRQ line 3 to a logic 0. The command register is programmed to the following settings after the Initialize routine is called...

- Disable IRQ
- Track
- Channel 1 Selected

Programming the SDC-36015

The PC interface to the SDC-36015 controls the digital lines configuring the card for different operations. The control register can be accessed by using the **SDC36015_Program** routine. When the "Uses SDC36015;" command is included in the application source code file, the available control options are defined as constants. This simplifies the controlling of the card by allowing direct commands to change the control words. The necessary bit manipulation is done. Refer to the following list of constants.

Command	Description
SDC36015_IRQENA	Enables the Irq 3 output capability
SDC36015_IRQDIS	Disables the Irq 3 output capability
SDC36015_INHIBIT	Inhibits all Converters
SDC36015_TRACK	Tracks on all Converters
SDC36015_CRESET	Resets all Turns Counters
SDC36015_C1	Selects Channel 1 as the Active Channel
SDC36015_C2	Selects Channel 2 as the Active Channel
SDC36015_C3	Selects Channel 3 as the Active Channel
SDC36015_C4	Selects Channel 4 as the Active Channel
SDC36015_C5	Selects Channel 5 as the Active Channel
SDC36015_C6	Selects Channel 6 as the Active Channel

The constants are passed into the **SDC36015_Program** routine and the desired bits are toggled in the Control Word.

SDC36015_Program (0 , SDC36015_CRESET); { Card 0 }

This routine is used for selecting the active channel. The active channel selects the converter to read LOS, BIT, Turns Count, and Angle Data.

Reading the BIT Flag

Both the RDC/SDC-1920X and SDC-1456X support a BIT Flag for error detection. If a fault condition exists in the active converter the flag is active. This fault may show momentarily if a large angular change is made.

The **SDC36015_Read_Bite** routine is used to read the error condition of the card. The bit is normally in a logic 1 state. The bit goes to a logic 0 if the error condition is present.

Var BiteState : Integer;

BiteState := SDC36015_Read_Bite (0);

Reading the LOS Flag

The RDC/SDC-1920X converters support a LOS (Loss-of-Signal) Flag for error detection. The **SDC36015_Read_Los** routine returns the status of the Los flag. This bit is normally in a logic 0 state. The bit goes to a logic 1 if the input synchro/resolver is either too low or missing.

Var LosState : Integer;

LosState := SDC36015_Read_Los (0);

Reading the Angle

The routine **SDC36015_Read_Angle** allows the reading of angle information in 16-bit binary form. The following call would read card number **1**, channel **0** into the variable called **Angle**. This variable is of a defined type Word (16-bit unsigned integer). Note that the converter should be in an inhibit state during the read to ensure the consistency of the LSB and MSB during the PC read cycles. This is controlled by the third parameter. If this parameter is passed the constant MODE_CONTINOUS the angle is not latched with the **Read_Angle** routine. If this parameter is passed the constant MODE_LATCH, the converter is inhibited, the data is read, and the converter is set to continue tracking the input angle.

Var Angle : Word;

Angle := SDC36015_Read_Angle (1, MODE_LATCH);

Reading the Turns Counter

The routine **SDC36015_Read_Count** allows the reading of the 4-bit turns counter associated with the active channel. This passed a card number and returns and integer. Note that only the last 4 bits are used reflecting the 4-bit turns counter.

Var Count : Integer;

Count := SDC36015_Read_Count (0); { Card 0 }

Reading the Status

If a fault condition BIT or LOS occurs on a channel, the associated status bit in the status register is set active. The value of this register changes constantly with the error conditions of the converters. If an interrupt condition occurs, however, the data is latched until the irq is cleared using the command register. Note the six channels are represented with channel 1 representing the LSB and channel 6 representing bit 6 of the returned value.

Var Status : Integer;

Status := SDC36015_Read_Status (0);

Closing the SDC-36015 Card

During an application it is good programming practice to INITIALIZE the SDC-36015 card at the beginning of the access to the card, and to CLOSE the card when access to the card is finished. This returns the card to the following command state.

- Disable IRQ
- Inhibit
- Channel 1 Selected

SDC36015_Close(0); { Close Card 0 }

Pascal Programming Algorithms

Converting to Decimal

The SDC36015_Read_Angle routine returns a 16-bit binary coded angle. This can be converted to a standard floating point real value with the following pascal code.

```
Var AngleBinary : Word;
   AngleDecimal: Real;

AngleBinary := SDC36015_Read_Angle ( 0 , MODE_LATCH );
AngleDecimal := ( AngleBinary * 360.0 ) / 65536.0 );
```

Reading Multiple Channels

Since only one channel can be active on the SDC-36015 card, the channel must be selected before reading data.

```
Var angle : Array [0..6] of Word;
Var bite : Array [0..6] of Bite;
int  X;

For X := 0 to 5 Do
      Begin
         SDC36015_Program(0,($1000*X)+SDC36015_C1);
         angle[X]:=SDC36015_Read_Angle(0,MODE_LATCH);
         bite[X]:=SDC36015_Read_Bite(0);
      End;
```

USING WITH LABWINDOWS

LabWindows Instrument Module

The SDC-36015 "C" Module has been compiled into a *LabWindows Instrument Module*. This can be very easily incorporated into your *LabWindows* applications. The necessary files associated with the *Instrument Module* are contained in the \SDC36015\LW subdirectory. Their description is as follows:

SDC36015.C "C" Source code to be compiled under *LabWindows*.

SDC36015.H "C" Include file to be compiled under *LabWindows*.

SDC36015.LBW *LabWindows* compiled source code for the module.

SDC36015.LWI *LabWindows* compiled include file for the module.

SDC36015.FPThis is the instrument's function panel file. This describes the function tree, function panels, and help files for working in the *LabWindows* environment.

Loading the Instrument

Since the instrument module has already been compiled, there are only three steps to using the SDC-36015 Instrument Module in your application.

1. Load up the *LabWindows* environment by typing LW at the DOS prompt.

2. Pull down the **INSTRUMENT** menu and select **LOAD**. Go to the **\SDC36015\LW** subdirectory and load 'SDC36015.FP'.

3. You can now fully utilize the functions described in Appendix A. It is recommended to begin by looking at example6.c which is contained in the **\SDC36015\EXAMPLE** subdirectory.

The LabWindows Application

Included on this disk is a user friendly GUI (Graphic User Interface) for use with the SDC-36015 Synchro/Resolver-to-Digital converter card. Located in the **\SDC36015\DEMO** subdirectory are all the files necessary to create an executable version of this interface. Please note the following file list:

15DEMO.CSDC36015 Demonstration Source Code
15DEMO.HSDC36015 Demonstration Include File
15DEMO.UIRSDC36015 *LABWINDOWS* User Interface
SRDDC.PCXILC DATA DEVICE CORPORATION LOGO
SETUP.DATSaved Default Configuration File

Combining with other Software

The demonstration software can easily be incorporated into existing software. Below is a listing of the MAIN() function in the "C" demonstration program. After **demo_initialize** is called, the **demo_loop** function can be called at any time to bring up the main display screen.

```
void main ()
{/*BEGIN MAIN*/
   demo_initialize ();
   demo_loop ();
   demo_close ();
}/*END MAIN*/
```

The Display Panel

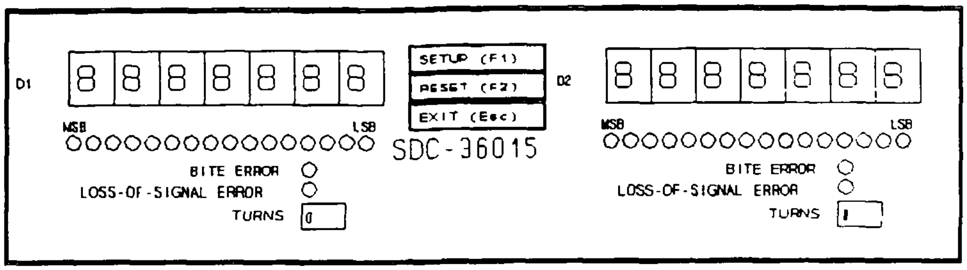


FIGURE 9. "LABWINDOWS" DISPLAY SCREEN

The display has a user interface similar to many bench-top instruments. The display includes two, seven character output displays and two 16 bit binary output displays. This enables the data from two active channels to be displayed simultanously. Also included for the display channels is a BIT, LOS, and Turns counter output display. The (SETUP-F1) button brings up the setup screen (see Setup Screen). The (RESET-F2) button resets the turns counter. When the demo loads, it begins reading angular/status information. The (EXIT-ESC) button is pressed to exit the program.

The Setup Panel

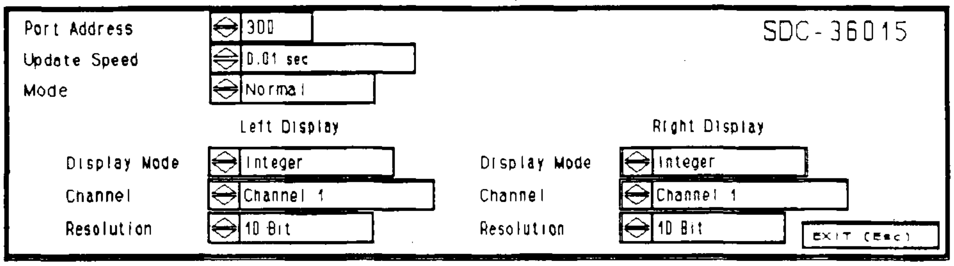


FIGURE 10. "LABWINDOWS" CONFIGURATION SCREEN

This screen allows the user to configure all the programmable options of the demo with simple selectors and buttons. Also this configuration is automatically saved so that the user can have the software customized to their setup.

The most important setting of the card is the PORT ADDRESS. This must be configured before the card is read. The value of this setting can be between 0x300 and 0x390 in increments of 16 (ex. 0x340,0x350,0x360...). The delay between reads can be set to 0.01, 0.1, or 1.0 seconds. The MODE selection either takes data from real ports on the PC (NORMAL), or simulates counting data (DEMO).

The left and right displays are independant. They can be set to read data from CHANNELS 1 through 6, RESOLUTION 10 through 16, and DISPLAY MODE to hex (hex display of 16-bit data 0x0000-0xFFFE), integer (integer display of 16-bit data 0-65535), or float (decimal converted to floating point angular 0.0-359.995).

NOTE: The setup is written to the card when the setup panel is is closed.

THE MENU SOFTWARE

Running from Dos

The menu software was designed to work on the IBM PC in 80 Column text mode. The software can be run by issuing the following DOS commands:

```
CD \SDC36015\MENU
        DDC36015
```

Menu

1. Select Active Card
2. Initialize - Set Base Address
3. Configuration
    1. Enable Irq
    2. Display Channels
    3. Select Two-Speed Channels
    4. Set Inhibit State
    5. Reset Turns Counter
    6. Reset Irq
    7. Return to Main Menu
4. Measure Angle
5. Quit

Using the Software

The active card is first selected. This enables the use of multiple cards with the same software. This can be set to 1 or 2. The Port Address of the card is then set using the Initialize selection.

The card can then be configured. The configuration menu allows control over the IRQ ENABLE control bit, display of selected channels (1-6), selection of up to three two-speed channels with speed ratio's from 2 through 128, control of the INHIBIT control bit, reset of the turns counter and Irq.

The Measure Angle menu command allows the display up to six channels including angular data in degrees, BIT and LOS flags, turns count, and two-speed combined angles.

APPENDIX A: FUNCTIONS

SDC36015_Initialize

"Initialize the SDC-36015 Card"

FORMAT
void SDC36015_Initialize (int Card, int Address);

Procedure SDC36015_Initialize (Card, Address : Integer);

PURPOSE

This routine must be called before the SDC-36015 can be accessed. The SDC-36015 software module was designed to work with multiple cards in the same system. The card number is passed in through the **Card** parameter. The port address of the card is passed in through **Address**. All following function calls to that card will use this address for communication. The address can be changed by calling this routine again. SDC36015_Initialize sets the card to the following configuration...

Irq Disabled
Track
    Channel 1 Selected

PARAMETERS

Card
Allows selection of a specific card to be used in reference to the function call. The valid range of this parameter is from 0 to 7. (the maximum number of cards is controlled with the MAXCARDS constant).

Address
This value is the port address of the card (the address must also be set in hardware with the dip switches located on the card assembly). The valid range of this address is from 000h to 3FCh in increments of 4. (ex. 300h,304h,308h)

EXAMPLE

```
SDC36015_Initialize ( 0, 0x300 ); /* Card 0 Port=0x300 */

SDC36015_Initialize ( 0, $300 ); { Card 0 Port=$300 }
```

RETURN VALUE

None.

SDC36015_Program

"Command Control the SDC-36015"

FORMAT

void SDC36015_Program (int Card, int Command);

Procedure SDC36015_Program (Card, Command : Integer);

PURPOSE

All control of the SDC-36015 card is done using this routine. The requested command is passed to the routine through the **Command** parameter.

PARAMETERS

Card
Allows selection of a specific card to be used in reference to the function call. The valid range of this parameter is from 0 to 7. (the maximum number of cards is controlled with the MAXCARDS constant).

Command
When the SDC-36015 module is included in the compilation of the application source code, the constants used for the various commands are defined. The following is a list of all valid constants that can be passed through the **Command** parameter.

Command	Description
SDC36015_IRQENA	Enables the Irq 3 output capibility
SDC36015_IRQDIS	Disables the Irq 3 output capibility
SDC36015_INHIBIT	Inhibits All Converters
SDC36015_TRACK	Tracks with All Converters
SDC36015_CRESET	Resets all Turns Counters
SDC36015_C1	Selects Channel 1 as the Active Channel
SDC36015_C2	Selects Channel 2 as the Active Channel
SDC36015_C3	Selects Channel 3 as the Active Channel
SDC36015_C4	Selects Channel 4 as the Active Channel
SDC36015_C5	Selects Channel 5 as the Active Channel
SDC36015_C6	Selects Channel 6 as the Active Channel

EXAMPLE

SDC36015_Program (1,SDC36015_TRACK); /*Card 1 Track Input Signal*/

SDC36015_Program (1,SDC36015_TRACK); {Card 1 Track Input Signal}

RETURN VALUE

None.

SDC36015_Read_Bite

"Read the Bite Error Flag"

FORMAT

void SDC36015_Read_Bite (int Card);

Function SDC36015_Read_Bite (Card : Integer) : Integer;

PURPOSE

This function reads the current state of the BIT flag. This bit is normally in a logic 1 state. The bit goes to a logic 0 if the converter can no longer track the input synchro/resolver signal. This fault may occur for a short time duration if a large angular change is made and the converter error has not settled on the angle.

NOTE: Error data is read from the currently active channel.

PARAMETERS

Card
Allows selection of a specific card to be used in reference to the function call. The valid range of this parameter is from 0 to 7. (the maximum number of cards is controlled with the MAXCARDS constant).

EXAMPLE

```
        int BiteState;
BiteState = SDC36015_Read_Bite ( 1 ); /* Card 1 */

        Var BiteState : Integer;
BiteState := SDC36015_Read_Bite ( 1 ); { Card 1 }
```

RETURN VALUE

Normally this routine should return a logic 1. It returns a logic 0 if a fault condition exists.

SDC36015_Read_Los

"Read the Loss-of-Signal Error Flag"

FORMAT

int SDC36015_Read_Los (int Card);

Function SDC36015_Read_Los (Card : Integer) : Integer;

PURPOSE

Returns the status of the LOS (Loss of Signal) flag. This bit is normally in a logic 0 state. The bit goes to a logic 1 if the input signal to the converter is either too low or missing.

NOTE: Error data is read from the currently active channel.

PARAMETERS

Card
Allows selection of a specific card to be used in reference to the function call. The valid range of this parameter is from 0 to 7. (the maximum number of cards is controlled with the MAXCARDS constant).

EXAMPLE

```
        int LosState;
LosState = SDC36015_Read_Los ( 1 ); /* Card 1 */

        Var LosState : Integer;
LosState := SDC36015_Read_Los ( 1 ); { Card 1 }
```

RETURN VALUE

This function normally returns a logic 0. If a logic 1 is returned, the synchro/resolver signal input into the converter is missing or too low.

SDC36015_Read_Angle

"Read the Output Angle"

FORMAT

WORD SDC36015_Read_Angle (int Card , int Mode);

Function SDC36015_Read_Angle (Card : Integer; Mode : Int) : Word;

PURPOSE

This returns the output angle of the SDC-36015's currently active channel. The routine is passed a valid Card number and a read MODE. The LSB and MSB are read and returned as a combined 16 bit unsigned integer (WORD).

NOTE: Angle data is read from the currently active channel.

PARAMETERS

Card
Allows selection of a specific card to be used in reference to the function call. The valid range of this parameter is from 0 to 7. (the maximum number of cards is controlled with the MAXCARDS constant).

MODE

This parameter can be set to MODE_CONTINOUS for a direct read of the LSB and MSB or MODE_LATCH. The MODE_LATCH option should be used to ensure the converter output is latched during the read cycles of the PC. This command latches the converter output, reads the LSB and MSB data, and then returns the converter to track.

EXAMPLE

```
angle = SDC36015_Read_Angle ( 0 , MODE_LATCH );

angle := SDC36015_Read_Angle ( 0 , MODE_LATCH );
```

RETURN VALUE

Bit	Value
0	0.005
1	0.011
2	0.022
3	0.044
4	0.087
5	0.175
6	0.350
7	0.703
8	1.406
9	2.813
A	5.625
B	11.25
C	22.50
D	45.00
E	90.00
F	180.0

SDC36015_Read_Count

"Read the Turns Counter"

FORMAT

int SDC36015_Read_Count (int Card);

Function SDC36015_Read_Count (Card : Integer;) : Integer;

PURPOSE

This returns the 4 bits of the turns counter of the active channel.

NOTE: Turns counter data is read from the currently active channel.

PARAMETERS

Card
Allows selection of a specific card to be used in reference to the function call. The valid range of this parameter is from 0 to 7. (the maximum number of cards is controlled with the MAXCARDS constant).

EXAMPLE

```
turns = SDC36015_Read_Count ( 0 );

turns := SDC36015_Read_Count ( 0 );
```

RETURN VALUE

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0

Bit	Description
0	D0 Turns Counter LSB
1	D1 Turns Counter
2	D2 Turns Counter
3	D3 Turns Counter MSB
4	N/C
5	N/C
6	N/C
7	N/C
8	N/C
9	N/C
A	N/C
B	N/C
C	N/C
D	N/C
E	N/C
F	N/C

SDC36015_Read_Status

"Read the Error Status Register"

FORMAT

int SDC36015_Read_Status (int Card);

Function SDC36015_Read_Status (Card : Integer;) : Integer;

PURPOSE

This returns the 6 bits of error status. Each bit is normall a logic 0. The bit associated with a channel goes to a logic 1 state if the BIT or LOS error occur for that channel. The status data normally updates with the error status of the channels. If IRQ is enabled and an error occurs, this register is latched on the IRQ line. The register can be cleared by issuing the IRQ reset command.

PARAMETERS

Card
Allows selection of a specific card to be used in reference to the function call. The valid range of this parameter is from 0 to 7. (the maximum number of cards is controlled with the MAXCARDS constant).

EXAMPLE

```
st = SDC36015_Read_Status ( 0 );

st := SDC36015_Read_Status ( 0 );
```

RETURN VALUE

Bit	Description
0	S0 Status Channel 1
1	S1 Status Channel 2
2	S2 Status Channel 3
3	S3 Status Channel 4
4	S4 Status Channel 5
5	S5 Status Channel 6
6	N/C
7	N/C
8	N/C
9	N/C
A	N/C
B	N/C
C	N/C
D	N/C
E	N/C
F	N/C

SDC36015_Close

"Close the SDC-36015 Card"

FORMAT

void SDC36015_Close (int Card);

Procedure SDC36015_Close (int Card);

PURPOSE

This is the last routine that should be called to the SDC-36015 module. This returns the SDC-36015 card to the below states. It is good practice to call this procedure at the end of hardware use.

- Disable Irq
- Inhibit
- Channel 1 Selected

PARAMETERS

Card
Allows selection of a specific card to be used in reference to the function call. The valid range of this parameter is from 0 to 7. (the maximum number of cards is controlled with the MAXCARDS constant).

EXAMPLE

SDC36015_Close (1); /* Card 1 */

SDC36015_Close (1); { Card 1 }

RETURN VALUE

None.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B: "C" SOFTWARE

```
/*
 * SDC36015.H
 *
 * "C" SOFTWARE INTERFACE LIBRARY FOR USE WITH THE SDC-36015
 * SYNCHRO RESOLVER TO DIGITAL CONVERTER CARD.
 *
 * ILC DATA DEVICE CORPORATION
 * 105 WILBUR PLACE
 * BOHEMIA, NY 11716-2426
 * (516) 567-5600
 *
 * V 1.2 CREATED WITH BORLAND "C" 3.1 10/5/94 BOB PASCAZIO x 7386
 */

/* TYPE DEFINITIONS */
typedef unsigned int WORD;

/* CONSTANTS */
#define MAXCARD             0x8
#define MAXCHANNEL          0x6
#define CARDMASK            0x7
#define TRACK               0x1
#define INHIBIT             0x0
#define NORMAL              0x1
#define BITE                0x0
#define MODE_CONTINOUS      0x0
#define MODE_LATCH          0x1
#define ENABLE              0x1
#define DISABLE             0x0

/* REGISTERS */
#define SDC36015_REG_CMD            0x00
#define SDC36015_REG_LSB            0x00
#define SDC36015_REG_MSB            0x01
#define SDC36015_REG_ERRCNT         0x02
#define SDC36015_REG_STAT           0x03
```

```
/* SDC-36015 COMMANDS */
                                          /* New  Reg# Posit  Mask     */
#define SDC36015_IRQENA  0x06BF          /* 0000 0    110    1011 1111 */
#define SDC36015_IRQDIS  0x16BF          /* 0001 0    110    1011 1111 */
#define SDC36015_INHIBIT 0x04EF          /* 0000 0    100    1110 1111 */
#define SDC36015_TRACK   0x14EF          /* 0001 0    100    1110 1111 */
#define SDC36015_CRESET  0x15DF          /* 0000 0    101    1101 1111 */
#define SDC36015_C1      0x00F8          /* 0000 0    000    1111 1000 */
#define SDC36015_C2      0x10F8          /* 0001 0    000    1111 1000 */
#define SDC36015_C3      0x20F8          /* 0010 0    000    1111 1000 */
#define SDC36015_C4      0x30F8          /* 0011 0    000    1111 1000 */
#define SDC36015_C5      0x40F8          /* 0100 0    000    1111 1000 */
#define SDC36015_C6      0x50F8          /* 0101 0    000    1111 1000 */

/* FUNCTION PROTOTYPES */
void SDC36015_Initialize ( int, int );
void SDC36015_Program (int, WORD );
WORD SDC36015_Read_Angle ( int, int );
int  SDC36015_Read_Bite ( int );
int  SDC36015_Read_Los ( int );
int  SDC36015_Read_Count ( int );
int  SDC36015_Read_Status ( int );
void SDC36015_Close ( int );

/* END SDC36015.H */
```

```
/*
 * SDC36015.C
 *
 * "C" SOFTWARE INTERFACE LIBRARY FOR USE WITH THE SDC-36015
 * SYNCHRO RESOLVER TO DIGITAL CONVERTER CARD.
 *
 * ILC DATA DEVICE CORPORATION
 * 105 WILBUR PLACE
 * BOHEMIA, NY 11716-2426
 * (516) 567-5600
 *
 * V 1.2 CREATED WITH BORLAND "C" 3.1 10/5/94 BOB PASCAZIO x 7386
 */


#include <SDC36015.H>
#include <CONIO.H>
#include <DOS.H>


WORD    SDC36015_Addr    [MAXCARD];
WORD    SDC36015_Cmd     [MAXCARD];
int     SDC36015_Init    [MAXCARD];


void SDC36015_Initialize ( int Cd, int Addr )
{/*--------------------------------------------------------------------*/
    Cd&=CARDMASK;                     /*RANGE CHECKING*/
    SDC36015_Addr [Cd] = Addr;        /*STORE ADDRESS*/

    /*DISABLE IRQ*/
    /*TRACK*/
    /*CHANNEL 1 SELECTED*/
    SDC36015_Cmd [Cd] = 0X10;

    /*WRITE CMD REGISTER*/
    outp (SDC36015_Addr [Cd]+SDC36015_REG_CMD,SDC36015_Cmd [Cd]);

    SDC36015_Init [Cd] = 1;                  /*SET INIT FLAG*/
/*---------------------------------------------------------------------*/}
```

```
void SDC36015_Program (int Cd, WORD Cmd)
{/*------------------------------------------------------------------------*/
   if (SDC36015_Init[Cd]) /*CHECK INIT FLAG*/
        {
        /*MASK OFF BIT(S)*/
        SDC36015_Cmd[Cd]&=(Cmd&0x00FF);
        /*AND NEW BIT(S)*/
        SDC36015_Cmd[Cd]|=(((Cmd&0xF000)>>12)<<((Cmd&0x0700)>>8));
        /*WRITE CMD REGISTER*/
        outp (SDC36015_Addr [Cd]+SDC36015_REG_CMD,SDC36015_Cmd[Cd]);

        if (Cmd==SDC36015_CRESET)                /*CHECK FOR CNTR RESET*/
             {
             /*YES - RESET BIT TO 0*/
             SDC36015_Cmd[Cd]&=0xDF;
             /*WRITE CMD REGISTER*/
             outp(SDC36015_Addr[Cd]+SDC36015_REG_CMD,SDC36015_Cmd[Cd]);
             }
        }
/*------------------------------------------------------------------------*/}
```

```
WORD SDC36015_Read_Angle ( int Cd, int Mode)
{/*-----------------------------------------------------------------------------*/

   WORD AngleData;       /* TEMP STORAGE */
   AngleData=0X0000;     /* CLEAR STORAGE */
   Cd&=CARDMASK;            /* RANGE CHECKING */

   /* CHECK INIT FLAG AND CHANNEL */
   if (SDC36015_Init[Cd])
       {
       /* CONTINOUS MODE */
       if (Mode==MODE_CONTINOUS)
          {
           /* YES - READ ANGLE */
           AngleData = (inp(SDC36015_Addr[Cd]+SDC36015_REG_MSB)<<8)|
                   (inp(SDC36015_Addr[Cd]+SDC36015_REG_LSB));
          }
       else
          {
           /* NO - LATCH MODE, SO INHIBIT */
           outp (
              SDC36015_Addr [Cd]+SDC36015_REG_CMD,
              (SDC36015_Cmd [Cd]&0xEF));
           /* READ ANGLE */
           AngleData = (inp(SDC36015_Addr[Cd]+SDC36015_REG_MSB)<<8)|
                   (inp(SDC36015_Addr[Cd]+SDC36015_REG_LSB));
           /* TRACK */
           outp (
              SDC36015_Addr [Cd]+SDC36015_REG_CMD,
              (SDC36015_Cmd [Cd]|0x10));
          }
       }
   /* RETURN DATA */
   return (AngleData);
/*-----------------------------------------------------------------------------*/}
```

```
int SDC36015_Read_Bite ( int Cd )
{/*--------------------------------------------------------------------------*/

    int BiteData;    /* TEMP STORAGE */
    BiteData=0;      /* CLEAR STORAGE */
    Cd&=CARDMASK;  /* RANGE CHECKING */

    /* CHECK INIT FLAG */
    if (SDC36015_Init[Cd])
        {
         /* LOAD ERROR AND COUNT REGISTER */
         BiteData = inp(SDC36015_Addr[Cd]+SDC36015_REG_ERRCNT);
         /* CHECK BITE */
         if (BiteData&0x20) BiteData=1; else BiteData=0;
        }
    /* RETURN DATA */
    return (BiteData);
/*--------------------------------------------------------------------------*/}

int SDC36015_Read_Los ( int Cd )
{/*--------------------------------------------------------------------------*/

    int LosData;     /* TEMP STORAGE */
    LosData=0;       /* CLEAR STORAGE */
    Cd&=CARDMASK;  /* RANGE CHECKING */

    /* CHECK INIT FLAG */
    if (SDC36015_Init[Cd])
        {
         /* LOAD ERROR AND COUNT REGISTER */
         LosData = inp(SDC36015_Addr[Cd]+SDC36015_REG_ERRCNT);
         /* CHECK LOS */
         if (LosData&0x10) LosData=1; else LosData=0;
        }
    /* RETURN DATA */
    return (LosData);
/*--------------------------------------------------------------------------*}
```

```
int SDC36015_Read_Count ( int Cd )
{/*------------------------------------------------------------------------*/

    int CountData;   /* TEMP STORAGE */
    CountData=0;     /* CLEAR STORAGE */
    Cd&=CARDMASK;  /* RANGE CHECKING */

    /* CHECK INIT FLAG */
    if (SDC36015_Init[Cd])
        {
         /* LOAD ERROR AND COUNT REGISTER */
         CountData = inp(SDC36015_Addr[Cd]+SDC36015_REG_ERRCNT);
         /* MASK OFF COUNT */
         CountData&=0x0F;
        }
    /* RETURN DATA */
    return (CountData);
/*------------------------------------------------------------------------*/}

int SDC36015_Read_Status ( int Cd )
{/*------------------------------------------------------------------------*/

    int StatusData;  /* TEMP STORAGE */
    StatusData=0;    /* CLEAR STORAGE */
    Cd&=CARDMASK;  /* RANGE CHECKING */

    /* CHECK INIT FLAG */
    if (SDC36015_Init[Cd])
        {
         /* LOAD STATUS REGISTER */
         StatusData = inp(SDC36015_Addr[Cd]+SDC36015_REG_STAT);
         /* MASK OFF BITS */
         StatusData&=0x3F;
        }
    /* RETURN DATA */
    return (StatusData);
/*------------------------------------------------------------------------*/}
```

```
void SDC36015_Close (int Cd)
{/*-----------------------------------------------------------------------*/
    /* RANGE CHECKING */
    Cd&=CARDMASK;
    /* CHECK INIT FLAG */
    if (SDC36015_Init[Cd])
        {
        /* LOAD FINAL COMMAND WORD:   */
        /* DISABLE IRQ                */
        /* INHIBIT                    */
        /* CHANNEL 1 SELECTED         */
        SDC36015_Cmd [Cd] = 0X00;
        /* WRITE CMD REGISTER */
        outp (
            SDC36015_Addr [Cd]+SDC36015_REG_CMD,
            SDC36015_Cmd  [Cd]
            );
        /* CLEAR INIT FLAG */
        SDC36015_Init [Cd] = 0;
        }
/*-----------------------------------------------------------------------*/}

/* END SDC36015.C */
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C: PASCAL SOFTWARE

```
{
 SDC36015.PAS

 PASCAL SOFTWARE INTERFACE LIBRARY FOR USE WITH THE SDC-36015
 SYNCHRO RESOLVER TO DIGITAL CONVERTER CARD.

 ILC DATA DEVICE CORPORATION
 105 WILBUR PLACE
 BOHEMIA, NY 11716-2426
 (516) 567-5600

 V 1.2 CREATED WITH TURBO PASCAL V 7.0 12/1/94 BOB PASCAZIO x 7386
}

UNIT SDC36015;

INTERFACE

{ REGISTER MAPPING }

CONST SDC36015_REG_CMD      =    $00;
      SDC36015_REG_LSB      =    $00;
      SDC36015_REG_MSB      =    $01;
      SDC36015_REG_ERRCNT   =    $02;
      SDC36015_REG_STAT     =    $03;

{ CONSTANTS }

CONST  MAXCARD              =    $8;
       MAXCHANNEL           =    $6;
       CARDMASK             =    $7;
       IRQCARD              =    $0;
       TRACK                =    $1;
       INHIBIT              =    $0;
       NORMAL               =    $1;
       BITE                 =    $0;
       MODE_CONTINOUS       =    $0;
       MODE_LATCH           =    $1;
       ENABLE               =    $1;
       DISABLE              =    $0;
```

```
{ CONTROL COMMANDS }
                                                { New  Reg#  Posit   Mask     }
                                                { XXXX  X    XXX   XXXX XXXX }
CONST SDC36015_IRQENA  = $16BF;                 { 0001  0    110   1011 1111 }
      SDC36015_IRQDIS  = $06BF;                 { 0000  0    110   1011 1111 }
      SDC36015_INHIBIT = $04EF;                 { 0000  0    100   1110 1111 }
      SDC36015_TRACK   = $14EF;                 { 0001  0    100   1110 1111 }
      SDC36015_CRESET  = $15DF;                 { 0000  0    101   1101 1111 }
      SDC36015_C1      = $00F8;                 { 0000  0    000   1111 1000 }
      SDC36015_C2      = $10F8;                 { 0001  0    000   1111 1000 }
      SDC36015_C3      = $20F8;                 { 0010  0    000   1111 1000 }
      SDC36015_C4      = $30F8;                 { 0011  0    000   1111 1000 }
      SDC36015_C5      = $40F8;                 { 0100  0    000   1111 1000 }
      SDC36015_C6      = $50F8;                 { 0101  0    000   1111 1000 }
```

```
{ FUNCTION PRIMATIVES }

Procedure SDC36015_Initialize ( Cd, Addr : Integer );
Procedure SDC36015_Program (Cd : Integer; Cmd : Word);
Function SDC36015_Read_Angle ( Cd, Mode : Integer) : Word;
Function SDC36015_Read_Bite ( Cd : Integer ) : Integer;
Function SDC36015_Read_Los ( Cd : Integer ) : Integer;
Function SDC36015_Read_Count ( Cd : Integer ) : Integer;
Function SDC36015_Read_Status ( Cd : Integer ) : Integer;
Procedure SDC36015_Close (Cd : Integer);

IMPLEMENTATION

Var SDC36015_Addr : Array [0..MAXCARD] of Word;
   SDC36015_Cmd  : Array [0..MAXCARD] of Integer;
   SDC36015_Init : Array [0..MAXCARD] of Integer;


Procedure SDC36015_Initialize ( Cd, Addr : Integer );
{---------------------------------------------------------------}
Begin
    Cd:=Cd AND CARDMASK;             {RANGE CHECKING}
    SDC36015_Addr [Cd] := Addr;     {STORE ADDRESS}

    {DISABLE IRQ}
    {TRACK}
    {CHANNEL 1 SELECTED}
    SDC36015_Cmd [Cd] := $10;

    {WRITE CMD REGISTER}
    Port[SDC36015_Addr [Cd]+SDC36015_REG_CMD] := SDC36015_Cmd [Cd];

    SDC36015_Init [Cd] := 1;                {SET INIT FLAG}
End;
{---------------------------------------------------------------}
```

```
Procedure SDC36015_Program (Cd : Integer; Cmd : Word);
{————————————————————————————————————————————————————————————————————}
Begin
    If (SDC36015_Init[Cd]=1) Then {CHECK INIT FLAG}
      Begin
          {MASK OFF BIT(S)}
          SDC36015_Cmd[Cd] := SDC36015_Cmd[Cd] AND (Cmd AND $00FF);
          {AND NEW BIT(S)}
          SDC36015_Cmd[Cd] := SDC36015_Cmd[Cd] OR
          (((Cmd AND $F000) SHR 12) SHL ((Cmd AND $0700) SHR 8));
          {WRITE CMD REGISTER}
          Port[SDC36015_Addr [Cd]+SDC36015_REG_CMD] := SDC36015_Cmd [Cd];

          If (Cmd=SDC36015_CRESET) Then {CHECK FOR CNTR RESET}
            Begin
                {YES - RESET BIT TO 0}
                SDC36015_Cmd[Cd] := SDC36015_Cmd[Cd] AND $DF;
                {WRITE CMD REGISTER}
                Port[SDC36015_Addr [Cd]+SDC36015_REG_CMD] :=
                SDC36015_Cmd [Cd];
            End;
      End;
End;
{————————————————————————————————————————————————————————————————————}
```

```
Function SDC36015_Read_Angle ( Cd, Mode : Integer) : Word;
{----------------------------------------------------------------------------}
Var AngleData : Word;       { TEMP STORAGE }
Begin
    AngleData:=$0000;    { CLEAR STORAGE }
    Cd:=Cd AND CARDMASK; { RANGE CHECKING }

    { CHECK INIT FLAG AND CHANNEL }
    If (SDC36015_Init[Cd]=1) Then
      Begin
          { CONTINOUS MODE }
          If (Mode=MODE_CONTINOUS) Then
            Begin
                { YES - READ ANGLE }
                AngleData := (Port[SDC36015_Addr[Cd]+SDC36015_REG_MSB]
                            SHL  8 OR
                            Port[SDC36015_Addr[Cd]+SDC36015_REG_LSB] );
            End
         Else
           Begin
               { NO - LATCH MODE, SO INHIBIT }
               Port[SDC36015_Addr [Cd]+SDC36015_REG_CMD]
                            := SDC36015_Cmd [Cd] AND $EF;
               { READ ANGLE }
                AngleData := (Port[SDC36015_Addr[Cd]+SDC36015_REG_MSB]
                            SHL  8 OR
                            Port[SDC36015_Addr[Cd]+SDC36015_REG_LSB] );
               { TRACK }
               Port[SDC36015_Addr [Cd]+SDC36015_REG_CMD]
                            := SDC36015_Cmd [Cd] OR $10;
           End;
      End;
      { RETURN DATA }
      SDC36015_Read_Angle := AngleData;
End;
{----------------------------------------------------------------------------}
```

```
Function SDC36015_Read_Bite ( Cd : Integer ) : Integer;
{-----------------------------------------------------------------------}
Var BiteData : Integer;     { TEMP STORAGE }
Begin
    BiteData:=0;            { CLEAR STORAGE }
    Cd:=Cd AND CARDMASK;  { RANGE CHECKING }

    { CHECK INIT FLAG }
    If (SDC36015_Init[Cd]=1) Then
      Begin
          { LOAD ERROR AND COUNT REGISTER }
          BiteData := Port[SDC36015_Addr[Cd]+SDC36015_REG_ERRCNT];
          { CHECK BITE }
          If ((BiteData And $20)<>0) Then BiteData:=1 Else BiteData:=0;
      End;
    { RETURN DATA }
    SDC36015_Read_Bite := BiteData;
End;
{-----------------------------------------------------------------------}

Function SDC36015_Read_Los ( Cd : Integer ) : Integer;
{-----------------------------------------------------------------------}
Var LosData : Integer;     { TEMP STORAGE }
Begin
    LosData:=0;            { CLEAR STORAGE }
    Cd:=Cd AND CARDMASK;  { RANGE CHECKING }

    { CHECK INIT FLAG }
    If (SDC36015_Init[Cd]=1) Then
      Begin
          { LOAD ERROR AND COUNT REGISTER }
          LosData := Port[SDC36015_Addr[Cd]+SDC36015_REG_ERRCNT];
          { CHECK BITE }
          If ((LosData And $10)<>0) Then LosData:=1 Else LosData:=0;
      End;
    { RETURN DATA }
    SDC36015_Read_Los := LosData;
End;
{-----------------------------------------------------------------------}
```

```
Function SDC36015_Read_Count ( Cd : Integer ) : Integer;
{————————————————————————————————————————————————————————————}
Var CountData : Integer;   { TEMP STORAGE }
Begin
    CountData:=0;          { CLEAR STORAGE }
    Cd:=Cd AND CARDMASK;  { RANGE CHECKING }

    { CHECK INIT FLAG }
    If (SDC36015_Init[Cd]=1) Then
      Begin
          { LOAD ERROR AND COUNT REGISTER }
          CountData := Port[SDC36015_Addr[Cd]+SDC36015_REG_ERRCNT];
          { MASK OFF COUNT }
          CountData := CountData AND $0F;
      End;
    { RETURN DATA }
    SDC36015_Read_Count := CountData;
End;
{————————————————————————————————————————————————————————————}
```

```
Function SDC36015_Read_Status ( Cd : Integer ) : Integer;
{------------------------------------------------------------------}
Var StatusData : Integer;   { TEMP STORAGE }
Begin
    StatusData:=0;          { CLEAR STORAGE }
    Cd:=Cd AND CARDMASK;  { RANGE CHECKING }

    { CHECK INIT FLAG }
    If (SDC36015_Init[Cd]=1) Then
      Begin
          { LOAD STATUS REGISTER }
          StatusData := Port[SDC36015_Addr[Cd]+SDC36015_REG_STAT];
          { MASK OFF COUNT }
          StatusData := StatusData AND $3F;
      End;
    { RETURN DATA }
    SDC36015_Read_Status := StatusData;
End;
{------------------------------------------------------------------}

Procedure SDC36015_Close (Cd : Integer);
{------------------------------------------------------------------}
Begin
    { RANGE CHECKING }
    Cd:=Cd AND CARDMASK;
    { CHECK INIT FLAG }
    If (SDC36015_Init[Cd]=1) Then
      Begin
          { LOAD FINAL COMMAND WORD:   }
          { DISABLE IRQ                }
          { INHIBIT                    }
          { CHANNEL 1 SELECTED         }
          SDC36015_Cmd [Cd] := $00;
          { WRITE CMD REGISTER }
          Port[SDC36015_Addr [Cd]+SDC36015_REG_CMD] :=
          SDC36015_Cmd [Cd];
          { CLEAR INIT FLAG }
          SDC36015_Init [Cd] := 0;
      End;
End;
{------------------------------------------------------------------}

End.
{ END SDC36015.PAS }
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D: GLOSSARY

Interrupt Enable*: A logic low for this Bit enables the Interrupt circuitry on the card. In this condition when a LOS or BIT* is indicated by any of the converters on the card, AND the card is being addressed for an I/O Read or Write cycle, the interrupt circuitry will latch the Fault Indicator signals and generate an Interrupt (IRQ3) to the computer. The Interrupt Handler Routine must pulse this Bit High then Low to clear the interrupt. A logic High on this Bit disables the interrupt circuitry and forces the IRQ3 line to remain in the High (non-interrupt) state.

Counter Reset*: This Bit should normally be in the High state. When this Bit is pulsed Low then High the Turns Counter for every channel will be reset to zero. These turns counters are external to the hybrid converters and the SDC-14560 converters do not supply the necessary up/down and count signals required to drive them. **If turns counting is required the RDC/SDC-1920X series of converters must be used.**

Inhibit*: A High on this Bit enables the converters to track the Synchro/Resolver input signals. When this Bit is brought to the Low state all of the converters will freeze at the angle present at the time of the High to Low transition.

Channel Select: This three Bit address selects the active Synchro/Resolver converter channel during a Status/Count or Binary Angle I/O Read. Address 0 selects physical Channel 1.

BIT* Flag: This bit is normally at logic 1. A logic 0 indicates that the BIT flag has been asserted for the selected channel. This occurs if the loop error signal for the converter exceeds a preset level. This condition may arise momentarily if a large angular step is applied to the selected converter.

LOS Flag (1920X): This bit is normally at logic 0. A logic 1 indicates a loss of synchro or resolver signal to the selected converter.

Turns Count (1920X): These four bits contain the turns count information for the selected channel. The turns counters will increment/decrement, at the 0/360 degree transition, for each complete revolution of the input shaft angle. These turns counters are external to the hybrid converters and the 1920X series of converters must be utilized to drive them.

Fault (Channel 1-6): When a BIT* or LOS condition is detected by any converter AND the Interrupt circuitry is enabled (via the Control Word) AND the card is addressed for a I/O Read or Write cycle, the Status data is latched in the Status/Error Map register and the IRQ3 line is asserted causing an interrupt to the computer. The Interrupt Handler Routine must reset the IRQ3 line by pulsing the Interrupt Enable* bit in the Command Word register High then Low. A logic High Fault signal in any position of the Status/Error Map indicates that either a LOS or BIT* condition has occurred for the associated channel.

Note: If the SDC-36015 is populated with SDC-1456X hybrid converters (which do not support the LOS signals) jumper DS must be installed in the GND position to program the card to ignore these signals.

ILC DATA DEVICE CORPORATION®

ILC Data Device Corporation
105 Wilbur Place
Bohemia, NY 11716
(516) 567-5600