
AD16S

**Analog Input Interface for the Persistor CF-1
In Sandwich Card TM Format**

Oceanographic Embedded Systems

**1260 NE Seavy Ave.
Corvallis, OR 97330**

e-mail: mark@oes.to

Release 0.2 January 2000

Introduction

The AD16SS is a peripheral interface for the Persistor CF-1 that provides four channels of analog input with 16-bit resolution. Each input channel can be configured to accept signals of +/-10V, 0 to 5.00V, or 0 to 4.095V. The interface requires a +5V supply to be present on pin 5 of the CF-1 interface to provide power for the A/D converter. The AD16S complies with the preliminary Sandwich Card specification from Persistor Instruments and include an onboard PIC processor for card identification and to control the +5V power to the converter.

Setup and Testing

Setting up the AD16S is as simple as plugging the CF-1 into the AD16S, connecting power, serial I/O and analog inputs, then running one of the test programs. Generally, you will plug the AD16S into a base board of some kind, which will provide power to the AD16 and CF-1.

Serial Port Connections

The serial I/O connector provides the same RS-232 signals implemented on the Recipe cards from Persistor. The header is compatible with the IDC-10 to DB9F connector also provided by Persistor. The pin connections of this cable are shown in figure 1.

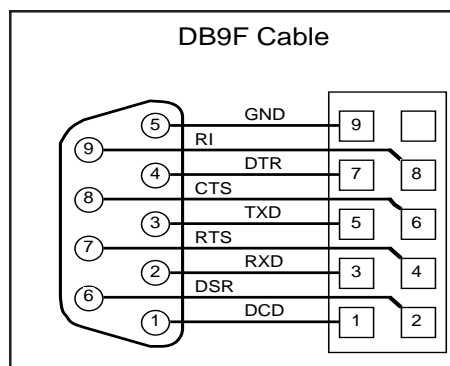


Figure 1. IDC10 to DB9F

Analog Input Connections

The AD16S can accept analog inputs either through the four pairs of solder pads on 0.2" centers or through the pads on 0.1" centers. The first set of pads is designed to allow easy connection of separate Signal/Ground pairs from external sensors. The second set of pads is compatible with a standard ribbon cable connector. The pin definitions for the connectors are shown in figure 2.

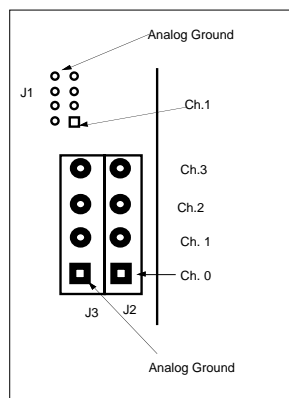


Figure 2. Analog Inputs

Power Supply Input

The AD16S accepts +3.3V power from Pin 11 (VLIN) of Connector C on the CF-1. It also requires +5V on pin 5 of Connector C. This will normally require a separate power supply base card or a recipe card set up to deliver regulated +5V to pin 5. Alternatively, you may use a regulated +5V supply to provide power to both the AD16S (through pin 5) and to the CF-1 (at pin 13—VBAT).

The +5V supply to the A/D converter can be switched off by issuing commands to the AD16S Sandwich Card Supervisor PIC chip. The PIC uses one output pin to control a MOSFET switch in the +5V power circuit. The switch is designed to automatically disconnect the +5V supply when the CF-1 is in the low-power Suspend mode.

Test Software

The diskette provided with the AD16S includes a number of compiled applications that you may use to test your AD16S and CF-1. You may also download this software, and other example programs as they are developed, from the OES web page (www.oes.to)

Hardware

The AD16S is assembled on two-sided printed circuit board using both surface-mount and through-hole components. The arrangement on the components on the board is shown in figure 3. There are some minor differences in the productions PC boards.

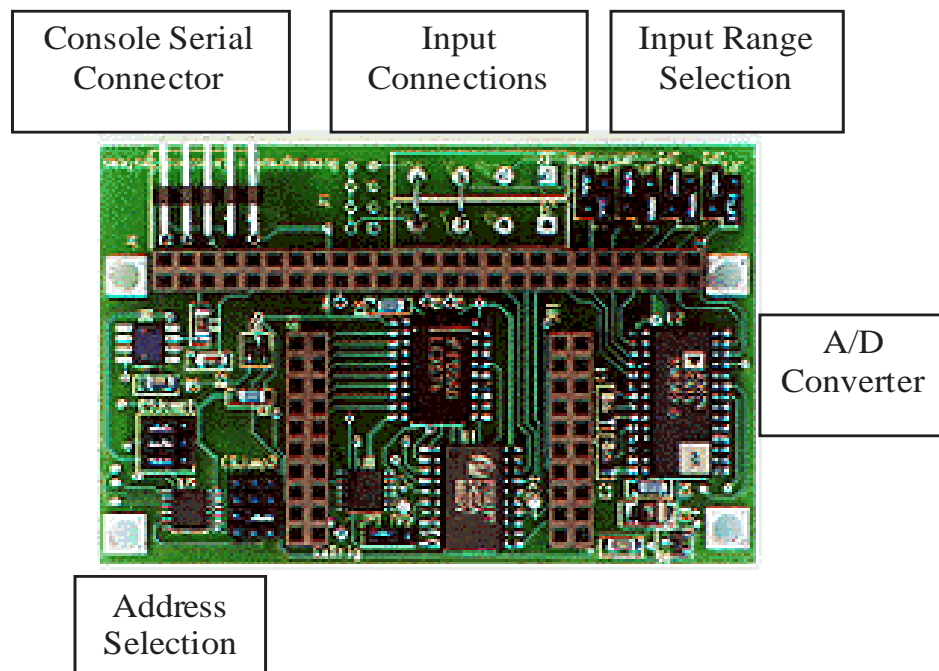


Figure 3. Component Arrangement on AD16S (prototype board shown)

Analog-to-Digital Converter and Interface

The AD16S uses the Analog Devices AD974 4-channel, 16-bit ADC. This converter provides a combination of low power, wide input range, and convenient serial interface. It also has an internal reference voltage and runs on a single +5V supply. The channel selection bits, serial data and clock, and start conversion signal are buffered by a 74LPT245 tri-state bus interface circuit. This IC provides I/O signals which are compatible with the 5V TTL levels of the ADC while operating from the 3.3V linear supply of the Persistor CF-1. The outputs of the 74LPT245 can be switched into a high-impedance mode when the +5V supply power to the ADC is shut down. A 74LC74 Octal buffer IC is used to provide four latched output bits to control the A/D multiplexer, the tri-state enable of the 74LPT245, and the PowerDown input of the A/D converter. The ADC is operated as a master device on the CF-1 QSPI interface to provide the greatest possible data acquisition speed. This mode of operation is not directly compatible with the Queued PicoBus operation supported by Persistor Instrument—which operates the CF-1 as the master and the peripherals as the slaves. However, all the QSPI signals of the AD16S are routed through the tri-state buffer and can be completely disconnected from the QSPI under software control.

Input Range Selection

There are four pairs of input jumpers on the AD16S. Each pair can be set up to select one of three input ranges or a test input according to figure 4. The jumpers are normally set for the +/-10V range when the card is shipped.

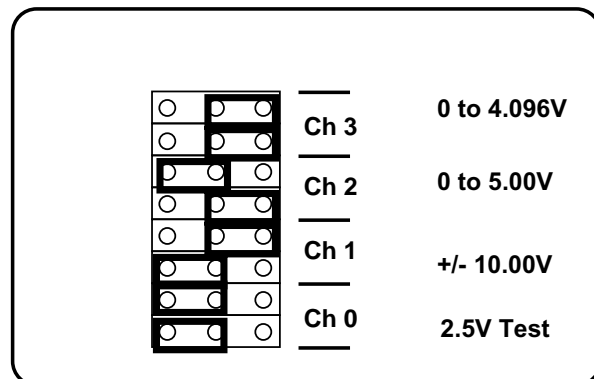


Figure 4. AD16S input range selection jumpers

Sandwich Card Address Selection

Persistor Instruments has selected the address ranges from FFFFF8000 to FFFFBFFF and FFFF0000 to FFF3FFF for Sandwich Card s. The first address range is decoded using CS8 and the logic on the card. The second range is decoded using CS10. Each of these ranges is divided into 16 blocks of 1Kbyte length for each sandwich card. The details of the addressing are covered in the “Sandwich Card Chip Select Reference” from Persistor.

The ADS16 is normally assigned to memory Slot 13 at address FFFFB400. This address is selected by setting the CSJUMP1 jumpers for +0 and +8, and the CSJUMP2 jumper for +5. Slot 13 is normally ‘undefined’ and is used so that slots 17 and 18 can be reserved for parallel A/D converters. The jumper setup for Slot 13 is shown in figure 5.

NOTE: CJ1 jumper positions may not exactly match those in the Persistor Sandwich Card documentation. The jumper positions on other sandwich cards may or may not follow the Persistor layout conventions. Please consult the documentation specific to the card when you set up the jumpers.

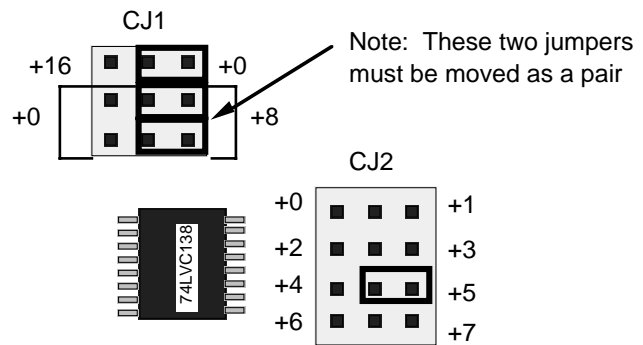


Figure 5. Sandwich Card Address Selection. Slot 13 Default selection is shown.,

Serial I/O Connector

The serial port header has the same pinout as the header on the hasty-pudding recipe card. It is compatible with the ribbon-cable and 9-pin connector provided with that interface.

A to D Converter Trigger Selection

The AD974 requires a low-going pulse of at least 100nSec width to start a conversion. The AD16S allows you to provide this signal from three different sources:

1. From an output write strobe generated by the address decoding circuitry (the factory default setting).
2. From a timer output bit on connector C of the CF-1.
3. From an external source applied to the ExTrig input pad.

Option 1 generates the convert signal when the software writes any byte to (Base+3) where Base is the Base Address of the memory segment used by the AD16S. The convert signal will be approximately 200nanoseconds in duration. This method of generating a strobe is much faster than the two separate write operations needed to clear and reset a standard I/O bit.

To use Option 2, you must connect a jumper wire between the ExTrig pad and one of the CF-1 timer output bits. The AD16S has test pads on pins 24, 26 and 34 for this purpose. These pins are the outputs of Double-Action timer modules in the CF-1. These pins can be configured to provide jitter-free triggering of the A/D at a wide range of precisely timed intervals. This method is best for collecting data at precise intervals for applications such as spectral analysis as the timing of collection is not affected by instruction timing which can cause jitter in the most carefully written interrupt-driven timing routines. If multiple channels must be collected, you can also switch the multiplexer and control the conversions with standard PinClear() and PinSet() function calls.

Option 3 allows you to connect an input signal to the A/D through the ExTrig pad to control the conversion. The input signal must be a low-going strobe of at least 100nSec. duration. Since the signal passes through the 74LPT245, it may use either 3.3V or 5V signal levels.

Software

AD16S Library

A library of software routines for the AD16S is provided in MetroWerks library format for the MetroWerks CodeWarrior development system. The library files (AD16S.lib and AD16S.h) provide routines to initialize the A/D interface, control the +5V power and ADC shutdown, and collect data from the ADC.

The source code to the AD16S library is proprietary information, but can be licensed by special arrangement with OES.

Environment Variable Setup

The AD16SLibrary routine depend on a properly set environment variable to set the base address of the card. Environment variables are stored in the virtual EEPROM of the CF-1, and can be set in the same manner as those in an MSDOS system. The PicoDos command:

```
SET SCB.13=AD16S.3000.1006.100
```

would set an environment variable for an AD16S in Slot 13. Since you must know the manufacturer's ID (3000) and the serial number (1006) for the card, we have provided an application to automatically read those variables from the AD16S, then scan all the slots to find the AD16S. When one is found, you are prompted to save an environment variable. The application is ADSETENV.RUN. It requires PicoDos 2.2 for proper operation.

Sample Program

The only test program included with the Rev 0. boards is ADSLTest which allows testing of the board with several commands:

- c Collects 1000 data points for each channel
- s Displays the mean and std. deviation for each channel using the data from the previous C command
- 0..3 Shows a histogram of data from the selected channel
- p Puts the AD16S into the low-power mode. +5V is shut off.
- w Wakes up the AD16 card.

If you use the 'p' command to shut down the A/D, any subsequent command will automatically wake up the A/D. However, the results of a collection or histogram display will be skewed as the AD974 is probably still settling.

ADS16Lib Library Routines

The ADS16Lib library contains software routines you can use to collect analog data with your ADS16 card. The library is provided as a binary library which you can integrate into your CodeWarrior® project. You have also received a header file defining the interfaces to the routines that you must include in your project. The following paragraphs describe the library routines and their usage.

The library defines a data structure to hold key information about each ADS16 found in the system. An array of four of these structures will allow future versions of the library to accommodate as many as four ADS16s in a system. (The current version of the library stops checking after the first card is found)

```
typedef struct {
    SCSDevice addev;
    vucptr adbase;
} adsdevtype;
```

The SCSDevice component holds a pointer to the Sandwich Card Supervisor device record for the card. This record provides information that allows the software to retrieve information about the sandwich card and to use the supervisor to control the +5V power switch on the card.

Four global unsigned short variables are defined in the library to allow you access to the A/D results when you collect data from all four channels at once.

```
ushort ad0,ad1,ad2,ad3;
```

Function Prototype

```
short ADSFindCards(bool verifycard);
```

Parameters

verifycard determines whether the routine will actually test the hardware. The testing may have side effects in systems with multiple sandwich cards.

Return Value

The result will be the number of cards found. The routine will have filled in that many entries in the adsdevs array.

Usage

This function is used to step through the sandwich cards in a system and look for ADS16s. If an ADS16 is found, the routine looks for a matching environment variable which defines the slot address of the card. If the environment variable is found, the entry in the `adsdevs` array is filled in.

Function Prototype

```
void SetMux(unsigned char chan, vucptr adsbase);
```

Parameters

Chan specifies which of the four channels (0..3) of the card will be selected. *adsbase* specifies the hardware base address of the card. This value is normally retrieved from the `adsdevs` array.

Usage

This function sets the channel for which data will be collected. It has the intended side effect of turning on the tri-state buffer which connects the A/D converter to the SPI pins of the CF-1.

Function Prototype

```
void SetupQSMSlave(void);
```

Parameters

None.

Return Value

None

Usage

This routine converts the QSM on the CF-1 to slave mode. The A/D on the ADS16 then controls the clocking of data into the QSM. This has two advantages: the data is clocked in at 6mHz (50% faster than possible with the CF-1 running at 16mHz); and the data is properly aligned in a 16-bit word. (As a QSM slave, the A/D requires two extra clocks, which results in a mis-aligned result in the QSM receive register.

Function Prototype

```
void GetADSAll(vucptr adsbase);
```

Parameters

adsbase defines the base address of the ADS16.

Usage

This routine collects data from all four channels of the A/D and stores the results in the global variables `ad0` to `ad3`. The routine requires about 40 microseconds to collect and store all four values (with the CF-1 running at 16mHz). In many cases, it may be easier to call this routine at interrupt time with a PIT chore, then simply allow your foreground program to treat the `ad0` to `ad3` values as magical variables that represent the analog voltages at any time. The PIT chore could also be set up to digitally filter the inputs as they are collected.

Function Prototype

```
unsigned short GetADS(unsigned char chan, vucptr adbase );
```

Parameters

Chan specifies which of the four channels (0..3) of the card will be selected. *adbase* specifies the hardware base address of the card. This value is normally retrieved from the `adsdevs` array.

Usage

This routine collects and returns the analog data from a single channel. Since the A/D always returns the result of the previous conversion, this routine actually does two conversions--the first may return the result from a different channel. The second conversion returns the result from the desired channel. Since two conversions are required, this routine takes about 12microseconds at 16mHz.

Return Value

The unsigned short integer represents the A/D counts. The value represented by the count depends on the setting of the input jumpers for the channel.

Function Prototype

```
void ADSLowPower(vucptr adbase);
```

Parameters

adbase defines the base address of the ADS16.

Usage

This routine sets the PWRDN bit on the AD974. This turns off the internal clocks of the A/D and reduces its power consumption. The overall current requirements of the ADS16 drop by about 10mA. This routine does not turn off the +5V power to the A/D, so you can start converting again in a few dozen microseconds.

Function Prototype

```
void ADSPowerDown(vucptr adsbase, SCSDevice adsdev);
```

Parameters

adsbase defines the base address of the ADS16. *adsdev* is a pointer to the data structure associated with the PIC sandwich card supervisor.

Usage

This function will set the A/D control buffer to the tri-state condition to isolate the A/D from the QSPI bus, then turn off the +5V power to the A/D using the control bit on the PIC sandwich card supervisor. It will actually take several milliseconds for the power to drain off the capacitors on the +5V supply.

Function Prototype

```
void ADSPowerUp(vucptr adsbase, SCSDevice adsdev);
```

Parameters

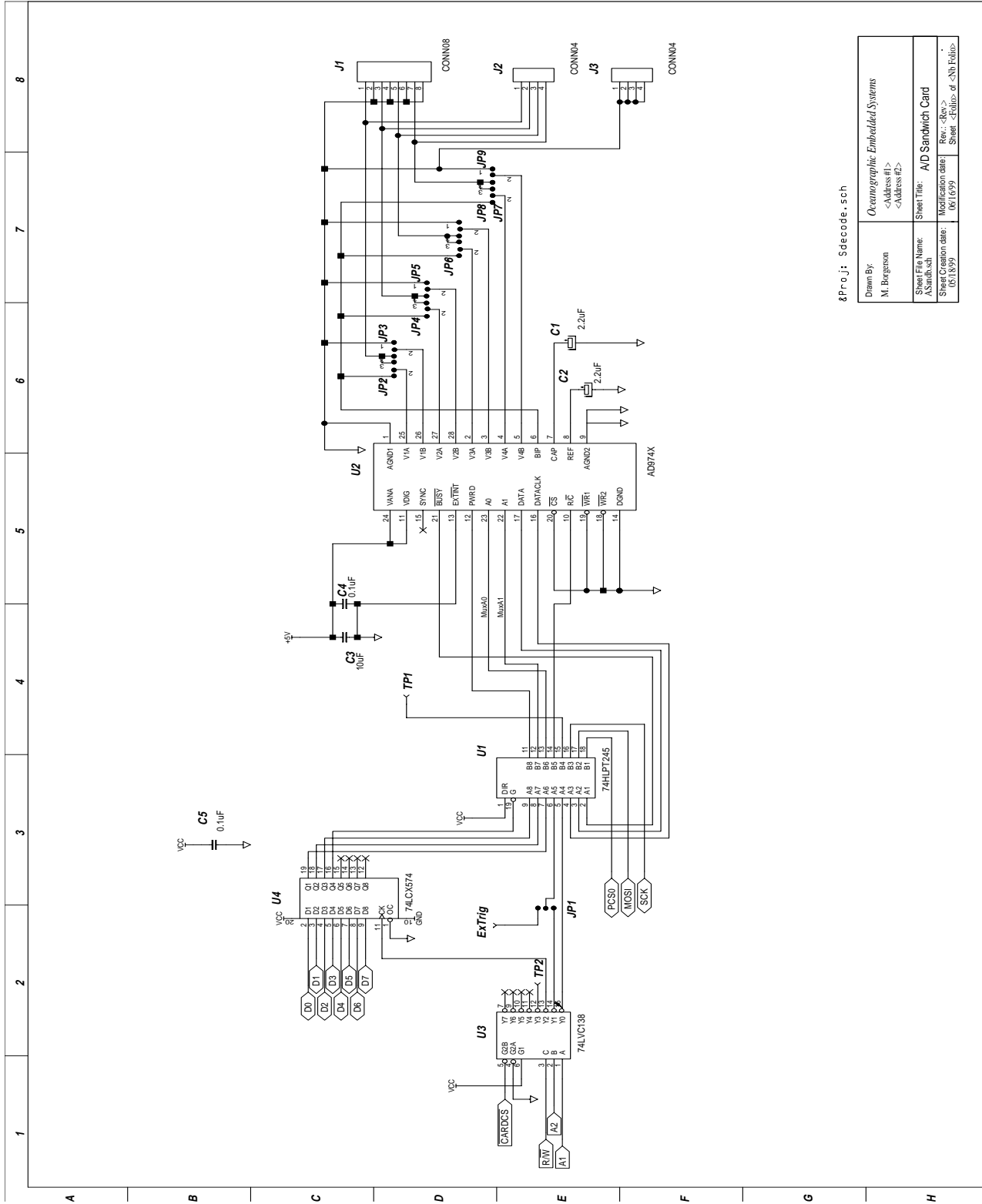
adsbase defines the base address of the ADS16. *adsdev* is a pointer to the data structure associated with the PIC sandwich card supervisor.

Usage

This function turns on the +5V power to the A/D using the switch controlled by the sandwich card supervisor. It will take several milliseconds for the +5V supply to stabilize. It will also take up to 100 milliseconds for the A/D to start up its internal clocks and stabilize its operation. For this reason, you should allow a generous delay interval when you shut down, then power up the A/D converter.

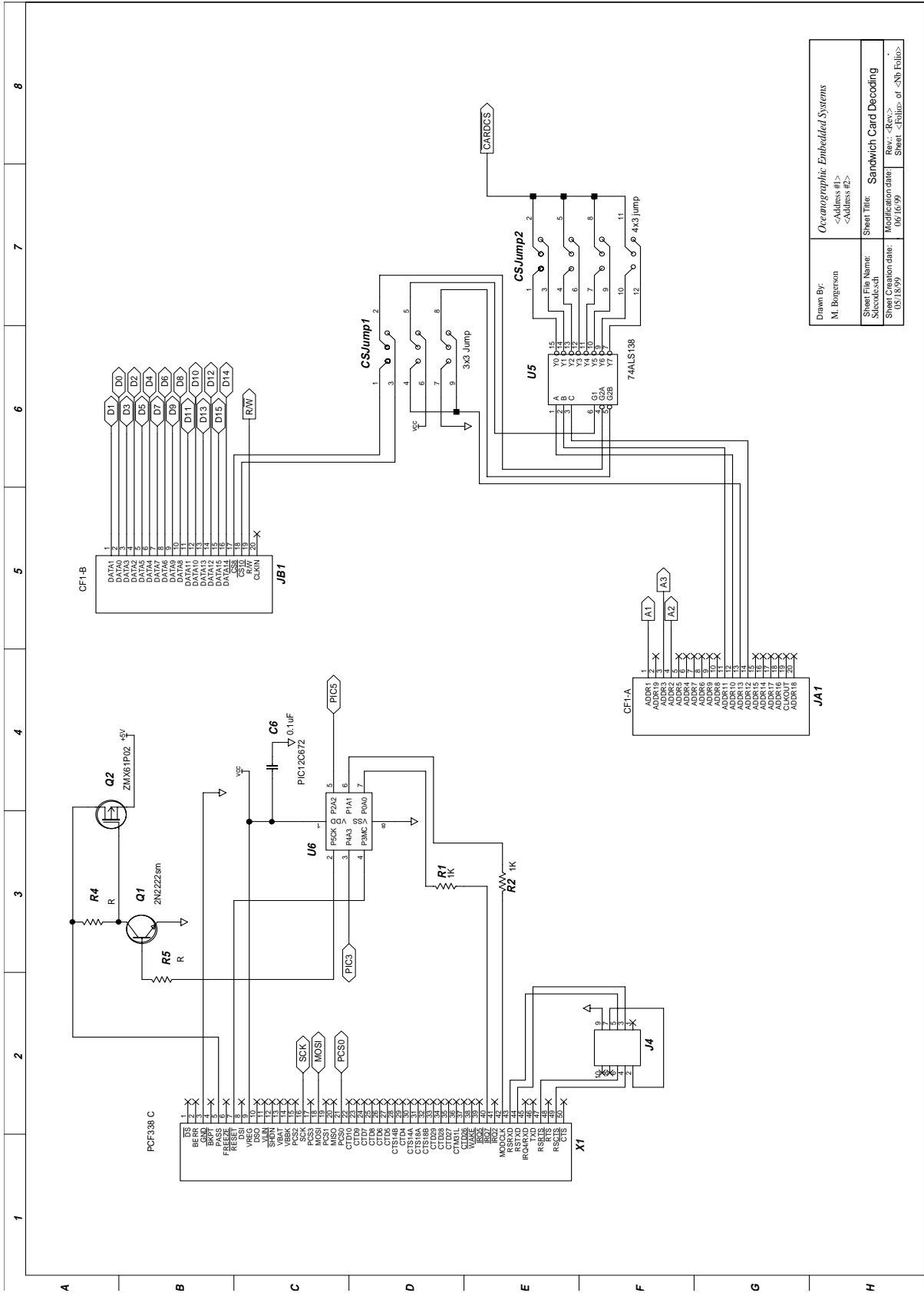
NOTES For Release 1:

The trigger selection jumper (JP1) is reversed from layout on the prototype circuit boards. To select the internal trigger signal, the jumper must connect the two pins nearest the 74LPT245. This is the default setting when the board is shipped.



8Proj: Sdecode.sch

Drawn By:	Oceanographic Embedded Systems
Sheet File Name:	<Address #1>
Sheet Title:	<Address #2>
Sheet File Name:	AD Sandwich Card
Sheet Title:	Rev: <Rev>
Modification date:	06/16/99
Sheet:	<Nb> of <Nb> Pages



Drawn By: M. Borgeson	Oceanographic Embedded Systems -<Address #1-> -<Address #2->
Sheet File Name: Slecwfc.sch	Sheet Title: Sandwich Card Decoding
Sheet Creation date: 06/16/99	Modification date: Rev.: -<Rev.> Sheet -<Folio> of -<Nb Folio>