An NCS Standard Interface for the XDS 900 Series Computers

W. Lushbaugh
Communications Systems Research Section

The Network Control System standard interface has been adopted as a standard computer interface for all future minicomputer-based subsystem development for the Deep Space Network. This article describes a standard interface adapter for existing XDS 900 series computers that will enable the design of equipment compatible with the presently available machines and easily transferable to new minicomputers as they become available.

I. Introduction

The Network Control System (NCS) standard interface has been adopted as a standard computer-interface for all future minicomputer-based subsystem development for the DSN. In order to design and test equipment for currently existing 900-series machines and be able to transfer that equipment conveniently to new minicomputers as they become available, a standard interface adapter (SIA) is needed which can be attached to the 900 series machines. The following article provides a design overview of such an SIA.

II. Review of 900 Series I/O Structure

The 900 series XDS computers are 24-bit machines with input/output (I/O) performed in the register mode, i.e., transfers to and from an external piece of equipment are on a 24-bit basis. An energize output machine (EOM) instruction and parallel input (PIN) and parallel output (POT) instructions perform I/O on a word-by-word central processing unit (CPU) controlled basis. Executing an EOM-POT (or PIN) sequence delivers (or receives) data from the device addressed by the EOM. An EOM-POT (or PIN) sequence cannot be interrupted by an external interrupt.

Two interrupts are typically used in an I/O sequence. One is a data interrupt to signify that the input/output buffer is full/empty, while the other interrupt is used to signal special conditions such as end of message or errors.

Reference 1 gives a detailed description of the XDS 900-series computer interfacing for those who find the above description too brief.

III. Review of the Standard Interface Format

The NCS standard interface output consists of fourteen signals and a power sense line. The fourteen signals consist of eight data signals, two function code signals and four control or handshaking signals. Three of the four combinations of function codes are available for tagging data transmission while the fourth, the 1-1 condition, is reserved for commands out of the computer or status in from the device connected to that computer. Two of the control signals are used as stimulus signals; one from computer (STC) and one from the device (STD). Both of these signals are unidirectional and go true for the complete duration of a message. The remaining two signals are bidirectional control signals called response (RSP) and ready (RDY), which control the handshaking of data across the interface. Response and ready each make one complete cycle (false to true to false) for each byte transferred across the interface.

IV. Programming Characteristics of the Interface

To the programmer, the standard interface appears to be a device, with two associated EOMs. One EOM is used to channel the input or output of the data to the interface while the other is used to communicate status information to and from the program.

A typical data-out sequence is as follows. A status EOM-POT sequence is executed by the program to alert the interface that the data output mode is starting. The hardware will generate a data interrupt at this point to notify the software that the output buffer is empty. The software responds to this interrupt by executing an EOM(DATA)-POT sequence which fills the output buffer and clears the interrupt. This sequence of data interrupts and data POTs will be continued until the desired number of words has been transferred from the computer. There will be one last data interrupt telling the program that the buffer is empty, at which time a status EOM-POT sequence is executed, which terminates the data output mode and automatically clears the data interrupt. This normal data-out mode only uses one of the two interrupts, i.e., the data interrupt. If at any time during this mode, the device connected to the other end of the SIA interrupts the data out mode by either a status insert or by turning the data transmission around by initiating the data input mode, the program would be notified by a service interrupt.

The data-in mode is started by the device. The hardware in the SIA holds the data-in buffer ready when not in use so that three bytes will be transferred to the computer before the first data interrupt occurs. This data interrupt is the same physical interrupt as used in the data out mode, but no confusion can result if the interface is programmed properly. For example, the status of the device can be read at any time to clarify which type of data interrupt is occurring.

V. Status Register

The status register is the means by which software and hardware communicate. The software delivers commands to the SIA via the output status register and reads back various status information from the input status register. Despite this split input/output function, the status register is a single 24-bit register.

A list of input and output bit assignments is given in Table 1. Following is a detailed description of each signal.

C_{23}-C_{16} Command Byte/Status Byte. These 8 bits are used to communicate to the device connected to the SIA. When a command is to be sent from the computer to the device (see bit 15) the contents of these 8-bit locations are stored in the SIA and transmitted to the device at the proper time in the handshaking sequence.

When the device delivers an 8-bit status byte to the computer, these 8 bits are stored in the SIA and delivered to the computer following an EOM (STATUS)-PIN sequence. These bits are only changed when a new status byte is delivered and so are meaningless unless C_7 is set (see C_7).

C_{15} Command Flip Flop. Setting this bit via an EOM POT STATUS sequence, causes the hardware to interrupt (at the proper time) whatever I/O process is happening and send a command to the device. If this bit is not set, the contents of C_{23} through C_{16} do not get copied into the SIA.

C_{14}-C_{13}. These two bits are the two function codes to be used in the data output mode or the received function codes in the data input mode. These two bits must be set to the desired value at the beginning of each data transmission AND MUST NOT BE CHANGED until the next message.

When the hardware in the SIA detects the data input mode, it switches these two inputs from the output function code flip-flops to the input function code flip-flops, so that these two bits have different meanings depending upon what the hardware is doing.

C_{12} Data Request. Setting this bit causes the hardware to attempt the data output mode to the device. This request will be honored as long as the device does not override it. Resetting this flip-flop terminates the data output mode.

C_{11}, C_{10} — B_1, B_0. B_1, B_0 are interpreted in the hardware as a binary number (B_0 = LSB) giving the number of bytes per word to be transferred in the data output mode, or the number of bytes received before a data interrupt is generated, in the data input mode. It is possible to change these bits during a message, but this should only be done during a data interrupt.

C_9 Reset SIA. Setting a 1 in this position triggers a 100-ns pulse that resets the SIA.

C_8 Reset Device. Setting a 1 in this position sets a flip-flop which sends a RESET COMMAND to the DEVICE. To release the device reset, a zero must be set in this position.

C_7 Status Byte Here. A one read in at this position means that a status byte was received from this device. This bit causes a status interrupt and may be reset by putting a one in this position or by RESET. When this bit is set, data output mode is temporarily cut off.

C_6 ΔSTD. This bit is set to 1 when STD (stimulus from the device) makes a 1 to 0 transition. This bit causes a status interrupt and can be reset by putting a 1 in this position or by RESET.

C_5 STD. This is a raw copy of the signal stimulus from device (STD).

C_4 Device Power. This bit = 1 when device power is off. This will cause a status interrupt to be generated as long as this signal is set, but this can be masked out of the interrupt by a 1 in this position. Setting a 1 in this position does not affect the value read in during a status PIN.

C_3 Device Override. This bit is set to 1, causing a status interrupt when a data byte is received while the data request (see C_{12}) is set.

C_2-C_1, R_1-R_{11}, Respectively. These two bits give the number of received bytes in UNARY. Both are reset to "0" on an EOM-PIN of data (or start or reset). A data interrupt is developed when $R_1 = R_{11} = 1$ and a third byte has arrived.

C_0 Data Interrupt. This bit is a copy of the data interrupt.

VI. Design Philosophy

Since it was felt that most noise pulses encountered in practice are high-energy short-duration voltage spikes, and furthermore that most system noise is synchronous with a system clock, it was decided to investigate the possibility of building an asynchronous (clockless) SIA. There was a further line of thought leading to this investigation, i.e., that there is no inherent demand at either end of the interface to have a clock. Certainly the 900-series computer end of the link could use computer timing signals where necessary and the interface would supply other necessary timing signals, e.g., data here, data taken, etc. There also is no need to supply a clock at the device end, for if the device does have a clock, it will not necessarily be the 5-MHz reference required in the DSN specification.

VII. Design Results

The design objectives were completely met in that, a completely clockless interface was built and demonstrated. The heart of the system is the low-pass filtered line receiver shown in Fig. 1. The line receiver is National Semiconductor's DM8820A, a dual line receiver in a 14-pin package. Figure 1 shows one of the control signals terminated in the standard 220/330 Ω terminator and feeding an RC network consisting of the 1 kΩ resistor and the 220 pF capacitor. This RC network forms part of the low-pass filter, as the device itself with the feedback as shown also has an inherent filtering capability of about 10 MHz. The 4.3 kΩ resistor to ground from pin 3 of the DM8820A causes the device to compare the signal on pin 1 to approximately 1.7 V when the output is high, while the 1 kΩ and the diode to the output drops the comparison voltage to about 1.1 V when the output is low. The combined effects of these external components is to add hysteresis, and increase the noise rejection to greater than half the voltage swing in both the high and low signal states.

Laboratory tests of the combined filter and line receiver gave the following results: (a) A square wave input varying between 0.3 and 3.0 V of 3 MHz or higher did not get through the line receiver; (b) A single positive-going pulse of less than 200 ns never got through the line receiver, and some circuits rejected pulses as wide as 250 ns; (c) A single negative going pulse (i.e., signal normally 3 V pulsing to 0.3 V) of less than 200 ns would never trigger the output of the line receiver, and some circuits rejected pulses of 250 ns. The variation in pulse widths rejected can be attributed to variations in external components as well as variations in the line receivers themselves.

Figure 2 shows RSP (on the upper trace) and RDY (on the lower trace) of an actual handshaking sequence between the SIA for a 900-series computer as it was connected to an external test box which had the full complement of handshaking logic. The signals are as they appear inside of a short (negligible delay) cable. The horizontal calibration is 500 ns/cm, while the vertical scale is 1 V/cm, with ground at the bottom reticle and center reticle for the lower and upper traces, respectively. The upper trace showing RSP from the test box (a device), starts the sequence asserted, as the SIA on the 910 computer does not have any data ready. When the 24 bits of data are loaded into the SIA, RDY is asserted by the SIA, as seen on the lower trace of Fig. 2. Approximately 300 ns later, the device recognizes the assertion of RDY and generates a 100-ns data strobe pulse (not shown) after which RSP goes high. Approximately 250 ns later the SIA recognizes the raising of RSP, and it completes the cycle by raising RDY. A new cycle is then initiated by the device about 350 ns later, when it observes the raising of RDY when it again lowers RSP. The total length of the handshaking sequence is about 1.3 μs. Three handshaking cycles are shown in Fig. 2, the sequence ending there, since the buffer in the SIA is now empty. All three pulses on both traces appear synchronously in Fig. 2, since the relative timing of each is derived from fixed delays rather than a system clock. This synchronous feature considerably aided in the speedy debugging of the initial design and will certainly be helpful in repairing a malfunctioning SIA or device in the future.

VIII. Conclusions

An SIA for XDS 900-series computers has been built without using a system clock. Rather than validating the signal by interrogating it with two clock pulses, the signal is assumed valid only when it is present long enough (200 to 250 ns) to pass through a low-pass filter. The resulting SIA has a peak data transmission rate of approximately 770 kilobytes/s (minus software I/O time). The unit has been successfully tested using a completely functional clockless test unit. The unit has also been successfully tested over 100 m (300 ft) of cable and connected to the precision signal power measurement unit (PSPM). The PSPM is a device with an SIA adaptor, developed by the Digital Projects Group, which uses the double clocking method of filtering the received signals.

A device is now under development which will convert the twin-coax intercomputer communications link (Ref. 2) to a device with an SIA adaptor. This will enable any computer with an SIA adaptor to communicate via coaxial cable to another computer up to 600 m away.

References

1. Martin, W. L., "Digital Communication Tracking: Interfacing with SDS 900 Series Computers," in *The Deep Space Network, Space Programs Summary* No. 37-39, Vol. III, pp. 52–65. Jet Propulsion Laboratory, Pasadena, Calif., May 31, 1966.

2. Lushbaugh, W. A., "A Driver/Receiver Unit for an Intercomputer Communications Link," in *The Deep Space Network Progress Report*, Technical Report 32-1526, Vol. XV, pp. 109–113, Jet Propulsion Laboratory, Pasadena, Calif., June 15, 1973.

Table 1. XDS 900-Series 14 line interface status register

900-Series Bit Number				
23	Least sig. bit		D1	
22			D2	
21			D3	
20		Command byte on pot	D4	Line of 14 line interface
19		Status byte on pin	D5	
18			D6	
17			D7	
16	Most sig.bit		D8	
15	CMD F.F. Set one EDM Pot status. Reset by start and when CMD sent			
14 13	F_0 } Function codes to be used in data out mode, or F_1 } Received function codes			
12	DAT RQST. Indicates data output mode (not needed for CMD out)			
11 10	B_0 } B_1, B_0 are interpreted in the hardware as a binary number (B_0 = LSB) giving the number of bytes per word to B_1 } be transferred in output mode, or the number of bytes received before a data interrupt is generated, in input mode			
9	Reset. Setting a one in this position triggers a one-shot pulse to reset the 900 series end of the comm. link			
8	Reset device. Setting a 1 in this position sets a flip-flop which sends a reset CMD to the device. To release the reset a 0 has to be plotted in this position			
7*	Status byte here. Cuts off data out transfer until cleared			
6*	ΔSTD. Set to 1 when STD makes a 1 to 0 transition			
5	STD. Raw copy of STD line			
4*	Device power = 1 when device power is off, masked out of status interrupt by entering a 1 in this position			
3*	Device override. Set to one when a data byte is received when the hardware is in data out mode			
2 1	R_1 } Number of data bytes received in unary. Both are reset to 0 on EOM-PIN data or start. A data in interrupt is R_{11} } developed when $R_1 = R_{11} = 1$ and a third byte has arrived			
0	Data interrupt			

*Status bits 3, 4, 6, and 7 are either masked or reset by potting a 1 in the corresponding position of the status register. Any one of these bits set generates a status interrupt.

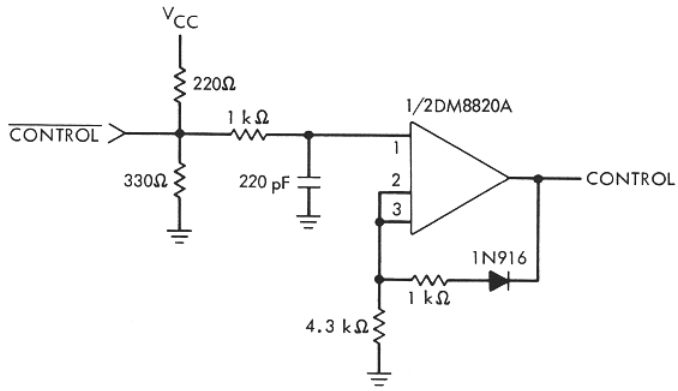

Fig. 1. Filtered line receiver with hysteresis

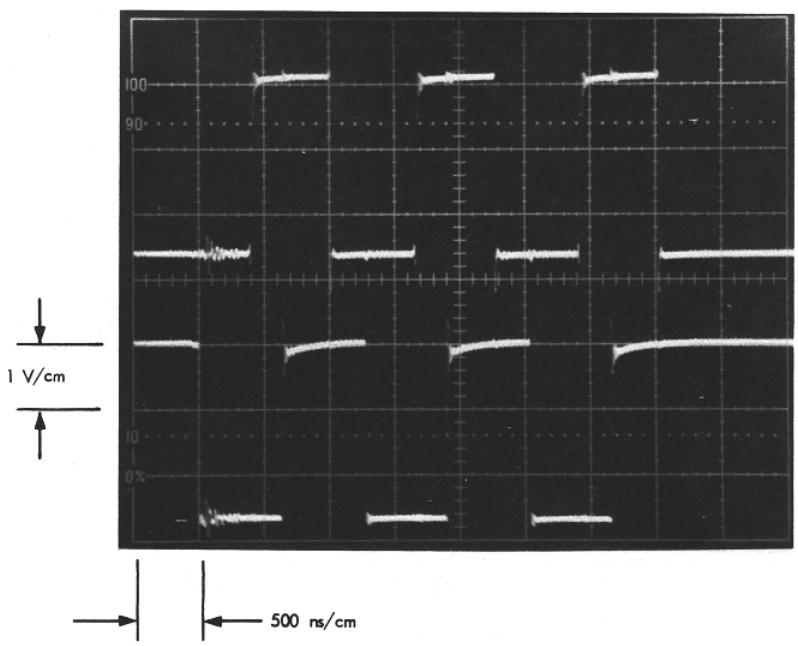

Fig. 2. RSP (TOP) RDY (BOT)