A COMPUTERIZED CORROSION MONITORING SYSTEM

Ed Schmauch
Conoco Inc.
PO Box 1267
Ponca City, OK 74603

Introduction

A system has been developed to automatically monitor corrosion by an electrochemical technique. The system strategy employed is to dedicate a microcomputer system to each electrochemical probe, and serially interface all microcomputers to a single personal computer (PC). The dedicated microcomputer systems are the topic of this paper. Each dedicated microcomputer controls the electrochemical scanning of its probe, acquires the electrochemical data, temporarily stores the data, and transmits the data to the PC upon request where it is analyzed and stored. All input/output (I/O) is interrupt-driven. The time critical work is done within assembly coded interrupt service routines (ISRs), while less critical work is done in high-level FORTH.

Microcomputer Hardware

The dedicated microcomputer systems are built around the New Micros Inc. 100 Squared single board computer which is based on the Rockwell R65F12 FORTH-based microprocessor. The R65F12 is an enhanced 6502 CPU which contains the fig-FORTH kernel in on-chip ROM. The R65F12 also contains 40 TTL I/O lines, 2 counter/timers, a serial port, and 192 bytes of on-chip RAM. The corrosion monitoring system has an additional 10 kilobytes of RAM, 2 K of which is used for variables and buffers and 8 K of which is used for electrochemical data. The 8 K is sufficient to retain data from repetitive scans for up to 12 hours, thus relieving the PC of any real-time responsibilities. The corrosion monitoring program is 4 K of headerless FORTH code which is burned into a 2732 PROM. TTL I/O lines are used to interface to both the digital-to-analog converter (DAC), which controls the electrochemical scanning, and the analog-to-digital converter (ADC), which acquires the data. TTL lines are also used to output the corrosion rate to a liquid crystal display. The system has been tested very successfully in the laboratory and there are plans to install it at an oil field water handling plant in the near future.

Electrochemical Corrosion Monitoring

An electrochemical scan consists of varying the current output from the probe and measuring the resulting probe potential. From this potential as a function of current data, the corrosion rate can be calculated. The DAC drives a Howland current pump circuit [1] which controls the current output of the electrochemical probe. An electrochemical scan starts at zero current output. The DAC is incremented by a certain step value (typically 0.1 mA) at regular time intervals (typically 1 minute). The potential of the probe is measured at each DAC setting with the ADC and typically ranges from -50 to 50 mV. When the DAC reaches the set maximum value (typically 0.5 mA), the scan direction is reversed and the DAC is decremented in the same stepwise manner through zero until it

reaches the set minimum value (typically -0.5 mA). At this point the scan direction is reversed again and the DAC is stepped back to zero. A scan both starts and ends at zero current output, simplifying repetitive scanning.

From the potential as a function of current data obtained during an electrochemical scan, the corrosion rate of the probe can be determined. Corrosion is measured as a penetration rate, typically milli-inches per year (MPY). Since the probe is constructed of the same type of steel as the pipe in which it is installed, this gives a measure of the corrosion rate of the pipe. Electrochemical determinations of corrosion rate are most useful when the corrosive media is a highly conductive liquid, such as salt water. The dedicated microprocessor estimates the corrosion rate using a simplified linear model of the corrosion process [2] and outputs this value to the liquid crystal display. More rigorous analysis of the electrochemical data resulting in a more accurate determination of the corrosion rate is done by the PC. Electrochemical scan parameters can be set by the PC through the serial interface.

Data Storage

The electrochemical data is stored in a linked list [3]. A pointer contains the address of the beginning of the data from the oldest electrochemical scan. Stored with each scan is a pointer to the next more recent scan. Zero is stored in the pointer with the most recent scan. When the PC requests electrochemical data, the data for the oldest scan is serially transmitted to the PC and the oldest scan pointer is updated to point to the next oldest scan. A zero in the oldest scan pointer indicates that no scans are currently stored, all have been transmitted to the PC.

The time critical work is done within assembly coded interrupt service routines (ISRs), while less critical work is done in high-level FORTH. Data passage between the ISRs and high-level FORTH is done through circular buffers [4]. Each circular buffer (CB) is 256 bytes plus 2 one-byte pointers. The first pointer contains the relative address where the next byte of data will be put into the CB and the second points to where the next byte of data will be fetched from the CB. The pointers are updated with each put to or fetch from a CB. When the two pointers are equal, the CB is empty. Data can be put into or fetched from CBs in either a byte or word format. CBs are used for ADC conversion integers, received serial characters, and serial characters to be transmitted.

Multitasking

A very simple multitasking scheme is employed which uses a fourth CB. When an ISR determines that a high level FORTH task is required, it places the code field address (CFA) of the FORTH word into the task CB. The high-level multitasker continuously checks the task CB and executes any contents:

```
: TASKER   BEGIN   BEGIN   TSK_CB 2GET_CB   UNTIL   EXECUTE   AGAIN ;
```

2GET_CB is used to fetch 2 bytes (1 word) from the CB. It expects the parameter field address (PFA) of the CB (in this example, TSK_CB) on the top-of-stack and leaves a flag indicating if there is any data in the CB. If the flag is true, the word fetched from the CB (in this example, the CFA of the task to be executed) is second from top-of-stack. The major advantage of this multitasker is that it utilizes CB operations which are required for other parts of the program. This sharing of operations simplifies program development and maintenance. TASKER is executed after system initialization, therefore high-level FORTH words are only executed under the direction of ISRs.

Interrupt-driven Input/Output

The dedicated microcomputer system utilizes five interrupts. Since the R65F12 has a single interrupt request line, on interrupt, program execution is vectored to a service routine which saves the accumulator and X index register, tests the interrupt flag register, and jumps to the appropriate routine to service the interrupt.

R65F12 counter/timer A is used as a real-time clock and is set to interrupt every 25 milliseconds (40 Hz). The clock ISR first updates the date and time. Then it checks if it is time to update the DAC value or to initiate data acquisition with the ADC. For maximum temporal precision, the actual writing to the DAC and initiating of the ADC is done within the clock ISR. The less time critical functions dealing with the DAC scanning and ADC initiation are handled by high-level FORTH words. These functions include determining the time and value for the next update of the DAC, determining the time for next initiation of the ADC, determining when the scan direction should reverse, and determining when a scan is complete and the next scan is starting. These high-level words are executed by the multitasker when the clock ISR places the appropriate CFA into the task CB.

ADC completion triggers an interrupt through an edge sensitive TTL input on the R65F12. The ADC ISR gets the data from the ADC register and places it in the ADC CB. A number of ADC conversions (typically 100) are box-car averaged [5] for each potential reading. If the ADC ISR determines that the number of ADC values required for averaging have been acquired and placed in the CB, the ADC ISR halts the ADC and places the CFA of the high-level ADC word in the task CB. Therefore, the high-level ADC word is not executed after each ADC completion, but only after a complete box-car. The high-level ADC word gets the raw ADC integers from the ADC CB, determines the average ADC value, and stores the average ADC value (potential) along with the corresponding DAC value (current) in the linked list for electrochemical data.

The PC controls each dedicated microcomputer system by issuing serial commands. Reception of a serial character by the microcomputer triggers an interrupt. The serial receive ISR reads the character from the the serial port receive register and places it in the receive CB. If the received character is a carriage return, the receive ISR puts the CFA of the high-level receive word into the task CB. Therefore, the high-level serial receive word is not executed for each character, but only after a complete command is received. The high-level receive word fetches the characters from the receive CB, interprets the PC command, and takes the

appropriate action. PC commands include setting the DAC scan parameters, setting ADC acquisition parameters, setting the time and date, and requesting that electrochemical data be transmitted.

Transmitting serial characters to the PC serial multiplexer requires that request-to-send and clear-to-send be used. Two interrupts are therefore required, one for serial tranmit register empty and one for clear-to-send asserted. When the PC requests data from a dedicated microcomputer, the microcomputer fetches the data from the electrochemical linked list, translates the binary numbers to ASCII, and puts the ASCII characters into the serial transmit CB. As soon as the first character is in the CB, request-to-send is asserted using a TTL output of the R65F12. When the PC serial multiplexer asserts clear-to-send, this triggers an interrupt through an R65F12 edge sensitive TTL input. The clear-to-send ISR enables interrupt on serial transmit register empty. The serial transmit ISR fetches the next character from the serial transmit CB and outputs it through the serial port. The serial transmit ISR always checks if clear-to-send is still asserted before outputting a character. If clear-to-send is no longer asserted, the serial transmit ISR disables interrupt on transmit register empty. When clear-to-send is reasserted, the clear-to-send ISR will re-enable interrupt on transmit register empty. When the transmit CB is emptied, the transmit ISR disables interrupt on transmit register empty and un-asserts request-to-send.

Conclusion

Assembly coded ISRs and high-level FORTH combine to give the corrosion monitoring dedicated microcomputer system maximum performance and maintainability. The use of assembly coded ISRs gives the system the required performance to easily handle the time critical tasks. The use of high-level FORTH to handle the non-time critical tasks eases program development and maintenance.

References

[1] R. A. Pease, EDN, January 20, (1983), p. 85.

[2] M. Stern and A. L. Geary, Journal of the Electrochemical Society, Vol. 104, No. 1, (1957), p. 56.

[3] A. Ralston, ed., Encyclopedia of Computer Science, Van Nostrand Reinhold Co., Dallas, (1976), p.1224.

[4] Ibid., p.188.

[5] R. E. Dessy, Laboratory Automation, American Chemical Society, Washington D.C., (1978), p. 157.