Microcomputer-Based Fault Detection Using Redundant Sensors

HECTOR P. POLENTA, ASOK RAY, SENIOR MEMBER, IEEE, AND JOHN A. BERNARD, SENIOR MEMBER, IEEE

Abstract—**The design of a prototype device that implements a redundancy management scheme for the on-line detection and isolation of faulty sensors in strategic facilities like nuclear reactors, hazardous chemical plants, and advanced aircraft is presented. Such a device is potentially useful for reducing the number of display devices in the control room and relieving the plant operator(s) from the tasks of assimilation and analysis of redundant sensor data as well as for enhancing the processing capabilities of the main computer. The fault detection device can be used as an integral part of intelligent instrumentation systems. The device has been built using an 8-bit microcomputer system and commercially available electronic hardware. The software is not restricted to any specific hardware. It is completely portable. The operations of the prototype device have been successfully demonstrated for real-time validation of sensor data at the MITR-II nuclear research reactor.**

I. Introduction

INTELLIGENT instrumentation systems in complex and strategic facilities such as nuclear reactors, hazardous chemical plants, and advanced aircraft are generally provided with redundant sensors to enhance safety and reliability [1]–[4]. If all redundant measurements are displayed in a central area, the instrumentation panels become crowded and are difficult to interpret. Moreover, human operators must assimilate and analyze these data and, should sensors disagree, determine which ones are correct. This situation can be mitigated through the use of computer-aided fault detection and isolation (FDI) techniques as reported in the current literature [4]–[13]. Implementation of such FDI techniques as part of intelligent instrumentation systems have been shown to be successful in nuclear reactors and aircraft [6]–[14].

A large plant may have many sets of redundant measurements, one for each essential process variable. Thus the same FDI procedure must be repeatedly executed for each redundant measurement set during a sampling interval. If this is done on a single processor, these operations will be sequential and may consume a nonnegligible portion of the total available time slot during each sampling interval. This may occur even though the execution time for a single FDI operation is not significant. Therefore, multiprocessing for parallel execution of FDI procedures for individual sets of redundant measurements is desirable, especially if this can be accomplished in a cost-effective manner.

A cost-effective approach for implementing FDI procedures in a multiprocessing environment is to base the FDI system on a factory-designed and -assembled microcomputer board and to construct the associated data acquisition system from commercially available integrated circuits (ICs). This permits each microprocessor module to be located near the group of sensors that it serves. Hence fewer wires need to be used for communication to the main computer. The microcomputer transmits the status of each redundant sensor and an estimate of the measured variable to the control room via an appropriate man–machine interface that is controlled by the main computer.

The concept of parallel processing for fault detection of individual plant variables has been introduced in one of our recent publications [11], which reports the development and implementation of a fault-tolerant control system using a single 16-bit microprocessor. The task of fault detection was carried out by subprograms that were placed at the lowest level in the hierarchy of system software, and these subprograms were repeatedly called (once for each process variable) during each sample time. The software structure allows systematic parallel operations of individual tasks corresponding to the respective subprograms if the existing single processor is replaced by a distributed processor that will consist of the main processor and a number of functionally identical microprocessors. Following a bottom-up approach, the low-level subprograms for fault detection can be emulated by functionally equivalent hardware. In that case, only minor modifications will be required in the software of the main processor for interfacing with the individual microprocessors that will perform fault detection for the redundant sensor data of the respective process variables.

An FDI procedure which has been experimentally validated at MIT's Nuclear Research Reactor (MITR-II) [7], [8], [11] and at Argonne National Laboratory's Experimental Breeder Reactor (EBR-II) [13] using 16-bit minicomputers was chosen for implementation on a commercially available inexpensive 8-bit microprocessor. The procedure makes use of a redundancy management scheme for sensor arrays and is computationally efficient. A prototype device was built for this purpose and was tested on-line.

The goal of this paper is to present the design and implementation of a prototype redundancy management

Paper IUSD, approved by the Industrial Control Committee of the IEEE Industry Applications Society for presentation at the 1986 Industry Applications Society Annual Meeting, Denver, CO, September 28–October 2. This work was supported in part by the Argentine Naval Service, in part by the Charles Stark Draper Laboratory, and in part by the Massachusetts Institute of Technology. Manuscript released for publication February 15, 1988.

H. P. Polenta is at Carlos Villate 1916, 1636-Olivos-Bz.As., Argentina.

A. Ray is with the Mechanical Engineering Department, The Pennsylvania State University, University Park, PA, 16802.

J. A. Bernard is with the Nuclear Reactor Laboratory, Massachusetts Institute of Technology, 138 Albany Street, Cambridge, MA 02139.

IEEE Log Number 8821465.

scheme for on-line detection and isolation of faulty sensors which can be used as an integral part of intelligent instrumentation systems in strategic facilities like nuclear reactors, hazardous chemical plants, and advanced aircraft. The paper is organized in six sections and one appendix. The concept of the fault detection and isolation algorithm and its implementation procedure are briefly discussed in Section II. Sections III and IV describe the salient features of electronic hardware and software design, respectively. The results of experimentation conducted at the MIT nuclear research reactor and the derived conclusions are presented in Sections V and VI, respectively. The Appendix provides three examples to illustrate the function of the FDI algorithm described in Section II.

II. Description of the Fault Detection and Isolation Procedure

The theoretical basis and details of mathematical derivations of the FDI procedure are given in [8]. The underlying principle of redundancy management for detection and isolation of faulty sensors is briefly described in this section.

A. Background of the FDI Algorithm

The redundant measurements of a process variable are modeled as

$$m = Hx + \epsilon \tag{1}$$

where m is the $(l \times 1)$ vector of measurements that are generated from sensors, H is the measurement matrix of dimension $(l \times n)$ and rank n, and x is the true value of the n-dimensional measured variable. The vector ϵ represents measurement errors such that, for normal functioning of each measurement, $|\epsilon_i| < b_i$, where b_i is the specified error bound for the measurement m_i.

A measure of relative consistency between redundant measurements is given by the projection of the measurement vector m onto the left null space of the measurement matrix H such that the variations in the underlying component Hx in (1) are eliminated and only the remaining effects of the error vector ϵ can be observed. An $((l - n) \times l)$ matrix V is chosen such that its $(l - n)$ rows form an orthonormal basis for the left null space of H, i.e.,

$$VH = 0 \qquad VV^T = I_{l-n}. \tag{2}$$

The column space of V is referred to as the "parity space" of H and the projection of m onto the parity space as the "parity vector" [15], which is represented as

$$p = Vm = V\epsilon. \tag{3}$$

From (2), it follows that

$$V^TV = I_l - H[H^TH]^{-1}H^T. \tag{4}$$

The columns $v_1, v_2, \cdots, v_l$ of V, that are projections of the measurement directions (in R^l) onto the parity space are called failure directions since the failure of the ith measurement m_i implies the growth of the parity vector p in (3) in the direction of v_i. For nominally unfailed operations, the magnitude $\|p\|$ of the parity vector remains small. If a failure occurs, p may (in time) grow in magnitude along the failure subspace, i.e., the subspace spanned by the specific column vectors associated with the failed measurements. If the fault is time-varying, then the failure directions (and hence the failure subspace) may also be time-varying. The increase in the magnitude of the parity vector signifies abnormality in one or more of the simultaneous redundant measurements, and its direction can be used for identification of abnormal measurement(s). A geometric interpretation of this FDI methodology, along with further details, is given in [8]. The parity vector p in (3) is related to the familiar residual vector η by

$$\eta = V^Tp \tag{5}$$

where $\eta = m - H\hat{x}$ and $\hat{x} = (H^TH)^{-1}H^Tm$, the least-squares estimate of x. Because of the property (2) of VV^T, the residual vector and parity vector have identical norms, i.e.,

$$\eta^T\eta = p^Tp. \tag{6}$$

B. Implementation of the FDI Algorithm

The FDI procedure presented in this paper is applicable to scalar measurements only, i.e., the dimension of the measured variable x in (1) is equal to 1. (A general FDI procedure for nonscalar measurements such as the velocity or acceleration of an object in space is discussed in [8].) For l measurements of a scalar process variable, an FDI decision can be made using relative angular orientation and magnitude of the $(l - 1)$ dimensional parity vector with respect to various subspaces spanned by one or more failure directions, i.e., columns of the projection matrix V [16]. An alternative approach is to make a decision from the magnitudes of projections of the parity vector on all $l(l - 1)/2$ different one-dimensional subspaces that are, in the parity space, orthogonal complements of the subspaces spanned by $(l - 2)$ out of l failure directions. The norms of these projections are identically equal to the norms of one-dimensional parity vectors that are directly generated from the pair of measurements corresponding to the two failure directions which are not included in the aforesaid set of $(l - 2)$ failure directions [8].

For scalar measurements, the measurement matrix in (1) can be chosen as $H = [1 \quad 1 \quad \cdots \quad 1]^T$ without loss of generality. For a pair of scalar measurements m_i and m_j, the magnitude of the one-dimensional parity vector in (3) is directly proportional to $(m_i - m_j)$, which can be compared with the sum $(b_i + b_j)$ of the respective error bounds for a consistency check. Any two scalar measurements m_i and m_j at the sampling instant k are defined to be consistent if

$$|m_i(k) - m_j(k)| < (b_i(k) + b_j(k)). \tag{7}$$

Thus the relative consistencies of l scalar measurements can be evaluated by concurrent checking of $l(l - 1)/2$ pairs of measurements.

In this context, the inconsistency index of a measurement m_i is defined at a given sample time as

$$I_i = \sum_{j=1}^{k} f[|m_i - m_j| > (b_i + b_j)], \qquad i = 1, 2, \cdots, k \tag{8}$$

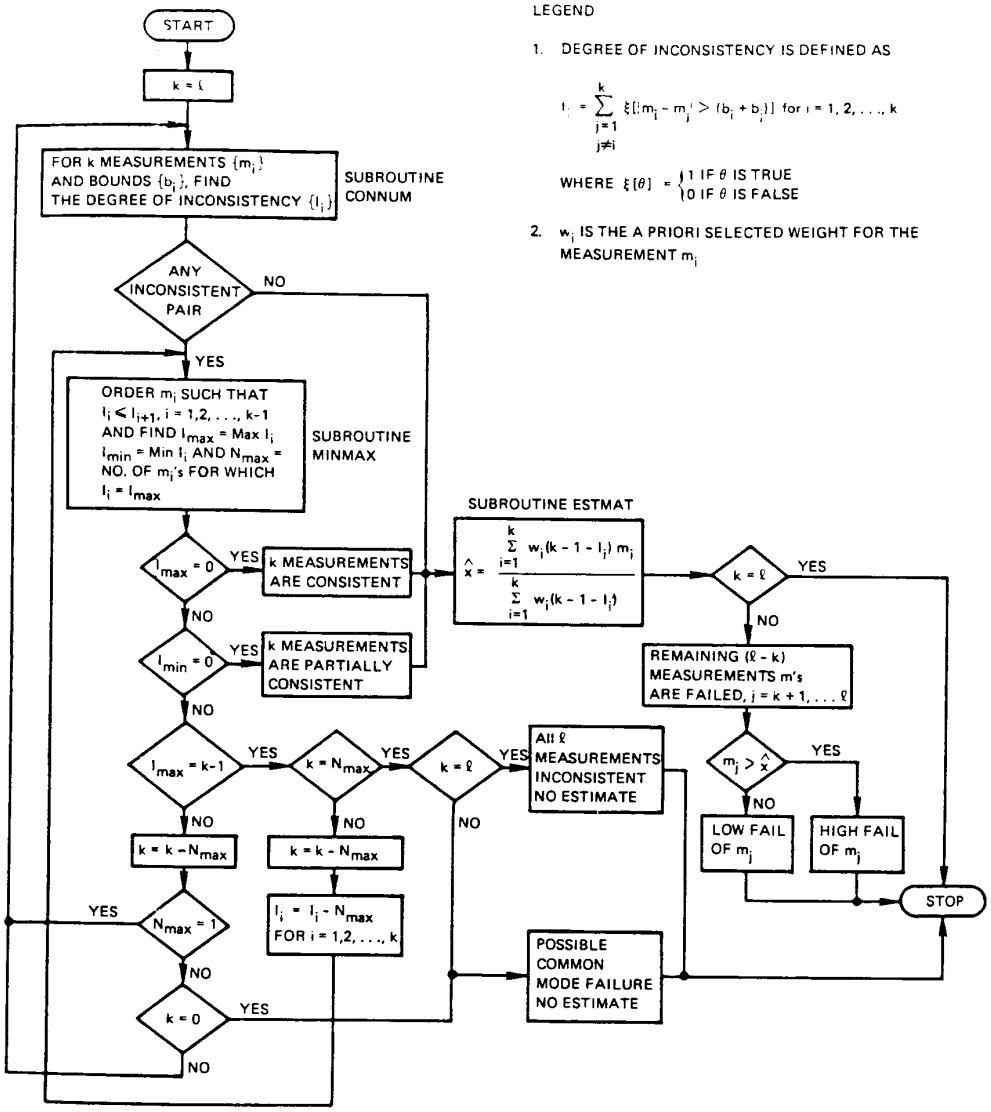

Fig. 1. Flowchart for redundancy management scheme.

where the indicator function f is defined as

$$f[*] = \begin{cases} 0, & \text{if } * \text{ is false} \\ 1, & \text{if } * \text{ is true.} \end{cases} \tag{9}$$

For each measurement m_i, the degree of inconsistency I_i provides k distinct levels of characterization ranging from 0 to $(k-1)$. If $I_i > I_j$, then m_i is more likely to be faulty than m_j at the given time sample. A flowchart of the redundancy management algorithm is given in Fig. 1. The consistency or inconsistency of each measurement is determined at each time sample on the basis of the inconsistency indices. This is explained by examples in the Appendix.

The FDI technique presented in this paper is a sequential decision-making procedure that systematically seeks out the largest consistent subset for use in estimation of the measure variable. The remaining measurements are tested to determine if they should be isolated as faulty. This is accomplished by assigning each sensor a counter. The allowed values of this counter are the nonnegative integers. Given a measurement m_i, the counter for its associated sensor will be incremented by one unit if m_i is inconsistent at a particular sample time. Similarly, the counter is decremented by one unit if m_i is consistent and the value of the counter is positive. (*Note:* The value of the counter is never allowed to become negative.) If a counter reaches a preset positive integer N, then the respective sensor is isolated as faulty. The choice of N is dictated by the allowable probability of false alarms. Typical values of N are 2 or 3.

If j sensors have already been isolated as faulty and if, out of the remaining $(k-j)$ measurements, n measurements ($n < (k-j)$) are found to be inconsistent at a given sample time, then the estimate $\hat{x}$ of the measured variable is obtained by a

weighted average of the remaining $(k - j - n)$ measurements at that sample time:

$$x = \left(\sum_{i=1}^{k-j-n} m_i w_i (k-j-1-I_i) \right) \Big/ \left(\sum_{i=1}^{k-j-n} w_i (k-j-1-I_i) \right) \tag{10}$$

where w_i is the *a priori* selected weight for the individual measurement m_i and $(k - j - 1 - I_i)$ is the adaptive weight of m_i at the given sample time. Since sensors with wider tolerance are relatively less accurate, it is reasonable to take w_i as proportional to a negative power of the error bound b_i. The procedure for obtaining estimates is illustrated by examples in the Appendix.

An alternative means for computation of the estimate is the midvalue selection [15] from the set of consistent measurements. Midvalue selection is simple to implement and does not involve arithmetic multiplicative operations such as taking the weighted average. However, it is more prone to inaccuracy due to the roundoff errors on an 8-bit microprocessor. The software for the implemented FDI procedure has both options of the weighted average and the midvalue selection.

III. Description of Electronic Hardware

The approach chosen for implementing the redundancy management scheme was to base the system on a factory-designed and -assembled microcomputer board and to construct the sensor data acquisition interface from commercially available IC chips. A block diagram of the system hardware that shows the details of the data acquisition subsystem and its interface with the microcomputer subsystem is given in Fig. 2. The details of the microcomputer system are given in [17] and, therefore, are not repeated here.

A. Microcomputer Subsystem

The first step in the design process was to determine the number of bits for the data bus, arithmetic operations, and analog-to-digital conversion. It is industry practice to take the measurements with a precision of one in 1000. This requires the use of a 10-bit analog-to-digital converter (ADC) and either simple software on a 16-bit machine or rather complex operations on an 8-bit machine.

It was determined that an inexpensive method for implementing this system would be to use an 8-bit microcomputer interfaced with an 8-bit ADC. As a result, the resolution would be one in 256 instead of one in 1000, i.e., the third digit of the display would not be accurate. Given that each measurement would not be considered individually but would be processed together with other redundant measurements of the same process variable via (10), this lack of precision would not significantly influence the estimate of the measured variable. Therefore it was decided to pursue this approach, which, if successful, would yield a simple reliable inexpensive means of implementing the sensor redundancy management procedure using commercially available microcomputers and IC chips.

The Motorola MEK6809D4/MEK6809KPD microcomputer system [17] based on the MC6809 8-bit microprocessor was chosen primarily because it offered some of the capabilities of 16-bit processing, a multiplication instruction, and advanced addressing techniques like position-independent code. The microcomputer system incorporates 4 kbyte of static RAM, eight 24-pin ROM sockets (allowing for the installation of a wide variety of ROMs, PROMs, and EPROMs), an interface, and an MEK6809KPD keyboard/display unit including a 25-key keypad and eight seven-segment displays. Memory-mapped I/O devices, controlled by the microcomputer, form part of the sensor data acquisition subsystem. These devices consist of

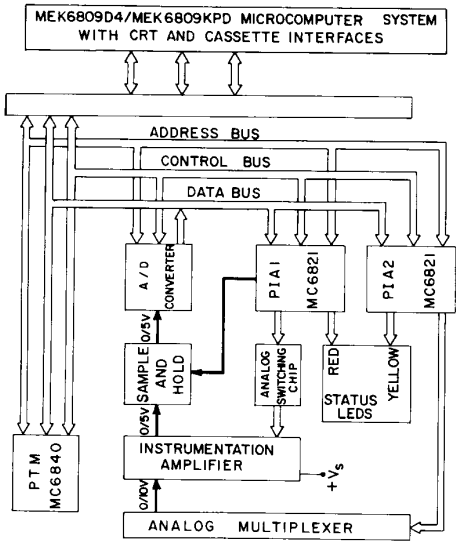

Fig. 2. Electronic hardware schematic.

1) eight addresses for the programmable timer module (PTM),
2) four addresses for the peripheral interface adapter (PIA1),
3) four addresses for the peripheral interface adapter (PIA2), and
4) one address for the (ADC).

B. Sensor Data Acquisition Subsystem

The data acquisition subsystem is designed to receive a maximum of eight sensor signals. The sampling period for data acquisition is selectable between 0.2 and 2.0 s. This range was chosen for compatibility with conventional process control systems. The sampling period is adjusted via an MC6840 programmable timer module that functions as the real-time clock.

The analog sensor signals (that are inputs to the data acquisition system) are often contaminated with plant noise (induced by motors, switches, and stray magnetic/electric fields) and common-mode voltages that result from the circulation of stray currents through the wires connecting sensors to the data acquisition center. Should problems of this type occur, the accuracy of the measurements will be degraded. Therefore, to avoid difficulties of this type and

thereby be certain of achieving the desired accuracy, a data acquisition system with balanced inputs was chosen. An analog multiplexer (MUX) connects sequentially, under computer control, each balanced input to a balanced output which is fed to an instrumentation amplifier (IA). The IA converts the input signal from balanced to unbalanced (referred to the system ground), provides common-mode rejection and a high input impedance, and amplifies the input by a factor of 4, 2, 1, or 0.5 to yield a stable output. This is shown in Table I.

TABLE I

Range (V)	Amplifier Gain	Input Resolution (mV)
0.00–1.24	4.0	4.88
1.25–2.49	2.0	9.77
2.50–4.99	1.0	19.53
5.00–10.00	0.5	39.00

The gain-changing feature is implemented by switching the appropriate resistors via an IC chip that contains four analog switches. The switches are controlled by the microcomputer through the peripheral interface adapter (PIA1) as shown in Fig. 2.

The output of the IA becomes the input to the sample and hold (S&H), which in turn implements the zero-order-hold algorithm. The S&H samples its input for an extremely short time and then holds the sampled data, permitting the switching of the MUX to the next channel in the sequence. While the switch is being made from one channel to the next, the current output of the S&H is digitized by the ADC that is connected to the microcomputer data bus.

The data acquisition process is initiated by a signal from the real-time clock. Channels 1–k are acquired, processed, and stored within a period of 2 ms. The software of the redundancy management procedure then isolates the faulty sensors and calculates the estimate of the measured variable from the valid sensor data. The display routine shows the results at the terminal while the microcomputer waits for the start of the next cycle. Fig. 2 shows a block labeled as "status LEDs." This block is a group of light-emitting diodes (LEDs) that indicate if and how the sensors have failed. There are eight red LEDs to indicate high failures of individual sensors and eight yellow LEDs to indicate low failures. These LEDs are driven by the microcomputer through and by a special interface to provide the required power.

In a distributed processing environment, a mainframe or a minicomputer is expected to gather information from a number of such microprocessor-based devices. The design of the prototype described here allows for interfacing with other computers. The easiest way is to use the RS-232C port. A complete description of the hardware design, the wiring of each block, and the procedures for adjustment and calibration of the prototype system is given in [18].

IV. Description of System Software

The software for the prototype system is divided into five major blocks.

1) System Management/Loader Routine: This routine organizes the operation of the microcomputer system and provides the application software with all the information and interfacing necessary for use of the hardware resources.

2) Initialization Routine: This routine is designed to establish the necessary initial conditions to set up the data acquisition process.

3) Data Acquisition Routine: This routine acquires the raw sensor data and preprocesses them to meet the requirements of the redundancy management algorithm. This routine has a built-in test capability for performing rationality checks. Each sensor reading is checked to be certain that it lies within a range of values that are reasonable for the quantity being measured. The permitted ranges are preset. If a sensor's output does not fall within the "rationale" range, it is automatically deleted from the set of valid sensors.

4) Redundancy Management Routine: This routine determines the consistency or inconsistency of all unfailed redundant measurements, provides an estimation of the measured variable from the consistent measurements, and isolates the faulty sensors via sequential testing.

5) Display Routine: This routine drives a CRT terminal via the RS-232C port and displays the estimate and sensor status information.

These routines are written in the Macro Assembly language using position-independent addressing code. A complete description of the software is given in [18]. This addressing approach allows the program to be positioned arbitrarily in the memory and facilitates software portability. The software is loaded from a tape cassette into the RAM of the prototype system described here. Regarding a commercial design, the software (without any major modifications) could be stored in the ROM area or automatically downloaded from a disk into the RAM when the operating system receives a request during startup.

The system startup and the normal operational cycle are schematically described in Fig. 3. During startup, the first routines to be run are the loader and the initializer, respectively. Next, the system enters the cyclic operational mode that repeats itself at every sampling period. These operations are briefly described below.

1) Upon initialization, the fast-interrupt request (FIRQ) mask is set and the microprocessor is given the synchronization instruction (SYNC). Upon receiving SYNC, the CPU halts until a peripheral device requests an interrupt. (Note: If the interrupt is disabled, the CPU continues with the next instruction in the program without any delay.) FIRQ is the only interrupt that can take the CPU out of the synchronization state, thereby causing it to halt. FIRQ is generated by the real-time clock at the beginning of each sampling interval.

2) The FIRQ is unmasked such that, should the process take longer than the sampling time, the FIRQ service routine (FIRQSVR) will be executed. The FIRQSVR is triggered to generate an audible warning signal whenever the sampling period is exceeded.

3) Next, the data acquisition routine is activated. It coordinates the hardware to acquire the sensor data and calls for the equalizer routine, which converts all sensor data to the

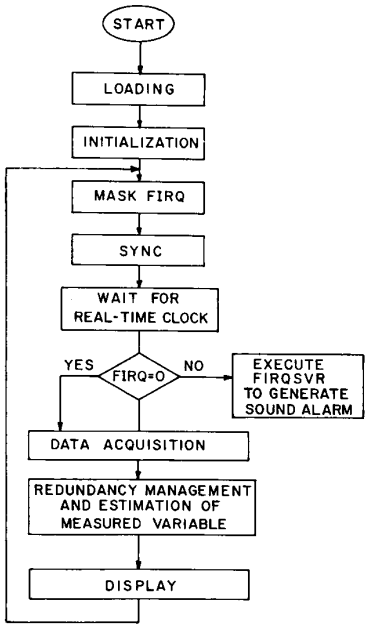

Fig. 3. System operation flowchart.

same scale, thereby allowing for meaningful comparisons to detect inconsistencies.

4) The redundancy management routine is now utilized. It identifies the failed sensor(s) and the kind of failure (high or low), discards both the inconsistent measurements and the invalid measurements from already failed sensors, produces an estimate of the measured variable, and generates the information for both the CRT display and the status LEDs.

5) Finally, the display routine presents the results on the CRT.

V. Experimental Results

The prototype device was tested on-line for detection and isolation of sensor failures at the MIT Nuclear Research Reactor (MITR-II). A description of the system configuration and instrumentation of this 5-MW fission reactor is given in [19]. The nuclear instrumentation used for the research reported in this paper consists of three neutron flux sensors and a gamma-ray sensor that correlates neutron power with the radioactivity (N-16) of the primary coolant. Four independent measurements of primary coolant flow are obtained from pressure differences across orifices. Primary coolant temperatures are measured as follows: two sensors for the hot leg, two sensors for the cold leg, and one sensor for the temperature difference between the legs. The noise and statistical characteristics of the MITR-II's flow, temperature, and neutron flux instrumentation are similar to those of commercial reactors.

The sensor readings (in volts) were represented with three digits. Of these, the most significant two digits were accurate, and the least significant digit was somewhat noisy. This behavior was expected because the prototype was built on a breadboard which precluded use of a single-point ground connection with separated digital and analog ground return paths. The estimate of the measured variable was generated using both the weighted average and midvalue selection techniques. The estimate obtained by the averaging approach was, as expected, less noisy. The fault detection and isolation capabilities of the prototype device were tested for both natural and induced sensor failures. No false alarms were reported during continuous operation of the MIT reactor over a period of one month. (Note: The MIT reactor operates 24 h/day for a total of approximately 100 h/week). The prototype was also tested by voiding water-filled shutters used to admit neutron beams to the experimental apparatus that surrounds the MIT reactor. Voiding certain of these shutters will cause the signal seen by one of the neutron sensors to change by 10–15 percent. The prototype correctly identified the failed sensor and deleted it from the estimate of the reactor's power.

The prototype device was tested for injected failures of sensors corresponding to errors in excess of the error bound. Typical cases are reported next.

Faulty Sensor Calibration: In a given sensor, a bias was introduced in excess of the error bound. The sensor was isolated as faulty and the estimate was generated from the remaining measurements.

Gradual Drift: Drift was introduced in a given sensor. The sensor was isolated after the accumulated drift exceeded the permissible error bound.

Failed Sensors: Some of the sensors were disconnected from the prototype device and the corresponding ports of the data acquisition system were short-circuited. The respective sensors were identified as faulty.

The prototype device was also tested during transient operation of the MIT reactor. For example, during a reactor shutdown process, the power estimate generated by the prototype device followed closely the accepted power level.

VI. Conclusion

A cost-effective means for implementing a fault detection and isolation procedure using redundant sensors has been presented. A prototype device has been built using an 8-bit MC6809 microprocessor and other commercially available electronic components. Given a set of redundant sensors for a process variable, the device identifies faulty sensor(s), as well as the type of failure(s) (high or low), and generates an estimate of the measured variable from the remaining unfailed sensors. The device has been successfully demonstrated for on-line fault detection on the 5-MW MIT research reactor.

Although this prototype device is based on a factory-designed and assembled microcomputer system, the associated software is not restricted to a specific hardware. It is portable to other microcomputer systems. Hence, while the hardware may require some redesign for use on other facilities, such would not be the case with the software.

This prototype device is designed to function in a multiprocessing environment where a number of such devices could simultaneously serve different groups of redundant sensors and feed the results to a main computer. In complex and strategic processes like nuclear reactors, hazardous chemical plants, and advanced aircraft, such a device is potentially

useful for 1) reducing the number of display devices in the control room, 2) relieving the control room operator(s) from the tedious tasks of assimilation and analysis of the redundant sensor data, and 3) enhancing the processing capabilities of the main computer.

Appendix
Examples for Illustration of the FDI Procedure

The following examples illustrate how the FDI procedure of the flowchart in Fig. 1 is used.

Example 1

Let three measurements m_1, m_2, and m_3 be available, i.e., $l = 3$.

1) If all measurements are consistent, i.e., $I_i = 0$, $i = 1, 2, 3$, then the estimate is $\hat{x} = (w_1m_1 + w_2m_2 + w_3m_3)/(w_1 + w_2 + w_3)$ from the consistent measurements m_1, m_2, and m_3.

2) Next, consider the case when m_1 is consistent with m_2 and m_3 but m_2 and m_3 are mutually inconsistent, i.e., $I_1 = 0$, $I_2 = 1$, and $I_3 = 1$. Since $I_{max} = \max_i I_i > 0$ and $I_{min} = \min_i I_i = 0$, the estimate is $\hat{x} = (2w_1m_1 + w_2m_2 + w_3m_3)/(2w_1 + w_2 + w_3)$, from the partially consistent measurements m_1, m_2, and m_3, and no measurement is isolated as inconsistent.

3) Suppose that the measurement m_1 fails, i.e., $I_1 = 2$, $I_2 = 1$, $I_3 = 1$. In this case, $I_{min} > 0$ and N_{max} (=number of measurements for which $I_i = I_{max}$) is one. On the next pass, the two measurements m_2 and m_3 are found to be consistent. The resulting estimate is $\hat{x} = (w_2m_2 + w_3m_3)/(w_2 + w_3)$, from the consistent measurements m_2 and m_3, and m_1 isolated as a high failure if $m_1 > \hat{x}$, or as a low failure if $m_1 < \hat{x}$.

4) If m_1, m_2, and m_3 are mutually inconsistent, i.e., $I_1 = I_2 = I_3 = 2$, then no estimate is possible because all measurements are inconsistent. In this case, a fault has been detected, but it cannot be isolated unless additional information is available.

Example 2

Let four measurements, m_1, m_2, m_3, and m_4, be available, i.e., $l = 4$.

1) If two measurements, e.g., m_2 and m_4, fail simultaneously and identically, then there are two consistent pairs, (m_1, m_3) and (m_2, m_4), that are mutually inconsistent, i.e., $I_1 = I_2 = I_3 = I_4 = 2$. Since $I_{min} > 0$ and $N_{max} = 4$, the outcome is that no estimate can be obtained because there is a possible common-mode failure. A fault is detected, and it can be isolated only if additional information is available to determine the failure of a specific pair. Usually, such information can be generated from other sources.

2) Next, suppose that m_1 and m_3 are consistent, but m_1 and m_4 fail simultaneously and nonidentically, i.e., $I_1 = 3$, $I_2 = I_3 = 2$, $I_4 = 3$. Since $I_{max} = 3$ and $N_{max} < 4$, only two measurements, m_2 and m_3, are considered in the next pass, with their degree of inconsistencies decremented by N_{max}, i.e., $I_2 = I_3 = 0$. The resulting estimate $\hat{x} = (w_2m_2 + w_3m_3)/(w_2 + w_3)$ from the consistent measurements m_2 and m_3, and m_1 and m_4 are isolated as inconsistent. Each faulty measurement is classified as high or low, depending on whether it is larger or smaller than $\hat{x}$.

Example 3

Let five measurements, m_1, m_2, m_3, m_4, and m_5, be available, i.e., $l = 5$.

1) Suppose two measurements, say m_2 and m_5, fail simultaneously and identically, i.e., $I_1 = 2$, $I_2 = 3$, $I_3 = 2$, $I_4 = 2$, $I_5 = 3$. Since $I_{min} > 0$, $I_{max} < 4$, and $N_{max} > 1$, the three measurements m_1, m_3, and m_4 that have $I_i < I_{max}$ are tested on the next pass. The resulting estimate is $\hat{x} = (w_1m_1 + w_3m_3 + w_4m_4)/(w_1 + w_3 + w_5)$, from the consistent measurements m_1, m_3, and m_4; and the remaining measurements m_2 and m_5 are isolated as faulty.

2) Next, suppose that m_2, m_3, and m_4 fail simultaneously but nonidentically, i.e., $I_1 = 3$, $I_2 = 4$, $I_3 = 4$, $I_4 = 4$, $I_5 = 3$. *Since* $I_{min} > I_{max} = 4$, and $N_{max} < 5$, the degree of inconsistency for the two measurements m_1 and m_5 are decremented by $N_{max} = 3$, i.e., $I_1 = 0$, and $I_5 = 0$. The resulting estimate is $\hat{x} = (w_1m_1 + w_5m_5)/(w_1 + w_5)$, from consistent measurements m_1 and m_5. The remaining measurements m_2, m_3, and m_4 are isolated as faulty.

3) Suppose that two measurements, m_2 and m_4, fail identically, and m_5 fails in a different way at the same instant, i.e., $I_1 = 3$, $I_2 = 3$, $I_3 = 3$, $I_4 = 3$, and $I_5 = 4$. Initially, m_5 is isolated as faulty, and since $N_{max} = 1$, the remaining measurements are reset to $I_1 = I_2 = I_3 = I_4 = 2$ on the next pass. The situation then reduces to case 1) in Example 2, and no estimate can be obtained unless additional information is available.

Acknowledgment

The authors acknowledge the contributions of Prof. David D. Lanning and Mr. Paul T. Menadier in this research and wish to thank the staff of the MIT Research Reactor for their assistance.

References

[1] C. H. Meijer *et al.*, "On-line power plant signal validation technique utilizing parity-space representation and analytic redundancy," Electric Power Res. Inst., Palo Alto, CA, Rep. NP-2110, Nov. 1981.

[2] C. H. Meijer and B. J. Frogner, "On-line power plant alarm and disturbance analysis system," Electric Power Res. Inst., Palo Alto, CA, Rep. NP-1379, Apr. 1980.

[3] S. Osder, "Chronological overview of past avionic flight control system reliability in military and commercial operations," in *Agradograph 224*, P. R. Kurzhals, Ed. AGRAD-AG-224, 1977.

[4] G. M. Stanley, "On-line data reconciliation for process control," presented at the American Institute of Chemical Engineers, Winter Annual Meeting, Nov. 1982, paper 11b.

[5] A. S. Willsky, "A survey of design methods for failure detection in dynamic systems," *Automatica,* vol. 12, pp. 601–611, Nov. 1976.

[6] T. T. Chien and M. B. Adams, "A sequential failure detection technique and its application," *IEEE Trans. Automat. Contr.,* vol. AC-21, pp. 750–757, Oct. 1976.

[7] A. Ray, M. Desai, and J. Deyst, "On-line fault diagnosis in a nuclear reactor by sequential testing," *IEEE Trans. Nucl. Sci.,* vol. NS-30, pp. 1850–1855, June 1983.

[8] A. Ray and M. Desai, "A redundancy management procedure for fault detection and isolation," *J. Dynamic Syst. Meas. Contr.,* pp. 248–254, Sept. 1986.

[9] J. L. Tylee, "A generalized likelihood ratio approach to detecting and

identifying failures in pressurizer instrumentation," *Nucl. Technol.*, vol. 56, pp. 484–492, Mar. 1982.
[10] R. M. Clark and B. Campbell, "Instrument fault detection in a pressurized water reactor pressurizer," *Nucl. Technol.*, vol. 56, pp. 23–32, Jan. 1982.
[11] A. Ray, "A microcomputer-based fault-tolerant control system for industrial applications," *IEEE Trans. Ind. Appl.*, vol. IA-19, pp. 1276–1283, Sept. 1983.
[12] J. L. Fisher *et al.*, "Signal validation for a BWR suppression pool," *Trans. Amer. Nucl. Soc.*, vol. 44, pp. 64–66, Aug. 1983.
[13] O. L. Deutsch *et al.*, "Development and testing of a real-time measurement validation program for sodium flowrate in the EBR-II," Charles Stark Draper Lab., Cambridge, MA, Rep. CSDL-R-1592, Oct. 1982.
[14] M. Desai, J. C. Deckert, and J. Deyst, "Dual sensor failure identification using analytic redundancy," *AIAA J. Guidance Contr.*, vol. 2, pp. 213–220, May/June 1979.
[15] J. E. Potter and M. C. Suman, "Thresholdless redundancy management with arrays of skewed instruments," in *Agradograph 224*, P. R. Kurzhals, Ed. AGRAD-AG-224, 1977.
[16] K. C. Daly, E. Gai, and J. V. Harrison, "Generalized likelihood test for FDI in redundant sensor configurations," *J. Guidance Contr.*, vol. 2, pp. 9–17, Jan.–Feb. 1979.
[17] *Motorola Memory Systems, MEK6809D4 Microcomputer Evaluation Board and MEK68KPD Keypad and Display Unit Users' Manual,* Motorola Inc., Austin, TX.
[18] H. P. Polenta, "Implementation and testing of a microcomputer based fault detection system," thesis, Dep. Nucl. Eng. Mass. Inst. Technol., Cambridge, Jan. 1983.
[19] *Reactor Systems Manual,* Mass. Inst. Technol., Cambridge, Rep. MITNRL-004, 1980.

Hector P. Polenta received the B.S. degree in telecommunications from the National University of La Plata, Argentina, and the M.S./N.E. degree from the Massachusetts Institute of Technology, Cambridge.

He is currently serving as a Naval Officer with the Argentine Navy. His research interests include the control of nuclear reactors and advanced electronics design.

Asok Ray (SM'83) for a photograph and biography, please see page 904 of this TRANSACTIONS.

John A. Bernard (M'84–SM'85) received the B.S. degree from Yale University, New Haven, CT, and the M.S./N.E. and Ph.D. degrees from the Massachusetts Institute of Technology, Cambridge, in 1970, 1979, and 1984, respectively.

His professional experience includes service as a Naval Officer on the nuclear-powered aircraft carrier Enterprise (1970–1974), a Shift Supervisor of the 5-MW MIT Research Reactor (1974–1979), and Superintendent of the MIT Research Reactor (1979–present). He currently holds the position of Principal Research Engineer at MIT and is responsible for the development of techniques for the closed-loop digital operation of nuclear reactors.

Dr. Bernard is a member of the American Nuclear Society and the author of more than 50 technical papers and reports.