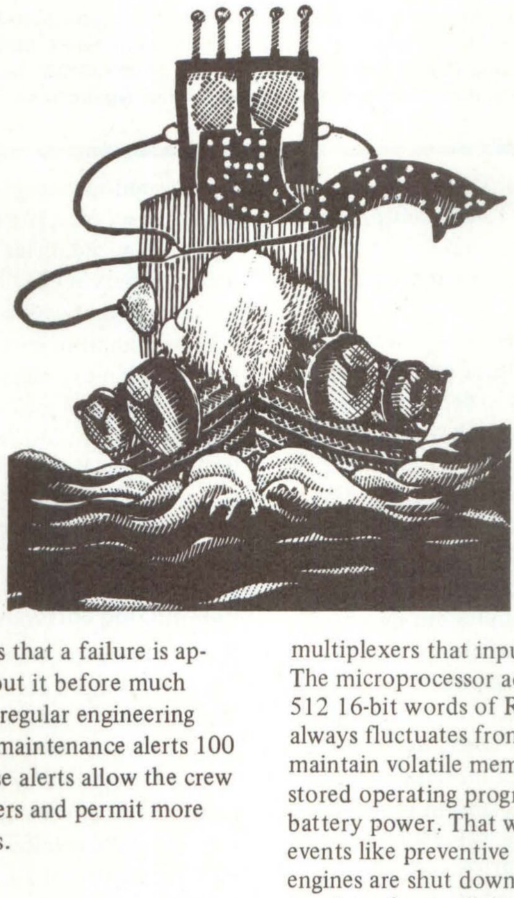
MICRO-BRAIN ON THE BOUNDING MAIN

How designers configured a microprocessor based data-acquisition and diagnostic system that detects engine malfunctions and oversees preventive maintenance aboard seagoing tugboats

by **Steve Tsolis** and **Tony Mathews**



Until recently, the marine industry had no analytical method of monitoring the performance of the eight-to-twenty-cylinder diesel engines that power modern vessels, and ship operators couldn't detect many minor malfunctions in those power plants until the malfunctions caused major breakdowns. And because the plants develop up to 4000 hp, those breakdowns often caused near-catastrophic failures.

We grew aware of these problems when a tugboat operator asked us to determine the cause of several massive crankshaft failures. As part of our investigation, we developed the Seaborne Integrated Diagnostic System (SIDS), a microprocessor controlled data-acquisition and diagnostic system that is now installed on several ocean- and river-going craft.

Using SIDS, a tugboat's crew knows that a failure is approaching in time to do something about it before much damage occurs. The system prints out regular engineering reports and also generates preventive-maintenance alerts 100 hrs or so before work is required. These alerts allow the crew to perform minor repairs during layovers and permit more efficient scheduling of major overhauls.

designing for computer novices

SIDS' major design constraints included size limitations, a hostile environment and operating personnel totally unversed in either electronics or computer operation. Primarily engines and fuel tanks surrounded by small hulls, tugboats are cramped. With this space premium in mind, we designed SIDS to fit in a 17"W x 10"H x 23"D cabinet. On one ship, we had to install the unit in the most readily available space – the ship's head.

In the monitor's severe operating environment, sensors fixed to the engines experience high temperatures as well as large-amplitude, low-frequency vibration. In addition, crews wash down the engines frequently with a caustic solution. Because some craft are under way for long periods of time, performing service en route is a difficult if not impossible task. Because ships' personnel aren't qualified to adjust or maintain μP-based equipment, we made SIDS completely automatic. If it fails, it fails gracefully and generates no false warnings. Achieving this capability would normally dictate using an expensive minicomputer system, but cost and size factors precluded this option.

Instead, we chose to use the National Semiconductor 16-bit IMP-16 microprocessor (Fig 1), which in our design accepts a 128-input section of 16 8-channel-type multiplexers that input a 12-bit integrating A/D converter. The microprocessor accesses 8K 16-bit words of PROM and 512 16-bit words of RAM. Because workboat power almost always fluctuates from the levels a computer requires to maintain volatile memory, we configured the system's RAM-stored operating programs and its real-time clock to utilize battery power. That way, the system can save time-related events like preventive maintenance schedules while the ship's engines are shut down. The microprocessor's output section consists of a parallel-to-serial converter and a printer driver; two of the IMP-16's user flags provide aural warning and enable the printer.

SIDS requires a 12-bit data precision to achieve 4-decimal-digit accuracy. We also require the monitor to scan, correct

Steve Tsolis *and* **Tony Mathews** *are president and vice president, respectively, of Advanced Electronic Devices, an Old Saybrook, CT, firm that specializes in microprocessor-system consulting and design.*

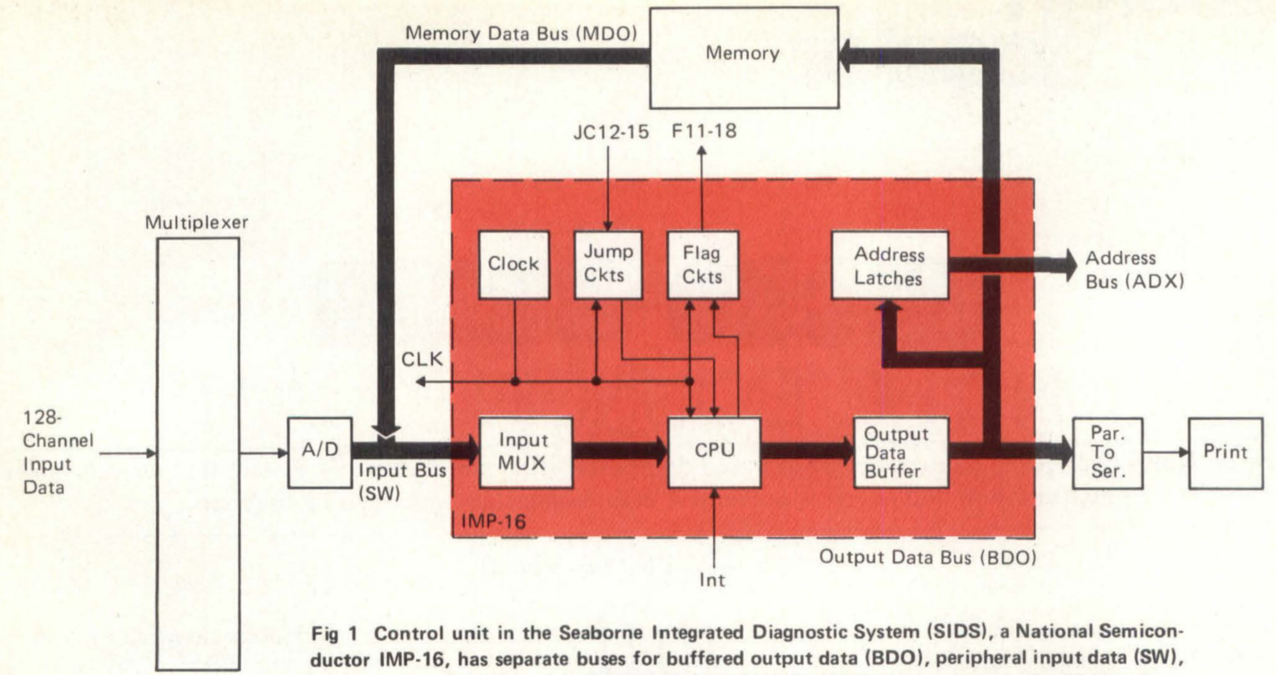


Fig 1 Control unit in the Seaborne Integrated Diagnostic System (SIDS), a National Semiconductor IMP-16, has separate buses for buffered output data (BDO), peripheral input data (SW), memory data (MDO) and buffered addresses (ADX). SIDS' designers chose the 16-bit microprocessor to meet arithmetic and bit-manipulation requirements.

and process one data point every 10 ms. The processing operation involves performing two or three 12-bit multiplications (or divisions) while sifting through a multi-level fault tree, so execution time formed the key factor in our selection of a 16-bit microprocessor.

But additional factors also influenced our choice. Sixteen-bit machines require fewer total bits of memory than 8-bit machines, and programs are easier to debug in 16-bit machines. We also liked the convenience of having a little more than sufficient arithmetic accuracy in one computer word; 8-bit processors require double-precision arithmetic to handle data from a 12-bit A/D converter.

After considering all these factors, we determined that while the IMP-16 could sift through the fault tree in less than 5 ms and 8-bit devices required about 8 ms, the combination of arithmetic functions and tree sifting placed 8-bit processors beyond our 10-ms target. Thus, we had to choose a 16-bit device.

Our choice narrowed to a decision between the five-chip IMP-16 and the single-chip Pace – also from National Semiconductor – the only 16-bit devices available in production quantities at the time of our design. The IMP-16 has separate 16-bit buses for buffered output data (BDO), peripheral input data (SW), memory data (MDO) and buffered addresses (ADX). Pace uses a single I/O bus architecture. In the IMP-16, memory locations and peripheral devices can have the same addresses.

The IMP-16 also has 16 user-available flags, one general interrupt condition, one vector interrupt, and four user jump conditions – features important in reducing component count in an interrupt-driven system. Both IMP-16 and Pace use the same basic set of instructions, but the IMP-16 has additional CROM (control read-only memory) that provides 17 instructions, including single-word arithmetic commands (multiply, divide, double precision add and subtract) as well as set bit, clear bit and test bit instructions. The IMP-16 multiply and divide commands produce 32-bit products and dividends in less than 200 μs – a factor that simplifies software in that device.

The amount of data that SIDS must process makes bit manipulation expedient; the monitor generates many status tables in which it must modify single bits to indicate status changes. Single-word IMP-16 commands set and clear any bit in an accumulator, while the SKBIT instruction increments the microprocessor's program counter if the specified bit is a logical "one". These bit-manipulation capabilities also led us to choose the IMP-16.

maximizing software use

A typical seagoing tugboat has a 4-engine plant consisting of two turbocharged diesel main engines and two diesel-driven alternators. In a SIDS-equipped ship, sensors measure the temperature of the induction air, the engine coolant and the lubrication system for each engine, as well as exhaust gas temperatures at each cylinder. Other sensors measure pressures in intake manifolds, lubrication systems and fuel lines. RPM tach generators and voltage sensors monitor engine speed and alternator output; cables from all sensors to the controller are generally long because of confined engine-room space.

The 100 mV signals from the sensors go to 8-input multiplexers, each equipped with a single amplifier on its output for isolation (Fig 2). Each channel is clocked at a 10-ms rate, and multiplexer output, through switches, feeds the single 12-bit A/D converter. Twelve-bit parallel data from the converter goes to the IMP-16 SW data bus. One of the IMP-16's user flags (F-12) controls start of conversion; end of conversion is signalled by an extra data bit inserted on the SW bus.

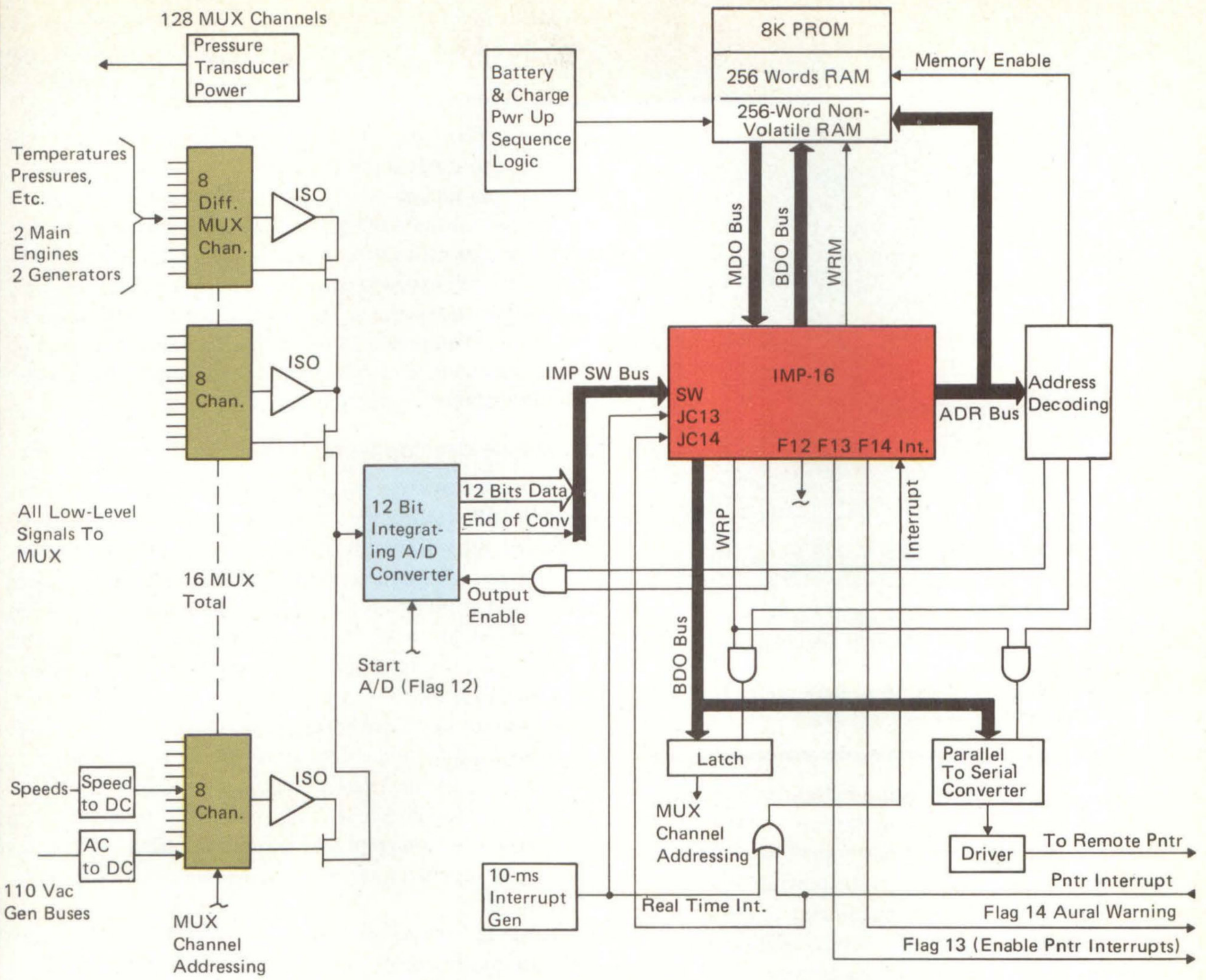


Fig 2 Temperature, pressure, engine-rotation and alternator-output data input SIDS through 16 8-input multiplexers, whose isolated outputs go to a 12-bit A/D converter and from there to the IMP-16's SW data bus. The microprocessor selects an input channel with a 16-bit word, decoded by latch circuits; it signals start of conversion with Flag 12 and end of conversion through Bit 13 on the SW data bus. Each conversion requires about 8 ms; a 10-ms real-time interrupt forces the system through Jump 13 and the interrupt pin.

Decoded address lines select the proper channel addresses.

We feel that system software should handle most tasks normally assigned to hardware. This arrangement reduces system complexity, lowers hardware cost and channels a large part of design cost into a one-time, non-recurring software-development effort. It also leads to some interesting instrumentation concepts. For example, the monitor accepts thermocouple signals directly; no secondary-reference junction exists, because reference signals come from PROM-resident tables keyed to nominal thermocouple output. Similarly, signal conditioning or gain setting schemes don't serve individual pressure sensors; signals are standardized through software. The IMP-16 provides all signal conditioning; it inserts scaling, gain and offsets prior to converting data to engineering units for calculation.

The software's processing of data input from the sensors involves multiplying by a conversion factor and adding a constant. Data stays temporarily in RAM for later required references.

diagnosing the diagnostics

During the monitor's software design phase, we realized that we needed more than seven bits to control the processing of input data, even though the system achieves multiplexer addressing with only seven bits. We gained this added capability without relying on look-up tables, which require large amounts of memory.

The monitor must establish whether incoming data should be tested for high limits, low limits or both; whether the data measures temperature, pressure, rotation or voltage; and whether a given failure requires crew warning. We achieved the required parameter selection for this process by using the nine additional available bits in each IMP-16 word (Fig 3).

The monitor can use sensor data in several ways to deduce a malfunction. But complex diagnostics are difficult to achieve without extensive fine tuning of the software for each engine – a process that produces long routines. Suppose, for example, that the exhaust gas temperature of one cylinder is constantly 20° lower than the temperature of all other cyl-

IMP-16 Bit	MUX Address	Bit Assignment
15	Bit 6 (Most Sig.)	0 = port, 1 = starboard
14		0 = data from engine, 1 = data from generator
13		1 = "left bank" required in error message
12		1 = "right bank" required in error message
11		1 = "stop engine" required
10		1 = failed upper limit 0 = failed lower limit
9		1 = "stop generator" required
8	Bit 5	0 = temperature, 1 = pressure
7	Bit 4	part of address
6	Bit 3	part of address
5	Bit 2	part of address
4	Bit 1	part of address
3	Bit 0	part of address
2		1 = don't output this MUX address
1		1 = perform upper limit test
0		1 = perform lower limit test

Fig 3 SIDS achieves multiplexer addressing with seven bits; it uses the remaining nine bits of a multiplexer address word to control the processing of input data. The monitor's software outputs each 16-bit MUX address from a table of addresses through which it cycles.

inders, and that no crankcase-pressure pulsations exist at a frequency equal to the number of rpm. Suppose too that the pressure in the intake manifold opposite the low exhaust gas temperature is constantly lower than the pressure in the other intake manifold. Most probably the clearance of one of the valves for the cylinder with low exhaust gas temperature is wrong. But for another engine of the same type, 10° or 30° could indicate the same condition.

To minimize the size of the diagnostic routines that allow for such variations, we configured SIDS' software so that input data is converted to proper units and then tested sequentially with other data (Fig 4). Such diagnostics give SIDS great power but don't occupy much memory; we had to write only 1000 lines of code for the executive program, the diagnostic routines and the tables.

To accommodate different customer requirements, we partitioned memory and wrote the program with a separate executive that we could commit to ROM after our final design review. We assigned constants subject to change, tables, data unique to specific engines, and tables and subroutines for optional equipment to other specific memory areas. Following checkout, we implemented these parameters in separate PROMs for production.

prototype development

We developed software and prototype hardware on National's IMP-16P prototyping system, equipped with the IMP-16 CPU card, 8K x 16 words of memory, and interfaces for a Teletype, a high-speed tape drive, a card reader, a high-speed printer and a PROM programming card. The system's memory size allows about 800 labels in the development program.

During prototype development, we used a Documation card reader for source program input; a Texas Instruments Silent 700 for keyboard access, cassette read and write and hardcopy output; and a Centronics printer for assemblies. We modified the firmware in National's Teletype interface card to increase the Silent 700's speed from 10 cps to 30 cps. And we wrote a program that allowed the Silent 700 to store and access support programs on magnetic tape cassettes at 1200 baud.

To speed software development, we used the IMP-16 resident assembler, editor, compiler and Debug programs. Because we owned a card reader, source editing did not pose

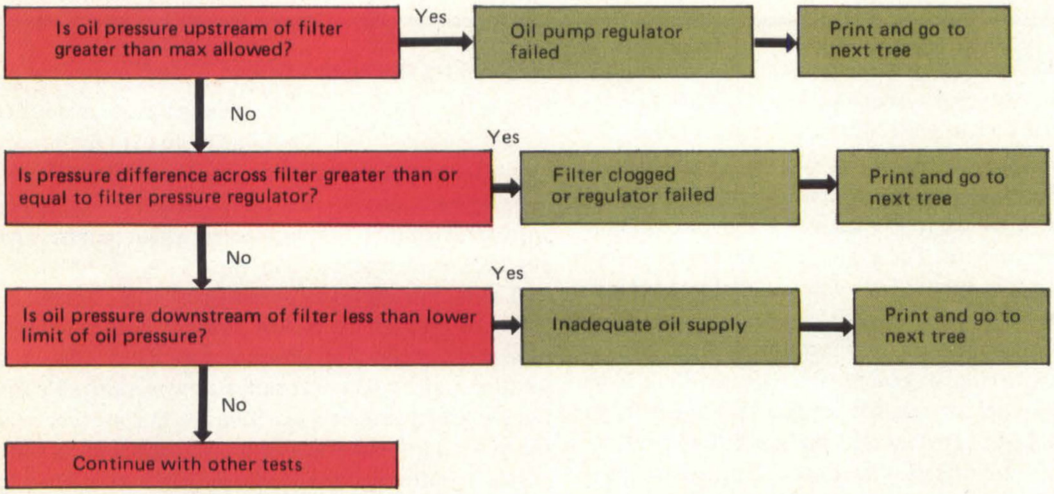


Fig 4 Sample fault tree illustrates how SIDS sequentially compares input data with PROM-resident limits. The monitor's executive program, diagnostic routines and tables are implemented in about 1000 lines of code.

the edit/loading problem it could have if we had used paper tape. We "terminal-built" many of the short programs – especially subroutines – on the Silent 700 using a "conversational assembler," to further reduce coding time. And we established systematic breakpoints to allow TTY printout of selected CPU and memory contents during debugging. A limited version of the Debug program served shipboard SIDS verification and checkout.

In its initial version – designed to determine the cause of crankshaft failure in one class of tugboats – SIDS was housed in the IMP-16 prototyper chassis. The assembly included the analog input section, sensor power supply, A/D converter, microprocessor CPU, memory, output interface and operator control panel. (The control panel in the prototyper is a completely independent peripheral; when installed, it allows complete access to CPU and memory, but when removed it does not affect system operation.)

Using this prototype SIDS, we ascertained that the shaft failure resulted from low (20 psi) oil pressure caused when the engine speed dropped significantly below idle as the propeller shaft was shifted into reverse. Since then, we have expanded SIDS to provide unattended shipboard diagnostics. To properly respond to these diagnostics, an engine must turn at constant speed for more than 15 min; otherwise, analysis is restricted to simple limit testing. Generators too must operate for at least 15 min before a detailed analysis.

When SIDS finds a failure, it prints a message in clear language to alert the crew to both the problem and to the required corrective action. It also prints advisory messages. For example, a worn injector produces a rich fuel/air mixture and generates lower-than-ordinary cylinder-head temperatures. While the condition doesn't cause damage immediately, the crew can replace the injector while under way to improve fuel consumption. The advisory message would read

```
PORT GENERATOR
EGT CYLINDER 16
LOW TEMPERATURE
960°F
```

SIDS also tracks PROM-resident preventive maintenance schedules with its real-time clock and prints those schedules to alert the skipper of impending yard time.

While the shipboard printer advises a crew of power-plant performance malfunctions and preventive-maintenance requirements, another function – trending – helps the ship's owners further increase the vessel's efficiency. Weekly, the crew tabulates readings from the printer and from fixed power-plant instruments and mails those tabulations to our headquarters, where they are coded and input to a computer program. The program compares the readings with historical data on the ship's equipment and analyzes trends that could lead to reduced engine performance, potential malfunctions not covered by the on-board system and conditions requiring correction during the ship's next overhaul period. ♦

COMING NEXT MONTH

★ Designing Consumer-System Peripherals

★ Microcomputers For Industry