FIND AND FIX Y2K EMBEDDED EQUIPMENT RISKS

© 1998 Dick Lefkon 212-539-3072
Asst. Prof. of Information Technology
New York University, NY, NY
profy2k@hotmail.com

EXECUTIVE SUMMARY

An understandable confusion appears to have led some IT professionals to believe that the Year 2000 Inventory/Assessment phase establishes estimates and parameters prior to startup of real work. The truth is opposite: A Year 2000 Inventory/Assessment ***IS*** the real work, getting an enterprise well beyond the one-third mark of project accomplishment, and providing complete information for a comprehensive project schedule to accomplish all remaining Year 2000 work.

Inventory/Assessment will result in minimizing later work on 85%-96% of your capital equipment. But it is necessary actually to look. A factory or processing center with "at least 95% workable" equipment, is akin to a NASA space vehicle with "at least 95% workable" equipment.

An early Y2K problem introduction by this author appeared as "Nuclear Disaster and the Millennium Trojan Horse," in The Proceedings from the 14th National Computer[Information Systems] Security Conference, October 1-4, 1991. More recently he has run Y2K Capital Equipment Inventory and fix teams in the United States Southwest, West, Midwest, Southeast and East.

While some companies and Federal agencies have already met their mandate to COMPLETE Inventory/Assessment (Y2K's "Phase 2"), others have entered 1999 without even BEGINNING it at most installations with capital equipment.

An isolated or hub-based enterprise may have one central routing point through which all goods flow. But for decentralized ones, every location can be unique and fairly independent. The processing equipment and fire

and other premises equipment may have been modified during service calls. Even if -- unlikely -- twelve consecutive processing centers test Y2K-OK, it remains necessary to Inventory/RiskAnalyze the thirteenth: Each critical operation needs skilled eyes and hands to enumerate its capital equipment, including those devices not documented in the corresponding file cabinet.

CLARIFYING AMBIGUOUS Y2K TERMINOLOGY

Year 2000 plans and schedules are frequently misunderstood by even the most intelligent and highly educated lay professionals. This stems in great part from the "Y2K" re-use of pre-existing words such as "Contingency," "Inventory" and "Risk Assessment."

A good IT manager is at a special disadvantage dealing with Y2K Capital Equipment work, exactly because she or he has previously run timely and succcessful IT efforts based on the effective prior application of almost completely opposite (old) meanings of basic (new) Y2K terms. For instance, any IT manager who has ever spent a few days on the telephone to develop a usable Impact Analysis for budgeting and planning, may be puzzled at the standard 15%-25% Y2K Project cost allocation to the Y2K Inventory/RiskAnalysis phase.

The non-Y2K professional's confusion -- and its resolution -- rest entirely on the contrast between taking a complete item census and performing an effective sample to obtain broad, often useful generalizations. As this writer has acknowledged in his 1975 college textbook, STATISTICS FOR MANAGEMENT AND THE HELPING SCIENCES (Addison Wesley, Reading, pp 50-51):

> "Studying a sample of your target population
> can [sometimes] be better than taking a census
> of the entire population:
>
> > "It is faster.
> > It is cheaper.
> > It uses less manpower.
> > It's less tiring."

He cautions, however, that sampling should only be used when "it is impossible -- or unwise -- to 'try everyone,' to take a census."

A good random sampling/analysis of your Capital Equipment will most likely produce results in line with those of other semi-automated operations:

85 % of devices will have no Y2K date problem.
15 % of devices will initially be suspect.
4 % of devices will need to be upgraded, be replaced, or have their function discarded.

An equivalent scoping exercise may have been performed during budgeting. However, knowing the "4 %" AVERAGE failure rate contributes absolutely nothing to survivability of an individual installation, any more than would awareness of a quarter-inch (1/4 ") bullet hole in a patient, in the absence of any knowledge of whether the bullet has hit the thigh, the heart, etc.

Consider that "4 %" again. If a processing center's security cameras display the wrong date beginning 1/1/00, a simple temporary remedy can be to affix a new handwritten date to the opposing wall each day. But if that location's essential operation depends on a particular model of sorting machine that will fail on 1/1/00 -- or 9/9/99, 2/29/00, 10/10/00, 1/1/01, 3/1/01, etc. -- only an on-site manual tagging and definitive part identification can dependably find and characterize the impending breakdown.

All decentralized organizations share these dual Y2K dilemmas:

- Even the best-maintained inventory file cabinets will be missing documents on potentially disaster-prone capital equipment, due to the passage of years and extinction of memory;

- Main Headquarters records of the many hundreds of outlying locations will be sketchier still, due to additional obfuscation contributed by distance and the judgment of individual chiefs.

The Y2K inventorying reality can plainly be seen:

- EVERY single physical location is unique and different.

- Two sites may differ by only a few critical automated devices -- yet just one of those (or its network connection) may require a Y2K upgrade to avoid derailing that site's work processing chain.

Therefore, if funds previously allocated to the Y2K Inventory/Analysis phase are reduced or partially withdrawn, the wise choice is to eliminate ALL Y2K outlays for the less critical of the Y2K-funded processing centers, and to perform a FULL MANUAL INVENTORY (and necessary follow-up) on the ones most critical to enterprise survivability. Using guesswork or stale purchasing lists in place of any physical inventory is going to invite risk. Riskier still would be to limit such an investigation to a device checklist sampled only from convenient sites elsewhere.

In all of this, calculating enterprise-wide averages cannot ensure survivability: A site with "at least 95% workable" equipment, is akin to a NASA space vehicle with "at least 95% workable" equipment.

IF Y2K ASSESSMENT DOESN'T SUMMARIZE, THEN JUST WHAT IS ITS IMPACT?

Every legitimate Y2K Capital Equipment Inventory/Assessment phase in every industry requires that every individual item and networked group be found, described correctly, assigned a criticality rating, and checked for survivability across the century rollover and certain other key dates. There can be no exceptions to this expenditure of effort -- although the use of tracking software and Y2K compliance databases can make verification process reliable and faster for multiple-instance equipment.

The "Assessment" part of Y2K Inventory/RiskAssessment refers to Individual item survivability and criticality, not a generalized average of any kind:

- If a weigh station won't function a year from this January, trucks may have to leave half-empty or risk breaking an axle.

- If PBX's fail, high-profit special orders can be neither received nor tracked.

- If a training VCR displays the wrong date, don't worry.

- If fire- or biohazard-detectors report false positives, automated operations can be wrongly curtailed or shut down entirely.

- If an "Assessment" produces forecasted averages but fails to pinpoint each individual potential malfunction, then rest assured your competitors will be poised to pick up the slack once your embarrassing Y2K failures become widely publicized.

The good news is: After a thorough inventory, if no capital equipment at a particular site is suspect, then good multi-date testing of everything may confirm that that processing center is Y2K-OK.

The bad news is: Unless you're Burger King, your processing equipment and fire control systems are not all identical. Even if -- unlikely -- twelve consecutive processing centers test Y2K-OK, it is still necessary to Inventory/Analyze the thirteenth.

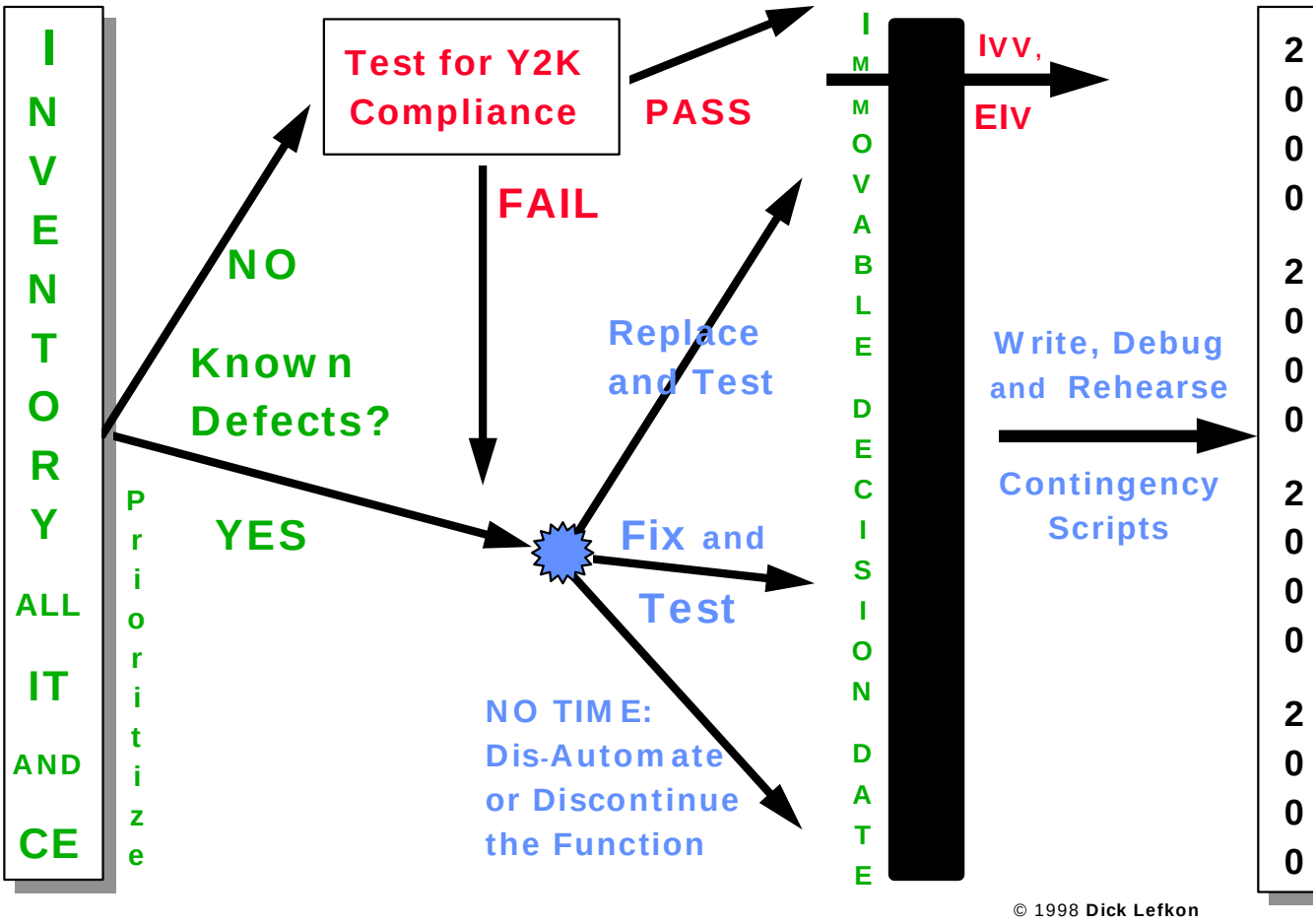


Figure 1 outlines the Y2K process for each capital equipment item, while illustrating at left that an item cannot be tested or fixed if it is not known.

WHAT TO DO NOW?

Since its first Y2K hearings in April of 1996, the House Government Management Information and Technology Subcommittee has pressed all Federal agencies to COMPLETE the Y2K Renovation/Replacement (Y2K Phase 3) of all critical systems by the end of 1998. Yet some haven't even completed Inventory/RiskAssessment (Y2K Phase 2) for their processing and premises equipment.

Time is short. While some capital items may be readily replaceable, others can require lead times of months, making it virtually impossible to preserve their function if they are discovered late in 1999 during a slowpaced Y2K inventory.

You can probably cut 75% off inventory time by avoiding recent grads and software specialists when assembling or contracting your Y2K capital equipment inventorying team(s):

- Collective hands-on experience of team members should cover HVAC, Fire/Security/Power, LAN/SW, Telecom and your specifics.

- Each team member needs extensive "been there, done that" experience in inventorying, acquiring, repairing and replacing equipment from the full line of vendors in his/her specialty area.

- At least some should have the ability to reprogram, reconstruct and/or test the embedded controllers. Although repair and testing come later, self-confidence that they can do so will enhance each item's inventory, as well as assignment of the right criticality and survivability ratings.

- Most team members should hold the EE or degree or equivalent.

- Each team member should possess a basic understanding of Y2K, and be satisfied to perform only the related to Y2K Inventory/Assessment.

WHAT INVENTORYING INVOLVES

Unless performed systematically, Year 2000 inventories are notoriously inadequate. Once Y2K software Renovation gets going, it’s not that rare for another 40% to 60% of modules to appear which were missed if the preceding Inventory Phase was done in a cursory manner.

Five prominent non-software risk areas are the centralized computer hardware, backbone/LAN hardware/software, telecom equipment/services,

distributed PC/desktop hardware/software, and this paper's main topic: non-IS embedded chip devices.

A concise but complete list of embedded device types can be copied or xerographed from the chapter by Harold Carruthers, "So You Can't Program Your VCR?," pp. 36-38 in this author's 1998 book, Year 2000: Best Practices for Y2K Millennium Computing (Prentice Hall, New York City; originally DPMA SIG-Mainframe, 1996). From Congressional testimony to book chapters, other authors have been prolific in permuting the contents of Carruthers' still-authoritative1996 list.

EMBEDDED CHIP DEVICES

Non-IS embedded-chip devices have been the big surprise of the Y2K effort. In 1996 and earlier, only relatively few of us allocated real resources for them in our project plans Fortunately, this author had been an electronics professor before entering IS, and the locating and Y2K remediation of embedded chips involved familiar tasks.

Whether or not you are comfortable with or amused by embedded-chip devices, they can be a real Y2K exposure if not renovated. For instance, your office building's chip-driven door locks might think February has only 28 days. Upon systematically mis-counting by one day, these devices could then proceed to lock you out one weekday per week from March of 2000 onward - and, also once weekly, admit the general public to your premises over the weekend!

Chip devices fall into two usage categories - premises and business function. Your survey instruments and cure methods will be different for the two.

Practically everyone has "premises" or "environmental" embedded chip devices - on elevators, fire suppressers, HVAC and physical security systems. In most cases these are administered by the Buildings & Grounds Department or equivalent. If your organization rents, this unit is not under your direct control: Ignore your inclination to settle for a perfunctory letter of inquiry. Instead, act as though the building manager is

just another one of your AVP's and bring your Inventory form and some patience to the first meeting. Explain why and how to fill it in.

At minimum, the form's columns should include:

* Category; e.g., Building Environment
* Device type; e.g., Heating Oil Meter
* Criticality
* Manufacturer/Vendor
* Vendor Telephone
* Vendor Contact
* Part Description
* Part Number
* Quantity
* Warranty Y/N
* Spec/Documentation is on file Y/N
* Interfaces with (list)
* Sunset/Replacement scheduled Y/N
* Obvious calendar code Y/N
* Calendar Trouble already
* Other Trouble already
* Contingency Workaround available upon failure
* Remarks (Will vendor have a Y2K-OK version? When?)

Additional analysis before planning your renovation might add columns listing the vendor's own Y2K classification of the device, a user group's classification, and the part numbers of risky components (or "N/A").

Now consider how and by whom the initial survey is completed. You probably don't want to assign this task to programmers. Despite their higher salaries they'll do less well than members of your operational departments and the purchasing office. The latter can check on existing warranties, spec sheets, vendor information, and successor vendors.

In the optimal case, people tasked with the building environment should report o oil meters' quantity, criticality, interfaces, calendar relevance and past troubles. Similarly shop floor staff are the best people to rate the importance and interfaces of their manufacturing robots - not costly software consultants.

RUNNING YOUR OWN Y2K INVENTORY

This is an inventory survey, not a complete solution or even - yet - follow-up research and scheduling. First mark down the primary information in the function-sequenced way people encounter it in daily worklife. Later on, when the time comes to contact Motorola or another vendor directly, you can always sort the inventory table on Columns 4 and 7 to produce an equivalent document which provides better support for telephone inquiries.

Remember, too, that about 85% of chip devices won't have any relevant calendar dependencies at all. Once this is explicitly verified and logged, a hospital with 15,000 such clinical devices has its burden reduced to about 2,250 - about 30 pages worth. Of these, perhaps 10% will be headed for discontinuation/replacement or be determined to produce strictly cosmetic effects such as a display timestamp. If today were mid-1998, you'd only need to address 1800 in 18 months - five per day, on average.

These devices can be classified for Y2K in at least four primary ways. First is isolated testing, such as turning power on-off-on or stay-on around midnight of such critical dates as 12/31/1999, 2/28/2000, 2/29/2000, and 12/31/2000. Second is the same test scheme in an integrated environment where appropriate. Third is component analysis based on the burned-in programs and the chips themselves. Last is human reassurances - from a vendor, organization website, neighbor enterprise, or your lawyer. Just remember that your operation will fail if its suppliers and payers fail, and the very act of requesting helpful data from your business partners can sensitize them to Y2K and help enable them (and therefore you) to stay afloat.

When scheduling which five devices to remediate that "first day," remember that the inventoried criticality and interfaces should yield the highest prioritization numbers. A device that could fail and cause loss of human life (or the corporate equivalent – cancelled insurability or charter) should be handled early, as should one which can cascade bad information into other critical systems - whether IS or not. Regardless of press releases, nobody will finish Y2K 100% so take your prioritization seriously.

IS AND TELECOM DEVICES

The survey form detailed for non-IS embedded-chip devices will work fairly well for the IS ones, too. The job is much easier because even a poorly-designed enterprise network might have thousands of nodes and routers and gateways and printers - but it won't have thousands of different models.

You might consider expanding the "Interfaces" column, though. It will probably need a list, not just Y/N or a single entry.

Most network hardware vendors claim to be in good shape this year, and when it comes time to verify, Legal might permit you to accept the word of the vendor or one of your affiliates. Keep in mind the costs of a model upgrade (or new vendor) in the cases where the manufacturer obsoletes what you own.

PC-attached devices are made in many flavors by many more vendors than standard network equipment. Just counting the ones for which you have spec sheets will drive this device count into the hundreds or higher. One assist - not perfect, but helpful - may be one of the PC inventorying programs offered commercially. Each PC's device and software inventory is written onto the insert floppy containing the executed inventory program. There are also LAN versions that can do the job centrally for networked PC's.

The third category, telecom hardware, is the most restricted set of all. You probably know exactly what you have because you have an enforceable organization-wide standard and, besides, you're likely paying a monthly charge for every phoneset. The telecom manufacturers claim to have made nearly all their customer-sited equipment (and firmware) Y2K-conformant.

After your eventual Y2K renovation/replacements, conformance validation need not be limited to explicit device testing that you perform or accept from others. All three categories in this section will be re-tested for Y2K compliance in your own organization's end-to-end validations later this year and next.

If you have devised good Y2K tests, this in-context validation will pick up flaws missed by individual device probes. That is a safety level unavailable

with most non-IS embedded chip devices. This single-level testing constraint is the single real downside to the fact that the isolated devices inherently limit the propagation of Y2K errors.

CPU-BASED MAINFRAME AND PC HARDWARE

Some mainframe and minicomputer hardware can never be made Year 2000 compliant. Your vendor will tell you the truth (and probably pull out a pencil and an order form). For mainframe/mini hardware you can use the inventorying methods just described for attached devices.

Mainframe/mini operating systems (the "real" computer) just special cases of shrink-wrapped vendor software, discussed in a following section.

The CPU-based PC hardware/software issue will itself be a non-issue next year. Vendors who needed to know, already knew last year exactly what was wrong with PC's for Year 2000 and how to remedy it. Right now, OEM's and resellers alike are still disguising and dumping their defective product. In another year most of it will be proudly owned by private individuals who are not reading this. At that point your organization will identify a supplier who furnishes a reliable spec sheet and replace everyone's desktop machine. Funding will come from the Y2K budget but your PC upgrade will be as much for social purposes as for anything else.

Just remember that "underprivileged" kid in the commercial who has merely a 33 megabaud modem. Somehow your organization will find the $2,000 this year or next to put a (Y2K-compliant) Intel Pentium IV-b 711 Mhz PC on your desk so your officemates won't make fun of you. At your salary level, the workhours lost to embarrassment and skulking would cost much more than that.

Just make sure you have an adequate number of Y2K-compliant PC's to use for your early testing.

Y2K-MOTIVATED MAINFRAME HARDWARE RISK/COST REDUCTIONS

Computers themselves have been a major category of Y2K-vulnerable embedded-chip devices. Noncompliant PC hardware is generally handled

by spot upgrading of BIOS chips and other identifiable components of newer machines, or by scrapping older machines and replacing them with recent-vintage 200-550 Mhz replacements.

The author and his technical teams have identified, repaired and replaced numerous computers, with platforms ranging from PC, to client-server, to OS390/MVS mainframes. The offending hardware need not be a 386 or old Pentium: Many IBM 308X mainframes, and approximately five thousand each of 43X1's and 937X's, will be more successful as paperweights than as CPU's when January of 2000 arrives.

Just from the standpoint of hardware compliance, two main nonexclusive replacement approaches have emerged:

- Enlarge the most recent IBM/Amdahl/Fujitsu mainframe, accommodating the same workflow on fewer and larger mainframes.
In general this is acquired directly through the manufacturer's regular sales organization. The second altenative is not.

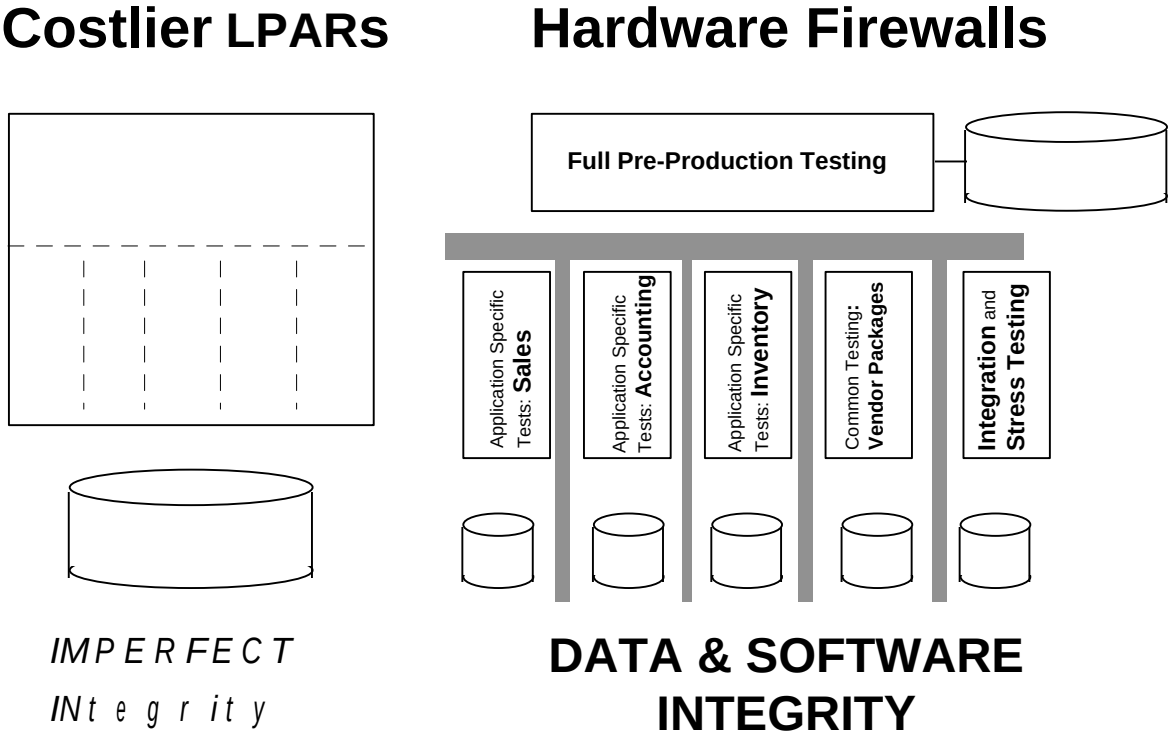
- Swap the tasks of the noncompliant box onto a "pygmy" mainframe, sucn as the IBM's R390 or Expert Systems' CDS-2000. Both of these have been on the market four years, are (from personal experience) straightforward to install, deliver processing power between 15 and 50 times the non-compliant machines they replace, and cost the same as or less than a single year's services of a mainframe consultant.

On the Y2K software side, "Time Machine" has come to mean an isolated computer used for remediation and testing of designated software applications. Isolation has been the key, because forward-date interaction with noncompliant production systems might corrupt or derail production:

- Software Licenses may expire, delaying further work
- Passwords and file retentions may expire
- Catalogued data and spooled output may disappear.

In tightly-controlled mainframe settings, mainframe Y2K software activities have successfully been confined within an LPAR or VM GUEST ln an active production machine. An argument for using one or more isolated pygmy mainframes as the Time Machine(s) is illustrated in Figure 2.

Figure 2 Y2K test Environments:



Besides the inherent risk reduction in the arrangement at right, there is a fairly drastic cost reduction as well. The cost reduction stems from the fact that each of the 8-MIP pygmies, based on an IBM-manufactured S/390 hardeware core, qualifies for IBM's so-called Entry System Licenses for the software:

- On a modest IBM Group 40 machine (25 MIPS), recent 3-year IBM software license costs have been $1MM for OS390, $228K DB2, $222K IMS, $283K CICS, $60K RACF, $64K COBOL.

- On each of three equivalent 8 MIP pygmies, the same licenses are $33K for OS390, $7K DB2 or IMS, $9K CICS, $2K RACF or COBOL. (CA also offers a similar but only-80%-off policy for the CDS-2000 pygmy)

Thus, deconstructing a more muscular IBM/Amdahl/Fujitsu machine into application-specific pygmy mainframes, can drastically reduce large-scale license costs for software used by relatively few, while still reducing risk.

TELECOM SERVICE AND OUTSOURCED FUNCTIONS

This portion of your inventory probably belongs on a legal pad, not on a table with many neat rows and columns. You currently have a very limited number of long-distance, Baby Bell and Bell-lookalaike phone services. Each has its own attributes, and the best shared attribute is that the phone service industry is in constant flux.

Most telecom companies have blandly assured their customers that their networks will be Year 2000 compliant by some time in 1999. That doesn't do your 1998 worldwide testing much good, does it?

Use some variation of the vendor contact letters featured in the March issue. This is a mission-critical system(s) even though it is entirely the "other guy's" fault. Stay on top of your telecom vendors and their potential replacements. If you don't like what you learn in your Inventory Phase, be prepared to make or urge an unpleasant decision regardless of their size. Just remember that Sperry-Univac was once the Big Computer Company, not IBM.

Especially when your end-to-end test time arrives, remember it is perfectly legal to switch vendors - or, at least, put on additional service from a conformant vendor for the duration of your worldwide test.

Outsourcers are a different category entirely. Your business is much more visible and important to them. And by this time they have clearly shown their hand. There is a difference between generic business outsourcers and Y2K Fix Houses.

Although maybe not perfect, your business-outsourcer(s) should be at least slightly ahead of the Y2K wave midpoint for your industry. That is because they have many clients with application needs like yours, and most of the Y2K problems you identify will already have arisen.

TECHNICAL LOOK AT EMBEDDED SYSTEMS

Most embedded systems are designed to work on their own, performing only one or two tasks independently of other systems. Fixing them will require not only a hardware change but also changes to the embedded software. In many cases, the companies that designed the devices no longer exist. In those whose doors are still open, management's concern with quarterly profit may have taken precedence over spending to fix a problem that has yet to occur.

One of the best ways to grasp the technical issues presented by an embedded system is to walk through the design of one. In keeping with the Security themeof these Proceedings, thedesignated example is a simple door security system whose functional requirements are that:

The status of four entry doors be monitored.
Four electric door locks (one for each door) be controlled.
The normal hours of business (when the doors will be unlocked) will go from 8 in the morning to 5 in the evening on Monday through Friday.
No entry be permitted on Saturday or Sunday.
No entry be permitted on company-specified holidays.

These requirements imply that the security system will need four inputs (for monitoring the entry doors) and four outputs (for controlling the electric door locks). It will also need a clock so that the system can keep track of times, days, and (for keeping track of holidays) dates.

The dates of company holidays will be programmed into the security system using a PC and a serial communication link. For simplicity's sake, assume that the PC used by company personnel to input holiday dates is already year 2000 compliant and has the menus needed for inputting the dates. All the security system must do is make use of that data.

EMBEDDED HARDWARE COMPONENTS

The common building blocks for designing an embedded system are a microprocessor for running the control program, ROM to store the program, RAM for holding variable data, a real-time clock , and peripheral devices for accepting inputs about the doors' status and providing outputs

capable of controlling the door locks. The component most critical to system performance, and usually the first to be selected, is the microprocessor.

To keep costs low, this has historically meant a choice of 8-bit processors from Intel Corp., Motorola Inc., and Zilog Inc. -- the 8051, HC11, and Z8 families, respectively. The Intel 8051 family of microprocessors is used in this example, but either of the others could also serve.

The design requirements imply that the system will also need ROM and RAM. The ROM is where the microprocessor's main program resides, the program that tells the microprocessor exactly what to do, step by step. Note that the control program can be changed only by replacing the ROM in which it resides or, in the case of electrically alterable ROM, reprogramming it. The RAM will be used for storing that data that varies--the status of the doors, the time, date, the number of days in operation, and any variables used by the internal program.

ONE-CHIP WONDERS

Like other microprocessors intended for embedded applications, the Intel 8051 has both RAM and ROM on the same chip as the processor. Such chips are often referred to as microcontrollers because they are so useful in control applications. If the control program and the variable data are small enough to fit in the on-chip ROM and RAM, the need for two external memory chips disappears and saves money.

The memory architecture of the Intel 8051 family comprises two areas: 4KB of on-chip program memory (ROM), and 128B of data memory (RAM). Including internal memory, the chip can access up to 64KB (65 536) distinct address locations in ROM and the same number in RAM, which can be used to reference memory locations in additional memory external to the chip, if needed.

The chip also has four 8-bit input/output ports, each connected to an external package pin. Four of these general-purpose port pins can be used to accept input signals from the doors. Each door has a microswitch that opens the circuit when the door is physically opened. This action causes a voltage change on the port pin to which it is connected. The four

extra general-purpose port pins serve to control the electrical locks on the doors.

A serial communications port for connection to a PC is also available in the Intel 8051 chip. However, another chip is needed to translate the voltage levels from the microprocessor into those required by the PC. The Maxim MAX232, a popular RS-232 receiver/driver chip, performs this task quite nicely. Further, the port can be can also be used for communicating with an alarm if the security system finds that the electronic doors have failed to lock.

RTC: REALTIME CLOCK

The last remaining hardware item to be covered is the real-time clock (RTC). This chip feeds the program running in the microprocessor with knowledge of exactly what the time-of-day, day-of-the-week, and year are. Semiconductor clock chips are available from a variety of vendors. Among the most popular are the variations available from Motorola Inc., National Semiconductor Corp., and Dallas Semiconductor Corp. Virtually every clock chip designed is based on the original architecture of the Motorola real-time clock. Many Pentium processors even use the same basic register layout as this first real-time clock.

The Dallas Semiconductor DS1287 clock is highly integrated. Its physical package contains not only the clock chip, but also a rechargeable lithium battery (to keep the clock running should external power be lost) and a 32-kHz crystal oscillator, which serves as a time base for the clock chip. The integrated package saves space on the circuit board and the cost of additional components, such as a battery and battery holder.

The DS1287 is totally compatible with the industry-standard Motorola clock. That is, the information it supplies to the system is in the form of seconds, minutes, hours, day-of-the-week, day-of-the-month, month, and a two-digit year [Table 1]. With the exception of one mode, all data is represented in two-digit binary-coded decimal (BCD) format. (The exception is the 12-hour clock mode of operation. If it is selected, then 1 bit of the hour byte is used as an AM/PM indicator. Otherwise, for 24-hour mode, the first BCD nibble represents the tens place (0, 1, or 2) and the second the ones place (0-9).)

Because Motorola-standard clocks use only two binary-coded decimal digits – a single byte -- to represent the year, they are not year 2000 compliant. What is missing is the century information (19XX, 20XX).

For the last 50 years or so, computer systems have assumed that the digits preceding the year and decade digits were 19 and ignored them when making calculations. So when asked to calculate the difference between 1999 and 2000, the system uses only the last two places of the date and comes up with an answer of 99 instead of 01. Further, the calculation of a whether the year is a leap year, if based on the assumption that 00 is 1900, is incorrect for 2000.

With the hardware design completed, all that remains is to write the software control program that will be embedded in the microcontroller's ROM. It begins by fetching time, day, and date information from the real-time clock, and then checking each item to determine the required state of the electric locks. If the time-of-day is between 8 a.m. and 5 p.m., and the day of the week is Monday through Friday, and it is not a company holiday, the doors are unlocked. At any other time the doors are locked. Should it be determined that any lock is in an incorrect state, the system sets off an alarm.

Also included is a routine that checks to see if the controller has received its yearly checkup ("Is maintenance = OK?"). This feature will reduce not just legal liability for, but real exposure to, criminal trespass as a result of a malfunction. Since the system is providing security during all off-hours, security would be compromised should the system fail for lack of proper maintenance. Second, in normal system operation, it will prevent a system malfunction that could lock the employees out of the building all day on a scheduled workday.

THE HIDDEN FLAWS

The complete system satisfies all practical requirements. Once installed, it functions properly and, over time, is taken for granted. As for any upgrade, the attitude is a common one: if it ain't broke, don't fix it.

Herein lies the problem. The system is working well, and its inherent limitations are invisible. Unfortunately, the designers of the semiconductor clock chips overlooked the looming problem of the year 2000 rollover. When those clock chips were first being designed, it was standard practice to drop the century data.

Further, designers were continually told that the life cycle of a new product was usually three and at best five years. Suppliers may well have been replacing their older products with new designs at that frequency, but in the real world, systems are replaced only if they suffer a catastrophic failure. Unlike the PC market, where system upgrades are held to be necessary every six months, embedded system designs based on 1970s technologies are still performing their designated functions today. As long as the system is performing its task properly, it will remain in place for years.

The technical problem stems from the two-digit year register internal to the clock chip. This year-counter will roll over to 00 when the year 2000 arrives; but to the embedded system, the date will be 1900 instead of the year 2000. One immediate and one belated misstep ensue. First, the maintenance check will find that the system has not been serviced for 99+ years, and therefore keep the doors locked until maintenance is performed. (Since 1 January 2000 is a Saturday, the flaw would not detected until 8 a.m. on Monday, 3 January, when everyone tried to report for work.)

The second obstacle would pop up in March, because of the awkward fact that the year 1900 was not a leap year but the year 2000 is. Since this embedded system uses a day-of-week indicator, it will begin to go awry on Tuesday, 29 February. To the system, 4 March will be a Friday, not a Saturday. If this system were connected to another system that has been corrected, both could become corrupted.

THE DOUBLE FIX

Correcting the problem presented by the security system calls for two changes, neither of them trivial. First, the real-time clock chip has to be replaced so that it can function properly after 31 December 1999. Second, the control program has to be altered so that it can properly calculate when maintenance was last performed.

In 1992, Dallas Semiconductor developed a semiconductor clock chip, the DS1687 RTC , that is year 2000 compliant; it was the first in the industry to provide a true four-digit year code. In the older chip, the user could store a century number in nonvolatile RAM, but this number was not automatically incremented by the clock chip. The new chip design makes use of a byte-wide counter for the upper two binary-coded decimal digits representing the hundreds of years that is automatically incremented [Table 1]. In addition, it compensates for the leap year that occurs in the year 2000. The chip may be retrofitted into any existing design that utilizes the older Motorola-standard technology.

The fix applies only to the hardware side of the embedded system problem--nothing has really changed unless the software in the ROM is revised to access the new data, The security system used as a design example uses a variable to store a date that indicates when the last maintenance check was performed. When the rollover occurs, the system assumes that it has been 100 years since it was last serviced, and pulls itself out of service, indicating that a malfunction has occurred. Any non-year-2000-compliant embedded controller replacement for this unit will also indicate a malfunction as soon as the actual date is entered into the system.

Assume the software embedded system does not care about the year field in the service variable. When the rollover occurs, nothing happens--until 29 February rolls around. When it does, the system is off by one day, affecting the day of the week perceived by the system. Wouldn't it be great to have the system assuming it is Sunday when it is really Monday? Until fixed, it will make this assuption each Monday!

Some microcontrollers are ROMless devices and so require an external ROM chip containing the software code for the embedded system. This chip can be updated and replaced without removing the microprocessor from the system. Processors with internal ROM contain embedded control software and have to be replaced to update a system. Some systems may make use of both internal and external ROM, and so both the processor and external ROM need replacing.

Solving the Y2K problem for embedded systems does not require an advanced degree in physics. So armed with the above knowledge, a good assembly-language programmer and a soldering-iron jockey with a steady

hand can go about renovating century-defective equipment. By having a firm idea of how embedded chip devices were designed and can be upgraded, even the least technical coordinator of a year 2000 project can arrange for the parts and personnel needed to trap and fix the problem.