CLASSIFICATION <div style="text-align: center; margin-top: 20px;"> <div style="display: inline-block; vertical-align: middle; text-align: left;"> BATTELLE <small>BATTELLE MEMORIAL INSTITUTE</small> <small>POST OFFICE BOX 999 / RICHLAND, WASHINGTON 99352</small> </div> <div style="display: inline-block; vertical-align: middle; text-align: left; margin-left: 10px;"> NORTHWEST <small>PACIFIC NORTHWEST LABORATORY</small> <small>POST OFFICE BOX 999 / RICHLAND, WASHINGTON 99352</small> </div> </div>	DOCUMENT IDENTIFICATION NO. BNWL-SA-5268 COPY AND SERIES NO. DATE DECEMBER 1974				
TITLE AND AUTHOR AUTOMATION OF A BRAZING PROCESS WITH AN INTEL 8008 MICROPROCESSOR T. A. SEIM	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%; padding: 2px;"> CONTRACT ☐ - 1830 ☐ - 1831 </td> <td style="width: 50%; padding: 2px;"> PROJECT NO. </td> </tr> <tr> <td colspan="2" style="padding: 2px;"> RESERVED FOR TECH. INFO. USE </td> </tr> </table>	CONTRACT ☐ - 1830 ☐ - 1831	PROJECT NO.	RESERVED FOR TECH. INFO. USE	
CONTRACT ☐ - 1830 ☐ - 1831	PROJECT NO.				
RESERVED FOR TECH. INFO. USE					

D I S T R I B U T I O N					
NAME	COMPANY	LOCATION	NAME	COMPANY	LOCATION
			<i>Conf-750304--2</i> MASTER		

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research and Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

ROUTE TO	PAYROLL NO.	COMPANY	LOCATION	FILES ROUTE DATE	SIGNATURE AND DATE

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Automation of a Brazing Process With
An Intel 8008 Microprocessor

T. A. Seim

BNW

WORK PERFORMED UNDER AEC CONTRACT AT(4511)-1830

Automation of a Brazing Process With An Intel 8008 Microprocessor

INTRODUCTION

The need for automatic control of a joining process operated by one of the AEC's production agencies became apparent after an abnormally high failure and rejection rate was experienced. Up to that point no automatic control whatsoever was used in the process, resulting in great variations of weld quality from part to part. Furthermore, operators were allowed considerable individual judgment in the adjustment of process parameters. Variances due to operator differences are unpredictable and undocumented.

TECHNICAL DISCUSSION

The process involved is a ring brazing operation similar to the simplified drawing in Fig. 1. A rod shaped component is placed in a hole of the mating part along with a ring of brazing compound. An induction heating coil positioned over the rod heats the parts to brazing temperature. This entire apparatus is placed in a vacuum chamber evacuated to about 10^{-5} torr.

Control of the process necessarily required measuring the temperature at the brazing joint. This was complicated by the limitation that the part could not be contacted (high surface finishes precluded that). Optical pyrometry was experimented with, but proved unworkable due to the vacuum environment. A non-contacting temperature sensor was developed based on the principle illustrated in Fig. 2.

The device uses two thermocouples and a surrounding 'heat bucket'. The principle of operation is to control the temperature of the heat bucket to within 5 deg C of the primary thermocouple. This is accomplished using a secondary thermocouple and heater coil attached to the heat bucket. A proportional temperature control loop maintains the thermal balance. The purpose of the heat bucket is to simulate a thermal 'black body' behind the primary thermocouple. This minimizes temperature gradients between the point being measured and the measuring thermocouple.

Given the ability to measure the braze joint temperature, the control system had to maintain the joint temperature within an 8 deg C error bound and generate a time-temperature profile. The control system also had to accommodate future changes in the profile and control algorithm.

Off-the-shelf process instrumentation was available which could have been wired into an adequate control system. The primary reason for selecting a microprocessor instead of a hard-wired system was for programmability of the control functions. It was felt that changes inherent in development projects could be better effected through reprogramming than rewiring. It also served as an opportunity to evaluate a new and important technology.

The programmability of microprocessors infers the ability to reprogram them. This bit of obvious logic manifests itself in several ways. Clearly many of the changes that occur in a control system throughout its lifetime could be accommodated by changing the program instead of the wiring. Complex functions which in a hard-wired system would mean adding more memory. Thus by allowing room for additional memory for the microprocessor in the development states long term flexibility is assured. This may be very impractical for a hardwired control system.

The fundamental control system takes the form of Fig. 3. Process actuation is achieved by adjusting the field voltage of the motor-generator used to power the heating coil. This is done using a variable transformer driven by a stepping motor. Control feedback is accomplished using the temperature sensor previously described. The operator's responsibilities (apart from readying

the process for operation) include: initiating a braze cycle, signaling a successful braze, and aborting a process cycle having difficulties. Subjective decisions on the part of the operator are minimized, a necessary condition if joint differences are to be eliminated.

A prime problem in using a microprocessor was adapting it to an instrumentation package suitable for an industrial environment. This problem was solved once it was learned that Lawrence Livermore Laboratories had designed the Intel 8008 microprocessor into their standard electronic hardware package. The card sets comprising the microprocessor are available commercially through Control Logic, Inc. Fig. 4 pictures printed circuit boards and an equipment box similar to the one used for the braze controller.

LLL also assisted in software development by providing a debug program which allows a user to interact with the microprocessor via an alphanumeric terminal. This, however, is merely the tip of the iceberg, so to speak, as far as the problems encountered during software development.

Developing a significant control program for a microprocessor requires considerable program development support. By 'significant' it is meant more than a few hundred program bytes. It is generally necessary to code programs in a symbolic code and use an assembler program to convert the code to machine language. Doing this manually becomes very time-consuming and error-prone, and must be repeated if the program is changed to any degree. Microprocessors are capable of assembling programs given sufficient memory and input-output peripherals, but this is counter to the desired goal of a minimal control system configuration. An alternative to this dilemma is to have a separate microprocessor for development purposes. This means investing in additional equipment and thus warrants considering other alternatives.

Some microprocessor manufacturers have developed software for assembling programs and other functions which are available on time-sharing services. A prospective user would lease a

terminal from the time-sharing service and have access to these programs and other useful file maintenance routines with no capital investment. For the user with no internal computer facilities this is a viable alternative.

If computer facilities are available there exists the alternative of adapting these to provide the requisite support. This usually involves writing significant programs such as assemblers, which may be beyond the abilities of the user. As microprocessors become more commonplace this kind of software may be available, either from the manufacturer or from a systems house. This is especially true if the computer is an IBM 360 or 370. Other possibilities include using software written in high level languages (such as FORTRAN). A word of caution is in order here. Programs such as these become machine dependent, that is to say, they may not run on another computer without extensive modification. Since they usually involve an appreciable cost, guaranteed operation should be insisted on. Reluctance to provide a guarantee may be interpreted as guaranteed difficulties.

One final alternative is available to those with some programming acumen and a computer having a macro assembler. Macro refers to representing a series of repetitive instructions by a single symbol. This feature can be used to represent the instruction set of a 'target' microprocessor. Basically the technique involves forming individual instructions using the macro definition features of the assembler. The user must be very familiar with the particular system to do this, but an assembler can be implemented with surprisingly little effort by this approach.

In this instance an in-house computer (SEL 840A) was available for software development. It did not have a macro assembler so an assembler for 8008 code was written for the computer. A simulator for the 8008 was also developed to debug microprocessor control programs. While a debug program was available for the 8008 which allowed the user to communicate with the microprocessor using a Teletype, the usefulness of the routine was impaired by the

inability to set 'breakpoints' and single step the program. It was also necessary to develop the control program without the 8008 hardware.

The simulator was particularly helpful by allowing not only simulation of the 8008, but also the controlled process. Without total system simulation it would have been impossible to fully test the control program. In general, a simulator which does not allow simulation of the process is of little value in debugging process control programs. Simulators on time-sharing services fall in this category. In the case of the braze controller, the control program ran without error the first time it was executed on actual microprocessor hardware.

The control program developed for the braze controller may be depicted as shown in Fig. 5. The main elements are the sequence controller, the time-temperature profile generator, and the three mode controller. The sequence controller insures that appropriate actions are taken during the various phases of the brazing operation. The controller is implemented in software as state driven logic from data stored in a table. The use of tables is an efficient and highly flexible means to implement control sequences. Furthermore it is a direct realization of the state diagram, which is a commonly used aid in designing control logic.

The time-temperature profile generator delivers a 'set point' to the three-mode controller. This profile was originally derived from a table, which permitted any arbitrary profile to be generated (also thermocouple linearization could be achieved by correcting entries in the table). This flexibility was not necessary in this application, so a piece-wise linear algorithm was substituted for the table.

The three-mode controller algorithm presented the greatest difficulty for the software design. The principal problem was that the 8008 (and the 8080) has no status checking for arithmetic overflow and signed arithmetic magnitude comparison. Overflows

must be checked so that incorrect control responses are not made (this usually results in control outputs in the opposite direction to that desired-- a control disaster). Since the 8 bit word size allows for a number range of -128 to +127 it is not possible to scale inputs to the computation so that overflow never occurs (multiplication, for instance, requires that the product be in this range, limiting the multiplier and multiplicand to +11). Software checking for overflows could be made, but this turns out to be nearly as complex as doing computations in double precision and not checking. The multiplications in the three-mode algorithm are made with single precision operands (8 bits) which result in double precision products (16 bits) that are summed and are converted back to single precision. One range check is made at this point to determine if the control output exceeds the range of the stepping motor range (about 310 degrees rotation or 185 steps for a 200 step/revolution motor).

The lack of application software at the time of the development necessitated writing all arithmetic subroutines in-house. Only later was an Intel developed multiplication subroutine incorporated in the program. This turned out to be a mixed blessing as the routine did not work for the most negative number, -128. As microprocessors mature the software situation will probably be alleviated.

The economics of using microprocessors is considerably different from those of hardwired controllers. The processor represents a fixed cost regardless of the complexity of the program. The cost of increased complexity is primarily a function of the cost of additional memory, which may be small in comparison to the overall system cost. A hardwired system, on the other hand, experiences costs proportional to overall complexity. Consequently there is a cross-over point with increasing complexity where the microprocessor becomes more cost effective than a hard-wired system.

These economic differences make it feasible to 'over design' a microprocessor based controller adding additional functions at low cost (or no cost since some memory space may be unused) once cost justification has been achieved. Built in diagnostic routines, for instance, could greatly simplify maintenance problems. The processor also might keep tabs on system performance to flag impending failures or aid in quality assurance.

A significant cost in the recording of this kind of information is the recorder itself. Storage of the charts themselves may be a considerable problem. A microprocessor could use its alterable memory for storing this information for relatively short periods and transmit it periodically for medium and long term storage. The receiving device may be either a large computer or some sort of peripheral device connected temporarily to the microprocessor by maintenance personnel.

In either case the communication interface would be standardized and a one-time investment. Serial data transmission, such as EIA standard RS232, would minimize cabling costs by using the full data bandwidth of the line. This is especially important where cabling can represent up to 40% of the entire control system cost.

Furthermore, the capability to transmit information back to the microprocessor is essentially free (the bulk of the cost in cabling is in labor). A system which starts out as several independent controllers could be later upgraded into an integrated control system when the situation merited it.

RESULTS

The 8008 microprocessor proved to be more than adequate for the control task. It exhibited excellent noise immunity and no failures of the processor have occurred. Although it is a relatively slow microprocessor in comparison with those currently available, it is capable of 40 three-mode control computations per second. The process is being controlled at 10 per second.

The programmability of the controller was a valuable asset. Improved control was achieved by changing the linear control algorithm to a piecewise linear algorithm. Display and data output functions were added at the request of production personnel. All of these changes would have been difficult and expensive to implement if the controller were hardwired.

CONCLUSION

The future of microprocessors in control applications will depend on such things as performance and reliability. But in the end they will succeed based upon one criterion-- economics. Microprocessors will have to achieve cost justification through either lower initial cost, longer service lifetime, or by lower maintenance costs due to higher reliability. This implies a comparison with an alternative design approach in contrast to an application uniquely suited to a microprocessor. For the near term microprocessors will be used largely as a substitute for existing or planned hardwired controllers with much less emphasis on exotic systems such as distributed processors.

Fig. 1 a ¹⁰⁰Brazing setup.

Fig 2. Non-Contacting ^{u.c.} temperature _{sensor} ^{u.c.}

Fig. 3. Brzecz control system.

Fig. 4

A specialized process braze controller was designed with a standard microprocessor. Standard components were used throughout for simplified maintenance.

Fig. 5. Control Program ^{u.c.}block ^{u.c.}diagram ^{u.c.}

REFERENCES

1. Altman, L., "Single Chip Microprocessors Open Up a New World of Applications," *Electronics*, April 18, 1974, pp. 81-100.
2. Cadzow, J. A., Discrete Time Systems, Prentice-Hall, Englewood Cliffs, NJ, 1973.
3. Cushman, R. H., "Experts Predict the Future Course of Microprocessor Components," *EDN*, February 20, 1974, pp. 24-26.
4. Cushman, R. H., "The Intel 8080: First of the Second-Generation Microprocessors," *EDN*, May 5, 1974, pp. 30-36.
5. Davis, S., "A Fresh View of Microcomputers," *Computer Design*, May 1974, pp. 67-78.
6. Gladstone, B., "Designing With Microprocessors Instead of Wired Logic Asks More of Designers," *Electronics*, Oct. 11, 1973, pp. 91-104.
7. Holt, R. M. and M. R. Lemas, "Current Microcomputer Architecture," *Computer Design*, February 1974, pp. 65-73.
8. Kildall, G.A., "High Level Language Simplifies Microcomputer Programing," *Electronics*, June 27, 1974, pp. 103-109.
9. Kuo, B. C., Analysis and Synthesis of Sampled Data Control Systems, Prentice-Hall, Englewood Cliffs, NJ, 1963.
10. Ogden, J. L., "Survey of 8-Bit Microprocessors Reveals Wide Choice For Users," *EDN*, June 20, 1974, pp. 44-50.
11. Schmid, H. "Monolithic Processors," *Computer Design*, Oct. 1974, pp. 87-95.
12. Seim, T. A., A Microprocessor Sampled Data Process Controller, BNWL-1795, Battelle-Northwest, Richland, WA, November 1973.
13. Weiss, C. D., "Software for MOS/LSI Microprocessors," *Electronic Design*, April 1, 1974, pp. 50-57.
14. Weissberger, A. J., "MOS/LSI Microprocessor Selection," *Electronic Design*, June 7, 1974, pp. 100-104.