/AN INVESTIGATION OF THE
USE OF A DIGITAL PROCESS MONITORING AND CONTROL SYSTEM
IN AN UNDERGRADUATE CHEMICAL ENGINEERING LABORATORY/

by

BERNARD THEODORE GIEFER, JR.

B.S., Purdue University, 1983

A REPORT

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

CHEMICAL ENGINEERING

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1989

Approved by:

Major Professor

TABLE OF CONTENTS

TABLES

ABBREVIATIONS

AC   – alternating current

ACE   – Application and Control Engine

A/D   – analog/digital

AR   – assembly register

CPU   – central processing unit

DBB   – Database Builder

DC   – direct current

DOS   – disk operating system

EGA   – enhanced graphics adapter

FIX   – Fully Integrated Control System

I/O   – input/output

Kb   – kilobyte

mA   – milli-amperes

Mb   – megabyte

ms   – milli-second

PC   – personal computer

PID   – proportional, integral, and derivative

PLC   – programmable logic controller

RAM   – random access memory

SAC   – Scan, Alarm, and Control

SCADA – Systems Control and Data Acquisition

TC   – thermocouple

Chapter I

PROJECT INTRODUCTION AND RATIONALE

Recently, sophisticated, low-cost, process monitoring and control software and hardware for personal computers, or PCs, has become available. This has enabled academic institutions, who have previously been constrained by budgetary concerns, to seriously consider development of PC-based, digital monitoring and control systems for their departmental process control instruction. Personal computer interfaced digital systems has brought within reach the desire of academic institutions to expose engineering students to the fundamentals of modern process monitoring and control on a scope that is not only intended as industrial simulation, but is also of industrial quality.

The rationale for computerized industrial process monitoring and control centers around the speed, accuracy, and reliability of modern digital systems. A

proliferation of PCs in the office place, and the push to minimize the physical size of monitoring and control systems while increasing capability, has been imparted in large measure by technological advances in semi-conductor design and production. The result has been a large commercial demand for PC-interfaced monitoring and control systems. The rapid expansion of the process control industry has made it incumbent upon educators that digital monitoring and control be integrated into the syllabus of all engineering curriculum.

In fact, realistic and modern automated process monitoring and control has become an essential intructional tool at universities; it provides unparalleled introduction to state-of-the-art industrial process monitoring and control hardware and software for aspiring engineers. The rapid increase in the number of PCs at academic institutions and the broad diversity (in cost and quality) of commercially available PC-based software packages has enabled colleges and universities to seriously consider the installation of industrial quality digital process monitoring and control systems on laboratory equipment.

Hence the charge for this project; research educationally viable process monitoring and control options which will permit an active operator interface

through a PC. The purpose is to expose chemical engineering students to an industrially credible process monitoring and control system. Then, select, procure, and install a digital monitoring and control system to upgrade existing analog features on the heat exchanger experiment in the undergraduate laboratory at Kansas State University's Department of Chemical Engineering.

Chapter II

THE PROCESS APPARATUS AND EXPERIMENTAL PROCEDURE

The system for which digital process monitoring and control is applied is depicted in figure 1. The apparatus features a conventional tube-in-shell heat exchanger where water passes through the tubes and steam passes through the shell. The exchanger is a Bell and Gossett Model 42-2 (figure 2). A pneumatically operated air-to-close control valve with a valve-stem positioner is used to regulate steam flow to the shell. For a temperature sensor, a jacketed copper-constantan thermocouple is situated at the center of the effluent water stream. Finally, an analog recorder controller is used to plot effluent water temperature. (Digitally stored historical trending will replace the mechanical recorder).

The heat exchanger experiment is conducted as follows:

To measure frequency response:

4

air pressure
to valve

steam
line
pressure

steam
inlet - - ►

tube-in-shell
heat exchanger

water
inlet - - ►

heat exchanger
steam pressure

water
outlet

steam/
condensate
outlet

thermocouple

controller

current-to-
pressure
transducer

air
line
pressure

air
filter

air
pressure
regulator

air supply - - ►
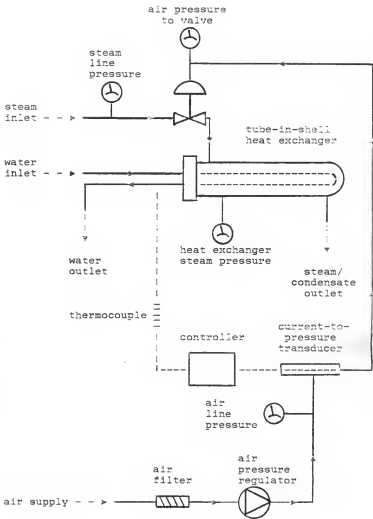
Figure 1. Schematic of the process apparatus.

5

|------------25.75 inches------------|

steam inlet

water inlet

water outlet

steam/condensate outlet
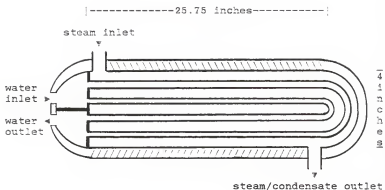
4 inches

Descriptors:

Shell: 4 inch inside diameter, 25.75 inches
         long; made of steel.

Baffles: 1-1 inch wide ring shaped baffle,
         13.5 inches from the tube sheaf.

Tubes: - all tubes made from 0.75 inch outside
           diameter 18 gauge copper.

         - 4 hair-pin curved tubes.

         - 2 tubes totaling 50 inches long.

         - 2 tubes totaling 46.75 inches long.

Tube support and fastening hardware: steel.

Figure 2. Bell and Gossett model SU-42-2 heat exchanger.

6

1) Completely open the pneumatic valve.

2) Connect the pneumatic valve to a 20 psi air supply.

3) Connect output of a sine wave generator to the input of the pneumatic valve.

4) Turn on the air, water, and steam supply valves.

5) Record the inlet sine wave and outlet temperature.

To measure step response: Proceed as for the frequency response, but replace sine wave forcing function with a manually induced step change from the controller.

For the purposes of this experiment, the sole point of control is the regulation of outlet water temperature by manipulation of the input steam flow. A block diagram for this control scheme is at figure 3. From an engineering standpoint, this may not be the "best" control solution, but this scheme lends itself to introductory educational instruction.

The primary goal of the experiment is:

1) to determine optimal PID controller settings from two methods: frequency response and from response to a step change,

then:

2) to investigate the closed-loop response to
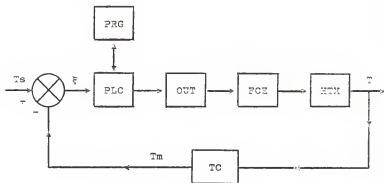
Abbreviations:

FCE – final control element (i.e., valve).

HTX – heat exchanger.

OUT – output module.

PRG – monitoring and control program.

PLC – programmable logic controller.

TC – thermocouple.

Tm – measured temperature.

Ts – temperature setpoint.

T – output temperature.

ξ – error (Ts–Tm).

Figure 3. Process control scheme for the heat exchanger.

8

various load and set-point changes,

and finally:

3) to model the closed-loop system using simulation methods.

To determine controller settings by the frequency response method, a sinusoidal variation in air pressure is applied to the control valve, which, providing the frequency of the sinusoid is sufficiently low, will produce a sinusoidal variation in the exit water temperature which is approximately proportional to, and the same frequency of, the air pressure sinusoid. This condition satisfied, the outlet water temperature is measured as a function of time. The amplitude ratio and the phase lag of the output (water temperature sinusoid) versus the input (air pressure sinusoid) is calculated. This process is repeated for several different input frequencies. Thereby, a BODE plot is generated by plotting amplitude ratio versus frequency on log-log scale, and phase lag versus frequency on semi-log scale. An example of this is at figure 4. The Ziegler-Nichols controller settings are then determined, where:

$$K_u = 1/A$$

$$P_u = 2\pi/\omega_{co}$$
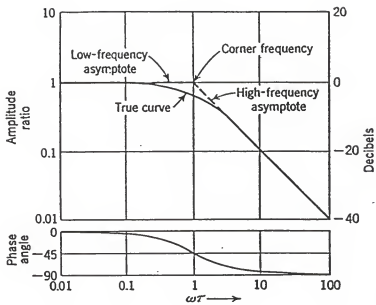
Where:

ω = frequency

τ = system time constant

Figure 4. Example BODE plot.

$$K_c = 0.6K_u$$

$$\tau_I = P_u/2$$

$$\tau_D = P_u/8$$

where: ω_{co} = cross over frequency.

A = amplitude ratio at ω_{co}.

The settings determined from a step response are calculated by the Cohen-Coon method. In this portion of the experiment, a step change in air pressure is introduced to the pneumatic valve and the response of the outlet water temperature is plotted against time. From the resulting "S"-shaped curve a tangent is drawn through the point of inflection (see example in figure 5). From the method of Cohen and Coon:

$$m = b/c$$

$$K_c = 1.35(1+m/5)/b$$

$$\tau_I = 2.5a(1+m/5)(1+3m/5)$$

$$\tau_D = 0.37a(1+m/5)$$

where: a = apparent dead time.

b = intercept of tangent line and temperature axis.

c = difference between beginning and final steady state values.
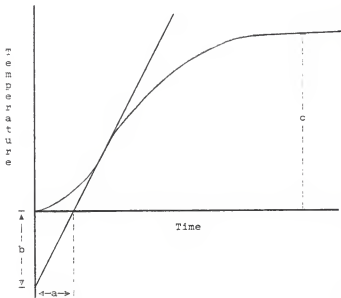
Where:

a = apparent dead time

b = temperature intercept

c = final steady state - initial steady state

Figure 5. Example Cohen-Coon plot.

The controller settings calculated by the Ziegler-Nichols and Cohen-Coon methods are then used as initial controller settings. The response of the controlled system when subjected to load and set-point changes is measured for variations in the controller settings. This data is then used to determine "good" controller settings necessary for the particular response characteristics required (for example, small response time or small overshoot).

Digital monitoring and control will not fundamentally alter the purpose or procedure of the experiment, but it will change the method. Instead of using an add-on sine wave generator to enable the measurement of the frequency response, the software will be programmed to sinusoidally drive the steam control valve. Rather than a manually induced step change in the steam flow to facilitate the determination of the Cohen-Coon controller settings, the steam valve will be driven automatically to produce the step change. Controller settings, too, will no longer be adjusted manually, but will be changed via the PC operator interface. Finally, instead of using a mechanical strip chart recorder, data will be stored digitally.

Chapter III

SOFTWARE SELECTION

The selection of monitoring and control software was to be the first acquisition, because available software/hardware interfaces limit hardware options. The software selection was conducted in three phases, as follows:

> 1) Phase 1: Compile a list, based on a thorough literature search, of available process monitoring and control software products.
>
> 2) Phase 2: Eliminate all products not in conformance with project guidelines:
>
> 3) Phase 3: Compare remaining software products and select the best product.

Phase 1 was conducted over the period of several months, during which time many of the major computer related publications were researched for available

software products and the product information was
solicited from the manufacturer. A list of over 25
monitoring and control software products emerged from
phase.

Phase 2 guidelines were:

1) <u>The system cost must be less than 8,000
   dollars</u>. The entire project was cost
   constrained by funds earmarked for it by the
   Department of Chemical Engineering.

2) <u>The final product must have "industrial"
   look and feel</u>. The primary purpose for this
   project was to install an industrially
   credible process control system that could
   be used for practical educational
   instruction.

3) <u>The program must run on an IBM compatible
   PC</u>. This decision was based on the
   popularity of IBM compatibles in general,
   but also the popularity of IBM compatibles
   in the control industry. The vast majority
   of commercially available control software
   is written for IBM compatibles. The current
   mix of PCs in the Department of Chemical
   Engineering (mostly IBM compatibles) also
   affected this decision. Note, however, that

the use of a PC interface is not always industrially necessary or practical. For instance, industrial applications may not require a complete PC for every control system when a PC bus and a single board are sufficient. Also, PCs are not generally designed to function in the harsh environment common to many production sites where extremes of humidity, temperature, and contaminants could severely retard PC performance.

4) The program must calculate Proportional, Integral, and Derivative (PID) control loops. This was necessary to expose students to a broad approach to control solutions, and also build in the most flexibility for the multiple adaptations for which the program is eventually intended.

5) The software and hardware must be "off-the-shelf". Generally, off-the-shelf products are readily available, are procured at lower cost, and require less programming than custom designs. They also have the benefit of being time tested, resulting in less debugging time.

6) The system must be capable of taking at
   least one data measurement per second. One
   second is the smallest increment needed for
   constructing response curves. However, it
   is preferable that the system be capable of
   an upgrade to faster scan times should an
   experiment demand it.

7) The graphics package must be easily
   manipulated. The graphics software must
   contain the flexibility to draw any control
   scheme, and the scheme must be easily read
   and understood. It is preferable that the
   package possess, or enable the creation of,
   a symbol library and permit the development
   of custom graphics.

8) The graphics package must provide color
   display. This adds industrial realism and
   sophistication, but also helps operators
   visually isolate individual components and
   functions of the control scheme.

9) The software must store and, on command,
   display historical data trends. This
   enables students to review the effect of
   particular control strategies or controller
   settings and permits comparison of the

17

effects of different controller settings and
control schemes. The ability to recall
historical data is also an aid in report
preparation.

10) <u>The system must allow data printing without
    disturbing data collection</u>. This enables
    the operator to obtain a printed copy of
    historical data for review and analysis
    while the experiment runs.

11) <u>The software must not severely constrain
    hardware options</u>. This consideration was
    necessary to ensure that a wide variety of
    hardware products were available from which
    to choose. It is also important to enhance
    the flexibility of the system should future
    expansion of the system be required.

12) <u>The computer should be free to run other
    programs without interrupting data sampling</u>.
    This is useful for purposes of designing a
    laboratory control process experiment that
    effectively utilizes the student's time.
    Since control experiments have a period of
    dead time required while data acquisition
    takes place, it is instructionally helpful
    if the students can utilize the laboratory

PCs for additional educational purposes
(such as plotting trends or running
simulations) while data collection proceeds.

Requirements 4 – 9 were not significantly limiting
criteria. These conditions turned out to be met by nearly
all of the monitoring and control packages available.
However, criteria 1, 2, 3, 10, 11, and 12 were not common
to every software product considered. So it was these 6
conditions which eliminated many of the products from
consideration.

Criteria 1 posed a problem primarily for products
which require the memory of a main-frame computer to run.
For instance, software produces marketed by Setpoint
Engineering are designed to run only on VAX systems.
Hardware requirements alone rule out these software
products.

The second criterion was subjective in nature, but
generally eliminated the software products designed for
minor applications or purely educational purposes. An
example of this is Enable Software.

The requirement for a PC interface (criteria 3)
eliminated many commercial systems which were designed
principally to operate on proprietary software and
hardware. For instance, the Taylor MOD System, marketed
by Combustion Engineering, is used by another department

19

at Kansas State University and at least one other academic institution. Though the product meets the other nine criteria, the software is compatible only with MOD Systems hardware and therefore did not meet the requirements of this project.

Requirements 8 and 10 eliminated many software programs which were intended only for industrial or educational applications where the control program could run continually in the foreground. An example of this type of educational product is UC Online, sold by the University of California at Berkeley.

The ninth condition eliminated many software packages which were designed primarily for particular hardware designs, such as products from Burr-Brown, Equinox Data, and Keithley.

Four products emerged from phase 2 analysis. They were:

1)   LT/Control;   Laboratory   Technologies Corporation;   Wilmington,   Massachusetts. LT/Control requires 640 Kb RAM. Besides PID control, it is capable of "bang-bang" (i.e. on-off) control. The software is pre-programmed to control many schemes, including feed forward and cascade, and has subroutines which can automatically change set points according to schedule or calculation. The software runs with either CGA or EGA

graphics, and permits data to be displayed graphically, numerically, and color coded according to limit status. LT/Control may be enhanced to permit background data acquisition and control, freeing the operator interfacing PC for other computational purposes. The data acquisition and analysis core is called Labtech NOTEBOOK. The system can control up to 16 I/O devices simultaneously with hardware interface through a variety of I/O cards (analog, digital, thermocouple, RTD, strain gauge, resistance, counters, frequency, or RS-232). With LT/Control, acquired data can be stored in programs such as LOTUS 1-2-3, and an alarm notation is displayed when a signal passes an alarm point. The program contains various regression calculation subroutines for fitting data, and also is capable of performing fast fourier transforms.

2)   ONSPEC;   Heuristics,   Incorporated; Sacramento, California. ONSPEC Control Software displays supervisory information and allows many tasks to be run concurrently. The software will run on IBM compatibles and some American Telephone and Telegraph, Hewlett-Packard, and Digital Equipment Corporation products, and is designed modularly to permit system "customization". Among the functions which may be acquired through the appropriate module are: plant documentation graphics, operating and target summaries, analog and digital data

21

logs, historical records, operating changes, trend graphs, process status symbols, output instrumentation, batch controls, multitasking, and recipe loading. The software supports up to six remote units and over 50 I/O interfaces. Graphic displays are created from a standardized process control character set that replaces the IBM foreign language character set. The eprom which supplies this set is plugged into the system color board. ONSPEC operates with a PASCAL compiler, and can be modularly updated with display, alarm, historical trending, and statistical quality control. Four modules of immediate interest to this project were Statistical Quality Control (SQC), Historical Data Analyzer (HDA), Modeling Kit, and Supertrends. SQC enables the alarm limits and services to be chosen. SQC control charts, frequency distribution displays, measurement data tables, and alarm actions are generated from historical data files and retrieved on or off line. HDA enhances the flexibility of the standard historical subroutine by allowing data to be stored on floppy disks for data analysis on a second PC. All trending and spreadsheet functions are identical to those contained in the standard control software. For graphics display, the Modeling Kit module permits dynamic simulation of the controlled system, via graphics which are fed from real time data.

22

Finally, the Supertrends module serves as an electronic
version of the mechanical pen. Two graphs may be
displayed per screen, with 7 "pens" per graph. The module
can operate in real time, or may be used to inspect trends
from the historical file.

3) Process Window; Taylor Industrial Software,
Limited; Edmonton, Canada. Process Window reads ladder
logic from the programmable logic controller (PLC) via a
RS-232 serial link, and stores this data in the PLC
database. The software will support multiple PLCs. The
user develops point descriptors, hardware I/O descriptors,
page titles, and rung comments. The DOCUMENTATION
software which is responsible for reading data from the
PLC produces ladder listing and descriptor table displays.
Descriptors are printed above all contacts and word
variables, and rung comments and page titles are printed.
The program displays a data table of utilized variables
and lists unused and unreferenced variables. All contacts
and word/register usage is cross referenced between ladder
logic and descriptor tables. Process Window can interface
13 different PLCs. The software displays full color
system configuration graphics, engineering units
conversion for all PLC variables, prepared static and
dynamic displays, and defines report and alarm formats.
Process Window has a capacity of 500 analog or 1000

discrete points per system. Analog inputs are converted into 12 BIT with sign or 16 BIT otherwise, and up to 5 analog inputs may be utilized per calculation. The package contains a historical trending routine which can display up tp 4 variables on the PC screen, and has an alarm routine which indicates analog high or low with percent deadband, indicate discrete conditions on or off, and display unacknowledged alarm indicators on all screens. The printing package is capable of generating 16 separately defined reports (initiated from keyboard, alarm, or time), and interfaces with LOTUS 1-2-3 for report format.

4) **The FIX; Intellution, Incorporated; Westwood, Massachusetts.** The FIX is a modularly designed software system that requires 512 Kb memory to run. Independent modules have been written to provide a variety of control functions and attendant data manipulation. The program is written to run on IBM compatibles, in "C" language, and provides concurrent program execution on a standard PC-DOS operating system. Alarm capabilities and data acquisition are part of the core base package, and of interest to this project is the CONTROL module that will execute feedforward and feedback control with PID or on-off calculation. The FIX also has a PIXEL Graphics package which will interface with either CGA or EGA, has

24

16 foreground and background colors, and a library of 128 different instrumentation symbols and a Historical Trending program that can display up to 8 data sets simultaneously in graph format. Other features are a C Data Base Access and an On-Line Spreadsheet interface which includes a LOTUS Symphony add-in. An EPSON printing program prints data. The FIX database can be accessed by user written or commercially available spreadsheets. The FIX will interface with 30 PLCs and communicates through IBM-bus hardware or serial I/O through a RS-232 or Intellution's ICM-4.

The next stage (phase 3) was comparison of the remaining programs. This was no mean feat, since there are currently no clear and universal standards which lent themselves to a systematic comparison of programs. Any software evaluation must take into consideration the three levels of control operation, i.e. routine, supervisory, and emergency. Expanding these levels of operation resulted in the following evaluation scheme:

1) How easily can the system be configured?

2) Can the process be simulated?

3) How difficult is it to make changes?

4) How long does operator training take?

5) Are product capabilities commensurate with product cost?

Each of the four products to emerge from phase 2 would have satisfied the requirements of this project, so the product selection was based on which software package most comprehensibly satisfied the phase 3 guidelines. This was The FIX. The FIX had a greater number of program enhancing options which provided a high degree of configurational flexibility. Though The FIX cost more than LT/Control or Process Window (and was about equivalent the price of ONSPEC), The FIX appeared to provide the greater value. The FIX has an "open architecture" design which permits better system configuration through the various FIX optional routines and, if needed, user written programs (which interface The FIX through the C-Data Base Access option). The superior color graphics capability of The Fix, provided by the Pixel Graphics option, permit graphic designs of all angles, sizes, and shapes (in 16 foreground and backgound colors), and has a "cut and paste" option which was unique. The FIX appeared to have a greater capacity to run foreground programs independent of background data acquisition routines. Finally, Lakeland Engineering of Kansas City, which markets The FIX in Kansas, provided very favorable delivery and service arrangements. It was for these reasons, therefore, that The FIX was recommended as the best course of action.

Chapter IV

THE FIX (FULLY INTEGRATED CONTROL SYSTEM)

The FIX (Fully Integrated Control System) by Intellution, Incorporated, is a computer program developed to run on 100% IBM compatible PCs and designed to monitor, control, generate alarms, and store process data. Through its "open architecture system", The FIX can be configured with any combination of FIX support software or user written programs.

The FIX was written in C language, and runs on a PC-DOS (version 3.0 or higher) operating system. Because PC-DOS was not modified for The FIX, other PC-DOS programs can run concurrently without system degradation. The FIX will run in 512 Kb memory, but 640 Kb will augment performance and is recommended by Intellution. The FIX also requires:

> 1) One 360 Kb double-sided, double density
>   floppy disk drive.

2) One 10 megabyte hard disk.

3) A color graphics adapter board.

4) A color monitor.

5) A parallel printer adapter.

Helpful options include a:

1) Second, backup, diskette drive.

2) Multi-function card containing battery powered time of day clock, additional RAM, and a parallel port printer.

3) Asynchronous serial adapter.

4) Parallel or serial printer.

5) Third party FIX supported cards.

6) Mouse (supported in Pixel Graphics DRAW only).

7) Adjunctive PC-DOS software packages.

8) 8087/80287 co-processor, ICM-4 card, or ARTIC card. ARTIC can only be used with the ACE (Application and Control Engine) version of The FIX.

The chosen FIX configuration for this project includes:

1) Expanded SCADA (Systems Control and Data Acquisition).

2) PIXEL Graphics.

3) CONTROL configurator.

4) Historical Trending (includes The FIX Shell).

5) C language database access.

6) On-Line spreadsheet interface (includes the Symphony add-in).

7) Epson printing program.

The two primary functions of any process monitoring and control software are to access the controlling hardware and to process pertinent data into useable form. The conceptualization of how The FIX was configured to do this is at figure 6.

The size of the computer space devoted the control scheme is set by SCADA. Expanded SCADA will support up to 3000 control blocks (which will be described in detail under Database Builder). Because some Database Builder blocks are used for control, calculational, or other functions in the control scheme, the actual number of I/O points will be considerably less than 3000.

The "master" program of The FIX is the Scanning, Alarm, and Control (SAC) program. It receives, conditions, and sends to the data base signals received from the process hardware, manages information flow, and scans blocks for alarm conditions. SAC runs as a background program in The FIX (performing the necessary control operations), and once started cannot be stopped

29

```
PROCESS DATABASE
controlled by Scan Alarm and
Control (SAC) Package
Analog/Digital Input — Analog /Digital Output
Calculation and Totalizer blocks
Trend blocks
Control blocks
```

```
THE FIX SHELL
DBB
DRAW   VIEW   CDBA
HTR   SPREADSHEETS   GEN
```

```
Up to four I/O drivers
(with ICM-4)
```
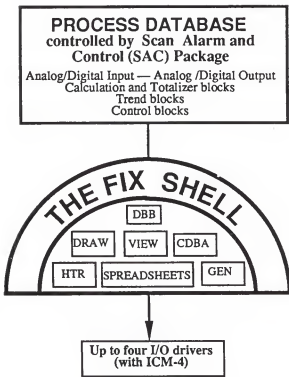
Abbreviations:

DBB   - Database Builder.

CDBA - C Language Database Access.

HTR   - Historical Trending.

GEN   - Report Generator.

Figure 6. Conceptualization of The FIX configuration.

except by system reboot. These background programs have the highest priority for execution of any FIX program, meaning that these programs are run first, and remaining time is then allocated to other programs if needed. Since these background programs are memory resident, multiple background functions can be executed simultaneously. When the system is operating normally, the process database is looked at every second. When the system experiences some overload, all blocks due for processing will be completed in sequential order, meaning there may be some perceivable system slowdown for overload conditions. Once a minute, all overloads of this nature will be reported; the proper operator response may be to increase the scan frequencies of some of the controlled loops. Approximately 60 blocks can be processed per second with an IBM XT (or compatible), and up to 3 times that for an IBM AT.

The core of The FIX control capability is three standard PC-DOS integrated software modules: Database Builder (DBB), which links the process hardware to the PC; DRAW, which links DBB to the terminal display, and VIEW, which displays process data on the PC screen (see Figure 7).

DBB is used to configure and maintain control blocks, which are linked together to define the control scheme. This is done by matching I/O hardware addresses, tag
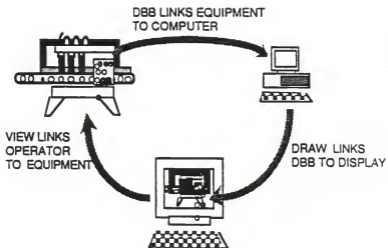
Figure 7. Relationship between DBB, DRAW, and VIEW.

names, and control parameters. DBB operates in the foreground of The FIX, and permits block manipulation while monitoring and control operations proceed. DBB blocks being edited are, however, removed from scan by SAC. Some blocks may be "chained" to other blocks, while some require specific sequencing. Primary blocks, usually associated with I/O hardware, are scanned periodically, and secondary blocks are used to perform a function based on input from a preceding block (figure 8). Blocks are identified by tags. Scanning times, which are input into the primary block, may be as fast as once per second, or as slow as once every 255 seconds. Each block has a specific function.

The second primary program is DRAW. DRAW is used to create the displays which interface the operator with the hardware, and is divided into graphic and link elements. The graphic element is used to indicate process configuration and sub-configurations. All commands are available from a screen displayed DRAW menu. A list of executable Pixel Graphics commands is in the manual. Links retrieve process values from the database. There are four types of links: text links, which display data from a DBB block; color links, which are used to define systemic color schemes; special links, which are used to retrieve system information, and display links, which

| Analog Input | Calculation | PID | Trend | Analog Output |
| Primary | Secondary | Secondary | Secondary | Primary |

Figure 8. Example of "chained" DBB blocks.

connect programs stored within The FIX. A compilation and description of all available links is also in the manual. Pixel Graphics permits precise drawing capability with either a keystroke or (preferably) a mouse. Pixel Graphics support a 640 by 350 pixel field, and requires an EGA with 256 K of additional memory. When creating screens in DRAW, it is advisable to edit screens (if needed) immediately after completion. This is due to the limited editing features supported by The FIX (most memory is devoted to control operations). It is also advisable to create a hierarchy of displays and sub-displays to simplify use.

The operator accesses The FIX database through VIEW (via links provided by DRAW). VIEW does not control the process; SAC, running in the background, executes all control functions. Operators in VIEW are prompted by menus which, based on tag names and cursor movements within the display, call up the requested links.

While DBB, DRAW, and VIEW form the core of The FIX, for the purposes of this project, the utility of the monitoring and control software is only evident when the Historical Trending, C Database Access, On-Line Spreadsheet, and the Epson Printing programs are added.

The Historical Trending package collects and saves data, and when prompted displays the data in graphical

form. Data collected is stored in PC-DOS by tag names which correspond to those used in other FIX functions. The Historical Trending software generates pixel graphic displays of data value (in the chosen engineering units) versus time. Data value, point status, and time are displayed for each point. Displays and reports do not interfere with on-line FIX data collection, alarming, or control features (as mentioned in the description of SAC). To operate Historical Trending, an EGA is required. Displays will have 640 by 200 pixel resolution, with eight available color options.

Historical Trending is subdivided into four modules: Historical Trend Assign (HTA), Historical Data Collection (HDC), and Historical Data Display (HDD), which run in the foreground of The FIX, and Historical Database Access (HDA) which runs as a background program.

The HTA routine is used to modify collection parameters for HDC, and can be used at any time (on or off-line). HTA is devised in two segments: the Trend Parameter menu and the Tag Assignment Menu. Collection rates and file sizes are established in the Trend Parameter menu. All process data collected from HDC is saved in compressed form on the hard disk. File lengths are 2, 4, 8, or 24 hours, and new files are automatically opened only after the previous file closes. Process data

can be sampled at 2 or 6 second intervals. Data in an
open file can be accessed, but only through the Display
File option which significantly increases system load.
Through the Tag Assignment menu, HTA stores data by tag
name, point status, and time of day. For each point, a
tag name must specify which DBB block data is to be
collected. A collection mode is selected to determine how
the data is handled (i.e. sample at specified scan rate,
or store only average, high, or low values within a
specified interval). A limit filter can be chosen to
"eliminate" insignificant values. This reduces storage
space since, dependent on the collection mode chosen, new
data values will be stored only if they differ from the
previous value by an amount greater than the limit chosen.
A scan rate is input which designates how often data is
processed and stored in historical data files. Possible
scan rates are 2, 6, 10, or, 30 seconds, 1, 5, 10, or 30
minutes, or 1 hour.

HDD displays and analyzes stored data. It can
display data on-line, without interrupting FIX data
collection or SAC functions. Up to 8 variables, in
appropriate engineering units, can be displayed
simultaneously, each of a different color if desired.
Points can be called up at random or as part of a data
group. Pan, slide, and zoom functions permit the display

to be adjusted as needed. Finally, data can be printed in tabular form, or accessed by spreadsheet files.

The final Historical Trending module, HDC, takes data from The FIX database, processes these values according to scan time and collection parameters, and compresses this data within constraints of the filter value chosen. HDC runs in The FIX as a background function, and is therefore uninterrupted by foreground routines.

The ability of The FIX to acquire data and maintain process monitoring and control concurrent with data referencing and manipulation is the function of The FIX Shell. The Shell serves as a multi-tasking "umbrella", prioritizing program access to the database. The Shell can support up to 16 prioritized programs simultaneously, and consists of three programs: GO.EXE, which orders the multi-tasking priorities and loads the programs; MENU.EXE, which loads and displays The FIX master menu; and the DOS Shell, which loads DOS functions. The Shell requires 5 Kb of memory. When the Shell is stopped, multi-tasked programs also stop. However, the real-time programs associated with data acquisition continue to run.

Finally, though The FIX eliminates many of the low-level programming details sometimes required to set up digital control, there may be occasion for custom designed

programs.   User written programs can   interface   the
database with the C database access module. C database
access will enable user   written programs written in C
language   to gain access to the internal FIX routines and
database.

Chapter V

MONITORING AND CONTROL HARDWARE

This chapter discusses the monitoring and control hardware equipment selected for the heat exchanger process scheme. The chosen hardware fully supports The FIX software, and meets the necessary interface requirements of The FIX monitoring and control software. All hardware purchased is a product of Gould, Inc.; the programmable logic controller (PLC) is a Gould 984-380, the thermocouple (TC) input module is a Gould B883-200, and a Gould B872-002 was selected as the analog output module. The housing for all hardware modules is a Gould model H819. All pieces of hardware will be described in the following sections.

Section V(a). THE PROGRAMMABLE LOGIC CONTROLLER

The heart of the monitoring and control hardware is the Gould 984 Model 380 programmable logic controller (PLC). The 380 is mid-range in the Gould 984 series of PLCs, all of which have common logic sets. Figure 9 portrays the 984-380 PLC.

The 984-380 PLC has up to 6 Kb words of user logic, and a 1920 K RAM (random access memory), providing a total user memory of nearly 8 Kb. The user logic and state RAM supports up to two Gould 800 series module housings, which serve as the local drop; the PLC will not support remote I/O. The maximum I/O capacity supported at the drops is 21 I/O modules of up to 256 discrete points of I/O.

The PLC is designed to withstand the temperature, humidity, and atmospheric extremes, and occurrences of shock and vibration which are common to a factory floor. It is housed in a metal frame which has a shielded backplane. The backplane permits internal communications within the housing, and protects the circuitry from electromagnetic and radio frequency interference.

A memory-protect switch (i.e. a front mounted key) precludes accidental access to the loaded program. The memory of the 984-380 PLC is backed up by a lithium battery which has a charged life of 14 days when fully

41
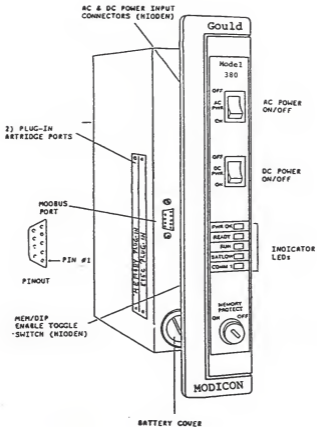
Figure 9. Schematic of the Gould 984-380 PLC.

energized.

A built-in power supply provides a minimum of 16.25 watts to the I/O. The 984-380 PLC runs on 115 or 230 volts AC or 24 volts DC. The AC power is primary, with the DC as a backup power source. The recommended AC power cable is three insulated leads of #14 gauge copper, suitable for supplying 115 or 230 volts at 5 amperes peaks for the turn-on, or at a worse case 0.4 amperes minimum. The DC power cable should be at least #18 gauge copper wire, capable of permitting 24 volts at a 24 amperes turn-on surge peak, and 1.7 continuous amperes at the worse case low voltage. Each supply source is turned on or off by a switch on the face of the 380 housing.

The host PC, in its supervisory and programming capacity, communicates with the 984-380 through its one MODBUS port. One MODBUS service scan per second may be programmed.

The 16 bit central processing unit (CPU) of the 984-380 PLC solves user logic at 5 milliseconds (ms) per thousand nodes (nominal speed). The CPU status indicators which appear on the module are:

> a) POWER OK (green LED): When on, indicates that input power and voltage output levels are ok.

43

b) READY (amber LED): Indicator is on when the
        PLC has passed power-up checks and remains
        on (in stopped and run modes) while system
        operates without error. When an error is
        detected, the LED switches off.

c) RUN (green LED): LED is on when the PLC is
        in the RUN mode and is solving logic.

d) BATTERY LOW (red LED): When on, indicates
        that a 14 day hold-up is all that remains in
        the effective life of the battery.

e) COMMUNICATION 1 (green LED): LED is on when
        the communications processor has a unit
        address and comminications are in progress.

DIP switch settings for the MODBUS port 1 are in
Table 1.

The space requirements for the 984-380 module are 12
inches clearance to the left so that the power supply
connectors can be accessed, 6 inches on the top and right
side to permit adequate air cooling, and 12 inches on the
bottom to allow for cable access.

The 984-380 PLC has executive operating software and
two levels of user logic memory which can be interchanged
by the operator. The 984 Enhanced Executive Cartridge
which was selected for this project features two primary

Table 1. DIP Switch Settings for MODBUS Port 1

1) <u>SWITCH FUNCTION</u>

<u>switch</u>	<u>down</u>	<u>up</u>
1	baud select	baud select
2	baud select	baud select
3	no parity	parity
4	odd parity	even parity
5	ASCII (7 data bits)	RTU (8 data bits)

2) <u>DIP SWITCH BAUD RATE SETTINGS</u>

<u>switch one</u>	<u>switch two</u>	<u>baud</u>
up	up	19200
down	up	9600
up	down	2400
down	down	1200

functions which are of primary use to this project: Move and EMTH.

The Move function contains a series of routines which enable the operator to manipulate data sets. These are the table to block and the block to table functions. The block to table (BLKT) function block enables large quantities of holding registers to be moved from a fixed source block to a destination block within a table in one scan whereas the block to table (TBLK) function enables the operator to move, in one scan, a large number of holding registers from a block within a table to a fixed destination block.

The BLKT function is shown in figure 10. The top node is the source register, which indicates the beginning of the fixed block of registers which will be moved. As indicated by the figure, only 4TTTT references are permitted for this node. The middle node shows the first register of the destination table. The entry 4MMMM is the first register of the destination table and defines the pointer register. BLKT uses moveable pointers to indicate the destination registers. The destination table actually starts at 4MMMM + 1 and acts like a series of blocks which hold data transferred from the source blocks. Pointer values are incremented to define where the source block is to be moved to; the source block is moved to the register

46

```
                  _____
                 |               |
  activate       |    source     |   move successful
 ───────────────►|    4TTTT      |  ────────────────►
                 |               |
                 |               |
                 |               |
  hold pointer   | destination pointer |  error/move
 ───────────────►|    4MMMM      |  ────────────────►
                 |               |     not possible
                 |               |
                 |               |
  reset pointer  |  BLKT length  |   (not used)
 ───────────────►|    NNN        |  ────────────────►
                 |               |
                 |_____|
```

Figure 10. Block to table (BLKT) function block.

47

1 unit above the pointer value:

start register = [(4MMMM + 1) + (pointer value *
length)]

and continues until the end of the BLKT:

end register = [start register + (block length –
1)]

The bottom node, then, indicates the legth of the block of registers which is to be moved, which also fixes the length of the source and destination blocks. This length can vary from 1 to 100. A BLKT may be executed to any position in the holding register provided that the entire desination block is within the range provided for.

There are three inputs to BLKT; *activate*, *hold pointer*, and *reset pointer*. The top input (refer back to figure 10) is *activate*, and permits a move to be made provided that the switch is 'on' and there is an adequate range of destination registers. The range of the destination registers is verified based on the beginning and end registers of the destination blocks, and the pointer value. *Hold pointer* is the middle input to the BLKT function. When *hold pointer* is 'on', the value of

48

the pointer does not change after each move, but, rather, remains the same. If *hold pointer* is 'off', the pointer value increments by one after every move. The last (bottom) input is *reset pointer*. When this is 'on', the destination register is reset to zero before a move is made, enabling the next move to be made to the block which immediately follows the pointer register; the destination pointer can be reset even if a move is not made. *Reset pointer* operates independent of the top and middle inputs to BLKT.

There are two outputs to BLKT (the third output is not used): *move successful* and *error/move not possible*. *Move successful* is the top output and shows 'on' when *activate* input is 'on' and a successful move has been made. The middle output is *error/move not possible*, and goes 'on' when *activate* input is 'on' a move is not possible, due to the block length exceeding the allocated range.

Figure 11 shows the table to block (TBLK) function. The top node is the source register, in which 4TTTT indicates the start of the table of registers from which the blocks of registers are to be moved from. The source table extends from:

$$\text{start register} = [4TTTT + (\text{pointer value} *$$

```
          _____
         |                   |
activate |     source        | move successful
-------->|     4TTTT         |-------->
         |                   |
         |                   |
         |                   |
hold pointer | destination pointer | error/move
-------->|     4MMMM         |-------->
         |                   | not possible
         |                   |
         |                   |
reset pointer| TBLK length   | (not used)
-------->|     NNN           |-------->
         |                   |
          _____
```

Figure 11. Table to block (TBLK) function block.

length)]

to:

ending register = [start register calculated +
(length - 1)].

The middle node is the destination pointer and designates
the block of registers to be moved from the table of
registers, where 4MMMM indicates the source block to be
moved. The destination of the moved block begins at 4MMMM
+ 1. Unlike BLKM, where the pointer indicated the fixed
source indicator, a movable pointer is used in TBLK to
designate the source block. The block destination and
length are constant, though the source block can be
changed with appropriate control inputs. The destination
area of the block can be within the overall table, or
precede it within the 4X register space. The bottom node
designates the length of the block of registers moved
(which can be from 1 to 100 registers). This defines both
the source and the destination block lengths. Table to
block moves can be executed for any series of registers
provided that the entire block is in the configured range.

As indicated by figure 11, there are three inputs
into the TBLK function. The top input is *activate*, and

when it is 'on', range checks of the block's beginning and end are made, based on the pointer's value. If a move can be satifactorily made, the move is executed. The middle input, *hold pointer*, determines whether the pointer value will change with every move. An input of 'on' holds the pointer value at a constant value, meaning that the same block of values is moved (unless the pointer is in some other manner incremented). An input of 'off' automatically increments the pointer value with each move. Finally, the bottom input, *reset pointer*, is independent of the top two inputs, and resets the pointer register to zero, even when a move is not made.

The three outputs of TBLK are really two: the bottom output is not used. The top output, *move successful*, shows 'on' when a move has been successfully executed. *Error/move not possible* is the middle output and goes 'on' when the *activate* input is 'on', but a move is not possible due to range restrictions.

The second function provided by the chosen executive cartridge is EMTH. EMTH functions provide the arithmetical computation capabilities of the Gould 984-380 PLC. It includes double precision integer add, subtract, multiply, divide, and square root, process square root, logarithm, and antilogarithm, and 30 floating point

capabilities. The form of the EMTH block is shown in figure 12.

The top node of the EMTH function requires either 3X or 4X registers, depending on the operation performed (see table 2). The middle node needs 2, 4, or 6 registers, also depending on the function executed, and only 4X references are permitted. Finally, the bottom node permits the selection of the desired function, of which there are 38. Table 2 indicates all inputs required and outputs provided for by the various arithmetical operations performed.

Section V(b). THE THERMOCOUPLE INPUT MODULE

The thermocouple input module chosen for this project was the Gould B883-200 Thermocouple (TC) Input Module. It is capable of handling up to ten thermocouple inputs (any combination or type B, E, J, K, N, R, S, T: see table 3), and relays real time temperature measurement data to the PLC. Features include reference junction temperature compensation, open circuit detection, signal linearization, automatic self calibration, and internal diagnostic checks (which are conducted with every scan of the module). Signals relayed to the PLC can be in degrees

53
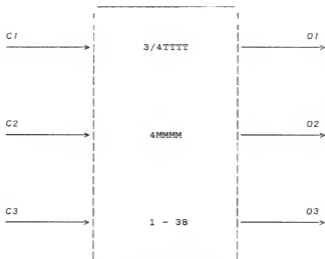
Figure 12.   The EMTH function block.

Table 2. The Enhanced Executive Cartridge EMTH Functions

Function	Code	Registers	Inputs	Outputs
Integer Addition	1	4X	C1	O1,O2
Integer Subtraction	2	4X	C1	O1,O2,O3
Integer Multiplication	3	4X	C1	O1,O2
Integer Division	4	4X	C1,C2	O1,O2,O3
Square Root	5	3X,4X	C1	O1,O2
Process Square Root	6	3X,4X	C1	O1,O2
Logarithm	7	3X,4X	C1	O1,O2
Antilogarithm	8	3X,4X	C1	O1,O2
Integer to FP Conversion	9	4X	C1	O1
Integer Plus FP	10	4X	C1	O1
Integer Minus FP	11	4X	C1	O1
Integer Times FP	12	4X	C1	O1
Integer Divided by FP	13	4X	C1	O1
FP Minus Integer	14	4X	C1	O1
FP Divided by Integer	15	4X	C1	O1
Integer/FP Comparison	16	4X	C1	O1
FP to Integer Conversion	17	4X	C1	O1,O3
FP Addition	18	4X	C1	O1
FP Subtraction	19	4X	C1	O1
FP Multiplication	20	4X	C1	O1
FP Division	21	4X	C1	O1
FP Comparison	22	4X	C1	O1,O2,O3
FP Square Root	23	4X	C1	O1
Change Sign	24	4X	C1	O1
Load Value of Pi	25	4X	C1	O1
Sine in Radians	26	4X	C1	O1
Cosine in Radians	27	4X	C1	O1
Tangent in Radians	28	4X	C1	O1
Arcsine in Radians	29	4X	C1	O1
Arccosine in Radians	30	4X	C1	O1
Arctangent in Radians	31	4X	C1	O1
Radians in Degrees	32	4X	C1	O1
Degrees in Radians	33	4X	C1	O1
FP Raised to Integer Power	34	4X	C1	O1
Exponential Function	35	4X	C1	O1
Natural Logarithm	36	4X	C1	O1
Common Logarithm	37	4X	C1	O1
Report Errors	38	4X	C1	O1,O2

where: FP = floating point

Table 3. Thermocouples Supported by the B883-200 Module

Thermocouple	Type	Application Range	Accuracy
Platinum/6% Rhodium– Platinum/30% Rhodium	B	200 to 1820 °C 392 to 3308 °F 0.1782 to 13.81 mV	±2.0 °C ±3.6 °F ±0.014 mV
Chromel–Constantan	E	−240 to 1000 °C −400 to 1832 °F −9.604 to 76.36 mV	±1.5 °C ±2.7 °F ±0.104 mV
Iron–Constantan	J	−100 to 1200 °C −148 to 2192 °F −4.632 to 69.54 mV	±1.6 °C ±2.9 °F ±0.092 mV
Chromel–Alumel	K	−100 to 1372 °C −148 to 2501 °F −3.553 to 54.87 mV	±1.7 °C ±3.0 °F ±0.066 mV
Nicrosil–Nisil	N	−224 to 1299 °C −371 to 2370 °F −4.248 to 47.47 mV	±1.7 °C ±3.0 °F ±0.057 mV
Platinum– Platinum/13% Rhodium	R	−50 to 1768 °C −58 to 3214 °F −0.226 to 21.11 mV	±2.3 °C ±4.1 °F ±0.027 mV
Platinum– Platinum/10% Rhodium	S	−50 to 1768 °C −58 to 3214 °F −0.2357 to 18.70 mV	±2.0 °C ±3.8 °F ±0.022 mV
Copper–Constantan	T	−270 to 400 °C −400 to 752 °F −6.105 to 9.286 mV	±1.6 °C ±2.9 °F ±0.039 mV

Celsius or Farenheit, or in millivolts. Some of the
primary functional components of the TC module are the
signal conditioning circuitry, input signal filters, an
input signal multiplexer, a high resolution dual slope A/D
converter, a microcomputer, and the I/O bus interface
hardware. Each input channel has voltage protection, a
single pole passive filter, and an open TC detection
circuit.

When the 883-200 TC module interfaces with a 984-380
PLC, as in this configuration, a S901 processor with a
J810/812 adapter, or a S908 processor with a J890/892
adapter is required. Figure 13 shows a generalization of
the interfaces between the module, the PLC, and the
inputs, and the outputs.

When the power is turned on to the module, a
diagnostics check is executed to ensure all circuitry is
properly functional. During this power-up and diagnostics
check, the PLC will show zeroes for any attempted TC
readings, and actual data values will only be transmitted
to the PLC when the TC module is switched to *RUN* mode.

The software contained within the TC module controls
the OURBUS interface and any A/D conversions which are
required. There are four conditions under which the TC
module may operate: *RUN*, *IDLE*, *CONFIGURATION*, and
MAINTENANCE. Generally, the PLC is responsible for

```
                  ┌──────────────────┐
  ----------►│                  │
                  │                  │          Power
  ----------►│                  │
                  │                  │     ◄------------
  ----------►│                  │
                  │                  │
  ----------►│                  │
                  │     B883-200     │
  ----------►│                  │
                  │   Thermocouple   │
  Up to 10     │                  │
                  │      Input       │
  Input TCs    │                  │
                  │      Module      │
  ----------►│                  │
                  │                  │     ◄------------
  ----------►│                  │
                  │                  │        I/O Bus
  ----------►│                  │
                  │                  │     ------------►
  ----------►│                  │
                  │                  │
  ----------►│                  │
                  └────────┬─────────┘
                           ⊥
                           =
                    Active Light
```
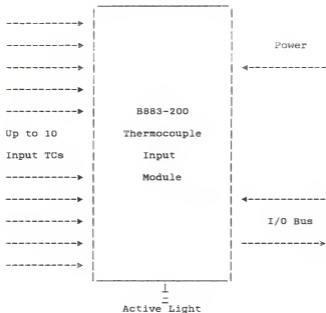
Figure 13.   Generalization of the B883-200 interfaces.

ordering the mode of operation, however, when the diagnostic routine detects an error, all entries are reset and the TC returns to *RUN* mode.

In *RUN* mode, the TC module is in its normal operating condition. Generally there are two simultaneous operations being performed in *RUN* mode; the TC module is reading the analog inputs from the thermocouples, and the PLC is asking the TC module for these readings. Both operations are on-going and independent; a request for data from the PLC does not prompt the TC module to execute a thermocouple scan.

The *IDLE* mode of the B883-200 indicates that all measurements have been temporarily suspended. The only ongoing operations are the ongoing system diagnostic checks.

Channel parameters may be changed in the *CONFIGURATION* mode of operation, of which there are two: the thermocouple type and the engineering units desired. The 10 channels are simply identified with a designation between 1 and 10, with 0 left as the identifier for the TC module itself. The only alterable parameter for the module is the input power frequency, be it 50 or 60 hertz. Any entry to the *CONFIGURATION* mode automatically clears all previously set parameters.

The final state of operation is *MAINTENANCE*. This mode provides access to the A/D calibration constants, but generally, these require no alteration; the constants have been preset at the factory and infrequently require adjusting.

At startup of the B883-200 TC input module, a complete diagnostics check is made. This check is run continuously throughout operation with every communication with the PLC or an input channel. An error detection results in an error code display. The *active* designator is on the face of the module; when the indicator is lit, the module's diagnostic routine has not detected any systemic errors.

The B883-200 can process one new input every 100 ms. The module actually is performing a continual scan of the input channels, but the use of noise filtering reduces the frequency of channel updates. Update time of all input channels is dependent on the number of active input channels. The length of each complete update cycle is:

complete update = active channels * 100 ms.

The TC module automatically computes and stores degrees Celsius, Farenheit, and millivolts, for access by the PLC. The maximum command response time is 7 ms.

The PLC initiates all communications between itself and the B883-200, with a command word followed by data words, and the module will respond to the PLC by echoing the command word preceding the requested data. All communications between the PLC and the module are in three 16-bit words. Three consecutive input registers (30XXX, 30XXX + 1, 30XXX + 2) and three consecutive output registers (40XXX, 40XXX + 1, 40XXX + 2) must be utilized. The first output register (command register) is 40XXX, and holds the command code. The following two output registers are the data registers. Conversely, the first input register, 30XXX, is the echo of the command register, and the following two registers are the data registers. The command codes must be a four digit number between 0000 and 9999 (see table 4). All command words share the same address space.

Generally, thermocouples are employed in high noise settings, which can pose a particular problem for the TC module since most thermocouple inputs are only in the range of a few microvolts per degree Celsius. This can only be compensated for to a limited amount by the internal filtering capacity of the B883-200. Also, the B883-200 has floating inputs, which permits the I/O to be referenced to either a ground or some other potential. This allows the module to accept high common mode voltages

Table 4. The B883-200 TC Input Module Command Code List

No Operation
    0000 - no operation (NOP)
System Configuration
    0010 - read TC module identification
    0019 - read TC module PROM combination number
    0020 - read TC module status
Channel Configuration Related
    0220 - set channel parameter
    0221 - read channel parameters
    0290 - enter configuration mode
    0291 - enter configuration mode
Start/Stop
    0300 - exit idle
    0301 - enter idle
Maintenance Related
    0904 - enter maintenance mode
    0905 - exit maintenance mode
    11XX - read physical channel value (in floating point
           format)
    2YXX - set channel compensation values (in floating
           point format)
    12XX - read M values (calibration constants)
    13XX - read B values (calibration constants)
Read Data
    10XX - read channel value

while maintaining an adequate filter of common mode noise. Figure 14 shows an example of the setup for a grounded thermocouple, and figure 15 depicts the typical setup for an isolated thermocouple.

Section V(c). THE ANALOG OUTPUT MODULE

The mechanism by which the digital outputs from the PLC were converted to their intended analog counterparts is via an analog output module. The particular equipment selected for this project was the Gould B872-002 Analog Output Module. The B872-002 is the simplest module to operate in the entire hardware setup, principally because the primary function of this equipment is simply to convert digital data relayed to it from the 984-380 PLC into useable electrical currents.

The B872-002 can manage up to four outputs simultaneously, and proportionately converts digitally relayed numerical values (ranging from 0000 to 4096) into a current range of 4 to 20 milli-amperes (mA). The module can only be mounted on the chosen 800 series housing (described later), which also provides the means of grounding the module. As with the B883-200 TC input module, the B872-002 analog output module requires either
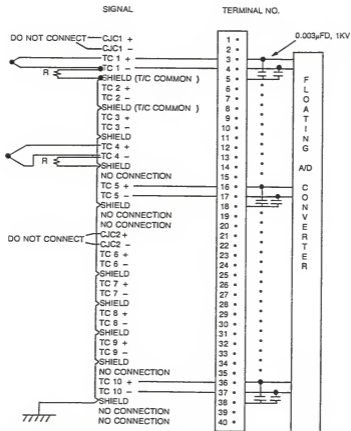
Figure 14. Typical grounded thermocouple interconnections.

Figure 15. Typical isolated thermocouple interconnections.

* R>1MΩ & R<100MΩ, ¼W, CARBON FILM.

a J810 remote I/O adapter or a J812 ASCII adapter module when used with the 984 series of Gould PLCs. Calibration of the module is recommended at 12 month intervals for the conditions anticipated for its use (i.e. below 40 °C). Table 5 lists some electrical characteristics of the module.

On the back of the B872-002 are two sets of DIP switches, labeled A and B. The switches labeled B are factory set to the off position, and should not be adjusted. However, set A is to be altered dependent upon whether the analog output of the indicated channel is to remain at the last value or reset to 0.0 mA as a result of a system shut down or standby situation. The four switches in set A correspond to the four output channels, and operate independently. The switch settings are portrayed in figure 16.

Finally, the face of the module has two indicator lights, which provide the operator a quick read of the module status (see table 6). The module is in the *fault* state when the *active* light is off, and all outputs are at 0.0 mA. Any *fault* condition, when remedied, can only be reset by turning power to the module off, then back on.

Table 5. Electrical Characteristics of the B872-002

current range	4 to 20 mA
maximum loop supply voltage	60 volts DC
minimum voltage drop	loop voltage - 7 V
	at 20 mA
maximum voltage drop	loop voltage -30 V
	at 20 mA
response time (to within	
0.025% of full)	30 ms
resolution	1 of 4096 counts
range of monitoring voltage	1 to 5 V DC
minimum load of	
monitoring voltage	1 megaohm
impedance of monitoring	
voltage (typical)	220 ohms
total output errors at 25 °C	±0.025%

resistance is the total resistance in the
field side circuit

loop voltage is voltage of the loop supply,
not to exceed 60 V DC

ON

(ALL-SW)- NORMAL-

B

OFF

OUTPUT
4 3 2 1

ON

RTZ- HOLD-

A

OFF
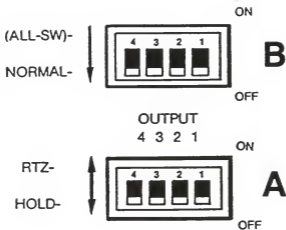
When the PLC stops:

a) HOLD keeps output at previous value

b) RTZ returns output to zero

Figure 16. DIP switch settig labels on the B872-002.

Table 6. Status Indicators on the B872-002 Module

Indicator	State	Condition
active	ON	PLC is running and properly communicating with module
	BLINKING	Module is failing internal diagnostics test
	OFF	PLC is in standby or has stopped, <u>or</u> communication error
fault	OFF	PLC and module are properly communicating
	ON	Communication error between PLC and module

Section V(d). THE MODULE HOUSINGS

A Gould model H819 housing was selected to provide the link between the hardware components. Beyond storing the modules and enabling the electrical connections among the modules, the H819 also provides a conduit to ground and electromagnetic and radio frequency interference. The housing can be configured to permit rack or panel mounting.

Chapter VI

PRELIMINARY PROGRAMMING

Configurating the control hardware begins with an understanding of the Gould system of Ladder Logic. The Gould Executive software is used in conjunction with the 984 Controller Ladder Lister to establish the data control system.

Function keys on the keyboard to the PC are used to access the controller logic data base. The keys and their functions are shown in figure 17. Numerical entries must be entered with the keys at the top of the keyboard, and not with the numerical keypad.

The first level to be accessed in Ladder Lister is the Reset Level. There are four (4) labels which will appear:

        1. Printer Type

        2. Set Date

        3. Return to Gould Exec

Table 7. PC function key definition in Ladder Lister

Function Key	Action
Del	Delete Node
Esc	Clear Error
Ins	Start Next
Pg Dn	Get Next
Pg Up	Get Previous
Alt C	Change Screen
Alt N	Delete Network
Alt P	Print
Alt F1	Erase
Alt F2	Get
Alt F3	Change Node
Alt F4	Search
Alt F5	Continue Search
Alt F6	Exit
Alt F7	Trace
Alt F8	Retrace
Alt F9	Reset
Alt F10	Clear Assembly Register

4. Attach Unit # (number).

To establish communication between the 984 and the PC, type the unit number in the AR displayed on the screen, and press F8. The system is now configured to the 984 specified. The resultant print-out for the configuration used for this project is at figure 18. Once the PLC has been properly configured, the screen displays a prompt for

1. Printer Type (function key F1)

2. Set Date (function key F2)

3. Release 984 (function key F8).

The printer type is set with the F1 key. The F2 (Set Date) key is used to specify the current date. F8 is used to execute Release 984, which detaches the 984 from the PC when the ladder listing is completed.

Once the 984 PLC and the PC have been is properly connected, proceed to the EXIT LEVEL by pressing Alt F6. The screen will then display:

1. CONTROLLER OPERATIONS (function key F1)

2. LADDER LISTER (function key F2)

3. EXAMINE MEMORY (function key F3).

When CONTROLLER OPERATIONS is accessed the following functions appear:

1. STOP 984 (function key F1)

2. START 984 (function key F2)

3. CLEAR 984 MEMORY (function key F3)

CONTROLLER: EXEC #: 02066

```
                 NETWORKS:             00001-00007
                 COILS:                NONE
                 DISCRETE INPUTS:      NONE
                 INPUT REGS:           NONE
                 HOLDING REGS:         NONE
                 NUMBER OF MESSAGES:   NONE
```

984 CONFIGURATION (16-BIT LOGIC WORD)

```
         TOTAL LOGIC WORDS : 05205    TOTAL MESSAGE WORDS :  00000

         TOTAL MESSAGES :    0000     HIGHEST MESSAGE USED : 0000

COILS:        0256                              DISCRETE INPUTS:  0032

INPUT REGS:   0016                              HOLDING REGS:     1900
```

SEG	USED	NETWORKS	SEG	USED	NETWORKS
0001	00144	0001 - 0007	0002	00001	NONE
0003	00001	NONE	0004	00001	NONE

Figure 17. PLC configuration.

74

4. RUN DEBUG (function fey F5)

5. PREVIOUS MENU (function key F8).

Once into each sub-program, the F7 key is used to PROCEED, and F8 is pressed to CANCEL.

LADDER LISTER is used to generate the formatted listings of the PLC logic data base. Once into the LADDER LISTER routine, the following is displayed on the screen:

1. Commence (F1)

2. List Ref/Networks (F2)

3. No Ref/Networks (F3)

4. List ASCII (F4)

5. No ASCII (F5)

6. Enter Title (F7)

7. Enter Control ID (F8).

First, give the ladder lister a title, with the F7 entry. The title can be a 60 character string. Then give the controller a six character identification with the F8 utility.

The F2 function key allows the user to designate which PLC data base references or networks are to be included in the Ladder List. Options available under List Ref/Networks are:

1. List Network (F1)

2. List Reference (F2)

3. List All (F3)

4. Xref/No Xref (F4)

5. Format/No Format (F5)

6. Set Parameters (F7)

7. Previous Menu (F8).

These utilities enable the operator to specify the particular segments of the PLC data base which are to be printed.

Use of F4 enables the operator to display pertinent ASCII labels, consistent with the references or networks selected for display. The sub-menu permits the operator to, again, designate the scope of the labels displayed. Conversely, the selection of F5 (No ASCII) indicated that no ASCII messages are desired on the ladder lister display.

Finally, to print out the ladder list consistent with parameters chosen, select COMMENCE (F1).

Selection of the EXAMINE MEMORY function allows the operator to review memory locations in the PLC.

The system was now ready to be programmed to conduct the type of control desired. For the purposes of this report, the first step was to get the TC Input Module ready to properly transmit the temperature data in a manner in which it could be manipulated to produce the desired output (i.e. the correct temperature units and

values). (Throughout this discussion on preliminary programming of the TC Module, it will be necessary to refer to table 4 (page 62) which describes the codes which are unique to the Module.

The Module configuration is established as follows from RUN:

        1. Enter idle mode (code 301).

        2. Enter configuration mode and set to "zero" state (code 290).

        3. Set channel parameters and load new configuration data (code 220).

        4. Exit configuration mode (code 291).

From which the module proceeds into power-up diagnostics.

At figure 19 is a depiction of the first portion of the program for the TC Module. When the module is powered up to run, it enters the idle mode, as designated by code 301. Assembly register (AR) 30001 transmits this command to the Module. This causes the TC Module to perform system diagnostics and prepare for the configuration mode. Once the diagnostics are completed (about 7 milli-seconds), the TC Module is commanded to enter the configuration mode (code 290), and designate AR 40005 as the command register for the output. (Recall from the above discussion on how the TC Module communicates with the PLC, designation of AR 30001 as the command code

77

register for the input effectively tabs AR 30002 and 30003
as data input registers, and designating AR 40005 as the
command code output register [simply the echo of the input
command] effectively assigns AR 40006 and 40007 as the
output data registers.) (Note that the registers for the
input could have been any three consecutive series of
numbers, with the provision that they be in the 30000's,
and the output registers could have been any three
consecutive series of numbers, provided that they were in
the 40000's; the assembly registers selected are arbitrary
with the proviso that they be unused for any other
purpose.)

With the idle mode diagnostics completed, the next
step was to set the channel parameters and load new
configuration data (code 290). This is depicted at figure
20. The channel number was established as "channel 1".
The parameter number was programmed as follows: the 8
indicates that a type "T" thermocouple is used for data
acquisition, and the 2 (following the 8) converts the data
into double precision C (these codes are contained and
described in the TC Module operating manual). Once
completed, the read configuration code (220) was returned
in AR 40005.

Finally, the configuration mode is exited (code 291).
Figure 21 shows how this is programmed. Code 300 exits

```
+------------------------------------------------------------------------( )
¦                                                                        00100
¦--I¨I-+-----+-
¦ 00100 ¦00301
+        +   +-
¦        ¦00000
+        +ADD +-
¦        +40005
++----+-        ++----+-
¦¦30001         ¦¦00290
++   +------++   +-
¦¦00301          ¦00000
++SUB +-         +ADD +-
+41000           +40005
```

```
           REF
     COIL= TYPE   CROSS REFERENCE BY NETWORKS

     00100 ]P[    00001
```

Figure 19. TC Module configuration: enter idle mode.

```
+               +-----+-----+
¦               ¦     ¦     ¦
++----+-        ++---+++----++-----+-
¦¦30001         ¦¦00001 ¦00082 ¦00220
++   +------++   +-+   +-+  +-   +-
¦¦00290         ¦00000 ¦00000 ¦00000
++SUB +-        +ADD +-+ADD +-+ADD +-
¦+41000         +40006 +40007 +40005
+
¦
+
*
```

```
        REF
   COIL= TYPE   CROSS REFERENCE BY NETWORKS
```

Figure 19. TC Module configuration: enter configuration

         mode.

the idle mode which the module was in during configuration, and starts the controller.

The PLC was then programmed to accept data from the TC Module. Figure 22 shows how this was programmed. Again, the starting point (as previously designated) was output AR 40005. To read channel 1, the appropriate command code had to be entered into the command register (40005). This was accomplished by putting 1001 ("read channel value [1000] at 1 [01]", i.e. 1000 + 01 = 1001) into AR 40005 by adding 999 (the largest constant available for programming, and a number which does not correspond to a commandnd 2 (also a number not corresponding to a command code). Then, the program was developed so that once channel 1 was read, the data values were transferred from the input registers (30002 and 30003) to the proper holding registers (in the 40000 series). For a double precision value (as was programmed previously) the first register will echo the command, the second will contain the sign with the whole number, and the last (third) register will hold the fraction. A double precision number can be any value within the range of $\pm$ 9999.9999. For example, suppose the data value input was +21.9, and that this data value was input by registers 30001 - 30003. Register 30001 would contain the command code, register 30002 would have the whole numbers and the

NETWORK: 00003   SEGMENT: 001

```
+ +----+-      + +----+-
| |30001       | |00291
+ +   +------+ +   +-
| |00220        |00000
+ +SUB +-      +ADD +-
| |41000       +40005
+ +----+-
| |30001
+ +   +------+----+-
| |00000        |00300
+ +SUB +-      +   +-
| |41000        |00000
*              +ADD +-
               +40005
```

            REF
     COIL# TYPE   CROSS REFERENCE BY NETWORKS

Figure 20.   TC Module configuration: set channel

        parameters and exit configuration mode.

        NETWORK: 00004   SEGMENT: 001

```
+ +----+-
| |40005
+ +   +------+----+-
| |00300        |00999
+ +SUB +-      +   +-
| |40002        |00002
+              +ADD +-
|              +40005     +------+------+------+------+
| +----+-      +------+---+++----+++----+++----+++
| |30001       |              |  .  |    |      |      |
+ +   +------+              |30002 |30003 |40008 |40011 |40010
| |41000       |              *   +-+  +-+  +-+  +-+  +-
+ +SUB +-      |            |00000 |00010 |00010 |00010 |40012
  +41000       |            +ADD +-+ADD +-+MUL +-+MUL +-+ADD +-
               |            +40000 +40011 +40009 +40012 +40014
```

            REF
     COIL# TYPE   CROSS REFERENCE BY NETWORKS
Figure 21.   Acquire and manipulate data.

                    81

sign from the data input (+21), and register 30003 would have the fractional portion (9). The data values in the input registers would be transferred to a holding register (30002 transferred to 40008 and 30003 transferred to 40011). The data in these holding registers would be multiplied by 10 (a manipulation required to properly align the data for display; will be discussed below). Since multiplication of two registers results in the product being parked in two consecutive registers (the product could be as large as 8 digits), the following is the result when data values in 40008 and 40011 were multiplied by 10: in 40009, 0000; in 40010, 0021; in 40012, 0000; in 40013, 9000. Registers 40010 and 40012 were added together so that the final number was stored in one register (40014), with the condition that the final digit in the AR represented tenths of a degree. Properly programming the DRAW routine inserted a decimal before the last digit in the register when the register value was displayed. This completed the data manipulation required to get a temperature reading accurate to the tenth.

The next series of programming was established to read and store the data values from the TC Module. Data values are read into the holding registers, but only one in every hundred is necessary for control purposes (due to

the scanning rate of the TC Module [once every 100 milli-second, when one channel was being scanned] as described above, and the capability of the PID software to execute only one computation every tenth of a second. Therefore, every hundredth data value would be available at .1 second intervals, which was precisely the rate at which data values could be handled by the PID loop. The way these values were retrieved is at figure 23. Relay coil 99 was the signal to start sampling. Every time that 99 came up, the first action was to zero out the 100 holding registers for the TC Module data values. Therefore, the series of registers starting at 40202 (and having values of zero) was transferred to the 100 registers starting at 40101. The registers between 40202 and 40301 (i.e. 100 holding registers) were zeroed out one per scan by the R T (register to table) move logic function. The transfer of zeroes from AR 40202-40301 to 40101-42000 was instantaneously executed with a BLKM (block move) command.

Figure 24 shows the first-in (FIN) routine. This function copies data from one register to another. When the top node received power, the data values were transferred to the next available destination register: in this situation, the 100 registers (i.e. o to 99, hence the program value 00099) starting with 40101 (40101 being the register succeeding the programmed 40100).

NETWORK: 00005   SEGMENT: 001

```
+--I^I----------------+----+-+----+-
: 00099              :40202 :40100
+------+----+        +    +-+    +-
:      :40017        :40101 :40100
+      +    +-        +BLKM+-+SUB +-
:      :40017        +00099 +40100
+      +SUB +-
:      +40017
+--I\I--+----+-
: 00099 :40017
+      +    +-
:      :40201
*--I^I--+R->T+-
  00099 +00099
```

           REF
      COIL# TYPE   CROSS REFERENCE BY NETWORKS

Figure 22. Storing data values.

NETWORK: 00006   SEGMENT: 001

```
+--I^I--+----+-
: 00098 :40014
+      +    +------------------------------------------------( )
:      :40100                                                 00097
+      + FIN+-
:      +00099
+
:
+
:
+
:
*
```

           REF
      COIL# TYPE   CROSS REFERENCE BY NETWORKS

      00097]\[00007

Figure 23. The first-in function.

84

Finally, figure 25 depicts the timer function. AR 40015 is programmed to indicate which data value will be used for the PID loop, and stores the running time (note that the time increment is a hundredth of a second).

NETWORK: 00007   SEGMENT: 001

```
+--I\I----I I-+--------------------------------------------------( )
¦ 00097  00099¦                                                     00099
+            +¦+----+-------------------------------------------------( )
¦            ¦40015                                                 00098
+--I\I---------+T.01+-
¦ 00098        +40016
¦
¦
+
¦
+
¦
*
```

```
        REF
  COIL# TYPE   CROSS REFERENCE BY NETWORKS

  00099 } [    00007
        }\[   00005
        ]P[   00005
  00098 }\[   00007
        ]P[   00006
```

        ***** END OF LISTING *****

Figure 24. The timer function.

Chapter VII

WORK TO BE COMPLETED

There remain four major areas of work before this process monitoring and control scheme can be properly utilized as an educational tool in the chemical engineering laboratory.

First, the DRAW routine must be thoroughly studied, and all of its intricacies mastered. What will be required in this regard is a complete drawing of the controlled hardware, together with all pertinent data blocks, so that the operator will be able to visualize the system being controlled, and to discern the operating conditions within the process.

Second, the historical trending package will have to be integrated into the operation in a manner which will be useful to the student. Historical trending capabilities of The FIX were looked at, but not in sufficient detail to permit a report at this time.

Third, the exercise methodology must be worked out in a format that can be readily used for the intended purpose of this project: to develop a laboratory experiment in process monitoring and control for undergraduate chemical engineering students.

Finally, a sufficient data base must be developed to enable future laboratory instructors to properly evaluate and critique student work.

AN INVESTIGATION OF THE

USE OF A DIGITAL PROCESS MONITORING AND CONTROL SYSTEM

IN AN UNDERGRADUATE CHEMICAL ENGINEERING LABORATORY

by

BERNARD THEODORE GIEFER, JR.

B.S., Purdue University, 1983

AN ABSTRACT OF A REPORT

submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

CHEMICAL ENGINEERING

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1989

ABSTRACT

The use of high-speed, digitally automated process monitoring and control equipment is becoming commonplace in the chemical process industries. In the past, this required expensive software and large minicomputers. Thus, the vast majority of universities could not afford to provide Chemical Engineering students with the opportunity of studying and using digital control systems similar to that of industry. This has changed dramatically as process control software has decreased in cost, increased in performance, and become available for use with inexpensive personal computers.

In this report, a commercial process monitoring and control product is investigated in terms of it's educational potential for Chemical Engineering undergraduates. The control system was installed on a small, tube-in-shell heat exchanger which has been used for an experiment in a senior laboratory class. The selection of the control system is discussed and it's installation and initial testing is reviewed. Plans for incorporation into the undergraduate laboratory are reviewed.