HMI/Control Software

System Overview:

HMI/Control is an application framework for building **HMI** and Control applications for industrial machinery in **Visual Basic**. It consists of custom controls and modules which make it easy to configure the control system by simply dropping controls on VB forms, and changing properties of those controls to define how the HMI/Control application will interact with the control hardware on the machine. The custom controls included in the system handle the following types of functions: Manual, Automatic. Time, Temp, Adjust, Setup, and Reporting. The control system is made up of a computer, a PLC, and a servo sub system. HMI/Control's own proprietary HMI (Human Machine Interface) software is used in the computer for handling the user interface, and controlling some functions on the machine. Most machine control I/O (Input/Output) is done through the **PLC**. This PLC can be a Siemens, Allen Bradley, Omron, or GE PLC. The software in the computer specifies the memory locations and timers used to interact with the logic running in the PLC. The system also supports **servo** control systems by **Delta Tau** and **Indramat**. These servo systems are used to control the indexing of the chains on these machines. Servo systems can also be used for presses and other functions on the machines using very basic control parameters.

The computer used should be an industrial computer, and can include an analog card and Encoder or Motion Control Card. The analog card used is a Metrabyte DAS8 or a clone of that card (AIO8 for example). The software used for this system is a program named **HMICTRL.EXE**, and is a Windows application providing a user interface to the operator of the machine.

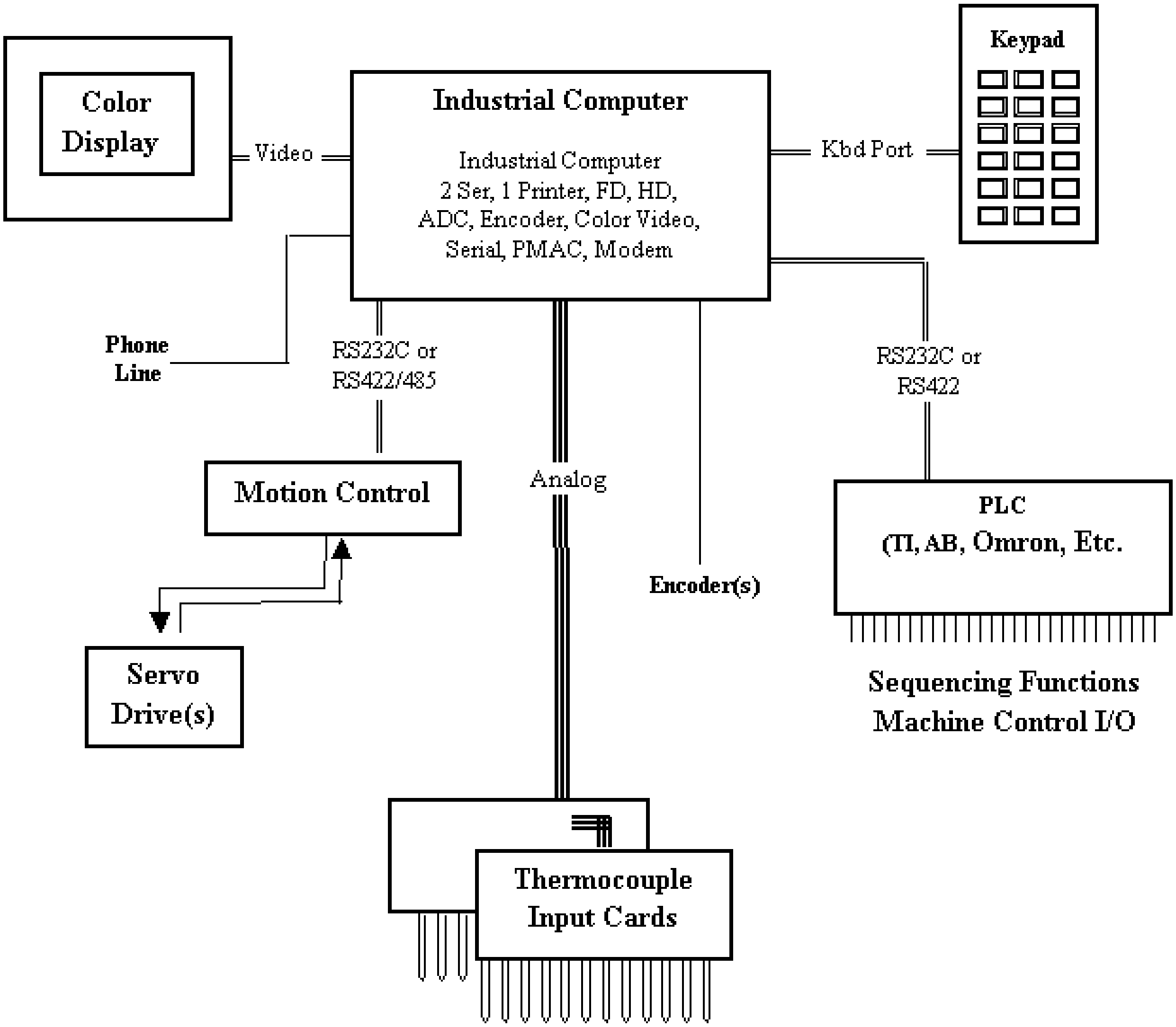
The screens used in this software are **MANUAL FUNCTIONS**, **TIMES**, **TEMPS**, **RECIPES**, **DOORS**, **COUNTERS**, **PRODUCTION**, **SETUP**, **ADJUST**, **SERVOS** and other supporting graph and data viewing screens. The data files define which controls appear on those screens, where those controls show up, what functions those controls perform, and how they interface with the PLC. The analog card is used for reading thermocouple information and the Control program uses this information to control temperatures. This card is also used for feedback for some adjustment functions. The encoder board or motion control board is used for reading encoders for X/Y adjustments.

The PLC's used by the **HMI/Control** program are connected to the computer by a serial port. The HMI/Control software includes drivers for various types of PLC's and communicates with them through PLC memory locations and timers. Individual bits in memory locations, whole memory locations, and timer presets are used to provide control data, diagnostics information, setpoints, and actual values to and from the PLC. The low level control logic and time-dependent functions of the machine are handled by the PLC, since the serial interface is not generally fast enough to handle real time control functions. A watch dog timer in the

PLC safeguards against computer lockups causing dangerous control system failures. Cycle timing and certain adjustments are performed by the PLC logic, and the timers in the PLC are programmed by the computer to control temperatures.

The servo systems supported by this HMI/Control software are **Delta Tau MCC** and **PMAC**, and **Indramat CLC** systems. **Delta Tau MCC** cards are stand alone cards connected to the computer by a serial port. **Indramat CLC** cards are more modern servo control cards integrated into the Indramat servo systems. They are connected to the computer by a serial port as well. **Delta Tau PMAC** cards are multi-axis servo control cards capable of controlling from two to eight axis simultaneously. The **PMAC** cards may be internal to the computer, or connected serially as a stand-alone card or as part of a **Delta Tau PMAC-PAC**. All servo control cards are programmed with a set feedrate, and index program based on user input to the HMI/Control program. These servo systems are then indexed automatically by the PLC logic, and are therefore synchronized with the other functions of the machine.

HMI/Control System BLOCK DIAGRAM



Theory of Operation:

On startup, the computer does a Power on Self-Test (P.O.S.T.), then boots up the operating system. This system is Windows 95/98. The HMI/Control software is then run from the startup group, or as the system shell. (**HMICTRL\HMICTRL.EXE** can be configured as the Windows shell by editing the WIN.INI file). The HMI/Control software first reads the configuration data files from disk and initializes internal variable data. It then checks the PLC connection by writing and reading to and from the PLC. If this test passes, the HMI/Control program will then initialize the PLC data. Normal operation of the machine can begin once the PLC is initialized.

The first operator screen displayed is the Manual Screen. The layout of this screen is defined by the

MANUAL form. Normally, the first thing the operator must do in running the machine is clear faults by using the Fault Reset control. This control clears the internal faults in the HMI/Control program such as **X/Y Following Error**, and **MCC Communication Error**, and then clears the latched faults in the PLC. If any fault conditions still exist, the fault messages will be displayed again. When all faults have been cleared, the other manual functions from the Manual Screen can be used, and the key functions listed on the bottom of the machine can be used. Adjustments from the Adjust Screen can also be done. All key functions can be performed anytime they appear on the screen. The manual functions are generally used to turn on and off the outputs on the PLC.

Different classes of functions are handled by the HMI/Control software in different ways. The manual functions and key functions each set a bit in a word in PLC memory to turn on a function. Some of these functions are automatically turned off under certain circumstances. The data for handling manual functions is listed in the **Properties** of the **ManButton** controls on the **MANUAL** form, and the data for handling key functions is listed in **Properties** of the **FncButton** controls on the **BUTTONS** form . How each function is handled is determined by the Properties of those controls. The automatic timed functions are controlled by writing to timers in the PLC. A start and stop timer is associated with each of these functions. These timer numbers are listed in **Properties** of the **TimeStart** and **TimeStop** controls on the **TIMES** form. Some of those properties are also used to determine if each of these timed functions is Form Dwell or Cycle Time. A Timing Graph Screen can display a line chart showing the relative start and stop times for any timed functions.

Temperature control is done from the Temperature Screen. The layout of this screen is defined by the **TEMPS** Form. This screen allows the operator to input set temperatures and view the actual temperatures for all heaters on the machine. Temperature control is normally done by reading temperatures from thermocouples and writing to timers in the PLC in order to control the output pulse width for each heater section. The PID (Proportional, Integral, Derivative) control loop is normally performed by the computer . The **TempSet** and **Temp** Controls on the**TEMPS** Form tell which thermocouple card/channel and which timers in the PLC are used for each zone. Temperatures can also be controlled by the PLC in which case the computer simply writes the set temperatures to the PLC, and reads the actual temperatures from the PLC. The actual temperatures can be graphed, and trends can be viewed on the temperature graph screens. Temperature data can be collected for graphing every 1, 10, or 120 seconds.

Automatic adjustments are done from the Adjust Screen. The controls on this screen are used to perform all adjust and home functions. There are two types of functions, PLC adjustments and X/Y adjustments. PLC type adjustments are done by writing set positions to words in the PLC, and setting the appropriate bit in the Move Word. The adjustment is then handled by the PLC logic. A home function can also be performed by setting the correct bit in the Home Word. This function will normally run until the Home Flag is seen by the PLC. The other type of adjustment is an X/Y adjustment, where the control loop is closed by the PC. In this case, the input device is tied to the PC, and the PC communicates with the PLC to control the motor itself. Properties of the **AdjSet** and **Adjust** controls are used to specify an adjust function as a PLC or X/Y type of adjustment. These controls are located on the **ADJUST** form.

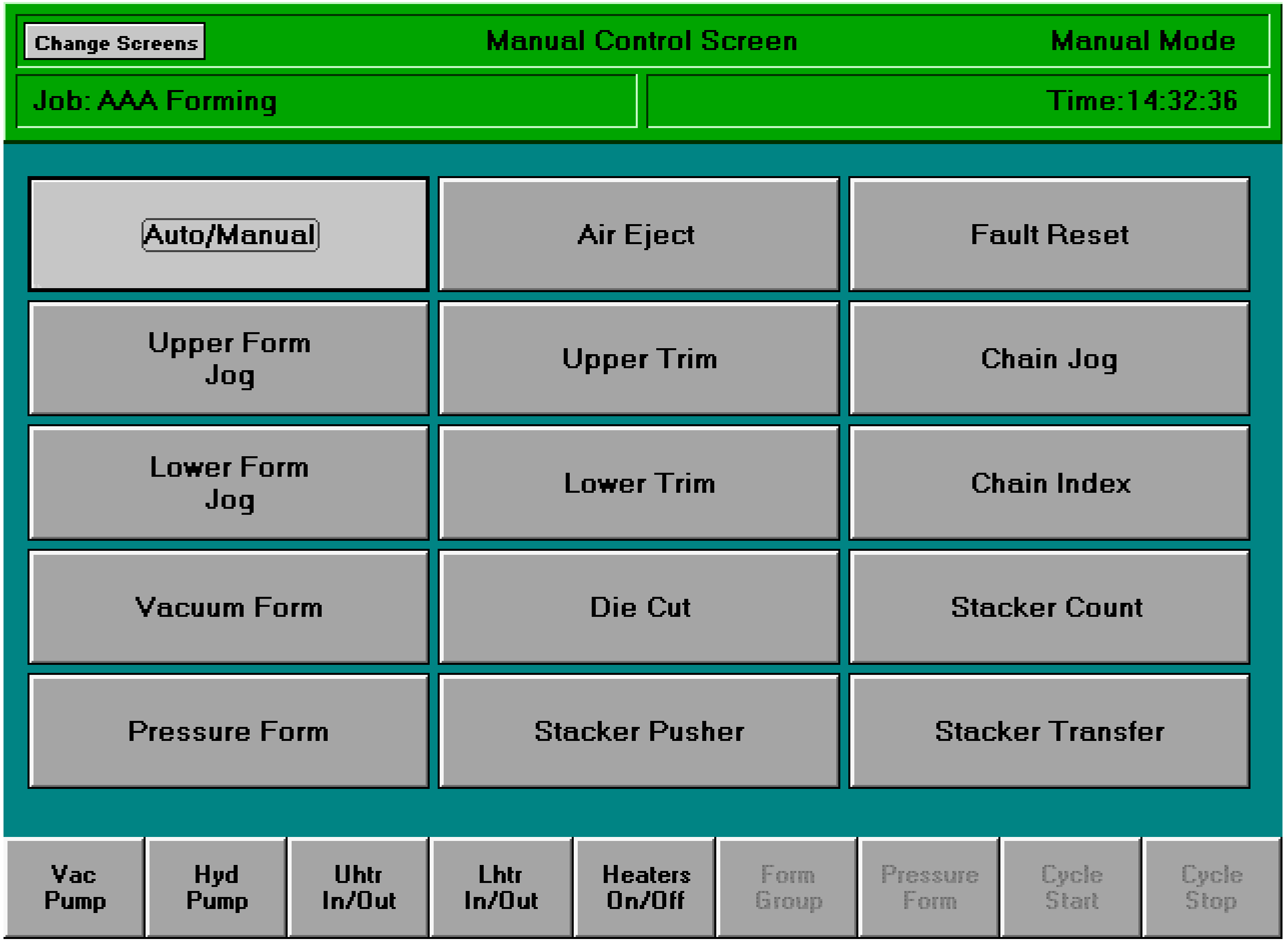
Recipe storage is available on all machines. This is actually the use of data files for storing recipes (groups of setpoints) for all standard variable data used on the machine, such as start and stop times and set temperatures. Production information is stored with each recipe file. This includes time and counter information and other data related to machine operation and expenses. production data for the current recipe can be changed from the Setup Screen and viewed on the Counter Screen and Production Screen.

Fault messages such as E-stop or door diagnostic information is shown on the Door Diagnostic Screen. The information on machine operation errors is communicated to the computer by setting a bit in the Status Words for each error condition as it is latched in the PLC logic. The computer reads these locations periodically to see if the machine has any error conditions. If so, the computer will go to the Door Diagnostic Screen and display that error. The individual machine faults and door diagnostics are determined by the Properties of the **Fault** and **Door** controls on the **DOORS** form.

Environment variables are used to control some of the functions on the machine. These variables can be set from the DOS prompt by saying **SET variable name = value** or by entering the line in the **HMICTRL.INI** file. Entries in **HMICTRL.INI** will override any DOS Environment variable settings.

There are environment variables to turn off error trapping, run in monochrome mode, limit the instantaneous heater current to this value, enable diagnostic error display, run in demo mode, save recipe on cycle stop, save recipe data to a temporary file on cycle stop, ignore encoder input while not performing a move, store comment files when storing recipes, logically tie together adjustments when the mechanical adjustments are attached to each other, disable all PLC keys, perform analog calibration from the center position, limit the cycle rate to this value, ask the user for a recipe name to search for when going to the recipe screen, write SLC500 heater timer presets to an integer file, set batch size for heater timer preset updates, check control parameters for duplicate addressing, etc., Enable/Disable Functions of Manual and Timing Screens, delay time between PLC communication, display dates in date, month, year format, refresh forms periodically, set number of samples for Temp Screen, enable filtering of analog input values, specify address and size of memory block in PLC for analog values when analog inputs are tied to the PLC, allow changes to index length while cycling, set backlash time for adjustments when changing direction, check servo parameters for testing servo sizing values, set default type of accel curve used for any axis if a curve type is not specified in the SERVOn.DEF file, set relative brightness for highlighting buttons on Manual Form, Adjust Form, Setup Form, etc. when getting and loosing focus, set time in seconds for periodic refreshing of heater timer presets, log adjust actual position changes when not adjusting, log PLC communication to Debug Window or history file, set Security level for performing common operations, and set accelerator keys for operating from the keyboard.

The Manual Screen


Manual Control Screen

Manual Mode

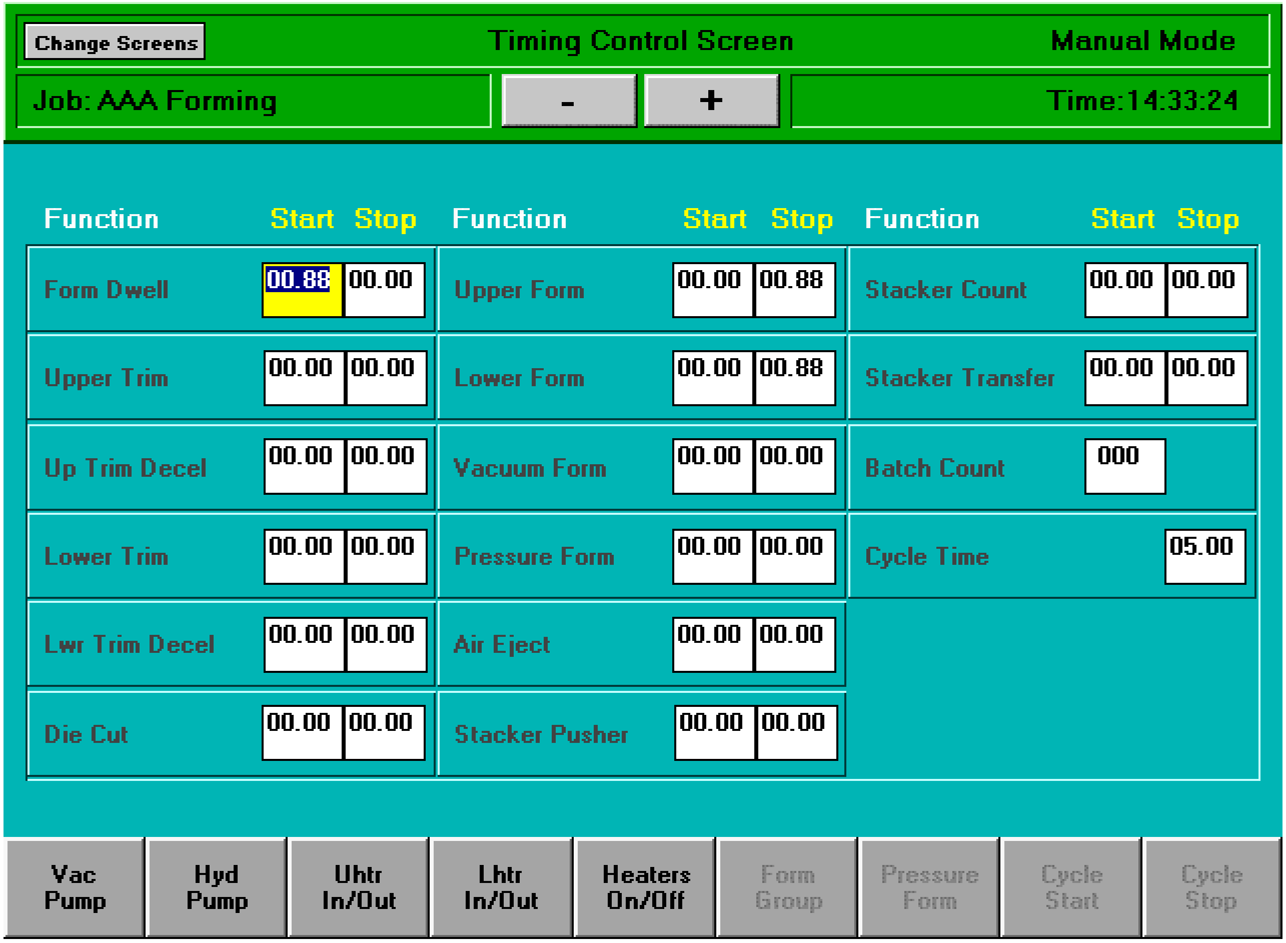
Job: AAA Forming

Time:14:32:36

The HMI/Control program uses the **MANUAL**Form and the **Properties** of the **Manual** controls on it to draw the Manual Screen and control the functions on that screen. Like all of the main screens in this system, certain controls showing up on the screen indicate where the data items Manual/Automatic, Date, Time, Recipe Name, Index Length, and Htr Config Override should be displayed. The **MANUAL** Form also has special controls for handling timer routines and communication functions. There are also Three communications controls. The **PLC** control is used to handle serial communications to the PLC. The properties of this control determine the communication parameters such as baud rate and parity, and the Comm Port number used to talk to the PLC. The **Servo** Control is used to for serial communication with the Motion Control card in the same way. Although this software is geared toward operating cyclic machines (i.e. thermoformers), the functions included in this system can be used to operate continuous processes. (such as extrusion) The screen shots shown here are just an example of what can be done with this software. The appearance of the screens made with this software could differ drastically from the general layout shown here.

The **MANUAL** Form has an **array** of **Manual** controls. These controls can be activated by using a touch screen, or by using the arrow keys and pushing enter. Pushing enter will cause that function to toggle on or off. Push it once it turns on. Push it again it turns off. That's going to happen unless it is an on/off button in which case it will turn on and then back off. The on/off state of these buttons is communicated to the PLC by using the word and bit specified in Properties of that control. Code is included in the Control module to handle highlighting button controls when they get/lose the focus.

The Function Timing Screen



Timing Control Screen

Manual Mode

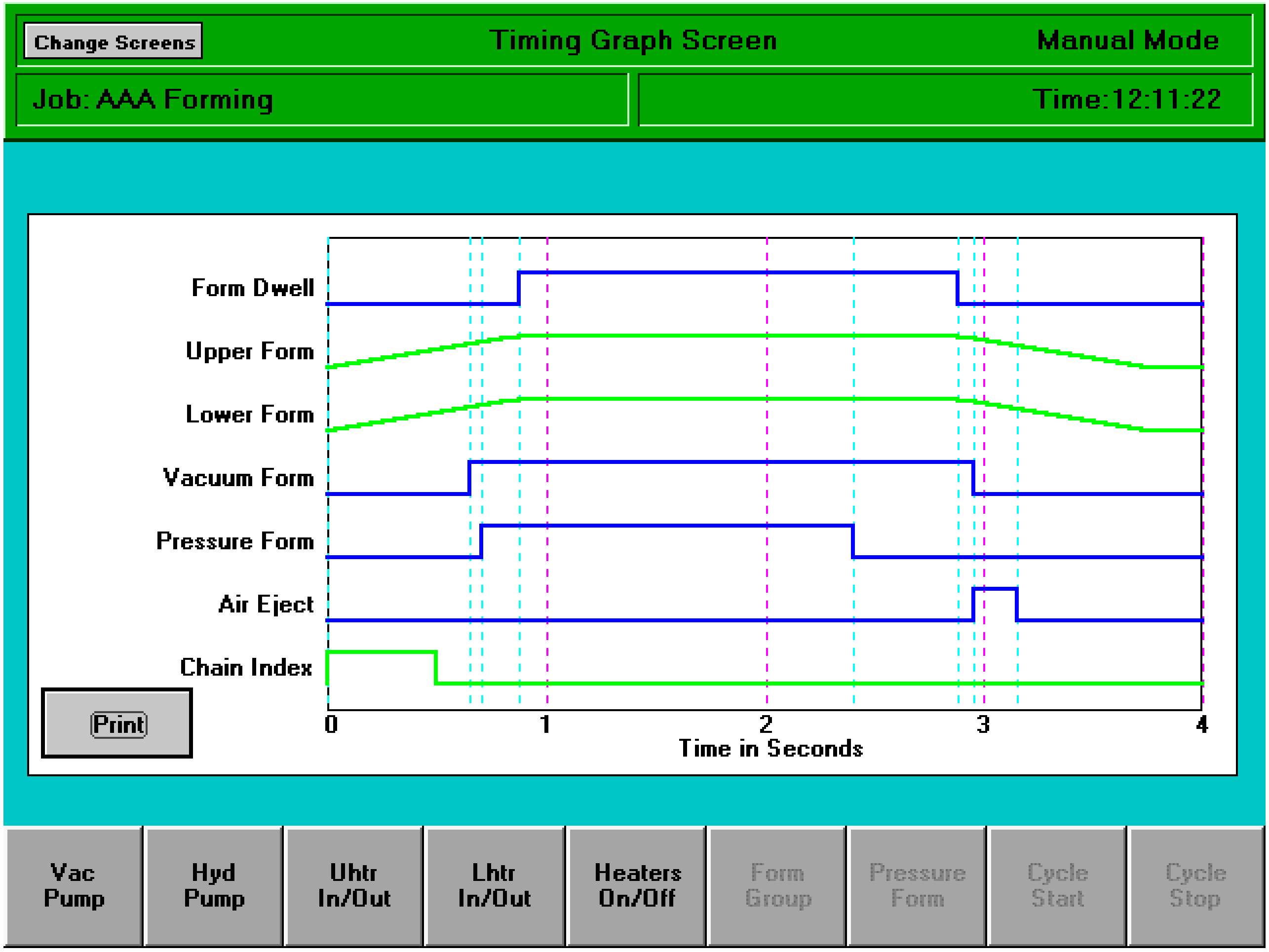
Job: AAA Forming

Time:14:33:24

Function	Start	Stop	Function	Start	Stop	Function	Start	Stop
Form Dwell	00.88	00.00	Upper Form	00.00	00.88	Stacker Count	00.00	00.00
Upper Trim	00.00	00.00	Lower Form	00.00	00.88	Stacker Transfer	00.00	00.00
Up Trim Decel	00.00	00.00	Vacuum Form	00.00	00.00	Batch Count	000	
Lower Trim	00.00	00.00	Pressure Form	00.00	00.00	Cycle Time		05.00
Lwr Trim Decel	00.00	00.00	Air Eject	00.00	00.00			
Die Cut	00.00	00.00	Stacker Pusher	00.00	00.00			

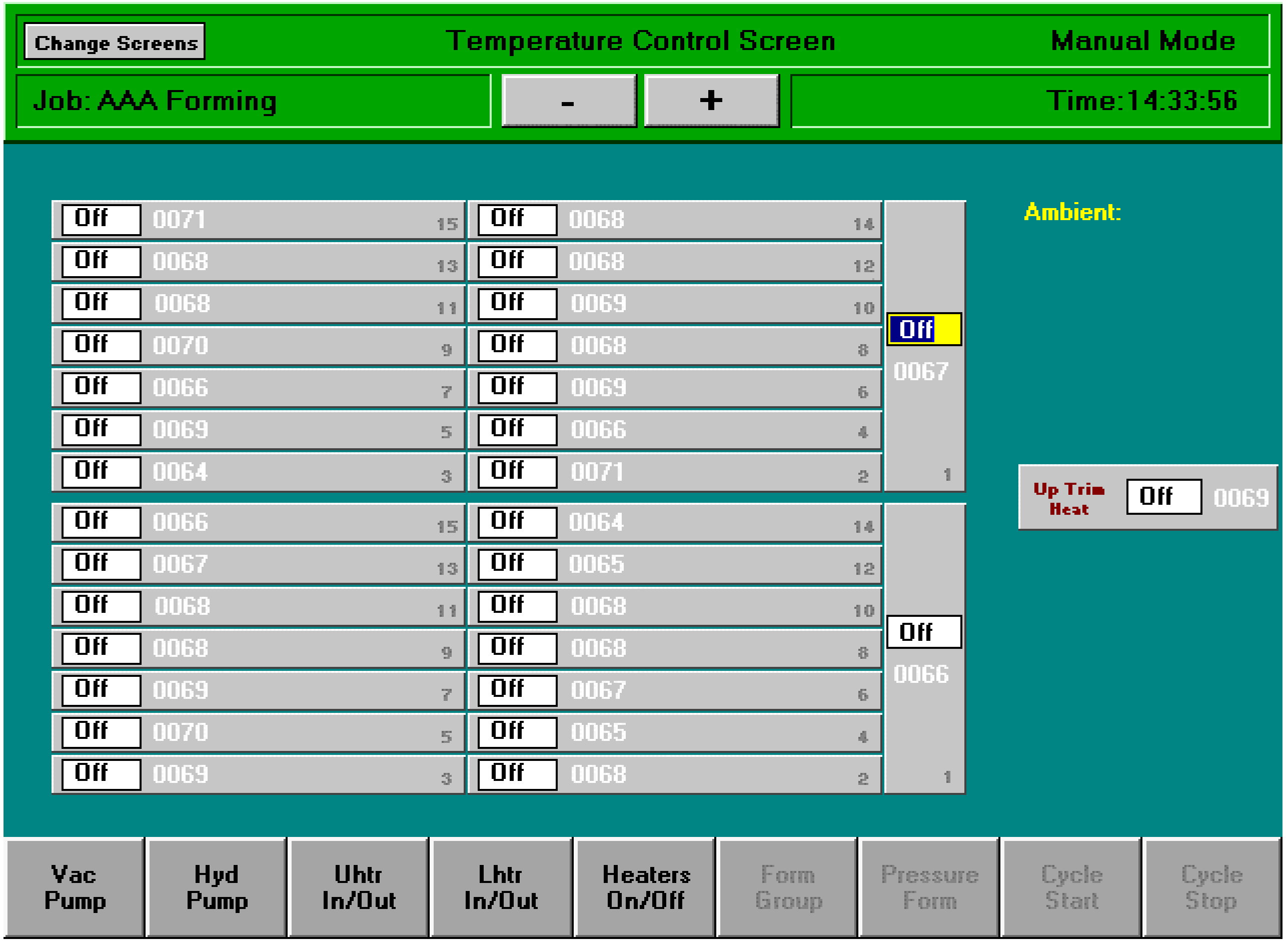
The HMI/Control program uses the **TIMES** Form and the **TimeName**, **TimeStart** and **TimeStop** Controls to draw the Function Timing Screen and control the functions on that screen. The **TIMES** Form defines the screen layout of the Function Timing Screen. Like all of the main screens in this system, certain controls showing up on the screen indicate where the data items Manual/Automatic, Date, Time, Recipe Name, Index Length, and Htr Config Override should be displayed. Servo setpoint Controls are used for setting servo parameters. The servo setpoint control will update a motion controller variable (**P Variable**). Servo setpoint controls can specify a maximum value for that set point.

The Timing Graph Screen



The **TimeGraph** Form Has **FncName** Controls on it. These controls are used to specify what Time Functions to display where on this form.

The Temperature Screen



Temperature Control Screen Manual Mode

Job: AAA Forming

Time:14:33:56

Off	0071	15	Off	0068	14			
Off	0068	13	Off	0068	12			
Off	0068	11	Off	0069	10	Off		
Off	0070	9	Off	0068	8	0067		
Off	0066	7	Off	0069	6			
Off	0069	5	Off	0066	4			
Off	0064	3	Off	0071	2	1		

Off	0066	15	Off	0064	14	
Off	0067	13	Off	0065	12	
Off	0068	11	Off	0068	10	Off
Off	0068	9	Off	0068	8	0066
Off	0069	7	Off	0067	6	
Off	0070	5	Off	0065	4	
Off	0069	3	Off	0068	2	1

Ambient:

Up Trim Heat Off 0069

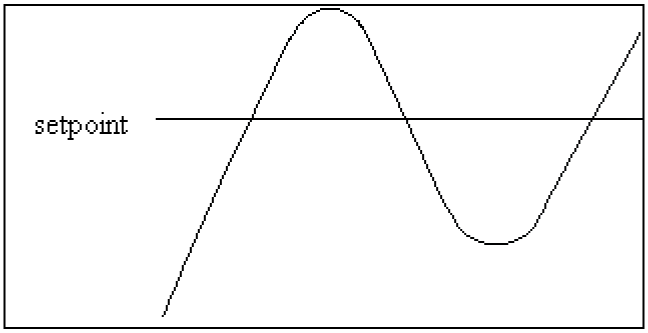
The HMI/Control program uses the **TEMPS** Form and the **Heater**, **HtrName**, **TempSet**, and **Temp** Controls on it to implement the Temperature Screen and control the functions on that screen. Like all of the main screens in this system, certain controls showing up on the screen indicate where the data items Manual/Automatic, Date, Time, Recipe Name, Index Length, and Htr Config Override should be displayed. Temperature control on this system is done either by PID control in the computer, or by writing to a control word in the PLC. Calculations are done in the computer for PID temperature control, and the Temp control timer pair are set to produce the desired heater output. The temperature control can also be set to use a single bit for heat/cool control. PID settings for the PID temperature controls can be changed by the operator at run time. A two point calibration function allows calibration of each temperature zone.

The **TempSet** controls are used for inputting the setpoints for temperature control. The **Temp** Controls are used for displaying temperature control actual values. These controls can be set so that zone is handled by writing to and reading from a separate Temperature Controllerl over a serial interface. The external temperature controller can use Modbus protocol. (such as used on the **Eurotherm 808** or **2408** temperature controllers) If Modbus protocol is not specified, then an Omron **E5AX** controller is assumed.

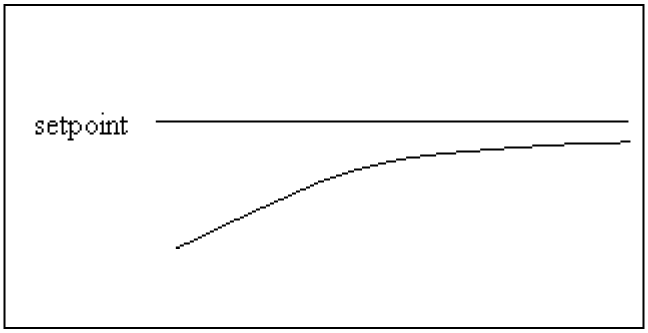
PID Temperature Control

PID process control is normally used to control the temperatures by the HMI/Control program. The outputs in the PLC are normally controlled using a Time-Proportioning technique which outputs a rectangular wave to the heater elements. This technique is sometimes referred to as Pulse Width Modulation or PWM. The basic function of a PID controller is to provide the output needed for the process value (actual temperature in this case) to follow the set value (set temperature) as closely as possible. Normally, the set temperature remains steady and the actual temperature must stay as close to that set temperature as possible. The error (difference between set and actual) is used to create the **PID** control's output using three factors; **Proportional Gain**, **Integral Gain** (sometimes called Reset), and **Derivative Gain**.

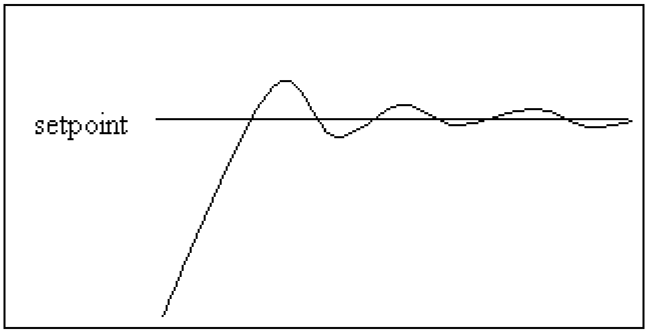
A simple Proportional controller outputs a control value inversely proportional to the process error. (Proportional control is really PID control with the Integral Gain and Derivative Value both set to zero.) The simplest form of Proportional control to implement would be to assume infinite proportional gain. The output would therefore go on (output = 100%) when the temperature is too low, and go off (output = 0%) when the temperature is too high. (This would be like the thermostat in your car). Normally, a proportional controller will not control as well as we need, because the control output is simply directly proportional to the difference between the set and actual values. If the Proportional Gain is too low, it will take a very long time to approach setpoint. If the Proportional Gain is too high, the actual temperature will swing far above and below the setpoint. Normally the desired action of a proportional controller is for the actual value to reach the setpoint and overshoot it a little, and then drop a little below the setpoint before oscillating above and below the setpoint within a reasonable range. The diagrams below show the effects of changing the Proportional Gain in a system.



The Proportional Gain is set **too high** here. The actual value reaches the setpoint and continues far beyond it, finally turning around and going far below it, never settling down close to the setpoint.

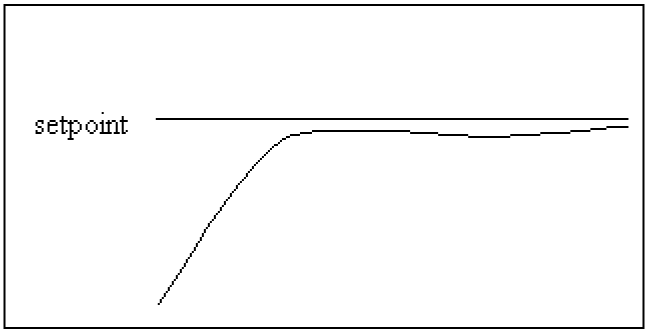


The Proportional Gain is set **too low** here. The actual value doesn't reach the setpoint in a reasonable time, and will not react well to changes in temperature or changes to the process. (for example, changes in ambient temperature and cooling effects) It also will not react quickly to changes in set temperature.



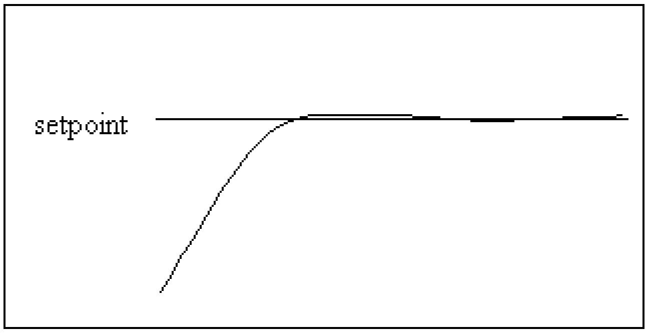
The Proportional Gain is a reasonable value here. The actual value reaches the setpoint and continues a little beyond it, then turns around and goes a little below it, settling down close to the setpoint and oscillating around that point over a reasonable range.

Derivative Gain is used to limit the amount of change in the actual value. The derivative function from calculus is the rate of change of the original function (the actual temperature here). Derivative Gain when increased will decrease the maximum rate of change of the actual value. When used with the Proportional control, a PD control is created. Appropriate values of Derivative Gain should result in a response something like that shown below.



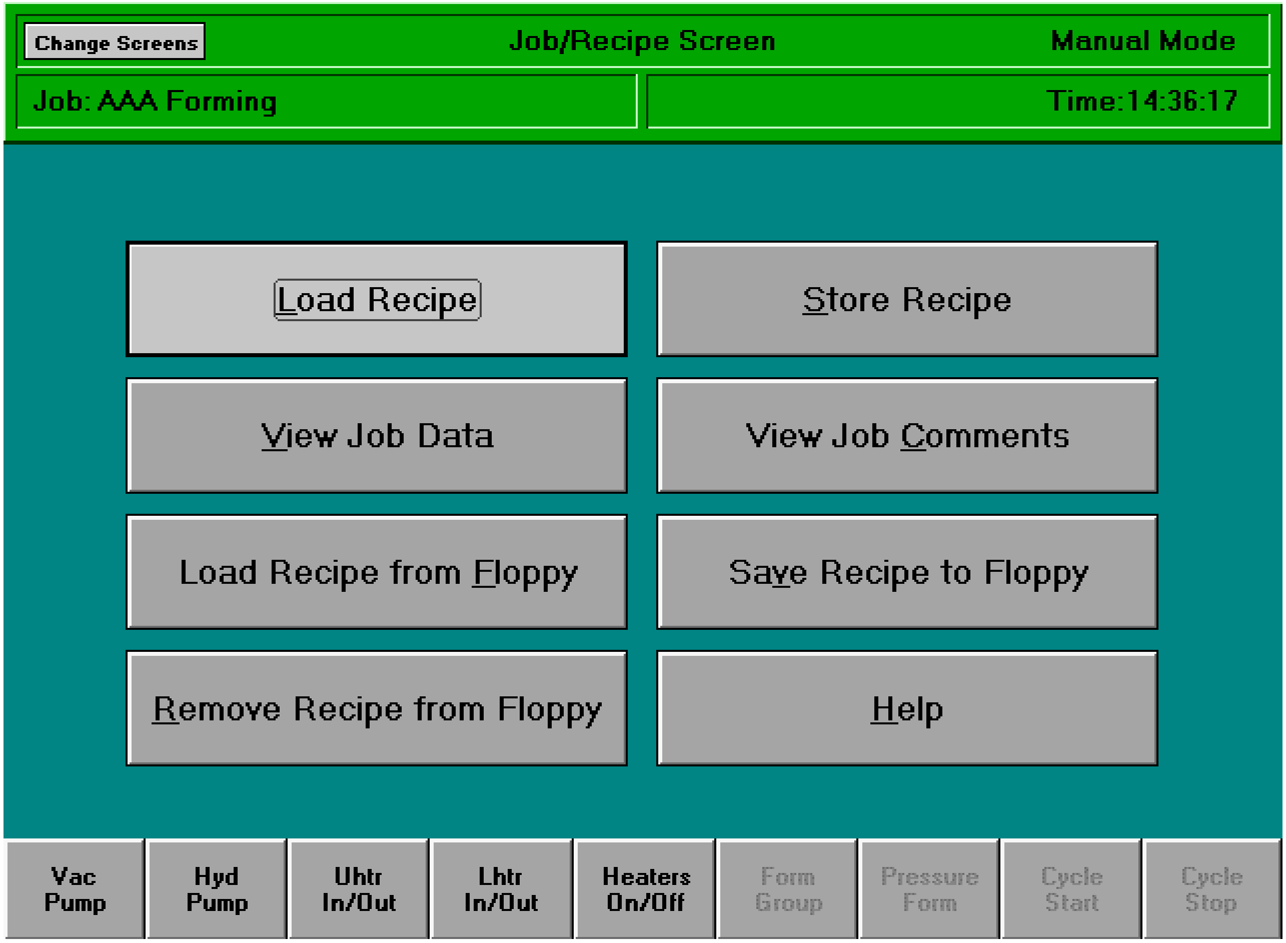
You can see here that the response of the PD controller is closer to the theoretical perfect response that we would like to see. The deviation in actual temperature is less, and there is less overshoot. It takes just slightly more time to get to the setpoint, but we are controlling close to the setpoint sooner.

The addition of an Integral function makes our PD controller into a PID controller. The Integral function from calculus is the inverse of the derivative function. Rather than representing the rate of change of the original function (actual temperature), it is the accumulation of what the original function has been over time. This function when added to a PD controller has the effect of boosting up the value that we settle out at by taking into account where we are coming from. The net result is that the actual value looks like that from the PD controller, but pushed up a bit. The response from a correctly tuned PID controller is illustrated below.



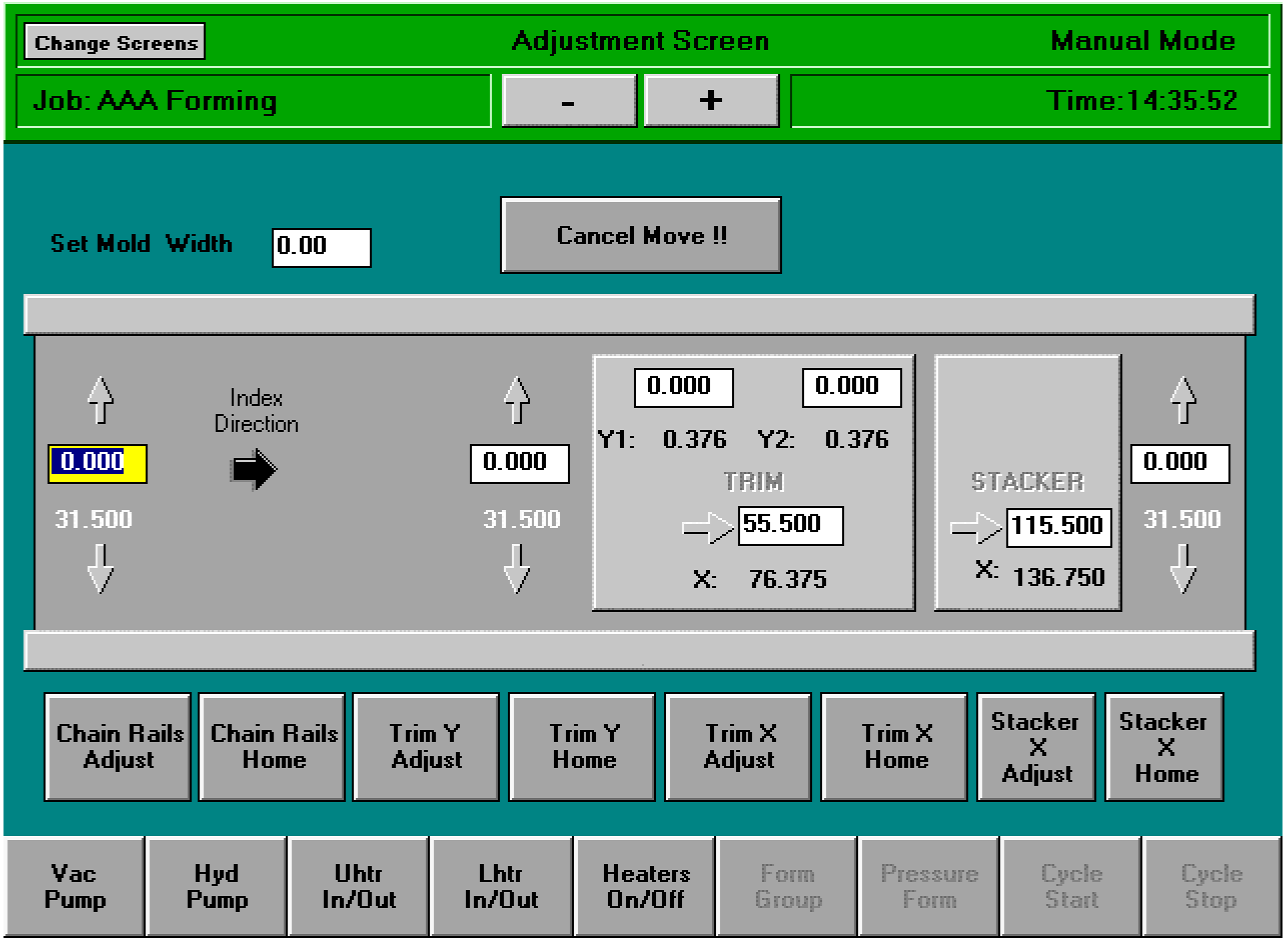
You can see here that the response of the PID controller is much closer to the theoretical perfect response. The deviation in actual temperature is very small, and there is very little overshoot. It takes less time to get to the setpoint, and we are controlling closer to the setpoint sooner.

The Recipe Screen



The HMI/Control program uses the **RECIPES** Form for the Recipe Management Screen and control the functions on that screen. Each option on recipes Form is implemented by a **PushButton** control. The Recipe Change function of these controls can be set to **LOAD FROM NETWORK, LOAD FROM FLOPPY, SAVE TO FLOPPY, REMOVE FROM FLOPPY, LOAD, STORE, VIEW COMMENTS, VIEW DATA,** or **HELP.**

The Adjustment Screen



Adjustment Screen   Manual Mode

Job: AAA Forming

Time:14:35:52

Set Mold Width 0.00

0.000 31.500

0.000 31.500

0.000 0.000

Y1: 0.376 Y2: 0.376

TRIM 55.500 X: 76.375

STACKER 115.500 X: 136.750

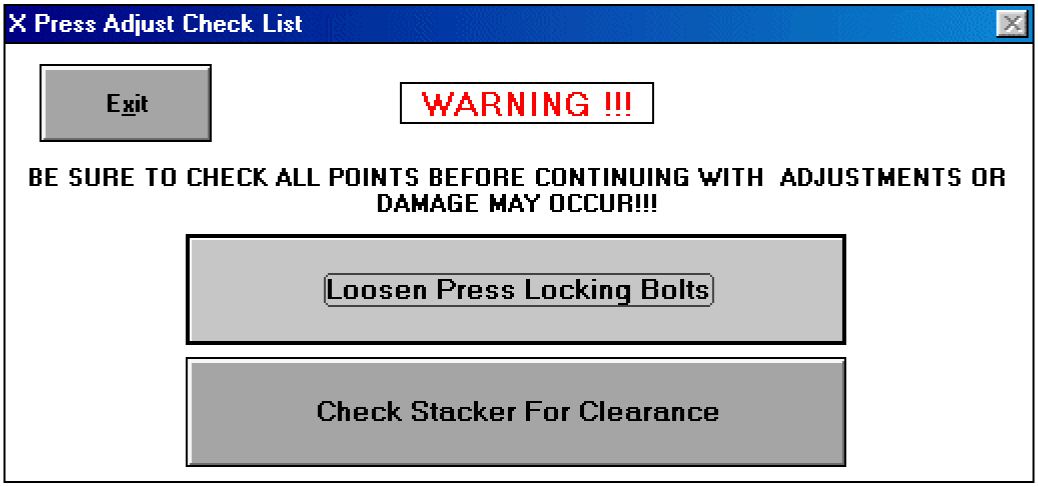
0.000 31.500

The HMI/Control program uses the **ADJUST** Form to draw the Adjustment Screen and control the adjust functions on that screen. The adjustments on these machines are done in one of two ways. Either the adjustment logic is handled entirely by the PLC or the move and reverse bits are used by the computer to do the adjustment. PLC type adjustments are done by writing the desired position to a word in the PLC, and then setting the Move Bit in the PLC. The home function for PLC type adjustments is done by setting the Home Bit in the PLC. The PLC will then handle the function completely on it's own. Cancel controls can be used to cancel PLC moves by setting the cancel bit. Adjustments done through the computer are called X/Y adjustments**.** X/Y adjustments can be grouped together by setting properties in the adjust controls. All adjustments grouped together will be incrementally moved closer to their desired position. These adjustments use the **MoveBit** and **ReverseBit** in the PLC to run the adjustment in forward or reverse. Speed bits in the PLC control the motor speed. (When variable speed motors are not used, the PLC can use timers to pulse the output to the motor at different rates based on the setting of the speed bits.) Properties can be set to determine what distance from the setpoint uses the high, medium, and low speeds of the motor (This is a somewhat crude form of proportional control). Adjustments can be set to automatically adjust when plus and minus are used. Also adjustment's can be limited to greater than the set Mold Width. Each Move or Adjust function on the ADJUST Form is implemented by a **MoveButton** control. The Adjustment functions are set to match these controls. The **MoveButton** controls can be set to perform the following functions: **RAILS ADJUST, RAILS HOME, PRESS ADJUST, PRESS HOME, PLC ADJUST, PLC HOME, X/Y ADJUST, X/Y HOME, X/Y MIN, X/Y MAX, X/Y MID, CANCEL,** and**SET X.**

CHECKLISTS

There are a number of checklists which are used when you are doing an adjustment in order to check with the operator and make sure the machine is mechanically ready to do the adjustment. The forms used to implement the Check List functions are standard VB Forms. The CheckList forms have an array of **PushButton** Controls. The Forms may have a **CancelButton**, which when pushed will cause the checklist execution to be canceled, unloading the form.

Checklist Form:

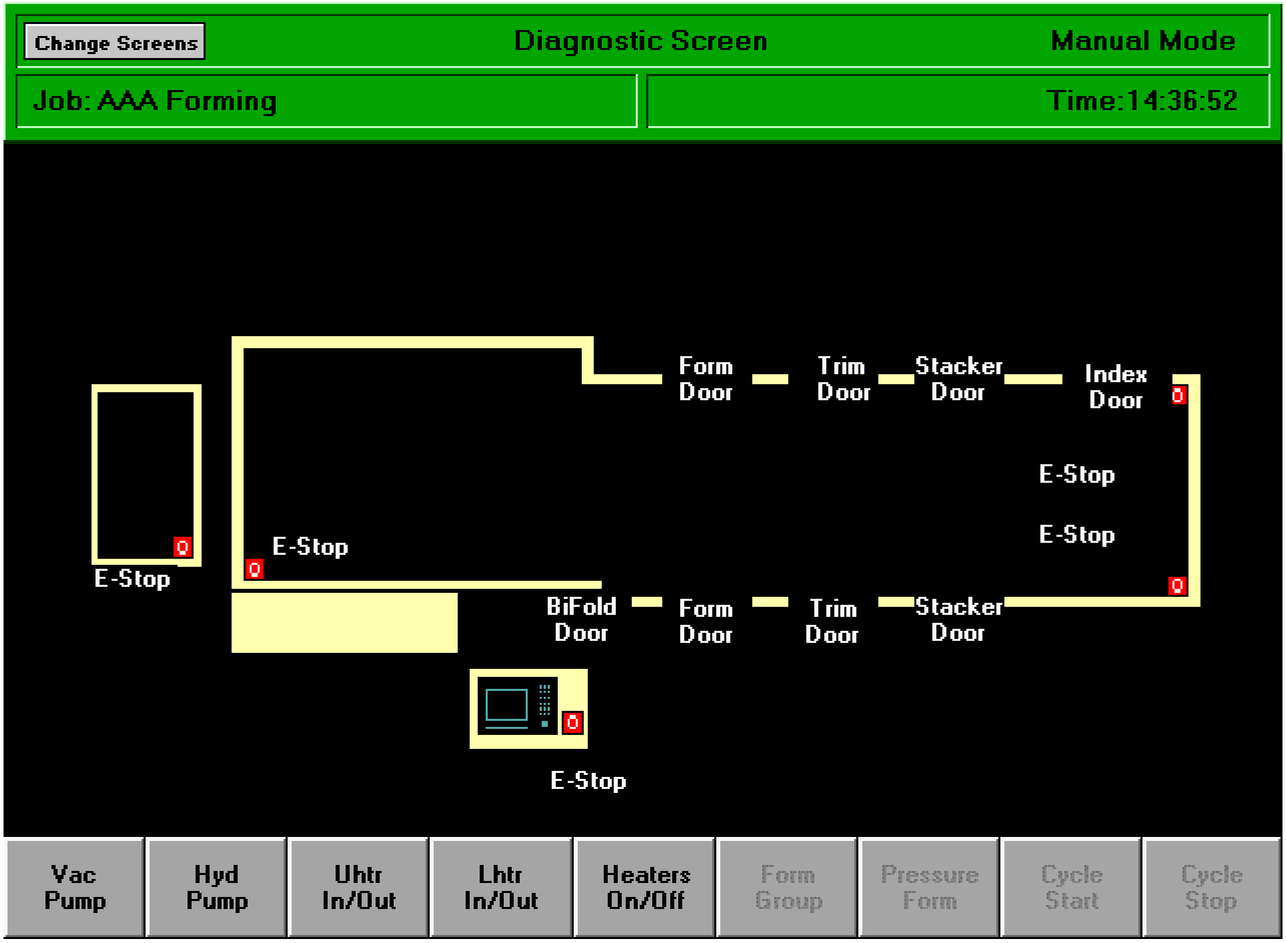


X Press Adjust Check List

WARNING !!!

BE SURE TO CHECK ALL POINTS BEFORE CONTINUING WITH ADJUSTMENTS OR DAMAGE MAY OCCUR!!!

The Diagnostic Screen



Diagnostic Screen

Manual Mode

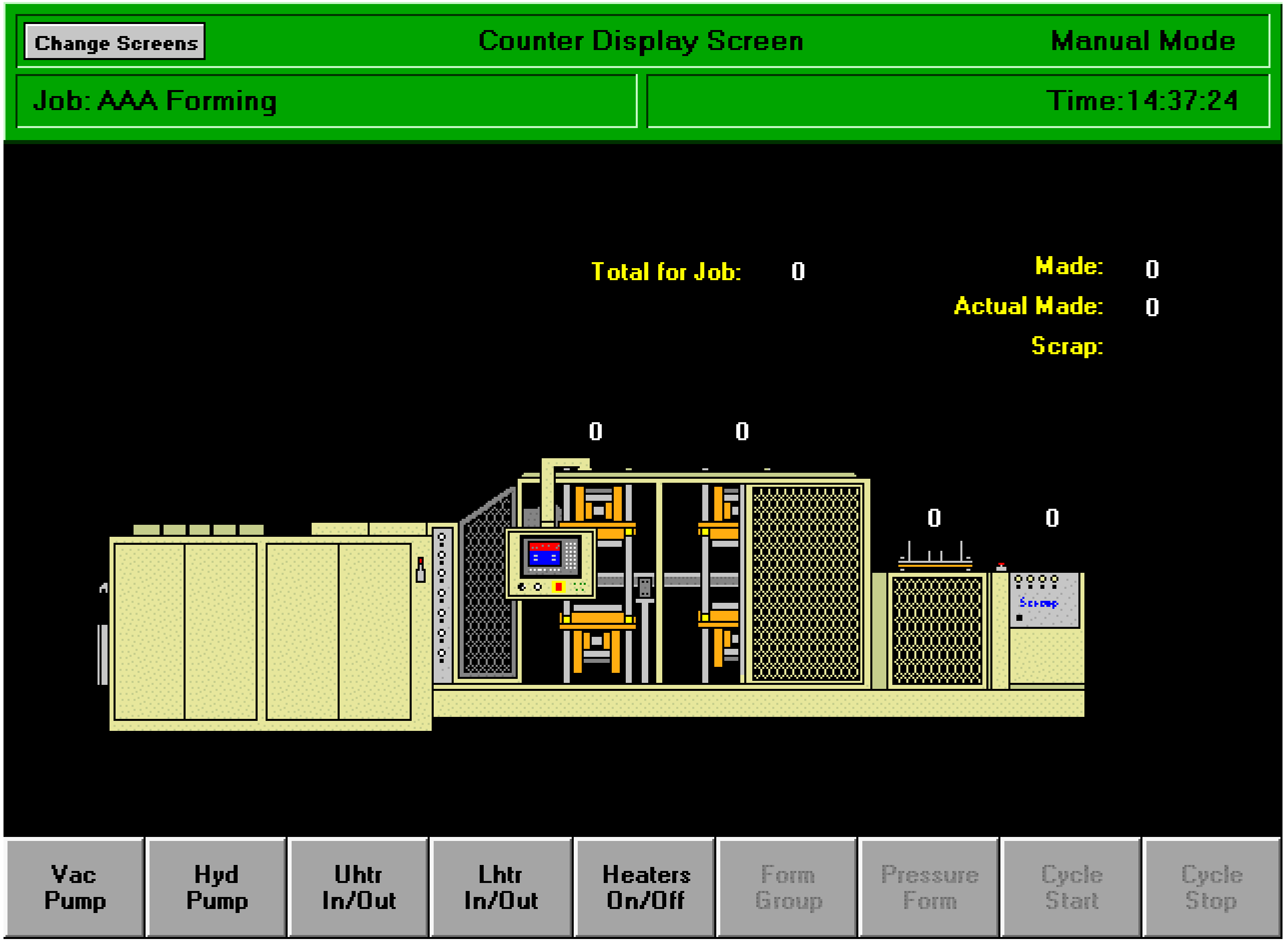
Job: AAA Forming

Time:14:36:52

The HMI/Control program uses the **DOORS** Form to display the Door Diagnostic Screen. The **DOORS** Form has a group of **Door** Controls. There is also a group of **Fault** Controls. The Status Words in the PLC are periodically polled to see if any of the Fault bits are set. If they are, then a fault condition exists. If the machine is cycling, then the machine Cycle Stops and all functions which are to be cleared on an error are cleared. On a fault, the screen changes to the Door Diagnostic Screen. If the **Fault** Control is set as a critical error, then the heaters are turned off. While the Door Diagnostic Screen is displayed, the Door Word in the PLC is read and the **Door** controls are displayed either white if they are OK, or blinking red if they are open (latched condition in PLC).

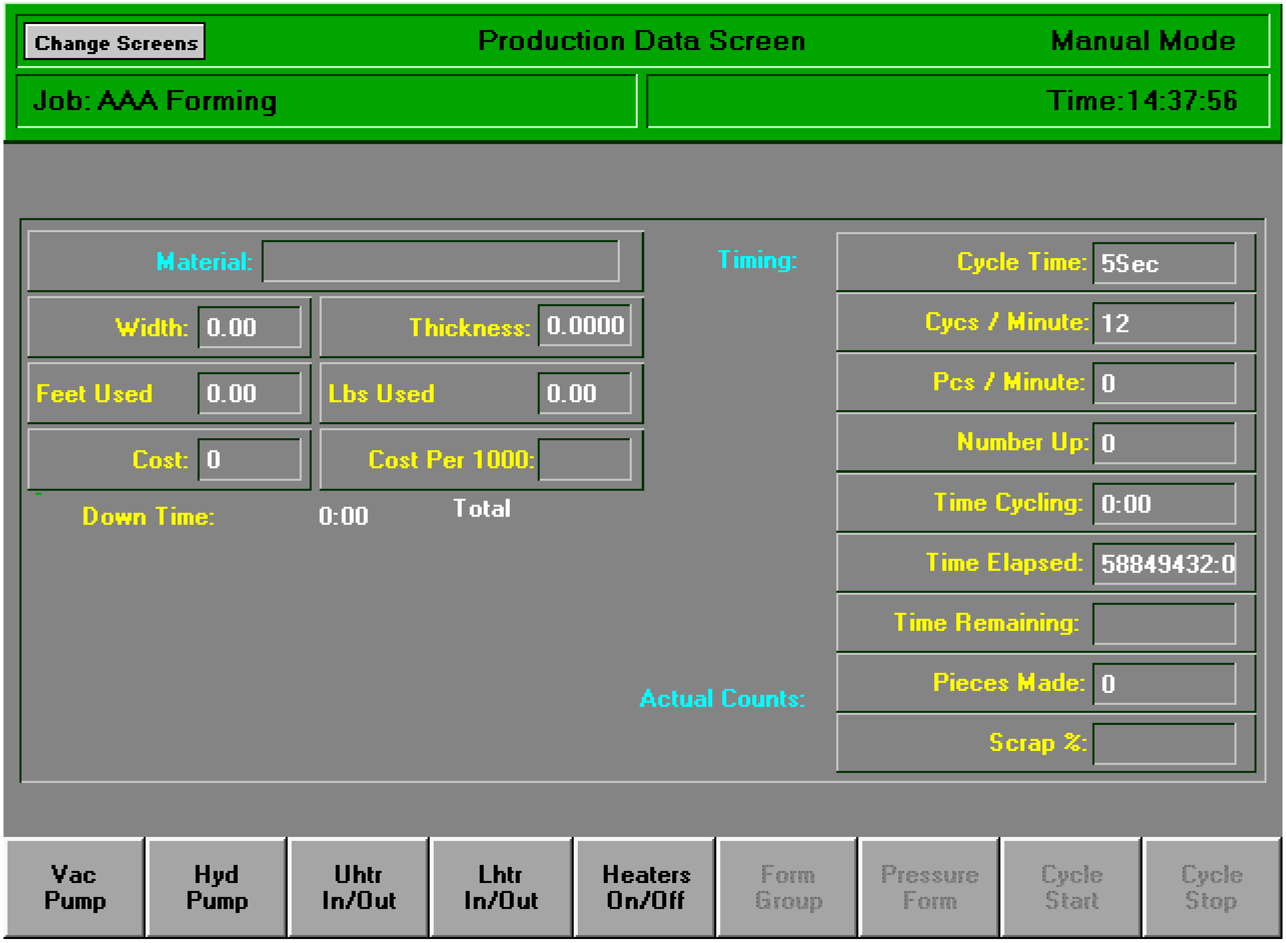
The Production Screens

The Counters Screen



There are two possible production screens corresponding to the two levels of production information available in the system. The machine has production level one if the number of counters is greater than zero. The machine has production level two if the downtime names at the end of HMICtrl.cfg are not blank The HMI/Control program uses the files **COUNTERS** Form and **PRODUCTION** Form for the Production Screens. **ProdData** Controls are placed on these forms to display Counter and Production information.

The Production Data Screen



Production Data Screen Manual Mode

Job: AAA Forming

Time:14:37:56

Material:

Width: 0.00

Thickness: 0.0000

Feet Used 0.00

Lbs Used 0.00

Cost: 0

Cost Per 1000:

Down Time: 0:00 Total

Timing:

Cycle Time: 5Sec

Cycs / Minute: 12

Pcs / Minute: 0

Number Up: 0

Time Cycling: 0:00

Time Elapsed: 58849432:0

Time Remaining:

Actual Counts:

Pieces Made: 0

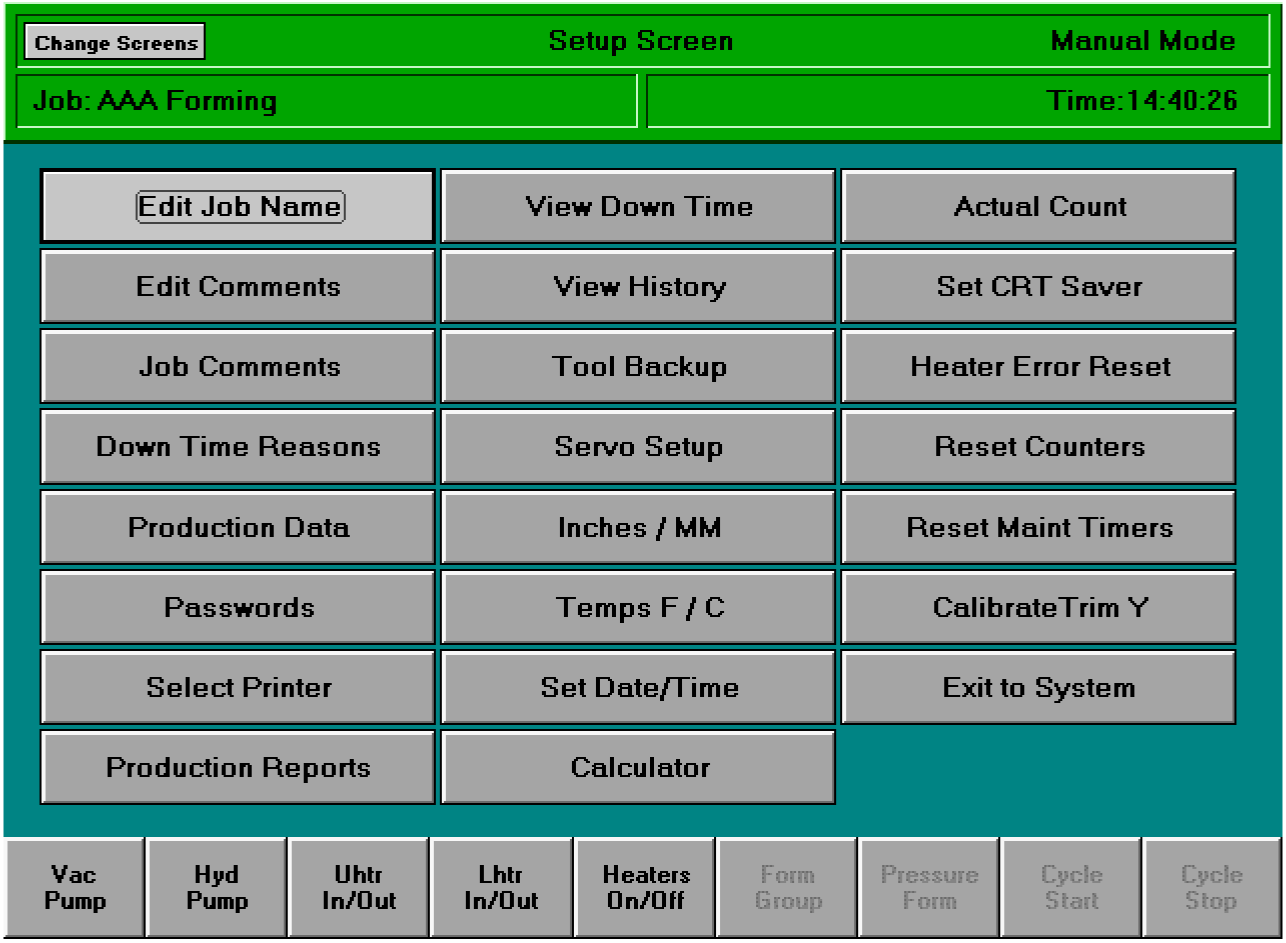
Scrap %:

Production reports are defined by the file **REPORT.PRN**. Production parameters in the reports can be specified as start and finish times for the timed functions. A line in report.prn that says **TEMPERATURE REPORT** will include the file temps.prn. Set temperatures are specified in tempdat1.prn. A line in report.prn that says **COMMENT SCREEN** will cause the comment screen for the current recipe to be included in the report.

The ProdData Function Values

The following function values can be used on the Production Forms and reports: Total Pieces to Make, Pieces Made, Formed Count, Trimmed Count, Stacked Count, Indexed Count, Material Name, Material Width, Material Thickness, "Feet Used" or "Meters Used", Feet or Meters Used, "Pounds" or "Kilograms", Pounds or Kilograms, Cycle Time in Seconds, Cycles per Minute, Number Up, Pieces per Minute, Hours Run, Hours Elapsed, Hours Remaining, Total Downtime, Downtime 1/2/3/4/5, Total Cost of Material, Actual Material Cost per 1000 Parts, Material Cost per 1000 Parts, Total Cost, Actual Cost per 1000, Cost per 1000, Actual Pieces Made, Scrap Percent

The Setup Screen



Setup Screen

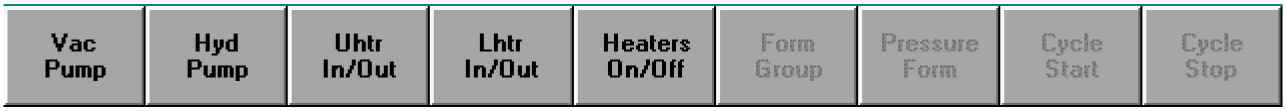
Manual Mode

Job: AAA Forming

Time:14:40:26

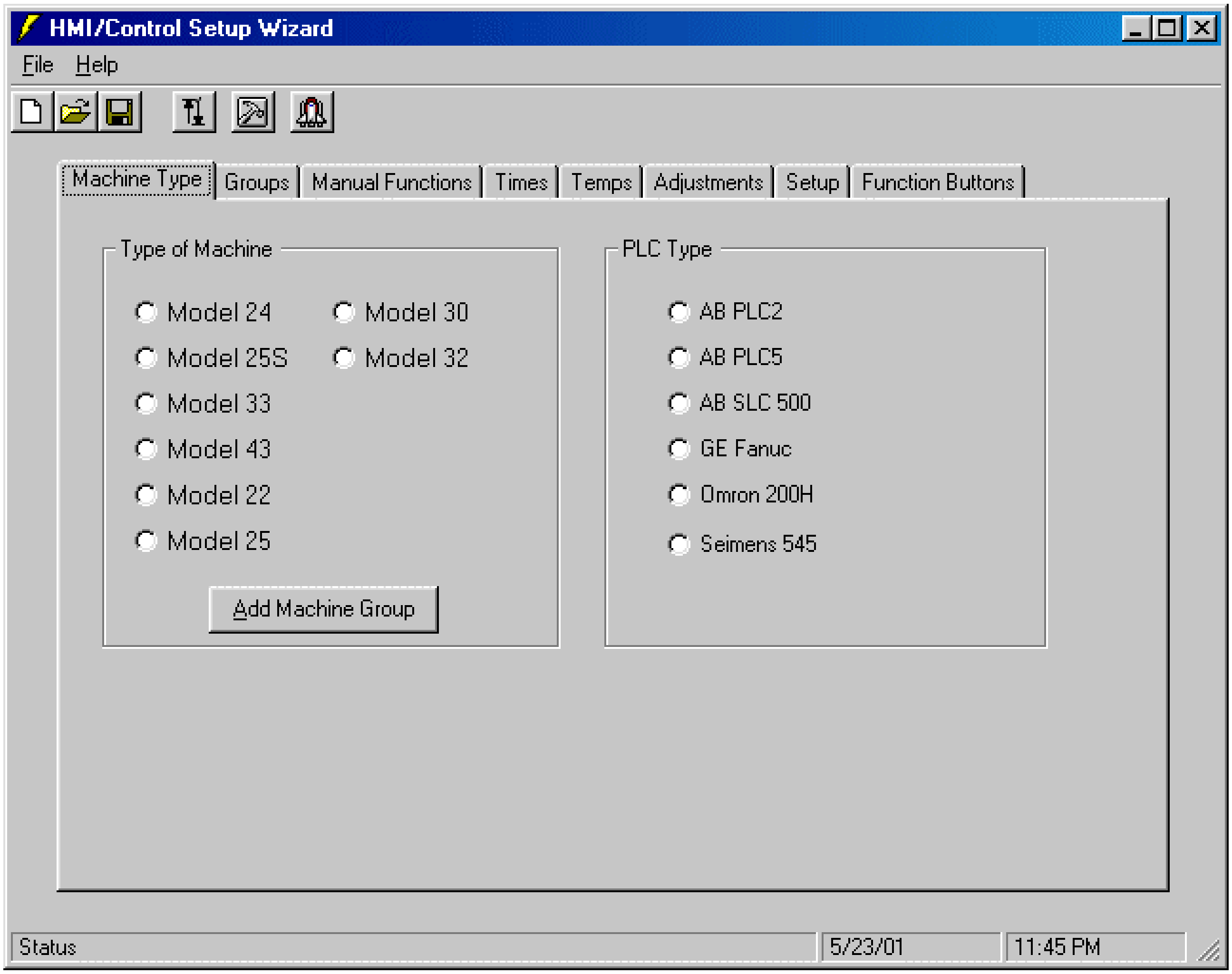
The HMI/Control program uses the **SETUP** Form to implement the Setup Screen and Setup functions for the HMI/Control program. The setup functions chosen from the **PushButton** Controls are: Actual Count, Reset Counters, Change Language, Select index speed, Edit Recipe Name, Edit Comments, Set CRT saver time, Set date/time, Enter Production data, Edit downtime names, Edit passwords / security, Exit from HMI/Control program to Windows shell, Change to/from inches, mm's, Change to/from F/C, Reset heater errors, Reset maintenance reminder timers, Backup recipe data, View history, View Downtime, Set default printer, Calibrate thermocouple values, Calibrate Y automatically, Calibrate Y manually, Calibrate step size, Calibrate offset in PLC, Calibrate X, Print Production reports, Set Target Timing, View job change checklist, Enable manual functions, Enable timing functions, Run another program

The Function Keys



The function keys appear across the bottom of the screen on the **BUTTONS** Form. The function keys each have an associated key code. The key with that key code will cause that function to toggle on and off and write to the PLC in a manner similar to the manual functions. **FncButton** Controls on this form are used to implement the functions assigned to the Function keys on the keyboard. The status of each function is displayed on the corresponding **FncButton** Control by highlighting the caption of that control in either the On Color or the Off Color the appearance of the these colors on the control can be testedat design time.

The Setup Wizard



SetupWiz is a setup wizard for creating the HMI/Control application for a thermoformer. It has a typical wizard appearance with tabs for Machine Type, Groups, Manual Functions, Times, Temps, Adjustments, Setup, and Function Buttons. By going through each of these tab forms and selecting the preconfigured options for the given machine, the programmer can create a custom machine control program in very little time.

The "preconfigured options" are setup in the SetupWiz directory by extracting the code for the controls/forms for that option and putting it in the appropriate sub-directory. (This means all of the text from the .FRM file.) These directories are read at run time and their names automatically appear as options in SetupWiz. After the machine functions are selected, the setup file can be saved, and the project can be built. The Build option will build the complete Visual Basic project, and the Launch option will put you in VB ready to "Create HMICtrl.Exe".

The most logical way to setup a machine is probably to start by selecting the type of machine. Selecting the machine type will cause certain program objects such as bitmaps and servo functions to be configured automatically. Default machine configurations with the most common options for a particular machine type should be included in the Machine Group. Therefore, programming an entire standard stock order can literally be done with the push of a button. Most machines will require you to read the order to find the options included on that machine, and select those options on the appropriate tab form before building the project.

Machine Type

The machine type (Model 25, Model 33, etc.) is selected on the Machine Type form. A separate sub-directory is created under MachType for each type of machine, and the appropriate bitmaps, servo configuration files, etc for that machine will be included in that directory. All of these files will become a part of the project for the machine you are setting up. If a Group.Wiz file is included in that directory, then that will become the standard 'Machine Group' for that type of machine.

The PLC type is also selectable from this screen. The sub-directories under the directory PLC will show up as choices for PLC Type. This selection will determine the configuration used for PLC initialization.

Groups

Groups include functions for any screen or screens which should logically be grouped together, for example, Form Start and Form Stop on the Timing Screen, or X Adjust on the Adjust Screen along with X Adjust Calibrate on the Setup Screen. Groups are Group.Wiz files stored in their own sub-directory under the Groups sub-directory. Any options in that .Wiz file will be included when that group is selected.

Manual Functions

The Manual Functions tab allows you to select the manual function buttons (used for activating machine functions, etc.) normally located at the Manual Screen. Functions extracted to the Manual directory will be available as ManualButton type controls on the Manual form. The controls will be arranged in a rectangular array to fill the form. These controls may be moved on the form by the programmer later if desired.

Time Functions

The Times tab allows you to select the Timed functions (used for controlling the Start/Stop times in automatic mode, counter presets, etc.) normally located at the Timing Screen. Functions extracted to the Times directory will be available as TimeStart/TimeStop type controls on the Times form. The controls will be arranged in a rectangular array to fill the form. These controls may be moved on the form by the programmer later if desired.

Temperature Functions

The Temps tab allows you to select the basic oven layout and any other temperature control/display functions. Functions extracted to the Temps directory will be available as TempSet/TempAct type functions on the Temperature Screen. These functions will be placed in their standard position on the Temps form, and may be moved by the programmer later if desired. The Temps\Oven sub-directory will have sub-directories with the forms and controls necessary to handle the different types of basic ovens. This is really just like a group, except that the controls on the Temps form will be rearranged based on the type of oven. (This is done because the ovens exist on almost every machine, and because they typically determine the layout of the other adjustments on the machine.

Setup Functions

The Setup tab allows you to select the Setup function buttons (used for selecting machine control options, calibration, selecting printers, etc.) normally located at the Setup Screen. Functions extracted to the Setup directory will be available as SetupButton type controls on the Setup form. The controls will be arranged in a rectangular array to fill the form. These controls may be moved on the form by the programmer later if desired.

Function Buttons

The Function Buttons tab allows you to select the function buttons (used for enabling/disabling machine functions, etc.) normally located at the bottom of the screen. Functions extracted to the Buttons directory will be available as FunctionButton type controls on the Buttons form. These controls will be placed on the Buttons form. If the Pendant check box is selected, the buttons will be placed in their standard position on the form, otherwise they will be arranged automatically to fill the Buttons form. These controls may be moved on the form by the programmer later if desired.

Computer Hardware

General Computer Hardware

The standard Industrial computers used for the computer are PC compatible systems purchased from **Industrial Computer Source**. These systems currently have a passive backplane with Pentium 166 CPU 's, 16 Meg of RAM , IDE hard disk drives, 3-1/2" diskette drives, 2 serial ports (RS232C), and VGA video output. All controllers and interfaces are included on the CPU card.

Analog I/O

Analog I/O in this system is normally through the use of an analog to digital converter and analog signal multiplexers. The **Metrabyte DAS-8** and **EXP16** are normally used. An **ICS** clone of the DAS-8 called the **AIO8** can also be used. The analog inputs are used for temperature, pressure, current, and adjustments. The DAS-8 and EXP16's are connected through ribbon cables which carry both analog and digital signals. Four digital signals from the DAS-8 control the onboard multiplexers on the EXP16's' Eight analog signals from up to eight EXP16's'go back to the DAS-8, which has its own multiplexer allowing it to read from up to eight different EXP16 cards. This allows up to 128 analog inputs. Temperatures are normally read from thermocouples connected to the EXP16 cards. Cold junction compensation is accomplished by reading the cold junction block on one of the EXP16's and doing the appropriate calculation in the program. **One** of the EXP16's, therefore must have the CJC jumper set to 7. The gain setting of each EXP16 can be set differently to get the desired range of input values for each function.

Computer Diagnostics

Diagnostic programs are included in the standard setup files to help troubleshoot common computer hardware problems. **DAS8.EXE** is a program which will determine if the DAS-8 card is installed properly, and this program when run will continuously output the analog value read from the DAS-8 card.**Hard disk** problems can be tested and corrected by using the **ScanDisk** and **Defrag** utilities under System Recipes. General addressing and IRQ conflicts can be tested by going into **Control Panel** and using **System/Device Manager** and looking for exclamation points which indicate a problem with the configuration. The **Resources** tabs for each device can be used to verify the hardware addresses and interrupts used for that device. Software problems can sometimes be caused by a computer virus. Don't take this statement to mean that anytime you see something unusual happen it must be caused by a virus. In fact, this is often not the case. Still, there is a way to test for known computer viruses. A program in the DOS directory named **MSAV.EXE** is the Microsoft Anti-Virus program. This program will not necessarily detect every virus. Many viruses may have been created since the creation date of that program. It will, however, test for some of the more common virus programs.

The **modem card** communication can be easily tested by going to Control Panel and using the **Modem/Diagnostics/More** Info functions. Serial communications can be tested by using the **Terminal** application. under applications on the Start menu. Serial connections to external modems and other devices can be tested in this way by trying different baud rates and typing in the Terminal window while watching for communication LED's to flicker on the receiving device. This works best at lower baud rates. If the comm port or cable is suspect, then a breakout box can be used. HMI/Control has **Black Box RS232 breakout boxes**. We also have built **RS422 /485 breakout boxes** which can be used to test RS422/485 communications in the same way as the RS232 breakout boxes from Black Box. These breakout boxes have LED's which indicate the transmit and receive signals. The cover can be removed for accessing jumpers to set the I/O pins for that device. The LED's can also be disabled to remove them from the circuit when they aren't needed. A simple test of the communication cable can be made by connecting the transmit and receive +/- together to make a loop-back tester. When this is done, characters typed into Terminal should be echoed back to the screen.

PLC Programming

Siemens - TiSoft

Siemens PLC's are programmed from a DOS application named TiSoft . The TiSoft program is stored in the TI directory. The program is run with the TI505 command. In general, the cursor keys will move the block cursor around the program, and the action of the function keys are listed on the bottom of the screen. The new version of TiSoft allows you to go online or offline. While offline, you can select a directory and program name. This will determine the synonyms displayed when you are programming, whether you are online or offline, since the synonyms are not stored in the PLC. If a plus sign appears at the right on the bottom of the screen, that means you can select different Fkey options by pressing the spacebar.

The HMI/Control program addresses memory locations in the Siemens PLC's as Vmem locations. Timer presets are addressed as the actual timer presets. Heater timer values are written in a batch mode to the Siemens PLC, so no timer numbers between the lowest Heater timer and the highest Heater timer should be used unless it is used by the HMI/Control program itself. Otherwise, any values in the PLC will be overwritten by the HMI/Control program.

Omron - SysWin

SysWin is a Windows application for programming the Omron PLC's. Omron PLC's are usually connected to the computer by an RS232 cable connected to a Host Link module. The HMI/Control program addresses memory locations in the Omron PLC's as Channel numbers. Timer presets are addressed as the DM locations used as timer presets. Heater timer values are written in a batch mode to the Siemens PLC, so no DM locations between the lowest Heater timer number and the highest Heater timer number should be used unless it is used by the HMI/Control program itself. Otherwise, any values in the PLC will be overwritten by the HMI/Control program.

Allen Bradley - APS/AI

Both APS and AI software are DOS applications used for programming the Allen Bradley PLC's. Communication with AB PLC's is done through an RS232 connection using DF1 protocol. The HMI/Control program addresses memory locations in the AB PLC's as words in the B3 file. Timer presets are addressed as the actual timer presets or as word locations in the N5 integer file. (In that case, the integer file is used to increase the throughput since only one word is needed for each timer, but the integer file values must explicitly be moved from the integer file to the appropriate timer presets by the PLC code.) Heater timer values are written in a batch mode to the Siemens PLC, so no timers between the lowest Heater timer and the highest Heater timer should be used unless it is used by the HMI/Control program itself. Otherwise, any values in the PLC will be overwritten by the HMI/Control program.

GE/Fanuc - LM90

The LM90 software is programming system used for programming GE /Fanuc PLC's. These PLC's are connected to the computer by a RS232 interface. The HMI/Control program addresses memory locations in the GE PLC's as N locations. Timer presets are addressed as the N locations used as timer presets. Heater timer values are written in a batch mode to the Siemens PLC, so no timer numbers between the lowest Heater timer and the highest Heater timer should be used unless it is used by the HMI/Control program itself. Otherwise, any values in the PLC will be overwritten by the HMI/Control program.

Time Based Functions

Time based functions in HMI/Control machines are implemented in the PLC by taking a pair of timers in the PLC, and running those timers continuously, resetting them when the main machine cycle timer times out. The output of the function associated with those timers is normally programmed to be on when the Auto bit is true and the start timer has timed out and the stop timer has not timed out, or when the Auto bit is false and the Manual bit from the PC is set.

Temperature Control

The temperature control section of the PLC program normally has two timers associated with each zone. The timers are written to by the PC, and the PLC simply turns on the heater output if the Pulse timer has timed out and the Off timer has not. Normally, the HeatersOn bit must be true as well as a heater safety timer bit which is set when a watchdog timer times out. MoldTrol On/Off functions are programmed by simply turning on the Heat output when the Heat bit is true, and turning on the Cool output when the Cool bit is true. (Both of these bits are set directly by the computer.) Heater error bits are set by the PC when a heater error condition is sensed, and heater contactors for as many zones as needed can be turned on when those bits are false.

Adjustments

Internal PLC type adjustments normally latch an adjust function on in the PLC when the Adjust or Move bit is set by the PC. These adjust functions stay latched while there are no errors, and set stays on the same side of actual or a home input is seen. Actual value is created by adding to the actual value when a pulse is seen, or from an absolute input value and an offset. The home function normally puts zero in the actual value, or puts the absolute input value in the offset for that adjustment. The offset value can also be set by the PC.

Counters

Counter functions for pcs made, pcs formed, etc. are implemented by counting off a one shot of the function to be counted. The counters in the PLC normally need to be cascaded together. In this case, the first counter counting out should reset it and increment the second counter. The accumulated counts of those two counters can written to two consecutive words which are read by the PC for display. However the counters are implemented, the first word must have the first three decimal digits and the second must have the next three decimal digits. (In other words 999 999)

Faults

Fault conditions sensed by the PLC should set the fault bit for that condition and latch itself until the Fault Reset bit is true.

Servo Systems

Servo systems can be tested by breaking their operation down into the most basic functions. The servo systems used in these machines consist of a motor, a drive, a motion controller, and a position feedback device (encoder). Sometimes the position feedback device is an artificially generated encoder signal from the drive, and sometimes the motion controller is not actually a separate piece of equipment but is integrated into the drive as is the case with the Indramat CLC card. The functions performed by the motor, drive, motion controller, and encoder are still distinct functions, however, which can be isolated from each other for troubleshooting purposes. When there is trouble with the servo system, the first thing to test is that the drive can control the motor. This can usually be tested directly from the controls on the drive without controlling from the motion control card. If this function doesn't work, there's no sense looking at the motion controller.

Once the drive controls the motor, it should be determined that the drive can respond to a command signal. This is usually a digital command signal to the drive from the motion controller. Each motion controller has a function to simulate what is called an "Open Loop" control signal. On the Delta Tau MCC and PMAC cards the command is "**O**". The open loop command can be used to test the response of the drive to known external commands without being affected by the encoder . This response will vary based on whether the drive is in speed mode or torque mode. A drive running in speed mode will run at a constant speed when given the open loop command. A drive running in torque mode will go at a constant torque. This may cause the motor to increase speed very quickly, so a very small open loop command should be tried on these systems. If the drive is responding OK , the next thing to do is test the position feedback (encoder). This can be done by moving the servo axis and watching the display on the motion control software to verify that the position display is changing in the right direction. Finally, after the feedback is verified, the motion controller can be put in closed loop mode, and the jog command can be executed (**J** on MCC, **J+** on PMAC). As long as jog works, motion control card may need to be tuned to the servo system's physical response, and the automatic mode moves can be tested.

Profiles

Different speed profiles can be programmed into the HMI/Control machines to program the motion control card for the incremental moves needed to index the material through the machine. These motion programming code for these different profiles differ for various motion control systems. The main purpose of these motion profiles is to smoothly accelerate the material to the desired speed, travel at that speed, and smoothly decelerate from that speed back down to a stop. The standard acceleration curves used for different ranges of index lengths are all modified "S" curves. These are called "S" curves because if the speed is graphed, the beginning of the graph looks like an "S". We called these particular curves modified "S" curves because the beginning of the curve does not look exactly like and "S", but more like the traditional trapezoidal speed graph with the corners of the trapezoid rounded off.

Delta Tau MCC

The modified "S" curve is programmed into the Delta Tau MCC card by using the time mode function along with the constant feedrate function. Time mode is programmed into this card by using the "T" command. Each move is an incremental move performed during the specified time in milliseconds. After the time mode moves are finished, the servo is programmed to travel at a constant speed using the "**F**" command. Finally, more time mode moves are executed until we have decelerated back down to a stop. In programming the MCC in this way, the important things to keep in mind are that the time mode moves must smoothly change from zero up to the top programmed speed. That top programmed speed must be the same speed specified in the constant speed moves by the "**F**" command. The final time mode section must decelerate down from this speed to a stop at the end of the program. The speed specified by "**I06**" times the "**F**" value divided by 1000 should be approximately 9% higher than the largest time mode move divided by the number of seconds specified in the "**T**" command.

Delta Tau PMAC

The Delta Tau PMAC card has not only the motion programs to be loaded, but also can have various parameters and PLC programs. Any standard programs or values to be entered into the PMAC card at startup can be placed in a PMAC Initialization file. The HMI/Control application can handle both PMAC-PC's and serial PMAC's. Programming a modified "S" curve into a Delta Tau PMAC card is considerably more simple than it is with the MCC card. The TA and TS commands can be used to simply specify the time used for the acceleration up to programmed speed, and the "S" portion of that time respectively.

Indramat CLC

The Indramat CLC system must be programmed using Indramat's **Visual Motion** software. This program is a Windows application, and should be a program named **CLC.EXE** in a directory named **Indramat**. A program must be loaded into the Indramat that matches the servo configuration of the machine. The "S" curve is implemented in the Indramat CLC system by programming a simple trapezoidal move into the **CLC**, and setting the **Bipolar Jerk Limit** to a value which will limit the jerk (second derivative of the speed function). The units of the **Bipolar Jerk Limit** parameter are not clearly defined, and the action of this parameter is affected by other values in the Indramat system. These values, therefore, should not be changed without good reason. Communication with the CLC card is done through a serial port.

Modems and Remote Access

Modem Hardware and Configuration

HMI/Control machines normally include an internal modem through which we can remotely operate the customer's computer. This allows us to modify configuration files, program the PLC, perform diagnostic functions, or download a new program, all without leaving HMI/Control. This will not happen, however, if you don't install and test the modem and remote control software properly. As part of building every computer, the modem must be tested. The most common modem address used with most modems is port address **3E8**H and **IRQ5**. These addresses should be used whenever possible to avoid confusion on the part of HMI/Control service people and customers. The modem test procedures described earlier will test that the modem is installed, but they won't verify that the modem will actually work over the phone line. When using an internal modem which appears like a serial port and uses the Hayes AT command set, the logic functions as well as the hardware appearance of the modem can be tested by going into the **Terminal** program, and typing the "**AT**" command. The modem should respond with "**OK**". For any modem, the only way to actually test the modem's operation completely is to hook it up to a phone line and communicate with it.

PCAnywhere

The software we use to communicate with the machines through these modems is **pcANYWHERE**. When pcANYWHERE is installed properly, it will appear as an icon on the taskbar, waiting for an incoming call. This is what's known as **Host** mode. That computer which was dialed into is the Host computer. The computer dialing in to operate the Host remotely is called the **Remote** computer. Most functions can be executed by the remote on the host as if the operator of the Remote computer were sitting at the keyboard of the host computer. There are also two important functions available to the remote operator. The **Chat** function allows the remote operator to type messages back and forth with the customer at the Host computer. The **File Transfer** function allows files to be transferred back and forth almost as if you were running File Manager or Explorer. These functions are both available from the control menu. The control menu is accessed by clicking on the icon in the top left of the title bar of your application window. (In this case, pcANYWHERE).

HMI/Control is a control programming package available on spec. If you would like to know more about our HMI and control solutions, or are interested in us quoting a similar system tailored to your needs, contact us at info@swe-eng.com We'll be glad to talk work you.

[Back]

Send mail to webmaster@swe-eng.com with questions or comments about this web site.
Please read our Usage Rules before using any information from this site.

Last modified: June 10, 2002