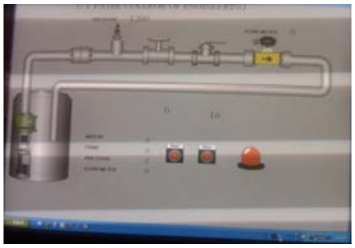
9. Introduction To SCADA System



9. INTRODUCTION TO SCADA SYSTEM

9.1 INTRODUCTION

SCADA is “Supervisory Control and Data Acquisition”. Real-time industrial process control systems used to centrally monitor and control remote or local industrial equipment such as motors, valves, pumps, relays, sensors, etc. SCADA is Combination of telemetry and Data Acquisition.

Previously without SCADA software, an industrial process was entirely controlled by PLC, CNC, PID & micro controllers having programmed in certain languages or codes. These codes were either written in assembly language or relay logic without any true animation that would explain the process running. It is always easy to understand the status of the process if it is shown with some animations rather then written codes. Hence SCADA software came to existence and with some exclusive features it became internal part of automation system.

SCADA is not just hardware but also software. It’s a concept. It’s a system as a combination of special hardware, software and protocols. SCADA is used to control chemical plant processes, oil and gas pipelines, electrical generation and transmission equipment, manufacturing facilities, water purification and distribution infrastructure, etc. For example, in a SCADA system a PLC can be used to control the flow of cooling water as part of an industrial process. At the same time the supervisor can use the Host control function to set the temperature for the flow of water. It can also have alarms and can record the flow of water temperature and report back to The RTUs and PLCs are responsible for data collection such as meter readings, equipment status etc and communicate back to the SCADA system. This data can be stored in a database for later analysis or monitored by a supervisor to take appropriate actions if required.

SCADA systems typically implement a distributed database, commonly referred to as a tag database, which contains data elements called tags or points. A point represents a single input or output value monitored or controlled by the system. Points can be either "hard" or "soft". A hard point is representative of an actual input or output connected to the system, while a soft point represents the result of logic and math operations applied to other hard and soft points. Most implementations conceptually remove this distinction by making every property a "soft" point (expression) that can equal a single "hard" point in the simplest case. Point values are normally stored as value-timestamp combinations; the value and the timestamp when the value was recorded or calculated. A series of value-timestamp combinations is the history of that point. It's also common to store additional metadata with tags such as: path to field device and PLC register, design time comments, and even alarming information.

9.1.1 WHAT SCADA CAN DO FOR YOU?

SCADA is not a specific technology, but a type of application. SCADA stands for Supervisory Control and Data Acquisition — any application that gets data about a system in order to control that system is a SCADA application.

A SCADA application has two elements:

1. The process/system/machinery you want to monitor a control — this can be a power plant, a water system, a network, a system of traffic lights, or anything else.
2. A network of intelligent devices that interfaces with the first system through sensors and control outputs. This network, which is the SCADA system, gives you the ability to measure and control specific elements of the first system.

You can build a SCADA system using several different kinds of technologies and protocols. This white paper will help you evaluate your options and decide what kind of SCADA system is best for your needs.

9.1.2 WHERE IS SCADA USED?

You can use SCADA to manage any kind of equipment. Typically, SCADA systems are used to automate complex industrial processes where human control is impractical — systems where there are more control factors, and more fast-moving control factors, than human beings can comfortably manage.

Around the world, SCADA systems control:

- **Electric power generation, transmission and distribution:** Electric utilities use SCADA systems to detect current flow and line voltage, to monitor the operation of circuit breakers, and to take sections of the power grid online or offline.

- **Water and sewage:** State and municipal water utilities use SCADA to monitor and regulate water flow, reservoir levels, pipe pressure and other factors.

- **Buildings, facilities and environments:** Facility managers use SCADA to control HVAC, refrigeration units, lighting and entry systems.

- **Manufacturing:** SCADA systems manage parts inventories for just-in-time manufacturing, regulate industrial automation and robots, and monitor process and quality control.

- **Mass transit:** Transit authorities use SCADA to regulate electricity to subways, trams and trolley buses; to automate traffic signals for rail systems; to track and locate trains and buses; and to control railroad crossing gates.

- **Traffic signals:** SCADA regulates traffic lights, controls traffic flow and detects out-of-order signals.

As I'm sure you can imagine, this very short list barely hints at all the potential applications for SCADA systems. SCADA is used in nearly every industry and public infrastructure project — anywhere where automation increases efficiency.

What's more, these examples don't show how deep and complex SCADA data can be. In every industry, managers need to control multiple factors and the interactions between those factors. SCADA systems provide the sensing capabilities and the computational power to track everything that's relevant to your operations.

9.1.3 HOW DOES A SCADA SYSTEM WORK?

A SCADA system performs four functions:

1. Data acquisition
2. Networked data communication
3. Data presentation
4. Control

These functions are performed by four kinds of SCADA components:

1. **Sensors** (either digital or analog) and **control relays** that directly interface with the managed system.

2. **Remote telemetry units (RTUs).** These are small computerized units deployed in the field at specific sites and locations. RTUs serve as local collection points for gathering reports from sensors and delivering commands to control relays.
3. **SCADA master units.** These are larger computer consoles that serve as the central processor for the SCADA system. Master units provide a human interface to the system and automatically regulate the managed system in response to sensor inputs.
4. The **communications network** that connects the SCADA master unit to the RTUs in the field.

9.1.4 The World's Simplest SCADA System

The simplest possible SCADA system would be a single circuit that notifies you of one event. Imagine a fabrication machine that produces widgets. Every time the machine finishes a widget, it activates a switch. The switch turns on a light on a panel, which tells a human operator that a widget has been completed.

Obviously, a real SCADA system does more than this simple model. But the principle is the same. A full-scale SCADA system just monitors more stuff over greater distances.

Let's look at what is added to our simple model to create a full-scale SCADA system:

9.2 Data Acquisition

First, the systems you need to monitor are much more complex than just one machine with one output. So a real-life SCADA system needs to monitor hundreds or thousands of sensors. Some sensors measure inputs into the system (for example, water flowing into a reservoir), and some sensors measure outputs (like valve pressure as water is released from the reservoir).

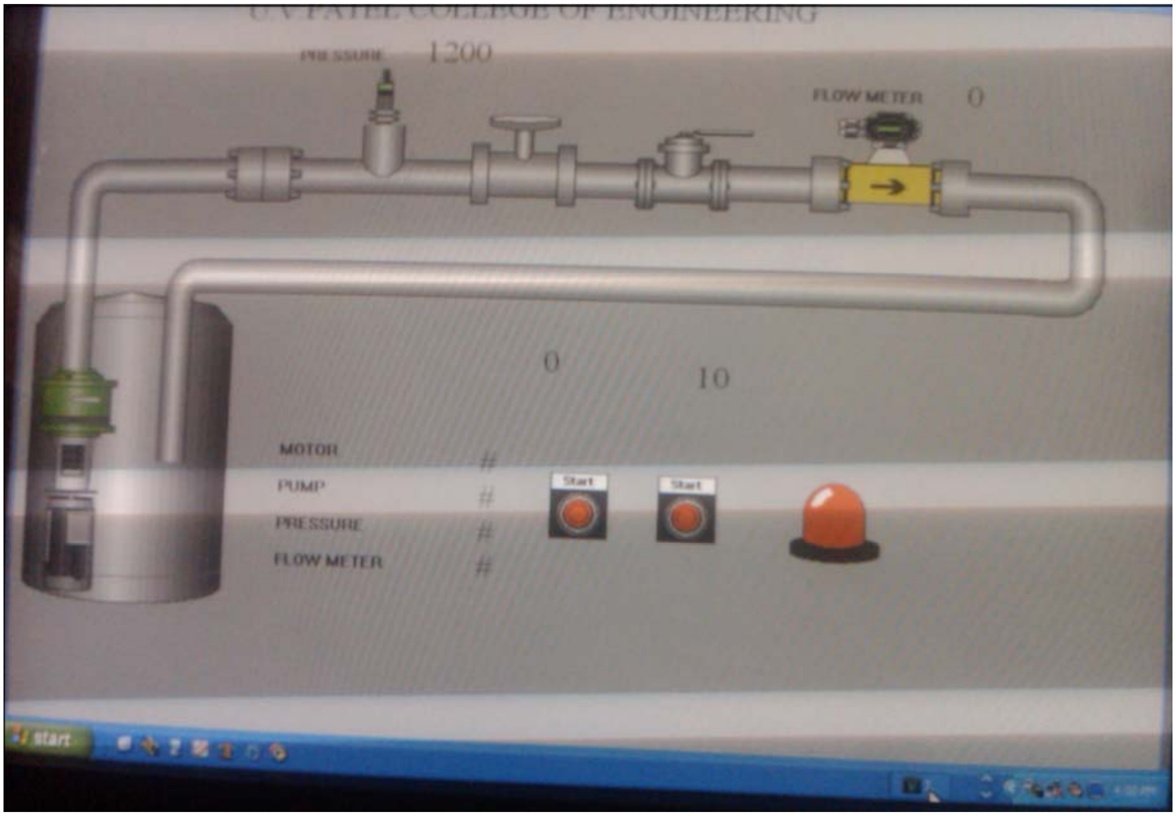


FIG 9.1: ACTUAL DEVELOPMENT OF SCADA SYSYTEM

Some of those sensors measure simple events that can be detected by a straightforward on/off switch, called a discrete input (or digital input). For example, in our simple model of the widget fabricator, the switch that turns on the light would be a discrete input. In real life, discrete inputs are used to measure simple states, like whether equipment is on or off, or tripwire alarms, like a power failure at a critical facility.

Some sensors measure more complex situations where exact measurement is important. These are analog sensors, which can detect continuous changes in a voltage or current input. Analog sensors are used to track fluid levels in tanks, voltage levels in batteries, temperature and other factors that can be measured in a continuous range of input.

For most analog factors, there is a normal range defined by a bottom and top level. For example, you may want the temperature in a server room to stay between 60 and 85 degrees Fahrenheit. If the temperature goes above or below this range, it will trigger a threshold alarm. In more advanced systems, there are four threshold alarms for analog sensors, defining Major Under, Minor Under, Minor Over and Major Over alarms.

9.3 DATA COMMUNICATION

In our simple model of the widget fabricator, the “network” is just the wire leading from the switch to the panel light. In real life, you want to be able to monitor multiple systems from a central location, so you need a communications network to transport all the data collected from your sensors.

Early SCADA networks communicated over radio, modem or dedicated serial lines. Today the trend is to put SCADA data on Ethernet and IP over SONET. For security reasons, SCADA data should be kept on closed LAN/WANs without exposing sensitive data to the open Internet.

Real SCADA systems don’t communicate with just simple electrical signals, either. SCADA data is encoded in protocol format. Older SCADA systems depended on closed proprietary protocols, but today the trend is to open, standard protocols and protocol mediation.

Sensors and control relays are very simple electric devices that can’t generate or interpret protocol communication on their own. Therefore the remote telemetry unit (RTU) is needed to provide an interface between the sensors and the SCADA network. The RTU encodes sensor inputs into protocol format and forwards them to the SCADA master; in turn, the RTU receives control commands in protocol format from the master and transmits electrical signals to the appropriate control relays.

9.4 DATA PRESENTATION

The only display element in our model SCADA system is the light that comes on when the switch is activated. This obviously won’t do on a large scale — you can’t track a light board of a thousand separate lights, and you don’t want to pay someone simply to watch a light board, either.

A real SCADA system reports to human operators over a specialized computer that is variously called a master station, an HMI (Human-Machine Interface) or an HCI (Human-Computer Interface).

The SCADA master station has several different functions. The master continuously monitors all sensors and alerts the operator when there is an “alarm” — that is, when a control factor is operating outside what is defined as its normal operation. The master presents a comprehensive view of the entire managed system, and presents more detail in response to user requests. The master also performs data processing on information gathered from sensors — it maintains report logs and summarizes historical trends.

An advanced SCADA master can add a great deal of intelligence and automation to your systems management, making your job much easier.

9.5 CONTROL

Unfortunately, our miniature SCADA system monitoring the widget fabricator doesn’t include any control elements. So let’s add one. Let’s say the human operator also has a button on his control panel. When he presses the button, it activates a switch on the widget fabricator that brings more widget parts into the fabricator.

Now let’s add the full computerized control of a SCADA master unit that controls the entire factory. You now have a control system that responds to inputs elsewhere in the system. If the machines that make widget parts break down, you can slow down or stop the widget fabricator. If the part fabricators are running efficiently, you can speed up the widget fabricator.

If you have a sufficiently sophisticated master unit, these controls can run completely automatically, without the need for human intervention. Of course, you can still manually override the automatic controls from the master station.

In real life, SCADA systems automatically regulate all kinds of industrial processes. For example, if too much pressure is building up in a gas pipeline, the SCADA system can automatically open a release valve. Electricity production can be adjusted to meet demands on the power grid. Even these real-world examples are simplified; a full-scale SCADA system can adjust the managed system in response to multiple inputs.

9.6 ARCHITECTURE

In this section we are going to details which describe the common architecture required for the SCADA products.

9.6.1 HARDWARE ARCHITECTURE

The basic hardware of the SCADA system is distinguished into two basic layers: the "client layer" which caters for the man machine interaction and the "data server layer" which handles most of the process data control activities. The data servers communicate with devices in the field through process controllers. Process controllers, e.g. PLC’s, are connected to the data servers either directly or via networks or field buses that are proprietary (e.g. Siemens H1), or non-proprietary (e.g. Profibus). Data servers are connected to each other and to client stations via an Ethernet LAN. Fig.1. shows typical hardware architecture.

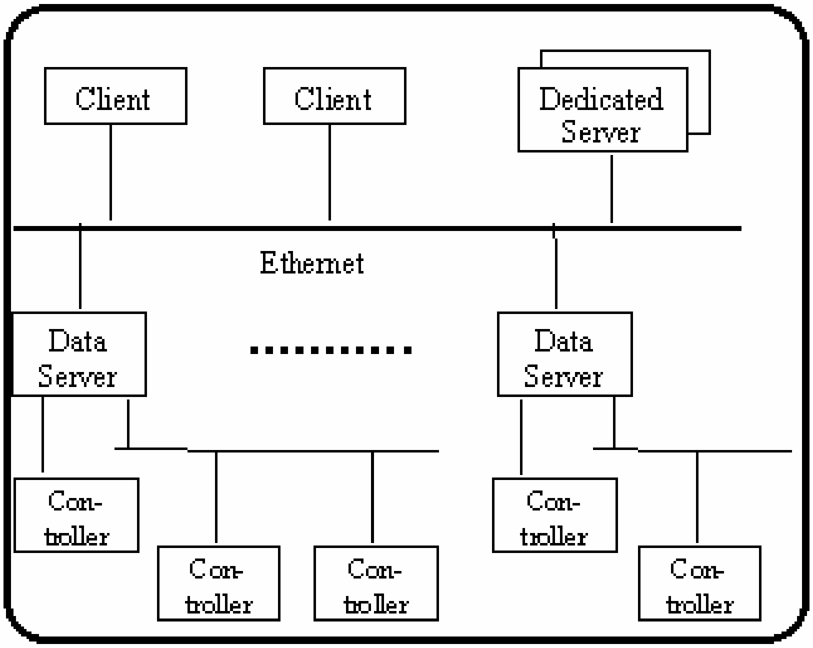


FIGURE 9.2: TYPICAL HARDWARE ARCHITECTURE

9.6.2 SOFTWARE ARCHITECTURE

The SCADA products are multi-tasking and are based upon a real-time database (RTDB) located in one or more servers. Servers are responsible for data acquisition and handling like polling controllers, alarm checking, calculations, logging and archiving) on a set of parameters, typically to which those are connected.

However, it is possible to have dedicated servers for particular tasks, e.g. historian, data logger, alarm handler. Fig. 2 shows a SCADA architecture that is generic for the product.

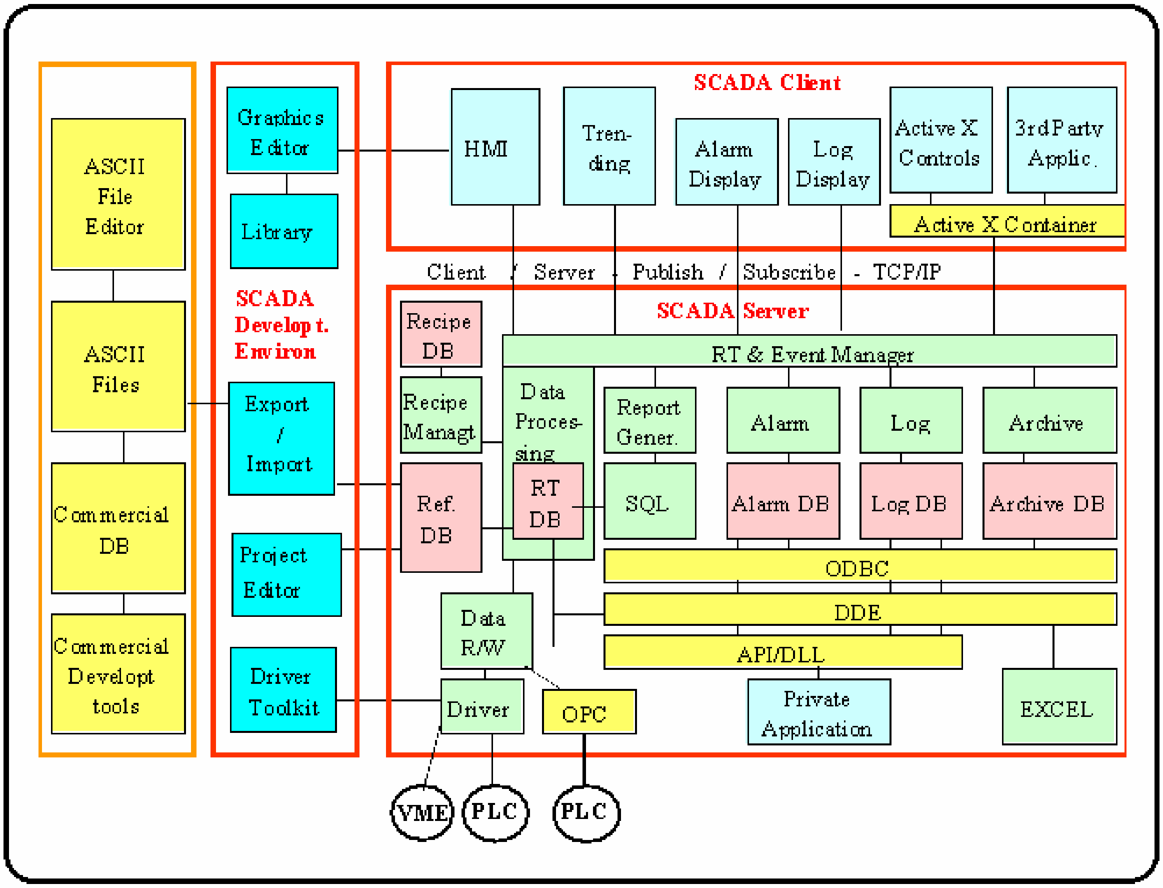


FIGURE 9.3: GENERIC SOFTWARE ARCHITECTURE

9.7 COMMUNICATION

9.7.1 INTERNAL COMMUNICATION

Server-client and server-server communication is in general on a publish-subscribe and event-driven basis and uses a TCP/IP protocol, i.e., a client application subscribes to a parameter which is owned by a particular server application and only changes to that parameter are then communicated to the client application.

9.7.2 ACCESS TO DEVICES

The data servers poll the controllers at a user defined polling rate. The polling rate may be different for different parameters. The controllers pass the requested parameters to the data servers. Time stamping of the process parameters is typically performed in the controllers and this time-stamp is taken over by the data server. If the controller and communication protocol used support unsolicited data transfer then the products will support this too.

The products provide communication drivers for most of the common PLCs and widely used field-buses, e.g., Modbus. Of the three field buses that are recommended are, both Profibus and Worldfip are supported but CANbus often not. Some of the drivers are based on third party products (e.g., Applicom cards) and therefore have additional cost associated with them. VME on the other hand is generally not supported.

A single data server can support multiple communications protocols; it can generally support as many such protocols as it has slots for interface cards. The effort required to develop new drivers is typically in the range of 2-6 weeks depending on the complexity and similarity with existing drivers, and a driver development toolkit is provided for this.

9.8 CLASSIFICATION BASED ON GENERATION

SCADA systems have evolved through 3 generations as follows:

- **FIRST GENERATION: "MONOLITHIC"**

  In the first generation, computing was done by mainframe systems. Networks didn’t exist at the time SCADA was developed. Thus SCADA systems were independent systems with no connectivity to other systems. Wide Area Networks were later designed by RTU vendors to communicate with the RTU. The communication protocols used were often proprietary at that time. The first-generation SCADA system was redundant since a back-up mainframe system was connected at the bus level and was used in the event of failure of the primary mainframe system.

- **SECOND GENERATION: "DISTRIBUTED"**

The processing was distributed across multiple stations which were connected through a LAN and they shared information in real time. Each station was responsible for a particular task thus making the size and cost of each station less than the one used in First Generation. The network protocols used were still mostly proprietary, which led to significant security problems for any SCADA system that received attention from a hacker. Since the protocols were proprietary, very few people beyond the developers and hackers knew enough to determine how secure a SCADA installation was. Since both parties had vested interests in keeping security issues quiet, the security of a SCADA installation was often badly overestimated, if it was considered at all.

- **THIRD GENERATION: "NETWORKED"**

These are the current generation SCADA systems which use open system architecture rather than a vendor-controlled proprietary environment. The SCADA system utilizes open standards and protocols, thus distributing functionality across a WAN rather than a LAN. It is easier to connect third party peripheral devices like printers, disk drives, and tape drives due to the use of open architecture. WAN protocols such as Internet Protocol (IP) are used for communication between the master station and communications equipment. Due to the usage of standard protocols and the fact that many networked SCADA systems are accessible from the Internet, the systems are potentially vulnerable to remote cyber-attacks. On the other hand, the usage of standard protocols and security techniques means that standard security improvements are applicable to the SCADA systems, assuming they receive timely maintenance and updates.

9.9 HUMAN MACHINE INTERFACE

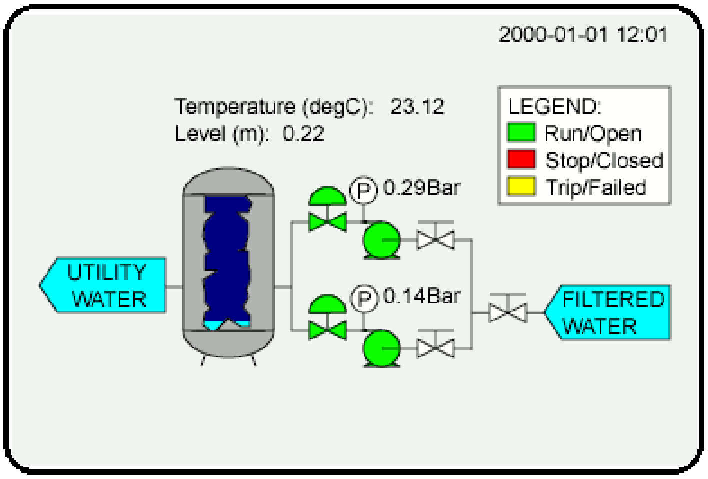


FIGURE 9.4: TYPICAL BASIC SCADA ANIMATIONS

A Human-Machine Interface or HMI is the apparatus which presents process data to a human operator, and through which the human operator controls the process.

An HMI is usually linked to the SCADA system's databases and software programs, to provide trending, diagnostic data, and management information such as scheduled maintenance procedures, logistic information, detailed schematics for a particular sensor or machine, and expert-system troubleshooting guides.

The HMI system usually presents the information to the operating personnel graphically, in the form of a mimic diagram. This means that the operator can see a schematic representation of the plant being controlled. For example, a picture of a pump connected to a pipe can show the operator that the pump is running and how much fluid it is pumping through the pipe at the moment. The operator can then switch the pump off. The HMI software will show the flow rate of the fluid in the pipe decrease in real time. Mimic diagrams may consist of line graphics and schematic symbols to represent process elements, or may consist of digital photographs of the process equipment overlain with animated symbols.

The HMI package for the SCADA system typically includes a drawing program that the operators or system maintenance personnel use to change the way these points are represented in the interface. These representations can be as simple as an on-screen traffic light, which represents the state of an actual traffic light in the field, or as complex as a multi-projector display representing the position of all of the elevators in a skyscraper or all of the trains on a railway.

An important part of most SCADA implementations is alarm handling. The system monitors whether certain alarm conditions are satisfied, to determine when an alarm event has occurred. Once an alarm event has been detected, one or more actions are taken (such as the activation of one or more alarm indicators, and perhaps the generation of email or text messages so that management or remote SCADA operators are informed). In many cases, a SCADA operator may have to acknowledge the alarm event; this may deactivate some alarm indicators, whereas other indicators remain active until the alarm conditions are cleared. Alarm conditions can be explicit - for example, an alarm point is a digital status point that has either the value NORMAL or ALARM that is calculated by a formula based on the values in other analogue and digital points - or implicit: the SCADA system might automatically monitor whether the value in an analogue point lies outside high and low limit values associated with that point. Examples of alarm indicators include a siren, a pop-up box on a screen, or a colored or flashing area on a screen (that might act in a similar way to the "fuel tank empty" light in a car); in each case, the role of the alarm indicator is to draw the operator's attention to the part of the system 'in alarm' so that appropriate action can be taken. In designing SCADA systems, care is needed in coping with a cascade of alarm events occurring in a short time, otherwise the underlying cause (which might not be the earliest event detected) may get lost in the noise. Unfortunately, when used as a noun, the word 'alarm' is used rather loosely in the industry; thus, depending on context it might mean an alarm point, an alarm indicator, or an alarm event.

10. Sensors

Aquamist
100
200
300
400 ml/m
FLOW
S
B
-W. INJECTION
-WATER LEVEL
SC
WL
WH
Made in England
AQUAMIST

10. SENSORS

10.1 FLOW SENSOR

FIG 10.1: FLOW SENSOR

A **flow sensor** is a device for sensing the rate of fluid flow. Typically a flow sensor is the sensing element used in a flow meter, or flow logger, to record the flow of fluids. As is true for all sensors, absolute accuracy of a measurement requires functionality for calibration.

There are various kinds of flow sensors and flow meters, including some that have a vane that is pushed by the fluid, and can drive a rotary potentiometer, or similar device.

Other flow sensors are based on sensors which measure the transfer of heat caused by the moving medium. This principle is common for microsensors to measure flow.

Flow meters are related to devices called velocimeters that measure velocity of fluids flowing through them. Laser-based interferometry is often used for air flow measurement, but for liquids, it is often easier to measure the flow. Another approach is Doppler-based methods for flow measurement. Hall effect sensors may also be used, on a flapper valve, or vane, to sense the position of the vane, as displaced by fluid flow.

10.2 PRESSURE SENSOR

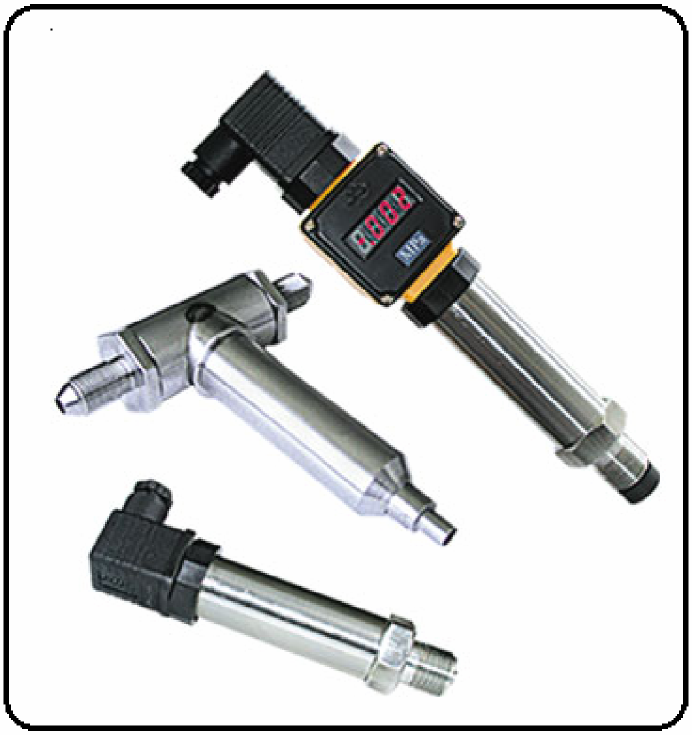


FIG 10.2: DIGITAL AIR PRESSURE SENSOR

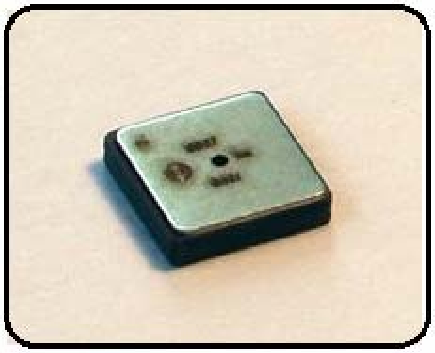


FIG 10.3: COMPACT DIGITAL BAROMETRIC PRESSURE SENSOR

A **pressure sensor** measures pressure, typically of gases or liquids. Pressure is an expression of the force required to stop a fluid from expanding, and is usually stated in terms of force per unit area. A pressure sensor usually acts as a transducer; it generates a signal as a function of the pressure imposed. For the purposes of this article, such a signal is electrical.

Pressure sensors are used for control and monitoring in thousands of everyday applications. Pressure sensors can also be used to indirectly measure other variables such as fluid/gas flow, speed, water level, and altitude. Pressure sensors can alternatively be called pressure transducers, pressure transmitters, pressure senders, pressure indicators and piezometers, manometers, among other names.

Pressure sensors can vary drastically in technology, design, performance, application suitability and cost. A conservative estimate would be that there may be over 50 technologies and at least 300 companies making pressure sensors worldwide.

There is also a category of pressure sensors that are designed to measure in a dynamic mode for capturing very high speed changes in pressure. Example applications for this type of sensor would be in the measuring of combustion pressure in an engine cylinder or in a gas turbine. These sensors are commonly manufactured out of piezoelectric materials such as quartz.

Some pressure sensors, such as those found in some traffic enforcement cameras, function in a binary (on/off) manner, i.e., when pressure is applied to a pressure sensor, the sensor acts to complete or break an electrical circuit. These types of sensors are also known as a pressure switch.

10.2.1 TYPES OF PRESSURE MEASUREMENTS

FIG 10.4: SILICON PIEZORESISTIVE PRESSURE SENSORS

Pressure sensors can be classified in term of pressure ranges they measure, temperature ranges of operation, and most importantly the type of pressure they measure. In terms of pressure type, pressure sensors can be divided into five categories:

- **ABSOLUTE PRESSURE SENSOR**

  This sensor measures the pressure relative to perfect vacuum pressure (0 PSI or no pressure). Atmospheric pressure, is 101.325 kPa (12.7 PSI) at sea level with reference to vacuum.

- **Gauge pressure sensor**

This sensor is used in different applications because it can be calibrated to measure the pressure relative to a given atmospheric pressure at a given location. A tire pressure gauge is an example of gauge pressure indication. When the tire pressure gauge reads 0 PSI, there is really 12.7 PSI (atmospheric pressure) in the tire.

- **Vacuum pressure sensor**

This sensor is used to measure pressure less than the atmospheric pressure at a given location. This has the potential to cause some confusion as industry may refer to a vacuum sensor as one which is referenced to either atmospheric pressure (ie measure Negative gauge pressure) or relative to absolute vacuum.

- **Differential pressure sensor**

This sensor measures the difference between two or more pressures introduced as inputs to the sensing unit, for example, measuring the pressure drop across an oil filter. Differential pressure is also used to measure flow or level in pressurized vessels.

- **Sealed pressure sensor**

This sensor is the same as the gauge pressure sensor except that it is previously calibrated by manufacturers to measure pressure relative to sea level pressure.

10.3 PRESSURE SENSING TECHNOLOGY

There are two basic categories of analog pressure sensors.

Force Collector Types These types of electronic pressure sensors generally use a force collector (such a diaphragm, piston, bourdon tube, or bellows) to measure strain (or deflection) due to applied force (pressure) over an area.

- **PIEZORESISTIVE STRAIN GAGE**

Uses the piezoresistive effect of bonded or formed strain gages to detect strain due to applied pressure. Common technology types are Silicon (Monocrystalline), Polysilicon Thin Film, Bonded Metal Foil, Thick Film, and Sputtered Thin Film. Generally, the strain gauges are connected to form a Wheatstone bridge circuit to maximize the output of the sensor. This is the most commonly employed sensing technology for general purpose pressure measurement. Generally, these technologies are suited to measure absolute, gauge, vacuum, and differential pressures.

- **CAPACITIVE**

Uses a diaghragm and pressure cavity to create a variable capacitor to detect strain due to applied pressure. Common technologies use metal, ceramic, and silicon diaphragms. Generally, these technologies are most applied to low pressures (Absolute, Differential and Gauge)

- **ELECTROMAGNETIC**

Measures the displacement of a diaphragm by means of changes in inductance (reluctance), LVDT, Hall Effect, or by eddy current principal.

- **PIEZOELECTRIC**

Uses the piezoelectric effect in certain materials such as quartz to measure the strain upon the sensing mechanism due to pressure. This technology is commonly employed for the measurement of highly dynamic pressures.

- **OPTICAL**

Uses the physical change of an optical fiber to detect strain due to applied pressure. A common example of this type utilizes Fiber Bragg Gratings. This technology is employed in challenging applications where the measurement may be highly remote, under high temperature, or may benefit from the technologies inherent immunity to electromagnetic interference.

- **POTENTIOMETRIC**

Uses the motion of a wiper along a resistive mechanism to detect the strain caused by applied pressure.

10.4 APPLICATIONS

There are many applications for pressure sensors:

- **PRESSURE SENSING**

This is the direct use of pressure sensors to measure pressure. This is useful in weather instrumentation, aircraft, cars, and any other machinery that has pressure functionality implemented.

- **ALTITUDE SENSING**

This is useful in aircraft, rockets, satellites, weather balloons, and many other applications. All these applications make use of the relationship between changes in pressure relative to the altitude. This relationship is governed by the following equation:

$$h = (1 - (P/P_{ref})^{0.190284}) \times 145366.45ft$$

This equation is calibrated for an altimeter, up to 36,090 feet (9,000 m). Outside that range, an error will be introduced which can be calculated differently for each different pressure sensor. These error calculations will factor in the error introduced by the change in temperature as we go up.

Barometric pressure sensors can have an altitude resolution of less than 1 meter, which is significantly better than GPS systems (about 20 meters altitude resolution). In navigation applications altimeters are used to distinguish between stacked road levels for car navigation and floor levels in buildings for pedestrian navigation.

- **FLOW SENSING**

This is the use of pressure sensors in conjunction with the venturi effect to measure flow. Differential pressure is measured between two segments of a venturi tube that have a different aperture. The pressure difference between the two segments is directly proportional to the flow rate through the venturi tube. A low pressure sensor is almost always required as the pressure difference is relatively small.

- **LEVEL / DEPTH SENSING**

A pressure sensor may also be used to calculate the level of a fluid. This technique is commonly employed to measure the depth of a submerged body (such as a diver or submarine), or level of contents in a tank (such as in a water tower). For most practical purposes, fluid level is directly proportional to pressure. In the case of fresh water where the contents are under atmospheric pressure, 1psi = 27.7 inH20 / 1Pa = 9.81 mmH20. The basic equation for such a measurement is

$$P = p * g * h$$

Where ,

P = Pressure,

P = Density of the Fluid,

g = Standard Gravity,

h = Height of fluid column above pressure sensor

- **LEAK TESTING**

A pressure sensor may be used to sense the decay of pressure due to a system leak. This is commonly done by either comparison to a known leak using differential pressure, or by means of utilizing the pressure sensor to measure pressure change over time.

10.5 REMOTE TERMINAL UNIT

An RTU, or Remote Terminal Unit is a microprocessor controlled electronic device which interfaces objects in the physical world to a distributed control system or SCADA system by transmitting telemetry data to the system and/or altering the state of connected objects based on control messages received from the system.

10.5.1 What to Look for in a SCADA RTU?

Your SCADA RTUs need to communicate with all your on-site equipment and survive under the harsh conditions of an industrial environment. Here’s a checklist of things you should expect from a quality RTU:

- **Sufficient capacities** to support the equipment at your site … but not more capacity than you actually will use. At every site, you want an RTU that can support your expected growth over a reasonable period of time, but it’s simply wasteful to spend your budget on excess capacity that you won’t use.

- **Rugged construction** and ability to withstand extremes of temperature and humidity. You know how punishing on equipment your sites can be. Keep in mind that your SCADA system needs to be the most reliable element in your facility.

- **Secure redundant power supply**. You need your SCADA system up and working 24/7, no excuses. Your RTU should support battery power and, ideally, two power inputs.

- **Redundant communication ports**. Network connectivity is as important to SCADA operations as a power supply. A secondary serial port or internal modem will keep your RTU online even if the LAN fails. Plus, RTUs with multiple communication ports easily support a LAN migration strategy.

- **Nonvolatile memory (NVRAM)** for storing software and/or firmware. NVRAM retains data even when power is lost. New firmware can be easily downloaded to NVRAM storage, often over LAN — so you can keep your RTUs’ capabilities up to date without excessive site visits.

- **Intelligent control.** As I noted above, sophisticated SCADA remotes can control local systems by themselves according to programmed responses to sensor inputs. This isn't necessary for every application, but it does come in handy for some users.

- **Real-time clock** for accurate date/time stamping of reports.

- **Watchdog timer** to ensure that the RTU restarts after a power failure.

11. Programmable Logic Controller

11. PROGRAMMABLE LOGIC CONTROLLER

11.1 INTRODUCTION

FIG 11.1: PLC & INPUT/OUTPUT ARRANGEMENTS

A programmable logic controller (PLC) or programmable controller is a digital computer used for automation of electromechanical processes, such as control of machinery on factory assembly lines, amusement rides, or lighting fixtures. PLCs are used in many industries and machines. Unlike general-purpose computers, the PLC is designed for multiple inputs and output arrangements, extended temperature ranges, immunity to electrical noise, and resistance to vibration and impact. Programs to control machine operation are typically stored in battery-backed or non-volatile memory. A PLC is an example of a real time system since output results must be produced in response to input conditions within a bounded time, otherwise unintended operation will result.

11.2 PROGRAMMING

PLC programs are typically written in a special application on a personal computer, then downloaded by a direct-connection cable or over a network to the PLC. The program is stored in the PLC either in battery-backed-up RAM or some other non-volatile flash memory. Often, a single PLC can be programmed to replace thousands of relays.

Under the IEC 6931-3 standard, PLCs can be programmed using standards-based programming languages. A graphical programming notation called Sequential Function Charts is available on certain programmable controllers. Initially most PLCs utilized Ladder Logic Diagram Programming, a model which emulated electromechanical control panel devices (such as the contact and coils of relays) which PLCs replaced. This model remains common today.

IEC 6931-3 currently defines five programming languages for programmable control systems: FBD (Function block diagram), LD (Ladder diagram), ST (Structured text, similar to the Pascal programming language), IL (Instruction list, similar to assembly language) and SFC (Sequential function chart). These techniques emphasize logical organization of operations.

While the fundamental concepts of PLC programming are common to all manufacturers, differences in I/O addressing, memory organization and instruction sets mean that PLC programs are never perfectly interchangeable between different makers. Even within the same product line of a single manufacturer, different models may not be directly compatible.

11.2.1 LADDER LOGIC

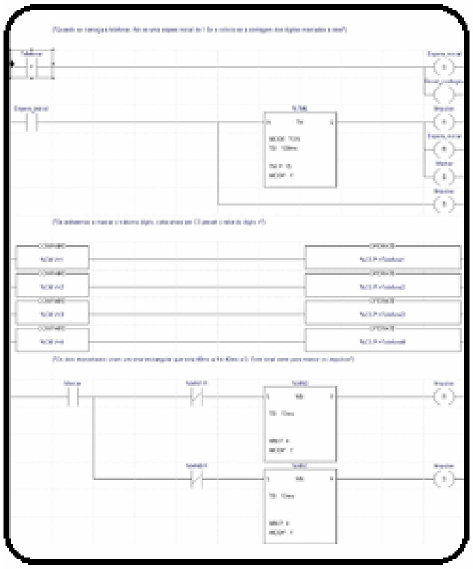


FIG 11.2: PART OF A LADDER DIAGRAM, INCLUDING CONTACTS AND COILS, COMPARES, TIMERS AND MONOSTABLE MULTIVIBRATORS

Ladder logic is a programming language that represents a program by a graphical diagram based on the circuit diagrams of relay-based logic hardware. It is primarily used to develop software for Programmable Logic Controllers (PLCs) used in industrial control applications. The name is based on the observation that programs in this language resemble ladders, with two vertical rails and a series of horizontal rungs between them.

11.2.2 ADDITIONAL FUNCTIONALITY

Additional functionality can be added to a ladder logic implementation by the PLC manufacturer as a special block. When the special block is powered, it executes code on predetermined arguments. These arguments may be displayed within the special block.

```
                          +-------+
 -----[ ]--------------------+  A    +----
  Remote Unlock              +-------+
                        Remote Counter


                          +-------+
 -----[ ]--------------------+  B    +----
  Interior Unlock            +-------+
                       Interior Counter


                   +--------+
 --------------------+ A + B  +-----------
                   + into C +
                   +--------+
```

In this example, the system will count the number of times that the interior and remote unlock buttons are pressed. This information will be stored in memory locations A and B. Memory location C will hold the total number of times that the door has been unlocked electronically.

PLCs have many types of special blocks. They include timers, arithmetic operators and comparisons, table lookups, text processing, PID control, and filtering functions. More powerful

PLCs can operate on a group of internal memory locations and execute an operation on a range of addresses, for example, to simulate a physical sequential drum controller or a finite state machine. In some cases, users can define their own special blocks, which effectively are subroutines or macros. The large library of special blocks along with high speed execution has allowed use of PLCs to implement very complex automation systems.

11.3 FEATURES

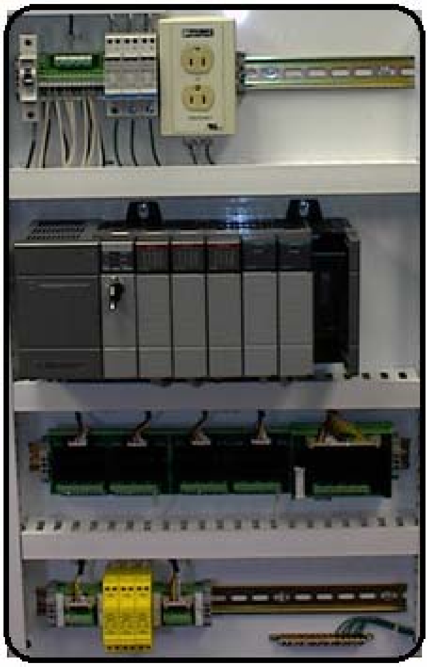


FIGURE 11.3: CONTROL PANEL WITH PLC (GREY ELEMENTS IN THE CENTER). THE UNIT CONSISTS OF SEPARATE ELEMENTS, FROM LEFT TO RIGHT; POWER SUPPLY, CONTROLLER, RELAY UNITS FOR IN- AND OUTPUT

The main difference from other computers is that PLCs are armored for severe conditions (such as dust, moisture, heat, cold) and have the facility for extensive input/output (I/O) arrangements. These connect the PLC to sensors and actuators. PLCs read limit switches, analog process variables (such as temperature and pressure), and the positions of complex positioning systems. Some use machine vision. On the actuator side, PLCs operate electric motors, pneumatic or hydraulic cylinders, magnetic relays, solenoids, or analog outputs. The input/output arrangements may be built into a simple PLC, or the PLC may have external I/O modules attached to a computer network that plugs into the PLC.

11.4 SYSTEM SCALE

A small PLC will have a fixed number of connections built in for inputs and outputs. Typically, expansions are available if the base model has insufficient I/O.

Modular PLCs have a chassis (also called a rack) into which are placed modules with different functions. The processor and selection of I/O modules is customised for the particular application. Several racks can be administered by a single processor, and may have thousands of inputs and outputs. A special high speed serial I/O link is used so that racks can be distributed away from the processor, reducing the wiring costs for large plants.

11.5 USER INTERFACE

PLCs may need to interact with people for the purpose of configuration, alarm reporting or everyday control.

A Human-Machine Interface (HMI) is employed for this purpose. HMIs are also referred to as MMIs (Man Machine Interface) and GUI (Graphical User Interface).

A simple system may use buttons and lights to interact with the user. Text displays are available as well as graphical touch screens. More complex systems use a programming and monitoring software installed on a computer, with the PLC connected via a communication interface.

11.6 COMMUNICATIONS

PLCs have built in communications ports, usually 9-pin RS-232, but optionally EIA-485 or Ethernet. Modbus, BACnet or DF1 is usually included as one of the communications protocols. Other options include various fieldbuses such as DeviceNet or Profibus. Other communications protocols that may be used are listed in the List of automation protocols.

Most modern PLCs can communicate over a network to some other system, such as a computer running a SCADA (Supervisory Control And Data Acquisition) system or web browser.

PLCs used in larger I/O systems may have peer-to-peer (P2P) communication between processors. This allows separate parts of a complex process to have individual control while allowing the subsystems to co-ordinate over the communication link. These communication links are also often used for HMI devices such as keypads or PC-type workstations.

11.7 EXAMPLES

(1) As an example, say a facility needs to store water in a tank. The water is drawn from the tank by another system, as needed, and our example system must manage the water level in the tank.

Using only digital signals, the PLC has two digital inputs from float switches (Low Level and High Level). When the water level is above the switch it closes a contact and passes a signal to an input. The PLC uses a digital output to open and close the inlet valve into the tank.

When the water level drops enough so that the Low Level float switch is off (down), the PLC will open the valve to let more water in. Once the water level rises enough so that the High Level switch is on (up), the PLC will shut the inlet to stop the water from overflowing. This rung is an example of seal-in (latching) logic. The output is sealed in until some condition breaks the circuit.

```
|                                          |
|  Low Level     High Level             Fill Valve     |
|------[/]------|------[/]----------------------(OUT)---------|
|            |                             |
|            |                             |
|            |                             |
|  Fill Valve  |                               |
|------[ ]------|                              |
|                                          |
|                                          |
```

An analog system might use a water pressure sensor or a load cell, and an adjustable (throttling) dripping out of the tank, the valve adjusts to slowly drip water back into the tank.

In this system, to avoid 'flutter' adjustments that can wear out the valve, many PLCs incorporate "hysteresis" which essentially creates a "deadband" of activity. A technician adjusts this deadband so the valve moves only for a significant change in rate. This will in turn minimize the motion of the valve, and reduce its wear.

A real system might combine both approaches, using float switches and simple valves to prevent spills, and a rate sensor and rate valve to optimize refill rates and prevent water hammer. Backup and maintenance methods can make a real system very complicated.

(2) EXAMPLE OF A SIMPLE LADDER LOGIC PROGRAM

The language itself can be seen as a set of connections between logical checkers (contacts) and actuators (coils). If a path can be traced between the left side of the rung and the output, through asserted (true or "closed") contacts, the rung is true and the output coil storage bit is asserted (1) or true. If no path can be traced, then the output is false (0) and the "coil" by analogy to electromechanical relays is considered "de-energized". The analogy between logical propositions and relay contact status is due to Claude Shannon.

Ladder logic has contacts that make or break circuits to control coils. Each coil or contact corresponds to the status of a single bit in the programmable controller's memory. Unlike electromechanical relays, a ladder program can refer any number of times to the status of a single bit, equivalent to a relay with an indefinitely large number of contacts.

So-called "contacts" may refer to physical ("hard") inputs to the programmable controller from physical devices such as pushbuttons and limit switches via an integrated or external input module, or may represent the status of internal storage bits which may be generated elsewhere in the program.

Each rung of ladder language typically has one coil at the far right. Some manufacturers may allow more than one output coil on a rung.

--( )-- a regular coil, energized whenever its rung is closed

--(\)-- a "not" coil, energized whenever its rung is open

--[ ]-- A regular contact, closed whenever its corresponding coil is energized

--[\]-- A "not" contact, open whenever its corresponding coil is energized

The "coil" (output of a rung) may represent a physical output which operates some device connected to the programmable controller, or may represent an internal storage bit for use elsewhere in the program.

(3) Here is an example of what one rung in a ladder logic program might look like. In real life, there may be hundreds or thousands of rungs.

For example:

```
 1. ----[ ]---------|--[ ]--|------( )
        X           |   Y   |       S
                    |       |
                    |--[ ]--|
                        Z
```

The above realises the function: S = X AND (Y OR Z)

Typically, complex ladder logic is 'read' left to right and top to bottom. As each of the lines (or rungs) are evaluated the output coil of a rung may feed into the next stage of the ladder as an input. In a complex system there will be many "rungs" on a ladder, which are numbered in order of evaluation.

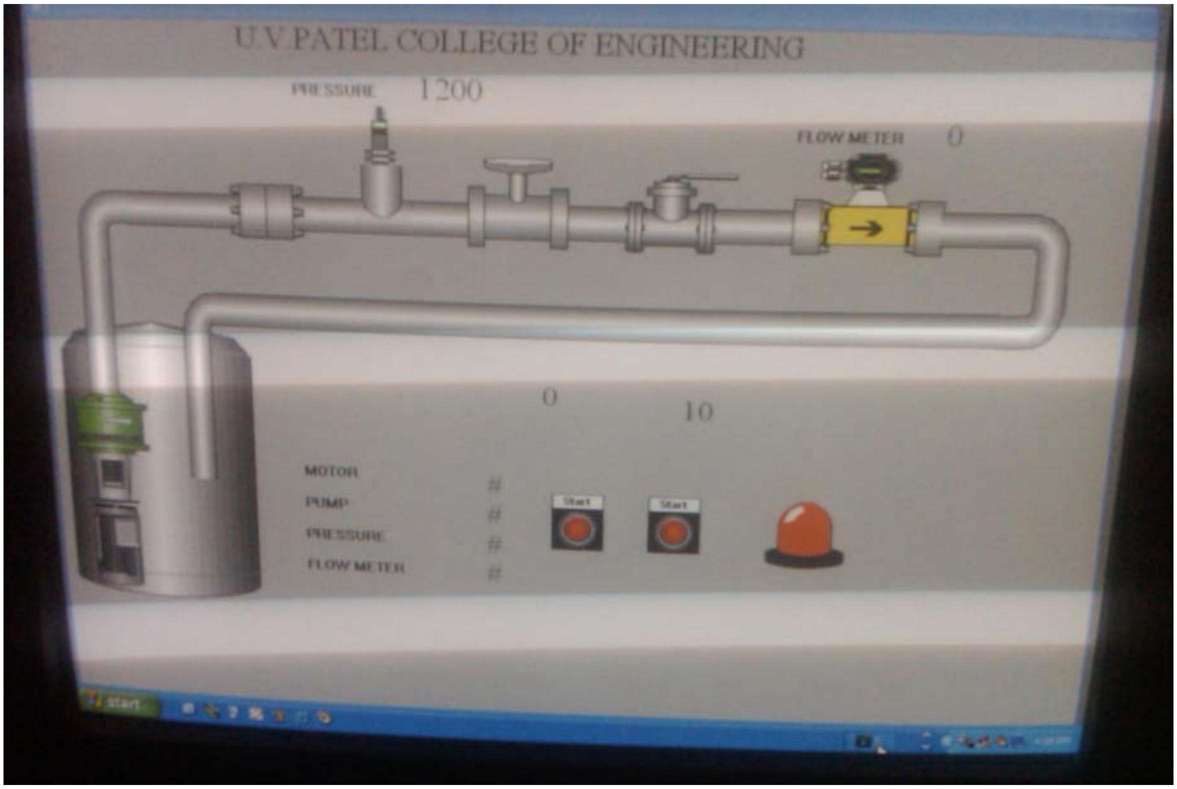
```
 1. ----[ ]-----------|---[ ]---|----( )
        X             |    Y    |     S
                      |         |
                      |---[ ]---|
                           Z
 2. ----[ ]----[ ]-------------------( )
        S      X                      T
```

2. T = S AND X where S is equivalent to #1. above

This represents a slightly more complex system for rung 2. After the first line has been evaluated, the output coil (S) is fed into rung 2, which is then evaluated and the output coil T could be fed into an output device (buzzer, light etc..) or into rung 3 on the ladder. (Note that the contact X on the second rung serves no useful purpose, as X is already defined in the 'AND' function of S from the 1st rung.)

This system allows very complex logic designs to be broken down and evaluated.

12. PROJECT WORK



12. PROJECT WORK

The SCADA can give the various advantages as described above. The system when carefully designed can be made to use all these advantages to optimize the system.

12.1 OBJECTIVE

The aim of our project is to operate the pump and to achieve the control of the flow rate and pressure head developed by the pump.

In the subsequent pages is shown the entire details of the construction and their functioning to achieve optimum consideration.

12.2 COMPONENTS

- Water tank.
- Pipe assembly (nipples, union, flange, bushing, flexible pipe)
- Motor.
- Vertical 1 hp, 10 stage centrifugal pump.
- Nipples.
- Manual valve.
- Non return valve.
- Sensors (pressure transmitter, flow meter)
- Computer system.
- Programmable logic controller.
- Contactor base push button starter.

12.3 OPERATION

The following description will show the various functions performed by the parts and to accomplish these with automation the ladder diagrams are shown. This program governs the PLC and the operation is performed according to these steps:

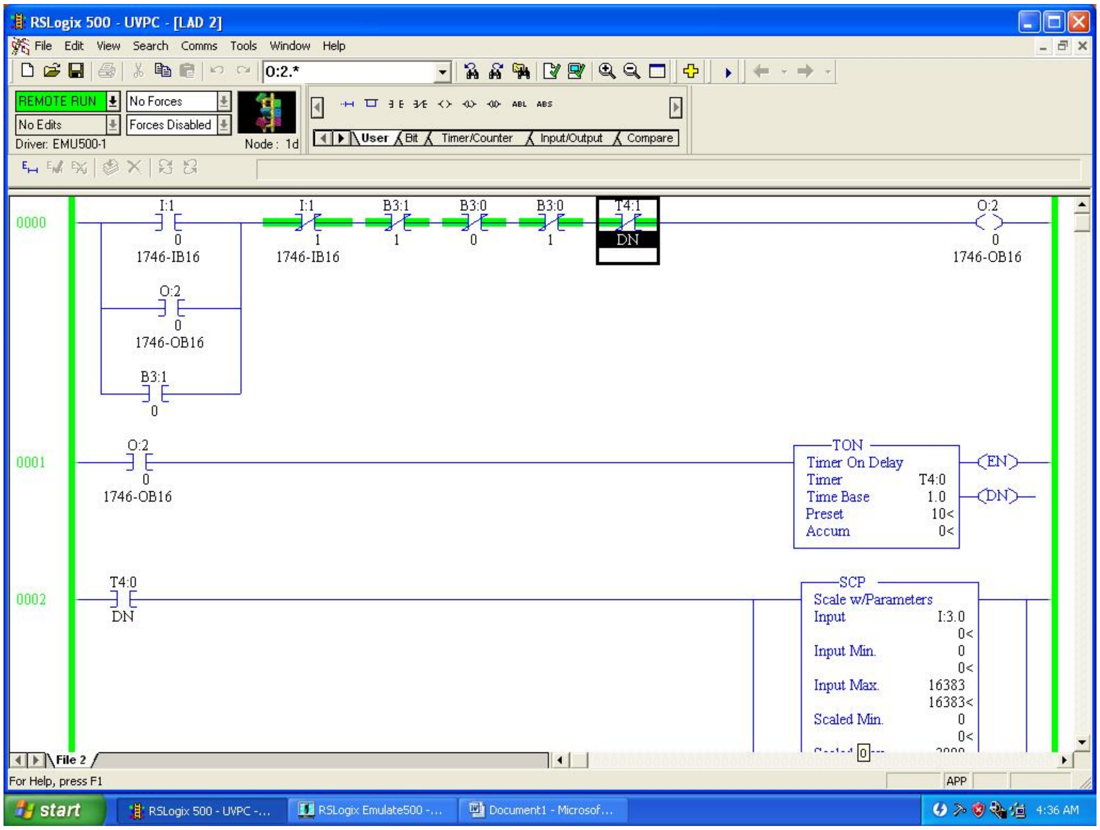
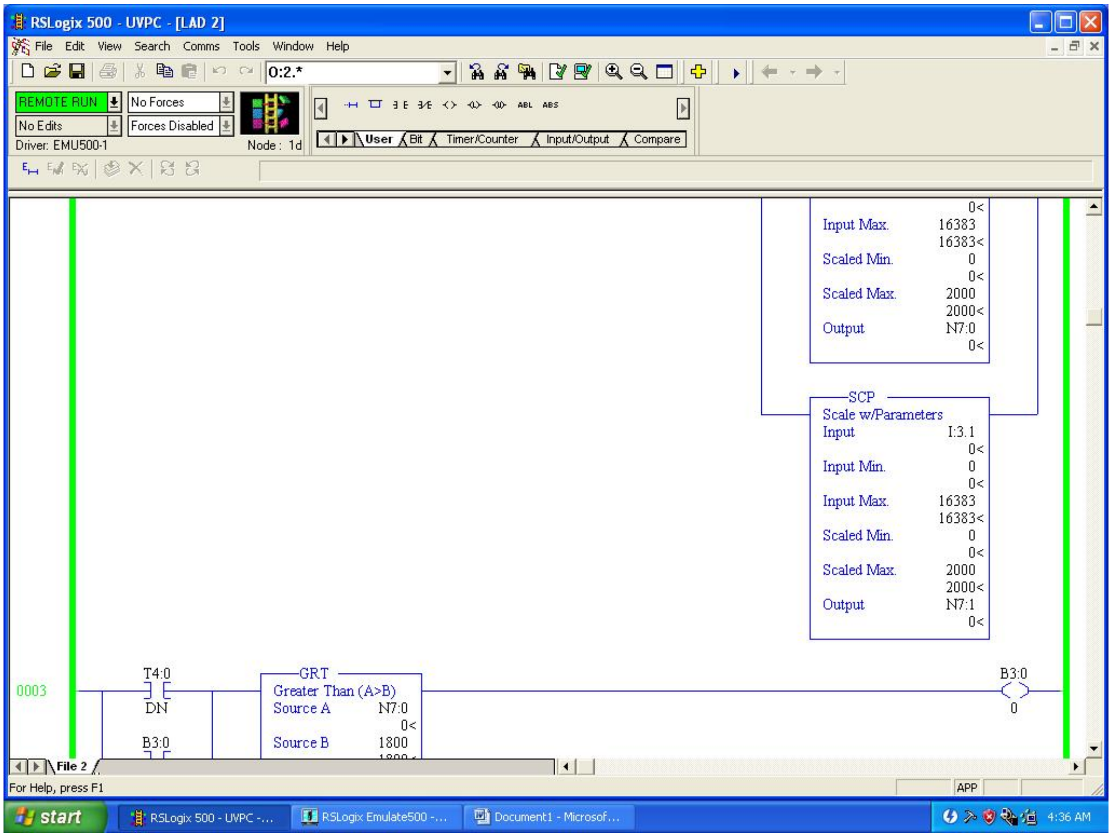


FIG 12.1:LADDER PROGRAMMING PART-1

The push button with contactor base is pushed to give a starting pulse to the motor. Now a push button is not constant value but is not continuous so a hold on the pulse is provided. If any time during operation the pump is to be shut dow then the stop push botton is pressed.

Now after the pump commences to operate the sensors start reading after the first 10 seconds have passed so that the sensors read the value when the the pump is stabilized.



```
                                        0<
Input Max.            16383
                      16383<
Scaled Min.               0
                         0<
Scaled Max.            2000
                      2000<
Output                 N7:0
                         0<

SCP
Scale w/Parameters
Input                 I:3.1
                         0<
Input Min.                0
                         0<
Input Max.            16383
                      16383<
Scaled Min.               0
                         0<
Scaled Max.            2000
                      2000<
Output                 N7:1
                         0<

0003   T4:0      GRT                                B3:0
       ] [       Greater Than (A>B)                 < >
       DN        Source A        N7:0                0
                                   0<
       B3:0      Source B        1800
```

FIG 12.2: LADDER PROGRAMMING PART-2

During the operation at every instant the fow and the pressure head are measured and the are compared with the required value or value range as per our requirement. The required range for the parameters controlled are:

- For pressure head: 35 kg/m2 to 60 kg/ m2.
- For discharge/ flow rate: 50 lpm to 90 lpm.

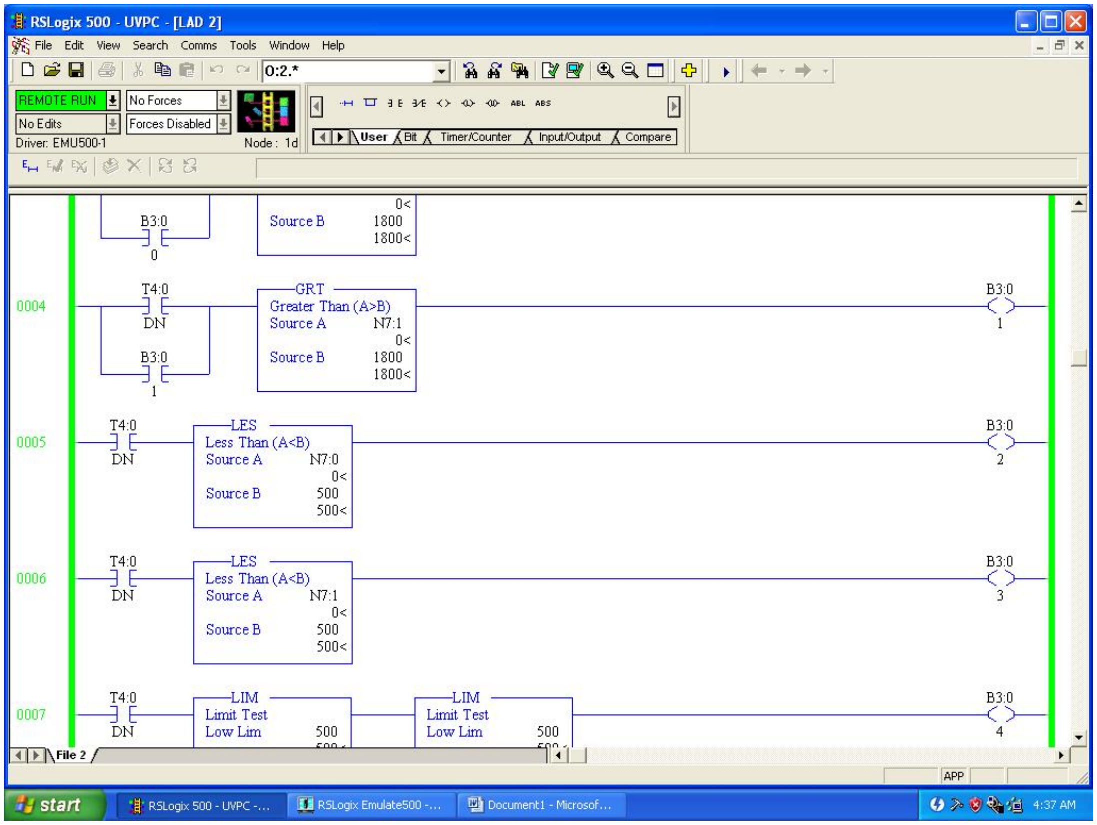


FIG 12.3:LADDER PROGRAMMING PART-3

Now this measured value is made to compare with the required range values.

However due to various faults in the system the performance always doesnot occur in the requiredrange. If the value is between the specified range the operation is continued. If the value obtained from the sensor is less than the range value it will send to the PLC A signal to operate the motor for 10 more seconds so that the the pump is given a chance to stabilize. However, if it doesnot take place the motor is tripped and the operation ceased.

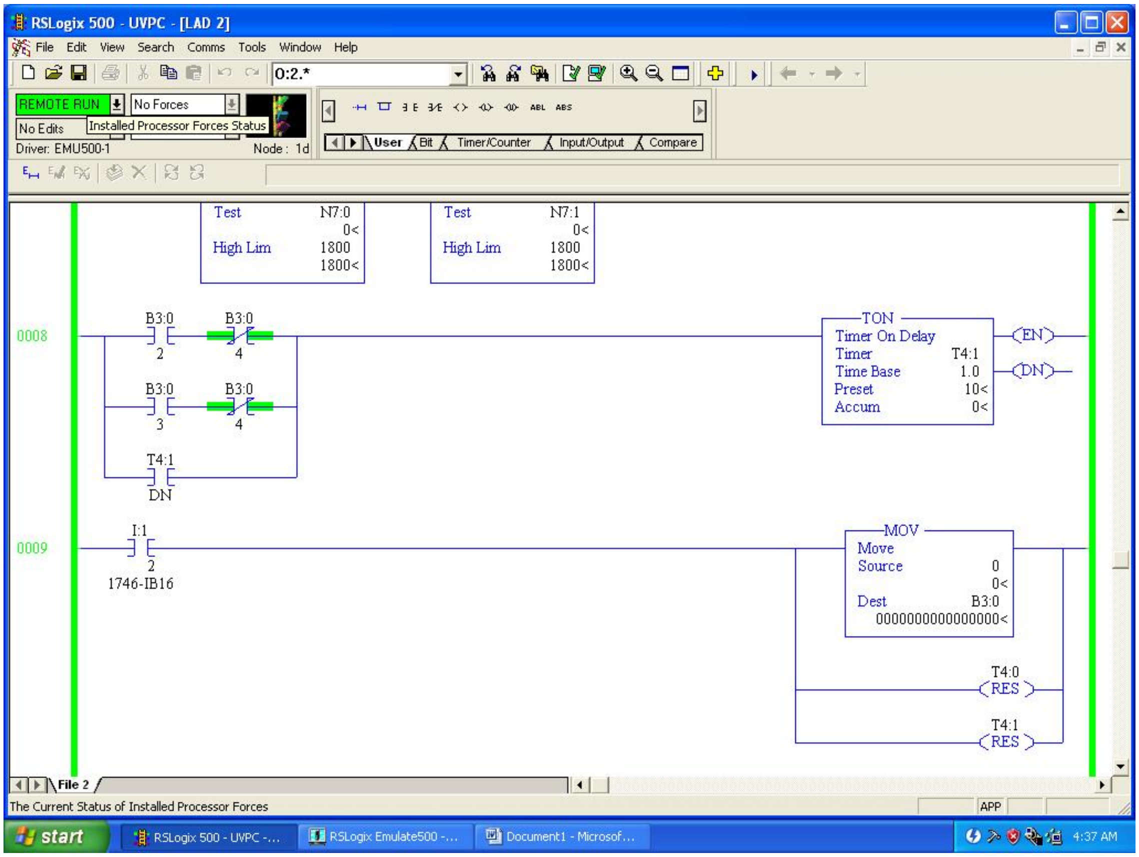


FIG 12.4: LADDER PROGRAMMING PART-4

In the other scenario the value occuring can be more than the range in this case the motor shall trip w/o any delay.

The process shall continue to operate like this until one presses the red stop push button.

IMPLEMENTATION OF SCADA

- Pictorial representation of the entire system.
- Monitoring the sytem at each point.
- Communication with PLC.
- Controlling of ON and OFF of the system.
- Supervising the pressure and flow parameters and it also provides the analog instantaneous value.
- For system hazards it gives an alarm indication.

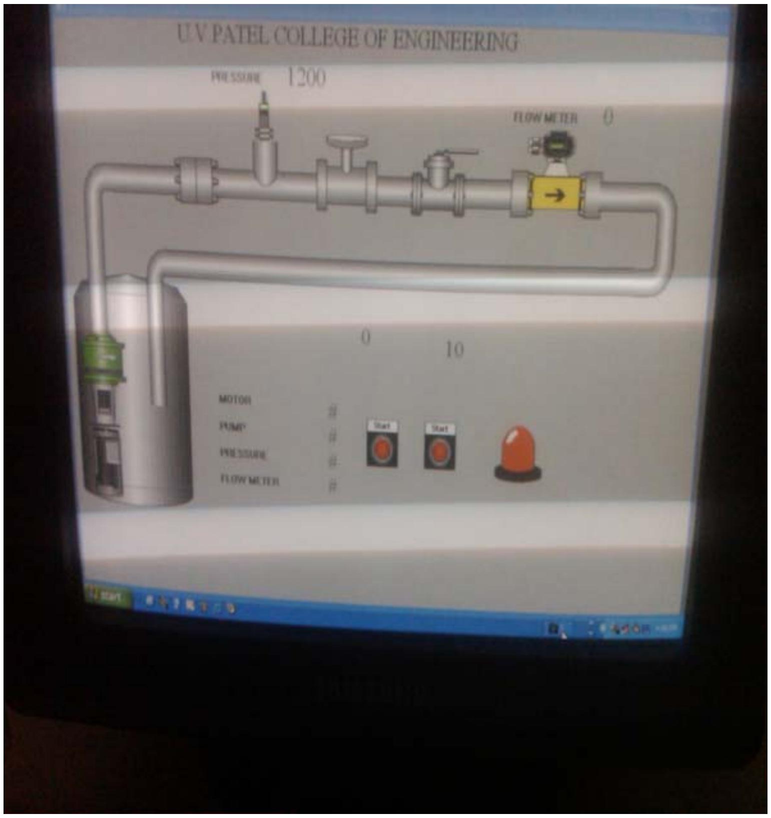


FIG 12.5 : SCADA IMPLEMENTATION

12.4 COSTING OF THE PROJECT

Component	**Cost(Approx.) in Rs.**
Pump & Motor	8,000
Starter	1,250
Pipeline Assembly	1,250
Non Return Valve	450
Ball Valve(Manual Valve)	270
Pressure Transmitter(Sensor)	2,350
Flow Meter(Sensor)	97,000
Wiring Cost	980
Programmable Logic Controller	40,000
Wonder ware Intouch SCADA Software	23,000
Total Costing	1,74,550

13. CONCLUSION

13.CONCLUSION

In our endevour to achieve the automated pump control process we conclude that it has been successful to our aim. The following is project concluding status:

The entire system to control the pump’s pressure and discharge rate available is computerized. The pump can be switched on and off by simple buttons on the monitor of the computer. Once the pump is on the entire process control is achieved on the computer and the any deviation In the process would lead to auto tripping off of the pump.

We have been successful In our endevour however we also been able to identify the purpose of this project. Such a control and monitoring system process in viable only in case of large systems as the cost incurred for small control and monitoring process is not viable. Manual labour is more beneficial at such places. However such system when applied to large processes on heavy duty floors in industries prove highly efficient and cost effective.