Implementation of Real-Time SCADA-Based HCI Manufacturing Process

Hla Myo Tun[1] Zaw Min Naing[2] Maung Latt[3] Win Khine Moe[4]

Abstract

This research presents SCADA based control system that is capable of Natural Fertilizer Manufacturing Process. In this research, the simulation of the whole process is conducted in real-time condition by using Visual C#.Net programming under Visual Studio 2008 software. The parallel port of PC and PIC microcontrollers have been communicated with parallel port interface Visual C#.Net program. Peripheral Interface Controller (PIC16F877A) is used as a control unit of natural fertilizer manufacturing process with C Programming instead of Programmable Logic Controller hardware. In interfacing section, parallel port was used to communicate to the personal computer with all process devices. However, it did not use so many input lines because it has limitation of I/O ports. Optocouplers are also included to divide different voltages between circuits and the computer. The monitoring system displays the real time manufacturing process on the computer. This automation process helps the natural fertilizer manufacturing process to reduce the amount of errors that occur, to reduce the human resources, to increase the efficiency and cost effectiveness.

Keywords:

1 Introduction

Most of industrial plants have a control center where is installed a SCADA (Supervisory Control and Data Acquisition). It is a software application specially designed to work on computers in the production control, providing communication with the devices (independent controllers, programmable robots, etc.) and controlling the process from the screen of the computer. In addition, it provides all the information that is generated in the process to diverse users, as much as the same level as to other supervisors within the plant: quality control, supervision, maintenance, etc.

It usually exist a computer, which carries out tasks of supervision and management of motors, as well as data processing and process control. The communication is made by means of special buses. All this is executed normally in real time, and is designed to give to the plant operator the possibility of supervising and controlling these processes [1]. The abstraction that process control allows has proven to be one of the greatest costs and time savers of the industrial age. Instead of

[1] Department of Electronic Engineering, Yangon Technological University hmyotun@gmail.com
[2] Department of Research and Innovation, Ministry of Education, drzaw290615@gmail.com
[3] Technological University (Toungoo) , Ministry of Education, mgmglatt2020@gmail.com
[4] Department of Research and Innovation, Ministry of Education winkmoe@gmail.com

many workers monitoring and altering parts of large systems, one or two operators can control the whole system from a central point.

This research outlines the various stages of operation included in the conversion of a manually operated natural fertilizer manufacturing plant towards an automated plant. The natural fertilizer manufacturing process contains four main stages: preparation of raw materials, production of mould, mixing of mould, rice husk ash and microorganism’s culture broth and packaging process. Firstly, if the start button is pressed, the whole process turns on. Then, mixer motor runs and the first conveyor moves towards one direction by necessary speed. Mould is filled in the tank1 until reaching the specified weight level and rice husk ash is filled in the tank2 until reaching the specified weight level. And then the conveyor moves towards one direction by necessary speed. Mould and rice husk ash are carried by conveyor and are filled in the mixer. Mould and rice husk ash are mixed in the mixer. The natural fertilizer gets mixture of mould, husk ash and microorganisms’ culture broth [2]. Natural fertilizer is filled in the bags until the specified weight level. The robot arm picks up the bag and places it another conveyor. These bags are carried by another conveyor and are bagged by sewing machine.

The weight and position feedback data from these sensors are converted the electrical signals by using signal conditioning circuit. By applying these electrical signals for the microcontroller not only the conditions of process is controlled but also the real time condition is monitored on the computer. The optocouplers are interfaced from the parallel port of the computer and the microcontroller.

2 Natural Fertilizer Manufacturing Process

Myanmar is an agricultural country, with a total of 18 million hectares of cultivable land, and out of which about nine million hectares are under crop cultivation. So, the demand for fertilizer is very high. The currently used agricultural inputs are mostly chemicals.

The use of chemical fertilizers has been widely practiced because of its almost immediate effect, which has replaced the traditional way of recycling, this nutrients to a point that this is now taken for granted. Moreover, intensive application of chemical fertilizers in agriculture has caused the damage to the ecological state of the agricultural system.

The use of natural fertilizer is an alternative to improve the condition of soil. Natural fertilizers do not contaminate the soil and atmosphere. It helps to produce healthy foods. It is well known that a considerable number of bacterial species are able to exert a beneficial effect upon plant growth. They are used as natural fertilizers or control agents for agriculture improvement. The main parts of natural fertilizer manufacturing process are shown in Figure.1. In natural fertilizer manufacturing process, the three main components are included. These are mixing process, filling process and packaging process. This research outlines the various stages of operation included in the conversion of a manually operated natural fertilizer manufacturing plant towards an automated plant. The quality of natural fertilizer can be increased because mixing time, filling time and bagging time are exactly using this SCADA system.

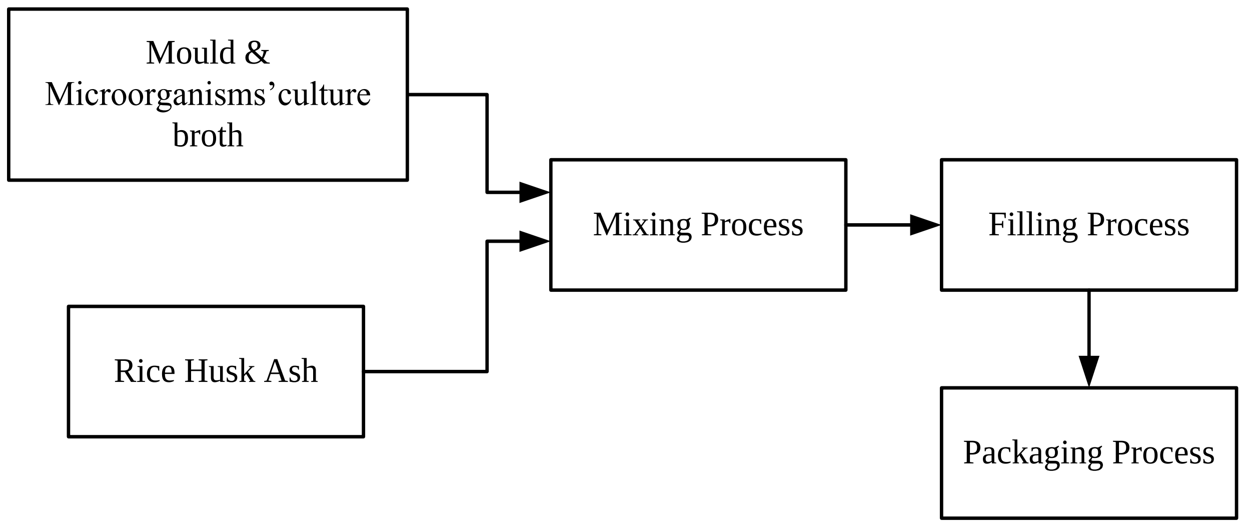


Figure 1: Block diagram of Natural Fertilizer Manufacturing Process [2]

3 SCADA System for Natural Fertilizer Manufacturing Process

In the production of natural fertilizer, the manufacturing process is used for controlling by Supervisory Control and Data Acquisition system. Most of modern industries use SCADA system because of security and reliability system and other facts. It is shown in Fig. 3.2. SCADA system includes a master station (personal computer). And this manufacturing process uses PLC controller unit in most industries. It is designed with PIC microcontroller instead of using PLC and other control circuits. So, Remote Terminal Unit (RTU) includes interfacing circuit, PIC microcontroller and devices used control driver circuit. Parallel port communication is used to connect master station and RTU. Three weight sensors and two position sensors are used to detect the amount of weight and the position of the object. The data from sensors is controlled by using controller circuit and interface circuit is used to match this data to understand the computer using parallel port and optocouplers. The operation of motors is driven by PIC microcontroller.

For a small factory, it is monitoring system using Visual C#.Net programming. It designs the real-time monitoring for natural fertilizer manufacturing process. So, it consists of Master station, which includes data acquisition and visualization programs and RTU, which includes PIC microcontroller based signal sensing and supervisory control. SCADA software has various types (Citech, Lab View, RSView32, WinCC,In Touch 7.1 and etc.). In this research work, own monitoring program for natural fertilizer manufacturing process is used instead of the SCADA software. This program is written by visual C#. Net programming under Visual Studio 2008 [3].

4 Implementation

The software written to control the natural fertilizer manufacturing process is also described. The features of the MPLAB compiler are used to program the PIC microcontroller.

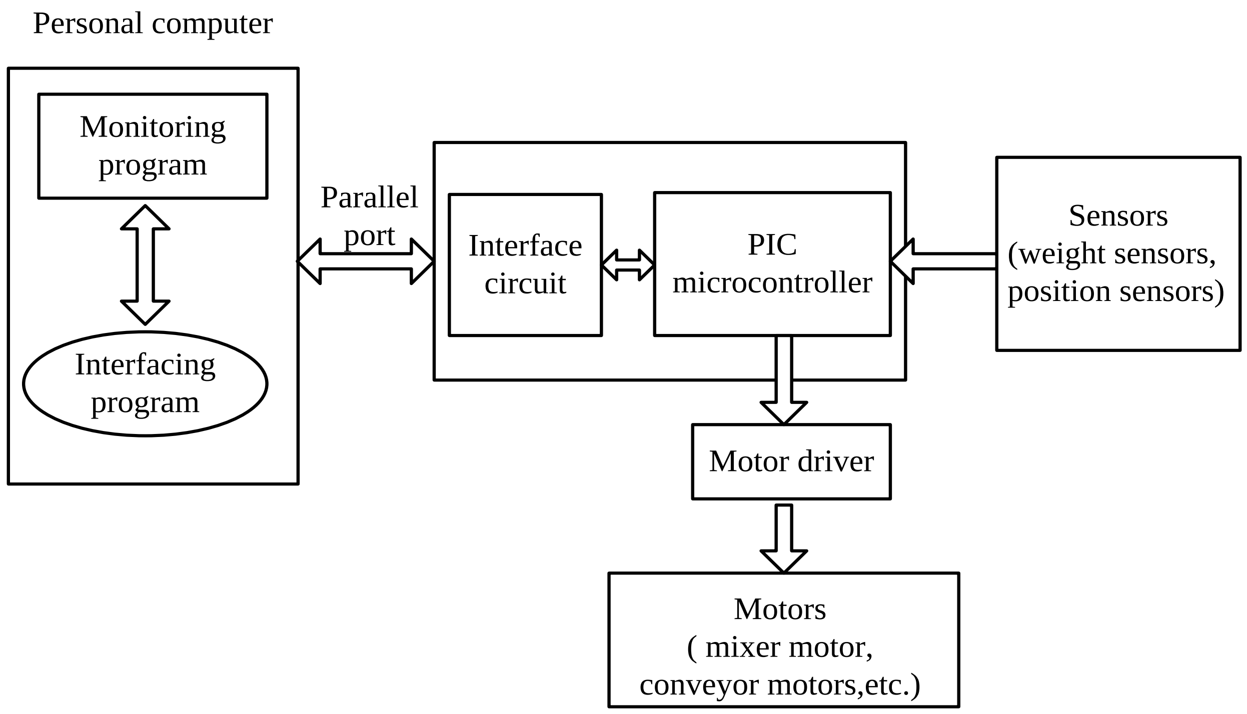


Figure 2: Block Diagram of SCADA System for Natural Fertilizer Manufacturing Process [3]

4.1 Interfacing and Monitoring

This system consists of two portions. The first one is to control the operation of SCADA based natural fertilizer manufacturing process from the controlling unit. The other is to display and simulate for the whole manufacturing process. The interfacing system includes optocouplers and parallel port. The monitoring system is written by Visual C#.Net programming from Visual Studio 2008 software.

A. Interface Portion of Optocoupler

The PC817 type optocouplers are used to shift the matching voltages between the computer and PIC16F877A. They are used to match the various different voltages. This system uses two grounds as common ground from the process and common ground from the computer. The selected pins of the parallel port are used as output pins of PC and its output signal (1, 2, 3 and etc.) is used to select the output lines of PIC. Pin 2 of parallel port cable is used as a start/stop signal pin to control the natural fertilizer manufacturing process.

B. Computerized Control Portion of the System

The parallel port interfacing is applied in this system to input the computer and to output the process. The standard parallel port uses three contiguous addresses, usually in one of these ranges. The first address in the range is the port’s base address, also called the data register. The second address is the port’s status register, and register, and the third address is the control register. The data register is used as output and the status port is used as input. The control register is not used in this research.

C. Interfacing for the Parallel Port Communication

The parallel port is used to communicate, control and collect data with operation mode linking with RTU. It uses library files ".dll" to control visual C#.net for input/output from actual process. Windows have other options for driving device, including DLLs. A visual C# program can call a DLL directly to access a DLL. A DLL (dynamic linked library) is a set of procedures that windows applications can call. When an application runs, it links to the DLLs declared in its program code, and the corresponding DLLs load into memory. Multiple applications can access the same DLL. The application calls DLL procedures much like any other subroutine or function.

Many programming languages enable to write and compile DLLs. Creating a DLL can be as simple as writing the code and choosing to compile it as a DLL rather than as an executable (.exe) file. Visual C# programs can call any DLL, whether it was originally written in Basic or another language. The parallel port in the original IBM PC, and any port that emulates the original port's design, is sometimes called the SPP, for standard parallel port, even though the original port had no written standard beyond the schematic diagrams and documentation for the IBM PC. Other names used are AT-type or ISA-compatible. The port in the original PC was based on an existing centronics printer interface. However, the PC introduced a few differences, which other systems have continued.

SPPs can transfer eight bits at once to a peripheral, using a protocol similar to that used by the original centronics interface. The SPP doesn't have a byte-wide input port, but for PC-to-peripheral transfers, SPPs can use a Nibble mode that transfers each byte 4 bits at a time. Nibble mode is slow, but has become popular as a way to use the parallel port for input. There are many ways to access a parallel port in software, but all ultimately read or write to the port's registers. To distinguish between I/O ports and system memory, the microprocessor uses different instructions and control signals for each. On the original PC, port address could range from 0FFh to 3FFh. Many newer parallel port decode an eleventh address line to extend the range to 7FFh. Many programs that access the parallel port use this table to get a port's address. This way uses only have to select LPT1, LPT2, or LPT3, and the program can find the address. Use the control bits as inputs on the PC only on SPPs or ports that emulate the SPP. If it uses the control lines as inputs, drive them with open-collector outputs. This will protect the port's circuits if a law control-port output should connect to a high output.

Visual CSharp has been the most popular choice for basic programmers developing window programs. It can add Inp and Out to the language in a dynamic linked library (DLL). A DLL contains code that any windows program can access, including the program written in Visual CSharp. This SCADA based software includes DLLs for port access: inpout32.dll, for use with 32-bits Visual C# program [4].

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Runtime.InteropServices;
public class PortAcess {
      [DllImport("inpout32.dll", EntryPoint= "Inp32")]
      public static extern int Inp32(int Port Address);
      [DllImport("inpout32.dll", EntryPoint= "Out32")]
      public static extern int Out32(int PortAddress, int Value);
      public static double an1;
      public static double an2;
      public static double an3;
      public static int sw1;
      public static int sw2;
      public static int motor1;
      public static int motor2;
      public static int motor3;
      public static int motor4;
      public static int motor5;
      public static int motor6;
      public static int motor7;
      public static int motor8;
      public static bool Tank1ArrowEnable;
      public static bool Tank2ArrowEnable;
      public static bool Sensor1ArrowEnable;
      public static bool RobotEnable;
      public static bool PackingConveyor;
      public static long TotalBag;
      public static bool SystemStart;
}
```

The program declares the Inp32 and Out32 contained in inpout32.dll and declares variables using software for natural fertilizer manufacturing process. A program that writes directly to parallel port has no way of knowing whether another application is already using the port. But sometimes a port is intended to use with a single application. If other applications have no reason to access the port, direct I/O with In and Out should cause no problems.

When receiving data from Operation mode three timers are used to different the condition of sensors and motors using natural fertilizer manufacturing process. With Inp and Out declared in program, it can use them much like Inp and Out in QuickBasic. On the user's system, the file Inpout32.dll should be copied to one of these locations: the default windows directory (usually\Windows), the default system directory (usually \ Windows \ System), or the application's working directory. These are the locations that windows automatically search when it loads a DLL. If for some reason the DLL is in a different directory, it will need to add its path to the filename in the declare statements.

D. Linking Windows System for the Process

The operation of the SCADA based natural fertilizer manufacturing process can be run and stopped from main page by the operator. The main page serves to collect data from the operation mode and display in real-time running and save the collected data. The graph line is drawn in approximately one second step until the STOP command is accepted. When the main page program is shutdown the data acquisition and processing is shutdown all process except receiving command from the main page. So, the application software can be viewed in pages, the main page of PC and other pages linking each other as software of SCADA system. The complete block diagram is shown in Figure.3. The main page can be divided into the following function. These are:

- Operation mode (real-time monitoring as hardware),
- Hardware mode (displaying components like real devices),
- Instruction mode (List of facts of SCADA system for Natural Fertilizer Manufacturing Process) and
- Control mode (including start, stop and exit for security) [6].

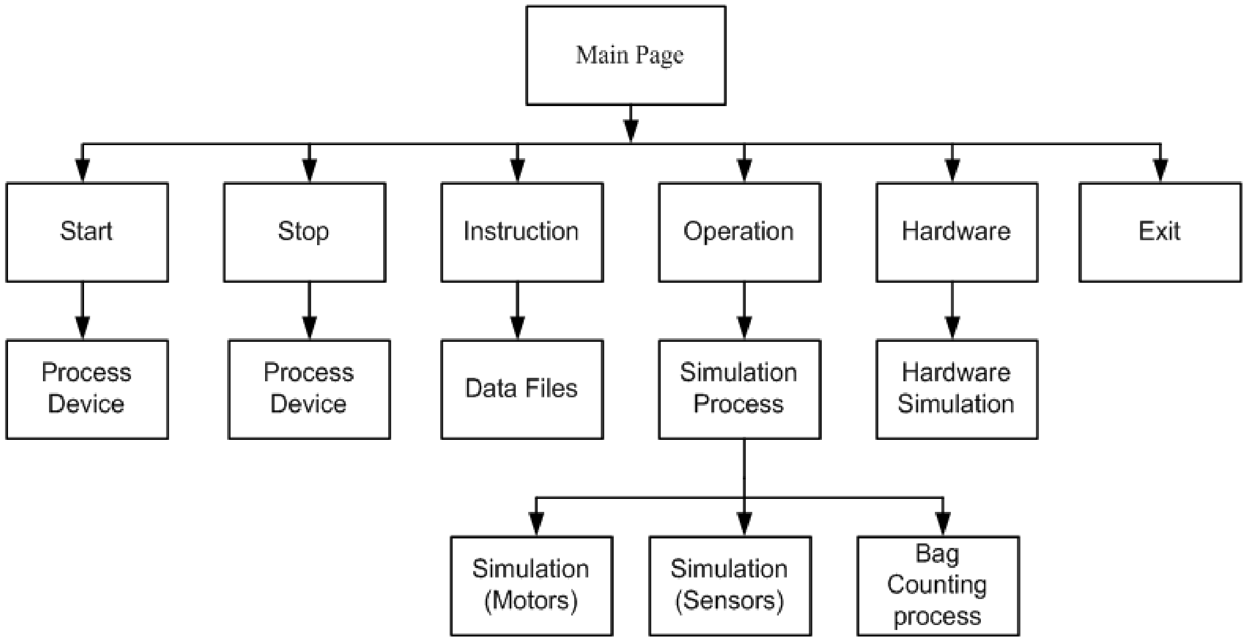


Figure 3: Block Diagram of Linking Window System

4.2 Design Program for Main Page

The Visual C#.Net portion of window application from Visual Studio 2008 IDE is used for Main page. It designs a window form using the software application data and uses data by depended the process from the properties. The devices used for the control are designed on the form by using the toolbox. And the codes of programming for each of the component are written on the code viewer by using Visual C#. Timer tools are used for timing of data interfacing and input/output for process data. A timer is used to raise an event at user-defined intervals. This windows timer is designed for a single-threaded environment where User Interface (UI) threads are used to perform processing. It requires that the user code have a UI message pump available and always operate from the same thread, or marshal the call onto another thread [5].

Gets or sets the time, in millisecond, between timer ticks. Windows forms timer component has an interval property that specifies the number of milliseconds that pass between one timer event and the next. Unless the component is disabled, a component is designed for a windows forms environment. The read or write signal of timer continues to receive the Tick event at roughly equal interval of time. This input/output system activates every time. It uses timer tool (timerIn_Tick) to watch the port system. The signal from timer state is flowed to selected pin of PC.

If five selected pins of parallel port sent to PIC as '1', PC would read weight level of variable resistor1 (instead of weight sensor1) for tank1. If five selected pins of parallel port sent to PIC as '2', PC would read weight level of variable resistor2 (instead of weight sensor2) for tank2. If five selected pins of parallel port sent to PIC as '3', PC would read weight level of variable resistor3 (instead of weight sensor3) for bagging. If five selected pins of parallel port sent to PIC as '4', PC would read position of bag from switches (instead of position sensor1 and position sensor2). The input signal of each port is compared with the particular selecting address. It is shown in Figure.4. It can be written:

```
PortAccess.Out32(888, 1);
delay();
input = PortAccess.Inp32(889);
PortAccess.Out32(888, 0);
delay();
b0 = (( input & 64 ) == 0)? 1: 0;
b1 = (( input & 128 ) == 0)? 0: 1;
b2 = (( input & 32 ) == 0)? 1: 0;
b3 = (( input & 16 ) == 0)? 1: 0;
b4 = (( input & 8) == 0)? 1: 0;
PortAccess. an1 = b0 * 1 + b1 * 2 + b2 * 4 + b3 * 8 + b4 * 16;
PortAccess.Out32(888, 2);
delay();
 input = PortAccess. Inp32(889);
PortAccess.Out32(888, 0);
delay();
b0 = (( input & 64 ) == 0)? 1: 0;
b1 = (( input & 128 ) == 0)? 0: 1;
b2 = (( input & 32 ) == 0)? 1: 0;
b3 = (( input & 16 ) == 0)? 1: 0;
b4 = (( input & 8 ) == 0)? 1: 0;
an2 = b0 * 1 + b1 * 2 + b2 * 4 + b3 * 8 + b4 * 16;PortAccess.Out32(888, 3);
delay();
input = PortAccess. Inp32(889);
PortAccess.Out32(888, 0);
delay();
b0 = (( input & 64 ) == 0)? 1: 0;
b1 = (( input & 128 ) == 0)? 0: 1;
b2 = (( input & 32 ) == 0)? 1: 0;
b3 = (( input & 16 ) == 0)? 1: 0;
b4 = (( input & 8 ) == 0)? 1: 0;
PortAccess.an3 = b0 * 1 + b1 * 2 + b2 * 4 + b3 * 8 + b4 * 16;
PortAccess.Out32(888, 4);
delay();
input = Inp32(889);
PortAccess.Out32(888, 0);
delay();
```

```
PortAccess.sw1 = ((input & 64) == 0)? 1:0;
PortAccess.sw2 = ((input & 128) == 0)? 0:1;
PortAccess.Out32(888, 0);
delay();
```

If five selected pins of parallel port sent to PIC as ‘7’, mixer motor (‘motor1’) would activate. If five selected pins of parallel port sent to PIC as ‘6’, mixer motor (‘motor1’) would not activate. If five selected pins of parallel port sent to PIC as ‘9’, conveyor motor1 (‘motor2’) would activate. If five selected pins of parallel port sent to PIC as ‘8’, conveyor motor1 (‘motor2’) would not activate. If five selected pins of parallel port sent to PIC as ‘11’, filling valve motor for tank1 (‘motor3’) would activate.

If five selected pins of parallel port sent to PIC as ‘10’, filling valve motor for tank1 (‘motor3’) would not activate. If five selected pins of parallel port sent to PIC as ‘13’, filling valve motor for tank2 (‘motor4’) would activate. If five selected pins of parallel port sent to PIC as ‘12’, filling valve motor for tank2 (‘motor4’) would not activate. If five selected pins of parallel port sent to PIC as ‘15’, filling valve motor for mixer (‘motor5’) would activate. If five selected pins of parallel port sent to PIC as ‘14’, filling valve motor for mixer (‘motor5’) would not activate. If five selected pins of parallel port sent to PIC as ‘17’, robot arm motors (‘motor6’) would activate. If five selected pins of parallel port sent to PIC as ‘16’, robot arm motors (‘motor6’) would not activate. If five selected pins of parallel port sent to PIC as ‘19’, conveyor2 motor (‘motor7’) would activate. If five selected pins of parallel port sent to PIC as ‘18’, conveyor2 motor (‘motor7’) would not activate. If five selected pins of parallel port sent to PIC as ‘21’, sewing motor (‘motor8’) would activate. If five selected pins of parallel port sent to PIC as ‘20’, sewing motor (‘motor8’) would not activate. It can be written:

```
if (motor1 == 0){
   PortAccess.Out32(888, 8);
   delay();
   PortAccess.Out32(888, 0);
   delay();
}
else{
   PortAccess.Out32(888, 9);
   delay();
   PortAccess.Out32(888, 0);
   delay();
}
```

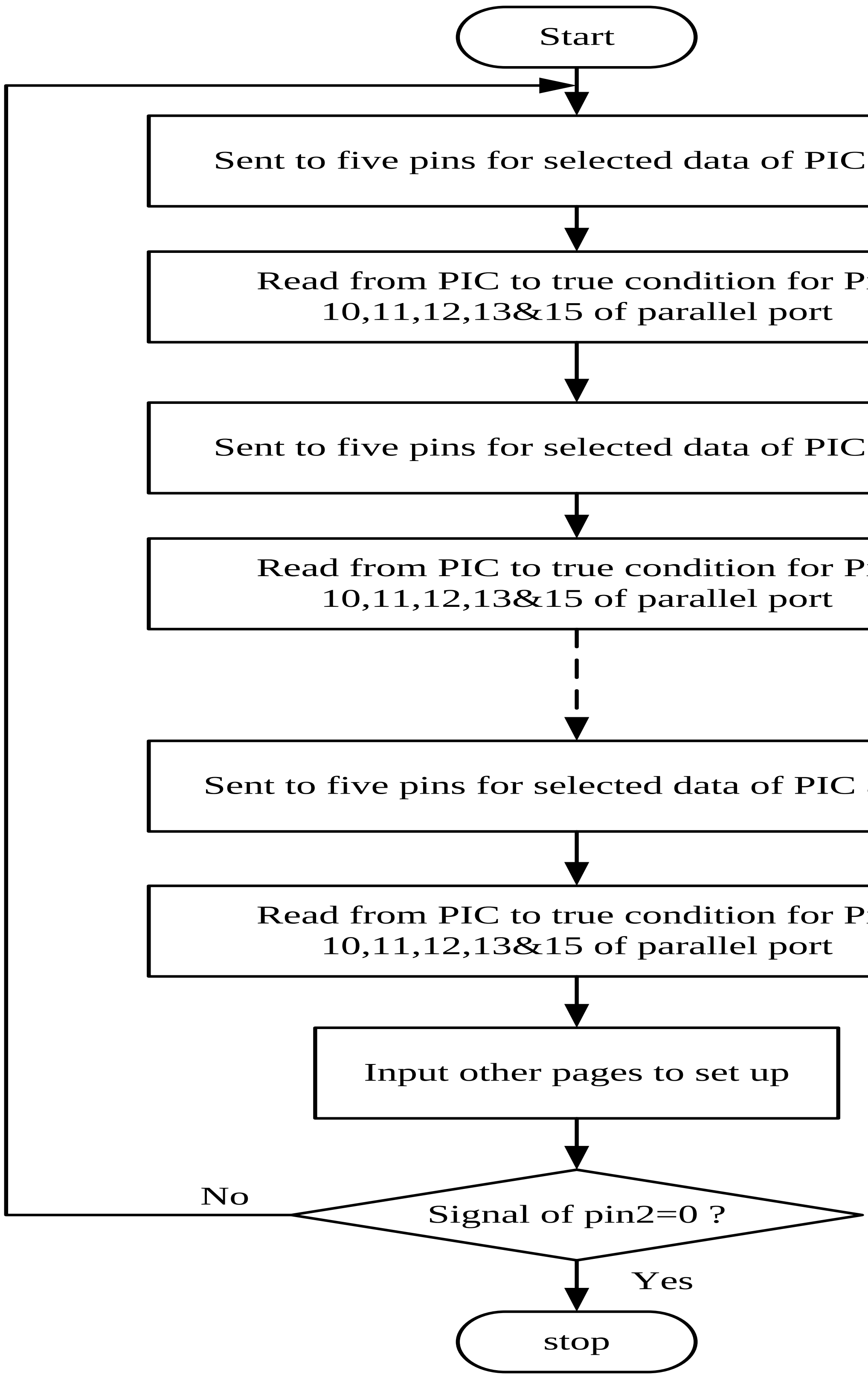

Start
Sent to five pins for selected data of PIC as 1
Read from PIC to true condition for Pin 10,11,12,13&15 of parallel port
Sent to five pins for selected data of PIC as 2
Read from PIC to true condition for Pin 10,11,12,13&15 of parallel port
Sent to five pins for selected data of PIC as 21
Read from PIC to true condition for Pin 10,11,12,13&15 of parallel port
Input other pages to set up
Signal of pin2=0 ?
No
Yes
stop

Figure 4: Flow Chart of Input/Output System

In the following program line, the input address are used 889 decimal numbers (379h).

```
inp = PortAccess.Inp32(889);
```

It includes the buttons as start, stop, instruction, hardware, and exit. Start/stop is used to true or false for the output system. It can be written:

```
For Stop, PortAccess.SystemStart = false;
For Start, PortAccess.SystemStart = true;
```

The main form is activated about the following codes and set the output address as 888 decimal numbers (378h);

```
PortAccess.Out32(888,0);
PortAccess.SystemStart = 0;
```

The error signal from the process activates every time. So it uses timer tool (tmr Error_Tick) to watch this condition. If error signal from input/output system is true, it plays the SystemSounds.Asterisk file in the computer for emergency sound and the back colour of error button changes black and red alternatively. If it is false, the back colour of error button is green. This system is always turns on. It is shown in Figure.5.

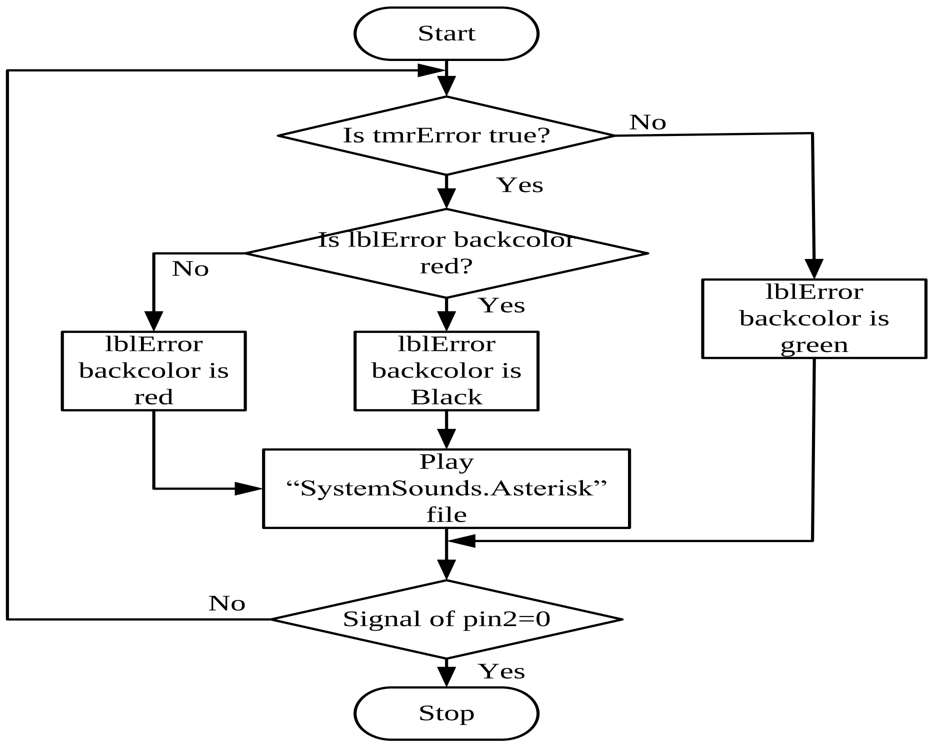

Start
Is tmrError true?
No
Yes
Is lblError backcolor red?
No
Yes
lblError backcolor is green
lblError backcolor is red
lblError backcolor is Black
Play "SystemSounds.Asterisk" file
No
Signal of pin2=0
Yes
Stop

Figure 5 Flow Chart of Error System

The operation button (btOperation) is used to open the simulation dialog box (frmOperation) for linking the real-time simulation process. It can be written:

```
frmOperation f2= new frmOperation();
f2.Show();
```

And the hardware button (btHardware) is utilized to open the hardware dialog box (frmHardware) for linking the whole process communication. It can be written:

```
frmHardware f3 = new frmHardware();
f3.Show();
```

The instruction button (btInstruction) is used to open the instruction dialog box (frmInstruction) including user guide for SCADA based Natural Fertilizer Manufacturing Process. It can be written:

```
frmInstruction f1 = new frmInstruction();
f1.Show();
```

The exit button (btExit) is used for closing the system. It includes the message box depending with the user's decision.

```
if (MessageBox.Show("Are you sure want to Exit?","SCADA",
MessageBoxButtons.YesNo,MessageBoxIcon.Question) == System.
   Windows.Forms.DialogResult.Yes) {
   Application.Exit();
}
```

4.3 Designing Program for Monitoring of Operation Page

This page uses the monitoring of components on the form (frmOperation). The signal of real-time monitoring for each device is get to display on the page by timer (timerIn_Tick) of main page.

So, each image is put on the form with picture box tool (imgName). Next, the text shows the condition of running process and the stopping process displays on the form by the condition of start button (low or high). It can be written:

```
If (PortAccess.SystemStart)
   lbState.Text="Running Condition of Process";
else
   lbState.Text="Stopping Condition of Process";
```

This system needs to test how the state of input pin is. So it uses timer tool from window application to display the components. The monitoring of running conveyor is used with input signal of timer tool (tmrConveyor_Tick) and it is shown in Figure.6. If the input signal of conveyor gets from input timer, the condition of running conveyor will show with the changing of images in the picture box on the form.

The signal of error is also used in the operation page. But the condition of error is shown by changing the colour of error button on the form. The flow of programme equals the program of above error. If the input pin (inError) is true, the color of error image (bnError) changes red and black alternatively. If it is not true, it displays the green colour.

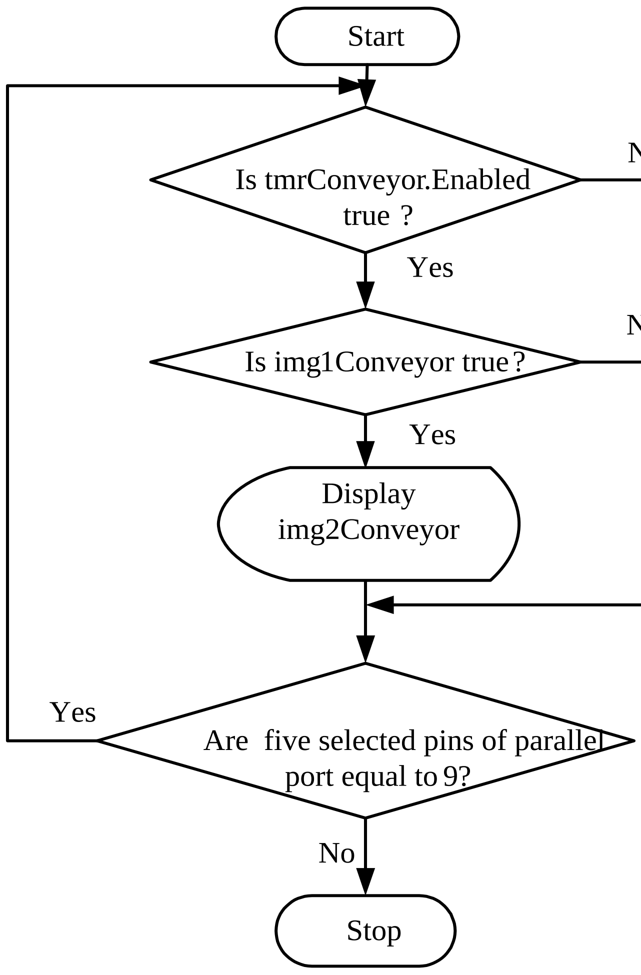


Figure 6: Flow Chart of Monitoring for Conveyor

The operation page includes four buttons. These are motor button, sensor button, counter button and back button. The motor button (btMotor) is used to open the motor dialog box (frmMotor) for linking the real-time simulation for running motors in the process. It can be written:

```
 frmMotor f = new frmMotor();
 f.Show();
```

The sensor button (btSensor) is utilized to open the sensor dialog box (frmSensor) for linking the real-time conditions of sensors in the process. It can be written:

```
frmSensor f1 = new frmSensor();
f1.Show();
```

And the counter button (btCounter) is used to open the counter dialog box (frmCounter) for linking the real-time number of bags finished overall packaging system in the process. It can be written:

```
frmCounter f2 = new frmCounter();
f2.Show();
```

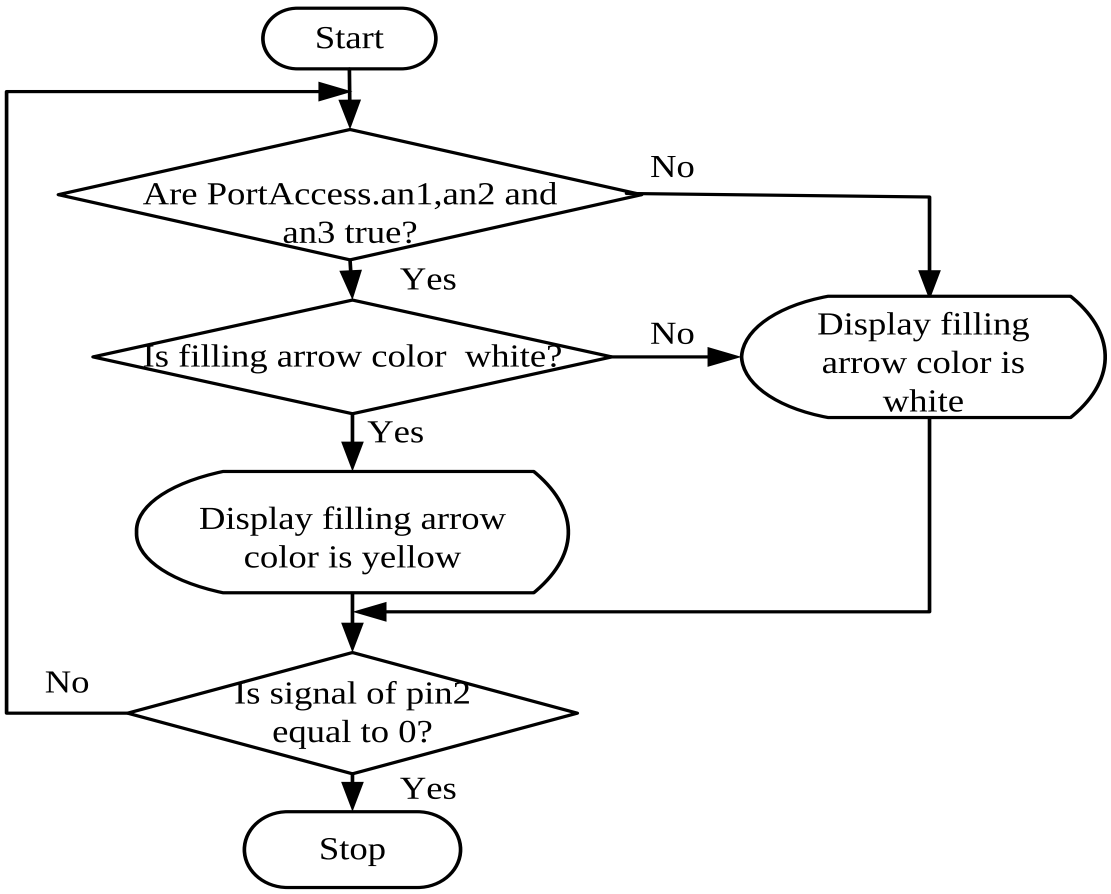

Start
Are PortAccess.an1,an2 and an3 true?
No
Yes
Is filling arrow color white?
No
Display filling arrow color is white
Yes
Display filling arrow color is yellow
No
Is signal of pin2 equal to 0?
Yes
Stop

Figure 7: Flow Chart of Monitoring for Filling Process

Finally, the back button (btBack) for closing the system is used. It comes back to the main page. In Figure.7, if weight sensor reached the specified weight level of tank1, the filling process would occur. The filling arrow color changes white and yellow alternatively. If weight sensor reached the specified weight level of tank2, the filling process would occur. The filling arrow color changed white and yellow alternatively. If weight level reached the specified weight level of bag, the filling process would occur. The filling arrow color changed white and yellow alternatively.

4.4 Implementation Program for Motor Page

This page is linked with the operation page to display the condition of motors included in the system. It is used by pressing the motor button (btMotor) from the operation page. The back button is used to back the operation page. The image for running motors has two images with design of motor. This system uses timer tool (timerIn_Tick) to get signals from input ports. The input signals of parallel port are used for mixing process (inMixer), transportion process (inConveyor1), filling process in the tank1 (inFillingValve1), filling process in the tank2 (inFillingValve2), carrying process (inConveyor2), filling process in the bag (inFillingValve3), the robot arm motors (inRobotArm) and sewing process (inSewing).

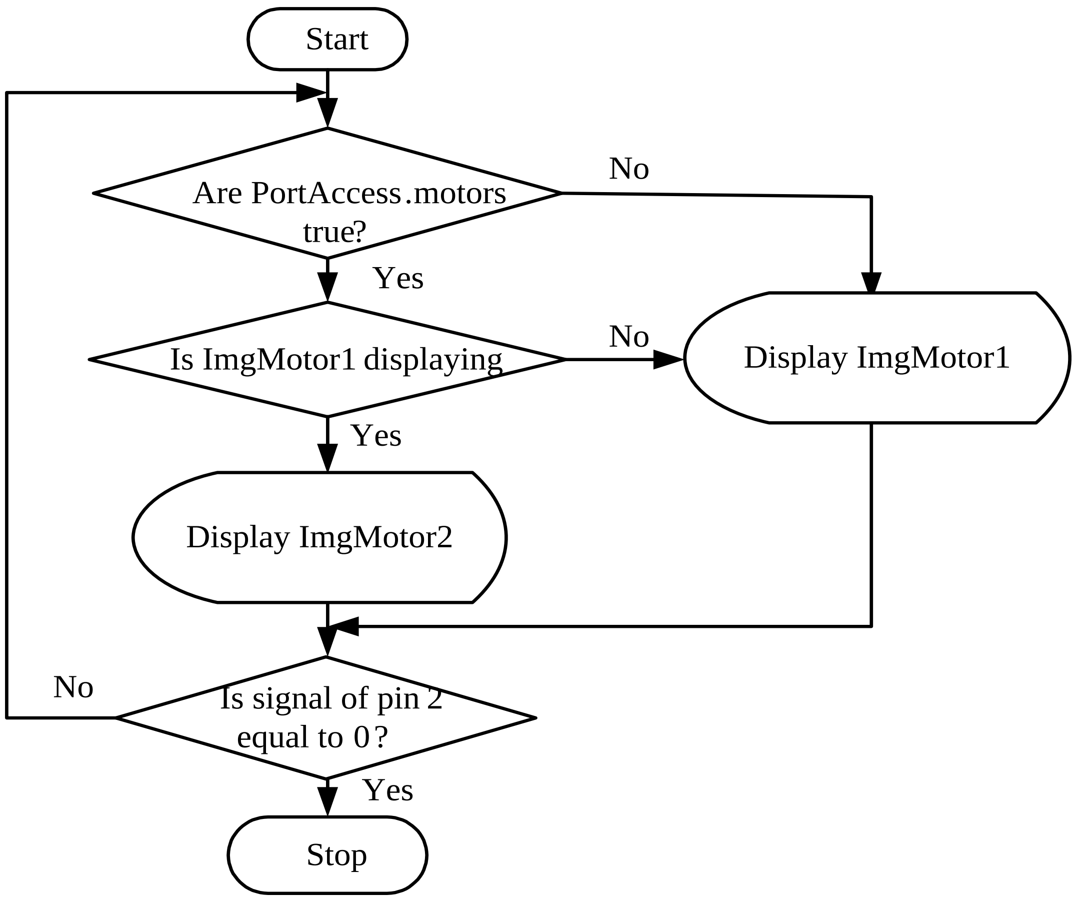


Figure 8: Flow Chart of Monitoring for Running Motors System

All of images for motors are displayed at the images in the picture boxes. If the signal of all input pins or each input pin is false, it displays the image of motor1 in each location of image as the images of mixer motor, conveyor motor1, filling motor1, filling motor2, conveyor motor2, filling motor3, robot arm motor and sewing motor (imgMixer, imgConveyor1, imgFM1, imgFM2, imgFM3, imgConveyor2, imgRAMs and imgSewing).

If the signal of each input pin is true, it displays the image of motor1 and the image of motor2 alternatively. It is shown in Figure.8. So, each image is put on the form with picture box tool (imgName). Next, the text shows the condition of running process and the stopping process displays on the form by the condition of start button (low or high). It can be written:

```
If (PortAccess.SystemStart)
   lbState.Text="Running Condition of Motors";
else
   lbState.Text="Stopping Condition of Motors";
```

4.5 Implementation Program for Sensor Page

This page is linked with the operation page to display the condition of sensors included in the system. It is used by pressing the sensor button (btSensor) form the operation page. The back button is used to go back the operation page. The image for the running sensors has two colours with design of sensor. This system uses timer tool (timerIn_Tick) to get signals from input ports.

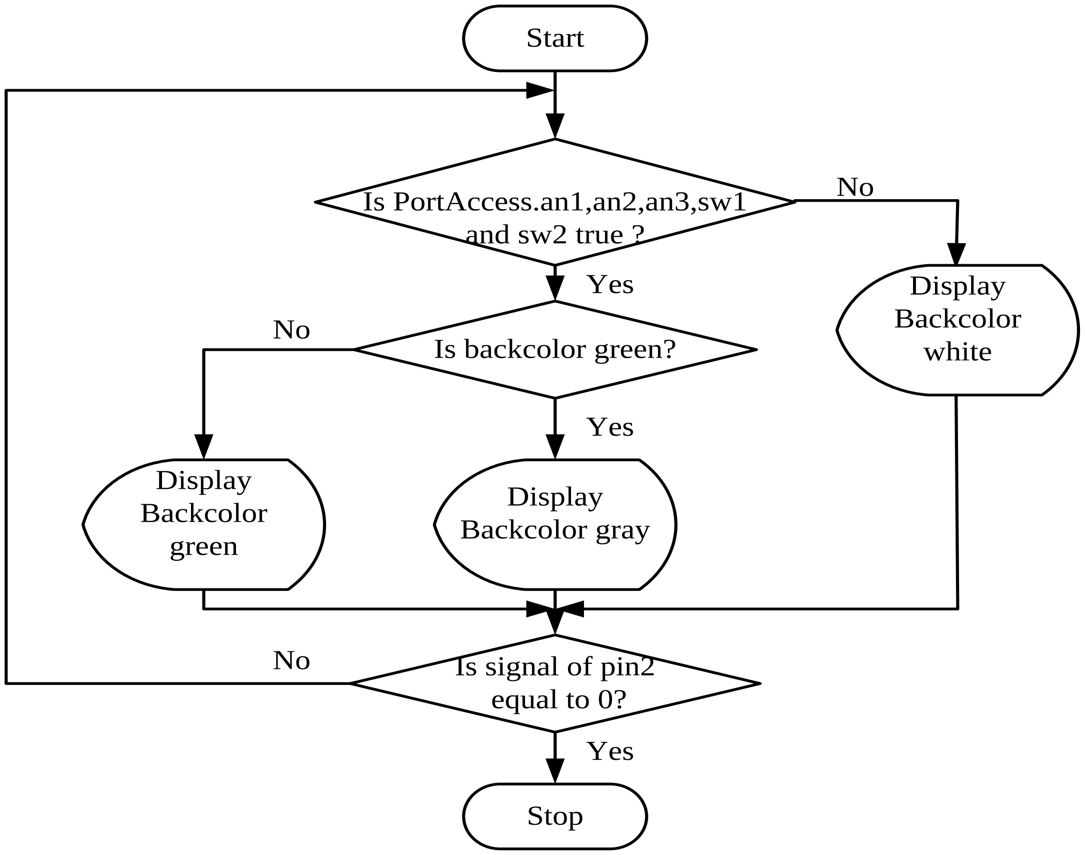
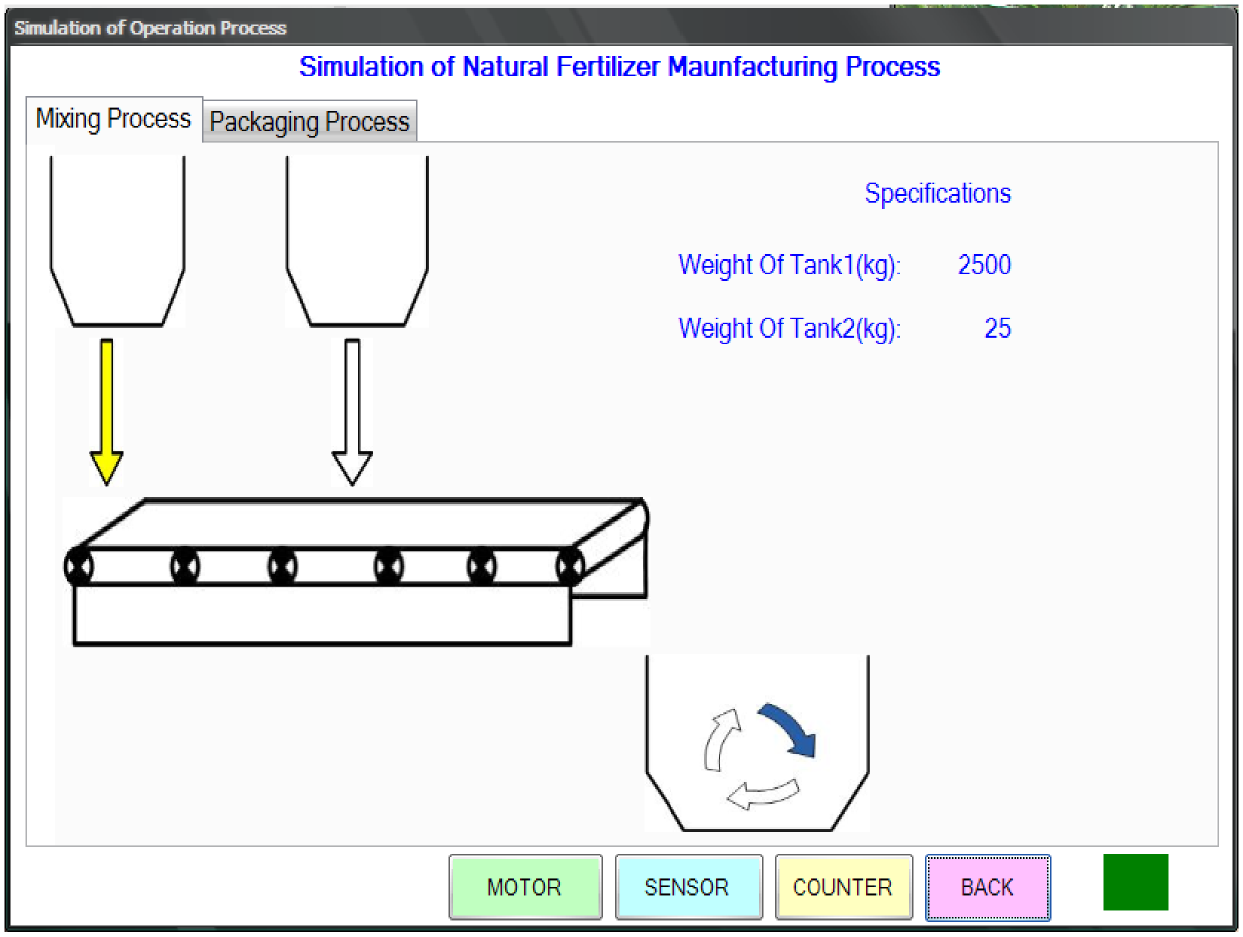


Figure 9: Flow Chart of Monitoring for Sensors

The input signals of parallel port are simultaneously used for weight sensor1 (inFillingValve1), weight sensor2 (inFillingValve2), weight sensor3, (inFillingValve3), position sensor1 (inRobotBase) and position sensor2 (inSewing). The image colors of sensor are displayed at the images in the pictur box. If the signal of all input pins or each input pin is true, it displays the green colour and gray color alternatively. If it is not, it displays the white colour image of sensor in each location of image. It is shown in Figure.9.

So, each image is put on the form with picture box tool (imgName). Next, the text shows the condition of running process and the stopping process displays on the form by the condition of start button (low or high). It can be written:

```
If (PortAccess.SystemStart)
   lbState.Text="Running Condition of Process";
else
   lbState.Text="Stopping Condition of Process";
```


Simulation of Operation Process
Simulation of Natural Fertilizer Maunfacturing Process
Mixing Process
Packaging Process
MOTOR
SENSOR
COUNTER
BACK

Specifications

Weight Of Tank1(kg): 2500

Weight Of Tank2(kg): 25

Figure 10: Screenshot of Simulation Result of Operation Window (Mixing Process)

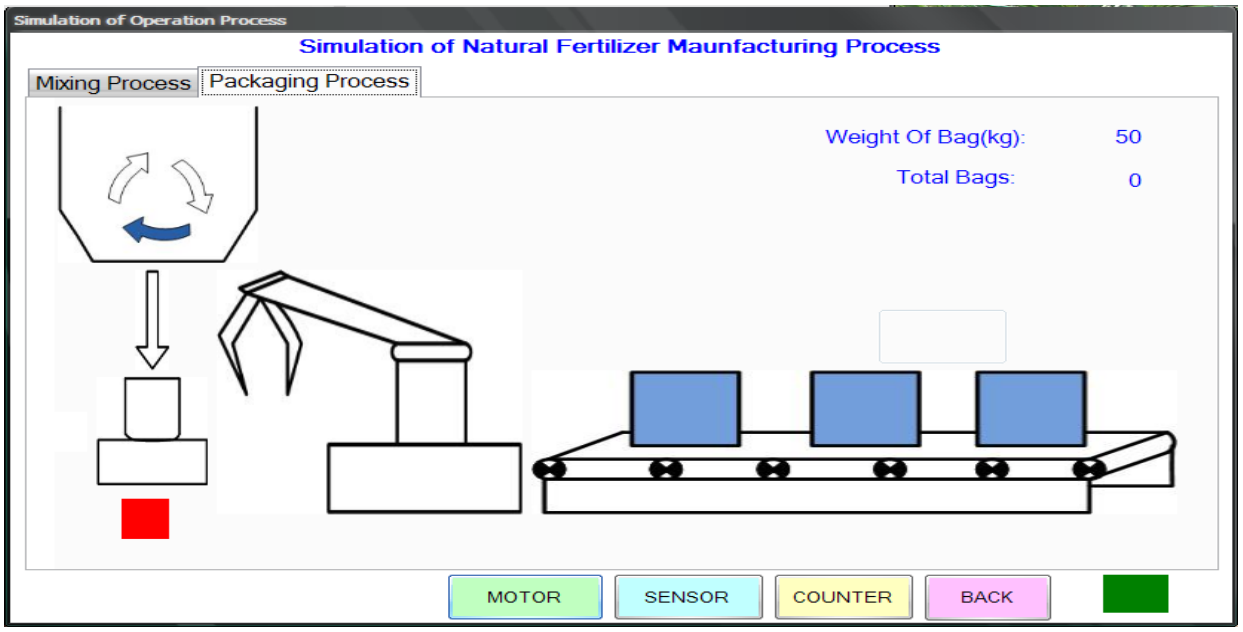
5 Simulation Result of SCADA Based Monitoring System

In SCADA based manufacturing process, parallel port pins are used to control the input and output signal of hardware devices. And it is linked window application in the monitoring system. The monitoring system includes main window which links other windows, operation windows which links the device operations, motor operation window which is running by each state, counter window which counts bags, sensor window which expresses the working condition of sensors, hardware window which links the monitoring of hardware device sample and instruction window which expresses as the user guide.

5.1 Simulation Result of Operation Window

This window includes two portions. Figure.10 illustrates mixing process and filling process. Figure.11 illustrates filling process and packaging process. This window is a main process to show for overall process. And it includes four buttons- motor, sensor, counter and back.

On the window form, it is designed with working condition of the sample components for the real devices of natural fertilizer manufacturing process by using their images of actual devices. It shows the monitoring of changing or moving condition of sample components. It uses input/output signal from main window for operation of device by using each signal. This window shows the amount of weight level of tank1 and tank2. The back button is used to go back the main window like to close the operation window. The motor button is utilized to display the running condition of motors due to each input signal by linking with it. The sensor button is used to express the working condition of sensors by linking with it. The counter window is used to display the number of bag.


Simulation of Operation Process
Simulation of Natural Fertilizer Maunfacturing Process
Mixing Process
Packaging Process
MOTOR
SENSOR
COUNTER
BACK

Weight Of Bag(kg): 50
Total Bags: 0

Figure 11: Screenshot of Simulation Result of Packaging Window

5.2 Simulation Result of Hardware Window

It includes the hardware images like the components of SCADA based system. If the user clicks the start button, the overall system will operate and the arrow will show on the form. It links with the main page. The back button is used to go back the main window. It is shown in Figure.12.

5.3 Simulation Result of Sensor Window

This window shows the working conditions of sensor by changing the colors of sensor images. It includes five sensors- weight sensor for Tank1, weight sensor for Tank2, weight sensor for bag, position sensor1 and position sensor2.

If the sensors get the signal from the input pins of parallel port, the changing color of sensor will show green and gray alternatively on the form. It is linked with the operation window. Figure.13 shows simulation result of the sensor window. This window describes the weight level of tank1 (weight sensor1), the weight level of tank2 (weight sensor2) and the weight level of bag (weight sensor3). The operator can see the amount of weight level. The back button is used to go back the operation window.

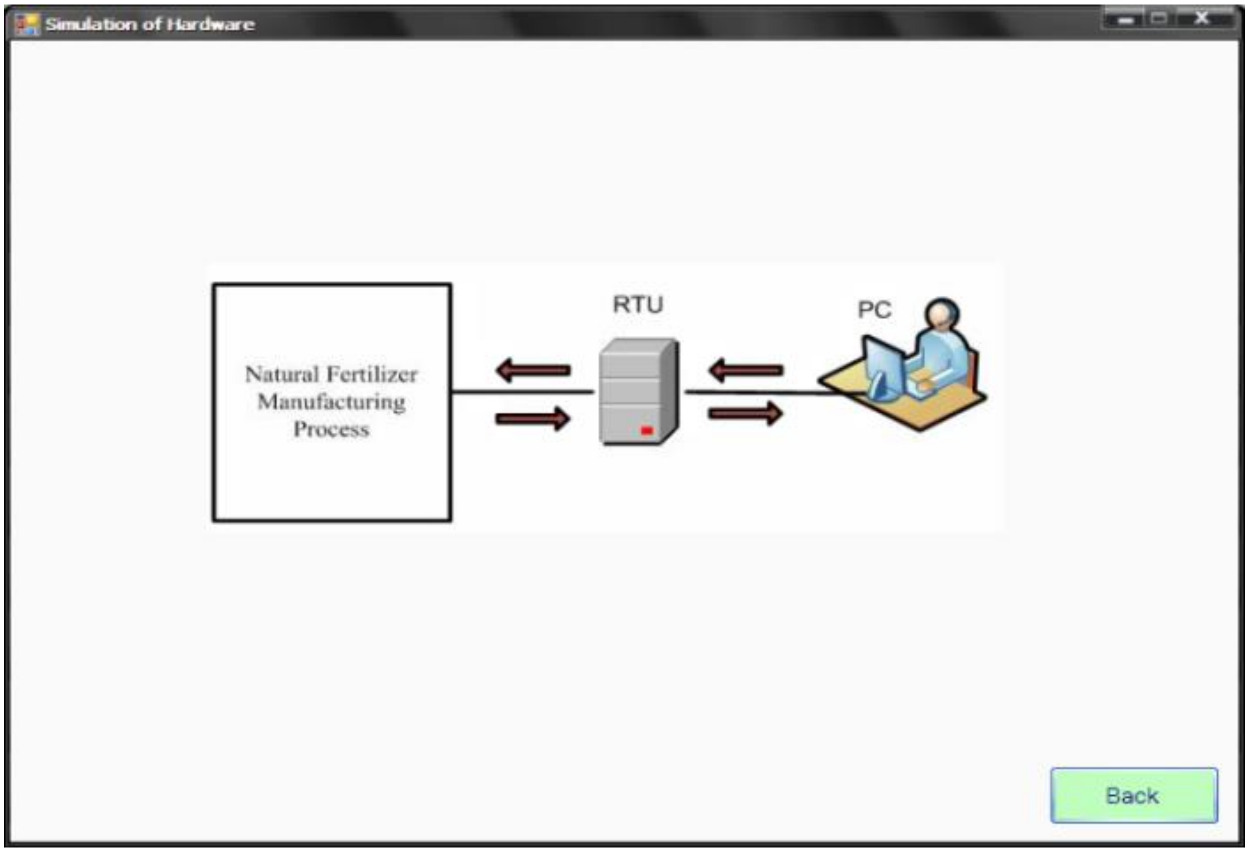


Figure 12: Screenshot of Simulation Result of Hardware Window

5.4 Simulation Result of Motor Window

It includes eight motors- mixer motor, conveyor motor1, tank1 valve motor, tank2 valve motor, mixer valve motor, robot arm motors (robot base motor, robot arm motor and robot gripper motor), conveyor motor2 and sewing motor. This window shows the images of motor like the moving of motors for the process.

If motors get the signal from the input pins of parallel port, the running condition of motors will show in the form. It is linked with operation window. The back button is used to go back the operation window. Figure.14 shows the screenshot of simulation result of the motor window.

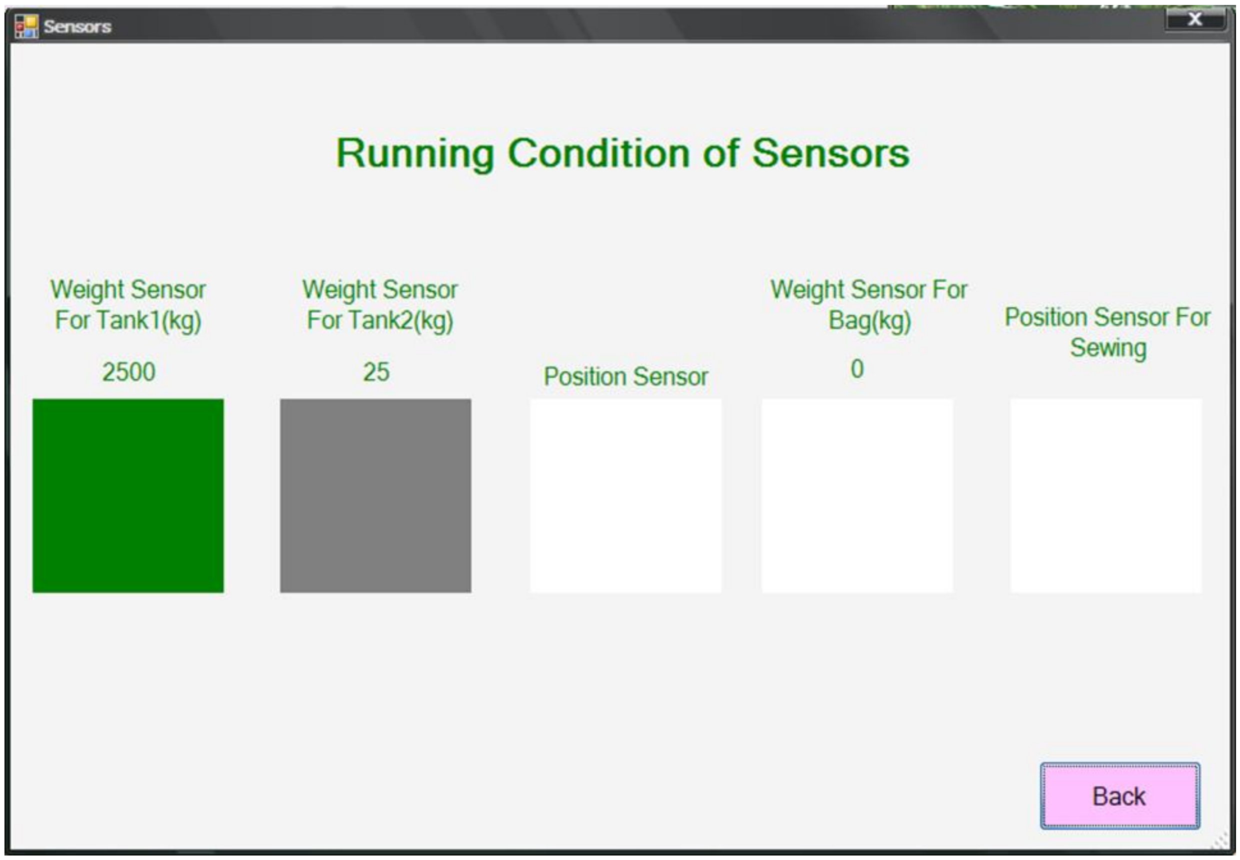

Sensors
Running Condition of Sensors
Weight Sensor For Tank1(kg)
2500
Weight Sensor For Tank2(kg)
25
Position Sensor
Weight Sensor For Bag(kg)
0
Position Sensor For Sewing
Back

Figure 13: Screenshot of Simulation Result of Sensor Window

5.5 Simulation Result of Counter Window

This window shows the images of bag like the number of bags for the real-time manufacturing process. It includes five bags for one cycle operation. It shows the number of bags which is packed after it has filled at the real devices from hardware process. It is linked with the operation window. The back button is used to go back the operation window. The operator can change the number of bags per one cycle. Figure.15 shows the screenshot of simulation result of the counter window.

5.6 Simulation Result of Exit Window

This window shows the decision making for the user to run or stop of the current condition of the overall system. It uses the message box including yes or no decision that is if it is yes, it will exit from the monitoring system. If it is not, this process will continue the operation. It is shown in Figure.16.

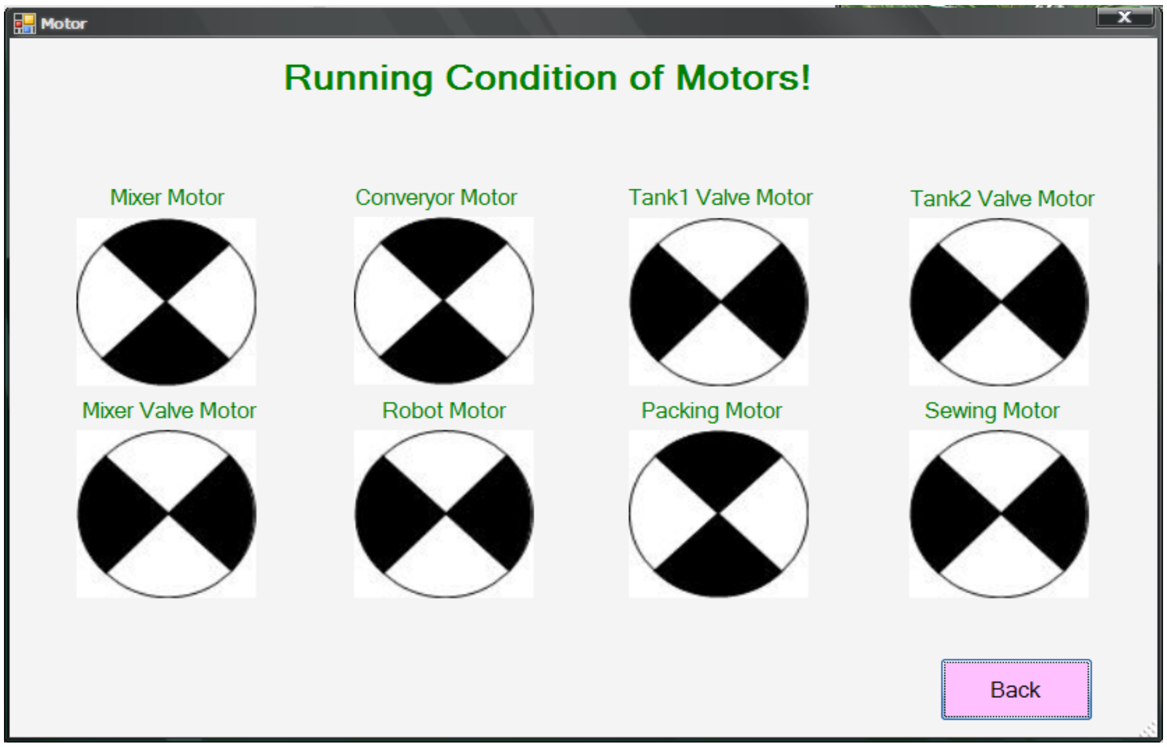


Figure 14: Screenshot of Simulation Result of Motor Window

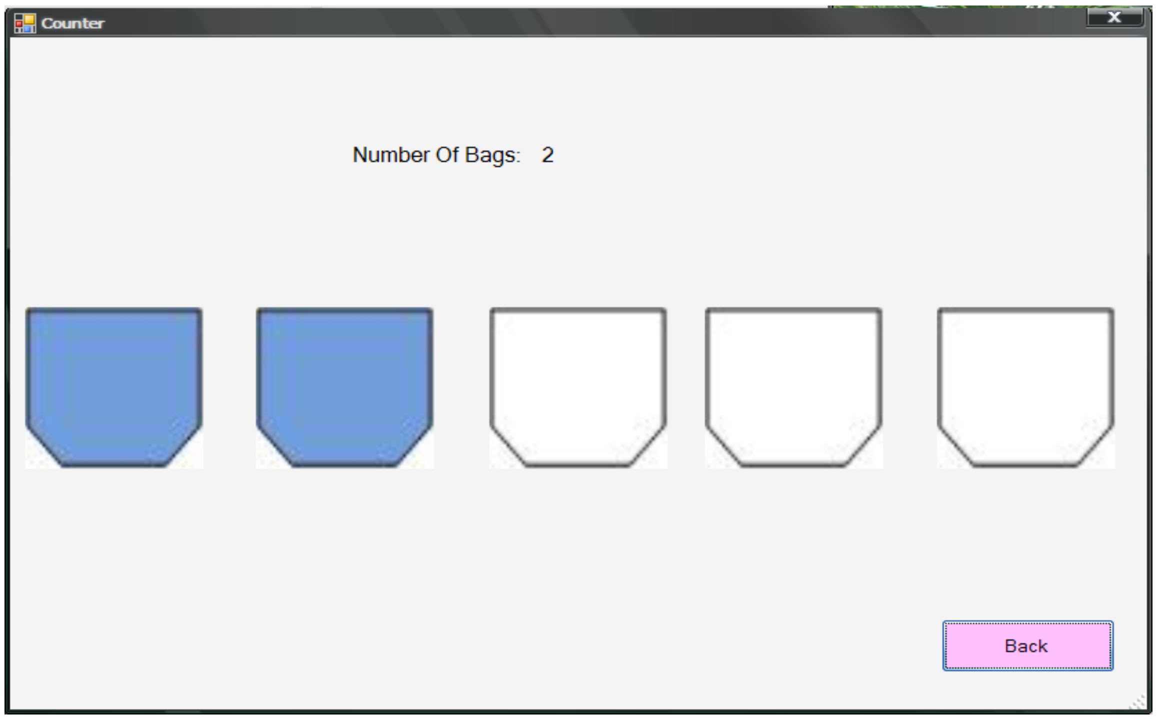
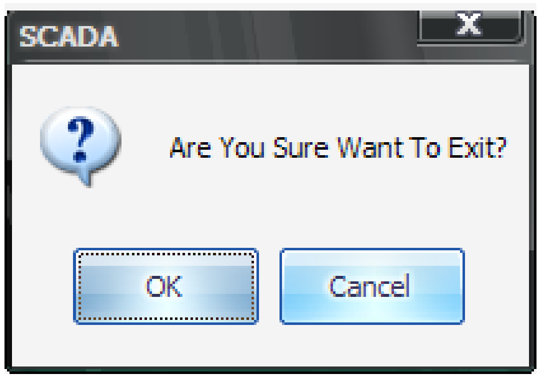


Figure 15: Screenshot of Simulation Result of Counter Window


SCADA

Are You Sure Want To Exit?

Figure 16: Screenshot of Simulation Result of Exit Window

6 Conclusion

In this work, a low cost data acquisition, processing and monitoring system based SCADA system has successfully developed. And the simulating and monitoring for the Natural Fertilizer Manufacturing Process use own programs without popular SCADA software. Communication system uses parallel port and interfacing circuit (including optocouplers). Hardware unit for this process is designed with the small I/O sample model and tested by connecting hardware and computer. And then it can start/stop with using monitoring system and can see all of running state for this manufacturing process. Although this system cannot use many devices that could be monitored or controlled, this paper has shown a basic application of process control. All of the equipment used in the demonstration meets the normal industry standards, and could be found in automation systems around the world. Although this is a very small demonstration of the process control compared to most industrial application, the processes and principles used for this paper are still the same regardless of the size of the system.

References

[1] Jose Angel Gomez Gomez: “Survey of SCADA Systems and Visualization of a real life process”, Sweden, (2000).

[2] Shwe Si Wa: “Natural Fertilizer Manufacturing Process”, Mandalay Technological University, Myanmar, (2011).

[3] Hla Myo Tun: “*Distributed Control System for Vehicle Spare Parts Manufacturing Plant” (Real Time Graphical User Interface Monitoring and Neworking System)*, Ph.D paper, (2008).

[4] John Sharp: “Microsoft Visual C# 2008 Step by Step”, by Microsoft Press, Washington, (2009).

[5] Bradley-Millspaugh: “*Programming in Visual C# 2008”*, By McGraw-Hill Primis, United States of America, (2009).

[6] Ronald L. Krutz, “Securing SCADA Systems”,Wiley Publishing Inc, Indianapolis, Indiana, (2005)