UACEE International Journal of Advances in Electronics Engineering
Vol:1 Issue:1 ISSN 2278 - 215X

Design of Conveyor Monitoring Controller with Wireless System

Mrs.Sheeja S.Suresh [1], Mr.P.H.Rangaree[2]
[1] Research scholar, G.H.Raisoni College of Engg./Electronics, Nagpur, India
Email: sheeja_suresh@rediffmail.com
[2] Asst. Professor, G.H.Raisoni College of Engg./Electronics and Telecommunication, Nagpur, India
Email: pankajrangari12@rediff.com

Abstract — **In this paper we design a conveyor monitoring controller with wireless system. Conveyor systems are one of the commonly used devices for the transportation of parts from one workplace to another and for horizontal transport in manufacturing systems. In today's world it is difficult to get manpower, besides transportation by means of truck or railways is also not economical in some applications. In such places wireless control of the conveyor controller plays an important role that is to reduce the manual work and save funds for the enterprise.**

This proposed design is highly suitable for applications running at extreme temperature and pressures where human control is difficult. Also immediate response at both the ends is achieved, including safety measures. In this paper we design a Prototype model of conveyor controller, BASCON-51 coding of microcontroller, wireless monitor using VB.net programming. Measurement Techniques with calculations of conveyor speed at the locate site controller along with counter calculations is also explained.

***Keywords*— Vb.net : Visual Basic.net**
BASCON51 : Basic Compiler-51 coding

I. INTRODUCTION

Conveyor systems are one of the commonly used devices for the transportation of parts from one workplace to another and for horizontal transport in manufacturing systems. In today's world it is difficult to get manpower, besides transportation by means of truck or railways is also not economical in some applications. In such places wireless control of the conveyor controller plays an important role that is to reduce the manual work and save funds for the enterprise.

Conveyors join processes together, deliver and take away goods automatically and on cue. They can store goods until the next person or machine wants them, sort out goods or merge them intelligently and can facilitate a number of functions that add value to products, such as packing and weighing.

Powered Belt Conveyors – In our design, this carry the product fully by a moving belt, which is supported by either a sheet metal slider bed or rollers mounted in the frame. It is most practical and economic means of transporting goods from level to level, most inexpensive of all powered conveyors and very safe, lowest cost powered conveyor, highest throughput capacity and speed, lowest noise rating, handles widest range of goods (bags, parcels, boxes, sacks), maintains gaps and orientation, can handle declines/inclines up to 30°, suits high volumes and is reversible, belt on roller options for heavier loads.

This proposed design is highly suitable for applications running at extreme temperature and pressures where human control is difficult. Also immediate response at both the ends is achieved, including safety measures.

Advantages of this design : This design provides a common design environment, which facilitates basic general design study of conveyor controller and handshaking signals through interfacing with remote site vide vb.net GUI.

- After getting the basic ideas, engineers can locate and correct errors early in system design, when the time and financial impact of system modification are minimized.
- Design reuse, for upgrades. Real time and fast processing is achieved
- The microcontroller and VB.net interfacing software and settings can be easily modified.
- Less number of operators required.
- Cost Effective system.
- I created my own software embedded timers in BASCON51 coding which enabled the microcontroller at the local site to have multitask features.
- This software and coding can be installed on laptop/ notebook computers successfully ensuring wireless control from any place via LAPTOP within the area of RF wireless range. Effective multitasking of local site microcontroller.
- Effective handshaking between the Local and Remote site.

II. METHODOLOGY

Design :

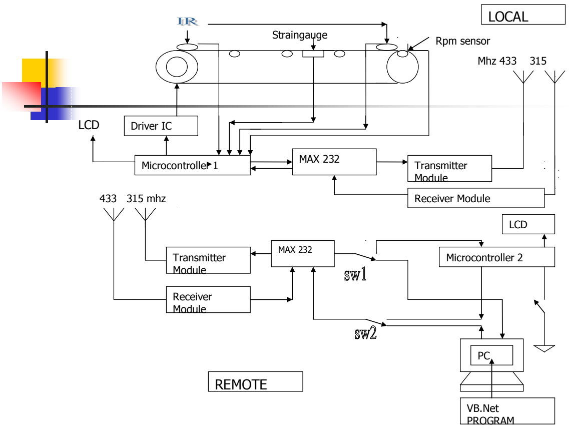

Fig 1: Design of the conveyor monitoring controller with wireless system.

Design description :

The Vb.net programme is used to create the GUI at the Remote site. In the GUI we have two buttons, one for opening the COM port and one for closing the COM port.One big window is used to display the status on the serial port, this window is called the Raw window.

One button for checking the detection of object, If the object is present then automatically the command for turning on the motor is issued from microcontroller1.After counting 10 object the motor stops automatically. Even if there is no object for 2 minutes then also the " Belt stopped " message appears on the LCD display and also on the raw window.

On the local site there is a On/off switch provided to switch On/Off the conveyor manually in case of emergency. This features is also included in the Vb.net coding from remote site.The local site i.e the actual conveyor design model there is a IR sensor named IR1 connected to the conveyor master roller, initially when no object is detected by the IR sensor, it outputs logic one to microcontroller 89S52 input port P1.0 at local site, which in turn keeps the motor driver P2.0=0 in off condition disabling the motor and also the master roller.

When the object is placed on the belt positioned at master roller, it is detected by the IR1 sensor and a logic zero is sent to local site microcontroller1, Then this microcontroller1 starts the motor though the motor driver IC, also this data signal is transmitted at 433 MHZ transmitter via MAX232 to the Remote site, and then all this communication on the serial port is displayed simultaneously on the VB.Net Raw window.Now the message "Object detected ! Process starts…" will come across the screen of the PC, While the counter is incrementing as it detects the object, anytime the conveyor can be stopped by clicking " motor off " button on Vb.net GUI at the remote site.

If the microcontroller achieved all the set parameter then this also is fed to the local site where the local site microcontroller keeps the motor in off condition. In this way we can control the conveyor by wireless handshaking signals between the local and remote site.

We can program the counter connected to the second IR sensor at the follower roller. As the object reaches the second IR2 sensor, it senses the object and increment the counter, this module is programmed here to count continuous 10 pieces of object intake.

There is a RPM sensor SW1 connected to the follower roller. This is a proximity sensor working on capacitive principle. When there is no object, output will be logic 1, when object is detected the output will be logic 0 i.e.0 volts. It is used to detect the motor speed, output will go to counter to count rpm, It is useful when the belt is broken or slipped and then to stop the driver motor though microcontroller.

Straingauge sensor is connected at the middle of the belt conveyor to measure the weight of the belt. It can measure up to 4.5 Kg, its output will be in millivolts, hence will be connected to signal conditioning ckt. for amplification. The strain gauge output is given to Load transmitter which in turn is transmitted to Remote Site to be displayed on GUI raw window, alongwith LCD display on Local site.

Microcontroller2 is kept for future purpose, like if the pc with Vb.net GUI has some problem, then in that case, this wireless system can also be controller though microcontroller2 with LCD display. Hence we have given a manual switch so that control can be given by VB.net GUI PC or through the Microcontroller2 in case of emergencies.

Also this wireless connectivity is possible through Notebook/ Laptop that is the VB.Net GUI can be installed on laptop successfully without any change in program. This way we can control the conveyor system wirelessly without using a desktop PC.

III.BASCON – 51 coding for local site microcontroller 89S52

```
' ************ Compiler Directives *************
$regfile = "80522.DAT"
$romstart = &H0
$crystal = 11059200
$baud = 9600
' ************* Pin Definitions *****************
Lcd = 16 * 2
Lcdpin = Pin , Db4 = P0.4 , Db5 = P0.5
Lcdpin = Pin , Db6 = P0.6 , Db7 = P0.7
Lcdpin = Pin , E = P0.2 , Rs = P0.0
```

```
Rw Alias P0.1
Prox1 Alias P1.0                 'FOR IR1
sw1 Alias P1.1                   'FOR RPM
Ir1 Alias P1.0                  'FOR OBJ DETECTION
Ir2 Alias P1.2                  'FOR OBJ CNTR
Belt Alias P2.0                 'FOR MOTOR ON
Dim S As Integer
Dim A As Integer , D As Integer
Dim C As Integer
Dim Count As Integer
Dim T0 As Single
Dim Rpm As String * 10
Dim T1 As Integer
Dim T2 As Integer
Dim T3 As Integer
Dim V0 As Single
Dim V1 As Byte
' ******* Main Program *******
  P2.0 = 0
  P2.1 = 1
  P2.2 = 1
  P2.3 = 1
  Rw = 0
  A = 0
  D = 0
  V1 = 0
  Lcdinitialised
      Cls
      Upperline
      Lcd "     HELLO      "
      'Goto Main3

   Main3:
      P2.0=Pwm_initialised,190;
 ' pulse width modulated 75% duty cyle pulse given to
motor to reduce speed
      Do
      Debounce Ir1 , 0 , Irsens , Sub
      Upperline
      Lcd "WAIT FOR OBJECT"
      Print "WAIT FOR OBJECT"
      Loop
    Irsens:
      Belt = 1
      Upperline
      Lcd "OBJECT DETECTED"
      Print "OBJECT DETECTED"
      Goto Main
      'Return
   Main:
      Do
      Debounce Prox1 , 0 , Strt1 , Sub
      T0 = 0
      D = D + 1          ' RPM check counter
      If D > 1000 Then
      D = 0
      Belt = 0
      Goto Trip
```

```
   End If
   Waitms 10
   Loop
  Strt1:
   Do
   Debounce Prox1 , 0 , Stp , Sub
   Thirdline
   Lcd A ; "             "
   A = A + 1
   Print "A=" ; A
     If A > 1000 Then
     A = 0
       Goto Trip
     End If
   Waitms 1
   Loop
  "
  Trip:
     S = 0
   Cls
   Do
   Debounce Sw1 , 0 , Jump , Sub
   Upperline
   Lcd "..BELT STOPPED.."
   Print "BELT STOPPED...."
   Belt = 0
   Waitms 10
   Loop
  Jump:
  Goto Main3

  Stp:
   T0 = 100 / A
   T0 = T0 * 60
   T0 = T0 + 14
   Rpm = Fusing(t0 , ##)
   Cls
   Lowerline
   Lcd "RPM=" ; Rpm ;

   A = 0
   D = 0
   Belt = 1
    If T0 < 25 Then
     Goto Trip
    End If
    T0 = 0
    Goto Main4
End

Main4:
  T1 = 1
  Do
  T1 = T1 + 1
  Debounce Ir2 , 0 , Rst , Sub
  If T1 > 1000 Then
  Goto Trip
  End If
 ' S = S + 1
```

```
    If S > 10 Then
     S = 0
    Goto Trip
    End If
      If T1 = 0 Then
      Goto Strt1
      End If
  Lowerline
  Lcd "Rpm=" ; Rpm ; " Obj=" ; S ; "   "
  Print "RPM=" ; Rpm;
  Print "Objects count=" ; S
  Fourthline
  Lcd "p1=" ; P1;
  Loop
Rst:
T1 = 0
S = S + 1
Return
```

IV. VB.Net GUI CODING

```
Imports PCComm
Public Class frmMain
    Private comm As New CommManager()
    Private transType As String =
String.Empty

    Private Sub
cboPort_SelectedIndexChanged(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles
cboPort.SelectedIndexChanged
        comm.PortName = cboPort.Text()
    End Sub

    ''' <summary>
    ''' Method to initialize serial port
    ''' values to standard defaults
    ''' </summary>
    Private Sub SetDefaults()
        cboPort.SelectedIndex = 0    '
for com1 selection because index start
from "0" only in vb.net
        cboBaud.SelectedText = "9600"
        cboParity.SelectedIndex = 0
        cboStop.SelectedIndex = 1
        cboData.SelectedIndex = 1
    End Sub

    ''' <summary>
    ''' methos to load our serial
    ''' port option values
    ''' </summary>
    Private Sub LoadValues()
        comm.SetPortNameValues(cboPort)
        comm.SetParityValues(cboParity)
        comm.SetStopBitValues(cboStop)
    End Sub
    ''' <summary>
    ''' method to set the state of
controls
    ''' when the form first loads
    ''' </summary>
    Private Sub SetControlState()
        rdoText.Checked = True ' for
sending data in text
        cmdSend.Enabled = False ' for
not to sending text on port
        cmdClose.Enabled = False   ' for
sending data on port to stopped
    End Sub
    Private Sub frmMain_Load(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Button1.Enabled = False
        Button2.Enabled = False
        LoadValues()
        SetDefaults()
        SetControlState()
    End Sub
    Private Sub cmdClose_Click(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles cmdClose.Click
        comm.ClosePort()
        'SetControlState()
        SetDefaults()
    End Sub
    Private Sub cmdOpen_Click(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles cmdOpen.Click
        Button1.Enabled = True
        Button2.Enabled = True
        comm.Parity = cboParity.Text
        comm.StopBits = cboStop.Text
        comm.DataBits = cboData.Text
        comm.BaudRate = cboBaud.Text
        comm.DisplayWindow = rtbDisplay
        comm.OpenPort()

        cmdOpen.Enabled = True
        cmdClose.Enabled = True
        cmdSend.Enabled = True
        timer1.enabled = True
    End Sub
    Private Sub cmdSend_Click(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles cmdSend.Click
        comm.Message = txtSend.Text
        comm.Type =
CommManager.MessageType.Normal
        comm.WriteData(txtSend.Text)
        txtSend.Text = String.Empty
        txtSend.Focus()
    End Sub
    Private Sub
rdoHex_CheckedChanged(ByVal sender As
System.Object, ByVal e As
System.EventArgs) Handles
rdoHex.CheckedChanged
        If rdoHex.Checked() Then
            comm.CurrentTransmissionType
=
PCComm.CommManager.TransmissionType.Hex
        Else
        comm.CurrentTransmissionType =
PCComm.CommManager.TransmissionType.Text
        End If
    End Sub
    Private Sub
cboBaud_SelectedIndexChanged(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles
cboBaud.SelectedIndexChanged
        comm.BaudRate = cboBaud.Text()
    End Sub
Private Sub
cboParity_SelectedIndexChanged(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles
cboParity.SelectedIndexChanged
        comm.Parity = cboParity.Text()
    End Sub
    Private Sub
cboStop_SelectedIndexChanged(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles
cboStop.SelectedIndexChanged
        comm.StopBits = cboStop.Text()
    End Sub
    Private Sub
cboData_SelectedIndexChanged(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles
cboData.SelectedIndexChanged
        comm.StopBits = cboStop.Text()
    End Sub
    Private Sub
rtbDisplay_TextChanged(ByVal sender As
System.Object, ByVal e As
System.EventArgs) Handles
rtbDisplay.TextChanged
        TextBox1.Text = rtbDisplay.Text
    End Sub
    Private Sub Timer1_Tick(ByVal sender
As System.Object, ByVal e As
System.EventArgs) Handles Timer1.Tick
        rtbDisplay.Clear()
        Timer1.Interval = 1000
    End Sub
    Private Sub Button1_Click(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

        txtSend.Text = "4"
        comm.Message = txtSend.Text
        comm.Type =
CommManager.MessageType.Normal
        comm.WriteData(txtSend.Text)
        txtSend.Text = String.Empty
        txtSend.Focus()
    End Sub
    Private Sub Button2_Click(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        txtSend.Text = "3"
        comm.Message = txtSend.Text
        comm.Type =
CommManager.MessageType.Normal
        comm.WriteData(txtSend.Text)
        txtSend.Text = String.Empty
        txtSend.Focus()
    End Sub

    Private Sub
TextBox1_TextChanged(ByVal sender As
System.Object, ByVal e As
System.EventArgs) Handles
TextBox1.TextChanged
    End Sub
    Private Sub GroupBox1_Enter(ByVal
sender As System.Object, ByVal e As
System.EventArgs) Handles
GroupBox1.Enter
    End Sub
               End Class
```

V. TOOLS

A methodology is a collection of methods, techniques, tools and documentation aids which help the system developers analyze, design and implement a software system. This section, describes the design methods used to conduct the research and the fact finding techniques. We used Various design techniques in BASCON 51 for programming the microcontroller, VB.net for interfacing desktop PC/laptop with microcontroller. Sensors used in design include proxy-IR sensors, capacitive IR sensor and strain gauge.

VI. RESULT

Studied and designed prototype model of Conveyor controller.

Studied and designed wireless monitoring system and designed VB.net code simulation for wireless control of conveyor.

Achieved successful handshaking between the Local and Remote side of the Conveyor.

VII. CONCLUSION

Design of Prototype model of conveyor controller, wireless monitor using VB.Net programming is done.

Measurement Techniques with calculations of conveyor speed at the locate site controller along with counter calculations is also explained.

VIII. FUTURE SCOPE

In future, with additional hardware and software we can give connectivity through wireless GSM vide satellite specially in areas of polar and desert sites where human control is difficult due to adverse climatic conditions.
Also in military and defense applications, this can be modified to provide automated transportation of explosive materials between distant places through conveyor controller systems,that is between military base units.

IX. REFERENCES

[01] TI – "Modeling of conveyors and their equipment interfaces", E.J.Williams, S.Sadakane, PY-1996, VL-0J A-Winter Simulation Conference (WSC1996)

[02] M. A. Alspaugh, "Longer Overland Conveyors with Distributed Power and Controls", Presented at Rockwell Automation Fair, St Louis, MO, November 15, 2005