The Code of CR1000

'CR1000 Series Datalogger Program

'To create a different opening program template, type in new

'instructions and select Template | Save as Default Template

'date:

'Declare Variable

Const STATION\_NAME = "CR1000" 'Define station name

Const On = true 'Assign "On" as Boolean True

Const Off = false 'Assign "Off" as Boolean False

Const Escape = "Hit Esc" 'Word indicates action to exit dead end

Dim Commands As String 'Define commands used to declare a web page

Public EXO2\_SDI12\_address = 0 'Define SDI12 address of EXO2 is 0

Public AWS\_SDI12\_address = 1 'Define SDI12 address of AWS600 is 1

Public RunSDI12 As Boolean 'Define SDI12 operation variable

Public Port(8) As Boolean 'Define Eight interface

Public StartFlag As Boolean 'Pump Control Process Variable

Public CountDown1 = 30 'Pump Count Down Variable

Public CountDown2 = 30 'Pump Count Down Variable

Public CountDown3 = 30 'Pump Count Down Variable

Public Flag(2) As Boolean 'Depth control variable pump

Public update\_complete As Boolean 'Indicates that update is complete

Public Handle As Long 'Holds the result of the TCPOpen function

Public file\_handle As Long 'Holds the result of the FileOpen function

Public TogglePump As Boolean 'Pump control Variable

Public http\_header As String \* 100 'Define a string that indicates the additional header information to include in the request

Public http\_post\_response As String \* 200 'Indicates where the results of the request will be written

Public http\_post\_tx 'Holds the result of HTTPPost function

Public InString As String \* 100 'Declare a string variable large enough to hold the input string

Public OutString As String \* 100 'Declare a string variable large enough to hold the output string

Public PTemp , batt\_volt, Counter, I 'Define PTemp, batt\_volt,Counter,I

Public Result 'Holds the result of the ModBus master query

Public CR1000State 'Holds the state of CR1000

Public ProductOil 'Define Product oil

Public CrudeOil 'Define Crude oil

Public ModIn(23) 'Define variable array which is used as the source of data to send, or the variable array that is used as the destination for data received

Public Rawdate 'Define Rawdate

Public Weather(5) As Float 'Define the weather instrument AWS600 measuring array

Public WQ(9) As Float 'Define quality meter EXO2 measuring array

Alias Port(5) = C5 'Define the interface C5

Alias Port(6) = C6 'Define the interface C6

Alias Port(7) = C7 'Define the interface C7

Alias Flag(1) = deep\_water 'Define deep water flag

Alias Flag(2) = shallow\_water 'Define shallow water flag

Alias WQ(1) = DO\_ 'Define Dissolved Oxygen

Alias WQ(2) = TU\_ 'Define turbidity

Alias WQ(3) = COND\_ 'Define the conductivity

Alias WQ(4) = PH\_ 'Define PH

Alias WQ(5) = CHL\_ 'Define the chlorophyll

Alias WQ(6) = BGA\_ 'Define Blue-Green algae

Alias WQ(7) = WT\_ 'Define water temperature

Alias WQ(8) = WiperPosition\_ 'Define WiperPosition

Alias WQ(9) = Power\_ 'Define Power\_

Alias Weather(1) = AirTemp 'Define air temperature

Alias Weather(2) = RH 'Define relative humidity

Alias Weather(3) = Pressure 'Define atmospheric pressure

Alias Weather(4) = WindDir 'Define the direction of the wind

Alias Weather(5) = WindSpd 'Define the speed of the wind

Units PTemp = Deg C

Units batt\_volt = Volts

Units DO\_ = mg/Litre

Units TU\_ = NTU

Units CHL\_= μg/L

Units COND\_= μs/cm

Units BGA\_= μg/L

Units WT\_ = Deg C

Units WiperPosition\_ = V

Units Power\_ = V

Units ProductOil = μg/L

Units CrudeOil = μg/L

Units AirTemp = Deg C

Units Pressure = hPa

Units WindSpd = meters/second

Units WindDir = Degrees

'Define Data Tables. Define storage area in memory.Table name is Half\_hour.

DataTable (System,True,-1) 'Time driven data storage

DataInterval (0,1,Min,0) 'Controls the interval

Minimum (1,batt\_volt,FP2,0,False) 'Stores batt\_volt minimum in low resolution format

FieldNames ("Supply\_Voltage")

Sample (1,PTemp,FP2) 'Stores PTemp sample in low resolution format

FieldNames ("Panel\_Temperature")

EndTable

DataTable (WaterQuality,True,-1)

DataInterval (0,1,Min,0)

Sample (1,ProductOil,FP2) 'Stores ProductOil sample in low resolution format

FieldNames("ProducOil")

Sample (1,CrudeOil,FP2) 'Stores CrudeOil sample in low resolution format

FieldNames("CrudeOil")

Sample (1,DO\_,FP2) 'Stores DO\_ sample in low resolution format

FieldNames("DO")

Sample (1,TU\_,FP2) 'Stores TU\_ sample in low resolution format

FieldNames("Turbidity")

Sample (1,COND\_,IEEE4) 'Stores COND\_ sample in low resolution format

FieldNames("Conductivity")

Sample (1,PH\_,FP2) 'Stores PH\_ sample in low resolution format

FieldNames("pH")

Sample (1,CHL\_,IEEE4) 'Stores CHL\_ sample in low resolution format

FieldNames("Chlorophyll")

Sample (1,BGA\_,IEEE4) 'Stores BGA\_ sample in low resolution format

FieldNames("BGA-PE")

Sample (1,WT\_,IEEE4) 'Stores WT\_ sample in low resolution format

FieldNames("Water\_Temperature")

Sample (1,WiperPosition\_,FP2) 'Stores WT\_ sample in low resolution format

FieldNames("Wiper\_Position")

Sample (1,Power\_,FP2) 'Stores WT\_ sample in low resolution format

FieldNames("Supply")

Sample (5,Weather(),IEEE4) 'Stores Weather() sample in high resolution format

Average (1,WindSpd ,FP2,False) 'Stores windspeed average in low resolution format

Maximum (1,WindSpd ,FP2,False,True) 'Stores windspeed maximum in low resolution format

Minimum (1,WindSpd ,FP2,False,True) 'Stores windspeed minimum in low resolution format

EndTable

PreserveVariables 'Retain in memory the values for variables declared by the Dim or Public statements

Function ConfigureCR1000()

SetStatus("RS232Power",-1)

SetStatus("BaudrateRS232",9600) 'Set the RS232 baud rate to 9600

SetStatus("pppInterface",1) '1=RS232

SetStatus("pppIPAddr","192.168.10.110")

SetStatus("pppDial","ATV1;AT+CGATT=0;AT+CGDCONT=1,""IP"",""internet"";ATDT\*99\*\*\*1#")

SetStatus("pppDialResponse","CONNECT")

'SetStatus("ServeAddress"??,"url.to.use")

SetStatus("HTTPEnabled",TRUE) 'Setup the web server on port 8888

SetStatus("HTTPPort",8888)

SetStatus("StationName",STATION\_NAME)

update\_complete=TRUE 'Finished

EndSub

'Custom Menu Declarations

DisplayMenu("\*\*CUSTOM MENU DEMO\*\*",-3) 'Create Menu; Upon power up, the custom menu is displayed. The system menu is hidden a blank line

SubMenu("") 'Dummy Sub menu to write a blank line a blank line

DisplayValue("",Escape)

EndSubMenu 'End of dummy submenu

SubMenu("Draw Water ") 'Create Submenu named Draw Water

MenuItem("Count to Draw Water",CountDown1) 'Create menu item CountDown

MenuPick(30,60,90,120) 'Create a pick list for CountDown

MenuItem("Manual PUMP 1",TogglePump) 'Manual PUMP 1 control Menu Item

MenuPick(On,Off)

EndSubMenu

SubMenu("Measure") 'Create Submenu named Measure

MenuItem("Count to Measure",CountDown2)

MenuPick(30,60,90,120)

MenuItem("Manual PUMP 2",TogglePump) 'Manual PUMP 2 control Menu Item

MenuPick(On,Off)

EndSubMenu

SubMenu("Drain away water ") 'Create Submenu named Drain away water

MenuItem("Count to Drain away Water",CountDown3)

MenuPick(30,60,90,120)

MenuItem("Manual PUMP 3",TogglePump) 'Manual PUMP 3 control Menu Item

MenuPick(On,Off)

EndSubMenu

EndMenu

'Main Program

BeginProg

Handle = TCPOpen("192.168.10.110",502,300,20)

ModbusMaster(Result,ComRS232,9600,1,1,ModIn(),1,23,3,200,0)

SerialOpen (ComRS232,9600,0,0,50000)

SerialFlush (ComRS232) 'Clear any characters in the serial input buffer

OutString = CHR(48) + CHR(67) + CHR(33) + CHR(48) + CHR(48) + CHR(49) + CHR(53) + CHR(48) + CHR(55) + CHR(60) + CHR(13) + CHR(62) + CHR(60) + CHR(10) + CHR(62) '0C!001507<CR><LF>

OutString = CHR(49) + CHR(67) + CHR(33) + CHR(49) + CHR(48) + CHR(49) + CHR(53) + CHR(48) + CHR(53) + CHR(60) + CHR(13) + CHR(62) + CHR(60) + CHR(10) + CHR(62) '1C!101505<CR><LF>

OutString = CHR(49) + CHR(68) + CHR(48) + CHR(33) + CHR(60) + CHR(13) + CHR(62) + CHR(60) + CHR(10) + CHR(62) '1D0! <CR><LF>

OutString = CHR(48) + CHR(68) + CHR(48) + CHR(33) + CHR(60) + CHR(13) + CHR(62) + CHR(60) + CHR(10) + CHR(62) '0D0! <CR><LF>

OutString = CHR(48) + CHR(68) + CHR(49) + CHR(33) + CHR(60) + CHR(13) + CHR(62) + CHR(60) + CHR(10) + CHR(62) '0D1! <CR><LF>

Scan (10,sec,0,0)

IPRoute ("api.cosm.com",0) 'Set the interface to be used when the datalogger sends an outgoing packet and multiple interfaces are active

PanelTemp(PTemp,250) 'Read the temperature of the datalogger wiring panel and output the value in degrees Celsius

Battery (batt\_volt) 'Read the battery voltage

VoltSe (ProductOil,1,mV5000,3,True,0,250,1.0,0) 'Make a single-ended voltage measurement ProductOil on SE3

VoltSe (CrudeOil,1,mV5000,4,True,0,250,1.0,0) 'Make a single-ended voltage measurement CruideOil on SE4

Counter = Counter + 1

ModIn(1) = batt\_volt

ModIn(2) = PTemp

ModIn(3) = DO\_

ModIn(4) = TU\_

ModIn(5) = COND\_

ModIn(6) = PH\_

ModIn(7) = CHL\_

ModIn(8) = BGA\_

ModIn(9) = WT\_

ModIn(10) = WiperPosition\_

ModIn(11) = Power\_

ModIn(12) = AirTemp

ModIn(13) = RH

ModIn(14) = Pressure

ModIn(15) = WindDir

ModIn(16) = WindSpd

ModIn(17) = ProductOil

ModIn(18) = CrudeOil

ModIn(19) = CountDown1

ModIn(20) = CountDown2

ModIn(21) = CountDown3

ModIn(22) = Result

ModIn(23) = Counter

SerialFlush(ComRS232) 'Clear any characters in the serial port RS232 buffer

'Retrieve the results from two SDI-12 sensors every three minutes

If TimeIntoInterval(0,3,Min) Then

RunSDI12 = True

If RunSDI12 = True Then

SDI12Recorder (WQ(),5,EXO2\_SDI12\_address,"MC!",1.0,0) 'Retrieve the results from an SDI-12 sensor EXO2

SDI12Recorder (Weather(),7,AWS\_SDI12\_address,"MC!",1.0,0) 'Retrieve the results from an SDI-12 sensor WS600

EndIf

EndIf

'Menu Item "Draw Water" Menu Support Code

CountDown1 = CountDown1 - 1 'Count down by 1

If CountDown1 <= 0 'Stop count down from pasing 0

CountDown1 = 0

EndIf

If CountDown1 > 0 Then

StartFlag = True 'Indicate countdown started

TogglePump = True 'Indicate pump started

If Flag(1) = True Then 'Pump and reversing control

PortSet(3,1) 'Flag(1) means of the deep water pump

Else

PortSet(3,0)

EndIf

If Flag(2) = True Then 'Flag(2) means of the shallow water pump

PortSet(4,1)

Else

PortSet(4,0)

EndIf

EndIf

If StartFlag = True AND CountDown1 = 0 Then 'Interprocess count down and manual Pump

TogglePump = False

StartFlag = False

EndIf

PortSet(1,TogglePump) 'Set control port according to result of processing

'Menu Item "Measure" Menu Support Code

CountDown2 = CountDown2 - 1

If CountDown2 <= 0

CountDown2 = 0

EndIf

If CountDown2 > 0 Then

StartFlag = True

TogglePump = False

EndIf

If StartFlag = True AND CountDown2 = 0 Then

TogglePump = False

StartFlag = False

EndIf

If StartFlag = True AND CountDown2 <> 0 Then

TogglePump = False

EndIf

PortSet(1,TogglePump)

'Menu Item "Drain away water" Menu Support Code

CountDown3 = CountDown3 - 1

If CountDown3 <= 0

CountDown3 = 0

EndIf

If CountDown3 > 0 Then

StartFlag = True

TogglePump = True

If Flag(1) = True Then 'Deep water pump start

PortSet(3,1)

Else

PortSet(3,0)

EndIf

If Flag(2) = True Then 'Shawllow water pump start

PortSet(4,1)

Else

PortSet(4,0)

EndIf

EndIf

If StartFlag = True AND CountDown3 = 0 Then

TogglePump = False

StartFlag = False

EndIf

PortSet(1,TogglePump)

'Post current ModIn() to cosm.com (free service)

'See https://cosm.com/docs/ on API documentation

http\_header = "X-ApiKey: " 'Enter unique ApiKey here

'Write CR1000 file to CPU drive

'Adds two users

file\_handle = FileOpen("CPU:CR1000.CR1","w+",0)

'What to write.Crucial formatting. MUST be in the correct format (json) to accept.

FileWrite(file\_handle,STATION\_NAME & CHR(13),0)

FileWrite(file\_handle,"anonymous::3" & CHR(13),0)

FileWrite(file\_handle,"user:nMg/CR1000=:1)" & CHR(13),0)

FileClose(file\_handle)

http\_post\_tx = HTTPPost ("http://api.cosm.com/v2/feeds/"& 12345 &"/datastreams/"& ModIn() &"/datapoints", "CPU:CR1000.json", http\_post\_response, http\_header)

'Finished

update\_complete = TRUE

'Call Output Tables to store values

CallTable System

CallTable WaterQuality

CR1000State = ConfigureCR1000()

NextScan

EndProg