Hydroelectric Power Plant Project

Need to add modes and sequences!!

Project Narrative:

The idea here is to program controls for a very CRUDE hydroelectric power plant. We start with a base program that simulates a changing environment and manages our analog input signals accordingly. Our job is to write a program that incorporates these signals and controls various analog and digital devices in order to run the system with a safe and stable process. We also want to create an HMI that will allow the Operator to interact easily and effectively with our system.

Theory of Operation:

Our plant is built on a river. The river flows at various (changing) speeds speeding up and slowing down regularly.

The river flows through a baffle (think of a gate that works like a proportional control valve) that we control via a servo motor. We will send a 4-20mA signal out to this device to open and close it. While our baffle cannot completely block the river (only reduce it), it does give us some control over our generator's rotor speed.

After passing through the baffle, the river flows through our generator and turns the rotor. The faster the river flows through the generator, the faster the rotor turns (higher RPM) and the more current our generator produces.

As the rotor turns faster, the rotor's bearings heat up more and more. An oil pump controlled by a VFD pumps oil to the bearings to cool them off.

We'll monitor oil flow to make sure that if the VFD is running, the oil is flowing, but not flowing too much either. We don't want to burn up a pump after all.

Once the current we're able to produce gets high enough, we close an interlock to the power station which allows our generator to produce power and send it to the station.

When we aren't running, we want to leave the baffle open and engage the brake on the rotor so it doesn't keep spinning.

(continued)

If anything goes wrong (an alarm occurs), we control an alarm horn to notify the Operator (whose probably off watching porn on his cellphone somewhere) He's a real shady dude, but he was willing to work for cheap, and we have a tight operating budget!

Naturally, our system has an E-Stop for emergencies. Although impractical, our E-Stop will be a button on our HMI.

IO:

Here are the devices, signals, scaling factors and polarities we're using for this project to keep everybody on the same page.

(Key: AI = Analog Input, AQ = Analog Output, DI = Digital Input, DQ = Digital Output)

Baffle:
Signals: 1 DQ with tag name "Baffle_En" and 1 AQ with tag name "Baffle_Out"
Output scaling value range: 0-27648
HMI Display values: 0-100% open
Polarity: Reverse (higher output signal value closes the baffle more)
Note: So sending a 0 value out to the baffle makes it 100% open while sending a 27648 closes it completely.

Rotor Speed:
Signals: 1 AI with tag name "RPM_In"
Input scaling value range: 0-27648
HMI Display values: 0-200 RPM
Polarity: Straight (higher input signal value means higher RPM)
Note: RPM means "revolutions per minute" if you didn't already know. I really hope you did though!

Oil Pump VFD:
Signals: 1 DQ with tag name "OilPumpVFDen_Out" and 1 AQwith tag name "OilPumpVFDsp_Out"
Output scaling value range: 0-27648
HMI Display values: 0-100%
Polarity: Straight (higher input signal value means higher oil pump output)
Note: The digital output enables the drive, while the analog output controls the speed. Obviously the pump doesn't do anything if it isn't enabled regardless of what speed value you send it.

Oil Flow:
Signals: 1 AI with tag name "OilFlow_In"
Input scaling value range: 0-27648
HMI Display values: 0-30 GPM
Polarity: Straight (higher input signal value means higher GPM)
Note: GPM means "gallons per minute" because f the metric system!*

(continued)

Oil Temperature:
Signals: 1 AI with tag name "OilTemp_In"
Input scaling value range: 0-27648
HMI Display values: 0-500 degrees
Polarity: Straight (higher input signal value means higher temperature)
Note: I didn't specify Celsius or Fahrenheit, did I? Use your imagination.

Power Station Interlock:
Signals: 1 DQ with tag name "StationInterlock_Out"
Polarity: Straight (energize the output to engage the interlock)

Alarm Horn:
Signals: 1 DQ with tag name "AlarmHorn_Out"
Polarity: Straight (energize the output to sound the horn)

Generator Brake:
Signals: 1 DQ with tag name "Brake_Out"
Polarity: Straight (energize the output to engage the brake)

E-Stop:
Signals: 1 DI with tag name "EStop_In"
Polarity: Reverse (closed E_Stop means everything is normal / open means there's a problem)

AC Current:
Input scaling value range: 0-27648
HMI Display values: 0-700 kW
Polarity: Straight (higher input signal value means higher kW)
Note: kW means "killowatts."

Alarm Conditions:

Note: All alarms should operate on a 5-second delay WITH THE EXCEPTIONS of the E-Stop (which should be immediate) and the Oil Low Flow and Brake Failure.

Note: Sound the horn for any alarm.

Note: Set all of your alarm delay times and setpoints up as tags so we can make them configurable on our HMI.

Overcurrent:
Setpoint: 550kW
Process Actions: Disengage interlock and begin normal shutdown sequence.
Alarm Inhibits: nothing

(continued)

High Oil Temp:
Setpoint: 400 degrees
Process Actions: Disengage interlock and begin normal shutdown sequence.
Alarm Inhibits: nothing

Rotor Overspeed:
Setpoint: 180 RPM
Process Actions: Disengage interlock and begin normal shutdown sequence.
Alarm Inhibits: nothing

Oil Low Flow:
Setpoint: < 5 GPM for greater than 10 seconds while VFD is engaged and brake is disengaged
Process Actions: Disengage interlock, deenergize the VFD, open the baffle and engage the brake.
Alarm Inhibits: VFD

Oil High Flow:
Setpoint: 30 GPM
Process Actions: Disengage interlock and begin normal shutdown sequence. Override VFD output to 50% of previous output (when fault occurred) until shutdown complete.
Alarm Inhibits: nothing

E-Stop:
Condition: E-Stop open
Process Actions: Disengage interlock, deenergize the VFD, open the baffle and engage the brake. All HOAs shift from AUTO mode to either OFF or HAND depending on desired state.
Alarm Inhibits: interlock, VFD

Brake Failure:
Condition: RPM > 0 with brake engaged for more than 30 seconds
Process Actions: Disengage interlock, deenergize the VFD, open the baffle and engage the brake.
Alarm Inhibits: nothing

Sequence of Operation / Modes:

Note: With the exception of Fault mode and POSSIBLY Idle mode, the remaining modes should be set up as steps of your sequence in your GRAPH program block. They should occur sequentially as listed here below.

Modes:
Warmup – When Start pushbutton is pressed, the sequence begins with this step. Allow the rotor to turn at a speed somethere between 20 and 50 RPMs and run the oil pump at some low output level (2-10 GPM maybe?) until the oil temp reaches 150 degrees. Once we’ve maintained at least 150 degrees for 10 seconds, we want to transition into the next mode.

(continued)

Stabilize **–** Generator should speed up until desired kW output is attained. Once we're stable within 5kW (+/-) of the setpoint for at least 10 seconds, we should transition into the next mode.

Generation – Keep everything running and close the interlock.

Cooldown – Once the Stop button has been pushed, open the interlock, slow the rotor to between 20 and 50 RPMs and maintain a high oil flow until the oil temp is below 150 degrees. Once we're below for ten seconds, we transition into Idle mode.
Idle **–** Interlock and baffle are deenergized and brake is engaged.

Fault – When we hit an alarm, the sequence should be aborted according to the alarm details (above in the Alarms section), and we should go into Fault mode (which is not part of our sequence).

HMI Detail:

Note: These are the MINIMUM requirements for our HMI. Inasmuch as you'll probably be showing this to potential employers someday trying to convince them to pay you vast sums of ca$h, it may be in your best interest to take the liberty and EXCEED these requirements. So please - feel free to consider your time spent on this project as an investment in your employability and earning potential.

Screens:
System Status (default screen)
Process Overview
Alarm Management
IO / HOA

System Status: Show system mode, alarm indicators, important process values, high-level system controls (like maybe a start and stop button).

Note: System should NOT start until all devices are in AUTO control mode and all alarms eliminated.

Process Overview: Show a graphical representation of our system with relevant process values appearing at their points of collection. You can decide where exactly those may be when not clear.

Alarm Management: Show alarm history, alarm reset and silence buttons, and indicators which show the presence of actual alarm bits (not just notifications).

IO / HOA: Give the Operator controls over all devices to include manual setpoints for analog devices. Also, display the status / values of inputs signals. You may also want to show alarm indicators (not notification bits) for any devices that may be inhibited.

Note: HAND mode should engage the device permanently until mode is changed to either OFF or AUTO.

Unspecified Information / Questions / Concerns:

Did I not provide some important detail you need to know to program the system? That's okay – use your best judgment. As a programmer, we have to do exactly that and much more often than you might think. This is a guideline, but in the end, it's YOUR PROJECT, so feel free to make your own determination on anything I didn't explicitly state.

For that matter, feel free to take liberties on any of the stuff I did explicitly state! Come to think of it, if you decide to program a sno-cone maker instead of this hydroelectric plant, that too would be just fine. I mean – who doesn't love a sno-cone?!