

Lab

HMI configuration for PLC control system: *Question 91, completed objectives due by the end of day 2, section 2*

Exam

Day 3 of next section – only a simple calculator may be used! **Complete mastery of these objectives due by the next exam date**

Specific objectives for the “mastery” exam:

- Electricity Review: Calculate voltages and currents in an ideal AC transformer circuit
 - Sketch proper wire connections for sourcing or sinking PLC I/O points and specified process switch trip conditions
 - Determine status of discrete PLC outputs given process switch stimuli and a RLL program
 - Calculate voltages and currents within balanced three-phase AC electrical circuits
 - Solve for a specified variable in an algebraic formula
 - Determine the possibility of suggested faults in a simple PLC circuit given a wiring diagram, RLL program listing, meter measurements, and/or reported symptoms
 - INST240 Review: Determine suitability of different level-measuring technologies for a given process fluid type
 - INST251 Review: Identify the graphed response of a controller as being either P, I, or D
 - INST263 Review: Determine effect of a fault in a selector or override control system
-

Recommended daily schedule

Day 1

Theory session topic: Introduction to Human-Machine Interface (HMI) panels

Questions 1 through 20; answer questions 1-9 in preparation for discussion (remainder for practice)

Day 2

Theory session topic: Applications and programming practice

Questions 21 through 40; answer questions 21-26 in preparation for discussion (remainder for practice)

Day 3

Theory session topic: Data comparison and arithmetic instructions

Questions 41 through 60; answer questions 41-49 in preparation for discussion (remainder for practice)

Day 4

Theory session topic: Data transfer instructions

Questions 61 through 80; answer questions 61-67 in preparation for discussion (remainder for practice)

Feedback questions (*81 through 90*) are optional and may be submitted for review at the end of the day

How To . . .

Access the worksheets and textbook: go to the *Socratic Instrumentation* website located at <http://www.ibiblio.org/kuphaldt/socratic/sinst> to find worksheets for every 2nd-year course section organized by quarter, as well as both the latest “stable” and “development” versions of the *Lessons In Industrial Instrumentation* textbook. Download and save these documents to your computer.

Maximize your learning: complete all homework *before* class starts, ready to be assessed as described in the “Inverted Session Formats” pages. Use every minute of class and lab time productively. Follow all the tips outlined in “Question 0” as well as your instructor’s advice. Do not take constructive criticism personally. Make every reasonable effort to solve problems on your own before seeking help.

Identify upcoming assignments and deadlines: read the first page of each course worksheet.

Relate course days to calendar dates: reference the calendar spreadsheet file (`calendar.xlsx`), found on the BTC campus Y: network drive. A printed copy is posted in the Instrumentation classroom.

Locate industry documents assigned for reading: use the Instrumentation Reference provided by your instructor (on CD-ROM and on the BTC campus Y: network drive). There you will find a file named `00_index.OPEN.THIS.FILE.html` readable with any internet browser. Click on the “Quick-Start Links” to access assigned reading documents, organized per course, in the order they are assigned.

Study for the exams: Mastery exams assess specific skills critically important to your success, listed near the top of the front page of each course worksheet for your review. Familiarize yourself with this list and pay close attention when those topics appear in homework and practice problems. Proportional exams feature problems you haven’t seen before that are solvable using general principles learned throughout the current and previous courses, for which the only adequate preparation is independent problem-solving practice every day. Answer the “feedback questions” (practice exams) in each course section to hone your problem-solving skills, as these are similar in scope and complexity to proportional exams. Answer these feedback independently (i.e. no help from classmates) in order to most accurately assess your readiness.

Calculate course grades: download the “Course Grading Spreadsheet” (`grades_template.xlsx`) from the Socratic Instrumentation website, or from the BTC campus Y: network drive. Enter your quiz scores, test scores, lab scores, and attendance data into this Excel spreadsheet and it will calculate your course grade. You may compare your calculated grades against your instructors’ records at any time.

Identify courses to register for: read the “Sequence” page found in each worksheet.

Receive extra instructor help: ask during lab time, or during class time, or by appointment.

Identify job openings: regularly monitor job-search websites. Set up informational interviews at workplaces you are interested in. Participate in jobshadows and internships. Apply to jobs long before graduation, as some employers take *months* to respond! Check your BTC email account daily, because your instructor broadcast-emails job postings to all students as employers submit them to BTC.

Impress employers: sign the FERPA release form granting your instructors permission to share academic records, then make sure your performance is worth sharing. Document your project and problem-solving experiences for reference during interviews. Honor all your commitments.

Begin your career: participate in jobshadows and internships while in school to gain experience and references. Take the first Instrumentation job that pays the bills, and give that employer at least two years of good work to pay them back for the investment they have made in you. Employers look at delayed employment, as well as short employment spans, very negatively. Failure to pass a drug test is an immediate disqualifier, as is falsifying any information. Criminal records may also be a problem.

file howto

General Values, Expectations, and Standards

Success in this career requires professional integrity, resourcefulness, persistence, close attention to detail, and intellectual curiosity. If you are ever in doubt as to the values you should embody, just ask yourself what kind of a person you would prefer to hire for your own enterprise. Those same values will be upheld within this program.

Learning is the top priority in this program. Every circumstance, every incident, every day will be treated as a learning opportunity, every mistake as a “teachable moment”. Every form of positive growth, not just academic ability, will be regarded as real learning.

Responsibility means *ensuring* the desired outcome, not just *trying* to achieve the outcome. If your efforts do not yield the expected results, only you can make it right.

Integrity means being honest and forthright in all your words and actions, doing your very best every time and never taking credit for the achievement of another.

Safety means doing every job correctly and ensuring others are not endangered. Lab safety standards include wearing closed-toed shoes and safety glasses in the lab room during lab hours, wearing ear protection around loud sounds, using ladders to reach high places, using proper lock-out/tag-out procedures, no energized electrical work above 30 volts without an instructor present in the lab room, and no power tool use without an instructor present in the lab room.

Diligence means exercising self-discipline and persistence in your studies, realizing that hard work is a necessary condition for success. This means, among other things, investing the necessary time and effort in studying, reading instructions, paying attention to details, utilizing the skills and tools you already possess, and avoiding shortcuts.

Mastery means the job is not done until it is done *correctly*: all objectives achieved, all problems solved, all documentation complete, and no errors remaining.

Self-management means allocating your resources (time, equipment, labor) wisely, and not just focusing on the nearest deadline.

Communication means clearly conveying your thoughts and paying attention to what others convey. Remember that no one can read your mind, and so it is incumbent upon you to communicate any and all important information.

Teamwork means working constructively with your classmates so as to maximize their learning as well as your own.

Initiative means recognizing needs and taking action to meet those needs without encouragement or direction from others.

Representation means your actions are a reflection of this program and not just of yourself. Doors of opportunity for all BTC graduates may be opened or closed by your own conduct. Unprofessional behavior during tours, jobshadows, internships, and/or jobs reflects poorly on the program and will negatively bias employers.

Trustworthiness is the result of consistently exercising these values: people will recognize you as someone they can rely on to get the job done, and therefore someone they would want to hire.

Respect means acknowledging the intrinsic value, capabilities, and responsibilities of those around you. Respect may be gained by consistent demonstration of valued behaviors, and it may be lost through betrayal of trust.

General Values, Expectations, and Standards (continued)

Punctuality and Attendance: late arrivals are penalized at a rate of 1% grade deduction per incident. Absence is penalized at a rate of 1% per hour (rounded to the nearest hour) except when employment-related, school-related, weather-related, or required by law (e.g. court summons). Absences may be made up by directing the instructor to apply “sick hours” (12 hours of sick time available per quarter). Classmates may donate their unused sick hours. Sick hours may not be applied to unannounced absences, so be sure to alert your instructor and teammates as soon as you know you will be absent or late. Absence on an exam day will result in a zero score for that exam, unless due to a documented emergency.

Mastery: any assignment or objective labeled as “mastery” must be completed with 100% competence (with multiple opportunities to re-try). Failure to complete by the deadline date caps your grade at a C–. Failure to complete by the end of the *next* school day results in a failing (F) grade for that course.

Time Management: Use all available time wisely and productively. Work on other useful tasks (e.g. homework, feedback questions, job searching) while waiting for other activities or assessments to begin. Trips to the cafeteria for food or coffee, smoke breaks, etc. must not interfere with team participation.

Orderliness: Keep your work area clean and orderly, discarding trash, returning tools at the end of every lab session, and participating in all scheduled lab clean-up sessions. Project wiring, especially in shared areas such as junction boxes, must not be left in disarray at the end of a lab shift. Label any failed equipment with a detailed description of its symptoms.

Independent Study: the “inverted” instructional model used in this program requires independent reading and problem-solving, where every student must demonstrate their learning at the start of the class session. Question 0 of every worksheet lists practical study tips. The “Inverted Session Formats” pages found in every worksheet outline the format and grading standards for inverted class sessions.

Independent Problem-Solving: make an honest effort to solve every problem before seeking help. When working in the lab, help will not be given to you unless and until you run your own diagnostic tests.

Teamwork: inform your teammates if you need to leave the work area for any reason. Any student regularly compromising team performance through absence, tardiness, disrespect, or other disruptive behavior(s) will be removed from the team and required to complete all labwork individually. The same is true for students found inappropriately relying on teammates.

Communication: check your email account daily for important messages from your instructor. Ask the instructor to clarify any assignment or exam question you find confusing, and express your work clearly and compellingly.

Academic Progress: your instructor will record your academic achievement, as well as comments on any negative behavior, and will share all these records with employers provided you have signed the FERPA release form. You are welcome to see these records at any time, and are encouraged to track your own academic progress using the grade spreadsheet template.

Office Hours: your instructor’s office hours are by appointment, except in cases of emergency. Email is the preferred method for setting up an appointment with your instructor to discuss something in private.

Grounds for Failure: a failing (F) grade will be earned in any course if any mastery objectives are past deadline by more than one school day, or if any of the following behaviors are demonstrated: false testimony (lying) to your instructor, cheating on any assignment or assessment, plagiarism (presenting another’s work as your own), willful violation of a safety policy, theft, harassment, intoxication, or destruction of property. Such behaviors are grounds for immediate termination in this career, and as such will not be tolerated here.

file expectations

Inverted session formats

The basic concept of an “inverted” learning environment is that the traditional allocations of student time are reversed: instead of students attending an instructor-led session to receive new information and then practicing the application of that information outside of the classroom in the form of homework, students in an inverted class encounter new information outside of the classroom via homework and apply that information in the classroom session under the instructor’s tutelage.

A natural question for instructors, then, is what their precise role is in an inverted classroom and how to organize that time well. Here I will list alternate formats suitable for an inverted classroom session, each of them tested and proven to work.

Small sessions

Students meet with instructors in small groups for short time periods. Groups of 4 students meeting for 30 minutes works very well, but groups as large as 8 students apiece may be used if time is limited. Each of these sessions begins with a 5 to 10 minute graded inspection of homework with individual questioning, to keep students accountable for doing the homework. The remainder of the session is a dialogue focusing on the topics of the day, the instructor challenging each student on the subject matter in Socratic fashion, and also answering students’ questions. A second grade measures each student’s comprehension of the subject matter by the end of the session.

This format also works via teleconferencing, for students unable to attend a face-to-face session on campus.

Large sessions

Students meet with instructors in a standard classroom (normal class size and period length). Each of these sessions begins with a 10 minute graded quiz (closed-book) on the homework topic(s), to keep students accountable for doing the homework. Students may leave the session as soon as they “check off” with the instructor in a Socratic dialogue as described above (instructor challenging each student to assess their comprehension, answering questions, and grading the responses). Students sign up for check-off on the whiteboard when they are ready, typically in groups of no more than 4. Alternatively, the bulk of the class session may be spent answering student questions in small groups, followed by another graded quiz at the end.

Correspondence

This format works for students unable to attend a “face-to-face” session, and who must correspond with the instructor via email or other asynchronous medium. Each student submits a thorough presentation of their completed homework, which the instructor grades for completeness and accuracy. The instructor then replies back to the student with challenge questions, and also answers questions the student may have. As with the previous formats, the student receives another grade assessing their comprehension of the subject matter by the close of the correspondence dialogue.

In all formats, students are held accountable for completion of their homework, “completion” being defined as successfully interpreting the given information from source material (e.g. accurate outlines of reading or video assignments) and constructive effort to solve given problems. It must be understood in an inverted learning environment that students *will* have legitimate questions following a homework assignment, and that it is therefore unreasonable to expect mastery of the assigned subject matter. What is reasonable to expect from each and every student is a basic outline of the source material (reading or video assignments) complete with major terms defined and major concepts identified, plus a good-faith effort to solve every problem. Question 0 (contained in every worksheet) lists multiple strategies for effective study and problem-solving.

Sample rubric for pre-assessments

- **No credit** = Any homework question unattempted (i.e. no effort shown on one or more questions); incomprehensible writing; failure to follow clear instruction(s)
- **Half credit** = Misconception(s) on any major topic explained in the assigned reading; answers shown with no supporting work; verbatim copying of text rather than written in your own words; outline missing important topic(s); unable to explain the outline or solution methods represented in written work
- **Full credit** = Every homework question answered, with any points of confusion clearly articulated; all important concepts from reading assignments accurately expressed in the outline and clearly articulated when called upon by the instructor to explain

The minimum expectation at the start of every student-instructor session is that all students have made a good-faith effort to complete 100% of their assigned homework. This does not necessarily mean all answers will be correct, or that all concepts are fully understood, because one of the purposes of the meeting between students and instructor is to correct remaining misconceptions and answer students' questions. However, experience has shown that without accountability for the homework, a substantial number of students will not put forth their best effort and that this compromises the whole learning process. Full credit is reserved for good-faith effort, where each student thoughtfully applies the study and problem-solving recommendations given to them (see Question 0).

Sample rubric for post-assessments

- **No credit** = Failure to comprehend one or more key concepts; failure to apply logical reasoning to the solution of problem(s); no contribution to the dialogue
- **Half credit** = Some misconceptions persist by the close of the session; problem-solving is inconsistent; limited contribution to the dialogue
- **Full credit** = Socratic queries answered thoughtfully; effective reasoning applied to problems; ideas communicated clearly and accurately; responds intelligently to questions and statements made by others in the session; adds new ideas and perspectives

The minimum expectation is that each and every student engages with the instructor and with fellow students during the Socratic session: posing intelligent questions of their own, explaining their reasoning when challenged, and otherwise positively contributing to the discussion. Passive observation and listening is not an option here – every student must be an active participant, contributing something original to every dialogue. If a student is confused about any concept or solution, it is their responsibility to ask questions and seek resolution.

If a student happens to be absent for a scheduled class session and is therefore unable to be assessed on that day's study, they may schedule a time with the instructor to demonstrate their comprehension at some later date (before the end of the quarter when grades must be submitted). These same standards of performance apply equally make-up assessments: either inspection of homework or a closed-book quiz for the pre-assessment, and either a Socratic dialogue with the instructor or another closed-book quiz for the post-assessment.

file format

Course Syllabus

INSTRUCTOR CONTACT INFORMATION:

Tony Kuphaldt
(360)-752-8477 [office phone]
(360)-752-7277 [fax]
tony.kuphaldt@btc.edu

DEPT/COURSE #: INST 232

CREDITS: 3 **Lecture Hours:** 11 **Lab Hours:** 44 **Work-based Hours:** 0

COURSE TITLE: PLC Systems

COURSE DESCRIPTION: In this course you will learn how to program data-handling functions in programmable logic controllers (PLCs) including comparison, arithmetic, and data transfer instructions. You will also learn to connect and program human-machine interface (HMI) panels to PLCs. **Pre/Corequisite course:** INST 231 (PLC Programming) **Prerequisite course:** MATH&141 (Precalculus 1) with a minimum grade of “C”

COURSE OUTCOMES: Program, analyze, and efficiently diagnose control systems incorporating programmable logic controllers (PLCs) and human-machine interface panels (HMIs).

COURSE OUTCOME ASSESSMENT: PLC/HMI programming, analysis, and diagnosis outcomes are ensured by measuring student performance against mastery standards, as documented in the Student Performance Objectives. Failure to meet all mastery standards by the next scheduled exam day will result in a failing grade for the course.

STUDENT PERFORMANCE OBJECTIVES:

- Without references or notes, within a limited time (3 hours total for each exam session), independently perform the following tasks. Multiple re-tries are allowed on mastery (100% accuracy) objectives, each with a different set of problems:
 - Calculate voltages and currents in an ideal AC transformer circuit, with 100% accuracy (mastery)
 - Sketch proper wire connections for sourcing or sinking PLC I/O points and specified process switch trip conditions, with 100% accuracy (mastery)
 - Determine status of discrete PLC outputs given process switch stimuli and a RLL program, with 100% accuracy (mastery)
 - Calculate voltages and currents within balanced three-phase electrical circuits, with 100% accuracy (mastery)
 - Solve for specified variables in algebraic formulae, with 100% accuracy (mastery)
 - Determine the possibility of suggested faults in a simple PLC circuit given a wiring diagram, RLL program listing, meter measurements, and/or reported symptoms, with 100% accuracy (mastery)
 - Program an HMI panel to fulfill a specified control system function
- In a team environment and with full access to references, notes, and instructor assistance, perform the following tasks:
 - Demonstrate proper use of safety equipment and application of safe procedures while using power tools, and working on live systems
 - Communicate effectively with teammates to plan work, arrange for absences, and share responsibilities in completing all labwork
 - Augment a PLC-controlled motor start/stop system with an HMI panel providing operator access to operating parameters
 - Generate an accurate wiring diagram compliant with industry standards documenting your team's motor control system
- Independently perform the following tasks with 100% accuracy (mastery). Multiple re-tries are allowed with different specifications/conditions each time:
 - Research equipment manuals to sketch a complete circuit connecting a PLC to either a discrete sensing device or a discrete control element (e.g. VFD), with all components randomly selected by the instructor
 - Diagnose a random fault placed in another team's PLC motor control system by the instructor within a limited time using no test equipment except a multimeter and ladder logic editing software, logically justifying your steps in the instructor's direct presence

COURSE OUTLINE: A course calendar in electronic format (Excel spreadsheet) resides on the Y: network drive, and also in printed paper format in classroom DMC130, for convenient student access. This calendar is updated to reflect schedule changes resulting from employer recruiting visits, interviews, and other impromptu events. Course worksheets provide comprehensive lists of all course assignments and activities, with the first page outlining the schedule and sequencing of topics and assignment due dates. These worksheets are available in PDF format at <http://www.ibiblio.org/kuphaldt/socratic/sinst>

- INST232 Section 1 (HMI programming, PLC comparison and arithmetic programming, PLC data transfer programming): 4 days theory and labwork
- INST232 Section 2 (PLC analog programming and project completion): 1 day theory and labwork + 1 day for mastery/proportional exams + 1 day for lab clean-up

METHODS OF INSTRUCTION: Course structure and methods are intentionally designed to develop critical-thinking and life-long learning abilities, continually placing the student in an active rather than a passive role.

- **Independent study:** daily worksheet questions specify *reading assignments*, *problems* to solve, and *experiments* to perform in preparation (before) classroom theory sessions. Open-note quizzes and work inspections ensure accountability for this essential preparatory work. The purpose of this is to convey information and basic concepts, so valuable class time isn't wasted transmitting bare facts, and also to foster the independent research ability necessary for self-directed learning in your career.
- **Classroom sessions:** a combination of *Socratic discussion*, short *lectures*, *small-group* problem-solving, and hands-on *demonstrations/experiments* review and illuminate concepts covered in the preparatory questions. The purpose of this is to develop problem-solving skills, strengthen conceptual understanding, and practice both quantitative and qualitative analysis techniques.
- **Hands-on PLC programming challenges:** daily worksheet questions specify realistic scenarios requiring students to develop real PLC and HMI programs on their PLC trainers to implement the desired control function(s).
- **Lab activities:** an emphasis on constructing and documenting *working projects* (real instrumentation and control systems) to illuminate theoretical knowledge with practical contexts. Special projects off-campus or in different areas of campus (e.g. BTC's Fish Hatchery) are encouraged. Hands-on *troubleshooting exercises* build diagnostic skills.
- **Feedback questions:** sets of *practice problems* at the end of each course section challenge your knowledge and problem-solving ability in current as well as first year (Electronics) subjects. These are optional assignments, counting neither for nor against your grade. Their purpose is to provide you and your instructor with direct feedback on what you have learned.

STUDENT ASSIGNMENTS/REQUIREMENTS: All assignments for this course are thoroughly documented in the following course worksheets located at:

<http://www.ibiblio.org/kuphaldt/socratic/sinst/index.html>

- INST232_sec1.pdf
- INST232_sec2.pdf

EVALUATION AND GRADING STANDARDS: (out of 100% for the course grade)

- Completion of all mastery objectives = 50%
- Mastery exam score = 10%
- Proportional exam score = 30%
- Lab questions = 10%
- Quiz penalty = -1% per failed quiz
- Tardiness penalty = -1% per incident (1 “free” tardy per course)
- Attendance penalty = -1% per hour (12 hours “sick time” per quarter)
- Extra credit = +5% per project (assigned by instructor based on individual learning needs)

All grades are criterion-referenced (i.e. no grading on a “curve”)

100% ≥ A ≥ 95%	95% > A- ≥ 90%	
90% > B+ ≥ 86%	86% > B ≥ 83%	83% > B- ≥ 80%
80% > C+ ≥ 76%	76% > C ≥ 73%	73% > C- ≥ 70% (minimum passing course grade)
70% > D+ ≥ 66%	66% > D ≥ 63%	63% > D- ≥ 60% 60% > F

Absence on a scheduled exam day will result in a 0% score for the proportional exam unless you provide documented evidence of an unavoidable emergency.

If you fail a mastery exam, you must re-take a different version of that mastery exam on a different day. Multiple re-tries are allowed, on a different version of the exam each re-try. There is no penalty levied on your course grade for re-taking mastery exams, but failure to successfully pass a mastery exam by the due date will result in a failing grade (F) for the course.

If any other “mastery” objectives are not completed by their specified deadlines, your overall grade for the course will be capped at 70% (C- grade), and you will have one more school day to complete the unfinished objectives. Failure to complete those mastery objectives by the end of that extra day (except in the case of documented, unavoidable emergencies) will result in a failing grade (F) for the course.

“Lab questions” are assessed in a written exam format, typically on the last scheduled day of the lab project. Grading is as follows: full credit for thorough, correct answers; half credit for partially correct answers; and zero credit for major conceptual errors.

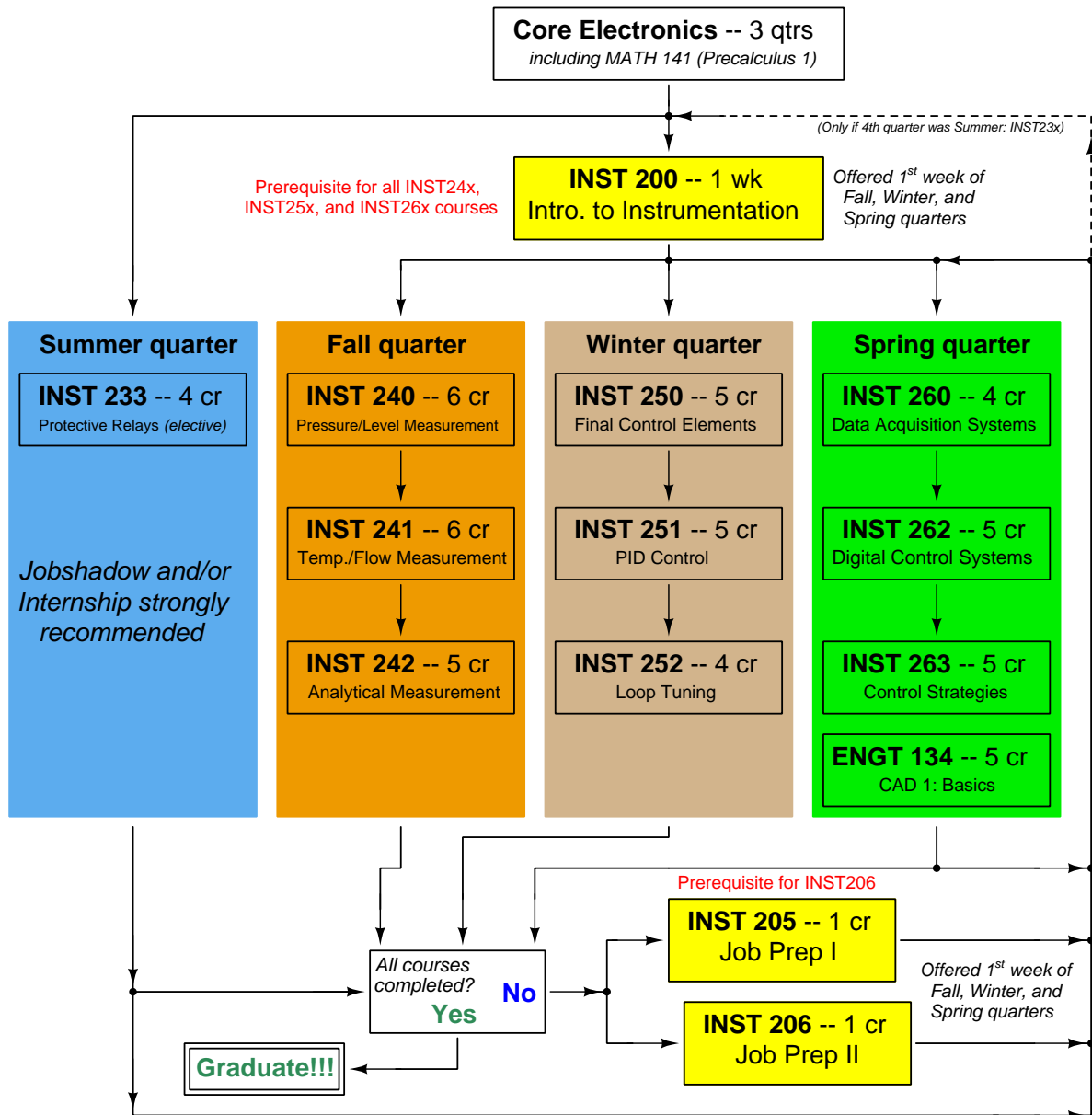
Individual preparation for Socratic dialogue sessions is measured by a “prep quiz” and/or personal inspection of your work by the instructor. A second (“summary”) quiz score for every Socratic session marks your participatory dialogue and ability to give reasoned answers to challenge questions on that session’s topic(s). In the event of absence, these scores may be credited by having your preparatory work and demonstration of understanding reviewed at any time before the end of the quarter in a one-on-one dialogue with the instructor.

Extra credit opportunities exist for each course, and may be assigned to students upon request. The student and the instructor will first review the student’s performance on feedback questions, homework, exams, and any other relevant indicators in order to identify areas of conceptual or practical weakness. Then, both will work together to select an appropriate extra credit activity focusing on those identified weaknesses, for the purpose of strengthening the student’s competence. A due date will be assigned (typically two weeks following the request), which must be honored in order for any credit to be earned from the activity. Extra credit may be denied at the instructor’s discretion if the student has not invested the necessary preparatory effort to perform well (e.g. lack of preparation for daily class sessions, poor attendance, no feedback questions submitted, etc.).

REQUIRED STUDENT SUPPLIES AND MATERIALS:

- Course worksheets available for download in PDF format
- *Lessons in Industrial Instrumentation* textbook, available for download in PDF format
→ Access worksheets and book at: <http://www.ibiblio.org/kuphaldt/socratic/sinst>
- Spiral-bound notebook for reading annotation, homework documentation, and note-taking.
- Instrumentation reference CD-ROM (free, from instructor). This disk contains many tutorials and datasheets in PDF format to supplement your textbook(s).
- Tool kit (see detailed list)
- Simple scientific calculator (non-programmable, non-graphing, no unit conversions, no numeration system conversions), TI-30Xa or TI-30XIIS recommended
- Portable personal computer with Ethernet port and wireless. Windows OS strongly preferred, tablets discouraged.
- Small “brick” PLC and HMI panel (*Automation Direct option*):
 - Automation Direct CLICK PLC model C0-00DD1-D (price \approx \$70) 8 discrete (DC) inputs, 6 discrete (DC) outputs
 - Automation Direct CLICK PLC model C0-02DD1-D (price \approx \$130) 4 discrete (DC) inputs, 4 discrete (DC) outputs, 2 analog inputs, 2 analog outputs, RS-485 Modbus communications port, real-time clock and calendar
 - Automation Direct CLICK 24 VDC power supply model C0-00AC (price \approx \$30) 24 VDC at 0.5 amp maximum output
 - Automation Direct C-More Micro HMI panel 3 inch EA1-S3ML-N (price \approx \$150)
 - *optional* Automation Direct C-More Micro touch-screen HMI panel 3 inch EA1-S3ML (price \approx \$190)
 - Automation Direct USB/serial adapter and cable part EA-MG-PGM-CBL (price \approx \$40) necessary for programming the C-More Micro HMI panel (also useful for programming the PLC)
 - **Note:** We have found the Automation Direct software works equally well through a 9-pin serial port as through a USB port (with converter), and is very “friendly” to use.
- Small “brick” PLC and HMI panel (*Allen-Bradley option*):
 - Rockwell (Allen-Bradley) MicroLogix 1000 model 1761-L10BWA (price \approx \$85 with BTC student discount at North Coast Electric) 6 discrete (DC) inputs, 4 discrete (relay) outputs
 - *or* Rockwell (Allen-Bradley) MicroLogix 1100 model 1763-L16BWA (price \approx \$240 with BTC student discount at North Coast Electric) 10 discrete (DC) inputs, 6 discrete (DC) outputs, 2 analog inputs, RS-485 communication port, 10 Mbit/s Ethernet communication port, embedded web server for remote monitoring of data points (series A or B programmable using free MicroLogix Lite software)
 - Rockwell (Allen-Bradley) cable part 1761-CBL-PM02 (price \approx \$30 with BTC student discount at North Coast Electric)
 - Automation Direct C-More Micro HMI panel 3 inch EA1-S3ML-N (price \approx \$150)
 - *optional* Automation Direct C-More Micro touch-screen HMI panel 3 inch EA1-S3ML (price \approx \$190)
 - Automation Direct cable part EA-MLOGIX-CBL (price \approx \$30) and adapter part EA-MG-SP1 (price \approx \$50) necessary for connecting the C-More Micro HMI panel to an Allen-Bradley MicroLogix 1000 PLC
 - Automation Direct USB/serial adapter and cable part EA-MG-PGM-CBL (price \approx \$40) necessary for programming the C-More Micro HMI panel
 - **Note:** Programming Allen-Bradley PLCs is best done using a PC with a 9-pin serial port. We have found trying to use a USB-to-serial adapter very troublesome with Allen-Bradley software!

Sequence of second-year Instrumentation courses



The particular sequence of courses you take during the second year depends on when you complete all first-year courses and enter the second year. Since students enter the second year of Instrumentation at four different times (beginnings of Summer, Fall, Winter, and Spring quarters), the particular course sequence for any student will likely be different from the course sequence of classmates.

Some second-year courses are only offered in particular quarters with those quarters not having to be in sequence, while others are offered three out of the four quarters and must be taken in sequence. The following layout shows four typical course sequences for second-year Instrumentation students, depending on when they first enter the second year of the program:

Possible course schedules depending on date of entry into 2nd year



file sequence

General tool and supply list

Wrenches

- Combination (box- and open-end) wrench set, 1/4" to 3/4" – *the most important wrench sizes are 7/16", 1/2", 9/16", and 5/8"; get these immediately!*
- Adjustable wrench, 6" handle (sometimes called "Crescent" wrench)
- Hex wrench ("Allen" wrench) set, fractional – 1/16" to 3/8"
- *Optional:* Hex wrench ("Allen" wrench) set, metric – 1.5 mm to 10 mm
- *Optional:* Miniature combination wrench set, 3/32" to 1/4" (sometimes called an "ignition wrench" set)

Note: *when turning any threaded fastener, one should choose a tool engaging the maximum amount of surface area on the fastener's head in order to reduce stress on that fastener. (e.g. Using box-end wrenches instead of adjustable wrenches; using the proper size and type of screwdriver; never using any tool that mars the fastener such as pliers or vise-grips unless absolutely necessary.)*

Pliers

- Needle-nose pliers
- Tongue-and-groove pliers (sometimes called "Channel-lock" pliers)
- Diagonal wire cutters (sometimes called "dikes")

Screwdrivers

- Slotted, 1/8" and 1/4" shaft
- Phillips, #1 and #2
- Jeweler's screwdriver set
- *Optional:* Magnetic multi-bit screwdriver (e.g. Klein Tools model 70035)

Electrical

- Multimeter, Fluke model 87-IV or better
- Alligator-clip jumper wires
- Soldering iron (10 to 40 watt) and rosin-core solder
- Resistor, potentiometer, diode assortments (from first-year lab kits)
- Package of insulated compression-style fork terminals (14 to 18 AWG wire size, #10 stud size)
- Wire strippers/terminal crimpers for 10 AWG to 18 AWG wire and insulated terminals
- *Optional:* ratcheting terminal crimp tool (e.g. Paladin 1305, Ferrules Direct FDT10011, or equivalent)

Safety

- Safety glasses or goggles (available at BTC bookstore)
- Earplugs (available at BTC bookstore)

Miscellaneous

- Simple scientific calculator (non-programmable, non-graphing, no conversions), TI-30Xa or TI-30XIIS recommended. Required for some exams!
- Portable personal computer with Ethernet port and wireless. Windows OS strongly preferred, tablets discouraged.
- Masking tape (for making temporary labels)
- Permanent marker pen
- Teflon pipe tape
- Utility knife
- Tape measure, 12 feet minimum
- Flashlight

An inexpensive source of tools is your local pawn shop. Look for tools with unlimited lifetime guarantees (e.g. *Sears* "Craftsman" brand). Check for BTC student discounts as well!

file tools

Methods of instruction

This course develops self-instructional and diagnostic skills by placing students in situations where they are required to research and think independently. In all portions of the curriculum, the goal is to avoid a passive learning environment, favoring instead *active engagement* of the learner through reading, reflection, problem-solving, and experimental activities. The curriculum may be roughly divided into two portions: *theory* and *practical*.

Theory

In the theory portion of each course, students independently research subjects *prior* to entering the classroom for discussion. This means working through all the day's assigned questions as completely as possible. This usually requires a fair amount of technical reading, and may also require setting up and running simple experiments. At the start of the classroom session, the instructor will check each student's preparation with a quiz. Students then spend the rest of the classroom time working in groups and directly with the instructor to *thoroughly* answer all questions assigned for that day, articulate problem-solving strategies, and to approach the questions from multiple perspectives. To put it simply: fact-gathering happens outside of class and is the individual responsibility of each student, so that class time may be devoted to the more complex tasks of critical thinking and problem solving where the instructor's attention is best applied.

Classroom theory sessions usually begin with either a brief Q&A discussion or with a "Virtual Troubleshooting" session where the instructor shows one of the day's diagnostic question diagrams while students propose diagnostic tests and the instructor tells those students what the test results would be given some imagined ("virtual") fault scenario, writing the test results on the board where all can see. The students then attempt to identify the nature and location of the fault, based on the test results.

Each student is free to leave the classroom when they have completely worked through all problems and have answered a "summary" quiz designed to gauge their learning during the theory session. If a student finishes ahead of time, they are free to leave, or may help tutor classmates who need extra help.

The express goal of this "inverted classroom" teaching methodology is to help each student cultivate critical-thinking and problem-solving skills, and to sharpen their abilities as independent learners. While this approach may be very new to you, it is more realistic and beneficial to the type of work done in instrumentation, where critical thinking, problem-solving, and independent learning are "must-have" skills.

Lab

In the lab portion of each course, students work in teams to install, configure, document, calibrate, and troubleshoot working instrument loop systems. Each lab exercise focuses on a different type of instrument, with a eight-day period typically allotted for completion. An ordinary lab session might look like this:

- (1) Start of practical (lab) session: announcements and planning
 - (a) The instructor makes general announcements to all students
 - (b) The instructor works with team to plan that day's goals, making sure each team member has a clear idea of what they should accomplish
- (2) Teams work on lab unit completion according to recommended schedule:
 - (First day) Select and bench-test instrument(s)
 - (One day) Connect instrument(s) into a complete loop
 - (One day) Each team member drafts their own loop documentation, inspection done as a team (with instructor)
 - (One or two days) Each team member calibrates/configures the instrument(s)
 - (Remaining days, up to last) Each team member troubleshoots the instrument loop
- (3) End of practical (lab) session: debriefing where each team reports on their work to the whole class

Troubleshooting assessments must meet the following guidelines:

- Troubleshooting must be performed *on a system the student did not build themselves*. This forces students to rely on another team's documentation rather than their own memory of how the system was built.
- Each student must individually demonstrate proper troubleshooting technique.
- Simply finding the fault is not good enough. Each student must consistently demonstrate sound reasoning while troubleshooting.
- If a student fails to properly diagnose the system fault, they must attempt (as many times as necessary) with different scenarios until they do, reviewing any mistakes with the instructor after each failed attempt.

Distance delivery methods

Sometimes the demands of life prevent students from attending college 6 hours per day. In such cases, there exist alternatives to the normal 8:00 AM to 3:00 PM class/lab schedule, allowing students to complete coursework in non-traditional ways, at a “distance” from the college campus proper.

For such “distance” students, the same worksheets, lab activities, exams, and academic standards still apply. Instead of working in small groups and in teams to complete theory and lab sections, though, students participating in an alternative fashion must do all the work themselves. Participation via teleconferencing, video- or audio-recorded small-group sessions, and such is encouraged and supported.

There is no recording of hours attended or tardiness for students participating in this manner. The pace of the course is likewise determined by the “distance” student. Experience has shown that it is a benefit for “distance” students to maintain the same pace as their on-campus classmates whenever possible.

In lieu of small-group activities and class discussions, comprehension of the theory portion of each course will be ensured by completing and submitting detailed answers for *all* worksheet questions, not just passing daily quizzes as is the standard for conventional students. The instructor will discuss any incomplete and/or incorrect worksheet answers with the student, and ask that those questions be re-answered by the student to correct any misunderstandings before moving on.

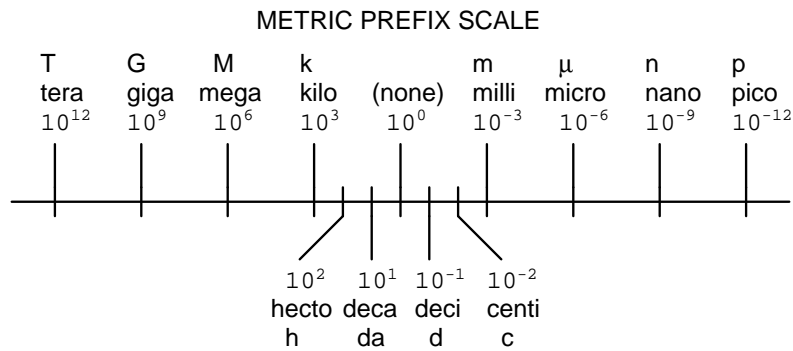
Labwork is perhaps the most difficult portion of the curriculum for a “distance” student to complete, since the equipment used in Instrumentation is typically too large and expensive to leave the school lab facility. “Distance” students must find a way to complete the required lab activities, either by arranging time in the school lab facility and/or completing activities on equivalent equipment outside of school (e.g. at their place of employment, if applicable). Labwork completed outside of school must be validated by a supervisor and/or documented via photograph or videorecording.

Conventional students may opt to switch to “distance” mode at any time. This has proven to be a benefit to students whose lives are disrupted by catastrophic events. Likewise, “distance” students may switch back to conventional mode if and when their schedules permit. Although the existence of alternative modes of student participation is a great benefit for students with challenging schedules, it requires a greater investment of time and a greater level of self-discipline than the traditional mode where the student attends school for 6 hours every day. No student should consider the “distance” mode of learning a way to have more free time to themselves, because they will actually spend more time engaged in the coursework than if they attend school on a regular schedule. It exists merely for the sake of those who cannot attend during regular school hours, as an alternative to course withdrawal.

Metric prefixes and conversion constants

- **Metric prefixes**

- Yotta = 10^{24} Symbol: Y
- Zeta = 10^{21} Symbol: Z
- Exa = 10^{18} Symbol: E
- Peta = 10^{15} Symbol: P
- Tera = 10^{12} Symbol: T
- Giga = 10^9 Symbol: G
- Mega = 10^6 Symbol: M
- Kilo = 10^3 Symbol: k
- Hecto = 10^2 Symbol: h
- Deca = 10^1 Symbol: da
- Deci = 10^{-1} Symbol: d
- Centi = 10^{-2} Symbol: c
- Milli = 10^{-3} Symbol: m
- Micro = 10^{-6} Symbol: μ
- Nano = 10^{-9} Symbol: n
- Pico = 10^{-12} Symbol: p
- Femto = 10^{-15} Symbol: f
- Atto = 10^{-18} Symbol: a
- Zepto = 10^{-21} Symbol: z
- Yocto = 10^{-24} Symbol: y



- **Conversion formulae for temperature**

- $^{\circ}\text{F} = (^{\circ}\text{C})(9/5) + 32$
- $^{\circ}\text{C} = (^{\circ}\text{F} - 32)(5/9)$
- $^{\circ}\text{R} = ^{\circ}\text{F} + 459.67$
- $\text{K} = ^{\circ}\text{C} + 273.15$

Conversion equivalencies for distance

- 1 inch (in) = 2.540000 centimeter (cm)
- 1 foot (ft) = 12 inches (in)
- 1 yard (yd) = 3 feet (ft)
- 1 mile (mi) = 5280 feet (ft)

Conversion equivalencies for volume

1 gallon (gal) = 231.0 cubic inches (in³) = 4 quarts (qt) = 8 pints (pt) = 128 fluid ounces (fl. oz.)
= 3.7854 liters (l)

1 milliliter (ml) = 1 cubic centimeter (cm³)

Conversion equivalencies for velocity

1 mile per hour (mi/h) = 88 feet per minute (ft/m) = 1.46667 feet per second (ft/s) = 1.60934
kilometer per hour (km/h) = 0.44704 meter per second (m/s) = 0.868976 knot (knot – international)

Conversion equivalencies for mass

1 pound (lbm) = 0.45359 kilogram (kg) = 0.031081 slugs

Conversion equivalencies for force

1 pound-force (lbf) = 4.44822 newton (N)

Conversion equivalencies for area

1 acre = 43560 square feet (ft²) = 4840 square yards (yd²) = 4046.86 square meters (m²)

Conversion equivalencies for common pressure units (either all gauge or all absolute)

1 pound per square inch (PSI) = 2.03602 inches of mercury (in. Hg) = 27.6799 inches of water (in.
W.C.) = 6.894757 kilo-pascals (kPa) = 0.06894757 bar

1 bar = 100 kilo-pascals (kPa) = 14.504 pounds per square inch (PSI)

Conversion equivalencies for absolute pressure units (only)

1 atmosphere (Atm) = 14.7 pounds per square inch absolute (PSIA) = 101.325 kilo-pascals absolute
(kPaA) = 1.01325 bar (bar) = 760 millimeters of mercury absolute (mmHgA) = 760 torr (torr)

Conversion equivalencies for energy or work

1 british thermal unit (Btu – “International Table”) = 251.996 calories (cal – “International Table”)
= 1055.06 joules (J) = 1055.06 watt-seconds (W-s) = 0.293071 watt-hour (W-hr) = 1.05506 x 10¹⁰
ergs (erg) = 778.169 foot-pound-force (ft-lbf)

Conversion equivalencies for power

1 horsepower (hp – 550 ft-lbf/s) = 745.7 watts (W) = 2544.43 british thermal units per hour
(Btu/hr) = 0.0760181 boiler horsepower (hp – boiler)

Acceleration of gravity (free fall), Earth standard

9.806650 meters per second per second (m/s²) = 32.1740 feet per second per second (ft/s²)

Physical constants

Speed of light in a vacuum (c) = 2.9979×10^8 meters per second (m/s) = 186,281 miles per second (mi/s)

Avogadro's number (N_A) = 6.022×10^{23} per mole (mol^{-1})

Electronic charge (e) = 1.602×10^{-19} Coulomb (C)

Boltzmann's constant (k) = 1.38×10^{-23} Joules per Kelvin (J/K)

Stefan-Boltzmann constant (σ) = 5.67×10^{-8} Watts per square meter-Kelvin⁴ ($\text{W}/\text{m}^2 \cdot \text{K}^4$)

Molar gas constant (R) = 8.314 Joules per mole-Kelvin (J/mol-K)

Properties of Water

Freezing point at sea level = $32^\circ\text{F} = 0^\circ\text{C}$

Boiling point at sea level = $212^\circ\text{F} = 100^\circ\text{C}$

Density of water at 4°C = $1000 \text{ kg}/\text{m}^3 = 1 \text{ g}/\text{cm}^3 = 1 \text{ kg}/\text{liter} = 62.428 \text{ lb}/\text{ft}^3 = 1.94 \text{ slugs}/\text{ft}^3$

Specific heat of water at 14°C = $1.00002 \text{ calories}/\text{g} \cdot ^\circ\text{C} = 1 \text{ BTU}/\text{lb} \cdot ^\circ\text{F} = 4.1869 \text{ Joules}/\text{g} \cdot ^\circ\text{C}$

Specific heat of ice $\approx 0.5 \text{ calories}/\text{g} \cdot ^\circ\text{C}$

Specific heat of steam $\approx 0.48 \text{ calories}/\text{g} \cdot ^\circ\text{C}$

Absolute viscosity of water at 20°C = 1.0019 centipoise (cp) = 0.0010019 Pascal-seconds (Pa·s)

Surface tension of water (in contact with air) at 18°C = 73.05 dynes/cm

pH of pure water at 25°C = 7.0 (*pH scale = 0 to 14*)

Properties of Dry Air at sea level

Density of dry air at 20°C and 760 torr = $1.204 \text{ mg}/\text{cm}^3 = 1.204 \text{ kg}/\text{m}^3 = 0.075 \text{ lb}/\text{ft}^3 = 0.00235 \text{ slugs}/\text{ft}^3$

Absolute viscosity of dry air at 20°C and 760 torr = 0.018 centipoise (cp) = 1.8×10^{-5} Pascal-seconds (Pa·s)

file conversion_constants

How to get the most out of academic reading:

- Articulate your thoughts as you read (i.e. “have a conversation” with the author). This will develop *metacognition*: active supervision of your own thoughts. Write your thoughts as you read, noting points of agreement, disagreement, confusion, epiphanies, and connections between different concepts or applications. These notes should also document important math formulae, explaining in your own words what each formula means and the proper units of measurement used.
- Outline, don’t highlight! Writing your own summary or outline is a far more effective way to comprehend a text than simply underlining and highlighting key words. A suggested ratio is one sentence of your own thoughts per paragraph of text read. Note points of disagreement or confusion to explore later.
- Work through all mathematical exercises shown within the text, to ensure you understand all the steps.
- Imagine explaining concepts you’ve just learned to someone else. Teaching forces you to distill concepts to their essence, thereby clarifying those concepts, revealing assumptions, and exposing misconceptions. Your goal is to create the simplest explanation that is still technically accurate.
- Write your own questions based on what you read, as though you are a teacher preparing to test students’ comprehension of the subject matter.

How to effectively problem-solve and troubleshoot:

- Rely on principles, not procedures. Don’t be satisfied with memorizing steps – learn *why* those steps work. Each one should make logical sense and have real-world meaning to you.
- Sketch a diagram to help visualize the problem. Sketch a graph showing how variables relate. When building a real system, always prototype it on paper and analyze its function *before* constructing it.
- Identify what it is you need to solve, identify all relevant data, identify all units of measurement, identify any general principles or formulae linking the given information to the solution, and then identify any “missing pieces” to a solution. Annotate all diagrams with this data.
- Perform “thought experiments” to explore the effects of different conditions for theoretical problems. When troubleshooting, perform *diagnostic tests* rather than just visually inspect for faults.
- Simplify the problem and solve that simplified problem to identify strategies applicable to the original problem (e.g. change quantitative to qualitative, or visa-versa; substitute easier numerical values; eliminate confusing details; add details to eliminate unknowns; consider simple limiting cases; apply an analogy). Often you can add or remove components in a malfunctioning system to simplify it as well and better identify the nature and location of the problem.
- Work “backward” from a hypothetical solution to a new set of given conditions.

How to manage your time:

- Avoid procrastination. Work now and play later, or else you will create trouble for yourself. Schedule your work appropriate to the *place* you’re in as well: e.g. don’t waste lab time doing things that could be done anywhere else, when there is work to be done that requires the lab.
- Eliminate distractions. Kill your television and video games. Study in places where you can concentrate.
- Use your “in between” time productively. Don’t leave campus for lunch. Arrive to school early. If you finish your assigned work early, begin working on the next assignment.

Above all, cultivate persistence. Persistent effort is necessary to master anything non-trivial. The keys to persistence are (1) having the desire to achieve that mastery, and (2) realizing challenges are normal and not an indication of something gone wrong. A common error is to equate *easy* with *effective*: students often believe learning should be easy if everything is done right. The truth is that mastery never comes easy!

file question0

Creative Commons License

This worksheet is licensed under the **Creative Commons Attribution 4.0 International Public License**. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California 94105, USA. The terms and conditions of this license allow for free copying, distribution, and/or modification of all licensed works by the general public.

Simple explanation of Attribution License:

The licensor (Tony Kuphaldt) permits others to copy, distribute, display, and otherwise use this work. In return, licensees must give the original author(s) credit. For the full license text, please visit <http://creativecommons.org/licenses/by/4.0/> on the internet.

More detailed explanation of Attribution License:

Under the terms and conditions of the Creative Commons Attribution License, you may make freely use, make copies, and even modify these worksheets (and the individual “source” files comprising them) without having to ask me (the author and licensor) for permission. The one thing you must do is properly credit my original authorship. Basically, this protects my efforts against plagiarism without hindering the end-user as would normally be the case under full copyright protection. This gives educators a great deal of freedom in how they might adapt my learning materials to their unique needs, removing all financial and legal barriers which would normally hinder if not prevent creative use.

Nothing in the License prohibits the sale of original or adapted materials by others. You are free to copy what I have created, modify them if you please (or not), and then sell them at any price. Once again, the only catch is that you must give proper credit to myself as the original author and licensor. Given that these worksheets will be continually made available on the internet for free download, though, few people will pay for what you are selling unless you have somehow added value.

Nothing in the License prohibits the application of a more restrictive license (or no license at all) to derivative works. This means you can add your own content to that which I have made, and then exercise full copyright restriction over the new (derivative) work, choosing not to release your additions under the same free and open terms. An example of where you might wish to do this is if you are a teacher who desires to add a detailed “answer key” for your own benefit but *not* to make this answer key available to anyone else (e.g. students).

Note: the text on this page is not a license. It is simply a handy reference for understanding the Legal Code (the full license) - it is a human-readable expression of some of its key terms. Think of it as the user-friendly interface to the Legal Code beneath. This simple explanation itself has no legal value, and its contents do not appear in the actual license.

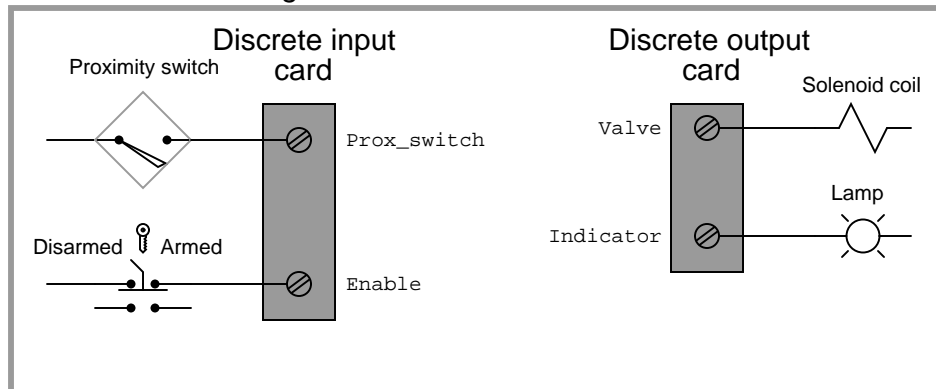
file license

Questions

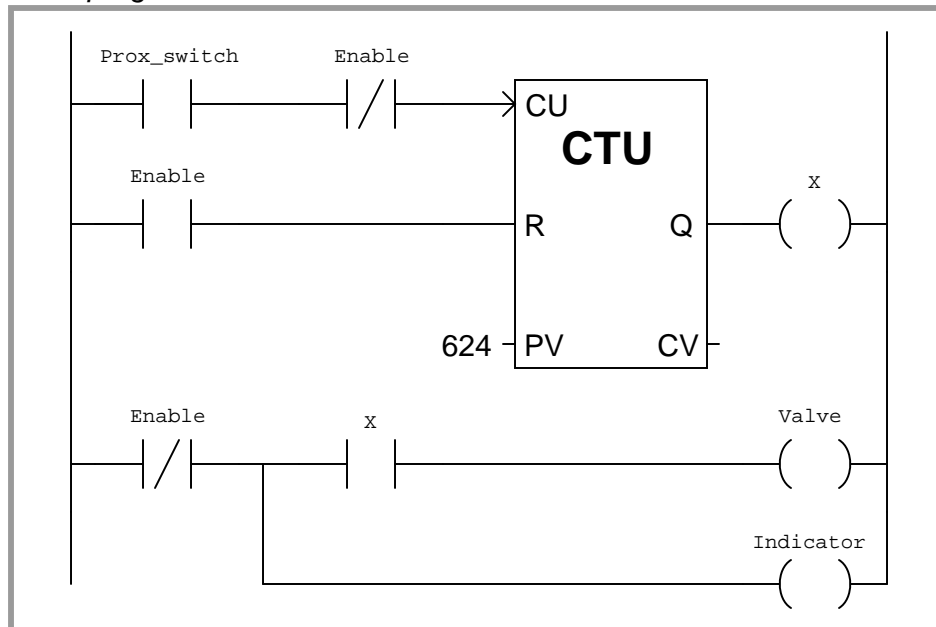
Question 1

Suppose we have an IEC 61131-3 compliant PLC connected to a proximity switch, a key switch, and two loads (a lamp and a solenoid coil actuating a valve) as shown in this illustration. The proximity switch counts objects passing by on a conveyor belt:

Real-world I/O wiring



PLC program



Examine the offline display of this PLC's relay ladder logic program, determining the status of the lamp and of the valve solenoid coil after the proximity switch counts 6 more objects passing by, assuming the Enable keyswitch is in the "Armed" position and the current count value is 620.

[file i04528](#)

Question 2

Read and outline the “Digital Representation of Numerical Data” section of the “Digital Data Acquisition and Network” chapter in your *Lessons In Industrial Instrumentation* textbook.

The purpose of your outline is to foster close reading of the text, to facilitate quick referencing of specific points within the text, to record questions of your own, and to practice clear writing. Your outline must meet the following standards for full credit: *every major idea contained in the text represented in your outline, entirely in your own words (i.e. no copying of text), written in a legible and comprehensible manner, of sufficient quality that others would find it informative.* Incomplete, illegible, cryptic, and/or plagiarized outlines will not receive full credit. A suggestion is one sentence of your own per paragraph of source text. Helpful additions include:

- Noting questions or points of confusion you have from the reading
- Page numbers from the source text for quick reference during discussion
- Images copied from the text (or sketched by you) to illustrate concepts
- References to previously learned concepts

[file i04395](#)

Question 3

Read and outline the “Human-Machine Interfaces” section of the “Programmable Logic Controllers” chapter in your *Lessons In Industrial Instrumentation* textbook.

The purpose of your outline is to foster close reading of the text, to facilitate quick referencing of specific points within the text, to record questions of your own, and to practice clear writing. Your outline must meet the following standards for full credit: *every major idea contained in the text represented in your outline, entirely in your own words (i.e. no copying of text), written in a legible and comprehensible manner, of sufficient quality that others would find it informative.* Incomplete, illegible, cryptic, and/or plagiarized outlines will not receive full credit. A suggestion is one sentence of your own per paragraph of source text. Helpful additions include:

- Noting questions or points of confusion you have from the reading
- Page numbers from the source text for quick reference during discussion
- Images copied from the text (or sketched by you) to illustrate concepts
- References to previously learned concepts

[file i04526](#)

Question 4

Identify the integer data types your PLC uses for the “accumulator” or “current” value registers in its *counter* instructions and *timer* instructions. Note that different data types may be used for each of these instruction types! Are the registers 16 bit or 32 bit? Are they signed or unsigned? How can you tell?

Suggestions for Socratic discussion

- Why does the data type matter to us?
- What is the largest number value reachable with each of the identified data types?
- Prove the bit size of your PLC’s counter and/or timer instructions using a demonstration program (such as the demo programs you created earlier for exploring counter and timer instructions).
- Suppose you needed the PLC to be able to count (or to time) to a value greater than that possible by the basic counter or timer instruction on its own. For example, your counter uses a 16-bit signed integer register, and you need to be able to count up to +1 million. How could you do this given the limitations of the signed 16-bit counter registers?

[file i03677](#)

Question 5

One of the most important steps in programming an HMI (Human-Machine Interface) to communication with a PLC is to configure its *tagname database*. This is a list of all data points within the PLC that the HMI must read from and/or write to, as well as any data points within the HMI itself needed for the graphical interface to operate.

Your task is to configure your own HMI panel twice: once to read and write data points for a Koyo CLICK PLC, and again to read and write data points for an Allen-Bradley SLC 500 PLC. Show your completed tagname database listings (captured screenshots are okay) to the instructor for verification.

Data points for Koyo CLICK PLC

- Discrete input X6 (assign HMI tagname “Speed_switch”)
- Discrete output Y2 (assign HMI tagname “Alarm_siren”)
- Counter C3 current value (assign HMI tagname “Overspeed_count”)
- Timer T5 current value (assign HMI tagname “Alarm_duration”)

Data points for Allen-Bradley SLC 500 PLC

- Discrete input I:3/5 (assign HMI tagname “Start_switch”)
- Discrete output O:1/2 (assign HMI tagname “Motor_contactor”)
- Counter C5:1 accumulator value (assign HMI tagname “Start_count”)
- Timer T4:0 accumulator value (assign HMI tagname “Run_time”)

Note that you do not need physical access to any of these PLC types in order to construct the HMI tagname databases for them. All you need to do is configure the HMI editing software as though the HMI *will eventually be* communicating with the appropriate PLC, and then build the database accordingly.

Suggestions for Socratic discussion

- Your HMI will likely give you options for which type of integer variables to select, 16 bit versus 32 bit, signed versus unsigned. How do you know what is the appropriate integer size for each PLC application?

[file i04590](#)

Programming Challenge – HMI control of sprinkler valves

Suppose an instrument technician wishes to have a PLC-controlled sprinkler system in his yard, with an HMI panel inside his house with “pushbutton” graphics on the screen where he may conveniently activate sprinkler water solenoid valves. To begin this project, the technician connects two solenoid valves to two discrete outputs on his PLC: one valve opens up to send water to his fruit tree sprinkler nozzles while the other valve opens up to fire a jet of water at the nearby fire hydrant where his neighbor’s dog likes to mark his territory.

Create a simple HMI project with two “pushbutton” icons on the screen. The first icon will directly activate the PLC output bit for the fruit tree sprinklers, and it needs to have a *toggle* action: pressing this icon once turns the bit on, and pressing it a second time turns it off. The second icon will directly activate the PLC output bit for the anti-dog water cannon, and it needs to have a *momentary* action: pressing this icon activates the water jet, and releasing it stops the water jet.

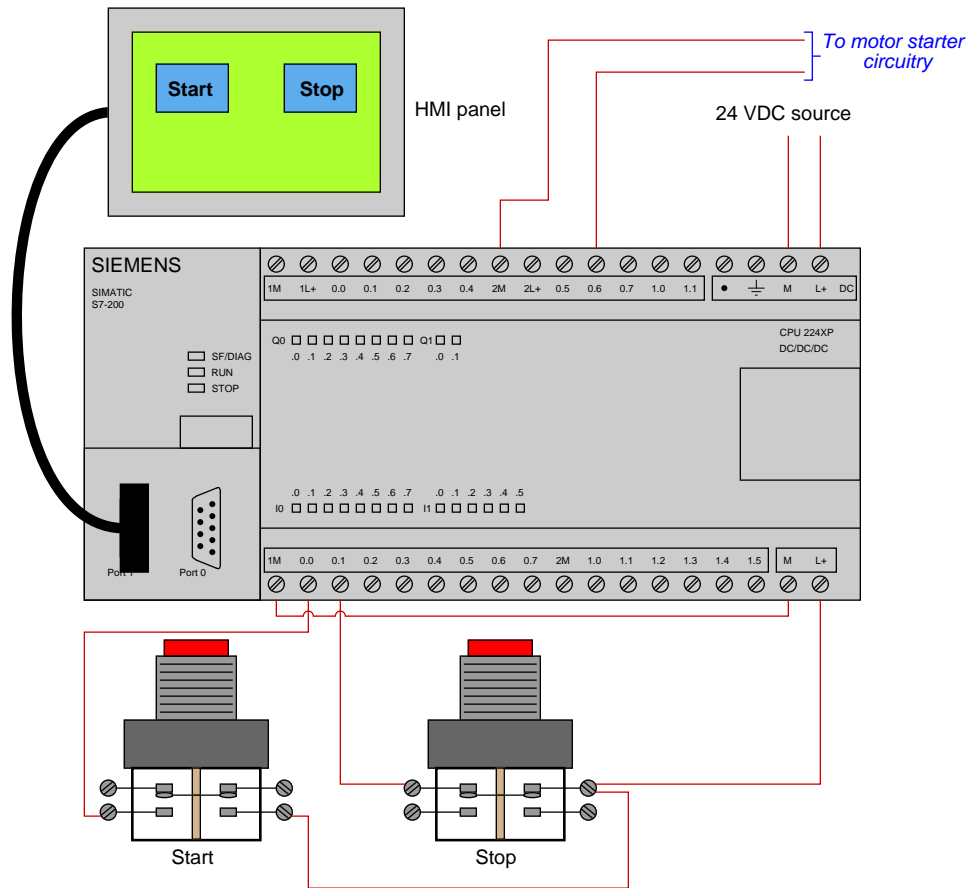
The PLC itself should have no instructions programmed in it (except perhaps for an END rung to avoid a processor error).

Suggestions for Socratic discussion
--

- What types of HMI data tags (boolean, integer, floating-point, ASCII, etc.) should be used for both these “pushbutton” objects?
- How do you specify the action (toggle, momentary, etc.) of the “pushbutton” icons on the HMI screen?
- What steps must you take to create appropriate tag names in the HMI for the PLC’s data points?
- Explain why the PLC should have no program in it, or conversely, what bad things could happen if a program existed in the PLC with coils addressed to the same output bits the HMI was attempting to write to.

Question 7

A technician is troubleshooting a problem with a PLC program she just wrote. The program is a simple motor start/stop control, with an HMI panel interfacing with the Siemens S7-200 PLC for another set of start/stop “buttons” to complement the hard-wired momentary-contact pushbuttons:



The problem is, the motor starts and stops just fine by pushing the real pushbuttons, but not by pressing the “button” objects on the HMI’s touch-screen display. For some reason, the HMI is unable to either start or stop the motor. The tagname database programmed into the HMI only has two tags: **Motor_Start** and **Motor_Stop**, assigned (with “write” permission) to addresses **I0.0** and **I0.1** respectively.

Identify what the problem is in this newly-programmed system, and also how to fix it. Additionally, determine whether the pushbutton switch inputs are wired to *source* current to the PLC or *sink* current from it.

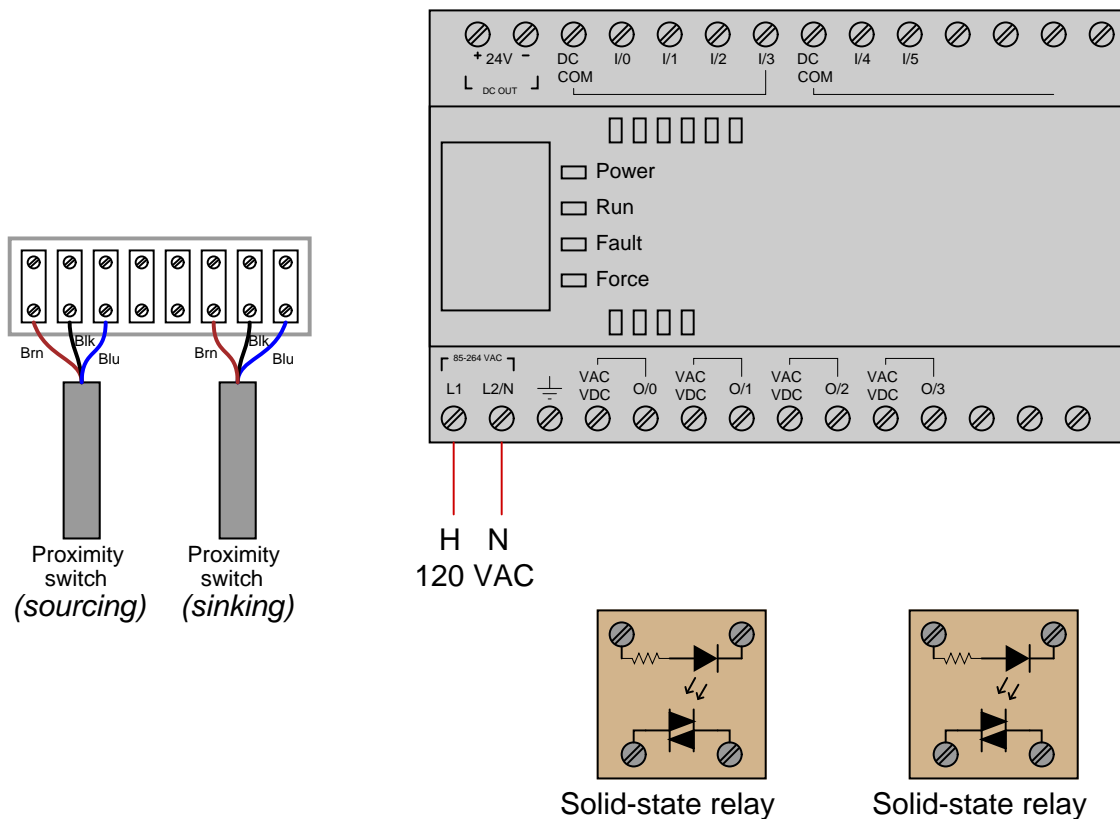
Suggestions for Socratic discussion

- If the “Start” pushbutton switch’s wire came disconnected from input **I0.0**, would the HMI panel then be able to start up the motor?
- If the “Stop” pushbutton switch’s wire came disconnected from input **I0.1**, would the HMI panel then be able to stop the motor?
- If the “Start” and “Stop” pushbutton switches were re-wired to connect to inputs **I0.5** and **I0.6** respectively, would this solve the problem?

file i03595

Question 8

Sketch the wires necessary to connect two proximity switches and two solid-state relay coils to the following Allen-Bradley MicroLogix 1000 PLC (model 1761-L10BWA, with 6 discrete DC inputs either sourcing or sinking, and 4 discrete relay contact outputs). Connect the sourcing switch to input I:0/0, the sinking switch to input I:0/4, and the two solid-state relays to outputs O:0/1 and O:0/2:



Suggestions for Socratic discussion

- What advantages do *solid-state* relays enjoy over their electromechanical counterparts?
- Can these solid-state relays switch DC, AC, or both?
- Identify how the behavior of a TRIAC differs from that of a bipolar or field-effect transistor.

[file i04524](#)

Programming Challenge and Comparison – Motor control with HMI and pushbutton

Suppose we need to have a PLC provide start/stop control for an electric motor, from two different locations. Near the motor we have a pair of momentary-contact pushbutton switches: one to start the motor and another to stop it. However, at the control room we have an HMI panel where the operators would like to have an additional set of momentary start and stop “pushbuttons” so they may start and stop the motor from that location as well.

Write a PLC program integrating these two sets of start/stop controls together, so that the motor may be controlled from either location. Also provide a “run” indicator on the HMI panel so operators there know when the motor is actually running. Demonstrate the program’s operation using switches connected to its inputs to simulate the discrete inputs in a real application.

When your program is complete and tested, capture a screen-shot of it as it appears on your computer, and prepare to present your program solution to the class in a review session for everyone to see and critique. The purpose of this review session is to see multiple solutions to one problem, explore different programming techniques, and gain experience interpreting PLC programs others have written. When presenting your program (either individually or as a team), prepare to discuss the following points:

- Identify the “tag names” or “nicknames” used within your program to label I/O and other bits in memory
- Follow the sequence of operation in your program, simulating the system in action
- Identify any special or otherwise non-standard instructions used in your program, and explain why you decided to take that approach
- Show the comments placed in your program, to help explain how and why it works
- How you designed the program (i.e. what steps you took to go from a concept to a working program)

Suggestions for Socratic discussion
--

- Why would it be a bad idea for the HMI to write to the same PLC addresses used for discrete inputs (e.g. `X1`, `I:0/0`, `I0.0`, etc.) in an attempt to have dual control over the motor? What bits in the PLC memory should the HMI write to in order to avoid this problem?
- How may we ensure consistent action at the motor if conflicting commands are given from the pushbuttons and HMI (e.g. “Start” pushbutton pressed while “Stop” HMI control activated)?

[file i02485](#)

Question 10

Convert the following numbers from binary (base-two) to decimal (base-ten):

- $10_2 =$
- $1010_2 =$
- $10011_2 =$
- $11100_2 =$
- $10111_2 =$
- $101011_2 =$
- $11100110_2 =$
- $10001101011_2 =$

Describe a general, step-by-step procedure for converting binary numbers into decimal numbers.
file i02164

Question 11

Convert the following numbers from decimal (base-ten) to binary (base-two):

- $7_{10} =$
- $10_{10} =$
- $19_{10} =$
- $250_{10} =$
- $511_{10} =$
- $824_{10} =$
- $1044_{10} =$
- $9241_{10} =$

Describe a general, step-by-step procedure for converting decimal numbers into binary numbers.
file i02165

Question 12

A numeration system often used as a “shorthand” way of writing large binary numbers is the *octal*, or base-eight, system. Based on what you know of place-weighted numeration systems, describe how many valid ciphers exist in the octal system, and the respective “weights” of each place in an octal number.

Also, perform the following conversions:

- 35_8 into decimal:
- 16_{10} into octal:
- 110010_2 into octal:
- 51_8 into binary:

Suggestions for Socratic discussion
--

- If binary is the “natural language” of digital electronic circuits, why do we even bother with other numeration systems such as hex and octal?
- Why is octal considered a “shorthand” notation for binary numbers?

[file i02166](#)

Question 13

A numeration system often used as a “shorthand” way of writing large binary numbers is the *hexadecimal*, or base-sixteen, system.

Based on what you know of place-weighted numeration systems, describe how many valid ciphers exist in the hexadecimal system, and the respective “weights” of each place in a hexadecimal number.

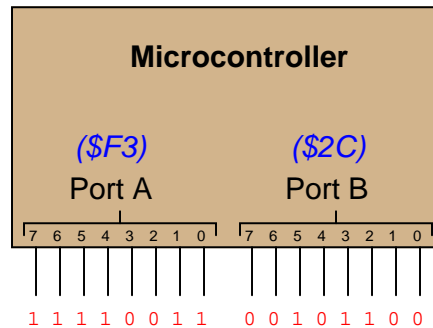
Also, perform the following conversions:

- 35_{16} into decimal:
- 34_{10} into hexadecimal:
- 11100010_2 into hexadecimal:
- 93_{16} into binary:

[file i02167](#)

Question 14

Digital computers communicate with external devices through *ports*: sets of terminals usually arranged in groups of 4, 8, 16, or more (4 bits = 1 *nybble*, 8 bits = 1 *byte*, 16 bits = 2 bytes). These terminals may be set to high or low logic states by writing a program for the computer that sends a numerical value to the port. For example, here is an illustration of a microcontroller being instructed to send the hexadecimal number **F3** to port A and **2C** to port B:



Suppose we wished to use the upper four bits of port A (pins 7, 6, 5, and 4) to drive the coils of a stepper motor in this eight-step sequence:

Step 1: 0001
Step 2: 0011
Step 3: 0010
Step 4: 0110
Step 5: 0100
Step 6: 1100
Step 7: 1000
Step 8: 1001

As each pin goes high, it drives a power MOSFET on, which sends current through that respective coil of the stepper motor. By following a "shift" sequence as shown, the motor will rotate a small amount for each cycle.

Write the necessary sequence of numbers to be sent to port A to generate this specific order of bit shifts, in hexadecimal. Leave the lower four bit of port A all in the low logic state.

file i02168

Question 15

When representing non-whole numbers, we extend the “places” of our decimal numeration system past the right of the decimal point, like this:

<i>Decimal place-weights</i>								
$\frac{2}{10^3}$	$\frac{5}{10^2}$	$\frac{9}{10^1}$	$\frac{6}{10^0}$	•	$\frac{3}{10^{-1}}$	$\frac{8}{10^{-2}}$	$\frac{0}{10^{-3}}$	$\frac{4}{10^{-4}}$

$$2 \times 10^3 = 2000$$

$$3 \times 10^{-1} = \frac{3}{10}$$

$$5 \times 10^2 = 500$$

$$8 \times 10^{-2} = \frac{8}{100}$$

$$9 \times 10^1 = 90$$

$$0 \times 10^{-3} = \frac{0}{1000}$$

$$6 \times 10^0 = 6$$

$$4 \times 10^{-4} = \frac{4}{10000}$$

How do you suppose we represent non-whole numbers in a numeration system with a base (or “radix”) other than ten? In the following examples, write the place-weight values underneath each place, and then determine the decimal equivalent of each example number:

<i>Binary place-weights</i>								
<u>1</u>	<u>0</u>	<u>0</u>	<u>1</u>	•	<u>1</u>	<u>0</u>	<u>1</u>	<u>1</u>

<i>Octal place-weights</i>								
<u>4</u>	<u>0</u>	<u>2</u>	<u>7</u>	•	<u>3</u>	<u>6</u>	<u>1</u>	<u>2</u>

<i>Hexadecimal place-weights</i>								
<u>C</u>	<u>1</u>	<u>A</u>	<u>6</u>	•	<u>3</u>	<u>2</u>	<u>B</u>	<u>9</u>

Question 16

Complete this table, performing all necessary conversions between numeration systems:

Binary	Octal	Decimal	Hexadecimal
10010			
		92	
			1A
	67		
1100101			
			122
		1000	
	336		
1011010110			

Question 17

In digital computer systems, binary numbers are often represented by a fixed number of bits, such as 8, or 16, or 32. Such bit groupings are often given special names, because they are so common in digital systems:

- byte
- nybble
- word

How many binary bits is represented by each of the above terms?

And, for those looking for more challenge, try defining these terms:

- nickle
- deckle
- chawmp
- playte
- dynner

Question 18

The IEEE 754-1985 standard for representing floating-point numbers uses 32 bits for single-precision numbers. The first bit is the *sign*, the next eight bits are the *exponent*, and the last 23 bits are the *mantissa*:

Sign	Exponent (E)	Mantissa (m)
0	00000000	00000000000000000000000

$\pm 0.m \times 2^{E-127}$ Single-precision, when exponent bits are all zero

$\pm 1.m \times 2^{E-127}$ Single-precision, when exponent bits are not all zero

Based on this standard, determine the values of the following single-precision IEEE 754 floating-point numbers:

Sign	Exponent (E)	Mantissa (m)
1	00101100	000011110000001100000000
0	11001100	110101100000000000000000
0	00000000	11111100000110001000000

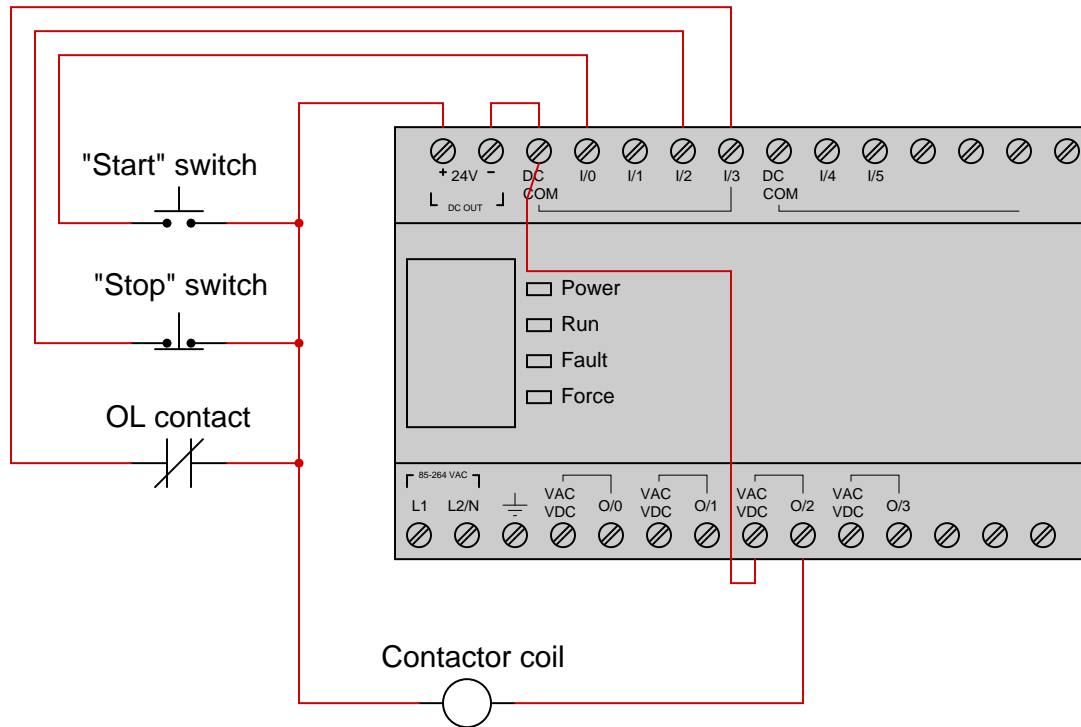
Finally, how do you represent the number 1 (1.0×2^0) in this floating-point format?
[file i01851](#)

Question 19

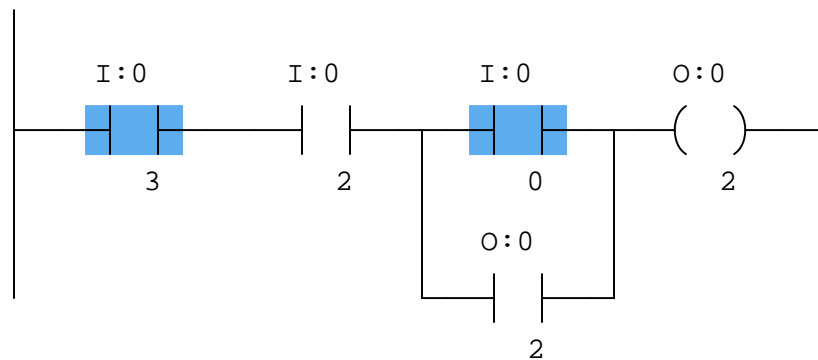
Question 20

Question 21

Suppose we have an Allen-Bradley MicroLogix 1000 controller connected to a pair of momentary-contact pushbutton switches and contactor controlling power to an electric motor as shown in this illustration:



This motor control system has a problem, though: the motor refuses to start when the “Start” pushbutton is pressed. Examine the “live” display of the ladder logic program inside this Allen-Bradley PLC to determine what the problem is, assuming an operator is continuously pressing the “Start” pushbutton as you examine the program:



Identify at least two faults that could account for all you see here.

[file i04530](#)

Programming Challenge – model rocket launch timer with HMI screen

Suppose we wish to automate a model rocket launchpad using a PLC to time the launch of the rocket. When the “Countdown” pushbutton (momentary contact) is pressed, the PLC will begin a counting sequence to launch the rocket. After 10 seconds, a discrete output point on the PLC will activate to power the rocket engine’s igniter.

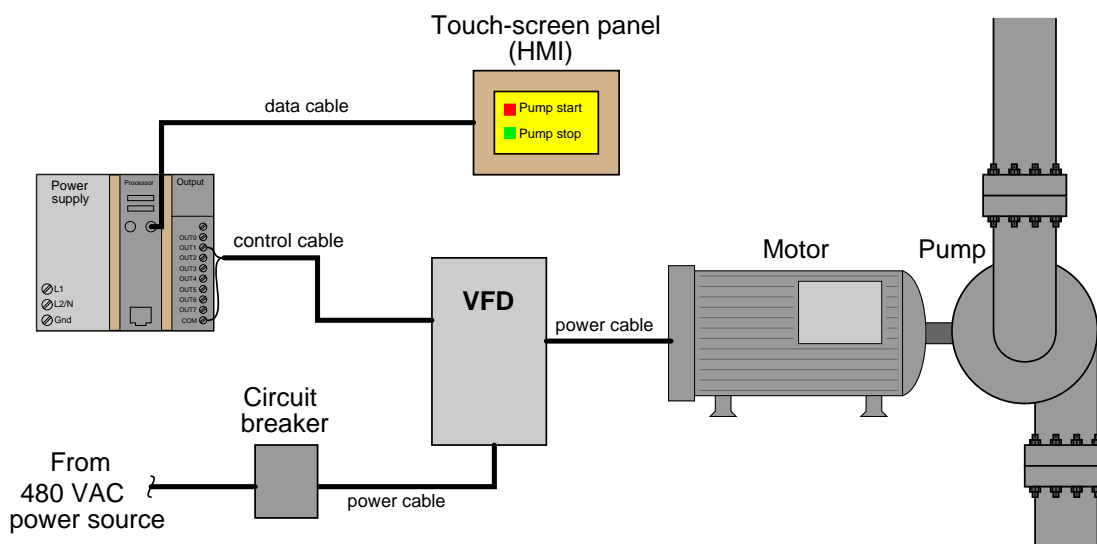
Write a PLC program to perform this countdown function, and program an HMI to display the 10-second count from beginning to end in the form of a bargraph.

Suggestions for Socratic discussion
--

- How can you make this function latching, so that no one needs to hold the “Countdown” pushbutton the entire 10 seconds, but rather merely needs to press it once and release?

Question 23

A control system for a chemical reaction process uses a VFD to power the charge pump introducing chemical fluids into a reaction vessel:



This system is newly constructed, and when the operators try starting up the pump by pressing the “Pump start” icon on the touch-screen, nothing happens. A technician temporarily connects a jumper wire across the two terminals at the VFD where the control cable lands. At this, the motor starts up and runs.

Identify the likelihood of each specified fault for this circuit. Consider each fault one at a time (i.e. no coincidental faults), determining whether or not each fault could independently account for *all* measurements and symptoms in this circuit.

Fault	Possible	Impossible
Circuit breaker off		
Touch-screen panel malfunctioning		
Programming error in PLC		
Faulted power cable between VFD and motor		
Faulted power cable between breaker and VFD		
Output card malfunctioning		
Open control cable		
Shorted control cable		
Open data cable		

Finally, identify the *next* diagnostic test or measurement you would make on this system. Explain how the result(s) of this next test or measurement help further identify the location and/or nature of the fault.

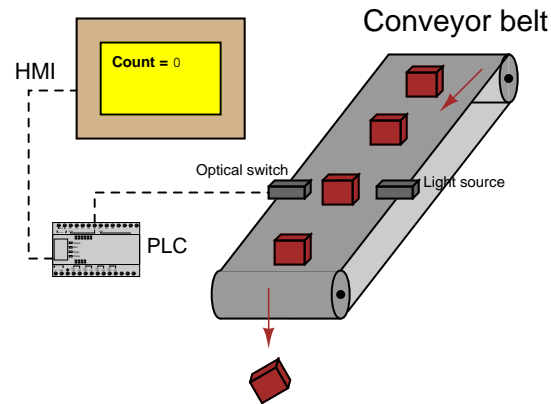
Suggestions for Socratic discussion

- For those who have studied centrifugal pumps, which way is liquid flow going through this pump, up or down?

[file i04151](#)

Question 24

A PLC counts packages coming by on a conveyor belt in a manufacturing facility. An optical sensor detects these packages as they travel by on the conveyor belt:



Unfortunately, something is not working correctly in this system. The HMI display continues to read a count value of zero no matter how many packages pass by the sensor switch. This very same system worked just fine three days ago, and had been working fine for one whole year before that.

Brainstorm at least five different faults that could account for this problem, and then devise a “next test” you would conduct to narrow the field of potential faults. The simpler this test (i.e. the least amount of time to conduct and the less complicated test equipment required), the better!

-
-
-
-
-

Next test:

Suggestions for Socratic discussion

- Explain how your proposed diagnostic test would either confirm or eliminate certain fault possibilities.
[file i03596](#)

Question 25

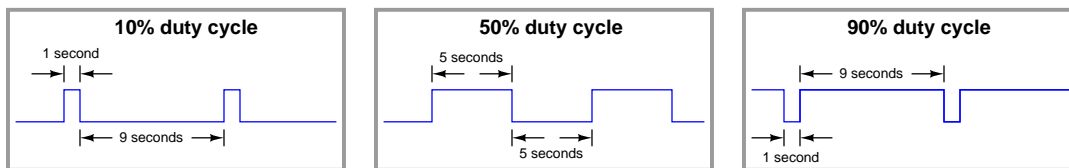
Suppose you need to connect an Allen-Bradley CompactLogix PLC discrete output card (part 1769-OB16) channel to the “Start” discrete input of an Allen-Bradley PowerFlex 4 VFD. Identify which terminals on the PLC need to be connected to which control terminals on the VFD, assuming the use of channel #3 on the PLC output card.

[file i01875](#)

Question 26

Programming Challenge and Comparison – HMI-driven PWM duty cycle control

Suppose we wish to use a PLC to control the average amount of electrical power delivered to an oven’s heating element. The simplest way to implement this control is to have the PLC output a pulsing discrete signal to a solid-state relay (SSR) which then switches AC power to the heating element, the “duty cycle” of that pulsing being adjustable between 0% and 100%, inclusive.



Write a PLC program providing this pulse-width-modulation (PWM) control, using an HMI screen to provide operators with arbitrary adjustment of the duty cycle. The frequency of this pulsing should be slow: 1 Hz or less.

When your program is complete and tested, capture a screen-shot of it as it appears on your computer, and prepare to present your program solution to the class in a review session for everyone to see and critique. The purpose of this review session is to see multiple solutions to one problem, explore different programming techniques, and gain experience interpreting PLC programs others have written. When presenting your program, prepare to discuss the following points:

- Identify the “tag names” or “nicknames” used within your program to label I/O and other bits in memory
- Follow the sequence of operation in your program, simulating the system in action
- Identify any special or otherwise non-standard instructions used in your program, and explain why you decided to take that approach
- Show the comments placed in your program, to help explain how and why it works
- How you designed the program (i.e. what steps you took to go from a concept to a working program)

Suggestions for Socratic discussion

- What type of timer instruction(s) are best suited for this application?
- Would there be an easy way to build a high limit into this system, so the operators could not increment the duty cycle value greater than 100%?
- How could you make the frequency adjustable from the HMI as well?

[file i03515](#)

Programming Challenge – Blinking alarm light

Suppose we wish to create a high-pressure alarm system for operators to alert them when the pressure inside a process vessel exceeds certain pressure limits. Two normally-open pressure switches connect to this vessel, and to separate inputs on the PLC. The first is a PSH (pressure switch high) set for a trip pressure of 150 PSI, and the second is a PSHH (pressure switch high-high) set for a trip pressure of 220 PSI. If the first pressure switch (150 PSI) activates, we want to make a warning light blink on and off. If the second pressure switch activates (220 PSI), *we want that same warning light to remain on steady*.

Write a PLC program to perform this function, and demonstrate its operation using switches connected to its inputs to simulate the discrete inputs in a real application.

Suggestions for Socratic discussion
--

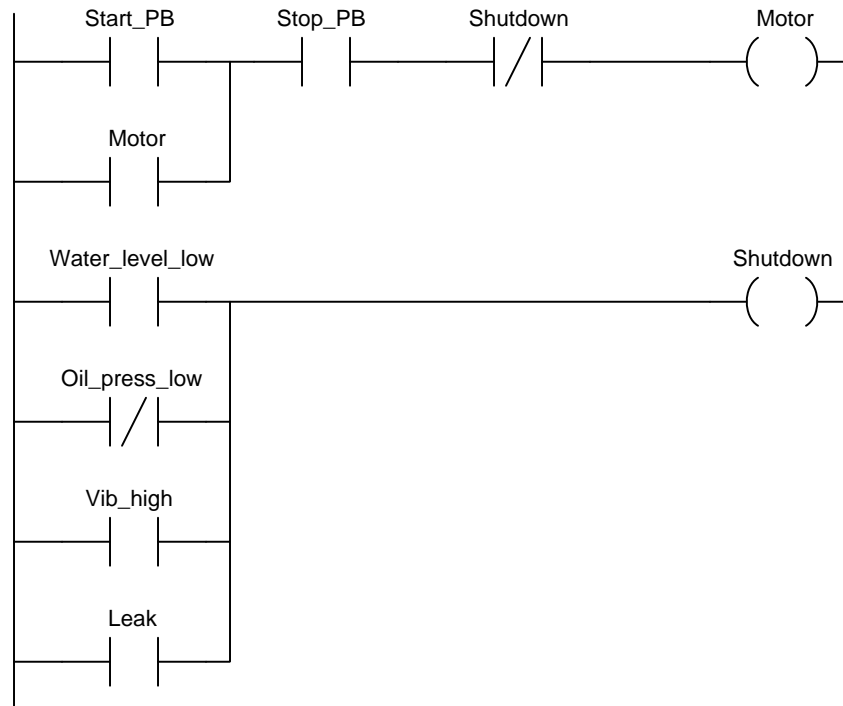
- How do you make the alarm light blink on and off if all the PLC input points are steady (non-pulsing)?
- Can you think of any alternatives to having the same light energize differently to represent two different process alarm conditions?

Question 28

A technician needs to write a PLC program to control a water pump driven by an electric motor. This water pump will be manually started and stopped by pushbutton switches, and shut down automatically by any one of several “permissive” switches. The operating statuses of these switches are listed here:

- **Start pushbutton** (normally-open): open when unpressed, closed when pressed
- **Stop pushbutton** (normally-closed): closed when unpressed, open when pressed
- **Low water level** (normally-closed): closed when level is low, open when level is adequate
- **Low oil pressure** (normally-open): open when pressure is low, closed when pressure is adequate
- **High vibration** (normally-closed): closed when still, open when vibrating
- **Water leak detector** (normally-open): open when dry, closed when wet (leak detected)

The technician’s first attempt is shown here, but it contains a serious error. Identify and correct this error:



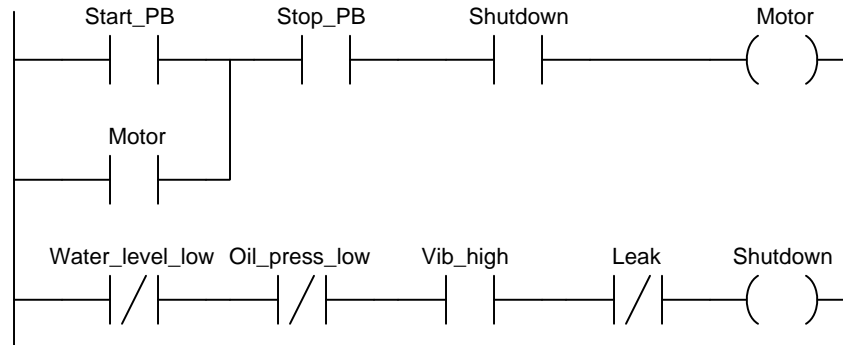
[file i02656](#)

Question 29

A technician needs to write a PLC program to control a water pump driven by an electric motor. This water pump will be manually started and stopped by pushbutton switches, and shut down automatically by any one of several “permissive” switches. The operating statuses of these switches are listed here:

- **Start pushbutton** (normally-open): open when unpressed, closed when pressed
- **Stop pushbutton** (normally-closed): closed when unpressed, open when pressed
- **Low water level** (normally-closed): closed when level is low, open when level is adequate
- **Low oil pressure** (normally-open): open when pressure is low, closed when pressure is adequate
- **High vibration** (normally-closed): closed when still, open when vibrating
- **Water leak detector** (normally-open): open when dry, closed when wet (leak detected)

The technician’s first attempt is shown here, but it contains a serious error. Identify and correct this error:



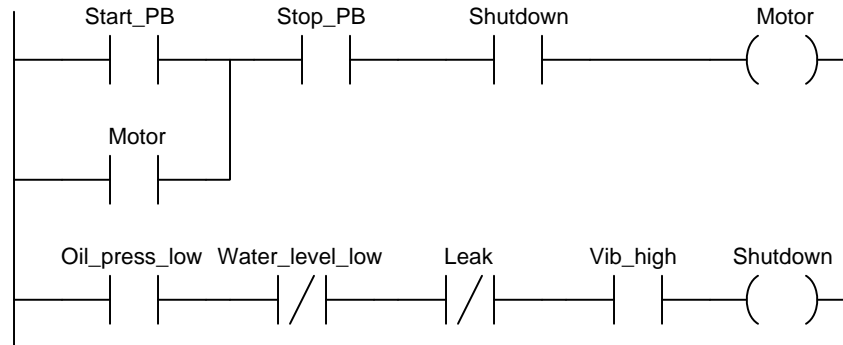
[file i02657](#)

Question 30

A technician needs to write a PLC program to control a water pump driven by an electric motor. This water pump will be manually started and stopped by pushbutton switches, and shut down automatically by any one of several “permissive” switches. The operating statuses of these switches are listed here:

- **Start pushbutton** (normally-open): open when unpressed, closed when pressed
- **Stop pushbutton** (normally-open): open when unpressed, closed when pressed
- **Low water level** (normally-closed): closed when level is low, open when level is adequate
- **Low oil pressure** (normally-open): open when pressure is low, closed when pressure is adequate
- **High vibration** (normally-closed): closed when still, open when vibrating
- **Water leak detector** (normally-open): open when dry, closed when wet (leak detected)

The technician’s first attempt is shown here, but it contains a serious error. Identify and correct this error:



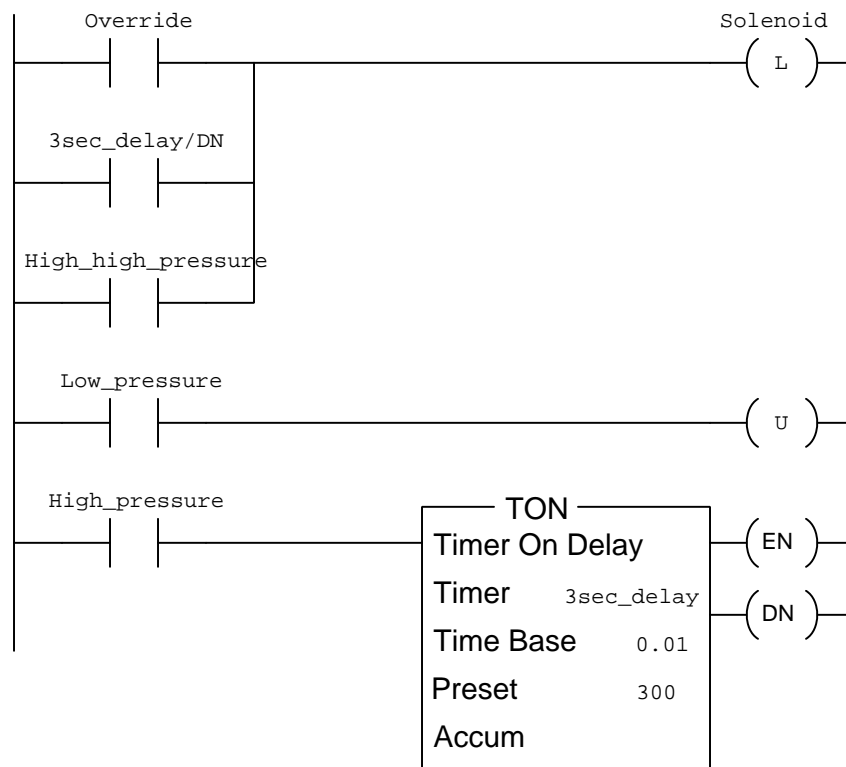
[file i02658](#)

Question 31

A technician needs to write a program for an Allen-Bradley CompactLogix PLC to control a pressure-relieving solenoid valve in a gas processing system. A pair of high-pressure control switches signals the PLC when to open the solenoid valve: one telling the PLC to open the valve after a 3-second time delay and the other (called the “high-high” switch, with a higher trip setting) telling the PLC to open the valve immediately. A pushbutton switch serves as a manual override to open the solenoid valve immediately when pressed. In all cases, the solenoid vent valve will remain open (energized) until pressure falls below the setting of a low-pressure gas switch. The operating statuses of these switches are listed here:

- **Override pushbutton** (normally-open): open when unpressed, closed when pressed
- **Low gas pressure** (normally-closed): closed when pressure is less than 10 PSI, open when pressure exceeds 10 PSI
- **High gas pressure** (normally-closed): closed when pressure is less than 30 PSI, open when pressure exceeds 30 PSI
- **High-high gas pressure** (normally-open): open when pressure is less than 40 PSI, closed when pressure exceeds 40 PSI

The technician’s first attempt is shown here, but it contains a serious error. Identify and correct this error:



[file i02853](#)

Question 32

Question 33

Question 34

Question 35

Question 36

Question 37

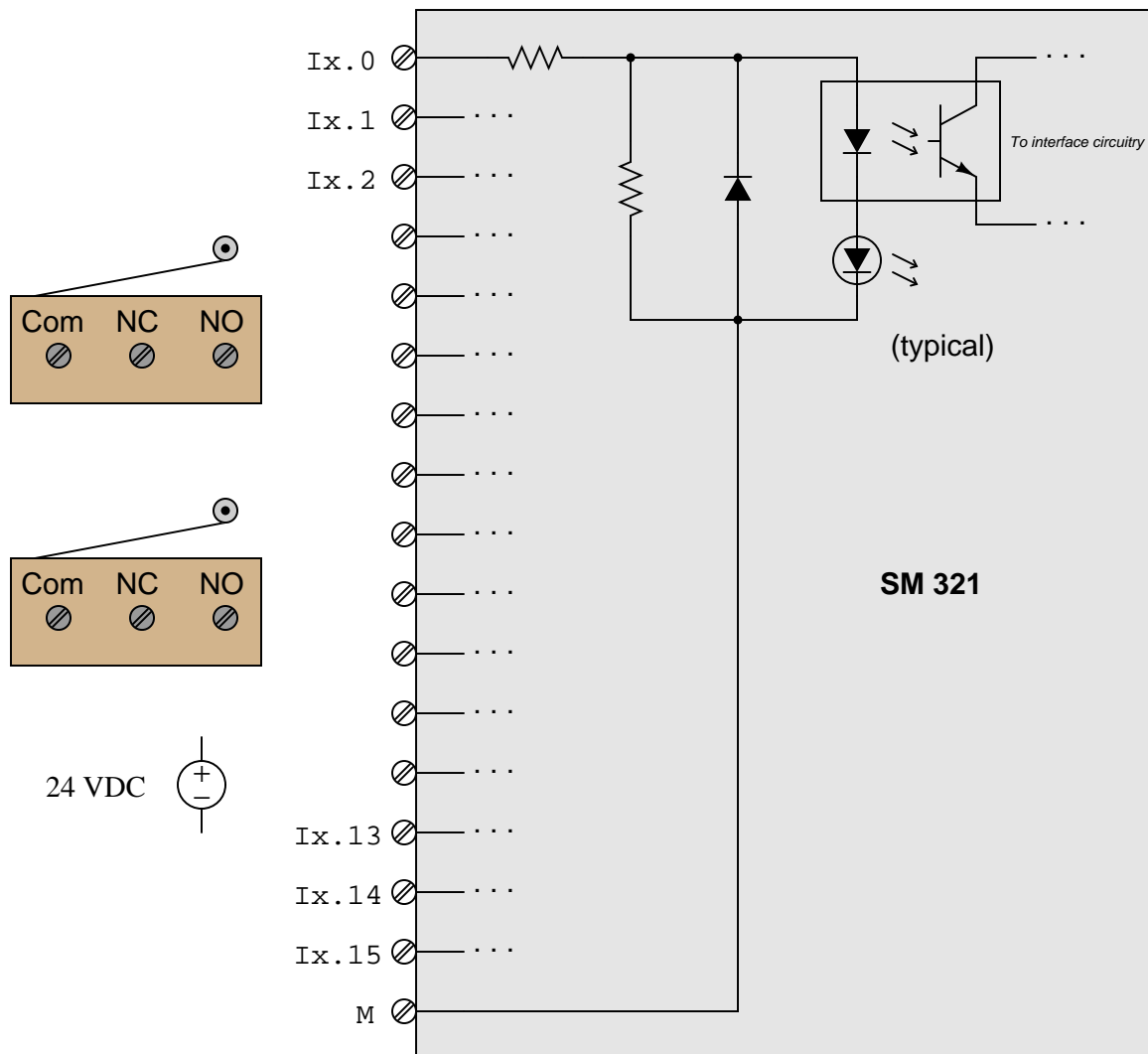
Question 38

Question 39

Question 40

Question 41

Sketch the wires necessary to connect two limit switches (normally-open contacts) to input channels **Ix.3** and **Ix.6** of a Siemens SM 321 discrete input card (model 6ES7321-1BL00-0AA0). The internal schematic diagram of the first channel (**Ix.0**) is shown as “typical” for all the channels:



Also, identify whether this is a *sinking* or a *sourcing* input module.

Suggestions for Socratic discussion

- If you have identified this module as sourcing, explain how its design would differ to make it *sinking*. If you have identified this module as sinking, explain how its design would differ to make it *sourcing*.
- Explain how this module’s internal circuitry could be modified to allow it to source *or* sink current, instead of doing just one of these functions.

[file i04536](#)

Question 42

Read selected portions of the Siemens “SIMATIC S7-200 Programmable Controller System Manual” (document A5E00307987-04, August 2008) and answer the following questions:

Locate the section discussing the PLC’s available *math instructions* and identify several of them.

Is there a “generic” math instruction capable of evaluating a typed expression (i.e. an instruction box where you can enter your own arbitrary formula to obtain an answer, like an Excel spreadsheet)?

What is the difference between a *double integer* and a normal integer in this PLC?

How is it possible to calculate roots other than square roots, when the only “root” instruction available in the set is “square root”?

Are *floating-point* numbers supported in the Siemens S7-200 series of PLC, or only integer numbers?

A reserved area in the S7-200 PLC’s memory called **SMB1** (Special Memory Byte #1) holds eight bits with significance to math operations. Identify the functions of some of these bits, and describe how their status (either 0 or 1) could be useful in a PLC program using math instructions.

Suggestions for Socratic discussion

- If you have access to your own PLC for experimentation, I urge you to write a simple *demonstration* program in your PLC allowing you to explore the behavior of these PLC instructions. The program doesn’t have to do anything useful, but merely demonstrate what each instruction does. First, read the appropriate section in your PLC’s manual or instruction reference to identify the proper syntax for that instruction (e.g. which types of data it uses, what address ranges are appropriate), then write the simplest program you can think of to demonstrate that function in isolation. Download this program to your PLC, then run it and observe how it functions “live” by noting the color highlighting in your editing program’s display and/or the numerical values manipulated by each instruction. After “playing” with your demonstration program and observing its behavior, write comments for each rung of your program explaining in your own words what each instruction does.
- The mathematical technique suggested by the manual for calculating roots other than two can utterly fail in certain cases. Identify at least one of those cases!
- The Siemens S7-200 PLC has the ability to divide two integer numbers and express both the quotient and a *remainder*. Explain how these two portions of the result are stored in the PLC’s memory.
- Apply the mathematical technique suggested for the S7-200 PLC to demonstrate how to take the cube root of some number you already know the answer to (e.g. $\sqrt[3]{64}$).

[file i02343](#)

Question 43

Read selected portions of the Allen-Bradley “MicroLogix 1000 Programmable Controllers (Bulletin 1761 Controllers)” user manual (document 1761-6.3, July 1998) and answer the following questions:

Locate the section discussing the PLC’s available *math instructions* and identify several of them.

Is there a “generic” math instruction capable of evaluating a typed expression (i.e. an instruction box where you can enter your own arbitrary formula to obtain an answer, like an Excel spreadsheet)?

What is the difference between a *double integer* and a normal integer in this PLC?

Are *floating-point* numbers supported in the MicroLogix 1000 series of PLC, or only integer numbers?

A reserved area in the MicroLogix PLC’s memory called *status* designates several bits with significance to math operations. Identify the functions of some of these bits, and describe how their status (either 0 or 1) could be useful in a PLC program using math instructions.

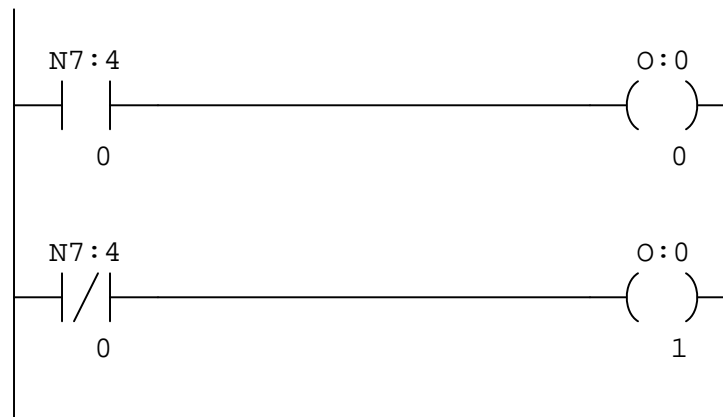
Suggestions for Socratic discussion

- If you have access to your own PLC for experimentation, I urge you to write a simple *demonstration* program in your PLC allowing you to explore the behavior of these PLC instructions. The program doesn’t have to do anything useful, but merely demonstrate what each instruction does. First, read the appropriate section in your PLC’s manual or instruction reference to identify the proper syntax for that instruction (e.g. which types of data it uses, what address ranges are appropriate), then write the simplest program you can think of to demonstrate that function in isolation. Download this program to your PLC, then run it and observe how it functions “live” by noting the color highlighting in your editing program’s display and/or the numerical values manipulated by each instruction. After “playing” with your demonstration program and observing its behavior, write comments for each rung of your program explaining in your own words what each instruction does.
- Note the *execution times* listed for each instruction in the MicroLogix 1000 manual. Why do you suppose this information might be important to you as a programmer or an end-user of a PLC?

[file i02344](#)

Question 44

This Allen-Bradley PLC program detects whether an integer number stored in register N7:4 is odd or even, energizing one of two lights to indicate the result:



Explain how this very simple program detects the oddness or evenness of the integer number, and also determine which output bit activates for the odd result and which activates for the even result.

Could this same programming technique be applied in a Siemens S7-200 PLC, say to determine whether the integer number stored in word VW88 was odd or even?

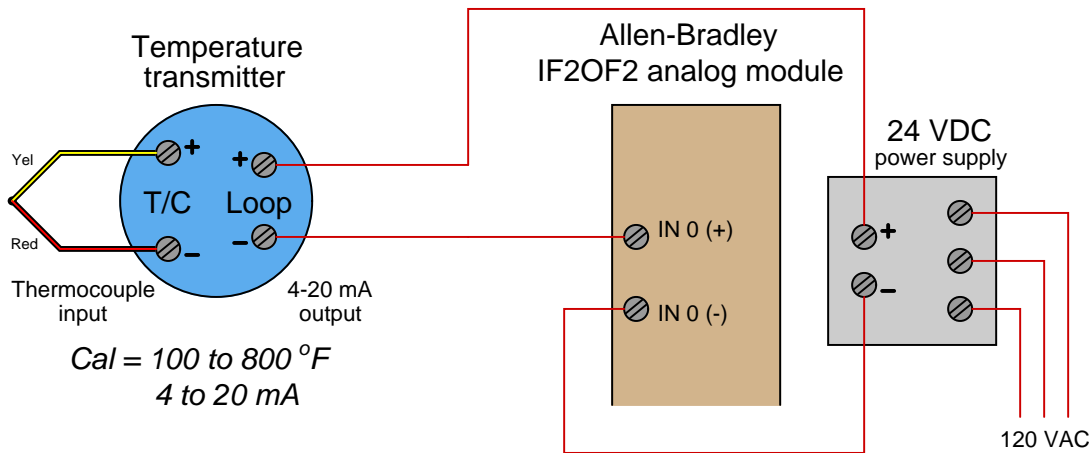
Suggestions for Socratic discussion
--

- Does the Koyo CLICK PLC offer similar bit-wise addressing capability, which could be used in the same way?

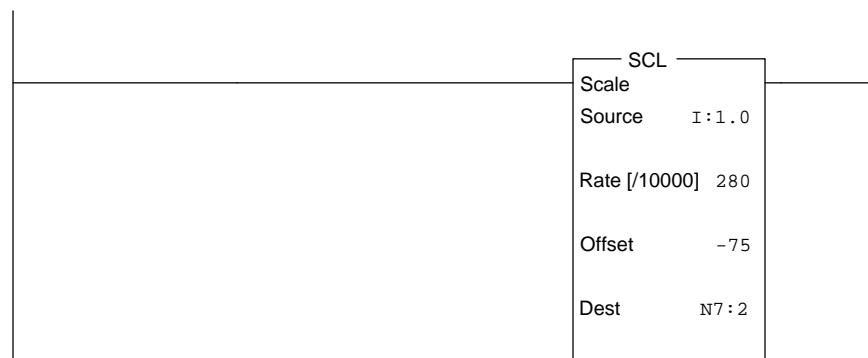
[file i02387](#)

Question 45

An Allen-Bradley MicroLogix 1100 PLC senses a 4-20 mA analog signal from a temperature transmitter using an analog input module (model IF2OF2). At the lower end of signal scale (100 degrees Fahrenheit = 4 mA DC signal), the input register value in the PLC (the analog-to-digital converter's “count” value) is 6240. At the high end of signal scale (800 degrees Fahrenheit = 20 mA DC signal), the input register value in the PLC is 31200:



A *scale* (SCL) instruction in the PLC's program is used to convert this raw analog-to-digital “count” value into units of degrees Fahrenheit in the PLC, storing the result in register N7:2 as an integer number:



Examine the parameters entered into this SCL instruction, and then calculate the actual integer value written to N7:2 at a temperature of 100 degrees F (4 mA signal) and the actual integer value written to N7:2 at 800 degrees F (20 mA signal). The results will *not* be exact due to rounding of the Rate and Offset parameters.

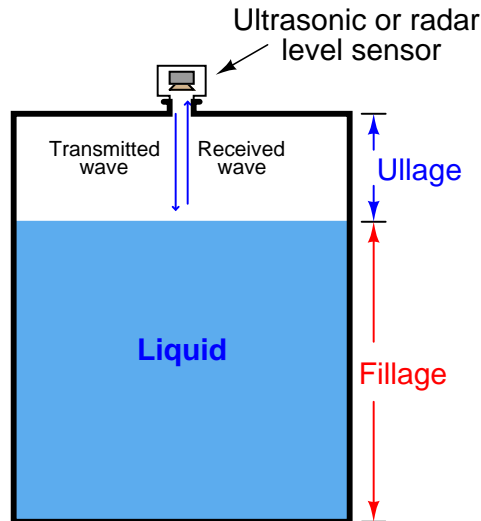
Suggestions for Socratic discussion

- The particular scaling chosen here is not the best for a realistic application, using an integer number to represent a temperature between 100 and 800 degrees with a resolution of ± 1 degree. Explain how we could represent a temperature range of 100.0 to 800.0 degrees instead.
- For those who have studied thermocouple types, identify the type of thermocouple used in this system.
- Identify the circumstance(s) under which the scale instruction (SCL) will generate an *overflow* condition in an Allen-Bradley PLC.

Question 46

Programming Challenge – Fillage/ullage calculator

Ultrasonic- and radar-based liquid level sensing instruments where the sensor is located on the top of a storage vessel and waves are sent down to the liquid level and then reflected back naturally measure the “air space” above the liquid. The technical term for this measurement is *ullage*, representing the empty space of the storage vessel:



However, operations personnel are often more interested in the *fillage* of a vessel (how full it is) rather than its ullage. Think of it as the classic question of whether the glass is half-full or half-empty, with an industrial flavor.

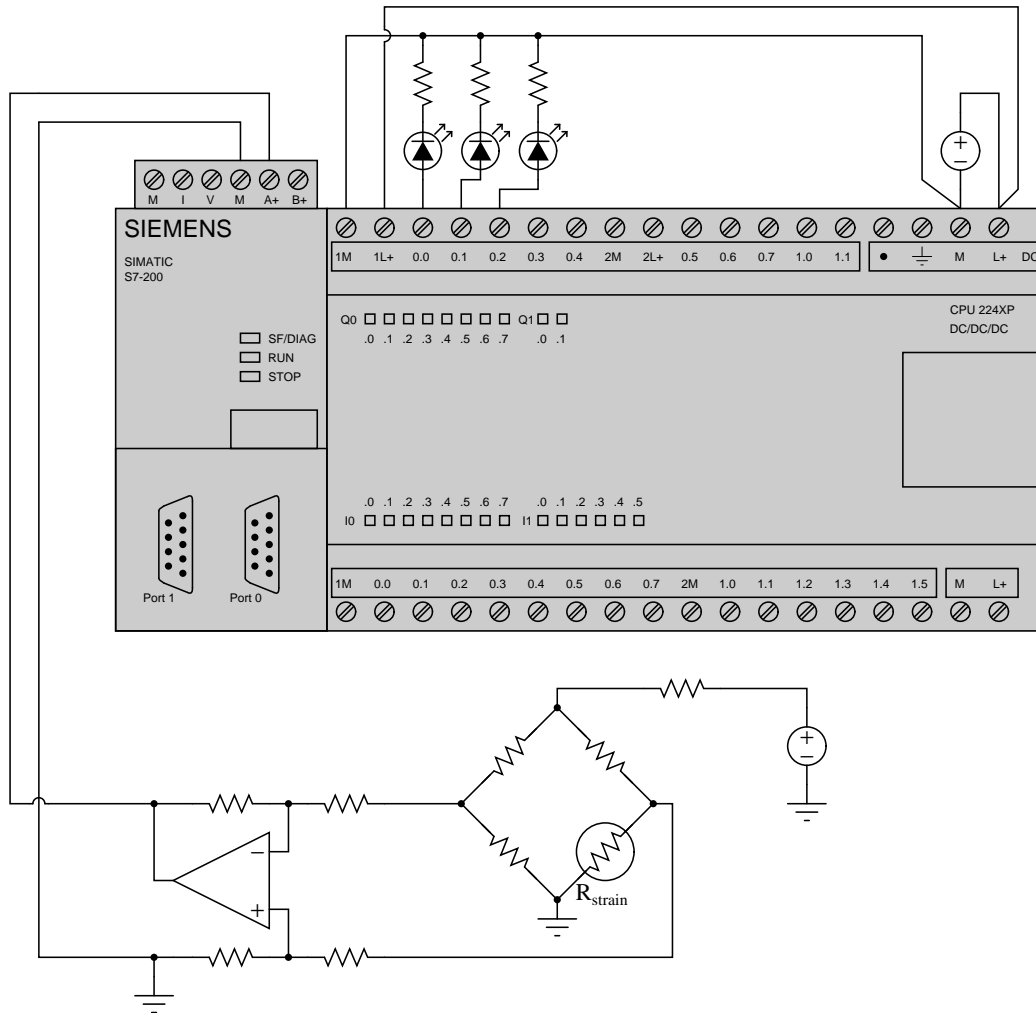
Write a PLC program to take the ullage value of an ultrasonic level sensor and convert this into a fillage value for a vessel, given a fixed (total) height for the vessel. Use an HMI with a “bargraph” indicator as a tank level (fillage) display. Since you probably do not have a level transmitter readily available to connect to your PLC for this exercise, feel free to simulate one by using a pair of discrete inputs to increment and decrement an up/down counter, generating a variable simulated value for the level transmitter. If your PLC happens to have an analog input channel, feel free to input a variable voltage signal to simulate the scale’s reading instead!

Suggestions for Socratic discussion

- For those who have studied level measurement technologies, what other liquid level-sensing technologies naturally sense ullage besides radar and ultrasonic?
- For those who have studied level measurement technologies, describe the difference between *guided-wave* radar level sensors and *unguided* radar level sensors.
- Determine how it is possible to format a vertical bargraph on your HMI display so that it looks like a filling tank (a very wide bargraph!), and link that bargraph’s tag name to the fillage variable in your PLC.

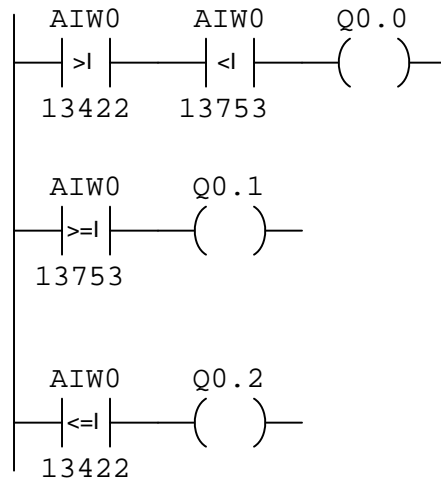
Question 47

A manufacturing facility uses an electronic scale to weigh batches of material in a packaging process. The scale weighs each batch about to be packaged, and determines whether the batch is too light, too heavy, or within tolerable limits. Three discrete output bits serve as the indicators of these statuses: one to energize when the batch is too light, one to energize when it's too heavy, and the third to energize when the batch weight is correct. The wiring for this system is shown here, with the bridge and differential amplifier circuit calibrated for an 8 volt signal output (to the PLC) at 1500 lbs scale weight, and a 0 volt signal at 0 lbs scale weight:



The analog inputs on the S7-200 PLC are scaled for 0 to 10 volts = 0 to 32000 counts.

Analyze this offline program display for the S7-200 PLC, explaining how the ladder-logic functions and also determining the weight limits for the three output bits:



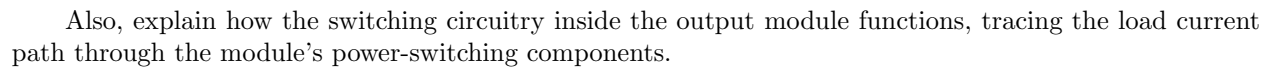
Finally, determine which bit represents “too light,” which bit represents “too heavy,” and which bit represents “correct” for weight.

Suggestions for Socratic discussion
--

- Explain how the PLC will interpret a failed-open strain gauge.

[file i03516](#)

Sketch the wires necessary to connect a solenoid and a relay (both devices having coils rated for 120 VAC) to output channels Qx.2 and Qx.4 of a Siemens SM 322 discrete output card (model 6ES7322-1FH00-0AA0), respectively. The internal schematic diagram of the first channel (Qx.0) is shown as “typical” for all the channels, revealing how TRIACs are used to switch 120 VAC power for each discrete output channel. Include any necessary power sources in your sketch to make the circuits functional:



file i04246

Programming Challenge and Comparison – Positive displacement flowmeter rate

A common design of flowmeter for residential water flow measurement is the *positive displacement* design, where the movement of water volume through the meter causes a mechanism to rotate, passing a known and fixed quantity of water volume through the meter for each revolution. The rotation of the flowmeter mechanism may be electrically transmitted by a magnetic reed switch actuated by a magnet on the flowmeter mechanism's rotating shaft. Actuating (closed and opened) one cycle per revolution, the reed switch produces a pulse signal representing a known and fixed measurement of water volume per switch "pulse."

Write a PLC program continuously calculating the flow rate of water through such a meter, given a meter factor of 1 gallon per switch pulse. The calculated flowrate needs to be displayed on an HMI, units of "GPM" (gallons per minute).

When your program is complete and tested, capture a screen-shot of it as it appears on your computer, and prepare to present your program solution to the class in a review session for everyone to see and critique. The purpose of this review session is to see multiple solutions to one problem, explore different programming techniques, and gain experience interpreting PLC programs others have written. When presenting your program (either individually or as a team), prepare to discuss the following points:

- Identify the "tag names" or "nicknames" used within your program to label I/O and other bits in memory
- Follow the sequence of operation in your program, simulating the system in action
- Identify any special or otherwise non-standard instructions used in your program, and explain why you decided to take that approach
- Show the comments placed in your program, to help explain how and why it works
- How you designed the program (i.e. what steps you took to go from a concept to a working program)

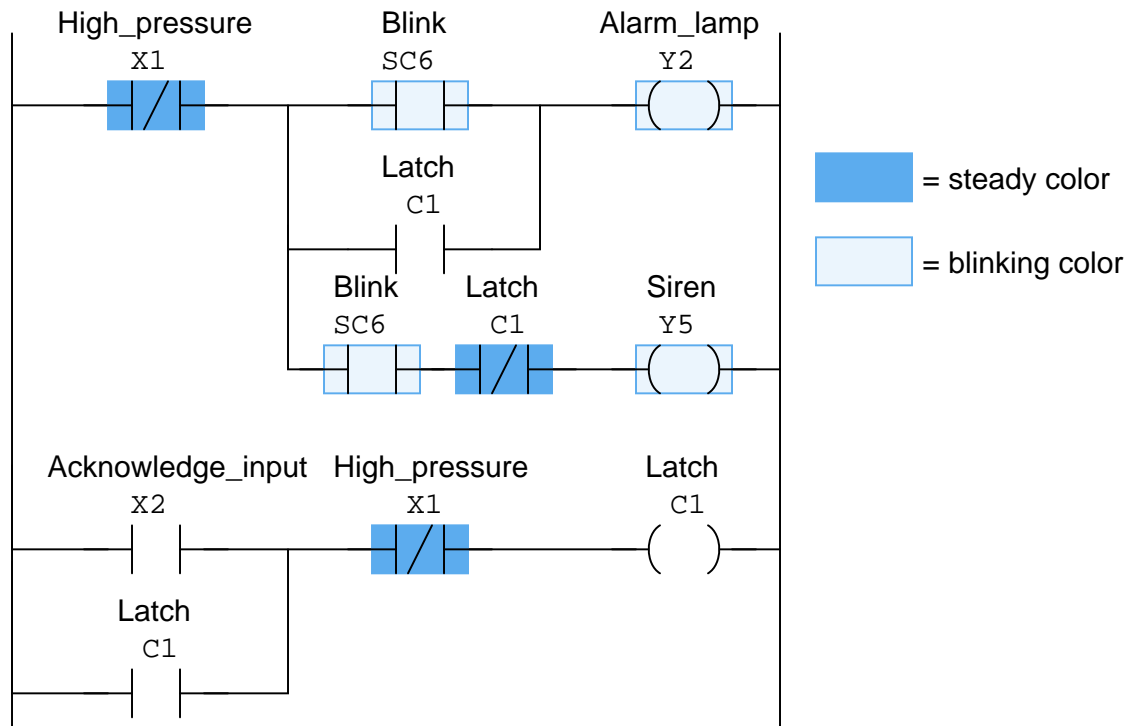
Suggestions for Socratic discussion
--

- How can you write your program to update the flow calculation more often than once per minute?
- One way to calculate flow is to count the number of gallons passed in one minute of time. Is there a way to calculate inversely: determining flow rate by measuring the amount of time elapsed between pulses?

[file i02486](#)

Question 50

A PLC-controlled alarm annunciator system has a problem, and you are called to diagnose what that problem is. The alarm lamp and siren are supposed to both “pulse” when a high-pressure condition is detected by a pressure switch, and then the siren is supposed to be silenced when the “Acknowledge” pushbutton is pressed. However, the siren continues to pulse (and so does the alarm lamp) despite repeated attempts by the operators to press the “Acknowledge” switch. By the time they call you, they have reached a point of formidable anger due to the incessant pulsing of the siren:



You have the ability to remotely monitor the PLC program (“online”) from the maintenance shop over a network cable, allowing you to view the working program without ever getting near the angry operators. What you see on your laptop computer screen is shown above.

Identify a likely cause for the inability of the operators to acknowledge this alarm. Also, identify a means by which you could silence the irritating noise during the time it takes you to further diagnose the problem and make repairs.

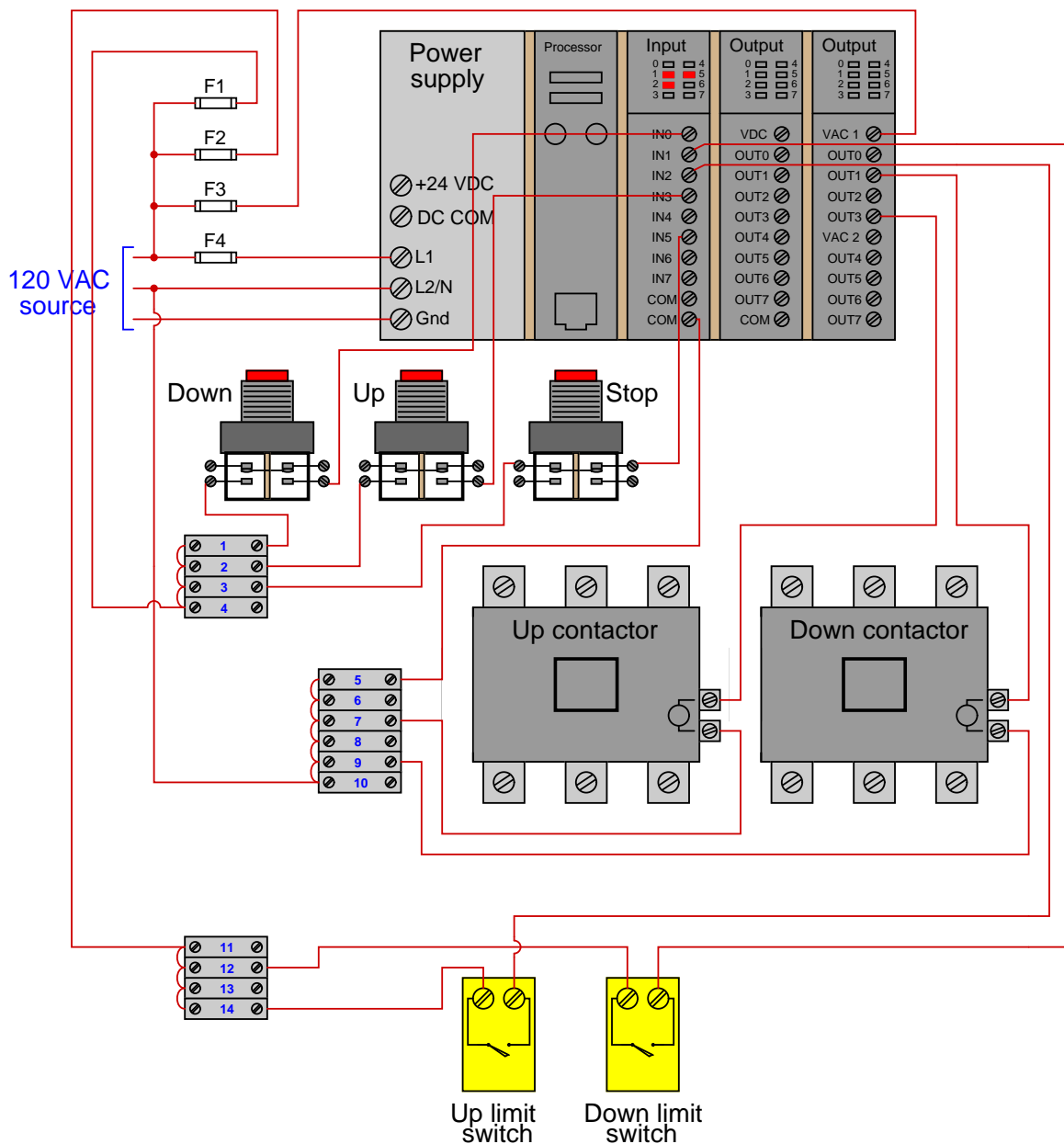
Suggestions for Socratic discussion

- Based on the program you see here, do you think the high-pressure switch contacts are wired NO or NC?
- Based on the program you see here, do you think the “Acknowledge” switch contacts are wired NO or NC?

[file i02537](#)

Question 51

This elevator control system has a problem. No matter which pushbutton is pressed, the elevator remains “stuck” in the full-down position. The following pictorial diagram shows the wiring of this system, along with the I/O card status lights as they appear with no one pressing any pushbuttons:



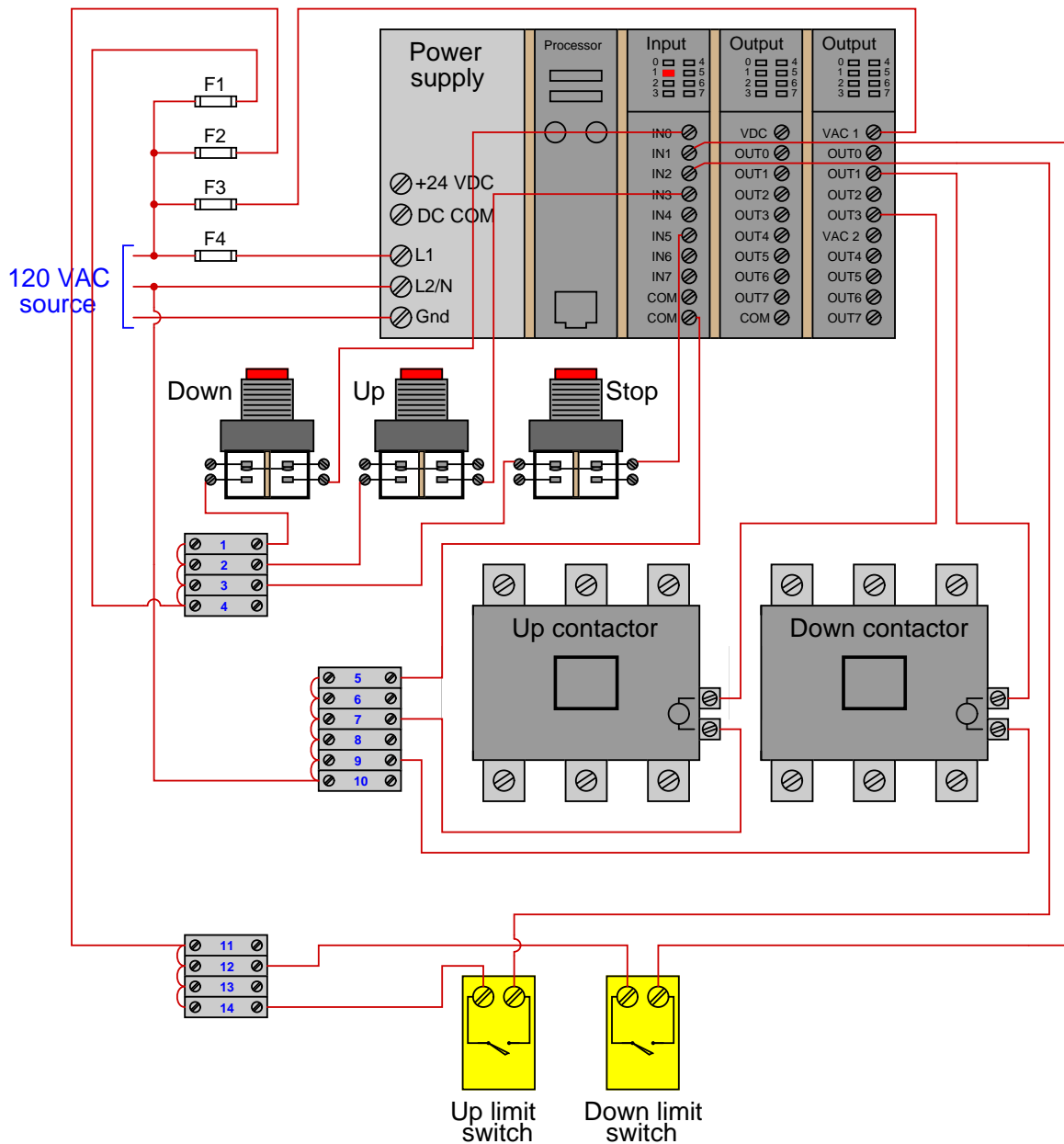
Pressing the “Down” and “Up” pushbuttons, you notice the LEDs for input channels 0 and 3 light up, respectively. No LEDs on the output card light up at any time.

Based on this information, determine a likely fault causing this elevator to remain stuck at the full-down position.

[file i02538](#)

Question 52

This elevator control system has a problem. No matter which pushbutton is pressed, the elevator remains “stuck” in the full-down position. The following pictorial diagram shows the wiring of this system, along with the I/O card status lights as they appear with no one pressing any pushbuttons:

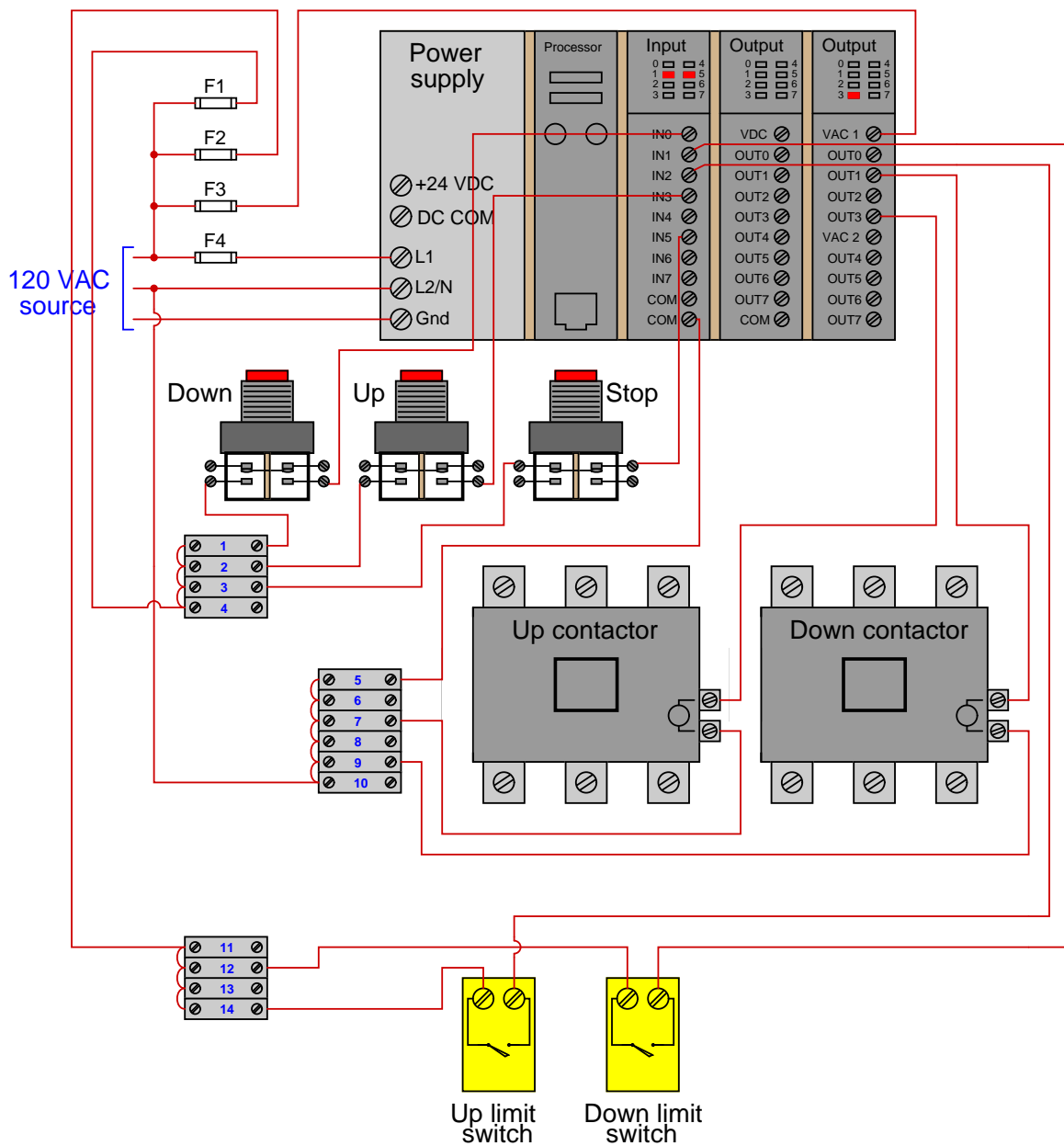


Based on this information, determine a likely fault causing this elevator to remain stuck at the full-down position.

file i02539

Question 53

This elevator control system has a problem. No matter which pushbutton is pressed, the elevator remains “stuck” in the full-down position. The following pictorial diagram shows the wiring of this system, along with the I/O card status lights as they appear with no one pressing any pushbuttons:



Pressing the “Down” and “Up” pushbuttons, you notice the LEDs for input channels 0 and 3 light up, respectively.

Based on this information, determine a likely fault causing this elevator to remain stuck at the full-down position.

[file i02544](#)

Question 54

Question 55

Question 56

Question 57

Question 58

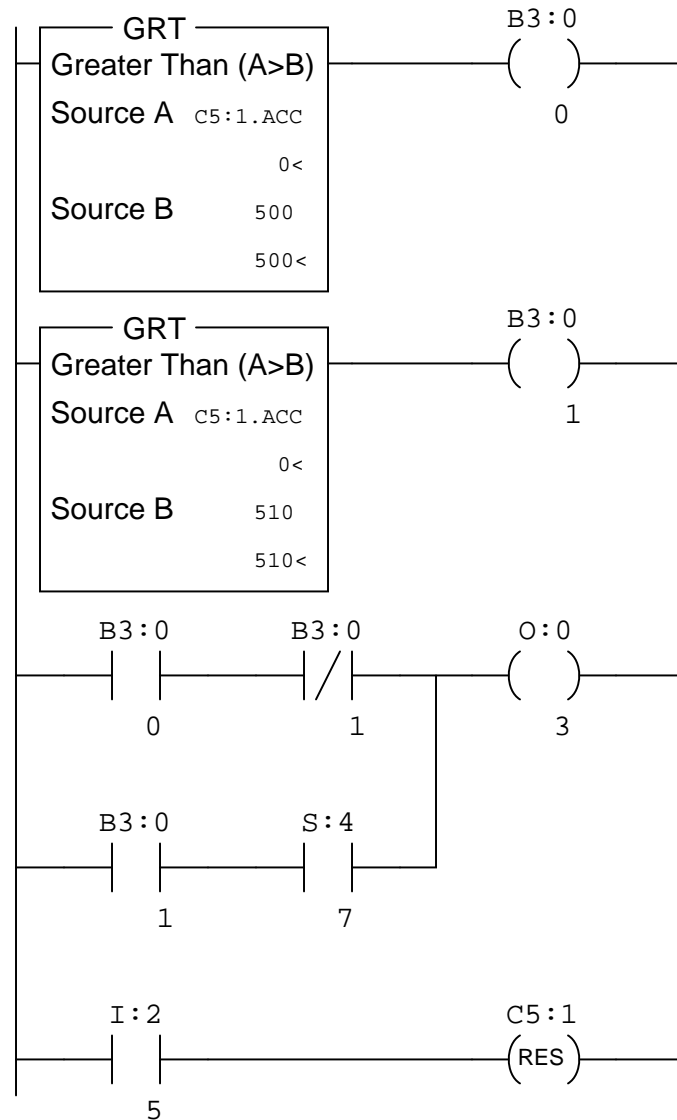
Question 59

Question 60

Question 61

Suppose we have an Allen-Bradley MicroLogix 1000 PLC controlling the starting and stopping of an air compressor. A timer instruction tracks the total accumulated run-time of the compressor, incrementing a counter (C5:1) for each hour of run-time passed. Thus, the accumulator value of counter C5:1 records the total number of hours the compressor has run.

Additional ladder logic code operates on this counter's accumulator value to energize a maintenance warning light. Note that the status contact S:4/7 is a free-running clock bit with a 1.28 second period:



Based on your examination of this program, determine what the maintenance lamp (connected to output O:0/3) should be doing when the compressor's run time reaches 505 hours.

Suggestions for Socratic discussion

- The ladder logic code incrementing counter C5:1 every hour is not shown here. Describe what this code might look like.

- Suppose the normally-open B3:0/0 contact instruction were changed to be normally-closed. How would this affect the operation of this program?

[file i04538](#)

Question 62

Read and outline the introduction and the “Modbus Overview” subsection of the “Modbus” section of the “Digital Data Acquisition and Networks” chapter in your *Lessons In Industrial Instrumentation* textbook.

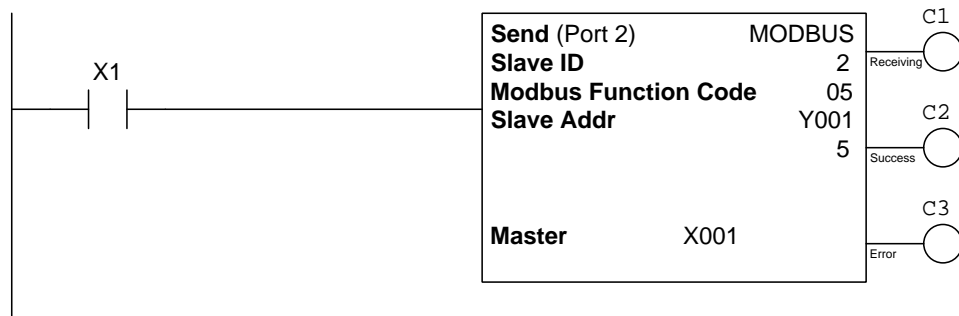
The purpose of your outline is to foster close reading of the text, to facilitate quick referencing of specific points within the text, to record questions of your own, and to practice clear writing. Your outline must meet the following standards for full credit: *every major idea contained in the text represented in your outline, entirely in your own words (i.e. no copying of text), written in a legible and comprehensible manner, of sufficient quality that others would find it informative.* Incomplete, illegible, cryptic, and/or plagiarized outlines will not receive full credit. A suggestion is one sentence of your own per paragraph of source text. Helpful additions include:

- Noting questions or points of confusion you have from the reading
- Page numbers from the source text for quick reference during discussion
- Images copied from the text (or sketched by you) to illustrate concepts
- References to previously learned concepts

[file i04467](#)

Question 63

Suppose a technician wants to have a toggle switch connected to input X1 of one PLC (a Koyo “CLICK” model) directly control output Y1 of another “CLICK” PLC, and writes this program to do it:



Unfortunately, the program does not function as intended. When the X1 toggle switch is turned on, output Y1 on the other PLC energizes as it should. However, when the X1 toggle switch is turned off, the other PLC's Y1 output remains energized.

Explain what is wrong with this program, and how to correct it. Note that this is a very common misconception for people new to communication instructions in PLC programs!

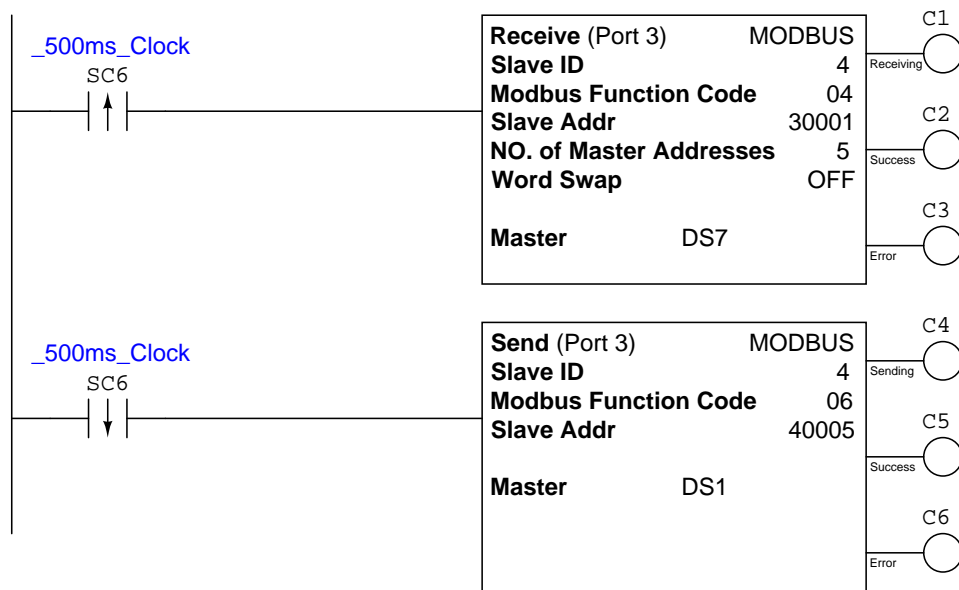
Suggestions for Socratic discussion

- Modify this program so that it will function as intended, sending the true bit state of X1 to the receiving PLC regardless of whether it is a “1” or a “0”

[file i03376](#)

Question 64

This Koyo CLICK PLC program reads data from and writes data to a Modbus device connected to the RS-485 communications port (Port 3):



Explain how this program functions, especially the use of the rising- and falling-edge SC6 contacts. How many integer registers being communicated between the CLICK PLC and the Modbus device? What type(s) of data are being read and written to the Modbus device?

It is important to note that if the Modbus device being communicated with is another PLC, that other PLC does *not* have to run complementary communication instructions in its ladder logic program (e.g. a “Send” instruction to match the first PLC’s “Receive” instruction, and so forth). The instructions you see in the first PLC’s program are self-sufficient – the other PLC is able to answer Modbus queries and commands without any need for additional ladder logic programming. Examine the “Operating Cycle” or “Scan Cycle” in the instruction manual for a typical PLC, and identify when in the cycle you think Modbus communications take place if not in the ladder logic program.

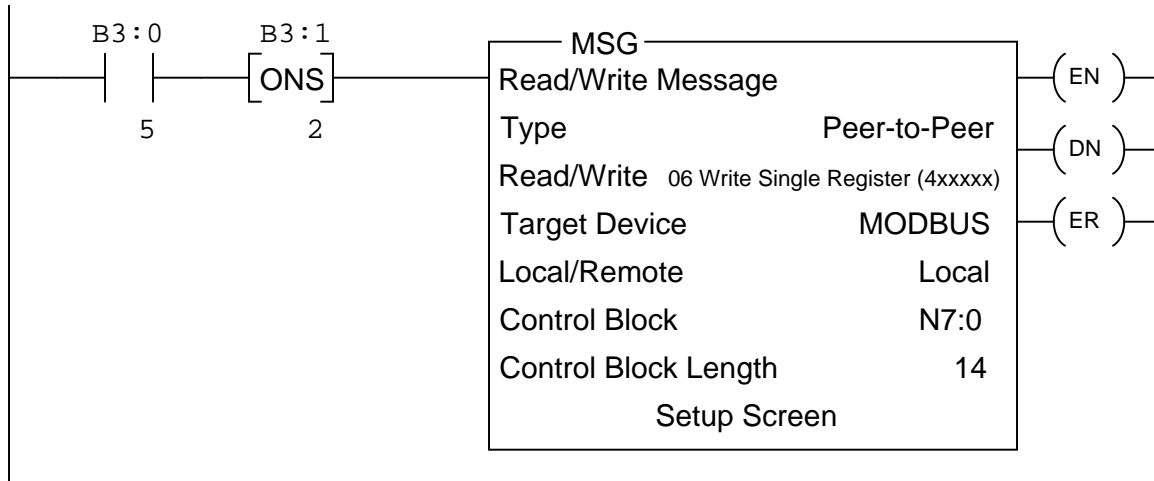
Suggestions for Socratic discussion

- What would happen if each Modbus communication instruction continually received “power” from the left-hand rail without waiting on the SC6 contact instructions?
- Identify some practical uses for the data stored in the C bits signifying “Sending,” “Receiving,” “Success,” and “Error.”

[file i03858](#)

Question 65

This Allen-Bradley SLC 500 PLC program writes data to a variable-frequency drive using Modbus protocol. The PLC receives data from an HMI panel in the form of an integer value and a momentary “ENTER pushbutton” bit signal (writing to bit B3:0/5 in the PLC). It is important to note that MicroLogix programs such as this one require a “one-shot” to drive any communication instructions such as **MSG**. If a **MSG** instruction is continually activated in a ladder-logic program it experiences trouble completing its task, as though each fresh scan of the program interrupts the instruction’s action begun on the previous scan:



Explain how this program functions, especially how the “one-shot” (ONS) instruction prevents the **MSG** instruction from “talking over itself”. How many integer registers are used in the SLC 500 PLC’s memory for storing data relevant to the **MSG** instruction? What type(s) of data are being written to the VFD?

Presently this program sends a Modbus message to another device once for every positive transition of the B3:0/5 bit. Suppose you wished to alter this program so that it transmits a Modbus message regularly every half-second or so. How could you edit the program to do this?

A lot of details seem to be missing from this instruction as it appears in the RSLogix 500 software display. Identify some of the pertinent parameters we do *not* see in this **MSG** instruction, and where the human programmer could navigate to in the RSLogix software to view them.

[file i04531](#)

Demonstration Program – communication instructions

An important technique for learning any programming language – Ladder Diagram PLC programming included – is to write simple “demonstration” programs showcasing and explaining how particular instructions and programming constructs are supposed to work. Since you have access to your own personal PLC, you can explore the elements of your PLC’s programming language like a scientist would explore new specimens: subject them to tests and record how they respond. This is how you will be able to teach yourself new models of PLC when you are working in your career, when you won’t have textbooks to follow or training to show you exactly what to do.

Write a minimal “demonstration” program for one of your PLC’s *communication* instructions, where a discrete input on your PLC controls a discrete output on a different (networked) PLC. An acceptable demonstration program must meet these three criteria:

- **Simple** – nothing “extra” included in the program to detract from the fundamental behavior of the instruction(s) being explored.
- **Complete** – nothing missing from the program relevant to the fundamental behavior of the instruction(s) being explored.
- **Clearly documented** – every rung clearly commented in your own words, every variable named.

Note: a common mistake when writing demonstration programs is to try to make them fit some practical application, like an electric motor start/stop control. The reason why this is a bad idea for a demonstration program is because single applications typically use a limited range of each instruction’s features. The purpose of a demonstration program is to serve as an experimental “laboratory” for testing any and all features of the instruction(s), not just to showcase some of the features. You will have plenty of later opportunities to write PLC programs for practical applications!

At minimum your program must demonstrate the following:

- How to configure the communication instruction to either send or receive data.
- How the two PLCs are addressed on the interconnecting network, if at all.
- How to start and stop communication (i.e. how to enable and disable the communication instruction in ladder logic).
- The status of any error bits associated with the communication instruction.

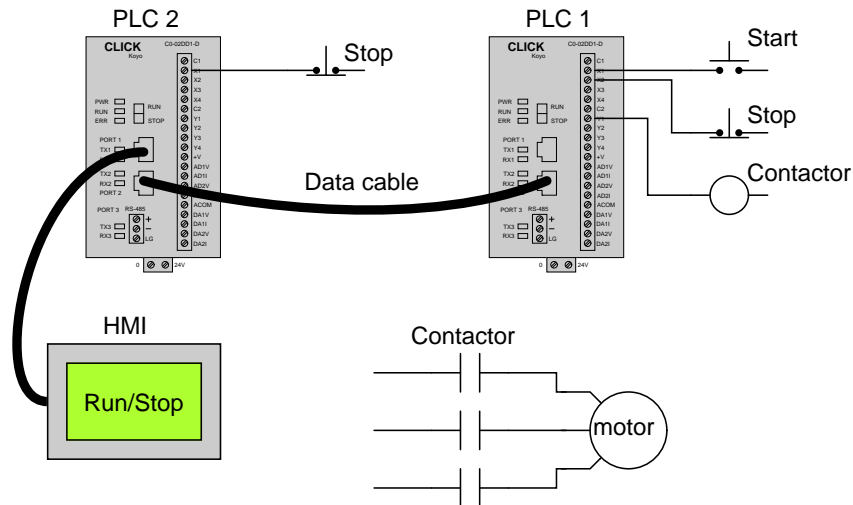
Here are some helpful tips to keep in mind when writing your demonstration program:

- Only one of the two PLCs connected to each other needs to have a communications instruction in it – *the other PLC doesn’t even have to be programmed with any ladder logic at all to have data read from or written to it!*
- Most communication instructions only need to be activated for a single scan in order to perform as intended. In fact, “energizing” a communication instruction every program scan is often a recipe for trouble!
- Make sure any memory locations being written to by a communication instruction are not also being written to by some other element of the PLC (e.g. an input bit being controlled by electrical inputs, a bit being written to by a coil in the program, etc.).

[file i03122](#)

Programming Challenge and Comparison – Remote PLC “stop” button for motor control system

Suppose we have an application where two PLCs are connected via a network cable. PLC 1 directly connects to the momentary-contact “Start” pushbutton, “Stop” pushbutton, and contactor coil for a simple motor control system. PLC 2 has its own “Stop” pushbutton connected, which is supposed to cause the motor at the first PLC to stop when pressed. An HMI connected to PLC 2 reads the status of the motor (whether it is running or stopped):



Work individually or in teams to write a PLC program using network communication instructions to perform the functions of remote stop and remote viewing of motor status. *Note: some PLC models are unable to act as network “master” devices, such as the Allen-Bradley MicroLogix 1000 series A and B PLCs.*

When your program is complete and tested, capture a screen-shot of it as it appears on your computer, and prepare to present your program solution to the class in a review session for everyone to see and critique. The purpose of this review session is to see multiple solutions to one problem, explore different programming techniques, and gain experience interpreting PLC programs others have written. When presenting your program (either individually or as a team), prepare to discuss the following points:

- Identify the “tag names” or “nicknames” used within your program to label I/O and other bits in memory
- Follow the sequence of operation in your program, simulating the system in action
- Identify any special or otherwise non-standard instructions used in your program, and explain why you decided to take that approach
- Show the comments placed in your program, to help explain how and why it works
- How you designed the program (i.e. what steps you took to go from a concept to a working program)

Suggestions for Socratic discussion

- Why can’t communication instructions be continuously “energized” in PLC programs, but rather must be only occasionally activated (i.e. a series contact closing for one scan every so often)?
- Does it matter which PLC is the master and which is the slave in this application?

[file i02492](#)

Question 68

Read Appendix C of the Allen-Bradley “PowerFlex 4 Adjustable Frequency AC Drive user manual” (document FRN 5.xx), and answer the following questions:

Describe what a VFD (Variable Frequency Drive) is useful for. What, exactly, does it do in a control system?

What does this section have to say about the “+” and “–” wires for Modbus RS-485 devices?

Identify some of the commands for the AC motor drive accessible as individual bits in register 8192.

Identify the register within the AC motor drive holding the frequency “reference” (command) value. This is the numerical value commanding the motor how fast to spin. Is this numerical value specified in integer, fixed-point, or floating-point format?

Suggestions for Socratic discussion

- Based on your reading of this manual, is there any danger in accidentally reversing the Modbus (RS-485) wire connections?
- The wiring diagram on page C-1 shows a 120 ohm termination resistor installed at the cable end. Can you think of any application where you might wish to use a different-value termination resistor on this network cable (i.e. something other than 120 Ω)?
- Page 1-9 of this manual describes a “reflected wave problem” that may manifest on long lengths of motor cable between the drive and the motor. Based on the description and the table of figures shown on that page, what does this problem consist of?
- Identify some of the error codes generated by this VFD which may be read via Modbus (held in register 8449).
- The network wiring diagram shown in figure C.1 shows an interesting method of grounding the shield conductor in each cable. Interpret this diagram, and then elaborate on alternative methods of shield grounding which would also work.

file i04469

Question 69

Read pages 4-46 through 5-8 of the Automation Direct “GS1 Series Drives user manual” (document GS1-M), and answer the following questions:

Describe what a VFD (Variable Frequency Drive) is useful for. What, exactly, does it do in a control system?

Identify the purpose of the “FA-ISONET” device referenced on pages 5-6 and 5-7.

Identify some of the status bits readable in register 48450 (2001 hex).

Identify the register within the AC motor drive holding the speed “reference” (command) value. This is the numerical value commanding the motor how fast to spin. Is this numerical value specified in integer, fixed-point, or floating-point format?

Suggestions for Socratic discussion

- Identify which layer of the OSI model the FA-ISONET device operates on.
- Can the Modbus bit rate of this VFD be arbitrarily set, or is it fixed at one communication speed?
- Identify which register(s) within the VFD you would have to write data into via Modbus in order to command the drive to “Run” and to “Stop”.
- Explain the significance of the speed reference value residing in a register address beginning with “4” within the Modbus addressing scheme.

[file i04470](#)

Question 70

Question 71

Question 72

Question 73

Question 74

Question 75

Question 76

Question 77

Question 78

Question 79

Question 80

Question 81

Suppose a PLC program contains a counter instruction that counts in unsigned 16-bit integer format. An HMI connected to this PLC, however, is configured to read and interpret this counter's accumulated value as a *BCD* number instead of unsigned binary.

If the PLC's counter has an accumulated value of 38199 (decimal), determine how the HMI will incorrectly interpret and display it.

Question 82

Suppose a technician needs to program a PLC to take the raw analog-to-digital “count” value from an analog input card and scale it to a value ranging 0 to 100 (%). The input card’s ADC count range is 0 to 65535. The standard formula for doing this conversion is as follows:

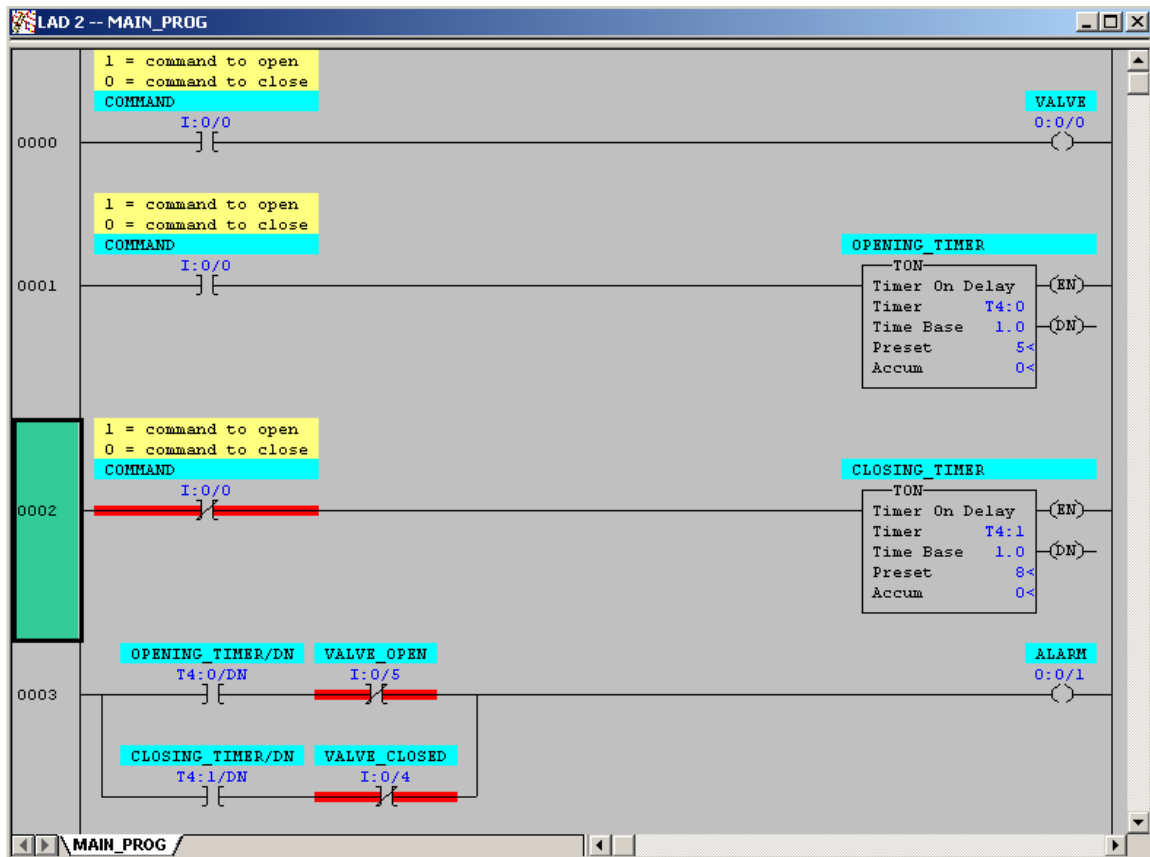
$$\text{Scaled output} = \frac{\text{Raw input}}{65535} \times 100$$

Ideally, this formula entered into a “Math” instruction in the PLC will convert any raw count value from the analog input channel into a 0 to 100% value. However, when the technician tries programming this formula into the PLC’s math instruction, the result is always either 0 or 100 and never any other values. After fruitlessly trying to figure out what is going wrong, a more experienced programmer walks by to observe and comments, “That’s because this PLC’s math instruction only does *integer* calculations.” The first technician is still perplexed, and comes to you for help.

First, explain why the formula does not compute as the technician expects it to. Second, recommend a fix so that the PLC will do a better job of scaling this ADC count value into percent.

Question 83

Examine this ladder logic program for an Allen-Bradley MicroLogix PLC controlling a valve's motion and checking to determine if the valve ever becomes stuck and cannot fully open or fully close:



Based on this program, determine the necessary connections for each limit switch monitoring the valve stem's position:

- VALVE_OPEN limit switch: *normally-open* (NO) or *normally-closed* (NC)?
- VALVE_CLOSED limit switch: *normally-open* (NO) or *normally-closed* (NC)?

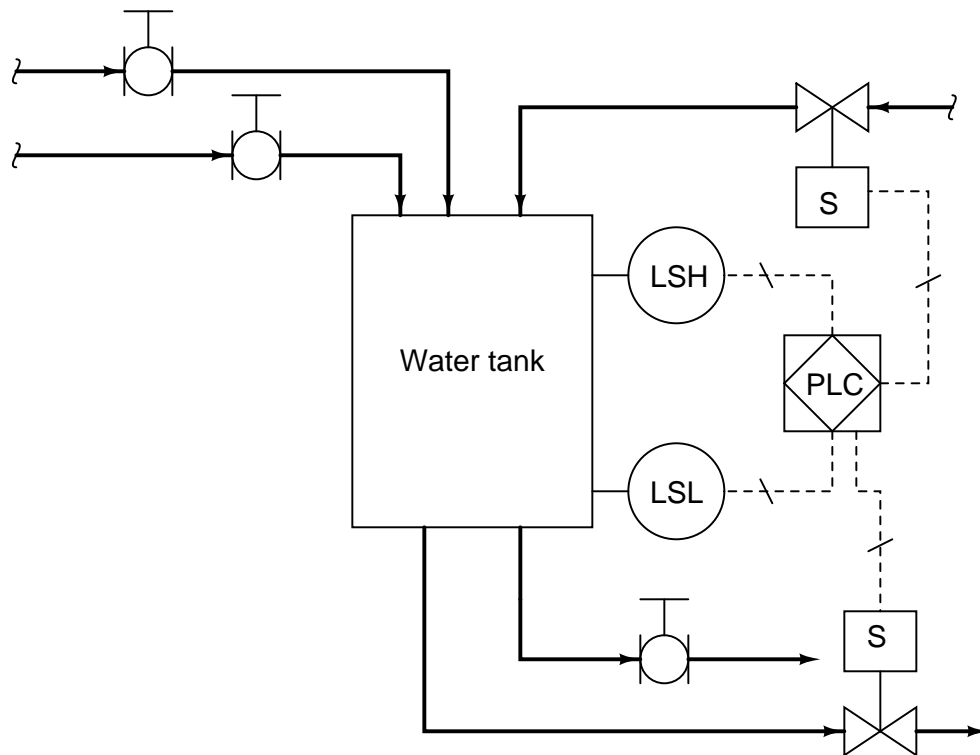
Furthermore, determine the energization status of the PLC output controlling the valve actuator:

- VALVE discrete output channel: *energize* to open the valve or *de-energize* to open the valve?

[file i02383](#)

Question 84

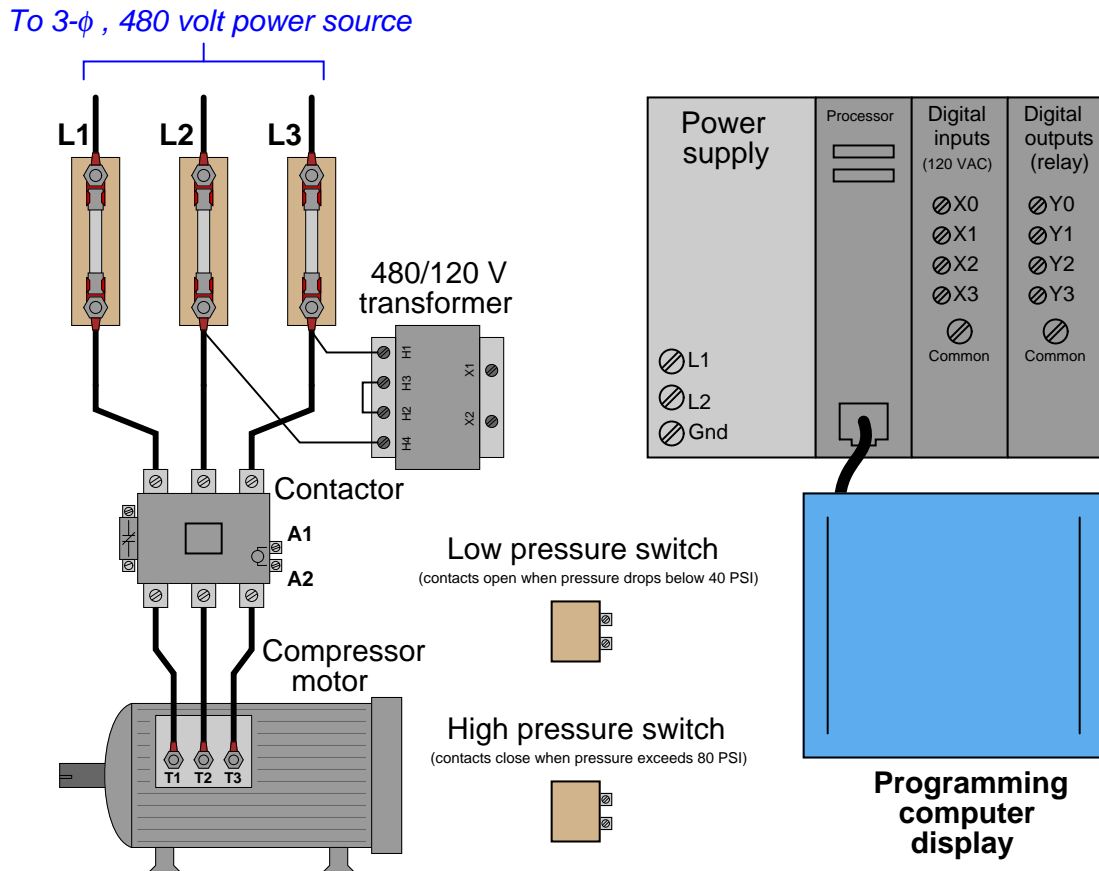
Suppose this PLC-controlled water level system suffers a switch failure, such that the low-level switch never activates to warn the PLC of a low-level condition in the tank:



What operational problems will result from this switch failure? Be as specific as you can in your answer!

Question 85

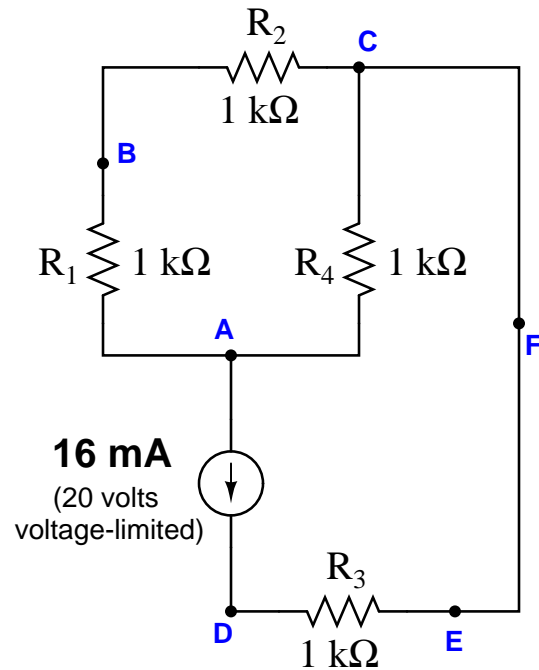
Sketch wires in this diagram to show how a PLC could be connected to a couple of pressure switches and a motor contactor to control the starting and stopping of a three-phase air compressor motor. Note that both pressure switches are normally-open: the contacts are open at atmospheric pressure (0 PSIG), and close as pressure rises. One switch has a setting of 40 PSI, and controls when the motor starts. The other switch has a setting of 80 PSI, and controls when the motor stops.



Also draw a simple ladder logic program in the computer display window for this compressor start/stop function.

Question 86

Suppose a voltmeter registers 0 volts between test points **C** and **B** in this circuit:



Identify the likelihood of each specified fault for this circuit. Consider each fault one at a time (i.e. no coincidental faults), determining whether or not each fault could independently account for *all* measurements and symptoms in this circuit.

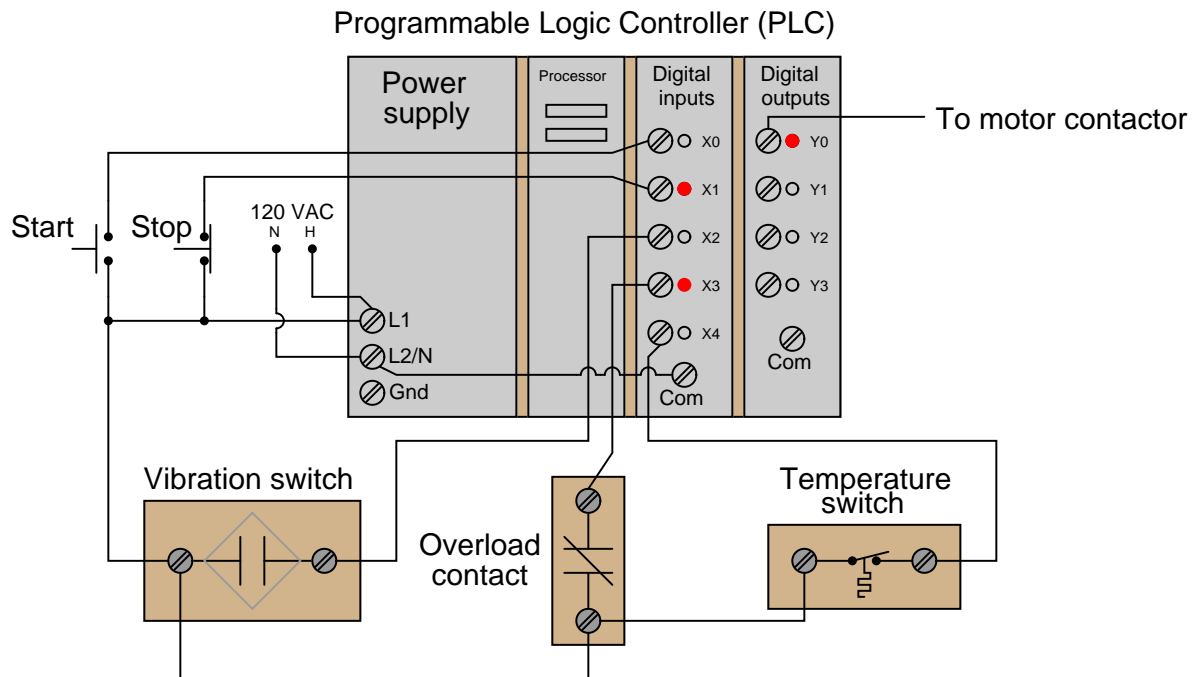
Fault	Possible	Impossible
R_1 failed open		
R_2 failed open		
R_3 failed open		
R_4 failed open		
R_1 failed shorted		
R_2 failed shorted		
R_3 failed shorted		
R_4 failed shorted		
Current source dead		

This question is typical of those in the “Fault Analysis of Simple Circuits” worksheet found in the *Socratic Instrumentation* practice worksheet collection, except that all answers are provided for those questions. Feel free to use this practice worksheet to supplement your studies on this very important topic.

[file i03156](#)

Question 87

A Programmable Logic Controller (PLC) serves as the logic solver (safety shutdown controller) for a large motor-driven pump. It is programmed to shut off the pump if any dangerous conditions occur:



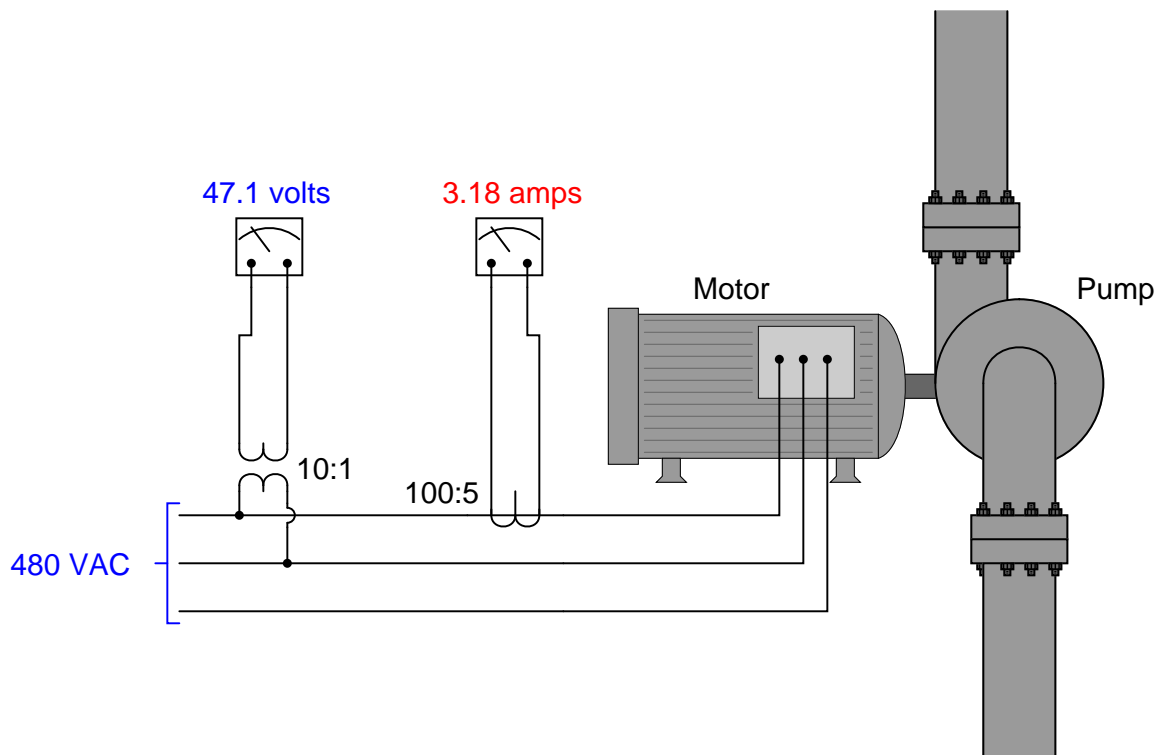
Red LED indicator lights on the input and output cards of the PLC indicate if those respective I/O channels are energized. The light statuses shown in the above diagram are when the pump is running as it should (i.e. no abnormal conditions).

Suppose this pump is running just as it should. Identify how you could electrically simulate each of the following “trip” conditions, causing the pump to shut down even though nothing is physically wrong:

- Electrically simulate a condition of dangerous vibration to the PLC
- Electrically simulate an overloaded motor condition to the PLC
- Electrically simulate a condition of dangerous temperature to the PLC
- Electrically simulate a tripped PLC (to the motor contactor)

Question 88

Suppose a three-phase 480 VAC electric motor is used to turn a pump:

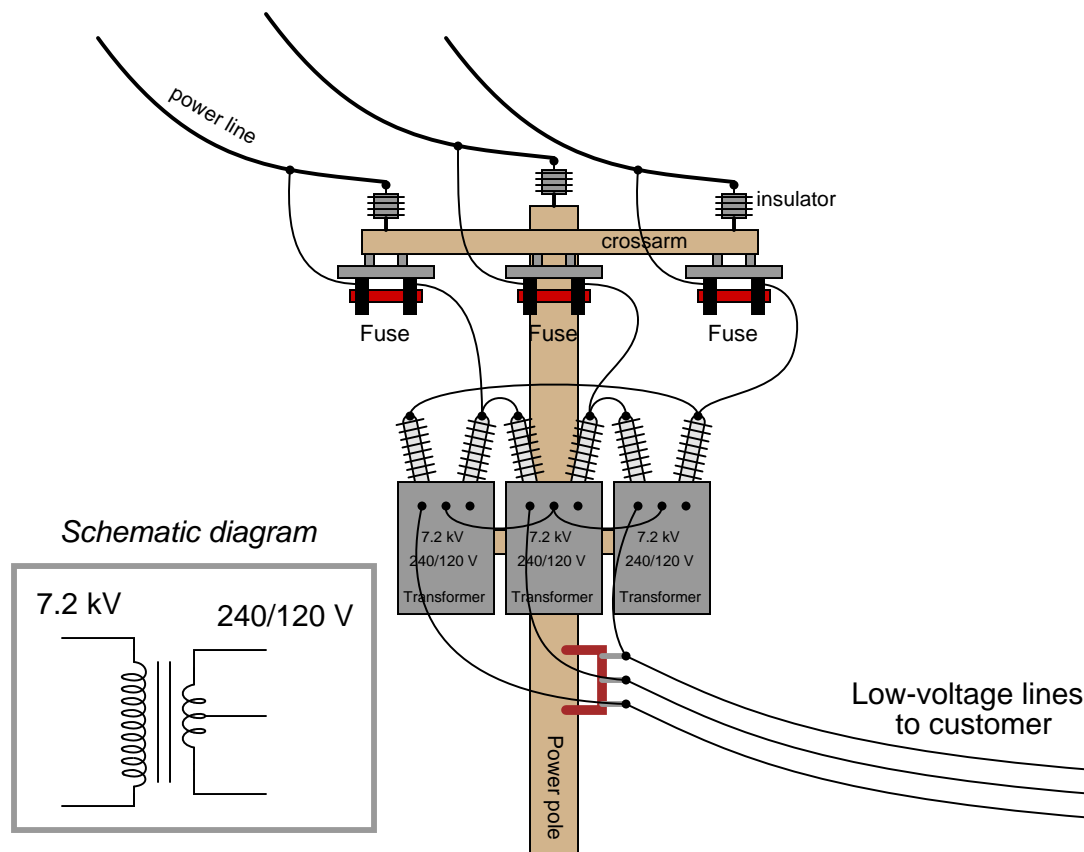


A *current transformer* (CT) with a ratio of 100:5 is used to measure line current. A *potential transformer* (PT) with a ratio of 10:1 is used to measure line voltage. Based on this measurement (assuming the motor presents a balanced load), calculate the following:

- The electrical power delivered to the motor (in horsepower)
- Assuming 93% efficiency, the mechanical power output by the motor (in horsepower)
- Assuming 93% efficiency, the heat dissipation of the motor (in kilowatts)

Question 89

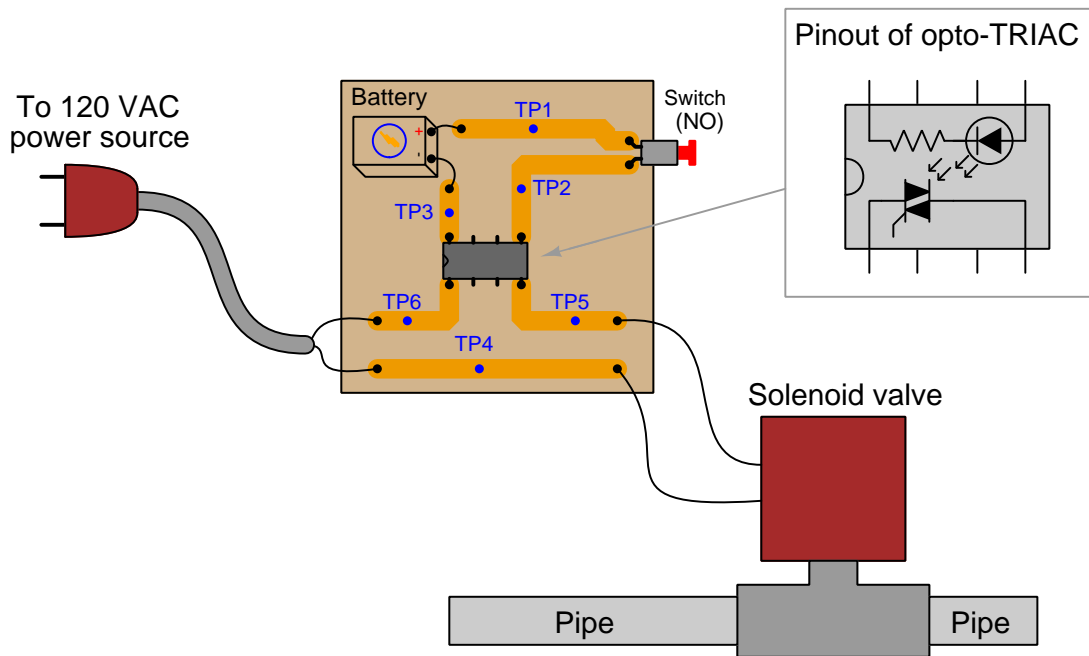
Examine the primary and secondary connections on this three-phase transformer bank, and then determine the line voltage to the customer, assuming 7.2 kV line voltage on the distribution power lines. The schematic diagram shown in the grey box is typical for each of the three transformers:



Assuming this is the last power pole on the line (i.e. the three 7.2 kV power lines dead-end at this transformer bank), calculate current through any one of those 7.2 kV lines if the line current to the customer is 49 amps.

Question 90

A technician is troubleshooting a faulty optically-isolated TRIAC power switching circuit. The solenoid valve is supposed to open up when the pushbutton switch is pressed and shut when the switch is released, but it remains open (passing liquid) no matter what state the switch is in. A mechanic replaces the solenoid valve, thinking it is frozen open. However, even the brand-new solenoid valve remains open and refuses to shut:



Leaving the switch in its normal ("unpressed") position, the technician measures approximately 0.1 volts AC between test points TP5 and TP6, and 9 volts DC (normal for the battery) between test points TP1 and TP3. Based on these voltage measurements, identify two possible faults (either one of which could account for the problem and all measured values in this circuit), and also identify two circuit elements that could not possibly be to blame (i.e. two things that you know *must* be functioning properly, no matter what else may be faulted). The circuit elements you identify as either possibly faulted or properly functioning can be wires, traces, and connections as well as components. Be as specific as you can in your answers, identifying both the circuit element and the type of fault.

- Circuit elements that are possibly faulted
 - 1.
 - 2.
- Circuit elements that must be functioning properly
 - 1.
 - 2.

Lab Exercise – introduction

Your team's task is to configure an HMI (Human-Machine Interface) for a system controlled by a PLC, as well as add a discrete electrical load and a three-pole contactor (energized by one of the PLC's discrete outputs) to the system. The HMI you choose to configure shall allow operator access to (at minimum) discrete input data points, discrete output data points, and either counter or timer instructions. Project ideas include:

- Air compressor control, with high and low air pressure switches
- Water sump pump control, with high and low water level switches
- Other alternatives? *Must be pre-approved by instructor!*

The following table of objectives show what you and your team must complete within the scheduled time for this lab exercise. Note how some of these objectives are individual, while others are for the team as a whole:

Objective completion table:

Performance objective	Grading	1	2	3	4	Team
Team meeting and prototype sketch (do <i>first!</i>)	mastery	–	–	–	–	
Circuit design challenge	mastery					– – – –
Final tagname database for HMI	mastery	–	–	–	–	
Final wiring diagram and system inspection	mastery					– – – –
Demonstration of working system	mastery	–	–	–	–	
Final PLC program inspection	mastery					– – – –
Troubleshooting	mastery					– – – –
Lab question: Wiring connections	proportional					– – – –
Lab question: Commissioning	proportional					– – – –
Lab question: Mental math	proportional					– – – –
Lab question: Diagnostics	proportional					– – – –
Decommission and lab clean-up	mastery	–	–	–	–	
Team tool locker inspection	mastery	–	–	–	–	

The only “proportional” scoring in this activity are the lab questions, which are answered by each student individually. A listing of potential lab questions are shown at the end of this worksheet question. The lab questions are intended to guide your labwork as much as they are intended to measure your comprehension, and as such the instructor may ask these questions of your team day by day, rather than all at once (on a single day).

It is essential that your team plans ahead what to accomplish each day. A short (10 minute) team meeting at the beginning of each lab session is a good way to do this, reviewing what's already been done, what's left to do, and what assessments you should be ready for. There is a lot of work involved with building, documenting, and troubleshooting these working instrument systems!

As you and your team work on this system, you will invariably encounter problems. You should always attempt to solve these problems as a team before requesting instructor assistance. If you still require instructor assistance, write your team's color on the lab whiteboard with a brief description of what you need help on. The instructor will meet with each team in order they appear on the whiteboard to address these problems.

Lab Exercise – team meeting and prototype sketch

An important first step in completing this lab exercise is to **meet with your instructor** as a team to discuss safety concerns, team performance, and specific roles for team members. If you would like to emphasize exposure to certain equipment (e.g. use a particular type of control system, certain power tools), techniques (e.g. fabrication), or tasks to improve your skill set, this is the time to make requests of your team so that your learning during this project will be maximized.

An absolutely essential step in completing this lab exercise is to work together as a team to **sketch a prototype diagram** showing what you intend to build. This usually takes the form of a simple electrical schematic and/or loop diagram showing all electrical connections between components, as well as any tubing or piping for fluids. This prototype sketch need not be exhaustive in detail, but it does need to show enough detail for the instructor to determine if all components will be correctly connected for their safe function.

For example, if you intend to connect field devices to a PLC (Programmable Logic Controller), your prototype sketch must show how those devices will connect to typical input/output terminals on the PLC, where electrical power will be supplied, the placement of all overcurrent protection devices (e.g. fuses) and their current ratings, etc. Prototype sketches need not show all intermediary connections between components, such as terminal blocks in junction boxes between the field device and the controller.

You should practice good problem-solving techniques when creating your prototype sketch, such as consulting equipment manuals for information on component functions and marking directions of electric current, voltage polarities, and identifying electrical sources/loads. Use this task as an opportunity to strengthen your analytical skills! Remember that you will be challenged in this program to do all of this on your own (during “capstone” assessments), so do not make the mistake of relying on your teammates to figure this out for you – instead, treat this as a problem *you* must solve and compare your results with those of your teammates.

Your team’s prototype sketch is so important that the instructor will demand you provide this plan before any construction on your team’s working system begins. *Any team found constructing their system without a verified plan will be ordered to cease construction and not resume until a prototype plan has been drafted and approved!* Similarly, you should not deviate from the prototype design without instructor approval, to ensure nothing will be done to harm equipment by way of incorrect connections. Each member on the team should have ready access to this plan (ideally possessing their own copy of the plan) throughout the construction process. Prototype design sketching is a skill and a habit you should cultivate in school and take with you in your new career.

When selecting a three-pole contactor to include in this project, choose one with a coil compatible with the discrete outputs of your PLC. If your PLC’s discrete outputs are a mis-match for the contactor coil, you will need to install an *interposing relay* to convert the PLC’s DC output control signal into a driving signal suitable for the contactor (e.g. a relay interposing between a PLC with 24 VDC discrete outputs and a contactor with a 120 VAC rated coil, or a relay interposing between a PLC with a 100 mA rated discrete output and a contactor coil drawing 500 mA).

PLC equipment manuals always provide sample diagrams showing how external components may connect to the I/O points. Feel free to use these sample diagrams as templates for your prototype sketch. *This is the most challenging portion of your wiring, so be sure to work with your teammates to get this right!*

Planning a functioning system should take no more than a couple of hours if the team is working efficiently, and will save you hours of frustration (and possible component destruction!).

Lab Exercise – planning the interface

An essential step in programming an HMI panel is to define the list of *tagnames* and *data types* the HMI will need to reference in order to fulfill its function. While this seems like something that may be done while in the process of programming the HMI's graphical display objects, it is actually best done *before* any HMI programming is done at all. Listing all the signals and variables the HMI will need to reference within the PLC it connects to as a first step helps ensure that the naming convention you choose for these tags make sense in the grand scheme of the system design, rather than being ad-hoc in nature. Well-named tags go a long way to making the HMI easier to understand and update in the future. It also helps identify changes that may need to be made to the PLC program to provide the HMI with all the data it will need.

The next step should be finding appropriate documentation for your HMI. You may locate this on the HMI manufacturer's website. Use this documentation to identify how to properly wire, power, and program the HMI display unit.

Planning the interface should take no more than an hour if the team is working efficiently, and will save you hours of frustration (and possible component destruction!).

Lab Exercise – circuit design challenge

Your instructor will randomly select one motor control or discrete sensing device and one brand/model of PLC I/O from the lists shown below, for which you must sketch an accurate circuit diagram showing how the PLC would connect to the motor control or sensor to control/receive its status. The instructor will also randomly select which channel to use on the PLC's I/O card (e.g. *use channel #4 on a 16-channel I/O card*). If additional electrical components are required (e.g. DC power source, electromechanical relay, etc.), those must be incorporated into your diagram as well.

This exercise tests your ability to locate appropriate information in technical manuals and sketch a correct discrete control circuit for a given PLC and sensor/motor control. The electronic Instrumentation Reference contains manuals for all brands and models listed below, and will be available to you during this exercise. Your instructor will request you show the pages from the appropriate manuals in conjunction with your finished sketch, to verify correct connections between the PLC and field device.

Switch options

- Proximity/limit switch (generic)
 - Mechanical limit switch (3 terminals: NO, NC, and COM)
 - Mechanical process switch (3 terminals: NO, NC, and COM)
 - Inductive proximity switch, sourcing (3 wires: +, –, and Out)
 - Inductive proximity switch, sinking (3 wires: +, –, and Out)
- Electronic level switch
 - Rosemount 2120 vibrating fork level switch with relay output
 - Rosemount 2120 vibrating fork level switch with PNP output
- Electronic flowmeter
 - Rosemount 8800C flow transmitter with pulse output

Motor Control options

- Variable Frequency Drives (VFDs)
 - Rockwell PowerFlex 4 (discrete “run” input in SNK mode)
 - Rockwell PowerFlex 4 (discrete “run” input in SRC mode)
 - Rockwell bulletin 1333 (2-wire Stop/Start control mode)
 - Automation Direct GS1 (discrete “forward” input)
 - Automation Direct GS1 (discrete “reverse” input)
- Soft-start
 - Rockwell bulletin 150 SMC-Flex (discrete “start” control)

Note: instructor will specify which channel to use on the PLC's I/O card.

Programmable Logic Controller options

- Automation Direct “CLICK” I/O cards
 - Input module: C0-08ND3 (8-channel DC)
 - Input module: C0-08NE3 (8-channel AC/DC)
 - Input module: C0-08NA (8-channel AC)
 - Output module: C0-08TD1 (8-channel DC sinking)
 - Output module: C0-08TD2 (8-channel DC sourcing)
 - Output module: C0-08TA (8-channel AC)
 - Output module: C0-04TRS (4-channel relay)
- Siemens S7-300 I/O cards
 - Input module: DI 32 x AC 120 V (6ES7321-1EL00-0AA0)
 - Input module: DI 32 x DC 24 V (6ES7321-1BL00-0AA0)
 - Input module: DI 16 x DC 24 V (6ES7321-1BH50-0AA0)
 - Input module: DI 16 x DC 48-125 V (6ES7321-1CH20-0AA0)
 - Output module: DO 32 x AC 120/230 V/1A (6ES7322-1FL00-0AA0)
 - Output module: DO 16 x DC 24 V/0.5A (6ES7322-1BH01-0AA0)
 - Output module: DO 16 x Rel (6ES7322-1HH01-0AA0)
- Rockwell (Allen-Bradley) ControlLogix 5000 I/O cards
 - Input module: 1756-IA16 (16-channel 120 VAC)
 - Input module: 1756-IB16 (16-channel 24 VDC)
 - Output module: 1756-OA8 (8-channel 120 VAC)
 - Output module: 1756-OB8 (8-channel 24 VDC)
 - Output module: 1756-OW16I (16-channel relay)
- Emerson ROC800 SCADA/RTU
 - Input module: 8-channel discrete DI
 - Input module: 6-channel AC discrete (AC I/O set for “input” mode)
 - Output module: 5-channel discrete DO
 - Output module: 5-channel discrete relay DO-Relay
 - Output module: 6-channel AC discrete (AC I/O set for “output” mode)

It is highly recommended that you approach this exercise the same as for the design of any other electrical circuit: carefully identify all *sources* and *loads* in the circuit, identify devices requiring DC versus devices requiring AC, trace directions of all currents (DC only), and mark the polarities of all voltages (DC only). Most of the mistakes made in this type of circuit design challenge may be remedied by careful consideration of these specific circuit-analysis details.

Note: a very effective problem-solving strategy for determining how to connect different electrical components together to create a DC circuit is to first identify whether each component acts as a *source* or a *load* in that circuit. Then, label voltage polarities (+ , -) and directions of current accordingly. Knowing which way current must flow through each component and which polarity each voltage must have is key to ensuring the inter-component connections are correct.

Common mistakes:

- Failing to use the provided Instrumentation Reference and instead wasting time looking up manuals on the internet.
- Neglecting to consider voltage ratings of field devices vs. PLC I/O channels.
- Not noticing which “common” terminals on PLC I/O cards serve which group of I/O points.
- Mis-matching sinking and sourcing DC devices.
- Misinterpreting “typical” schematic diagrams shown for I/O cards or multi-channel field devices. Note that a “typical” diagram shows you how just *one* of the channels is wired, because the diagram would become too crowded if that same diagram were repeated for every channel on the device!

- Assuming sinking/sourcing characteristic when undocumented (*remember, it's always safe to use an interposing relay if you are not sure!*).

Lab Exercise – documenting the system

Since this lab exercise involves the inclusion of a three-pole electrical contactor and load, you may have to modify your wiring diagram from the previous lab exercise. The same standards from the last lab exercise apply here: it must show every connection, every cable, every terminal block, etc.

When your entire team is finished updating your individual wiring diagrams, call the instructor to do an inspection of the system. Here, the instructor will have students take turns going through the entire system, with the other students checking their diagrams for errors and omissions along the way. During this time the instructor will also inspect the quality of the installation, identifying problems such as frayed wires, improperly crimped terminals, poor cable routing, missing labels, lack of wire duct covers, etc. The team must correct all identified errors in order to receive credit for their system.

After successfully passing the inspection, each team member needs to place their wiring diagram in the diagram holder located in the middle of the lab behind the main control panel. When it comes time to troubleshoot another team's system, this is where you will go to find a wiring diagram for that system!

Lab Exercise – troubleshooting

The most challenging aspect of this lab exercise is *troubleshooting*, where you demonstrate your ability to logically isolate a problem in the system. All troubleshooting is done on an individual basis (no team credit!), and must be done *on a system you did not help build*, so that you must rely on loop diagrams to find your way around the system instead of from your own memory of building it.

Each student is given a limited amount of time to identify both the general location and nature of the fault, logically justifying all diagnostic steps taken. All troubleshooting activities will take place under direct instructor supervision to ensure students are working independently and efficiently.

Failure to correctly identify both the general location and nature of the fault within the allotted time, and/or failing to demonstrate rational diagnostic procedure to the supervising instructor will disqualify the effort, in which case the student must re-try with a different fault. Multiple re-tries are permitted with no reduction in grade.

A standard multimeter is the only test equipment allowed during the time limit. No diagnostic circuit breaks are allowed except by instructor permission, and then only after correctly explaining what trouble this could cause in a real system.

The instructor will review each troubleshooting effort after completion, highlighting good and bad points for the purpose of learning. Troubleshooting is a skill born of practice and failure, so do not be disappointed in yourself if you must make multiple attempts to pass! One of the important life-lessons embedded in this activity is how to deal with failure, because it *will* eventually happen to you on the job! There is no dishonor in failing to properly diagnose a fault after doing your level best. The only dishonor is in taking shortcuts or in giving up.

Common mistakes:

- Neglecting to take measurements with your multimeter.
- Neglecting to check other measurements in the system (e.g. pressure gauge readings).
- Incorrectly interpreting the loop diagram (e.g. thinking you're at the wrong place in the system when taking measurements).
- Incorrect multimeter usage (e.g. AC rather than DC, wrong range, wrong test lead placement). This is especially true when a student comes to lab unprepared and must borrow someone else's meter that is different from theirs!

Remember that the purpose of the troubleshooting exercise is to foster and assess your ability to intelligently diagnose a complex system. Finding the fault by luck, or by trial-and-error inspection, is not a successful demonstration of skill. The only thing that counts as competence is your demonstrated ability to logically analyze and isolate the problem, correctly explaining all your steps!

Troubleshooting takes a lot of lab time, usually at least two 3-hour lab sessions for everyone in a full class to successfully pass. Be sure your team budgets for this amount of time as you plan your work, and also be sure to take advantage of your freedom to observe others as they troubleshoot, to better learn this art.

Lab Questions

- **Wiring connections**

- Determine correct wire connections between field components and a PLC I/O card to create a working PLC input or output circuit, based on diagrams of components with terminals labeled
- Correctly determine all electrical sources and loads, as well as all voltage polarities and current directions in a DC input or output circuit, based on diagrams of field components and the PLC's I/O card with terminals labeled

- **Commissioning and Documentation**

- Explain what a “boolean” data point is in a PLC or an HMI
- Explain what an “integer” data point is in a PLC or an HMI
- Explain what a “fixed point” data point is in a PLC or an HMI
- Explain what a “floating point” data point is in a PLC or an HMI
- Explain what an “ASCII” data point is in a PLC or an HMI

- **Mental math** (no calculator allowed!)

- Convert a binary number into decimal
- Convert a binary number into hexadecimal
- Convert a decimal number into binary
- Convert a hexadecimal number into binary
- Convert a hexadecimal number into decimal

- **Diagnostics**

- Examine a PLC program and identify any mistakes in it
- Determine whether or not a given diagnostic test will provide useful information, given a set of symptoms exhibited by a failed system
- Identify at least two plausible faults given the results of a diagnostic test and a set of symptoms exhibited by a failed system
- Propose a diagnostic test for troubleshooting a failed system and then explain the meanings of two different test results

file i03741

Answers

Answer 1

Both the lamp and the solenoid coil will be energized.

Answer 2

Answer 3

Answer 4

Answer 5

Answer 6

Answer 7

Both pushbutton switches are *sourcing* current to the PLC's inputs.

I'll let you figure out what the problem is in this system. *Hint: it's a very common mistake made by students first learning how to interface HMIs with PLCs.*

Answer 8

Note: standard electronic proximity switch wiring color codes are as follows:

- **Brown** = Positive (+) DC supply
- **Blue** = Negative (−) DC supply
- **Black** = Signal output

Answer 9

Answer 10

- $10_2 = 2_{10}$
- $1010_2 = 10_{10}$
- $10011_2 = 19_{10}$
- $11100_2 = 28_{10}$
- $10111_2 = 23_{10}$
- $101011_2 = 43_{10}$
- $11100110_2 = 230_{10}$
- $10001101011_2 = 1131_{10}$

Answer 11

- $7_{10} = 111_2$
- $10_{10} = 1010_2$
- $19_{10} = 10011_2$
- $250_{10} = 11111010_2$
- $511_{10} = 11111111_2$
- $824_{10} = 1100111000_2$
- $1044_{10} = 10000010100_2$
- $9241_{10} = 10010000011001_2$

Answer 12

There are only eight valid ciphers in the octal system (0, 1, 2, 3, 4, 5, 6, and 7), with each successive place carrying eight times the “weight” of the place before it.

- 35_8 into decimal: 29_{10}
- 16_{10} into octal: 20_8
- 110010_2 into octal: 62_8
- 51_8 into binary: 101001_2

Answer 13

There are sixteen valid ciphers in the hexadecimal system (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F), with each successive place carrying sixteen times the “weight” of the place before it.

- 35_{16} into decimal: 53_{10}
- 34_{10} into hexadecimal: 22_{16}
- 11100010_2 into hexadecimal: $E2_{16}$
- 93_{16} into binary: 10010011_2

Follow-up question: why is hexadecimal considered a “shorthand” notation for binary numbers?

Answer 14

Step 1: 10_{16}
Step 2: 30_{16}
Step 3: 20_{16}
Step 4: 60_{16}
Step 5: 40_{16}
Step 6: $C0_{16}$
Step 7: 80_{16}
Step 8: 90_{16}

Follow-up question: write the same sequence in decimal rather than hexadecimal:

Step 1:
 Step 2:
 Step 3:
 Step 4:
 Step 5:
 Step 6:
 Step 7:
 Step 8:

Answer 15

<i>Binary place-weights</i>							
1001.1011₂ = 9.6875₁₀							
$\frac{1}{2^3}$	$\frac{0}{2^2}$	$\frac{0}{2^1}$	$\frac{1}{2^0}$	•	$\frac{1}{2^{-1}}$	$\frac{0}{2^{-2}}$	$\frac{1}{2^{-3}}$
$\frac{1}{2^4}$							

<i>Octal place-weights</i>							
4027.3612₈ = 2071.471191406₁₀							
$\frac{4}{8^3}$	$\frac{0}{8^2}$	$\frac{2}{8^1}$	$\frac{7}{8^0}$	•	$\frac{3}{8^{-1}}$	$\frac{6}{8^{-2}}$	$\frac{1}{8^{-3}}$
$\frac{2}{8^{-4}}$							

<i>Hexadecimal place-weights</i>							
C1A6.32B9₁₆ = 49574.198135376₁₀							
$\frac{C}{16^3}$	$\frac{1}{16^2}$	$\frac{A}{16^1}$	$\frac{6}{16^0}$	•	$\frac{3}{16^{-1}}$	$\frac{2}{16^{-2}}$	$\frac{B}{16^{-3}}$
$\frac{9}{16^{-4}}$							

Binary	Octal	Decimal	Hexadecimal
10010	22	18	12
1011100	134	92	5C
11010	32	26	1A
110111	67	55	37
1100101	145	101	65
100100010	442	290	122
1111101000	1750	1000	3E8
11011110	336	222	DE
1011010110	1326	726	2D6

- byte = 8 bits
- nybble = 4 bits
- word = *depends on the system*

The term “word,” is often used to represent 16 bits, but it really depends on the particular system being spoken of. A binary “word” is more accurately defined as the default width of a binary bit grouping in a digital system.

Follow-up question: what binary grouping corresponds to a single hexadecimal character?

$$-1.00001110000001100000000 \times 2^{-83}$$

$$+1.11010110000000000000000 \times 2^{77}$$

$$+0.11111100000110001000000 \times 2^{-126}$$

The number 1 is represented as follows (literally, $1.0 \times 2^{127-127}$):

Sign	Exponent (E)	Mantissa (m)
0	01111111	000000000000000000000000

Answer 22

Answer 23

Answer 24

Answer 25

To answer this question, you are going to have to research the input and output connections for these specific devices, from the manufacturer’s literature.

Answer 26

Answer 27

Answer 28

The **Vib_high** contact instruction should be drawn as normally-closed rather than normally-open as shown in the technician’s first draft of the PLC program. The program is designed to shut down the motor if ever the **Shutdown** bit goes to a 1 state. This means the motor will shut down if any of the permissive contact instructions become colored (i.e. “conductive” to virtual power). We have been told that the real-world vibration switch is NC, which means its contact opens when vibration is excessive. This means high vibration causes that bit to be 0, which necessitates an NC contact instruction so that it will color under that condition.

Answer 29

The **Oil_press_low** contact instruction should be drawn as normally-open rather than normally-closed as shown in the technician’s first draft of the PLC program. The program is designed to shut down the motor if ever the **Shutdown** bit goes to a 0 state. This means the motor will shut down if any of the permissive contact instructions become uncolored (i.e. fails to “conduct” virtual power). We have been told that the real-world oil pressure switch is NO, which means its contact opens when oil pressure becomes too low. This means a low oil pressure condition causes that bit to be 0, which necessitates an NO contact instruction so that it will un-color under that condition.

Answer 30

The **Stop_PB** contact instruction should be drawn as normally-closed rather than normally-open as shown in the technician’s first draft of the PLC program. The program is designed to shut down the motor when that pushbutton is pressed (i.e. the **Stop_PB** contact instruction becomes uncolored (i.e. fails to “conduct” virtual power). We have been told that the real-world Stop pushbutton switch is NO, which means its contact closes when pressed. This means pressing the Stop pushbutton causes that bit to be 1, which necessitates an NC contact instruction so that it will un-color under that condition.

Answer 31

The **High_pressure** contact instruction should be drawn as normally-closed rather than normally-open as shown in the technician’s first draft of the PLC program. This particular switch signals a high-pressure condition by opening its contacts, since the real-world switch is normally-closed (NC). We want the 3-second timer to begin timing when this switch senses a high pressure, and so we need the contact instruction to color when the real-world high-pressure switch trips (opens). A PLC contact instruction that colors when it senses a “0” bit condition is a normally-closed (NC) instruction.

Answer 32

Answer 33

Answer 34

Answer 35

Answer 36

Answer 37

Answer 38

Answer 39

Answer 40

Answer 41

Answer 42

The suggestion for calculating roots other than two uses logarithms and exponentials (anti-logarithms). It is based on mathematical laws of logarithms and exponents such as these, which have been used in antiquity to multiply, divide, and raise to powers long before the advent of electronic calculators or computers:

$$\log(AB) = \log A + \log B \qquad AB = e^{(\log AB)} = e^{(\log A + \log B)}$$

$$\log\left(\frac{A}{B}\right) = \log A - \log B \qquad \frac{A}{B} = e^{\log\left(\frac{A}{B}\right)} = e^{(\log A - \log B)}$$

$$\log(A^B) = B \log A \qquad A^B = e^{\log A^B} = e^{B \log A}$$

$$\sqrt[B]{A} = A^{\frac{1}{B}} \qquad A^{\frac{1}{B}} = e^{\log A^{\frac{1}{B}}} = e^{\frac{1}{B} \log A} = e^{\frac{\log A}{B}}$$

Answer 43

Answer 44

Yes, this same technique may be applied in a Siemens S7-200 PLC. For a “word” (16 bit) integer stored in V memory register VW88, the contact bit address would be V88.0 because Siemens uses the decimal point character (.) as a bit delimiter. Allen-Bradley, by contrast, uses the forward slash character (/) as a bit delimiter.

Answer 45

At 100 deg F, N7:2 = 100 (*rounded up from 99.72*)

At 800 deg F, N7:2 = 799 (*rounded up from 798.6*)

Answer 46

Answer 47

00.0 = “Correct weight” = Between 786.45 lbs and 805.84 lbs

00.1 = “Too heavy” = Equal to or exceeds 805.84 lbs

00.2 = “Too light” = Equal to or less than 786.45 lbs

Answer 48

Answer 49

Answer 50

A likely cause of the failure is an “open” electrical fault on the acknowledge pushbutton circuit. Other possible faults include:

- PLC input X2 failed (low state)
- Operators pressing the wrong acknowledge button

A “quick fix” for the operators to shut off the annoying siren is to force bit Y5 to a 0 state while you diagnose the problem further and execute repairs.

Answer 51

The “Up” limit switch is failed shorted, making the PLC “think” the elevator is in the full-up position, which prevents it from trying to move upwards. The elevator will not move down either, because the “Down” limit switch properly indicates the platform’s full-down position, preventing any further downward motion.

We may tell this is the fault by examining the LED status indicators on the input card for the two limit switches. You will note that both channels 1 and 2 are lit, indicating *both* limit switches are sending signals to the PLC’s input card. According to the symbols shown on the limit switches themselves, these are normally-open (NO) switches, which means they should pass power to the PLC only if they detect the presence of the elevator platform. When the platform is not at one of those positions, the switch is supposed to return to its resting state (open) and thereby de-energize its respective PLC input channel.

The fact that both LEDs are lit is an indication something is wrong with the limit switches. Seeing that both limit switches have NO contacts, and knowing the elevator platform happens to be in the fully down position, we may conclude that the “up” limit switch is failed shorted.

Answer 52

The PLC is not seeing the signal it needs to from the NC “Stop” pushbutton, and so it “thinks” someone is pressing the Stop pushbutton. Possible faults include:

- Stop pushbutton switch failed open
- Open wire fault from Stop pushbutton to IN5 terminal
- Open wire fault from terminal 3 to Stop pushbutton
- Open wire fault from terminal 4 to fuse F1
- Fuse F1 blown

Answer 53

There is an electrical fault somewhere in the “Up” contactor coil circuit. Possible faults include:

- Up contactor coil failed open
- Output card channel 3 failed open
- Open wire fault from Up contactor coil to OUT3 terminal
- Open wire fault from terminal 7 to Up contactor coil
- Open wire fault from fuse F3 to VAC 1 terminal
- Fuse F3 blown

Answer 54

Answer 55

Answer 56

Answer 57

Answer 58

Answer 59

Answer 60

Answer 61

The maintenance warning light should be on steady.

Answer 62

Answer 63

Hint: everything inside the “Send” instruction is correct. The problem lies with the X1 contact instruction. *The distinction we need to make is that of DATA PAYLOAD versus TRIGGERING EVENT.*

Answer 64

The rising- and falling-edge contact instructions are necessary to keep the two Modbus instructions from “colliding” with each other over time. All in all, six integer registers are communicated between the PLC and the Modbus device.

Answer 65

Partial answer:

In order to alter the program for regular Modbus message transmissions, you can replace the B3:0/5 contact instruction with a contact instruction linked to one of the S:4 status bits (a continually counting binary number in the Allen-Bradley PLC’s free-running clock). Choose whichever status bit toggles at a frequency closest to 2 Hz (once every half-second).

For Koyo CLICK PLCs, the “Receive” and “Send” instructions are fairly self-explanatory.

For Allen-Bradley MicroLogix PLCs, the “Message” (MSG) instruction is the one to use, configured for “500CPU” Communication Command type. Follow these steps to ensure good operation:

- Configure the two PLCs which will be networked to each other via Ethernet with compatible IP addresses and subnets (e.g. 192.168.0.5 and 192.168.0.7, each with a subnet mask of 255.255.255.0).
- Either connect the two MicroLogix PLCs with a single Ethernet cable, or plug them into an Ethernet hub, allowing a laptop to still be connected to them for programming and monitoring purposes.
- Create a new “Message” file (e.g. **MG9**) and a new “Routing Information” file (e.g. **RI10**). These will need to be referenced within the setup screen of the MSG instruction.
- Reference this new Message file in the MSG instruction box (e.g. MSG File = **MG9:0**).
- Double-click the words “Setup Screen” in the MSG instruction box to access the setup screen where you may enter all the other configuration data for this instruction.
- Under the “This Controller” section, select Channel 1 (if you double-click on the entry field, a pull-down arrow appears, allowing you to select among options).
- Under the “This Controller” section, set the “Communication Command” to either **500CPU Read** or **500CPU Write**, depending on whether you wish to have the MSG instruction read (receive) data from another PLC, or write (send) data to another PLC.
- Under the “This Controller” section, set the “Data Table Address” to the memory location within this PLC accessed by the MSG instruction. If you are writing information to another PLC, this is the register where the transmitted data will come from. If you are reading information from another PLC, this is the register where the received data will be placed.
- Under the “This Controller” section, set the “Size in Elements” for the number of contiguous registers you wish to write or read. If you only wish to read or write a single 16-bit register, leave this parameter set for 1.
- Under the “Target Device” section, set the “Data Table Address” to the memory location within the other PLC accessed by the MSG instruction. If you are writing information to another PLC, this is the register in the other PLC where the transmitted data will be written to. If you are reading information from another PLC, this is the register in the other PLC where the received data will be read from.
- Under the “Target Device” section, select **Local**.
- Under the “Target Device” section, specify the Routing Information file you created earlier (e.g. **RI10:0**).
- Select the MultiHop tab on the Setup Screen window, and there specify the IP address of the target device (i.e. the IP address of the other PLC).

Answer 67

For Koyo CLICK PLCs, the “Receive” and “Send” instructions are fairly self-explanatory.

For Allen-Bradley MicroLogix PLCs, the “Message” (MSG) instruction is the one to use, configured for “500CPU” Communication Command type. Follow these steps to ensure good operation:

- Configure the two PLCs which will be networked to each other via Ethernet with compatible IP addresses and subnets (e.g. 192.168.0.5 and 192.168.0.7, each with a subnet mask of 255.255.255.0).
- Either connect the two MicroLogix PLCs with a single Ethernet cable, or plug them into an Ethernet hub, allowing a laptop to still be connected to them for programming and monitoring purposes.
- Create a new “Message” file (e.g. **MG9**) and a new “Routing Information” file (e.g. **RI10**). These will need to be referenced within the setup screen of the MSG instruction.
- Reference this new Message file in the MSG instruction box (e.g. MSG File = **MG9:0**).
- Double-click the words “Setup Screen” in the MSG instruction box to access the setup screen where you may enter all the other configuration data for this instruction.
- Under the “This Controller” section, select Channel 1 (if you double-click on the entry field, a pull-down arrow appears, allowing you to select among options).
- Under the “This Controller” section, set the “Communication Command” to either **500CPU Read** or **500CPU Write**, depending on whether you wish to have the MSG instruction read (receive) data from another PLC, or write (send) data to another PLC.
- Under the “This Controller” section, set the “Data Table Address” to the memory location within this PLC accessed by the MSG instruction. If you are writing information to another PLC, this is the register where the transmitted data will come from. If you are reading information from another PLC, this is the register where the received data will be placed.
- Under the “This Controller” section, set the “Size in Elements” for the number of contiguous registers you wish to write or read. If you only wish to read or write a single 16-bit register, leave this parameter set for 1.
- Under the “Target Device” section, set the “Data Table Address” to the memory location within the other PLC accessed by the MSG instruction. If you are writing information to another PLC, this is the register in the other PLC where the transmitted data will be written to. If you are reading information from another PLC, this is the register in the other PLC where the received data will be read from.
- Under the “Target Device” section, select **Local**.
- Under the “Target Device” section, specify the Routing Information file you created earlier (e.g. **RI10:0**).
- Select the MultiHop tab on the Setup Screen window, and there specify the IP address of the target device (i.e. the IP address of the other PLC).

Answer 68

Answer 69

Answer 70

Answer 71

Answer 72

Answer 73
Answer 74
Answer 75
Answer 76
Answer 77
Answer 78
Answer 79
Answer 80
Answer 81
This is a graded question – no answers or hints given!
Answer 82
This is a graded question – no answers or hints given!
Answer 83
This is a graded question – no answers or hints given!
Answer 84
This is a graded question – no answers or hints given!
Answer 85
This is a graded question – no answers or hints given!
Answer 86
This is a graded question – no answers or hints given!
Answer 87
This is a graded question – no answers or hints given!
Answer 88
This is a graded question – no answers or hints given!
Answer 89
This is a graded question – no answers or hints given!
Answer 90
This is a graded question – no answers or hints given!
Answer 91