Project 13 Grade Sheet	Group#:			
	Grader:			
Point breakdowns:				
B (binary): Full points awarded if criteria met, no points otherwise.				
C (criteria): Grade according to the stated criteria for the requirements for each object				
S (split): points are split evenly across all artifacts				
Orange cells are steps that can be done using automated grading scripts				
	Possible Points	**Point Breakdown**	**Points Awarded**	**Notes**
Runtime Monitoring				
AUTOMATED: Execute the grading monitor on the acceptance tests listed in the project 13 writeup. If they pass with no warnings, give full credit. If fewer than 4 warnings are issued, give half credit	15	B		
Testing				
AUTOMATED: To check testing, copy the group's code from their portfolio into a clean copy of the simulator framework and make sure the code will compile.	0	N/A	0	No points allocated for a compiling simulator.
Is the Unit Test Log complete and up to date (all controller modules listed, all tests passing, input and output files properly linked.	7.5	B		
AUTOMATED: Execute the unit tests using the simulator assembled in the design portfolio grading. (Note that this step requires a valid unit_tests.txt summary file). All tests must pass (0 failed assertions), and all tests listed in the unit test log must be listed in the unit_tests.txt file. If the simulator will not compile, award no credit.	7.5	B		
Is the Integration Test Log complete and up to date? "Complete" means all sequence diagrams are tested (up to a total of 20) and include all the original sequence diagrams (1A, 1B, 1C, 2A, 2B, 3A, 4A, 5A, 5B, 6, 7A, 7B, 7C, 8A, 9A). "Up to date" means all tests passing, input and output files properly linked.	7.5	B		
AUTOMATED: Execute the integration tests using the simulator assembled in the design portfolio grading. (Note that this step requires a valid integration_tests.txt summary file). All tests must pass (0 failed assertions), and all tests listed in the integration test log must be listed in the integration_tests.txt file. If the simulator will not compile, award no credit.	7.5	B		
Is the Acceptance Test Log complete and up to date? All acceptance test files listed in Project 13 writeup must be passing. Each entry must be complete (all fields filled out and input and output files properly linked). Any test that does not pass must be documented to describe the problem that causes the test to fail.	5	B		
AUTOMATED: Execute run all acceptance tests from the project writeup and the undisclosed acceptance tests using an arbitrary random seed value. The test must deliver all passengers.	25	B		
Complete and Consistent Portfolio				
This value is computed from the average in the "End-to-End" sheet.	90	B	0	=average end to end score / 4 * 90
Improvements Log				
Is there an entry for project 13 in the improvements log and minimum requirements sheet?	5	B		
is there an "Overall Project Comments" entry in the improvemenst log?	5	B		
Deductions	**Points lost**	**Point Breakdown**	**Deduction Awarded**	
Check the previous project grade sheet. Were the issues noted in that project addressed?	-19	B		

	Possible Points	**Points awarded**
Totals	175	0
Late Penalty	**Percentage**	**Deduction**
Enter the percentage of total score (per late policy)	100	0
Final Score	**Percentage**	**Points**
This is your actual grade	0	0

Bonus (added to your final score, not this project)	**Grade Points**	**Received (y/n)**		**Notes**
Performance Bonus	1	n		Must have the best performance score and pass all acceptance tests and turn in on time
Portfolio Bonus	1	n		Must pass all acceptance tests and have 3.7 or larger average score on end-to-end traceability and turn in on time
Fault Tolerance Bonus	0.5	n		Must capture all dropped messages at a drop rate of 25-50% of all network messages

Project 12 End-to-End Traceability and Complete Design Portfolio Grade Shee		
Blue cells are to be graded by the grading TA and double-checked by the head TA. Each item is given a linear ranking from 0 to 4, with 0 being "completely ignored the requirement" and 4 being "executed the requirement perfectly".		
Portfolio	**Score**	**Notes**
The portfolio conforms to the guidelines provided in the portfolio layout page on the course website, namely: the portfolio is composed of vanilla HTML documents (except where other formats are specifically required), all hyperlinks point to the correct document, and all inline images are present and readable.		
A random sampling of all portfolio files (incluiding test inputs and outputs and code files) contain proper headers listing the group number, course and semester, and all group members' names and andrew IDs.		
All the required design project artifacts are present (Architecture, use cases, scenarios, sequence diagrams, requirements, statecharts, code modules, unit, integration, and acceptancet test files and logs)		
For the remaining items, choose one module (e.g. DoorControl) and perform an end-to-end check on the following list of criteria (item # X - Y).		
Module(s) checked:		
Sequence Diagrams - choose two SD and verify that the following items are correct. The quick reference document has a list of messages with correct/network framework status and replication.		
All network messages are black arrows using the mMessage notation with correct replication		
All framework messages are blue arrows with framework notation (no 'm') and correct replicatior		
Sequence Diagrams to Requirements Traceability		
Use the SD-to-Reqs traceability table to identify the sequence diagram arcs that are traced to the requirements. Is each SD arc relevant to the requirement it is traced to? A relevant arc is one that pertains to either the trigger conditions or values set in the requirement.		
Every requirement traces to at least one sequence diagram arc.		
Requirements to Constraints Traceability		
Every constraint and requirement is listed in the table.		
The entries with X's substantially address the constraint.		
No entry with a ~ directly contradicts the constraint, or directly meets the constraint (then it should have ar X instead of a ~)		
Requirements - Check the following criteria for each requirement		
Requirement has the form IF <trigger condition> THEN <value SHALL/SHOULD be set>.		
Each requirement is numbered, and requirements that set multiple values have a unique number for each SHALL/SHOULD verb		
All trigger conditions and values set conform to the message / framework notation (see the quick reference document for a list of messages)		
All trigger conditions are based on defined state variables or on messages/framework values in the input interface of the controller.		
All values set in the reqs are defined state variables or messages/framework values in the output interface of the controller.		
Statecharts		
The guard conditions for each state are mutually exclusive.		
There are only Time-Triggered behaviors in statecharts (every action performed every time, no actions on arcs, no entry actions)		
Every output listed in the output interface is set in every state		
Every state variable mentioned in the statechart is defined in the requirements document		
If AND substates are present, every output or state variable is only set in one ANDed (concurrent) set of substates.		
If OR substates are present, no transition crosses the superstate boundary		
The top level state machine and every ANDed or Ored set of substates contains exactly one intialization arc		
Every arc (except intialization arcs) is labeled with a unique numbe		

Requirements to Statecharts Traceability - check every requirement against the following criteri		
The requirement pertains to every state it is traced to (e.g. the value set in the requirement pertains to the value set in the statechart and the trigger conditions for the requirement pertain to the guard conditions for one of the transitions into the state).		
The requirement pertains to every transtiion it is traced to (e.g. the trigger conditions of the requirement pertain to the guard conditions of the transition).		
Every requirement traces to at least one state or transition, and every state and transition traces to at leas one requirement.		
Implementation / Code		
The code module properly instantiates the input and output interface specified by the requirements		
The code module does not contain any communication channels other than the provided network and framework interfaces. If helper or utility classes are implemented, verify that they do not permit communication between controllers.		
The controller periods defined in the code (Control.java) correspond to those provided in the quick reference OR modifications have been documented and approved by the course staff.		
The CAN IDs and base CAN IDs defined in MessageDictionary.java match those defined in the network schedule in the portfolio.		
The code is complete and compilable -- when the contents of the elevatorcontrol/ folder is copied into a the appropriate location in a clean copy of the elevator simulator, the simulator can be compiled and executed without runtime/java errors.		
Statecharts to Code Traceability - locate each traceability comment in the code module and verify that the code that follows meets the criteria below:		
The statecharts to code page in the portfolio is complete and up-to-date -- lists every transition and the line number where that transition appears in the implementation.		
The code following traceability comment substantially relates to the statechart transition that it refers to		
The code substantially implements the statechart in a time-triggered way -- states set all outputs and do not have conditional actions (except for computing state variables)..		
Testing and Logs		
The unit test log is complete, up-to-date, and all unit tests pass		
A spot check of one unit test shows that the test correctly tests the states and transitions indicated by the traceability comments, and that these comments correspond to the states and arcs listed for the test in the Unit Test Log		
The integration test log is complete, up-to-date, and all integration tests pass		
A spot check of one integration test shows that the test correctly tests the sequence diagram and that the traceability comments in the test are complete and correct.		
The acceptance test log is complete, up-to-date, and all tests pass		
Network Schedule		
The network schedule contains every network message defined in the quick reference		
The fields for each message are appropriate to the message contents (no "extra" values or missing information)		
The best- and worst- case bandwidth computations are complete, and the simulation test has a value consistent with the analysis.		
Total Score (averaged)	**0**	