Conestoga College

Institute of Technology and Advanced Learning

Document Title	CAN (Controller Area Network) Diagnostics Design (CDD) Project Charter
Revision	2.0
Date Issued	May 6, 2024
Date Effective	May 6, 2024
Department	School of Engineering and Information Technology
Program	Integrated Telecommunication and Computer Technologies
Audience	ESE Semester 6 Students
Authors	Semester 6 Team
	Reviewers
	Michael A. Galle, Professor
	Ali Tehrani, Professor
	Monzur Kabir, Professor
	Jack Cole, Professor

Revision History

Revision	Description of Change	Effective Date
0.1	Charter of year 2014 version 1.0 with dates changed for 2015 delivery - MK	May 11, 2015
1.0	Published for the students - MK	May 11, 2015
1.1	Correction in page 4 and 7 to separate description of call-station panel from floor selection panel.	June 6, 2015
1.2	Charter of year 2015 version 1.1 with modifications for 2016 delivery - MG	May 9, 2016
1.3	Charter update to include new program changes - MG	May 8, 2017
1.4	Charter update - MG	May 7, 2018
1.5	Charter updated to include project modifications	May 6, 2019
1.6	Charter updated to include unique features and to include proposals for Capstone project in Phase III - MG	May 2020
1.7	Charter update - MG	May 5, 2021
1.8	Charter update – MG	May 5, 2022
1.9	Charter update – MG (Replacement of HCS12 with Arduino Controller)	May 5, 2023
2.0	Charter update – MG (Addition of Shield for STM32 Nucleo board)	May 6, 2024

Table of Contents

CAN (Controller Area Network) Elevator

Diagnostics & Control Design Project

1. Academic Need

This project provides students with an opportunity to study and apply design principles for networked systems. They will also apply their Physics, Mathematics, and Software Engineering understanding.

Students will practice the implementation of industry-standard data networking and distributed computing and enhance their abilities of system integration involving multidisciplinary knowledge and skills. For example, CAN networks may be selected by building system engineers to minimize the cost of cabling or by automotive engineers to minimize vehicle weight etc. They will also practice engineering safety procedures, and project management skills. This project delivers learning outcomes as specified in the latest course-outline of Engineering Project VI (EECE73125).

2. Deliverables

The project is divided into 3 phases. The deliverables of these phases are listed below:

Phase-1: (5 Weeks)

1. Design of a CAN messaging protocol that is common amongst all groups
2. A plan of CAN node configuration that is common amongst all groups
3. Fully functioning CAN network involving a variety of controlling and controlled devices and more than two CAN-nodes
4. Test plan and results

Phase-2: (7 Weeks)

5. An accessible (read/write) data repository
6. User interface for remote control
7. An internet based remote access system to monitor and control an industrial system

Phase-3: (2 Weeks)

8. Complete fully functioning CAN diagnostic system
9. Design and implement at least one unique feature or capability
10. Prepare Project Ideas for Capstone
11. Final presentation

All Phases

12. Project plan, weekly status report, meeting minutes, and engineering logbook

3. Project Description

The overall project plan (to be completed in 3 phases) is to build and control an elevator, leveraging the use of a CAN bus communications environment, coupled with networked systems to create a means of tracking the elevator's operation/diagnostic data. The elevator car's position and status must also be viewable/controllable to computers over the Internet. See the system diagram, below.

The **final project demonstration** begins with the elevator car at an initial position. As buttons are pressed at each floor, the system will direct the car to service the requests. The system will use a distance sensor to track the car position, determining the distance from the bottom of the elevator shaft, and thus determine the next floor number. Button presses will be handled over the CAN bus (these functions can also be carried out remotely using virtual/software interface)

In the first phase of the project, application of the above mentioned concepts and skills is demonstrated through the specification, design, programming, and testing of an embedded controller for low-level communications between buttons, displays, and sensors (implemented via hardware available in our lab). This controller will communicate with the sensors and call buttons using the CAN bus (see Figure-1).

Each call station (Floor controller) will consist of

- Buttons and LEDs (virtual/software/online also possible) to indicate the selection
- A display panel, which will show the elevator car's present position and a speaker will chime/announce when an elevator has reached its destination (floor)

There will be another controller (Elevator *Car* controller – call button at bottom of elevator) for:

- Floor selection buttons (possibly virtual buttons) with built-in LED for each button, and
- LEDs (possibly virtual indicators) that indicate the floor the elevator is on

There will be another controller (Supervisory Controller – the Raspberry Pi) that:

- Stores the server, database and application for remote control of the elevator
- Contains a *speaker* that may be used to announce the floor the elevator is on

There will be another controller (Elevator Controller – Arduino Controller) (already implemented) for

- Sensing the height of the elevator
- Controlling the motor to set the height (floor) of the elevator

The distance sensor is interfaced directly to the controller (Arduino Controller) to provide a distance estimate, which can be displayed on an LCD or 7-segment LED display and ultimately used for positional feedback.

The elevator **motor** is interfaced with the controller via servo electronics.

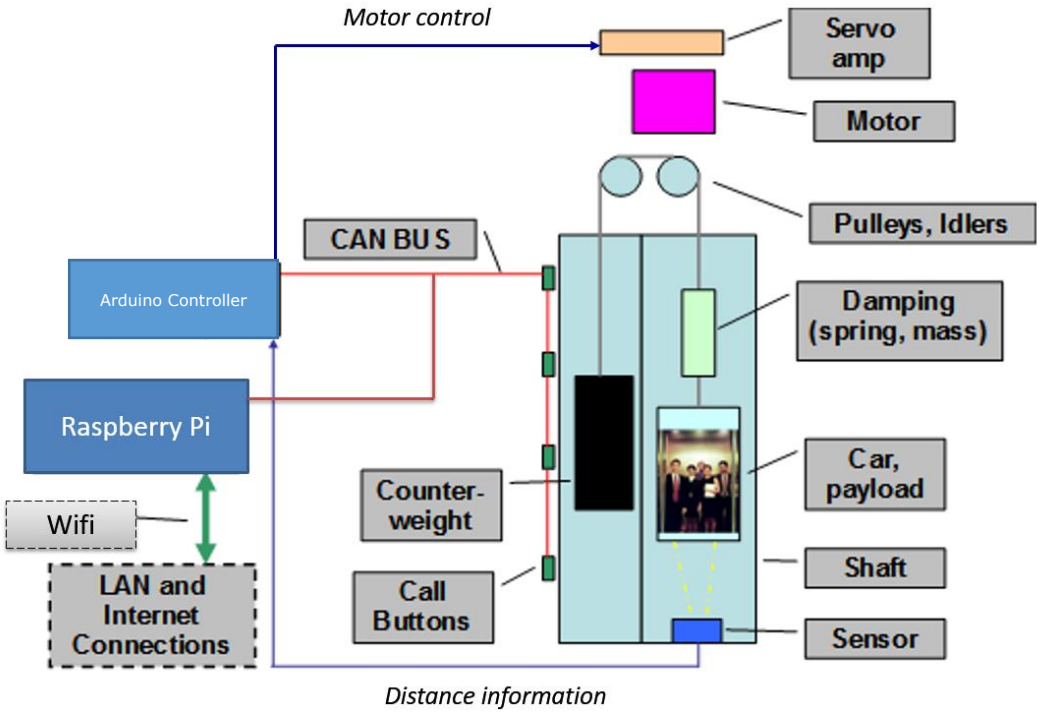

Figure-1

(Note: Above figure depicts an approximate schematic of the system. Actual implementation may be different from this.)

The minimal project (in **Phase 1**) is a CAN network consisting of the controller, the call stations, the distance sensor, LCD/7 segment display and a supervisory controller. Extra credit projects will extend both the hardware design and the software capabilities to include additional functionality; however, these must be approved by the faculty team prior to embarking on extra credit work (where possible). Examples of possible extensions include a passenger on/off detection and counter system to enable the tracking of elevator utilization, or a security camera control system.

In **Phase 2**, the elevator controller will also interface to the supervisory controller on a Local Area Network (LAN) (already accessible over the internet), and thence to the Internet (see Figure-2). Events from the elevator's operation will be logged to the database in the supervisory controller, which will contain software designed, programmed and tested by the student group. This latter software will consist of two parts:

1. **A data server (database)** which will monitor and store data from the controller/CAN network, and

2. **A diagnostics display and control program (website / graphical user interface)**

An aspect of elevator control (such as shutting the elevator down completely or requesting the elevator via a phone app rather than the floor button) will be included.

Finally, in Phase 2, software will be developed to retrieve elevator status information from the elevator module, over an Internet connection.

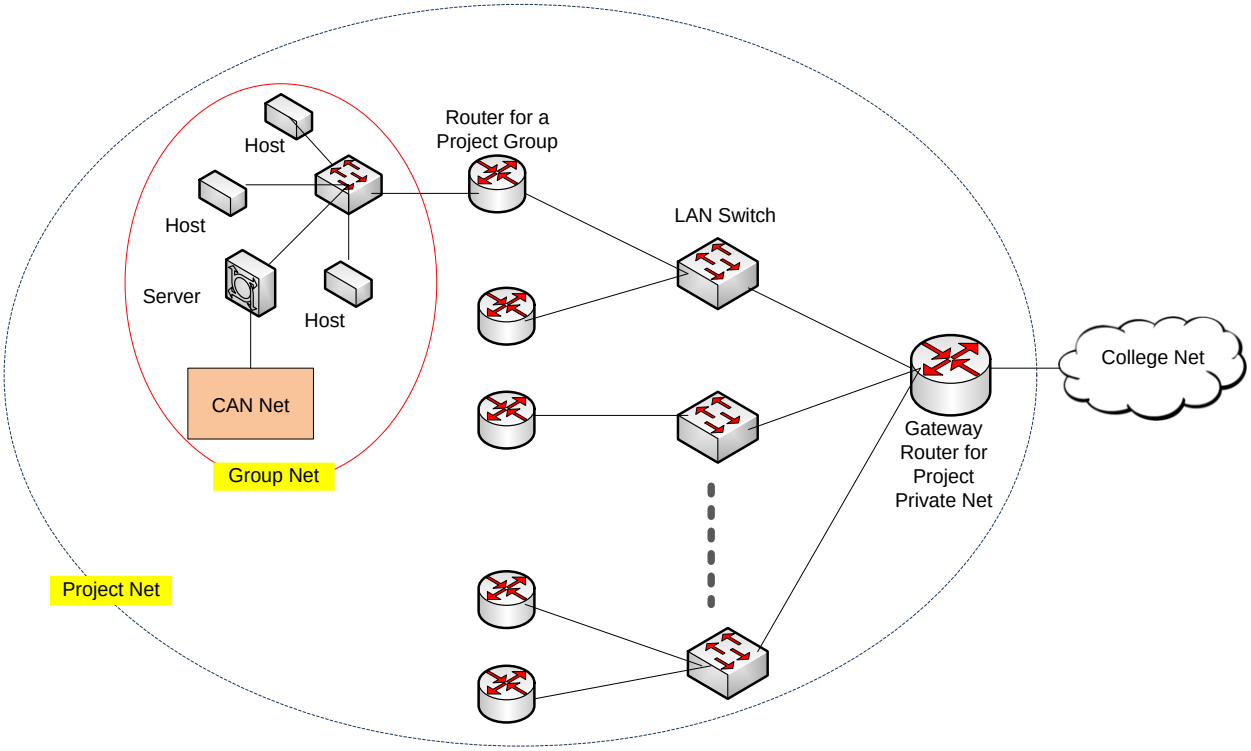

Figure-2

(Note: Above figure depicts an approximate schematic of the system. Actual implementation may be different from this.)

In **Phase 3** the two modules (developed in phases 1 and 2) will be *integrated* to complete the system. This phase also encourages each student group to incorporate features that are not specifically stated in this charter document. This phase includes a final (video) demonstration, oral (video) presentation and report. This phase will also include the preparation of ideas for the next project (Capstone).

Students will work in groups of two to four students (if applicable) (possibly 4-5 students due to number of available elevators).

4. Specific Requirements

This project will meet the following criteria (requirements are identified as **M**ust, **S**hould, **C**ould or **W**on't):

MUSTs

Phase 1 (4 Weeks)

1. The **elevator system** will cover at least *three floors*, with **one CAN node per floor**. Each node will have *up/down switches and lights* (we will also begin building buttons and indicators on a webpage by debrief 2. Students *may include a PCB shield added to the STM32 Nucleo board*)
2. The **elevator car will have a node** with *door open/close switches & lights*, *floor-selection switches & lights*, and *LCD/7-segment display* for displaying current location of the elevator. (we will begin building buttons and indicators on webpage by debrief 2. *Students may include a PCB shield added to the STM32 Nucleo board*)
3. A **controller CAN node** (already installed in the elevator as the Arduino Controller) will perform the following tasks:
   - Detect the *location of the elevator car* (with the distance sensor) and distribute this information to all call-stations for displaying on LCD/7-segment display.
   - Control the motor used to set the floor of the elevator.
   - Detect up/down request at any floor and switch on the respective LED.
   - Detect floor selection at any floor and switch on the respective LED.
   - Detect door open/close request at any floor and switch on an LED.

   * Note that some of the functionality between the **controller** and **elevator car** CAN nodes can be shared
4. Students will implement a **CAN messaging protocol** (see Protocol document provided in eConestoga)

It is recommended that you start to think about how you will implement the elevator operation logic as a finite state machine at this stage (see stage-3)

Phase 2 (7 Weeks)

5. Develop a **user interface** (may be web-based) which will:
   - Implement a *repository to log all the information the CAN nodes* display on their LCD/7-segment display, and a program for data logging and retrieval.
   - Display *diagnostics reports* (all the information the CAN nodes display on their LCD/7-segment display).

- Provide *user action buttons/icons* to send commands to CAN-based elevator system in order to control the elevator operation and/or collect system diagnostic data.

(Note: student groups are not required to implement any communication between the logging and control PC and CAN-system at this stage. This interface will be implemented in phase-3. However, it is highly recommended that groups work ahead)

6. Develop a **server program** on the to provide data from the repository upon request but **only to authorized users**.

7. Write a **client program** (may be web application based) that can access the data **server** and display a diagnostics report and control the operation of the elevator.

8. Write a client program must include the ability for running the elevator in **'sabbath' mode**, where the elevator moves from floor to floor without the need for users to press the call button. This could be automated operation based on date/time (i.e. Saturday).

9. Build a **server program** for announcing the floor number that the elevator is on to user/passengers who may be visually impaired. (May connect a speaker to the Supervisory controller and play recorded messages that announce the floor number)

Phase 3 (2 Weeks)

1. Research on ways to **connect the CAN network with internet** and select one from the options.
2. Implement a **CAN node** to *interface* to establish CAN-LAN communication and diagnostic data logging into the data repository.
3. Develop a **finite state machine** for the *elevator operation logic*, and implement it accordingly. (Students are advised to start working on this as early as possible).
4. Complete at least one **additional capability or feature** that shows **ingenuity of design**

   Examples:

   4.1. Include hardware aids for debugging including test points, LEDs, start/stop switches, analog input knob

   4.2. New features for the elevator (e.g. voice control, facial recognition)
5. Demonstrate the fully functional integrated system (video)
6. Prepare ideas for the Capstone Project in Semester 7 (important!)
7. Produce a Final Presentation (video)

For All Phases

1. Practice safety procedures

2. Practice project management, teamwork, and professionalism
3. Create an appropriate test plan and document test results
4. Create appropriate design documentation
5. Maintain engineering logbook

5. Milestones and Schedules

- Project Start Date: **May 6, 2024**. Project End Date: **August 9, 2024**
- This document listed the milestones for each phase in Section-3 (MoSCoW list). The schedule for each phase is given in the following table.

Deadline	Milestone Deliverable
Week 5	• All the tasks for Phase-1
Week 12	• All the tasks for Phase-2
Week 14	• All the tasks for Phase-3

- Each student-group is responsible for developing a **project plan** and weekly schedule for each phase by the end of first week of that phase. The project-plan will break down the tasks of each phase into weekly tasks.
- Each week a student-group will complete its weekly tasks as indicated in the project plan, and will **write in their (online) engineering log book, produce a project status report, appropriate test-plan and document test results**.
- There will be a weekly briefing to provide more specific information. Some of the briefing sessions will be used to provide students with additional knowledge and information, which may be required for the project but not adequately covered earlier.

6. Project Evaluation

- Students' **project plan** for each phase will be evaluated
- Students' **project work will be evaluated weekly**. Approved project plan will be the basis of the weekly evaluation. Demonstration, engineering log, project status report, appropriate test-plan and documented test results are examples of the items that will be considered during the evaluation. Safety practice and professionalism are also important aspects of the evaluation.
- At the **end of each phase** the work of each group will be evaluated on the basis of the specific requirement as stated in Section 4 (MuSCoW list).
- **Final Presentation (demonstration)** will be marked using evaluation-rubrics. The rubrics will be *made available to the students in advance*. External people may be invited as audience and/or evaluators.

Marks allocation:

- Marks allocation for each phase will be as per the distribution indicated in the course outline of the Engineering Project VI course. The mark for each phase is mostly for the overall group performance, however the evaluator reserves the right to allocate the marks on an individual basis if there is any significant imbalance between individual student's workload and achievements.
    - **Phase-1, 2:** about **80%** of the available marks will be for **technical achievements** and **20%** for **others** (project management, group dynamics, safety and professionalism)
    - **Phase-3:** approximately **50%** of the available marks will be for **technical achievements** including final demonstration, **20% for ingenuity of the additional capabilities or features, 5% for developing Project ideas for the Capstone project and 25% for the final presentation**.
- The course outline also allocates some marks for individual performance. This mark will be based on the student's active participation, attitude, aspects of professionalism, and contribution to project. Individual student's performance will be observed throughout the project duration, and the mark will be assigned at the end of the project.

7. Project Issues and Constraints

7.1. Risk

- Some required knowledge might not be delivered (by the instructors) in a timely manner. This will be remedied as soon as it is detected.
- Any external factors that are beyond the control of the project team

7.2. Constraints

- This project shall be completed in 13 weeks.

- Each student group shall complete this project separately from other groups. While it is expected that students in a group would divide the workload each student requires understanding and to be able to explain other members' work.
- Copying of source code is not permitted unless the academic-term lead professor, or his representative, gives written permission.

7.3. Assumptions

- Each student will dedicate one hour of outside study time for each scheduled class hour.
- Each student will attend all scheduled classes, briefings and debriefings.
- Each student will make wise and efficient use of scheduled class time.
- Each student bench is equipped with the required computer and software.
- Each student bench is equipped with logic analysers and troubleshooting equipment.
- Failure to attend classes, or briefings, or debriefings, or to submit material according to given schedules may result in academic penalties at the discretion of the professor or academic team.

7.4. Project Schedule

See semester schedule for briefing and debriefing time.

7.5. References

- STM32 documentation
- PCAN USB/CAN documentation
- Arduino online documentation: https://docs.arduino.cc/
- CodeWarrior documentation (Deprecated)
- HCS12 documentation from Freescale Semiconductor (Deprecated)
- Axeman Development board documentation (Deprecated)
- Huang,H-W. The HCS12/9S12: An Introduction to Software and Hardware Interfacing. Thomson-Delmar, 2006. ISBN 1-4018-9812-2 (Deprecated)
- Alberto Leon-Garcia and Indra Widjaja. Communication Networks (2nd edition). McGraw-Hill
- Norman S. Nise. Control Systems Engineering (6th edition), John-Wiley Publishers
- Specification documents for various selected components

Appendix A – Graduate Attributes

Professional Body of Knowledge

KB	KB1	KB2	PA	PA1	PA2	PA3	IV	IV1	IV2	IV3	ED	ED1	ED2	ED3	ED4	ED5	ET	ET1	ET2	ET3	ET4
Knowledge base	Facts	Concepts	Problem analysis	Decomposition	Methodology	Validation	Investigation	Research	Measure	Experiment	Design	Problem Defn & Research	Preliminary Design	Detailed Design	Implementation	Verification & Validn	Use of engineering tools	Models/Simulations	Measurement Tools	Manufacturing Tools	CAD Systems
								A	A	D			A	A	A	A			A		

Employability Skills

TM	TM1	TM2	TM3	CM	CM1	CM2	PR	PR1	PR2	PR3	PR4	LL	LL1	LL2	LL3	LL4
Individual and team work	Personal Contribution	Collaboration	Infrastructure	Communication skills	Log Engineering Info	Convey Engineering Info	Professionalism	Work Ethic	Professional Conduct	Professional Contribution	Professional Practice	Life-long learning	Autonomous Learning	Applying Knowledge & Skills	Self Direction & Reflection (Metalearning)	Learning Strategies (Metacognition)
	A	A			A	A		A	A					A		

Professional Responsibility

SC	SC1	SC2	SC3	EE	EE1	PM	PM1	PM2	PM3	PM4	PM5
Impact on society and the environment	Environmental Awareness	Product Life Cycle	Balance & Tradeoff	Ethics and equity	Ethical Responsibility	Economics and project management	Project Scheduling	Resource Allocation & Costing	Risk Management	Business Planning	Economic Analysis
	A				A		A				

LEGEND		
I	**Introduced**	first experience/use
D	**Developed**	continued experience/use
A	**Applied**	integration/extension of knowledge & skills