Säkerhetssystem med MD407

Utveckling av ett MD407-baserat säkerhetssystem med CAN-kommunikation

Philip Antonsson, Felix Bråberg, Linus Haraldsson, Gabriel Käll,
Erik Nilsson, Sebastian Sjögren, Simon Widerberg

Datatekniskt projekt

2020-11-01

Innehåll

Ordlista

Buss En elektrisk bana som används för dataöverförning mellan datorkomponenter [1].

CAN(Controller Area Network) Ett asynkront protokoll för kommunikation mellan separata hårdvaruenheter[2, s. 249].

Discord En plattform för gruppsamtal över internet.

GitHub Webbhotell för lagring av källkod.

Gren En version av mjukvara som lagras på Github.

IO Input-Output, representerar in- och utmatning av data.

Ping En signal som begär ett svar från en enhet [3].

Struct En datastruktur, d.v.s. en bit av bestämt minne i datorn där data är grupperad och lagrad för lätt åtkomst.

SysTick SysTick är en timer baserad på hårdvarans klockcyklar som kan konfigureras för att regelbundet skapa avbrott.

USART Universal Synchronous/Asynchronous Reciever/Transmitter, en gränssnittsenhet som omvandlar data till bitströmmar. Används för att möjliggöra kommunikation mellan två hårdvaruenheter[4].

1 Introduktion

1.1 Syfte

År 2018 var det cirka 82 000, eller 1,8 procent, av rikets hushåll som rapporterade fall av bostadsinbrott i Sverige. Detta antal var detsamma år 2016 och 2017 [5]. Enligt Statistiska centralbyrån har den vanligaste lägenhetstypen två rum och kök [6]. Det är dock inte en majoritet som bor så, och bostadsstorlekar samt bostadstyper varierar. Därmed är det viktigt att säkra larm- och låssytem finns till för medborgares säkerhet, och att de är anpassade för deras bostäder. Projektets syfte är därför att ta fram ett larm- och låssystem baserat på mikrokontrollern MD407 som också är modulärt.

1.2 Mål

Projektets mål var att utveckla ett larm- och låssystem med hjälp av en centralenhet samt två periferienheter och en störenhet. De två periferienheterna var ett dörrlarm och ett rörelselarm. Övergripligt används periferienheterna för att interagera med de fysiska sensorerna och larmen. Centralenheten används för att hantera all information samt kommunikation. Störenheten användes för att stresstesta systemet. Enheterna skall kommunicera med varandra över en gemensam buss med hjälp av protokollet Controller Area Network (CAN).

Två utökningar av grundsystemet planerades. Den ena var att rörelselarmet inte skall larma till centralenheten förrän det befinner sig någon inom en viss variabel räckvidd från enheten. Alltså skall rörelselarmet endast larma lokalt tills räckvidden blir överskriden av någon. Den andra planerade utökningen var att ljudet som uppgör larmet skulle sticka ut och vara så tydligt som möjligt. Prioriteringen av utökningarna var i samma ordning som de introducerades.

1.2.1 Kravspecifikation

Följande kravspecifikation har utvecklats för att uppnå målet och täcka de nödvändiga funktionerna av ett säkerhetssystem:

1. Enheterna ska kunna kommunicera genom CAN-bussen på ett konsekvent sätt.
2. Centralenheten ska lagra aktuell information om alla enheter.
3. Centralenheten ska presentera den viktigaste informationen för användaren genom USART, Universal Synchronous/Asynchronous Reciever/Transmitter.
4. Periferienheterna ska vara konfigurerbara genom USART.
5. Det globala larmet ska utlösas ifall centralenheten förlorar kontakten med en periferienhet.
6. Periferienheterna ska tillämpa ett lokalt larm om sensorer blir triggade, och larma globalt om det behövs.
7. Om larmet har gått kan det avaktiveras från centralenheten genom att en kod matas in på en knappsats.

8. Användaren ska kunna koppla upp systemet med de periferienheter som behövs, och inte vara låst till en speciell konfiguration.

Punkt ett togs fram för att ett system behöver förutsägbar kommunikation för att vara pålitligt. Annars kan en mängd problem uppstå, som att sidoeffekter förekommer vid konfiguration, e.t.c. Punkt två ansågs också vara väsentlig; att bedöma när ett larm ska utlösas kräver aktuell information om systemet. För att användaren ska kunna ta del av denna aktuella information måste den presenteras genom USART på ett förståeligt sätt. Informationen presenteras för användaren så den kan konfigurera perifirienheterna genom USART. Därför behövs punkt tre och fyra som krav. De tre punkterna innan punkt åtta är relaterade till systemets primära funktion - att larma på och av vid rätt tillfälle. Därför är krav fem, sex och sju nödvändiga för ett fungerande system. Punkt åtta är obligatorisk för systemets modularitet, och krävs därför i detta larmsystem.

1.3 Arbetsmetod

Arbetet delades i början av projektet upp i åtta delar, vilka beskrivs i sektion 3. Varje gruppmedlem blev tilldelad en primär deluppgift samt en eller flera närliggande uppgifter att assistera med.

Projektets mjukvara kodades i C med hjälp av textredigeraren Codelite. Tre enheter med olik funktionalitet skulle utvecklas, och det bestämdes att en uppsättning kod per enhet var optimalt. Detta gjorde mjukvaruutvecklingen mer effektiv eftersom koden för de tre enheterna kunde skrivas parallellt, och tester kunde utföras individuellt innan systemets sammankoppling. Testerna på enheternas mjukvara utfördes i hierarkisk ordning. De minsta delarna av programmen testades först, eftersom dessa delar inte förlitar sig på någon annan kod. Efter det testades större och större block av kod, och till slut hela enheten. Testerna på enhetsnivå dokumenterades och diskuteras i avsnitt 5.

För att abstrahera bort svårskriven lågnivåkod användes ”STM32F4xx Standard Peripherals Library”. Det är ett programvarubibliotek för processorn i MD407 som underlättar användningen av diverse hårdvara som enheten stödjer. En del hårdvara tillhörande till dörrenheten och larmenheten behövdes dock utvecklas på egen hand. Den utvecklades med hjälp av en kopplingsplatta, el-komponenter, och egenskrivna funktioner.

Arbetet på de enskilda delenheterna skedde på separata grenar i github. Om en gren skulle slås ihop med master grenen krävdes att följande kriterier uppfylldes.

1. Koden var godkänd av en annan gruppmedlem.
2. Om en skriven funktion inte var trivial krävdes en testfunktion för att bevisa dess funktionalitet.
3. Koden jämfördes mot och rättades efter kodkonventionsdokumentet.
4. Funktionalitetstester utfördes och testdokumenten bifogades.

2 Teknisk bakgrund

Detta kapitel ger en teoretisk översikt över de grundläggande komponenterna i larmsystem, och hur de fungerar. I samband med detta förklaras en del av hårdvaran som används i projektet ingående. Efter det beskrivs teorin bakom kommunikationstandarden CAN.

2.1 Larmsystem

Larmsystem kan variera när det gäller utrustning och komplexitet, men det finns gemensamma komponenter som utgör en bas för systemen. Denna bas består ofta av *sensorer*, *centralutrustning* och *larmdon* [7, s. 15]. Sensorerna är larmsystemets ”sinnen”. Deras uppgift är att upptäcka ändringar i tillstånd. Tillstånden kan vara tryck, ljud, ljus, slutande eller brytande kontakt, [7, s. 15]. Denna ändring i tillstånd signaleras till centralutrustningen genom systemets kommunikationsstandard. Ett gemensamt kommunikationsprotokoll används för det valda mediet.

Om sensorerna är larmsystemets ”sinnen” är centralutrustningen larmsystemets ”hjärna”. Centralenheten tar emot information från sensorerna, och dess uppgift är att tolka vad denna information betyder [7, s. 18]. När centralenheten har tolkat informationen bestämmer den vad som ska göras härnäst. Om någon av sensorerna har gett ett oroväckande utslag kanske centralenheten skickar en signal till ett larmdon.

2.2 Projektets kommunikationstandard, CAN

CAN, Controller Area Network, är en databuss som utvecklades för fordonsindustrin. Denna buss tillåter kommunikation mellan flera enheter, där alla enheter delar samma kommunikationsmedium. Protokollet har också förmågan att självdiagnostisera och reparera fel i meddelanden.

[8, s. 2]

För att förhindra och alternativt optimera kollisioner använder sig CAN av protokollet CSMA/CD+AMP. Detta protokoll gör så att noder ”lyssnar innan de pratar”, de väntar en specificerad tid innan de börja sända information. Om en kollision skulle uppstå används meddelandenas prioritet för att lösa den – meddelandet med högre prioritet fortsätter att sända. När ett larmsystem utvecklas kan detta användas till systemets fördel; utvecklare kan definera att larmmeddelanden alltid har högsta prioritet.

[8, s. 3]

CAN-meddelanden är förpackade i så kallade CAN-ramar och har en standardiserad struktur. Den CAN-ram som anses vara av standardstruktur visualiseras i figur 1. Diverse fält i ramen beskrivs nedan:

ID-fältet Innehåller meddelandets ID.

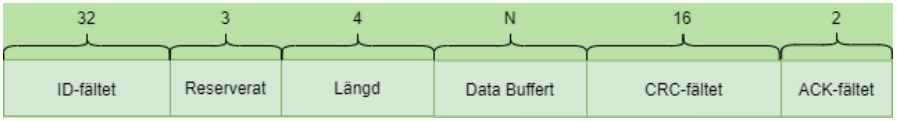
Reserverat Innehåller information angående vad för typ av ram det är samt vilket typ av ID ramen använder sig av.

Längd Längden på databufferten i bytes.

Data Buffert Bufferten där all data ligger. Längd-fältet beskriver databuffertens storlek.

CRC-fältet *Cyclic Redundancy Check* är en felkontrollskod. Den används för att undersöka om ett meddelande har kommit fram oskadat.

ACK *Acknowledge* skickas av mottagande noder för att tala om för avsändarnoden att meddelandet kom fram.



Figur 1: En CAN-ram av standardstruktur. Talen beskriver antalet bytes. N innebär att databuffertens längd kan variera. Den bestäms av värdet i längdfältet.

2.3 Projektets sensortyper

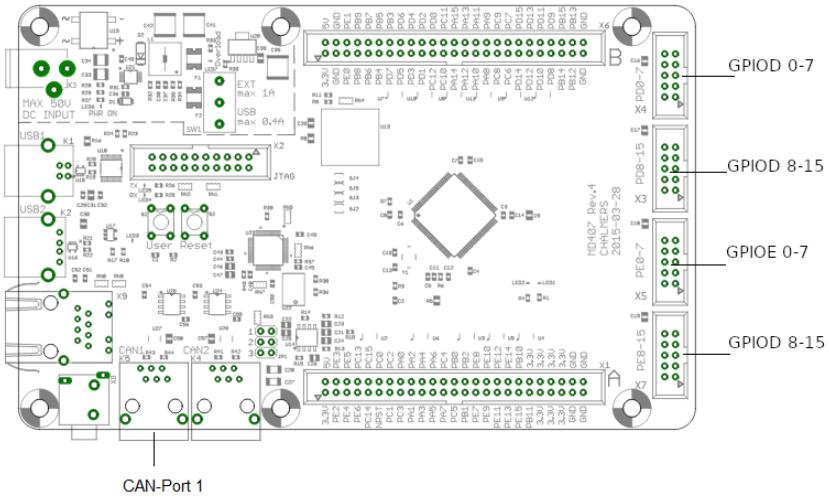
I detta projekt används tre typer av sensorer: en magnetkontakt, en vibrationssensor och en avståndssensor. Magnetkontakten som används består av två delar: en magnet och en mekanisk kontakt. Kontakten påverkas av magnetens fält, och om denna är kopplad till en elektrisk krets, kan denna krets slutas eller brytas [7, s. 15]. Så länge magneten är nära kontakten hålls kretsen sluten, och sensorn behöver ingen strömförsörjning för att detta ska fungera. Det här används för att kontrollera om en dörr är stängd.

Till skillnad från magnetkontakten behöver resten av sensorerna strömförsörjning. Vibrationssensorn som används kallas *SW-18010P*. Sensorn har en liten fjäder innanför sitt metallskal, och när denna fjäder vidrör skalet sluts kretsen. Den reagerar därför på rörelse och vibration från alla möjliga riktningar. I projektet används det för att detektera inbrott, t.ex. att en ruta krossas. Avståndssensorn *HC-SR04* använder ultraljud för att rapportera hur nära något är i dess synfält. Sensorn avger ultraljudsvågor, och mäter hur lång tid det tar dem att komma tillbaka. Tiden och frekvensen av vågorna används för att uppskatta hur långt de färdades. Den kan mäta avstånd från 2 till 400 centimeter med upp till tre millimeter exakthet, och används för att detektera om någon eller något närmar sig.

[9] , [10]

2.4 MD407 - En översikt

Mikrokontrollern MD407 har många anslutningsportar på sitt kretskort. Här beskrivs de portarna som är relevanta för projektet, och var de befinner sig (se figur 2).

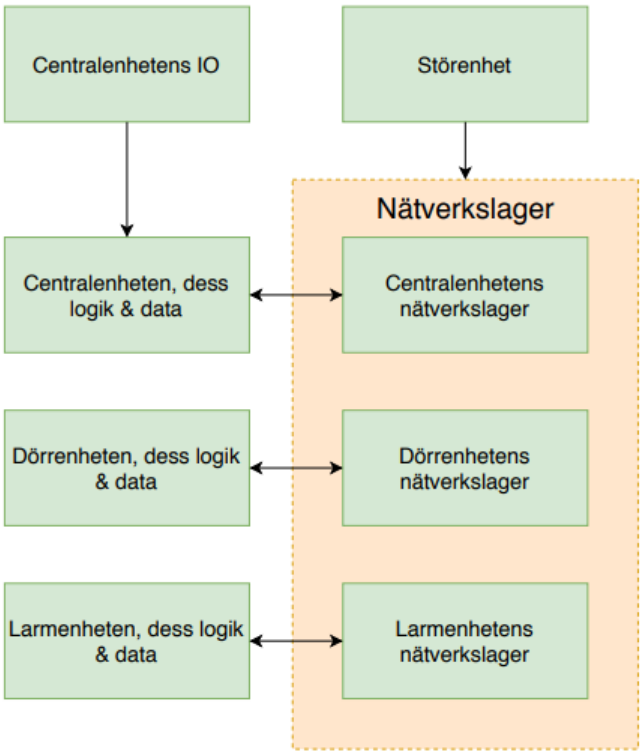


Figur 2: Portar på MD407 som är relevanta för projektet. De gröna cirklarna inuti GPIO-portarna representerar en anslutningspunkt, där en kabel kan anslutas. En anslutningspunkt kan beskrivas som en ”pin” [11].

”CAN-port 1” används för att koppla samtliga enheter till CAN-bussen. GPIO-portarnas nummer refererar till vilka ”pinnar” portarna har: GPIOD 0-7 har pinnar 0-7.

3 Design

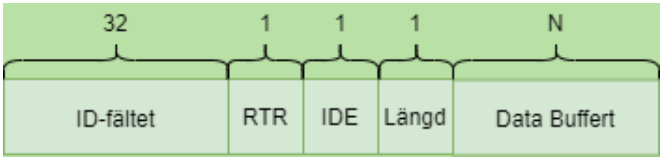
För att säkerhetssystemet ska fungera korrekt måste det klara av en mängd olika uppgifter. Bland annat: kontrollera om en dörr är öppen; avljuda larm; läsa och tolka värden från sensorer, och ta emot inmatning av användare. Systemet delas upp i moduler för att effektivt utföra de nödvändiga funktionerna. Detta underlättar även utvecklingen av projektet. En grafisk översikt av systemet visas nedan i figur 3. Rektanglarna är moduler och pilarna är de funktioner som anropas mellan dem.



Figur 3: Grafisk representation av systemet

3.1 Nätverklagret

Ett CAN-protokoll designades för projektet. Den ram som designades syns i figur 4. ID-fältet innehåller identifieringsinformation för meddelandet. RTR(Remote Transmission Request) innehåller information angående typen av ram. IDE(Identifier Extension) säger vilken typ av identifiering som ramen har. Längd ger längden på databufferten och databuffert innehåller datan i fråga. Databufferten kan som längst vara åtta bytes lång. Utöver protokollet planerades också ett bibliotek för användandet av CAN. Ett sådant bibliotek hjälper uppfyllandet av krav ett under 1.2.1.



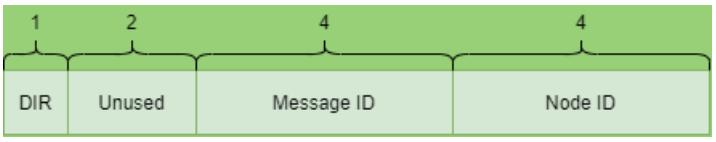
Figur 4: Den designade CAN-ramen

ID-fältet innehåller i sin tur data om meddelandet. Fältet visualiseras i figur 5. Samtliga sektioner i fältet beskrivs nedan:

DIR är meddelandets riktning. Är biten satt till noll är meddelandet ämnat till centralenheten. Är biten ett så är meddelandet från centralenheten.

Message ID är meddelandets ID. Den säger vad det är för slags meddelande som skickats.

Node ID har olika betydelser beroende på vad DIR-biten är satt till. Är DIR-biten satt till noll säger detta fält vilken enhet meddelandet är ämnat för. Men om DIR-biten är satt till ett beskriver fältet vilken nod meddelandet kom ifrån.



Figur 5: CAN-ramens ID-fält, talen beskriver antalet bytes.

Det fanns olika sorter av meddelanden som designades. Dessa syns tabell 1 nedan. Genom att ha olika meddelandetyper blir det lättare att skriva hanteringsfunktioner för CAN-meddelanden då programmet utför rutiner baserat på typerna. Prioriteringen av diverse meddelandetyper är densamma som ordningen i tabellen, där högsta prioriteten är längst upp i tabellen.

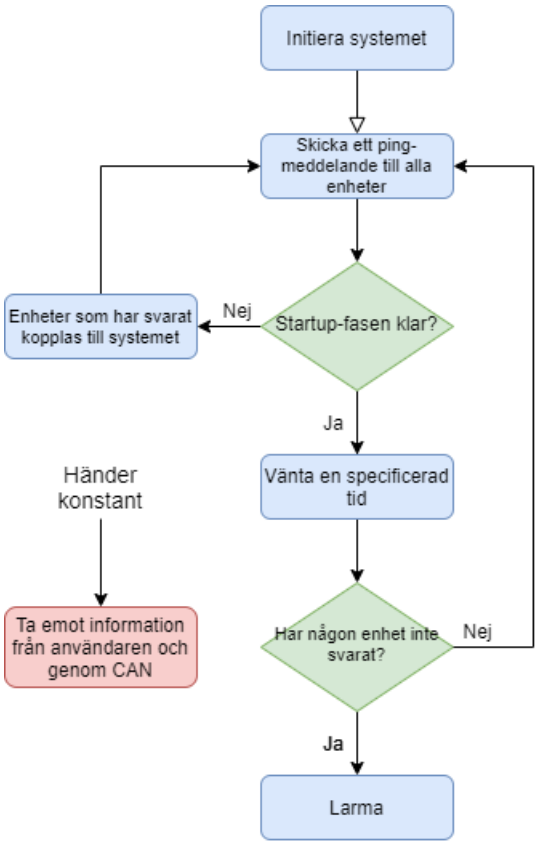
Tabell 1: Alla typer av meddelanden. Högst prioritering längst upp i tabellen.

Typ	Betydelse
Alarm	Säger åt larmenheten att den ska slå larm.
Ping	Ett vanligt pingmeddelande. När en periferienhet får ett pingmeddelande lägger enheten upp ett eget pingmeddelande på bussen som svar.
Konfigurering	Innehåller nya konfigurationer till en enhet.

3.2 Centralenheten

Om man sammanfattar centralenhetens uppgift ska den fungera som administratör för alla enheter. I 1.2.1 nämns det att den ska lagra aktuell information

om dessa och konfigurera om dem, men också att den ska larma om någon enhet blir frånkopplad. För att uppfylla dessa krav designades centralenhetens logik efter flödesschemat som visas i figur 6. Datahanteringen utvecklades enligt kravspecifikation två, och logiken gällande enheterna enligt kravspecifikation fem.



Figur 6: Flödesschema för centralenhetens logik

I schemat beskrivs det att data konstant tas emot från användaren och CAN. Denna information handlar mest om de kopplade enheterna, och behöver lagras, både för användaren och systemet i helhet. Därför använder sig centralenheten av en lista, där varje plats i den innehåller information om en kopplad modul. Informationen inuti listan är en ”struct”, en datarepresentation av modulen. Dessa ”structs” underlättar när specifik information ska hämtas gällande modulerna, och detta görs väldigt ofta i systemet.

Schemat beskriver också centralenhetens logik; hur centralenheten kopplar enheter till systemet och detekterar om någon av dem blir frånkopplade. När uppstarts-fasen är klar är alla enheter anslutna till systemet, förutsatt att de svarade på ping-meddelandena som skickades. Centralenheten börjar då regelbundet skicka ut ping-meddelanden, väntar en specificerad tid, och undersöker

om någon enhet inte har svarat i tid. Då larmar centralenheten globalt. Det här säkerhetsställer att inga delar av systemet kan kopplas bort utan att ett larm går.

Uppstarts-fasen är designad med avseende på modularitetskravet (krav åtta). Systemet är inte ”låst” till en speciell uppsättning enheter; alla som svarar på centralenhetens ping-meddelanden blir anslutna. Användningen av listan med ”structs” möjliggör denna process. Teoretiskt sett kan en godtycklig mängd enheter av varierande typ kopplas till systemet. Detta underlättar också för användaren - systemets delar är anslutna så fort uppstart-fasen är färdig, utan någon konfiguration.

3.3 Centralenhetens IO

Centralenheten IO består av en knappsats och USART. IO funktionerna för dessa utvecklades enligt kravspecifikation 3, 4 och 7 (se 1.2.1). Därmed är knappsatsen designad för ta emot en sifferkod samt att lagra den för centralenhetens användning. USART utformades dels för att skicka utmatningar från centralenheten som presenteras för användaren. USART designades även för att ta emot inmatningar från användaren vilket tillåter konfigurationer och status meddelanden. Konfigurationerna sker via olika kommandon som användaren kan skriva in i USART.

Både knappsatsen och USART utvecklades för att inte avbryta programmet, vilket tillåter ett kontinuerligt program där alla funktioner kan köras vid alla tidpunkter. Detta krävs för en korrekt fungerande centralenhet då ping meddelanden måste mottas och hanteras konstant. (3.2, figur 6).

3.4 Dörrenheten

Säkerhetssystemet designades för att kunna hantera flera dörrar. Ett alternativ hade varit att ha en dörr per MD407-kort. Dock, eftersom att ett MD407-kort har förmågan att hantera ett flertal dörrsensorer utnyttjas det. Då blir det enklare samt effektivare att koppla upp ett större antal dörrar. Antalet uppkopplade dörrar bestäms i centralenheten. Varje dörr förses med ett unikt ID för att kunna skilja dem åt samt konfigurera dem separat. Konfigurationen sker genom att det skickas och mottages meddelanden över CAN med hjälp av CAN-protokollet i enlighet med krav ett och fyra (1.2.1).

En dörrs larmfunktion kan aktiveras och avaktiveras. En avaktiverad dörr signaleras med en grön diod. Enligt kravspecifikationen har dörrenheten både lokalt och globalt larm. Det lokala larmet, en röd diod, går efter dörren varit öppen längre än en given tid. Det används för att inte larma globalt i onödan. Exempelvis i fallet att dörren öppnas av misstag och sedan omedelbart stängs igen. Meddelande om globalt larm skickas till centralenheten då dörren varit öppen längre än en ytterligare angiven tidsgräns. Centralenheten avgör sedan hurvida larmenheten ska tjuta. Hur lång tid innan globalt samt lokalt larm ges kan konfigureras via CAN.

3.5 Larmenheten

Larmenhetens huvudsakliga uppgift är att utlösa ett ljudbaserat larm om rätt krav möts. Dessa krav är antingen att centralenheten skickar ett meddelande över CAN om att utlösa larmet eller att larmenhetens sensorer har mätt ett visst värde. Sensorerna består utav en vibrationssensor och en avståndssensor som skickar mätdata till larmenheten.

Kommunikationen från Centralenheten via CAN-bussen har högsta prioritet eftersom även centralenheten snabbt måste kunna utlösa larmet vid behov. Det har ordnats genom att mottagna meddelanden skapar avbrott i systemet. Avbrottshanteraren för vidare meddelandet till en hanteringsfunktion beroende på vilken typ av meddelande det är. De meddelandetyper som hanteras är konfigureringsmeddelanden enligt krav fyra (1.2.1), meddelanden för att växla av/på larmet och pingmeddelanden.

Enligt krav sex (1.2.1) har larmenheten en utbyggnad i form av ett lokalt larm. Detta lokala larm består av en lampa som tänds när någon eller något närmar sig avståndssensorn. Syftet för utbyggnaden är att det ska avråda folk från handlingar som skulle leda till ett globalt larm.

3.6 Störenheten

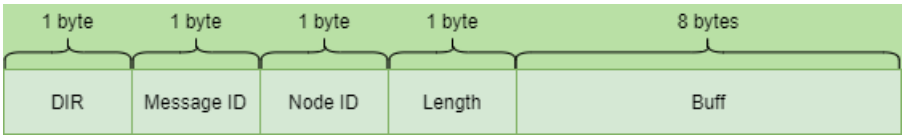
Störenheten utvecklades för att simulera en defekt periferienhet genom att skicka stora mängder data på CAN-nätverket. Enheten kan skicka faktiska meddelanden eller slumpad data. Förmågan att skicka båda typer utvecklades för att ha möjlighet att testa en bredare mängd problem som skulle kunna uppstå.

4 Teknisk beskrivning

Denna sektion beskriver hur projektets delsystem implementerades utefter systemets design och krav. Systemet är uppdelat i moduler enligt figur 3. Nedanstående rubriker går in i mer detalj om respektive modul.

4.1 Nätverkslagret

För att standardisera hur meddelanden skickas och tas emot skrevs ett CAN-bibliotek. Biblioteket innehåller funktioner som skickar samt hanterar mottagande av CAN-meddelanden. Biblioteket kan användas på alla periferienheter men funktioner för att hantera enhetsspecifika meddelanden måste manuellt implementeras.



Figur 7: Structen som representerar ett CAN-meddelande.

I koden skapades också en struct som representerade ett CAN-meddelande. Denna struct bestod av fyra bytes av riktnings- och destinationsdata samt längd på databufferten. Efter dessa finns det en buffert på åtta bytes för resterande data. Det första byte-segmentet beskriver huruvida meddelandet kommer från eller till centralenheten, det andra beskriver meddelandets typ, och det tredje representerar CAN-ramens ID och det fjärde talar om längden på meddelandets databuffert(i bytes). En visualisering av structen finns i figur 7.

Meddelandetyperna, som nämns i tabell 1, har definierats som konstanta heltalsvärden i koden. Prioriteringen av meddelandetyperna räknas ut genom en summering av ID för noden i meddelandet och ID för meddelandetypen.

4.2 Centralenhetens logik och datahantering

När systemet startas aktiverar centralenheten en ”uppstarts-fas”. Den inleds med att centralenheten skickar ut ett ping-meddelande till alla enheter. När centralenheten märker att den har fått ett svar extraherar den informationen från meddelandet, skapar en datarepresentation av modulen, och lägger in den i en lista. Informationen som läggs in är bland annat modulens unika ”id” och modulens typ, etc. Om en modul inte rapporterar inom tidsramen för uppstarts-fasen läggs den inte in i listan. Då ignoreras följande meddelanden från denna modul.

Efter uppstarsfasen är klar skickar centralenheten ping-meddelanden till alla enheter varje sekund. Om en av enheterna inte hinner svara innan nästa meddelande skickas, sänder centralenheten en larmsignal till alla larmenheter. En larmsignal skickas också om centralenheten märker att dörrenheten larmar, och dörren i fråga är aktiverad.

För att lagra information gällande modulerna använder sig centralenheten av en speciellt utformad datatyp. Datatypen och dess fält beskrivs i figur 8.

Datatypen "MODULE"
nodeid - Modulens unika identifikationsnummer
type - Enhetstypen (T.ex. MODULE_ALARM)
connected - Är enheten fortfarande kopplad?
Konfigurationsfält
armed - Är enheten aktiverad, kan den larma?
cfg1
cfg2
cfg3
cfg4

Figur 8: Datatypen ”MODULE”, gröna fält är konfigurationsfält.

Enhet	Dörrenheten	Larmenheten
cfg1	doorid	triggered
cfg2	doortime_glb	distancecfg
cfg3	doortime_loc	-
cfg4	-	-

Figur 9: Modulernas konfigurationsfält, ’-’ betyder att fältets värde är odefinerat.

Konfigurationsfälten beskriver modulens nuvarande inställningar. Det första fältet beskriver om enheten är larmad eller inte, och resten beror på vilken modultyp enheten är. I figur 9 beskrivs det vad ”cfg1” till ”cfg4” representerar för detta projektets moduler.

Dörrenhetens ”doorid” anger hur många dörrar som dörrenheten är konfigurerad för. De följande fälten, localtime och globaltime är hur länge en dörr kan vara öppen innan dörrenheten slår lokalt respektive globalt larm. När det gäller larmenheten är ”cfg1” ”triggered”, och beskriver om larmet är triggat. Det andra fältet representerar vilket avstånd som avståndsmätaren ska larma vid. Värdet i detta fält kan vara från 1-255, mätt i centimeter.

4.3 Centralenhetens IO

Centralenhetens IO-enheter knappsatsen och USART har sitt eget buffertsystem. Ett buffertsystem hanterar en inmatning ifrån USART och knappsatsen genom att lagra endast en bokstav eller siffra i bufferten. Programmet går sedan vidare för att kolla om det finns indata från andra källor. Därefter kan

inmatningen fortsätta, vilket tillåter programmet att köra alla sina funktioner kontinuerligt.

Konfiguration av programmet sker via USART-kommandon som konfigurerar modulernas register (se 4.2 figur 9 för lista av alla register). Ett kommando har strukturen [funktion] och potentiellt [modul ID], där funktion står för följande funktioner:

- arm - Armerar en modul genom att sätta modulens armed register till ett givet en modul-ID.
- disarm - Desarmerar en modul genom att sätta modulens armed register till noll givet en modul-ID.
- status - Returnerar all information om en modul vilket inkluderar en lista över modulens register och deras innehåll givet en modul ID.
- config - Tillåter användare att konfigurera en moduls register givet en modul-ID. Detta sker genom att efter användaren skrivit in ”config [Modul-ID]"uppmanas de att skriva in information till varje register i modulen med kommatecken emellan, t.ex ”0,15” för en larmenhet.
- printlist - Returnerar en lista av alla kopplade modulers typ och ID.
- help - Returnerar en lista av alla möjliga kommandon i USART och deras parametrar.

Processen från en inmatning till ett exekverat kommando börjar med att användaren skriver in bokstäver i USART vilka lagras i bufferten. När användaren trycker ”retur” skickas de bokstäver som har lagrats i bufferten vidare till en funktion som bearbetar inmatningen. På grund av kommando strukturen delas inmatningen upp i två delar, ett ord ”funktion” och ett ”modul-ID”. Det inskrivna ordet jämförs sedan med en lista med möjliga kommandon. Om det finns i listan körs det kommandot med hjälp av modul-ID, annars avbryts kommandot.

Vid larm aktiveras knappsatsen där användaren kan skriva in koden. Inmatningen lagras i knappsatsens buffer tills lika många siffror som längden på lösenordet har tagits emot. Därefter verifieras koden och om rätt kod har angetts avlarmas systemet. Användaren får upprepade försök att skriva in rätt kod. Lösenordets inmatning kan återställas genom att trycka på C (clear) på knappsatsen.

4.4 Dörrenheten

När dörrenheten kopplas upp för första gången körs ett antal initierande funktioner. En av dem aktiverar SysTick som används för tidsavläsning genom att ett värde ökas varje gång SysTick ger avbrott. Systick ger avbrott varje givet antal klockcykler, vilket anges i initialiseringen. När en dörr med larm aktiverat öppnas sätts den dåvarande tiden som en tidsstämpel. Programmet granskar därefter med jämna mellanrum om dörren varit öppen längre än tillåtet och larmar då lokalt eller globalt beroende på konfigurationen. Om dörren stängts innan tidsgränsen nåtts nollställs tidsstämpeln och eventuellt lokalt larm stängs av. Ett globalt larm som gått måste stängas av från centralenheten.

För att klara av att hålla reda på hurvida dörrarna är öppna eller stängda använder den en uppsättning dörrsensorer som ger information om dörren är öppen eller stängd. Dörrenheten håller reda på hur länge sedan sensorns krets bröts och beroende på det utlöser den passande larm.

En struct för dörrar har skapats. Dörrstructen innehåller följande data.

- Dörrens ID
- Vilken GPIO-pinne dörren är kopplad till
- Om den är öppen eller stängd
- Röd diodstatus
- Larmfunktion på-/avslagen
- Tidsgräns för globalt larm
- Tidsgräns för lokalt larm
- Tidsstämpel

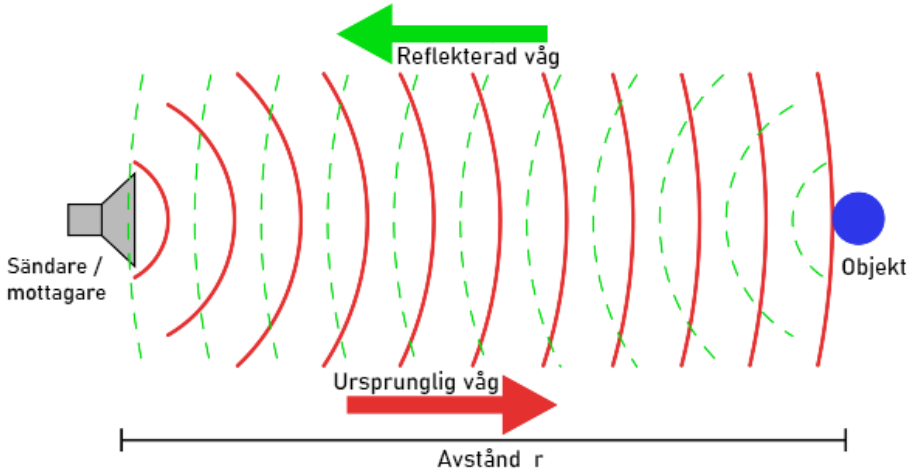
Upp till åtta dörrar stöds. För fler än åtta dörrar behövs flera dörrenheter. När programmet startas initieras alla åtta dörrar med standardvärden. När centralenheten sedan skickar det första konfigurationsmeddelandet (se 4.2) konfigureras alla dörrar upp till den givna dörren efter inställningarna i meddelandet. Alla dörrar som då inte konfigureras tas då bort ur systemet.

4.5 Larmenheten

Larmenhetens inportar är konfigurerade till pinnarna GPIOE0-15. Därför kopplas avståndssensorns ekosignal samt vibrationssensorns utsignal via dessa pinnar. GPIOD0-15 är reserverade som utportar, de kopplas till avståndssensorns inport och lokala larmets lampor.

Larmenhetens programlogik är uppbyggd för att lyssna på sensorerna och centralenheten kontinuerligt och märka om ett larm måste utlösas. Vibrationssensorn är direkt kopplad till larmfunktionen i och med att tröskelvärdet konfigureras analogt. Till skillnad från vibrationssensorn måste programlogiken hantera både avståndssensorns in- och utdata. För att aktivera avståndssensorn skickas en tio mikrosekunders högnivåsignal till sensorns inport (markerad Trig"). Ekot mäts därefter och ger en högnivåsignal på utporten (markerad "Echo") proportionell till tiden pulsen färdas (se figur 10). Tidsintervallet jämförs sedan med värdet “time_to_trigger_sound” som bestäms från det kalibrerade eller konfigurerade värdet från centralenheten. Om tiden som mätts är mindre än “time_to_trigger_sound”, dvs objektet är närmare än det konfigurerade avståndet utlöses larmet.

Utöver detta tänder larmenheten en varningslampa så fort något närmar sig avståndssensorn. Avståndet för att tända lampan är dubbelt så långt som det konfigurerade värdet från centralenheten.



Figur 10: Avståndssensor. Avståndet räknas ut med: $r = (t * v_{ljud})/2$,
t = tiden av högnivåsignalen från echo, v_{ljud} = ljudets hastighet (343m/s).

Ifall larmenheten mottager ett meddelande från centralenheten hanteras det omedelbart med ett avbrott. Avbrottshanteraren för vidare meddelandet till en hanteringsfunktion beroende på vilken typ av meddelande det är. De meddelandetyper som hanteras är konfigureringsmeddelanden, meddelanden för att växla av/på larmet och pingmeddelanden. Ett konfigureringsmeddelande innehåller:

- En byte som bestämmer om larmet ska vara armed (se figur 8).
- En byte som bestämmer om larmet ska slå larm/sluta slå larm direkt (se figur 9).
- En byte som antingen kan:
  - Ge en order att kalibrera sensorn efter de objekt som finns framför den i nuläget, detta sker när byten är noll.
  - Ge ett bestämt avstånd som avståndssensorn ska konfigureras efter.

När ett pingmeddelande är mottages skickas ett svarsmeddelande tillbaka. Dessutom sparar funktionen tidspunkten då larmenheten senast tog emot ett pingmeddelande. Eftersom larmenheten är den enda enheten i systemet som kan producera ett larm måste det finnas någon motåtgärd om larmenheten skulle kopplas ifrån systemet. Den motåtgärden består av att om larmenheten slutar tar emot ping-meddelanden från centralenheten i åtminstone en sekund larmar enheten automatiskt, dock är det värt att notera att centralenheten inte får ett meddelande om larmet är igång.

4.6 Störenheten

Störenheten har förmågan att skicka både korrekt strukturerade CAN-meddelanden eller slumpade datasträngar. Mängden data och frekvensen på meddelanden är

inte konfigurerbara genom systemet utan måste föranpassas i koden. Detta anses acceptabelt då enheten är enkel och endast används när tester utförs på systemet.

4.7 Användning

Programmet startar med en ”uppstartsfas”, där alla moduler kopplas till programmet automatiskt (se 3.2). Alla enheter är initialt icke-armerade, vilket betyder att de inte kan användas. För att armera modulen behövs USART-kommandon användas (se 4.3). En användare som aldrig har använt programmet tidigare kan t.ex först skriva ”help” för att se möjliga kommandon. Sedan kan användaren skriva ”printlist” för att se vilka moduler är kopplade och deras ID. Slutligen kan användaren börja konfigurera eller armera/disarmera modulerna med ”arm/disarm [modul-ID]” eller ”config [modul-ID]”, där [module-ID] är en av ID värdena från printlist.

När ett lokalt larm utlöses tänds en lampa, och på dörrenheten startas en timer. För att larma av innan ett globalt larm utlöses behöver användaren skriva in rätt pin-kod med knappsatsen. Om användaren skriver fel får den återkoppling via USART, och får ett nytt försök. Om det skrivs in mindre än fyra siffror kan användaren fortfarande börja om genom att trycka på knappen ”C” (clear).

För att ansluta dörrar till en dörrenhet skall dörrsensorer kopplas till GPIOE 0-7, röda dioder till GPIOD 0-7 och gröna dioder till GPIOD 8-15. Om färre dörrar än åtta vill anslutas skall dessa anslutas till de lägra pinnarna. Sedan konfigureras antalet dörrar genom USART. Om t.ex två dörrar ska anslutas kopplas två dörresensorer till GPIOE 0-1, Två röda dioder kopplas till GPIOD 0-1 och två gröna dioder kopplas till GPIOD 8-9.

5 Resultat

Ett säkerhetssystem har utvecklats med hjälp av tre mikrokontroller och ett antal sensorer. En mikrokontroller fungerar som centralenhet, denna kommunicerar med två periferienheter som i sin tur har uppkopplade sensorer. Säkerhetssystemets funktionalitet har verifierats med ett flertal tester. Testerna utfördes på enhetsnivå och på systemet som helhet. De utförda testerna visar att systemet fungerar stabilt med pålitlig kommunikation mellan enheter och sensorer, samt att sensorerna kan konfigureras så de agerar som önskat. Det övergripande testet, T17 (bilaga 17), som presenteras nedan visar översiktligt hela systemets funktionalitet, och mer detaljerade resultat finns under rubrikerna nedan.

5.1 Nätverkslagret

CAN-nätverket som helhet fungerar väl. Meddelanden tas emot och hanteras korrekt. Prioritering av meddelanden fungerar utan krockar enligt test T5 (bilaga 5). Systemet klarar av när en störenhet skickar stora mängder data på CAN-bussen. I test T6 (bilaga 6) visas att systemet klarar av när en störenhet skickar ping-meddelanden konstant utan fördröjning så länge systemet skickar riktiga meddelanden mer sällan än 500 gånger per sekund.

En störenhet som skickar slumpade meddelanden kan systemet inte hantera då viss data som skickas inte är av samma struktur som ett faktiskt meddelande. Detta är en identifierad svaghet då CAN-protokollet är okrypterat. Dessa resultat visar att systemet uppfyller krav ett enligt 1.2.1.

5.2 Centralenheten

Test T1 (bilaga 1) visar att centralenheten korrekt skapar en lista med alla uppkopplade enheters ID vid startup-fasen (se 4.2). Detta test verifierar att systemet ansluter enheter modulärt, men visar även att kommunikationen mellan de anslutna enheterna fungerar som den ska. Enheterna svarar punktligt på alla ping-meddelanden som centralenheten skickar ut, och när en enhet blir frånkopplad går larmet. Dessa tester pekar på att systemet uppfyller krav ett, fem, och åtta enligt 1.2.1.

T19 (bilaga 19) visar att information om enheternas konfigurationer lagras och uppdateras korrekt, enligt krav fem. Att meddelanden från enheter med okända ID-nummer ignoreras påvisas i T6 (bilaga 6). Därav uppfylls krav två enligt 1.2.1.

Centralenhetens IO-funktioner har förmågan att utföra samtliga kommandon begärda från USART. Dessa inmatningar används av centralenheten för att konfigurera periferienheterna eller begära information om systemet. Denna funktionalitet visades i T15 (bilaga 15). Knappsatsen brukades för att stänga av ett aktivt larm, och detta fungerade vid upprepade försök i test T10 (bilaga 10). Dessa tester visar att krav tre är uppfyllt enligt 1.2.1.

5.3 Dörrenheten

Test T12 (bilaga 12) visar att dörrenheten kan larma lokalt. Testet visar också att en dörr kan tända en diod för att markera att den är ställd till att inte larma. Test T13 (bilaga 13) visar på att dörrar kan ställas att larma eller inte från centralenheten. Slutligen visar test T15 (bilaga 15) en större överblick på att funktionaliteten är korrekt. Där demonstreras att tidsgränser går att konfigurera och att enheten kan utlösa det globala larmet. Dessa resultat visar att dörrenheten uppfyller krav fyra och sex enligt 1.2.1.

5.4 Larmenheten

Avståndssensorn och vibrationssensorn är funktionella. Avståndssensorn larmar lokalt när något kommer närmre än ett konfigurerat avstånd, vilket visas i test T2 (bilaga 2). T4 (bilaga 4) visar att det globala larmet aktiveras korrekt när avståndet är under ett andra tröskelvärde. Systemtestet T17 (bilaga 17) visar att larmenheten kan konfigureras av centralenheten eller själv kalibrera larmavståndet på order från centralenheten. Resultaten ovan visar att larmenheten uppfyller krav fyra och sex enligt 1.2.1.

Test T7 (bilaga 7) visar att om inget pingmeddelande från centralenheten är mottaget inom önskat tidsintervall utlöses larmet. Det globala larmet kan även stängas av och på från centralenheten. Detta visas i test T8 (bilaga 8). Dessa resultat visar att larmenheten uppfyller krav fem och sju enligt 1.2.1.

Det utfördes även ett test på avståndssensorns felmarginal, T4 (bilaga 4), vilket visade att det uppmätta avståndet var mycket konsekvent och hade en felmarginal på under en centimeter.

6 Diskussion

Slutresultatet uppnår de krav som sattes på arbetet, se 1.2.1. Larmenheten, dörrenheten och centralenheten arbetar tillsammans för att uppfylla de krav som ställdes för ett fungerande lås- och larmsystem. Centralenheten övervakar enheternas status regelbundet. Vid otillåtet intrång skickar dörrlarmet respektive rörelselarmet en signal att larma. Kommunikationen mellan enheterna sker på ett konsekvent sätt med det designade CAN-protokollet. Centralenheten lägger till enheter på ett modulärt sätt, enligt krav åtta. Däremot är det inte perfekt då ID-nummer inte delas ut dynamiskt utan är hårdkodade hos enheterna. Trots detta är systemet på bra väg att vara modulärt.

Utöver grundsystemet utfördes också två utökningar av rörelselarmet. Rörelselarmet larmar lokalt inom ett visst distansintervall från rörelesesensoren och sedan globalt då intervallet överskrids. Sedan har dessutom en distinkt larmsignal framställts.

6.1 Struktur

Under projektets utveckling har det lagts en viss vikt på att generalisera hanteringen av modultyperna. Denna generalisering underlättas mest av modulstructen som beskrivs i sektion 4.2. Att använda en gemensam datatyp för modultyperna gör det enklare att samla alla enheters information på samma ställe, i en sorts lista. Listan med enheter skulle kunna filtreras, sorteras och sökas i, vilket är en stor fördel. Om systemet skulle utökas blir dessa funktioner praktiska, mest för användaren.

När information om en specifik modul ska uppdateras är användingen av en lista lämplig; listan söks igenom, och enhetens modul-struct uppdateras. Detta gör det möjligt för centralenheten att lagra aktuell information om alla enheter, enligt krav två i 1.2.1. Att samtlig information om enheterna lagras på samma plats underlättar också för utvecklare. En ny modultyp skulle kunna läggas till i systemet utan att behöva ändra på projektets struktur.

Denna struktur gör dock systemet mer komplicerat, och detta kan ha negativa effekter. En komplex struktur gör det svårare att identifiera misstag eller logiska fel. Därför kan systemets säkerhet och funktionalitet bli svårare att verifiera. Men om ett komplext systems kod är en välskriven och verifierad bas för systemet främjar detta framtida utveckling. Då underlättar den befintliga koden förståelse och vidareutveckling av systemet.

En stor förbättring när det gäller generalisering skulle vara att skapa en universell modultyp för sensorer. I det nuvarande läget har larmenheten hand om vibrationssensorn och rörelsesensorn, medan dörrenheten har hand om flera magnetkontakter (eller dörrsensorer). Om en sensormodul implementeras skulle den kunna hantera alla sensorer som tidigare var kopplade till larmenheten och dörrenheten samtidigt, och utöver det skulle den kunna hantera ännu fler sensortyper. Modulens datahantering kan implementeras som centralenhetens: en generell struct för sensorer finns, och information lagras om dessa i modulens minne. Alla funktioner som är specifika för att konfigurera samt avläsa

sensorerna skulle då kunna skrivas separat, och importeras av modulen som ett programbibliotek. Detta underlättar processen att lägga till en ny sensortyp.

6.2 Funktionalitet och IO

Funktionellt skall systemet tillåta användaren att påverka systemet samt presentera information och visa resultatet av användarens påverkan. Användaren har två olika sätt att interagera med systemet: genom USART-kommandon och genom en knappsats. Detta tillåter att alla perifirienheter kan konfigureras fullständigt vilket täcker kraven på funktionalitet.

Systemets förmåga att ge användaren feedback är i nuläget begränsad. Centralenheten begär inte information från periferienheterna när de konfigureras. Därmed kommer ingen automatiskt utdata om ändringarnas effekt. Den lokala statusen kan fortfarande ses manuellt med USART-kommandon, men den kan vara olika från statusen på periferienheterna. Det uppnår kraven på funktionalitet, men operationen kan förenklas och optimeras.

Projektets moduler läggs nu till automatiskt vid uppstart, förutsatt att alla periferienheter har sina ID-nummer fördefienerade. Denna process är dock inte helt konfigurationsfri; användaren behöver defienera alla periferienheters ID i respektive källkod innan koden laddas. Detta skulle kunna förbättras och en utveckling av systemet är därför att implementera automatisk ID-utdelning för att göra det mer användarvänligt. Vid uppstart skulle centralenheten då dela ut ett unikt ID-nummer till alla enheter. Detta skulle göra systemet helt konfigurationsfritt och användaren skulle bara behöva koppla in de nödvändiga enheterna och trycka ”start”.

Om den automatiska ID-utdelningen implementeras kan man anse att systemet är fullständigt modulärt. Det kan då kopplas flera dörr- och larmenheter till systemet, och det antalet moduler som kan vara kopplade samtidigt kan ändras. Ett icke-modulärt system, som har ett fast antal enheter, kan vara enklare att använda och det kan också vara säkrare. Att inte ha en uppstartfas som lägger till alla tillkopplade enheter minimerar risken att en oönskad modul kan läggas till i systemet, och att ändra på systemet kanske blir svårare eftersom CAN-systemet inte skulle vara lika generellt. Den negativa sidan med ett sådant system är så klart att det inte är modulärt.

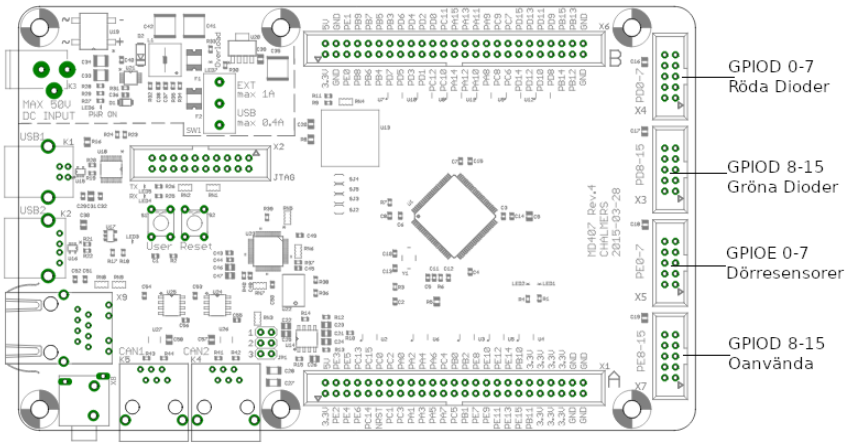
6.3 Delsystemen

Nätverkslagret tillfredsställer i slutändan projektets krav för kommunikation. Enheterna kan kommunicera genom en gemensam CAN-buss där meddelandekollisioner löses utav det skapade CAN-biblioteket. Genom prestandatester som beskrivs i 5 säkerställs, med hög sannolikhet, att nätverkslagret är robust för sabotage och tekniska fel. Däremot hade förbättringar kunnats göra som exempelvis att kryptera datan som skickas över bussen. Detta nämns mer i 6.4. En annan förbättring hade kunnat vara att inkludera ett så kallat *ACK-fält* i CAN-ramen. Det denna gör beskrivs i 2.2. Att ha detta fält hade kunnat säkerställa att meddelanden som läggs ut på bussen faktiskt når den avsedda mottagaren. Då

hade potentiellt snabbheten hos CAN-meddelandena försämrats då mer information måste utbytas. Däremot bör denna prestandaförändring vara oväsentlig.

Larmenheten hanterar dess uppgifter enligt ställda krav. Avbrott prioriteras och utför ändringar omedelbart utan att påverka prestandan av sensorerna. En potentiell utökning av antalet möjliga sensorer av samma typ skulle gynna systemet i hög grad. Det skulle kräva en omstrukturering av GPIO-pinnarna samt att alla sensorfunktioner i programlogiken måste modifieras för att hantera fler än bara en modul i taget. Det skulle resultera i en drastisk minskning av antal MD407-kort för fullständiga system, med en potentiellt minimal försämring av prestanda.

Dörrenheten är satt att hantera max åtta dörrar. Ett MD407-kort har 32 GPIO-pinnar. Varje dörr kräver tre, en för dörrsensorn och två för dioderna. Den fysiska begränsningen är alltså tio dörrar. Valet att ha max åtta dörrar gjordes med anledning av enkelhet för användaren. På MD407-kortet är pinnarna uppdelade i fyra grupper, pinnarna 0-7 och pinnarna 8-15 för både GPIOE och GPIOD, se bild 11. Det finns alltså plats för två dörrsensorer och fyra dioder till på GPIOE 8-15. Nackdelen med det blir att användaren som skall koppla upp systemet får en mindre intuitiv och rörigare uppkoppling då en GPIO-port kommer vara både utport och inport, alltså ha både dörrsensorer och dioder. Därav valdes åtta dörrar.



Figur 11: GPIO-pinnarnas användning på dörrenheten[11]

6.4 Säkerhet

Vissa åtgärder togs för att förbättra säkerheten. Alla funktioner i programmet kontrollerar sin indata innan något görs med den för att säkerställa att inget oförutsägbart sker i programmet. Utöver det kan moduler bara läggas till i

modullistan under uppstarts-fasen, och efter det ignoreras alla mottagna meddelanden med okända nodID-nummer. Detta ger en viss säkerhet mot intrång i systemet, men garanterar inte detta till 100 %.

I det nuvarande läget finns det många möjligheter att utveckla systemets säkerhet. Projektets implementation av CAN är funktionellt, men inte säkert. De säkerhetsåtgärder som vidtogs för modullistan nämns i stycket ovan, men skyddet som uppförs av dessa kan enkelt kringgås. Om en enhet ansluts till nätverket kan den skicka meddelanden i en annan enhets namn. Allt som krävs är att den externa enheten skickar ett meddelande med samma ”nodeid” som en modul i modullistan. Denna enhet kan antingen gissa sig fram till detta ID, eller extrahera informationen ur ett CAN-meddelande från bussen.

Anledningen varför alla anslutna enheter kan läsa informationen på bussen är för att meddelandena inte är krypterade. Att implementera kryptering i kommunikationssystemet skulle vara ett stort steg för systemets säkerhet. Det skulle göra det betydligt svårare för en enhet att skapa meddelanden som följer projektets kommunikationsstandard, förutsatt att enheten inte har tillgång till den informationen. Attacker där en enhet kopierar ett redan krypterat meddelande och skickar det igen är svårare att hantera. Det finns lösningar till det problemet, men de är avancerade.

När det gäller IO kan säkerheten utvecklas. Ett användarsystem för USART med olika behörighetsnivåer skulle kunna implementeras. Där skulle en administratör exempelvis kunna hantera vilka pin-koder som larmar av, och lägga till nya användare på en lägre eller lika hög behörighetsnivå.

6.5 Utvärdering av tester

De flesta av de utförda testerna var funktionella. Alltså testade de att en viss funktion fungerade korrekt. De minsta möjliga funktionerna testades först. När tester på mer omfattande funktioner gjordes var det då redan säkerställt att eventuella fel inte låg på underliggande kod. Det förenklade felsökningen och utvecklingen.

Endast en handfull mängd prestationstester utfördes och de var relaterad till störenheten, CAN-bussen och rörelsesensorn. Om fler prestationstest hade utförts hade systemets gränser varit bättre definierade. Detta hade förtydligat förbättringmöjligheter och säljpunkter för systemet. Ett säkerhetstest med en tredje part hade kunnat utföras för att värdera säkerheten och upptäcka svagheter i systemet. Det har dock inte utförts i brist av tid.

För kod skrevs tester för att bevisa korrekt funktionalitet. De testerna skrevs efter att given icke-trivial funktion var klar. Om testerna hade gjorts i förväg hade de varit användbara verktyg för utvecklingen. Det är också oklart hur mycket tid som hade sparats när man räknar med skapandet av testerna. Dessutom låg svårigheten generellt inte i mjukvaran utan i hårdvaran som testades med funktionalitetstester.

7 Slutsats

Arbetet att utveckla ett säkerhetssytem med mikrokontrollern MD407 anses vara lyckat. Det skapade säkerhetssytemet uppfyller de krav som ställts i 1.2.1. Systemet är dessutom utökat med en svårmissad larmsignal och förmågan för larmenheten att larma lokalt eller globalt beroende på hur nära något är. Systemet levererades i tid.

Systemet är modulärt på det sättet att inläggning av enheter vid uppstart sker automatiskt vilket är i enlighet med krav åtta (1.2.1). Enheternas identifikationsnummer måste dock definieras av användaren i koden. Att lägga till automatisk ID-utdelning vid uppstart skulle vara en bra vidareutveckling av systemet. En gemensam datatyp brukas även för alla enheter i centralenheten. Detta gör det möjligt att samla deras information på samma ställe och underlättar även framtida enhetsutökningar. Enhetslistan kan sökas igenom och uppdateras. Likaså har CAN-funktionerna generaliserats, utom de som tolkar meddelandena då det för dessa ibland är nödvändigt att integrera dem med enhetens egna logik.

Testningen utfördes först på mindre funktioner, varpå större och större stycken testades tills dess att hela systemet testats. Mer tester hade kunnat göras och mer vidareutvecklingsmöjligheter hade då kunnat hittas. Tester kunde även ha konstruerats i förväg som gjort det möjligt att spara mer tid.

All skriven kod är välkommenterad och följer samma kodkonvention och systemet är tämligen modulärt. Allt det här underlättar och möjliggör för ytterligare utvecklingar av systemet.

Referenser

[1] databus. (n.d.), "Collins english dictionary – complete and unabridged, 12th edition 2014," 2014, [Online] Tillgänglig: https://www.thefreedictionary.com/databus hämtad: 2020-10-04.

[2] N. M. Kevin M.Lynch and M. L. Elwin, *"Embedded Computing and Mechatronics with the PIC32"*. Newnes, 2016.

[3] P. Christensson, "Ping definition," 2018, [Online] Tillgänglig: https://techterms.com/definition/ping hämtad: 2020-10-03.

[4] D. Howe, "Usart definition," 2020, [Online] Tillgänglig: https://encyclopedia2.thefreedictionary.com/USART hämtad: 2020-10-04.

[5] S. L. Maria Molin and Åsa Irlander Strid, "Nationella trygghetsundersökningen 2019," 2019.

[6] S. centralbyrån, "Drygt 4,8 miljoner bostäder i sverige," 2018. hämtad från, [Online] Tillgänglig: https://www.scb.se/hitta-statistik/statistik-efter-amne/boende-byggande-och-bebyggelse/bostadsbyggande-och-ombyggnad/bostadsbestand/pong/statistiknyhet/bostadsbestandet-2017-12-31/ hämtad: 2020-10-28.

[7] J. Åke Lundh, *"LARM övervaknings- och säkerhetssystem Faktabok"*. ATHENA lär AB, 2008.

[8] S. Corrigan, *"Introduction to the Controller Area Network (CAN)"*. Texas Instruments Incorporated, 2016.

[9] "Ultrasonic ranging module hc-sr04," inget år, [Online] Tillgänglig: http://www.elecfreaks.com/store/download/product/Sensor/HC-SR04/HC-SR04_Ultrasonic_Module_User_Guide.pdf hämtad: 2020-10-07.

[10] M. Gonzales, "Product details specification, vibration sensor switch sw-18010p," 2011, [Online] Tillgänglig: https://www.scribd.com/document/333270565/Datasheet-sw18010P hämtad: 2020-10-07.

[11] F. Östman, "Md407 – en arm-baserad laborationsdator för utbildning," 2014.

Bilagor

Test - T1

Projektnamn/projektgrupp: 13
Datum: 2020-10-09 tid: 16:45

Testfalls-id:
T1

Testobjekt:
Larmenheten

Testarens namn:
Felix Bråberg, Gabriel Käll, Erik Nilsson, Linus Haraldsson, Philips Antonsson, Simon Widerberg

Beskrivning:

Larmenheten, centralenheten och dörrenheten kopplas upp. Centralenheten pingar enheterna och lägger till de enheters som svara ID i modullistan. Sedan kopplas dörrenheten ifrån systemet.

Förväntat resultat:

Alla enheter ska läggas till i modullistan och Larmenheten skall gå igång när centralenheten inte får svar från dörrenheten.

Resultat:

Modullistan fungerade som förväntat och Larmenheten gick igång.

Analys:

Enheterna hinner pinga tillbaka inom det angivna tidsintervallet. När en enhet kopplades ur så gick larmet också igång vilket tyder på att den grundläggande kommunikation är i fas.

Test - T2

Projektnamn/projektgrupp: 13

Datum: 2020-10-13 tid: 16:45

Testfalls-id:
T2

Testobjekt:
Larmenheten

Testarens namn:
Simon Widerberg

Beskrivning:

Hela systemet ska vara uppkopplat. Sen ska någon röra sig framför rörelsesensorn på det avstånd som skall dra igång det lokala larmet.

Förväntat resultat:

Larmenhetens lokala larm ska dra igång

Resultat

Larmenhetens lokala larm triggades

Analys:

Larmenhetens lokala larm fungerar korrekt

Test - T3

Projektnamn/projektgrupp: 13

Datum: 2020-10-22 tid: 16:15

Testfalls-id:
T3

Testobjekt:
Larmenheten

Testarens namn:
Simon Widerberg, Linus haraldsson

Beskrivning:

Detta test ska visa felmarginalen på avståndsmätaren vid två meters avstånd. Detta utförs genom att ett avstånd framför avståndsmätaren på två meter mättes upp med tumstock och sedan skriver avståndsmätaren ut vilket avstånd den mäter upp i centimeter. Detta ska utföras tio gånger. Avståndsmätaren räknar avståndet i heltal.

Förväntat resultat:

vi förväntar oss en felmarginal på en centimeter.

Resultat

Alla 10 tester mätte 200 cm.

Analys:

Resultatet var bättre än förvänat. avståndsmätaren har en felmarginal på mindre än en centimeter på två meters avstånd.

Test - T4

Projektnamn/projektgrupp: 13

Datum: 2020-10-09 tid: 13:45

Testfalls-id:
T4

Testobjekt:
Larmenheten

Testarens namn:
Simon Widerberg, Linus Haraldsson

Beskrivning:

Hela systemet ska vara uppkopplat. Sedan ska någon röra sig framför rörelsesensorn på det avstånd som skall aktivera det globala larmet.

Förväntat resultat:

Larmenhetens globala larm ska aktiveras.

Resultat

Larmenhetens globala larm aktiveras.

Analys:

Larmenhetens globala larm fungerar korrekt.

Test - T5

Projektnamn/projektgrupp: 13

Datum: 2020-10-16 tid: 13:00

Testfalls-id:

T5

Testobjekt:

Störenheten, CAN-nätverket

Testarens namn:

Linus Haraldsson, Gabriell Käll, Felix Bråberg

Beskrivning:

En störenhet kopplas upp till CAN-kommunikationen mellan centralenheten och larmenheten. Störenheten ska först spamma lågprioritetsmeddelanden medan centralenheten skickar ett larmmeddelande, vilket har hög prioritet, för att testa om systemet prioriterar korrekt.

Förväntat resultat:

Högprioritetsmeddelandena ska fungera som vanligt.

Resultat:

Högprioretetsmeddelandena mottogs som förväntat av larmenheten vilken larmade korrekt.

Analys:

Prioriterande av meddelanden verkar fungera korrekt.

Test - T6

Projektnamn/projektgrupp: 13

Datum: 2020-10-16 tid: 14:00

Testfalls-id:

T6

Testobjekt:

Störenheten, CAN-nätverket

Testarens namn:

Linus Haraldsson, Gabriell Käll, Felix Bråberg

Beskrivning:

En störenhet kopplas upp till CAN-kommunikationen mellan centralenheten och larmenheten. Störenheten ska konstant skicka meddelanden till larmenheten för att störa kommunikationen. Först ska meddelanden ha fel ID så det inte tas emot av larmenheten. Andra körningen ska meddelanden faktiskt ha rätt ID och då behöva bearbetas av larmenheten. Centralenheten ska under testerna pinga med hög frekvens för att märka när systemet saktas ned.

Förväntat resultat:

Meddelanden med fel ID ska inte ha stor påverkan på systemet.
Meddelanden som behöver hanteras av enheten förväntas sakta ner systemet och möjligtvis förhindra kommunikation mellan enheterna.

Resultat:

När störenhetens meddelanden inte hanterades av någon enhet hade de ytterst liten effekt på systemet. När störenhetens meddelanden behövde hanteras blev systemet märkbart långsammare och fick problem med kommunikationen.

När centralenheten skickade meddelande 1000 gånger per sekund medan störenheten störde tappades kontakten regelbundet. Vid 500 gånger per sekund fungerar systemet återigen som förväntat trots störenheten.

Analys:

Systemets prestanda är bra och kan hantera lågprioriterade meddelanden skickade med hög frekvens så länge systemets egna meddelanden skickas med en frekvens under 500 gånger per sekund.

Test - T7

Projektnamn/projektgrupp: 13

Datum: 2020-10-13 tid: 13:35

Testfalls-id:

T7

Testobjekt:
Larmenheten

Testarens namn:
Linus Haraldsson, Simon Widerberg och Felix Bråberg

Beskrivning:

Larmenheten ska larma om den inte blir pingad av centralenheten inom 5 sekunder. Först pingar centralenheten i 10 sekunder och då förväntas inget larm. Sedan kopplas centralenheten av.

Förväntat resultat:

Medan centralenheten pingar ska systemet fungera felfritt. När centralenheten blir frånkopplad ska larmenheten larma efter 5 sekunder.

Resultat:

Allt fungerade som förväntat.

Analys:

Larmenheten kollar nu om den är uppkopplad eller inte. Möjligt tillägg är att larmet ska stängas av ifall enheten åter igen får ping från centralenheten.

Test - T8

Projektnamn/projektgrupp: 13
Datum: 2020-10-13 tid: 13:35
Testfalls-id:
T8
Testobjekt:
Larmenheten, Centralenheten, Dörrenheten, USART
Testarens namn:
Linus Haraldsson, Simon Widerberg

Beskrivning:

Dörren skall skicka iväg ett larm-meddelande till centralenheten, därefter skall centralenheten skicka vidare meddelandet till larmenheten. Efter det skickar centralenheten två larm-meddelanden får att starta och stänga av larmet. Larmenheten skall också starta sitt egna larm och centralenheten skall då kunna skicka ett meddelande för att stänga av detta. Alla medelanden att stänga av larmet ska ske genom USART.

Förväntat resultat:

Centralenheten skall ta emot dörrenhetens larm-meddelande för att sedan skicka ett larm-meddelande till larmenheten som senare ska larma. Sedan skall centralenhetens två larmmeddelanden kunna stänga av och sätta på larmenhetens larm. Till sist skall larmet kunna sätta på sitt egna larm och centralenhetens sista meddelande skall kunna stänga av det.

Resultat:

Alla meddelanden anlände korrekt till Larmenheten och växlade av/på larmet som förvätnat.

Analys:

Resultatet visar att systemet kan använda CAN för att kommunicera med varandra. Systemet larmar korrekt och stänger sedan av larmet med keypad. USART visade korrekta statusmeddelanden. Detta visar att systemet kommunikation gällande larm är funktionell.

Test - T9

Projektnamn/projektgrupp: 13
Datum: 2020-10-13 tid: 13:45
Testfalls-id:
T9
Testobjekt:
Keypad
Testarens namn:
Sebastian Sjögren, Gabriel Käll

Beskrivning:

Enheterna Keypad ska vara uppkopplad och programmet ska larma vid start vilket aktiverar keypad. När programmet körs ska en användare mata in värden på keypad till programmet, vilket används för att larma av programmet.

Förväntat resultat:

De inmatade värdena från keypad skall matas in korrekt. Beroende på vilket värde som skrevs in ska vissa specialfall ske, t.ex att systemets avlarmas eller att inmatningen blir rensad.

Resultat:

Programmet fungerar ej. Inmatningar från keypad kan inte avläsas av programmet, vilket betyder att inga av de förväntade resultaten kan uppnås.

Analys:

Resultatet pekar på ett icke funktionerande program. Problemet antas ligga i uppkopplingen till den fysiska enheten keypad, då programmet fungerade som förväntat i en simulator. Som åtgärd kommer det analyserar hur keypad är inkopplad till md407-kortet och vilka pinnar som används i koden.

Test - T10

Projektnamn/projektgrupp: 13
Datum: 2020-10-14 tid: 18:00
Testfalls-id:
T10
Testobjekt:
Keypad
Testarens namn:
Sebastian Sjögren

Beskrivning:

Enheterna Keypad ska vara uppkopplad och programmet ska larma vid start vilket aktiverar keypad. När programmet körs ska en användare mata in värden på keypad till programmet, vilket används för att larma av programmet.

Förväntat resultat:

De inmatade värdena från keypad skall matas in korrekt. Beroende på vilket värde som skrevs in ska vissa specialfall ske, t.ex att systemets avlarmas eller att inmatningen blir rensad.

Resultat:

Programmet fungerar som det förväntade resultatet. Systemets avlarmas vid rätt kod och reagerar som förväntat.

Analys:

Resultatet pekar på ett funktionerande program. Därmed är funktionaliteten av keypad enheten bevisad.

Test - T11

Projektnamn/projektgrupp: 13
Datum: 2020-10-16 tid: 15:00
Testfalls-id:
T11
Testobjekt:
Larmenhet
Testarens namn:
Simon Widerberg, Felix Bråberg

Beskrivning:

När larmenheten blir tillsagd att stänga av larmet skall det ske omedelbart utan att avsluta larmsignalen. Det sker genom att vi lade till några tomma return statements i alarmfunktionen om triggered blir satt till 0.

Förväntat resultat:

Larmet avslutas omedelbart.

Resultat:

Larmet avslutades omedelbart.

Analys:

Om larmenheten nu får en order att stänga av larmet sker det utan att avsluta larmsignalen.

Test - T12

Projektnamn/projektgrupp: 13

Datum: 2020-10-02 tid: 15:00

Testfalls-id:

T12

Testobjekt:

Dörrenheten

Testarens namn:

Philip Antonsson, Erik Nilsson

Beskrivning

Dörrenhetens MD047-kort har två dioder(röd och grön) per dörrsensor kopplade till port D(0-7) och flera dörrsensorer till pinar på port E. Att dioderna tänds och släcks när dörren öppnas och stängs skall verifieras. Att betenedet stämmer överens ifall dörren är satt till armed eller unarmed verifieras också.

Förväntat resultat:

När en dörrsensorn är satt att larma och öppnas bör dess röda dioden tändas och när sensorn sluts skall den släckas, den gröna dioden skall vara släckt. När dörren är satt att inte larma skall dess röda diod alltid vara släckt och den gröna alltid tänd.

Resultat

Vid första körningen av testet då två dörrar öppnades samtidigt stängdes ljuset för den andra dörren av. Koden rättades till och när testet kördes en gång till gav det korrekta resultat.

Analys

Vid första körningen tolkades att ljuset ändrades vid fel tillfälle som ett fel i när ljusändringsfunktionen kallades. Det rättades och felet var lösts. Att testet ger förväntat resultat betyder att koden ändrar en dörrs status och då utför tillhörande funktioner korrekt. Samt att den åtskiljer olika dörrar korrekt. Dörrenheten är redo att vidarutvecklas.

Test - T13

Projektnamn/projektgrupp: 13

Datum: 2020-10-04 tid: 16:00

Testfalls-id:
T13

Testobjekt:
Dörrenheten

Testarens namn:
Philip Antonsson, Erik Nilsson, Gabriel Käll

Beskrivning:

Ett test utfördes för att testa att dörrenheten kunde ta emot, tolka och genomföra meddelanden som skickas via CAN. Två MD407-kort kopplades samman varav en hade en dörrsensor och två dioder kopplade(röd och grön). Den andra enheten skickar ett konfigurationsmeddelande om att stänga av larmfunktionen(armed) för den inkopplade dörren.

Förväntat resultat:

Om dörrens larmfunktion är på, det vill säga att armed är satt till 1, lyser den röda dioden när dörren öppnas och släcks när den stängs igen. När den sedan stängs av borde den röda dioden sluta lysa när dörren öppnas och den gröna dioden skall alltid vara tänd.

Resultat:

Det förväntade resultatet var vad som skedde vilket visar på att funktionerna för CAN och de för genomföringen av instruktionerna fungerade. Dock om dörren var öppen när dens alarmed status sattes till 0 förblev den röda dioden tänd vilket senare korrigerades i koden.

Analys:

Att resultatet i stort sätt speglade det förväntade tolkas som att koden fungerade och tyder på att konfigurationsmeddelanden kan tas emot, tolkas och utföras korrekt.

Test - T14

Projektnamn/projektgrupp: 13
Datum: 2020-10-15 tid: 15:00
Testfalls-id:
T14
Testobjekt:
Keypad,rörelsesensor,CAN,centralenhet,larmenhet,dörrenhet.
Testarens namn:
Felix Bråberg, Gabriel Käll, Erik Nilsson, Linus Haraldsson, Philips Antonsson, Simon Widerberg, Sebastian Sjögren.

Beskrivning:

workaround för att kolla att pings mottas från rätt plats.

Förväntat resultat:

Enheter skall få korrekt pings, sedan ska rörelsesensor aktiveras så att larmet går av, sedan ska keypad användas för att avlarma.

Resultat:

Programmet fungerar som det förväntade resultatet. Systemets avlarmas vid rätt kod och reagerar som förväntat ...

Larm mottar ping (1,3).

Central har node id 13,

Larm nod-id 11

Central får ping från 3 och 11.

Larm får ping från 11.

Dörr får ping från 11.

Analys:

Resultatet betyder att enheterna kan använda CAN för att kommunicera med varandra. Systemet larmas korrekt när antingen en enhet dras ut och slutar skicka meddelande till centralenheten eller när sensorn går av. Sedan stängs larmet av med keypad. Detta bevisar att systemet med flera enheter och deras kommunikation är funktionell.

Test - T15

Projektnamn/projektgrupp: 13
Datum: 2020-10-15 tid: 15:45
Testfalls-id:
T15
Testobjekt:
USART,dörrenhet
Testarens namn:
Gabriel Käll, Erik Nilsson, Philip Antonsson, Sebastian Sjögren.

Beskrivning:

Testar om dörrarna kan armeras genom USART kommandon och att larm kan därefter skickas från dörrenhet till centralenhet. Larmet på dörren skall sedan stängas av med keypad. Det ska även vara möjligt att konfigurera antalet dörrarnas, se deras status och ändra tidsgränser med USART kommandon.

Förväntat resultat:

Enheter skall först få korrekt pings för att etablera fungerande kommunikation. Sedan ska givet antal dörrar korrekt armeras beroende på vad som skrivs in i USART. Ett lokalt larm aktiveras av att en dörrenhet kopplas ifrån och att den givna lokala tidsgränsen passerats. Efter en viss given global tidsgräns vidarebefordras larmet till centralenheten som larmar systemet. Därefter ska keypad användas för att avlarma, vilket ska avläsas med USART.

Resultat:

Programmet fungerar som det förväntade resultatet. De rätta dörrarna armeras vilket gör så att de larmar, efter korrekta tidsgränser, när de kopplas ut ur systemet. Systemets avlarmas sedan vid rätt kod och reagerar som förväntat.

Analys:

Resultatet betyder att enheterna kan använda CAN för att kommunicera med varandra. Config kommandot kan skrivas in genom USART för att armera och konfigurera dörrenheten. Systemet larmas korrekt när en enhet dras ut och slutar skicka meddelande till centralenheten. Sedan stängs larmet av med keypad och USART visade korrekta statusmeddelanden. Detta bevisar att systemet kommunikation och centralenhetens inmatningar från USART och keypad är funktionell.

Test - T16

Projektnamn/projektgrupp: 13
Datum: 2020-10-15 tid: 16:15
Testfalls-id:
T16
Testobjekt:
Keypad,rörelsesensor,USART,CAN,Larmenhet,Centralenhet.
Testarens namn:
Gabriel Käll, Simon Widerberg, Linus Haraldsson, Sebastian Sjögren.

Beskrivning:

Testar om larmenheten kan armeras genom USART kommandon och att larm kan beroende på armed status skicka ett larm meddelande från larmenhet till centralenhet. Larmet på centralenheten skall sedan stängas av med keypad.

Förväntat resultat:

Enheter skall först få korrekt pings för att etablera fungerande kommunikation. sedan ska dörrarna korrekt armeras beroende på vad som skrivs in i USART. Larmet aktiveras av att en dörrenhet dras ut, vilket först ska skapa ett lokalt larm som efter en viss tid vidarebefordras till centralenheten som larmar systemet. Därefter ska keypad användas för att avlarma.

Resultat:

Programmet fungerar ej som förväntat. När ett config meddelande skickas från centralenheten mottas den av larmenheten, men den hanteras inkorrekt vilket betyder att larmenheten inte armeras. Då larmenheten aldrig armeras kan den inte larma och resten av testet kan inte utföras.

Analys:

Resultatet betyder att enheterna kan använda CAN för att kommunicera med varandra. Config kommandot kan skrivas in och mottas av centralenheten, men armering sker ej då centralenhetens funktion för att motta konfig meddelande är felt strukturerad. För att lösa detta kommer centralenhetens och larm enhetens send/receive funktioner jämföras och skrivas om så att de är kompatibla.

Test - T17

Projektnamn/projektgrupp: 13
Datum: 2020-10-27 tid: 16:00
Testfalls-id:
T17
Testobjekt:
USART, Keypad, dörrenhet, centralenhet och larmenhet
Testarens namn:
Gabriel Käll, Erik Nilsson, Philips Antonsson, Sebastian Sjögren, Linus Haraldsson, Simon Widerberg och Felix Bråberg

Beskrivning:

Detta test ska röra hela larmsystemet och visa att sensorer, kommunikation och interagering med systemet fungerar som förväntat.

Först ska uppstartsfasen utföras där både larmenheten och dörrenheten ska läggas in centralenhetens enhetslista. Därefter ska det gå att konfigurera vardera sensor enskilt utan att påverka resten av systemet, samt ska armering och disarmeringsfuktionerna fungera.

Larmenheten ska larma korrekt och periferienheternas lokala larm ska också utlösas som förväntat. Det globala larmet ska gå att stängas av via knappsatsen.

Förväntat resultat:

Uppstartsfas:
Vi förväntar oss att båda periferienheterna tar emot ping-meddelandena och svarar som förväntat och därmed läggs till i centralenhetens enhetslista.

Konfigurering:
När dörrsensorn får konfigurationsmeddelandet konfigureras den korrekt. Vid konfiguration av antal dörrar, larmfunktion och larm-tidsgränser ändras dessa och när modulen desarmeras fortsätter den att uppdatera dörrarnas värden men det globala larmet går inte.

Avståndssensorn konfigureras till samma avstånds som skrivet i USART och kalibrerar vid konfigureringsmeddelanden när värdet “Distance” är noll. Sensorerna och larmet blir avstängt vid desarmering och återupptas vid armering.

Larmning och avlarmning:
Centralenheten vidarebefordrar larmmeddelanden från dörrenheten och lyckas skicka egna larmmeddelanden ifall enheter blir oväntat bortkopplade. Larmenheten tar emot dessa meddelanden och utlösa larmet. Larmenheten larmar vid utslag från dess sensorer eller då enheten blir bortkopplad från resten av systemet.

Knappsatsen larmar av då rätt kod blir inmatad.

Resultat:

Periferienheterna dyker upp i centralenhetens enhetslista. Dörrenhetens olika värden kunde konfigureras korrekt och vid desarmering slutade centralenheten vidarebefordra det globala larmet till larmenheten medan dörrenhetens andra funktioner fortfarande fungerade som vanligt. Liknande resultat ser vi för larmenheten då larm utlöses när de ska och vars sensorer agerar som förväntat. Konfigurering fungerade och distansmätarens avstånd kan både kalibreras och sättas av centralenheten. Alla funktioner upphör korrekt vid desarmering.

Analys:

Enheterna dök upp i enhetslistan och ping-meddelanden togs emot och skickades ut vilket var det som förväntades. Detta tolkas som att uppstartsfasen fungerat korrekt. Eftersom testets utfall stämmer överens med det förväntade resultatet tyder på att vi har ett robust och funktionellt system som hanterar de krav vi ställt på det. Att dörrenhetens och larmenhetens värden ändras vid konfigurationsmeddelanden tyder på att de kan ta emot och implementera meddelanden från centralenheten på ett korrekt sätt. Testets resultat visar också på att desarmeringen fungerade som förväntat.

Test - T18

Projektnamn/projektgrupp: 13

Datum: 2020-10-27 tid: 17:00

Testfalls-id:

T8

Testobjekt:

Störenheten, CAN-nätverket

Testarens namn:

Linus Haraldsson, Gabriell Käll, Felix Bråberg

Beskrivning:

En störenhet kopplas upp till CAN-kommunikationen mellan centralenheten och larmenheten. Störenheten ska konstant skicka slumpade meddelanden på CAN-nätverket. Datasträngarna som skickas har strukturen av vanliga CAN-meddelanden men innehållet slumpas fram med hjälp av en funktion som returnerar pseudoslumptal beroende på systemklockan.

Förväntat resultat:

På grund av den simpla strukturen av meddelanden och avsaknaden av någon sorts kryptering förväntas systemet inte kunna hantera denna sorts attack.

Resultat:

Systemet agerade oförutsägbart och var inte längre pålitligt.

Analys:

Systemet klarar som förväntat inte av att en stor mängd slumpade meddelanden skickas på CAN-nätverket.

Test - T19

Projektnamn/projektgrupp: 13
Datum: 2020-10-27 tid: 16:45
Testfalls-id:
T19
Testobjekt:
Centralenheten
Testarens namn:
Gabriel Käll

Beskrivning:

I startupfasen ska centralenheten pinga alla uppkopplade enheter i systemet och lägga till de enheter som svarar i modullistan. Sedan ska larmenheten och dörrenheten omkonfigureras. Cantralenhetens information om enheternas konfiguration ska uppdateras korrekt.

Förväntat resultat:

Centralenheten ska korrekt uppdatera hur enheterna är konfigurerade

Resultat:

Centralenheten uppdaterade korrekt hur enheterna är konfigurerade

Analys:

Centralenheten ska nu ha all information som behövs för att veta att systemet fungerar som förväntat.