FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Design Document

Group Nr. 4

Sandra Ljubic
Rostyslav Samonov

Contents

1 Project description

Im diesem Dokument wird das Projekt einer sprachgesteuerten Lampe vorgestellt, die auf einen gewissen Befehl hin, das Licht ein- und ausschalten kann.

1.1 Problem Description

Figure 1: system description

Immer wieder gibt es in Krankenhäusern Probleme mit Keimen. Vorallem gefährlich ist das während Operationen im OP Saal, wenn der Doktor das Licht an- oder auschalten muss und dabei mit dem Lichtschalter in Berührung kommt. Bei dem Kontakt werden Keime auf die Handschuhe übertragen, die dann bösartige bis tötliche Folgen für den Patienten haben können. Aus diesem Grund wird in vorliegender Arbeit das Projekt einer sprachgesteuerten Lampe vorgestellt.

2 Software architecture and design

2.1 Software modules

Safety related modules

1. Modul STM32 Raspberry Pi - Testdurchlauf-Safety: Description: Alle Safety-related Modules werden nach Start des Systems 1x getestet.
   Functions:
   a) Test-Pakete werden gezählt, verschlüsselt und geschickt
   b) Test-Pakete werden empfangen, gezählt und entschlüsselt
   c) Orangenes LED Blinken wird geprüft (STM32 Sprachaufnahme und Raspberry Pi Übertragung)
   d) Rotes LED Leuchten wird geprüft (STM32 Raspberry Pi Fehlerzustand)
   e) Grünes LED Leuchten wird geprüft (STM32 Raspberry Pi Einsatzbereitschaft)
   f) Blaues LED Leuchten wird geprüft (Raspberry Pi Licht eingeschaltet)
   Siehe Requirement 6.1.1

2. Modul STM32 - Zählen der zu übertragenen Pakete:
   Description: Die zu übertragenen Pakete werden auf 16 Pakete hochgezählt, um durch die Überprüfung nach der Übertragung, die Vollständigkeit der Daten zu gewährleisten.
   Functions: Hochzählen der zu übertragenen Pakete auf 16 Stück
   Data: signed int Value
   Siehe Requirement 6.1.2

3. Modul Raspberry Pi - Überprüfung der Anzahl der übertragenen Pakete:
   Description: Die übertragenen Pakete werden gezählt um die Vollständigkeit der Daten zu gewährleisten. Dieser Wert muss 16 betragen.
   Functions: Zählen der Pakete, Vergleichen der Anzahl der Pakete mit Zahl 16.
   Data: signed int Value
   Siehe Requirement 6.1.3

4. Modul STM32 - LED Sprachaufnahme:
   Description: Eine orangene LED blinkt, solange die Sprachaufnahme läuft.
   Functions: Eine orangene LED blinkt mit 3 Hz, solange die Sprachaufnahme läuft.
   Siehe Requirement 6.1.4

5. Modul STM32 - LED Einsatzbereitschaft:
   Description: Eine grüne LED leuchtet, wenn der STM32 einsatzbereit ist.
   Functions: Eine grüne LED leuchtet, wenn der STM32 einsatzbereit ist. Ist der STM32 in einem Fehlerzustand, wird die grüne LED ausgeschaltet.
   Siehe Requirement 6.1.5

6. Modul STM32 - LED Fehlerzustand:
   Description: Eine rote LED leuchtet, sobald ein Fehlerzustand (Safe State) auftritt.
   Functions: Eine rote LED leuchtet.
   Siehe Requirement 6.1.6

7. Modul STM32 - Fehlerzustand (Safe State):
   Description: Wenn ein Fehler auf dem STM32 erkannt wird, wird der STM32 in einen Fehlerzustand versetzt und alle Funktionalitäten werden deaktiviert, nur die rote LED aus Safety Related Modul 6 ist aktiviert. Jede Verbindung zum Raspberry Pi ist unterbrochen. Ein manueller Reboot des Systems ist erforderlich.
   Functions: Deaktivieren der Funktinalitäten des STM32 und aktivieren der roten LED aus Safety related Module 6.
   Siehe Requirement 6.1.9

8. Modul Raspberry Pi - LED Übertragung:
   Description: Eine orangene LED blinkt, solange die Übertragung läuft.
   Functions: Eine orangene LED blinkt mit 3 Hz, solange die Übertragung läuft.
   Siehe Requirement 6.1.7

9. Modul Raspberry Pi - LED Einsatzbereitschaft:
   Description: Eine grüne LED leuchtet, wenn der Raspberry Pi einsatzbereit ist.
   Functions: Eine grüne LED leuchtet, wenn der Raspberry Pi einsatzbereit ist. Tritt ein Fehler auf, wird die LED vorübergehend ausgeschaltet.
   Siehe Requirement 6.1.5

10. Modul Raspberry Pi - LED Fehleraufkommen:
    Description: Eine rote LED leuchtet, sobald ein Fehler auftritt, um den Nutzer zu informieren.
    Functions: Eine rote LED leuchtet.
    Siehe Requirement 6.1.6

11. Modul Raspberry Pi - UART Kabelfunktionalitätsüberprüfung:
    Description: Um zu sehen, ob die Rx (receive) Leitung auf dem Raspberry Pi funktioniert, prüft man die orangene LED. Blinkt sie nach der Sprachaufnahme bedeutet das, dass das Kabel Daten überträgt und somit funktionsfähig ist, leuchtet sie nicht, ist das Kabel defekt.
    Functions: Um zu sehen ob die Rx (receive) Leitung auf dem Raspberry Pi funktioniert, blinkt die orangene LED während der Datenübertragung.
    Siehe Requirement 6.1.8

12. Modul Raspberry Pi - Fehleraufkommen:
    Description: Wenn ein Fehler auf dem Raspberry Pi auftritt wird die rote LED aktiviert, die grüne LED wird vorübergehend ausgeschaltet.

Functions: Aktivieren der roten LED.
Siehe Requirement 6.1.9

Security related modules

1. Modul STM32 - Verschlüsselung der .wav Daten (binary):
   Description: Das .wav Daten-Paket wird für die Übertragung symmetrisch verschlüsselt.
   Functions: Das .wav Daten-Paket werden symmetrisch verschlüsselt.
   Data Input: binary Daten
   Data Output: verschlüsselte binary Daten
   Siehe Requirement 6.2.1

2. Modul Raspberry Pi - Entschlüsseln der .wav Daten (binary):
   Description: Das .wav Daten-Paket wird entschlüsselt.
   Functions: Die symmetrisch verschlüsselten binary Daten werden entschlüsselt.
   Data Input: verschlüsseltes binary Daten
   Data Output: entschlüsselte binary Daten
   Siehe Requirement 6.2.2

3. Modul STM32 - Buffer Overflow Check:
   Description: Um Buffer Overflows zu vermeiden, wird die Sprachaufnahme nach 1 Sekunde automatischc beendet und es werden genau 16 Datenpakete an der Raspberry Pi versendet.
   Functions: Beenden der Sprachaufnahme nach 1 Sekunde und befüllen von genau 16 Paketen.
   Siehe Requirement 6.2.3

4. Modul STM32 - DoS Timeout:
   Description: Um zu vermeiden, dass eine DoS Attacke durchgeführt werden kann, verwenden wir einen Button mit 50 Millisekunden Debounce, wodurch der Button nur maximal alle 50 Millisekunden ausgelöst werden kann, was zu gering für eine DoS Attacke ist.
   Functions: Button mit 50 Millisekunden Debounce
   Siehe Requirement 6.2.4

5. Modul STM32 und Raspberry Pi - MAC:
   Description: Um die Integrität der Daten zu gewährleisten, wird den Daten auf dem STM32 ein MAC angehängt, der nach erhalt der Daten auf dem Raspberry Pi geprüft wird.
   Functions: Nutzen eines MAC.
   Siehe Requirement 6.2.5

Modules with no influence on Safety and Security

1. Modul STM32 - Aktivierung Aufnahme:
   Description: Zur Aktivierung der Sprachaufnahme wird ein User Button verwendet, welcher sich auf dem STM32 befindet.
   Functions: Auslösen eines Interrupts durch den User-Button.
   Data: GPIO Input Interrupt
   Siehe Requirement 6.3.1

2. Modul STM32 - Durchführung der Aufnahme:
   Description: Um die Sprachaufnahme durchzuführen wird das MEMS-Mikrofon auf dem STM32 aktiviert. Nach Ablauf von 1 Sekunde, wird die Aufnahme automatisch beendet.
   Functions: Aktivierung des MEMS-Mikrofons, Durchführung der Aufnahme von Sprachbefehlen und beenden der Aufnahme nach 1 Sekunde.
   Data: Voice Input in binary Format, Timer
   Siehe Requirement 6.3.2

3. Modul STM32 - Befüllen der Daten-Pakete:
   Description: Um die Sprachaufnahme zu Übertragen, werden die Daten der Aufnahme in 16 Daten-Pakete in vordefinierter Größe gelegt und dieses dann übertragen. Bei der nächsten Befüllung, werden die Daten des alten Pakets überschrieben.
   Functions: Befüllen der Daten-Pakete.
   Data: binary Pekete
   Siehe Requirement 6.3.3

4. Modul STM32 - Übertragung der verschlüsselten binary Daten über UART:
   Description: Um die Sprachaufnahme zur Verarbeitung auf den Raspberry Pi zu schicken, wird sie über einen Bit-Strom über UART übertragen.
   Functions: Übertragen eines Bit-Stroms über UART.
   Data: verschlüsselte binary Daten
   Siehe Requirement 6.3.4

5. Modul Raspberry Pi - Empfangen der Daten:
   Description: Die Daten werden auf dem Rasberry Pi empfangen.
   Functions: Empfangen und Speichern der Daten.
   Data: verschlüsselte binary Daten
   Siehe Requirement 6.3.5

6. Modul Raspberry Pi - Verarbeiten der entschlüsselten Daten durch die Spracherkennungssoftware (Rhasspy):
   Description: Die entschlüsselten Daten werden verarbeitet und ein Befehl oder ein Fehler wird ausgegeben.

Functions: Die entschlüsselten Daten werden an die Software Rhasspy übergeben und in der Software verarbeitet und der entsprechende Befehl wird ausgegeben.
Data Input: binary Daten
Data Output: Aktivierungs-Befehl oder Fehlerausgabe
Siehe Requirement 6.3.6

7. Modul Raspberry Pi - Licht-Aktivierung/Deaktivierung:
   Description: Die Lichtquelle (blaue LED) wird durch einen Befehl aktiviert oder deaktiviert.
   Functions: Aktivieren und Deaktivieren einer Lichtquelle (blaue LED) durch den übertragenen Befehl.
   Data: Aktivierungsbefehl
   Siehe Requirement 6.3.7

2.2 Libraries

STM32 in C

1. stm32f4xx_HAL
2. Rhasspy (Spracherkennung)
3. TinyCrypt (Verschlüsselung)

Raspberry Pi in Python

1. gpiozero
2. serial
3. time
4. os
5. pycryptodome
6. json
7. request
8. sys

2.3 Interrupts

Folgende Interrupts werden in dem System verwendet. Sie sind nach absteigender Priorität sortiert.

1. Interrupt durch den User-Button.
2. Interrupt durch den Timer zum Toggeln der LEDs.

2.4 Pinout STM32

Folgende Pinouts des STM32F407 Discovery Boards wurden gewählt, um die in diesem Design Dokument beschirebenen Funktionalitäten zu gewährleisten.

1. UART1 Tx Leitung auf GPIO C12
2. LED1 (orange) auf GPIO D13
3. LED2 (grün) auf GPIO D12
4. LED3 (rot) auf GPIO D14
5. LED4 (blau) auf GPIO D15
6. USER BUTTON auf GPIO A0
7. MEMS Mikrofon auf GPIO A6

2.5 Pinout Raspberry Pi

Folgende Pinouts des STM32F407 Discovery Boards wurden gewählt, um die in diesem Design Dokument beschirebenen Funktionalitäten zu gewährleisten. Diese können sich im Laufe der Realisierung des Systems noch ändern. Dieses Dokument wird entsprechend erweitert.

1. UART2 Rx Leitung auf GPIO 15
2. LED5 (orange) auf GPIO 2
3. LED6 (grün) auf GPIO 18
4. LED7 (rot) auf GPIO 17
5. LED8 (blau) auf GPIO 3

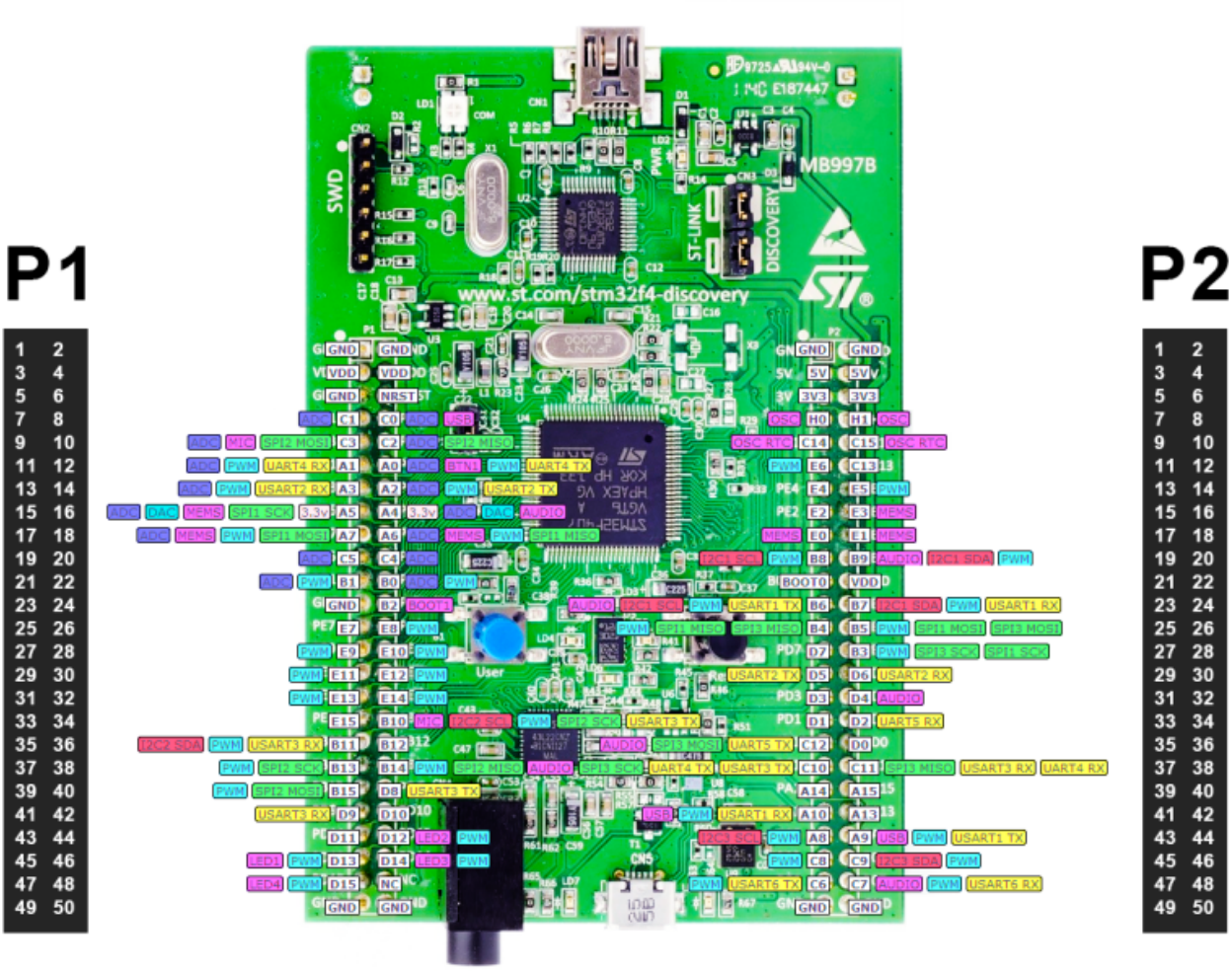

Figure 2: STM32 Pinouts

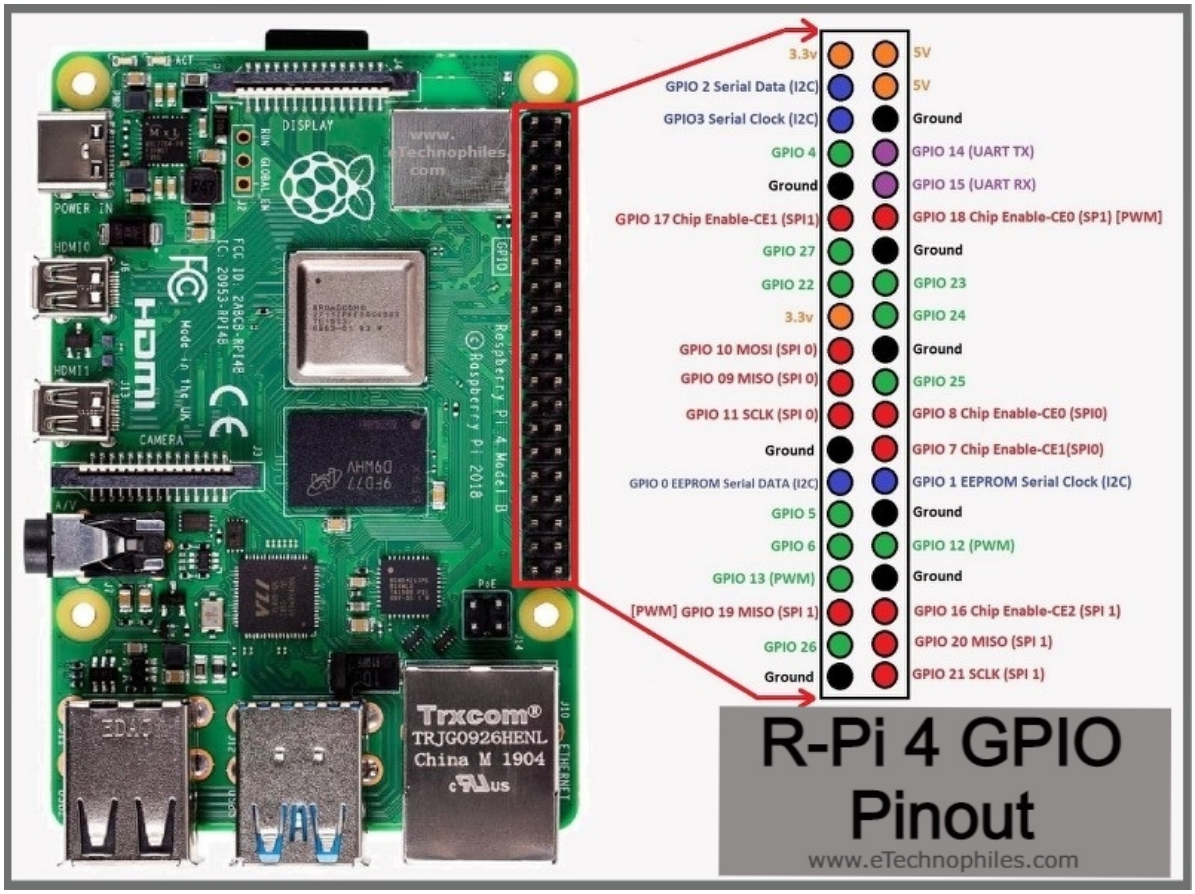

Figure 3: Raspberry Pi Pinouts

3 Program flowchart

In diesem Abschnitt zeigen wir die Prozessschritte des Programms. Auf dem ersten Flow Chart ist der Prozessfluss des STM32 zu sehen, auf dem zweiten der des Raspberry Pi.

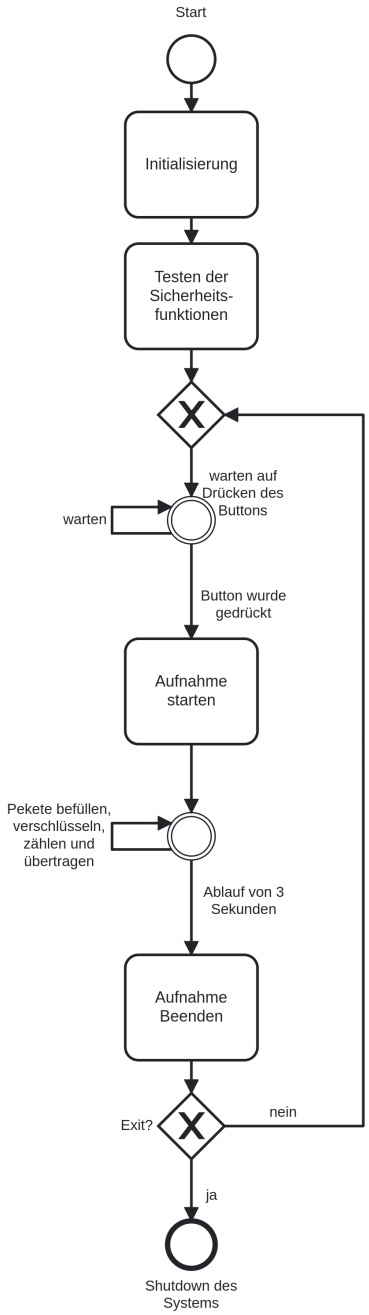

Figure 4: STM32 Flow Chart

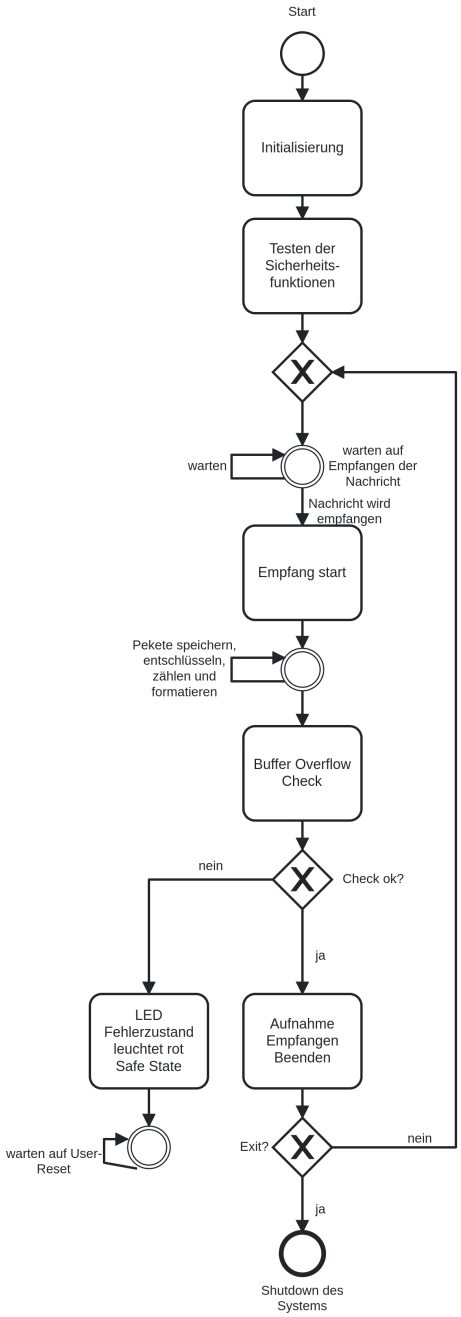

Figure 5: Raspberry Pi Flow Chart

4 Hazard identification

4.1 Identified hazards and countermeasures

1. Hazard: Counter Overflow (bei den geschickten Paketen).
   Gegenmaßnahme: Zurücksetzen desr Counter nach Übertragen der Pakete.
   Siehe Safety Related Module 2

2. Hazard: UART Kabelfunktionalitätsüberprüfung, falls das Kabel defekt wird.
   Gegenmaßnahme: Das Funktionieren der Leitung soll durch blinken einer orangenen LED während der Datenübertragung angezeigt werden. Blinkt die LED während Datenübertragung nicht, liegt ein Problem vor.
   Siehe Safety Related Module 11

3. Hazard: Daten werden über UART nicht vollständig übertragen.
   Gegenmaßnahme: Die Anzahl der übertragenen Pakete wird nach der Übertragung aller Pakete überprüft und muss genau 16 betragen.
   Siehe Safety Related Modules 2 und 3

4.2 Identified hazards without countermeasures

1. Hazard: Probleme bei der Stromversorgung. Gegenmaßnahme wird nicht implementiert

2. Hazard: LEDs leuchten nicht, obwohl sie sollten. Gegenmaßnahme wird nicht implementiert

3. Hazard: STM32 MEMS-Mikro fällt aus (mechanischer Defekt). Gegenmaßnahme: Überprüfung der Größe des Audiofiles nach der Übertragung, um festzustellen, ob dieses eine Aufnahme enthält

5 Threat identification

5.1 Identified threats and countermeasures

1. Threat: DoS Attacke
   Gegenmaßnahme: Debouncing des Buttons auf 50 Millisekunden
   Siehe Security Related Module 4

2. Threat: Datenmanipulation
   Gegenmaßnahme: Nutzung eines MAC
   Siehe Security Related Module 5

3. Threat: Buffer Overflow.
   Gegenmaßnahme: Die Sprachaufnahme wind nach Ablauf von 1 Sekunde automatisch gestoppt und es werden genau 16 Pakete mit Daten befüllt.
   Siehe Security Related Module 3

5.2 Identified threats without countermeasures

1. Threat: Spoofing
   Gegenmaßnahme wird nicht implementiert

2. Threat: Elivation of Privelege auf OS.
   Gegenmaßnahme wird nicht implementiert

3. Threat: Manipulation von Testsetzen im Vornhinein (Spracherkennung)
   Gegenmaßnahme wird nicht implementiert

6 Requirements

6.1 Safety related requirements

6.1.1 Requirement:
Beim Start des Systems werden alle Safety-relevanten Module in einem Testdurchlauf abgearbeitet. Am Ende des Testdurchlaufs kennzeichnen die Status-LEDs den Erfolg des Testablaufs. Siehe Safety Related Modules 5 und 9

6.1.2 Requirement:
Um die Vollständikeit der übertragenen Daten-Pakete zu gewährleisten, werden die Daten vor dem Verschicken auf dem STM32 in genau 16 Pakete gefüllt. Nach der Übertragung werden sie auf dem Raspberry Pi gezählt. Der Wert muss mit dem Wert 16 übereinstimmen. Siehe Safety Related Module 2

6.1.3 Requirement:
Um die Vollständikeit der übertragenen Daten-Pakete zu gewährleisten, werden diese nach Erhalt auf dem Raspberry Pi zur Überprüfung gezählt. Der Wert muss mit dem Wert 16 übereinstimmen. Siehe Safety Related Module 3

6.1.4 Requirement:
Während die Sprachaufnahme läuft und die Daten übertragen werden, soll eine LED orange blinken, welche den Aufnahmezustand und Datenfluss signalisiert. Siehe Safety Related Module 4

6.1.5 Requirement:
Eine LED auf dem STM32 und dem Raspberry Pi soll die Einsatzbereitschaft des Systems dem Benutzer signalisieren. Siehe Safety Related Modules 5 und 9

6.1.6 Requirement:
Eine LED auf dem STM32 und auf dem Raspberry Pi soll den Fehlerzustand des Systems dem Benutzer signalisieren. Siehe Safety Related Module 6

6.1.7 Requirement:
Eine LED soll dem Benutzer signalisieren, dass die Übertragung der Aufnahme läuft. Siehe Safety Related Module 8

6.1.8 Requirement:
Die UART Leitung zwischen dem STM32 und dem Raspberry Pi soll auf ihre Funktionsfähigkeit getestet werden. Siehe Safety Related Module 12

6.1.9 Requirement:
Falls ein Fehler in einer Funktion im System erkannt wird, soll das System in den Fehlerzustand übergehen.

6.2 Security related requirements

6.2.1 Requirement:
Um die Daten während der Übertragung vor unbefugtem Mitlesern zu schützen, müssen diese vor der Übertragung, auf dem STM32, symmetrisch verschlüsselt werden.

6.2.2 Requirement:
Damit die verschlüsselten Daten wieder gelesen werden können, müssen diese nach der Übertragung, auf dem Raspberry Pi wieder entschlüsselt werden.

6.2.3 Requirement:
Um einen Buffer Overflow, durch nicht stoppende Datenübertragung, auf dem Raspberry Pi zu verhindern, soll die Sprachaufnahme nach 1 Sekunde automatisch enden.

6.2.4 Requirement:
Um eine DoS Attacke zu verhindern, soll ein Button mit 50 Millisekunden Debounce verwendet werden.

6.2.5 Requirement:
Die Integrität der Daten soll durch einen MAC gewehrleistet werden.

6.3 Requirements with no influence on Safety and Security

6.3.1 Requirement:
Damit die Aufnahme gestartet werden kann, soll es einen Button geben, den der User bedient.

6.3.2 Requirement:
Um die Sprachaufnahme durchzuführen, soll das MEMS-Mikrofon auf dem STM32 aktiviert werden. Nach Ablauf von 1 Sekunde, soll die Aufnahme automatisch beendet werden.

6.3.3 Requirement:
Die aufgenommenen Daten sollen zur Übertragung in 16 Pakete vordefinierter Größe gefüllt werden. Die Pakete werden sofort verschickt. Die Daten des nächsten Pakets, ersetzen die Daten des vorigen.

6.3.4 Requirement:
Die symmetrisch verschlüsselten Daten sollen über einen Bit-Strom über UART an den Raspberry Pi zur weiteren Verarbeitung geschickt werden.

6.3.5 Requirement:
Auf dem Raspberry Pi sollen die verschlüsselten Daten empfangen und gespeichert werden, um sie weiterzuverarbeiten.

6.3.6 Requirement:
Die entschlüsselten Daten sollen auf dem Raspberry Pi durch die Spracherkennungssoftware Rhasspy verarbeitet werden. Der entsprechende Befehl oder eine Fehlermeldung werden gegeben.

6.3.7 Requirement:
Die Lichtquelle soll durch einen Befehl ein- oder ausgeschaltet werden.