\subsubsection{UAV} Text\\

Text

\subsubsection{Nucleo} STM32F767 Nucleo-144

Wie oben erwähnt, die Sensorfusion Algorithmus wird implementiert und überprüft in hinzugefügte Hochleistungsprozessor -- \glqq Nucleo STM32F767ZIT6\grqq, die von STMicroelectronics entwickelt wird und wegen hoher Leistung heutzutage weit benutzt wird. Mikrocontroller enthaltet Prozessor und zugleich auch Peripheriefunktion, d.h. nicht nur Arbeits- und Programmspeicher sondern auch komplexe Peripheriefunktionen wie z.B. PWM-Ausgänge, CAN-, USB-, I2C, SPI (angewendet in unserem Protokoll) usw. werden komplett und kompakt auf demselben Chip integriert, was eine kostengünstige und flexible Möglichkeit für Benutzer bietet, neue Konzepte auszuprobieren und Prototypen zu erstellen.

Dank der kompletten Entwicklungswerkzeugen erleichtert man sehr die Entwicklungsarbeit. Einerseits, mit zahlreiche verpackte Bibliotheksfunktionen ermöglicht man schnelle Entwicklung verstärkt sehr die Lesbarkeit der Code, was ein wichtiger Grund die Popularität des STM32-Familien ist. Andererseits, mit GUI-Werkzeug wie CubeMX darf man einfach und sicher das System wie z.B. Konfiguration von dem ganzen Uhrbaum initialisieren.

Mikrokontrolle STM32767 in unsere Nucleo besitzt ARM Cortex-M7 Kern, der im maximal 216 MHZ arbeiten könnte. Zudem erhält er 2 MByte Flash-Speicher, 512Kbytes SRAM und 144 Pins, während der Stromverbrauch aber sehr gering -- c.a. 250mA (Alle Peripheriegeräte aktiviert wird. TODO BibReference reference datasheet P124!!!) ist. Und Debugger/Programmierer-Werkzeug ST-LinkV2 ist schon implementiert. Ausreichende Peripheriegeräte werden implementiert, z.B. insgesamt vier I2C Schnittstellen sechs SPI Schnittstellen. SPI3, die von uns benutzt wird, könnte maximal als 25 Mbit/s Kommunikation Geschwindigkeit erreichen. Diese Geschwindigkeit ist schon viel größer als die Geschwindigkeit, welche HLP erreichen könnte.

\begin{figure}

\centering

\includegraphics[width=0.7\linewidth]{pics/Nucleo.jpg}

\caption{Nucleo}

\label{fig:comickopf-eps-converted-to}

\end{figure}

\subsubsection{Sensor Data} \

Unsere Daten zur Protokoll und Sensorfusion basiert aus zwei Teilen:die integriert in UAV Sensoren, äußere Sensoren. Firefly besitzt wie normale UAV das Gyro, Kompass, GPS. etc. Sie sind genug für normale Regelung in normaler Situationen aber nicht gut wie z.B. Indoor-Situation. Außerdem ist Druckmesser nicht sehr präzise in Höhe Bestimmung. Um besser zu positionieren und Attitüde zu schätzen, wir werden noch wie z.B. Optischer Fluss, Kamera, usw. hinzufügen.

Die innere Sensoren von UAV sind im folgende:

\begin{itemize}

\item

Gyro: Die zeige die lineare Beschleunigung und Winkelgeschwindigkeit.

\item Kompass: Es messt das Magnetfeld der Erde.

\item GPS: Durch GPS wird z.B. globale Position, Geschwindigkeit und Zeit.

\item Pressuresensor: Die Höhe könnte im Natur damit bestimmen.

\end{itemize}

Die äußere Sensoren von UAV sind im folgende:

\begin{itemize}

\item

Optischer Fluss: GPS könnte nicht innerhalb des Gebäudes arbeiten, während optische Fluss ist eine gute Ersetzung. Wir benutzen PX4FLOW, die erfolgreich als Position System GPS ersetzt hat. (Ref) % An Open Source and Open Hardware Embedded Metric Optical Flow CMOS Camera for Indoor and Outdoor Applications Dominik Honegger, Lorenz Meier, Petri Tanskanen and Marc Pollefeys ETH Zurich, Switzerland

\item

LIDAR-Lite v3. Für Indoor-Höhe-Bestimmung drinnen. Es ist viel schneller als Ultraschall und robust, zuverlässig.

\end{itemize}

Außerdem werden noch die anderen Sensoren spät hinzugefügt. z.B. Kamera, Microsoft Kinect, usw. um besser die verschiedenen Situationen anzupassen. Die Kapazität unseres Protokolls ist noch ausreichend für die Hinzufügung.

\begin{figure}

\centering

\includegraphics[width=0.4\linewidth]{pics/PX4FLOW.jpg}

\includegraphics[width=0.4\linewidth]{pics/Lidar.jpg}

\caption{äußere Sensoren}

\label{fig:otherSensor}

\end{figure}

\subsection{Serial Peripheral Interface (SPI)}

Das Serial Peripheral Interface (kurz SPI) ist ein im Jahr 1987 von Susan C. Hill Et al., damals bei dem Halbleiterhersteller Motorola (heute NXP Semiconductors), entwickeltes Bus-System und stellt einen \glqq lockeren \grqq Standard für einen synchronen seriellen Datenbus (Synchrones Serial Port) dar, mit den digitalen Schaltungen nach dem Master-Slave-Prinzip miteinander verbunden werden können(BibReference.)

% Patent US4816996: Queued serial peripheral interface for use in a data processing system. Angemeldet am 24. Juli 1987, veröffentlicht am 28. März 1989, Anmelder: Motorola (heute: NXP USA Inc.), Erfinder: Susan C. Hill, Joseph Jelemensky, Mark R. Heene.).

SPI benutzt nicht Adresse zum Slave Auswahl aber einfach einen Stift als \glqq Chip Select \grqq. SPI ist ein wirklich vollduplexfähiger Bus und hat hohe Kommunikationsgeschwindigkeit im Vergleich andere häufige angewendete serielle Bus wie USART, I2C. Die typische Verbindung von SPI ist wie Bild \ref{fig:spi-verbindung} gezeigt(Sternverbindung). Auch ist es gezeigt, dass jede Master-Slave Kommunikation im General vier Leitungen angefordert. Sind jeweils:

\begin{itemize}

\item SCLK ( Serial Clock) auch SCK, wird vom Master zur Synchronisation ausgegeben

\item MOSI ( Master Output, Slave Input) oder SIMO (Slave Input, Master Output)

\item MISO ( Master Input, Slave Output) oder SOMI (Slave Output, Master Input)

\item ${\overline{SS}}$ (Slave Select), oder $\overline{CS}$ (Chip Select).

\end{itemize}

\begin{figure}

\centering

\includegraphics[width=0.9\linewidth]{pics/SPIVerbindung.png}

\caption{SPI-Verbindung durch Kaskadierung der Slaves}

\label{fig:spi-verbindung}

\end{figure}

Der Protokollablauf ist wie Bild \ref{fig:spiablauf} gezeigt. Legt Master mit der Leitung \glqq Slave Select \grqq fest, mit welchem Slave er kommunizieren will, ist der jeweilige Slave aktiv. Es wird ein Wort vom Master zum Slave und gleichzeitig ein anderes Wort vom Slave zum Master transportiert.

Die Polarität von Clock und Datenübertragung sind nicht bei Motorola definiert. Sie werden durch \glqq CPHA\grqq, der Clock Idle definiert, und \glqq CPOL\grqq, der definiert, in welche Flanken der Chip Daten übernommen, festgelegt.

Mit jeder Taktperiode wird ein Bit übertragen. Beim üblichen Bytetransfer sind also acht Taktperioden für eine vollständige Übertragung nötig. Master verschiebe durch MOSI ein Bit aus dem Register und übernimmt durch MISO ein Bit ins diese Register. Slave benutzt aber MISO und MOSI. Eine vollständige Übertragung ist den Austausch zwischen den Registern von Master und Slave. Es können auch mehrere Worte hintereinander übertragen werden. Eine Übertragung ist beendet, wenn das Slave-Select-Signal endgültig auf High gesetzt wird.

\begin{figure}

\centering

\includegraphics[width=0.9\linewidth]{pics/SPIAblauf}

\caption{}

\label{fig:spiablauf}

\end{figure}