# HARDWARE-IN-THE-LOOP – SIMULATION ALS BESTANDTEIL DES ENTWICKLUNGSPROZESSES FÜR DAS AUTOMATISCHE FLUGSTEUERUNGSSYSTEM DER STEMME S15

A. Kaden, G. Walde, B. Boche, R. Luckner,
Technische Universität Berlin, Institut für Luft- und Raumfahrt,
Marchstraße 12, 10587 Berlin, Deutschland

**Zusammenfassung**

Zur hochpräzisen automatischen Steuerung eines Luftarbeitsflugzeugs wird ein automatisches Flugsteuerungssystem für den Motorsegler STEMME S15 im Rahmen des Projekts LAPAZ im nationalen Luftfahrtforschungsprogramm (LUFO IV) entwickelt. LAPAZ steht für *Luft-Arbeits-Plattform für die Allgemeine Zivilluftfahrt*. Zur Erprobung dieses automatischen Flugsteuerungssystems wurde eine Bodentestanlage in Form eines Hardware-in-the-Loop (HIL) – Simulators aufgebaut. Mit ihm soll die korrekte Integration des Flugsteuerungssystems in das Flugzeug (STEMME S15 Prototyp) verifiziert und die korrekte Funktion der Flugregelgesetze überprüft werden. Die HIL-Simulation ist Teil eines kostengünstigen Entwicklungsprozesses für sicherheitskritische Systeme, welcher im Rahmen des Projekts aufgebaut wird. Der vorliegende Bericht gibt einen Überblick über den Entwicklungsprozess und beschreibt das Konzept, das Funktionsprinzip sowie den Aufbau des HIL-Simulators. Als Beispiel zur Validierung des Simulators werden Flugversuchs- und Simulationsdaten der ersten automatischen Landung der STEMME S15 miteinander verglichen.

## SYMBOLE

| | |
|---|---|
| H | Höhe |
| V | Geschwindigkeit |
| $\alpha$ | Anstellwinkel |
| $\gamma$ | Bahnwinkel |
| $\Theta$ | Längslagewinkel |

## ABKÜRZUNGEN UND INDIZES

| | |
|---|---|
| A/D | Analog / Digital |
| AD | Air Data |
| AFCP | Automatic Flight Control Panel |
| AFCS | Automatic Flight Control System |
| AHRS | Attitude and Heading Reference System |
| CAS | Calibrated Air Speed |
| CPM | Core Processing Module |
| EGT | Exhaust Gas Temperature |
| FCL | Flight Control Law |
| FMRA | Fachgebiet Flugmechanik, Flugregelung und Aeroelastizität (TU Berlin) |
| GND | Ground |
| GPS | Global Positioning System |
| ILS | Institut für Luftfahrtsysteme (Universität Stuttgart) |
| IOM | Input / Output Module |
| LA | Laser Altimeter |
| MSL | Mean Sea Level |
| RTWECs | Real-Time Workshop® Embedded Coder™ Stall |
| SEPHIR | Simulator for Educational Projects and Highly Innovative Research |
| SIMIF | Simulation Interface |
| TCU | Turbo Control Unit |
| UDP | User Datagram Protocol |
| WOW | Weight on Wheel |

## 1 EINLEITUNG

Ein Flugregelungssystem ist integraler Bestandteil eines Flugzeugs. Es besteht aus Soft- und Hardware. Seine Funktionen sind sicherheitskritisch. Da es außerdem hoch komplex ist, unterliegt sein Entwicklungsprozess strengen Regeln, die in SAE / ARP 4754 [1] beschrieben sind. Die Gefahr, Fehler bei der Entwicklung zu machen, ist groß. Der Nachweis der absoluten Fehlerfreiheit eines solchen hochintegrierten Systems, ob analytisch oder durch entsprechende Tests, ist nahezu unmöglich. Ein strukturierter Systementwicklungsprozess soll gewährleisten, dass die Anzahl sicherheitskritischer Entwurfsfehler minimal ist. Mit großem Aufwand werden in einer eigenen ingenieurwissenschaftlichen Disziplin, dem *Systems Engineering*, Entwicklungsmethoden aufgebaut und optimiert. Ziel ist die Realisierung eines Produkts, welches dem Kundenwunsch entspricht und behördlich zugelassen werden kann. Um den Zulassungsprozess erheblich zu erleichtern, gibt es in der Luftfahrt Richtlinien und Standards zur Systementwicklung. Richtlinien zur Entwicklung von Software sind in RTCA / DO-178B [2] beschrieben. Ein wichtiges Element des Entwicklungsprozesses ist der Funktionstest. Dieser wird an verschiedenen Prüfständen von Teilsystemen bis hin zum Gesamtsystem durchgeführt.

Insbesondere für kleine und mittelständige Unternehmen ist es wichtig, kostengünstige Entwicklungsmethoden zu verwenden, um die Ziele Realisierung und Zulassung des Gesamtsystems zu erreichen. Eine Möglichkeit ist die geschickte Wahl der Prüfstände. Hierbei ist darauf zu achten, dass kosten- und zeiteffizient gearbeitet werden kann, ohne die Qualität der Tests zu verringern. Im vorliegenden Bericht wird eine Bodentestanlage vorgestellt, die einen *Iron Bird* unter Nutzung des Flugzeugprototyps in Verbindung mit einer HIL-Simulation realisiert.

## 2 FORSCHUNGSPROJEKT LAPAZ

Die Abkürzung LAPAZ steht für *Luft-Arbeits-Plattform für die Allgemeine Zivilluftfahrt*. Ziel des Forschungsvorhabens ist die Entwicklung und Demonstration eines zuverlässigen, hoch präzise arbeitenden, automatischen Flugsteuerungssystems für Luftarbeitsflugzeuge. Das Flugzeug soll in der Lage sein, automatisch starten und landen zu können sowie als stabilisierte Plattform auch bei turbulenter Atmosphäre Messflüge durchzuführen. Als Einsatzmöglichkeiten sind Geoexploration, Umwelt-, Katastrophenschutz, Fischerei-, Küsten- und Grenzüberwachung zu nennen. Dabei stellt ein Motorsegler mit ca. 1 t Abflugmasse und Nutzlasten von 100-300 kg eine leistungsfähige und gleichzeitig kostengünstige sowie effiziente Lösung dar. Beispielflugzeug ist der Motorsegler STEMME S15.

LAPAZ ist ein vom Bundesministerium für Wirtschaft und Technologie (BMWI) im Rahmen des nationalen Luftfahrtforschungsprogramms (LUFO IV) gefördertes Technologieprojekt. An dem Forschungsvorhaben wirken drei Projektpartner mit, wobei die Aufgabenteilung an die jeweiligen Kompetenzbereiche angelehnt ist. Das Institut für Luftfahrtsysteme (ILS) der Universität Stuttgart ist für die Entwicklung der fehlertoleranten Plattform des Flugsteuerungssystems inklusive aller Redundanzmechanismen zuständig. Das Fachgebiet Flugmechanik, Flugregelung und Aeroelastizität (FMRA) der TU Berlin entwickelt die Flugregelung, das dafür notwendige flugmechanische Modell, die Mensch-Maschine-Schnittstelle sowie einen speziell ausgelegten Entwicklungsprozess für eine spätere Zulassung des Systems. Die Firma STEMME AG stellt das Flugzeug und ist für die Integration von Soft- und Hardware in die Flugzeugzelle sowie die Durchführung von HIL-Simulationen und Flugversuchen verantwortlich.

Mit der ersten automatischen Landung des STEMME S15 Prototyps am 22. März 2012 auf dem Flugplatz Neuhardenberg wurde ein wichtiger Meilenstein des Projekts erreicht. Um 17:44 lokaler Zeit setzte das Arbeitsflugzeug gesteuert durch das automatische Flugsteuerungssystem präzise und sicher auf der Landebahn 08 auf.

Im Hinblick auf die spätere Zulassung als Arbeitsflugzeug, wird vom FMRA ein Entwicklungsprozess für komplexe, sicherheitskritische Systeme aufgebaut und die Softwareentwicklung daran orientiert. Im folgenden Abschnitt wird eine kurze Übersicht zum Entwicklungsprozess gegeben.

### 2.1 Entwicklungsprozess

Der Entwicklungsprozess im Projekt LAPAZ erfolgt nach dem V-Modell (siehe BILD 1). Zu Beginn werden alle funktionalen und nichtfunktionalen Anforderungen an das automatische Flugsteuerungssystem in einer *Top Level* Spezifikation festgelegt. Diese werden schrittweise (Flugzeug → System → Gerät → Komponente) *top-down* bis zur Hardware- und Software-Anforderung verfeinert. Nach der Kodierung bzw. Fertigung folgt *bottom-up* die schrittweise Integration und Verifikation. Die Prozessschritte enden mit Validationstests (der Abnahme durch den Kunden). Hierzu muss angemerkt werden, dass das V-Modell eine Idealisierung darstellt. In der Realität auftretende Iterationen werden nicht abgebildet. Fällt z.B. in der Systemintegration ein Fehler auf, dessen Ursache in einer fehlerhaften Systemrealisierung liegt, so erfolgt von der Systemspezifikation ausgehend ein erneuter Durchlauf (Iteration) der Prozessschritte des V-Modells.

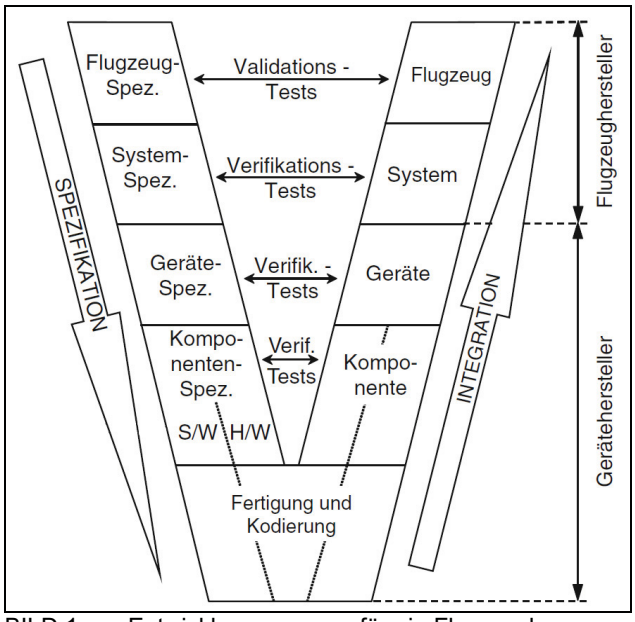



BILD 1. Entwicklungsprozess für ein Flugregelungssystem nach dem V-Modell [3]

Im Projekt LAPAZ wird ein inkrementelles Vorgehen verfolgt, d.h. es werden nicht alle Funktionen gleichzeitig erstellt, sondern von einer Grundfunktion ausgehend schrittweise neue Funktionen ergänzt. Diese werden nach dem V-Modell entwickelt und im Flugversuch erprobt. Hierfür muss zu Beginn die Reihenfolge der Erstellung der Teilfunktionalitäten festgelegt werden. Dies erfolgt einerseits durch Betrachtung der Abhängigkeiten der einzelnen Funktionen, andererseits nach deren Umsetzungsrisiko bzw. der Wichtigkeit der Funktion für das Endprodukt. Der Vorteil dieses Vorgehens gegenüber der Entwicklung der gesamten Funktionalität in einem Stück, ist das frühe Erlangen von Erkenntnissen aus dem Flugversuch, die damit frühzeitig in den Entwicklungsprozess zurückfließen können. Dadurch ist eine schrittweise Verbesserung möglich, sodass die bereits existierenden Funktionalitäten einem Reifeprozess unterliegen. Das inkrementelle Vorgehen führt zu einer wachsenden Softwarearchitektur. Um suboptimale Strukturen zu vermeiden, sollte die grundsätzliche Architektur von Anfang an vollständig entwickelt und festgelegt werden. Eine Restrukturierung (*Refactoring*) sollte nur im Notfall erfolgen.

Die Entwicklung der Flugregelgesetze (FCL – *Flight Control Laws*) erfolgt modellbasiert mit Hilfe von Simulink® und Stateflow® die zum Softwarepaket MATLAB® gehören. Der Quellcode der FCL wird durch den Einsatz des Real-Time Workshop® Embedded Coder™ (RTWEC – Grundprodukt und Erweiterung zur Codegenerierung unter MATLAB) erzeugt. Durch Kompilierung entsteht eine lauffähige Applikation, die auf dem Zielsystem integriert wird. Die Integrationsarbeiten erfolgen stufenweise vom Teil- zum Gesamtsystem.

Der gesamte Integrationsprozess wird von Verifikations- und Validationstests begleitet, die sowohl die Funktionalität entsprechend der Spezifikationen (Verifikation) als auch die Erfüllung der Anforderungen (Validation) nachweisen sollen. Die ersten Tests erfolgen bereits auf Mo-

dellebene mit Hilfe einer PC-Simulation sowie einer sogenannten Software-in-the-Loop (SIL) – Simulation[1]. Nach der Codegenerierung mittels RTWEC und der Kompilierung, muss nachgewiesen werden, dass das Modell sich genauso verhält wie der erzeugte Quellcode und die Applikation auf dem Zielsystem. Durch dieses Vorgehen bleiben die Tests auf Modellebene gültig. Der Nachweis erfolgt durch den Vergleich von Testvektoren. Diese werden in der Modellsimulation erzeugt und als Eingangsparameter für einen sog. Hostsimulator sowie einen Prüfstand des Zielsystems beim Projektpartner ILS genutzt. Die Ausgangsparameter werden anschließend verglichen, um die Übereinstimmung nachzuweisen.

Die HIL-Simulation ist ein weiterer wichtiger Bestandteil des Entwicklungsprozesses. Dieser Verifikationsschritt ist insbesondere deshalb wichtig, da nicht alle Anforderungen auf Modellebene getestet werden können. Hinzu kommt, dass das Testen der Software auf dem Zielsystem eine grundlegende Forderung der Richtlinie RTCA / DO-178B [2] darstellt. Dabei wird eine validierte und repräsentative Testumgebung gefordert. In Kapitel 6.4.1 *Test Environment* wird dies wie folgt beschrieben:

*„An excellent test environment includes the software loaded into the target computer and tested in a high fidelity simulation of the target computer environment."* [2]

In Abschnitt 3 werden das Konzept, der Aufbau und die Funktionsweise des HIL-Simulators beschrieben. Mit diesem können folgende Eigenschaften für die untersuchten Pfade geprüft werden:

1) Integrationsfähigkeit der Applikation (FCL) in die gesamte Softwarearchitektur (Kapitel 6.4.3 b, *Software Integration Testing*, [2])
2) Kompatibilität der Software mit der Hardware (Kapitel 6.4.3 a, *Hardware/Software Integration Testing*, [2])
3) Erfüllung der funktionalen Anforderungen an die FCL-Software
4) Mensch-Maschine-Interaktion

Für die Entwicklung der Regelgesetze und die anschließenden Tests (PC-, SIL- und HIL-Simulation) wird ein hoch genaues flugmechanisches Simulationsmodell der STEMME S15 verwendet. Dieses wird in Abschnitt 3.4 näher beschrieben.

## 2.2 STEMME S15 Prototyp

Das Luftarbeitsflugzeug S15 ist eine Variante des Motorseglers S6 (siehe BILD 2) der STEMME AG, welcher für kommerzielle Anwendungen konzipiert ist und nach EASA CS-23 [4] zugelassen werden soll. Er hat eine Spannweite von 18 m und eine maximale Abflugmasse von ca. 1 t. Im Projekt LAPAZ wurde der Prototyp der S15 um die für die Flugregelung nötige Sensorik, Aktuatorik sowie Soft- und Hardware erweitert. Die Antriebseinheit der S15 besteht aus einem turbogeladenen Viertaktmotor BOMBARDIER-ROTAX 914 S, einem Untersetzungsgetriebe und einem Propeller mit variabler Blattstellung. Als Kommandogrößen werden Propellerdrehzahl und Drosselklappenstellung vorgegeben. Die tatsächliche Motordrehzahl sowie der Einstellwinkel der Propellerblätter werden geregelt. Die Steuerung der Flugzeugbewegung um die drei körperfesten Achsen erfolgt konventionell mittels Höhen-, Seiten- und Querruder. Als Querruder dienen die beiden äußeren der drei Hinterkantenklappen der jeweiligen Flügelhälfte. Alle drei haben die Funktion der Wölbklappen. Zusätzlich verfügt die S15 über aerodynamische Bremsklappen. Die Trimmung erfolgt über einen stationären Höhenruderausschlag. Zur Entlastung des Piloten wird der Steuerknüppel mit Hilfe einer Feder und einer aktuatorgesteuerten Vorspannung kraftfrei gehalten. Die Bugradsteuerung ist in Abhängigkeit des Einfederweges mit dem Seitenruder gekoppelt.

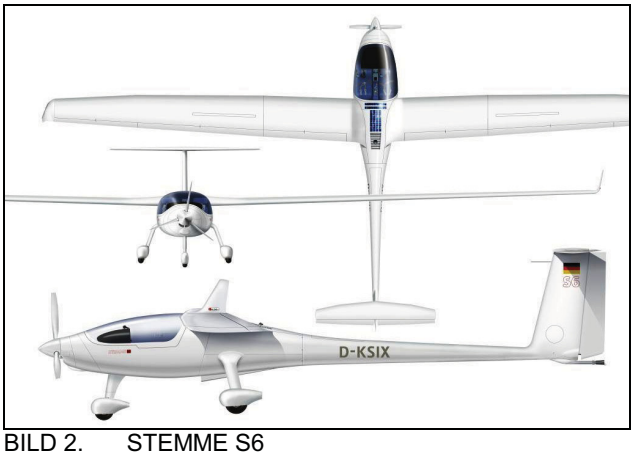
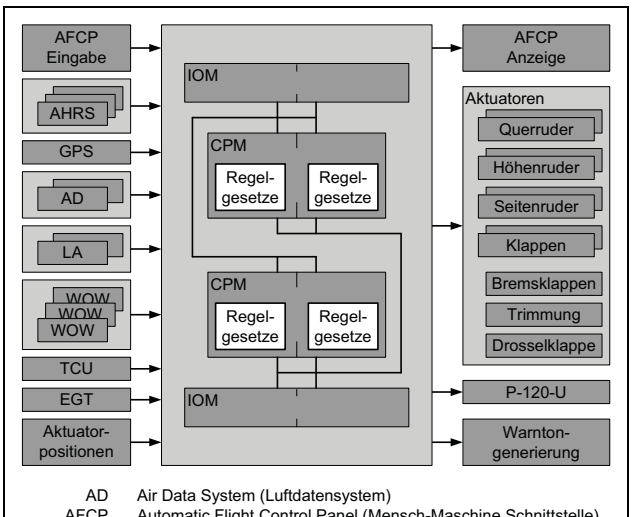



BILD 2. STEMME S6

Im Folgenden wird ein kurzer Überblick zur Plattform des automatischen Flugsteuerungssystems (AFCS – *Automatic Flight Control System*) gegeben. Eine ausführliche Beschreibung ist [5], [6] und [7] zu entnehmen. Details zur Reglerentwicklung sind in [8] angeführt. BILD 3 zeigt das AFCS schematisch.




| | |
|---|---|
| AD | Air Data System (Luftdatensystem) |
| AFCP | Automatic Flight Control Panel (Mensch-Maschine Schnittstelle) |
| AHRS | Attitude and Heading Reference System (Trägheitsplattform) |
| CPM | Core Processing Module (Duplexrechner) |
| EGT | Exhaust Gas Temperature (Abgastemperatursensor) |
| GPS | Global Positioning System (Navigationssystem) |
| IOM | Input / Output Module (Duplexrechner) |
| LA | Laser Altimeter (Laserhöhenmesser) |
| P-120-U | Steuergerät zur Drehzahlregelung des Propellers |
| TCU | Turbo Control Unit (elektronischer Ladedruckregler) |
| WOW | Weight on Wheel (Bodenkontaktsensor) |

BILD 3. Übersicht zum automatischen Flugsteuerungssystem AFCS

Das AFCS ist ein sicherheitskritisches System. Durch die Anwendung spezieller Redundanzmechanismen ist die Plattform fehlertolerant. Der modulare Aufbau sowie die

[1] Echtzeitsimulation im Forschungssimulator SEPHIR (*Simulator for Educational Projects and Highly Innovative Research*)

Skalierbarkeit ermöglichen eine leichte Anpassung an neue Aufgabengebiete. Basiskomponenten der Plattform sind die Module CPM (*Core Processing Module*) und IOM (*Input / Output Module*). Es handelt sich dabei um Duplexrechner mit grundsätzlich gleichem Aufbau. CPM und IOM bilden gemeinsam eine Rechnereinheit. Das gesamte Flugsteuerungssystem verfügt über zwei dieser Einheiten, was eine Duo-Duplex-Architektur ergibt. CPM und IOM verfügen jeweils über zwei gleichartige Systemseiten die sich gegenseitig überwachen und ein striktes Fail-Passive-Verhalten haben. Auf dem CPM sind die Flugregelgesetze implementiert. Aufgabe des IOM ist die Kopplung der Plattform an die Außenwelt, d.h. an Sensoren, Aktuatoren, Anzeigen und Bedienung. Dabei ist gewährleistet, dass jedem CPM stets alle Sensoren und Stellglieder zur Verfügung stehen. Als Eingangsgrößen benötigt die Flugregelung Informationen über Lage und Position sowie über Flug- und Betriebszustand der S15. Die entsprechenden Daten werden mit Hilfe verschiedener Sensoren ermittelt. Die redundante Auslegung erfüllt die Anforderungen an ein sicherheitskritisches System. Die Lage, die Position und der aerodynamische Flugzustand werden mit folgenden Sensoren erfasst:

- drei Trägheitsplattformen (AHRS – *Attitude and Heading Reference System*),
- ein Navigationssystem (GPS – *Global Positioning System*),
- zwei Luftdatensysteme (AD – *Air Data*) und
- zwei Laserhöhenmesser (LA – *Laser Altimeter*).

Jedes Fahrwerksbein verfügt über einen Bodenkontaktsensor (WOW – *Weight on Wheel*). Der Einfederweg signalisiert, ob sich das Flugzeug am Boden befindet. Daten zum Betriebszustand des Motors liefern der elektronische Ladedruckregler (TCU – *Turbo Control Unit*) als Bestandteil des ROTAX Motors sowie der Abgastemperatursensor (EGT – *Exhaust Gas Temperature*). Der Ladedruckregler übermittelt Motordrehzahl und Drosselklappenstellung. Die Ausschläge der Steuerorgane werden über die Aktuatorstellungen approximiert und auf die Reglereingänge zurückgeführt. Als Stellglieder dienen elektromechanische Aktuatoren, die in das vorhandene Gestänge eingreifen. Die Trimmung erfolgt mit Hilfe des bestehenden Aktuators der manuellen Steuerung. Die Drosselklappenstellung wird über einen Aktuator kommandiert, der am Leistungshebel der S15 angreift. Insgesamt sind 11 Aktuatoren eingebaut. Quer-, Seiten- und Höhenruder sowie die Wölbklappen sind jeweils über zwei redundante Aktuatoren unabhängig voneinander ansteuerbar. Das Drehzahlkommando wird über das vorhandene Steuergerät (P-120-U) vorgegeben. Die Reglerbedieneinheit (AFCP – *Automatic Flight Control Panel*) als Mensch-Maschine-Schnittstelle wurde speziell für den S15 Prototyp entwickelt.

## 3 HARDWARE-IN-THE-LOOP – SIMULATOR

Aus der Definition des Entwicklungsprozesses ergeben sich Anforderungen an den HIL-Simulator. Der Ersatz eines *Iron Birds* durch den Prototyp der STEMME S15 in der HIL-Simulation erfüllt dabei nicht nur Forderungen nach Zeit- und Kosteneffizienz, er minimiert darüber hinaus auch mögliche Fehlerquellen. Im folgenden Unterkapitel werden Anwendungsmöglichkeiten des HIL-Simulators vorgestellt. Anhand dieser wird ein Anforderungskatalog entwickelt, welcher die Basis für den Konzeptentwurf darstellt.

### 3.1 Konzeptentwurf

Der HIL-Simulator dient vorrangig der Funktionsprüfung von Hard- und Software des Flugsteuerungssystems am Boden. Die korrekte Funktion der Flugregelgesetze im integrierten Zustand soll damit verifiziert werden. Zusätzlich gibt es weitere Anwendungen. Der Testpilot kann das geplante Flugversuchsprogramm vorher hinsichtlich der Durchführbarkeit untersuchen. Das Simulieren von Notsituationen schafft die Möglichkeit, Prozeduren zur Übernahme der manuellen Steuerung durch den Piloten zu testen, zu trainieren und deren Sicherheit nachzuweisen. Anhand aufgezeichneter Daten können bereits absolvierte Flugversuche nachgestellt und analysiert werden. Die Reproduzierbarkeit von Versuchen ermöglicht den Vergleich verschiedener Reglerversionen. Des Weiteren kann der HIL-Simulator zur Demonstration des automatischen Flugsteuerungssystems z.B. auf Messen oder vor dem Kunden dienen.

Aus diesen Anwendungsmöglichkeiten leitet sich der folgende Anforderungskatalog ab:

- Einbeziehung möglichst aller realen Systemkomponenten des Flugsteuerungssystems,
- Erzeugen notwendiger Sensor- und Motordaten am Boden durch Echtzeitsimulation,
- Erfassen der Gesamtübertragungsfunktion der Ruderansteuerung durch Messen der Stellgliedausschlagwinkel,
- Übertragen und Aufbereiten der Stellgrößen als Eingang der S15-Simulation,
- Aufbereiten und Übertragen der simulierten Sensordaten an das Flugsteuerungssystem,
- Mobilität der Testanlage für den Transport,
- Robustheit der Testanlage für den Einsatz am ungeschützten Teststandort,
- Sicht- und Instrumentendarstellung über Projektor und TFT-Bildschirme.

Der Anforderungskatalog ist Basis für den Konzeptentwurf der Bodentestanlage bezüglich Aufbau und Funktionsprinzip. BILD 4 zeigt eine schematische Darstellung des HIL-Simulators. Dieser besteht aus dem Flugzeug (STEMME S15 Prototyp) und einer Simulationseinheit.

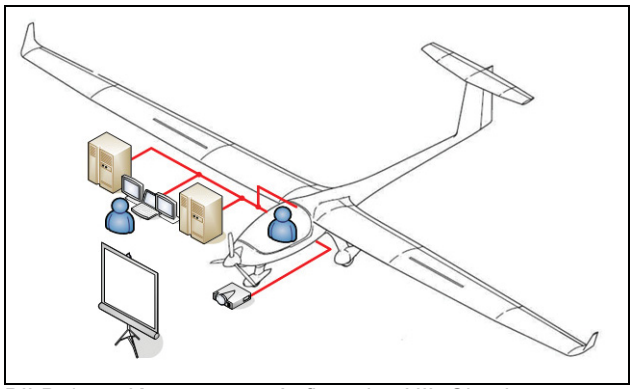


BILD 4. Konzept zum Aufbau des HIL-Simulators

Die Simulationseinheit ist mobil und robust ausgelegt. Die gesamte Hardware ist in zwei fahrbare Transportkoffer integriert und kann temporär mit der S15 verbunden werden. Der Arbeitsplatz des Versuchspiloten befindet sich im Cockpit. Von dort aus steuert er den Autopiloten über das AFCP im Frontpanel. Auch der manuelle Flug über

die Steuerorgane der S15 ist möglich. Zur Orientierung dient dem Piloten die Sichtdarstellung über den Projektor.

## 3.2 Funktionsprinzip

Die grundsätzliche Aufgabe des HIL-Simulators besteht darin, den STEMME S15 Prototyp inklusive des gesamten Flugsteuerungssystems am Boden betreiben zu können. Im Folgenden wird das Funktionsprinzip des Simulators beschrieben. BILD 5 zeigt eine grafische Übersicht dazu.

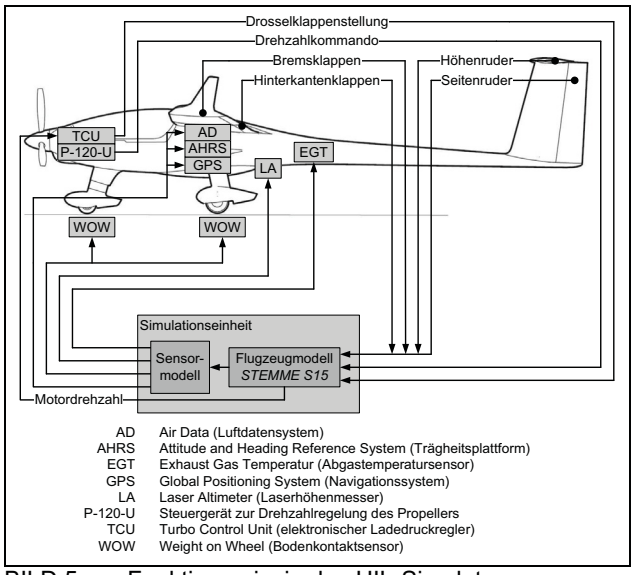



BILD 5. Funktionsprinzip des HIL-Simulators

Aufgrund der fehlenden Flugzeugbewegung ist die gesamte Sensorik des Flugsteuerungssystems außer Betrieb. Dies betrifft:

- die Luftdatensysteme (AD),
- die Trägheitsplattformen (AHRS),
- das Navigationssystem (GPS),
- die Laser-Höhenmesser (LA) und
- die Bodenkontaktsensoren (WOW).

Des Weiteren kommt der Betrieb des Motors aus sicherheitstechnischen sowie aus ökologischen und ökonomischen Gründen nicht in Frage. Außerdem wäre eine realistische Funktion der Antriebseinheit aufgrund der fehlenden Anströmung nicht gewährleistet.

Sensor- und Motordaten dienen der Flugregelung als Eingangsgrößen. Um den korrekten Betrieb der Flugsteuerung zu gewährleisten, müssen die Messwerte in einer Flugsimulation erzeugt werden. Dies geschieht in der Simulationseinheit. Die Simulation beinhaltet ein nichtlineares Flugzeugmodell der STEMME S15. In diesem ist auch die Antriebseinheit modelliert. Als Ausgangsgrößen stehen die aktuelle Drehzahl sowie die Abgastemperatur des Motors zur Verfügung. Das flugmechanische Modell der S15 besteht aus einer sechs Freiheitsgradsimulation des starren Flugzeugs, die um die für die Reglererprobung wichtigen elastischen Freiheitsgrade erweitert wurde. Außerdem enthält die Simulation ein Sensormodell. Dieses teilt sich in ein Dynamikmodell (Totzeiten usw.) und ein Fehlermodell (Rauschen, Bias, Drift usw.) auf. Die simulierten Sensor- und Motordaten werden den Flugsteuerungsrechnern als Eingangsgrößen übermittelt.

Die Flugsimulation benötigt als Eingang die Informationen der Steuerorgane. Die Ausschlagwinkel der Stellflächen werden mit Hilfe von Potentiometern gemessen. Somit entfällt die Modellierung der Gesamtübertragungsfunktion der Ruderansteuerung (Aktuatorik, Gestängekinematik usw.) in der Simulation[2]. Die Antriebseinheit benötigt als Eingangsgrößen die Drosselklappenstellung und das Propeller-Drehzahlkommando. Die Stellung der Drosselklappe liefert die TCU durch interne Messung mit Hilfe eines Potentiometers. Über das Steuergerät zur Drehzahlregelung des Propellers (P-120-U) wird das Drehzahlkommando vorgegeben.

## 3.3 Aufbau

Der HIL-Simulator hat einen ähnlichen Aufbau wie der vom FMRA betriebene Forschungssimulator SEPHIR. Es wird auf Hardware-Komponenten zurückgegriffen, die sich in der Praxis bewährt haben. Dies bietet zusätzlich den Vorteil, dass bestehende Software-Module durch minimalen Programmieraufwand für den HIL-Simulator adaptierbar sind. Die Simulatoren sind modular und überwiegend aus Standard-Einzelkomponenten aufgebaut. Gegenüber einem Gesamtsystem ist mit geringeren Beschaffungs- und Wartungskosten zu rechnen. Des Weiteren ist das System flexibel erweiterbar.

BILD 6 zeigt die wichtigsten Hardware-Komponenten der Simulationseinheit und deren Interaktion.

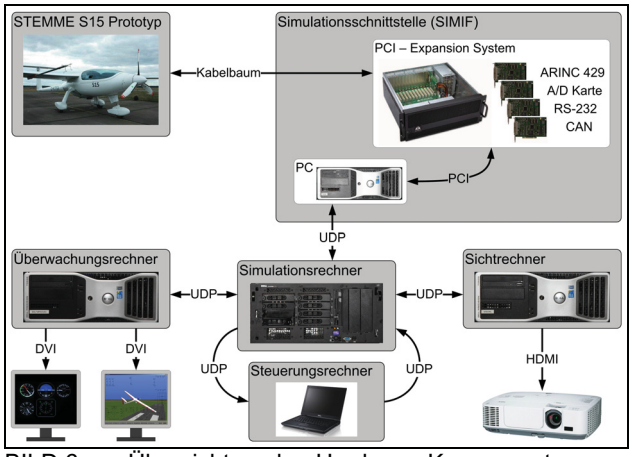



BILD 6. Übersicht zu den Hardware-Komponenten

Auf dem Simulationsrechner wird der eigentliche Simulationsprozess ausgeführt. Zur Verfügung steht ein Hochleistungsrechner auf dem ein UNIX-Betriebssystem installiert ist. Dieses wurde um Echtzeitfunktionalitäten erweitert und dahingehend optimiert, zeitkritische Anwendungen auszuführen. Die Simulation ist mit 125 Hz getaktet.

Die Simulationsschnittstelle (SIMIF – *Simulation Interface*) ermöglicht den Datentransfer zwischen der S15 und dem Simulationsrechner. Sie besteht aus einem konventionellen PC und einem PCI-Expansion-System zur Erweiterung der Steckkartenkapazität. Die Schnittstellenkarten umfassen die Übertragungsstandards ARINC 429, RS-232 und CAN und ermöglichen die Analog / Digital (A/D) Wandlung. Die Ausgänge der Karten sind über

[2] Die Wirkung der Luftkräfte auf Gestänge und Aktuatoren wird nicht simuliert. Dazu wäre eine aufwändige Kraftsimulation an allen Stellflächen erforderlich.

einen Kabelbaum mit den entsprechenden Komponenten des Flugsteuerungssystems der S15 verbunden. Der Prozessablauf des Datentransfers erfolgt durch folgende Schritte in der aufgeführten Reihenfolge:

- Daten vom S15 Prototyp empfangen,
- Daten via UDP an die Simulation senden,
- Daten via UDP von der Simulation empfangen und
- Daten an den S15 Prototyp senden.

Diese vier Schritte bilden einen Prozesstakt. Das Programm wird ebenfalls mit einer Frequenz von 125 Hz ausgeführt. Auf entstehende Latenzzeiten durch Phasenverschiebungen zum Simulationsprozess wird in Abschnitt 4.1 eingegangen.

Die gesamte Kommunikation zwischen den einzelnen Rechnern der Simulationseinheit findet per UDP (*User Datagram Protocol*) statt. Die Steuerung der Simulation erfolgt mit Hilfe eines Notebooks. Dabei sind verschiedene Simulationsparameter in Echtzeit veränderbar. Die Informationen über Position und Bewegung des Flugzeugs werden vom Sichtrechner verarbeitet und mit Hilfe eines Projektors dargestellt. Die entsprechende Software (PHILOSIM) stammt von der Firma PHILOTECH. Die Anzeige einzelner Instrumente und einer Flugzeug-Außenansicht wird mit Hilfe eines separaten Rechners (Überwachungsrechner) generiert und auf den TFT-Monitoren abgebildet.

BILD 7 zeigt den HIL-Simulator an der STEMME S15 während der Funktionsprüfung des automatischen Flugsteuerungssystems.

BILD 7. Betrieb des HIL-Simulators an der S15

## 3.4 Simulationsumgebung

In der ersten Phase des Projekts LAPAZ wurde in zahlreichen Messflügen das Verhalten der S15 identifiziert. Mit Hilfe der gewonnen Daten wurde ein flugmechanisches Modell entwickelt, welches in MATLAB und Simulink umgesetzt ist (siehe [9]). Teile dieses Modells sind Basis für die HIL-Simulation. Folgende Effekte werden im Gesamtmodell simuliert:

1) Flugmechanik in sechs Freiheitsgraden gekoppelt mit aeroelastischen Freiheitsgraden,
2) 2-Punkt Aerodynamik mit Stall,
3) Antriebseinheit,
4) Elastischer Mast mit Anstellwinkelfahne,
5) Bodeneffekt,
6) Fahrwerk,
7) Gelände,
8) Position in WGS84 Koordinaten,
9) Turbulenz,
10) ISA Atmosphäre,
11) Sensorik,
12) Aktuatorik + Steuergestänge und
13) AFCS-Mechanismen (Konsolidierung, Degradation).

Für die HIL-Simulation werden lediglich die Effekte 1) bis 11) benötigt, da die anderen Effekte durch das Flugzeug und den Regler direkt erzeugt werden.

Die Einbindung des Simulationsmodells in den HIL-Simulator erfolgt mit Hilfe eines Rahmenmodells sowie einer Echtzeit-Simulationsumgebung. Die Architektur des entstehenden Programms ist in BILD 8 dargestellt.

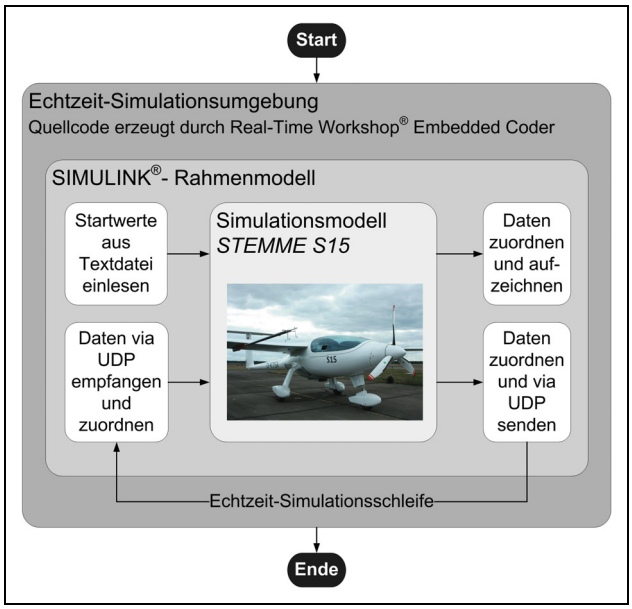



BILD 8. Architektur der Simulationsumgebung

Das Simulink-Rahmenmodell beinhaltet Komponenten für den Datentransfer zwischen der Simulation und den einzelnen Rechnern der Simulationseinheit. Entsprechende UDP-Schnittstellen sind für die Übertragung der Datenpakete zuständig. Dazu wurden eigene *S-Funktion*-Blöcke in der Programmiersprache *C* erstellt. Die empfangenen Daten werden aufbereitet und den Eingängen des Simulationsmodells zugeordnet. Notwendige Startwerte der Simulation liest eine entsprechende Funktion aus einer Text-Datei ein. Die Modellausgänge sind vor dem Senden ebenfalls aufzubereiten und zu ordnen. Dabei werden die Daten für das jeweilige Übertragungsprotokoll der einzelnen Sensoren vorbereitet (z.B. ARINC 429). Das Rahmenmodell verfügt über einen Block zur Datenaufzeichnung in eine externe Datei. Dabei kann der Benutzer über Beginn und Dauer der Aufzeichnung entscheiden. Die Daten werden binär gespeichert.

Rahmen- und Simulationsmodell werden mit einer Simulationsschrittweite von 8 ms berechnet, dies ergibt eine Taktfrequenz von 125 Hz. Sie sind allerdings als Simulink-Modell zunächst nicht echtzeitfähig. Im Allgemeinen läuft die Simulation in Abhängigkeit der Rechnerleistung

schneller oder langsamer als Echtzeit ab. Bezüglich einer Echtzeit-Simulationsumgebung gilt es, die Simulation zu beschleunigen. Ein geeignetes Werkzeug stellt auch hier der RTWEC dar. Er erzeugt *C* Quellcode aus Simulink-Diagrammen. Hauptaugenmerk liegt dabei auf der Simulationsbeschleunigung. Der von RTWEC generierte Quellcode ist für die Echtzeitanwendung zu erweitern. Den Rahmen liefert die Code-Datei `ert_main.c`, die vom RTWEC als Vorlage zur Verfügung gestellt wird. Der gesamte Simulationsprozess wird darin gesteuert. Innerhalb der Hauptfunktion `main()` wird das Gesamtmodell mit Hilfe von POSIX-Programmierschnittstellen [10] durch die Echtzeit-Uhr des Simulationsrechners getaktet.

## 4 VALIDIERUNG

Die Gültigkeit der HIL-Simulation ist nachgewiesen wenn gezeigt werden kann, dass sie zu jeder Zeit das Verhalten des S15 Prototyps abbildet. Verschiedene Untersuchungen wurden in diesem Zusammenhang durchgeführt. Die Validierung des Simulationsmodells der S15 erfolgte im Rahmen der Identifizierung (siehe [9]). Ein Teil der durchgeführten Messflüge wurde zur Modellbildung und der andere Teil zur Verifizierung des flugmechanischen Verhaltens verwendet. Während der Integrationsphase des HIL-Simulators am Flugzeug wurden die simulierten Sensorsignale bezüglich des Übertragungsprotokolls analysiert. Zusätzlich findet automatisch eine Validierung durch das plattformseitige Monitoring des AFCS statt. Eine Plausibilitätsprüfung der erzeugten Sensorwerte erfolgte anhand aufgezeichneter Daten. Auf die Untersuchung der zeitlichen Verzögerungen durch die HIL-Simulation sowie dem Vergleich von Flugversuch und Simulation wird in den nächsten beiden Abschnitten eingegangen.

### 4.1 Latenzzeiten der Simulationseinheit

Die HIL-Simulation kann grob in drei Hauptprozesse unterteilt werden:

1) Automatische Flugsteuerung,
2) Datentransfer und
3) S15-Simulation.

Prozess 1) umfasst die komplexen Flugsteuerungsmechanismen des AFCS von der Verarbeitung der Sensordaten über die Regelung und die Kommandogenerierung bis hin zur Messung der Stellgliedpositionen. Prozess 1) erfolgt auf dem STEMME S15 Prototyp, Prozesse 2) und 3) auf der Simulationseinheit. Neben den Hauptprozessen finden weitere Arbeitsschritte im Rahmen der Datenverarbeitung statt. Im Einzelnen betrifft dies:

- Datenverarbeitung durch Schnittstellenkarten (z.B. A/D-Wandlung),
- Datenübertragung zwischen PCI-Expansion-System und SIMIF-Rechner via PCI,
- Datenübertragung zwischen SIMIF- und Simulationsrechner via UDP.

Diese Prozesse laufen im Bereich von Nanosekunden ab und werden in der weiteren Betrachtung vernachlässigt.

In BILD 9 ist der zeitliche Verzug durch die Generierung der Sensordaten[3] graphisch dargestellt. Das Diagramm bildet alle drei Hauptprozesse ab, wobei die Taktung von Prozess 1) nicht aufgeschlüsselt ist. Die Dauer zwischen Empfang der Reglerkommandos und Generieren der neuen Sensorwerte wird als Latenzzeit definiert. Die Prozesse 2) und 3) laufen auf unterschiedlichen Rechnern (SIMIF- und Simulationsrechner). Die Wahl der gleichen Taktfrequenz (125 Hz) führt zu einem isochronen[4] Prozessablauf. In Abhängigkeit der Phasenverschiebung ergeben sich unterschiedliche Latenzzeiten. Im günstigen Fall a) beträgt die Latenzzeit mindestens 8 ms (oberes Diagramm). Die Phasenverschiebung im ungünstigen Fall b) ergibt eine Latenzzeit von maximal 17 ms (unteres Diagramm). Die Updateraten der wichtigsten Sensoren liegen bei maximal 50 Hz (20 ms Abtastrate). Somit befinden sich die Latenzzeiten im Toleranzbereich.

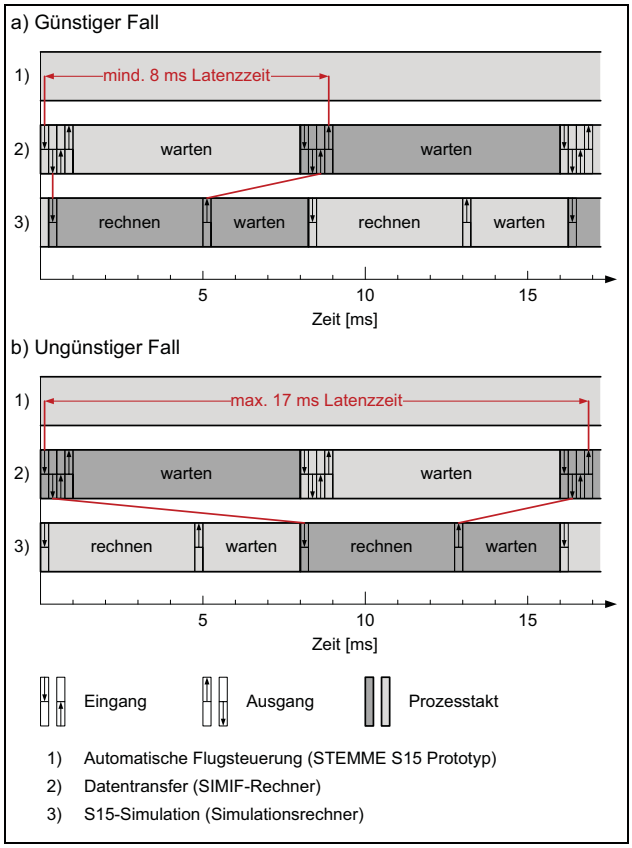



BILD 9. Latenzzeiten der Simulationsschnittstelle in Abhängigkeit der Phasenverschiebung

### 4.2 Vergleich HIL-Simulation und Flugversuch

In diesem Abschnitt werden Ergebnisse aus HIL-Simulation und Flugversuch anhand aufgezeichneter Daten verglichen. Dabei ist das Hauptaugenmerk auf das Regelverhalten des automatischen Flugsteuerungssystems gerichtet. Folgende Anforderungen ergeben sich für die Untersuchungen:

1) Definition reproduzierbarer Flugmanöver,
2) Wahl gleicher Randbedingungen (Referenzzustand, Atmosphäre, Masse, Schwerpunkt, usw.) und
3) Wahl der gleichen Quelle für die Datenaufzeichnung.

[3] Die simulierten Totzeiten sowie die Updateraten der Sensoren sind nicht berücksichtigt.

[4] Die Isochronität hängt von der Genauigkeit der Rechner- und Prozesstaktung ab.

In diesem Abschnitt wird exemplarisch ein Vergleich zwischen der ersten automatischen Landung des S15 Prototyps am 22. März 2012 in Neuhardenberg und den zuvor erfolgten HIL-Simulationen durchgeführt. Das Flugzeug bewegt sich während des automatischen Landevorgangs entlang einer definierten dreidimensionalen Trajektorie mit vorgegebener Geschwindigkeit, somit ist Anforderung 1) erfüllt. Anforderung 3) wird durch die Datenaufzeichnung innerhalb des automatischen Flugsteuerungssystems entsprochen. Dort werden alle Sensorwerte in der Form erfasst, wie sie auch der Regelung zur Verfügung stehen. Anforderung 2) wird im betrachteten Fall nur bedingt erfüllt. Dies liegt in erster Linie an der Wahl eines HIL-Versuchs, welcher vor dem eigentlichen Flugversuch stattgefunden hat. Die Nachbildung atmosphärischer Störungen ist dadurch nicht möglich. In TAB 1 sind die Bedingungen für Simulation und Flugversuch zusammengefasst.

| | HIL-Simulation | Flugversuch |
|---|---|---|
| **Turbulenz** | - | gering |
| **Wind** | - | leichter Seitenwind |

TAB 1. Atmosphärische Bedingungen für HIL-Simulation und Flugversuch

Die Seitenwindkomponente spielt bei der Längsbewegung des Flugzeugs eine untergeordnete Rolle. Die Betrachtungen werden sich deshalb auf die Bewegung in der Symmetrieebene beschränken. Masse und Schwerpunkt liegen für Flugversuch und Simulation etwa im gleichen Bereich.

BILD 10 und BILD 11 stellen wichtige Parameter der Flugzeuglängsbewegung über der Distanz zur Landebahnschwelle[5] dar. In BILD 10 sind die Sensorwerte für Längslagewinkel $\Theta$, Bahnwinkel $\gamma$, Höhe über MSL (*Mean Sea Level*) $H_{MSL}$ und kalibrierte Fluggeschwindigkeit (CAS – *Calibrated Air Speed*) $V_{CAS}$ aufgetragen. BILD 11 vergleicht die Höhe über MSL mit der Höhe über Grund (GND – *Ground*) $H_{GND}$. Grundsätzlich kann festgestellt werden, dass die Sensorwerte der HIL-Simulation einen relativ kontinuierlichen Verlauf zeigen, während die Flugversuchsdaten verrauscht sind. Dies ist den atmosphärischen Störungen zuzurechnen. Während Flugversuch und Simulation war die Reglerfunktion zur Böenabminderung nicht aktiv.

Das vertikale Flugprofil, welches im dritten Diagramm von BILD 10 durch den Verlauf der Höhe über MSL dargestellt ist, hat während des Endanflugs einen nahezu identischen Verlauf für Simulation und Flugversuch. Dies trifft auch grundsätzlich für den Bahnwinkel im zweiten Diagramm von BILD 10 zu. Lediglich der Punkt, an dem der Anfangsgleitwinkel von $\gamma = -5°$ eingenommen wird variiert. Der Punkt wird durch den Regler nicht in Abhängigkeit der Distanz zur Landebahnschwelle, sondern in Abhängigkeit der Anflughöhe bestimmt. Diese ist im Flugversuch höher gewählt. Folglich muss der Gleitpfad in größerer Entfernung von der Schwelle eingenommen werden, damit das Flugzeug dem vertikalen Flugprofil folgen kann. Bis zu einer Distanz von 286 m vor der Landebahnschwelle (−286 m) gibt der Regler einen Bahnwinkel von $\gamma = -5°$ vor. Ab diesem Punkt wird der Gleitpfad auf $\gamma = -3°$ reduziert. Dieser Übergang ist im zweiten und dritten Diagramm von BILD 10 gut zu erkennen. Diesbezüglich zeigen Flugversuch und HIL-Simulation prinzipiell das gleiche Regelverhalten. Sowohl der Längslagewinkel (erstes Diagramm BILD 10) als auch die Fluggeschwindigkeit (viertes Diagramm BILD 10) haben im Sinkflug deutlich unterschiedliche Werte für Simulation und Flugversuch. In dieser Flugphase kommandiert der Regler das 1,3-fache der Überziehgeschwindigkeit ($V_s$). Für Simulation und Flugversuch entspricht dies annähernd dem gleichen Wert von ca. 33 m/s. Zu erkennen ist, dass die S15 in der Realität dem Kommando folgen kann, wogegen sie in der Simulation etwa 3 m/s zu schnell fliegt. Ursache ist der Widerstand der aerodynamischen Bremsklappen, der im flugmechanischen Modell zu gering angenommen wurde. Die höhere Fahrt bei gleichem stationären Bahnneigungsflug, hat einen geringeren Anstellwinkel $\alpha$ zur Folge. Der Zusammenhang:

(1) $\Theta = \gamma + \alpha$ , (ohne Wind)

erklärt den geringeren Längslagewinkel in der HIL-Simulation.

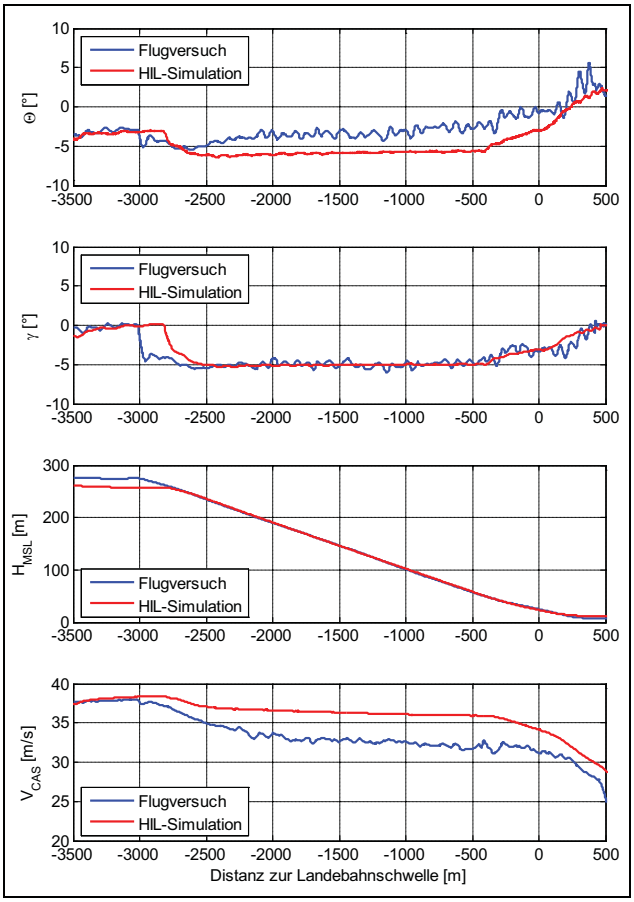



BILD 10. Längslagewinkel $\Theta$, Bahnwinkel $\gamma$, Höhe $H_{MSL}$ und Fahrt $V_{CAS}$ über der Distanz zur Landebahnschwelle

Im dritten Diagramm von BILD 10 ist zu erkennen, dass die simulierte Höhe über MSL, nach dem Aufsetzen der S15 auf der Landebahn (ca. 450 m nach der Schwelle), von der real gemessenen Höhe abweicht. BILD 11 soll dies verdeutlichen. Das erste Diagramm zeigt, dass die Abweichung etwa 3 m beträgt. Die Messung der Höhe über MSL erfolgt GPS-basiert. Die veröffentlichte Höhe der Landebahn 08 in Neuhardenberg beträgt laut AIP (*Aeronautical Information Publication*) 10 m über MSL. Entsprechend ist die Landebahn für die Simulation in

[5] Die Landebahnschwelle liegt im Ursprung der Abszisse.

dieser Höhe platziert. In der Realität misst der GPS-Sensor des AFCS eine Landebahnhöhe von ca. 6-7 m MSL. Dieser Unterschied liegt an der Meßgenauigkeit, die bei EGNOS (*European Geostationary Navigation Overlay Service*) gestützter GPS-Höhenbestimmung erreichbar ist.

Hervorzuheben ist, dass die erste automatische Landung trotzdem erfolgreich durchgeführt werden konnte. Dies ist nicht nur der robusten Auslegung, sondern auch dem Design des Reglers zu verdanken. Bis zum Erreichen der Landebahnschwelle wird die Höhe über MSL als Regelgröße verwendet. Das vertikale Flugprofil gibt vor, dass das Flugzeug die Schwelle in 15 m GND überfliegt. Dazu muss dem Regler die Information über die Höhe der Landebahn zur Verfügung stehen (für HIL-Simulation und Flugversuch wurde als Landebahnhöhe 10 m MSL eingestellt). Entsprechend überfliegt die S15 im Flugversuch die Schwelle 3 m zu hoch in ca. 18 m GND (zweites Diagramm BILD 11). Dieser Wert liegt im Toleranzbereich, sodass die Landung nicht abgebrochen werden muss. Über der Landebahn wird die Höhe über GND als Regelgröße verwendet. Der Regelfehler wird bis kurz vor dem Aufsetzen abgebaut. Der S15 Prototyp kann somit wieder dem vorgegebenen Flugprofil folgen.

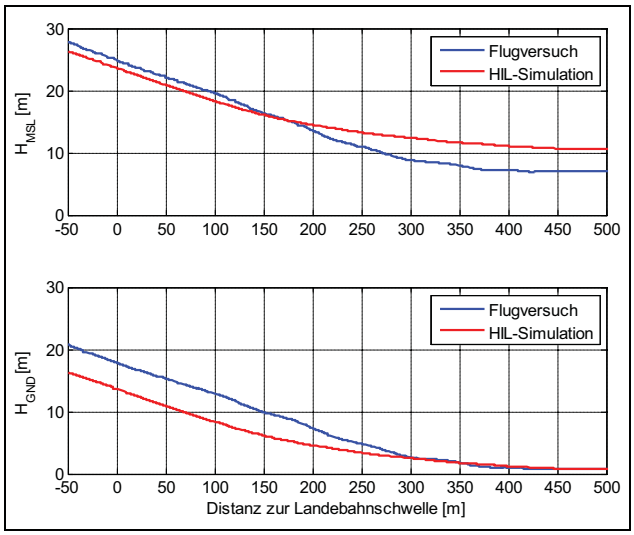



BILD 11. Höhe über MSL und GND über der Distanz zur Landebahnschwelle

## 5 FAZIT UND AUSBLICK

Im Rahmen des Forschungsprojekts LAPAZ wurde ein Hardware-in-the-Loop – Simulator aufgebaut. Dieser wird erfolgreich für die Funktionsprüfung des automatischen Flugsteuerungssystems der STEMME S15 verwendet. Der HIL-Simulator trägt als wichtiges Element des Entwicklungsprozesses maßgeblich zum Erfolg des Forschungsvorhabens bei. Vor den Integrationstests auf dem HIL-Simulator werden die Rechnerplattform, ihr Redundanzmanagement und die korrekte Integration der FCL-Software auf einem speziell dafür konzipierten Testsystem beim Projektpartner ILS rigoros getestet. Die Durchführung von HIL-Simulationen vor den eigentlichen Flugversuchen minimiert die Risiken für Testpilot und Flugzeug. Das Konzept, den S15 Prototyp in die Simulation einzubinden, hat sich bewährt. Im Vergleich zu einem *Iron Bird* konnten mit der mobilen Simulationseinheit Kosten und Zeit für Aufbau und Entwicklung sowie Nutzfläche bezüglich der Lagerung gespart werden. Aufbau und Einschalten des HIL-Simulators nimmt weniger als 15 min in Anspruch. Das Cockpit des Prototyps ist für den Testpiloten eine gewohnte Arbeitsumgebung. Notfälle sind realistisch darstellbar und die entsprechende Pilotenreaktion kann analysiert bzw. trainiert werden.

Der Vergleich von Simulation und Flugversuch in Abschnitt 4.2 hat gezeigt, dass der HIL-Simulator das Verhalten des STEMME S15 Prototyps realistisch wiedergibt. Wichtige Erkenntnisse aus den HIL-Simulationen konnten in die Weiterentwicklung des flugmechanischen Modells und des Simulators fließen. Das automatische Flugsteuerungssystem unterliegt einem fortlaufenden Entwicklungsprozess. Der modulare Aufbau des Simulators erleichtert die Erweiterung des Systems im Bedarfsfall.

## SCHRIFTTUM

[1] SAE: *Certification Considerations for Highly-Integrated or Complex Aircraft Systems*, ARP 4754, Systems Integration Requirements Task Group, AS-1C, ASD, SAE, Warrendale 1996

[2] RTCA: *Software Considerations in Airborne Systems and Equipment Certification*, DO-178B, RTCA, Inc., Washington, D.C. 1992

[3] Brockhaus, R.; Alles, W.; Luckner, R.: *Flugregelung*, 3. Auflage, ISBN 978-3-642-01442-0, Springer-Verlag, Berlin Heidelberg 2011

[4] EASA: *Certification Specification for Normal, Utility, Aerobatic and Commuter Category Aeroplanes*, CS-23, EASA 2003

[5] Hesse, S.; Reichel, R.; Görke, S.; Dalldorff, L.: *Eine skalierbare Plattform für sicherheitskritische, automatische Flugsteuerungssysteme der allgemeinen Zivilluftfahrt*, Deutscher Luft- und Raumfahrt Kongress 2009

[6] Hesse, S.; Görke, S.: *An Affordable, Fault-Tolerant Automatic Flight Control System for the Utility Aircraft Stemme S15*, CEAS EuroGNC 2011, Conference in Guidance, Navigation & Control in Aerospace, München 2011

[7] Polenz, S.; Cake, F.; Görke, S.; Reichel, R.: *SAFAR eine Fly-by-Wire Steuerung für ein Flugzeug der General Aviation (DIAMOND DA42)*, Deutscher Luft- und Raumfahrt Kongress 2011

[8] Lamp, M; Luckner, R.: *Flight Control Law Development for the Automatic Flight Control System LAPAZ*, CEAS EuroGNC 2011, Conference in Guidance, Navigation & Control in Aerospace, München 2011

[9] Meyer-Brügel, W.; Luckner, R.: *Flight Mechanical Simulation Models for Design and Test of Automatic Flight Control Functions*, CEAS EuroGNC 2011, Conference in Guidance, Navigation & Control in Aerospace, München 2011

[10] Burns, A.; Wellings, A.: *Real-Time Systems and Programming Languages: Ada 95, Real-Time Java and Real-Time C / Posix (Third Edition)*, ISBN: 0-201-72988-1, Addison Wesley Langmain, March 2001