IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

Design And Verification Of Flight Data Acquisition System Using Uvm

[1]Dr.Komala K, [2]Aditya R D, [3]Chinmai Gowda M, [4]Darshan K S, [5]Hanmant B N
[1]Associate Professor, [2]UG Student, [3]UG Student, [4]UG Student, [5]UG Student
[1]Department of Electronics and Communication Engineering,
[1]Sri Siddhartha Institute of Technology, Tumkur, India

Abstract: The Flight Data Acquisition Unit (FDAU) collects discrete, analog, and digital parameters from various aircraft sensors and avionics, forwarding them to the Flight Data Recorder (FDR). Modern systems employ real-time communication and advanced networking to globally monitor aircraft performance. Key sensed parameters include fuel level, pressure, temperature, airspeed, altitude, and vertical acceleration. An interface signal module handles data format and level conversion, while the FPGA based acquisition module supports parallel data collection. This paper presents a configurable data acquisition system using an FPGA+CPU architecture, leveraging FPGA is parallelism to capture diverse flight parameters efficiently. The system is scalable, highly integrated, and suitable for monitoring engine, airframe, and flight parameters, aiding both accident investigation and preventive maintenance.

Index Terms -**Aircraft, Analog, Avionics, CPU, FPGA, FDR,Flight Data Acquisition, Highly integrated.**

I. Introduction

The aviation industry's rapid technological advancements have led to an exponential increase in flight datavolume and complexity, necessitating more efficient and reliable data acquisition systems. Traditional systems, often reliant on multiple acquisition and control boards, struggle to meet these demands due to limitations in speed, scalability, and adaptability. To address these challenges, we propose a configurable Flight Data Acquisition System (FDAS) leveraging Field-Programmable Gate Arrays (FPGA) for real-time, parallel data processing and the Universal Verification Methodology (UVM) for robust design verification.

FPGAs offer customizable hardware solutions capable of handling high-speed data processing and interfacing with various sensors and communication protocols, making them ideal for the dynamic requirements of modern avionics systems. The integration of UVM provides a standardized, reusable, and scalable verification framework, ensuring the correctness and reliability of the FPGAbased FDAS. By simulating diverse flight scenarios and sensor inputs, UVM facilitates comprehensive testing, enabling the identification and rectification of potential issues early in the development process. This approach ensures robust system performance and accelerates development for aviation-grade safety and reliability.

II. Literature Survey

In [1] this literature surveythe paper discusses on computing and storing multiple types of flight data in real time. While the system effectively captures diverse flight parameters, it exhibits significant complexity in its design framework. This complexity necessitates considerable time and effort from professional technicians for operation and maintenance, potentially impacting efficiency and scalability in practical applications.
In [2] this literature survey the paper presents an FPGA-based implementation of a multi-protocol data acquisition system, utilizing both parallel and serial data transfer protocols, including SPI, I²C, and One-Wire, all modeled using VHDL. The system is designed to interface with digital signals from multichannel sensors and various ADC protocols, aiming to enhance flexibility in data communication.

In [3] this literature surveythe paper discusses on verifying digital subsystem designs by selecting commonly used data buses as the subject under test. The primary contribution lies in demonstrating the setup of a verification environment using these data buses. However, the work primarily serves as a demonstration of establishing a verification framework, rather than providing a comprehensive verification methodology or in-depth analysis of the subsystem's performance. Consequently, while it offers insights into the initial stages of verification environment setup, it lacks detailed exploration of verification strategies and their effectiveness in complex scenarios.

In [4] this literature survey the paper presents the design and implementation of an FPGA-based Data Acquisition (DAQ) system, demonstrating how an entire system can be realized within a single FPGA fabric. This approach aims to reduce development time and cost by minimizing the need for external components. However, the system is limited to interfacing with only one sensor, which restricts its applicability in scenarios requiring multi-sensor data acquisition. Consequently, while the design showcases the potential for compact and cost-effective DAQ systems, its limited scalability poses challenges for broader applications.

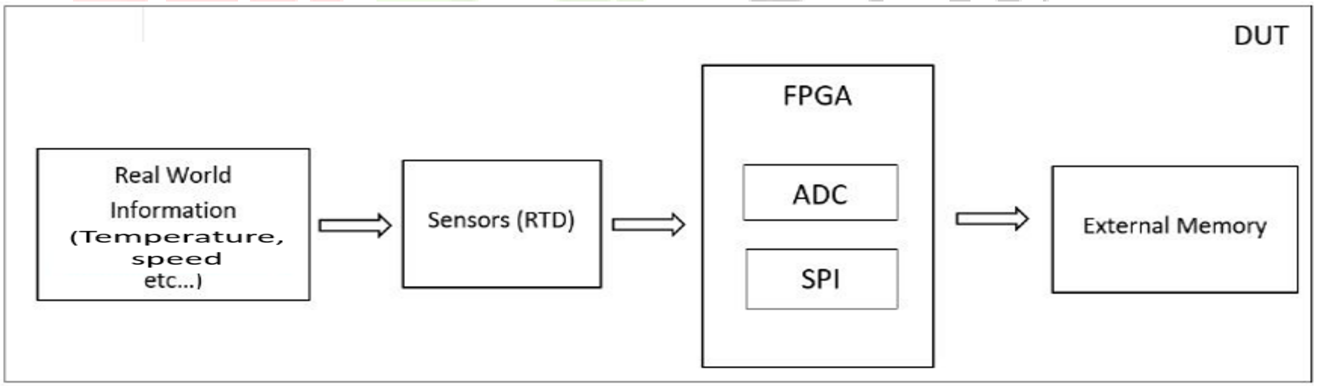
In [5] this literature survey the paper explores FPGA-based sensor data acquisition and noise reduction, emphasizing the need for real-time data integrity, which UVM can ensure through comprehensive verification setups.

In [6] this literature survey the paper presents a verification study of the ARINC-429 data bus digital design, utilizing the Universal Verification Methodology (UVM) to describe the DO-254 verification process. The study demonstrates UVM's effectiveness in certifying airborne electronics by implementing functional coverage, constraint random verification, and automated tests. The approach ensures comprehensive verification of custom micro-coded components against requirements, aligning with the stringent safety standards mandated for avionics hardware. This work highlights UVM's role in enhancing the reliability and safety of flight data systems through systematic and automated verification processes.

In [7] this literature survey The paper explores cloud-based solutions formanagingandprocessingspaceflight test handling data, and focusing enhanced system on real scalability.

In [8] this literature survey The paper discusses on measuring network delay in AFDX systems during flight tests, emphasizing the importance of accurate data acquisition, which UVM can address through precise simulation checks.

III. Methodology

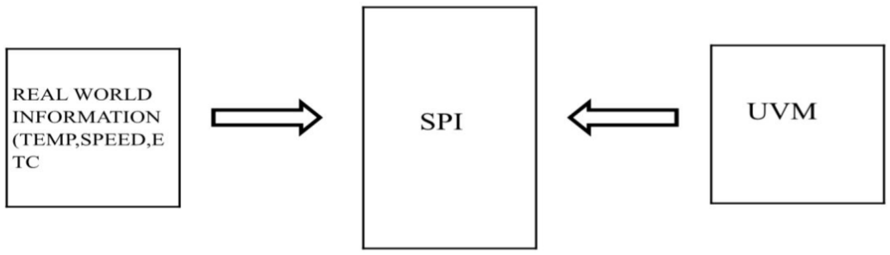


A flight data acquisition unit (FDAU) receives various discrete, analog and digital parameters from a number of sensors and avionic systems and then routes them to a flight data recorder (FDR) and, if installed, to a Quick Access Recorder (QAR). Information from the FDAU to the FDR is sent via specific data frames, which depend on the aircraft manufacturer. An FDAU collects, samples, conditions, and digitizes analog signals representing aircraft functions but does not normally have the capability to condition digital signals **.**

The system consists of ADC, interface signal module, CPU data processing module, FPGA data acquisition module. This system mainly comprises sensor, which collects the data from the extrnal world. The configurable aircraft data acquisition system mainly collects engine sensor parameters, fuselage sensor parameters, altitude, atmosphere and other flight data. Analogue to Digital Converter, or ADC, is a data converter which allows digital circuits to interface with the real world by encoding an analogue signal into a binary code. To process the analog signal onto digital devices like as FPGA it should be converted as digital form. After the signal conversion, data is processed using FPGA.

Every FPGA will have an on-board ADC which converts analog data obtained from sensors to digital data. To send data between microcontrollers and small peripherals such as ADC, shift registers, sensors we need an interface module. The interface signal module is used to collect and output various flight data, complete signal format and level conversion.
Therefore, the main function of the FPGA data acquisition module is to collect many kinds of signal types. FPGA, as the core of the data acquisition system, collects and stores the data. This system is divided into three modules: the front-end signal processing module, FPGA data acquisition module, and data storage module.

Verification using UVM



Universal Verification Methodology (UVM) is a standardized method for verifying integrated circuit designs introduced in 2011. UVM, an Accellera standard, is widely used globally. UVM is based on object-oriented programming in SystemVerilog, promoting reusability which saves time and money in projects.
The concept underpinning the Universal Verification Methodology (UVM) is centered on Coverage Driven Verification (CDV). This methodology amalgamates automated test generation, self-verifying testbenches, and coverage metrics to delineate advancements in design verification. The CDV process diverges from the conventional directed-testing paradigm. In this approach, a testbench developer initiates an organized plan by establishing verification objectives. These objectives are subsequently pursued through a meticulously crafted testbench, which generates valid stimuli and transmits them to a Device Under Test (DUT). Progress is assessed through coverage monitors integrated into the simulation environment, facilitating the identification of unexercised functionalities. Comprehensive design validation can be achieved by varying testbench parameters and randomization seeds.
To expedite the verification objectives, the simulation can be refined through the imposition of test constraints within the testing environment.While the Constrained-Direction Verification (CDV) methodology accommodates both directed and constrained-random approaches, the synthesis of these strategies is often deemed optimal: users can allow constrained-random testing to undertake the majority of the workload before transitioning to the labor-intensive, deterministic tests that target specific scenarios, which are challenging to achieve through random means. Within the Universal Verification Methodology (UVM) library, meticulously designed for this methodology, all requisite tools are delineated for inclusion in the architectural framework. It is incumbent upon the user to establish their own verification environment utilizing these tools. Each element within the UVM serves a distinct and purposeful function. Concurrently, the component communication network within the verification environment is automatically facilitated by the UVM. This enables the verification team to concentrate their efforts on the paramount task of delivering input data and meticulously observing the output data.
Verification requirements determined by the requirements capture process are converted to the verification design level requirements and linked to each other. It is expected that all requirements will be met as a result of the tests which run on the device under test (DUT) with the UVM environment and constraints that determined by the requirements. There are three coverage methods to measure whether all requirements are adequately met or not. These are functional coverage, code coverage, and assertion coverage. The entire verification process is completed when these three coverage metrics reach their target values. The coverage information is obtained as a report from the tools.

➢ **Tools used for the project work**

1. **Software tools:**

- ModelSim - Modelsim is a program created by Mentor Graphics used for simulating VHDL and Verilog designs. Simulation is a critical step of designing FPGAs.
- QuestaSim - QuestaSim is part of the Questa Advanced Functional Verification Platform and is the latest tool in Mentor Graphics tool suite for Functional Verification.
- QuartusPrime – To compile designs, perform timing analysis, simulate a design's reaction to different stimuli and to configurate the target device with the programmer.

2. **Hardware Kit:**Intel FPGA

IV. Block Diagram and Implementation

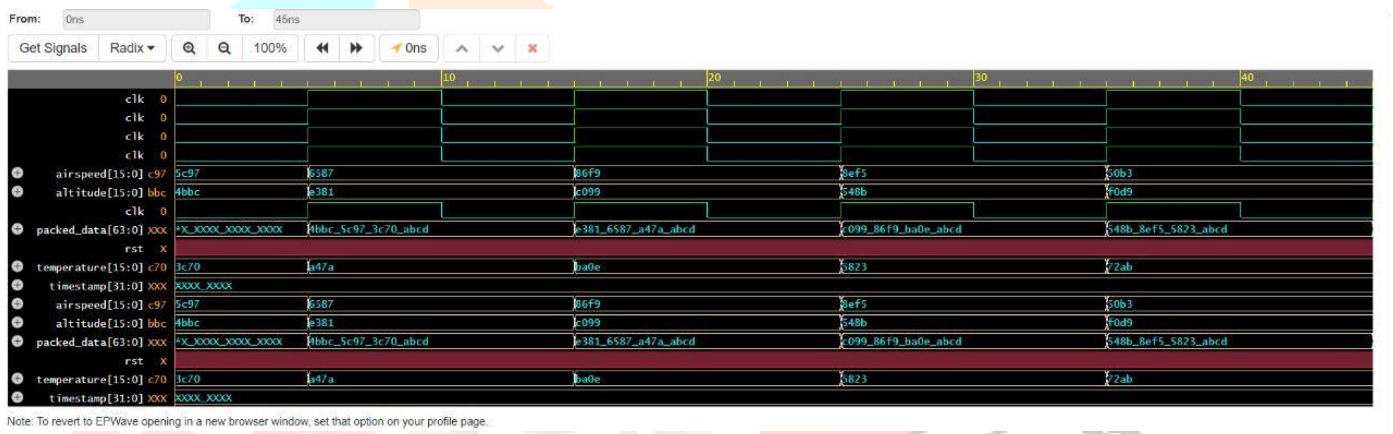
Collecting data from the external world involves gathering a wide array of information such as engine sensor parameters, fuselage sensor parameters, altitude, atmosphere conditions, and various other flight data. This data is then processed by converting the analog signals into digital format using an Analog to Digital Converter (ADC). The ADC plays a crucial role as it enables digital circuits to interact with real-world data by translating analog signals into binary code.
To facilitate the communication between the analog and digital domains, an interface signal module is utilized. This module is responsible for collecting the flight data, ensuring complete signal formatting, and performing level conversions as needed. For instance, it processes signals from sensors measuring temperature, pressure, velocity, and other critical parameters essential for safe flight operations.
In the data acquisition process, the Field-Programmable Gate Array (FPGA) initiates the signal conversion by sending signals to the ADC and then awaits the converted data. Once the ADC completes the conversion, it sends an acknowledgment back to the FPGA, indicating that the data is ready for retrieval. The FPGA then reads and processes the data, repeating this cycle continuously to ensure real-time data acquisition.
Moreover, the FPGA data acquisition module utilizes a parallel operation mode to efficiently capture various types of flight data. This approach enhances the system's capability to handle multiple data streams simultaneously, contributing to improved performance and reliability during flight operations.
To validate the system's functionality and performance, a sophisticated verification methodology known as Universal Verification Methodology (UVM) is employed. UVM offers an open-source environment that enables hardware designers to verify their designs effectively. By using UVM, engineers can ensure that the system meets all specified requirements and functions as intended, enhancing overall system quality and reliability.

V. Results And Discussions

Every FPGA will have an on-board ADC which converts analog data obtained from sensors to digital data. To send data between microcontrollers and small peripherals such as adc,shift registers, sensors we need an interface module. A serial peripheral interface (SPI) is an interface that enables the serial (one bit at a time) exchange of data between two devices, one called a master and the other called a slave . The SPI module allows a duplex, synchronous, serial communication between the MCU and peripheral devices. Software can poll the SPI status flags or SPI operation can be interrupt driven.
The four associated SPI port pins are dedicated to the SPI function as: Slave select (SS), Serial clock (SCK), Master out/slave in (MOSI), Master in/slave out (MISO). The main element of the SPI system is the SPI data register. The 8-bit data register in the master and the 8-bit data register in the slave are linked by the MOSI and MISO pins to form a distributed 16-bit register.
When a data transfer operation is performed, this 16-bit register is serially shifted eight bit positions by the SCK clock from the master; data is exchanged between the master and the slave. Data written to the master SPI data register becomes the output data for the slave, and data read from the master SPI data register after a transfer operation is the input data from the slave.

When a transfer is complete, received data is moved into a receive data register. This 8-bit data register acts as the SPI receive data register for reads and as the SPI transmit data register for writes. A single SPI register address is used for reading data from the read data buffer and for writing data to the shifter. For verifying the result, we took input from the temperature sensor and convert it to digital values.
There is a MASTER and a SLAVE mode. The MASTER device provides the clock signal and determines the state of the chip select lines, i.e. it activates the SLAVE it wants to communicate with. CS and SCKL are therefore outputs.The SLAVE device receives the clock and chip select from the MASTER, CS and SCKL are therefore inputs.This means there is one master, while the number of slaves is only limited by the number of chip selects.
A SPI device can be a simple shift register up to an independent subsystem. The basic principle of a shift register is always present. Command codes as well as data values are serially transferred, pumped into a shift register and are then internally available for parallel processing. Here we already see an important point, that must be considered in the philosophy of SPI bus systems: The length of the shift registers is not fixed, but can differ from device to device.
Normally the shift registers are 8Bit or integral multiples of it. Of course there also exist shift registers with aan odd number of bits. If a SPI device is not selected, its data output goes into high-impedance state (hi-Z), so that it does not interfere with the currently activated devices. When cascading several SPI devices, they are treated as one slave and therefore connected to the same chip select.

Simulation results:


From: 0ns To: 45ns

Note: To revert to EPWave opening in a new browser window, set that option on your profile page.

Later, we need to implement this on FPGA to transfer data from ADC to FPGA. Since we are designing a system for a flight management, accurate flight data cannot be measured in a ground level, so we will create a data sheet having a different flight parameters values and will design a system to compare those values with the measured values. The main aim of a project is to verify the system. After designing a system we'll verify it using a Universal verification methodology where flight data acquisition system acts as a DUT.

VI. Conclusion

According to the requirement of the flight data acquisition, the system uses the FPGA parallel processing mode to modularized the design of the signal acquisition, and uses the bus control to initialize the configuration register in the FPGA interface modules, so as to realize the flexible configuration of multiple types of flight data acquisition and storage. The FPGA-based data acquisition system can convert analog signals from a sensor into digital signals through an ADC, which is the most important part of the whole system.Sucessfully we have written a code for SPI interface and simulated to get proper results.

VII. References

[1]Ning Jia, Hang Chen and Jun Tian "A Design of Configurable Multi-type Flight Data Acquisition System", IEEE conference on communication and computing,2021

[2]Guojin Peng, Wei Zhang, Manting Liu "A method of Data Acquisition Network Delay Measurement for AFDX Avionics System in Flight test",7th international conference on signal
and image processing, 2022

[3]John W. Dyer and Benjamin Douglas "Portable airborne Data Acquisition for Flight Testes" , "IEEE international instrumentation and measurements Technology Conference Proceedings",2019

[4]G.V.Jayaramaiah and Chetan.Umadi "Fpga implementation of multi-protocol data acquisition system using vhdl"July-2020

[5]Ufuk Sakarya and Ibrahim Hokelek "Universal Verification Methodology Application of ARINC429 for Airborne Electronic Hardware Certification",2021

[6]. Harikrishnan.K, Vishwas.H.N, Vineetha Jain K.V, Dr.Ramesh Chinthala, " Sensor data acquisition and de-noising using fpga", 2020

[7]J. AN, S. LI, et al. "Design of Architecture of Spaceflight Test Data Cen ters Using Cloud Platform," Journal of Spacecraft TT&C Technology, v ol. 35 pp.: 137-145, 2016.

[8]D. Laney, 3D data management: controlling data volume, velocity, and variety, META Group Research Note, 2019.

[9]B.Zhao, Design and research of engine parameter collector system[D].Xi'an,Northwestern Polytechnical University, 2018.

[10]J. Zhng, G. Peng, B. Liu, W. Zhang, Hardware design of data acquisition system for a certain aircraft engine. Modern Electronics Technique," Xi'an, vol. 37,pp. 67-69, November 2014.