Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

FPGA-Based Data Acquisition and Telemetry System with Fault Detection using Verilog HDL

Anamika Chaturvedi
Department of Electronics Engineering
Rajkiya Engineering College
Sonbhadra, Uttar Pradesh, 231206, India

Shrivilas Mishra
Department of Electronics and Communication Engineering
Madan Mohan Malaviya University of Technology
Gorakhpur, Uttar Pradesh, 273010, India

Abstract - All types of scientific experiments, as well as industrial applications, aerospace applications, and applications for monitoring embedded systems, require Data Acquisition Systems (DAQ) to collect, process, and transmit sensor data in real-time. The goal of this study is to create a design for a FPGA-based data acquisition and telemetry system based on Xilinx and Verilog HDL. The complete system consists of four primary components: sensor data generator, telemetry packet generator, fault detection circuit, and UART interface. Sensor data (Temperature, Voltage, Current) is continuously being acquired and formatted into telemetry packets to be transmitted over the communication link. The system also has a fault detection circuit that will identify abnormal operating conditions based upon preset thresholds. This FPGA DAQ was designed and tested using Xilinx Vivado software through behavioral simulation only. The final synthesized and implemented design only consumed a small percentage of the Spartan-7 (XC7S50CSGA324-1) FPGAs' resources. The proposed architecture can be scaled very easily and is an efficient means of performing real-time monitoring and telemetry.

Keywords - FPGA, Data Acquisition System, Telemetry, Verilog HDL, UART Communication, Fault Detection, Spartan-7, Vivado.

I. INTRODUCTION

FPGA, or field programmable gate array, plays a key role in the creation and application of Data Acquisition Systems (DAS) and has been used in applications such as modern-day scientific instruments, industrial automation systems, aerospace system and embedded monitoring systems. The purpose of a DAS is to take data from sensors, process that data, and then send it to another system (monitoring unit or control unit) for future analysis. As the need for real time monitoring increases; processor-based DASs are becoming limited in their processing speed, scalability and deterministic operation [1].

FPGA represents an attractive option for implementing DAS because of their ability to process data in parallel, reconfigure (change functionality of previously configured, logic elements) and low latency and compute efficiently. Compared to microcontroller-based configuration solutions, an FPGA can perform many of the same processes concurrently which makes them ideal for real-time data acquisition and telemetry applications. The significant benefit experienced through the use of FPGA based DAS is that they have been used extensively in applications such as high energy physics experiments, aerospace telemetry, industrial monitoring and telecommunications systems [2],[3]. Telemetry is a vital component of monitoring architecture today allowing sensor data from remote locations to be transmitted back to a central monitoring station. Continuous monitoring of parameters such as temperature, voltage and ampere is crucial for validating that a system is operating correctly. Typically, the acquired data is packaged into structured packets prior to transmission to provide for efficient transmission and interpretation of the sensor data [4].

An additional requirement for DAQ systems is the capability to perform fault detection in the field. Faults can result from component wear out, electrical noise, environmental changes, or mis-calibration of sensors. Detecting an anomaly early helps improve reliability and prevent unexpected failures of a system. Techniques that use threshold-based detection are often employed because they are simple, have minimal hardware requirements, and can be easily implemented with FPGAs [5].

Universal Asynchronous Receiver Transmitter (UART) communication is one of the most widely used serial communication protocols in embedded systems. Its associated low number of external resources compared to other protocols and simple integration into designs are two reasons for its popularity. Using a UART to perform telemetry transmission provides an efficient and cost-effective means for transferring sensor data from FPGA platforms to external monitoring devices [6].

The use of FPGA DAQ systems for real-time monitoring and telemetry has recently been proven by studies conducted by Choi et al., who created a multi-channel FPGA system that processes and communicates data rapidly [7], and by Bonequi et al., who demonstrated the use of an FPGA DAQ system to make measurements in the field of electrophysiology [8]; both studies provide evidence of the growing significance of FPGA technology within the scope of modern DAQ systems.

Due to these advantages, this paper describes the design and implementation of an FPGA-based Data Acquisition and Telemetry system utilizing Verilog HDL, which consists of four core components: (1) Sensor Generator, (2) Telemetry Packet Generator, (3) Fault Detection Unit, and (4) UART Transmitter. The system will continuously measure and format sensor parameters such as temperature, voltage, and current into a telemetry packet. A fault detection mechanism based on pre-defined thresholds will identify any potential failures, and data will be transmitted through a UART interface. This complete design and implementation will be accomplished using the AMD Vivado synthesis tools and the Spartan-7 (XC7S50CSGA324-1) FPGA target device.

Summarized Contributions of this Project:

1. FPGA Based Modular DAQ Architecture Designed in Verilog Hardware Description Language.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

2. Telemetry Packet Generation Designed for Structured Sensor Data Transmission.
3. Threshold-Based Fault Detection Mechanism Designed.
4. UART Based Telemetry Communication Fitted into the Project.
5. All Aspects of the Project Verified through Simulation, Synthesis, and Implementation Using AMD Vivado Design.

II. PROPOSED SYSTEM ARCHITECTURE AND METHODOLOGY

The FPGA-based Data Acquisition and Telemetry System suggested is intended to acquire sensor data, generate telemetry packets, recognize conditions that are outside of normal operation, and transmit the processed data via a serial communication interface. The complete architecture was designed in the AMD Vivado design environment using Verilog HDL and implemented in a Spartan-7 (XC7S50CSGA324-1) FPGA. The overall system architecture is illustrated in Figure 1.

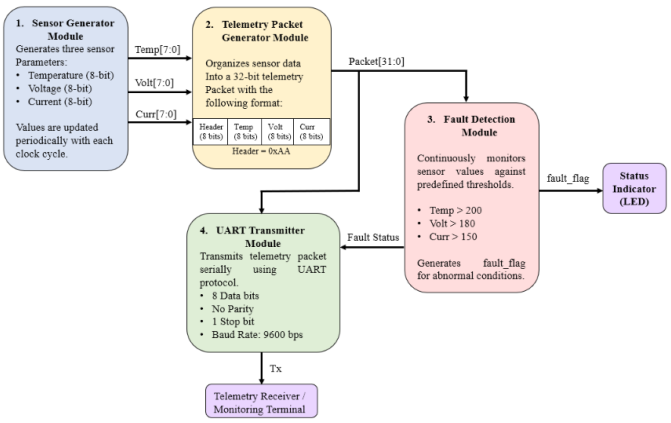

Fig. 1 Overall System Architecture of the Proposed FPGA-Based Data Acquisition and Telemetry System

The architecture consists of four primary functional modules: The Sensor Generator, the Telemetry Packet Generator, the Fault Detection Unit, and the UART Transmitter. In combination, these modules create an end-to-end data acquisition and telemetry system. FPGAs provide significant advantages for this type of application due to parallel processing capabilities, minimal latency, and deterministic performance characteristics [2],[3].

A. System Architecture

As illustrated in Figure 1, the Sensor Generator Module generates three virtual sensors which represent temperature, voltage, and current readings. These virtual sensor values can be used in later processing stages to emulate real-world sensor readings. Furthermore, the generated sensor readings are continuously updated every clock cycle, which allows for testing of real-time monitoring conditions.

The Telemetry Packet Generator aggregates the sensor measurements into a single 32-bit telemetry packet. This packet-based method is extensively utilized in telemetry systems as it facilitates data management, provides ease of decoding, and improves communication reliability [4]. A base header is included in the telemetry packet as well as data related to temperature, voltage, and current.

At the same time, the sensor data is inputted to the Fault Detection Unit. The fault detection module consistently checks the incoming sensor data to see if any of the readings are above their set threshold. When any reading is outside of its range it creates a fault flag. Using fault detection on the basis of thresholds is a common method used within embedded systems as it is simple to implement, requires minimal hardware and can be accomplished in real time [5].

The telemetry packet will then be passed through the UART Transmitter module to transmit the generated telemetry packet. The UART was chosen because it is an inexpensive, simple and resource-conscious serial communication method for FPGAs [6]. The transmitter converts the parallel data from the telemetry packet into serial data for use in communicating with an external computer or monitoring device.

In conclusion, the proposed system receives sensor data from temperature sensors, organizes it into telemetry packets, enables real-time fault-monitoring and fault-detection of the data, and transmits the resulting telemetry information via a UART communication interface.

The arrangement of different components in a modular way allows for scalability, meaning that new sensors and communication protocols can be added for future upgrades of the system.

B. Methodology

The methodology used in this project adopts a modular approach to designing an FPGA. At first each module was created and tested on its own through the use of behavioral simulations. After successful testing, all of the modules were connected together to form an overall structure to create the top-level DAQ.

The process of designing was done in the following steps:

1. Development of Sensor Generator module to imitate the characteristics of a temperature, voltage and current sensor.
2. Design Telemetry Packet Generator in a way that creates packets with a structure.
3. Implementation of a fault detection unit using a set threshold.
4. Design UART Transmitter for sending telemetry data via serial communication.
5. Integration of all modules creates one DAQ using FPGA technology.
6. Functional testing using Vivado in the form of behavioral simulations.
7. Synthesize and implement on the FPGA using the AMD Vivado design suite.
8. Analysis of resource utilization and verification of implementation.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

C. *Telemetry Packet Structure*

The telemetry packets created by the proposed scheme comprise a 32-bit packet containing a sync header plus sensor measurements. The telemetry packet format is illustrated in the following section:

Header	Temperature	Voltage	Current
8 bits	8 bits	8 bits	8 bits

The sync header (0xAA) is fixed as a sync byte for packet identification. The rest of the fields are used to encode sensor measurements from the Sensor Generator module therefore this packetized data structure allows for both efficient communication between devices and simple packet decoding at the receiving end [4].

D. *Fault Detection Strategy*

The proposed fault detection mechanism continuously monitors sensor parameters while comparing them with predefined thresholds, defining a fault condition whenever:

- Temperature > Temperature Threshold
- Voltage > Voltage Threshold
- Current > Current Threshold

If one or more of these conditions exist, a fault flag is declared and will remain in an active state until normal operating conditions return to the system. Threshold-based monitoring techniques are widely used in telemetry and embedded monitoring applications because of their ease of implementation and quick response time [5]. Additionally, the generated fault flag may be interfaced with a status LED to provide immediate visual indication of abnormal operating conditions

Because of this modular system architecture with its packet-based telemetry structure and integrated fault monitoring mechanism, the proposed telemetry/DAQ system can be implemented as a very efficient real-time monitoring solution based on an FPGA DAQ/Telemetry system.

III. SYSTEM IMPLEMENTATION AND RESULTS

An FPGA-based Data Acquisition system is described in this document. The Data Acquisition System has been implemented using the Verilog HDL through the use of the AMD Vivado Development tools version 2025.2. A modular architecture was used for implementation, allowing for the independent verification of functional blocks (components) before integration into a complete system. The Data Acquisition system is comprised of four (4) Key Components, which are: Sensor Generation, Telemetry Packet Generation, Fault Detection Unit, and UART Transmission.

The data acquisition system was developed following the common development approach for FPGAs which includes the utilities of RTL (Register Transfer Level) design, behavioral simulation, synthesis, implementation and performance evaluation. FPGAs are well known as an effective method of monitoring (real-time) due to their ability to process a large number of operations in parallel, provide a deterministic response time and generate output data with low latency [2],[3].

Telemetry packets containing the sensor data from the SG were passed through three subsequent systems TPG, FDU, and UT. Before connecting the complete DAQ system the functional blocks were verified for functionality using behavioral simulation. After successful completion of verification, the modules were synthesized and implemented on the Spartan-7 (XC7S50CSGA324-1) FPGA device. The next sections contain details about the design development and results. The design, simulation, synthesis, and implementation of the DAQ system all occurred within the hardware and software setup described in Table 1.

Table 1: Development and Simulation Environment.

Parameter	Specification
Design Language	Verilog HDL
FPGA Design Tool	AMD Vivado 2025.2
Target FPGA Device	Spartan-7 XC7S50CSGA324-1
Simulator	Vivado XSim
Verification Method	Behavioral Simulation
Sensor Types	Temperature, Voltage, Current
Telemetry Header	0xAA
Packet Width	32-bit
Clock Generation	Testbench Generated Clock
Development Platform	Windows 11

A. *Sensor Generator Module*

To facilitate the test and verification of the DAQ architecture, a Sensor Generator module was developed to create sensor value patterns that simulate sensor measurements in real-time. The module creates three independent sets of sensor data (temperature, voltage, and current) within the FPGA development environment. The simulated sensor data are incrementally updated after each clock cycle to create a continuously changing operating environment.

Generated temperature, voltage, and current sensor measurements mimic the types of sensors used in commercial environments (manufacturing, building automation, etc.) and any type of telemetered sensors that one may encounter when monitoring, controlling, or measuring something in an industrial environment [1]. The verification of the complete DAQ system was accomplished via this internal generation of sensor data without requiring external hardware sensors nor other infrastructure.

The generated sensor data are shown in the following behavioral simulation in Figure 2.

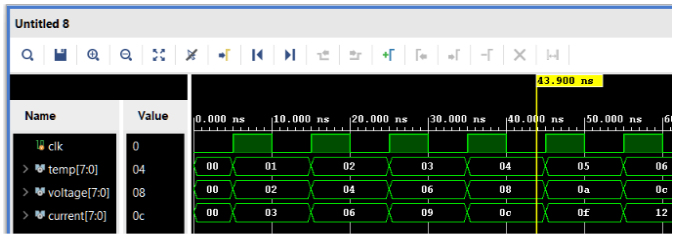

Fig. 2 Behavioral simulation waveform of the virtual sensor generator showing periodic increment of temperature, voltage, and current sensor values with each clock cycle.

The sensor data represented in Figure 2 all indicate an increase with time, showing that the generation logic for the sensors was functioning correctly throughout this process. Furthermore, the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

values generated from these sensors serve as reliable input data for both the telemetry packet generator and the fault detection modules. These results validate that the Sensor Generator module can generate dynamic sensor data for desired applications using FPGAs for DAQ.

B. Telemetry Packet Generator

The purpose of the Packet Generator within Telemetry Systems is to arrange measurements from different types of Sensors into a structure that will allow for transmission over the Network and monitoring through some sort of software package. The use of Packets for the transmission of data in Telemetry Systems provides a way to improve Synchronous Transmission of data, ease of Decoding, and improve the Reliability of Communication by establishing a defined structure for the Data being transmitted [4].

For the Telemetry Packet Generation, this proposed approach uses values of Voltage, Current, and Temperature from the Sensor Generator and adds those values into a Telemetry Packet that includes a Predefined Synchronization Header. Each Telemetry Packet contains Four Bytes of Data; therefore, the total width of the Packet is 32 Bits. The Telemetry Packet Structure is given below:

Header	Temperature	Voltage	Current
8 bits	8 bits	8 bits	8 bits

The Synchronisation Header is always a Fixed Value of 0xAA (Hexadecimal) to allow for the identification and Decoding of the Packet at the Receiving end. The remaining Three Bytes of Data represent the Measurement Values for the Temperature, Voltage, and Current created by the Sensor Generator.

The Behavioral Simulation Waveform of the Telemetry Packet Generator is shown in Figure 3.

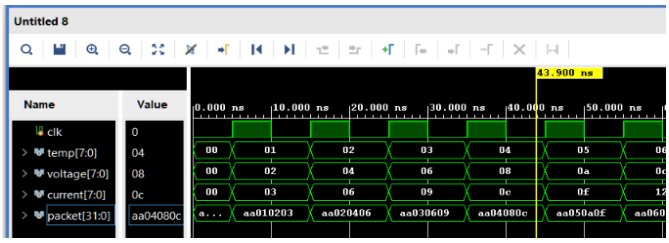

Fig. 3 Behavioral simulation waveform of telemetry packet generation showing successful concatenation of packet header (AAh), temperature, voltage, and current sensor data into a 32-bit telemetry frame

Upon analysing the packet header with the generated sensor values shown in Figure 3, a complete 32-bit telemetry frame has been successfully formed. Both the waveform confirms a properly assembled packet and shows that the telemetry packet generator continuously updates telemetry data whenever there is a change to the sensor values. Telemetry packets were generated during the simulation and are presented in Table 2.

Table 2: Telemetry Packet Verification.

Header	Temp	Voltage	Current	Packet
AA	01	02	03	AA010203
AA	02	04	06	AA020406
AA	03	06	09	AA030609
AA	04	08	0C	AA04080C
AA	05	0A	0F	AA050A0F

The results obtained show successful packet generation from multiple sensor combinations. For example, if the temperature is 01h, the voltage is 02h and the current is 03h, the generated telemetry packet will look like AA010203. The other combinations of sensors also produce telemetry frames that are properly formatted and maintain the synchronisation header. The results verify that the Telemetry Packet Generator was able to properly structure the various sensor data into a format for communicating via serial transmission/telemetry applications. The packetization method used in this implementation provided a means of efficiently and scalabley allowing future expansion with additional sensor channels and/or new communication protocols.

C. Fault Detection Unit

The Fault Detection Unit is a device designed to continuously check various sensor parameters and identify any unusual operating conditions in the proposed DAQ architecture. In telemetry and embedded systems, monitoring for faults is a significant need since it allows for the identification of unsafe operating conditions at an earlier time. This ultimately improves the overall system reliability [5].

The fault detection method that has been implemented uses three preset limit values for temperature (200), voltage (180), and current (150) as fault detection thresholds. Therefore, as sensors operate, their output will be continuously compared to these three limit values. Anytime a sensor exceeds its designated threshold value, a "fault" flag will be set to indicate that the sensor has operated outside of its normal range.

The fault detection logic represents this process in the following way:

Fault Flag = (Temperature > 200) OR (Voltage > 180) OR (Current > 150).

This method of using set thresholds provides an easy and low-cost hardware means of detecting out-of-range values during ongoing real-time monitors [5].

The behavior simulation waveform for the Fault Detection Unit is detailed in Figure 4.

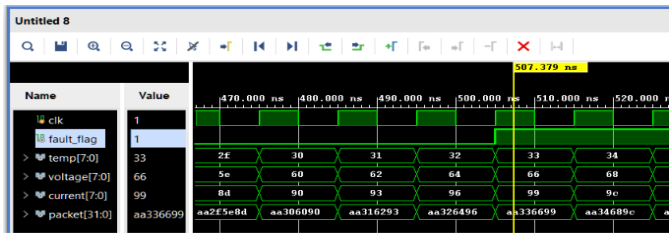

Fig. 4 Fault detection waveform showing assertion of the fault flag when sensor values exceed predefined threshold limits

Figure 4 depicts the behavior of the fault flag under normal operation. At all times the fault flag remains at logic level 0 unless the threshold value is exceeded, at which point the fault flag transitions to logic level 1. The example in figure 4 shows the current sensor value exceeds its defined threshold of 150 determining if an anomaly has been detected as the sensor value at that instant was 153 (0x99 in hexadecimal). The fault flag also shows detection of an anomaly has been achieved. The

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

waveform from the figure shows that the correct threshold-based fault detection logic was implemented.

Table 3: Fault Detection Test Cases

Test Case	Temperature	Voltage	Current	Fault Flag	Observation
TC1	50	80	100	0	Normal Operation
TC2	120	140	130	0	Normal Operation
TC3	51	102	153	1	Current Threshold Exceeded
TC4	201	120	100	1	Temperature Threshold Exceeded
TC5	100	181	120	1	Voltage Threshold Exceeded
TC6	220	190	170	1	Multiple Threshold Violations

The results in Table 3 illustrate the operation of the fault detection design (i.e., how it behaves) when subjected to different set of environmental or use conditions. In cases TC1 and TC2 (i.e., normal operation), all sensed values (i.e., the measurements taken from the physical world by the sensors) are within requirements and the fault flag value (i.e., whether or not there is (in)correct sensor performance) is 0. When the measurement parameter being sensed exceeds its limit (the parameter being monitored has crossed its prescribed threshold), the fault flag becomes immediately set to 1, which indicates that there is an issue with the sensor performing its function as intended (abnormal operating condition detected).

An example of this type of sensor failure is seen in TC3. The current sensor measures 153, which is greater than the threshold of 150, and activates the fault flag by setting it to 1, which indicates an abnormal operating condition. A similar situation occurs with TC4 (over temperature) and TC5 (over voltage). In TC6, all parameter measurements are simultaneously over the prescribed threshold requirements (i.e., they are being measured above their threshold values). The fault detection system is able to accurately determine the abnormal operating conditions in each instance.

The results of these simulations demonstrate that the architecture that was proposed can accurately identify abnormal sensors while maintaining low complexity for the implementation of the detector function. The ability to make this type of measurement is of great importance and will significantly improve the reliability of the proposed telemetry solution.

D. UART Transmitter Module

The UART Transmitter Module has been created to enable Serial Communication for the proposed DAQ architecture built on an FPGA. The UART protocol (Universal Asynchronous Receiver Transmitter) is among the most popular Serial Communication protocols used in embedded systems, due to the low complexity and cost associated with the hardware; as well as having reliable data transfer capabilities [6].

The main purpose of the UART transmitter is to take in Parallel Telemetry Data and convert it to a serial bit stream, which can be sent through a Communication Channel. The architecture of the UART is implemented using a finite state machine (FSM) that generates the Start Bit, transmits Data Bits, and appends a Stop Bit, which are all required for Asynchronous Serial Communication.

The operation of the Transmitter follows this Sequence:

1. Idle State
2. Start Bit Generation
3. Data Bit Transmission
4. Stop Bit Generation
5. Return to Idle State

When the UART receives a transmission request, it captures the input data and begins Serial Transmission of that data. While the data is being transmitted, the Busy Signal is held high, indicating that data is currently being sent out of the Transmitter. Once a full frame of data has been sent out, the Busy Signal goes low, allowing the Transmitter to accept the next transmission request. The behavioral simulation waveform for the UART Transmitter Module can be viewed in Figure 5.

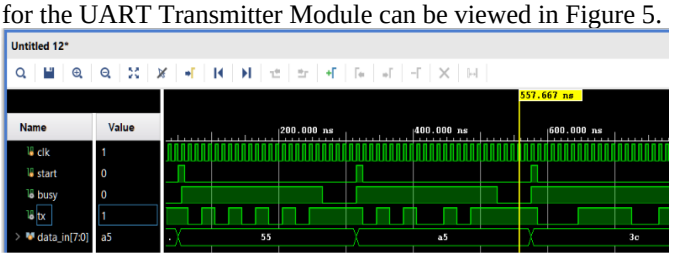

Fig. 5 UART transmitter verification waveform showing serial transmission of test bytes (0x55, 0xA5, and 0x3C) with corresponding start, busy, and TX signals

The UART transmitter has been proven to work correctly when creating the expected serial data transfer sequence by successfully transmitting a start bit, 8 data bits, and a stop bit as demonstrated in figure 5. The transmitted waveform also indicates that the state machine for serializing has performed correctly and that the input (parallel) data has been serialized as intended.

To validate the operation of the transmitter under various conditions (input source) several test bytes were transmitted using the UART module as listed in table 4.

Table 4: UART Verification Test Cases.

Test Case	Input Byte	Expected Result	Observed Result
TC1	0x55	Serial transmission on TX	Pass
TC2	0xA5	Serial transmission on TX	Pass
TC3	0x3C	Serial transmission on TX	Pass

All test patterns were transmitted to completion with the same values being sent with the results for TC1, TC2, and TC3

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

displaying the appropriate values demonstrating that the serial system is functioning correctly; the busy status signal was high throughout the time frame of the transmission and went low after the UART frame had completed transmitting.

Based upon these results it can be concluded that the UART Transmitter functions correctly and is suitable for use in telemetry applications, as well as, due to the modular design being capable to be integrated into other systems in the future (external monitoring systems, serial interfaces, etc.)

E. Top-Level System Integration

All of the individual component modules (Sensor Generator, Telemetry Packet Generator, Fault Detection Unit and UART Transmitter) have been combined into a single FPGA-based Data Acquisition and Telemetry System after they have passed individual tests. The purpose of the top-level integration was to verify that all functional blocks operate in concert with one another and to determine the overall system performance under realistic operating conditions.

The integrated architecture executes a full data acquisition process once sensor data is generated, telemetry packets are created, faults have been detected, and serial data has been transmitted. The Sensor Generator generates continuous temperature, voltage, and current readings which are processed at the same time as they generate telemetry packets and detect faults. Once generated, the telemetry packets are transmitted via the UART communication interface. In parallel, the Fault Detection Unit continuously monitors the parameters of the sensors and sets a fault flag when any sensors show signs of malfunctioning.

The top-level integration verifies that all components of the system interact correctly and that information flows are correctly established through each stage of the proposed system architecture. The successful integration is a major accomplishment because it demonstrates that independently verified component modules function together to form an integrated FPGA-based DAQ system.

In Figure 6, you can see the simulated behavior of the complete integrated architecture.

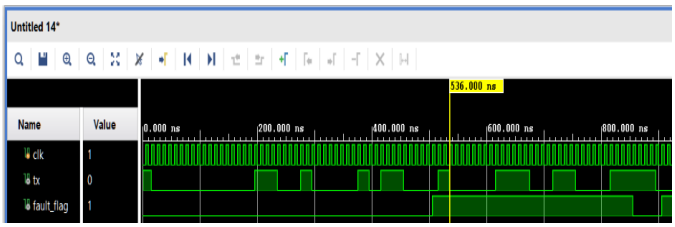

Fig. 6 Top-level DAQ system integration waveform showing UART transmission activity and fault flag assertion during system operation

Telemetry Packet Generator processed generated sensor readings successfully and created formatted telemetry packets (or frames). The Fault Detection also continuously monitored the sensor state (or condition) and provided fault flagging (or triggering) whenever any threshold violations occurred. The generated telemetry output was transmitted via the UART interface, verifying overall end-to-end system operation.

To validate the operation of the system, various integration testing scenarios were completed, and the verification results for each integration test scenario are summarized in Table 5.

Table 5: Verification Summary for Integration Testing.

Module	Function	Status
Sensor Generator	Generates sensor data	Pass
Packet Generator	Creates telemetry packet	Pass
Fault Detector	Detects threshold violations	Pass
UART Transmitter	Serial data transmission	Pass
Top-Level Integration	End-to-end system operation	Pass

The successful testing of the entire functional modules demonstrated that sensor values were accurately created and packetized, fault conditions detected, and telemetered data transmitted via UART. During simulation there was no data corruption or synchronization issues present.

The results achieved fully demonstrate that the combined architecture developed in this study meets the specifications needed for a fully functional FPGA-Based DAQ and telemetering system. The successful completion of a complete verification of the functional modules validates the effectiveness of this design approach (modular design methodology) for building integrated systems. Additionally, the results confirm the ability of the architecture to support real-time monitoring and telemetry applications.

F. Synthesizing FPGA Resources

Once the functional verification was successfully completed, the entire FPGA-based DAQ and telemetering system was then synthesized using the AMD Vivado Design Suite targeting the Spartan-7 (XC7S50CSGA324-1) FPGA device. The synthesis phase translates the Verilog HDL description into an FPGA-based hardware implementation which consists of logical, register and interconnect resources (i.e., logic cells, flip-flops and routing resources) available on the target FPGA device.

The synthesis result was achieved without any critical design errors, confirming that the proposed configurational design is a suitable candidate for implementation as hardware. The synthesized FPGA device using Vivado (after synthesis and configuration) is depicted in Figure 7.

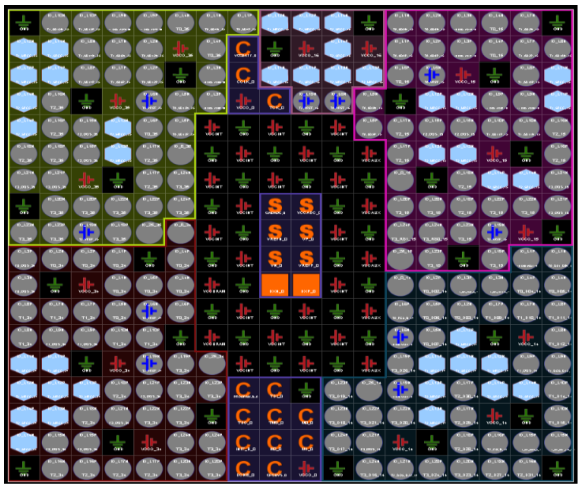

Fig. 7 Synthesized FPGA device view showing the target SPARTAN -7 (xc7s50csga324-1) package and available FPGA resources used for implementation of the DAQ system

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

The generalized synthesized design was implemented in the Spartan-7 FPGA architecture (Figure 7). The synthesis results confirmed that the proposed DAQ architecture was successfully translated to FPGA hardware resources without losing the intended functionality of the system.

To evaluate hardware efficiency, resource utilization statistics for the FPGA resources were collected from the Vivado synthesis report. The relevant results are shown in Table 6.

Table 6: FPGA Resource Utilization.

Resource	Used	Available	Utilization (%)
Slice LUTs	34	32,600	0.10%
Slice Registers	40	65,200	0.06%
Bonded IOBs	3	210	1.43%
BUFGCTRL	1	32	3.13%

The resource utilization report shows that the DAQ design uses a small part of the available FPGA resources. It uses 40 slice registers and 34 slice LUTs, which is only 0.06% and 0.10% of the available FPGA resources, respectively. The DAQ design also requires only 1 BUFGCTRL resource and 3 bonded IOBs to implement the entire DAQ system.

The low resource utilization indicates that the proposed design is efficient for the hardware on which it is implemented. Despite having integrated sensor generation, telemetry packet form, fault detection, and UART communication into a single design, the resources used by the design are minimal. This will be beneficial for applications that monitor embedded systems and must use the least amount of FPGA resources for implementation.

The low resource utilization of the proposed design will allow for significant future growth within the design. Additional channels and new sensors can be added without requiring many additional FPGA resources. Similarly, other communication methods or algorithms to process the data or to diagnose faults within the system will not require a significant hardware consumption in order to be added to the design.

The synthesis results demonstrate that the FPGA-based DAQ system is capable of achieving the functionality proposed while being both efficient and scalable.

G. FPGA Implementation Results

The AMD Vivado Design Suite completed the synthesis and implementation of the FPGA-based Data Acquisition and Telemetry System onto the Spartan-7 (XC7S50CSGA324-1) FPGA device. The implementation process placed and routed synthesized elements using physical resources on the FPGA. The physical resources for FPGA include all of the devices or connections that interconnect and connect the devices together using signal (i.e., electrical lines) and timing (i.e., clock signals).

The synthesized modules have all successfully been placed and routed with no functional conflicts, thus confirming successful synthesis implementation, verifying that the proposed telemetry system architecture can be surgically implemented onto an FPGA architecture.

The implemented FPGA device was captured as seen in Figure 8.

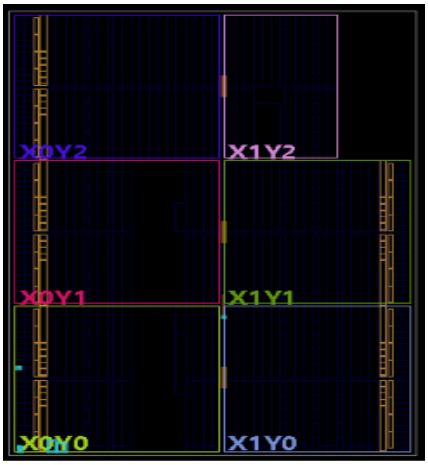

Fig. 8 Implemented FPGA design layout on the Xilinx Spartan-7 (xc7s50csga324-1) device showing the placement of the DAQ system resources

Figure 8 shows that the designed implementation fits into a very small area of the FPGAs overall available fabric; from the placement results obtained, a large portion of the total logic resources available have been efficiently utilized, and that the radial distances for proper routing connectivity have also been satisfied throughout the entire design. In relation to the implementation results, the compact resource usage evident from the layout of the implemented device matches the utilization report obtained during synthesis (Table 6).

The implementation results also show that although several different types of functional modular designs (e.g., sensor generation, telemetry packet creation, fault monitoring, and UART communication) were included in the architecture, the overall resource footprint needed to implement these modules within the FPGA is very small; therefore, this type of implementation provides significant flexibility for future modifications and enhancements to the design, as well as for adding other types of hardware/hardware capabilities.

In addition, successful routing and placement of the circuit have confirmed that the proposed architecture is hardware compatible with the Spartan-7 family of FPGA devices. The results obtained from the implementation validate that a complete DAQ system can be implemented in an economically viable way on a low-cost FPGA device while still having enough resources available for growth in future years.

In summary, the implementation results demonstrate the practicality of the proposed FPGA-based Data Acquisition and Telemetry System and confirm that the architecture can scale appropriately and efficiently from a hardware perspective.

IV. CONCLUSION

This paper discusses an FPGA data acquisition/telemetry system created with Verilog HDL. The overall system design combines sensor data generation, telemetry packets, fault detection, and communication via UART on one FPGA platform. The system's modular design methodology was intended to allow for simple system development, verification, and future scalability.

The behavior simulation results confirmed the modules are functioning properly when functioning individually and as an integrated system with the complete DAQ architecture. The waveforms generated during simulation show that sensor data generation, packet creation, fault monitoring, and UART output

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

were all successful. The implementation of both the synthesis and implementation on an FPGA using the AMD Vivado design suite was successful on the Spartan-7 (XC7S50CSGA324-1) FPGA.

Resource usage analysis showed that the architecture consumed a very small amount of the FPGAs resources, utilizing approximately 0.10% of slice LUTs and 0.06% of slice registers. The low consumption of resources by this hardware system indicates that it is efficient and has plenty of room for future expansion and additional functionality.

In general, the data collected supports that the new Data Acquisition & Telemetry System (DATS) built on an FPGA provides an effective, flexible and efficient data collection method for time-sensitive telemetry and monitoring applications. The creation and testing of working sensor monitor applications and telemetry technology was performed with a low-cost, high-volume FPGA platform creating a sound foundation for future development of such systems using FPGAs.

REFERENCES

[1] J. Fraden, Handbook of Modern Sensors: Physics, Designs, and Applications, 5th Edition, Springer, 2016

[2] S. Brown and Z. Vranesic, Fundamentals of Digital Logic with Verilog Design, 3rd Edition, McGraw-Hill, 2014.

[3] P. P. Chu, FPGA Prototyping by Verilog Examples, Wiley, 2008.

[4] D. Patranabis, Telemetry Principles, Tata McGraw-Hill, 2003.

[5] R. Isermann, Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance, Springer, 2006.

[6] J. W. Valvano, Embedded Systems: Real-Time Interfacing to ARM Cortex-M Microcontrollers, CreateSpace, 2014.

[7] S. Choi et al., “FPGA-Based Multi-Channel Real-Time Data Acquisition System,” Electronics, vol. 13, no. 15, 2024.

[8] I. D. Bonequi et al., “A Field-Programmable Gate Array (FPGA)-Based Data Acquisition System for Multi-Electrode Cellular Electrophysiology,” HardwareX, 2022