A Highly Reliable GPS Synchronized Clock with Sub-Second Precision Time Output

Submitted

by

Aditya Sharma 19115011
Akanksha Verma 19115015
Anchit Proch 19115024
Rahul Gupta 19115095
Raman Yadav 19121024
Vatsal Dixit 19115128
Mamta Meena 14115069

Project Supervisor

Prof. Vishal Kumar Saxena

Department of Electrical Engineering
Indian Institute of Technology Roorkee
2022

Declaration

We hereby declare that the work, which is presented here, entitled **A Highly Reliable GPS Synchronized Clock with Sub-Second Precision Time Output**, submitted for the completion of course EEN 300: Industry Oriented Problem (IOP). We also declare that we have been doing our work under the supervision and guidance of **Prof. Vishal Kumar Saxena, Electrical Engineering Department, Indian Institute of Technology Roorkee**. The matter presented in this report is not submitted for award of any other degree of any other institutes.

Date: 19th April 2022

Signature(s): Aditya sharma
Akanksha
Anchit proch
Rahulgupta
Ramanyadav
Vatsal dixit
Mamta meena

Certificate

This is to certify that the above statement made by the candidate is true to best of my knowledge and belief.

Prof. Vishal Kumar Saxena
Professor
Department of Electrical Engineering
Indian Institute of Technology Roorkee

Acknowledgment

This Industry Oriented Problem (IOP) Project could not have been carried out without the constant and enormous support and help we received from our project supervisor Prof. Vishal Kumar Saxena. We would also like to thank Mr. Ashish Kumar Sharma (Ph.D. at EED) wholeheartedly for his guidance and the many hours he had spent through virtual and in-person discussion sessions to rectify our errors and suggest new directions to increase the project's usability. We would like to thank Tinkering Labs, Team Robocon IIT Roorkee, and IIT Roorkee Motorsports for giving us access to soldering stations and testing equipment. Finally, we would like to thank Prof. G.N. Pillai (HoD, EED) and Prof. Vishal Kumar Saxena for providing 24x7 access to Embedded Systems Laboratory, EED.

Abstract

Accurate and precise timing is highly desirable in multiple economic activities around the world. The Global Positioning System (GPS) satellites translate precise time from atomic clocks contributing highly precise (within 100 billionths of a second) clocks to transmitting signals. This enables the receivers on Earth to produce high-quality synchronization signals without owning and calibrating highly accurate atomic clocks. Although the GPS receiver module has ultra-low noise 1Hz signals (1PPS), their reliability and low resolution remain a concern for various applications related to radio astronomy, communication networks, and power transmission systems. A highly reliable, GPS time-synchronized time-stamping methodology with sub-second precision time output is proposed to mitigate this problem. This concept is validated on the Zync7 Evaluation and Development Board (Zed Board), where the onboard Phase Locked Loop (PLL)-based clocks are utilized to synchronize and interpolate the GPS 1PPS signal.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction and Motivation

1.1 Introduction

Modern-day microprocessor-based systems require precise clocks for their timing synchronization and operational efficiency [1]. The accuracy, universality, and low-cost nature of the Global Positioning System (GPS) clock has led to significant improvements in receiver technologies for many medium and small-scale organizations adopting its use for local time-stamping. The GPS receiver module records time in UTC (Universal Co-ordinate Time), which depends on the vibration of individual atoms in atomic clocks. GPS-based time synchronization is one of the most widely utilized and accepted technologies for precise timing generation, enabling a clock-generating device to correct any local time for its users [2].

1.2 Motivation

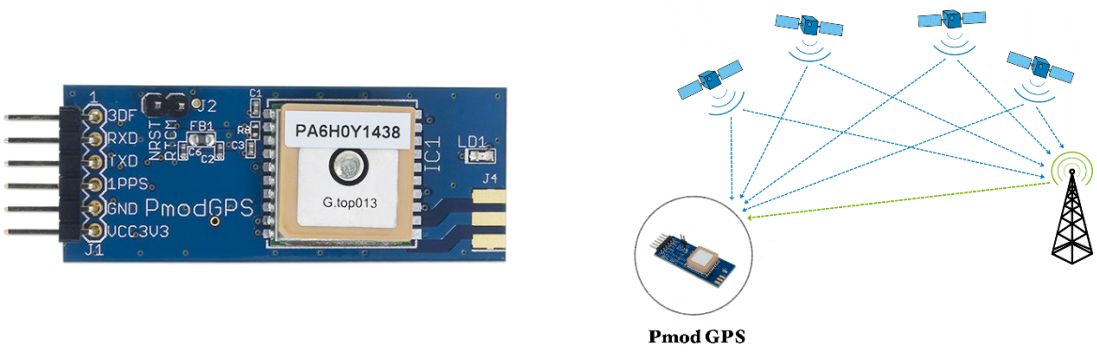
The GPS clock is also employed to synchronize electric power measurement and control protective device systems for power transmission applications. An FPGA-generated clock is compared with the GPS reference time, and logic determination is carried out, achieving fault detection and positioning. Also, the micro-second clock precision required for wireline communication systems is another potential application for GPS-based clocks. Although the GPS receiver produces precise 1Hz signals (1PPS), their reliability and low resolution remain a concern for various applications related to radio astronomy, communication networks, and power transmission systems. This work aims to design and implement a GPS-based sub-second precision clocking solution to mitigate the described challenges while improving its reliability substantially.

Chapter 2

Methodology

2.1 GPS Receiver

The GPS receiver module (PmodGPS) is an integral part of the GPS-based time detection system. It is a device that provides a 1 Pulse Per Second (1PPS) signal to the processing board along with the integrated location latitude, longitude, and altitude through the UART communication protocol [3]. The PmodGPS consists of the MediaTek GPS MT3329 on a GlobalTop FGPMMOPA6H GPS antenna module to receive positional data over UART. By default, the PmodGPS operates using the National Marine Electronics Association (NMEA) protocols for data output from the GPS module, although the Radio Technical Commission for Maritime Services (RTCM) pin Differential Global Positioning System (DGPS) data using RTCM protocols is also available. It is also designed for synchronization and timing applications, allowing the receiver to generate accurate timing with only one satellite being tracked. This property also allows for satellite tracking and GPS synchronization in weak-signal areas.



(a) (b)

Fig. 2.1. a) The GPS receiver module (PmodGPS) from Digilent Inc. and b) the GPS signal transmission process with the minimum required four co-planar GPS satellites in a 'visible' slot

TABLE 2.1
Pin Description

Pin	Signal	Description
1	3DF	3D-Fix Indicator
2	RX	Receiver
3	TX	Transmitter
4	1PPS	1 Pulse per Second
5	GND	Power Supply
6	VCC	Power Supply (3.3/5V)

2.2 Component Selection

2.2.1 Why Zedboard?

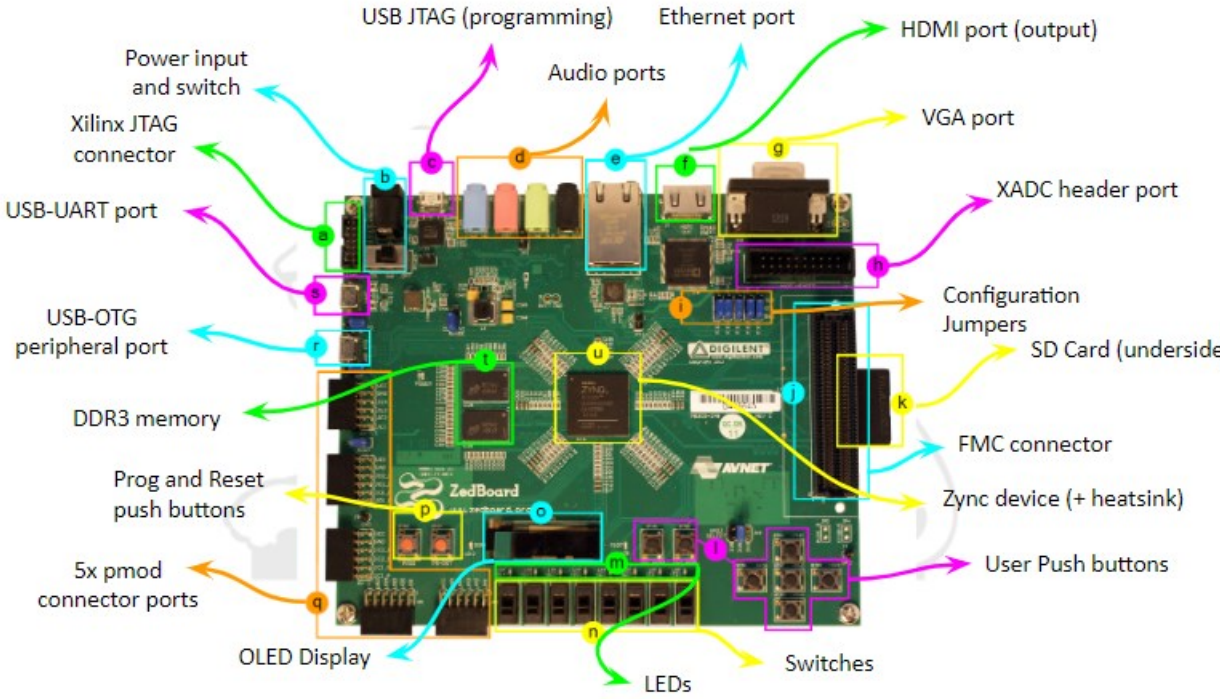


Fig. 2.2 The Zynq Evaluation and Development (Zedboard)

An important design decision for the project was the choice of the primary processor, especially the use of a Field-Programmable Gate Array (FPGA) against a low-cost microcontroller. Here, we cite some reasons for preferring an FPGA (Zedboard) over a microcontroller:

1. **Parallel Processing**: Generally, a microcontroller executes all operations sequentially, while an FPGA (Zedboard) is a field-programmable gate array that performs all operations through parallel computations. For our application, the data that requires high-precision and reliable time-stamping must be processed alongside the generated time with simultaneous operation. All processes must be completed within its frequency cycle (16MHz) for a microcontroller, defeating the project's purpose.
2. **Computation Power**: An FPGA's floating-point operations are easy to handle, supporting our requirement of time generation in microsecond accuracy through the continuous use of division and addition operation.
3. **Communication Protocols**: A critical application of this work is in power grid communication protocols (Local Area Networks), which require the availability of LAN and Ethernet ports.

2.3 Architecture

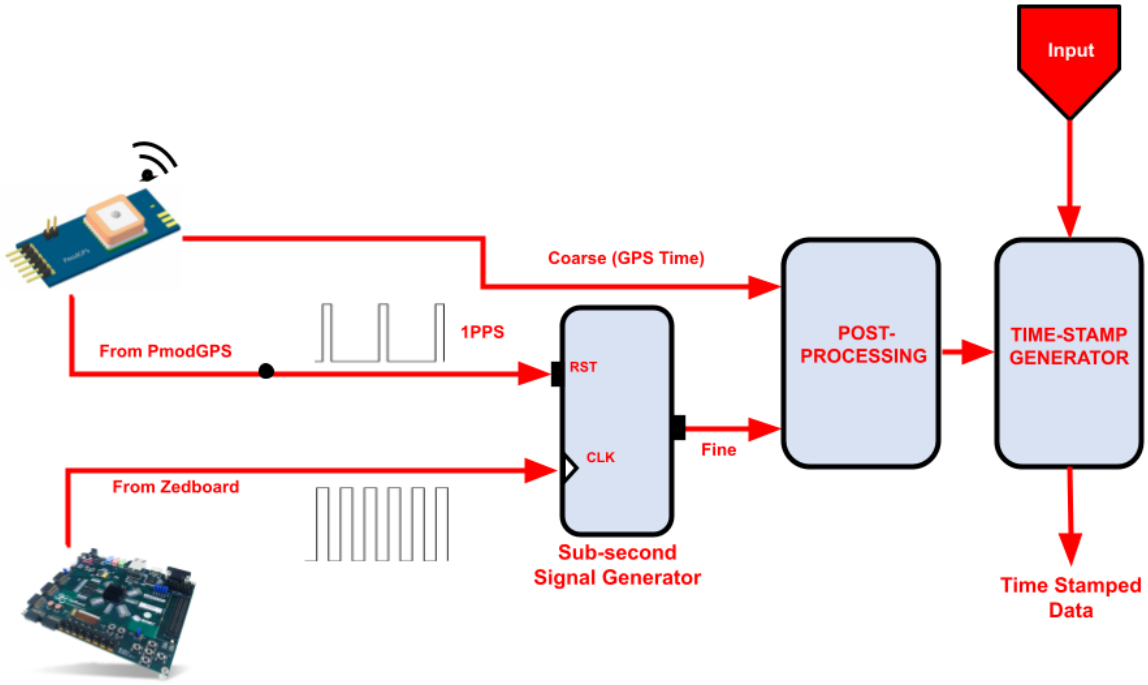


Fig. 2.3. System Architecture

This section describes the procedure for processing and utilizing the GPS signal received from the PmodGPS using the Zedboard.

2.3.1 Clock Interpolation

In order to produce finer clock resolution, we implement an RTL-based Sub-second Timing Signal Generator. This block receives a 1MHz CLOCK from the Zedboard's PLL and a RESET signal from the PmodGPS's 1Hz 1PPS signal and operates as follows:

1) The signal generator increments its output count at every rising edge of the input clock, i.e., after every 1 microsecond, until the 1PPS GPS signal resets its value to 0 at its rising edge.
2) Ideally, the output value would increase to 10^6(i.e., the number of rising edges in a second) before being reset to 0 by the rising 1PPS signal after 1 second. However, the unreliable nature of the 1PPS rising edge input allows the count to increment even after a second.
3) This problem requires an increase in count output bus length from 20 (for the ideal case) to 28, considering a maximum delay of 2 minutes before the next 1PPS rising edge. The 'fine' output from this block is sent to the next step, i.e., the processing block (see Fig. 2.2).

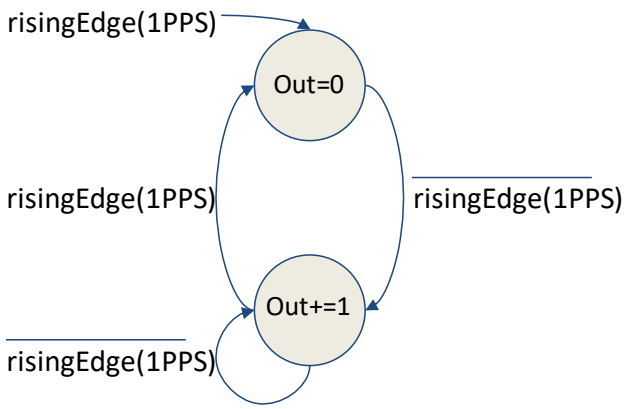


Fig. 2.4. The state-machine diagram for the signal-generation block

2.3.2 Synchronization

The 'fine' clock output, the 1PPS signal, and the GPS-based UTC-time from the GPS Module are packaged and sent to the processing unit (see Fig. 2.2). If the 1PPS rising edge is not seen within 120 cycles after the previous 1PPS, the output is generated from the fine clock, indicating that the receiver has missed the 1PPS signal (see Fig. 2.4). Subsequently, the output time is either updated to the newest available GPS time or an extrapolated version of the last received time through the sub-second clock output (see Fig. 2.5). This computation substantially improves the reliability of the GPS UTC by extending its availability by using a Zedboard-generated 'finer' 1MHz clock source.

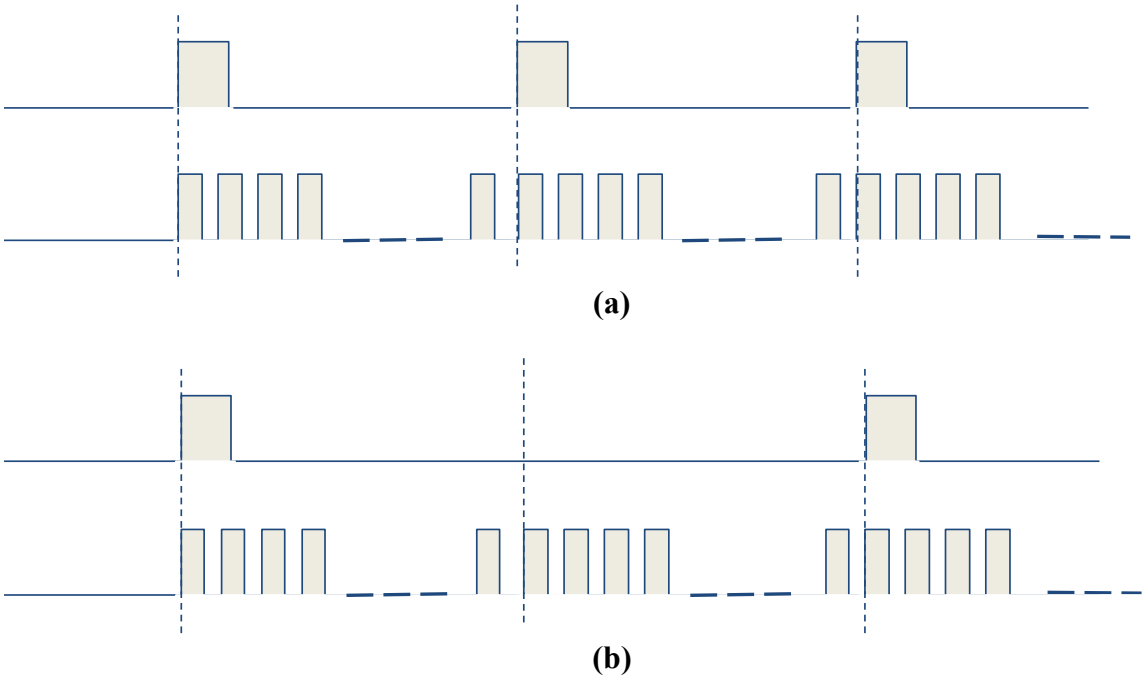


Fig. 2.5. The timing diagram for the 1PPS (top) and clock output (bottom) when a) the 1PPS arrives after one cycle and b) when the 1PPS signal does not arrive after one cycle

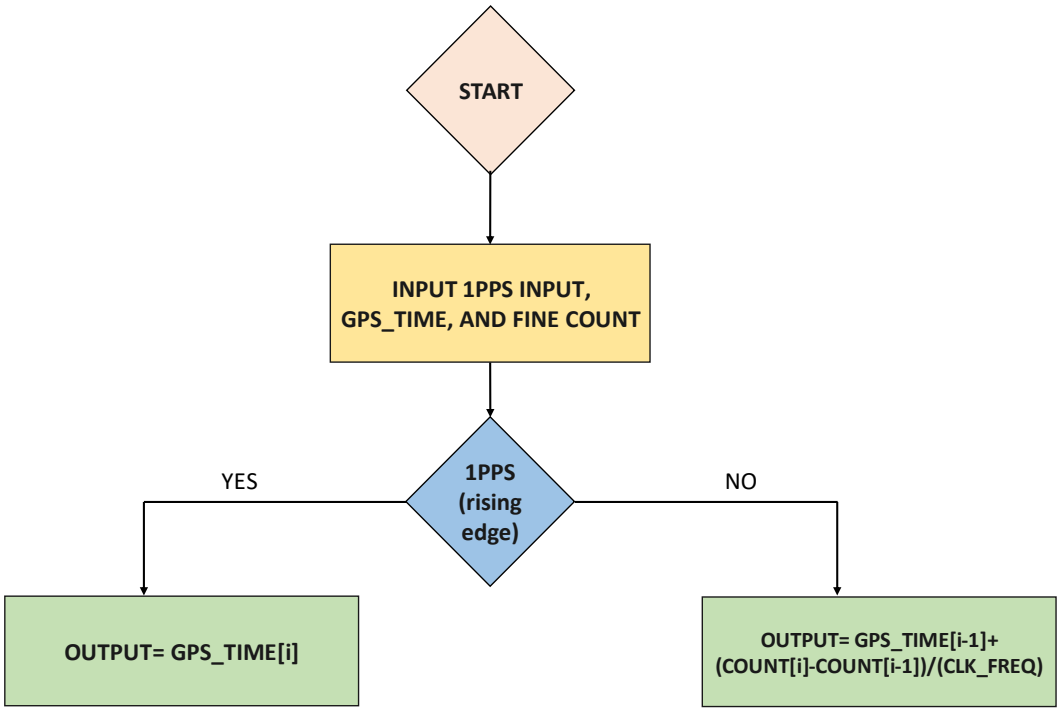


Fig. 2.6. A flowchart depicting the time processing algorithm

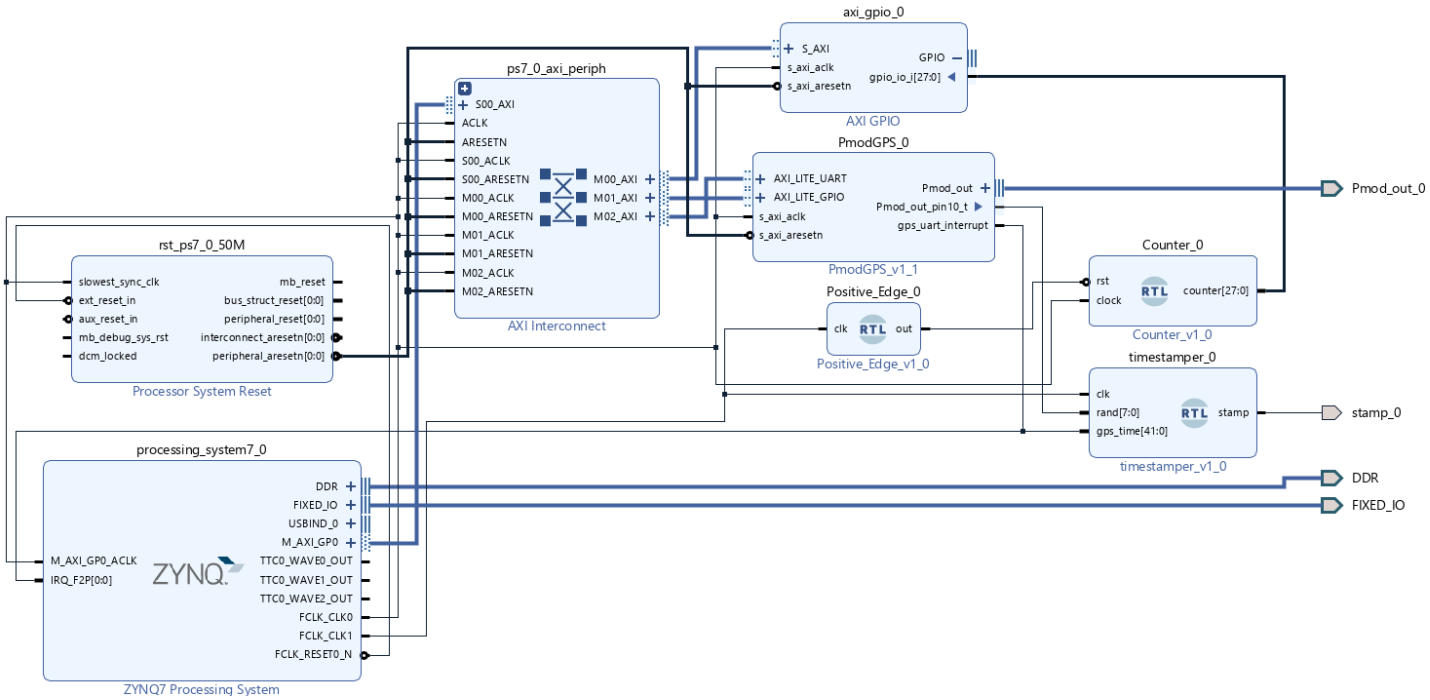


Fig. 2.7. Overall connection diagram for the Vivado project

2.4 Case Study

The reliable, high-resolution GPS clock could be used in multiple industrial-grade applications ranging from power systems to radio astronomy. However, the applications in Table 2.2 are deemed the most suitable considering the microsecond resolution generated from this clock. Here, we primarily target the communication protocols (LANs) in power grid substations which require microsecond level precision for synchronization and operation.

TABLE 2.2
Applications of time-dependent data

Grid application	Timing requirements (minimum reporting resolution and accuracy relative to UTC)
SCADA	Every 4-6 seconds reporting rate
Sequence of events recorder	50 µs to 2 ms
Digital fault recorder	50 µs to 1 ms
Protective relays	1 ms or better
Communications protocols	
Substation local area network communication protocols (IEC 61850 GOOSE)	100 µs to 1 ms synchronization
Substation LANs (IEC 61850 Sample Values)	1 µs

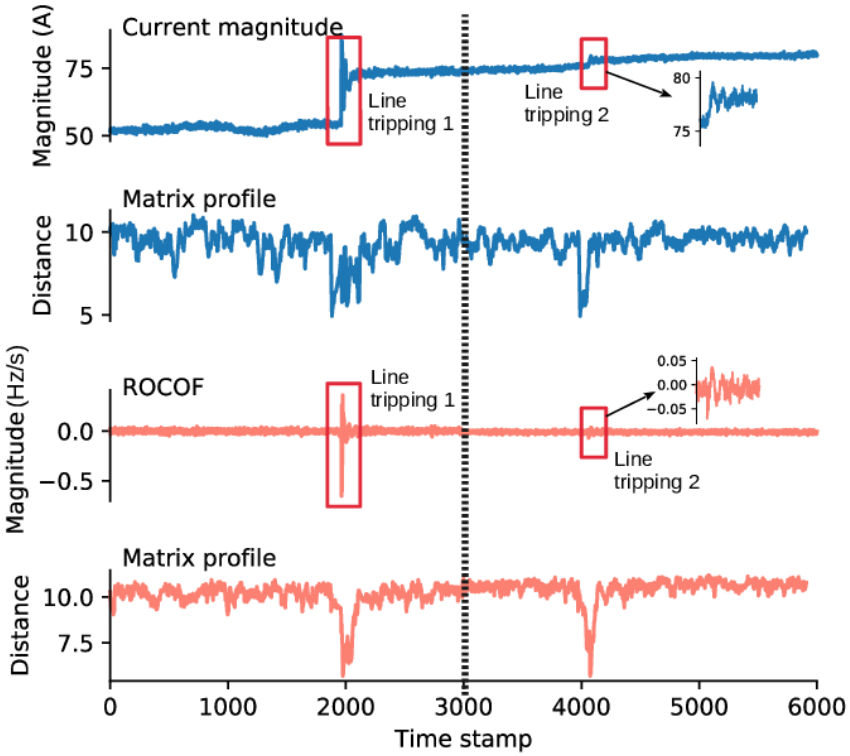


Fig. 2.8. Typical time-stamping process in a power substation

Chapter 3

Results

3.1 Signal Generator

Verilog Code:

```
`timescale 1ns / 1ps

module Counter_1 (reset,clk,q);
  input clk;
  input reset;
  output reg[27:0]q=28'd0;


  always@(posedge clk , posedge reset)
    begin

     if(reset)
       q<=0;
     else
       q<=q+1;
   end

endmodule
```

Simulated Waveforms:

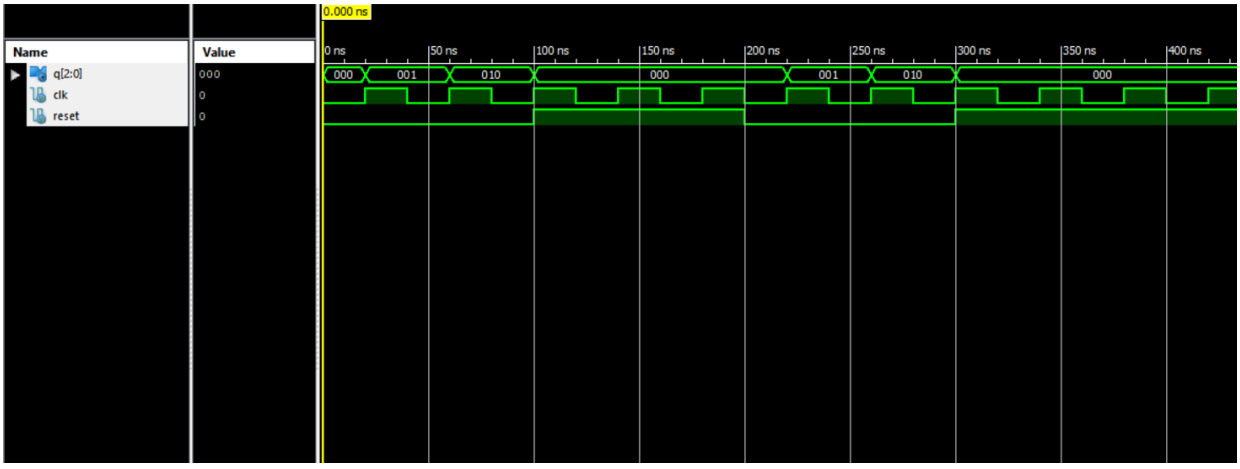


Fig. 2.9. Simulated waveforms: Timing Signal Generator

3.2 Processing Unit

C++ Code:

```
int main()
{
    init_platform();
    XGpio input;
    DemoInitialize();


    XGpio_Initialize(&input, XPAR_AXI_GPIO_0_DEVICE_ID);
    // Data direction register : input-1 , output - 0

    XGpio_SetDataDirection(&input, 1, 0xffffffff);
    long long int a;

    while(1)
    {

        a=XGpio_DiscreteRead(&input, 1);
        xil_printf("Sampled Data: ");
        xil_printf("0x%x", rand()%65536);
        DemoRun();
        xil_printf("(Sub-Second):%d\r\n", a);

    }
    cleanup_platform();
    return 0;
}
void DemoRun() {

      if (GPS.ping) {
         GPS_formatSentence(&GPS);
         if (GPS_isFixed(&GPS)) {
            string a = GPS_getTime(&GPS);
            int Hour = (int (a[0:1]) + 5)%24;
            int Minutes = (int (a[2:3]) + 30)%60;
            int Seconds = int (a[4:5]);

                printf("Time Stamp: %d", Hour);
            printf(":%d", Minutes);
            printf(":%d", Second);
            //xil_printf("Latitude: %s\n\r", GPS_getLatitude(&GPS));
            //xil_printf("Longitude: %s\n\r", GPS_getLongitude(&GPS));
            //xil_printf("Altitude: %s\n\r", GPS_getAltitudeString(&GPS));
            //xil_printf("Number of Satellites: %d\n\n\r", GPS_getNumSats(&GPS));
         }

      }

   DisableCaches();
}
```

Results:

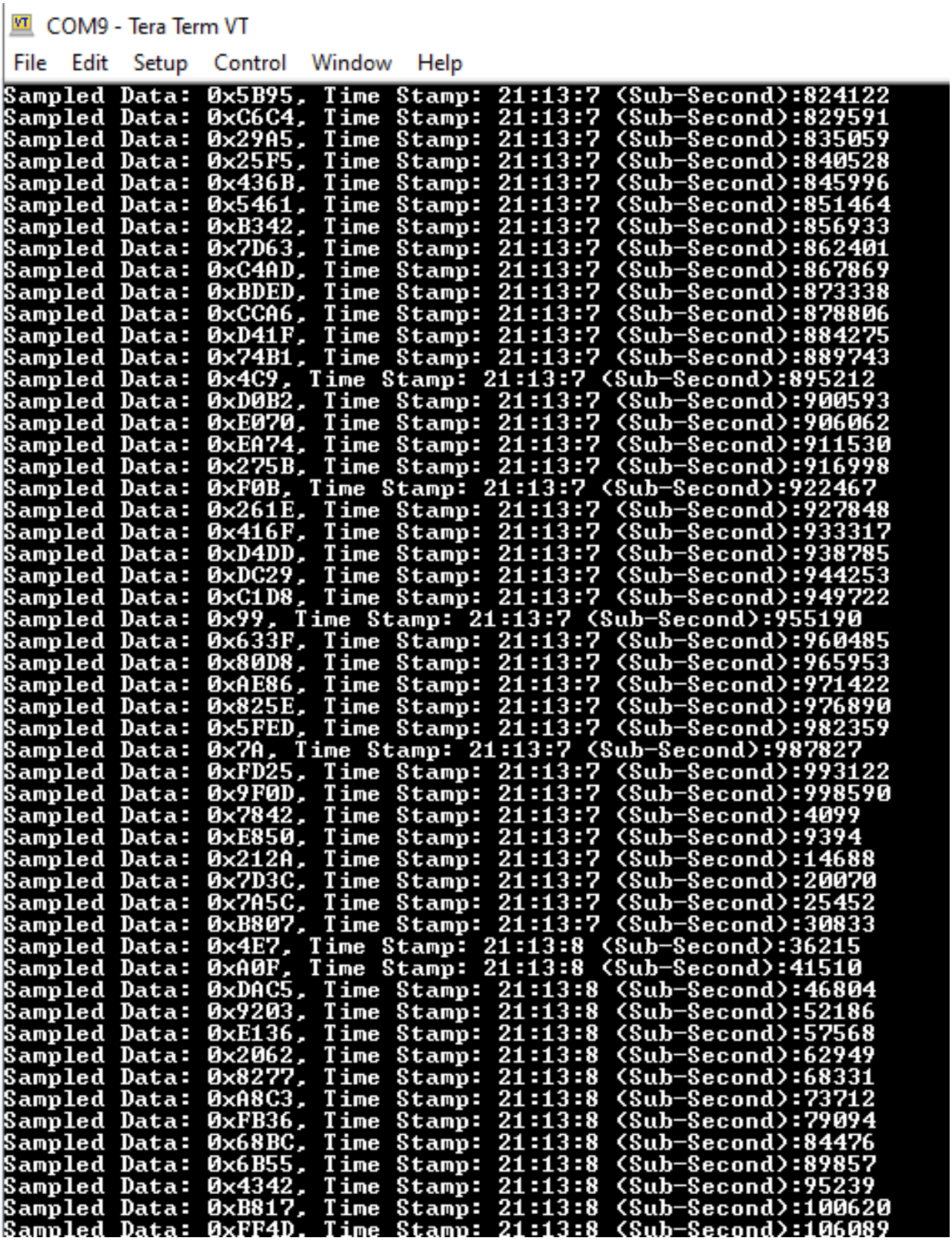


Fig. 2.10. Final Output on Serial Monitor

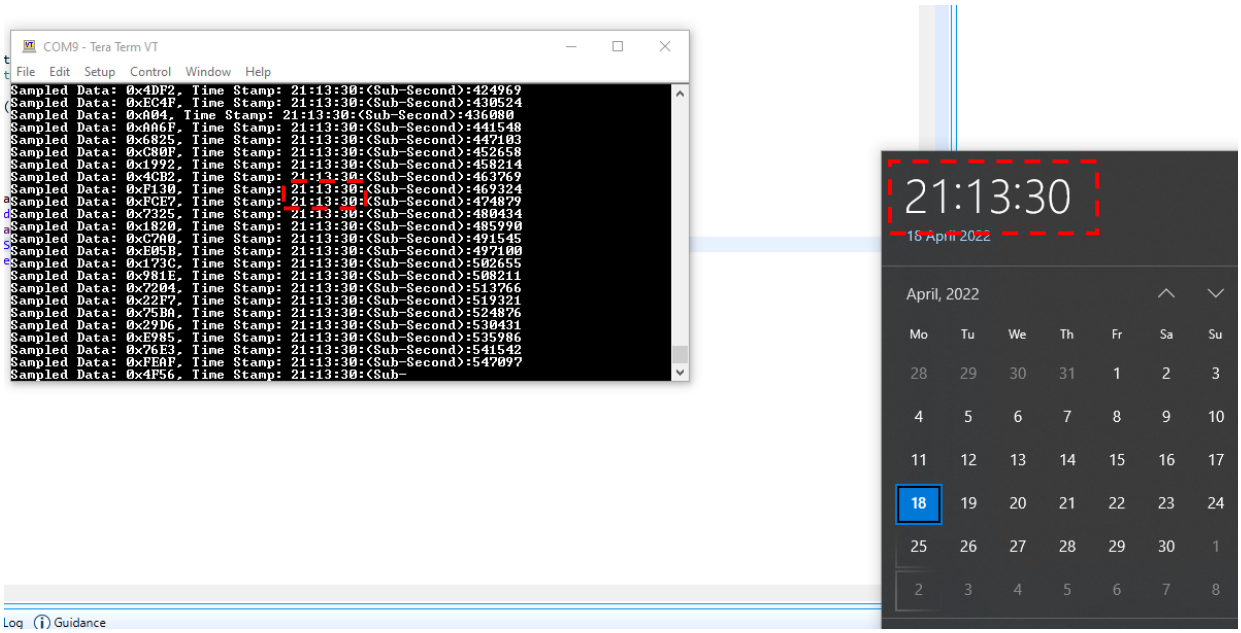


Fig. 2.11 Test Run Results

3.3 GPS Module

C++ Code:

```
void DemoRun() {
   while (1) {
      if (GPS.ping) {
         GPS_formatSentence(&GPS);
         if (GPS_isFixed(&GPS)) {
            printf("UTC TIME: %f\n",GPS_getTime(&GPS));
            xil_printf("Latitude: %s\n\r", GPS_getLatitude(&GPS));
            xil_printf("Longitude: %s\n\r", GPS_getLongitude(&GPS));
            xil_printf("Altitude: %s\n\r", GPS_getAltitudeString(&GPS));
            xil_printf("Number of Satellites: %d\n\n\r", GPS_getNumSats(&GPS));
         } else {
            xil_printf("Number of Satellites: %d\n\r", GPS_getNumSats(&GPS));
         }
         GPS.ping = FALSE;
      }
   }
   DisableCaches();
}
```

```
double GPS_getTime(PmodGPS *InstancePtr) {
   return atof(InstancePtr->GGAdata.UTC);
}

int GPS_getNumSats(PmodGPS *InstancePtr) {
   return atoi(InstancePtr->GGAdata.NUMSAT);
}

double GPS_getPDOP(PmodGPS *InstancePtr) {
   return atof(InstancePtr->GSAdata.PDOP);
}

double GPS_getAltitude(PmodGPS *InstancePtr) {
   return atof(InstancePtr->GGAdata.ALT);
}

double GPS_getSpeedKnots(PmodGPS *InstancePtr) {
   return atof(InstancePtr->VTGdata.SPD_N);
}

double GPS_getSpeedKM(PmodGPS *InstancePtr) {
   return atof(InstancePtr->VTGdata.SPD_KM);
}

double GPS_getHeading(PmodGPS *InstancePtr) {
   return atof(InstancePtr->VTGdata.COURSE_T);
}
char *GPS_getLatitude(PmodGPS *InstancePtr) {
    return InstancePtr->GGAdata.LAT;
}

char *GPS_getLongitude(PmodGPS *InstancePtr) {
    return InstancePtr->GGAdata.LONG;
}
```

Simulated Waveforms/ Results:

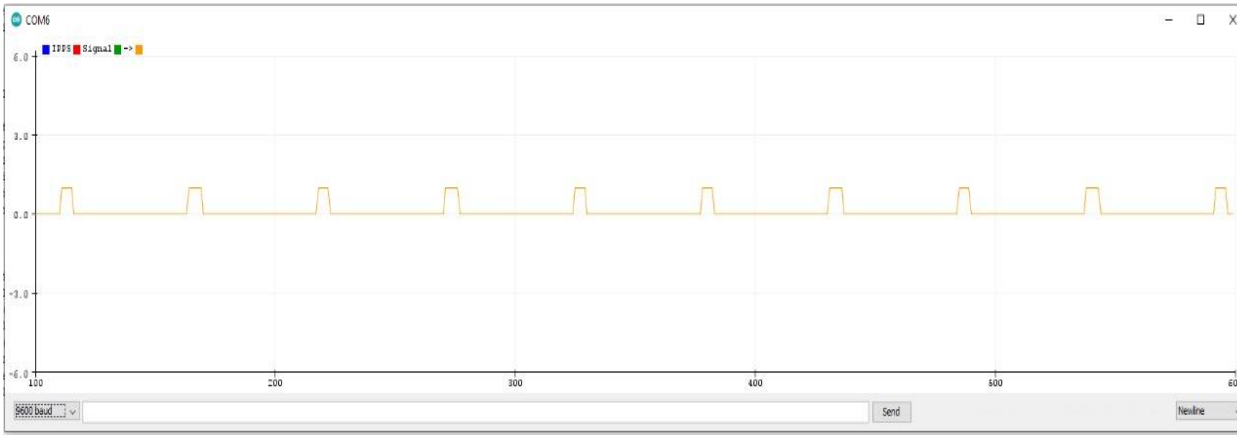


Fig. 2.11 1PPS Signal from PmodGPS

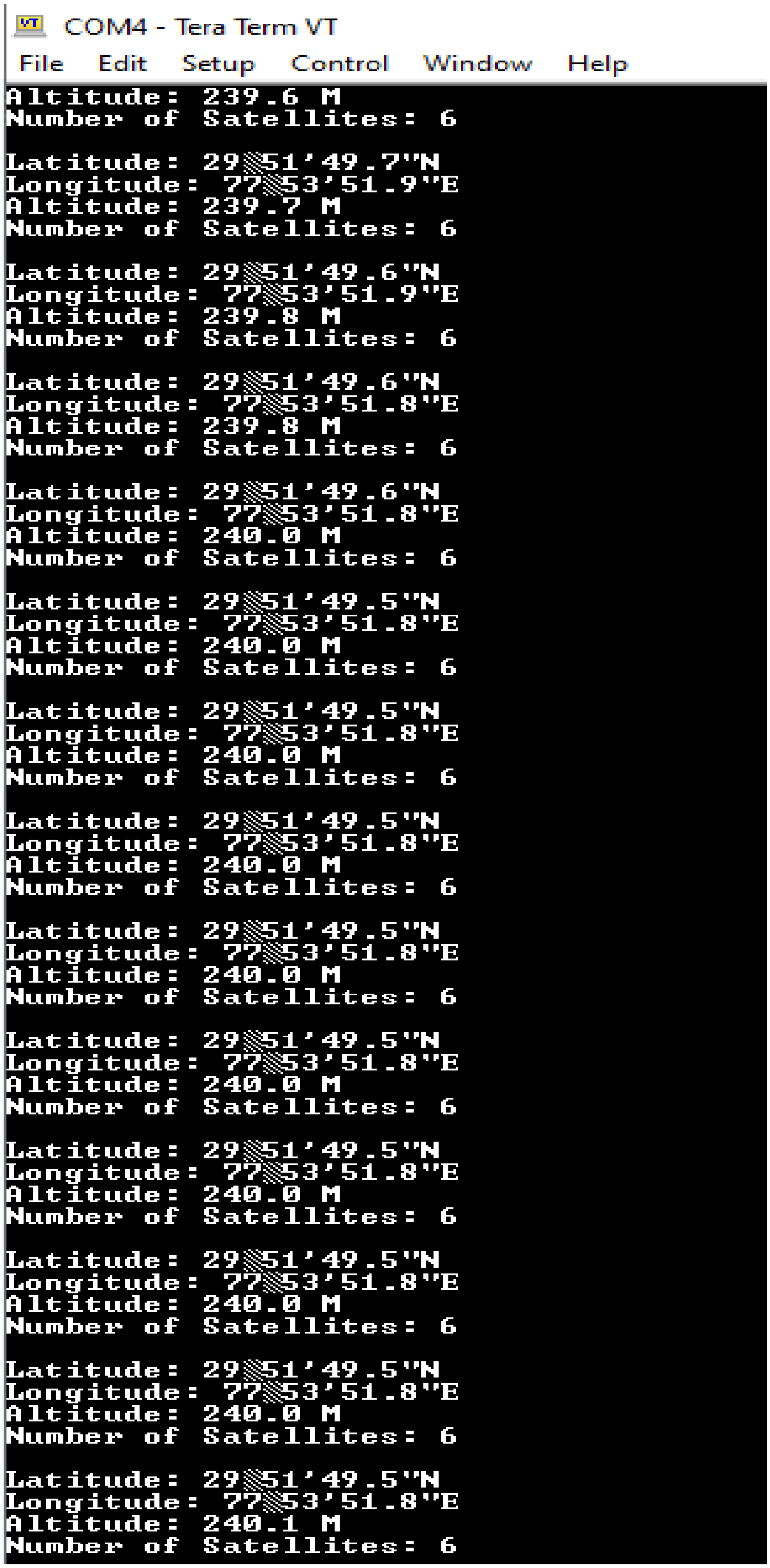


Fig. 2.12 PmodGPS Output

3.4 Test Input

Verilog Code:

```
`timescale 1ns / 1ps


module lookup(
    input clk
    );

        reg [7:0] rand;
        integer fd;
        initial begin
        fd = $fopen("out.txt", "w");
        @(posedge clk)
       $fwriteb(fd, $random);
        $fclose(fd);
        end
endmodule
```

Simulated Waveforms:

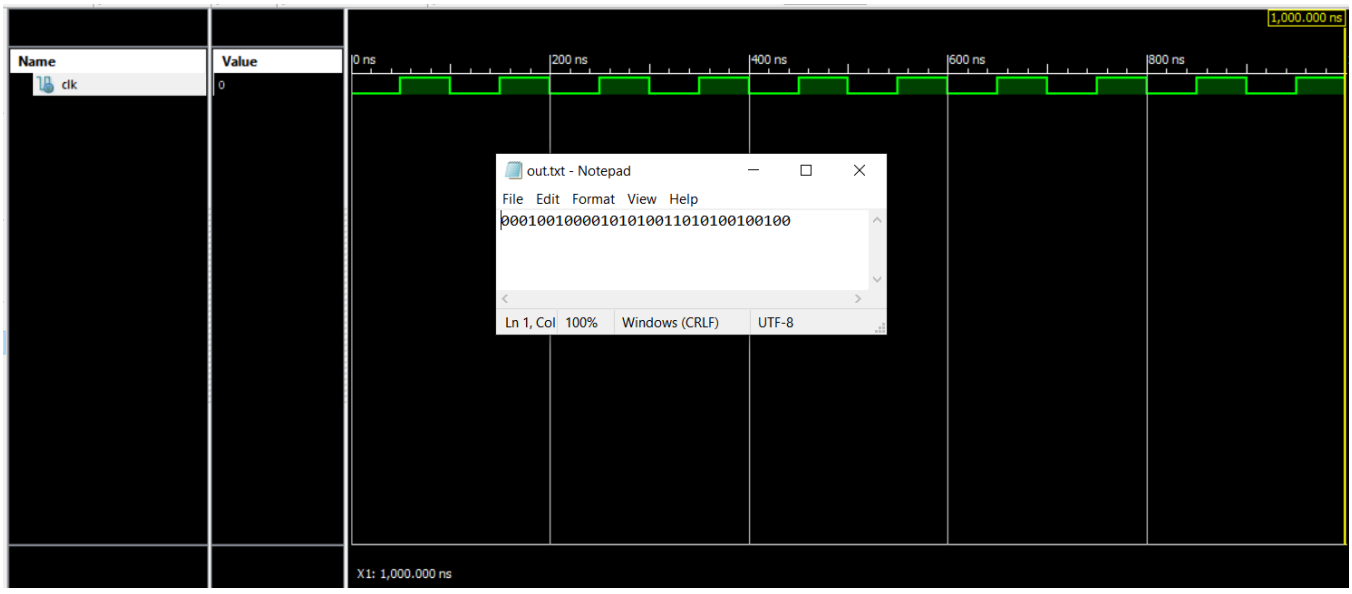


Fig. 2.13 Timestamped Data Logging

3.5 Positive Edge Detector

Verilog Code:

```
`timescale 1ns / 1ps

module Pos_Edge( input clk, input sig, output pe);

reg sig_dly;

always @(posedge clk) begin
 sig_dly <= sig;
 end

 assign pe = sig & ~sig_dly;


endmodule
```

Simulated Waveforms:

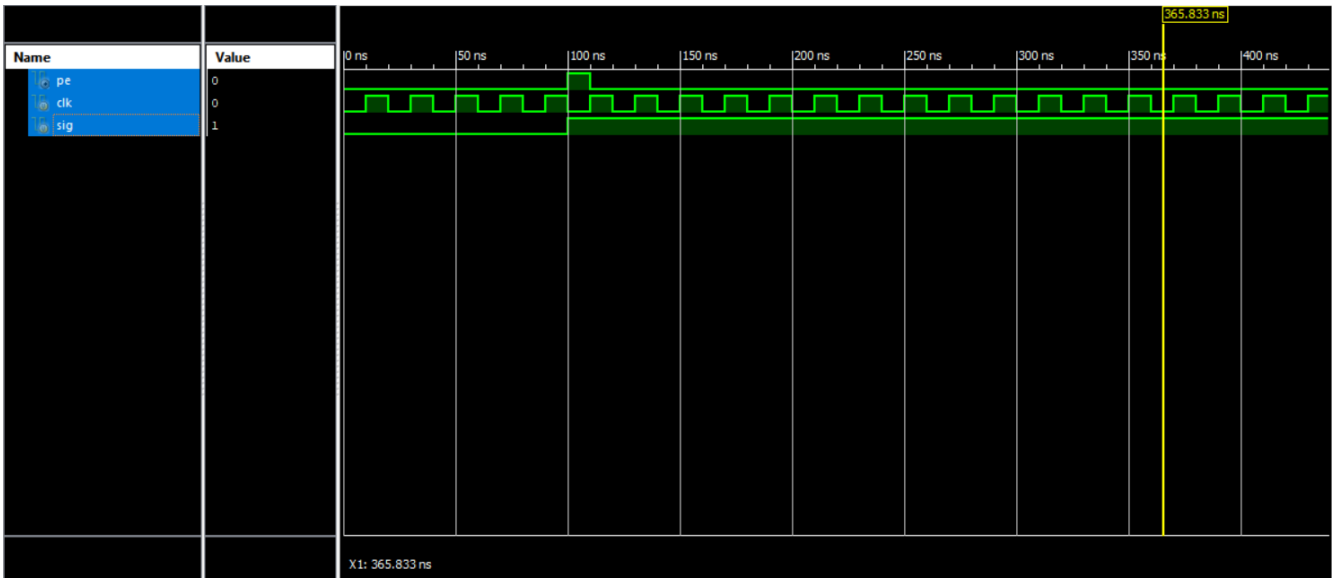


Fig. 2.14 Positive Edge Detector

Appendix I

- **Xilinx Workflow**

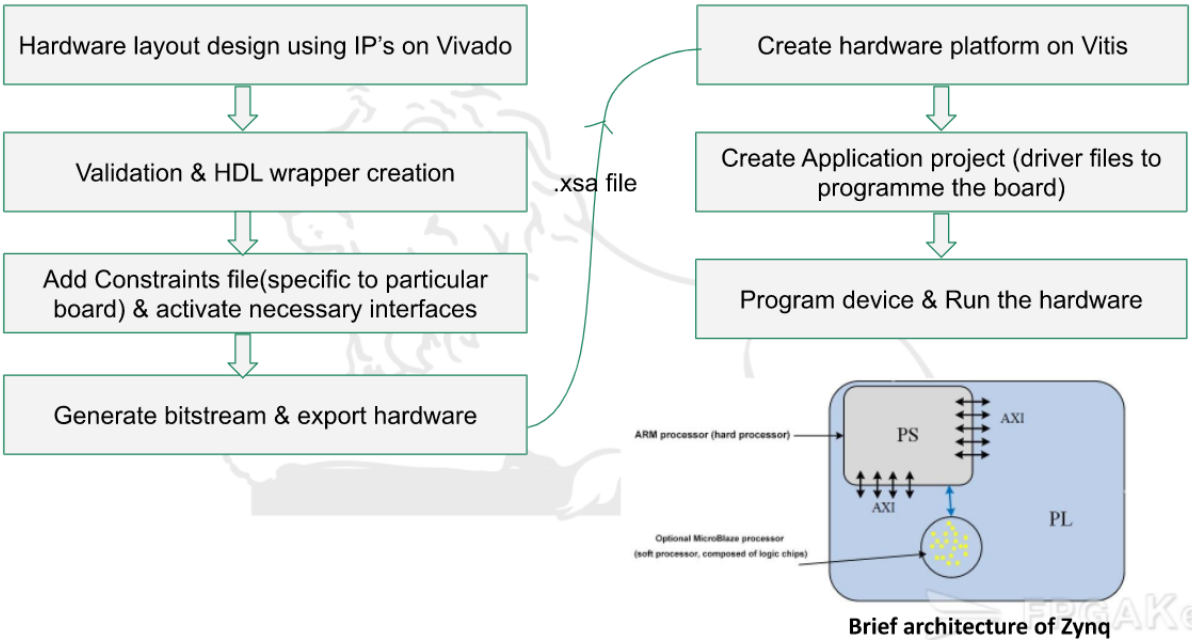


Figure A1.1

- **More Hardware Details**

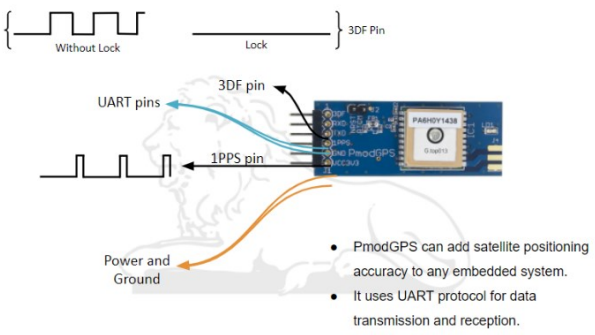


Figure A1.2

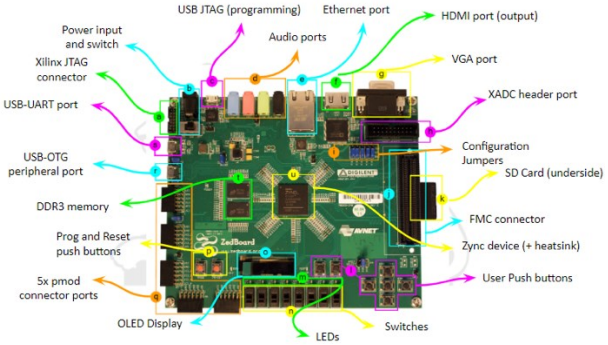


Figure A1.3

- Currently, the UART protocol is used for the transmission of time-stamped data.
- The Zedboard clock is capped to 1MHz for this project since more sophisticated high-speed communication protocols than UART would be required to transfer time-stamped data when clock speed is increased to much higher frequencies.
- Random data is generated and time-stamped for the demonstration in communication applications.

References

[1] The ZYNQ BOOK TUTORIAL: Tutorial for Zybo and ZedBoard

[2] GlobalTop-FGPMMOPA6H-Datasheet-V0A-Preliminary

[3] https://gisgeography.com/trilateration-triangulation-gps/

[4] T.M.Schmidl and D.C.Cox, “Robust Frequency and timing synchronization for OFDM,” IEEE Trans. Commun.,vol.45, no.12, pp.1613-1621, Dec 1997.

[5] Sterzbach, B. GPS-based Clock Synchronization in a Mobile, Distributed Real-Time System. Real-Time Systems 12, 63–75 (1997)

[6] D. C. Jefferson, S. M. Lichten and L. E. Young, "A test of precision GPS clock synchronization," Proceedings of 1996 IEEE International Frequency Control Symposium, 1996, pp. 1206-1210.

[7] Attila Pásztor and Darryl Veitch. 2002. PC based precision timing without GPS. SIGMETRICS Perform. Eval. Rev. 30, 1 (June 2002), 1–10.

[8] PmodGPS™ Reference Manual