International Journal of Electrical and Computer Engineering (IJECE)
Vol. 12, No. 4, August 2022, pp. 3466~3476
ISSN: 2088-8708, DOI: 10.11591/ijece.v12i4.pp3466-3476

Design of field programmable gate array-based data processing system for multi global positioning system receiver

Zainul Abidin[1], Nauval Aryawiratama[2], Adharul Muttaqin[2], Ryoichi Miyauchi[3]
[1]Department of Computer Engineering, Faculty of Computer Science, Universitas Brawijaya, Malang, Indonesia
[2]Department of Electrical Engineering, Faculty of Engineering, Universitas Brawijaya, Malang, Indonesia
[3]Department of Electrical Engineering, Faculty of Engineering, Tokyo University of Science, Tokyo, Japan

Article Info

Article history:

Received Aug 17, 2021
Revised Feb 11, 2022
Accepted Mar 2, 2022

Keywords:

Ballistic object
Field programmable gate array
Multi GPS receiver
Root mean square error
Tracking

ABSTRACT

A global positioning system (GPS) sensor is needed for a ballistic/moving object to do position tracking. In previous study, a multi GPS processing system was made using several microcontrollers and data processing cannot be done simultaneously. Therefore, it was considered as ineffective system. In this research, field programmable gate array (FPGA)-based data processing system for multi-GPS receiver was proposed. The proposed system was designed to reduce root mean square error (RMSE). There are two main processes in the proposed system which work in parallel, i.e. data parsing and data processing. Raw data from GPS receiver was collected and calculated to get average value, then sent it through serial communication to show result. Experimental results confirm the RMSE value of the proposed system is smaller than the conventional one. The RMSE for latitude, longitude, and altitude decrease by 38.46%, 58.28%, and 24.80%, respectively.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Zainul Abidin
Department of Computer Engineering, Faculty of Computer Science, Universitas Brawijaya
Veteran Street 8, Lowokwaru, Malang, East Java, 65145, Indonesia
Email: zainulabidin@ub.ac.id

1. INTRODUCTION

Recently, detecting and tracking moving objects, such as ballistic object become interesting issues among researchers. Development of technology leads to quick and precise coordinate's determination of target position. It will affect to firing accuracy and effectiveness [1]–[4]. Ballistic object trajectory tracking system is very important in flight test. The system can be based on inertial sensors (Inertial Navigation System), radar transponders, Doppler tracking, and global positioning system (GPS). Tracking using commercial GPS cannot work well in maximum altitude. Another thing that often happens is the GPS signal is not well received by the antenna placed on the nose-cone of the object [5].

In order to conduct tracking, it is necessary to have a GPS sensor. This device is used to receive data from 24 satellites. At least, three satellites must be connected to receive data properly. Each satellite will send a navigation message to the earth continuously. The receiver on earth will capture and track the satellites to obtain and process information in real time to get accurate data such a latitude, longitude, altitude, speed, time, and so on [6].

The accuracy of commercial type GPS data has a variation of about 20 meters, so it still needs to be improved. In order to solve this issue, Widada [5] has proposed a method of combining several GPS receivers to improve the accuracy and reliability of the tracking system. The study made a prototype using 4 GPS receivers placed on the nose-cone as shown in Figure 1, each GPS is connected to a microcontroller for data parsing. Then the data from each microcontroller is sent alternately to the main microcontroller to

calculate the average value, and transmit using radio telemetry to the launch station. The study is claimed to reduce root mean square error (RMSE) for latitude around 30% and longitude around 30%, while for altitude around 40% [5].

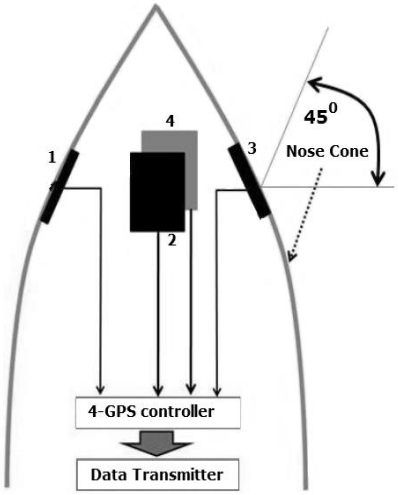


Figure 1. Prototype using 4 GPS receivers

The method is considered ineffective because the proposed system cannot receive GPS data simultaneously and also needed more than one microcontroller. Field programmable gate array (FPGA) is a device that has typical function in digital logic [7], [8]. FPGA-based system has been applied to various applications, such as in microelectronics, industrial control, robotics, and internet of things (IoT) [9]–[11]. According to Deepchand's research, FPGA is able to perform multi-channel universal asynchronous receiver transmitter (UART) communication processes and multi-channel pulse-width modulation (PWM) generator [12], [13]. The FPGA is able to realize parallel data processing so that it can support data processing in real time [14]–[19]. Whereas Maxfield stated that FPGA has more than enough capabilities to implement soft-core processor such as embedded microcontroller with various I/O which have very broad functions [20]. The speed of FPGA based embedded systems is faster than the microcontroller one. This was proven by Haskell and Hanna's research [21]. In addition, another advantage of FPGA is the designer can freely determine the I/O to be implemented in the system, so that an FPGA has higher flexibility than a microcontroller [13]. Based on the background, in this research, a multi-GPS receiver data processing system based on FPGA was designed and evaluated. With the simultaneous reading, the system is proposed to produce GPS data output with lower RMSE compare to the previous research.

2. RESEARCH METHOD

This research was conducted by considering several specifications. Discussions are presented in this section related to system architecture, system flowchart, subsystem design, and experimental method. Main subsystems were designed to realize functions as data parser, data processor, and UART transmitter.

2.1. System architecture

The multi GPS receiver data processing system consists of 3 main blocks of digital system i.e. data parser block, data processing block, and UART transmitter block as shown in Figure 2. The data parser block has a function to parse the data that has been received from the GPS sensor. Then, the data processing block calculates the average of GPS data received. While, UART Transmitter block serves to send data serially to other devices such as personal computer (PC) in order to monitor the results.

The digital system is designed and verified using Verilog hardware description language (VHDL). Simulation of input and output of a digital system can be controlled using the VHDL [22]. The VHDL provides alternative approach to circuit design just by using text descriptions and independent of schematics.

Each GPS receives raw data which must be interpreted following sentences of national marine electronics association (NMEA). The NMEA sentences consist of $GPGGA, $GPGLL, $GPVTG, $GPRMC, and $GSGGA [23]. Information of latitude, longitude, and altitude are required when the GPS is applied to

moving/ballistic object. In addition, number of satellites connected to the GPS is important as well to ensure position information. Furthermore, the $GPGGA sentence contains the required information. Therefore, data parser must be implemented to extract $GPGGA from the GPS raw data [23]–[25].

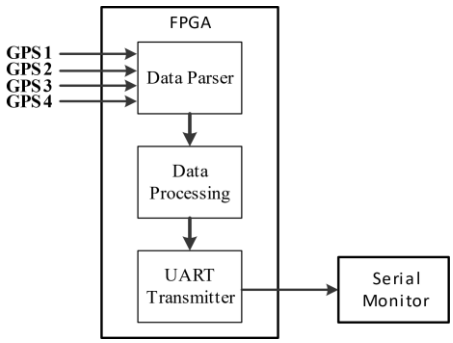


Figure 2. Main block diagram of the proposed system

2.2. System flowchart

The proposed system runs based on flowchart shown in Figure 3. The process starts with taking data from four GPS sensors simultaneously using the FPGA board. After the raw data from the GPS sensors is obtained, the data parsing process is applied so that only GPS data with the NMEA "$GPGGA" format will be processed. After the data is obtained, the parsing process is carried out again to get latitude, longitude, altitude, the number of satellites data. The next process is checking the number of satellites. If the number of satellites from one or several GPS sensors is 3 or more, the data will go directly to the next process, i.e. the calculation of the average data. However, if the number of satellites from one or several GPS sensors is less than 3, then the data owned by the GPS sensors are filled in with a 0 value and immediately forwarded to the next process, i.e. the calculation of average data. After the average is calculated, the next process is merge the average result data and also the input data from each GPS into one data packet. Then, the last process is sending the serial data to display on PC and return to the process of collecting data from four GPS sensors.

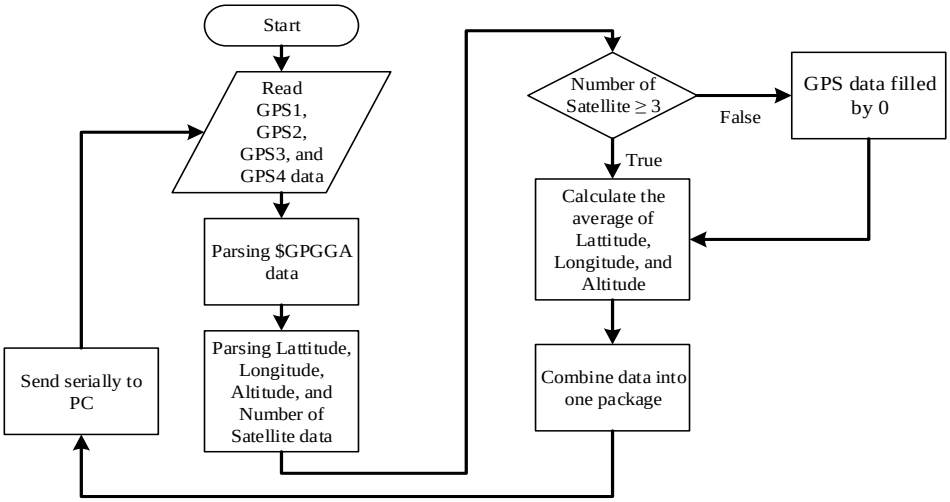


Figure 3. System flowchart

2.3. Subsystem

First subsystem to be discussed is data parser. As illustrated in Figure 4, there are 4 inputs from GPS sensors, each GPS sensors is connected to a parser module. The input is transferred directly to Rx port of the parser module. Data from GPS will be received serially with UART communication via the UART Receiver module. Next process is the data will be sorted using the brute-force string matching algorithm. This

algorithm works by matching characters every 8 bits of data. Figure 5 illustrates the algorithm [25]. In this research, the data of "$GPGGA" must be sorted. The code shows the fix data from GPS, however, only data of latitude, longitude, altitude, and also the amount of satellites will be extracted later. The character matching process is conducted by shifting. In order to do the parsing process, it is necessary to make a state diagram so that the process runs systematically. The data parsing scheme can be explained by state diagram shown in Figure 6. The state diagram shows that there will be a transition between states through character detection process. If it has not or does not detect the right character, then the process will turn back to the "Dollar" state. Process of transition between states can be explained by Table 1. After being sorted, the output data will be in the form of latitude, longitude, altitude and the number of satellites for each GPS sensor. This data will then be forwarded to the data processing block.

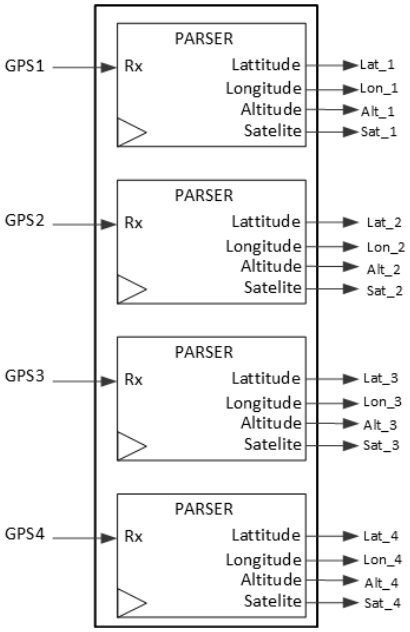


Figure 4. Data parser block

Figure 5. $GPGGA parsing scheme

Second subsystem is data processing block. This block is useful to process data from data parser block outputs. As shown in Figure 7, this block consists of *ASCII_to_integer*, *buffer_ASCII*, *buffer_module*, validator, average, and *integer_to_ASCII* entities. Data from the data parser block will be firstly forwarded to *buffer_ASCII* entity. This entity serves to hold the input data so that the input data entered randomly can be arranged together. The *ASCII_to_integer* entity is used to convert the input data which is still in the ASCII form into integer data, so that it can proceed to the next process. As for *buffer_ASCI*I entity, data from each GPS will be hold temporarily. Data will be issued when *eject_data* port gets logic 1.

After the data is converted to an integer, then the data is forwarded to the *buffer_module* entity which is used to hold the data first. Latitude, longitude, and altitude data will be removed from this buffer according to the input provided from the *gps_valid* port that comes from the entity validator. This entity checks the number of satellites available. If GPS is able to receive data from 3 or more satellites, then the data from the GPS is considered to be valid.

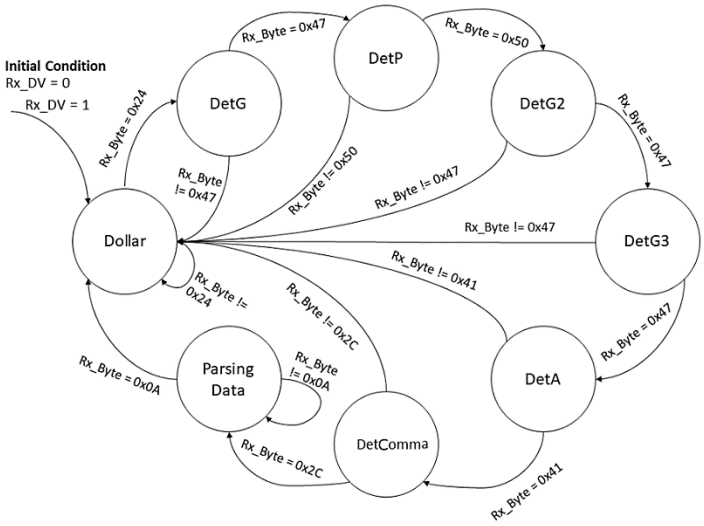


Figure 6. Parsing state diagram

Table 1. State table of parsing

Initial state	Destination state	Transition trigger	Explanation
Dollar	DetG	Rx_Byte = 0x24	"$" character detection
DetG	DetP	Rx_Byte = 0x47	"G" character detection
DetP	DetG2	Rx_Byte = 0x50	"P" character detection
DetG2	DetG3	Rx_Byte = 0x47	"G" character detection
DetG3	DetA	Rx_Byte = 0x47	"G" character detection
DetA	DetComma	Rx_Byte = 0x41	"A" character detection
DetComma	Parsing Data	Rx_Byte = 0x2C	"," character detection
Parsing Data	Dollar	Rx_Byte = 0x0A	New line detection

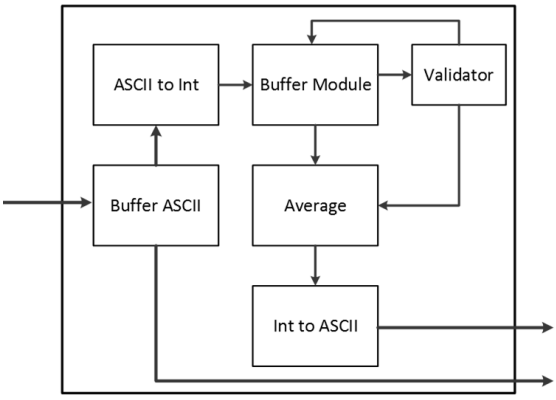


Figure 7. Data processing block

Table 2 shows data will be forwarded to the next process, namely the average calculation process that occurs at the average entity. In the average entity, input data will be calculated on average by adding data from a valid GPS and then doing the division process in accordance with the data from the valid GPS. The last process is conversion from integer data into ASCII. This process is carried out by the *integer_to_ASCII* entity. Data needs to be changed to ASCII because it will be used for serial transmission. Double dabble algorithm is used in this process. This algorithm is basically changing binary data into binary-coded decimal (BCD) with a width of 4 bits [26], [27]. The basic principle of this algorithm is to do a bit shift. Because ASCII data requires an 8 bits data width, the end of the data will be added with a 4 bits width as MSB with the value of "0011". This is important because it refers to the ASCII code in Table 3. Then, well known UART Transmitter block is responsible for sending data every 8 bits. Data will be sent with 9600 bps baud rate.

Table 2. Buffer table

GPS_Valid	GPS1	GPS2	GPS3	GPS4
0000	No	No	No	No
0001	Yes	No	No	No
0010	No	Yes	No	No
0011	Yes	Yes	No	No
0100	No	No	Yes	No
0101	Yes	No	Yes	No
0110	No	Yes	Yes	No
0111	Yes	Yes	Yes	No
1000	No	No	No	Yes
1001	Yes	No	No	Yes
1010	No	Yes	No	Yes
1011	Yes	Yes	No	Yes
1100	No	No	Yes	Yes
1101	Yes	No	Yes	Yes
1110	No	Yes	Yes	Yes
1111	Yes	Yes	Yes	Yes

Table 3. ASCII code

Character	Decimal	Binary
0	48	00110000
1	49	00110001
2	50	00110010
3	51	00110011
4	52	00110100
5	53	00110101
6	54	00110110
7	55	00110111
8	56	00111000
9	57	00111001

2.4. Experimental method

In order to evaluate the results, experiments were conducted with simulation and actual implementation on the FPGA. The simulator used in this study is ModelSim. The simulation is necessary to confirm the proposed system is able to provide output in accordance with the expectation. The next process is the implementation to Altera Cyclone IV FPGA board. There are 2 experiments to be performed, i.e.:

a) Testing with 1 GPS sensor

The purpose of this test is to confirm the proposed system can run well by receiving data from 1 GPS sensor. In order to compare the performance, this test is conducted by connecting the GPS sensor to Arduino and FPGA board. The parameters to be observed are the output data in the form of latitude, longitude, and altitude of the sensor. The number of data taken from this test is 500.

b) Testing with 4 GPS sensors

The purpose of this test is to confirm the proposed system can receive data from 4 GPS sensors simultaneously. In order confirm the performance, this test is applied to Arduino-based system (5 microcontrollers) and FPGA-based system. The same parameters are observed, i.e. latitude, longitude, and altitude of the sensor. This test generates 500 data. After testing, the RMSE will be calculated from the data obtained. The RMSE can be calculated using (1).

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(x_{obs,i}-x_{static,i})^2}{n}} \quad (1)$$

where:

RMSE = Root mean square error
xobs,i = Real-time data value
xstatic,i = Sampling data value
n = Sample number.

3. RESULTS AND DISCUSSION

Evaluation results are presented in this section. The evaluation results consist of simulation and experimental results. Prototypes based on Arduino and FPGA were fabricated to conduct the experiments.

3.1. Data parser simulation result

Data parser is tested through simulation by providing raw data from GPS as the input and taking only the $GPGGA sentence. The simulation uses data of latitude, longitude, altitude, satellite number as shown in Table 4. The simulation result of the parsing process is mentioned in Figure 8. There was a state change for $GPGGA parsing. This means that $GPGGA sentence can be found. At the end of the timing diagram, there is "ParsingData" state and followed by a rising edge on the "done" signal. These mean that the parsing process has been completed. Furthermore, Figure 9 shows the parsing simulation results of latitude, longitude, altitude and number of satellite. The result shows that the output of the parsing process is in accordance with the test data provided in Table 4.

Table 4. $GPGGA test data

	Lat	*Lon*	*Alt*	*Sat*
GPS1	0756.89465	11238.31502	469.0	04
GPS2	0755.87746	11236.31419	465.0	04
GPS3	0756.88464	11231.31145	465.0	05
GPS4	0755.87475	11238.31611	478.0	04

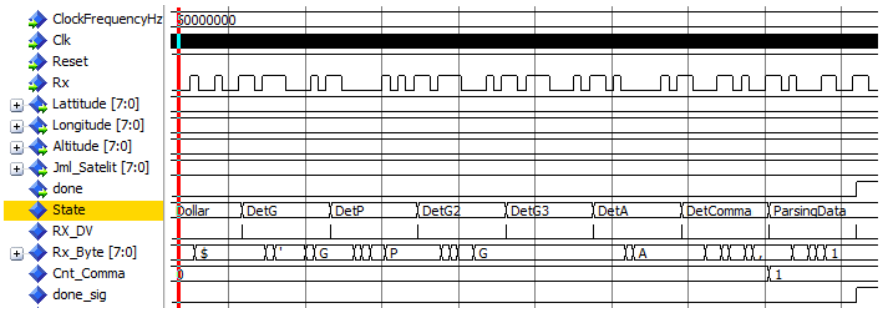


Figure 8. Parsing state simulation result

Figure 9. Latitude, longitude, altitude and satellite number parsing results

3.2. Data processing simulation result

In this test, the output data from the parser will be directly processed. Simulation is carried out for 550 ms. With the test data as in Table 4, the results are presented in Figure 10. It shows input data from data parser is synchronized. The input data is also calculated for its mean value, so that the average results of latitude, longitude, and altitude can be obtained.

3.3. Top level entity simulation result

Top level entity is tested through simulation by providing raw data from GPS. Simulation is carried out for 550 ms. This simulation is a combined simulation of data parser and data processing blocks with

additional UART Transmitter block. Figure 11 shows that the control state changes from idle to digit 1 when the *done_conv* signal originating from the *int_to_ASCII* entity has logic 1. In addition, the counter will also run to perform 8-bit data transactions with 9600 baud rate. Figure 12 shows the condition for writing process. Red circle number 1 shows the detail phenomenon of the signal *done_conv* experienced a falling edge. Then the red circle number 2 illustrate the data transmission has begun. It is marked by the falling edge of the TX signal. After that, in the "tulis state" (means write state), there will be a process of writing data on the UART transmitter block. Every 8 bits that come from RAM are released from the TX port serially with a baud rate of 9600.

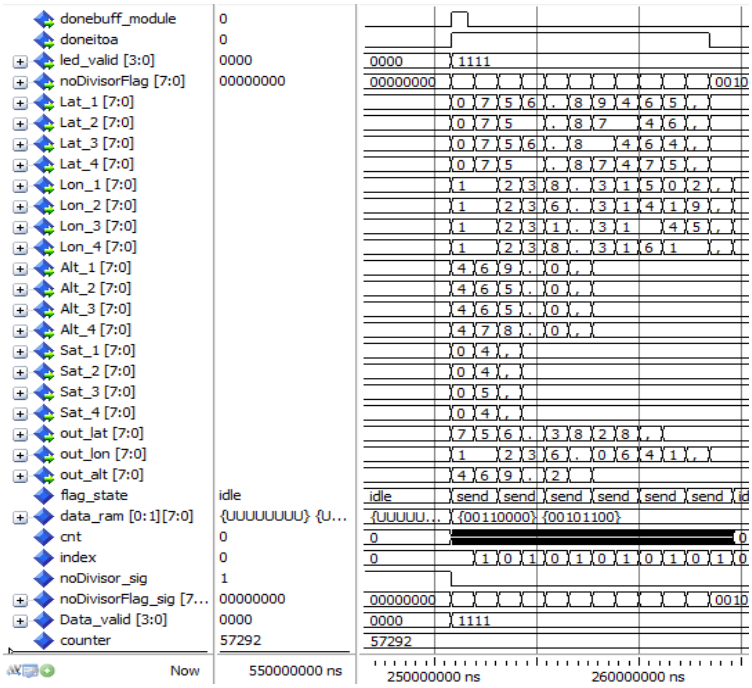


Figure 10. Data processing simulation results

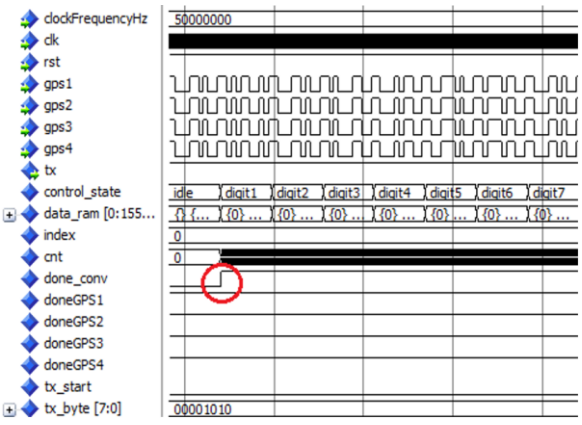


Figure 11. Top level entity simulation results

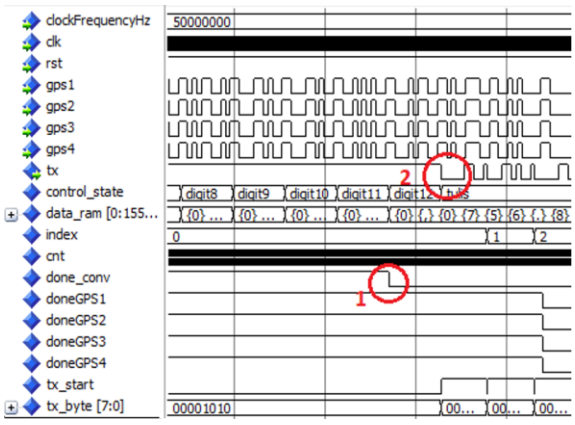


Figure 12. Data writing process

3.4. Prototype testing result

Experiment was conducted by using prototypes see Figures 13 and 14. The prototypes were tested by put the prototypes in a stationary position and taking sample of 500 data. After RMSE calculations, the following data are obtained. Table 5 summarize the testing result of the prototype with Arduino. The Arduino-based prototype decreases the RMSE until 58.87%, 33.05%, and 40.46% for latitude, longitude, and altitude, respectively. Based on Table 6, the prototype with FPGA can decrease the RMSE until 38.46%, 58.28%, and 24.80% for latitude, longitude, and altitude, respectively.

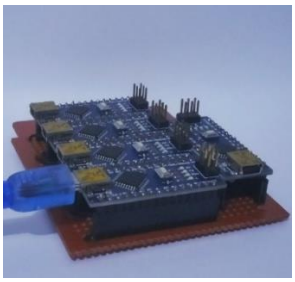


Figure 13. Arduino based prototype

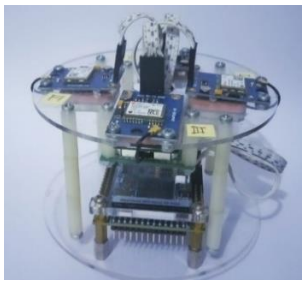


Figure 14. FPGA-based prototype

Table 5. Arduino-based RMSE

Parameter	1 GPS RMSE	4 GPS RMSE	RMSE Decrease (%)
Latitude (°)	4.51e-03	1.85e-03	58.87%
Longitude (°)	3.34e-03	2.24e-03	33.05%
Altitude (m)	5.201	3.096	40.46%

Table 6. FPGA-based RMSE

Parameter	1 GPS RMSE	4 GPS RMSE	RMSE Decrease (%)
Latitude (°)	2.86e-03	1.76e-03	38.46%
Longitude (°)	3.34e-03	1.39e-03	58.28%
Altitude (m)	0.558	0.419	24.80%

From the results summarized in Tables 5 and 6, the RMSEs of the latitude parameter for 1 GPS and 4 GPS at FPGA are 1.57 and 1.05 smaller than the Arduino, respectively. At the longitude parameter, the FPGA based prototype has RMSEs of 0.99 and 1.60 smaller than Arduino-based one for 1 GPS and 4 GPS, respectively. Whereas the RMSE at altitude parameter for 1 GPS and 4 GPS on FPGA are 9.31 and 7.37 smaller than Arduino-based one, respectively. From the experiment results, the RMSEs of FPGA-based prototype is smaller than the Arduino-based prototype. It means the FPGA-based prototype has better performance than the Arduino-based prototype.

4. CONCLUSION

In this study, a multi GPS receiver data processing system has been proposed and successfully implemented on the FPGA. The proposed system can produce output in accordance with the input given during the simulation. For each parameter, the proposed system produces a smaller RMSE than an Arduino microcontroller-based system. Therefore, the proposed system has better performance than the Arduino microcontroller-based one.

In the future, a data storage system on secure digital (SD) cards will be added to the proposed system, so that data retrieval process can be simpler. Furthermore, direct testing on moving objects will also be conducted to see the performance of the system in the tracking process. In order to confirm the system accuracy, testing at location with certain values of latitude, longitude, and altitude must be conducted.

ACKNOWLEDGEMENTS

Authors would like to thank *Lembaga Penelitian dan Pengabdian Masyarakat Universitas Brawijaya (LPPM - UB)* through *Hibah Peneliti Pemula* (Award No. 536.4.1/UN10.C10/PN/2021).

REFERENCES

[1] M. Brzozowski, M. Pakowski, M. Nowakowski, M. Myszka, and M. Michalczewski, "Radiolocation devices for detection and tracking small high-speed ballistic objects-features, applications, and methods of tests," *Sensors*, vol. 19, no. 24, Dec. 2019, doi: 10.3390/s19245362.

[2] M. Brzozowski, M. Pakowski, M. Nowakowski, M. Myszka, and M. Michalczewski, "Radars with the function of detecting and tracking artillery shells - selected methods of field testing," in *2019 IEEE 5th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, Jun. 2019, pp. 429–434, doi: 10.1109/MetroAeroSpace.2019.8869656.

[3] A. K. Maini, *Handbook of defence electronics and optronics*. Chichester, UK: John Wiley & Sons, Ltd, 2018.

[4] D. K. Barton, *Modern radar system analysis*. Artech Print on Demand; First Edition, 1988.

[5] W. Widada, "Method of combining multi-GPS receivers to improve accuracy and reliability of the tracking system of sounding rocket," *Jurnal Teknologi Dirgantara*, vol. 12, no. 1, pp. 1–10, 2014.

[6] Pan Ming, "FPGA-based GPS application system design," in *2009 Chinese Control and Decision Conference*, Jun. 2009, pp. 1384–1387, doi: 10.1109/CCDC.2009.5191575.

[7] S. M. Trimberger, *Field-programmable gate array technology*. Boston, MA: Springer US, 1994.

[8] S. M. Trimberger and J. J. Moore, "FPGA security: motivations, features, and applications," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1248–1265, Aug. 2014, doi: 10.1109/JPROC.2014.2331672.

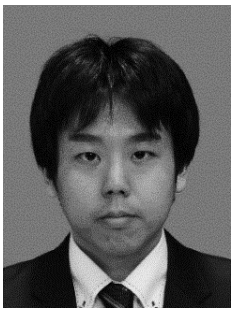
[9] E. Monmasson, L. Idkhajine, M. N. Cirstea, I. Bahri, A. Tisan, and M. W. Naouar, "FPGAs in industrial control applications," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 2, pp. 224–243, May 2011, doi: 10.1109/TII.2011.2123908.
[10] J. J. Rodriguez-Andina, M. D. Valdes-Pena, and M. J. Moure, "Advanced features and industrial applications of FPGAs—a review," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 4, pp. 853–864, Aug. 2015, doi: 10.1109/TII.2015.2431223.
[11] M. Elnawawy, A. Farhan, A. Al Nabulsi, A. R. Al-Ali, and A. Sagahyroon, "Role of FPGA in internet of things applications," in *2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, Dec. 2019, pp. 1–6, doi: 10.1109/ISSPIT47144.2019.9001747.
[12] D. Jaiswal, R. Jain, and M. Z. Alam, "Design of multi channel UART controller based on FIFO and FPGA," *International Journal of Advanced Research in Computer Engineering & Technology*, vol. 1, no. 4, pp. 419–426, 2012.
[13] A. Muttaqin, S. D. Finnadi, Z. Abidin, and K. Araki, "FPGA based synchronous multi-channel PWM generator for humanoid robot," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 1, pp. 249–256, Feb. 2021, doi: 10.11591/ijece.v11i1.pp249-256.
[14] I. M. Popescu, B. Popa, R. Prejbeanu, and I. Cosmin, "Evaluation of parallel and real-time processing performance for some vibration signals using FPGA technology," in *2018 19th International Carpathian Control Conference (ICCC)*, May 2018, pp. 365–370, doi: 10.1109/CarpathianCC.2018.8399657.
[15] Z. Kokosinski and B. Malus, "FPGA implementations of a parallel associative processor with multi-comparand multi-search operations," in *2008 International Symposium on Parallel and Distributed Computing*, 2008, pp. 444–448, doi: 10.1109/ISPDC.2008.42.
[16] N. Fujita, R. Kobayashi, Y. Yamaguchi, and T. Boku, "Parallel processing on FPGA combining computation and communication in openCL programming," in *2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, May 2019, pp. 479–488, doi: 10.1109/IPDPSW.2019.00089.
[17] F. A. Alquaied, A. I. Almudaifer, and M. A. AlShaya, "A novel high-speed parallel sorting algorithm based on FPGA," in *2011 Saudi International Electronics, Communications and Photonics Conference (SIECPC)*, Apr. 2011, pp. 1–4, doi: 10.1109/SIECPC.2011.5877001.
[18] W. Chen *et al.*, "FPGA-based parallel implementation of SURF algorithm," in *2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS)*, Dec. 2016, pp. 308–315, doi: 10.1109/ICPADS.2016.0049.
[19] Z. Zhou, X. Xin, S. Zhaolin, and L. Nan, "FPGA based high-speed parallel transmission system design and implementation," in *2010 International Conference on Intelligent System Design and Engineering Application*, Oct. 2010, pp. 767–770, doi: 10.1109/ISDEA.2010.438.
[20] C. Maxfield, *The design warrior's guide to FPGAs: devices, tools and flows*. UK: Newnes; 1st edition, 2004.
[21] R. E. Haskell and D. M. Hanna, "A VHDL--forth core for FPGAs," *Microprocessors and Microsystems*, vol. 28, no. 3, pp. 115–125, Apr. 2004, doi: 10.1016/j.micpro.2004.01.002.
[22] S. Palnitkar, *Verilog HDL: A guide to digital design and synthesis*. USA: Prentice Hall; 2nd edition, 2003.
[23] A. A. Bin Ariffin, N. H. A. Aziz, and K. A. Othman, "Implementation of GPS for location tracking," in *2011 IEEE Control and System Graduate Research Colloquium*, Jun. 2011, pp. 77–81, doi: 10.1109/ICSGRC.2011.5991833.
[24] R. H. R., V. L. W. B., M. N. S. Zainudin, and M. M. Ismail, "FPGA-based global positioning system," *International Journal of Electrical & Computer Sciences (IJECS)*, vol. 12, no. 5, pp. 11–14, 2012.
[25] Z. Abidin, N. Aryawiratama, A. Muttaqin, E. D. Arisandi, and C. Martineac, "Design of FPGA based data parser for global positioning system," in *2020 10th Electrical Power, Electronics, Communications, Controls and Informatics Seminar (EECCIS)*, Aug. 2020, pp. 159–162, doi: 10.1109/EECCIS49483.2020.9263455.
[26] S. Gao, D. Al-Khalili, J. M. P. Langlois, and N. Chabini, "Efficient realization of BCD multipliers using FPGAs," *International Journal of Reconfigurable Computing*, vol. 2017, pp. 1–12, 2017, doi: 10.1155/2017/2410408.
[27] S. Gao, D. Al-Khalili, and N. Chabini, "An improved BCD adder using 6-LUT FPGAs," in *10th IEEE International NEWCAS Conference*, Jun. 2012, pp. 13–16, doi: 10.1109/NEWCAS.2012.6328944.

BIOGRAPHIES OF AUTHORS

Zainul Abidin received the B.Eng. degree in electrical engineering from Universitas Brawijaya, Indonesia, in 2008. He received M.Eng. degree in electrical engineering from Universitas Brawijaya and University of Miyazaki, Japan, in 2011. His Ph.D. degree was granted by University of Miyazaki in 2017. Currently, he is an Assistant Professor at Universitas Brawijaya. His research interests include analog/digital circuit design, biological signal processing, and instrumentation design. He can be contacted at email: zainulabidin@ub.ac.id.

Nauval Aryawiratama was born in Pontianak, Indonesia, on November 26, 1997. He received the B.Eng. from Universitas Brawijaya, Indonesia, in 2020. He is currently working as an engineer. He can be contacted at email: narwiratama@gmail.com.

Adharul Muttaqin received the B.Eng. and M.Eng. degrees in electrical engineering from Bandung Institute of Technology, Indonesia, in 2000 and 2003, respectively. He is currently working as a lecturer in Department of Electrical Engineering, Universitas Brawijaya, Indonesia. His research interests include circuit and application design using field programmable gate array (FPGA), wireless sensor network, artificial intelligent, and microelectronics. He can be contacted at email: adharul@ub.ac.id.



Ryoichi Miyauchi was born in Kagoshima, Japan, on March 3rd, 1986. He received B. E. and M. E. degrees from Faculty of Engineering, University of Miyazaki, Miyazaki, Japan, in 2008 and 2010, respectively, and Dr.Eng. degree from Interdisciplinary Graduate School of Agriculture and Engineering, University of Miyazaki, Miyazaki, Japan, in 2019. From 2010 to 2015, he joined the Hitachi ULSI Systems Co., Ltd., Tachikawa, Japan. He was engaged in development of application software for Windows and development of voice recognition system for car navigation system. In 2019, he joined the Department of Electrical Engineering, Faculty of Engineering, Tokyo University of Science, Tokyo, Japan as an Assistant Professor. His main research interests are biological signal measurement system, analog integrated circuit and sensor interface. He can be contacted at email: rmiyauchi@rs.tus.ac.jp.