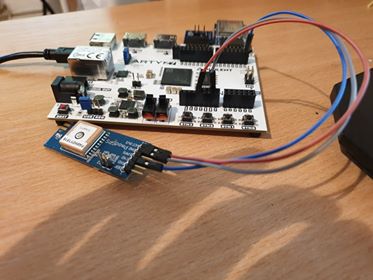


**Enhanced GPS with accelerometer**



Contents

[Enhanced GPS with accelerometer 2](#_Toc7984823)

[Instructions for Use 3](#_Toc7984824)

[Used Technologies and Algorithms 4](#_Toc7984825)

Project movie:

<https://youtu.be/cKSG60GwrWU>

# Enhanced GPS with accelerometer

,,Enhanced GPS with accelerometer’’ implemented on ArtyZ7 monitors and acquires physical data from GPS and accelerometer and transfers the data through WIFI protocol on a PC. On the PC an application written in Java computes the current position based on the acquired data: GPS coordinates and accelerometer X,Y,Z values. The GPS and accelerometer data could be viewed through a GUI (graphic user interface) implemented on the PC.

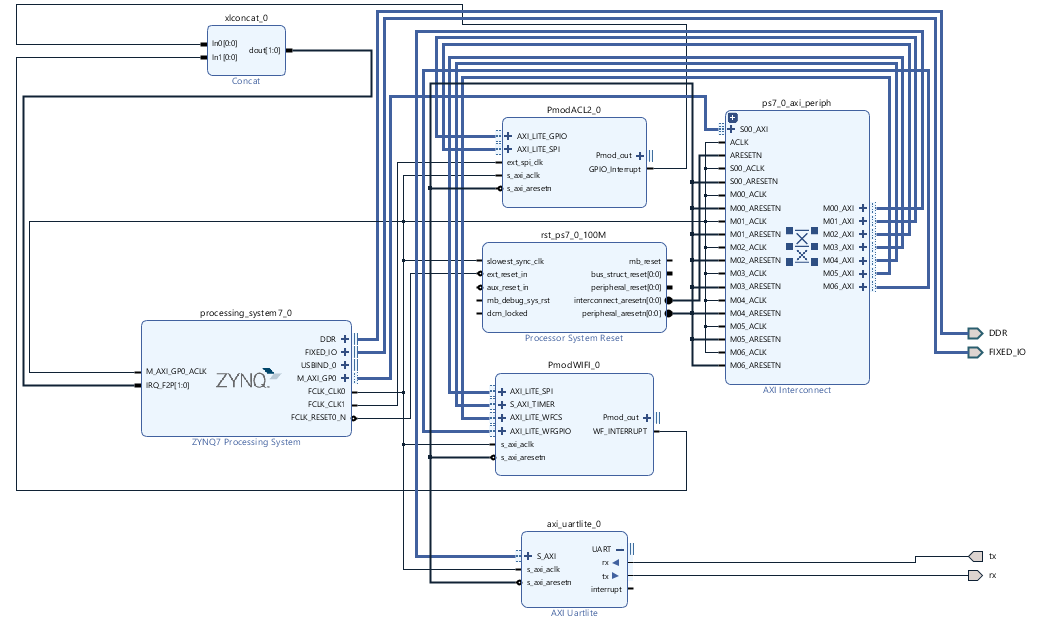
The purpose of the project is to complement loss of GPS signal or inaccurate GNSS signal measurement with motion analyzer.

This device uses PMod circuits for data acquisition: PMOD GPS, PMOD ACL and PMOD WiFi. The application implemented on ArtyZ7 board has 2 important stages:

1. A system on chip based on ARM and IP Cores from Digilent library is implemented . The IP cores used are: PmodACL2, PmodGPS and Pmod Wifi.
2. A software programming in C++ language is used for data acquisition and data transfer through Wifi.

The data are received on the PC and processed in order to enhanced the GPS location.

The implemented System on Chip may be seen in Figure 1.



Main systems used are:

* Interrupt System (IS)
* Data transfer through WiFi
* Direction vectors detection
* GPS data request

As for the software application we will use coordinates and GUI algorithms to print on the screen our data from board , transferred via WiFi. For this kind of process, we’ll need a TCP (Transmission Control Protocol) based on a client (board) and a server (notebook).

As for the WEB part , we’ll show current position and direction vector requested by board. It must be specified , of course , that those coordinates are derived from the real ones so that it won’t brake some state rules for position intimacy.

# Instructions for Use

In terms of usage, the devices requires in the initial phase:

* a power supply
* Internet connection ( for the server )
* a WiFi network card;
* JAVA 7

The first step consists in initializing the necessary programs in order to be able to work with the board : Vivado ( for the hardware part, FPGA ), SDK ( for the board’s programming, C ++), Eclipse (for the server/ data visualization).

Turn on Vivado and then select the project. After turning it on, click File -> Launch SDK. This will launch an SDK instance. This is where we initially program the board by pressing the Program FPGA button ( to note that the board has to be connected through USB firstly) and select “Program”. This will transfer the .bit file and configure the FPGA.

The next step consists in programming the board using a C++ code. Select the project ( right click ) -> Run as -> Launch on Hardware (System debugger). After having programmed it, you will have 10 seconds ( 5 seconds loading time + 5 seconds initializing time) to turn on the server.

You have to follow some steps in order to run a server instance. First of all, turn on the HotSpot Mobile protocol and configure it ( name: “stefan1234”, password: “test1234”). The next step is turning on the server in Eclipse. Launch Eclipse, upload the project and click “run”.

Once these steps have been followed, a connection has to get established between the server and the client ( the laptop and the board ) so that we could access the controls in the application. To note that the application uses JAVA, therefore it is necessary that JAVA runtime should be installed.

# Used Technologies and Algorithms

From the software point of view, as far as the server part is concerned, in order to provide a both fast and stable connection, we will use the security data transmission protocol TCP ( Transmission Control Protocol). This ensures the fluency, the safety, the accuracy and the validity of the data in order to monitor in real time the data transmission ( Real Time - Virtual Time < 10ms).

As for the application, we’ll be using a user-friendly program , created with OpenGL library , LWJGL ([Lightweight Java Game Library](https://www.lwjgl.org)) which ensures data synchronization with user commands. In this way we check 2 of the main targets: time and reliability.

Now , pointing to the hardware , we’ll meet one of the biggest program language , C++ used with some diverse and complex libraries from Xilinx , more exactly , PMods . Implementing and programming PMod circuits sets an important stage in device’s functionality (data request) and it works with the main program written in C++ , translated then in binary code for processor(ARM). This process requires an well written and correct code , fluent(to ensure data transmission in real time), certain written code (to ensure data integrity) and of course Interrupt system(which sets up components priority).

As for used algorithms , we’ll mention: Vector direction algorithm. This kind of algorithm use X axis (initially used as speed vector) and Y axis (as angular speed vector). ,,X’’ will be added to the GPS coordinate ,,x’’ and with some complex math operations , will also set ,,y’’ coordinate based on ,,Y’’ axis. In the end , we’ll reach out 2 results ,,x’’ and ,,y’’ , the new coordinates for the actual positions. To ensure correctly data working of the device , we’ll initialize the ,,path’’ with last 3 couple of points (coordinates) received from GPS.

Frequently Asked Questions (FAQ)

Q:**Where would I need this device?**

A:As we have not mentioned a role for this device , it might have multiple usages. For example , you can use it as car tracking , you would know where is your car every second. This might be a very useful anti-stealth tool.

Q:**Can I use this device to monitor my steps?**

A:Yes you can. Unfortunately we haven’t made a support or algorithm that monitor your steps such as smartphones or smartwatches uses. But , you can do some math operations from our logging of movements and you can easily measure your steps.

Q:**What do I need to do to work with this device?**

A:Umm , actually nothing complicated. This device was developed to be friendly with users and help them make it work even if u lost/don’t know the device’s guide. Just plug in the USB , program device and server with 2 clicks and then enjoy your results. Simple isn’t it?

Q:**Is there any source code stealth/used from external sites?**

A:No , all we implemented is originally coded and developed by Stefan Voica with assistance from Alexandra Stanciu.. Of course special thanks to Digilent for support and Xilinix!

Q:**Is there any additional software/hardware required to run this device?**

A:Our device runs on every platform that runs Java so we consider it universal application compatibility. You just need java to run our software and of course the device to be turned on.

Example usage

As we decided to write also an example of how our product work, we might want to mention some essential steps to be done before you can actually try this usage example.

We will implement a car GPS to track our vehicle’s location. For this we need:

* Car/vehicle
* Server(laptop)
* GPSwA(our device)
* Micro-usb cable

First , we will start with our device. We have to turn on the laptop , plug the usb cable to laptop and device. We will wait for the laptop to recognize the device(usually beeps) and we are ready to go on the next step.

Now that we configured the basic steps , we are ready to program the device. Open vivado , then file->launch SDK ->ok. Wait for vivado to start SDK and load it’s resource and bitstream. When its done , program the fpga first by pressing “program fpga” and then select “program”. The bitstream will be uploaded and we are ready to program the device with the C++ code. Select the project -> run as -> launch on hardware. The program will now upload and will be ready to connect to laptop.

For the third step , start the hotspot with configuration listed on the bottom and start the software. If right configured , it will auto peer and will start the transmission of data from device to server. The process might take up to 1 – 1:30 minutes depending on the wifi card and signal power.

When start streaming the data from a side to another , it will auto log the data to .dta files on the local folder. We implemented offline log to make developers and users use data as statistics or certain usage. The program works synchronized , such as the log is written and the data is shown to user in the same time without any bug.

|  |  |
| --- | --- |
| Network name | stefan1234 |
| Network password | test1234 |