Introduction

Here I would like to share my experience of building an electric skateboard in a step by step fashion. I did this as part of my internship at Siemens. They had developed a new inverter platform called TAPAS based on GaN. You can read more about it here.
The idea was initially a spare time activity but ultimately we ended up spending a lot of time to get the thing to work the way we wanted. A first prototype was developed with a wired switch(i know it is hard to believe , but we had humble beginnings) that ran a BLDC motor at a with a fixed speed controller. The software was based on Instaspin FOC speed control, a product developed by TI.

Components

The components used in the construction of e-skateboard are as follows :
1. TAPAS board from Siemens.
2. LiPo battery x 2 (11.1 V each). This is the one we used.
3. Any well built skateboard with wheels.
4. BLDC motor- Turnigy aerodrive SK3-5055-280KV BLDC motor.
5. cables to interconnect, fuse, fuse holder, mechanical fittings and accessories.
6. Remoteand receiver from Hobby King.

All the source code for the project is available here. The code was tested on CCS v7.1.
We will discuss the mechanical construction of the skateboard first , then the hardware modifications to suit the system and then about the software(based on TI Instaspin-FOC).

Mechanical construction

The skateboard is shown in figure 1 .
After reviewing some of the electric skateboards available online, we decided to go for a single wheel drive as a first try. We took apart one of the rear wheels , drilled four holes to connect a pulley, attached another pulley to the motor shaft and connected both pulleys using a belt (All these can be found generally in DIY kits like this one here or can be bought individually).

Now we have a skateboard with a single wheel belt drive. We built an Aluminium housing considering the dimensions of the skateboard, the ground clearance needed and the components that had to be fitted in(The idea was to include the TAPAS board, the two LiPo batteries and the connecting cables all inside the Aluminium housing and only the three phase cables should come out).
Adjust the pulley wheel assembly till the belt is tight enough to run without slip and the belt does'nt rub on the sides.

Figure 1: Skateboard

(a) Wheel

(b) wheel and motor

Electrical construction

Here we discuss the electrical connections and cabling(the TAPAS board is discussed in a separate section). The electrical connection scheme is as below,

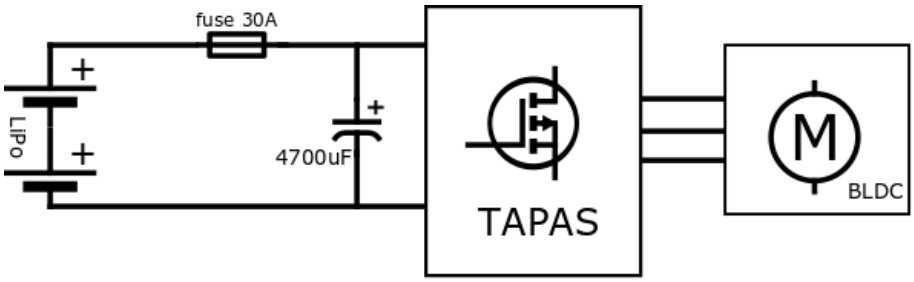

Figure 3: electrical schematic

2.5 sq mm cables , along with suitable connecters are used for the entire system. The two LiPo batteries are connected in series along with a 30A fuse for safety (the fuse also serves as the ON/OFF switch). A 4700 µF capacitor is added to the DC link as an additional support to prevent any voltage dip from affecting the inverter.

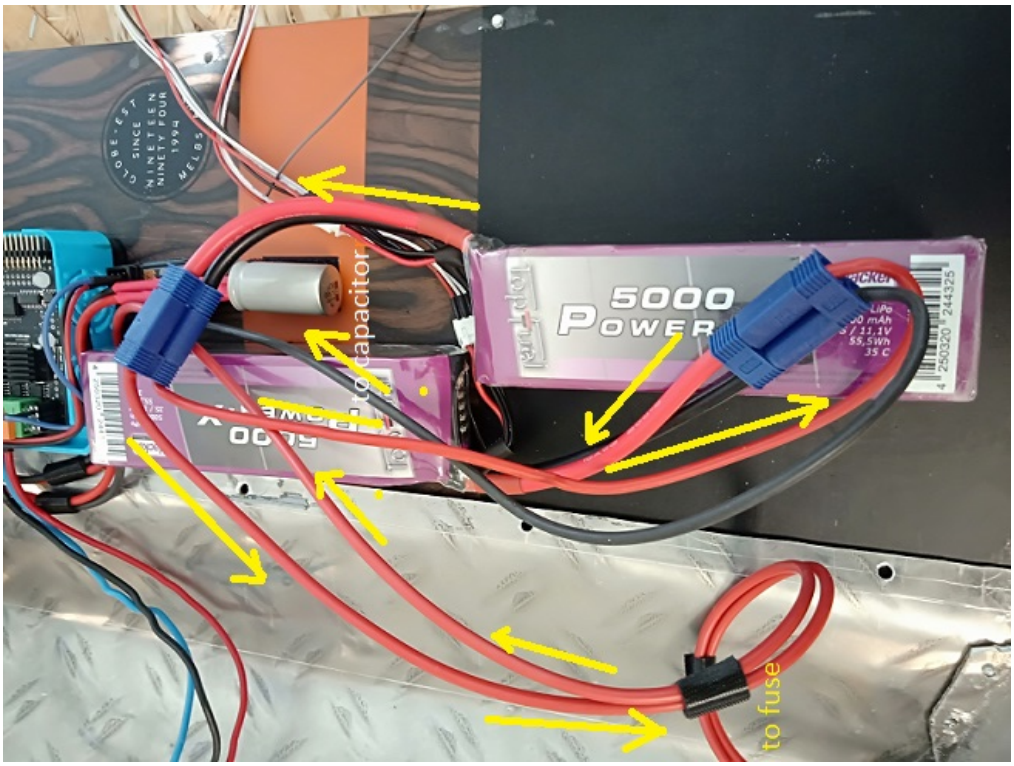

Figure 4: electrical schematic

TAPAS

Hardware

Initially the system was tested without any modifications to the TAPAS board. It was noticed that the motor does not start smoothly from all rotor positions. This phenomenon was particularly noticeable when a significant load torque was applied to the motor. The skateboard vibrated back and forth a bit at start when a heavy person was standing on it. The problem was thought to be due to the starting phase of the sensorless FOC algorithm, but TI Instaspin manual stated that with their FAST estimator motor should pick up rotations very easily even from zero speed when using force angle feature.
This led to further investigations with the board. The PWM frequency was 90KHz which was very close to the cut off frequency of the on board filter. In addition to this the current sensor output to the ADC was very noisy due to 5V to 3.3V level translation. Also the bandwidth of the current sensors was 120KHz.
It was decided to lower the PWM frequency as a way around these problems.The PWM frequency was lowered to 45KHz. This meant that we had to disable the on board LC filters (as the cut off frequency of the LC filter was around 90KHz). So the capacitors connected to each phase output were removed and heat sinks were attached to inductors . This is shown in the figure 5.
Similar heat sinks were also mounted on GaN FETs on bottom side of the PCB for cooling purpose. Also a reasonably big electrolytic capacitor was added to the input to augment the on board DC Link capacitor.

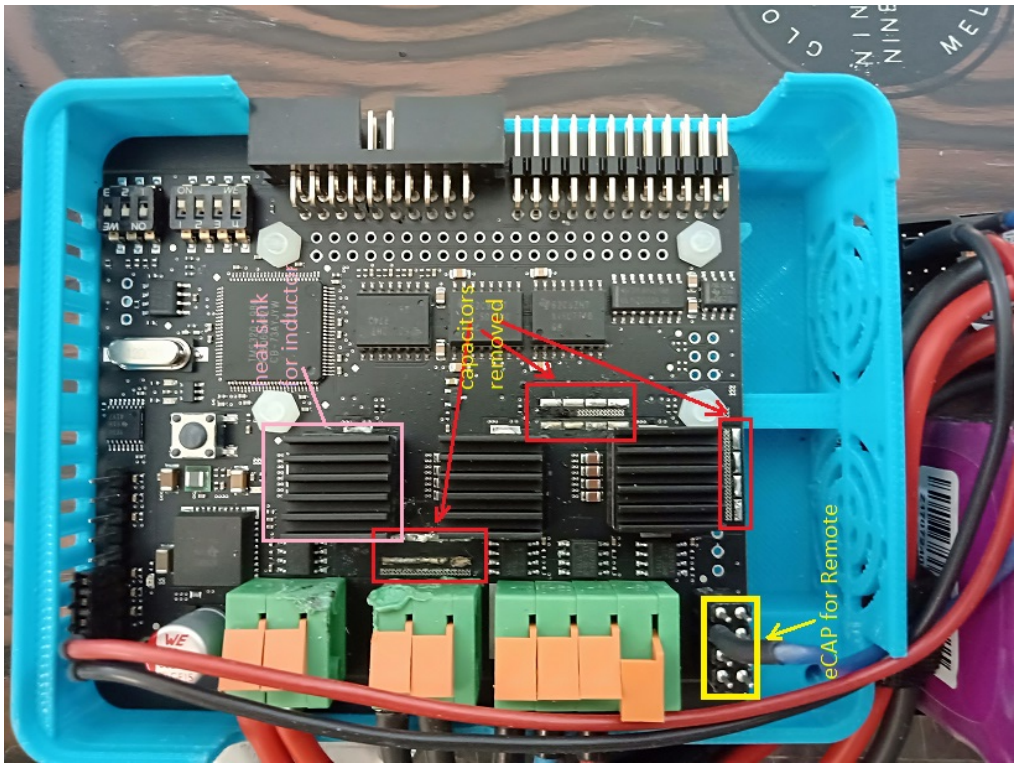

Figure 5: Modfifications to TAPAS board

Software

The software was developed in C, by modifying the project lab05a offered by TI as part of Instaspin FOC suite. Project 05a was chosen as we did not want any problems with the speed controller(making fault identification difficult). The control is now much simpler with two simple PI controllers. CCS is set up as instructed here to be TAPAS compatible. The software architecture is the same as described in the Instaspin lab tutorials- lab 05a.
The motor parameters are defined in user.h. The pole pair number is obtained from the manufacturer(name plate or data sheet). The motor we selected is a Permanent Magnet Brushless DC(BLDC) motor, so it has zero rotor resistance. Rs, Ld,Lq and rated flux are obtained from the FAST estimator. Maximum motor current is set to 30A (select this value based on your motor maximum current given in your data sheet.).The flux estimation frequency was set to 60Hz by trial and error method.

First Run

Proceed further, if you have the skateboard ready, the TAPAS board modified and electrical wirings completed Now lets look into MOTOR_Vars_INIT located in main.h file.The enableSys, enableForceAngle and enableUserParam Flags are set to true. The Run_Identify is automatically enabled after a finite time (you can see this in projlab05a.c) .EnableSys flag enables the entire system, force angle gives an angle reference during zero speed start and enableUserParam takes the motor parameters from user.h if the motor was previously identified. Refer TI Instaspin user manuals in case of any confusion with variables.
If you are running the code for the first time or if your motor is different from the one we use, you have to run the identification routines. In this case set enableUserParam to false. Go online, then check the state of gMotorVars. Set Run_Identify flag to false , set enableRsCalc and enableOffsetCalc to true and then re-enable Run_Identify. If everything works fine you can see your motor rotating, estimator estimating Rs and then Ld, Lq and flux values. If everything is completed successfully, motor identified flag is set to true and you can see values in Rs, Ld, Lq , rated flux and also in Current and voltage offsets. Copy the values of motor parameters, current and voltage offsets into user.h before running the motor next time.

The remote receiver is connected to eCAP via pin 6 on JP13. GPIO19 is configured as eCAP in hal.c. A servo control routine is written in servoCTRL.c , servoCTRL.h header files that reads the remote press as a pulse of width varying between 1ms(0%) and 2ms(100%). Now to run the motor, you recompile the code with the modifications above, download the new executable to DSP, go online and set enableUserParam to true. With this your skateboard is ready to hit the road.
All the changes are commented in the source code. A detailed understanding of the underlying theory can be obtained by going through the InstaSpinlabs and InstaSpin user manuals.

I found these references quite interesting and compelling that I feel obliged to share.
1.Motor Control Class A very lucid style introduction to motor control.
2.VESC . An open source ESC that contains tons of useful info. We have used his solution in our ISR to eliminate the case if the motor doesnt rotate after giving the command for a finite duration of time.

This is just a starting point. Lots of improvements are possible like adding more motors, sending data to a remote device to measure performance and so on.
Happy skateboarding!!