UNIVERSITY OF CALIFORNIA
College of Engineering
Department of Electrical Engineering and Computer Sciences

EECS 249 **A. Sangiovanni-Vincentelli**
Fall 2012 **P. Nuzzo**

Lab 2: Aircraft Electric Power Distribution Control
Due Tuesday, November 6, 2012

Lab 2 focuses on control protocol design using Matlab/Simulink. Check the Matlab/Simulink documentation manual and the several demos as a reference on modeling heterogeneous systems with Simulink. We acknowledge John Finn for helping out with the preparation of this assignment.

The goal of Lab 2 is to design and model the Bus Power Control Unit (BPCU) of an aircraft Electrical Power System (EPS). An overview of aircraft EPS, describing typical system configurations and specifications will be posted on Piazza as a reference. However, for this lab, we focus on a much simpler "plant", represented by the single-line diagram (SLD) in Fig. 1. An SLD is a simplified notation to represent three-phase systems.

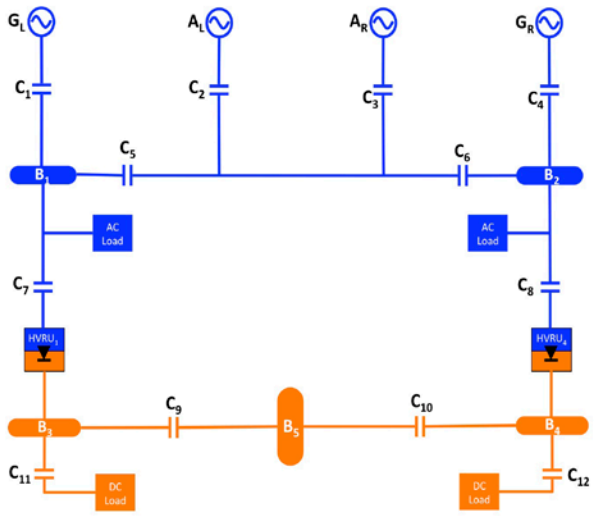

Fig. 1: A simplified single-line diagram with high-voltage AC and DC components.

The plant includes a set of generators (G_L and G_R), auxiliary power units (A_L, A_R), buses ($B_{1\text{-}4}$), rectifier units ($HVRU_1$, $HVRU_4$), AC and DC loads interconnected through power switches, called contactors ($C_{1\text{-}12}$). Given the reference SLD, the goal is to design the BPCU to drive the contactors while guaranteeing that AC and DC loads are always powered, even in the presence of faults in the power sources. To be more specific, we express the BPCU requirements in terms of *assumptions* of the BPCU on its environment (including the "plant") and *guarantees* that the BPCU must offer. This terminology will already introduce you to some concepts from contract-based design, which we are going to discuss later in class. Moreover, such a reasoning scheme can make it easier for you to check whether your final implementation satisfies the requirements.

The BPCU makes the following assumptions on its environment:

A1) The airplane is always operating in the cruising (flying) phase of its mission;

A2) At least one power source is always "healthy" (i.e. it is operational and can be inserted into the network to deliver power);
A3) Failures can only affect the power sources; once a power source becomes "unhealthy" (i.e. it is not operational and cannot be inserted into the network to deliver power), it will never return to be "healthy" (e.g., turned back on) during the cruising phase of the mission;
A4) An AC bus is correctly powered if the root-mean-square (RMS) voltage at its loads is between 110 V and 120 V and the frequency is 400 Hz.

Under the above assumptions, the BPCU offers the following guarantees:

G1) At start-up all the power source contactors are "open";
G2) In normal conditions (i.e. no faults or failures in the system) G_L and G_R are "on" and provide power for the left side and the right side of the system, respectively; auxiliary power units are "off"; C_9 and C_{10} are open ("off");
G3) No AC bus is powered by more than one power source at the same time, i.e. AC power sources can never be paralleled;
G4) It never happens that both the APUs are inserted into the network at the same time;
G5) AC buses cannot be unpowered for more than a well-defined length of time;
G6) DC buses must always stay powered, at least in a "reduced performance" mode, which occurs when only one HVRU is used;
G7) The left AC bus B_1 must always be powered from the first available source from the ordered list (G_L, A_L, A_R, G_R);
G8) The right AC bus B_2 must always be powered from the first available source from the ordered list (G_R, A_R, A_L, G_L).

To simulate your controller, a continuous-time model of the "plant", `Lab2_AircraftEPS.mdl`, has been implemented in Simulink, by exploiting the `SimPowerSystems` extension, and will be posted on Piazza for your convenience. As an example, the continuous-time model for the generator consists of a mechanical engine (turbine), a three-phase synchronous generator, and a local Generator Control Unit (GCU), driving the field voltage of the generator to provide a stable output voltage across a range of possible loads. Even if the model is not using a conservative electrical network model of computation, it can still capture a few dynamic behaviors that are relevant to your design, including timing behavior, current and voltage levels at different loads. The model can be discretized to speed up simulations, it can run in "normal" or "accelerator" mode, and it can seamlessly interface with discrete control algorithms developed, for instance, in `StateFlow`.

You should create a simulation set-up that shows the BPCU working in the presence of faults in the generators, and email the project files to nuzzo@eecs.berkeley.edu. In addition, you should submit a report addressing the following aspects:

a) Brief description of your design: How did you model the BPCU? How did you implement it in Simulink?
b) How did you decide on the set of failure events to use in order to validate your design and show that it actually satisfies the requirements?
c) How was your learning experience with Simulink? What aspects did you like or dislike most?

Finally, here are a few considerations to keep in mind while running the Simulink model:

- The revolutions-per-minute (rpm) input profile for the generators' turbines is only defined on the time interval [0, 10] seconds. Therefore, make sure you extend the generator profile if you need to run longer simulations.
- Depending on the input profile for the generators' turbines, some start-up time might be needed for the generators to reach the desired frequency (400 Hz), which is why we require that all the power source contactors are "open" at start-up. The behavior of your BPCU should be verified only after all generators have reached their steady state behavior (i.e. their output voltages and frequencies are within the desired ranges).
- In addition to the three-phase power outputs, each generator has an input and an output port. The input port allows you to inject faults. When the input is equal to one, as shown in Fig. 2a, the generator is healthy; when the input is zero, the generator fails and can no longer be used.

Therefore, you can replace the constant input block shown in the screenshot in Fig. 2a with a step or pulse signal, based on your needs. The output ports (LGEN, APU1, APU2 and RGEN), as shown in Fig. 2b, can be used by the BPCU to measure the generator output voltages (since the BPCU has no access to the generators' input signals).

- To generate events for your BPCU, you will need to monitor each generator's output voltage and make sure that the RMS voltage is between *V_min* and *V_max*. As an example, you can implement such an RMS "sensor" out of Simulink standard library blocks, as shown in Fig. 3. *Gen_status* is one if and only if the signal RMS voltage is in the desired range. Also, the RMS block in Fig. 3 requires a frequency parameter, which should be set to 400 Hz.
- To visualize simulation results, each load block includes an oscilloscope, plotting the voltage across it and the current through it. The generator frequency can be read from the output *Out1* (port 2) of the synchronous generator sub-block inside each generator block. Finally, the "status" of each contactor gets damped to a sink block directly connected to the control input of each contactor.

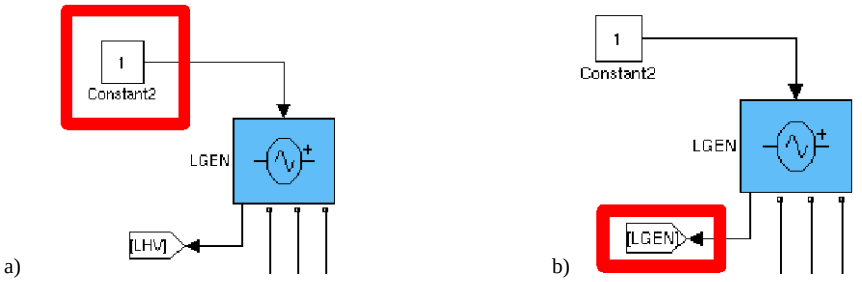

Fig. 2: (a) Injecting generator faults and (b) sensing the generator's status.

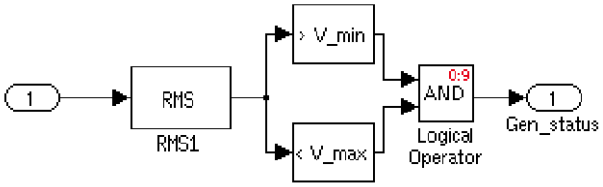

Fig. 3: Simulink block to monitor the generator's RMS voltage and detect violations of the desired range.