Senior Design

ENG EC 464

To: Future Senior Design Team, Users

From: Power Pooches, Team 14

Date: 5/3/2015

Subject: Software Documentation

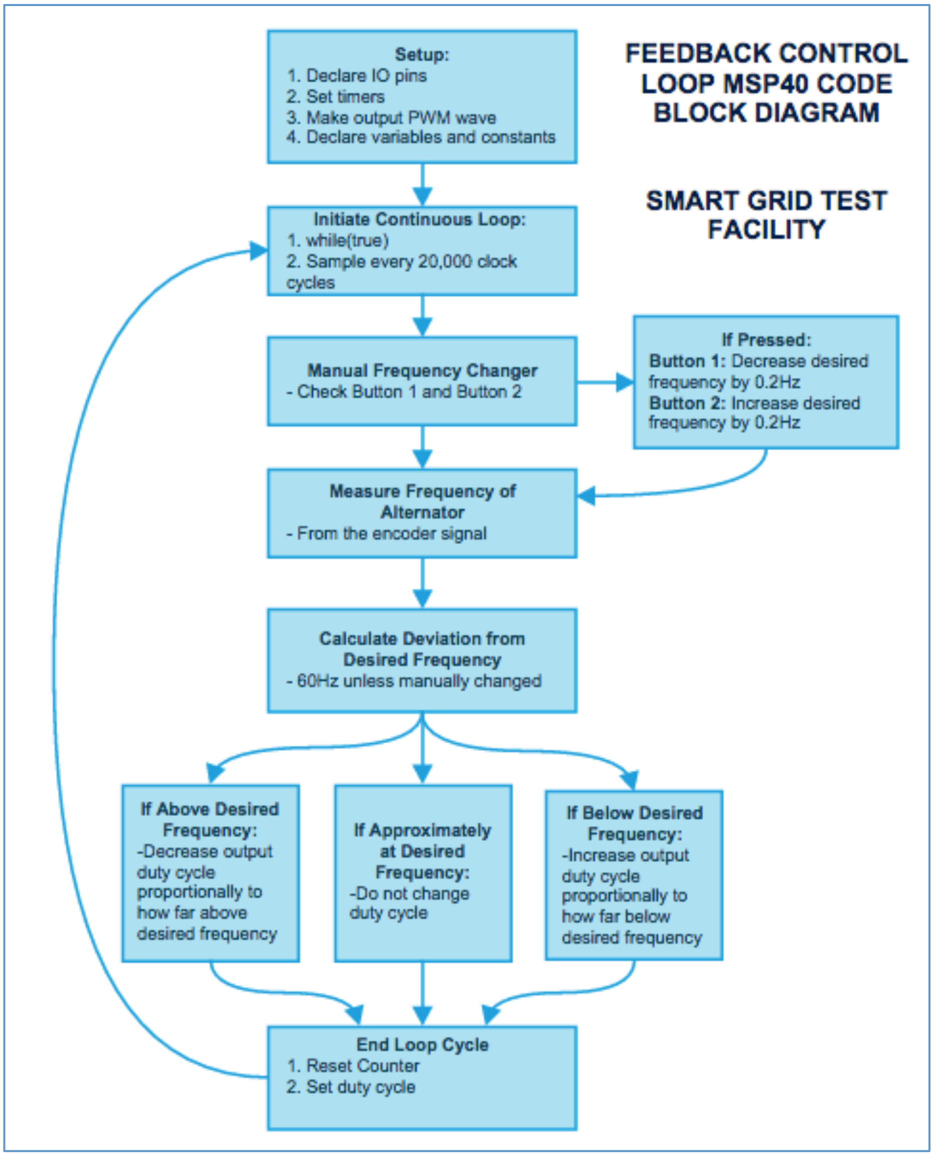
1.0 FEEDBACK CONTROL LOOP: MSP430

1.1 The code for both generators is in this USB drive. The code is well commented and is written in C and also some Arduino code, which is similar to C/C++. Because the microcontrollers were programmed on an Apple computer, Code Composer was not used. Instead, Energia was used, which can be downloaded simply from the Internet (available for Mac, Windows, Linux at http://energia.nu/download/). The environment looks very similar to the Arduino coding environment. The version used during the creation of this project was V.0101E0014.

1.2 The generators have their own (A and B) code, and the programs are not identical. The code takes into account experimental factors that depend on the specific motor and alternator, as noted in the programs.

1.3 To edit the code, save a copy of the original code and make a duplicate for edits. Compile the code using the check button at the top ("verify"). Resolve any errors and upload to the MSP430 using the "upload button," shaped like a rightward arrow.

1.4 Make sure to select the exact board under Tools >> Board (the design team used an MSP430 G2553).


FEEDBACK CONTROL LOOP MSP40 CODE BLOCK DIAGRAM
SMART GRID TEST FACILITY
Setup:
1. Declare IO pins
2. Set timers
3. Make output PWM wave
4. Declare variables and constants
Initiate Continuous Loop:
1. while(true)
2. Sample every 20,000 clock cycles
Manual Frequency Changer
- Check Button 1 and Button 2
If Pressed:
Button 1: Decrease desired frequency by 0.2Hz
Button 2: Increase desired frequency by 0.2Hz
Measure Frequency of Alternator
- From the encoder signal
Calculate Deviation from Desired Frequency
- 60Hz unless manually changed
If Above Desired Frequency:
-Decrease output duty cycle proportionally to how far above desired frequency
If Approximately at Desired Frequency:
-Do not change duty cycle
If Below Desired Frequency:
-Increase output duty cycle proportionally to how far below desired frequency
End Loop Cycle
1. Reset Counter
2. Set duty cycle

The diagram above shows the flow of information occurring in the MSP430 code. This applies to the MSP430 used on both feedback control loops.

2.0 DATA ACQUSITION

2.1 Setup: Setup of the data acquisition system is comprised of three parts: installation of support software, installation of MATLAB programming, and hardware setup.

2.1.1 To get started, log onto a windows computer with .NET framework installed and install the Labjack driver available at **labjack.com/support/u3** . **(specific information regarding driver installation)**

2.1.2 Connect the USB cable, and windows should prompt with “Found New Hardware” and the Found New Hardware Wizard will open. Here you have the option of which items you specifically wish to install, however it is recommended to accept all defaults. At a minimum install LJControlPanel.

2.1.3 Next, check to make sure the support software is installed by clicking on the Windows Start menu, going to the LabJack group, and running LJControlPanel. Once the program launches, click the **Find Devices** button, and if the software is installed correctly, and entry should appear under U3 as **USB-1**.

2.1.4 Once the LABJACK driver and software is installed, unzip the 'runlabjack.zip' file in a directory of choice. If you prefer it could be in the default directory Documents/MATLAB, however a new directory is recommended. Once the unzipped folder titled 'runlabjack' is in the directory, set the path to the folder and its contents. To set the path right click on the 'runlabjack' folder in the MATLAB file navigator and select 'add to path' then 'selected folders and subfolders'.

2.2 The Code: The main reason for utilizing MATLAB is engineering students at Boston University have extensive experience with it. This means future BU students/project groups can incorporate their own MATLAB based control systems into the system and utilize the LABJACK wherever necessary. With our project we've included basic example code that calculates data points and plots V-I waveforms. This is what the 'runlabjack' folder contains.

2.2.1 There are three functions within the folder: configurelabjack.m, GetData.m, and AnalyzeData.m. In addition, basic configuration is done via the main script runlabjack.m.

configurelabjack.m: takes the configuration decimal (refer to users manual for more info) and the timing parameter and configures the device using LABJACK corporation provided functions.

GetData.m: has DAQ unit collect analog data points from the sensor board and stores it in a .NET-based array.

AnalyzeData.m: converts the analog values collected by the LABJACK back to their actual values via mathematical operations and plots the waveforms. Then calculates VRMS, IRMS, V-I phase, and power factor. In addition, there is the option of FFT filtering if desired.

2.3 Abnormal Operation/Things for consideration: It should be noted the current sensor (ACS712) will have a noise of about +/- 4 milliAmps. If you run the software and the output plot for current seems very noisy, look at the axis for values. If the axis is in the range of +/- 20 milliAmps or so (.02 Amps), what is plotted is the extensive noise the ACS712 puts out.

2.3.1 In addition, the LABJACK does not like configuration decimals that do not include the first two pins FIO0 and FIO1. If you come across an error regarding line 12 in GetData.m, it is usually because FIO0 and FIO1 are not included.

2.3.2 Lastly, if you see saturated waveforms in the plots but the maximum voltage values are not around +/- 12 volts, check the ground connections of the hardware. Most likely, one or more are loose/not connected.

2.4 Future Capabilities: It should be noted that because we've provided a MATLAB code does not mean you have to use either the code or MATLAB with the system. Should you wish, LABJACK provides support for other languages including C++, Labview, and Python. This makes it very flexible for future projects. If you wish to program in any of the languages including MATLAB, LABJACK offers great support on their website in addition to coding examples. Have Fun!