All pertinent documents will be added to GitHub under the alias “LegallyKF5RCL”

The Initial goal is to get the chip modules up and working, then progress can be made developing software code. The follow modules are needed: SPI, UART, ADC.

Module operations needed:

SPI – Chip Select, Data Out, Clock.

UART – TX and RX lines.

ADC – 2 channels, 10 bit conversions.

Module hardware specifications

SPI

Chip select: 1 GPIO.

Data out: SDO1 pin.

Clock: SCK1 pin.

UART

TX: 1 GPIO

RX: 1 GPIO

ADC

2 Channels: 2 GPIO’s

First, I need to get UART up and working first. This will allow for robust debugging, and will be critical at early stages to verify the working status of other modules. To achieve this I will first get UART TX to operate properly. This way, at the least the chip can talk to me. Next RX will give me the ability to talk to the chip. To complete this task item and for verification sake, the chip will be programmed to echo any message that I send to the chip; performed in a debugging mode with warning messages included.

The final stage will be a filter in order to remove the sharp steps from changing the DAC. Without trying to teach myself how to make a feasible high-pass filter, I’m going to try a parallel RC circuit in simulation first. That was a bad idea. Instead a simulation of a series RC circuit will be used.

Now I will explain the logic behind using a simple RC circuit to filter the output. The capacitance of the capacitor should be sufficiently large to smooth the sharp jumps when the DAC changes value. This can be accomplished by making the output of the function generator to be the voltage across the capacitor. However, the capacitor cannot be too large, as this would cause the capacitor to discharge or charge slower than the DAC is attempting to pull the voltage. The dual issue is that if the capacitor is too small the capacitor will charge and discharge too fast and the smoothing will disappear at low frequencies. Since the system must not cause slew rate problems, my design will focus on determining the fastest rate at which the ideal signal changes, and match it with the maximum rate of charging in the capacitor.

Errors

There is a bizarre error in my Debugging routines. When I write to the SPI1BUF, UART prints out 0xFF. As of this commit, this is no longer a problem.

An error was occurring because I was trying to write to UART\_TX buffer too fast after setting up the module. After Inserting a delay of 0xFFFF \* 2 instruction cycles, it fixed itself.