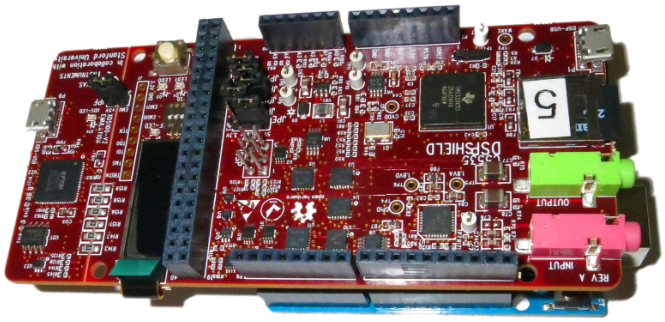
DSP Shield GPIO & I2C Communication

Application Note DSP-01

William J. Esposito 10/2014

Notes on operation and interaction with the DSP Shield GPIO and other I2C peripherals



It is important to note that most of the peripherals on the DSP shield communicate via the I2C bus, including the display, audio codec (used for configuration only, audio data is transmitted via the I2S bus over DMA), and most importantly, the Arduino GPIO headers as well as LEDs 0-2 and the user input dipswitches.

This means that GPIO reads and writes pass across the I2C bus and take a few hundred microseconds (and if misconfigured, can take ten milliseconds or longer). This also means that doing a GPIO write or read in an interrupt can potentially collide with an I2C event in the main program loop, and that a failure may occur even when an I2C transaction is interrupted for an extended period of time without an intervening I2C event.

The solution is to simply exercise care when communicating with the GPIO pins. Careful attention should when accessing GPIO from one location and using the I2C bus in an interrupt that may occur asynchronously.

When a GPIO event fails, it may cause future reads to fail as well. When this is the case, the digitalWrite or digitalRead function will return a “2” indicating an error. When that happens, the easiest thing to do is simply handle the error by resetting the I2C bus (controlled by the Wire library) with a call to:

```
Wire.endTransmission();
Wire.begin();
```

It is possible to integrate this into the digitalRead/Write functions, but has not yet been implemented.

With this in mind, it is quite possible to keep GPIO on the DSP Shield rock-solid.