Adam Berry-Huff

Hardware Description

Even though I may not need to draw a full 2.5 amps for all my circuit components, I chose the 2.5 amp wall wart (J1) just to ensure that I could supply my circuit with as much current as it would need. The MSP430G2553 MCU (U1) in active mode and running at 1 MHz only draws 230 uA of current. When the RN-42 Class 2 Bluetooth Module (U2) is enabled, connected, and transferring data it draws 50 mA maximum current. The MC33926 throttle control H-bridge motor driver (U3) draws 20 mA of current when enabled and in use. The LD1117 linear voltage regulator (U4) draws 10 mA maximum current. The PA-15-4-11 linear actuator (M1) is rated to draw 9 amps of current at full load. Since this motor can uphold up to 30 pounds of force, this is a massive over-estimate for how much current this motor will draw. The motor will be horizontal so it will not even have the full force of gravity working against it. After some testing, I determined that this motor runs at the rate I would like it to when drawing about 1.75 amps of current. Factoring in a 20 mA current draw for each of the two LEDs (LED1 and LED2), the worst case scenario current draw adds up to grand total of 1.87 amps of current. By using this 2.5 amp wall wart, this leaves a 630 mA of leeway current even when the circuit is running at maximum power.

The B005M5OCU2 wall wart (J1) is also rated at 12 volts DC. This value was chosen because the linear actuator is rated to run at 12 volts DC. A LD1117 linear regulator (U4) will then be used to regulate the voltage down to the logic level necessary for powering both the RN-42 Bluetooth Module (U2) and the MSP430G2553 MCU (U1).

As previously mentioned, the RN-42 Class 2 Bluetooth Module (U2) is powered by the output of the LD1117 linear voltage regulator (U4). This +3.3 volt input is connected to the VDD pin of the Bluetooth module. The optional analog inputs (AIO0 and AIO1) and the pulse code modulation pins (PCM\_IN, PCM\_OUT, PCM\_CLK, and PCM\_SYNC) are not connected on this chip because I will not be transferring data using analog signals. Most of the GPIO pins are not being used as well and are not connected as specified in the datasheet. I chose to communicate via UART communication with this Bluetooth Module so the UART\_RX and UART\_TX are connected to the MSP430G2553 MCU (U1) UARTRXD and UARTTXD pins (pins P1.1 and P1.2 respectively). The UART\_RTS and UART\_CTS pins on the Bluetooth module are connected to ground to enable the transmitter of the Bluetooth module. The SPI pins (SPI\_MISO, SPI\_CSB, SPI\_MOSI, and SPI\_CLK) and USB pins (USB\_D- and USB\_D+) are not being used either since I am using UART communication. The DATA pin (GPIO5) is connected to an LED and a current-limiting resistor in series. The LED blinks when data is being transmitted from the Bluetooth module or received by the Bluetooth module. The CONNECT pin (GPIO2) is connected to an LED and a current-limiting resistor in series. This LED is illuminated when the device is paired with another Bluetooth device and is off when no devices are paired with the Bluetooth device. The BAUD pin is configured to set the baud rate to 9600 when pulled high and 115000 baud when pulled low. In this case, the device is configured to 9600 baud. Finally, the DISCOVER pin will auto-discover and connect to devices it has been previously paired with when they are in range when pulled high. For this particular application, this is the desired effect.

The MC33926 throttle control H-bridge DC motor driver (U3) is also powered by a logic level input. However, due to the charge pump circuitry of the motor driver, the motor input voltage (VPWR) can be regulated down to also power the chip itself. The motor driving voltage (+12V) comes directly from the wall wart. The charge pump capacitor can be anywhere from 30 nF to 100 nF. I chose the recommended datasheet value of 33 nF. I tied the positive output disable pin (D1) to ground so the positive output terminal is never disabled if the chip is not disabled. Since the negative output disable pin is active low, I tied it to the chips enable pin so when the chip is enabled so is the negative output terminal. The enable pin of the motor driver chip is tied to a MSP430G2553 MCU (U1) GPIO pin so the motor driver can be enabled when a software interrupt triggers and disabled when the MCU goes into low power mode. This will bring the power draw down significantly when the system is dormant. The IN1 and IN2 pins control the OUT1 and OUT2 pin polarity respectively. If IN1 is logic level high and IN2 is logic level low, OUT1 is positive and OUT2 is negative. If IN1 is logic level low and IN2 is logic level high, OUT1 is negative and OUT2 is positive. By controlling the polarity of the linear actuator, I can extend and retract it through GPIO pins of the MSP430G2553 MCU (U1). The FB (feedback) pin of the motor driver outputs an analog signal that has a voltage proportional to the current draw of the motor attached to the driver. By connecting this pin to an analog-to-digital convertor through some feedback circuitry you can tell when the linear actuator gets fully retracted or extended because the internal limiter of the actuator cuts off current. When this occurs, you can disable the H-bridge by setting the enable pin to logic level low. This will ensure that, even though the internal limiter stops the actuator from being harmed, the motor and H-bridge will not have unnecessary current draw.

As you can see from the schematic, this circuit is simple, yet functional. Through the use of a wall wart and regulator, I am able to power both my logic level devices and my 12 V motor. A Bluetooth signal can be obtained using the RN-42 Bluetooth module. This signal can trigger a software interrupt that will in turn enable the motor driver and drive the motor into the desired position because going back into low power mode. Keep in mind that the current calculations quoted in this description are worst case calculations and this circuit will typically be drawing way less power than quoted.