**Introduction**

The following describes the hardware of the regenerative motor controller. The hardware flow diagram (figure 1) will be used as a starting point for the description. Throughout this description I will be addressing the functional blocks shown inside the motor controller block.

![Hardware Flow Diagram](image)

**Figure 1. Hardware Flow Diagram**

The motor controller is a microcontroller based system that uses feedback to control a three-phase BLDC motor. A three-phase BLDC motor consists of a permanent magnet rotor and a stator split into three phases. Three-phase motors are used because they are efficient, durable and require little maintenance. The design cost is a larger and more complicated motor controller.
Hardware Description

In the introduction you saw that there are design problems associated with choosing a BLDC motor, the main problem is that the motor must be driven by three half-bridges. Each bridge has a high and low-side FET that requires its own pulse width modulated (PWM) signal. Each PWM signal must be buffered, amplified and isolated before it drives the gates of the FETs. There are six identical gate drive sections on the controller schematic, one for each FET. The FETs are International Rectifier IRL540N MOSFETs, they have an $I_{ds}$ rating of 36A, $V_{DSS}$ of 100V and an $R_{ds(on)}$ of 0.044Ω. The FETs are mounted on the heat sink shown in figure 3 so that they maintain stable operating characteristics even when driving a loaded motor.
The drive circuit of a particular FET includes the following: a MAX627 (U7, U8 and U9), a 0.33µF coupling capacitor, a 270kΩ resistor a gate drive transformer, a 0.22µF coupling capacitor, a 1N4004 diode, a 150kΩ resistor in and a 7Ω resistor. The MAX627 FET driver can source or sink 1.5 Amps, greater than the 490 mA requirement. The 0.33µF coupling capacitor is used to ac couple the gate drive signal and effectively resets the core of the transformer each cycle, which prevents saturation. The 270kΩ is used because the MAX627 doesn’t like to see inductive loads, specifically the transformer. On the other side of the transformer there is another coupling capacitor, this time to ensure that the gate gets a little bit of negative drive to help turn it off faster and prevent turn on due to low-side switching noise. The resistor (R18-20, R30-32) in shunt allows voltage to be developed at the gate of FET while the diode prevents negative drive of more than V_{forward}. The 7Ω resistor in

Figure 3: Heat Sink
Hardware Description

series is used to prevent oscillations by damping the response of the gate drive network.

Notice that the MAX627 is supplied by 14V; this because the FET has a lower ‘on’ resistance ($R_{ds(on)}$) as the gate to source voltage ($V_{GS}$) is increased. Consequently a power supply that can deliver 14V at 1A is necessary to run 2 gate-drive circuits at any given moment. In addition a 5V signal is needed to provide the microcontroller, the CAN and the instrumentation amplifier with safe power. The 5V will be provided by an LM7805 1A linear regulator. To accomplish this, two LM2576 (U2, U3) buck converters are applied, one with 8V output for the 5V regulator and the other for 14V output for the gate drive. The input of each converter is the 36V battery and the output of the converter is the input voltage switched followed by a catch diode and an inductor for current storage and a filter capacitor.

To determine how the motor is to be commutated the motor has built-in Hall-Effect sensors that encode the rotary position of the motor. These sensors are read by the input capture port on the microcontroller, the motor position can be decoded and the motor speed can be determined as well. In addition to position and speed feedback, the motor current is measured to protect the motor in case of a stall or to slow down the motor because of over current. The voltage across a 0.01 ohm resistor is proportional to the current in the path from the positive battery terminal to the drain of the high-side FET's. This voltage is measured using an instrumentation amplifier with a gain of 18 set by R10 and the analog to digital (ATD) port of the microcontroller. Software will be used to calibrate the current sense reading.

Since the controller uses batteries it has to be conscious of the battery voltage to ensure that they are not overcharged in regeneration mode or undercharged in drive mode. A simple voltage divider using R1
and R2 is used to sense the battery voltage level and software will be used to calibrate the battery sense reading.

**Microcontroller Interface**

The MC9S12D64 (D64) has resources that are needed to complete the motor controller design. The D64 resources that I will be using in the motor controller are:

- Input capture port (IC0-2)
- Pulse-width modulator port (PWM0-5)
- Analog to digital converter (AN0, AN1)
- General purpose I/O (PJ6, PJ7, PT3-5)
- Motorola scalable controller area network (MSCAN)

The general purpose I/O will run the user interface and interface with some of the power stage equipment. PT3-5 will be outputs and will turn the user-interface LED’s on and off according to the state that the controller is in: ON, DRIVE or REGEN. PJ7 is configured as an input to watch the on/off switch, when the switch goes off, PJ7 is tied to the 5V line and a safe shutdown process will be initiated. PJ6 is configured as an output and controls the 14V power supply line by pulling the enable pin on the LM2576 low, it is named 14V_OFF and is therefore active high.

The CAN interface consists of an automotive grade transceiver PCA82C251, a 120Ω termination resistor and an RJ-11 jack. The CAN interface will be used to communicate specifically with the Bike Computer project designed by Phillipe Ng, but more generally a device can connect to the motor controller is it meets the following criteria:

- Runs on 5V with a max current of 200 mA or has its own power supply.
- Has an RJ-11 connection.
- Has a 120 ohm termination resistor.
Hardware Description

- Communicates using the same format for messages as the Controller.

The oscillator circuit will supply the D64 with an 16MHz clock that will be multiplied by the phase-lock loop circuit to achieve 24MHz. The D64 has 64kbytes of Flash ROM and 4kbytes of RAM. The RAM will be used by the Real-Time Operating System (RTOS) for storage of temporary variables created by the motor control software and by the CAN software. The ROM will hold the program code.