Miquel Rudé Moreno

Design and realization of a testbench for high dynamic performance permanent magnet synchronous motors

Master’s thesis

Automatic control laboratory

Swiss Federal Institute of Technology (ETH) Zurich

Spring semester 2010/2011

Supervisors

Prof. Dr. Manfred Morari

Dr. Sebastien Mariethoz
Contents

Introduction 2

1. System overview 3

2. Hardware design 5
   2.1. Board overview 5
   2.2. PCB modules 7
   2.3. Assembly, soldering and test 17
   2.4. PCB corrections 19

3. Load design 20

4. Control system 21
   4.1. Motor control routine 21
   4.2. FPGA design 22

5. Tests 28

6. Used software 32

Conclusion 33

Outlook 34

Nomenclature 35

Bibliography 36

Appendix 37
Introduction

Nowadays applications in which the control of the torque, the speed or the position of an electrical motor working at high speed is needed are very common. In this sense, this project has as its main goal the development and realization of a testbench for a high dynamic performance permanent magnet synchronous motor.

As a main task, this includes designing and testing all the hardware and software needed to be able to implement and test different controllers for the motor.

In a first part the remaining hardware to control the motor will be designed and assembled. This hardware will be a PCB including two inverters, current sensors and AD converters to obtain digital measurements of the position and the currents that are required to execute the control algorithm. As the available motor has no load and is not fixed, a second part of the project will consist in choosing a suitable load and designing a mechanical system to couple it to the motor and fix both. Finally an available board containing an FPGA and a DSP will be used to implement the software routines. The FPGA will be used to drive the AD’s and the RD converters and also to interface the PCB with the DSP. The DSP will receive the digital measurements through the FPGA and will use them to execute a control algorithm that will generate the PWM signals. Finally these signals will be sent back to the PCB inverters through the FPGA.

![Figure 0.1: PMSM and load](image)
1. System overview

In this chapter there will be given an overview of the required system to control the motor and the elements that it has, giving also some details of their characteristics.

A main schematic of the system and their elements is given below:

\[ \text{Figure 1.1: Motor control system overview} \]

As it can be seen in the schematic above the main element in the system is the PMSM, [1]. This is a 4 pole 3 phase motor working with a Y connection. It has a rated output of 5 W and a rated speed of 8000 min\(^{-1}\). This motor also has a resolver, [2], that provides 2 analog signals that will be used to obtain a digital measurement of the position. Coupled to the PMSM there is another brushless motor acting as a load (see chapter 3 for more details). In order to obtain the required digital data a set of 6 AD converters will be used. Their main function will be sampling the analog signal of the currents flowing through each phase of the motors. Moreover, to sample the position a special IC will be used, which will generate the appropriate signals to excite the resolver and will convert its outputs to a digital format. These measurements will be used then to run a control algorithm that will calculate the duty cycles and generate the signals of the PWM needed to drive each motor. Finally these signals will be applied to two 3-phase inverters connected to the motors.

Some parts of the system are not available and must be designed. The inverters, the RD and the set of AD’s will be placed in a PCB that is a modification of an existing board designed in another project. All the details of this design are explained in the
second chapter. Also the load must be selected and a mechanical system designed. This step is explained in the third chapter. Finally the control system is already available. For this, a DSP and an FPGA will be used and all the details about the software routines implemented there are explained in the fourth chapter.
2. Hardware design

In order to control the PMSM it is necessary to design a PCB containing the elements of the system that are not available. In this chapter it will be seen all the design process of the PCB, starting with an overview of its parts, then a detailed description of each one of them and its design and finally viewing how it is assembled, tested and its errors.

2.1. Board overview

In a previous master thesis, [3], a PCB to control the ASV was designed. This project will be modified and adapted to control the targeted PMSM. The final board is a 4 layer PCB with a size of 160x100 mm. The top and the bottom layers are used to carry the routed signals. One of the inner layers is used to carry the ground and the other one to carry the different power supplies. The schematics, the PCB layers printings as well as the list of its components can be found in the appendix A. The board structure is shown in the figure 2.1.

The board has 6 main units: the power supply, the RD converter module, two motor control units and two multiuse units. Moreover, connectors to interface the PCB with the FPGA, the resolver and to drive the motors are also available. Details of the mentioned units as well as the connectors and interfaces will be presented in the next sections.
2. Hardware design

Figure 2.1: PCB overview
2.2. PCB modules

2.2.1 Power supply

The power supply unit provides the +5V and the +3.3 V required to feed the IC’s of the PCB. To obtain these voltages 2 switched step-down voltage regulators, LM2575 3.3 and LM2575 5.0, are used, [4]. Their input ranges from 7 V to 40 V and their maximum output current is 1 A. In this project a +24 V external power supply will be used to feed them because this voltage is also required for some circuitry of the RD converter (see section 2.2.2 for more details). Followed to the regulators there are two LC filters. The second one is optional and used to improve the quality of the output voltage. Moreover, to assure the output stability there is a feedback signal placed between them.

![Power supply module diagram](image)

*Figure 2.2: Power supply module*

The use of the regulators is controlled with the pins nON/nOFF. In this case they are connected to ground, which means they are always on. The interface with the power supply is through the connector T1. It has 5 input and output pins and 5 ground pins. Table 2.1 shows its pin outline. The regulated voltages are accessible at the pins +5 V and +3.3 V. The regulators can be feed either with the +PV or the +BAT pins. The +SERVO pin was used as a power supply for the servos in the previous project and is unused in this one.
2. Hardware design

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+5.0 V</td>
<td>Control voltage in-/output (5V)</td>
</tr>
<tr>
<td>2</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>3</td>
<td>+3.3 V</td>
<td>Control voltage in-/output (3.3 V)</td>
</tr>
<tr>
<td>4</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>5</td>
<td>+BAT</td>
<td>Regulator input</td>
</tr>
<tr>
<td>6</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>7</td>
<td>+PV</td>
<td>Regulator input</td>
</tr>
<tr>
<td>8</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>9</td>
<td>+SERVO</td>
<td>Unused</td>
</tr>
<tr>
<td>10</td>
<td>GND</td>
<td>Ground</td>
</tr>
</tbody>
</table>

*Table 2.1: Power supply interface*

2.2.2 RD converter unit

The main element of this unit is the RD converter Smartcoder AU6802N1, [5]. This IC is thought to be used with a brushless resolver such as the Smartsyn. Its main function is providing the excitation signals for it and converting its outputs to digital data. This data is provided both in a parallel and in a serial format with a selectable resolution of 10/12 bits.

![RD converter unit](image)

*Figure 2.3: Power supply unit*

The chip needs a supply voltage of DC +5V with a tolerance of ±5%, which is compatible with the +5 V output of the step down regulator explained in the previous section. This power line will be used to feed the 2 VCC (analog power supply) and the 2 VDD (digital power supply) pins. In all of them a by-pass capacitor of 100 nF is inserted between the power line and the pin.
The digital signals of the IC use the following logic:

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min.</th>
<th>Max.</th>
</tr>
</thead>
<tbody>
<tr>
<td>High level input voltage</td>
<td>( V_{IH} )</td>
<td>2 V</td>
<td>( V_{DD} )</td>
</tr>
<tr>
<td>Low level input voltage</td>
<td>( V_{IL} )</td>
<td>0 V</td>
<td>0.8 V</td>
</tr>
<tr>
<td>High level output voltage</td>
<td>( V_{OH} )</td>
<td>( V_{DD} - 0.1 ) V</td>
<td>-</td>
</tr>
<tr>
<td>Low level output voltage</td>
<td>( V_{OL} )</td>
<td>-</td>
<td>0.1 V</td>
</tr>
</tbody>
</table>

*Table 2.2: Operating range of digital signals*

The Smartcoder requires a 20 MHz clock signal in the pin CLKin to work properly. Resistors R55, R56 and R57 set the source used to generate this signal. By soldering R55 and R56 a crystal resonator, [6], is used. Soldering R57 allows using an external clock generated in the FPGA.

In order to choose between the different working modes of the Smartcoder there is a set of configuration pins that must be connected either to “H” or “L” logic level. This can be done with the 1 kΩ resistors R41 to R54. MDSEL pin sets the resolution of the digital data, in this case it is connected to “L”, giving a resolution of 12 bits. Pin ACMD sets the acceleration mode. This pin is connected to “H”, which means that the dynamic accuracy in case of high angular acceleration is improved by enlarging 32 times the proportional gain of an internal PI controller. The excitation signals of the resolver must have a frequency of 10 kHz. This can be done by fixing the pins FSEL1 to “L” and FSEL2 to “H”. Moreover, the output mode is set to parallel bus interface mode by setting OUTMD to “H”. Finally, the remaining pins XSEL1 and XSEL2 are disconnected because they are only required to set the number of poles when the output mode is pulse interface mode.

One of the main functions of the chip is generating the two excitation signals (R1 and R2) for the resolver. To do that the Smartcoder generates in the pin RSO a 1 V\( \text{p-p} \) sine wave based on a 31–level interpolation DA. This signal is used as an input to a voltage booster amplifier circuit (see appendix A for details) to get the signals R1 and R2, using two amplification stages. The circuit is supplied with a +24 V power line. The circuit constants are calculated according to the recommendations of the Smartcoder specification:

\[
R_{\text{EXT}} \leq \frac{Z_{RO}}{10} \quad (Z_{RO} \text{ = input impedance of Resolver} = 60 \, \Omega) \rightarrow R_{23}, R_{24} = 5.6 \, \Omega,
\]

\[
R_f \geq 50 \, \text{kΩ} \rightarrow R_{11}, R_{12}, R_{13}, R_{14} = 51 \, \text{kΩ}
\]

\[
C_i \cdot R_i \geq 5 \cdot 10^{-4} \rightarrow C_2, C_3 = 2200 \, \text{pF} ; \quad R_7, R_8, R_9, R_{10} = 270 \, \text{kΩ}
\]

\[
C_f \cdot R_f \leq 5 \cdot 10^{-6} \rightarrow C_4, C_5 = 82 \, \text{pF}
\]

Resistors R15 to R18 and R19 to R22 are not specified in the datasheet. The chosen values for them is 51 kΩ for the first ones and 100 Ω for the second ones. For the components
D1 to D4, the standard diodes N4148 are used, [7]. Npn general-purpose transistors, [8], and their pnp complementaries, [9], are used for Q1, Q2, Q3 and Q4. The four operational amplifiers required for the circuit can be found in the IC AD824AR, [10]. They are feed also with the +24 V power line and have enough bandwidth (2 MHz) for this application. The remaining values of the resistors and capacitors are specified in the Smartcoder datasheet. The excitation signals obtained are shown in figure 2.2.

![Figure 2.4: Resolver excitation signals](image)

As it can be seen these are two sine wave signals with some noise, each one with an amplitude around 1.1 V (the picture shows 0.11 V due to the probe gain) and a frequency of 10 kHz.

In order to improve the quality of the RSO output the RD has two feedback pins (R1E and R2E) that must be connected to the signals R1 and R2 using a small circuit to adjust their phase. This circuit is made with the resistors R37 to R40 with a value of 120 kΩ and the capacitors C12 and C13 with a value of 100 nF.

Finally the last part of the interface with the resolver is the circuit used to read its outputs (signals S1, S2, S3 and S4). This circuit has three main functions. The first one is setting a gain for the signals. This gain is given by:

\[
G = \frac{70}{(R_{I1} + R_{I2} + 20)}
\]

The chosen values are \(R_{I1} = R_{I2} = 20\, \text{kΩ}\), applicable to resistors R27 to R30 and R33 to R36. In this case, according to the datasheet, all these resistors must have a tolerance
2. Hardware design

of 0.25 %. The second one is eliminating their noise using the capacitors C10 and C11 (3.3 pF) and the resistors R27 to R30 (20 kΩ) to implement two low-pass filters. Finally, resistors R25, R26, R31 and R32 are used for detecting any breaking of Resolver signal lines. Figure 2.3 shows an example of resolver output signals before entering to the IC.

Figure 2.3: Resolver output

To interface with the resolver a 9 pin Sub-D connector is placed in one edge of the PCB. Its pin outline is shown in table 2.3.

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S3</td>
</tr>
<tr>
<td>2</td>
<td>S4</td>
</tr>
<tr>
<td>3</td>
<td>R1</td>
</tr>
<tr>
<td>4</td>
<td>Unused</td>
</tr>
<tr>
<td>5</td>
<td>Unused</td>
</tr>
<tr>
<td>6</td>
<td>S1</td>
</tr>
<tr>
<td>7</td>
<td>S2</td>
</tr>
<tr>
<td>8</td>
<td>R2</td>
</tr>
<tr>
<td>9</td>
<td>Unused</td>
</tr>
</tbody>
</table>

Table 2.3: Resolver interface pin outline

The last part of the Smartcoder is its digital interface. This interface has redundancy, providing both a serial and a parallel data of the position. The serial interface works with an external clock (SCK) with a maximum frequency of 2 MHz. In this case a 1 MHz clock generated in the FPGA is used. Signal SCSB indicates the starting of a conversion and must be kept at a high level for 300 ns at least. When the conversion is done the
digital data is transmitted through the pin DATA on falling edges of SCK. The output is a 12 bit stream starting with the MSB plus 1 bit of parity. The parallel interface is controlled with the signals CSB, RDB and INHB(RD). CSB and RDB must be applied either at the same time or RDB after CSB. A maximum time of 40 ns after the falling edge of RDB is required for the output to be valid. Optionally the signal INHB(RD) can be used to lock the data. The 12 bit data is shown in the pins D0 to D11. Additionally a parity bit is also shown in the pin PRTY.

All these digital signals must be interfaced with the FPGA board. As the FPGA input/outputs use 3.3 V LVCMOS logic and the RD converter digital signals are based on a 0 V – 5 V logic, the IC’s SN74LVC16T245, [11], and SN74LVC4245, [12], are required to adapt them. The first one is a 16 bit bidirectional transceiver containing two 8-bit banks. The data flow direction of each bank is specified through the pins 1DIR and 1OE for the first one and through 2DIR and 2OE for the second. Both are configured to be used as outputs to the FPGA by setting the mentioned pins to GND. The IC has two power supplies, VCCA and VCCB, which specify the logic levels of each side of the chip. For this application the two VCCA pins are connected to the +3.3 V and the two VCCB pins are connected to the +5V line, using for each pin a 100 nF by-pass capacitor. The signals connected to each pin are shown in table 2.4.

<table>
<thead>
<tr>
<th>Bank</th>
<th>Input pin (from the RD)</th>
<th>Output pin (to the FPGA)</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1B1</td>
<td>1A1</td>
<td>ERRHLR</td>
</tr>
<tr>
<td></td>
<td>1B2</td>
<td>1A2</td>
<td>PRTY</td>
</tr>
<tr>
<td></td>
<td>1B3</td>
<td>1A3</td>
<td>D11</td>
</tr>
<tr>
<td></td>
<td>1B4</td>
<td>1A4</td>
<td>D10</td>
</tr>
<tr>
<td></td>
<td>1B5</td>
<td>1A5</td>
<td>D9</td>
</tr>
<tr>
<td></td>
<td>1B6</td>
<td>1A6</td>
<td>D8</td>
</tr>
<tr>
<td></td>
<td>1B7</td>
<td>1A7</td>
<td>D7</td>
</tr>
<tr>
<td></td>
<td>1B8</td>
<td>1A8</td>
<td>D6</td>
</tr>
<tr>
<td>2</td>
<td>2B1</td>
<td>2A1</td>
<td>D5</td>
</tr>
<tr>
<td></td>
<td>2B2</td>
<td>2A2</td>
<td>D4</td>
</tr>
<tr>
<td></td>
<td>2B3</td>
<td>2A3</td>
<td>D3</td>
</tr>
<tr>
<td></td>
<td>2B4</td>
<td>2A4</td>
<td>D2</td>
</tr>
<tr>
<td></td>
<td>2B5</td>
<td>2A5</td>
<td>D1</td>
</tr>
<tr>
<td></td>
<td>2B6</td>
<td>2A6</td>
<td>D0</td>
</tr>
<tr>
<td></td>
<td>2B7</td>
<td>2A7</td>
<td>DATA</td>
</tr>
<tr>
<td></td>
<td>2B8</td>
<td>2A8</td>
<td>UNUSED</td>
</tr>
</tbody>
</table>

*Table 2.4: Bus transceiver SN74LVC16T245 pin outline*

The LN74LVC4245 is used to adapt the inputs from the FPGA. To do that the pin OE is connected to GND and the pin DIR to +5V. In this case the IC only contains one 8 bit bank. Again, the power pins specify the logic level of each side. In this case the pin VCCA is connected to +5 V and the two VCCB pins to +3.3 V, using a 100 nF by-pass capacitor in each case. Its pin outline is shown in table 2.5.
2. Hardware design

<table>
<thead>
<tr>
<th>Bank</th>
<th>Input pin (from the FPGA)</th>
<th>Output pin (to the RD)</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B1</td>
<td>A1</td>
<td>SCSB</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>A2</td>
<td>SCK</td>
</tr>
<tr>
<td></td>
<td>B3</td>
<td>A3</td>
<td>CLKN</td>
</tr>
<tr>
<td></td>
<td>B4</td>
<td>A4</td>
<td>ERRSTB</td>
</tr>
<tr>
<td></td>
<td>B5</td>
<td>A5</td>
<td>INHB(RD)</td>
</tr>
<tr>
<td></td>
<td>B6</td>
<td>A6</td>
<td>RDB</td>
</tr>
<tr>
<td></td>
<td>B7</td>
<td>A7</td>
<td>CSB</td>
</tr>
<tr>
<td></td>
<td>B8</td>
<td>A8</td>
<td>UNUSED</td>
</tr>
</tbody>
</table>

*Table 2.5: Bus transceiver SN74LVC4245 pin outline*

The connection of all these signals with the FPGA is done through the connectors JP1, JP2 and JP4.

2.2.3 Motor control unit

This module contains three main parts: the inverter, the current sensors and the AD converters. Each one of them is explained in detail below.

![Figure 2.6: Motor control unit](image)

**Inverter**

To drive each motor a three-phase inverter is required. The selected component to do that is the triple half-bridge inverter L6234, [13]. Each half bridge is controlled with two different signals: an enable signal (EN1, EN2 and EN3) and a PWM signal (IN1, IN2
and IN3). This last signal controls the enabling of the MOSFET gates of the corresponding half bridge. If IN = 'H' the upper MOSFET is activated, while the lower one is activated if IN = 'L'. The two inverters are supplied with +24 V in the DC side. The interface with the motors is made through the connectors T2_1 and T2_2. Both have the same pin outline, shown in table 2.6. Pins 1, 2, 4, 6 and 7 were used in the previous project for a Hall sensor and are unused in this one.

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VHALL</td>
<td>Unused</td>
</tr>
<tr>
<td>2</td>
<td>GNDHALL</td>
<td>Unused</td>
</tr>
<tr>
<td>3</td>
<td>M3</td>
<td>Motor phase a</td>
</tr>
<tr>
<td>4</td>
<td>H3</td>
<td>Unused</td>
</tr>
<tr>
<td>5</td>
<td>M2</td>
<td>Motor phase b</td>
</tr>
<tr>
<td>6</td>
<td>H2</td>
<td>Unused</td>
</tr>
<tr>
<td>7</td>
<td>M1</td>
<td>Motor phase c</td>
</tr>
<tr>
<td>8</td>
<td>H1</td>
<td>Unused</td>
</tr>
<tr>
<td>9</td>
<td>VBUS</td>
<td>Inverter power supply</td>
</tr>
<tr>
<td>10</td>
<td>GND</td>
<td>Ground</td>
</tr>
</tbody>
</table>

Table 2.6: Motor interface pin outline

**Current sensors**

In order to obtain measurements of the currents in each phase current sensors are required. The main requirement when choosing them is that they must have high accuracy. For this reason a 0.05 Ω shunt resistor and a current shunt monitor, [14], will be used. To avoid an excessive heating up large footprint resistors are used. The current is measured as the voltage drop across the resistor. This voltage is read by the current sensor, which has an internal gain $G = 20$, so the total gain of the system is 1 V/A. Its output is configured with the pins Vref1 and Vref2. By setting Vref1 to +3.3 V and Vref2 to GND the sensor allows reading both positive and negative currents in the range from -1.65 A to 1.65 A, giving an output voltage of 1.65 V if the current is 0 A.

**AD converters**

A set of 6 AD converters, 3 for each motor control unit, is available in the PCB in order to convert different variables to a digital format. The selected AD is the ADC122S051, [15]. This is a two channel, 12 bit converter with a serial output. The converter is based on a successive approximation register and has an internal track and hold circuit. It can sample in the range from 200 to 500 ksp. This sample rate is controlled by an external clock connected in the pin SCLK. In this case a 5.625 MHz clock from the FPGA is used. The chip is supplied by connecting the pin $V_A$ to the +3.3 V line, using two by-pass capacitors. The voltage in this pin specifies the range of its analog inputs and its digital output. Finally the pin DIN specifies which channel is going to be converted in the next
sample. The list of the measured variables in each channel of each AD converter is shown in table 2.7.

<table>
<thead>
<tr>
<th>Component</th>
<th>Channel</th>
<th>Signal name</th>
<th>Measured variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>U8_1</td>
<td>1</td>
<td>Idc1</td>
<td>DC current motor 1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Ia1</td>
<td>Phase a current motor 1</td>
</tr>
<tr>
<td>U9_1</td>
<td>1</td>
<td>Vdc1</td>
<td>DC voltage motor 1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Ib1</td>
<td>Phase b current motor 1</td>
</tr>
<tr>
<td>U10_1</td>
<td>1</td>
<td>Vref1</td>
<td>Reference voltage</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Ic1</td>
<td>Phase c current motor 1</td>
</tr>
<tr>
<td>U8_2</td>
<td>1</td>
<td>Idc2</td>
<td>DC current motor 2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Ia2</td>
<td>Phase a current motor 2</td>
</tr>
<tr>
<td>U9_2</td>
<td>1</td>
<td>Vdc2</td>
<td>DC voltage motor 2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Ib2</td>
<td>Phase b current motor 2</td>
</tr>
<tr>
<td>U10_2</td>
<td>1</td>
<td>Vref2</td>
<td>Reference voltage</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Ic2</td>
<td>Phase c current motor 2</td>
</tr>
</tbody>
</table>

Table 2.7: Measured variables in each channel

The current signals of the motor are provided by the current sensors. A small circuit including a 150 Ω series resistor and a 4.7 nF parallel capacitor is inserted between the sensors and the AD to filter their noise. To sample the DC voltage of each motor, lines +VBUS1 and +VBUS2 are connected to a voltage divider containing a 1 kΩ and a 7.5 kΩ resistors. Thus, the output voltage corresponding to the 0 V to 24 V is 0 V to 2.8 V. To measure the DC current a LC filter and a current sensor available in the multiuse units explained below are used. Finally, as the selected converters do not have an internal voltage reference two channels are used to sample a stable voltage of +3.3 V provided by the voltage reference LM4120, [16], with an accuracy of 0.2 %.

2.2.4 Multiuse unit

The multiuse units contain two main blocks. The first one is thought to provide a measurement of the DC current. An LC filter containing a 330 µ inductance, a 100 nF and a 330 µF capacitor enables filtering this current. Then the same system with a shunt resistor and a current sensor used for measuring the motor currents is available.

The second block was designed in the previous project to enable optional current or voltage measurement. It consisted of a differential amplifier with an adjustable gain fixed with resistors. This block is unused in this project.
2. Hardware design

The interface with this unit is through the connectors T3_1 and T3_2 with a pin outline shown in the table 2.8.

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Filter in</td>
<td>LC filter input</td>
</tr>
<tr>
<td>2</td>
<td>Filter GND</td>
<td>LC filter ground</td>
</tr>
<tr>
<td>3</td>
<td>Filter out</td>
<td>LC filter output</td>
</tr>
<tr>
<td>4</td>
<td>Filter GND</td>
<td>LC filter ground</td>
</tr>
<tr>
<td>5</td>
<td>Current in</td>
<td>Current input</td>
</tr>
<tr>
<td>6</td>
<td>Current out</td>
<td>Current output</td>
</tr>
<tr>
<td>7</td>
<td>+ Diff amp</td>
<td>Unused</td>
</tr>
<tr>
<td>8</td>
<td>- Diff amp</td>
<td>Unused</td>
</tr>
</tbody>
</table>

*Table 2.8: Multiuse unit interface pin outline*

If one of these units is used to measure the currents in the DC side then the +24 V line must be connected to the pin filter in, the pin filter out must be short circuited with current in and finally the pin current out must be connected to the input +VBUS of the corresponding motor control unit.

Additionally, the PCB contains two connectors, JP3 and JP7, which can be used to connect another available board with an inverter and current sensors to the FPGA. Connector JP7 connects the board with the PCB and then all the signals are routed to JP3 to interface with the FPGA.
2.3. Assembly, soldering and test

This section shows all the steps followed since the PCB is received until it has been completely tested. The procedure consists in assembling and testing each unit of the PCB separately, starting with the power supply, then the RD unit and finally the motor control and multiuse units. Most of the components are soldered with the soldering iron and normal tin. For those with a small pitch the microscope, soldering paste and a small tip for the soldering iron will be required. To test the PCB the lab oscilloscope will be used.

The first unit to be soldered is the power supply. All the electrolytic capacitors and the diodes must be mounted according to their right polarity. The two regulators dissipate heat. To ensure that the heat transfer is good their body pin must be soldered with a large amount of tin to the main pad, using a large iron tip.

When testing the power supply for the first time one should check with the fingers if the regulators are heating up, which would mean there is a short circuit somewhere. The output of the two converters can be checked in the test points TST3 for the +5 V line and TST4 for the +3.3 V line. The test point TST1 is also available to check the input line +BAT.

The next assembled module is the RD converter. The first step to be realized is soldering the components AU6802N1, with a pitch of 0.65 mm, and SN74LVC14T245, with a pitch of 0.5 mm. They must be placed in the right orientation according to the circle that indicates their pin 1. These two components should be soldered first using solder paste and a thin tip and with the help of the microscope. After soldering them it should be checked optically with the microscope if there is any short circuit between consecutive pins due to leftovers of solder paste. Another important issue is checking that all the pins are properly connected with their respective pads, because it can cause errors that are difficult to recognize in later stages. The next step is soldering the rest of the external circuitry, containing capacitors, resistors and some IC, as well as all the connectors. This can be done easily using tin and the soldering iron with a normal tip. Once it has been done the board can be supplied again and the whole unit should be tested. The first thing to test is the supplies of all the IC. A second step is checking whether the excitation signals of the resolver have the right frequency and amplitude without connecting it. The test point TST5 contains two pins that allow making this operation. If this test is fine the resolver can be connected. As this means connecting a load to the system the excitation signals must be checked again. In this case they should have the same frequency, lower amplitude and some distortion in their waveform. Finally the output signals of the resolver can be checked. They should have a maximum amplitude 4 times lower than the excitation signals due to the resolver.
gain and one should be able to see how its amplitude is modulated by manually changing the rotor position.

The last step is soldering the motor control units and the multiuse units. The inverters have two body pins that must be soldered generously to their pad to ensure a good heat transfer. To test them the set of test points TST6 and TST7 are available, which allow to test the signals IN1, IN2, IN3, EN1, EN2 and EN3 of each inverter. The electrolytic capacitors C46_1, C46_2, C54_1 and C54_2 must be soldered with the right polarity. The rest of the components can be soldered easily. The board can then be supplied again to test the current sensors. To do that the set of test points TST8 and TST9 are available. A voltage of +1.65 V should be measured in this case because the phase currents are null at this stage. The digital outputs of the AD converters and the RD converter as well as the inverter outputs cannot be tested until the DSP/FPGA board is programmed properly to drive them.

The complete list of the set points can be found in the appendix A. The final board once it has been assembled is shown in figure 2.8.
2.4. PCB corrections

Two mistakes were done during the PCB design. They are explained below. However, the project files are left as they were sent to the manufacturer.

The footprint of the diodes D1, D2, D3 and D4, that indicates their polarity, is wrong; they must be soldered in the opposite direction.

The pins 1 and 2 of the connectors JP1, JP2, JP3, JP4, JP6_1 and JP6_2, one connected to ground and the other one isolated, should be changed. The pin 1 and pin 2 of the FPGA connectors are connected to +3.3 V and ground, so this would cause a short circuit when interfacing the 2 boards. To correct it all the pins 1 should be isolated and all the pins 2 connected to ground. In the realized board this problem was solved by cutting the wire that carries the +3.3 V signal.
3. Load design

The PMSM cannot be driven without any load coupled, because it could produce an excessive speed or acceleration of its shaft and even its burning. To avoid this problem a load must be chosen to offer the motor a resistive torque and a mechanical system to couple both ensuring a stable operation must be manufactured.

One option for the load is coupling to the motor a simple inertia element, like an aluminium cylinder. This option is simple to implement but the disadvantage is that the resistive torque cannot be controlled. To solve this the selected option is using another controlled motor as a load. This will allow to choose different resistive torque.

To select the motor some criteria must be fulfilled. It should have the same voltage, 24 V in the DC side, as the PMSM, similar dimensions and similar nominal speed. The brushless motor EC45 from maxon, [17], fulfils these requirements. It has a nominal speed of 7290 min$^{-1}$, a diameter of 45 mm and a shaft diameter of 4 mm.

Once the motor has been chosen the next and last step is manufacturing a mechanical system to couple both. The system should ensure a good alignment of both shafts to avoid their bending and low vibration of it. This second requirement is easy to fulfil in this case because the torque the motor is working with is relatively low. The design and manufacturing was made by the workshop team. The assembled system is shown in figure 3.1. It is based on an aluminium support that holds the two motors. Each one is screwed to an aluminium profile with a hole in the middle to insert their shaft. Finally a small cylinder with two holes couples them ensuring a good alignment.

![Figure 3.1: Motor and load assembled](image_url)
4. Control system

In the first section of this chapter it will be seen which is the motor control routine and how the control system is implemented. In the second section the FPGA design required to drive the motors will be explained.

4.1. Motor control routine

To implement the control system required for the targeted PMSM a board containing a DSP, [18], and a daughter board containing an FPGA designed in a previous semester thesis, [19], will be used. The main routine to control it is shown in the figure 4.1.

![Motor control routine diagram](image)

*Figure 4.1: Motor control routine*

This routine is based on a fixed PWM frequency, in this case, 10 kHz. At the end of each PWM period a new conversion starts in the AD’s and the RD. To drive them the FPGA will be used. Then, after certain time, when the conversions are done and the digital data of the measurements is available the FPGA will generate an interrupt inside the DSP and the rest of the period will be available to execute the desired control algorithm. In this algorithm the DSP will use the data of the conversions and will calculate the value of the duty cycles that must be applied in the next period. Finally the calculated values will be read by the FPGA and it will generate the appropriate signals to drive both inverters.
4. Control system

4.2. FPGA design

The FPGA will be used to implement different functionalities required to drive the motor. To do that a previous design thought to work with a PMSM will be modified and adapted using VHDL. This design will work with a global clock of 90 MHz. A brief description of the original design and the new blocks and entities included in this one are explained below.

4.2.1. Interface between the DSP and the FPGA

All the design inside the FPGA is commanded by the DSP. The interface between them is based on a 16 bit data bus either to read data from the FPGA or to write it and an 8 bit address bus both to select the data and the bank that wants to be read/write. To control this interface the set of signals CS, WR and RD is available. Additionally there is a signal to generate a reset in the FPGA.

The data bus is controlled by the original entity iobusmux that allows bidirectional communication between the DSP and the FPGA. This entity contains 16 data lines selected with a 4 bit address. In order to be able to read all the sampled values in the PCB the lines in the addresses 0010 and 0011, unused in the original project, will be connected. These lines are called dataa and datab and are also connected to the output of two entities combineout. Each one of them is an 8 line multiplexer where the measured magnitudes in the PCB are connected. The outline of these entities is shown in the table 4.1.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Address</th>
<th>Signal</th>
<th>Magnitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combineout/dataa</td>
<td>000</td>
<td>Posdata</td>
<td>Position</td>
</tr>
<tr>
<td></td>
<td>001</td>
<td>Ia1</td>
<td>Phase a current motor 1</td>
</tr>
<tr>
<td></td>
<td>010</td>
<td>Ib1</td>
<td>Phase b current motor 1</td>
</tr>
<tr>
<td></td>
<td>011</td>
<td>Ic1</td>
<td>Phase c current motor 1</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>Ia2</td>
<td>Phase a current motor 2</td>
</tr>
<tr>
<td></td>
<td>101</td>
<td>Ib2</td>
<td>Phase b current motor 2</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>Ic2</td>
<td>Phase c current motor 2</td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>Vcont1</td>
<td>DC voltage motor 1</td>
</tr>
<tr>
<td>Combineout/datab</td>
<td>000</td>
<td>Vcont2</td>
<td>DC voltage motor 2</td>
</tr>
<tr>
<td></td>
<td>001</td>
<td>Icont1</td>
<td>DC current motor 1</td>
</tr>
<tr>
<td></td>
<td>010</td>
<td>Icont2</td>
<td>DC current motor 2</td>
</tr>
<tr>
<td></td>
<td>011</td>
<td>Vref1</td>
<td>Reference voltage</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>Vref2</td>
<td>Reference voltage</td>
</tr>
<tr>
<td></td>
<td>101</td>
<td>Unused</td>
<td></td>
</tr>
<tr>
<td></td>
<td>110</td>
<td>Unused</td>
<td></td>
</tr>
<tr>
<td></td>
<td>111</td>
<td>Unused</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.1: Combineout entities outline
4. Control system

4.2.2. AD drivers

To drive the AD’s in the PCB two entities called adcontrol will be used. Each one of them will drive the three AD’s of one motor control unit. To do that they will generate the appropriate clock, chip select and the address of the channel that wants to be sampled and they will read the serial output of the AD’s and write their value in 16 bit registers. The VHDL code of these entities is available in the appendix B. This code is based on a state machine and a counter working with the 90 MHz clock. The routine to do that is explained below.

To sample the two channels a double sampling window will be generated at the end of each PWM period with the signal chip select. This window is shown in the figure 4.1.

![Figure 4.1: Double sampling window at 10 kHz](image)

The signal pulse_ad, created in the block modvec at the end of the PWM period will indicate the start of the conversions. Inside this sampling window a 5.625 MHz clock is generated. The reason for choosing this frequency is because it can be easily generated with the 90 MHz clock dividing it by 16, which is a power of two, and the obtained frequency is inside the clock range of the AD’s, which goes from 3.2 and 8 MHz. Then the AD’s will use the first 4 clock cycles to track and hold the value that must be sampled and the next 12 to provide the serial data. This data appears on falling edges of the clock and the FPGA will read it on the next rising edge to ensure a correct sampled value. The conversion of the two channels takes approximately 33 clock cycles, 16 to convert each channel and an additional one to create the second
chip select. This means that the total time to convert is around 6 \( \mu \)s, which represents only the 6 % of the period, leaving the rest available for the DSP to execute the control algorithm. A detail of the window and the generated clock is shown in the figure 4.2.

![Figure 4.2: Chip select and clock](image)

The signal called DIN, required to select which channel is going to be sampled, is also generated in the state machine. This signal is a stream of three bits that must be applied during the clock cycles 3, 4 and 5. The AD will keep their value in a register on rising edges of the clock and it will be used to indicate the channel sampled in the next sampling window. In this case the second channel will be sampled first by applying the stream 010 in the second sampling window and then the first channel will be sampled by applying the stream 000 in the first one. Thus, the measurement of the currents in each phase will be sampled at the beginning simultaneously with the position as it will be seen below.

Finally in the end of the conversions the read data is copied in 16 bit registers and will be available for the DSP until the end of the period.

### 4.2.3. RD converter

The block RD converter, containing 4 different entities, will be the responsible to drive the resolver to digital converter. Each one of them is explained below and their VHDL codes can be found in the appendix B.
4. Control system

Parallel data

This entity will generate the three signals required to control the parallel interface of the RD converter. The signal CSB will indicate when the conversion starts, the signal RDB will enable reading the parallel output and finally the signal INHB(RD) will lock this output the required time. In this case this will be done using again a state machine and a counter that will generate one sampling window at the end of each period. The start of this conversion is also indicated with the signal pulse_ad. The sampling window is shown in the figure 4.3.

![Figure 4.3: Sampling window for the RD converter](image)

The three signals will be kept at a low level during the same time as the required for the AD’s to convert the two channels. Then at the end of the sampling window the parallel output of the RD will be copied in a 16 bit register so that the digital data of the position is available for the DSP the rest of the period. At the same time this entity will generate during one clock cycle a signal called ad_end to indicate that all the conversions are done. This signal is connected in the block synchrogen that generates the signal DSPINT to create an interrupt in the DSP. Inside it the interrupt is selected by the DSP among 4 different signals: the mentioned ad_end and the signals sawend, pulse_ad and pulseint.
4. Control system

**Sckgen1**

This entity uses the 90 MHz clock to generate a 1 MHz clock required to drive the serial output of the RD. To do that a frequency divider by 90 is used. The generated clock is shown in the figure 4.4.

![Figure 4.4: 1 MHz clock generated in the entity sckgen1](image)

**Serial data**

The entity serial data drives the serial output of the RD converter, which requires only two signals: a clock, called SCK, and a chip select, called SCSB. To create the clock the one generated in the entity skcgen1 is used and connected to an output of this entity. Then a state machine is created based on this clock and a counter. This state machine creates in the beginning a high level of the signal SCSB during one clock cycle and then it reads the serial output of the converter and after 15 clock cycles it copies the read data in a 16 bit register. In this case the conversion takes 15 μs, much longer than the required for the AD's. The signal SCSB is shown in the figure 4.5. Moreover, unlike the entities adcontrol and parallel data this one is not synchronised with the signal pulse_ad and works independently.
4. Control system

Figure 4.5: Signal SCSB

Multiplex2

This entity allows selecting which is the output format of the position that will be read by the DSP. If the internal signal sel is set to 1 the parallel format is selected, if it is set to 0, the serial format is selected. In this version the signal is set to 1 so that all the system is synchronous with the signal pulse_ad and the conversions are as fast as possible.

4.2.4. Driving the inverters

Finally all the design required to drive the two inverters is already available in the original project and only a small modification concerning the inverter will be explained here. Basically this design reads the frequency of the PWM fixed by the DSP and generates the 6 signals for each inverter with the duty cycle calculated read there. Each one of these 6 signals controls one MOSFET of an inverter and is generated with the entities called antichev. The problem is that the inverters used in the PCB work with 3 signals to control the 6 MOSFET’s and 3 additional signals that allow enabling or disabling each leg. To solve this problem 3 of the original signals at the output of the entities antichev have been kept connected and the other 3 have been disconnected. Then an entity called memory16 is used to create the 3 enable signals for each inverter. This entity puts all of them to 0 when a reset is produced, disabling the 2 inverters while the motors are starting up in order to make the system safer.
5. Tests

Once the PCB has been designed and tested and the FPGA project has been implemented the next step before starting the control of the motor is testing that all the AD’s and the RD are sampling properly. To do that the DSP is programmed so that at every interrupt it reads all the data lines inside the FPGA and stores their value in an array. All the data is a 12 bit word that after being stored is plotted using MATLAB as an integer in order to check that the obtained values are correct.

The first test is the position. To do that the rotor position is changed manually during some seconds and the measurements are then plotted. The result is shown in the figure 5.1.

![Figure 5.1: Position test](image)

As it can be seen in the graphic when the rotor makes one round the obtained digital value changes from 0x000 to 0xFFF and then it starts again. In addition the difference between some consecutive samples will allow the DSP to identify the rotating direction. Finally the figure 5.2, which is a detail of the graphic above, shows that the samples have no noise.
5. Tests

Once the position has been tested the next step is testing the AD’s and the different magnitudes they measure. The first test is made without any supply in the inverters. In this case all the currents are 0 A, the output of the sensors should be more or less 1.65 V and the sampled value should be around 0x100. Figure 5.3 shows an example of a sampled current.
The test shows a set of samples with a mean of 0x10B, a value a little bit higher than the expected 0x100, and two bits of noise.

The voltage reference of 3.3 V provided by an external regulator connected to the AD’s is also tested. The result is shown in the figure 5.4.
5. Tests

As it can be seen the channel is saturated at 0xFFF as expected because the voltage and the supply of the AD are equal.

Finally the last test made was applying a load to one inverter and see how the currents are sampled. In this case the load consisted of 2 resistors and two inductors in series connected between two phases of the inverter output. Then a software routine that created a PWM at each phase was implemented so that the expected current is a sine wave with some ripple. The result is shown in figure 5.5

![Current test applying a PWM](image)

*Figure 5.5: Current test applying a PWM*

It is clearly seen in the figure that some of the samples are not being, collapsing to the value 0x100 or 0x000 sometimes. This problem may be due to a bad synchronization between the AD’s and the FPGA when reading their digital output and could not be solved due to a lack of time. Without taking into account these samples the current sensors seem to work properly when a load is applied.
6. Used software

PCB design

To design the PCB the software Altium Designer Summer 2009 was used. All the files of the project, including the schematics, the component and footprint libraries and the PCB are included in the project CD in the folder PCB. They are left in the state they were when the PCB was ordered. If it has to be manufactured again the corrections proposed in section 2.4 should be realized.

FPGA design

The software Xilinx ISE 10.2 was used to design the circuitry inside the FPGA. This software allows creating and editing it using both VHDL language and schematic edition. It also creates an output file with the format .bit that will be used later to program the FPGA using the DSP. The project files are also included in the project CD inside the folder FPGA project.

DSP/FPGA board programming

The DSP board was programmed with the software 6713 DSK CCStudio V3.1 using an USB port. Two projects are available in the project CD. The first one, called FPGA programming allows loading the FPGA design mentioned above using the DSP. To do that one must follow some steps. First the bit file must be created using Xilinx ISE 10.2. Then this file must be converted to a C file using a software tool like ITHALGO 2.0. This C file must be copied inside the FPGA programming project and finally the project must be rebuilt with CCStudio and the program file loaded to the DSP. The second project, called controller, contains the controller implementation needed to drive the motor.
Conclusion

The realization of a testbench for a high dynamic performance PMSM was a great milestone. In a first step the best topology to control the PMSM had to be selected. To implement it, it was necessary to design a PCB, order the components, assemble them and test it. The greatest challenge in this part was soldering those components with the smallest pitch and test that the PCB was working properly. Some errors were found during the tests and some modifications had to be done to ensure a proper operation of PCB. Then another motor had to be chosen and a mechanical system designed to couple both. This task was relatively easy thanks to the help of the workshop. Then a project for the FPGA to drive the motor and test the AD’s and the RD was realized. It was really interesting to learn how to drive AD converters using an FPGA with the help of the VHDL language. Finally this project was loaded in the DSP/FPGA board and the AD’s and RD could be partially tested due to lack of time. After finishing testing them the testbench will finally be ready to test different controllers for the motor, which was the main goal of this Master thesis.
Outlook

Now the next step is fully test the AD’s. To do that, different things must be done:

- Check the VHDL code that drives them to see if the string of bits given in serial format are read by the FPGA at the correct time
- Once it has been solved a PWM can be applied again to a load containing resistors and inductances and it must be checked whether the expected values for the current calculated theoretically are the same as the sampled ones

If the mentioned tests are fine then the system is ready to test the motor in this sequence:

- First the motor should be tested in open-loop by applying a voltage/frequency controller at each phase of it.
- If this is fine then the system is ready to test different controllers, starting with small torque controllers and continuing with more complex ones to control the torque, the speed or the position
# Nomenclature

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD</td>
<td>Analog to digital converter</td>
</tr>
<tr>
<td>ASV</td>
<td>Autonomous solar vehicle</td>
</tr>
<tr>
<td>DA</td>
<td>Digital to analog converter</td>
</tr>
<tr>
<td>DC</td>
<td>Direct current</td>
</tr>
<tr>
<td>DSP</td>
<td>Digital signal processor</td>
</tr>
<tr>
<td>FPGA</td>
<td>Field programmable gate array</td>
</tr>
<tr>
<td>IC</td>
<td>Integrated circuit</td>
</tr>
<tr>
<td>MSB</td>
<td>Most significant bit</td>
</tr>
<tr>
<td>PCB</td>
<td>Printed circuit board</td>
</tr>
<tr>
<td>PMSM</td>
<td>Permanent magnet synchronous motor</td>
</tr>
<tr>
<td>PWM</td>
<td>Pulse width modulation</td>
</tr>
<tr>
<td>RD</td>
<td>Resolver to digital converter</td>
</tr>
<tr>
<td>USB</td>
<td>Universal series bus</td>
</tr>
<tr>
<td>VHDL</td>
<td>Very high speed integrated circuit hardware description language</td>
</tr>
</tbody>
</table>
Bibliography

[1] Tamagawa Seiki Co. Ltd. TS4734 brushless servomotor specification
[8] NXP. 45 V, 500 mA npn general-purpose transistors datasheet
[9] NXP. 45 V, 500 mA pnp general-purpose transistors datasheet
[15] National Semiconductor. 2 channel, 12 bit AD converter ADC122S051 datasheet
[18] Spectrum Digital. TMS320C6713 DSK technical reference
[19] Thomas Haag. Design and realization of a DSP board for hybrid system controller
Appendix

A. PCB design
   A.1. PCB Schematics
   A.2. PCB layers outline
   A.3. Order list
   A.4. List of test points

B. VHDL codes
   B.1. Adcontrol
   B.2. Parallel data
   B.3. Sckgen1
   B.4. Serial data
   B.5. Multiplex2
A.1. PCB Schematics
A.2. PCB layers outline
A.3. Order list
## PART LIST

### Board overview

<table>
<thead>
<tr>
<th>Module</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>RD</td>
<td>32.62</td>
</tr>
<tr>
<td>Power supply</td>
<td>31.92</td>
</tr>
<tr>
<td>Motor control and multiuse</td>
<td>182.68</td>
</tr>
<tr>
<td>ID</td>
<td>Part</td>
</tr>
<tr>
<td>----</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>C1</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C2</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C3</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C4</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C5</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C6</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C7</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C8</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C9</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C10</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C11</td>
<td>Capacitor SMD 1026</td>
</tr>
<tr>
<td>C12</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C13</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C14</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C15</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C16</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C17</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C18</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C19</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C20</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C21</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C22</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C23</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C24</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C25</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>C26</td>
<td>Capacitor SMD 1206</td>
</tr>
<tr>
<td>D1</td>
<td>Diode SMD 1206 N4148</td>
</tr>
<tr>
<td>D2</td>
<td>Diode SMD 1206 N4148</td>
</tr>
<tr>
<td>D3</td>
<td>Diode SMD 1206 N4148</td>
</tr>
<tr>
<td>D4</td>
<td>Diode SMD 1206 N4148</td>
</tr>
<tr>
<td>JP1</td>
<td>Header 3X2</td>
</tr>
<tr>
<td>JP2</td>
<td>Header 5X2</td>
</tr>
<tr>
<td>JP4</td>
<td>Header 7X2</td>
</tr>
<tr>
<td>JP5</td>
<td>9 pin Sub D connector</td>
</tr>
<tr>
<td>Q1</td>
<td>Transistor SOT23 NPN</td>
</tr>
<tr>
<td>Q2</td>
<td>Transistor SOT23 PNP</td>
</tr>
<tr>
<td>Q3</td>
<td>Transistor SOT23 NPN</td>
</tr>
<tr>
<td>Q4</td>
<td>Transistor SOT23 PNP</td>
</tr>
<tr>
<td>Resistor</td>
<td>Value</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>R1</td>
<td>10 kΩ</td>
</tr>
<tr>
<td>R2</td>
<td>10 kΩ</td>
</tr>
<tr>
<td>R3</td>
<td>10 kΩ</td>
</tr>
<tr>
<td>R4</td>
<td>10 kΩ</td>
</tr>
<tr>
<td>R5</td>
<td>10 kΩ</td>
</tr>
<tr>
<td>R6</td>
<td>270 kΩ</td>
</tr>
<tr>
<td>R7</td>
<td>270 kΩ</td>
</tr>
<tr>
<td>R8</td>
<td>51 kΩ</td>
</tr>
<tr>
<td>R9</td>
<td>51 kΩ</td>
</tr>
<tr>
<td>R10</td>
<td>51 kΩ</td>
</tr>
<tr>
<td>R11</td>
<td>100 kΩ</td>
</tr>
<tr>
<td>R12</td>
<td>100 kΩ</td>
</tr>
<tr>
<td>R13</td>
<td>100 kΩ</td>
</tr>
<tr>
<td>R14</td>
<td>5.6 Ω</td>
</tr>
<tr>
<td>R15</td>
<td>5.6 Ω</td>
</tr>
<tr>
<td>R16</td>
<td>20 kΩ</td>
</tr>
<tr>
<td>R17</td>
<td>20 kΩ</td>
</tr>
<tr>
<td>R18</td>
<td>20 kΩ</td>
</tr>
<tr>
<td>R19</td>
<td>20 kΩ</td>
</tr>
<tr>
<td>R20</td>
<td>120 kΩ</td>
</tr>
<tr>
<td>R21</td>
<td>120 kΩ</td>
</tr>
<tr>
<td>R22</td>
<td>120 kΩ</td>
</tr>
<tr>
<td>R23</td>
<td>120 kΩ</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>R42</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R43</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R44</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R45</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R46</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R47</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R48</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R49</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R50</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R51</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R52</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R53</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R54</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R55</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R56</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R57</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R58</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>R59</td>
<td>Resistor SMD 1206</td>
</tr>
<tr>
<td>U1</td>
<td>R/D converter AU6802N1</td>
</tr>
<tr>
<td>U2</td>
<td>Bus transceiver SN74LVC16T245</td>
</tr>
<tr>
<td>U3</td>
<td>Bus transceiver SN74LVC4245</td>
</tr>
<tr>
<td>U4</td>
<td>Operational amplifier AD824AR</td>
</tr>
<tr>
<td>Y1</td>
<td>Crystal 20 MHz</td>
</tr>
</tbody>
</table>

### Power supply module

<table>
<thead>
<tr>
<th>ID</th>
<th>Part</th>
<th>Value</th>
<th>Order Number</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>C27</td>
<td>Capacitor ELKO THT</td>
<td>100 μF</td>
<td>Farnell 8813108</td>
<td>0.520</td>
</tr>
<tr>
<td>C28</td>
<td>Capacitor ELKO SMD</td>
<td>33 μF</td>
<td>Farnell 9753834</td>
<td>0.840</td>
</tr>
<tr>
<td>C29</td>
<td>Capacitor ELKO THT</td>
<td>330 μF</td>
<td>Farnell 1144622</td>
<td>0.540</td>
</tr>
<tr>
<td>C30</td>
<td>Capacitor ELKO THT</td>
<td>330 μF</td>
<td>Farnell 1144622</td>
<td>0.540</td>
</tr>
<tr>
<td>C31</td>
<td>Capacitor ELKO SMD</td>
<td>33 μF</td>
<td>Farnell 9753834</td>
<td>0.840</td>
</tr>
<tr>
<td>D5</td>
<td>Diode SMB, DO-214AA</td>
<td>1A, 40V</td>
<td>Farnell 8647771</td>
<td>0.340</td>
</tr>
<tr>
<td>D6</td>
<td>Diode SMB, DO-214AA</td>
<td>1A, 40V</td>
<td>Farnell 8647771</td>
<td>0.340</td>
</tr>
<tr>
<td>D7</td>
<td>Diode SMB, DO-214AA</td>
<td>1A, 40V</td>
<td>Farnell 8647771</td>
<td>0.340</td>
</tr>
<tr>
<td>D8</td>
<td>Diode SMB, DO-214AA</td>
<td>1A, 40V</td>
<td>Farnell 8647771</td>
<td>0.340</td>
</tr>
<tr>
<td>L1</td>
<td>Inductor EPCOS Power</td>
<td>33 μH</td>
<td>Farnell 1644441</td>
<td>1.620</td>
</tr>
<tr>
<td>L2</td>
<td>Inductor EPCOS Power</td>
<td>330 μH</td>
<td>Farnell 1644653</td>
<td>1.820</td>
</tr>
<tr>
<td>L3</td>
<td>Inductor EPCOS Power</td>
<td>33 μH</td>
<td>Farnell 1644441</td>
<td>1.620</td>
</tr>
<tr>
<td>L4</td>
<td>Inductor EPCOS Power</td>
<td>330 μH</td>
<td>Farnell 1644653</td>
<td>1.820</td>
</tr>
<tr>
<td>T1</td>
<td>Header (POWER)</td>
<td>2x5 3.5 90°</td>
<td>Farnell 1857832</td>
<td>4.660</td>
</tr>
</tbody>
</table>
**Motor control and multiuse modules**

<table>
<thead>
<tr>
<th>ID</th>
<th>Part</th>
<th>Value</th>
<th>Order Number</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>C32_1</td>
<td>Capacitor SMD 1206</td>
<td>22 nF</td>
<td>Farnell 1740716</td>
<td>0.310</td>
</tr>
<tr>
<td>C32_2</td>
<td>Capacitor SMD 1206</td>
<td>22 nF</td>
<td>Farnell 1740716</td>
<td>0.310</td>
</tr>
<tr>
<td>C33_1</td>
<td>Capacitor SMD 1206</td>
<td>4.7 nF</td>
<td>Farnell 1414737</td>
<td>0.410</td>
</tr>
<tr>
<td>C33_2</td>
<td>Capacitor SMD 1206</td>
<td>4.7 nF</td>
<td>Farnell 1414737</td>
<td>0.410</td>
</tr>
<tr>
<td>C34_1</td>
<td>Capacitor SMD 1206</td>
<td>4.7 nF</td>
<td>Farnell 1414737</td>
<td>0.410</td>
</tr>
<tr>
<td>C34_2</td>
<td>Capacitor SMD 1206</td>
<td>4.7 nF</td>
<td>Farnell 1414737</td>
<td>0.410</td>
</tr>
<tr>
<td>C35_1</td>
<td>Capacitor SMD 1206</td>
<td>4.7 nF</td>
<td>Farnell 1414737</td>
<td>0.410</td>
</tr>
<tr>
<td>C35_2</td>
<td>Capacitor SMD 1206</td>
<td>4.7 nF</td>
<td>Farnell 1414737</td>
<td>0.410</td>
</tr>
<tr>
<td>C36_1</td>
<td>Capacitor SMD 1206</td>
<td>4.7 nF</td>
<td>Farnell 1414737</td>
<td>0.410</td>
</tr>
<tr>
<td>C36_2</td>
<td>Capacitor SMD 1206</td>
<td>4.7 nF</td>
<td>Farnell 1414737</td>
<td>0.410</td>
</tr>
<tr>
<td>C37_1</td>
<td>Capacitor SMD 1206</td>
<td>4.7 nF</td>
<td>Farnell 1414737</td>
<td>0.410</td>
</tr>
<tr>
<td>C37_2</td>
<td>Capacitor SMD 1206</td>
<td>4.7 nF</td>
<td>Farnell 1414737</td>
<td>0.410</td>
</tr>
<tr>
<td>C38_1</td>
<td>Capacitor SMD 1206</td>
<td>1 µF</td>
<td>Farnell 1617260</td>
<td>0.163</td>
</tr>
<tr>
<td>C38_2</td>
<td>Capacitor SMD 1206</td>
<td>1 µF</td>
<td>Farnell 1617260</td>
<td>0.163</td>
</tr>
<tr>
<td>C39_1</td>
<td>Capacitor SMD 1206</td>
<td>1 µF</td>
<td>Farnell 1617260</td>
<td>0.163</td>
</tr>
<tr>
<td>C39_2</td>
<td>Capacitor SMD 1206</td>
<td>1 µF</td>
<td>Farnell 1617260</td>
<td>0.163</td>
</tr>
<tr>
<td>C40_1</td>
<td>Capacitor SMD 1206</td>
<td>1 µF</td>
<td>Farnell 1617260</td>
<td>0.163</td>
</tr>
<tr>
<td>C40_2</td>
<td>Capacitor SMD 1206</td>
<td>1 µF</td>
<td>Farnell 1617260</td>
<td>0.163</td>
</tr>
<tr>
<td>C41_1</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell 9406557</td>
<td>0.172</td>
</tr>
<tr>
<td>C41_2</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell 9406557</td>
<td>0.172</td>
</tr>
<tr>
<td>C42_1</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell 9406557</td>
<td>0.172</td>
</tr>
<tr>
<td>C42_2</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell 9406557</td>
<td>0.172</td>
</tr>
<tr>
<td>C43_1</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell 9406557</td>
<td>0.172</td>
</tr>
<tr>
<td>C43_2</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell 9406557</td>
<td>0.172</td>
</tr>
<tr>
<td>C44_1</td>
<td>Capacitor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C44_2</td>
<td>Capacitor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C45_1</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell 9406557</td>
<td>0.172</td>
</tr>
<tr>
<td>C45_2</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell 9406557</td>
<td>0.172</td>
</tr>
<tr>
<td>C46_1</td>
<td>Capacitor ELKO THT</td>
<td>330 µF</td>
<td>Farnell 1144622</td>
<td>0.540</td>
</tr>
<tr>
<td>C46_2</td>
<td>Capacitor ELKO THT</td>
<td>330 µF</td>
<td>Farnell 1144622</td>
<td>0.540</td>
</tr>
<tr>
<td>C47_1</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell 9406557</td>
<td>0.172</td>
</tr>
<tr>
<td>C47_2</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell 9406557</td>
<td>0.172</td>
</tr>
<tr>
<td>C48_2</td>
<td>Capacitor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component</td>
<td>Type</td>
<td>Value</td>
<td>Manufacturer</td>
<td>Code</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------</td>
<td>--------</td>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td>C48_2</td>
<td>Capacitor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C49_1</td>
<td>Capacitor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C49_2</td>
<td>Capacitor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C50_1</td>
<td>Capacitor SMD 1206</td>
<td>220 nF</td>
<td>Farnell</td>
<td>1833854</td>
</tr>
<tr>
<td>C50_2</td>
<td>Capacitor SMD 1206</td>
<td>220 nF</td>
<td>Farnell</td>
<td>1833854</td>
</tr>
<tr>
<td>C51_1</td>
<td>Capacitor SMD 1206</td>
<td>10 nF</td>
<td>Farnell</td>
<td>9406425</td>
</tr>
<tr>
<td>C51_2</td>
<td>Capacitor SMD 1206</td>
<td>10 nF</td>
<td>Farnell</td>
<td>9406425</td>
</tr>
<tr>
<td>C52_1</td>
<td>Capacitor SMD 1206</td>
<td>1 µF</td>
<td>Farnell</td>
<td>9406514</td>
</tr>
<tr>
<td>C52_2</td>
<td>Capacitor SMD 1206</td>
<td>1 µF</td>
<td>Farnell</td>
<td>9406514</td>
</tr>
<tr>
<td>C53_1</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell</td>
<td>9406557</td>
</tr>
<tr>
<td>C53_2</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell</td>
<td>9406557</td>
</tr>
<tr>
<td>C54_1</td>
<td>Capacitor ELKO THT</td>
<td>100 µF</td>
<td>Farnell</td>
<td>8813108</td>
</tr>
<tr>
<td>C54_2</td>
<td>Capacitor ELKO THT</td>
<td>100 µF</td>
<td>Farnell</td>
<td>8813108</td>
</tr>
<tr>
<td>C55_1</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell</td>
<td>9406557</td>
</tr>
<tr>
<td>C55_2</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell</td>
<td>9406557</td>
</tr>
<tr>
<td>C56_1</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell</td>
<td>9406557</td>
</tr>
<tr>
<td>C56_2</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell</td>
<td>9406557</td>
</tr>
<tr>
<td>C57_1</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell</td>
<td>9406557</td>
</tr>
<tr>
<td>C57_2</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell</td>
<td>9406557</td>
</tr>
<tr>
<td>C58_1</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell</td>
<td>9406557</td>
</tr>
<tr>
<td>C58_2</td>
<td>Capacitor SMD 1206</td>
<td>100 nF</td>
<td>Farnell</td>
<td>9406557</td>
</tr>
<tr>
<td>D9_1</td>
<td>Diode SMD 1206 1N4148</td>
<td></td>
<td>RS</td>
<td>652-6003</td>
</tr>
<tr>
<td>D9_2</td>
<td>Diode SMD 1206 1N4148</td>
<td></td>
<td>RS</td>
<td>652-6003</td>
</tr>
<tr>
<td>D10_1</td>
<td>Diode SMD 1206 1N4148</td>
<td></td>
<td>RS</td>
<td>652-6003</td>
</tr>
<tr>
<td>D10_2</td>
<td>Diode SMD 1206 1N4148</td>
<td></td>
<td>RS</td>
<td>652-6003</td>
</tr>
<tr>
<td>JP6_1</td>
<td>Header 7x2</td>
<td></td>
<td>Farnell</td>
<td>9838252</td>
</tr>
<tr>
<td>JP6_2</td>
<td>Header 7x2</td>
<td></td>
<td>Farnell</td>
<td>9838252</td>
</tr>
<tr>
<td>JP3</td>
<td>Header 7x2</td>
<td></td>
<td>Farnell</td>
<td>9838252</td>
</tr>
<tr>
<td>JP7</td>
<td>Header 12x2</td>
<td></td>
<td>Farnell</td>
<td>9838481</td>
</tr>
<tr>
<td>L5_1</td>
<td>Toroid Inductor</td>
<td>1 mH</td>
<td>Farnell</td>
<td>1610402</td>
</tr>
<tr>
<td>L5_2</td>
<td>Toroid Inductor</td>
<td>1 mH</td>
<td>Farnell</td>
<td>1610402</td>
</tr>
<tr>
<td>L5_1</td>
<td>Inductor EPCOS Power</td>
<td>330 µH</td>
<td>Farnell</td>
<td>1644653</td>
</tr>
<tr>
<td>L5_2</td>
<td>Inductor EPCOS Power</td>
<td>330 µH</td>
<td>Farnell</td>
<td>1644653</td>
</tr>
<tr>
<td>R60_1</td>
<td>Resistor SMD 1206</td>
<td>1 kΩ</td>
<td>Farnell</td>
<td>1717745</td>
</tr>
<tr>
<td>R60_2</td>
<td>Resistor SMD 1206</td>
<td>1 kΩ</td>
<td>Farnell</td>
<td>1717745</td>
</tr>
<tr>
<td>R61_1</td>
<td>Resistor SMD 1206</td>
<td>7.5 kΩ</td>
<td>Farnell</td>
<td>1841771</td>
</tr>
<tr>
<td>R61_2</td>
<td>Resistor SMD 1206</td>
<td>7.5 kΩ</td>
<td>Farnell</td>
<td>1841771</td>
</tr>
<tr>
<td>R62_1</td>
<td>Resistor SMD 1206</td>
<td>150 Ω</td>
<td>Farnell</td>
<td>9335943</td>
</tr>
<tr>
<td>R62_2</td>
<td>Resistor SMD 1206</td>
<td>150 Ω</td>
<td>Farnell</td>
<td>9335943</td>
</tr>
<tr>
<td>R63_1</td>
<td>Resistor SMD 1206</td>
<td>150 Ω</td>
<td>Farnell</td>
<td>9335943</td>
</tr>
<tr>
<td>R63_2</td>
<td>Resistor SMD 1206</td>
<td>150 Ω</td>
<td>Farnell</td>
<td>9335943</td>
</tr>
<tr>
<td>Part Number</td>
<td>Description</td>
<td>Value</td>
<td>Catalog Number</td>
<td>Supplier</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>-------</td>
<td>----------------</td>
<td>----------</td>
</tr>
<tr>
<td>R64_1</td>
<td>Resistor SMD 1206</td>
<td>150 Ω</td>
<td>Farnell 9335943</td>
<td>0.042</td>
</tr>
<tr>
<td>R64_2</td>
<td>Resistor SMD 1206</td>
<td>150 Ω</td>
<td>Farnell 9335943</td>
<td>0.042</td>
</tr>
<tr>
<td>R65_1</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R65_2</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R66_1</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R66_2</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R67_1</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R67_2</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R68_1</td>
<td>Resistor SMD 2512</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R68_2</td>
<td>Resistor SMD 2512</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R69_1</td>
<td>Resistor SMD 2512</td>
<td>0.050 Ω</td>
<td>Farnell 1634327</td>
<td>0.460</td>
</tr>
<tr>
<td>R69_2</td>
<td>Resistor SMD 2512</td>
<td>0.050 Ω</td>
<td>Farnell 1634327</td>
<td>0.460</td>
</tr>
<tr>
<td>R70_1</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R70_2</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R71_1</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R71_2</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R72_1</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R72_2</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R73_1</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R73_2</td>
<td>Resistor SMD 1206</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R74_1</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R74_2</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R75_1</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R75_2</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R76_1</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R76_2</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R77_1</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R77_2</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R78_1</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R78_2</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R79_1</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R79_2</td>
<td>Resistor SMD 1206</td>
<td>270 Ω</td>
<td>Farnell 9336257</td>
<td>0.040</td>
</tr>
<tr>
<td>R80_1</td>
<td>Resistor SMD 2512</td>
<td>0.050 Ω</td>
<td>Farnell 1634327</td>
<td>0.460</td>
</tr>
<tr>
<td>R80_2</td>
<td>Resistor SMD 2512</td>
<td>0.050 Ω</td>
<td>Farnell 1634327</td>
<td>0.460</td>
</tr>
<tr>
<td>R81_1</td>
<td>Resistor SMD 2512</td>
<td>0.050 Ω</td>
<td>Farnell 1634327</td>
<td>0.460</td>
</tr>
<tr>
<td>R81_2</td>
<td>Resistor SMD 2512</td>
<td>0.050 Ω</td>
<td>Farnell 1634327</td>
<td>0.460</td>
</tr>
<tr>
<td>R82_1</td>
<td>Resistor SMD 2512</td>
<td>0.050 Ω</td>
<td>Farnell 1634327</td>
<td>0.460</td>
</tr>
<tr>
<td>R82_2</td>
<td>Resistor SMD 2512</td>
<td>0.050 Ω</td>
<td>Farnell 1634327</td>
<td>0.460</td>
</tr>
<tr>
<td>T2_1</td>
<td>Header (MULTIUSE)</td>
<td>2x5 3.5 90°</td>
<td>Farnell 1857832</td>
<td>4.660</td>
</tr>
<tr>
<td>T2_2</td>
<td>Header (MULTIUSE)</td>
<td>2x5 3.5 90°</td>
<td>Farnell 1857832</td>
<td>4.660</td>
</tr>
<tr>
<td>T2_1</td>
<td>Socket (MULTIUSE)</td>
<td>2x3 3.5</td>
<td>Digikey 281-1851-ND</td>
<td>7.280</td>
</tr>
<tr>
<td>Part</td>
<td>Description</td>
<td>Vendor</td>
<td>Part Number</td>
<td>Price</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------</td>
<td>------------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>T2_2</td>
<td>Socket (MULTIUSE)</td>
<td>Digikey</td>
<td>281-1851-ND</td>
<td>7.280</td>
</tr>
<tr>
<td>T3_1</td>
<td>Header (MULTIUSE)</td>
<td>Farnell</td>
<td>1857831</td>
<td>3.710</td>
</tr>
<tr>
<td>T3_2</td>
<td>Header (MULTIUSE)</td>
<td>Farnell</td>
<td>1857831</td>
<td>3.710</td>
</tr>
<tr>
<td>T3_1</td>
<td>Socket (MULTIUSE)</td>
<td>Digikey</td>
<td>281-1850-ND</td>
<td>6.330</td>
</tr>
<tr>
<td>T3_2</td>
<td>Socket (MULTIUSE)</td>
<td>Digikey</td>
<td>281-1850-ND</td>
<td>6.330</td>
</tr>
<tr>
<td>U7_1</td>
<td>Voltage Reference LM4120</td>
<td>Farnell</td>
<td>9778810</td>
<td>1.970</td>
</tr>
<tr>
<td>U7_2</td>
<td>Voltage Reference LM4120</td>
<td>Farnell</td>
<td>9778810</td>
<td>1.970</td>
</tr>
<tr>
<td>U8_1</td>
<td>A/D converter ADC122S051</td>
<td>Farnell</td>
<td>1007995</td>
<td>4.300</td>
</tr>
<tr>
<td>U8_2</td>
<td>A/D converter ADC122S051</td>
<td>Farnell</td>
<td>1007995</td>
<td>4.300</td>
</tr>
<tr>
<td>U9_1</td>
<td>A/D converter ADC122S051</td>
<td>Farnell</td>
<td>1007995</td>
<td>4.300</td>
</tr>
<tr>
<td>U9_2</td>
<td>A/D converter ADC122S051</td>
<td>Farnell</td>
<td>1007995</td>
<td>4.300</td>
</tr>
<tr>
<td>U10_1</td>
<td>A/D converter ADC122S051</td>
<td>Farnell</td>
<td>1007995</td>
<td>4.300</td>
</tr>
<tr>
<td>U10_2</td>
<td>A/D converter ADC122S051</td>
<td>Farnell</td>
<td>1007995</td>
<td>4.300</td>
</tr>
<tr>
<td>U11_1</td>
<td>Difference Amplifier AD628</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U11_2</td>
<td>Difference Amplifier AD628</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U12_1</td>
<td>Current Shunt Monitor AD8210</td>
<td>Farnell</td>
<td>1274226</td>
<td>4.600</td>
</tr>
<tr>
<td>U12_2</td>
<td>Current Shunt Monitor AD8210</td>
<td>Farnell</td>
<td>1274226</td>
<td>9.400</td>
</tr>
<tr>
<td>U13_1</td>
<td>Operational Amplifier AD8031AR</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U13_2</td>
<td>Operational Amplifier AD8031AR</td>
<td>Unused</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U14_1</td>
<td>Motor Driver Three Phase SO20 L6234PD</td>
<td>Farnell</td>
<td>1467717</td>
<td>12.490</td>
</tr>
<tr>
<td>U14_2</td>
<td>Motor Driver Three Phase SO20 L6234PD</td>
<td>Farnell</td>
<td>1467717</td>
<td>12.490</td>
</tr>
<tr>
<td>U15_1</td>
<td>Current Shunt Monitor AD8210</td>
<td>Farnell</td>
<td>1274226</td>
<td>4.600</td>
</tr>
<tr>
<td>U15_2</td>
<td>Current Shunt Monitor AD8210</td>
<td>Farnell</td>
<td>1274226</td>
<td>4.600</td>
</tr>
<tr>
<td>U16_1</td>
<td>Current Shunt Monitor AD8210</td>
<td>Farnell</td>
<td>1274226</td>
<td>4.600</td>
</tr>
<tr>
<td>U16_2</td>
<td>Current Shunt Monitor AD8210</td>
<td>Farnell</td>
<td>1274226</td>
<td>4.600</td>
</tr>
<tr>
<td>U17_1</td>
<td>Current Shunt Monitor AD8210</td>
<td>Farnell</td>
<td>1274226</td>
<td>4.600</td>
</tr>
<tr>
<td>U17_2</td>
<td>Current Shunt Monitor AD8210</td>
<td>Farnell</td>
<td>1274226</td>
<td>4.600</td>
</tr>
</tbody>
</table>
A.4. List of test points
<table>
<thead>
<tr>
<th>Test point group</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>TST1</td>
<td>+VBAT</td>
</tr>
<tr>
<td>TST2</td>
<td>GND</td>
</tr>
<tr>
<td>TST3</td>
<td>+5 V</td>
</tr>
<tr>
<td>TST4</td>
<td>+3.3 V</td>
</tr>
<tr>
<td>TST5</td>
<td>R1</td>
</tr>
<tr>
<td></td>
<td>R2</td>
</tr>
<tr>
<td>TST6</td>
<td>EN1_1</td>
</tr>
<tr>
<td></td>
<td>IN1_1</td>
</tr>
<tr>
<td></td>
<td>EN2_1</td>
</tr>
<tr>
<td></td>
<td>IN2_1</td>
</tr>
<tr>
<td></td>
<td>EN3_1</td>
</tr>
<tr>
<td></td>
<td>IN3_1</td>
</tr>
<tr>
<td></td>
<td>EN3_2</td>
</tr>
<tr>
<td>TST7</td>
<td>EN1_2</td>
</tr>
<tr>
<td></td>
<td>IN1_2</td>
</tr>
<tr>
<td></td>
<td>EN2_2</td>
</tr>
<tr>
<td></td>
<td>IN2_2</td>
</tr>
<tr>
<td></td>
<td>EN3_2</td>
</tr>
<tr>
<td></td>
<td>IN3_2</td>
</tr>
<tr>
<td>TST8</td>
<td>la_1</td>
</tr>
<tr>
<td></td>
<td>lb_1</td>
</tr>
<tr>
<td></td>
<td>lc_1</td>
</tr>
<tr>
<td>TST9</td>
<td>la_2</td>
</tr>
<tr>
<td></td>
<td>lb_2</td>
</tr>
<tr>
<td></td>
<td>lc_2</td>
</tr>
<tr>
<td>TST10</td>
<td>Unused</td>
</tr>
<tr>
<td>TST11</td>
<td>Unused</td>
</tr>
</tbody>
</table>
B.1. Adcontrol
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

library UNISIM;
use UNISIM.VComponents.all;

entity adcontrol is
    port(
        start : in std_logic;
        reset : in std_logic;
        clk : in std_logic;
        din1, din2, din3 : in std_logic;
        cs : out std_logic;
        dout : out std_logic;
        clkout : out std_logic;
        data1, data2, data3, data4, data5, data6 : out std_logic_vector (15 downto 0)
    );
end adcontrol;

architecture behave of adcontrol is

    signal count : integer range 0 to 540;
    signal sdata1, sdata2, sdata3, sdata4, sdata5, sdata6 : std_logic_vector(15 downto 0);

begin

    process (clk, reset, start)
    begin
        if reset = '0' then
            count <= 0;
            sdata1 <= x"0000";
            sdata2 <= x"0000";
    
sdata3 <= x"0000";
sdata4 <= x"0000";
sdata5 <= x"0000";
sdata6 <= x"0000"

elsif clk'event and clk = '1' then
  if start = '1' then
    count <= 0;
    sdata1 <= x"0000";
    sdata2 <= x"0000";
    sdata3 <= x"0000";
    sdata4 <= x"0000";
    sdata5 <= x"0000";
    sdata6 <= x"0000"
    data1 <= x"0000";
    data2 <= x"0000";
    data3 <= x"0000";
    data4 <= x"0000";
    data5 <= x"0000";
    data6 <= x"0000"
  elsif start = '0' then
    if count >= 0 and count < 4 then
      count <= count + 1;
      cs <= '1';
      dout <= '0';
      clkout <= '1';
    elsif count >= 4 and count < 8 then
      count <= count + 1;
      cs <= '0';
      dout <= '0';
      clkout <= '1';
    elsif count >= 8 and count < 16 then
      count <= count + 1;
      cs <= '0';
      dout <= '0';
      clkout <= '0';
    elsif count >= 16 and count < 24 then
      count <= count + 1;
      cs <= '0';
      dout <= '0';
      clkout <= '1';
    elsif count >= 24 and count < 32 then
      count <= count + 1;
      cs <= '0';
      dout <= '0';
      clkout <= '0';
    elsif count >= 32 and count < 40 then
      count <= count + 1;
      cs <= '0';
dout <= '0';
clkout <= '1';
elsif count >= 40 and count < 48 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count >= 48 and count < 56 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 56 and count < 64 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count >= 64 and count < 72 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 72 and count < 80 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count = 80 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
sdata1(11) <= din1;
sdata3(11) <= din2;
sdata5(11) <= din3;
elsif count >= 81 and count < 88 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 88 and count < 96 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count = 96 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
sdata1(10) <= din1;
sdata3(10) <= din2;
sdata5(10) <= din3;
elsif count >= 97 and count < 104 then
count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
elsif count >= 104 and count < 112 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elsif count = 112 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
sdata1(9) <= din1;
sdata3(9) <= din2;
sdata5(9) <= din3;
elsif count >= 113 and count < 120 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
elsif count >= 120 and count < 128 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elsif count = 128 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
sdata1(8) <= din1;
sdata3(8) <= din2;
sdata5(8) <= din3;
elsif count >= 129 and count < 136 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elsif count >= 136 and count < 144 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elsif count = 144 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
sdata1(7) <= din1;
sdata3(7) <= din2;
sdata5(7) <= din3;
elsif count >= 145 and count < 152 then
count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
elsif count >= 152 and count < 160 then
  count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elsif count = 160 then
  count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
sdata1(6) <= din1;
sdata3(6) <= din2;
sdata5(6) <= din3;
elsif count >= 161 and count < 168 then
  count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elsif count >= 168 and count < 176 then
  count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elsif count = 176 then
  count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
sdata1(5) <= din1;
sdata3(5) <= din2;
sdata5(5) <= din3;
elsif count >= 177 and count < 184 then
  count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elsif count >= 184 and count < 192 then
  count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elsif count = 192 then
  count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
sdata1(4) <= din1;
sdata3(4) <= din2;
sdata5(4) <= din3;
elsif count >= 193 and count < 200 then

count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
elseif count >= 200 and count < 208 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elseif count = 208 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
sdata1(3) <= din1;
sdata3(3) <= din2;
sdata5(3) <= din3;
elseif count >= 209 and count < 216 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
elseif count >= 216 and count < 224 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elseif count = 224 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
sdata1(2) <= din1;
sdata3(2) <= din2;
sdata5(2) <= din3;
elseif count >= 225 and count < 232 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
elseif count >= 232 and count < 240 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elseif count = 240 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
sdata1(1) <= din1;
sdata3(1) <= din2;
sdata5(1) <= din3;
elseif count >= 241 and count < 248 then
count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
elif count >= 248 and count < 256 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elif count = 256 then
    count <= count + 1;
cs <= '1';
dout <= '0';
clkout <= '1';
sdata1(0) <= din1;
sdata3(0) <= din2;
sdata5(0) <= din3;
elif count >= 257 and count < 264 then
    count <= count + 1;
cs <= '1';
dout <= '0';
clkout <= '1';
elif count >= 264 and count < 272 then
    count <= count + 1;
cs <= '1';
dout <= '0';
clkout <= '0';
elif count >= 272 and count < 276 then
    count <= count + 1;
cs <= '1';
dout <= '0';
clkout <= '1';
elif count >= 276 and count < 280 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
elif count >= 280 and count < 288 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elif count >= 288 and count < 296 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
elif count >= 296 and count < 304 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '0';
elif count >= 304 and count < 312 then
    count <= count + 1;
cs <= '0';
dout <= '0';
clkout <= '1';
elif count >= 312 and count < 320 then
count <= count + 1;
end if;
cs <= '0';
dout <= '0';
clkout <= '0';
elif count >= 320 and count < 328 then
count <= count + 1;
end if;
cs <= '0';
dout <= '0';
clkout <= '1';
elif count >= 328 and count < 336 then
count <= count + 1;
end if;
cs <= '0';
dout <= '0';
clkout <= '0';
elif count >= 336 and count < 344 then
count <= count + 1;
end if;
cs <= '0';
dout <= '0';
clkout <= '1';
elif count >= 344 and count < 352 then
count <= count + 1;
end if;
cs <= '0';
dout <= '1';
clkout <= '0';
if count = 352 then
count <= count + 1;
end if;
cs <= '0';
dout <= '1';
clkout <= '1';
sdata2(11) <= din1;
sdata4(11) <= din2;
sdata6(11) <= din3;
elif count >= 353 and count < 360 then
count <= count + 1;
end if;
cs <= '0';
dout <= '1';
clkout <= '1';
elsif count >= 360 and count < 368 then
count <= count + 1;
end if;
cs <= '0';
dout <= '0';
clkout <= '0';
elif count = 368 then
count <= count + 1;
end if;
cs <= '0';
dout <= '0';
clkout <= '1';
sdata2(10) <= din1;
sdata4(10) <= din2;
sdata6(10) <= din3;
elsif count >= 369 and count < 376 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 376 and count < 384 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count = 384 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
    sdata2(9) <= din1;
    sdata4(9) <= din2;
    sdata6(9) <= din3;
elsif count >= 385 and count < 392 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 392 and count < 400 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count = 400 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
    sdata2(8) <= din1;
    sdata4(8) <= din2;
    sdata6(8) <= din3;
elsif count >= 401 and count < 408 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 408 and count < 416 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count = 416 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
    sdata2(7) <= din1;
    sdata4(7) <= din2;
    sdata6(7) <= din3;
elsif count >= 417 and count < 424 then  
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 424 and count < 432 then  
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count = 432 then  
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
    sdata2(6) <= din1;
    sdata4(6) <= din2;
    sdata6(6) <= din3;
elsif count >= 433 and count < 440 then  
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 440 and count < 448 then  
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count = 448 then  
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
    sdata2(5) <= din1;
    sdata4(5) <= din2;
    sdata6(5) <= din3;
elsif count >= 449 and count < 456 then  
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 456 and count < 464 then  
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count = 464 then  
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
    sdata2(4) <= din1;
    sdata4(4) <= din2;
    sdata6(4) <= din3;
elsif count >= 465 and count < 472 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 472 and count < 480 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count = 480 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
    sdata2(3) <= din1;
    sdata4(3) <= din2;
    sdata6(3) <= din3;
elsif count >= 481 and count < 488 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 488 and count < 496 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count = 496 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
    sdata2(2) <= din1;
    sdata4(2) <= din2;
    sdata6(2) <= din3;
elsif count >= 497 and count < 504 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
elsif count >= 504 and count < 512 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '0';
elsif count = 512 then
    count <= count + 1;
    cs <= '0';
    dout <= '0';
    clkout <= '1';
    sdata2(1) <= din1;
    sdata4(1) <= din2;
    sdata6(1) <= din3;
elsif count >= 513 and count < 520 then
  count <= count + 1;
  cs <= '0';
  dout <= '0';
  clkout <= '1';
elsif count >= 520 and count < 528 then
  count <= count + 1;
  cs <= '0';
  dout <= '0';
  clkout <= '0';
elsif count = 528 then
  count <= count + 1;
  cs <= '1';
  dout <= '0';
  clkout <= '1';
  sdata2(0) <= din1;
  sdata4(0) <= din2;
  sdata6(0) <= din3;
elsif count = 529 then
  count <= count + 1;
  cs <= '1';
  dout <= '0';
  clkout <= '1';
  data1 <= sdata1;
  data2 <= sdata2;
  data3 <= sdata3;
  data4 <= sdata4;
  data5 <= sdata5;
  data6 <= sdata6;
else
  cs <= '1';
  dout <= '0';
  clkout <= '1';
end if;
count <= count;
end if;
end if;
end process;
end behave;
B.2. Parallel data
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

library UNISIM;
use UNISIM.VComponents.all;

entity paralleldata is
  port(
    start : in std_logic;
    reset : in std_logic;
    clk : in std_logic;
    data : in std_logic_vector (12 downto 0);
    adend : out std_logic;
    csb : out std_logic;
    rdb : out std_logic;
    inhb : out std_logic;
    pardata : out std_logic_vector (15 downto 0)
  );
end paralleldata;

architecture behave of paralleldata is
begin
  signal count : integer range 0 to 540 := 0;
  process (reset, clk, start)
  begin
    if reset = '0' then
      count <= 0;
      pardata <= x"0000";
    elsif clk'event and clk = '1' then
      if start = '1' then
        count <= count + 1;
      end if;
      if count = 540 then
        count <= 0;
      end if;
      case count is
      when 0 =>
        adend <= '0';
        csb <= '0';
        rdb <= '0';
        inhb <= '0';
        pardata <= x"0000";
      when 1 =>
        adend <= '1';
        csb <= '0';
        rdb <= '1';
        inhb <= '0';
        pardata <= x"0000";
      when 2 =>
        adend <= '0';
        csb <= '0';
        rdb <= '0';
        inhb <= '0';
        pardata <= x"0000";
      when others =>
        adend <= '0';
        csb <= '0';
        rdb <= '0';
        inhb <= '0';
        pardata <= x"0000";
      end case;
    end if;
  end process;
end behave;
adend <= '0';
csb <= '0';
rdb <= '0';
inhb <= '0';
count <= 0;
pardata <= x"0000";
else
  if count >= 0 and count < 4 then
    count <= count + 1;
    csb <= '1';
    rdb <= '1';
    inhb <= '1';
    adend <= '0';
  elsif count >= 4 and count < 528 then
    count <= count + 1;
    csb <= '0';
    rdb <= '0';
    inhb <= '0';
    adend <= '0';
  elsif count = 528 then
    count <= count + 1;
    adend <= '0';
    csb <= '1';
    rdb <= '1';
    inhb <= '1';
  elsif count = 529 then
    count <= count + 1;
    adend <= '0';
    csb <= '1';
    rdb <= '1';
    inhb <= '1';
    pardata(0) <= data(0);
    pardata(1) <= data(1);
    pardata(2) <= data(2);
    pardata(3) <= data(3);
    pardata(4) <= data(4);
    pardata(5) <= data(5);
    pardata(6) <= data(6);
    pardata(7) <= data(7);
    pardata(8) <= data(8);
    pardata(9) <= data(9);
    pardata(10) <= data(10);
    pardata(11) <= data(11);
    pardata(12) <= data(12);
    pardata(13) <= '0';
    pardata(14) <= '0';
    pardata(15) <= '0';
  elsif count = 530 then
    count <= count + 1;
    adend <= '1';
    csb <= '1';
    rdb <= '1';
    inhb <= '1';
  else
count <= count;
adend <= '0';
csb <= '1';
rdb <= '1';
inhb <= '1';
end if;
end if;
end if;
end process;
end behave;
B.3. Sckgen1
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

library UNISIM;
use UNISIM.VComponents.all;

entity sckgen1 is
    port(
        clkin : in std_logic;
        clkout : out std_logic;
        reset : in std_logic
    );
end sckgen1;

architecture behave of sckgen1 is

    signal count : integer range 0 to 89;

begin
    process (reset, clkin)
    begin
        if reset = '0' then
            count <= 0;
        elsif clkin'event and clkin = '1' then
            if count = 89 then
                count <= 0;
            elsif count < 45 then
                count <= count + 1;
                clkout <= '1';
            elsif count >= 45 and count < 89 then
                count <= count + 1;
                clkout <= '0';
            end if;
        end if;
    end process;
end behave;
end if;
end if;
end process;
end behave;
B.4. Serial data
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
library UNISIM;
use UNISIM.VComponents.all;

entity serialdata is
  port(
    clk : in std_logic;
    reset : in std_logic;
    data : in std_logic;
    sck : out std_logic;
    scsb : out std_logic;
    serdata : out std_logic_vector (15 downto 0)
  );
end serialdata;

architecture behavioral of serialdata is
  signal count : integer range 0 to 14 := 0;
  signal sdata : std_logic_vector (15 downto 0) := x"0000";
begin
  sck <= clk;
  process (reset,clk)
  begin
    if reset = '0' then
      count <= 0;
      sdata <= x"0000";
    elsif clk'event and clk = '1' then
      if count = 14 then
        count <= 0;
        sdata <= x"0000";
      else
        count <= count + 1;
      end if;
    end if;
  end process;
end behavioral;
if count = 0 then
  scsb <= '1';
  serdata <= sdata;
  count <= count + 1;
elsif count = 1 then
  scsb <= '0';
  count <= count + 1;
elsif count = 2 then
  sdata(12) <= data;
  count <= count + 1;
elsif count = 3 then
  sdata(11) <= data;
  count <= count + 1;
elsif count = 4 then
  sdata(10) <= data;
  count <= count + 1;
elsif count = 5 then
  sdata(9) <= data;
  count <= count + 1;
elsif count = 6 then
  sdata(8) <= data;
  count <= count + 1;
elsif count = 7 then
  sdata(7) <= data;
  count <= count + 1;
elsif count = 8 then
  sdata(6) <= data;
  count <= count + 1;
elsif count = 9 then
  sdata(5) <= data;
  count <= count + 1;
elsif count = 10 then
  sdata(4) <= data;
  count <= count + 1;
elsif count = 11 then
  sdata(3) <= data;
  count <= count + 1;
elsif count = 12 then
  sdata(2) <= data;
  count <= count + 1;
elsif count = 13 then
  sdata(1) <= data;
  count <= count + 1;
elsif count = 14 then
  sdata(0) <= data;
  count <= 0;
end if;
end if;
end process;
end behavioral;
B.5. Multiplex2
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

library UNISIM;
use UNISIM.VComponents.all;

entity multiplex2 is
    port(
        data1 : in std_logic_vector (15 downto 0);
        data2 : in std_logic_vector (15 downto 0);
        dataout : out std_logic_vector (15 downto 0)
    );
end multiplex2;

architecture behave of multiplex2 is
    signal sel : std_logic := '1'; --if mode = 0 selects serial data, if mode = 1 selects parallel data;

begin
    dataout <= data1 when sel = '0' else data2;
end behave;