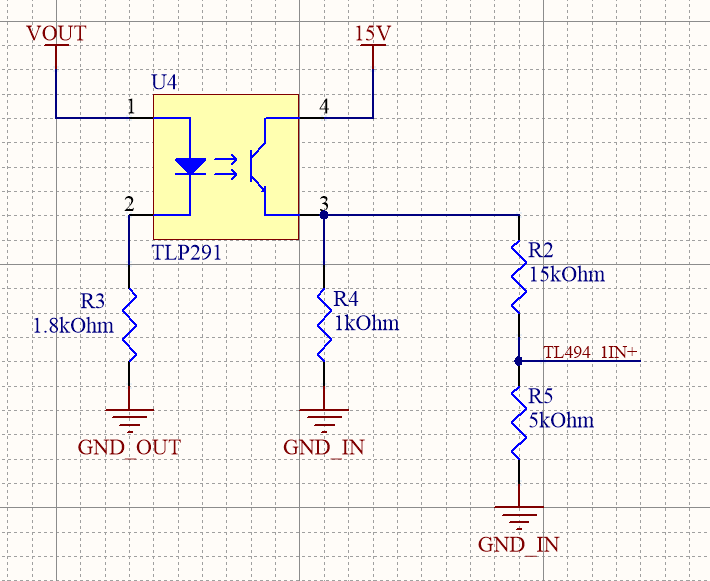
**Analog Optocoupler**

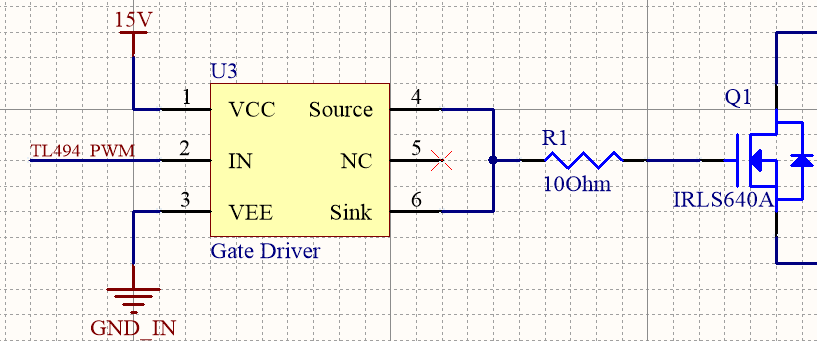
Since we need full isolation between input and output sides, we need to send the output voltage information to input side, where controller is placed, in an isolated way. We used TLP291 analog optocoupler to transfer voltage level to controller. Circuit schematic of the optocoupler is given in Figure A1. Depending on the output voltage level, optocoupler LED sends light to photoreceiver and voltage is read by controller. Supply voltage of the photoreceiver is taken from 15V output of buck converter IC. 15kΩ and 5kΩ resistors divides output voltage according to voltage reference of the controller.



**Figure A1. Circuit schematic of analog optocoupler**

**Gate Driver**

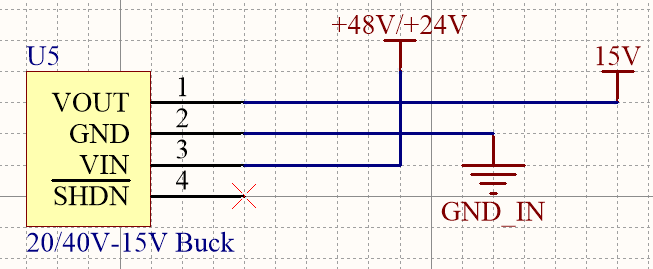
Analog controller cannot provide enough power to turn MOSFET on. We used ZXGD3009DY as a low side gate driver. It provides up to 40V 1A output current which is enough for our MOSFET application. Supply voltage is taken from output of the buck converter. Controller PWM output signal is connected to input of the driver and output is connected to gate of the MOSFET. Connections of the driver is given in Figure A2.



**Figure A2. Circuit schematic of gate driver**

**Buck Converter**

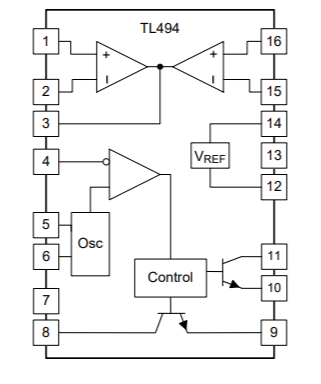
Maximum VGS voltage of the MOSFET is 20V. To obtain gate driving voltage and to supply analog controller and optocoupler, we used D36V6F15 DC/DC voltage converter. Selected buck converter can convert 15.2-50V to 15V. Input of the converter is connected to main input of the circuit and between 24-48V. Output of the buck converter is fixed and 15V. Circuit schematic of the converter is given in Figure A3.



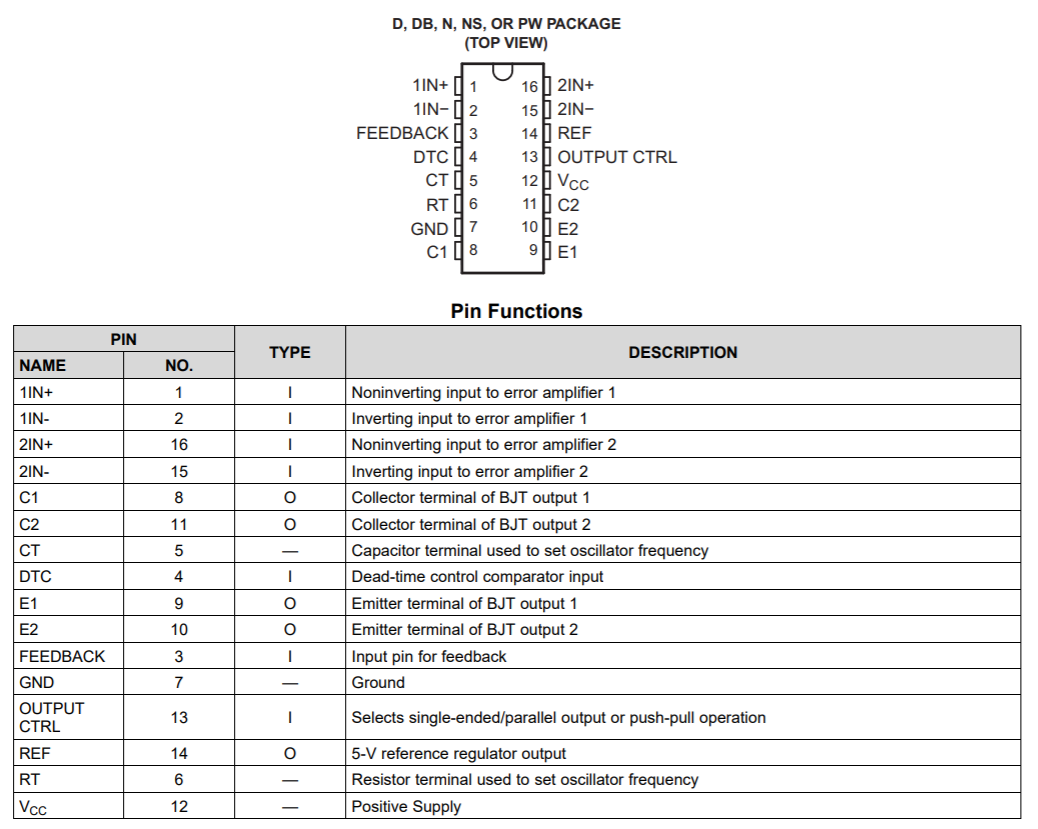
**Figure A3. Circuit schematic of buck converter**

**Controller Design**

As a PWM generator and closed loop feedback control, analog controller TL494 is chosen because of its useful properties, which is listed in this part. The design is done using the datasheet of Texas Instruments as a guide. The simplified block diagram and pin layout for this controller can be seen in Figure A4-A5.



**Figure A4. Simplified block diagram of TL494**



**Figure A5. Pin layout of TL494 and their functions**

1. Oscillator

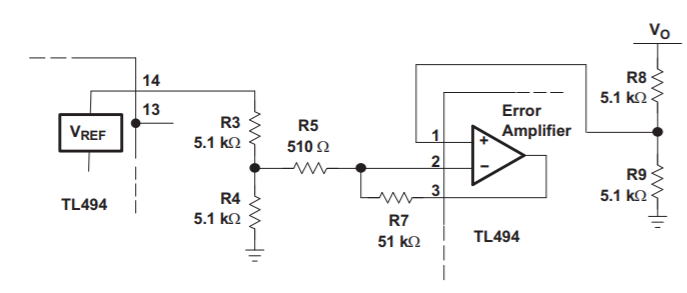
TL494 allows up to 300kHz oscillation frequency. We have decided to use 40kHz, which will be set with connecting a resistor and capacitor at RT and CT pins. Values of these will be selected using the following formula:

‬RT=25kΩ

CT=1nF

1. Closed loop feedback control

TL494 have 2 error amplifiers, voltage and current feedback. Voltage feedback will be used in this project. Typical design of this part can be seen in Figure A5. R7/R5 ratio determines the gain, and higher gain makes the response faster and error smaller. However, it also decreases stability. Therefore, a gain of 10 will be chosen.



**Figure A5. Error amplifier of TL494**

R3=1kΩ

R4=1kΩ

R5=500Ω

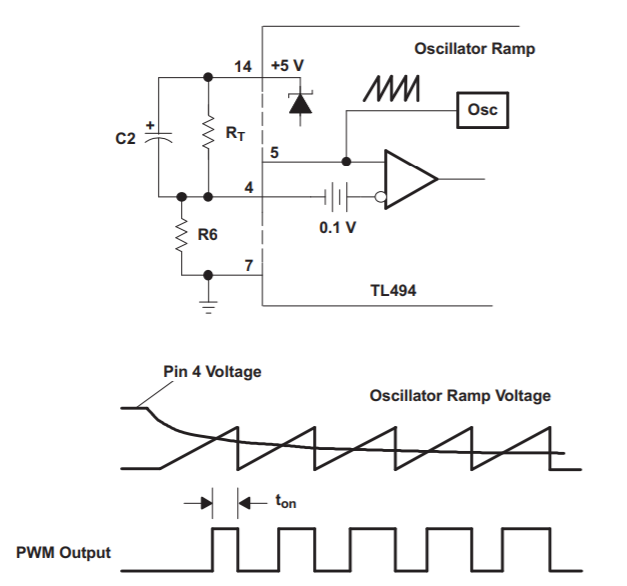
R7=5kΩ

R8= 15kΩ (shown in the TL291 circuit schematic)

R9=5kΩ (shown in the TL291 circuit schematic)

1. Dead Time Control and Soft Start

TL494 allows to set a minimum dead time control, which can be used to limit duty cycle. DTC pin can be adjusted from 0 V to 3.3 V to achieve such limit. This is very helpful for forward converter, since duty cycle should be limited such that there is enough time to reset the transformer. In our case, N1/N3 is 1 and maximum duty cycle of 0.5 is allowed.



**Figure A6. Soft start circuit**

The controller also allows for a simple soft starting circuit using DTC pin. A typical circuit is given in Figure A6. Here, R6 and RT will act as a voltage divider and determine the dead time limit. In our case, they will be selected such that Pin 4 voltage is 1.6 V. From voltage division, R6/(RT+R6) should be equal to 1.6/5 and we have 0.5 duty cycle limit.

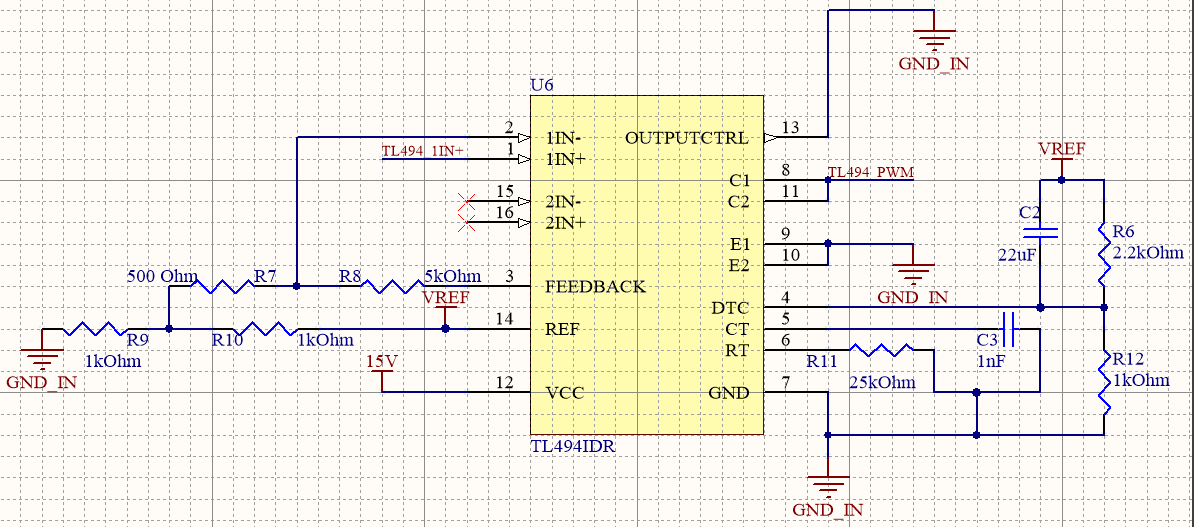
For a soft starting at 40-kHz, C2=22uF is selected and it provides 22ms soft start time which corresponds 880 switching cycle.

RT=2.2kΩ

R6=1kΩ

C2=22uF

General circuit schematic of the controller is given in Figure A7.



**Figure A7. Circuit schematic of TL494 controller**