ON Semiconductor®

Test Procedure for NCV7520GEVB Evaluation Board

This document serves as instructions to test the functionality of the demonstration board for the NCV7520 FLEXMOS Hex Low Side MOSFET Pre-Driver part. This board has many onboard components, a full Arduino UNO, a computer GUI, and its connected loads, so a test procedure is helpful to ensure full functionality

Equipment:

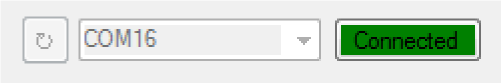
- NCV7520 Demo Board
- Two banana jack cables
- 12-24V, 1A DC power supply
- USB A to B cable
- PC running NCV7520 Demonstration Software

Setup

1) Ensure all jumpers are populated. All GATx and DRNx jumpers are populated, VCC and 3V3 jumpers are populated, VIN_RAW is populated, VINDUT should be shorting the center pin with the VIN pin, VDD should short the center pin with 5V.
2) Attach banana jacks to PWRGND and VLOAD jacks on board
3) Attach USB A to B cable from the onboard Arduino to the PC running the demo software

Startup

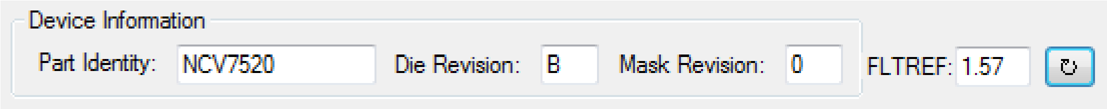
1) Power on DC power supply, set to 12V.
2) Run NCV7520 Demo Software on PC
3) Press and release ARDUINO RESET button on board, RSTB and ENB indicator lights with toggle
4) Click Refresh button to the left of the “Connect” button
5) Click drop-down menu next to “Connect” button. The COM port that the Arduino is connected to should be present, if not repeat step 3 and 4.
6) Select correct port and hit connect, the button will turn from red to yellow and if a successful connection is made will turn green and say “Connected” and the drop down menu will be disabled.



Transmitting Data, Update Device Information

1) With the above steps completed the board is ready to receive data from the GUI. On startup of the GUI the compiled SPI frame should be 0x1, the command to turn all gates off and all channels in “latch off” mode. Click “Send Data” to transmit this frame.
2) The board will not change its appearance, but a hex value will be added to the “Received” column of the “SPI Traffic” section of the GUI.
3) That first transmission should be the Device Revision Information SPI frame the part is programmed to send after the first transmission.
4) This is automatically interpreted and the “Device Information” section of the GUI is updated.

5) If the SPI Received value populated into the SPI Traffic register is not one 4 bit hex value (ex. 0xC111) click “Device RESET” in the bottom right corner of the GUI and try transmitting 0x1 again.
6) Additionally, click the refresh button next to FLTREF to read the voltage at the FLTREF pin on the DUT.

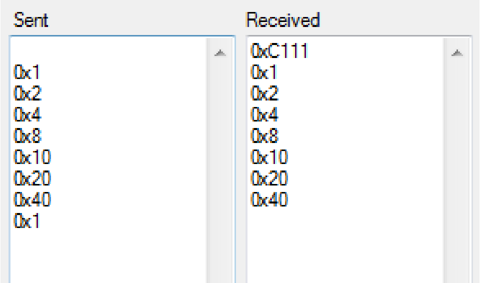


Device Information

Part Identity: NCV7520 Die Revision: B Mask Revision: 0 FLTREF: 1.57

SPI Control

1) Select “R0: Gate and Mode Select” under the “SPI Control” tab in the main section of the GUI.
2) Check the “Gate” check box in the CH0 section leaving “Latch Off” selected as is.
3) Click “Send Data.” The SPI Traffic section will be updated with the new sent frame and what the GUI received from the DUT.
4) This part’s SPI transmissions are “one frame behind” what is sent. This new entry in the received column is the information pertaining to the transmission of 0x1 we sent previously, and should have echoed that transmission. The GUI is designed to stagger the SPI traffic window so that the sent and received information can be more easily interpreted. While just sending commands to turn on and off gates, these should all be echoes, otherwise is an error.
5) Uncheck the CH0 gate box and check CH1 and click “Send data.” Repeat for all CHx having only one gate turn on at a time. Following the transmission to turn on the gate of CH5, uncheck the box and transmit 0x1, turning off all gates. The on-board LEDs labeled “GAT[x]” will light up correspondingly. The SPI Traffic register should look as follows:



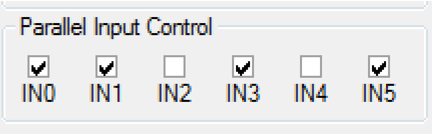
Sent	Received
	0xC111
0x1	0x1
0x2	0x2
0x4	0x4
0x8	0x8
0x10	0x10
0x20	0x20
0x40	0x40
0x1	

6) Click “Reset SPI Traffic” to clear values.
7) Turn on all CHx by selecting all of their gates and clicking “Send Data.” The hex frame should be 0x74 and all GAT[x] LEDs should be on.
8) Select register R1 and it should compile 0x6001 automatically. Click “Send Data.”
9) Repeat for R2 and R3. Their frames should be 0x429E and 0x607F respectively. These should all have responses echoing their transmissions.
10) Return to R0 and click “Send Data” twice with a brief pause in between (this keeps the GUI from failing to interpret two strings of data too close together and jumbling them up.) When returning to the Gate Control register the check boxes are reset to their initial off values.

Parallel Control

1) Click the check box labeled IN0 on the section labeled “Parallel Input Control.” The LEDs next to “IN0” and “GAT0” should turn on.

2) Leave IN0 checked and repeat for each INx box checking periodically that the correct LEDs are turning on.
3) Uncheck all INx boxes. This is important because these will override the PWM control and the SPI control.

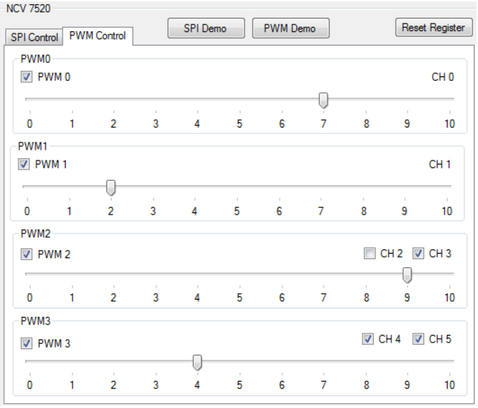


Parallel Input Control

☑ IN0 ☑ IN1 □ IN2 ☑ IN3 □ IN4 ☑ IN5

PWM Control

1) Select the "PWM Control" tab.
2) Check PWM 0 corresponding to CH0. At first no lights should turn on.
3) Drag the slider from 0 to 10 and back to 0 noticing that the LED gets brighter until it is at full at 10 and turn off at 0.
4) Repeat for PWM 1.
5) Check PWM 2, GAT2 and GAT3 LEDs will turn on, click the slider that is at 0 to turn them off.
6) Slide from 0 to 10 as before noticing how both channels are affected.
7) Uncheck CH2 and repeat 6. Check CH2 and uncheck CH3 and repeat 6.
8) Uncheck PWM 2Repeat steps 5, 6, and 7 for PWM 3.



☑ PWM 0 CH 0

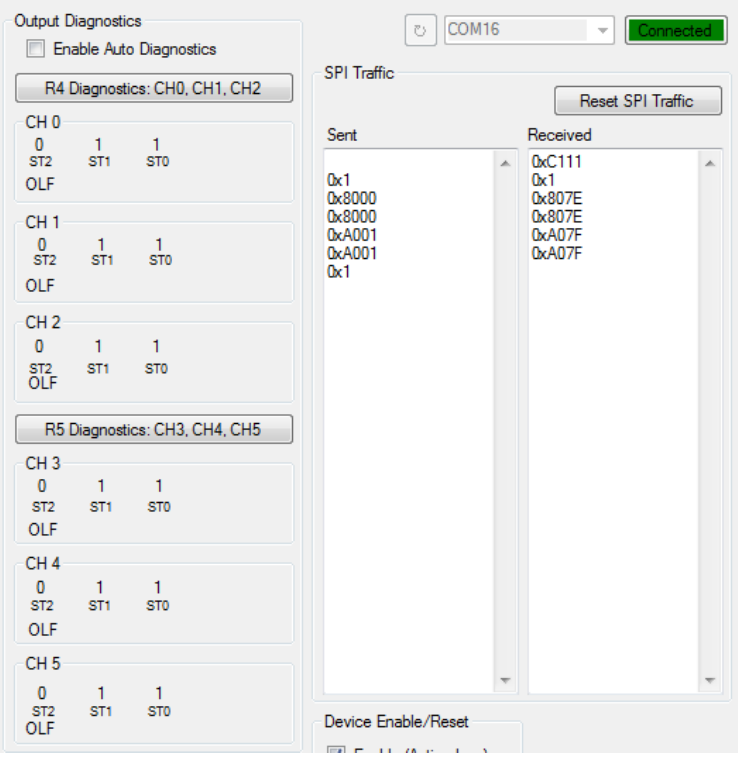
☑ PWM 1 CH 1

☑ PWM 2 □ CH 2 ☑ CH 3

☑ PWM 3 ☑ CH 4 ☑ CH 5

Output Diagnostics

1) Return to the SPI Control tab. Ensure R0 is selected and click "Reset Register" and "Send Data" to turn all gates off.
2) Click "R4 Diagnostics: CH0, CH1, CH2". 0x8000 will be sent. At first nothing will change, due to the "one frame behind" property of the part. Click R4 Diagnostics again.
3) This time a valid response should be received and the CH0, CH1, and CH2 diagnostic frames should be updated. They should all report 011 OLF at this point.
4) Repeat for R5.
5) Occasionally the SPI Received will be jumbled and the GUI will have trouble



□ Enable Auto Diagnostics

COM16

CH 0
0 ST2 1 ST1 1 ST0
OLF

CH 1
0 ST2 1 ST1 1 ST0
OLF

CH 2
0 ST2 1 ST1 1 ST0
OLF

CH 3
0 ST2 1 ST1 1 ST0
OLF

CH 4
0 ST2 1 ST1 1 ST0
OLF

CH 5
0 ST2 1 ST1 1 ST0
OLF

```
Sent        Received
0x1         0xC111
0x8000      0x1
0x8000      0x807E
0xA001      0x807E
0xA001      0xA07F
0x1         0xA07F
```

interpreting and therefore will not update the diagnostic information, requesting diagnostics by clicking R4 or R5 buttons again will clear this up.

6) Check “Enable Auto Diagnostics”
7) Wait a few seconds and 0x8000 and 0xA001 will have been sent automatically several times. This is the only way to get the GUI to update the diagnostics automatically, they are kept in the SPI traffic window so that any transmission errors can be seen diagnosed.
8) Clicking “About” will open a small window explaining the diagnostic codes and their priorities.

Demo Modes

1) Two demo modes are available in the GUI. These are run in the Arduino’s program rather than in the GUI’s processing. This was implemented from a version of the Arduino script developed to serve as a fully standalone demo board for in-house use. This means there is nothing on the GUI showing that the demo is running. Upon button click the sequence will run once and will not repeat.
2) Click SPI Demo. The sequence is as follows: 000001, 000010, 000100, 001000, 010000, 100000, 010000, 001000, 000100, 000010, 000001, 000011, 000111, 001111, 011111, 111111, 011111, 001111, 000111, 000011, 000001, 000000, 111111, 000000
3) Click PWM Demo. The sequence pulses from low to high and back to low for CH0, CH1, CH2, CH3, CH4, CH5, CH4, CH3, CH2, CH1, CH0, and then all channels at a slower rate.

If all instructions were able to be completed and the board and GUI performed as descried, the board is fully functional.