ON Semiconductor

Is Now

To learn more about onsemi™, please visit our website at
www.onsemi.com

NCP81111 Programming Application Note

The NCP81111 is a mixed signal controller with an I^{2}C programming interface. Once installed on an application board the IC can be power up and configured using a custom Windows GUI available from ON Semiconductor (Figure 1). The GUI uses an off the shelf I^{2}C to USB translator to communicate with the IC (Figure 2). The I^{2}C to USB device is powered by the USB port. ON also offers a Sample Programming board. This board has a socket on it and can configure the device before it is installed in the application board (Figure 3).

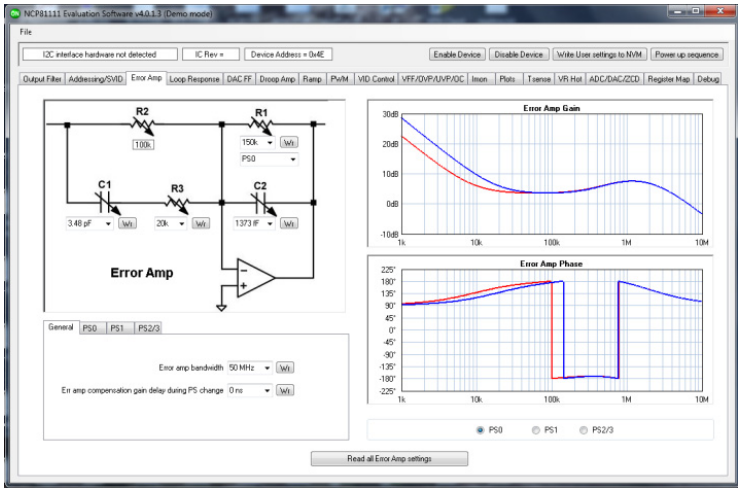


Figure 1. NCP81111 GUI

The I^{2}C to USB device should be plugged into the the NCP81111 sample programming board. The IS2 to USB device provides power programming board to power up the IC with 5 V for programming. The power should be disabled when installing or removing the part from the socket. The controller must be powered up and enabled in order for the device to be programmed.

The user can load a preconfigured configuration text file using the File load user registers function in the top left corner of the GUI. The user can also make adjustments in the GUI and configure the device after it has been installed in the application board as long as other devices on the bus do not interfere with the I^{2}C to USB communications. Table 1 shows a typical user configuration file.

The I^{2}C to USB device is available from these websites:

- http://www.robot−electronics.co.uk
- http://www.robotshop.com
- http://www.ftdichip.com/Drivers/VCP.htm

Select the windows 64bit Virtual com port driver from this web page for Windows 10.

ON Semiconductor®

www.onsemi.com

APPLICATION NOTE

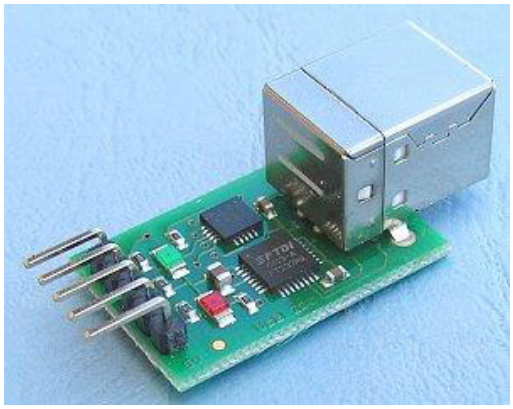


Figure 2. I^{2}C to USB Device

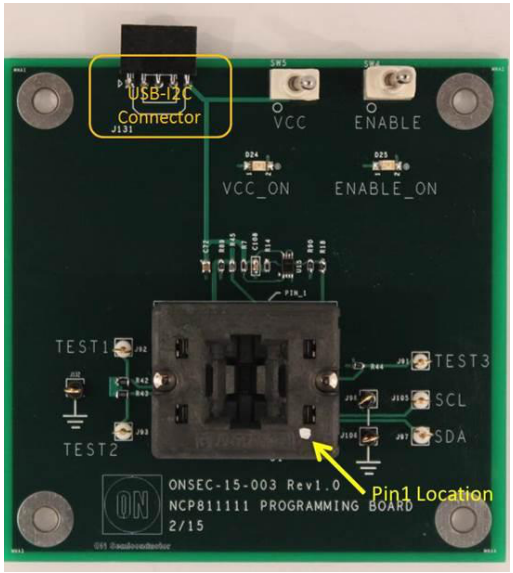


Figure 3. NCP81111 Sample Programming Board

Table 1. TYPICAL USER CONFIGURATION TEXT FILE

Register	Data	Description
005	0x02	Protocol ID
017	0x5F	VR ICC Max
018	0x64	VR Temp Max
019	0x14	VR SR Fast
020	0x05	VR SR Slow
021	0x79	VR vboot
022	0xFB	VR Vout Max
023	0x79	VR VID
024	0x02	VR Power State
025	0x00	VR VID offset
026	0x00	VR Multi VR config
027	0x35	VR Scratch Pad
030	0x00	VR Ramp Mode
031	0x02	VR Slow SR Select
040	0x00	User NVM Spare 1
041	0x00	User NVM Spare 2
042	0x00	User NVM Spare 3
043	0x00	User NVM Spare 4/Addr Offset
044	0x19	Ramp Current PS1
045	0x19	Ramp Current PS23
050	0xC4	VR Config 1
051	0x20	VR Config 2
052	0x20	VR Config 3
053	0x08	SVID Timeout Config
054	0x28	VR Slew Rate Fast Up
055	0x28	VR Slew Rate Fast Down
056	0x0A	VR Slew Rate Slow Up
057	0x0A	VR Slew Rate Slow Down
058	0x05	VR Slew Rate Soft Start
059	0x00	DACFF A Coeff Lo
060	0x00	DACFF A Coeff Hi
061	0x00	DACFF B Coeff
062	0x28	OVP Config 1
063	0x72	OVP Config 2
064	0xDA	PWM Config 1
065	0x26	PWM Config 2
066	0x4D	PWM Config 3
067	0x16	PWM Config 4
068	0x33	PWM Config 5
069	0x06	Error_Comp_PS Delay/PWM Config 6
070	0x18	PWM Config 7
071	0xC8	PWM Config 8
072	0x20	PWM Config 9
073	0x3F	PWM Config 10
074	0x1C	Ramp Config 1
075	0x1C	Ramp Config 2
076	0x1C	Ramp Config 3
077	0x03	Ramp Config 4
078	0x19	Ramp Config 5
079	0x12	Droop Config 1
080	0x05	Droop Config 2
081	0xF5	IMON Config 1
082	0x5C	IMON Config 2
083	0x01	IMON Config 3
084	0xF6	IMON Config 4
085	0x16	OC Config 1
086	0x06	OC Config 2
087	0x04	OC Config 3
088	0x55	Err Amp Int Config 1
089	0x07	Err Amp Int Config 2
090	0x1C	Err Amp Int Config 3
091	0x01	Err Amp Int Config 4
092	0x0C	Err Amp Int Config 5
093	0x13	Err Amp Int Config 6
094	0x13	Err Amp Int Config 7
095	0xB5	RC Comp Config 1
096	0x24	RC Comp Config 2
097	0xFB	Phase Shed Timout/RC Comp Config 3
098	0x94	VFF Config 1
099	0x00	VFF Config 2
100	0xDE	Tsense Config 1
101	0x00	Tsense Config 2
102	0x18	Tsense Config 3
103	0x3C	Tsense Config 4
104	0x03	Tsense Config 5
105	0xA2	Tsense Config 6
106	0x02	Tsense Config 7
107	0x28	Tsense Config 8
108	0xFD	Tsense Config 9
109	0x03	Tsense Config 10
110	0x00	Phase Double/Disable ZCD/ADC Config
111	0x0C	DAC Fine Trim PS0
112	0x00	DAC Fine Trim PS1
113	0x00	DAC Fine Trim PS23
114	0x23	Droop Temp EN/ZCD Offset Trim
115	0x00	Imon Offset PS1
116	0x00	Imon Offset PS23
117	0x00	Test1 Mux Select
118	0x00	Test2 Mux Select
119	0x00	Test3 Mux SelectA

Basic I²C Communications with the NCP81111

The NCP81111 supports the I²C protocol standard.
For a single-byte write the sequence is as follows:

S	I2C_ADDR+W	A	USER_REG_WRITE	A	USER_ADDR	A	D0	A	P

This will insert data into the register as shown:

Working Registers

Data	Address
D0	USER_ADDR

For a multi-byte write command the sequence is as follows:

S	I2C_ADDR+W	A	USER_REG_WRITE	A	USER_ADDR	A	D0	A	D1	A	D2	A	…	A	P

This will insert a block of data into the registers as shown:

Working Registers

Data	Address
D0	USER_ADDR
D1	USER_ADDR+1
D2	USER_ADDR+2
…	…

USER_NVM_RELOAD

This command will reload the User NVM settings from the NVM into the working registers. The sequence is as follows:

S	I2C_ADDR+W	A	USER_NVM_RELOAD	A	P

The command will reload all the registers at once and should complete in less than **50 µs** (worst case). This can be used to restore User settings after altering the working registers via the I²C interface. The reload is forced and does not require the settings to be configured. This command can be followed by a TEST_READ_STATUS_CURRENT command to check for read errors.

This command will update the ECC_USED_HISTORY, ECC_FAIL_HISTORY, and TIMEOUT_HISTORY registers at the bit positions corresponding to the addresses which were read (refer to the *HISTORY commands for detailed explanation).

USER_NVM_WRITE

This is the primary method for writing the User NVM settings into the NVM. The sequence is as follows:

S	I2C_ADDR+W	A	USER_NVM_WRITE	A	P

The command will write all the current User settings from the working registers into the NVM. It should complete in less than **988 ms** (worst case, 380 ms typical case). It is recommended to issue a TEST_READ_STATUS_CURRENT command following this command to check if any errors occurred during the write procedure.

Due to the long and variable write time there are three methods to determine when the procedure is complete:

1. The first method is to periodically poll the status using the TEST_READ_STATUS_CURRENT command and wait for the **LS_BUSY** flag to clear.
2. The second method is to use the Test Mux interface to route the **LS_BUSY** flag to the **TEST2** pin and monitor for the falling-edge of the **LS_BUSY** signal.
3. The third method is to simply wait **988 ms** which is long enough that the NVM will have either completed successfully or had timeout errors which forced it to complete by that time. A status check will indicate if any errors occurred.

This command will update the PROG_FAIL_HISTORY register at the bit positions corresponding to the addresses which were written (refer to the *HISTORY commands for detailed explanation).

Table 2.

Class	CMD (8-bits Hex)	Command	Description
User	01h	USER_REG_READ	Read a working set register (USER Read access required)
	02h	USER_REG_WRITE	Write a working set register (USER Write access required)
	03h	USER_NVM_RELOAD	Reload all user programmable working registers from NVM (regardless of whether **user_nvm_configured** is set or not)
	04h	USER_NVM_WRITE	Write all user programmable working registers to NVM
	05h	USER_POWER	Enable/Disable the device without power-cycling. Provides software reset capability and can control NVM, register, and auto-calibration behavior

USER_REG_READ

This command can read one or more bytes from the working register set.

The address (USER_ADDR) specified with this command is a working set address from the user address range (refer to the USER column in the Register Map). Only registers which have read access (shown as (R) or (RW) in the USER column) can be read with this command. If the command is specified with an address that does not have read access the device will respond with NA (not-acknowledge).

However, if a block of registers are read which start from a valid address, then via the auto-incrementing address point to an address that does not have read access, then for those invalid registers the return value will be 00h (zeros). The invalid registers do not stop the command, and the device will respond with an A (acknowledge). This allows a single USER_REG_READ command to read a contiguous block of data even if it spans addresses that are not valid.

Note that this command requires a repeated START sequence to change the data direction. Also, for the final byte received by the master it must signal end of data to the device by responding with a NA (not-acknowledge). This allows the device to release the data line so the master can send the STOP sequence.

If a long sequence of data is read, which due to the auto-incrementing address exceeds the allowable address range, then the device will return zero values (00h) for bytes beyond the address boundry.

For a single-byte read the sequence is as follows:

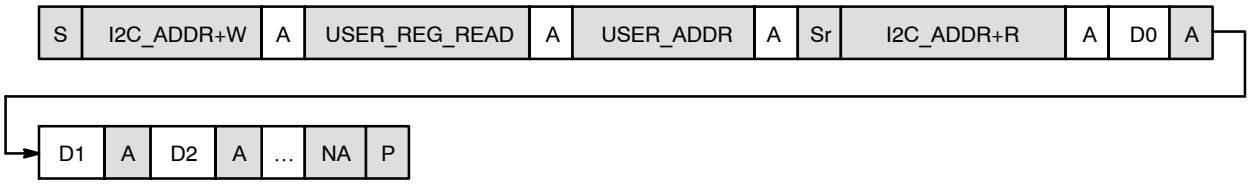
S	I2C_ADDR+W	A	USER_REG_READ	A	USER_ADDR	A	Sr	I2C_ADDR+R	A	D0	NA	P

This will read the data from the working register map as shown:

Working Registers

Data	Address
D0	USER_ADDR

For a multi-byte read command the sequence is as follows:



This will read the data from the working registers as shown:

Working Registers

Data	Address
D0	USER_ADDR
D1	USER_ADDR+1
D2	USER_ADDR+2
…	…

Notes on the Programming Sequence

The part must be powered up and enabled to program the device. The NCP81111 can be ordered as a blank part. These parts are disabled by the user_nvm_configured by bit 7 in register 50 set to zero. If the part has not been properly configured for the application or disabled with the disable bit and it has already been installed, there is the risk that the part could damage the output stage and itself when it is first enabled. In this case it is best to disable the DRMOS in the application board manually before applying power to the system. The DRVON signal is the best signal to do this with. Disconnect DRVON from the controller and tie it low so the DRMOS do not respond to the PWM and SMOD signals.

When programming the part the default I^2C address is 44h but the part can be programming to show up at other addresses. Do not program the part to show up at the 00h address you will no longer be able to talk to the part as this is not a valid address. When programming the device the user always writes to the working registers and when all the registers are configured then the user must do a NVM write command to store the data.

When writing to the I^2C address register care should be taken because this will change the address the part responds to. The next command must be sent to the new address if it is changed.

The device can be resistor programmed to show up initially at other addresses when enabled. Please see the datasheet for more details.

Notes on Using a Logic Analyzer

In the I^2C protocol the address is left-shifted 1 bit and the Read/Write bit is in the bit 0 position. For an address of 0x44 a logic analyzer will appear as 0x88. Analyzers which decode I^2C will correctly identify the address as 0x44. See the I^2C Specification for more details.
http://i2c.info/i2c−bus−specification

Notes on Memory Performance

The non-volatile memory Endurance is 100k programming cycles and the retention is 10 years at 150C. The Memory meets the JEDEC JESD47G and TSMC IP 9000 requirements.

PUBLICATION ORDERING INFORMATION

LITERATURE FULFILLMENT:
Literature Distribution Center for ON Semiconductor
19521 E. 32nd Pkwy, Aurora, Colorado 80011 USA
Phone: 303−675−2175 or 800−344−3860 Toll Free USA/Canada
Fax: 303−675−2176 or 800−344−3867 Toll Free USA/Canada
Email: orderlit@onsemi.com

N. American Technical Support: 800−282−9855 Toll Free USA/Canada
Europe, Middle East and Africa Technical Support:
Phone: 421 33 790 2910

ON Semiconductor Website: **www.onsemi.com**

Order Literature: http://www.onsemi.com/orderlit

For additional information, please contact your local Sales Representative