**smartNIV- Communication Protokoll**

Command list:

Commands are organized by Bytes. First Byte is the CMD-Byte. Following Bytes are Arguments ARG1…ARGn

Structure: CMD ARG1 ARG2 … ARGn

Test Transmission:

CMD: MATLAB: UART\_TX\_TEST = 0x01

ARGs = none

RETURN: = ACSI = Test\0

= HEX = 0x54 0x65 0x73 0x74 0x00

Turn on/off LED on Controller Electronic:

CMD: MATLAB: LED\_SET = 0x02

ARG1: = LED Off: 0x00

= LED On: 0x01

RETURN: = none

Get sensor measurement data once:

CMD: MATLAB: GET\_SENSOR\_CONT = 0x03

ARG1: MATLAB: xMin = 0x00 – 0x05

ARG2: MATLAB: xMax = 0x00 – 0x05

ARG3: MATLAB: yMin = 0x00 – 0x05

ARG4: MATLAB: yMax = 0x00 – 0x05

ARG5: MATLAB: mValues = 0x00 – 0xFF

ARG6: MATLAB: delaySwitch = 0x0000 – 0xFFFF (µSeconds)

ARG7: MATLAB: delayMeas = 0x0000 – 0xFFFF (mSeconds)

RETURN: = 36 Values = 0 – 4095 (uint16)

**To get voltage sensorValue, calculate for each of 36 Values:**

sensorValue =

((double(RETURN) / double(dacNMax)) \* dacRef) - dacRefOffset

*With: dacNMax = 4095, dacRef = 2.5, dacRefOffset = dacRef/2 = 1.25*

*Note: Measurable sensorValue Range between -1,25 V & 1,25 V*

Get sensor measurement data continuously, till stop command is send:

CMD: MATLAB: GET\_SENSOR\_CONT = 0x04

ARG1: MATLAB: xMin = 0x01 – 0x06 (uint8)

ARG2: MATLAB: xMax = 0x01 – 0x06 (uint8)

ARG3: MATLAB: yMin = 0x01 – 0x06 (uint8)

ARG4: MATLAB: xMax = 0x01 – 0x06 (uint8)

ARG5: MATLAB: mValues = 0x00 – 0xFF (uint8)

ARG6: MATLAB: delaySwitch = 0x0000 – 0xFFFF (µSec, uint16)

ARG7: MATLAB: delayMeas = 0x0000 – 0xFFFF (mSec, uint16)

RETURN: = 36 digitalValues = 0 – 4095

(uint16)

*Notizen:*

* *Was hier beim RETURN ankommt ist ein Count von ADC (Analog Digital Converter).*
* *mValues: Wenn z.B. 10, dann misst er hardwaretechnisch 10x und bildet daraus den Mittelwert und schickt dann den Mittelwert. Muss mindestens 1 sein.*
* *delaySwitch: Wartet x µSec zwischen den einzelnen Sensormessungen*
* *delayMeas: Wartet x mSec zwischen zwei Gesamtmessungen (alle Sensoren)*
* *getFrequency: Die echte Frequenz (hardwaretechnisch). Es wird die Zahl geschickt, die der Mikrocontroller gebraucht hat für die Messung*

**To get voltage sensorValue, calculate for each of 36 Values:**

sensorValue = ((double(RETURN) / double(dacNMax)) \* dacRef) - dacRefOffset

*With: dacNMax = 4095 (2^12 Bit), dacRef = 2.5 (Spannungsbereich vom ADC, Volt entspricht dem Maximalwert 4095), dacRefOffset = dacRef/2 = 1.25 (Offset, um positive und negative Werte messen zu können. Auf der Platine gibt es aber nur positive Spannungen, deswegen setzt man einfach nur den Nullpunkt in die Mitte)*

*Note: Measurable sensorValue Range between -1,25 V & 1,25 V*

*Notizen:*

* *In Wirklichkeit gab es scheinbar ein Problem, dass nur bis 2,4 gemessen werden konnte. Man wusste nicht, warum. Ist jedoch nicht so relevant, ob die Range komplett ausgeschöpft wird oder nicht*
* *Wenn mehr als 4095 kommen sollten (technisch theoretisch möglich), kann man das wegschmeißen. Dies kann ein Übertragungsfehler sein, ggf. eine Sicherheitsfrage.*

Set offset automatically by measurement through Controller Electronics:

CMD: MATLAB: SET\_OFFSET\_MEASURE = 0x05

ARGs: = none

RETURN: = 36 digitalValues = 0 – 4095

(uint16)

*Notizen:*

* *Offset-Values, wie sie zum Zeitpunkt der Messung sind, werden als RETURN-Wert zurückgegeben. Gibt den Spannungszustand des Patches zurück. Und gibt auch die Werte, also die Differenz an, die notwendig war, um die Spannung auf 0 zu setzen. D.h. danach sind sie 0. Wenn man also direkt danach nochmal Messen würde, wäre das Ergebnis 36x 0.*

**To convert digitalValue to analogValue, calculate for each of 36 Values:**

analogValue =

( (double(digitalValue) / double(dacNMax)) \* dacRef ) - dacRefOffset

*With: dacNMax = 4095, dacRef = 2.5, dacRefOffset = dacRef/2 = 1.25*

*Note: Measurable sensorValue Range between -1,25 V & 1,25 V*

Set offset manually by sending offset values to Controller Electronics:

CMD: MATLAB: SET\_OFFSET\_MANUAL = 0x06

ARGs: = 36 digitalValues = 0 – 4095

(uint16)

*Notizen:*

* *Man kann das Patch in diesem Zustand programmieren.*

**To convert digitalValue from analogValue calculate:**

digitalValue: round( ( (analogValue + dacRefOffset) / dacRef ) ) \* dacNMax

RETURN: = none

Select/set voltage gain:

CMD: MATLAB: GAIN\_SELECT = 0x07

ARG1: = 0x00 – 0x07

**ARG1 to real Gain Value:**

*0x00 = 50, 0x01 = 100, 0x02 = 200, 0x03 = 400, 0x04 = 600, 0x05 = 800, 0x06 = 1000, 0x07 = 2*

RETURN: = none

*Notizen:*

* *Das ist die Verstärkung von der Brücke zum ADC. Auf der Platine wird nur die Brückenspannung gemessen (in Millivolt) und diese muss verstärkt werden. Das ADC hat trotzdem eine feste Spannung.*
* *Das Gain schiebt die Spannung in einen Bereich, der vom ADC wahrgenommen wird.*
* *Alle anderen Werte als hier beschrieben würden vermutlich ignoriert werden. Ggf. gibt es einen Default-Wert, falls ein anderer Wert eingegeben wird. Dies müsste man noch prüfen. Ggf. ist der Default-Wert 0x07 = 2.*

Stop continuous measurement:

CMD: MATLAB: MEASUREMENT\_STOP = 0x08

ARGs: = none

RETURN: = none

*Notizen:*

* *Es werden dann keine Werte mehr geschickt. Es wird aber noch gelauscht. Er behält seinen Spannungszustand und geht idle (es werden keine Messungen mehr gemacht)*

Get resulting measurement frequency:

CMD: MATLAB: GET\_SFREQ = 0x09

ARGs: = none

RETURN: MATLAB: SFreq\_count = 0x0000 – 0xFFFF (uint16)

*Notizen:*

* *Er zählt einige Messungen und vergleicht dann intern, wie lange dafür gebraucht wurde. Das Verschicken ist auch schon in diesem Kommando drin. Er misst und hat dann in der Variable den Count gespeichert und dies kann man dann abfragen.*
* *Es ist nicht klar, was in der Variable steht, bevor eine Messung gemacht wird. Einheit ist in Millisekunden.*
* *Es wird davon ausgegangen, dass der ganze Prozess der Frequenzbestimmung nicht mehr als 78 Millisekunden dauern kann.*

**To calculate Time and Frequency:**

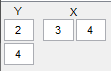
Time = SFreq\_count \* 1000 / 78125; (mSeconds)

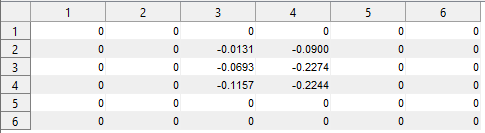
Frequency = 78125 / SFreq\_count; (Hz)

*Note: Frequency can only be obtained after a measurement*

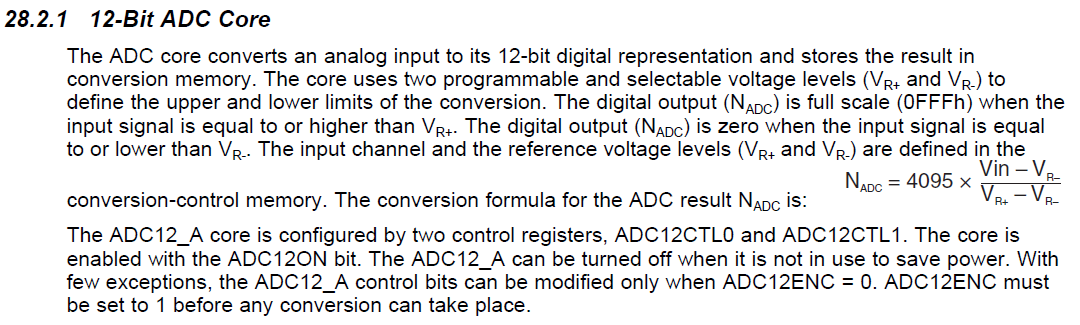
**Notes:**

**Example 1:** xMin, xMax, yMin, yMax





**ADC-Transfer function:**



Vin = Measured Voltage, VR+ = 2.5 V, VR- = 0 V