

NVA6100 PRELIMINARY DATASHEET

IMPULSE RADAR TRANSCEIVER SYSTEM

KEY FEATURES

- Single-chip impulse-based radar transceiver.
- Selectable low power RF output:
 - Low-band 1st order Gaussian pulse generator.
 - Medium-band 1st order Gaussian pulse generator.
- Programmable input amplifier gain for increased dynamic range.
- Ultra high-speed programmable sampler, giving up to 4 mm spatial resolution in normal mode.
- Interleaved sampling for sub-mm resolution.
- Staggered PRF for output spectrum smoothing and increased maximum unambiguous range.
- Iterations feature for 1/f noise suppression.
- Fully programmable sampling window:
 - 512 simultaneous sampling points.
 - 0 - 60 m detection range.
- A minimum of external components required.
- Easily adaptable to FCC/ETSI regulated UWB-masks through simple off-chip filtering.
- Focus feature for increased sensitivity without reduction in speed.
- Swept-threshold input quantizer with 13-bit resolution for increased sensitivity.
- Signal recovery through 32-bit digital integration.
- QFN32, 5x5 mm packaging.
- Powerful development tools available.

PRODUCT DESCRIPTION

NVA6100 is a fully integrated nanoscale impulse radar transceiver, designed for low-power high-performance applications. Based on Novelda's *Continuous Time Binary Value*-architecture, NVA6100 provides a low-cost highly integrated, highly robust solution for a wide range of remote sensing applications.

NVA6100 employs 32-bit digital integration and 512 parallel samplers for maximum frame depth and sensitivity, as well as a fully programmable frame offset for an extensive detection range.

Adjustable dynamic range is provided through the programmable input amplifier, while the high-speed, high-resolution sampler provides free space sampling resolution of up to 4 mm, while sub-mm resolution is attainable in interleaved sampling mode.

In a typical application setting, the NVA6100 yields unprecedented sensor performance with a minimum of external components.

APPLICATIONS

- Medical diagnostics.
- Life sign monitoring.
- Energy automation.
- Through-wall imaging.
- Tank gauging.
- Materials evaluation.
- Cable inspection.
- Object tracking.

TABLE OF CONTENTS

TABLE OF CONTENTS.....	2
1 ABBREVIATIONS.....	4
2 ABSOLUTE MAXIMUM RATINGS	5
3 ELECTRICAL CHARACTERISTICS.....	5
3.1 GENERAL OPERATING CONDITIONS.....	5
3.2 TRANSMITTER.....	6
3.2.1 IPG0, Medium-Band Pulse Generator.....	6
3.2.2 IPG1, Low-Band Pulse Generator.....	6
3.3 RECEIVER.....	7
3.4 POWER CONSUMPTION	7
4 PIN ASSIGNMENT	8
4.1 TERMINAL FUNCTIONS.....	8
5 CIRCUIT DESCRIPTION	10
6 SERIAL PERIPHERAL INTERFACE	11
6.1 EXTERNAL SIGNALS	11
6.2 USAGE	11
6.2.1 Command byte	11
6.2.2 Length byte	12
6.3 ADDRESS ORGANIZATION	12
6.4 ENDIANNESS.....	13
6.5 POWER-UP INITIALIZATION	13
6.6 ASYNCHRONOUS OUTPUTS	13
7 TRANSMITTER.....	14
7.1 OUTPUT PULSE SHAPES AND FREQUENCY SPECTRUMS.....	14
7.1.1 IPG0, Medium-Band Pulse Generator.....	14
7.1.2 IPG1, Low-Band Pulse Generator.....	15
8 THRESHOLDER.....	16
8.1 THRESHOLDER POWERDOWN.....	16
9 HIGH-SPEED SAMPLER	16
10 SWEEP CONTROLLER.....	17
10.1 OVERVIEW.....	17
10.2 PRINCIPLE OF OPERATION.....	17
10.3 CONFIGURING SWEEP BOUNDARIES.....	18
10.4 CONFIGURABLE STEP-SIZE.....	18
10.5 FOCUSED AVERAGING.....	18
10.6 BI-DIRECTIONAL SWEEPS.....	19

10.7	MONITORING SWEEP PROGRESS	20
11	TIMING CONTROLLER	21
11.1	DELAYED CLOCK FRAME OFFSET	21
11.2	STAGGERED PRF CONTROL	21
11.3	LFSR.....	21
12	TIMING CALIBRATION.....	22
12.1	CIRCUIT DESCRIPTION	22
12.2	EQUIVALENT CLOCK CYCLE DELAY	23
12.3	FRAMEOFFSET CALIBRATION.....	24
12.4	AVERAGE SAMPLING RATE	25
13	CONFIGURATION REGISTERS	26
13.1	CONFIGURATION REGISTER DETAILS	27
14	REFERENCE APPLICATION CIRCUIT	36
15	PACKAGE DIMENSIONS AND RECOMMENDED LAYOUT (QFN32)	37
16	ORDERING INFORMATION	38
17	CONTACT INFORMATION	38
17.1	NOVELDA CORPORATE HEADQUARTERS	38
17.2	NOVELDA R&D DEPARTMENT	38
17.3	NOVELDA TECHNICAL SUPPORT.....	38
18	DOCUMENT HISTORY	38
	DISCLAIMER.....	39

1 ABBREVIATIONS

DAC	Digital to Analog Converter
ESD	Electrostatic Discharge
FPS	Frames Per Second
HSC	High-Speed Comparator
I/O	Input / Output
LFSR	Linear Feedback Shift Register
LSB	Least Significant Byte
LSb	Least Significant Bit
MSB	Most Significant Byte
MSb	Most Significant Bit
MUR	Maximum Unambiguous Range
NA	Not Available
NC	Not Connected
PCB	Printed Circuit Board
PG	Pulse Generator
PRF	Pulse Repetition Frequency
QFN	Quad-Flat pack No leads
RF	Radio Frequency
RX	Receive(r)
SD	Sample Delay
SPD	Send Pulse Delay
SPI	Serial Peripheral Interface
TSP	Total Sequence Period
TX	Transmit(ter)
UWB	Ultra Wide Band
VGA	Variable Gain Amplifier

2 ABSOLUTE MAXIMUM RATINGS

Note that the absolute maximum ratings are stress ratings, and functional operation of the device under such conditions is not guaranteed. Long-term exposure to absolute maximum rating conditions may affect device reliability, and permanent damage may occur if these ratings are violated.

Parameter	Min.	Max.	Unit
Supply voltage core, VDDx	-0.3	1.26	V
Supply voltage I/O, VDDIO	-0.3	2.75	V
Input voltage, digital I/O	-0.3	3.6	V
Input RF level		6	dBm
Storage temperature	-65	150	°C
Reflow solder temperature		260	°C

Table 1. Absolute maximum ratings.



Caution! This is an electrostatic sensitive device. Failure to observe proper handling and installation procedures may result in performance degradation or terminal damage to the device.

3 ELECTRICAL CHARACTERISTICS

If nothing else is noted, all electrical characteristics are measured on the NVA6100 development modules (NVA-R630 and NVA-R640 respectively), with all VDDx = 1.2 V, except for VDDIO = 2.5 V, and ambient temperature $T_{amb} = 25\text{ °C}$.



Important notice. Please be advised that all data presented in this section are expected performance characteristics only, and are subject to change at any time, without further notice.

3.1 GENERAL OPERATING CONDITIONS

Parameter	Min.	Typ.	Max.	Unit
Supply voltage core, VDDx	1.14	1.2	1.26	V
Supply voltage I/O, VDDIO	2.25	2.5	2.75	V
Ambient operating temperature	0		70	°C
Main clock frequency, f_{CLK}			100	MHz
SPI clock frequency, f_{SCLK}			25	MHz
Logic input low voltage, V_{IL}	-0.3		0.7	V
Logic input high voltage, V_{IH}	1.7		3.6	V
Logic output low voltage, V_{OL}			0.7	V
Logic output high voltage, V_{OH}	1.7			V

Table 2. General operating conditions.

3.2 TRANSMITTER

3.2.1 IPG0, MEDIUM-BAND PULSE GENERATOR

Parameter	Min.	Typ.	Max.	Unit	Note
Order of Gaussian pulse approximation		1			
Lower -10 dB cutoff frequency, f_L					¹
Slow	620	660	690	MHz	
Nominal	780	845	890	MHz	
Fast	880	1060	1150	MHz	
Upper -10 dB cutoff frequency, f_H					¹
Slow	6960	7145	7300	MHz	
Nominal	9180	9550	9735	MHz	
Fast	10300	10410	10600	MHz	
Nominal output power			-53	dBm/MHz	^{2,3}
Instantaneous output amplitude					
Slow	440	470	510	mV	
Nominal	430	450	480	mV	
Fast	340	370	410	mV	
TSP for staggered PRF		65535		pulses	
Staggered PRF step resolution		31		ps	
Optimum load impedance		50		Ω	

Table 3. Medium-band pulse generator characteristics.

3.2.2 IPG1, LOW-BAND PULSE GENERATOR

Parameter	Min.	Typ.	Max.	Unit	Note
Order of Gaussian pulse approximation		1			
Lower -10 dB cutoff frequency, f_L					¹
Slow	420	435	440	MHz	
Nominal	445	450	460	MHz	
Fast	480	485	495	MHz	
Upper -10 dB cutoff frequency, f_H					¹
Slow	3115	3165	3230	MHz	
Nominal	3490	3555	3635	MHz	
Fast	3970	4065	4150	MHz	
Nominal output power			-45	dBm/MHz	^{2,3}
Instantaneous output amplitude	450	500	550	mV	⁴
TSP for staggered PRF		65535		pulses	
Staggered PRF step resolution		31		ps	
Optimum load impedance		50		Ω	

Table 4. Low-band pulse generator characteristics.

¹ The pulse generator's frequency band is programmable in 3 steps: slow, nominal and fast.

² Delivered to a single ended 50 Ω load.

³ PRF = 48 MHz.

⁴ Output amplitude of IPG1 is unaffected by bandwidth settings.

3.3 RECEIVER

Parameter	Min.	Typ.	Max.	Unit	Note
Receiver sensitivity					¹
Gain 0/1		-73		dBm	
Gain 2/3		-80		dBm	
Gain 4		-90		dBm	
Gain 5/6		-95		dBm	
Dynamic range				dB	²
Input voltage reflection coefficient, S11				dB	²
Programmable input amplifier gain range		29		dB	³
Input threshold resolution		13		bits	
Samples per frame		512		samples	⁴
Equivalent programmable sampling rate					⁵
Sampling rate 0	35.5	39.0	41.0	GS/s	
Sampling rate 1	18.0	20.0	21.0	GS/s	
Sampling rate 2		3.8		GS/s	
Sampling rate temperature coefficient					²
Sampling rate 0		-60		MS/s/°C	
Sampling rate 1		-31		MS/s/°C	
Sampling rate 2					
Minimum frame offset (trigger delay)		-350		ns	
Maximum frame offset (trigger delay)		350		ns	
Frame offset step, CoarseTune	0.90	0.95	1.10	ns	^{5,6}
Frame offset step, MediumTune	30.2	31.1	34.8	ps	
Frame offset step, FineTune		1.7		ps	
Frame offset jitter				ps	²
Frame offset temperature coefficient				% / °C	²
Optimum source impedance		50		Ω	

Table 5. Receiver characteristics.

3.4 POWER CONSUMPTION

Parameter	Min.	Typ.	Max.	Unit	Note
Power consumption		113		mW	
Power consumption, powerdown mode		27		mW	

Table 6. Power consumption characteristics.

¹ Receiver sensitivity at 10 dB SNR. Radar Settings were as follows: DACMin = 4014, DACMax = 4178, PulsesPerStep = 100, Iterations = 10.

² No or only partial data available at the time of writing.

³ Input amplifier gain is programmable in 8 steps from -6 to 23 dB.

⁴ Downsampling enables selectable frame sizes of 512, 256, 128 or 64 samples.

⁵ Both sampling rate and frame offset delays are dependent on PRF. A calibration circuit is included for measuring and calibrating these delays. Please refer to Section 12 for details.

⁶ PRF = 6 MHz.

4 PIN ASSIGNMENT

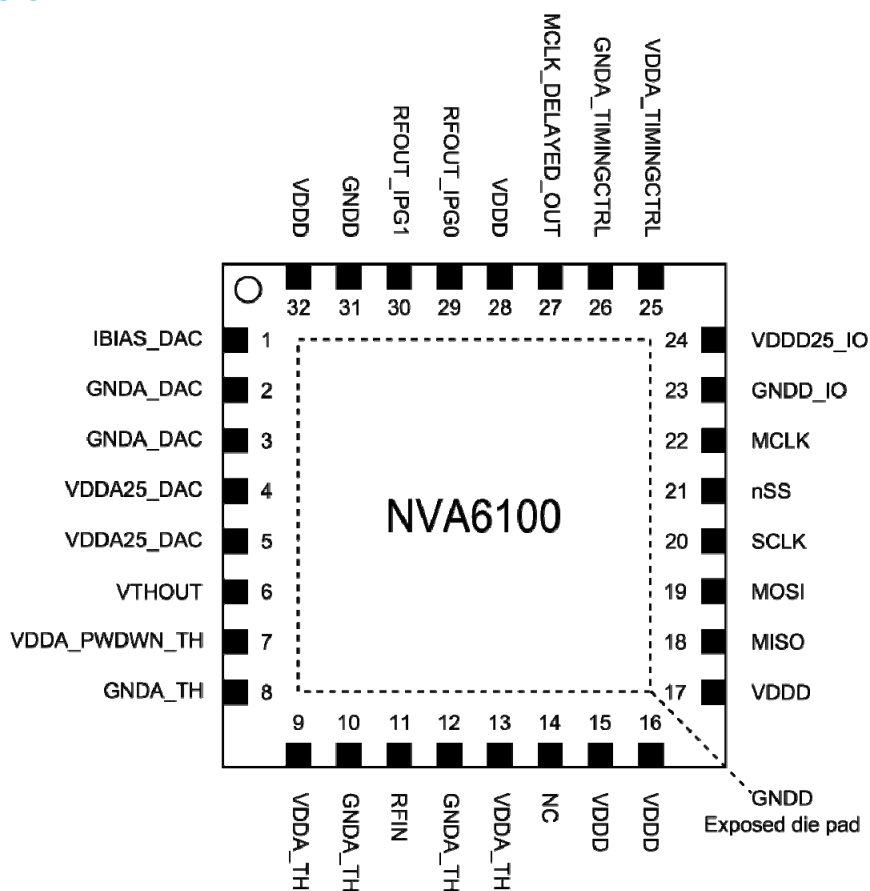


Figure 1. NVA6100 pinout, top view QFN32 package.

4.1 TERMINAL FUNCTIONS

Pin	Pin name	Pin type	Description
-	GNDD	Ground (digital)	Exposed die pad. Provides ground connection for the digital core. Must be connected to a solid ground plane.
1	IBIAS_DAC	Analog I/O	Reference current for internal DAC.
2	GNDA_DAC	Ground (analog)	DAC ground connection.
3	GNDA_DAC	Ground (analog)	DAC ground connection.
4	VDDA25_DAC	Power (analog)	DAC power supply (2.5 V).
5	VDDA25_DAC	Power (analog)	DAC power supply (2.5 V).
6	VTHOUT	Digital output	Threshold output for debugging purposes.
7	VDDA_PWDN_TH	Power (analog)	Input threshold power supply (1.2 V) for powerdown function.
8	GNDA_TH	Ground (analog)	Input threshold ground connection.
9	VDDA_TH	Power (analog)	Input threshold power supply (1.2 V).
10	GNDA_TH	Ground (analog)	Input threshold ground connection.
11	RFIN	RF input	RF input from receiving antenna.
12	GNDA_TH	Ground (analog)	Input threshold ground connection.
13	VDDA_TH	Power (analog)	Input threshold power supply (1.2 V).
14	NC	-	Not connected.
15	VDDD	Power (digital)	Digital core power supply (1.2 V).
16	VDDD	Power (digital)	Digital core power supply (1.2 V).

Pin	Pin name	Pin type	Description
17	VDDD	Power (digital)	Digital core power supply (1.2 V).
18	MISO	Digital output	SPI Master In Slave Out.
19	MOSI	Digital input	SPI Master Out Slave In.
20	SCLK	Digital input	SPI Clock input.
21	nSS	Digital input	SPI Slave Select (active low).
22	MCLK	Digital input	Master Clock input.
23	GNDD_IO	Ground (digital)	Digital I/O post-driver ground connection.
24	VDDD25_IO	Power (digital)	Digital I/O post-driver power supply (2.5 V).
25	VDDA_TIMINGCTRL	Power (analog)	Power supply (1.2 V) for analog parts of timing controller and high-speed sampler.
26	GNDA_TIMINGCTRL	Ground (analog)	Ground connection for analog parts of timing controller and high-speed sampler.
27	MCLK_DELAYED_OUT	Digital output	Clock output with programmable delay.
28	VDDD	Power (digital)	Digital core power supply (1.2 V).
29	RFOUT_IPG0	RF output	Medium-band impulse generator RF output.
30	RFOUT_IPG1	RF output	Low-band impulse generator RF output.
31	GNDD	Ground (digital)	Digital core ground connection.
32	VDDD	Power (digital)	Digital core power supply (1.2 V).

Table 7. NVA6100 terminal functions.



Important notice. The exposed die pad serves both an electrical and a thermal purpose, and **must** be connected to a solid ground plane, providing the digital ground connection for the chip.

5 CIRCUIT DESCRIPTION

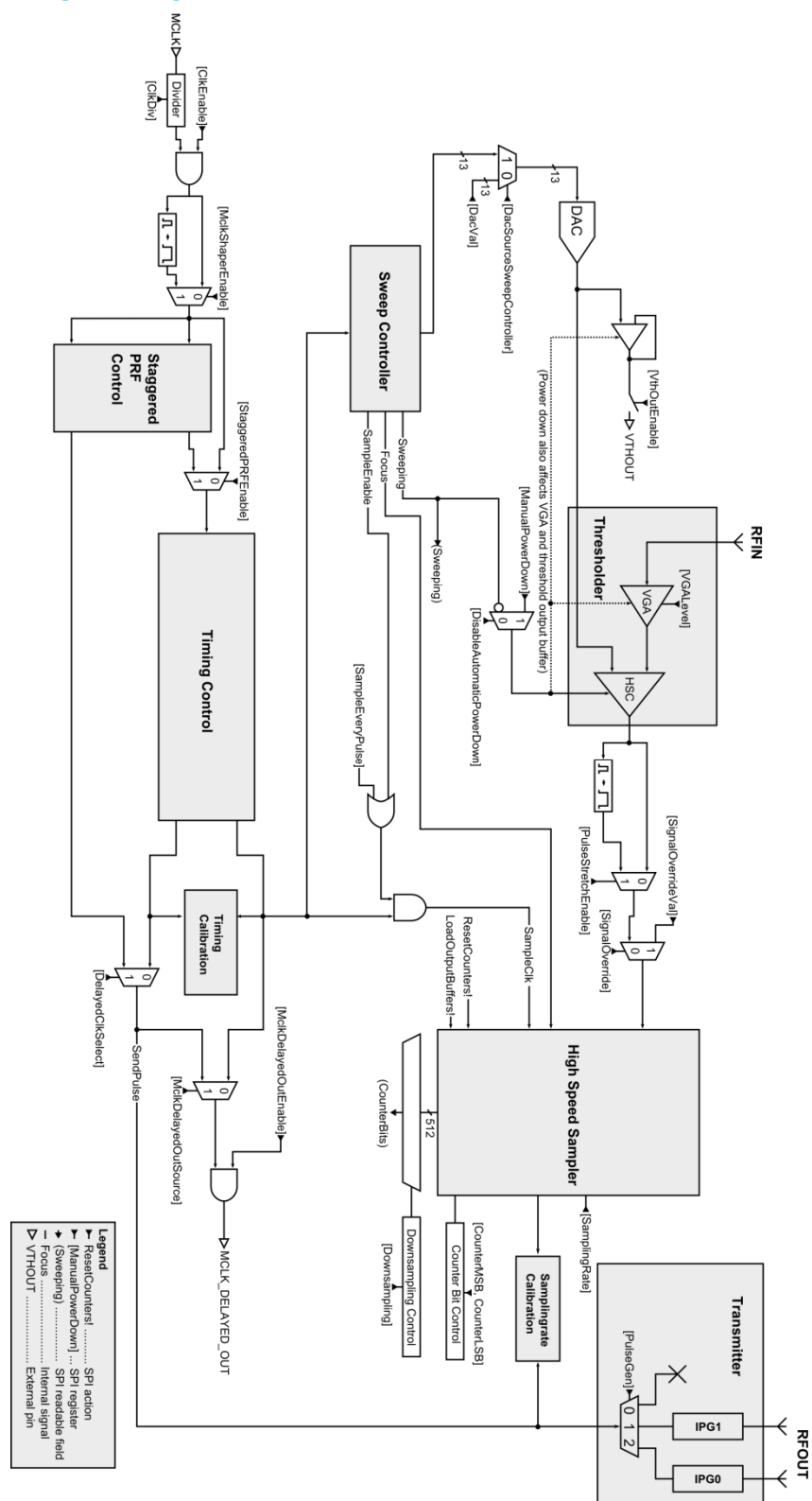


Figure 2. NVA6100 block diagram.

6 SERIAL PERIPHERAL INTERFACE

The 4-wire Serial Peripheral Interface (SPI) provides digital off-chip communication through slave mode operation in a synchronous data-link to an external controller, operating in Master Mode. It is used to read and write data to configuration registers, read the outputs from the sampler, or to execute action strobes.

6.1 EXTERNAL SIGNALS

Active high input data is transferred serially from the master through the Master Out Slave In (MOSI) input pin and read by the slave at the rising edge of the Serial Clock (SCLK) signal as long as the active low Slave Select (nSS) input pin is driven low. Output data is read by the master through the Master In Slave Out (MISO) pin on rising SCLK edges. When the chip is inactive (the nSS pin is driven high), the MISO pin is in high impedance mode. To avoid floating inputs to the master, it is recommended to use an external pull-up resistor on the MISO signal.

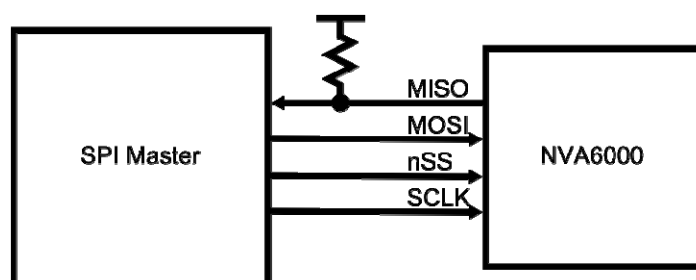


Figure 3. Example connectivity with external SPI master.

6.2 USAGE

All transmissions begin with a *command* byte and a *length* byte. Their specific meaning and format are described in the following sections. In write operations the length byte is immediately followed by a number of data bytes, and in read operations the length byte is immediately followed by a number of zero valued bytes. Action strobe operations are executed on the falling edge of the last SCLK pulse in the length byte. All transmissions are done with the most significant bit (MSb) first. Table 8 shows the different bytes and their usage. Byte 0 is transferred first, the byte 1 and so on.

Byte #	0	1	2	...	2+L
Data	Command	Length	Data[0]	Data[...]	Data[L-1]

Table 8. General usage of SPI.

6.2.1 COMMAND BYTE

The command byte contains a seven bit address field and an R/W bit, and is formatted as shown in Table 9. To execute an action strobe, the R/W field must be set to write.

Bit#	Field	Description
7 (MSb)	R/W	Select read or write operation. 0: Read operation is selected. 1: Write operation is selected.
6 downto 0	Address	Address of the field to read from or write to, or to the action strobe to execute.

Table 9. SPI command byte organization.

6.2.2 LENGTH BYTE

The length byte contains a seven bit length field and a continue bit.

Bit#	Field	Description
7 (MSb)	Continue	Enables partial readouts from fields. 0: Operation starts on the first bit in the field. 1: Operation continues from last time.
6 downto 0	Length	Number of bytes to be read or written in the current operation.

Table 10. SPI length byte organization.

The SPI can handle blocks of data at up to 127 bytes at a time. For reading or writing blocks larger than 127 bytes, the continue bit is set to one, indicating that the user intends to continue from the last read or write position. This keeps any built in address pointers from resetting. The continue bit is only used in memories, see the Table 11 for more information. R/W registers have separate read and write pointers and they are persistent over nSS. Reading and writing beyond specified sizes is not allowed. The length field must be set to zero when executing/addressing action strobes. Any additional data bytes will send 8 strobes each.

6.3 ADDRESS ORGANIZATION

There are a total of 128 different address fields, each addressable through the address part of the command byte. There are three different field types: register, memory and action strobe. Their specific attributes are described in the following section. A complete map of all available SPI fields, together with their type can be found in Section 13.1.

Register	
Continue	Not supported. Users must write the exact register length each time, but can choose to read only the first byte(s) of the register.
Read/Write	Can be both read and writable.
Data length	Actual register size may vary, but is never bigger than 127 bytes. Size of each field is specified in SPI address map.
nSS behavior	Data is persistent over nSS reset. Read/write pointers are reset over nSS reset. Aborting a write operation with nSS will leave the register with an undefined value, and the register needs to be reset. Aborting a read operation with nSS is safe.
Memory	
Continue	Yes.
Read/Write	Can be both read and writable.
Data length	0 to 127 bytes <i>per read/write cycle</i> . Total possible memory size is unlimited. Do not write beyond memory borders defined in SPI address map, as doing so will result in undefined behavior.
nSS behavior	Memory pointers are undefined after power-up. This means that the first read/write operation after power-up should be done with continue=0, resetting the internal pointer. Writing/reading from the location with length=0 and continue=0 can be used to reset memory pointers without actual memory reading/writing. Memory pointers are kept over nSS reset. Data is persistent over nSS reset. Aborting a write operation with nSS (pulling nSS in the middle of a transfer) will leave the register in an undefined state, and the register needs to be reset.

Action strobes	
Continue	No.
Read/Write	Write only (Reading from a strobe location gives undefined results).
Data length	0. Additional data bytes will issue 8 strobes each (one for each of the falling SCLK edges). This may cause undefined results.

Table 11. Address field types.

6.4 ENDIANNESS

Any integer greater than one byte in size, is transferred with the most significant byte first. Registers and memory fields can be a single integer of a given size, or a sequence of integers, maybe even of different sizes. The configuration of each field is specified in the SPI map in Section 13.1.

6.5 POWER-UP INITIALIZATION

Make sure to set nSS=1 before setting nSS=0 again and starting SPI communication. This is necessary in order to get a proper initialization of the SPI circuit. All registers, memory, memory pointers etc. will have undefined values unless otherwise noted, and must be initialized one by one.

6.6 ASYNCHRONOUS OUTPUTS

The SweepControllerStatus register contains 16 bits which can be used to monitor certain internal signals. These bits are synchronous to the MCLK pin, and not the SCLK pin as the rest of the SPI registers. For this reason they are termed asynchronous bits. The register can be read as a normal SPI register by temporarily disabling the clock signal on the MCLK pin, or the individual bits can be output as asynchronous signals on the MISO pin.

To output an asynchronous output on the MISO pin, start by transferring the Command and Length byte as normal. The MSb of SweepControllerStatus (bit #15) is now routed directly to the MISO pin, and can be monitored externally. To output bit N, toggle the SCLK pin 15-N times. For instance, to read bit # 13, the SCLK pin has to be toggled two times. To end the procedure and return the SPI to normal synchronous mode, either reset the nSS pin or toggle the SCLK the remaining N+1 times.

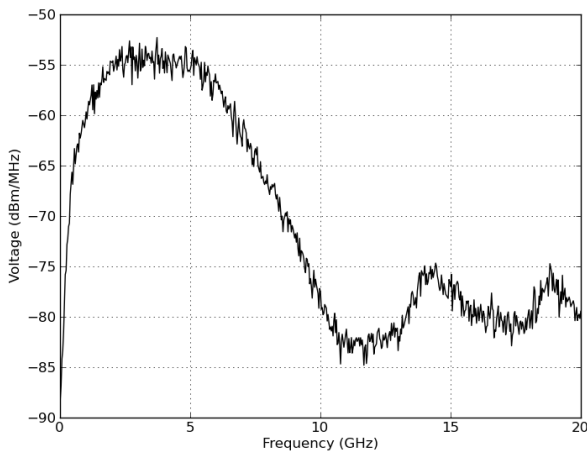
7 TRANSMITTER

NVA6100 employs two separate pulse generators, both 1th order Gaussian pulse approximation, but with different output frequency spectrums. The signal `PulseGen` in the `PGCtrl` register selects the active pulse generator. Only one pulse generator may be used at the time.

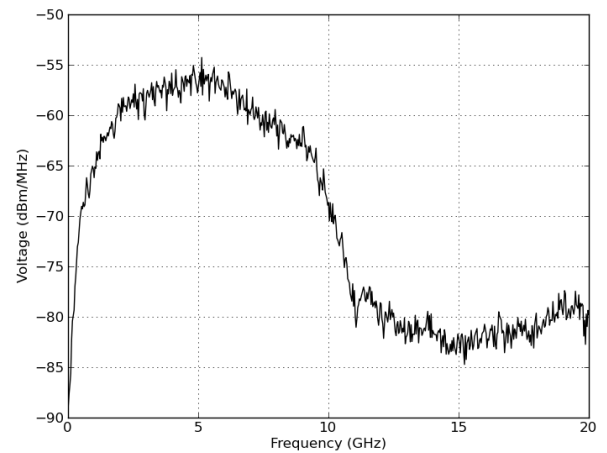
Note that the two generators do not share a common output pin, meaning that special care should be taken when designing the supporting hardware, depending on the target application. This is illustrated in the application example in Section 14.

7.1 OUTPUT PULSE SHAPES AND FREQUENCY SPECTRUMS

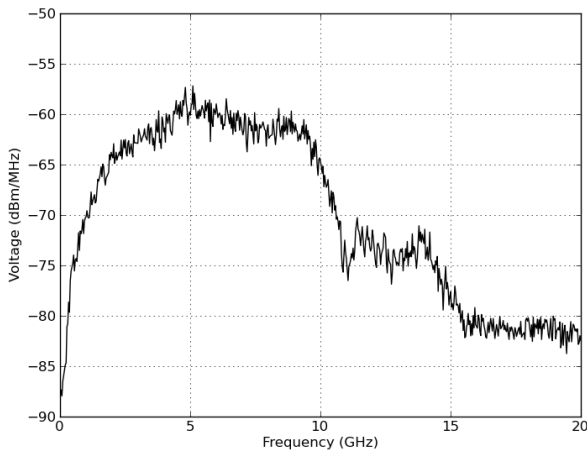
7.1.1 IPG0, MEDIUM-BAND PULSE GENERATOR



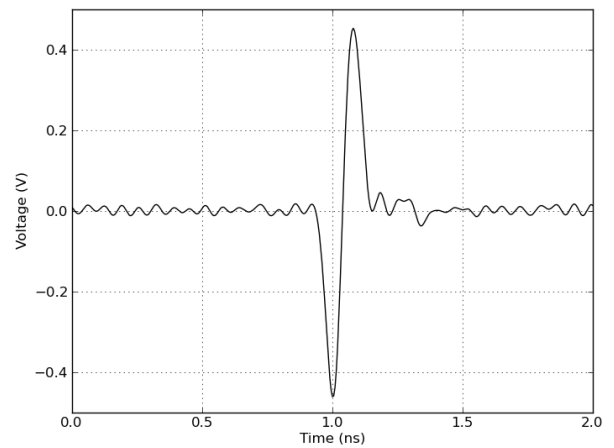
a) Typical frequency spectrum, slow delay line.



b) Typical frequency spectrum, nominal delay line.



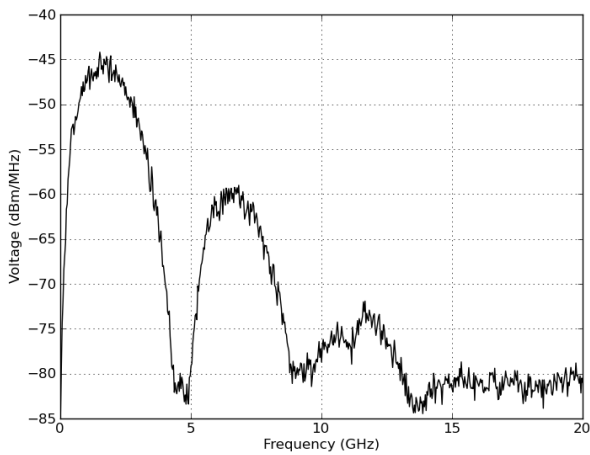
c) Typical frequency spectrum, fast delay line.



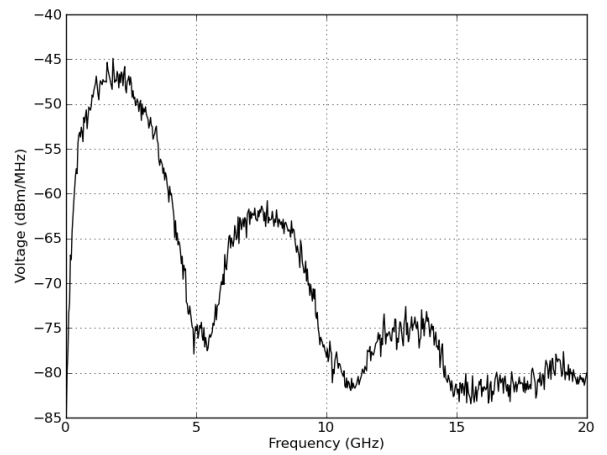
d) Example time domain output.

Figure 4. Medium-band pulse generator output.

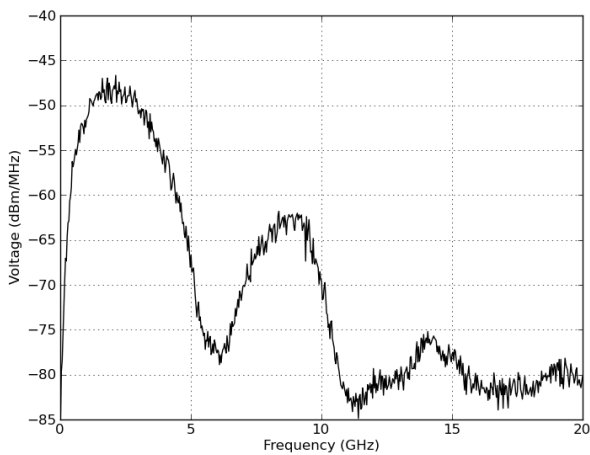
7.1.2 IPG1, LOW-BAND PULSE GENERATOR



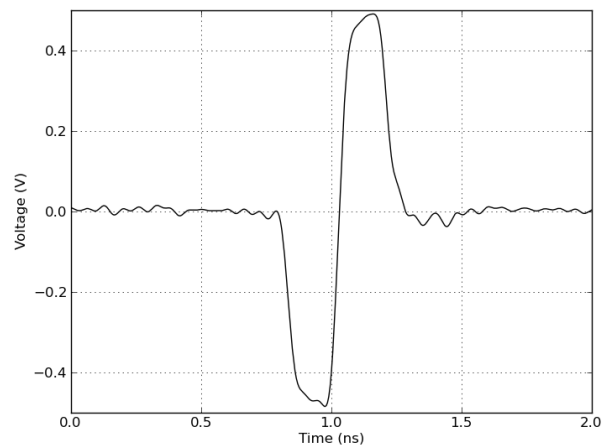
a) Typical frequency spectrum, slow delay line.



b) Typical frequency spectrum, nominal delay line.



c) Typical frequency spectrum, fast delay line.



d) Example time domain output.

Figure 5. Low-band pulse generator output.

8 THRESHOLDER

The input thresholder consists mainly of two blocks, the VGA and HSC. The VGA provides a variable input gain whereas the HSC compares the output signal from the VGA to a DC-threshold voltage and decides whether the signal is below or above this threshold in addition to providing further amplification. The output from the HSC is digital in nature as it is either high or low, although continuous in time. The input thresholder is controlled through the `ThresholderCtrl` register.

8.1 THRESHOLDER POWERDOWN

The Thresholder block contains an optional powerdown function for manual or automatic shutdown when entering power saving mode. Use of this feature is only available when supported by the surrounding hardware environment: i.e. pins 9 and 13 must be NC. The powerdown function is controlled by the `ThresholderPowerdown` register.

Note that both `ThresholderCtrl` and `ThresholderPowerdown` registers retain their values, even when the input thresholder is in powerdown mode.

9 HIGH-SPEED SAMPLER

The high-speed sampler samples the reflected incident signal at a sampling rate determined by the `SamplerCtrl` register. The output reads as a frame consisting of 512 consecutive samples, at 32 bits resolution. The high-speed sampling clock is synthesized from `MCLK` through delay lines.

10 SWEEP CONTROLLER

10.1 OVERVIEW

The Sweep Controller is a configurable state machine that allows the NVA6100 to execute sweeps independently of any external controller. The user triggers the sweep sequence by issuing the `StartRadar` command strobe. Shortly after, the `Sweeping` signal goes high, indicating that a sweep is in progress. This signal can be monitored asynchronously through the SPI, allowing the SPI master to concentrate on other tasks in the meantime. Once the sweep sequence is complete, the `Sweeping` signal goes low indicating that a new frame is ready in the sampler.

Figure 6 shows a block diagram of the Sweep Controller and its inputs and outputs. The Sweep Controller is clocked from the Timing Controller. In addition to the clock signal, there are two action strobes controlling the operation of the Sweep Controller: `StartRadar` and `ResetSweepController`. As mentioned earlier, the `StartRadar` triggers a new sweep sequence. The `ResetSweepController` action strobe aborts a sweep in progress. The DAC is controlled through the 13 bit wide `DACValue` bus. Internally, the `Sweeping` signal is used to automatically power down the Thresholder between sweeps in order to reduce the overall power consumption. See Section 8.1 for more information on configuring the NVA6100 for automatic power down. The `SampleEnable` signal enables the Sweep Controller to enable and disable the sampler when appropriate. It can, for instance, temporarily disable the High Speed Sampler while updating the `DACValue` to allow extra settling time for the DAC. The `Focus` signal is used to signal the High Speed Sampler when the DAC sweep is in the “focus” area, see Section 10.5 for more information on the focus feature of NVA6100.

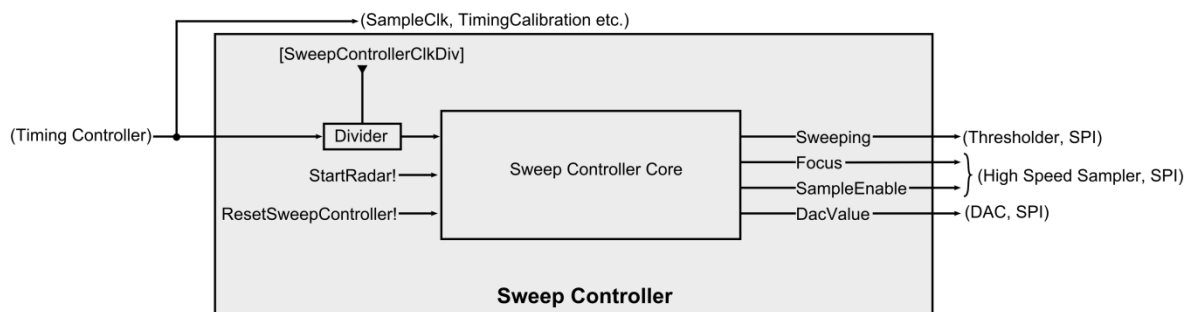


Figure 6. Sweep Controller block diagram.

10.2 PRINCIPLE OF OPERATION

The state diagram in Figure 7 shows the principle operation of the Sweep Controller state machine. Initially, the Sweep Controller is in the idle state. Issuing the `ResetSweepController` action strobe will also send the Sweep Controller into this state, independently of what state it currently is in. Once the `StartRadar` action strobe has been issued, the Sweep Controller proceeds to set the DAC to its initial value. The initial DAC value is either `DACMax` or `DACMin`, depending on the Bi-directional sweep settings (see Section 10.3). After this, the Sweep Controller waits a configurable number of MCLK cycles, giving the DAC time to settle at its initial value.

Then the Sweep Controller proceeds to either the “focused averaging” or the “unfocused averaging” state, depending on the setting of the `FocusMin` and `FocusMax` registers. The internal `SampleEnable` signal is now high, enabling the High Speed Sampler to sample and accumulate the received signal at each rising MCLK edge. Once the configured number of pulses have been sampled

and averaged, the `SampleEnable` signal goes low again. The Sweep Controller then holds the DAC value for a configurable number of MCLK cycles. This allows the High Speed Sampler to settle.

The `DACValue` is now compared to either the `DACMax` or the `DACMin` register, depending on the bi-directional sweep settings. If the `DACValue` is within the boundaries defined by these register, it is incremented, and the Sweep Controller repeats the above described procedure starting from the DAC setup stage.

If the `DACValue` is outside the boundaries defined by the `DACMax` or `DACMin` registers, the sweep sequence is finished and the Sweep Controller will either go back to its idle state, return the `Sweeping` signal to zero, or perform another iteration of the whole sweep sequence, depending on the Bi-directional sweep settings (Section 10.6).

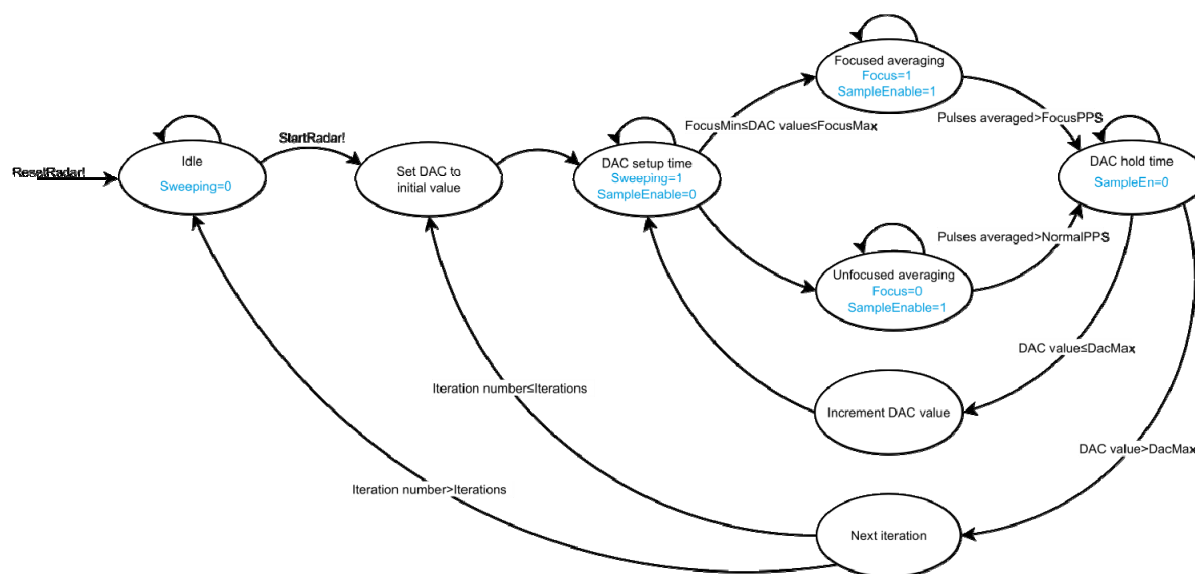


Figure 7. Sweep Controller principle of operation.

10.3 CONFIGURING SWEEP BOUNDARIES

The DAC sweep boundaries are configured through the `DACMin` and `DACMax` register. If the input voltage at the receiver is higher than `DACMax` or lower than `DACMin` it is cut and information may be lost.

To achieve maximum dynamic range, set `DACMin` to 0 and `DACMax` to 8191. This will sweep the DAC over the entire input range.

To improve speed, carefully observe sampled signal while increasing `DACMin` and decreasing `DACMax` until the signal swing fills the entire vertical span of the frame.

10.4 CONFIGURABLE STEP-SIZE

The size of the DAC increment is configurable through the `DACStep` register. Higher `DACStep` values means less averaging and lower sensitivity but higher speed and vice versa.

10.5 FOCUSED AVERAGING

The level of averaging, or the averaging factor, is configured by the two pulses-per-step (PPS) registers `FocusPulsesPerStep` and `NormalPulsesPerStep`. Increasing the PPS improves the final SNR, but also leads to longer readout time. The vertical focus feature is used to increase the

readout speed for a given level of averaging without having to reduce the dynamic range by changing DACMin and DACMax.

When the DACValue is lower than FocusMin or higher than FocusMax, the averaging factor is given by the NormalPulsesPerStep register. When the DACValue is between FocusMin and FocusMax, the averaging factor is given by the FocusPulsesPerStep register.

To configure the Sweep Controller for focused averaging, set the FocusOverride register to 0 and configure DACMin and DACMax for maximum dynamic range as described in Section 10.3. Then set FocusMin and FocusMax to suitable values, making sure that the portion of the signal that requires extra sensitivity is within the focus boundaries. The focus level is the level of extra averaging to be performed inside the focus area. There are ten different focus levels to choose from, where 1x is the lowest and 512x is the highest. The focus level is configured through the UnfocusedCounterIncrement, FocusPulsesPerStep and NormalPulsesPerStep registers.

The relationship between FocusPulsesPerStep and NormalPulsesPerStep is linked to the focus level. Correct setting of these two registers is important to achieve correct behavior. Start by selecting the desired focus level, and the desired level of averaging outside the focus area (NormalPulsesPerStep). Then configure the UnfocusedCounterIncrement and FocusPulsesPerStep registers according to Table 12.

Focus level	UnfocusedCounterIncrement	PPS setting
1x	0	FocusPPS = NormalPPS
2x	1	FocusPPS = 2 x NormalPPS
4x	2	FocusPPS = 4 x NormalPPS
8x	3	FocusPPS = 8 x NormalPPS
16x	4	FocusPPS = 16 x NormalPPS
32x	5	FocusPPS = 32 x NormalPPS
64x	6	FocusPPS = 64 x NormalPPS
128x	7	FocusPPS = 128 x NormalPPS
256x	8	FocusPPS = 256 x NormalPPS
512x	9	FocusPPS = 512 x NormalPPS

Table 12. Configuring Focus.

To disable focused averaging, set the FocusOverride and FocusOverrideValue registers to 1 and adjust the averaging factor through the FocusPulsesPerStep register. The UnfocusedCounterIncrement, NormalPulsesPerStep, FocusMin and FocusMax registers are all ignored when focused averaging is disabled.

10.6 BI-DIRECTIONAL SWEEPS

The Sweep Controller can be configured to perform sweeps in six different modes. Figure 8 illustrates the time-voltage relationship of the output of the DAC for each of the modes (numbered A through F in the figure).

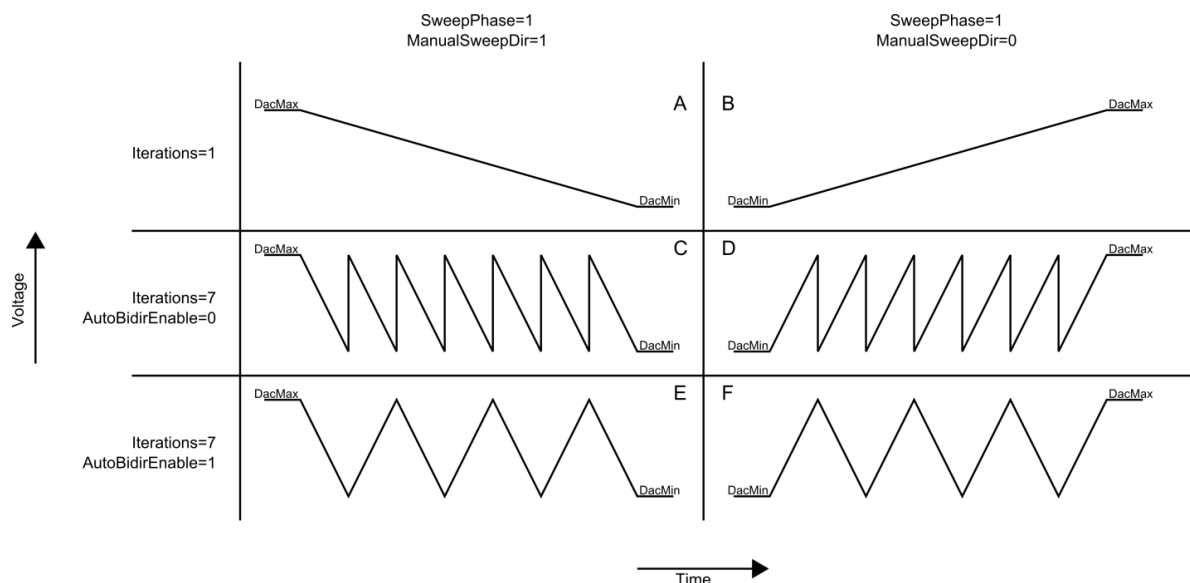


Figure 8. Time-voltage relationship of the different sweep modes, together with their corresponding register settings.

NVA6100 uses averaging to suppress noise, but this works best if the noise is white. In cases where the noise has a $1/f$ characteristic (also known as flicker noise) the “noise signal” might be the same (or very similar) for two subsequent samples. In such cases, using the average of these two samples will not suppress the noise. Plots A and B in Figure 8 illustrates the DAC output voltage for a regular sweep with low $1/f$ noise suppression.

To increase the low frequency noise suppression, the sweep sequence can be split over several iterations without increasing the total level of averaging. This spreads out the samples taken for a given DACValue, and increases the chance that the noise signal will have changed. The resulting DAC output voltage is illustrated in plots C and D in Figure 8. Here, the `Iterations` register has been increased. To maintain the same total level of averaging, the two PPS-registers have to be decreased accordingly.

Because stepping the DAC from its minimum to its maximum value requires longer settling time than stepping it by one LSB, the bi-directional sweeps (E and F) allows multiple iterations without extra settling time.

10.7 MONITORING SWEEP PROGRESS

The best way to monitor progress of a sweep is to check the state of the `Sweeping` signal. This signal is available as the MSb of the `SweepControllerStatus` register. Internally, this signal is synchronized with the rising edge of the `MCLK` signal, and not the `SPI` clock. This allows the signal to be monitored by the `SPI` master as described in Section 6.6.

11 TIMING CONTROLLER

The Timing Controller generates timing signals to the transmitter and the high speed sampler from the master clock signal, MCLK. The controller implements a programmable 0 – 353 ns FrameOffset in steps of ~1.7 ps, giving a spatial frame offset up to ~53 meters in free space. The FrameOffset is realised by three different delay elements, CoarseTune: 351x1 ns, MediumTune: 63x31 ps and FineTune: 63x1.7 ps.

The actual delay of an element is highly dependent upon environmental factors, such as temperature and power variations, and the delays presented in this chapter should be regarded as average values.

11.1 DELAYED CLOCK FRAME OFFSET

It is possible to delay sampling of the pulse to a multiple of the sampling period. This feature can be used in conjunction with the programmable delay elements and gives the possibility to sample signals with a time of flight up to several kilometres. Whether there is any signal left to sample at these distances, is obviously limited by the initially emitted power.

11.2 STAGGERED PRF CONTROL

The staggered PRF circuit consists of a double set of 6 16-bit Linear Feedback Shift Registers (LFSRs). Each LFSR generates a pseudo-random bit sequence. The initial state of the register is called a seed and the generated stream of bits is fully deterministic, the output is determined by the previous state. Each of the 6 LFSRs are programmable. The 6 LFSRs control a 6-bit programmable delay element similar to the MediumTune. This delay element ensures a small variation in reception time from impulse to impulse, giving a large unambiguous range.

11.3 LFSR

The bit positions that determine the next state are called taps. The taps are sequentially XOR'ed with the output bit and fed back to the input. The input bit is often referred to as bit 1 and the output bit, in this case, as bit 16. The arrangement of taps determines the length and sequence of the output stream and is described by a characteristic polynomial. A maximum-length polynomial for a 16 bit LFSR is $x^{16} + x^{13} + x^{12} + x^{11} + 1$ which corresponds to the tapping bit [16, 13, 12, 11] in the register. The LFSR is implemented such that the seed is given when setting the taps: the tapped bits are set to 1, while the rest are set to zero.

12 TIMING CALIBRATION

To minimize the impact of environmental variations on the sampled signal, the NVA6100 employs a multi-purpose Timing Calibration circuit, enabling measurement of the different delay elements relative to the clock frequency. This enables the user to implement compensation algorithms, effectively negating the effects of variations in temperature and supply voltage. The timing calibration unit operates in three different modes:

- Equivalent Clock Cycle Delay: characterization of the number of `CoarseTune`, `MediumTune` and `FineTune` delay elements per clock cycle.
- FrameOffset Calibration: characterization of the number of `MediumTune` (and `FineTune`) delays in a specific `CoarseTune` delay element.
- Average Sampling Rate: characterization of the number of `Coarse`-, `Medium`- and `FineTune` delay elements in the sampling delay line.

Note that any absolute measurements involving `FineTune` requires a very high-precision MCLK.

It is recommended that the user implements Timing Calibration algorithms for all three modes in the application software, and that these functions are run whenever a significant change in the radar environment occurs, as well as at start-up. Remember that the absolute values of the delay elements are heavily dependent on the system clock frequency, so a change in clock frequency necessitates a new calibration.

Note that the following descriptions are not complete algorithms, but rather guidelines and background information to help understand the concept and use of the Timing Calibration circuit. Please refer to the appropriate application example for the complete procedure.

12.1 CIRCUIT DESCRIPTION

Figure 9 describes the general block diagram of the measurement and calibration circuit. NVA6100 employs two parallel programmable delays, `SampleDelay` and `SendPulseDelay` (gray boxes in the figure). These signals are synthesized in the TimingController circuit, from the MCLK signal.

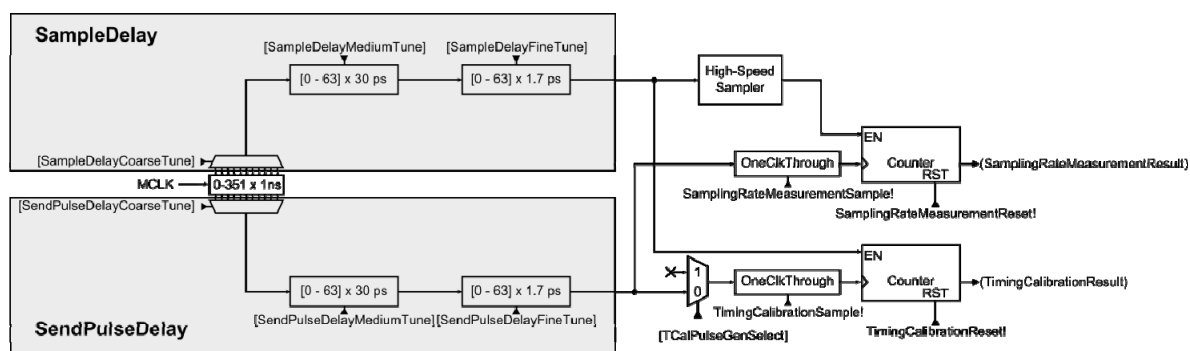


Figure 9. Timing Calibration circuit block diagram.



Note. Measurement of the Pulse Generator delay line length is not available in the current revision of NVA6100. Therefore, `TCalPulseGenSelect = 0` is required for all calibration and measurement.

12.2 EQUIVALENT CLOCK CYCLE DELAY

Determining the exact number of *CoarseTune*-, *MediumTune*- and *FineTune* delay elements required to represent a single period of the master clock (MCLK) is imperative for calculating absolute distances in a dynamic radar environment. Figure 10 illustrates a simplified version of the desired output, where CTx represent *CoarseTune* elements, and MTx represents *MediumTune* elements.

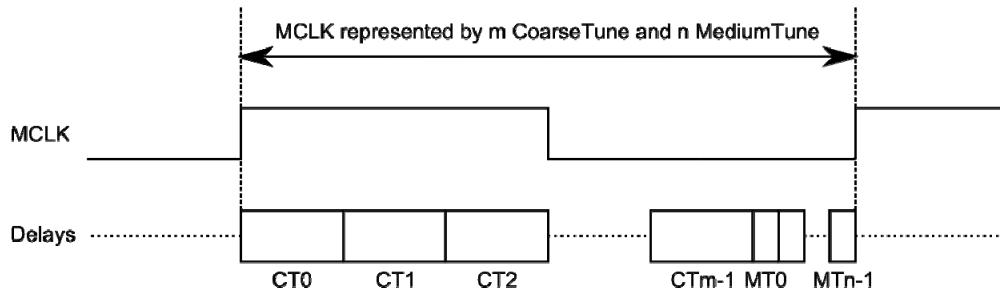


Figure 10. MCLK represented in terms of *CoarseTune* and *MediumTune*.

Determining the total relative delay in a single clock cycle is done by searching for two consecutive rising edges of MCLK. All delays associated with *SendPulseDelay* are kept at 0, while *SampleDelayCoarseTune* is incremented until both edges have been detected, i.e. the *SendPulseDelay* signal samples the pulse delayed by *SampleDelay* (see Figure 9). Figure 11 illustrates the desired output (note that rising edges appear as falling edges in the sampled output, because of the direction of sweeping). Having determined the number of *CoarseTune* elements required, the procedure is repeated with *MediumTune* (and *FineTune*, if required).

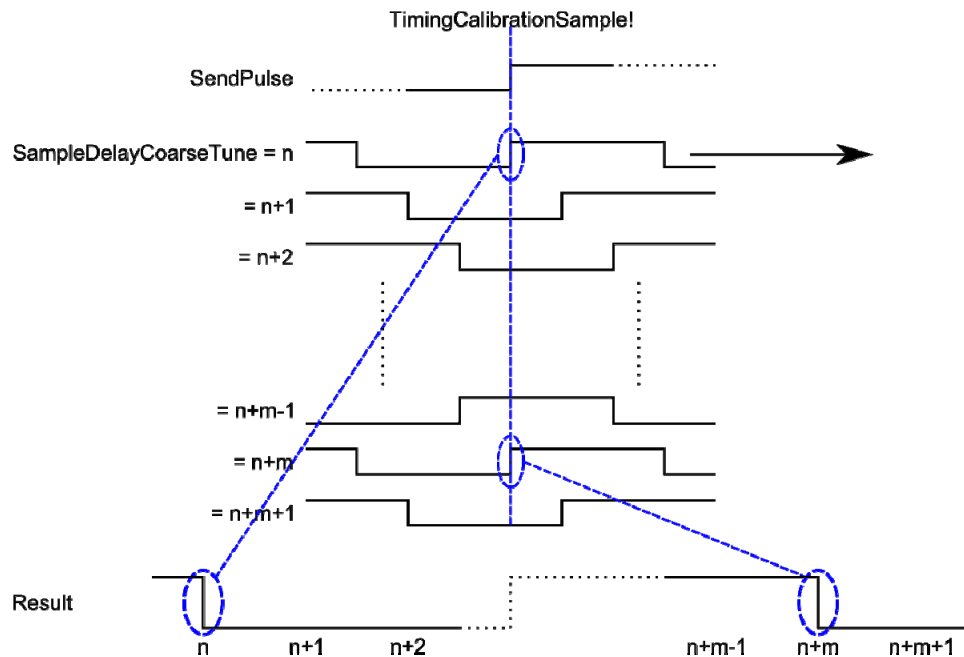


Figure 11. *SampleDelayCoarseTune* is incremented until two rising edges are detected.

12.3 FRAMEOFFSET CALIBRATION

Characterizing the mean number of `SampleDelayMediumTune` delay elements in a `SampleDelayCoarseTune` delay element (and also the mean number of `SampleDelayFineTune` in a `SampleDelayCoarseTune` element), is important for being able to accurately determine distance, as well as tracking objects over more than a single frame. Also, for increased precision and elimination of any process-related variation, finding the number of `SampleDelayMediumTune` delays in one *specific* `SampleDelayCoarseTune` delay element is of interest.

This mode uses both the `SampleDelay` and the `SendPulseDelay` delay lines.

1. Set `SampleDelayCoarseTune` to 1 higher than `SendPulseDelayCoarseTune`.
2. Increment `SendPulseDelayMediumTune` and sample until an edge is detected.

The `SendPulseDelay` delay line now contains enough delay (`CoarseTune` + `MediumTune`) to guarantee that we sample an edge in the `SampleDelay` delay line. Since the absolute delay of the two delay lines are not equal (they have a small relative offset, i.e. `SendPulseDelay` is a bit longer than `SampleDelay`), the current `SendPulseDelayMediumTune` value does not directly correspond to the value we want to calibrate, i.e. `SampleDelayMediumTune`. Therefore, we need to compare `SendPulseDelayMediumTune` to `SampleDelayMediumTune` to complete the calibration.

3. Set `SampleDelayCoarseTune` = `SendPulseDelayCoarseTune`, increment `SampleDelayMediumTune` and sample until an edge is detected.

Finding the number of `SampleDelayFineTune` in a `SampleDelayMediumTune` element is done in a similar fashion. However, for certain settings, the `FineTune` delay may not be long enough to compensate for all due to process and environmental variations, so a combination of `CoarseTune`, `MediumTune` and `FineTune` may be required. See the appropriate application example for details.

12.4 AVERAGE SAMPLING RATE

The Timing Calibration circuit also includes functionality for determining the sampling rate of the system, represented by `CoarseTune`, `MediumTune` and `FineTune` delay elements. Using the results from the previous calibration/measurement modes, the absolute sampling rate can be calculated. Knowing the sampling rate may be crucial for further signal conditioning/filtering, as well as determining absolute distances intra-frame. Note from Figure 9 that the result from this measurement is stored in a separate register, `SamplingRateMeasurementResult`.

1. Set all `SampleDelay` delay elements to 0.
2. Increment `SendPulseDelayCoarseTune`, `MediumTune` and `FineTune`, and sample (using `SamplingRateMeasurementSample`) successively until an edge is detected.

Having determined the total amount of `SendPulseDelay` delay elements in the High-Speed Sampler's delay line, we need to convert this back to `SampleDelay` delay elements. (As in Section 12.3, `SendPulseDelayMediumTune` and `SampleDelayMediumTune` are two separate delay lines, and we need to compare `SendPulseDelayMediumTune` to `SampleDelayMediumTune` to complete the calibration).

3. Set `SampleDelayCoarseTune` = `SendPulseDelayCoarseTune`, increment `SampleDelayMediumTune` and sample (using `TimingCalibrationSample` this time) until an edge is detected.
4. The result is the relative delay through the entire sampler's delay line. Combine with results from the Equivalent Clock Cycle Delay and FrameOffset Calibration measurements, and divide by the number of samplers for the equivalent sampling rate.

13 CONFIGURATION REGISTERS

The configuration of the NVA6100 is done through programming of the configuration registers accessible through the 8-bit Serial Peripheral Interface. Table 13 provides a complete overview of the address map for all the configuration and status registers, as well as the action commands. A detailed description of all registers and commands are available in Section 13.1.

7 registers are action commands. Writing to these registers will initiate a change of state, or a specific action. There are 24 8-bit registers, 25 16-bit registers and 2 32-bit registers, as well as one 16-kbit memory register, for radar data readout. Please see Section 6 for details on how to use the SPI.

After chip power-up, all registers need to be configured for correct operation. Refer to Section 6.5 for a sample initialization procedure.

Address	Field/Register name	Attributes	Type
0x00	ForceZero	R	8 bit
0x01	ForceOne	R	8 bit
0x02	ChipID	R	16 bit
0x03 - 0x1F	Reserved	-	-
0x20	SamplerOutputBuffer	R	0 - 16 kbit
0x21	SamplerReadoutCtrl	RW	16 bit
0x22	SamplerCtrl	RW	8 bit
0x23	ThresholderPowerdown	RW	8 bit
0x24	LoadOutputBuffer	W	Action
0x25	ResetCounters	W	Action
0x26	SamplerInputCtrl	RW	8 bit
0x27	ThresholderCtrl	RW	8 bit
0x28	CounterBitSelectorOutput	R	16 bit
0x29 - 0x2B	Reserved	-	-
0x2C	SamplingRateMeasurementResult	R	8 bit
0x2D	SamplingRateMeasurementSample	W	Action
0x2E	SamplingRateMeasurementReset	W	Action
0x2F	Reserved	-	-
0x30	FocusPulsesPerStep	RW	32 bit
0x31	NormalPulsesPerStep	RW	32 bit
0x32	DACFirstIterationSetupTime	RW	16 bit
0x33	DACFirstStepSetupTime	RW	16 bit
0x34	DACRegularSetupTime	RW	16 bit
0x35	DACLastIterationHoldTime	RW	16 bit
0x36	DACLastStepHoldTime	RW	16 bit
0x37	DACRegularHoldTime	RW	16 bit
0x38	SweepMainCtrl	RW	8 bit
0x39	DACMax	RW	16 bit
0x3A	DACMin	RW	16 bit
0x3B	DACStep	RW	16 bit
0x3C	Iterations	RW	16 bit
0x3D	FocusMax	RW	16 bit
0x3E	FocusMin	RW	16 bit
0x3F	Reserved	-	-
0x40	FocusSetupTime	RW	8 bit
0x41	FocusHoldTime	RW	8 bit
0x42	SweepClkCtrl	RW	8 bit
0x43	StartSweep	W	Action
0x44	ResetSweepController	W	Action
0x45 - 0x46	Reserved	-	-
0x47	SweepControllerStatus	R	16 bit
0x48 - 0x4F	Reserved	-	-
0x50	PGCtrl	RW	8 bit
0x51 - 0x57	Reserved	RW	8 bit
0x58	DACCtrl	RW	16 bit
0x59 - 0x5F	Reserved	-	-
0x60	MClkCtrl	RW	8 bit
0x61	StaggeredPRFCtrl	RW	8 bit
0x62	StaggeredPRFDelay	RW	8 bit

Address	Field/Register name	Attributes	Type
0x63	StaggeredPRFReset	W	Action
0x64	LFSR5TapEnable	RW	16 bit
0x65	LFSR4TapEnable	RW	16 bit
0x66	LFSR3TapEnable	RW	16 bit
0x67	LFSR2TapEnable	RW	16 bit
0x68	LFSR1TapEnable	RW	16 bit
0x69	LFSR0TapEnable	RW	16 bit
0x6A	TimingCtrl	RW	8 bit
0x6B	SampleDelayCoarseTune	RW	16 bit
0x6C	SampleDelayMediumTune	RW	8 bit
0x6D	SampleDelayFineTune	RW	8 bit
0x6E	SendPulseDelayCoarseTune	RW	16 bit
0x6F	SendPulseDelayMediumTune	RW	8 bit
0x70	SendPulseDelayFineTune	RW	8 bit
0x71	TimingCalibrationCtrl	RW	8 bit
0x72	TimingCalibrationSample	W	Action
0x73	TimingCalibrationReset	W	Action
0x74	TimingCalibrationResult	R	8 bit
0x75	MClkOutputCtrl	RW	8 bit

Table 13. Configuration registers overview.

13.1 CONFIGURATION REGISTER DETAILS

FORCEZERO (0x00)

Bit	Variable name	Details
[7:0]	ForceZero	All bits hardcoded to logic zero. Can be used as a simple functional test of the chip to verify connectivity etc.

FORCEONE (0x01)

Bit	Variable name	Details
[7:0]	ForceOne	All bits hardcoded to logic one. Can be used as a simple functional test of the chip to verify connectivity etc.

CHIPID (0x02)

Bit	Variable name	Details
[15:0]	ChipID	Always reads as 0x0306.

SAMPLEROUTPUTBUFFER (0x20)

Bit	Variable name	Details
[16383:0]	SamplerOutputBuffer	The result from the last sweep is placed in this register after the LoadOutputBuffer SPI action is called. Users can choose to read all or parts of this register by adjusting settings in SamplerReadoutCtrl.

SAMPLERREADOUTCTRL (0x21)

Bit	Variable name	Details
[15:12]	–	–
[11:7]	CounterMSb	Selects the most significant bit to read. Can be any value from 0 to 31, but must always be higher than or equal to CounterLSb.
[6:2]	CounterLSb	Selects the least significant bit to read. Can be any value from 0 to 31, but must always be lower than or equal to CounterMSb.
[1:0]	Downsampling	Selects the level of downsampling. 0: No downsampling, all samplers are read out. 1: Sampler number [0, 2, 4, ..., 512] are read out. 2: Sampler number [0, 4, 8, ..., 512] are read out. 3: Sampler number [0, 8, 16, ..., 512] are read out.

SAMPLERCTRL (0x22)

Bit	Variable name	Details
[7:6]	–	–
[5:2]	UnfocusedCounterIncrement	Selects the unfocused incrementation exponent of the counters in the sampler when the focus feature is enabled. The counters are incremented by a value 2^n . See also NormalPulsesPerStep and FocusPulsesPerStep in the Sweep Controller user interface documentation, Section 10.
[1:0]	SamplingRate	Selects the effective sampling interval in the sampler. 0: 26 ps. 1: 52 ps. 2: 280 ps. 3: (Not used).

THRESHOLDERPOWERDOWN (0x23)

Bit	Variable name	Details
[7:3]	–	–
[1]	ManualPowerdown	When this bit is set, power is turned off at the HSC. Requires DisableAutomaticPowerdown to be set.
[0]	DisableAutomaticPowerdown	Unless this bit is set, the Sweep Controller automatically powers down the HSC when inactive, in order to save power.

LOADOUTPUTBUFFER (0x24)

Load SamplerOutputBuffer with the result from the last sweep.

RESETCOUNTERS (0x25)

Clear the result of the last sweep from the samplers. Always call this before initiating a new sweep.

SAMPLERINPUTCTRL (0x26)

Bit	Variable name	Details
[7:1]	–	–
[0]	SampleEveryPulse	Override SampleEnable signal from the Sweep Controller. When enabled, the High Speed Sampler will sample its inputs on every rising MCLK edge, even if the Sweep Controller is inactive. In combination with the SignalOverride bits this can be used to perform a logic self test on the sampler circuitry.

THRESHOLDERCTRL (0x27)

Bit	Variable name	Details
[7]	–	–
[6]	SignalOverride	Enable overriding of the input of the High Speed Sampler.
[5]	SignalOverrideValue	Overrides the input of the High Speed Sampler: 0: The input is forced low. 1: The input is forced high. Correct operation of this bit requires SignalOverride to be set.
[4:2]	VGACfg	Set gain level in the VGA: 0: N/A. 1: Gain 4. 2: Gain 6. 3: Gain 2. 4: Gain 7. 5: Gain 3. 6: Gain 5. 7: Gain 1. Intervals between gain levels are not linear.
[1]	PulseStretchEnable	Enable pulse stretching of the thresholder output. This mode is required when SamplingRate = 2 is used.
[0]	VthOutEnable	Enable DAC output on external VTHOUT pin.

COUNTERBITSELECTOROUTPUT (0x28)

Bit	Variable name	Details
[15:0]	CounterBitSelectorOutput	Internal signal, available for test purposes.

SAMPLINGRATEMEASUREMENTRESULT (0x2C)

Bit	Variable name	Details
[7:0]	SamplingRateMeasurementResult	Sampling rate measurement result register.

SAMPLINGRATEMEASUREMENTSAMPLE (0x2D)

Perform one sample of the sampling rate measurement.

SAMPLINGRATEMEASUREMENTRESET (0x2E)

Reset the SamplingRateMeasurementResult register.

FOCUSPULSESPERSTEP (0x30)

Bit	Variable name	Details
[31:24]	–	–
[23:0]	FocusPulsesPerStep	Number of pulses per step to use within the focus range. If set to a different value than NormalPulsesPerStep, the UnfocusedCounterIncrement register in the High Speed Sampler must be updated accordingly. See Section 10.5 for more details.

NORMALPULSESPERSTEP (0x31)

Bit	Variable name	Details
[31:24]	–	–
[23:0]	NormalPulsesPerStep	Number of pulses per step to use outside of the focus range. If set to a different value than FocusPulsesPerStep, the UnfocusedCounterIncrement register in the High Speed Sampler must be updated accordingly. See Section 10 for more details.

DACFIRSTITERATIONSETUPTIME (0x32)

Bit	Variable name	Details
[15:0]	DACFirstIterationSetupTime	DAC setup time in MCLK periods. Pauses the sampling momentarily at the first DAC step of the first iteration in a multi-iteration sweep, giving the DAC time to settle at a new value before sampling is commenced. In some implementations the first DAC step might involve stepping from DACMax to DACMin (or DACMin to DACMax), which requires a longer setup time than the regular DAC steps.

DACFIRSTSTEPSETUPTIME (0x33)

Bit	Variable name	Details
[15:0]	DACFirstStepSetupTime	DAC setup time in MCLK periods. Pauses the sampling momentarily at the first DAC step of each sweep, giving the DAC time to settle at a new value before sampling is commenced. In some implementations the first DAC step might involve stepping from DACMax to DACMin (or DACMin to DACMax), which requires a longer setup time than the regular DAC steps.

DACREGULARSETUPTIME (0x34)

Bit	Variable name	Details
[15:0]	DACRegularSetupTime	DAC setup time in MCLK periods. Pauses the sampling momentarily at each DAC step, giving the DAC time to settle at a new value before sampling is commenced. Depending on the setting of the DACStep register, this may or may not be necessary. In general, larger steps require longer setup time.

DACLASTITERATIONHOLDTIME (0x35)

Bit	Variable name	Details
[15:0]	DACLastIterationHoldTime	DAC hold time in MCLK periods. Pauses the sampling and holds the current DAC value momentarily at the last DAC step of the last sweep, before stepping the DAC again.

DACLASTSTEPHOLDTIME (0x36)

Bit	Variable name	Details
[15:0]	DACLastStepHoldTime	DAC hold time in MCLK periods. Pauses the sampling and holds the current DAC value momentarily at the last DAC step of each sweep, before stepping the DAC again.

DACREGULARHOLDTIME (0x37)

Bit	Variable name	Details
[15:0]	DACRegularHoldTime	DAC hold time in MCLK periods. Pauses the sampling and holds the current DAC value momentarily at each DAC step, before stepping the DAC again.

SWEEPMAINCTRL (0x38)

Bit	Variable name	Details
[7:5]	-	-
[4]	FocusOverrideValue	Override the value of the internal Focus signal 0: Focus is forced low. 1: Focus is forced high. Correct operation requires FocusOverride to be set.
[3]	FocusOverride	Enable overriding of internal Focus signal.
[2]	SweepPhase	Sets the initial direction of a bi-directional sweep. 0: First sweep starts at DACMin, ends at DACMax. 1: First sweep starts at DACMax, ends at DACMin.
[1]	ManualSweepDir	Sets the sweep direction if bi-directional sweeps are disabled. 0: All sweeps starts at DACMin, ends at DACMax. 1: All sweeps starts at DACMax, ends at DACMin.
[0]	AutoBidirEnable	Enable bi-directional sweeps. 0: Bi-directional sweeps are disabled. 1: Bi-directional sweeps are enabled.

DACMAX (0x39)

Bit	Variable name	Details
[15:13]	-	-
[12:0]	DACMax	Max value of DAC sweep. Always set this field higher than or equal to DACMin.

DACMIN (0x3A)

Bit	Variable name	Details
[15:13]	-	-
[12:0]	DACMin	Min value of DAC sweep. Always set this field lower than or equal to DACMax.

DACSTEP (0x3B)

Bit	Variable name	Details
[15:13]	-	-
[12:0]	DACStep	Step size of DAC sweep.

ITERATIONS (0x3C)

Bit	Variable name	Details
[15:0]	Iterations	Number of sweep iterations to perform. Increasing the number of iterations increases the total level of averaging performed per readout, similar to increasing NormalPulsesPerStep/FocusPulsesPerStep.

FOCUSMAX (0x3D)

Bit	Variable name	Details
[15:0]	FocusMax	Defines the focus max value.

FOCUSMIN (0x3E)

Bit	Variable name	Details
[15:0]	FocusMin	Defines the focus min value.

FOCUSSETUPTIME (0x40)

Bit	Variable name	Details
[7:0]	FocusSetupTime	Setup time, in MCLK periods, for the Focus signal going from the sweep controller to the sampler.

FOCUSHOLDTIME (0x41)

Bit	Variable name	Details
[7:0]	FocusHoldTime	Hold time, in MCLK periods, for the Focus signal going from the sweep controller to the sampler.

SWEEPCLKCTRL (0x42)

Bit	Variable name	Details
[7:4]	-	-
[3]	DelayOutputSignals	Legacy register, set to zero.
[2:1]	SweepControllerClkDivider	Sweep Controller Clock divider. $f_{\text{SweepControllerClk}} = \frac{f_{\text{Mclk}}}{2^{\text{SweepControllerClkDivider}}}$
[0]	FocusSetupHoldEnable	Use setup/hold time for focus signal.

STARTSWEEP (0x43)

Signals the Sweep Controller to start a new sweep.

RESETSWEEPCONTROLLER (0x44)

Aborts any running sweep and resets the internal Sweep Controller state machines. This action does not reset any of the SPI registers. Always remember to call this action at startup, to ensure correct SweepController state.

SWEEPCONTROLLERSTATUS (0x47)

All bits in this register are asynchronous relative to SCLK (they are synchronous relative to MCLK). This means that if they are read out while MCLK is free running, their value might be invalid. They are meant to be used to monitor the progress of a sweep, or for debug and testing purposes.

Bit	Variable name	Details
[15]	Sweeping	Goes high when a StartSweep action strobe is received and stays high for the duration of the sweep. This bit should be used as the primary source for monitoring the progress of a sweep.
[14]	Focus	Internal signal, available for test purposes.
[13]	SampleEnable	Internal signal, available for test purposes.
[12:0]	DACValue	Internal signal, available for test purposes.

PGCTRL (0x50)

Bit	Variable name	Details
[7:6]	PulseGen	Selects active pulse generator. 0: Pulse generator is turned off. 1: Low-band pulse generator. 2: Medium-band pulse generator. 3: Pulse generator is turned off.
[5:3]	-	-
[2:1]	PulseGenFineTune	Selects center frequency of the selected pulse generator. 0: Slow. 1: Nominal. 2: Fast.
[0]	PulseGenMeasureEn	Enables measurement of the selected pulse generator's center frequency. For debug and characterization purposes.

DACCTRL (0x58)

Bit	Variable name	Details
[15]	DACSourceSweepController	Selects source for the DAC: 0: DAC is configured by the DACValueManual register. Allows manual configuration of the DAC, mainly used for testing purposes. 1: DAC is updated by Sweep Controller. Should be the default behavior for most applications.
[14:13]	–	–
[12:0]	DACValueManual	Sets the value of the DAC. For this register to have any effect, DACSourceSweepController must be set to zero.

MCLKCTRL (0x60)

Bit	Variable name	Details
[7:5]	–	–
[4]	ClkEnable	Enables distribution of the external master clock (MCLK).
[3:1]	ClkDivider	MCLK divider. $f_{DividedClk} = \begin{cases} \frac{f_{Mclk}}{2^{ClkDivider}}, & 0 \leq ClkDivider \leq 4 \\ 0, & 5 \leq ClkDivider \leq 7 \end{cases}$
[0]	DividedClkShaper	Shapes the pulse width of the divided clock to ~3.4 ns. 0: Disable. 1: Enable.

STAGGEREDPRFCTRL (0x61)

Bit	Variable name	Details
[7:3]	–	–
[2]	StaggeredPRFEnable	Enables the LFSR register, giving a pseudo random staggered Pulse Repetition Frequency. 0: Disable. 1: Enable.
[1]	DelayedClkSelect	Select a delayed version of the staggered MCLK as source for the pulse generator. The delay to the sampling circuit is programmed in StaggeredPRFDelay. Pulse generator source: 0: SendPulse delay line. 1: Delayed staggered MCLK.
[0]	DelayedClkSampleWhenReady	Enabling this will suppress the sampling of pulses until the staggered frame offset clock is delayed according to StaggeredPRFDelay. 0: Disable. 1: Enable.

STAGGEREDPRFDELAY (0x62)

Bit	Variable name	Details
[7:0]	StaggeredPRFDelay	Delay the sample signal by 0–255 DividedClk periods.

STAGGEREDPRFRESET (0x63)

Reset the StaggeredPRFDelay counter and LFSR registers to their default state.

LFSR5TAPENABLE, LFSR4TAPENABLE, ..., LFSR1TAPENABLE, LFSR0TAPENABLE (0x64 – 0x69)

Bit	Variable name	Details
[15]	–	–
[14:0]	LFSRXTapEnable	Determine which bits to feed back in the 16 bit LFSR register. Bit n in this SPI register corresponds to tapping of bit n+1 in the LFSR register. The output bit (bit 16) in the LFSR register is always tapped.

TIMINGCTRL (0x6A)

Bit	Variable name	Details
[7:4]	–	–
[3]	SendPulseDelayShaper	Shapes the pulse width of the send pulse delay clock to ~3.4 ns. 0: Disable. 1: Enable.
[2]	SampleDelayShaper	Shapes the pulse width of the sample delay clock to ~3.4 ns. 0: Disable. 1: Enable.
[1]	SampleDelayFineTuneEnable	Enables the fine tune delay line in the sample delay clock path. An extra delay of ~1.6 ns will be added to the overall sample delay.
[0]	SendPulseDelayFineTuneEnable	Enables the fine tune delay line in the send pulse delay clock path. An extra delay of ~1.6 ns will be added to the overall send pulse delay.

SAMPLEDELAYCOARSETUNE (0x6B)

Bit	Variable name	Details
[15:9]	–	–
[8:0]	SampleDelayCoarseTune	Delays the sample signal using 0-351 delay elements, each with a delay of ~1 ns.

SAMPLEDELAYMEDIUMTUNE (0x6C)

Bit	Variable name	Details
[7:6]	–	–
[5:0]	SampleDelayMediumTune	Delays the sample signal using 0-63 delay elements, each with a delay of ~31 ps.

SAMPLEDELAYFINETUNE (0x6D)

Bit	Variable name	Details
[7:6]	–	–
[5:0]	SampleDelayFineTune	Delays the sample signal using 0-63 delay elements, each with a delay of ~1.7 ps.

SENDPULSEDELAYCOARSETUNE (0x6E)

Bit	Variable name	Details
[15:9]	–	–
[8:0]	SendPulseDelayCoarseTune	Delays the send pulse signal using 0-351 delay elements, each with a delay of ~1 ns.

SENDPULSEDELAYMEDIUMTUNE (0x6F)

Bit	Variable name	Details
[7:6]	–	–
[5:0]	SendPulseDelayMediumTune	Delays the send pulse signal using 0-63 delay elements, each with a delay of ~31 ps.

SENDPULSEDELAYFINETUNE (0x70)

Bit	Variable name	Details
[7:6]	–	–
[5:0]	SendPulseDelayFineTune	Delays the send pulse signal using 0-63 delay elements, each with a delay of ~1.7 ps.

TIMINGCALIBRATIONCTRL (0x71)

Bit	Variable name	Details
[7:1]	–	–
[0]	TCalPulseGenSelect	Select source for timing calibration: 0: SendPulse signal. 1: Pulse generator delay line output.

TIMINGCALIBRATIONSAMPLE (0x72)

Perform one sample of the selected source.

TIMINGCALIBRATIONRESET (0x73)

Reset the TimingCalibrationResult register.

TIMINGCALIBRATIONRESULT (0x74)

Bit	Variable name	Details
[7:0]	TimingCalibrationResult	Timing calibration result register.

MCLKOUTPUTCTRL (0x75)

Bit	Variable name	Details
[7:2]	-	-
[1]	MClkDelayedOutEnable	Enable the output of the internally delayed MCLK signals.
[0]	MClkDelayedOutSource	0: Sample signal. 1: SendPulse signal.

14 REFERENCE APPLICATION CIRCUIT

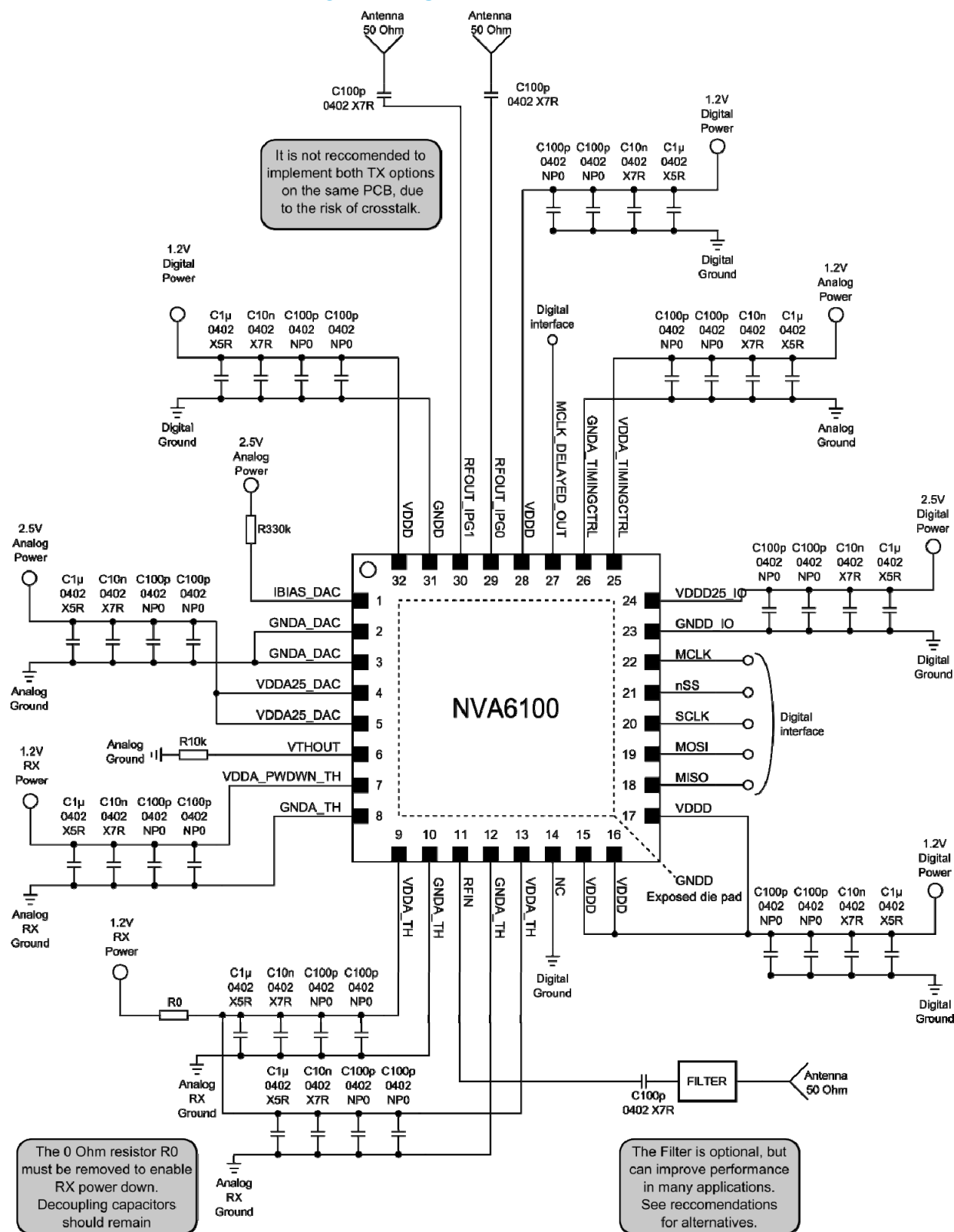


Figure 12. Typical application circuit.

15 PACKAGE DIMENSIONS AND RECOMMENDED LAYOUT (QFN32)

A recommended layout footprint for the QFN32 package is shown in Figure 13. Note that the figure is an illustration only, and not to scale. Physical package dimensions and other data are available upon request.

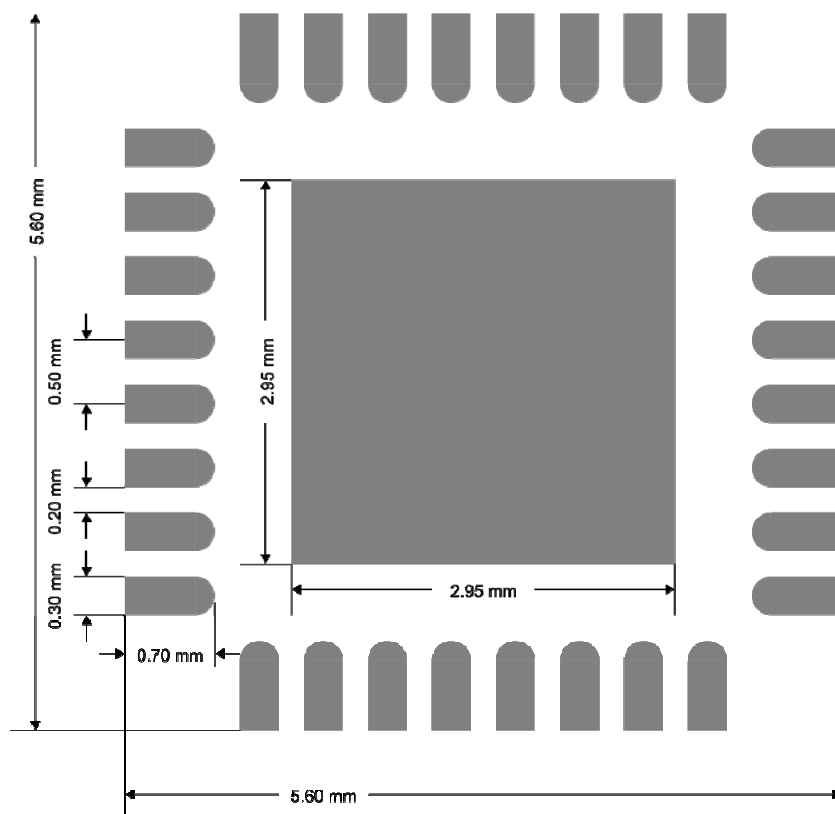


Figure 13. Recommended package layout.

16 ORDERING INFORMATION

Visit www.novelda.no, or contact sales@novelda.no for information on sales and distribution.

17 CONTACT INFORMATION

17.1 NOVELDA CORPORATE HEADQUARTERS

Novelda AS
Garverivegen 2
NO-3850 KVITSEID
NORWAY
Phone: +47 35 05 32 23

17.2 NOVELDA R&D DEPARTMENT

Novelda AS
Forskningsveien 2A
NO-0373 OSLO
NORWAY
Phone: +47 22 49 41 19

17.3 NOVELDA TECHNICAL SUPPORT

Visit www.novelda.no for the latest updated FAQ and errata, or contact support@novelda.no.

18 DOCUMENT HISTORY

Revision	Date	Description
0.31	2010-12-23	Added receiver sensitivity data. Updated and clarified Timing Calibration routines. Updated Figure 2.
0.3	2010-12-10	Updated sampling rate and frame offset resolution data. Added power consumption data. Added recommended package layout. Added Timing Calibration basic routines.
0.21	2010-10-06	Changed document status to "Preliminary". Core power supply operating conditions changed to 1.2 V. Valid operating temperature changed to 0 - 70 °C. Key data in chapter 3. <i>Electrical characteristics</i> updated.
0.1	2010-06-16	Advance information datasheet, initial release. This document now replaces NVA6100DIFF.

DISCLAIMER

Novelda AS reserve the right to make corrections, modifications, enhancements, improvements and other changes to its products and services at any time, and to discontinue any product or service without prior notice. Customers are encouraged to obtain the latest information before placing orders, and should verify that the information is up-to-date and complete. All products are sold subject to Novelda's terms and conditions of sale supplied at the time of order acknowledgement.

The information provided in this document represents Novelda's knowledge and beliefs as of the time of writing. Novelda assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using Novelda components.