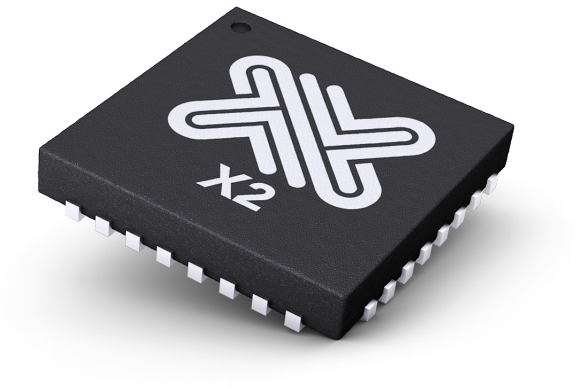
X2

Impulse Radar Transceiver

Key Features

- Single-chip impulse radar transceiver.
- Sub-mm accuracy.
- Low power consumption, typically < 120 mW.
- Industrial operating temperature range, -40 °C - 85 °C.
- Short startup time, < 1 ms.
- Single-ended RF terminals.
- Low power transmitter, with highly configurable output frequency band.
- Extensive hardware support for advanced timing measurement.
- Serial Peripheral Interface.
- QFN32 5x5 mm, plastic leadless package.



Product Description

The X2 impulse radar transceiver from Novelda gives access to advanced sensor technology in one single IC. The flexible transceiver makes X2 the perfect choice for implementing high accuracy, high resolution sensing systems with low power consumption. With access to all of X2's advanced functions, extensive technical documentation, application examples and the RadarLib API from Novelda, development of state of the art impulse radar sensors have never been easier.

Applications

- Sensors for home and building automation applications.
- Security sensor applications where you need to see through obstacles.
- Hidden sensors for esthetic reasons or to make them tamper proof.
- Ranging applications requiring sub-mm accuracy.
- Sensors for robot vision.
- Sensors for industrial automation.
- General sensing applications for presence, ranging and speed.

Ordering Information

For orders, please contact Novelda sales through the contact form on our webpage: http://www.xethru.com.

Table of Contents

1. Abbreviations

Acronym	Description
ADC	Analog-to-Digital Converter
CMU	Clock Management Unit
CT	CoarseTune
DAC	Digital-to-Analog Converter
DCO	Digitally Controlled Oscillator
FMeas	DCO Frequency Measurement
FO	Frame Offset
FT	FineTune
HSC	High Speed Comparator
LFSR	Linear Feedback Shift Register
LNA	Low Noise Amplifier
LSB	Least Significant Byte
LSb	Least Significant bit
MClk	Master Clock
MISO	Master In Slave Out
MOSI	Master Out Slave In
MSB	Most Significant Byte
MSb	Most Significant bit
MT	MediumTune
MUR	Maximum Unambiguous Range
NA	Not Available
NC	Not Connected
nSS	Slave Select (active low)
PG	Pulse Generator
PGD	Pulse Generator Delay
PRF	Pulse Repetition Frequency
QFN	Quad-Flat pack No leads
RF	Radio Frequency
RX	Receive(r)
SC	Sweep Controller
SClk	SPI Clock
SD	SampleDelay
SDL	Sampler Delay Line
SPD	SendPulseDelay
SPI	Serial Peripheral Interface
SW	StopWatch
TC	Timing Controller
TX	Transmit(ter)

Table 1.1. Abbreviations.

2. Electrical Characteristics

2.1. Explanation of Test Levels

Level I	Devices are 100% production tested.
Level II	Devices are production tested, with modifiers.
Level III	Devices are sample tested, with modifiers.
Level IV	Parameter is guaranteed by design.
Level V	Parameter is an expected value only.

2.1.1. Modifiers

(vc)	Tested for supply voltage corners: core ±5%, I/O ±10%.
(vn)	Tested at nominal supply voltages only.
(ti)	Tested for industrial grade temperature range, -40 °C to 85 °C.
(tr)	Tested at room temperature only, 25 °C.

2.2. X2 Power Domains

X2 has seven power domains, as explored in Chapter 3. The power domains are denoted as either *core* or *I/O*, categorized as follows:

Core power supplies	VDDA_HSC, VDDA_RX, VDDD, VDDD_TCTRL, VDDD_TX.
I/O power supplies	VDDA25_DAC, VDDD25_IO.

2.3. Absolute Maximum Ratings

Note that the absolute maximum ratings are limiting values, to be applied individually, and under which functional operation of the device 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	-0.3	1.26	V
Supply voltage, I/O	-0.3	2.75	V
Input voltage, digital I/O	-0.3	3.6	V
Input RF voltage level (peak-peak)		1.2	V
Ambient operating temperature	-40	85	°C

Table 2.1. Absolute maximum ratings.

Parameter	Standard	Max.	Unit
Storage temperature	JESD22-A103C[1]	150	°C
Reflow soldering temperature	J-STD-020[1]	260	°C
Moisture Sensitivity Level	JESD22-A113[1]	Level 2a	
ESD, Charge Device Model	JESD22-C101E[2]	200	V
ESD, Human Body Model	JS-001-2012[2]	1	kV
Latch-up	JESD78D, Class I[2]	Pass	

[1]For reference only. The package is generically qualified by manufacturer, Novelda does not guarantee adherence to standard.

[2]For reference only. Devices are sample tested only, Novelda does not guarantee adherence to standard.

Table 2.2. Environmental sensitivity.

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.

2.4. General Operating Conditions

Parameter	Test level	Min.	Typ.	Max.	Unit
Power supply and reference					
Supply voltage, core	III(ti)		1.2		V
Supply voltage, I/O	III(ti)		2.5		V
DAC reference current, I_{REF}	IV		10		µA
CMOS logic inputs					
Logic '0' voltage, V_{IL}	I	-0.3		0.7	V
Logic '1' voltage, V_{IH}	I	1.7		3.6	V
CMOS logic outputs					
Logic '0' voltage, V_{OL}	I			0.7	V
Logic '1' voltage, V_{OH}	I	1.7			V

Table 2.3. General operating conditions.

2.5. RX Parameters

Parameter	Test level	Min.	Typ.	Max.	Unit
Radar frame length	IV		256		samplers
Input port match, S11					
6.0 - 8.5 GHz	III(vn-tr)		-10.1	-9.6	dB
7.2 - 10.2 GHz	III(vn-tr)		-9.3	-8.9	dB
3.1 - 10.6 GHz	III(vn-tr)		-7.5	-7.2	dB
Receiver gain					
6.0 - 10.2 GHz	V		9.0		dB
Input RF voltage level at 1 dB gain compression point (peak-peak)	V		160		mV

Table 2.4. RX parameters summary.

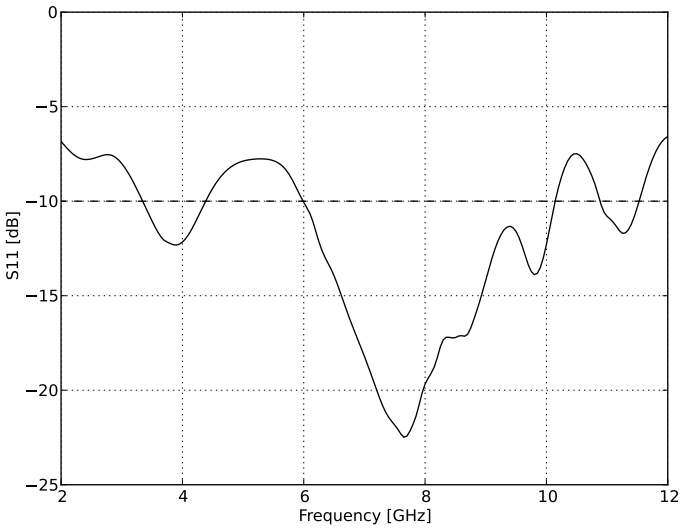


Figure 2.1. Typical receiver S11.

2.6. TX Parameters

Parameter	Test level	Min.	Typ.	Max.	Unit
Derivative of Gaussian pulse approximation	IV		11		
Output port match, S22					
6.0 - 10.2 GHz	III(vn-tr)		-3.8		dB
Output pulse center frequency[1]					
PGSelect = 0	III(vn-ti)		5.3		GHz
PGSelect = 1	III(vn-ti)		5.4		GHz
PGSelect = 2	III(vn-ti)		5.7		GHz
PGSelect = 3	III(vn-ti)		6.1		GHz
PGSelect = 4	III(vn-ti)		6.4		GHz
PGSelect = 5	III(vn-ti)		6.8		GHz
PGSelect = 6	III(vn-ti)		7.3		GHz
PGSelect = 7	III(vn-ti)		7.7		GHz
PGSelect = 8	III(vn-ti)		7.8		GHz
PGSelect = 9	III(vn-ti)		8.2		GHz
PGSelect = 10	III(vn-ti)		8.8		GHz
Bandwidth, -10 dB					
PGSelect = 0	III(vn-ti)	1.65	1.75	1.90	GHz
PGSelect = 1	III(vn-ti)	1.65	1.80	1.95	GHz
PGSelect = 2	III(vn-ti)	1.75	1.85	2.10	GHz
PGSelect = 3	III(vn-ti)	1.85	2.05	2.25	GHz
PGSelect = 4	III(vn-ti)	1.95	2.15	2.30	GHz
PGSelect = 5	III(vn-ti)	2.10	2.30	2.45	GHz
PGSelect = 6	III(vn-ti)	2.25	2.35	2.55	GHz
PGSelect = 7	III(vn-ti)	2.30	2.50	2.85	GHz

Parameter	Test level	Min.	Typ.	Max.	Unit
PGSelect = 8	III(vn-ti)	2.30	2.50	2.90	GHz
PGSelect = 9	III(vn-ti)	2.35	2.65	3.20	GHz
PGSelect = 10	III(vn-ti)	2.65	3.10	4.40	GHz
Mean output power[2]					
PGSelect = 0	III(vn-ti)	-11.0	-10.7	-10.5	dBm
PGSelect = 1	III(vn-ti)	-11.2	-10.8	-10.6	dBm
PGSelect = 2	III(vn-ti)	-11.6	-11.2	-11.0	dBm
PGSelect = 3	III(vn-ti)	-12.2	-11.8	-11.5	dBm
PGSelect = 4	III(vn-ti)	-12.3	-12.0	-11.7	dBm
PGSelect = 5	III(vn-ti)	-12.9	-12.6	-12.2	dBm
PGSelect = 6	III(vn-ti)	-13.9	-13.3	-12.9	dBm
PGSelect = 7	III(vn-ti)	-14.6	-14.0	-13.4	dBm
PGSelect = 8	III(vn-ti)	-14.8	-14.0	-13.4	dBm
PGSelect = 9	III(vn-ti)	-16.0	-14.8	-14.1	dBm
PGSelect = 10	III(vn-ti)	-17.2	-16.4	-15.1	dBm
Peak-to-peak output amplitude					
PGSelect = 0	III(vn-tr)		0.69		V
PGSelect = 1	III(vn-tr)		0.69		V
PGSelect = 2	III(vn-tr)		0.72		V
PGSelect = 3	III(vn-tr)		0.71		V
PGSelect = 4	III(vn-tr)		0.72		V
PGSelect = 5	III(vn-tr)		0.69		V
PGSelect = 6	III(vn-tr)		0.65		V
PGSelect = 7	III(vn-tr)		0.62		V
PGSelect = 8	III(vn-tr)		0.62		V
PGSelect = 9	III(vn-tr)		0.57		V
PGSelect = 10	III(vn-tr)		0.54		V
PGSelect = 11	III(vn-tr)		0.52		V

[1]The center frequency is heavily influenced by environmental factors, such as temperature and supply voltage. The actual center frequency at any given time should be calculated with the PGD Stopwatch function, as explained in Section 8.2. Use the data in this table carefully, and for reference only.

[2]Power average (i.e. true rms), at PRF = 100 MHz, integrated over the corresponding -10 dB bandwidth. Average output power scales with PRF with the relationship: -10*log10(100/PRF) [dB], where PRF is expressed in MHz.

Table 2.5. TX parameters summary.

2.7. Timing Characteristics

Parameter	Test level	Min.	Typ.	Max.	Unit
External clock frequency, f_{EXTCLK}	III(vn-ti)			100	MHz
StopWatch precision (standard deviation, σ)					
SWMeasTargetCycles = 10^4	III(vn-tr)		16		ps
SWMeasTargetCycles = 10^5	III(vn-tr)		5		ps
SWMeasTargetCycles = 10^6	III(vn-tr)		1.6		ps

Parameter	Test level	Min.	Typ.	Max.	Unit
SWMeasTargetCycles = 10^7	III(vn-tr)		0.5		ps
Mean Frame Offset delay unit resolution					
SampleDelay CoarseTune, SDCT	III(vn-tr)		920		ps / step
SampleDelay MediumTune, SDMT	III(vn-tr)		24		ps / step
SendPulseDelay CoarseTune, SPDCT	III(vn-tr)		920		ps / step
SendPulseDelay MediumTune, SPDMT	III(vn-tr)		24		ps / step
Mean system sampling rate	III(vn-tr)		39		GS/s
Staggered PRF sequence length	IV			2^{20}	cycles
Staggered PRF sequence spread (full length)	V		± 1.8		ns
SPI[1]					
Clock frequency, f_{SCLK}	V		15		MHz
nSS setup time	V	2			ns
nSS hold time	V	2			ns
MOSI setup time	V	2			ns
MOSI hold time	V	2			ns
MISO propagation delay	V			4	ns

[1]See Chapter 5 for a complete description of the X2 SPI protocol.

Table 2.6. Timing characteristics.

2.8. Current Consumption

Parameter	Test level	Min.	Typ.	Max.	Unit
Mean active mode currents[1]					
VDDA_RX	III(vn-tr)		15.5		mA
VDDA25_DAC	III(vn-tr)		5.7		mA
VDDD25_IO	V		0.1		mA
MClkDiv = 0					
VDDA_HSC	III(vn-tr)		20.4		mA
VDDD	III(vn-tr)		33.3		mA
VDDD_TCTRL	III(vn-tr)		13.7		mA
VDDD_TX	III(vn-tr)		1.0		mA
MClkDiv = 1					
VDDA_HSC	III(vn-tr)		21.0		mA
VDDD	III(vn-tr)		24.9		mA
VDDD_TCTRL	III(vn-tr)		7.9		mA
VDDD_TX	III(vn-tr)		0.8		mA
MClkDiv = 2					
VDDA_HSC	III(vn-tr)		21.6		mA
VDDD	III(vn-tr)		20.6		mA
VDDD_TCTRL	III(vn-tr)		5.1		mA
VDDD_TX	III(vn-tr)		0.6		mA
MClkDiv = 3					

Parameter	Test level	Min.	Typ.	Max.	Unit
VDDA_HSC	III(vn-tr)		21.9		mA
VDDD	III(vn-tr)		18.4		mA
VDDD_TCTRL	III(vn-tr)		3.6		mA
VDDD_TX	III(vn-tr)		0.5		mA
Peak active mode currents					
VDDA_HSC	III(vn-tr)		40		mA
VDDD	III(vn-tr)		110		mA
Standby currents[2]					
VDDA_HSC	III(vn-tr)		18.2		mA
VDDA_RX	III(vn-tr)		15.5		mA
VDDD	III(vn-tr)		15		mA
VDDD_TCTRL	III(vn-tr)		2.2		mA
VDDD_TX	III(vn-tr)		0.5		mA
VDDA25_DAC	III(vn-tr)		5.7		mA
VDDD25_IO	V		0.1		mA

[1]Unless otherwise noted, active mode currents have been measured with the default radar settings at PRF = 100 MHz and MClkDiv = 0.

[2]The *standby current* is the measured steady-state current while the radar is inactive, and does not imply a low power or power saving mode. The standby current have been measured with the default radar settings, with no external clock applied.

Table 2.7. Current consumption.

3. Pin Assignment

3.1. Pinout

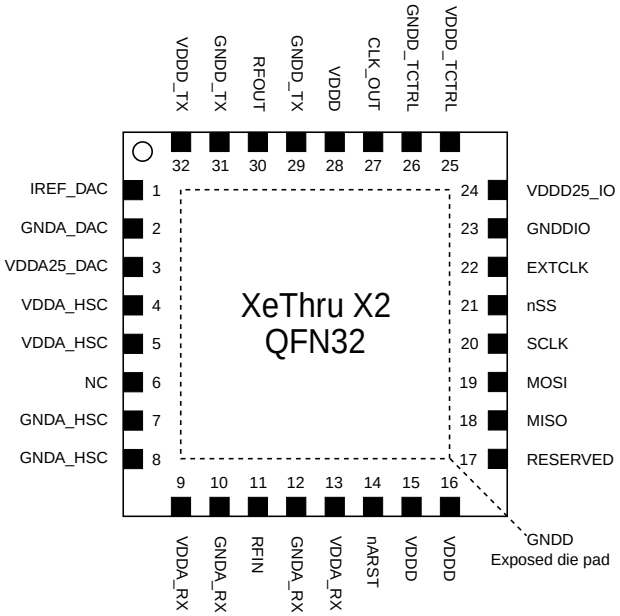


Figure 3.1. X2 pinout, top view, QFN32 package.

3.2. 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	IREF_DAC	Analog input	Reference current for internal DAC.
2	GNDA_DAC	Ground (analog)	DAC ground connection.
3	VDDA25_DAC	Power (analog)	DAC power supply (2.5 V).
4	VDDA_HSC	Power (analog)	HSC power supply (1.2 V).
5	VDDA_HSC	Power (analog)	HSC power supply (1.2 V).
6	NC	-	
7	GNDA_HSC	Ground (analog)	HSC ground connection.
8	GNDA_HSC	Ground (analog)	HSC ground connection.
9	VDDA_RX	Power (analog)	LNA power supply (1.2 V).
10	GNDA_RX	Ground (analog)	LNA ground connection.
11	RFIN	RF input	RF input from receiving antenna.
12	GNDA_RX	Ground (analog)	LNA ground connection.
13	VDDA_RX	Power (analog)	LNA power supply (1.2 V).
14	nARST	Digital input	Asynchronous reset (active low).
15	VDDD	Power (digital)	Digital core power supply (1.2 V).
16	VDDD	Power (digital)	Digital core power supply (1.2 V).
17	RESERVED	Digital input	Pull down to GNDD during normal operation.
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.

Pin	Pin name	Pin type	Description
21	nSS	Digital input	SPI Slave Select (active low).
22	EXTCLK	Digital input	External 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	VDDD_TCTRL	Power (digital)	Timing Controller and Sampler power supply (1.2 V).
26	GNDD_TCTRL	Ground (digital)	Timing Controller and Sampler ground connection.
27	CLK_OUT	Digital output	Clock output for external trigger.
28	VDDD	Power (digital)	Digital core power supply (1.2 V).
29	GNDD_TX	Ground (digital)	Transmitter ground connection.
30	RFOUT	RF output	RF output to transmitting antenna.
31	GNDD_TX	Ground (digital)	Transmitter ground connection.
32	VDDD_TX	Power (digital)	Transmitter power supply (1.2 V).

Table 3.1. X2 terminal functions.

4. Circuit Overview

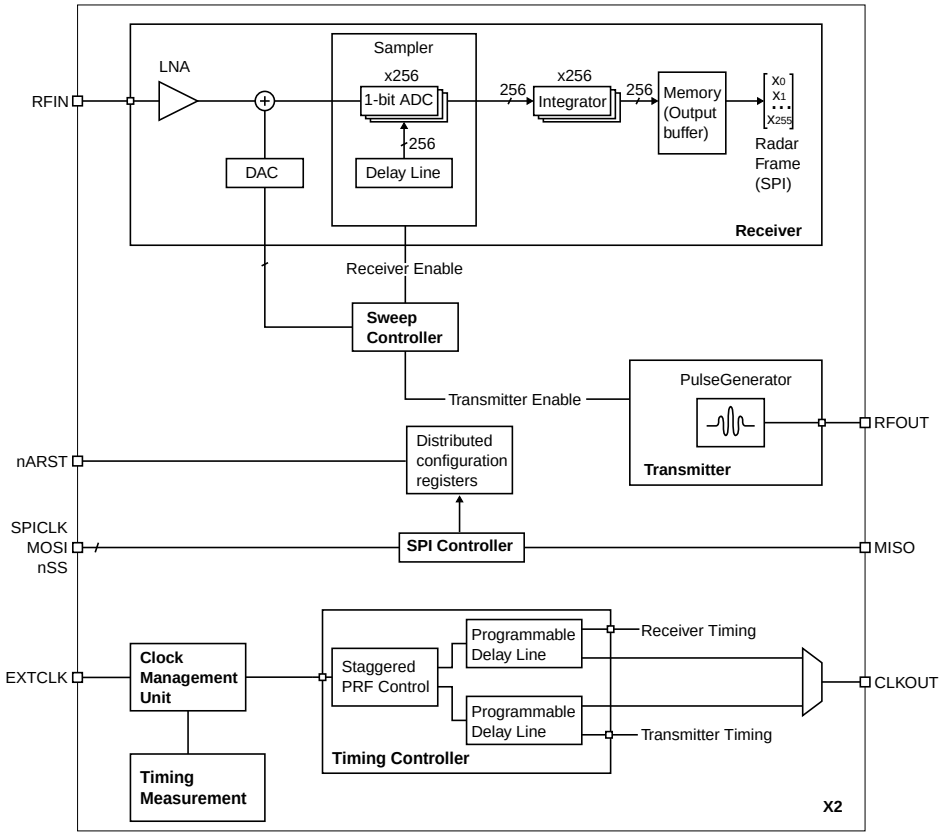


Figure 4.1. X2 architecture overview.

X2 is a complete transceiver for Ultra Wide Band/Impulse Radar applications. The basic components are a transmitter, a receiver and related control circuits, as shown in Figure 4.1. The system is configurable through a 4-wire Serial Peripheral Interface (SPI).

The receive path (RX) of the X2 consists of a Low Noise Amplifier (LNA), a Digital-to-Analog Converter (DAC), 256 Analog-to-Digital Converters (ADC) and 256 32-bit digital integrators, as well as an output memory buffer, accessible through the SPI. The RX is tightly integrated with the transmitter and is designed for coherent integration of the received energy.

The transmit path (TX) of the X2 consists of a pulse generator capable of generating pulses at a rate of up to 100 MHz. The output frequency can be measured and adjusted over a large frequency range to accommodate for regulatory requirements.

The timing of the RX and TX is individually adjustable through two sets of parallel delay lines. The relative difference in timing between the RX and TX path determines the spatial offset of the observed radar frame, and can be measured with the dedicated timing measurement circuit.

5. Serial Peripheral Interface

The Serial Peripheral Interface (SPI) is a 4-wire serial bus used for configuration and reading output from the radar system. The X2 is a SPI slave device connected to the SPI master as outlined in Figure 5.1.

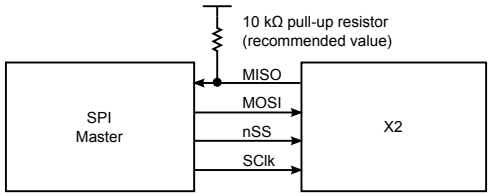


Figure 5.1. SPI master-slave connection.

The serial data transfer input (Master Out Slave In, MOSI), and output (Master In Slave Out, MISO), to the X2 are synchronized by the SPI clock (SClk). The Slave Select signal (nSS) must be low before and during transactions. Several transactions can be performed sequentially without waiting while nSS is low. MOSI is always read on the rising edge of SClk and MISO changes value on the falling edge of SClk (SPI mode 0, CPOL/CPHA = 0). See Figure 5.2 and Section 2.7 for timing specifications. Pulling nSS high is required after power up to reset the internal state of the SPI controller. A pull-up resistor is recommended to drive MISO to a defined voltage level as this output is floating while nSS is high in the default configuration. See the AsyncOutput configuration register for additional details and configuration options.

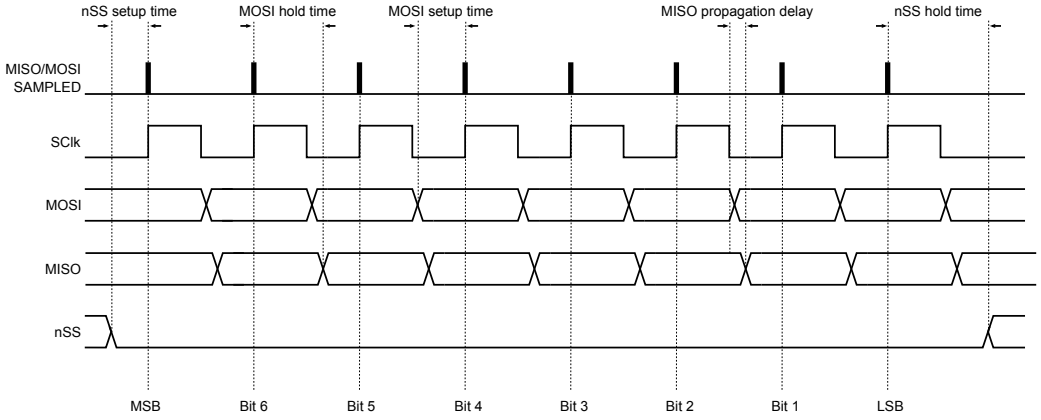


Figure 5.2. SPI timing requirements.

SPI transactions are either read, write, or action commands. In all cases, the most significant bit in each byte is transferred first, and the byte ordering is little endian (least significant byte first). The two first bytes are a command header signifying direction (read or write), address, continuation and length according to Table 5.1.

Byte #	Bit #	Field	Description
0	7 (MSb)	Read/Write	Read/Write select (0 = read selected, 1 = write selected).
0	6:0	Address	Address of the field to read from or write to, or the action to execute. The available registers are listed in Chapter 12.
1	7 (MSb)	Continue	Continued payload transaction (0 = operation starts from the first bit in the field, 1 = operation continues from the previous transaction). See Section 5.5 for additional details.
1	6:0	Length	Number of bytes to read or write in the current operation. See Section 5.4 for additional details.

Table 5.1. SPI transaction two byte header.

5.1. SPI Read Transactions

Read transactions read configuration registers and radar data from the X2. Read transactions have two phases: first, the master sends the command header requesting read operation, the address of the register to read from, and the number of bytes in the payload data (length), according to Table 5.1 on MOSI. In the second phase the data is sent to the master on MISO. Activity on MOSI is ignored in the second phase. The read transaction is illustrated in the timing diagram in Figure 5.3.

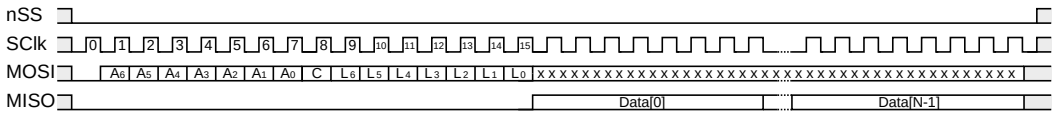


Figure 5.3. Bus activity diagram showing signal activity during SPI read transactions.

5.2. SPI Write Transactions

Write transactions write configuration settings to the X2. The master first sends the command requesting a write transaction, the register address to write to, and the number of bytes in the payload data (length), according to Table 5.1. On the clock cycle following the command, the data is written sequentially. The write transaction is illustrated in the timing diagram in Figure 5.4. MISO is low during this transaction.

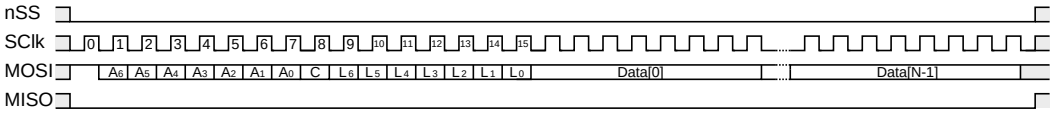


Figure 5.4. Bus activity diagram showing signal activity during SPI write transactions.

5.3. SPI Actions

Actions are a special case of write transactions that execute specific tasks rather than modify configuration registers. In this case the payload length is zero. The SPI action is illustrated in the timing diagram in Figure 5.5.

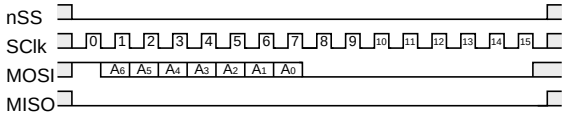


Figure 5.5. Bus activity diagram showing signal activity during an SPI action.

5.4. Extended Payload Transactions

Both read and write transactions support extended payload transactions by setting the length field to 127. In this case payload data is read or written until the master pulls nSS high.

5.5. Continued Transactions

The length field in the command header limits the transaction payload length to 126 bytes, unless extended payload transactions are selected. Continuing regular read or write transactions is possible by setting the Continue bit in the command header. In this case the read or write transaction continues from the previous read or write position of the requested field. Fields have separate pointers, and the read or write position is kept even if nSS is pulled high between transactions.

6. Sweep Controller

The purpose of the Sweep Controller is to coordinate the operation of the receiver during the acquisition of a radar frame. This process is referred to as a sweep because an important function of the Sweep Controller is to continuously program, or sweep, the DAC to generate appropriate threshold voltages in the receiver, see Chapter 7 for a description of the receiver system.

The sweep sequence is initiated by issuing the StartSweep executable action. The Sweep Controller sets the Sweeping bit in the SweepControllerStatus register while the sweep is in progress. The sweep status can be monitored by polling the register or the MISO output pin can be reconfigured to monitor the sweeping status asynchronously by programming the AsyncOutput configuration register. Sampling of the radar frame is complete when Sweeping goes low.

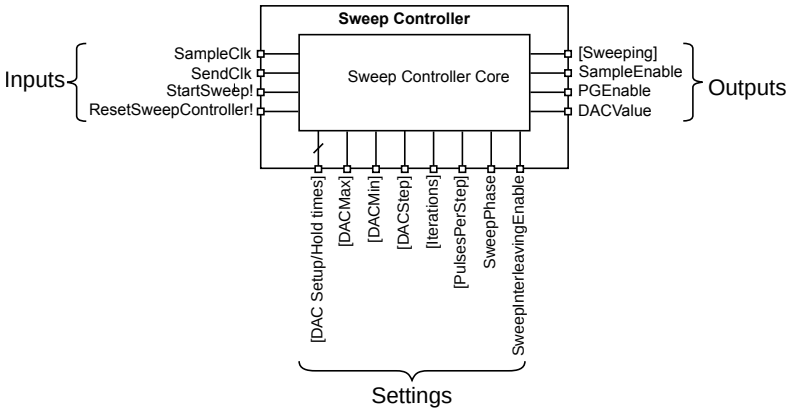


Figure 6.1. Sweep Controller block overview.

6.1. Principle of Operation

The state diagram in Figure 6.2 shows the principle of operation of the Sweep Controller state machine. Initially, the Sweep Controller is in the idle state. Issuing the ResetSweepController executable action will reset Sweep Controller to this state. Once the StartSweep executable action has been issued, the Sweep Controller proceeds to set the DAC to its initial value. The initial DACValue is either DACMax or DACMin, depending on the SweepPhase. After this, the Sweep Controller waits DACFirstIteration-SetupTime SampleClk cycles, giving the DAC time to settle at its initial value.

The internal SampleEnable signal is now high, enabling the Sampler to sample and accumulate the received signal at each rising SampleClk edge. After PulsesPerStep number of pulses are sampled and integrated, the SampleEnable signal goes low for DACRegularStepHoldTime SampleClk cycles.

The DACValue is then compared to either the DACMax or the DACMin register, depending on the direction of the sweep. If the DACValue is within the boundaries defined by these register, DACValue is decremented/incremented by DACStep and SampleEnable is held low for DACRegularStepSetupTime SampleClk cycles, before PulsesPerStep number of pulses are sampled. The Sweep Controller repeats changing the DACValue until it is beyond the boundaries.

If the DACValue is beyond 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 repeat the above sequence Iterations number of times.

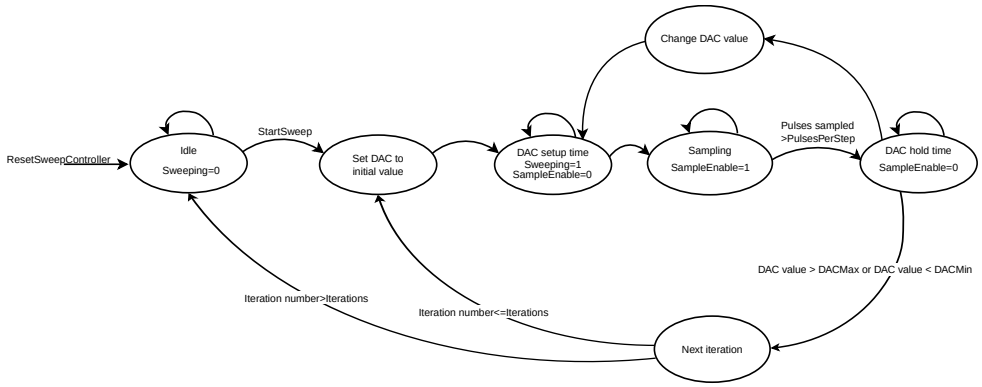


Figure 6.2. Sweep Controller state diagram.

6.2. Configuring Sweep Boundaries

The DAC sweep boundaries are configured through the DACMin and DACMax registers. Signal information is lost if the amplified input signal is outside the DAC values given by DACMin and DACMax.

Narrower DAC sweep boundaries will decrease the sweep time or allow more integration.

6.3. Configurable Step-Size

The size of the DAC increment is configurable through the DACStepCoarse and DACStepFine registers. Note that DACStepCoarse only has a function if SweepInterleaving is enabled, otherwise the DACStep is set by DACStepFine. Higher DACStep values means less integration, but higher speed and vice versa. See registers DACStep and SweepMainCtrl.

6.4. Sweeping the DAC

The Sweep Controller can be configured to perform sweeps in six different modes. Figure 6.3 illustrates the time-voltage relationship of the output of the DAC for each of the modes (numbered A through F in the figure).

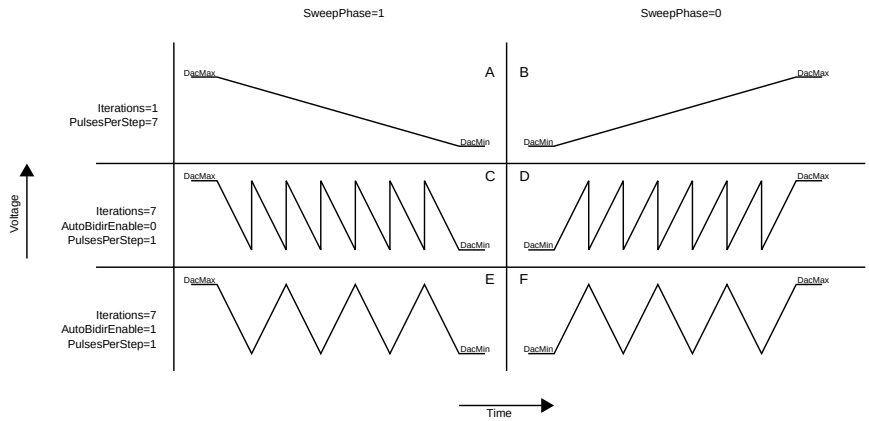


Figure 6.3. Time-voltage relationship of the sweep modes, together with their corresponding register settings.

There are several settings that control the behavior of the sweep. Plots A and B in Figure 6.3 illustrates the DAC output voltage for a sweep with Iterations = 1.

The sweep sequence can be split over several iterations without increasing the total level of integration. This spreads out the samples taken for a given DACValue. The resulting DAC output voltage is illustrated in plots C and D in Figure 6.3. Here, the Iterations register has been increased. To maintain the same total level of integration, the PulsesPerStep register have to be decreased accordingly.

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.

6.5. Sweep Interleaving

SweepInterleaving splits a sweep into several sub-sweeps where each sub-sweep uses DACStepCoarse as DACStep through the sub-sweep. Each sub-sweep is then offset with DACStepFine until all DAC values have been covered. See registers DACStep and SweepMainCtrl.

6.6. Processing Gain and Sweep Time

X2 uses coherent integration to achieve processing gain and the level of processing gain increase with higher integration. Increased integration can be achieved by increasing the number of pulses per step, by programming the PulsesPerStep configuration register, or by increasing the number of iterations, by programming the Iterations configuration register. The total integration is the product of these two values. The amount of processing gain is approximately doubled with twice the integration, resulting in a Signal-to-Noise Ratio (SNR) enhancement of 3 dB.

The Pulse Repetition Frequency (PRF) and total amount of integration determine the frame rate. Depending on the requirements of a given application the radar can be configured with more processing gain at the cost of lower frame rate, or higher frame rate at the cost of a lower SNR. The number of clock cycles required to complete the sweep with default DAC setup and hold times is

$$N_{\mathrm{SC}} = \mathrm{Iterations} \cdot (\mathrm{PulsesPerStep} + 1) \cdot \lfloor \frac{\mathrm{DACMax} - \mathrm{DACMin}}{\mathrm{DACStepFine}} \rfloor + 9 \tag{6.1}$$

The frame rate can then be calculated as

$$f_{\mathrm{framerate}} = \frac{\mathrm{PRF}}{N_{\mathrm{SC}}} \tag{6.2}$$

7. Receiver

The X2 receive path (RX) consists of a Low-Noise Amplifier (LNA), a Digital-to-Analog Converter (DAC), 256 1-bit high-speed Analog-to-Digital Converters (ADCs) and digital integrators, as well as an output memory buffer for radar frame readout over SPI. A block diagram of the RX path is shown in Figure 7.1

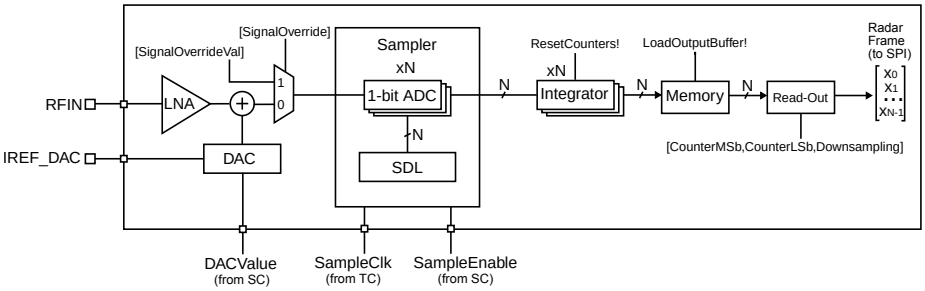


Figure 7.1. Receiver block diagram.

Sweeping of the DAC and sampling of the RF input signal is controlled by the Sweep Controller, see Chapter 6, through the signals DACValue and SampleEnable. The RF input signal is preamplified by the LNA and sampling is triggered by the SampleClk, originating from the TimingController, see Chapter 9. The SamplerDelayLine (SDL) generates a time offset between the 256 1-bit ADCs, synthesizing the system sampling rate.

7.1. Receiver Interface

After a sweep is completed, the captured radar frame is stored internally as an array of 32-bit integers. Before they can be accessed, the radar frame must be copied to the output buffer by issuing the LoadOutputBuffer executable action. At this point, the integrator can be reset by issuing the ResetCounters executable action and a new sweep can be started.

For information on how to access the output buffer, see Section 5.1 and Section 5.4. The number of bits that needs to be transferred for each frame depends on the number of samplers and the number of bits needed per sampler, which is configurable through the SamplerReadoutCtrl register.

7.2. Sampling Rate Estimation

The sampling rate of the system is given by the time offset between the individual sampling points of the radar frame. The exact value of this offset is not guaranteed by design, and is influenced by environmental factors such as temperature and supply voltage. X2 includes functionality that measures the actual sampling rate through StopWatch timing measurement, as explained in Section 11.1.1.

After a StopWatch measurement, the total delay of the Sampler Delay Line (SDL) is given by

$$\tau_{\mathrm{SDL}} = \frac{\mathrm{SDLMeasResult}}{\mathrm{SWMeasCyclesCounter} \times f_{\mathrm{MeasClk}}} \tag{7.1}$$

The mean radar frame sampling rate, f_s, can then be estimated with the following expression

$$f_s = \frac{N+1}{\tau_{\mathrm{SDL}}} \tag{7.2}$$

where N is the number of sampling points on the chip (256 on X2).

8. Transmitter

The X2 transmit path (TX) employs a high-order Gaussian approximation impulse generator, capable of transmitting high bandwidth impulses over a large range of frequencies. Figure 8.1 shows the basic block diagram of the transmitter. PGSelect and SendEveryPulse are user configurable functions, while PGEnable and SendClk are signals originating from the Sweep Controller and Timing Controller, respectively.

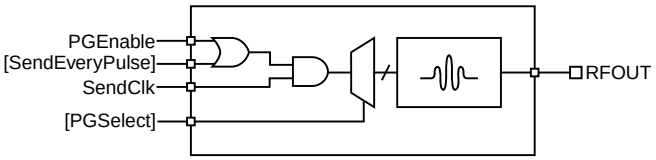


Figure 8.1. Transmitter block diagram.

The PGCtrl configuration register controls all Pulse Generator (PG) functions. The output center frequency (and hence relative bandwidth) is programmable through the PGSelect segment. There are 11 valid settings for pulse transmission, where higher values of PGSelect corresponds to higher center frequency of the transmitted pulse. During normal radar operation, pulse transmission is only desirable as long as sampling is enabled, i.e. during a sweep. However, the SendEveryPulse segment allows the user to override this behavior, transmitting pulses on every clock cycle.

8.1. Pulse Shape and Spectrum

Examples of output pulse frequency spectra (at PRF = 100 MHz) and time domain plots for all valid settings of PGSelect are given in Figure 8.2 - Figure 8.23.

8.1.1. Pulse Generator Output Spectra

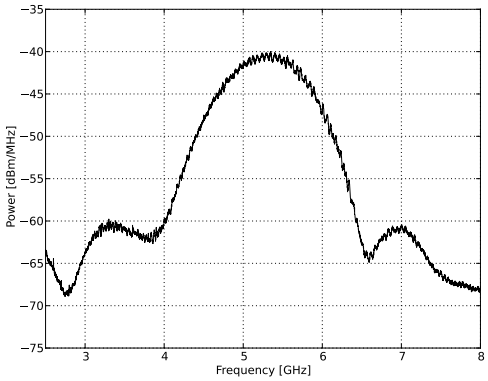


Figure 8.2. PGSelect = 0.

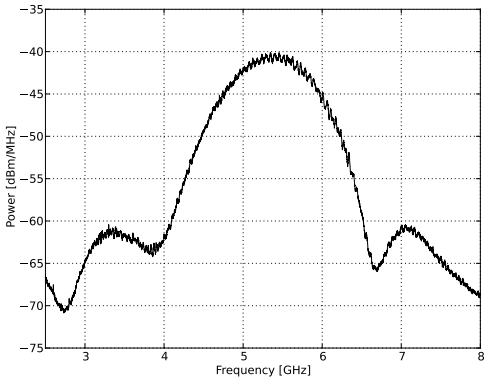


Figure 8.3. PGSelect = 1.

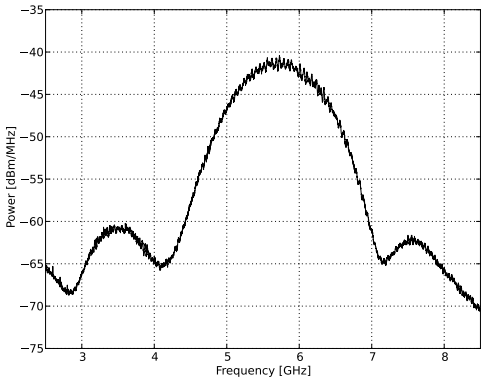


Figure 8.4. PGSelect = 2.

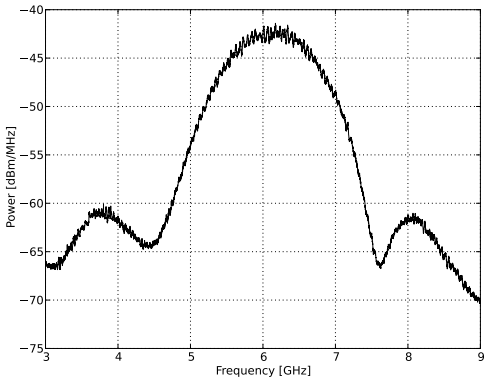


Figure 8.5. PGSelect = 3.

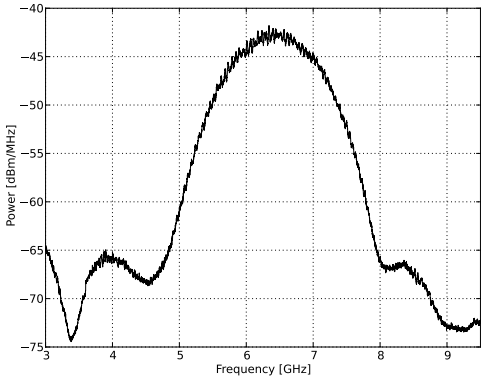


Figure 8.6. PGSelect = 4.

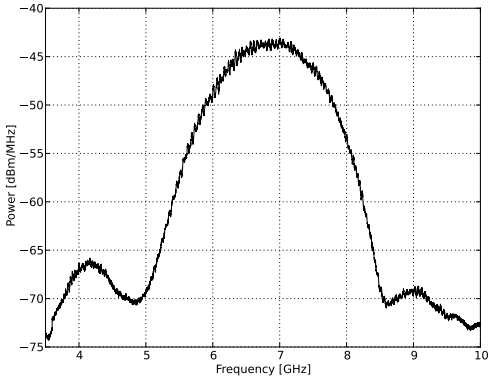


Figure 8.7. PGSelect = 5.

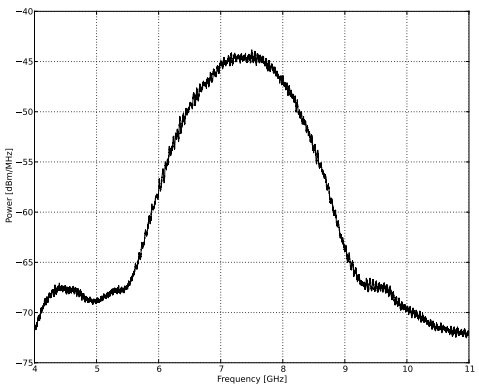


Figure 8.8. PGSelect = 6.

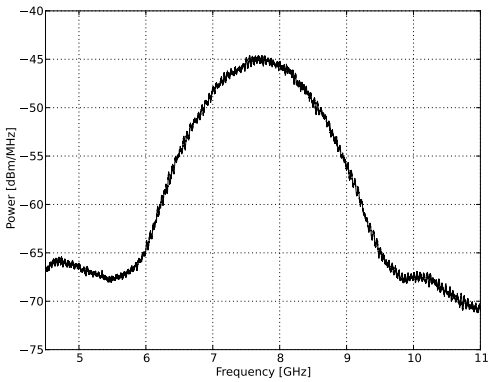


Figure 8.9. PGSelect = 7.

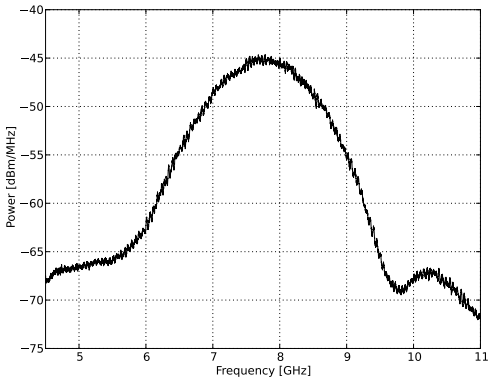


Figure 8.10. PGSelect = 8.

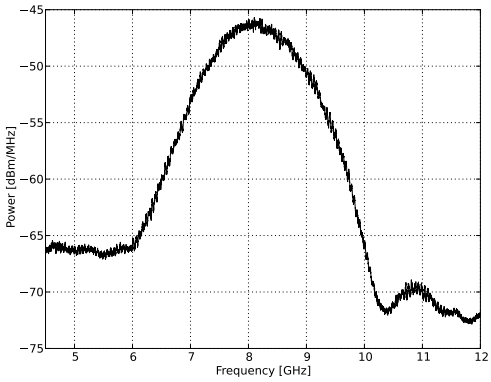


Figure 8.11. PGSelect = 9.

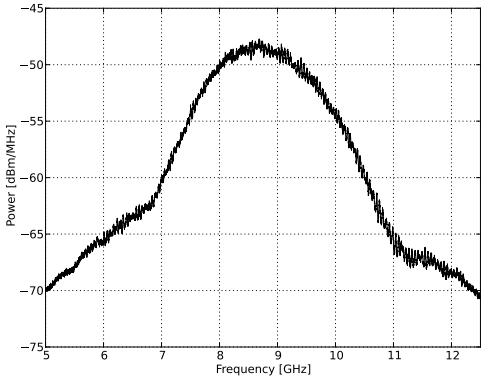


Figure 8.12. PGSelect = 10.

8.1.2. Pulse Generator Time Domain Output

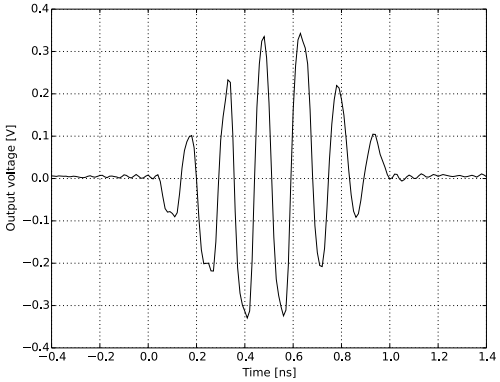


Figure 8.13. PGSelect = 0.

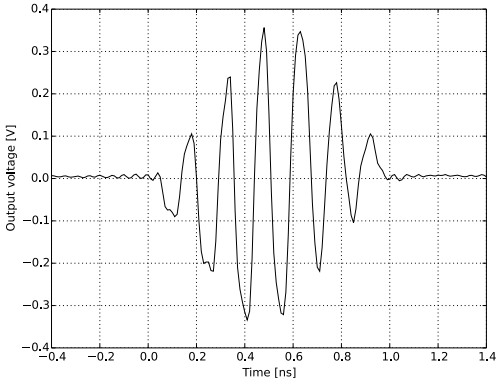


Figure 8.14. PGSelect = 1.

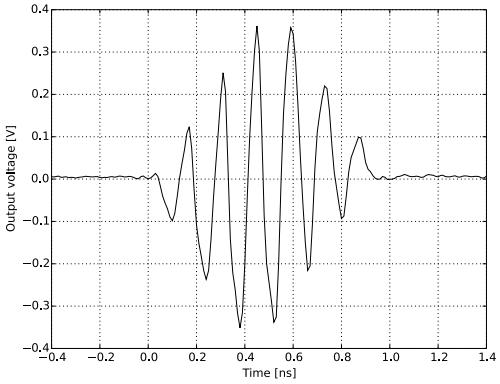


Figure 8.15. PGSelect = 2.

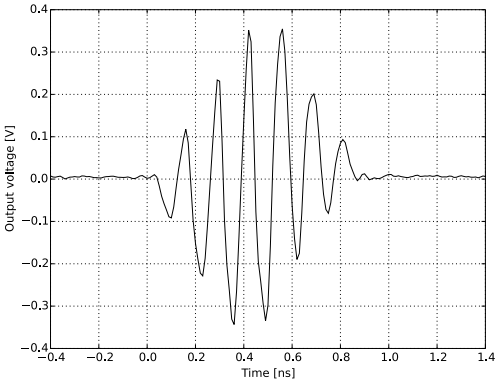


Figure 8.16. PGSelect = 3.

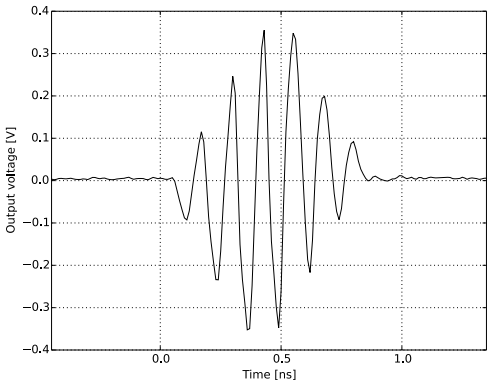


Figure 8.17. PGSelect = 4.

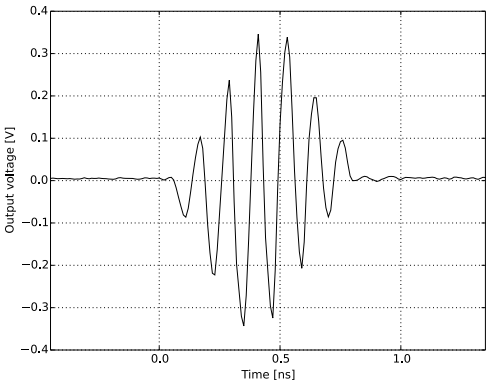


Figure 8.18. PGSelect = 5.

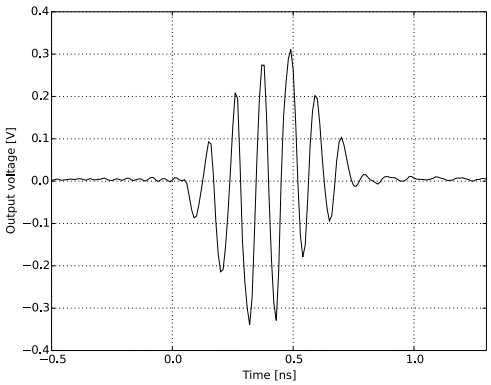


Figure 8.19. PGSelect = 6.

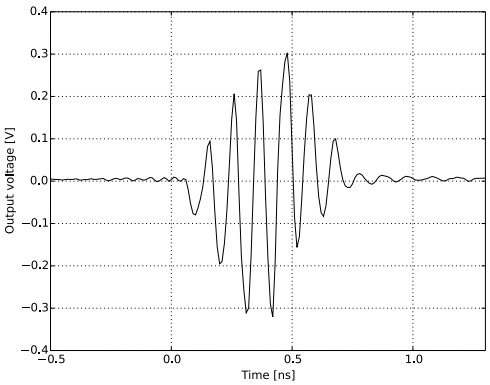


Figure 8.20. PGSelect = 7.

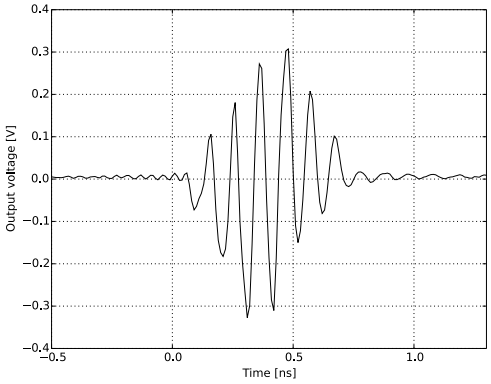


Figure 8.21. PGSelect = 8.

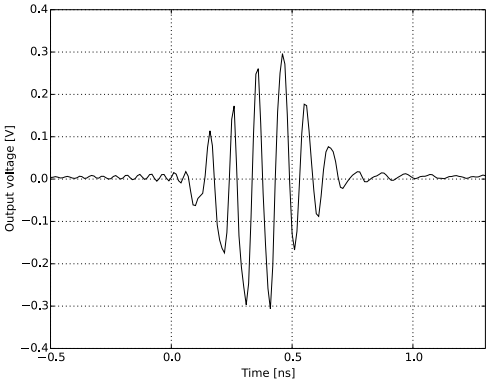


Figure 8.22. PGSelect = 9.

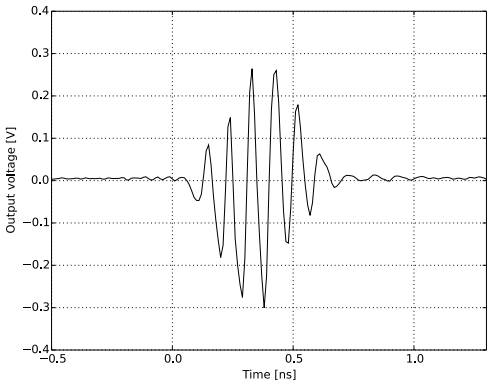


Figure 8.23. PGSelect = 10.

8.2. Output Center Frequency Estimation

The center frequency of the X2 PG output pulse is influenced by environmental factors, such as ambient temperature and supply voltage, and internal factors such as temperature variations due to variable computational load and power duty cycling. Knowledge and control of this parameter is important for many applications, including regulatory compliance. The center frequency at any given time is measurable through the StopWatch timing measurement function PGD, as explained in Chapter 11. The actual measured delay PGD is calculated with the formula:

$$\mathrm{PGD} = \frac{\mathrm{PGMeasResult}}{\mathrm{SWMeasCyclesCounter} \times f_{\mathrm{MeasClk}}} \tag{8.1}$$

The center frequency can then be estimated by the difference in time between the result of the measurement of the PGSelect in question and PGSelect = 12. The center frequency can be estimated through solving the following relationship for f_c:

$$\mathrm{PGD}(\mathrm{PGSelect} = \mathrm{n}) - \mathrm{PGD}(\mathrm{PGSelect} = 12) = af_c^3 + bf_c^2 + cf_c + d \tag{8.2}$$

Where a = -0.004388, b = 0.1083, c = -0.9831 and d = 4.033. PGD() is time in nanoseconds and f_c is the center frequency in GHz.

Figure 8.24 shows SW PGD measurement data for all valid PGSelect settings over the entire operational temperature range, correlated with the measured center frequency. The dashed line indicates the relationship in Equation 8.2.

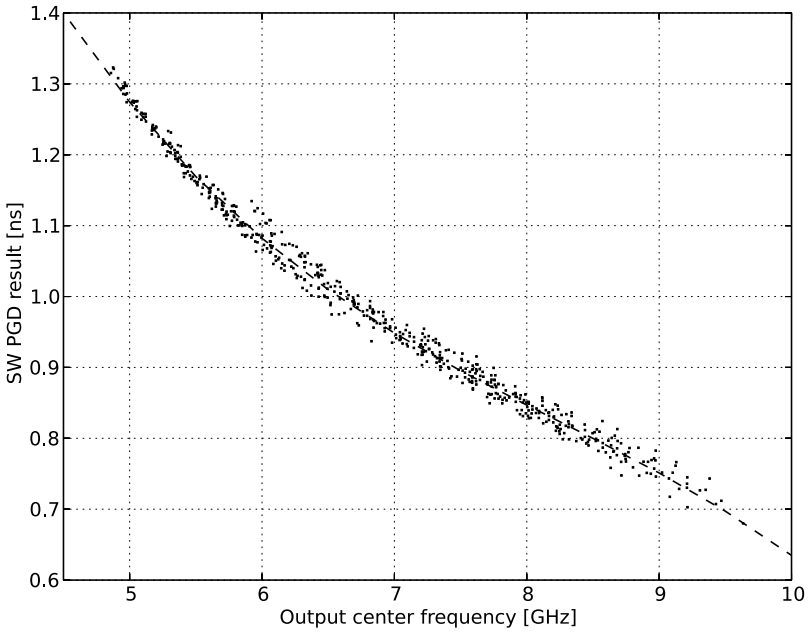


Figure 8.24. PG StopWatch results.

9. Timing Controller

A block diagram of the X2 Timing Controller is shown in Figure 9.1.

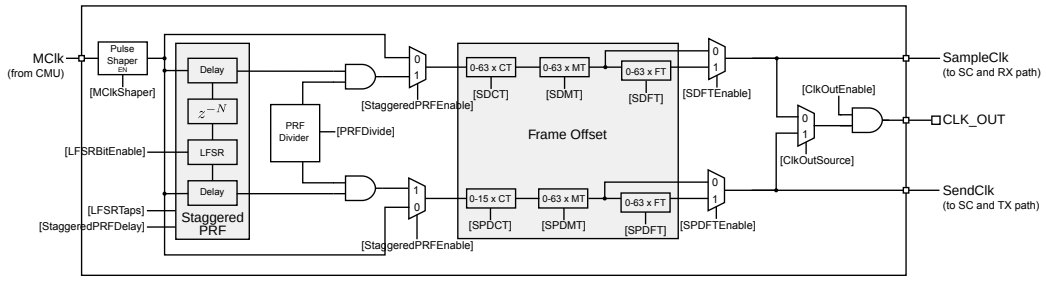


Figure 9.1. Timing controller block diagram.

9.1. Frame Offset

The X2 Timing Controller synthesizes the timing signals used for transmitting and sampling pulses. The Main Clock (MClk) is supplied from the CMU, see Chapter 10, and the Timing Controller splits this clock into two paths, each with separately programmable delays. The two clock paths are *SampleClk*, affected by the SampleDelay (SD) triggering the Sampler and *SendClk*, affected by the SendPulseDelay (SPD) triggering the Transmitter. The difference in time between these two signals, including a constant delay, make up the radar Frame Offset (FO); the start of the sampled radar frame.

$$\mathrm{FO} = \mathrm{SD} - \mathrm{SPD} + T_{\mathrm{application}} \tag{9.1}$$

$T_{application}$ is an application specific delay constant, including all on- and off-chip intrinsic delays, such as propagation delays due to internal routing, PCB routing, amplifiers, filters and antennas.

An important parameter of the X2 is the Pulse Repetition Frequency (PRF), the rate of pulse transmission and sampling of the radar system. The PRF is configurable through the CMU by configuring MClk, or directly in the Timing Controller by configuring the PRFDivide register. This relationship is given by:

$$\mathrm{PRF} = \frac{f_{\mathrm{MClk}}}{\mathrm{PRFDivide}} \tag{9.2}$$

Where f_{MClk} is the MClk frequency.

9.1.1. Configuring the Frame Offset

Each of the two delay paths constituting Frame Offset consist of three programmable delay lines with different delay granularity. CoarseTune (CT), MediumTune (MT) and FineTune (FT) are each individually programmable through registers SDCT, SDMT, SDFT, SPDCT, SPDMT and SPDFT. Refer to Section 2.7 for measurement data on delay unit resolution.

The FineTune elements are enabled by the SDFTEnable and SPDFTEnable segments of the TimingCtrl register. FTCtrl in the same register controls how FineTune elements are addressed. For power saving, CoarseTune elements can be individually enabled through the CoarseTuneCtrl register.

The StaggeredPRFDelay register can also be used to generate the desired Frame Offset. This register accepts values from 0 to 255, delaying the SampleClk signal with that amount of MClk cycles relative to SendPulse, creating FO in steps of MClk periods. The executable action StaggeredPRFReset must be issued each time StaggeredPRFDelay is updated. The StaggeredPRFEnable bit from LFSRCtrl must be set for StaggeredPRFDelay to be enabled. StaggeredPRFDelay can also be used in combination with CT/MT/FT to create an arbitrary FO.

The two delay lines SampleDelay and SendPulseDelay are individually measurable through the StopWatch function. Please refer to Chapter 11 for details on how to perform these measurements. After a StopWatch measurement, the current SD is calculated by:

$$\tau_{\mathrm{SD}} = \frac{\mathrm{SDMeasResult}}{\mathrm{SWMeasCyclesCounter} \times f_{\mathrm{MeasClk}}} \tag{9.3}$$

9.1.2. Frame Offset and Distance

Knowing the velocity factor of the transmission medium, FO can be translated directly to the spatial offset between the position of the radar and the start of the observed radar frame, d_{frame}, by the following relationship:

$$\mathrm{d_{frame}[m]} = \mathrm{FO[s]} \times \frac{c}{VF} \tag{9.4}$$

Where c is the speed of light and VF is the velocity factor of the transmission medium.

9.2. Staggered PRF

When the time taken for an echo of a transmitted pulse to return from a target is greater than the pulse repetition period, range ambiguity may occur. In simple radar systems, echoes from targets must be detected and processed before the next transmitter pulse is generated, greatly limiting the Maximum Unambiguous Range (MUR) of the system. In order to extend the MUR, X2 supports Staggered PRF, a transmission process where the time between each coherent pulse transmission/sample event changes slightly, in a patterned and readily-discernible repeating manner. This allows the X2 , on a pulse-to-pulse basis, to differentiate between returns from its own transmissions, as well as returns from interfering radar systems. The pulse staggering inserts a pseudo-random time offset between each pulse transmission/sample event, effectively extending the MUR well beyond the practical detection range.

X2 implements the random function of the Staggered PRF as a 20-bit configurable Linear Feedback Shift Register (LFSR), providing a pseudo-random code sequence which is translated into a pseudo-random time offset. Staggered PRF is enabled by default, and disabled by setting StaggeredPRFEnable = 0. The staggered PRF sequence spread is configurable through the LFSRBitEnable segment of the LFSRCtrl register. The SPI action StaggeredPRFReset is used to restart the staggering sequence. The LFSR polynomial is user configurable through the LFSRTaps register. Its default value ensures a maximum length pseudo-random spreading sequence.

9.3. External clock output

X2 supports the synchronization of several radars through the CLK_OUT pin. The ClkOutputCtrl register is used for controlling this function. ClkOutEnable enables routing of either SampleClk or SendClk, selectable through ClkOutSource, to CLK_OUT. This function can be used for a variety of applications, including radar beamforming and triggering of external pulse generators.

10. Clock Management

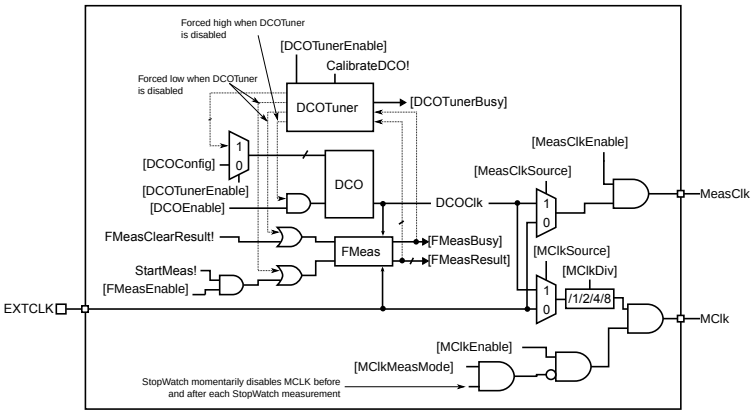


Figure 10.1. Clock management circuit schematic.

The Clock Management Unit (CMU) generates the internal clock signals in the X2. An outline of the clock management circuit is shown in Figure 10.1. The generated MClk is used to time the sending and receiving of radar pulses (see Chapter 9), and MeasClk is used for timing measurements (see Chapter 11). The internal clock signals are derived from the external clock input, EXTCLK, or an internally generated Digitally Controlled Oscillator (DCO) clock signal. These clock sources can be selected independently as sources for MClk and MeasClk by programming MClkSource and MeasClkSource in the CMUConfig configuration register respectively. However, it is recommended to use the DCO as the source for MeasClk and EXTCLK as the source for MClk to be able to reliably perform timing measurements (see Chapter 11). If MClk and MeasClk are generated from the same source, timing measurements will not yield the expected results. MClk and MeasClk are gated by programming the MClkEnable and MClkEnable bits. Additionally, the MClk clock source can be divided by 2, 4, or 8 by programming MClkDiv.

10.1. The DCO Clock Source

The DCO is a free-running on-chip oscillator that generates a clock signal when the DCOEnable bit in the CMUConfig configuration register is set. The output frequency is selected by the DCOConfigVal segment in the same register. The DCO must be disabled prior to programming DCOConfigVal. The actual oscillator frequency is influenced by environmental conditions, such as temperature and supply voltage. Therefore, the DCO subsystem contains a Frequency Measurement (FMeas) function to reference the DCO frequency to the external clock frequency, which is assumed to be a precise and known value. Further, the DCO frequency can be set automatically by means of the DCO tuner. The DCO tuner algorithmically configures the DCO until the target frequency setting is obtained. The tuning is initiated by issuing the CalibrateDCO executable action. Because the DCO is free-running, this action must be executed regularly to compensate for frequency drift resulting from changes in operating temperature, supply voltage and internally generated oscillator phase noise. As the DCO tuner is programming the DCO during the calibration, the DCO clock signal is not usable during calibration.

10.1.1. Manually Configuring and Enabling the DCO

The following steps are required to configure and enable the DCO (other registers are assumed to be at their default value):

1. Configure the DCO setting if desired by programming the DCOConfigVal segment in the CMUConfig configuration register. The default setting will run the DCO at a suitable frequency for timing measurements, but the absolute frequency depends on environmental conditions and must be measured prior to use. See Section 10.1.2.

2. Configure the DCO output divider if desired by programming the DCOOutputDiv bit in the CMUConfig configuration register. The default setting outputs the DCO clock signal directly. Setting DCOOutputDiv = 1 halves the frequency of the DCO clock signal.

3. Start the DCO by setting DCOEnable = 1 in the CMUConfig configuration register.

To reconfigure the DCO setting the DCO must be disabled by setting DCOEnable = 0 before programming DCOConfigVal.

10.1.2. Measuring the DCO Frequency

The following steps are required to measure the current DCO frequency when the DCO is enabled and running (other registers are assumed to be at their default value):

1. Set FMeasEnable = 1. See the CMUConfig configuration register for details.
2. Configure FMeasNumCycles to specify the number of clock cycles used to estimate the DCO frequency. The number of clock cycles influence the precision, accuracy and time required to run the frequency measurement used in the tuning algorithm. See Section 10.1.4 for details.
3. Clear the frequency measurement register by executing the FMeasClearResult SPI action.
4. Run the frequency measurement by executing the StartMeas SPI action.
5. Wait until the frequency measurement is finished by monitoring the FMeasBusy status bit in the DCOTunerStatus configuration register.
6. Read the frequency measurement result from FMeasResult. The DCO frequency measurement result is converted to the absolute DCO frequency using the following equation:

$$f_{\mathrm{DCO}} = f_{\mathrm{EXT}} \times \frac{\mathrm{FMeasResult}}{\mathrm{FMeasNumCycles}} \tag{10.1}$$

Where f_{DCO} is the absolute DCO frequency and f_{EXT} is the external clock frequency.

10.1.3. Automatically Tuning the DCO Frequency

The following steps are required to algorithmically set the DCO frequency using the DCO tuner function when the DCO is enabled and running (other registers are assumed to be at their default value):

1. Set DCOTunerEnable = 1. See the CMUConfig register for details.
2. Configure FMeasNumCycles to specify the number of clock cycles used to estimate the DCO frequency. The number of clock cycles influence the precision, accuracy and time required to run the frequency measurement used in the tuning algorithm. See Section 10.1.4 for details.
3. Configure the DCOTarget register. The DCOTarget setting is related to the target DCO frequency by the following equation:

$$\mathrm{DCOTarget} = \mathrm{FMeasNumCycles} \times \frac{f^{*}_{\mathrm{DCO}}}{f_{\mathrm{EXT}}} \tag{10.2}$$

Where f^{*}_{DCO} is the target DCO frequency and f_{EXT} is the external clock frequency.

4. Run the DCO calibration algorithm by issuing the CalibrateDCO executable action.
5. Wait until the DCO tuner finishes configuring the DCO by monitoring the DCOTunerBusy status bit in the DCOTunerStatus configuration register.

After calibration the DCO is running at a frequency close to the target frequency and no further action is required. The DCOTunerResult field in the DCOTunerStatus is the DCO setting found by the DCO tuner algorithm. This value is analogues to the DCOConfigVal in Section 10.1.1. However, DCOConfigVal is not controlling the DCO setting when DCOTunerEnabled is active. Keep DCOTunerEnable = 1 during operation, and repeat from step 2 regularly to compensate for frequency drift resulting from changes in environmental conditions and oscillator phase noise accumulation.

10.1.4. DCO Frequency Measurement Precision, Accuracy and Run Time

The frequency measurement function can be used directly as described in Section 10.1.2 or indirectly by means of the DCO tuner as described in Section 10.1.3. In both cases the FMeasNumCycles configuration register is used to specify the number of clock cycles of the external clock to use for the measurement. This value is directly proportional to the run time of the measurement:

$$\frac{\mathrm{FMeasNumCycles}}{f_{\mathrm{EXT}}} \quad (\mathrm{Run\ time}) \tag{10.3}$$

Where f_{EXT} is the frequency of the external clock signal. FMeasNumCycles scales the result of the frequency measurement, FMeasResult. The value of the result is inversely proportional to the theoretical precision of the measurement:

$$\frac{2}{\mathrm{FMeasResult} \times f_{\mathrm{DCO}}} \quad (\mathrm{Absolute\ precision}) \tag{10.4}$$

Where f_{DCO} is the measured DCO frequency. Finally, the accuracy of the measurement depends on both FMeasNumCycles and FMeasResult, and the accuracy of the external clock signal:

$$\mathrm{Accuracy}_{\mathrm{EXT}} \times \frac{\mathrm{FMeasNumCycles}}{\mathrm{FMeasResult}} \quad (\mathrm{Accuracy}) \tag{10.5}$$

Where $\mathrm{Accuracy}_{\mathrm{EXT}}$ is the accuracy of the external clock signal.

11. Timing Measurement

Delay elements are employed for timing generation throughout the X2 signal path. These delay lines generate the timing for Frame Offset, sampling rate and pulse transmission. As the absolute delays are influenced by temperature and supply voltage, the ability to accurately measure these delays is imperative for radar signal quality and precision. X2 employs a timing measurement technique powered by an on-chip StopWatch (SW) circuit in order to precisely and dynamically measure delays, requiring minimal user effort. SW provides the ability to measure:

SampleDelay (SD)	The delay line generating SampleClk, described in Section 9.1.
SendPulseDelay (SPD)	The delay line generating SendClk, described in Section 9.1.
Sampler Delay Line (SDL)	The delay relating to the system sampling rate, described in Section 7.2.
Pulse Generator Delay (PGD)	The delay relating to the center frequency of the transmitted pulse, described in Section 8.2.

11.1. Principle of Operation

Figure 11.1 shows a functional block diagram of the StopWatch circuit, including the relevant delay lines. Each of the delay lines can be measured in parallel and can be measured while performing a sweep. In preparation for an SW measurement, MeasClkEnable, MClkMeasMode and SWMeasEnable must be set high. SDLMeasEnable must be set high to measure the sampling rate. SendEveryPulse must be set high to measure the Pulse Generator Delay. Measurements are started by issuing the StartMeas executable action which initiates both the SW and FMeas measurements at the same time. The Staggered PRF can be enabled while measuring delays with SW. SWMeasBusy is high during a measurement and the length of a measurement is controlled by the SWSweepMode.

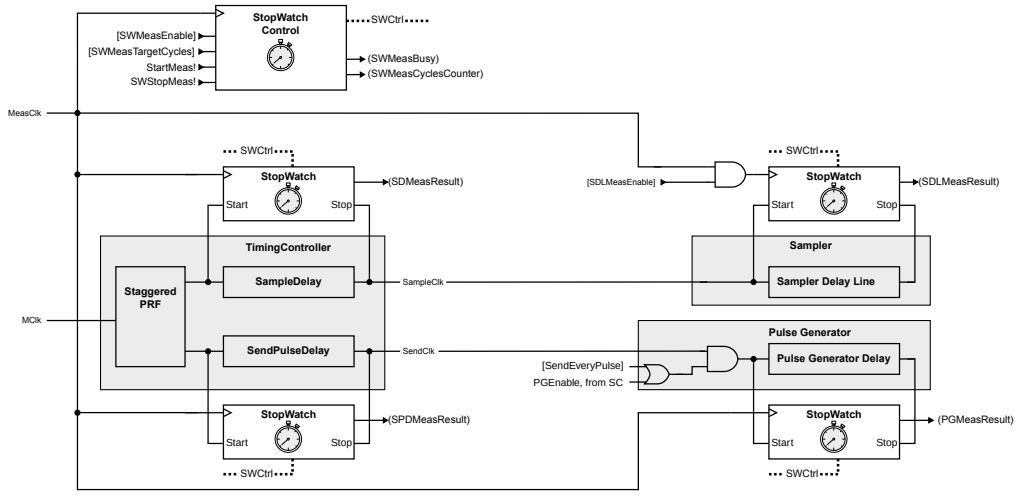


Figure 11.1. Block diagram showing signals important for timing measurement functionality.

After a SW measurement is completed, the measured values are available in SDMeasResult, SPDMeasResult, PGMeasResult, SDLMeasResult and SWMeasCyclesCounter.

The internal DCO generates the SW measurement clock, MeasClk. MeasClk must be uncorrelated and the frequency of MeasClk must be known in order to convert the output of the SW circuits to a delay in seconds. For more details on how to configure and measure the DCO frequency, refer to Chapter 10.

11.1.1. Performing a StopWatch Measurement

1. Set the DCO frequency, recommended frequency is approximately 211 MHz, see Section 10.1.3 for details.

2. The main SW functionality, including SD and SPD measurements, is enabled by setting SWMeasEnable = 1, MeasClkEnable = 1 and MClkMeasMode = 1.

   a. Measuring of the SDL is enabled by setting SDLMeasEnable = 1.

   b. Measuring of the PGD is enabled by setting SendEveryPulse = 1. Note that this will enable pulse transmission on every MClk cycle.

3. Clear the frequency measurement register by issuing the FMeasClearResult executable action.

4. The StartMeas executable action starts the measurement routines of both the FMeas and StopWatch measurement circuits.

5. Wait until the MeasClk frequency measurement is complete, signaled by FMeasBusy going low and the SW measurement is complete signaled by SWMeasBusy going low.

6. The SW measurement related results are available in SDMeasResult, SPDMeasResult, PGMeasResult, SDLMeasResult and SWMeasCyclesCounter.

11.1.2. Calculating the Measured Delays

The measured delay can be calculated with the formula:

$$\tau = \frac{\text{MeasurementResult}}{\text{SWMeasCyclesCounter} \times f_{\text{MeasClk}}} \tag{11.1}$$

where τ is the measured delay, MeasurementResult is one of SDMeasResult, SPDMeasResult, PGMeasResult, SDLMeasResult and $f_{MeasClk}$ is the frequency of the measurement clock. SWMeasCyclesCounter relates directly to the precision of the delay measurements. Please refer to Section 2.7 for data on this relationship. The $f_{MeasClk}$ can be calculated by the relation given in Equation 10.1

11.2. StopWatch Measurement Time

The StopWatch measurement time is given by:

$$t_{\text{meas}} = 2 \times \frac{\text{SWWaitCycles}}{f_{\text{MeasClk}}} + \frac{\text{SWMeasTargetCycles}}{f_{\text{MClk}}} \tag{11.2}$$

and will in practice be dominated by SWMeasTargetCycles; how many times the delay is measured. Measurement with a resolution of 16 ps std, $f_{MeasClk}$ = 211 MHz, f_{MClk} = 100 MHz gives a measurement time of ~1 ms.

12. Configuration Registers

Address	Name	Type
0x00	ForceZero	Register
0x01	ForceOne	Register
0x02	ChipID	Register
0x03	AsyncOutput	Register
0x20	SamplerOutputBuffer	Memory
0x21	SamplerReadoutCtrl	Register
0x24	LoadOutputBuffer	Action
0x25	ResetCounters	Action
0x26	SamplerInputCtrl	Register
0x2A	RXFECtrl	Register
0x31	PulsesPerStep	Register
0x32	DACFirstIterationSetupTime	Register
0x33	DACFirstStepSetupTime	Register
0x34	DACRegularStepSetupTime	Register
0x35	DACLastIterationHoldTime	Register
0x36	DACLastStepHoldTime	Register
0x37	DACRegularStepHoldTime	Register
0x38	SweepMainCtrl	Register
0x39	DACMax	Register
0x3A	DACMin	Register
0x3B	DACStep	Register
0x3C	Iterations	Register
0x43	StartSweep	Action
0x44	ResetSweepController	Action
0x47	SweepControllerStatus	Register
0x50	PGCtrl	Register
0x51	DACCtrl	Register
0x58	CMUConfig	Register
0x59	FMeasNumCycles	Register
0x5A	DCOTarget	Register
0x5B	FMeasResult	Register
0x5C	DCOTunerStatus	Register
0x5D	StartMeas	Action
0x5E	CalibrateDCO	Action
0x5F	FMeasClearResult	Action
0x60	PRFDivide	Register
0x61	LFSRCtrl	Register
0x62	StaggeredPRFDelay	Register
0x63	StaggeredPRFReset	Action
0x64	LFSRTaps	Register
0x6A	TimingCtrl	Register

Address	Name	Type
0x6B	SDCT	Register
0x6C	SDMT	Register
0x6D	SDFT	Register
0x6E	SPDCT	Register
0x6F	SPDMT	Register
0x70	SPDFT	Register
0x71	CoarseTuneCtrl	Register
0x76	ClkOutputCtrl	Register
0x77	PGMeasResult	Register
0x78	SDLMeasResult	Register
0x79	SDMeasResult	Register
0x7A	SPDMeasResult	Register
0x7B	SWStopMeas	Action
0x7C	SWCtrl	Register
0x7D	SWMeasCyclesCounter	Register
0x7E	SWMeasTargetCycles	Register
0x7F	SWMeasBusy	Register

Table 12.1. Configuration registers overview.

0x00 ForceZero

Read-only 8-bit register.

Bit	Segment name	Default value	Description
[7:0]	ForceZero	0x00	Always returns 0x00.

0x01 ForceOne

Read-only 8-bit register.

Bit	Segment name	Default value	Description
[7:0]	ForceOne	0xFF	Always returns 0xFF.

0x02 ChipID

Read-only 16-bit register.

Bit	Segment name	Default value	Description
[15:0]	ChipID	0x0309	Always returns the 16-bit chip identification number 0x0309.

0x03 AsyncOutput

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:4]	-	NA	

Bit	Segment name	Default value	Description
[3]	MISOHiZEnable	0x01	When enabled, the MISO pin is in high impedance mode when nSS is high, and active when nSS is low. Disable to output AsyncOutput even when nSS is high.
[2:0]	AsyncOutput	0x00	Configures the MISO pin function. 0: MISO 1: Sweeping 2: DCOTunerBusy 3: FMeasBusy 4: SWMeasBusy

0x20 SamplerOutputBuffer

Read-only memory.

The result from the last sweep is placed in this register after the LoadOutputBuffer (0x24) SPI action is called. Users can choose to read all or parts of this register by adjusting settings in SamplerReadoutCtrl (0x21).

0x21 SamplerReadoutCtrl

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:12]	-	NA	
[11:7]	CounterMSb	0x1F	Selects the most significant bit to read from each sampler. Can be any value from 0 to 31, but must always be higher than or equal to CounterLSb.
[6:2]	CounterLSb	0x00	Selects the least significant bit to read from each sampler. Can be any value from 0 to 31, but must always be lower than or equal to CounterMSb.
[1:0]	Downsampling	0x00	Selects the level of downsampling. 0: No downsampling, all samplers are read out. 1: Sampler number [0, 2, 4, ...] are read out. 2: Sampler number [0, 4, 8, ...] are read out. 3: Sampler number [0, 8, 16, ...] are read out.

0x24 LoadOutputBuffer

Executable Action.

Loads SamplerOutputBuffer (0x20) with the result from the last sweep.

0x25 ResetCounters

Executable Action.

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

0x26 SamplerInputCtrl

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:1]	-	NA	
[0]	SampleEveryPulse	0x00	Overrides SampleEnable signal from the Sweep Controller. When enabled, the Sampler will sample its inputs on every rising clock edge, even if the Sweep Controller is inactive.

0x2A RXFECtrl

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:4]	-	NA	
[3]	SignalOverride	0x00	Enables signal override at the sampler input. The override value is configured with the SignalOverrideVal bit.
[2]	SignalOverrideVal	0x00	Sets the signal override value.
[1]	Reserved	0x00	Always write as 0.
[0]	Reserved	0x00	Always write as 0.

0x31 PulsesPerStep

Read-writable 24-bit register.

Bit	Segment name	Default value	Description
[23:0]	PulsesPerStep	0x14	Number of pulses integrated per DAC step.

0x32 DACFirstIterationSetupTime

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:0]	DACFirstIterationSetupTime	0x0A	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.

0x33 DACFirstStepSetupTime

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:0]	DACFirstStepSetupTime	0x01	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.

0x34 DACRegularStepSetupTime

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:0]	DACRegularStepSetupTime	0x01	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 (0x3B) register, this may or may not be necessary. In general, larger steps require longer setup time.

0x35 DACLastIterationHoldTime

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:0]	DACLastIterationHoldTime	0x00	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.

0x36 DACLastStepHoldTime

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:0]	DACLastStepHoldTime	0x00	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.

0x37 DACRegularStepHoldTime

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:0]	DACRegularStepHoldTime	0x00	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.

0x38 SweepMainCtrl

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:4]	-	NA	
[3]	SweepInterleavingEnable	0x00	Enables Sweep Interleaving.
[2]	SweepPhase	0x00	Sets the initial direction of a bi-directional sweep. 0: First sweep starts at DACMin (0x3A), ends at DACMax (0x39). 1: First sweep starts at DACMax (0x39), ends at DACMin (0x3A).

Bit	Segment name	Default value	Description
[1]	Reserved	0x00	Always write as 0.
[0]	AutoBidirEnable	0x01	If this bit is set, bi-directional sweeps are enabled.

0x39 DACMax

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:13]	-	NA	
[12:0]	DACMax	0x1FF8	Max value of DAC sweep. Always set this field higher than or equal to DACMin (0x3A).

0x3A DACMin

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:13]	-	NA	
[12:0]	DACMin	0x00	Min value of DAC sweep. Always set this field lower than or equal to DACMax (0x39).

0x3B DACStep

Read-writable 32-bit register.

Bit	Segment name	Default value	Description
[31:26]	-	NA	
[25:13]	DACStepCoarse	0x08	Coarse step size of DAC sweep. Always set equal to or higher than DACStepFine. These bits have no function if Sweep Interleaving is disabled.
[12:0]	DACStepFine	0x08	Fine step size of DAC sweep if Sweep Interleaving is enabled, normal step size if Sweep Interleaving is disabled.

0x3C Iterations

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:0]	Iterations	0x0A	Number of sweep iterations to perform.

0x43 StartSweep

Executable Action.

Signals the Sweep Controller to start a new sweep.

0x44 ResetSweepController

Executable Action.

Aborts any running sweep and resets the internal Sweep Controller state machines. This action does not reset any of the SPI registers.

0x47 SweepControllerStatus

Read-only 8-bit register.

Bit	Segment name	Default value	Description
[7:1]	-	NA	
[0]	Sweeping	0x00	High while a sweep is in progress.

0x50 PGCtrl

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:6]	-	NA	
[5]	Reserved	0x00	Always write as 0.
[4]	SendEveryPulse	0x00	When disabled, pulse transmission is only enabled while a sweep is in progress. Set this bit to enable pulse transmission on every clock cycle. Note that this bit must be enabled to run Stop-Watch measurement for output center frequency estimation.
[3:0]	PGSelect	0x05	0-10: Pulse transmission is enabled. Higher values corresponds to increasing transmitter output center frequency. 11: Do not use. 12: Pulse transmission is disabled. This setting is used as a baseline during the SW center frequency estimation. 13-15: Pulse transmission is disabled.

0x51 DACCtrl

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:14]	-	NA	
[13]	DACOverride	0x00	When this bit is set, the DAC is controlled by DACOverrideVal. When this bit is cleared, the DAC is controlled by the Sweep Controller.
[12:0]	DACOverrideVal	0x00	Can be used to force the DAC to a specific value. Requires DACOverride to be high.

0x58 CMUConfig

Read-writable 40-bit register.

Bit	Segment name	Default value	Description
[39:36]	-	NA	
[35:20]	DCOTunerWaitCycles	0x0A	Number of cycles the DCOTuner waits after updating the DCO settings.

Bit	Segment name	Default value	Description
[19]	DCOOutputDiv	0x00	0: The DCO output is not divided and no specific duty cycle can be guaranteed. 1: The DCO output is divided by two and the duty cycle is guaranteed to be 50%.
[18:10]	DCOConfigVal	0xD8	Configures the DCO frequency. Higher numbers correspond to lower clock frequencies. This setting is only used if the DCOTuner is disabled. Always disable the DCO (see DCOEnable) before changing this value.
[9]	DCOEnable	0x00	0: The DCO is disabled. 1: The DCO is enabled.
[8]	MClkSource	0x00	0: MClk is sourced from the EXTCLK pin. 1: MClk is sourced from the DCO.
[7:6]	MClkDiv	0x00	MClk divide-by value. $f_{\mathrm{MClk}} = \frac{f_{\mathrm{ClkSource}}}{2^{\mathrm{MClkDiv}}}, \quad 0 \leq \mathrm{MClkDiv} \leq 3$ $f_{\mathrm{MClk}} = 0, \quad \mathrm{MClkDiv} > 3$
[5]	MClkEnable	0x01	Enables internal distribution of the MClk signal.
[4]	DCOTunerEnable	0x00	0: DCOTuner is disabled. The DCO must be manually configured through DCOConfigVal. 1: DCOTuner is enabled. The DCO is configured by the DCOTuner.
[3]	FMeasEnable	0x00	0: FMeas is disabled. 1: FMeas is enabled. DCOClk is measured when StartMeas (0x5D) SPI Action is called.
[2]	MeasClkSource	0x01	0: MeasClk is sourced from the EXTCLK pin. 1: MeasClk is sourced from the DCO.
[1]	MClkMeasMode	0x00	Allows the StopWatch circuit to disable the MClk, must be enabled during SW timing measurement.
[0]	MeasClkEnable	0x00	Enables internal distribution of the MeasClk signal. Note that SDLMeasEnable must be set in order to distribute MeasClk to the Sampler Delay Line. Must be enabled during SW timing measurement.

0x59 FMeasNumCycles

Read-writable 32-bit register.

Bit	Segment name	Default value	Description
[31:0]	FMeasNumCycles	0x2710	Sets number of EXTCLK measurement cycles to use for FMeas.

0x5A DCOTarget

Read-writable 32-bit register.

Bit	Segment name	Default value	Description
[31:0]	DCOTarget	0x526C	Sets the target value for the DCOTuner.

0x5B FMeasResult

Read-only 32-bit register.

Bit	Segment name	Default value	Description
[31:0]	FMeasResult	0x00	Contains the result of the previous DCOClk measurement.

0x5C DCOTunerStatus

Read-only 16-bit register.

Bit	Segment name	Default value	Description
[15:11]	-	NA	
[10:2]	DCOTunerResult	0x00	DCO configuration value result from the last CalibrateDCO (0x5E) session.
[1]	DCOTunerBusy	0x00	High while the DCOTuner is busy.
[0]	FMeasBusy	0x00	High while the FMeas circuit is busy.

0x5D StartMeas

Executable Action.

Starts the measurement routines of both the FMeas and StopWatch measurement circuits. FMeas and StopWatch can be individually disabled with the FMeasEnable and StopWatchEnable control bits.

0x5E CalibrateDCO

Executable Action.

Signals the DCOTuner to start the DCO tuning routine. The DCOTunerBusy bit will go high and stay so until the tuning routine has completed.

0x5F FMeasClearResult

Executable Action.

Clears the results from FMeas counters.

0x60 PRFDivide

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:0]	PRFDivide	0x01	Provides the ability to use a lower PRF without reducing the accuracy of the StaggeredPRFDelay. Note that if this register is set to 0, sampling will be disabled completely. $$\mathrm{PRF} = \frac{f_{\mathrm{MClk}}}{\mathrm{PRFDivide}}$$

0x61 LFSRCtrl

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7]	-	NA	
[6:1]	LFSRBitEnable	0x3F	Bitwise enabling of each of the 6 bits from the LFSR register. Sets the staggered PRF sequence spread. 0x3F: Full spread length. 0x1F: Spread length / 2. 0x0F: Spread length / 4. 0x07: Spread length / 8. 0x03: Spread length / 16. 0x01: Spread length / 32. 0x00: Staggered PRF disabled.
[0]	StaggeredPRFEnable	0x01	0: Disables the Staggered PRF feature. 1: Enables the Staggered PRF feature. A pseudo random time offset, controlled by an on-chip LFSR register, is inserted between each pulse transmission in order to create a staggered Pulse Repetition Period.

0x62 StaggeredPRFDelay

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:0]	StaggeredPRFDelay	0x00	Delays the SampleClk signal by 0-255 MClk periods, relative to the SendClk signal. Can be used together with the SampleDelay/Send-PulseDelay for Frame Offset. StaggeredPRFReset (0x63) must be called each time the StaggeredPRFDelay is updated. Requires the StaggeredPRFEnable bit to be set.

0x63 StaggeredPRFReset

Executable Action.

Resets the StaggeredPRFDelay circuit (and thus the pseudo-random sequence) to the known state given by LFSRTaps (0x64).

0x64 LFSRTaps

Read-writable 24-bit register.

Bit	Segment name	Default value	Description
[23:20]	-	NA	

Bit	Segment name	Default value	Description
[19:0]	LFSRTaps	0x90000	LFSR feedback taps. Setting bit n results in tapping bit position n of the LFSR register, effectively configuring the polynomial of the pseudo random pulse dithering function Staggered PRF.

0x6A TimingCtrl

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:6]	-	NA	
[5]	MClkShaper	0x00	The MClkShaper shapes the pulse width of MClk to approximately 3 ns. 0: MClkShaper is disabled. 1: MClkShaper is enabled.
[4]	Reserved	0x00	Always write as 0.
[3]	Reserved	0x00	Always write as 0.
[2]	SDFTEnable	0x00	Enables the FineTune delay in the SampleDelay clock path.
[1]	SPDFTEnable	0x00	Enables the FineTune delay line in the SendPulseDelay clock path.
[0]	FTCtrl	0x00	Controls how the FineTune elements are addressed. 0: FT elements are enabled sequentially according to the 6 lower bits of the SDFT/SPDFT (0x6D/0x70) registers. The remaining 58 bits have no function. 1: Each FT element is individually enabled/disabled, according to the SDFT/SPDFT (0x6D/0x70) registers.

0x6B SDCT

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:6]	-	NA	
[5:0]	SDCT	0x00	Controls the number of CoarseTune delay elements in the SampleClk signal path.

0x6C SDMT

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:6]	-	NA	
[5:0]	SDMT	0x00	Controls the number of MediumTune delay elements in the SampleClk signal path.

0x6D SDFT

Read-writable 64-bit register.

Bit	Segment name	Default value	Description
[63]	-	NA	
[62:0]	SDFT	0x00	Controls the FineTune delay elements in the SampleClk signal path. See the description of the FTCtrl bit in the TimingCtrl (0x6A) register. Note that if SDFTEnable is disabled, this register has no effect.

0x6E SPDCT

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:4]	-	NA	
[3:0]	SPDCT	0x00	Controls the number of CoarseTune delay elements in the SendClk signal path.

0x6F SPDMT

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:6]	-	NA	
[5:0]	SPDMT	0x00	Controls the number of MediumTune delay elements in the SendClk signal path.

0x70 SPDFT

Read-writable 64-bit register.

Bit	Segment name	Default value	Description
[63]	-	NA	
[62:0]	SPDFT	0x00	Controls the FineTune delay elements in the SendClk signal path. See the description of the FTCtrl bit in the TimingCtrl (0x6A) register. Note that if SPDFTEnable is disabled, this register has no effect.

0x71 CoarseTuneCtrl

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:6]	-	NA	
[5:2]	SDCTEnable	0x08	Controls the number of active CoarseTune blocks in the SampleClk signal path. Each block contains 8 CT elements and there are a total of 8 CT blocks available. The valid range is from 0 to 8. Setting the value higher than 8 has the same effect as setting it to 8. Take special care not to

Bit	Segment name	Default value	Description
			disable CT elements that are in use by always making sure the following is true: SDCTEnable*8 >= SDCT
[1:0]	SPDCTEnable	0x02	Controls the number of active CoarseTune blocks in the SendClk signal path. Each block contains 8 CT elements and there are a total of 2 CT blocks. The valid range is from 0 to 2. Setting the value higher than 2 has the same effect as setting it to 2.

0x76 ClkOutputCtrl

Read-writable 8-bit register.

Bit	Segment name	Default value	Description
[7:2]	-	NA	
[1]	ClkOutEnable	0x00	0: CLK_OUT is disabled (forced low). 1: CLK_OUT is enabled, and outputs a selectable internal clock signal.
[0]	ClkOutSource	0x00	0: SampleClk is output on the CLK_OUT pin. 1: SendClk is output on the CLK_OUT pin.

0x77 PGMeasResult

Read-only 40-bit register.

Bit	Segment name	Default value	Description
[39:0]	PGMeasResult	NA	Contains the result of the previous PGD measurement.

0x78 SDLMeasResult

Read-only 40-bit register.

Bit	Segment name	Default value	Description
[39:0]	SDLMeasResult	NA	Contains the result of the previous SDL measurement.

0x79 SDMeasResult

Read-only 40-bit register.

Bit	Segment name	Default value	Description
[39:0]	SDMeasResult	NA	Contains the result of the previous SD measurement.

0x7A SPDMeasResult

Read-only 40-bit register.

Bit	Segment name	Default value	Description
[39:0]	SPDMeasResult	NA	Contains the result of the previous SPD measurement.

0x7B SWStopMeas

Executable Action.

Stops a measurement in progress. If a sweep is in progress, the measurement is stopped after the sweep is complete.

0x7C SWCtrl

Read-writable 16-bit register.

Bit	Segment name	Default value	Description
[15:11]	-	NA	
[10]	Reserved	0x00	Always write as 0.
[9:3]	SWWaitCycles	0x1E	MClk is disabled before an SW measurement can begin. $\frac{SWWaitCycles}{f_{\mathrm{MeasClk}}} > \max(\tau_{\mathrm{SD}}) + \tau_{\mathrm{SDL}}$
[2]	SWMeasEnable	0x01	Enable stop watch measurements.
[1]	SWSweepMode	0x00	Controls the end condition of the StopWatch measurement. 0: Measurement ends after SWMeasTargetCycles (0x7E) cycles. 1: Measurement ends when the Sweeping signal goes low.
[0]	SDLMeasEnable	0x00	Enables StopWatch measurements of Sampler Delay Line.

0x7D SWMeasCyclesCounter

Read-only 40-bit register.

Bit	Segment name	Default value	Description
[39:0]	SWMeasCyclesCounter	0x00	Contains the actual number of cycles used for the previous StopWatch measurement.

0x7E SWMeasTargetCycles

Read-writable 40-bit register.

Bit	Segment name	Default value	Description
[39:0]	SWMeasTargetCycles	0xFFFF	Target number of StopWatch measurement cycles.

0x7F SWMeasBusy

Read-only 8-bit register.

Bit	Segment name	Default value	Description
[7:1]	-	NA	
[0]	SWMeasBusy	0x00	High while a StopWatch measurement is in progress.

13. Package Dimensions and Recommended Layout (QFN32)

Novelda uses green packages exclusively, ensuring RoHs compliance, free of lead (Pb), halides (Cl, Br) and anitomy (Sb)

13.1. Recommended Layout

A recommended layout footprint for the QFN32 package with corresponding physical sizes is shown in Figure 13.1.

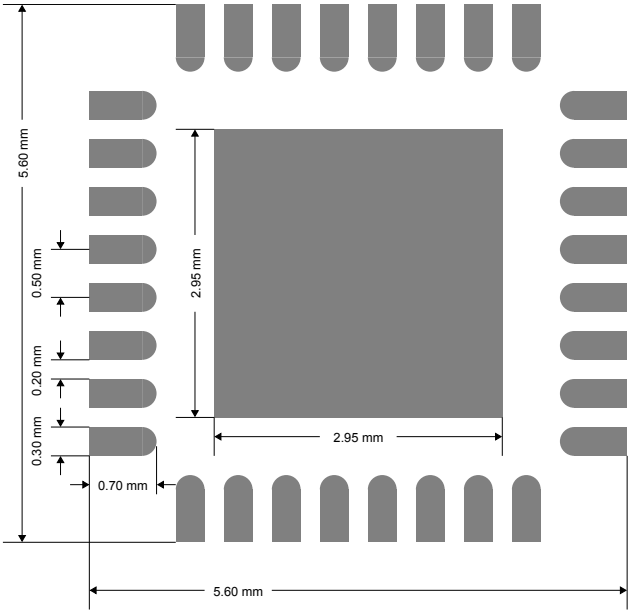


Figure 13.1. QFN32 package footprint.

13.2. Package Dimensions

Physical package dimensions for the QFN32 are shown in Figure 13.2 and Table 13.1.

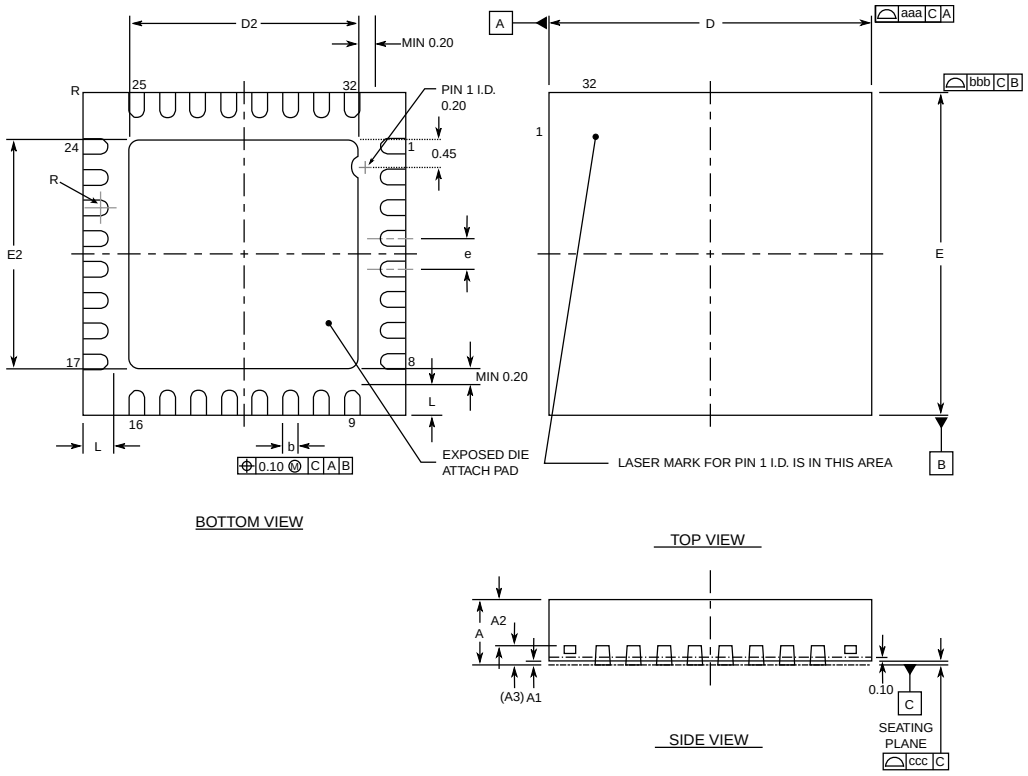


Figure 13.2. QFN32 technical drawing with package dimensions.

	Symbol	Min. [mm]	Nom. [mm]	Max. [mm]
Total thickness	A	0.8	0.85	0.9
Stand off	A1	0.007	0.013	0.05
Mold thickness	A2		0.65	0.70
L/F thickness	A3		0.20 REF.	
Lead width	b	0.18	0.25	0.30
Body size	D		5 BSC	
EP size	D2	3.60	3.70	3.80
Lead pitch	e		0.5 BSC	
Body size	E		5 BSC	
EP size	E2	3.6	3.7	3.8
Lead length	L	0.35	0.40	0.45
Lead radius	R	0.09		
TOLERANCES OF FORM AND POSITION				
Package edge tolerance	aaa		0.1	
Mold flatness	bbb		0.1	
Coplanarity	ccc		0.05	

Table 13.1. QFN32 package dimensions.

14. Disclaimer

The information provided in this document represents Novelda's knowledge and beliefs as of the time of writing. Novelda AS reserves 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. Information is supplied upon the condition that the persons receiving same will make their own determination as to its suitability for their purposes prior to use. In no event will Novelda be responsible for damages of any nature whatsoever resulting from the use of or reliance upon information.

All products are sold subject to Novelda's terms and conditions of sale supplied at the time of order acknowledgement. No representations or warranties, either express or implied, of merchantability, fitness for a particular purpose, that the products to which the information refers may be used without infringing the intellectual property rights of others, or of any other nature are made hereunder with respect to the information or the product to which the information refers. In no case shall the information be considered a part of our terms and conditions of sale.

Document History

Rev.	Release date	Change description
F	01-Oct-2015	Added content to Table 13.1, QFN32 Lead radius. Updated figure 13.2, QFN32 package drawing.
E	10-Sep-2015	Added content to Table 13.1, QFN32 package dimensions.
D	02-Sep-2014	Fixed typo on front page
C	15-Aug-2014	Fixed time scale, figs. 8.13 - 8.23.
B	03-Jun-2014	Updated content to X2 name change. Updated pinout drawing and table. Updated SDCTEnable description. Updated green packaging description. Updated max reflow temperature in abs max ratings. Fixed mistake in stopwatch measurement time section.
A	30-Oct-2013	Preliminary datasheet, initial release.