SiT95314 Quad PLL Frequency Translator / Jitter Cleaner / Network and Port Synchronizer

PRELIMINARY

Description

SiT95314 offers programmable Quad Fractional Frequency translation based jitter cleaning clock synthesizer parts with flexible input to output frequency translation options. Ultra High performance PLL supports up to 4 Diff/8 Single Ended input clocks that are common for all the 4 fractional translations and provides 4 clock outputs. The clock outputs can be derived from any of the 4 PLLs in a fully flexible manner.

It is fully programmable with the I2C/SPI interface or an on chip one time programmable non-volatile memory for factory pre-programmed devices.

Applications

- Carrier Ethernet
- OTN Equipment
- Microwave Backhaul
- Gigabit Ethernet
- Wireless Infrastructure
- Network Line Cards
- Small Cells
- Data Center/Storage
- SONET/SDH
- Test / Instrumentation
- Broadcast Video

Features

- Ultra-high Performance PLLs
- Fully Integrated design with no external components
- Sub 70 fs Typical RMS integrated jitter (12k-20M)
- 122.88M Output with excellent close in noise performance
- Fully Flexible Output and Input Mux: High level of flexibility in output allocation for PLLs
- JESD204B/C Support for data converter clocks
- External EEPROM Support
- Frequency Control DCO: DCO Control on all outputs (down to 0.001 ppt)
- Phase Control DCO: Fine phase adjustment knob for phase of all outputs from a PLL (adjustment accuracy < 1ps) in both closed loop and open loop modes
- Best in class hitless switching performance: PBO with sub 25 ps hit, Phase Propagation & Frequency Ramp with programmable freq/phase slopes
- Fully integrated Jitter and wander attenuation options down to 0.09 mHz
- Repeatable input to output delays with output relative delay adjust
- Internal ZDB Mode with < 0.5 ns Input to Output delay variation. Independently available for each PLL
- Outputs can be phase aligned on independent sync pulse
- 44 QFN 7 mm x 7 mm Package

Product Family	Inputs / Outputs	Input Freq	Output Frequency	RMS Jitter	Packages
SiT95314	4 Diff / 8 SE Inputs	8KHz – 2.1 GHz	0.5 Hz – 2.94912 GHz	~ 70 fs typ (QFN)	44 QFN

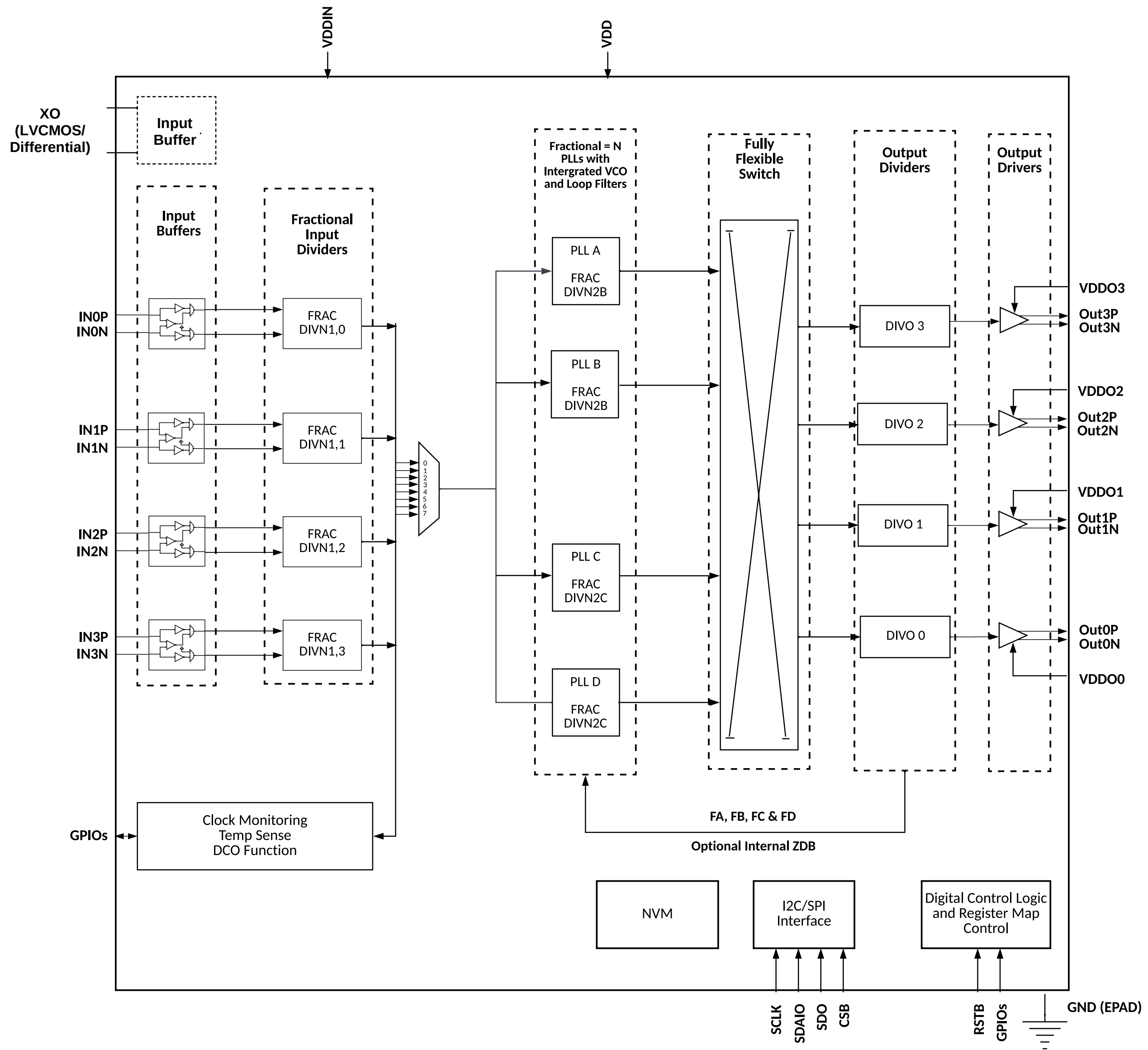
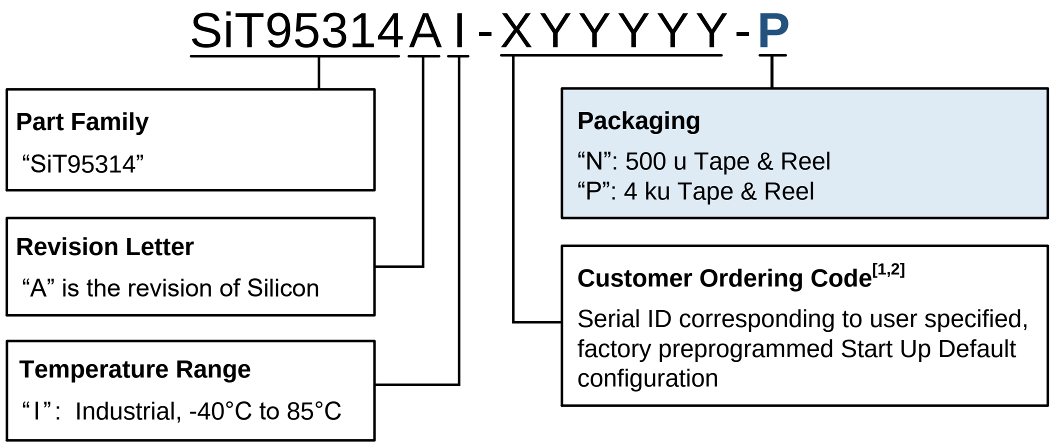


Figure 1. Functional Overview SiT95314

Contact SiTime for MEMS Reference Clocks for best jitter.

Table of Contents

Ordering Information


SiT95314 A I - X Y Y Y Y Y - P
Part Family
"SiT95314"
Revision Letter
"A" is the revision of Silicon
Temperature Range
"I": Industrial, -40°C to 85°C
Packaging
"N": 500 u Tape & Reel
"P": 4 ku Tape & Reel
Customer Ordering Code[1,2]
Serial ID corresponding to user specified, factory preprogrammed Start Up Default configuration

Notes:

1. X = "A" and "B" customer device, "C" to "Z" reserved.
   a. A:Denotes blank devices;
   b. B: Denotes Pre-configured devices, contact SiTime for the specifics
2. Y = 0..9, A…Z for custom serial ID.

Electrical Characteristics

Table 1. Absolute Maximum Ratings

Description	Conditions	Symbol	Min	Typ	Max	Units
Supply Voltages	Input Supply PLL Supply	V_{DDIN} V_{DD}	-0.5		+3.63	V
Output bank supply voltage	Output Driver Supply	V_{DDO}	-0.5		+3.63	V
Input voltage, All Inputs	Relative to GND Clock Inputs GPIO Inputs	V_{IN}	-0.5		+3.63	V
XO Inputs	Relative GND	V_{XO}	-0.5		+1.8	V
I2C Bus input voltage	SCLK SDIO	V_{INI2C}	-0.5		+3.63	V
SPI Bus input voltage	SDO_A1 CSB_A0	V_{INSPI}	-0.5		+3.63	V
Storage temperature	Non-functional, Non-Condensing	T_S	-55		+150	°C
Programming Temperature		T_{PROG}	+25		+85	°C
Maximum Junction Temperature in Operation		T_{JCT}			+125	°C
Programming Voltage (for Programming the OTP (Fuse Memory).		V_{PROG}	2.375	2.5	2.625	V
ESD (human body model)	JESD22A-114	ESD_{HBM}			2000	V
ESD (charge device model)		ESD_{CDM}			500	V
Latch Up	JEDEC JESD78D	LU			100	mA
Moisture Sensitivity Level		MSL	3			

Notes:

1. Exceeding maximum ratings may shorten the useful life of the device.
2. Stresses beyond those listed under Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or at any other conditions beyond those indicated under the DC Electrical Characteristics is not implied.
3. Exposure to Absolute-Maximum-Rated conditions for extended periods may affect device reliability or cause permanent device damage.
4. Any pin category not covered here has a default minimum rating of -0.5 V and a default maximum rating of 3.63 V.

Table 2. Operating Temperature and Thermal Characteristics

Description	Conditions	Symbol	Min	Typ	Max	Units
Ambient temperature		TA	–40	–	+85	°C
Junction temperature		TJ			+125	°C
SiT95314 : 44 QFN Package						
Thermal Resistance Junction to Ambient	Still Air	$θ_{JA}$	TBD			°C/W
	Air Flow :1m/s		TBD			
	Air Flow: 2m/s		TBD			
Thermal Resistance Junction to Case		$θ_{JC}$	TBD			°C/W
Thermal Resistance Junction to Board		$θ_{JB}$	TBD			°C/W
SiT95314						
Analog Input Pathways and XTAL Pathways	3.3 V range: ± 10%	VDDIN	2.97	3.3	3.63	V
PLL Supply	1.8 V range: ±5%	V_{DD}	1.71	1.80	1.89	V
Output Driver Supply	1.8 V range: ±5%	V_{DDO}	1.71	1.80	1.89	V
	2.5 V range: ±5%		2.375	2.50	2.625	V
	3.3 V range: ±10%		2.97	3.3	3.63	V

Description	Conditions	Symbol	Min	Typ	Max	Units
Power Dissipation (VDD=1.8V, VDDIN=3.3V, VDDO = 3.3 V)						
Total Power Dissipation (3.3 V LVDS Outputs @ 156.25M)	4 PLLs, 4 Outputs 4 DE Inputs	Pd		1.2		W
Supply Current						
VDDIN	All Four DE Inputs assumed to be enabled	I_{DDIN}		57		mA
VDD	Four PLLs and All 4 Outputs enabled	I_{DD}		423		mA
VDDO[4,7]	LVPECL, output pair terminated 50 Ω to V_{TT} (VDDO – 2 V).	I_{DDO} [1,2,3,5]		46		mA
	CML, output pair terminated 50 Ω to VDDO	I_{DDO} [1,5]		21		mA
	HCSL, output pair with HCSL termination	I_{DDO} [1,5]		35		mA
	LVDS output pair terminated with an AC or DC Coupled diff 100 Ω	I_{DDO} [1,5]		21		mA
	LVCMOS, 250 MHz, 2.5 V output, 5-pF load	I_{DDO} [5,6]		22		mA

Notes:

1. LVPECL standard is supported for V_{DDO} = {2.5 V, 3.3 V}. HCSL, CML and LVDS standards are supported for V_{DDO} = {1.8 V, 2.5 V, 3.3 V}.
2. LVPECL mode provides 6 mA of common mode current on each output.
3. A 50 Ω Termination resistor with a DC bias of V_{DDO} – 2 V for LVPECL standards is supported for V_{DDO} = {2.5 V, 3.3 V}.
4. IDDOx Output driver supply current specified for one output driver in the table. This includes current in each of the output module that includes output dividers, drivers and clock distributions.
5. Refer to the Output Termination Information in Output_Slave_Description the data sheet for the description of the various terminations that are supported.
6. Both P and N terminals are active in LVCMOS mode.
7. Current consumption accounts for load current. LVCMOS current include load current also.

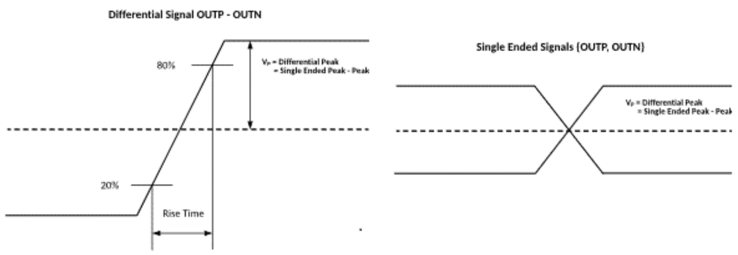
Table 3. Input Clock Characteristics

Parameter	Conditions	Symbol	Min	Typ	Max	Unit
Input Frequency Range	Differential	f_{IN}	8K	—	2100M	Hz
	All Single-ended signals (including LVCMOS)		8K	—	250M	Hz
PLL Input Frequency Range [1]	PLL Input Frequency	f_{IN_DPLL}	8K	-	12.5M	Hz
Input Buffer with Differential DC or AC Coupled (See Input Slave Description for Termination information)						
Input Common Mode Voltage Range (for DC Coupled Differential Inputs)	Differential DC coupled inputs; Defined as cross point	V_{CMR}	0.25	-	VDDIN – 0.85	V
Voltage Swing (Differential Amplitude Peak or Single Ended Peak to Peak for the differential signal)	f_{IN} < 400 MHz	VP_{IN} [3]	100	—	-	mV
	400 MHz < f_{IN} < 750 MHz		225	—	-	mV
	750 MHz < f_{IN} < 2100 MHz		350	—	-	mV
Slew rate		SR	400	-	-	V/us
Input Capacitance		C_{in}	-	1	-	pf
Input Resistance	Differential DC (on each input)	R_{in}	-	26	-	kΩ
	Differential AC		-	10	-	
Single Ended AC Coupled input (IN0P/N, IN1P/N, IN2P/N, IN3P/N)						
Voltage Swing (Vpp)	AC-Coupled fIN < 250 MHz	f_{IN,SE_AC}	500	_	3600	mV
Slew Rate [4]		SR	400	—	—	V/µs
Duty Cycle		DC	40	—	60	%
Input Capacitance		C_{IN}	—	0.3	—	pF
Input Resistance	AC Coupled SE	R_{IN}	—	24	—	kΩ

Parameter	Conditions	Symbol	Min	Typ	Max	Unit
Single Ended DC-Coupled Input (IN0P/N, IN1P/N, IN2P/N, IN3P/N)						
Input Voltage		V_{IL}	–0.2	—	0.4	V
		V_{IH}	0.8	—	VDDIN	V
Slew Rate [4]		SR	400	—	—	V/µs
Duty Cycle		DC	40	—	60	%
Input Resistance		R_{IN}	—	26	—	kΩ
Reference Clock (Applied to X1), Can be external XO						
Reference Clock Frequency	Range for best jitter	F_{IN_REF}	48	-	192	MHz
	Overall supported range		25	-	192	MHz
Input Voltage Swing [5]	Single Ended peak to peak	V_{IN_SE}	365	-	2000	mVpp_se
Slew rate		SR	400	-	-	V/us
Duty Cycle		DC	45	-	55	%

Notes:

1. Differential and SE P Side has fractional divider with integer divide range of 2 to 65535
2. For proper device operation, the input frequency is internally divided down to 12.5 MHz or less (PLL Phase Detector maximum frequency = 12.5 MHz). This is achieved using internal dividers in the chip.
3. VPIN is the single-ended peak-peak of the input signal which is equal to the differential peak. This is the swing requirement for both AC and DC coupled differential inputs where the swing is considered at the pin inputs.


Differential Signal OUTP - OUTN
80%
20%
Rise Time
V_P = Differential Peak
= Single Ended Peak - Peak
Single Ended Signals (OUTP, OUTN)
V_P = Differential Peak
= Single Ended Peak - Peak

4. Minimum slew rate specification is for best noise performance.
5. Max Voltage at X1 pin should be < 1.8 V.

Table 4. Serial Data and Clock Input, GPIOs

Parameter	Condition	Symbol	Min	Typ	Max	Unit
Input Voltage		V_{IL}	—	—	0.3 x V_{DDIO}[1]	V
		V_{IH}	0.7 x V_{DDIO}[1]	—	VDDIO	V
Input Capacitance		C_{IN}	—	1	—	pF
Input Resistance		R_{IN}	—	25	—	kΩ
Minimum Pulse Width	FINC, FDEC [2][3]	PW	100	—	—	ns
Minimum RESET duration		T_{RES}	100			µs
Update Rate	FINC, FDEC [2][3]	F_{UR}	1	—	-	µs
Minimum RESET Pulse Width			100	-	-	us
Minimum time between RSTb release and SPI/I2C ready			1	-	-	ms

Notes:

1. VDDIO is software selectable voltage between VDDIN and VDD
2. FINC/FDEC are the increment and decrement for the DCO opertaion from the pins
3. Minimum Pulse width/update rate are specified for free run PLL DCO

Table 5. Output Serial and GPIO Pin

Parameter	Test Condition	Symbol	Min	Typ	Max	Unit
All VDDIO based GPIOs						
Output Voltage	I_{OH} = -2 mA	V_{OH}	V_{DDIO} x0.75	—	—	V
	I_{OL} = 2 mA	V_{OL}	—	—	V_{DDIO} x 0.25	V

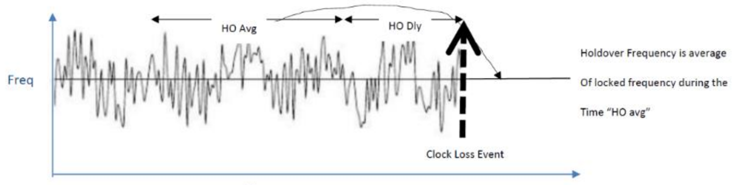
Table 6. Output Clock Characteristics

Description	Conditions	Symbol	Min	Typ	Max	Units
Differential output frequency continuous support [1],[2]	Differential output standards	$F_{OUT,DIFF}$ [1,3]	0.5		800M	Hz
Differential output frequency Selective Higher Frequency Support (For >800MHz outputs)[3]	Differential Output Standards	$F_{OUT,DIFF}$ [1,3]	61.44M* {15, 20, 24, 30, 32, 40, 48} = 921.6M, 1228.8M, 1574.56M, 1843.2M, 1966.08M, 2457.6M, 2949.12M			Hz
Single ended output frequency	LVCMOS outputs	$F_{OUT,SE}$	0.5		250 M	Hz
PLL loop bandwidth	Programmable	F_{BW}	0.00009		4000	Hz
Jitter peaking[4]	Meets SONET Jitter Peaking requirements in closed loop (see footnote)	J_{PEAK}			0.1	dB
Time delay before the Historical average for output frequency is considered.	Programmable in register map	H_{DELAY} [5]	0.002		35	s
Length of time for which the Average of the frequency is considered	Programmable in register map	H_{AVG} [5]	0.004		70	s
Power Supply to I2C or SPI interface ready	No I2C or SPI transaction valid till 10ms after all power supplies are ramped to 90% of final value.	T_{START}			10	ms
Hold Time for GPIO Latching[6]	Hold time for GPIO latched inputs available on the GPIOs after the RSTB pin is driven from low to high	T_{HOLD} [6]	1			ms
DCO Mode Frequency Step Resolution[7]	Frequency Increment or Decrement resolution. This is controlled through the register map.	$F_{RES,DCO}$ [7]	0.001			ppt
Output Phase Shift [8]	Resolution for output delay between clocks from same PLL. Resolution Programmable per output clock with this resolution for a total delay range of ±T/2 where T is the time period of the output clock	T_{RESO} [8]		35		ps
Output Skew[9]	Skew between outputs from the same PLL set up with same phase shift code	T_{SKEW} [9]	-50		50	ps
Input Phase Shift Resolution[10]	Programmable delay resolution for all outputs that are locked to a particular input. Input phase shift is programmable per input clock with this resolution for a total delay of ±T/2 where T is the time period of the input clock	T_{RESI} [10]	1			ps
PLL Lock Time[11]	Standard Mode	T_{LOCK}		300m		s
Maximum Phase Hit[12]	Default PBO Hitless Switching Mode (no phase propagation)	T_{MAX} [12]	-25		25	ps

Description	Conditions	Symbol	Min	Typ	Max	Units
Input to Output Delay Variation in external ZDB mode (external ZDB with feedback on the PCB)[13]	Any output to any input external feedback is possible. Supported for all PLLs. Multiple PLLs in external ZDB are supported concurrently	T_{ZDELAY}	-100		100	ps
Internal ZDB Mode Input to Output Delay Variation[14]	Supported for all PLLs. Multiple PLLs in internal ZDB are supported concurrently.	$T_{ZDELAY,INT}$	-500		500	ps
Uncertainty in Input to Output Delay	Maximum variation in the static delay from input to output clock between repeated power ups of the chip	ΔT_{DELAY}	-225		225	ps
PLL Pull Range		ω_P		500		ppm

Notes:

1. The continuous frequency support implies that all output frequencies till 800 MHz are available with no gaps.
2. The VCOs support two ranges: A Low Band Range of 4915.2 MHz to 5898.24 MHz and a High Band Range of 6875 MHz to 7812.5 MHz. Along with the fully flexibly output multiplexer for the output clocks, this provides for customer use cases to be easily supported with several concurrent frequencies for application scenarios possible from a single PLL.
3. Specific multiples of 61.44 MHz are provided for Wireless applications. Please contact SiTime Semiconductor for more frequency options and details.
4. Jitter peaking limit of < 0.1 dB can be enabled as an option for cases (such as SONET) where there is such a critical requirement. For other cases, the jitter peaking can be made slightly higher to enable faster transients and lock characteristics.
5. Hitless Switching enables PLL to switch between input clocks when the current clock is lost,
    a. Clock Loss can be defined as a specified number of consecutive missing pulses.
    b. Priority list for the input clocks can be set in the register map independently for each PLL.
    c. Output is truly hitless (no phase transient and 0 ppb relative error in frequency) for exactly same frequency input clocks that are switched in Phase Build Out Mode.
    d. Hitless switching support is both revertive and non-revertive
    e. Revertive / Non-revertive Support: Assume Clock Input 0 is lost and switch is made to Clock Input 1. Then, PLL reverts to Clock Input 0 when it becomes valid again in Revertive mode. It does not switch back to Clock Input 0 even when it becomes valid again in the non-Revertive mode.
    f. Entering hold over mode is supported with the frequency frozen at a historical average determined from the H_{DELAY} and H_{AVG} settings.



6. The SiT95314 chips provide a GPIO latching function that allows for certain GPIOs to function as latched GPIOs that are latched along with the release of chip reset (using the RSTB pin) and can the same pin is released for other functions in steady state.
    a. This requires the RSTB pin to be held low and released from low to high ONLY when all supplies to the chip have crossed 90% of their final value.
    b. This further requires that the GPIOs whose inputs are used for GPIO latching should "Hold" the expected value for latching for at least 1ms of time after the RSTB pin has reached 90% of the VDDIO supply.
7. The 0.001 ppt specification is for the smallest frequency step resolution available. Larger frequency step resolutions up to 100 ppm can be used also. The frequency resolution for the DCO mode frequency step is independently programmable for each DCO step.
    a. DCO steps are applicable on both XO referenced PLLs (free run) and input referenced PLL (sync mode)
    b. DCO step size is programmed in the programmable interface (PIF) using the serial I2C or SPI interface
    c. DCO is enacted in response to a trigger. This trigger can be provided either with a register based write through the serial interface or a pin-based trigger where a GPIO is programmed for the purpose of the increment and decrement DCO trigger.
8. The delay referred to here is delay between outputs from the same PLL. Such a delay can delay a clock by as much as 180 degrees which is half of a time period of the output clock with the resolution of 35 ps.
    a. All output clocks from one specific PLL are phase aligned by default. Relative delay adjustment is then possible on each clock individually as defined by the T_{RES} parameter for a total delay range of ±T/2 where T is the time period of the output clock.
9. This is the skew between outputs from the same PLL such that they are set up with the same relative output phase shift code.
10. The delay referred to here is from input to output hence it appears in the system as an input phase shift. All outputs from a PLL that is locked to a particular input can move by as much as 180 degrees of the input clock for the PLL (which is half the time period for the PLL input clock) and with a resolution of +/-10ps.
11. For low PLL Loop Bandwidths, wake up time can be very large unless the speed up features are used. The speed up feature provides the user options to use a completely independent time constants for the wake-up transitioning to the regular bandwidth after frequency and phase are locked
    a. Fast Lock Bandwidth needs to be less than 100 times smaller than the input clock frequency (divided input at PLL phase detector) for stable and bounded (in time) lock trajectory of the PLL
12. This test is for 2 inputs at 8M that are switched to get a 156.25M output.
13. Both input and feedback at 8MHz with the delays exactly matched and same slew for both for the chip
14. Internal ZDB mode is supported concurrently on any number of PLLs. The delay specification here is for delay between the input port and output port of the chip.

Table 7. Fault Monitoring Indicators

Description	Conditions	Symbol	Min	Typ	Max	Units
Clock Loss Indicator Thresholds	Clock Loss Indicators can be set on any of the 8 inputs. Loss of 2 - 16 consecutive pulses can be used to indicate a clock loss. Programmable in the register map.	CL_X [1,4, 5]	2		16	Pulses
Fine Frequency Drift Indicator Thresholds	Fine Drift Indicators programmable from ±2 ppm upto ±510 ppm in steps of ±2 ppm.	FD_X [2,3,4]	±2		±510	ppm
Coarse Frequency Drift Indicator Thresholds	Coarse Drift Indicators programmable from ±100 ppm upto ±1600 ppm in steps of ± 100 ppm.		±100		±1600	ppm
Phase Lock Loss Indicator	Provides phase locking indication for lock of the input and feedback clock phases with respect to each other for each PLL. This is particularly useful for 1 PPS lock	LLPH	1n		100u	s
Lock Loss Indicator Threshold	Lock Loss Indicator threshold is programmable in the range specified from the following choices for setting and clearing LL: {±0.05, ±0.1} ppb, {±0.5, ±1} ppb, {±0.2, ±0.4} ppm, {±2, ±4} ppm, {±20, ±40} ppm, {±200, ±400} ppm, {±2000, ±4000}	LL	±0.05ppb		±4000ppm	

Notes:

1. Clock Loss Indicators are used for:
   a. Hitless Switching Triggers
   b. Update in Status Registers in the register map
2. Frequency Drift Indicators can use any one of the 8 input clocks, internal clockInSync clock or the Crystal / Reference input clock as the golden reference with respect to which FDx for all other clocks can be recorded in the Status Registers. FDx thresholds for each clock input for each clock can be set independently.
3. Coarse and Fine Frequency Drift indicators can be concurrently enabled. This enables the user to detect fast drifting frequencies since detecting fine drifts will take longer measurements.
4. Clock loss and Lock loss indicators are available as alerts on GPIO pins.
5. Clock Loss can be combined with either of the frequency drift monitors (coarse and fine) to trigger the hitless switching event in the PLLs. The trigger for a hitless switching event in the PLL can therefore be either the Clock Loss event or either of Clock Loss or Frequency Drift.

Table 8. Crystal Requirements (SiT95314)

Description	Conditions	Symbol	Min	Typ	Max	Units
High Fundamental Frequency Crystal Reference (HFF)						
Crystal Frequency	Can be supported with a fundamental crystal of 100-160 MHz range.	$XTAL_{IN}$	100		160	MHz
C0 cap for crystal	Small range around CL only	$XTAL_{C0}$			2	pF
CL cap for crystal		$XTAL_{CL}$		5		pF
ESR for crystal	ESR defined at frequency of oscillation	$XTAL_{ESR}$			40	Ω
Rm1 for crystal		$XTAL_{Rm1}$			20	Ω
Power delivered to crystal	Drive Level to the crystal	$XTAL_{PWR}$		100		µW
Low Frequency Fundamental Crystal (LFF)						
Crystal Frequency	Can be supported with a fundamental crystal > 25 MHz range. For Best Performance use an LFF crystal > 48 MHz	$XTAL_{IN}$	25		96	MHz
C0 cap for crystal	Small range around CL only	$XTAL_{C0}$			2	pF
CL cap for crystal [2]		$XTAL_{CL}$		8		pF
ESR for crystal	ESR defined at frequency of oscillation	$XTAL_{ESR}$ [1]			40	Ω
Rm1 for crystal		$XTAL_{Rm1}$			40	Ω
Power delivered to crystal	Drive Level to the crystal	$XTAL_{PWR}$		100		µW

Notes:

1. ESR relates to the motional resistance Rm with the relationship $ESR = Rm\ (1 + C0/CL)^2$
2. The table specifies the Ci, Ce and Cs for the 48 MHz XTAL with different CL. Ci is the internal differential capacitance offered by the
3. chip whereas Ce (Ce=Ce1=Ce2) and Cs(Cs=Cs1=Cs2) are the Single Ended external and the stray cap on the PCB

XTAL	Ci	Ce	Cs	Unit
48MHz (CL = 8pF)	7.5	0	1	pF

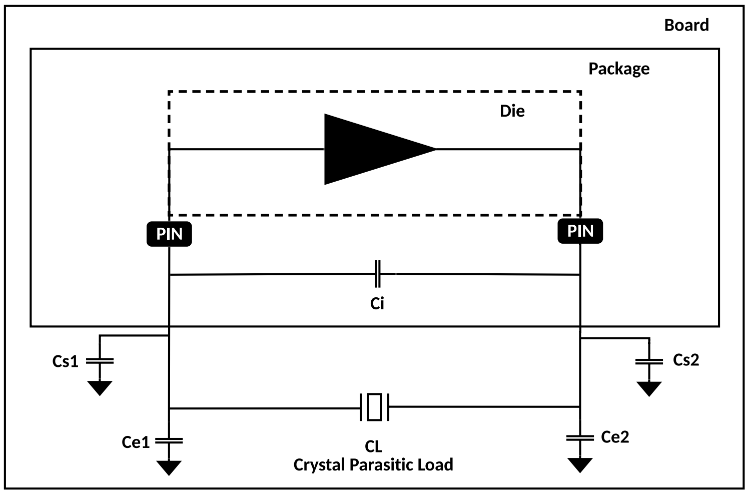


Table 9. Output RMS Jitter in Frequency Translation Modes

Description	Conditions	Symbol	Min	Typ	Max	Units
RMS Jitter for 12 kHz – 20 MHz Integration Bandwidth **F_{IN} = 38.88 MHz,** **PLL BW = 10 Hz, Single PLL Profile**	F_{OUT} = 156.25 MHz Ext. 76.8 MHz MEMS [1]	RMS_{JIT} [1]		~70		fs rms

Notes:

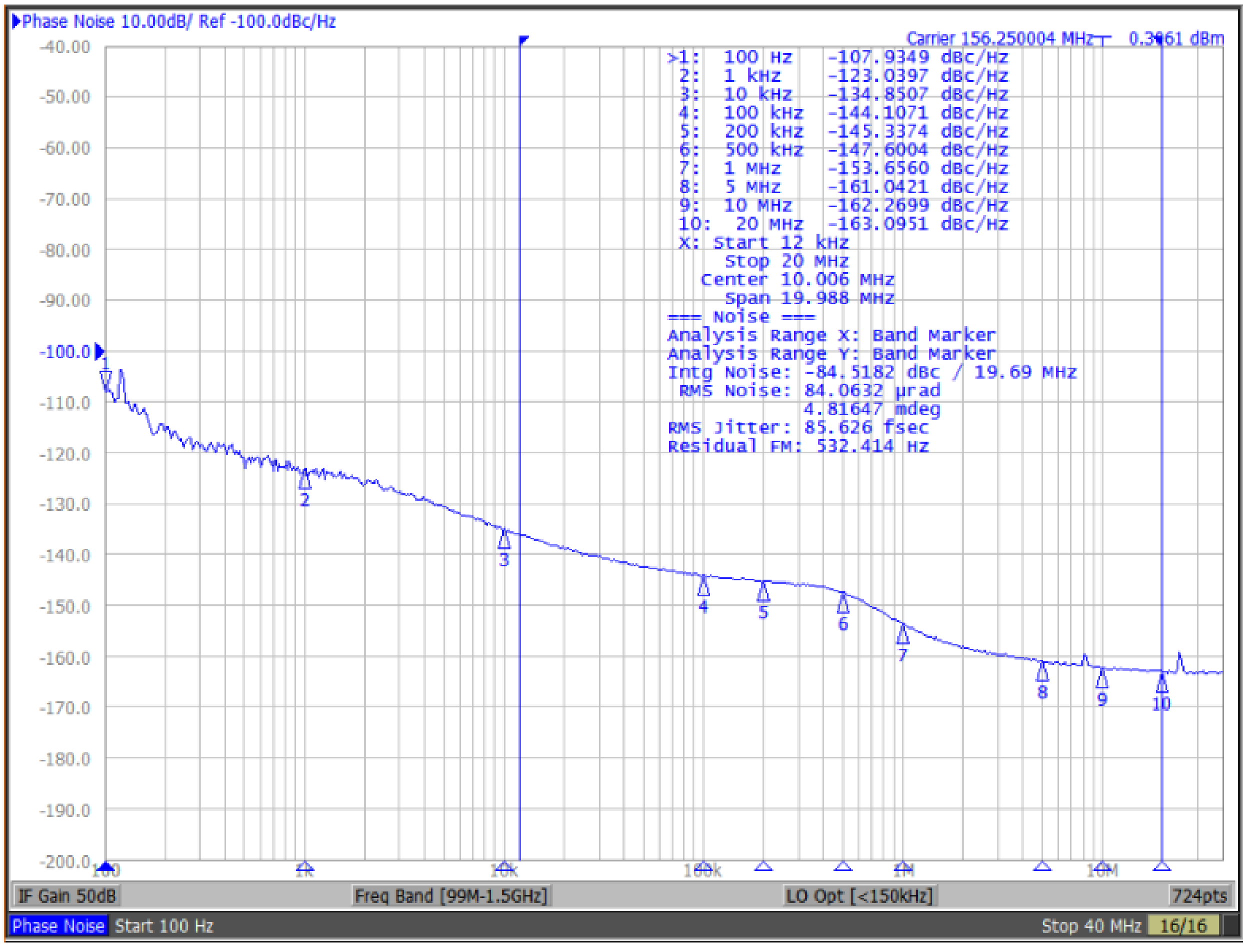
1. Jitter is 85fs if a 48MHz Crystal is used – assuming it meets the specifications mentioned in this data sheet

Table 10. Close-In Offset Phase Noise

Description	Offset Frequency	Symbol	Min	Typ	Max	Units
Phase Noise Skirt **F_{IN} = 38.88 MHz** **F_{OUT} = 122.88 MHz,** **PLL BW = 10 Hz** **48 MHz Reference** [2]	100 Hz	PN[1]		-116		dBc/Hz
	1 kHz			-127		
	10 kHz			-136		
	100 kHz			-145		
	1 MHz			-155		
	10 MHz			-163		

Notes:

1. External XO and Clock Inputs are fed from SMA100 source



```
>1:  100 Hz   -107.9349 dBc/Hz
 2:  1 kHz    -123.0397 dBc/Hz
 3:  10 kHz   -134.8507 dBc/Hz
 4:  100 kHz  -144.1071 dBc/Hz
 5:  200 kHz  -145.3374 dBc/Hz
 6:  500 kHz  -147.6004 dBc/Hz
 7:  1 MHz    -153.6560 dBc/Hz
 8:  5 MHz    -161.0421 dBc/Hz
 9:  10 MHz   -162.2699 dBc/Hz
 10:  20 MHz  -163.0951 dBc/Hz
 X: Start 12 kHz
      Stop 20 MHz
    Center 10.006 MHz
      Span 19.988 MHz
=== Noise ===
Analysis Range X: Band Marker
Analysis Range Y: Band Marker
Intg Noise: -84.5182 dBc / 19.69 MHz
 RMS Noise: 84.0632 µrad
            4.81647 mdeg
RMS Jitter: 85.626 fsec
Residual FM: 532.414 Hz
```

SiT95314 : 156.25MHz Output (LVDS) with 48MHz Crystal

Table 11. Power Supply Rejection

Description	Conditions	Symbol	Min	Typ	Max	Units
F_{OUT} = 156.25 MHz, F_{SPUR} = 100 kHz, BW = 100 Hz **PSRR on PLL Supply**	VDD = 1.8 V	$PSRR_{VDD}$		-80		dBc
F_{OUT} = 156.25 MHz, F_{SPUR} = 100 kHz, BW = 100 Hz **PSRR on Input Supply**	VDDIN = 3.3 V	$PSRR_{VDDIN}$		-100		dBc
F_{OUT} = 156.25 MHz, F_{SPUR} = 100 kHz, BW = 100 Hz **PSRR on Output Driver Supply**	VDDO = 3.3 V	$PSRR_{VDDO}$		-90		dBc

Notes:

1. The PSRR is measured with a 50 mVpp sinusoid in series with the supply and checking the spurious level relative to the carrier on the output in terms of phase disturbance impact.
2. Output PSRR measured with LVDS standard which is the recommended standard for AC Coupled terminations

Table 12. Adjacent Output Cross Talk

Description	Conditions	Symbol	Min	Typ	Max	Units
156.25 M and 155.52 M on adjacent outputs	LVDS Output	X_{TALK}		-75		dBc

Notes:

1. Measured across adjacent outputs- All adjacent outputs are covered and the typical value for the worst-case output to output coupling is reported.
2. This cross talk between outputs is mainly package dependent therefore terminated outputs are used for reporting these numbers ensuring that there is signal current in the bond wires.

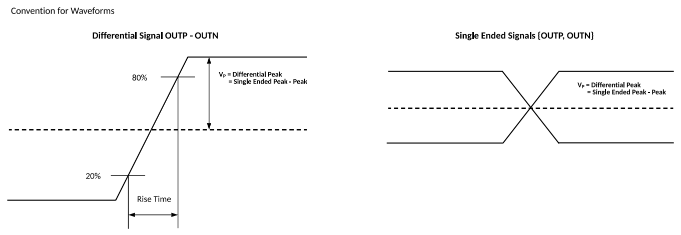
Table 13. Output Clock Specifications

Descriptions	Conditions	Symbol	Min	Typ	Max	Units
DC Electrical Specifications - LVCMOS output (Complementary Out of Phase Outputs or One CMOS Output per Output Driver)						
Output High Voltage	4 mA load, VDD = 3.3 V	V_{OH}	VDDO-0.3		-	V
Output High Voltage	4 mA load, VDD = 1.8V and 2.5 V	V_{OH}	VDDO-0.4		-	V
Output Low Voltage	4 mA load	V_{OL}			0.3	V
DC Electrical Specifications - LVCMOS output (In Phase Outputs)						
Output High Voltage	4 mA load, VDD = 3.3 V	V_{OH}	VDDO-0.35		-	V
Output High Voltage	4 mA load, VDD = 2.5 V	V_{OH}	VDDO-0.45		-	V
Output High Voltage	4 mA load, VDD = 1.8 V	V_{OH}	VDDO-0.5		-	V
DC Electrical Specifications – LVDS Outputs (VDDO = 1.8 V, 2.5 V or 3.3 V range)						
Output Common-Mode Voltage	VDDO = 2.5 V or 3.3 V range	V_{OCM}	1.125	1.2	1.375	V
Output Leakage Current	Output Off, VOUT = 0.75 V to 1.75 V	I_{OZ}	-20		20	µA
AC Electrical Specifications LVCMOS Output Load: 10 pF < 125 MHz, 7.5 pF < 150 MHz, 5 pF > 150 MHz > 200 MHz						
Output Frequency		f_{OUT}	0.5		250M	Hz
Output Duty cycle	Measured at 1/2 VDDO, loaded, f_{OUT} < 125 MHz	t_{DC}	45		55	%
Output Duty cycle	Measured at 1/2 VDDO, loaded, f_{OUT} > 125 MHz	t_{DC}	40		60	%
Rise/Fall time	VDDO = 1.8 V, 20-80%, Highest Drive setting	t_{RFCMOS}			2	ns
Rise/Fall time	VDDO = 2.5 V, 20-80%, Highest Drive setting	t_{RFCMOS}			1.5	ns
Rise/Fall time	VDDO = 3.3 V, 20-80%, Highest Drive setting	t_{RFCMOS}			1.2	ns
AC Electrical Specifications (LVPECL, LVDS, CML)						
Clock Output Frequency		f_{OUT}	0.5		2949.12 M	Hz
LVPECL Output Rise/Fall Time	20% to 80% of AC levels. Measured at 156.25 MHz for PECL outputs.	t_{RF}			350	ps
CML Output Rise/Fall Time	20% to 80% of AC levels. Measured at 156.25 MHz for CML outputs	t_{RF}			350	ps
LVDS Output Rise/Fall Time	20% to 80% of AC levels. Measured at 156.25 MHz for LVDS outputs.	t_{RF}			350	ps
HCSL Output Rise/Fall Time	20% to 80% of AC levels. Measured at 156.25 MHz for HCSL outputs.	t_{RF}			350	ps
Output Duty Cycle	Measured at differential 50% level, 156.25 MHz	t_{ODC}	45	50	55	%
Programmable Output differential peak LVDS DC Coupled Output[1],[2]	Measured at 156.25M Output (Programmable Typical Levels mentioned)	VP		100		mV
				200		
				300		
			300	400	500	
				500		
				600		
				700		
			500	800	1000	

Descriptions	Conditions	Symbol	Min	Typ	Max	Units
Programmable Output differential peak HCSL Coupled Output[1],[2]	Measured at 156.25M Output (Programmable Typical Levels mentioned) DC Coupled True HCSL termination assumed. See foot note	VP		500		mV
				550		
				600		
				650		
				700		
				750		
			400	800	1000	
Programmable Output differential peak CML Coupled Output[1],[2]	Measured at 156.25M Output (Programmable Typical Levels mentioned) DC Coupled True CML termination assumed. See foot note	VP		50		mV
				100		
				150		
				200		
				250		
				300		
				350		
			250	400	600	
Programmable Output differential peak LVPECL Coupled Output[1],[2]	Measured at 156.25M Output (Programmable Typical Levels mentioned) DC Coupled True LVPECL termination assumed. See foot note	VP		500		mV
				540		
				580		
				620		
				660		
				680		
				700		
			450	720	900	

Notes:

1.


Convention for Waveforms
Differential Signal OUTP - OUTN
80%
V_P = Differential Peak = Single Ended Peak - Peak
20%
Rise Time
Single Ended Signals {OUTP, OUTN}
V_P = Differential Peak = Single Ended Peak - Peak

2. Please see Output_Slave_Description related to the output slave to determine the output termination supported as per standard.
   a. LVDS standard is recommended for most AC Coupled termination cases OR DC coupled differential 100 Ohm loading
   b. LVPECL Standard mentioned here supports the true traditional standard with DC coupled 50 Ohm terminations to VDDO- 2V or its corresponding Thevenin equivalent network.
   c. CML Standard mentioned here supports the true traditional standard with DC coupled 50 Ohm terminations to VDDO supply.
   d. HCSL Standard mentioned here supports the true traditional standard with DC coupled 50 Ohm terminations to Ground which are then AC or DC coupled to a differential receiver

Pin Description

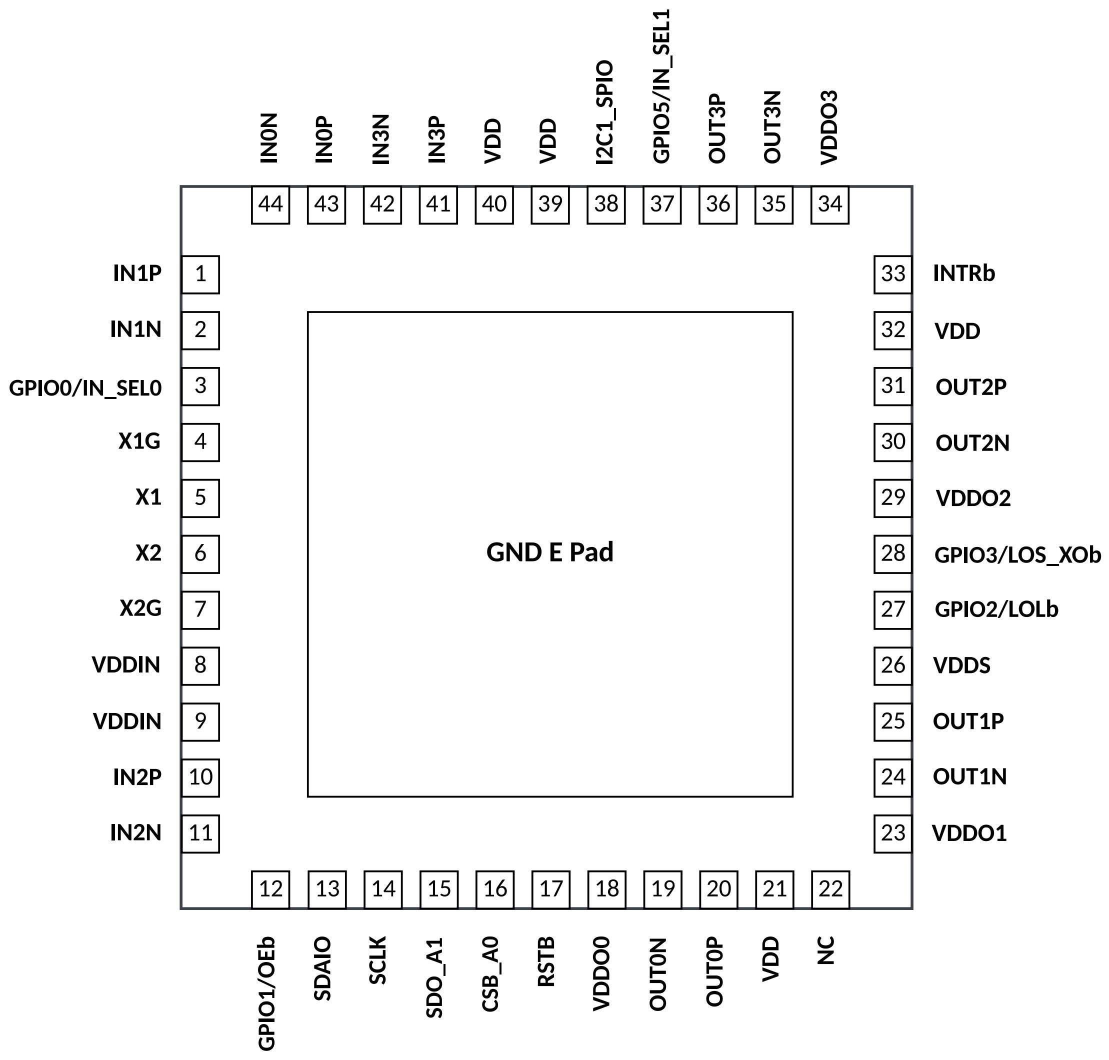


Figure 2. SiT95314 Pin Configuration

Table 14. Pin Description

Name	I/O Type	Pin No	Voltage Level	Default Pull Up/ Down	Description
IN1P	Input	1	-0.5V – 3.6 V[4]		Clock + for differential clock input. Single ended Input clock.
IN1N	Input	2	-0.5V – 3.6 V		Clock - for differential clock input. Single ended Input clock.
GPIO0/IN_SEL0	I/O	3	VDDIN (Pin 8) or VDD (Pin 32)[1]	Pull Down	Input Clock Selection for Manual selection of active clock. Can be left floating or pulled down to GND if not used. Pin can be programmed as other GPIO features.2
GPIO5/IN_SEL1	I/O	37	VDDIN (Pin 8) or VDD (Pin 32) [1]	Pull Down	
X1G	Gnd	4			Optional Crystal X1 Pin and accompanying ground pin. External reference clock input.
X1	Input	5			
X2	Output	6			Optional Crystal X2 Pin and accompanying ground pin. Leave unconnected if External oscillator is connected to X1.
X2G	Gnd	7			

Name	I/O Type	Pin No	Voltage Level	Default Pull Up/ Down	Description
VDDIN	Power	8 9	3.0V – 3.6V		Analog Power Supply for Input buffers, dividers and Input Ref Clk buffer.
IN2P	Input	10	-0.5V – 3.6 V		Clock + for differential clock input. Single ended Input clock.
IN2N	Input	11	-0.5V – 3.6 V		Clock - for differential clock input. Single ended Input clock.
GPIO1/OEb	I/O	12	VDDIN (Pin 8) or VDD (Pin 32) [1]		Used to disable (when 1) all the output clocks. Can be left floating or pulled down to GND if not used. Pin can be programmed as other GPIO features.[2]
SDAIO	I/O	13	VDDIN (Pin 8) or VDD (Pin 32) [1]	Pull Up	I2C Serial Interface Data (SDA) / SPI Input data (SPI 4 wire)) and SPI Input Output (SPI 3 Wire)
SCLK	I/O	14	VDDIN (Pin 8) or VDD (Pin 32) [1]	Pull Up	I2C Serial Interface Clock or SPI Clock Input.
SDO_A1	Output	15	VDDIN (Pin 8) or VDD (Pin 32) [1]	Pull Down	Serial Data Output (SPI 4 wire Interface). In I2C mode this is the A1 address pin (see Serial Interface section).
CSB_A0	Input	16	VDDIN (Pin 8) or VDD (Pin 32) [1]	Pull Up	Chip Select for the SPI Interface. In I2C mode this is the A0 address pin (see Serial Interface section).
RSTB	Input	17	VDDIN (Pin 8) or VDD (Pin 32) [1]	Pull Up	Active low reset Input. Reset should be held low and released from low to high only when all supplies to the chip have crossed 90% of their final value.
VDDO0	Power	18	1.71V-3.6V		Analog Power Supply for OUT0 drivers
OUT0N	Output	19	VDDO0 (Pin 18)		Clock – for differential Output. Single ended CMOS Output.
OUT0P	Output	20	VDDO0 (Pin 18)		Clock + for differential Output. Single ended CMOS Output.
NC/Reserved		22			No Connection. Leave floating
VDDO1	Power	23	1.71V-3.6V		Analog Power Supply for OUT1 drivers
OUT1N	Output	24	VDDO0 (Pin 23)		Clock – for differential Output. Single ended CMOS Output.
OUT1P	Output	25	VDDO0 (Pin 23)		Clock + for differential Output. Single ended CMOS Output.
VDDS	Power	26	1.71V-3.6V		Power Supply for Pin 27 and Pin 28 GPIO supply.
GPIO2/LOLb	I/O	27	VDDS (Pin 26)	Pull Down	Active Low. Loss of Lock Indicator for any or all PLLs. Pin can be programmed as other GPIO features.[2]
GPIO3/LOS_XOb	I/O	28	VDDS (Pin 26)	Pull Down	Active Low. Loss of Lock Indicator for any or all PLLs. Pin can be programmed as other GPIO features.[2]
VDDO2	Power	29	1.71V-3.6V		Analog Power Supply for OUT2 drivers
OUT2N	Output	30	VDDO0 (Pin 29)		Clock – for differential Output. Single ended CMOS Output.
OUT2P	Output	31	VDDO0 (Pin 29)		Clock + for differential Output. Single ended CMOS Output.
INTRb	I/O	33	VDDIN (Pin 8) or VDD (Pin 32) [1]	Pull Down	Active low indicator of programmable sticky notifies.
VDDO3	Power	34	1.71V-3.6V		Analog Power Supply for OUT3 drivers
OUT3N	Output	35	VDDO0 (Pin 34)		Clock – for differential Output. Single ended CMOS Output.
OUT3P	Output	36	VDDO0 (Pin 34)		Clock + for differential Output. Single ended CMOS Output.
I2C1_SPI0	Input	38	VDDIN (Pin 8) or VDD (Pin 32) [1]	Pull Up	Used as I2C/SPI select. 1: I2C 0: SPI
VDD	Power	21 32 39 40	1.71 V – 1.89 V [3]		Digital Power Supply for PLL
IN3P	Input	41	-0.5V – 3.6 V		Clock + for differential clock input. Single ended Input clock.

Name	I/O Type	Pin No	Voltage Level	Default Pull Up/ Down	Description
IN3N	Input	42	-0.5V – 3.6 V		Clock - for differential clock input. Single ended Input clock.
IN0P	Input	43	-0.5V – 3.6 V		Clock + for differential clock input. Single ended Input clock.
IN0N	Input	44	-0.5V – 3.6 V		Clock - for differential clock input. Single ended Input clock.
Ground	Gnd	ePad			Ground Pad

Notes:

1. GPIO supply is software selectable voltage between VDDIN & VDD
2. Refer to GPIO_Latching section for more details on GPIO function.
3. Pin 21, 32, 39, 40 power supply levels should be same and shorted on board.
4. The limits on the -ve voltage levels can be relaxed for AC coupled inputs, contact SiTime for more details

Functional Description

The SiT95314 is a jitter cleaning frequency translation device that offers four independent PLLs. The fully integrated part offers low integrated jitter for higher data rate links along with low close-in noise and full JESD204B/C support for RF clocks. These features provide a unique feature rich definition in a highly integrated part.

One of the four differential (or 8 single ended) clocks can be routed to four independent PLLs. A fully flexible output RF multiplexer allows any PLL's output to be routed to any output. This offers a very flexible frequency translation arrangement with independent control of each PLL in terms of jitter attenuation, bandwidth control and input clock selection with redundancy. The hierarchy of the clocks, nomenclature of the various frequency dividers as well as the clock translation pathways available on the chip are shown in Figure 3, Figure 4, and Figure 5.

The digital architecture of the chip is partitioned into a master digital controller and seven slave controllers. The master controller and each of the seven controllers has an associated volatile programmable interface (PIF). The overall PIF structure is a register map that is divided into several pages according to function. Each controller (master and slaves) has an associated unique Page number. Each Page has an independent 8 bit addressable PIF memory. In all the pages, the last address, FF, holds the current page number and is reserved for changing the page. The current page to be communicated with, can be set by writing the page number in hexadecimal form in the address FF on any page.

- Page 0,1: Master System
- Page 2: Input System
- Page 3,4: Output System
- Page A, 1A: PLL A
- Page B, 1B: PLL B
- Page C, 1C: PLL C
- Page D, 1D: PLL D
- Page 6, 7: Clock Monitor System

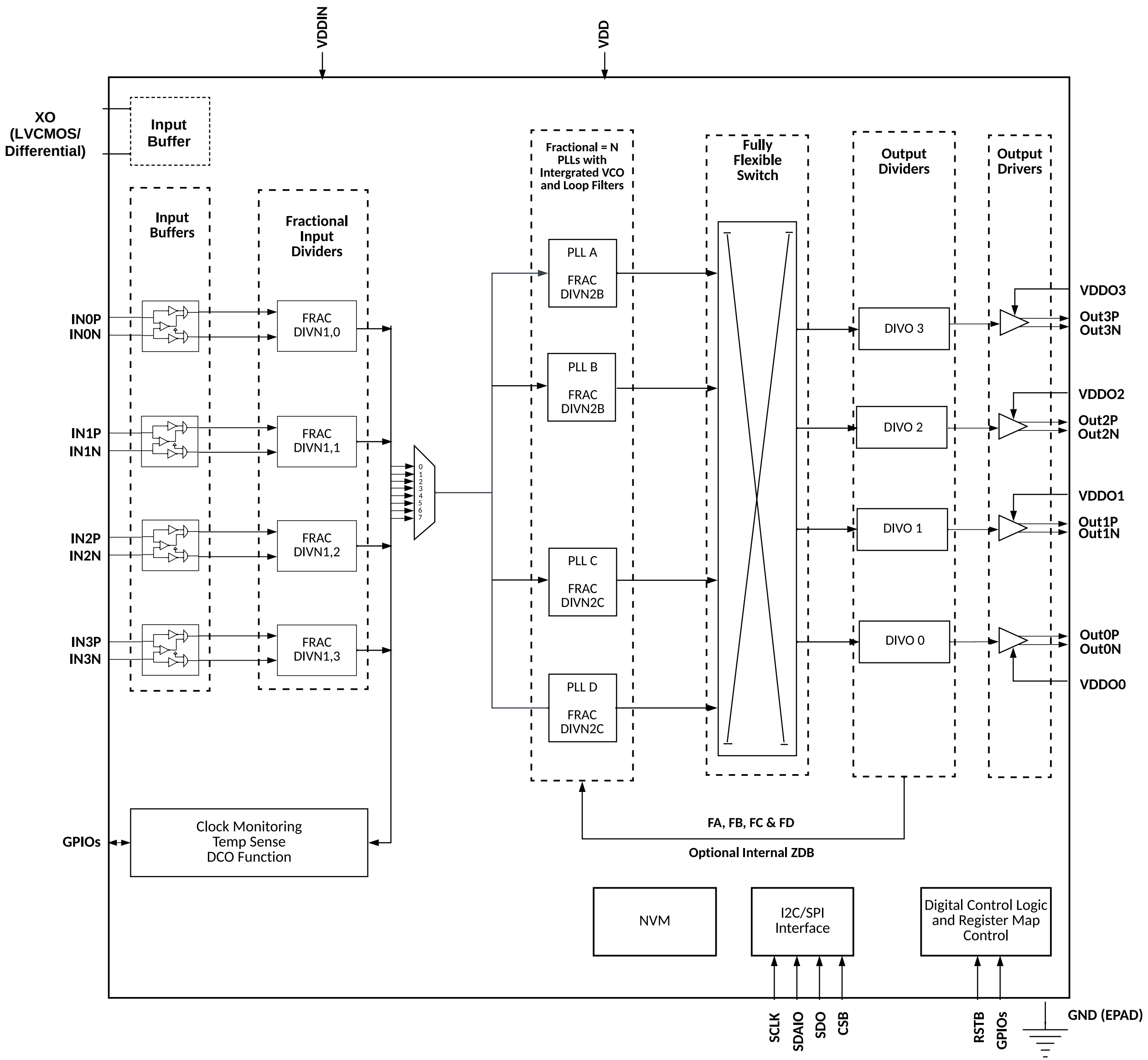


Figure 3. SiT95314 Functional Block Diagram

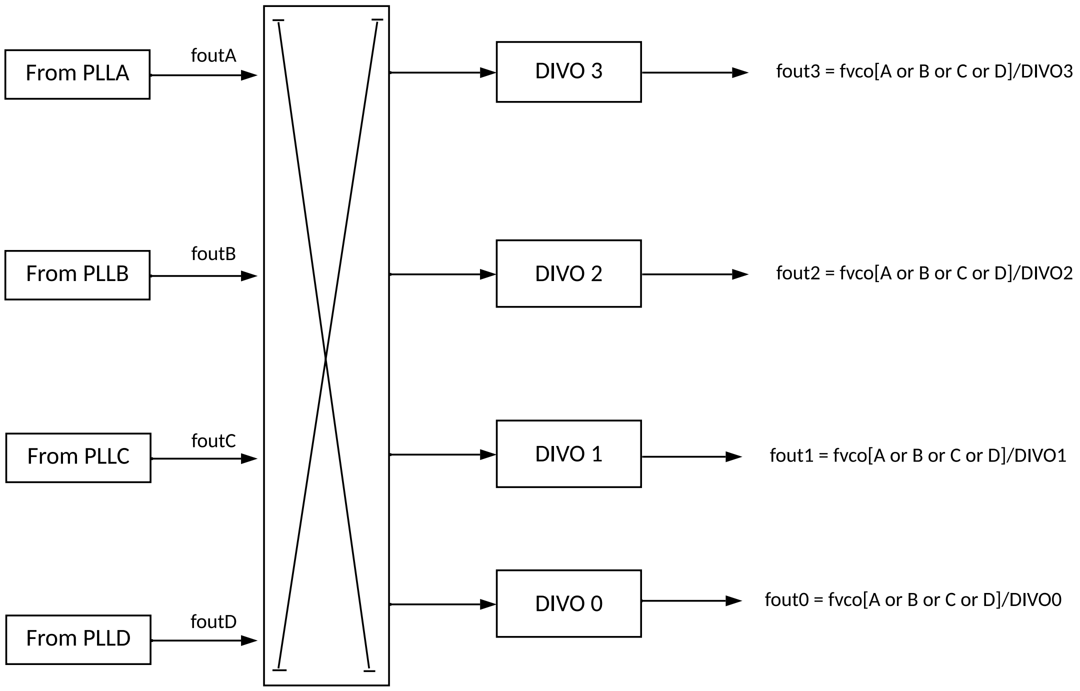

From PLLA
foutA
From PLLB
foutB
From PLLC
foutC
From PLLD
foutD
DIVO 3
DIVO 2
DIVO 1
DIVO 0
fout3 = fvco[A or B or C or D]/DIVO3
fout2 = fvco[A or B or C or D]/DIVO2
fout1 = fvco[A or B or C or D]/DIVO1
fout0 = fvco[A or B or C or D]/DIVO0

Figure 4. Output Clock Distribution

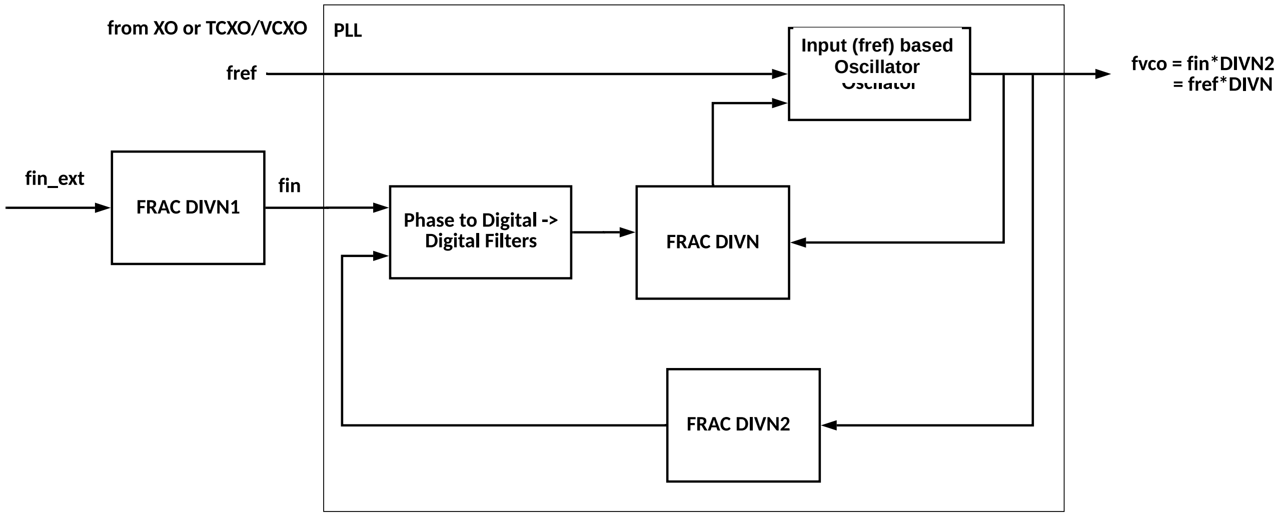


Figure 5. PLL Internals

Note: Similar for all PLLs

Table 15. PIF Description

Page	Contents	Summary of contents
0	Master	All Generic Information related to the chip Chip Configuration details Control for the master sequencer FSM Input Reference Clock Related Information
1		
2	Input Slave	Input 0N, 0P, 1N, 1P, 2N, 2P, 3N, 3P related information (Input type, DIVN1 divider configuration)
3	Output Slave	ODR 0, 1, 2, 3 ODR Standards, DIVO, Programmable delay configurations for each
4		
6	ClkMon Slave	Clock Loss related Function. Frequency Drift related function
7		
A	PLLA Slave	All PLL related functionality
1A		
B	PLL B Slave	
1B		
C	PLL C Slave	
1C		
D	PLL D Slave	
1D		

Master and Slaves: Architecture Description and Programming Procedures

Overview of the programming procedure

The SiT95314 part can be programmed using different methods. The chip has a serial SPI/I2C port to access the Programmable Interface (PIF) that comprises of the register map. Additionally, key parts of that register map can be populated from the on chip one time programmable memories (NVM OTP eFuse memories) as well as from an on board I2C addressable EEPROM. The on board EEPROM is provided as an optional additional feature adding more flexibility to the programmable autonomous functions for the part at wake up. The chip can operate with all features being available without the on board EEPROM also.

EEPROM is enabled on Special part numbers. Generic part doesn’t support EEPROM.

The part can be used in one of the following three configurations:

Configuration 1: Completely autonomous wake up and reaching the final operational state with output clocks available with no SPI/I2C serial port access needed. In such cases there are three options available.

Option 1: This can be with a configuration that is loaded entirely in the on-chip OTP eFuse memory.

Option 2: Alternatively, this can be with a configuration that is loaded entirely in the on board I2C addressable EEPROM.

Option 3: Alternatively, this can be with a configuration that is loaded partly from the OTP and partly from the on board I2C addressable EEPROM. (All of the above three autonomous wake up options involve the use of the lock pattern that indicates an autonomous wake up. The use of the lock pattern for the master and slave memories is described in more detail in Master and Slaves: Architecture Description and Programming Procedures

Configuration 2: Partial autonomous wake up such that one PLL and a few outputs are available at wake up without any SPI/I2C serial port access, while the remaining part of the chip is programmed using the SPI/I2C serial interface.

In this case, the on-chip OTP memory normally contains the partial wake up configuration that enables a single PLL and up to 3 outputs. However, if the user desires such a partial

wake up configuration can also be stored in and obtained from the on board EEPROM.

After the initial configuration with up to 3 outputs being available from a single PLL autonomously, the serial SPI/I2C ports can be used to program the remaining PLLs and outputs as well as enable more outputs from the PLL that was pre-programmed to provide the autonomous outputs at the wake up.

Configuration 3: In this case the chip has no output clocks available without the programming by the customer using the SPI/I2C serial port. Hence a custom profile is programmed by the user in the field using the serial SPI/I2C port. This is the most commonly used configuration for setting up the chip.

The chip can get partial settings from both the on-chip OTP and the optional on board EEPROM after which it is programmed with the desired configuration from the serial SPI/I2C interface. This feature of using partial configuration from the OTP memory and/or the optional EEPROM can still be done even if there is no output clock autonomously available.

Alternatively, the entire chip programming sequence is handled using the SPI/I2C profile. This will be the most commonly used setting within Configuration 3.

The on chip NVM memory in every case contains basic configuration information that is recorded during the automatic test procedure in production. At every wake up, the part downloads the basic information from the on chip NVM OTP memory. After that, the chip then reaches its operational steady state condition in any of the three configurations listed above.

Master and Slaves: Architecture Description and Programming Procedures

The chip comprises of a Page based mechanism that is split between a Master page and several Slaves. As a first step to understand the wake up arrangement, the memory arrangement of the Master and Slaves is described.

The Master controller is the first system to autonomously wake up on the application of power to the chip due to on-chip power on reset circuitry. All generic system information resides in the Master controller memory and it proceeds to wake up the Slaves as required based on this information. The relative wake up sequences of the Master and the various Slaves are described in more detail after the description of the memory structures.

The Master memory structure is shown in Figure 6. It contains a one-time programmable non-volatile memory (NVM) that stores the settings for the chip associated with the master controller. The master controller also contains a volatile PIF bank (NVMCopy) that has an exact copy of the NVM at every chip power up. This volatile PIF (NVMCopy) is the memory that is addressable using the serial interface (I2C / SPI) on Page 0 and can be overwritten from the I2C / SPI interface. This volatile PIF can also be loaded with settings which are stored on external I2C EEPROM. Procedure to configure the device from external I2C EEPROM is described after description of wake up sequence of the Master and the various slaves. The "Chip Settings" is the memory space that is not addressable from the I2C / SPI / EEPROM control and is the actual control for the chip.

For all configuration wake ups where the chip is configured using the I2C/SPI interface, the autonomous wake up is not needed. For the cases where an autonomous wake up is desired, the lock pattern is used. The NVM contains a two bit "Lock Pattern" that can be set to "10" or "01" to 'lock' the chip configuration once the final configuration is determined and wake up of the entire chip is desired in this configuration using a completely autonomous wake up. Additionally, there is a bit in the NVM that is an active low indicator of a manual wake up. This bit set to "1" along with the 'lock' for the configuration leads to an autonomous wake up of the chip using the 'locked' configuration. Any number of different configurations can alternatively be tried at all times using only the volatile NVMCopy PIF section which is the default mode for the cases where an autonomous wake up is not desired. This is useful for evaluations as well as allowing real time programming of the chip in various configurations with complete flexibility in the field.

The Master Controller Finite State Machine (FSM) described later in this section controls the device behaviour in accordance with the configuration in this memory structure and as per the wake up mode.

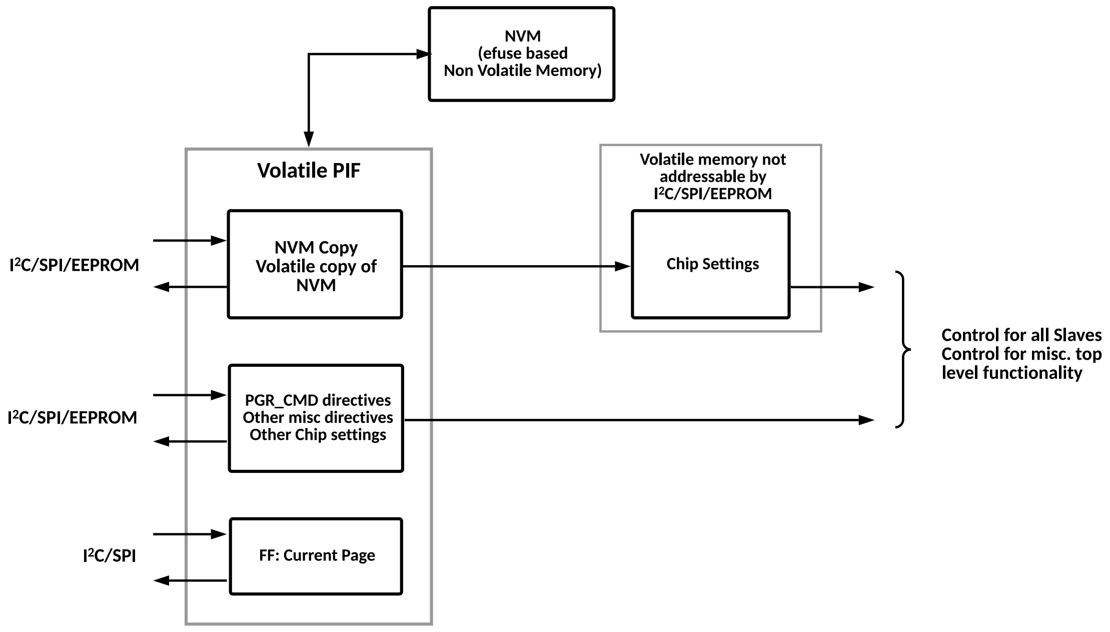


Figure 6. Master Memory Structure

The memory structure for each slave is shown in Figure 7 and is similar in construction to the master controller memory structure with some minor differences. The NVMCopy volatile PIF for the slave is addressable by the serial interface with the unique Page number associated with the slave. The "Slave Settings" is the memory space that is not addressable from the I2C / SPI / EEPROM control and is the actual control for the slave.

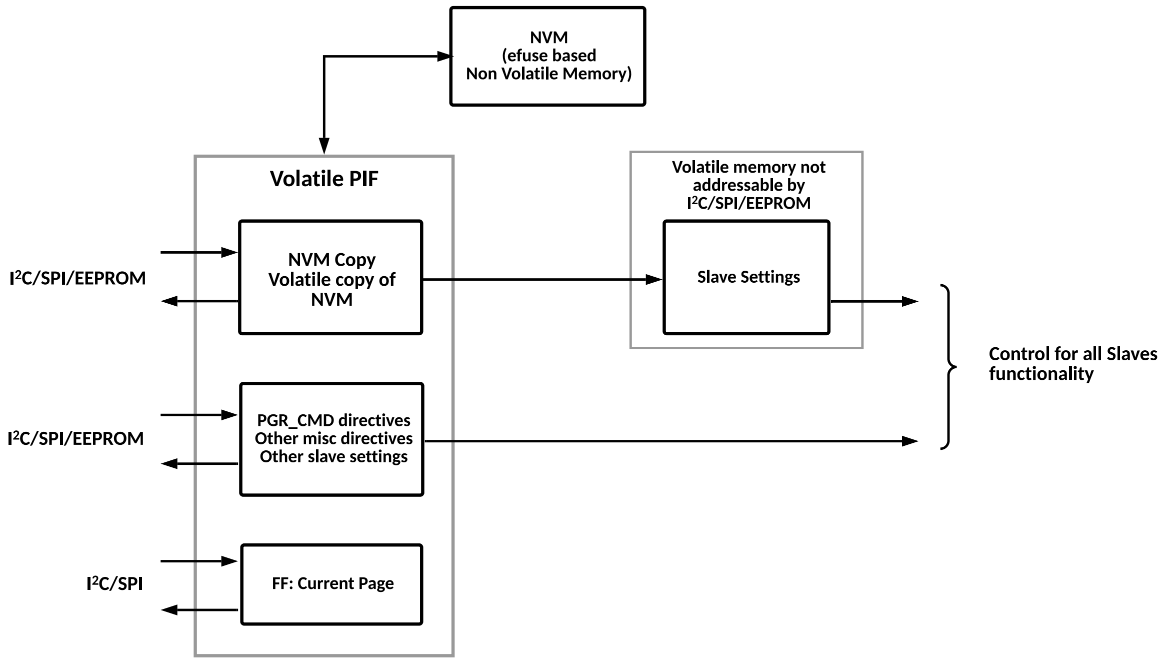


Figure 7. Slave Memory Structure

The Master Wake-Up Finite State Machine (FSM) is shown in more detail in Figure 8. At every power up of the device (or release from hard reset), the power-on-reset circuitry resets all systems and then autonomously releases only the master controller from reset. After releasing reset, the status of the GPIO pins will be latched. After hold time of 1ms GPIOs can release their reset levels and assume their normal operation modes.

Latching is disabled by default in SiT95314. GPIO latching and its corresponding latching functions are factory programmed by SiTime with non 00 code marking. Please contact SiTime for more details.

The NVM contents are copied to the NVMCopy volatile space on Page 0 which is in turn copied to the "Chip Settings". Default Chip configuration is as shown in Figure 16.

Table 16. Default Chip Configuration

Chip Configuration	Configuration Mode	Default settings	Description
SPI Mode 3wire/4wire	Page0 reg0x33[5]	0	If SPI mode is selected using I2C1_SPI0 pin then 0 : 4Wire SPI 1 : 3Wire SPI
I2C Address A2	Page0 reg0x33[6]	0	I2C address A2 will be set by this value.
EEPROM Access Enable	Page0 reg0x33[4]	0	0 : External I2C EEPROM access is not available 1 : External I2C EEPROM is available

Chip can be also configured from the GPIOs by enabeling the GPIO latching. If GPIO latching is enabled then GPIOs can be configured as shown in Table 18.

GPIO latching mechanism is described in section GPIO Latching. The latching of the GPIOs at release of reset allows the chip to re-use the same GPIOs for other functions in the steady state hence making a more efficient use of the same resource. Using NVM settings GPIO latching can be enabled. If GPIO latching is enabled, the master controller decodes the latched GPIO input using Chip Settings as shown in Table 19. If GPIO latching is disabled then initial chip configuration will be done according to NVM settings shown in Table 16.

During wakeup master controller configures the GPIOs for regular operations with default settings as shown in Table 17. GPIOs can be configured for different functions as described in section GPIO Modes during Regular Operation.

Table 17 Default GPIO Configuration

GPIO	PIN	Default Direction	Default Pull Up / Pull Down	Default Function	Description
GPIO0	3	Input	Weak Pull Down	IN_SEL0	Manual Input Clock Select Bit0
GPIO5	37	Input	Weak Pull Down	IN_SEL1	Manual Input Clock Select Bit1
GPIO1	12	Input	Weak Pull Down	OEb	Output Disable Control
GPIO3	28	Input	Weak Pull Down	LOS_XOb	XO Loss Status
GPIO2	27	Output	Weak Pull Down	LOLb	All PLL Loss of Lock Status

During wakeup master controller looks for I2C EEPROM access, if I2C EEPROM access is available and I2C mode is selected through I2C1_SPI0 pin, the master configures one of available GPIOs as EEPROM read done output, then it becomes the I2C master and starts searching the configuration block in available EEPROMs. If respective configuration block with Header and CRC is found in EEPROM then it re-configures the volatile PIF and Chip Settings with EEPROM data. After successful EEPROM read, the master notifies on a GPIO that the EEPROM read is done and becomes either SPI or I2C slave depending upon the configuration word read from the EEPROM. If EEPROM read fails due to any error then it will notify respective error along with EEPROM read done indication and becomes I2C slave. After EEPROM read done notification, status of EEPROM read can be checked by reading respective EEPROM error notify register or one of the available GPIO can be configured to show the EEPROM error notify output. If there is no EEPROM error notify along with EEPROM read done, it means EEPROM read is successful. If EEPROM read fails, the data inside volatile PIF may get corrupted so master does not update the Chip Settings; Master becomes either I2C slave and proceeds to Program Command Wait State. Configuring Chip Settings from EEPROM is described later down the same section. If EEPROM access is not available or EEPROM read is successful, the master controller now decides if the chip configuration is locked and it is an autonomous wake up of the entire chip or if a manual wake up is desired through the PIF based on the contents in the "Chip Settings".

In case a "Lock" is detected and an autonomous wake up is desired, the Master controller proceeds to enable the input reference clock buffer and associated fref pathways followed by the Slave systems in a pre-determined sequence. This finally leads the chip to the "Active State" with all desired outputs available as a result of all slave systems released from reset by the master controller. This is according to the requested settings that are programmed in the Master and the Slave NVM banks.

For the case where the final chip settings are not frozen hence the "Lock" pattern is not exercised, the master controller FSM reaches the Program Command Wait State (PRG_CMD). The desired chip settings can be written in the NVMCopy on Page 0 using the serial interface and desired slave sub-systems can be enabled. Several PRG_CMD state directives are available that are exercisable only in this state (refer for the master Finite State Machine). Using these directives, the desired settings written in NVMCopy can now be copied to "Chip Settings" followed by issuing the directive for the FSM to proceed to the "Active" state where each slave can now be manually written with the desired settings and in turn asked to proceed to its "Active" state.

A similar "Lock" pattern is available in the NVM bank of each slave. The currently used NVM bank for a slave can be locked for the autonomous wake up of each slave. The slave wake-up FSM is shown in Figure 9 and it similarly has a PRG_CMD state with associated directives. On Proceed to Active state directive on the slave, the slave controller wakes up the various blocks in its sub-system with the correct pre-determined sequence.

The NVM bank for the master and each slave can be programmed with a PRG_CMD directive in that state to lock a configuration / setting specific to the respective sub-system.

Each FSM (Master and Slaves) allows an escape sequence to go back to PRG_CMD state from its Active State. This can be used to selectively change the settings for that particular sub-system.

Note that the NVM for the master controller and for all slaves should be locked after writing desired settings for a completely autonomous wake up of the entire chip. For evaluations of the chip as well as cases where flexible on-the-fly programmable settings are desired, the chip can be used without engaging the NVM banks at all by using the NVMCopy space for the master and each slave in conjunction with the PRG_CMD directives. It is also possible to lock some of the slaves (to not re-write their settings for each wake up) while use programmable settings for other slaves.

This provides complete flexibility in terms of programming and using the chip in all scenarios.

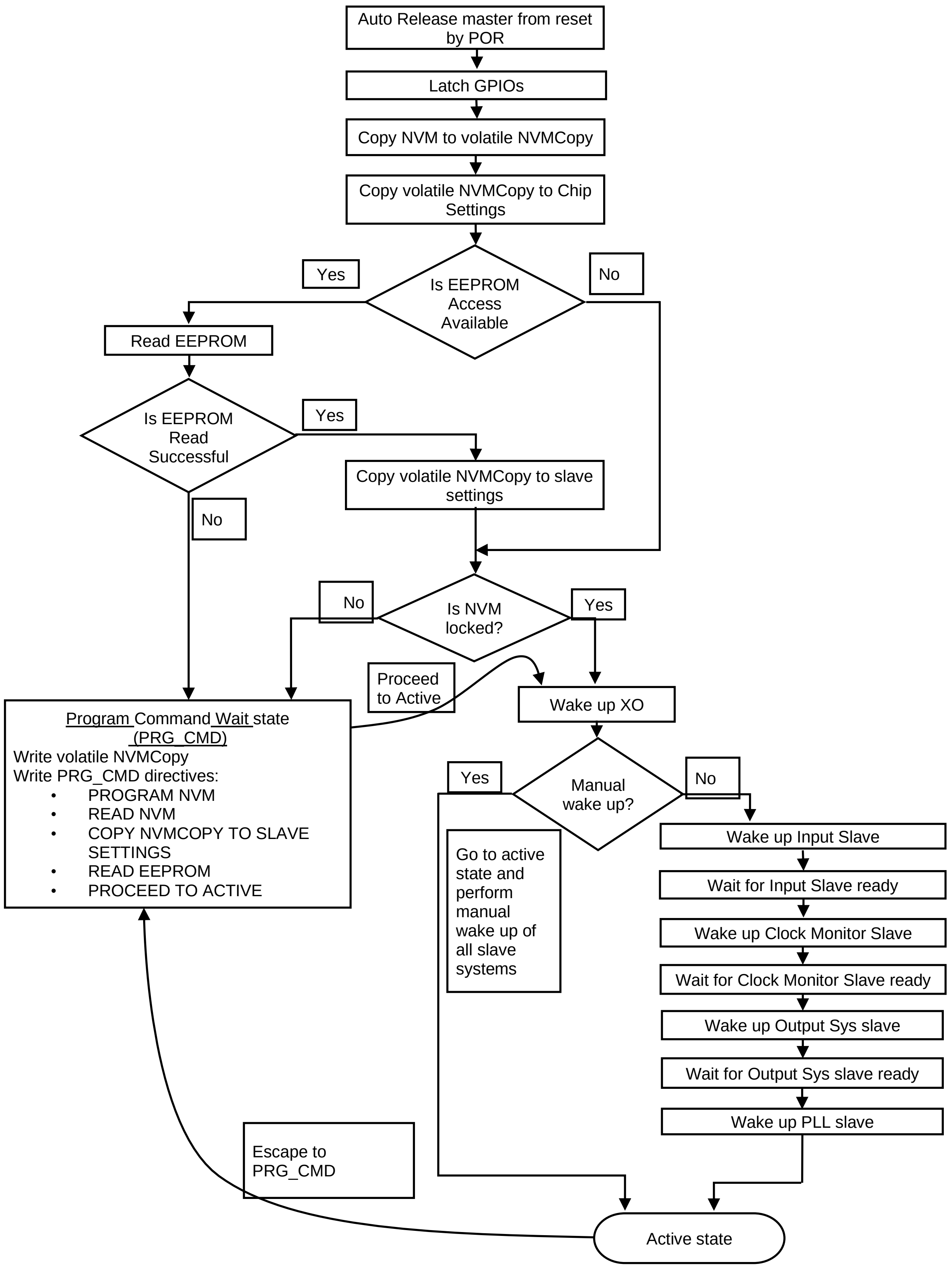

Auto Release master from reset by POR
Latch GPIOs
Copy NVM to volatile NVMCopy
Copy volatile NVMCopy to Chip Settings
Is EEPROM Access Available
Yes
No
Read EEPROM
Is EEPROM Read Successful
Yes
No
Copy volatile NVMCopy to slave settings
Is NVM locked?
No
Yes
Proceed to Active
Wake up XO
Program Command Wait state (PRG_CMD)
Write volatile NVMCopy
Write PRG_CMD directives:
PROGRAM NVM
READ NVM
COPY NVMCOPY TO SLAVE SETTINGS
READ EEPROM
PROCEED TO ACTIVE
Manual wake up?
Yes
No
Go to active state and perform manual wake up of all slave systems
Wake up Input Slave
Wait for Input Slave ready
Wake up Clock Monitor Slave
Wait for Clock Monitor Slave ready
Wake up Output Sys slave
Wait for Output Sys slave ready
Wake up PLL slave
Escape to PRG_CMD
Active state

Figure 8. Master Wake-Up Finite State Machine

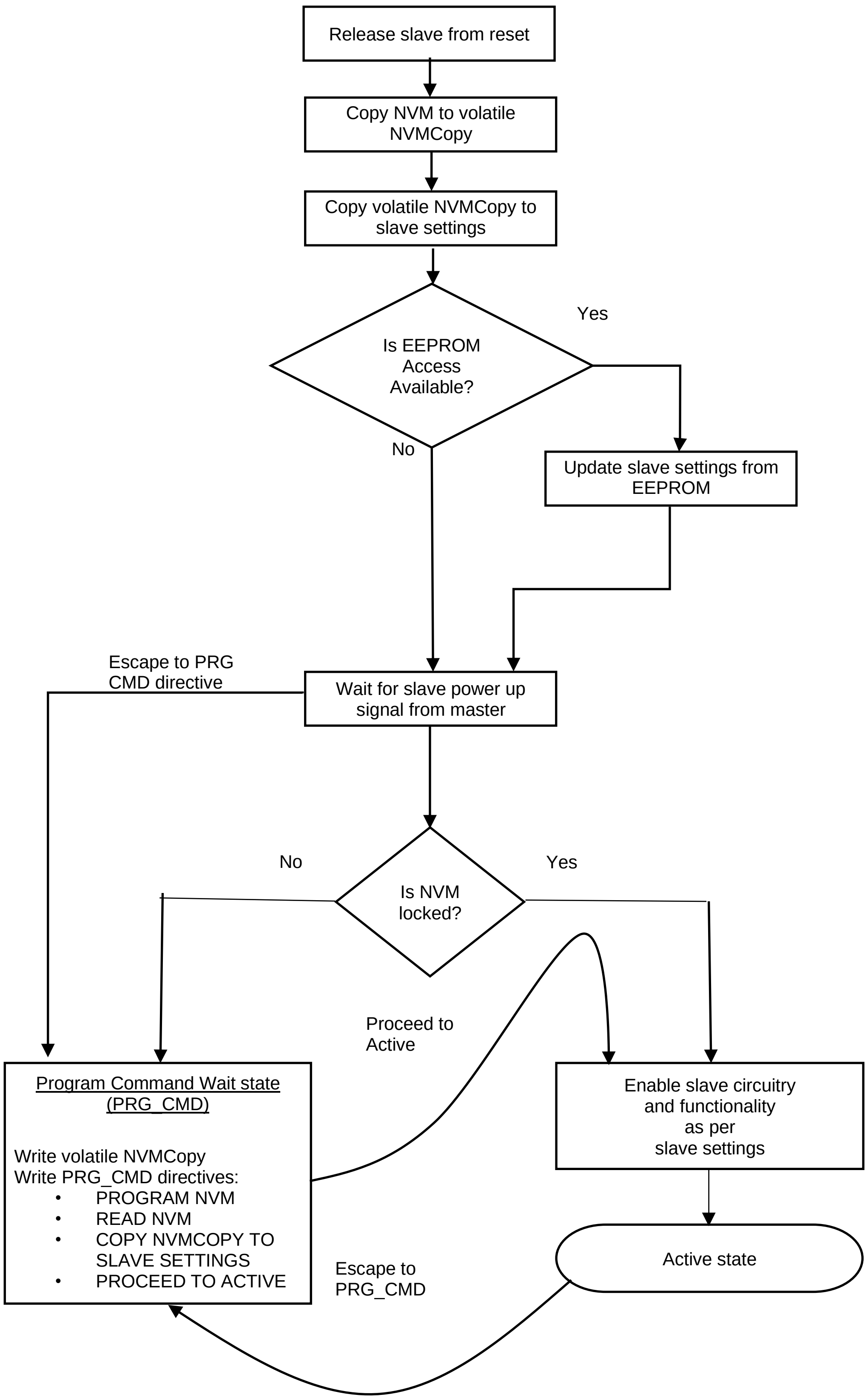

Release slave from reset
Copy NVM to volatile NVMCopy
Copy volatile NVMCopy to slave settings
Is EEPROM Access Available?
Yes
No
Update slave settings from EEPROM
Escape to PRG CMD directive
Wait for slave power up signal from master
No
Is NVM locked?
Yes
Proceed to Active
Program Command Wait state (PRG_CMD)
Write volatile NVMCopy
Write PRG_CMD directives:
PROGRAM NVM
READ NVM
COPY NVMCOPY TO SLAVE SETTINGS
PROCEED TO ACTIVE
Enable slave circuitry and functionality as per slave settings
Active state
Escape to PRG_CMD

Figure 9. Slave Wake-Up Finite State Machine

GPIO Latching

GPIO Latching is used to configure the chip during wakeup. GPIO latching is disabled as default factory settings and chip is configured as shown in Table 16.When reset is applied available GPIOs will be configured as shown in Table 18. After releasing reset, the status of the GPIO pins will be latched. After hold time of 1ms GPIOs can release their reset levels and assumes their normal operation modes. Latched value of GPIOs will be decoded later when NVM contents are copied to Chip Settings. Using GPIO latch configuration from Chip Settings, latched value of GPIOs will be decoded as shown in Table 19. The use of GPIO latching ensures optimal use of resources by making the same GPIOs that are latched at wake up available for other alternate functions during regular operation.

Table 18. GPIO Latching Configuration

GPIO	PIN	Default Direction	Default Pull Up / Pull Down	Function	Description
GPIO0	3	Input	Weak Pull Down	User Defined	User can define the functionality of these GPIOs using GPIO latch configuration as shown in Table 19
GPIO5	37	Input	Weak Pull Down	User Defined	
GPIO1	12	Input	Weak Pull Down	User Defined	
GPIO3	28	Input	Weak Pull Down	User Defined	
GPIO2	27	Input	Weak Pull Down	User Defined	

Table 19. GPIO Latch Decoding

gpio_latch_cfg[2:0]	Function	Description
3'd0	Unused default	If any GPIO is configured with this settings then latched value will be ignored
3'd1	EEPROM access disable	Disable EEPROM access during Wakeup sequence 0 : If I2C mode selected using I2C1_SPI0 pin, device will try to read the configuration block from external EEPROM. 1 : Device will not attempt to read the configuration block from external EEPROM. By default, no GPIO is used for this purpose, so the device will try to read the configuration block from external EEPROM.
3'd2	I2C slave address A2	If I2C mode is selected using I2C1_SPI0 pin, this GPIO will be used as I2C slave address A2. In SPI mode latched value will be ignored. By default, no GPIO is used for this purpose, so default I2C slave address will have a 0 for bit A2.
3'd3	SPI mode 3wire 4wire	If SPI mode is selected using I2C1_SPI0 pin, this GPIO will be used as control for 3-wire/4-wire mode of SPI. In I2C mode latched value will be ignored. 0 : 4wire mode 1 : 3wire mode By default, no GPIO is used for this purpose, so device will be in 4-wire SPI mode.
3'd4	Spare	If any GPIO is configured with this settings then latched value will be ignored
3'd5	Spare	If any GPIO is configured with this settings then latched value will be ignored
3'd6	Spare	If any GPIO is configured with this settings then latched value will be ignored
3'd7	Spare	If any GPIO is configured with this settings then latched value will be ignored

By default GPIO latching function is not enabled on SiT95314. GPIO latching and its corresponding latching functions are factory programmed by SiTime with non 00 code marking. Please contact SiTime for more details.

Using GPIO latch configuration any of the GPIO can be used as following:

- EEPROM Access Disable control: If device is configured as I2C slave during the start-up sequence using the I2C1_SPI0 pin, a high input value on a GPIO programmed with this function prevents a device from attempting to read configuration data from an external I2C EEPROM. If no GPIOs are configured in this mode then the device will attempt read configuration data from an external I2C EEPROM. If multiple GPIOs are configured to perform this function then any one of them being active will disable EEPROM accesses, so it is recommended that no more than one GPIO be programmed for this function.
- I2C Address bit A2: If a GPIO is configured to this function and serial port is selected as I2C during the start-up sequence using the I2C1_SPI0 pin, latched value of configured GPIO will be used as I2C slave address A2. If no GPIOs are configured in this mode, bit A2 of the slave serial port base address will be zero. If more than one GPIO is programmed with this functionality, only the one with the highest index will be used (e.g., if both GPIO5 and GPIO1 are programmed to do this, only GPIO5 would be used).
- SPI Mode 3-Wire/ 4-Wire: If device is configured as SPI slave during the start-up sequence using the I2C1_SPI0 pin, a high input value on a GPIO programmed with SPI Mode 3-Wire/ 4-Wire function set the device in 3-wire SPI mode. If no GPIOs are configured in this mode, then the device will be set to 4-wire SPI mode. If multiple GPIOs are configured to perform this function then any one of them being active will set the device in 3-wire SPI mode, so it is recommended that no more than one GPIO be programmed for this function.

Status vs Notify available on the Chip

SiT95314 provides various Status and notify bits that can be accessed from the register map. Below are the details of the procedure to be followed to access the same.

The alarm registers are a set of three types of registers distributed between the various pages as described in Table 26.The Status registers are the current dynamic status of a defect. The live status defects are active high with a ‘1’ indicating the defect is present.

- The Notify registers are the sticky bits for a defect. Sometimes, we may get a very short pulse for the status and it may not be possible for the external user to capture the same. Hence a notify register is provided. The notify register is set to 1 whenever there is a rising edge of the corresponding status register. This is a sticky bit and stays at 1 till the user writes a 1 to that specific bit to clear it.
- Each notify has a masking bit to enable or disable its operation. The notify sticky bit operates only if the corresponding masking bit is set to 1. If the masking register bit is set to 0, notify will not be asserted even when status toggles. The default value for the mask register is 0xff so all the notify signals are enabled. Once the user writes a 1 to clear the notify, the notify bit can again go high on the next rising edge of the status.

These registers operate on the internal 4 MHz RC clock. When there is a defect (i.e. status) of any bit in register it gets asserted and de-asserted in a live mode. User can read the corresponding register location to see the current status at any time.

On the other hand, the default value of the notify and masking register is 0xff – user has to write 0xff to both these registers to clear them at the beginning and use all notifies.

GPIO Modes during Regular Operation

Using Chip Settings, all the available GPIOs can be configured to one of the following modes. Also each GPIO can be pulled up, pulled down or set to high impedance state individually using respective Chip Settings. GPIOs will be configured to respective mode in master wake up sequence when NVM content is copied to Chip settings. This is the regular steady state operational mode for the GPIO which can be configured differently compared to the functionality that is latched at the release of reset. GPIOs can also be reconfigured with settings from EEPROM or from the volatile register writes during chip configuration. All the GPIOs are fully configurable so that any GPIO can perform any function of the following:

- Input Modes:
  - General Purpose Input - In this mode of operation, the GPIO pin will act as an input that is latched to specified internal register. That register can be read over the serial port.
  - DCO Select – This is used such that the PLL that is used for DCO can be selected. The same function is available directly from the register map as well alternatively.
  - DCO Increment – A low to high transition on the DCO increment pin causes a DCO increment on the selected PLL. The same function is available directly from the register map as well alternatively.
  - DCO Decrement - A low to high transition on

the DCO decrement pin causes a DCO decrement on the selected PLL. The same function is available directly from the register map as well alternatively.
    - Output Disable Control – This is an OEb pin such that it can be used as an active high to disable all the outputs.
    - SYS ref – SYSREF trigger related to the JESD204B function
    - Divider_restart – Divider restart can be used to dynamically change the relative delays between outputs from the same PLL where the delays are pre-loaded in the register map using the serial port and this pin is used as a GPIO trigger for the same
    - PLL Syncb Select –Selects which PLL to be used for the SYNCB trigger
    - PLL Syncb trigger – The actual SyncB trigger that is used for the selected PLL
    - Manual Clock Select: GPIO can be used as manual clock selection for PLLs.
    - Clock Disqualification: GPIO can be assigned as trigger to disqualify selected clock as active clock to selected PLL.
- Output Modes : GPIOs can be in individually configured as the following outputs. GPIO output polarity is programmable.
    - General Purpose Output - In this mode of operation, the GPIO pin will act as an output that is driven to the logic level specified in an internal register. That register can be written over the serial port.
    - Alarm Signal Outputs - Each GPIO can be independently configured to act as a Single-Purpose alarm or Aggregated alarm output. If multiple GPIOs are configured the same configuration settings, they will all have the same output values.
    - Clock Loss - In this mode the GPIO will act as Clock Loss output. There are three types of Clock Loss indication as
        - Clock loss - status as well as sticky notify bit can be give out on GPIO.
        - Frequency Drift - status as well as sticky notify bit can be given out on GPIO.
        - Clock loss with frequency drift - Only status bit can be given out on GPIO.
    - Loss of Lock Status - In this mode the GPIO will act as loss of lock indicator of respective PLL. There are three types of loss of lock indicators as
        - Inner Loop Loss of lock - Status as well as sticky notify bit can be give out on GPIO.
        - Outer Loop Loss of lock - Status as well as sticky notify bit can be give out on GPIO.
        - Phase loss of lock - Status as well as sticky notify bit can be give out on GPIO.
    - Holdover Freeze - In this mode the GPIO will act as Holdover indicator, which indicates that respective PLL is entered in Holdover mode. Status as well as sticky notify bit can be give out on GPIO.
    - EEPROM Status - In this mode the GPIO will act as EEPROM Status output to send out EEPROM defects. Status as well as sticky notify bit can be given out on GPIO.
    - EEPROM Read Done Indicator – In this mode GPIO will act as EEPROM read done Indicator to indicate that EEPROM read is done and SiT95314 became the slave. External micro controller can check the EEPROM status whether EEPROM read was successful or not. If there is no EEPROM defect then EEPROM read is successful.
    - Aggregated Alarm output - In this mode the GPIO will act as the logical OR of all alarm indicators that are enabled to drive this output. This output will be asserted if any of the “sticky” bits or status signals are asserted and enabled to cause the Alert (aggregated alarm). To clear this output, all contributing “sticky” bits must be individually cleared. This output will be active-high to indicate one or more alarms are asserted.

Configuring the Device from External EEPROM

SiT95314 can be configured from external EEPROM. During Wakeup sequence master checks for EEPROM access. If a GPIO is latched as 0 and decoded as eeprom access disable control and serial interface is configured to I^{2}C mode, SiT95314 tries to read the configuration block from EEPROM.

If EEPROM access is available and I^{2}C mode is selected, one of the GPIO will be configured to send out the EEPROM read done signal and the Master will start searching the configuration block in available EEPROMs with 400 KHz I^{2}C frequency and with EEPROM device address 7’b1010000. If there is no acknowledge from EEPROM then the Master increments EEPROM device address to 7’b1010001 and repeats the search. This will be repeated up to EEPROM device address is 7’b1010111. If there is still no acknowledge, it will repeat the search with 100 KHz I^{2}C frequency. If EEPROM controller receives acknowledge from the EEPROM then it starts reading the data from address 0x00000 of that EEPROM. While reading the data, it proceeds with checking the EEPROM size, device id, page id, page size and crc id. If mismatch occurs it sets the respective defect flag and starts looking for next available block or EEPROM until it gets valid configuration block. While reading the configuration data from EEPROM Master starts calculating the CRC, if calculated CRC matches with the received CRC, then all defect flags will be de-asserted and EEPROM read done indication will be sent out on the configured GPIO. If configuration block is not found in all possible data blocks or there is no acknowledge from EEPROM then sticky notifies for respective defect will be set and EEPROM read done signal will be sent out on configured GPIO. Once outer controller receives EEPROM read done signal it can look for the defect notifies, if none of the notifies are set then EEPROM read is successful and master proceeds to next step.

EEPROM data needs to be organized as shown in Table 20.

SiT95314 configuration data needs to be organized in blocks of 64Kb. One 64Kb block will have configuration data for one SiT95314 device. EEPROM can contain multiple configuration data blocks for multiple devices depending upon the size of EEPROM.

Table 20. EEPROM Data Organization

word_address	EEPROM data	size
0x00000	EEPROM Size	64 Kb block
0x00001	Device ID	
0x00002	Config Word	
0x00003	Page ID	
0x00004	Page Size	
0x00005	Page Register Address	
0x00006	Page Register Data	
0x00007	Page Register Address	
.	Page Register Data	
.	.	
.	.	
.	Page ID	
.	Page Size	
.	Page Register Address	
.	Page Register Data	
.	Page Register Address	
.	Page Register Data	
.	.	
.	.	
.	CRC ID	
.	crc[3]	
.	crc[2]	
.	crc[1]	
.	crc[0]	
.	.	
0x01FFF	.	
0x02000	Device ID	64 Kb block
0x02001	Config Word	
0x02002	Page ID	
0x02003	Page Size	

word_address	EEPROM data	size
0x02004	Page Register Address	
0x02005	Page Register Data	
0x02006	Page Register Address	
0x02007	Page Register Data	
.	.	
.	.	
.	Page ID	
.	Page Size	
.	Page register Address	
.	Page register Data	
.	Page register Address	
.	Page register Data	
.	.	
.	.	
.	CRC ID	
.	CRC[3]	
.	CRC[2]	
.	CRC[1]	
.	CRC[0]	
.	.	
0x03FFF	.	

EEPROM size :

0x00 -> 64 Kb EEPROM, 1 block of 64Kb.

0x01 -> 128 Kb EEPROM, 2 blocks of 64Kb.

0x02 -> 256 Kb EEPROM, 4 blocks of 64Kb.

0x03 -> 512 Kb EEPROM, 8 blocks of 64Kb.

0x04 -> 1024 Kb EEPROM, 16 blocks of 64Kb.

If block's starting address is 0x00000 then it should contain EEPROM size at location 0x00000, else if block's starting address is other than 0x00000 then it should not contain EEPROM size.

Table 21 shows available data blocks with their starting address in EEPROM.

Table 21. EPPROM Data Blocks

Blocks	starting address	EEPROM Size				
		64Kb	128Kb	256Kb	512Kb	1024Kb
1	00000	Yes	Yes	Yes	Yes	Yes
2	02000	No	Yes	Yes	Yes	Yes
3	04000	No	No	Yes	Yes	Yes
4	06000	No	No	Yes	Yes	Yes
5	08000	No	No	No	Yes	Yes
6	0A000	No	No	No	Yes	Yes
7	0C000	No	No	No	Yes	Yes
8	0E000	No	No	No	Yes	Yes
9	10000	No	No	No	No	Yes
10	12000	No	No	No	No	Yes
11	14000	No	No	No	No	Yes
12	16000	No	No	No	No	Yes
13	18000	No	No	No	No	Yes
14	1A000	No	No	No	No	Yes
15	1C000	No	No	No	No	Yes
16	1E000	No	No	No	No	Yes

- **Device ID:**

  Device ID field indicates to which SiT95314 device this configuration block belongs. This device id will be read by Master and compared with I2C device address of SiT95314 device, if device id matches configuration data from this block will be read otherwise EEPROM controller will start looking for next blocks.
- **Config Word:**

  Config Word field contains configuration data. After successful EEPROM Read, device will be configured with respective settings. Config_Word [0] is used to reconfigure the serial interface after successful EEPROM read.

  If Config_Word [0] = 0 then SiT95314 will be configured as SPI slave.

  If Config_Word[0] = 1 then SiT95314 will be configured as I2C slave.
- **Page ID:**

  Page ID field indicates to which page the configuration data belongs. This page id will be read by Master and respective page will be enabled to write the configuration data.
- **Page Size:**

  Page Size field contains a number which indicates how many registers of respective page needs to be configured from EEPROM.

  Page Register Address & Page Register Data:

  Page Register Address contains register address of respective page in which configuration data i.e. Page Register Data has to be written.
- **CRC ID:**

  CRC ID indicates that following four bytes (CRC [3], CRC [2], CRC [1] and CRC [0]) are the 32-bit CRC for respective configuration block.

Input Reference Clock Connectivity Options

The CMOS XO/TCXO output and the termination components should be placed as close as possible to the X1/X2 pins

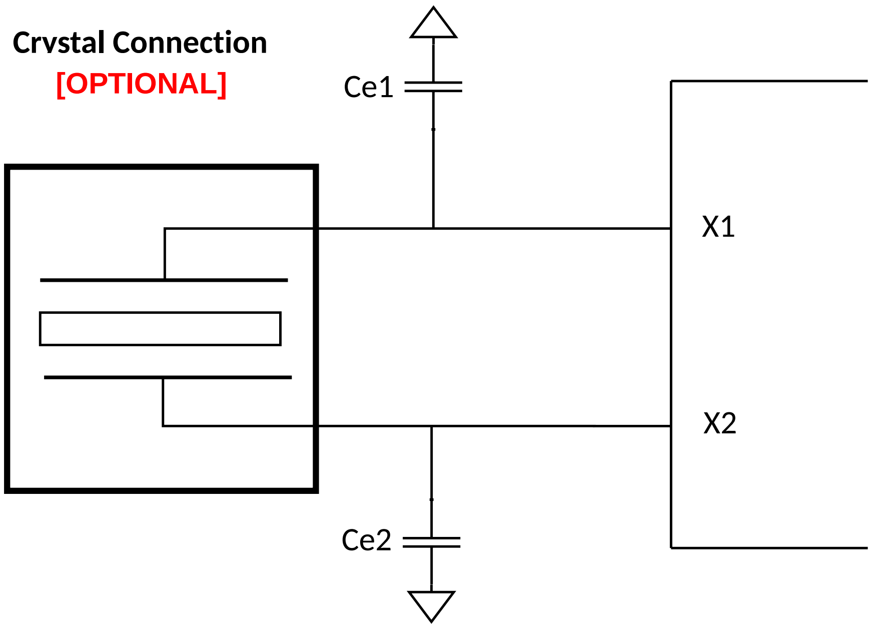


Figure 10. Crystal Connections

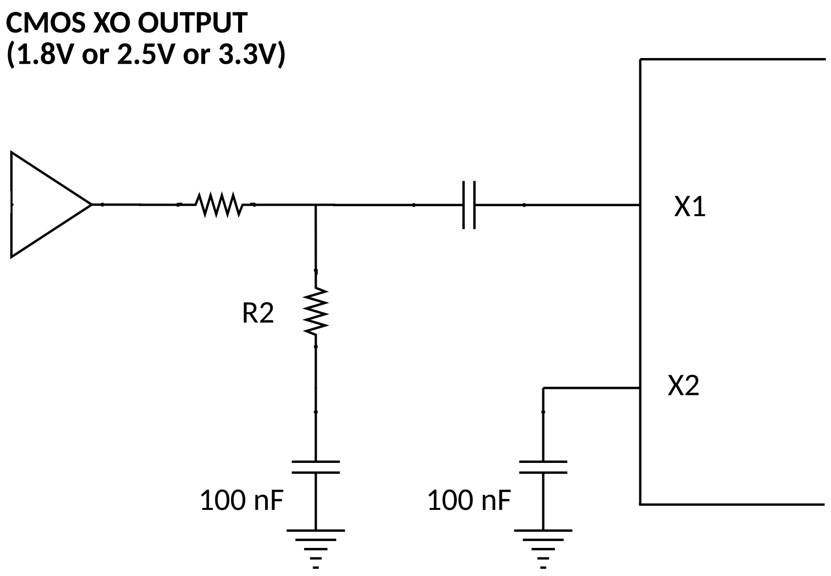


Figure 11. CMOS XO Connections

CMOS XO Driver Supply	R1 (Ohms)	R2 (Ohms)
1.8 V	0	DNP
2.5 V	274	732
3.3 V	453	549

Differential XO/Clock

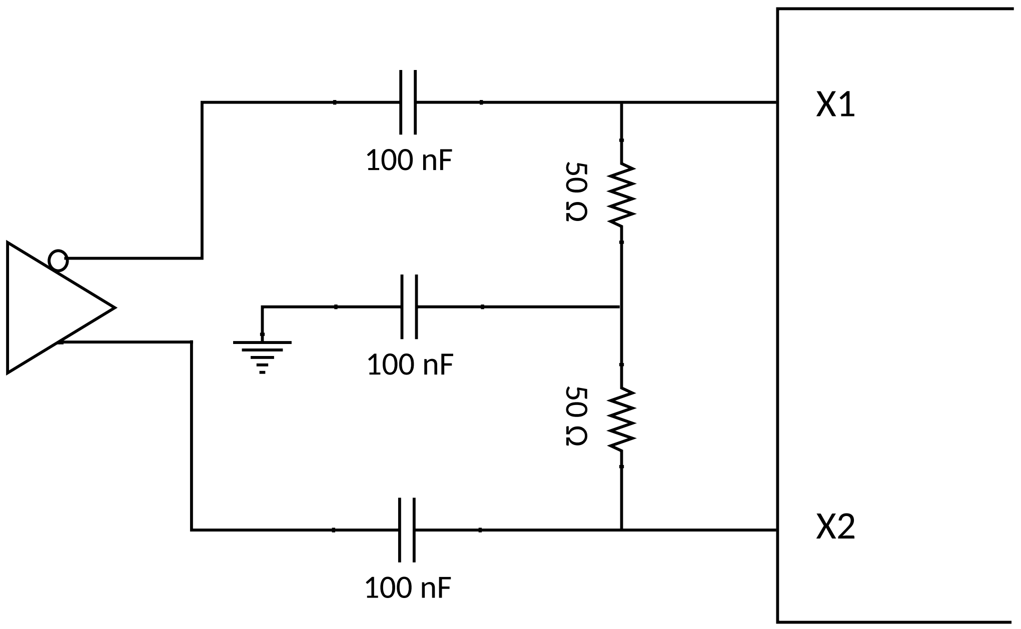


Figure 12. Differential XO Connections

Input Slave Description

Four independent differential clock inputs (and 8 independent single ended clock inputs) are available on the chip that can be routed to PLLs with complete flexibility.

Both single ended and AC coupled differential clock inputs are possible. The input clock receiver settings (to receive a single ended or differential clock) as well as the input clock divider settings are configurable on Page 2 that is assigned to the Input Slave. It is possible to bypass the input clock divider and use the input clock directly as an input to the PLL.

The various input termination interface options available for SiT95314 are shown in this section.

Apart from the above given interface examples, SiT95314 differential input buffer can accept 3.3 V/2.5 V domain universal input clock standards like LVPECL, LVDS, CML and other differential signals that meet input common mode voltage, slew rate and swing requirements specified in Table 3. Differential buffers supports a wide common mode and input signal swing.

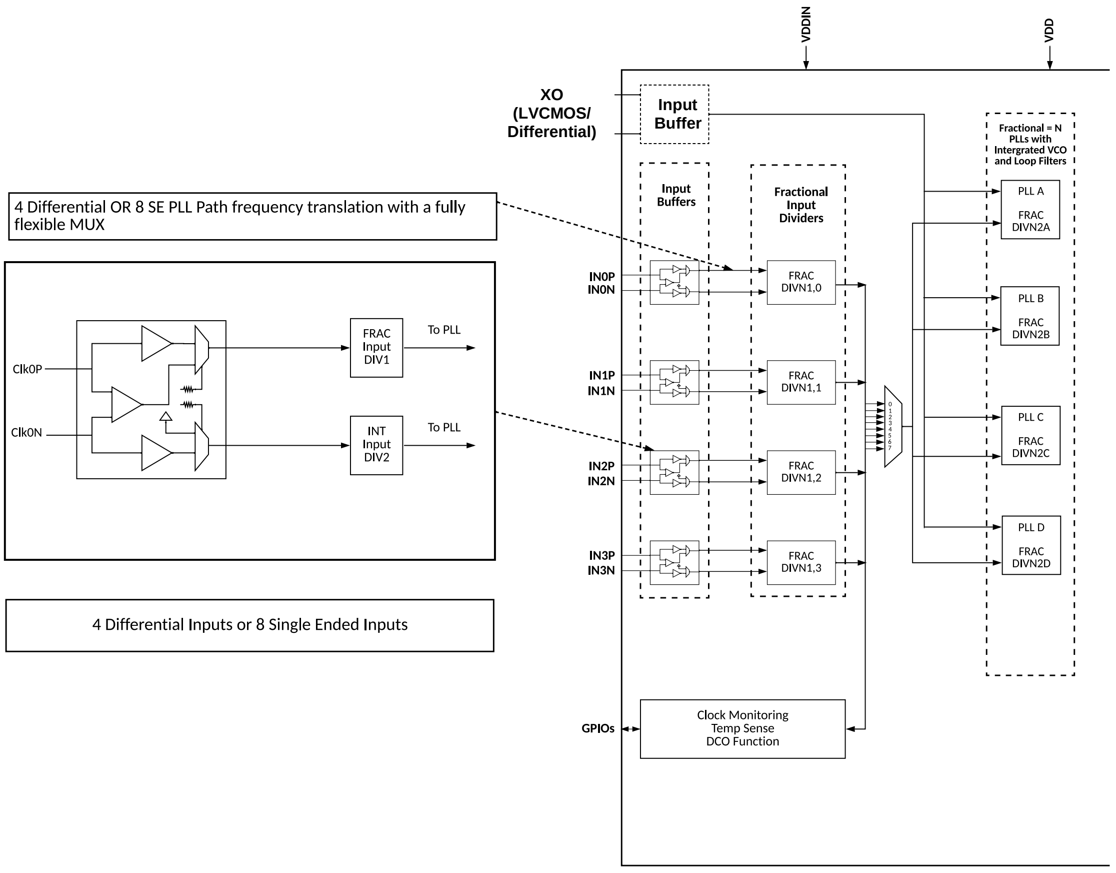

VDDIN
VDD
XO (LVCMOS/ Differential)
Input Buffer
Fractional = N PLLs with Intergrated VCO and Loop Filters
4 Differential OR 8 SE PLL Path frequency translation with a fully flexible MUX
Input Buffers
Fractional Input Dividers
PLL A FRAC DIVN2A
PLL B FRAC DIVN2B
PLL C FRAC DIVN2C
PLL D FRAC DIVN2D
IN0P
IN0N
IN1P
IN1N
IN2P
IN2N
IN3P
IN3N
FRAC DIVN1,0
FRAC DIVN1,1
FRAC DIVN1,2
FRAC DIVN1,3
Clk0P
Clk0N
FRAC Input DIV1
INT Input DIV2
To PLL
To PLL
4 Differential Inputs or 8 Single Ended Inputs
GPIOs
Clock Monitoring Temp Sense DCO Function

Figure 13. SiT95314 Input Slave Architecture

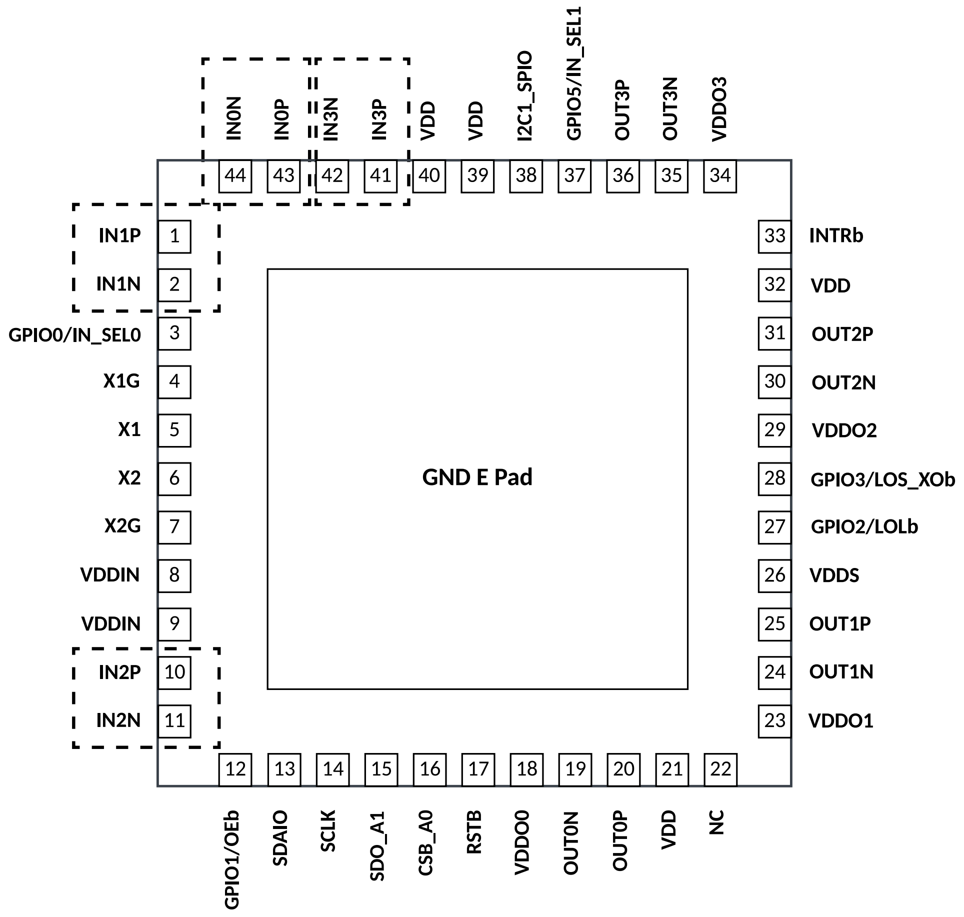


Figure 14. SiT95314 4 Diff/8 SE Input Drivers

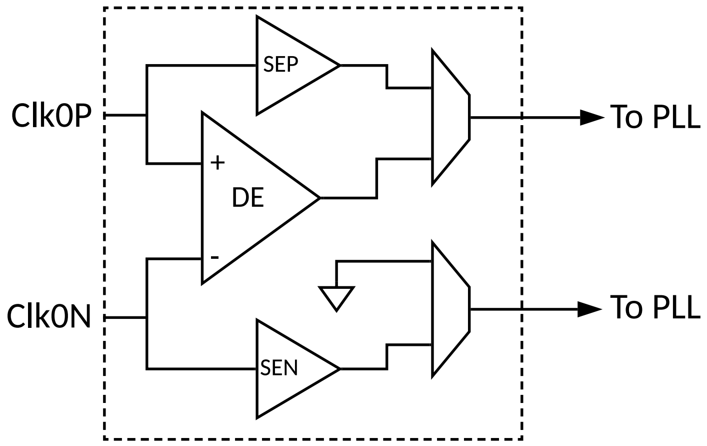
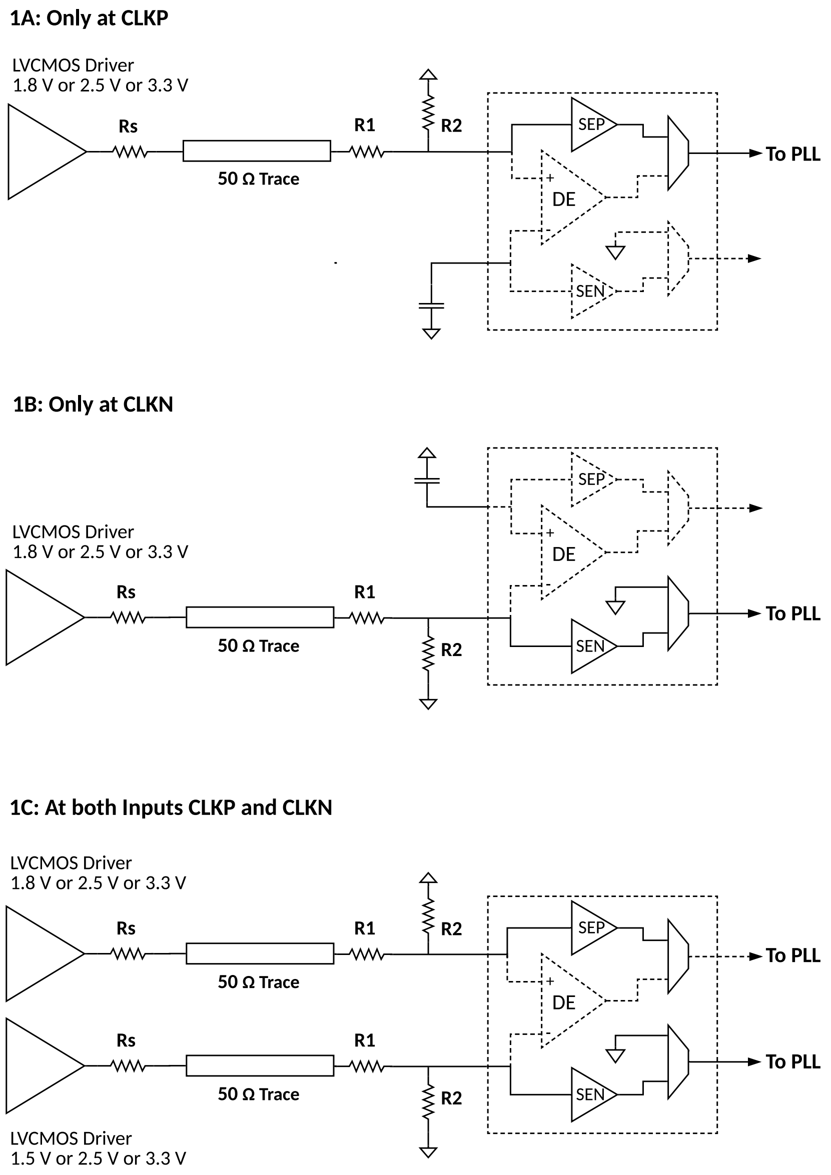


Figure 15. Input Buffer Structure

Note:DE – Differential Buffer; SEP – Single Ended Buffer at CLKP; SEN – Single Ended Buffer at CLKN



Driver Supply	R1 (ohm)	R2 (ohm)
1.8 V	140	665
2.5 V	325	475
3.3 V	445	365
Rs is resistance to match Ro_driver + Rs = 50 ohm		

Figure 16. Interface Type-I: Single Ended DC Coupled

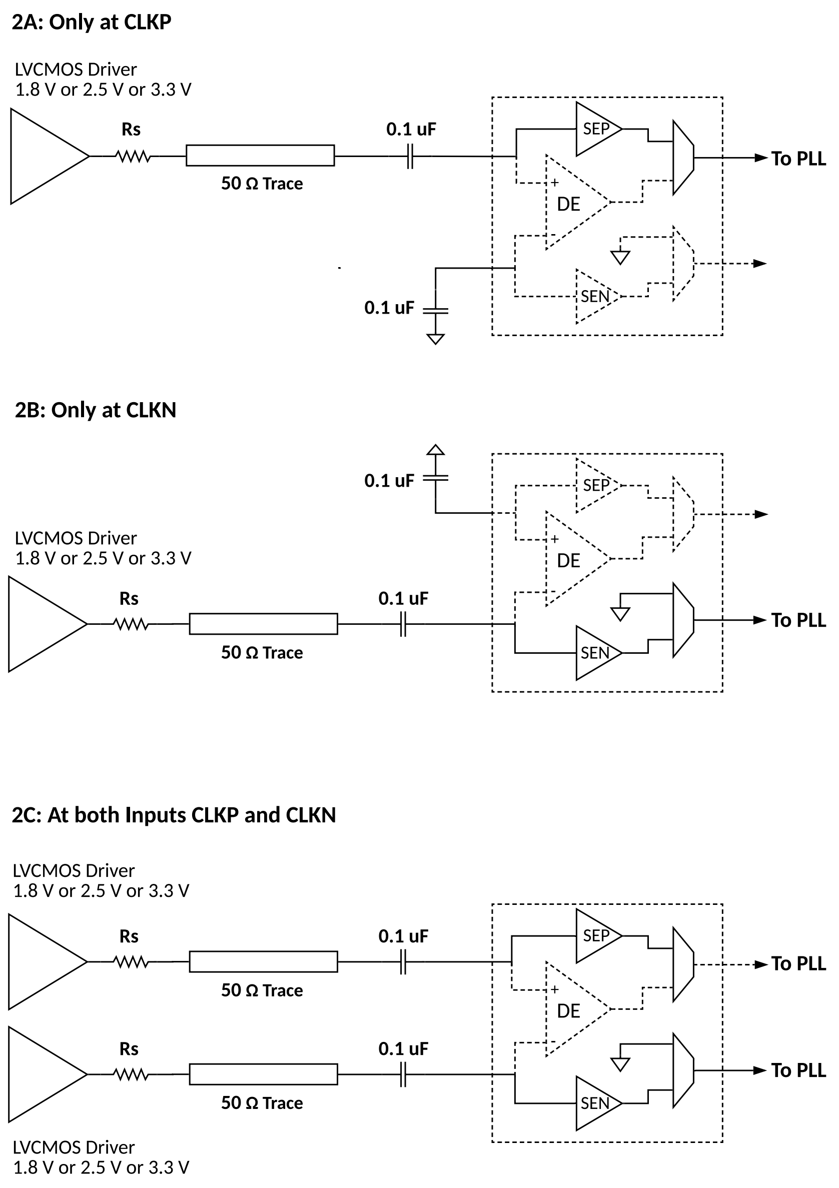
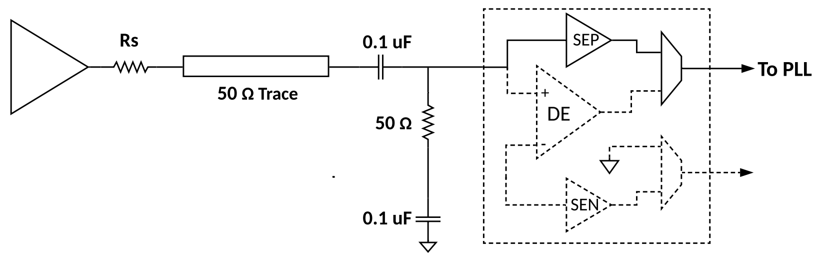


Figure 17. Interface Type II Single Ended Direct AC Coupled

3A: Only at CLKP



3B: Only at CLKN

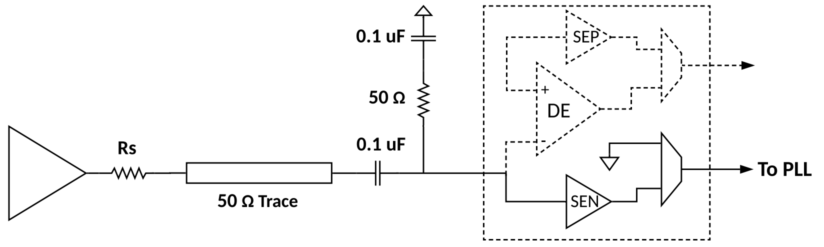


Figure 18. Interface Type III SE AC Coupled with 50-ohm termination

4A

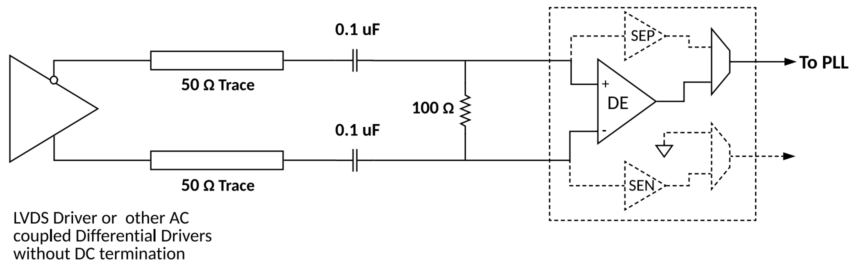

0.1 uF
50 Ω Trace
100 Ω
0.1 uF
50 Ω Trace
SEP
DE
SEN
To PLL
LVDS Driver or other AC coupled Differential Drivers without DC termination

Figure 19. Interface Type IV Differential AC Coupled 4A

4B

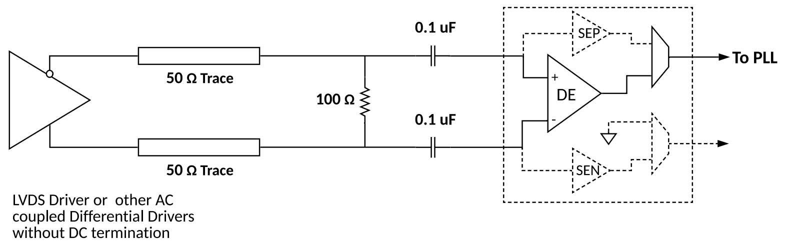

0.1 uF
50 Ω Trace
100 Ω
0.1 uF
50 Ω Trace
SEP
DE
SEN
To PLL
LVDS Driver or other AC coupled Differential Drivers without DC termination

Figure 20. Interface Type IV Differential AC Coupled 4B

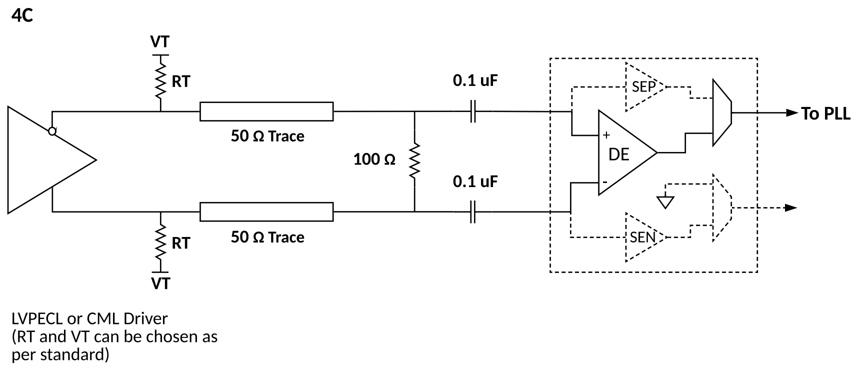


Figure 21. Interface Type IV Differential AC Coupled 4C

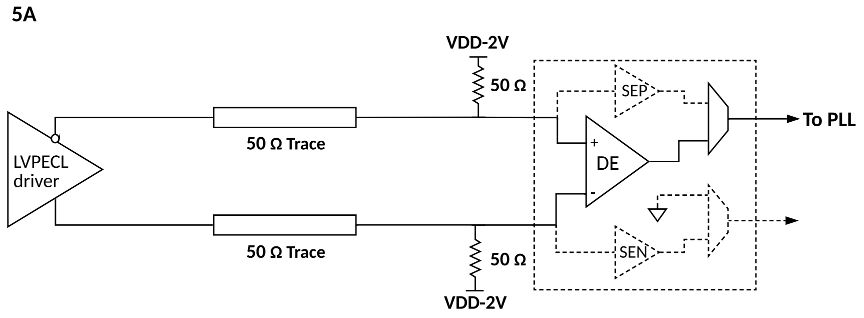


Figure 22. Interface Type V Differential DC Coupled 5A

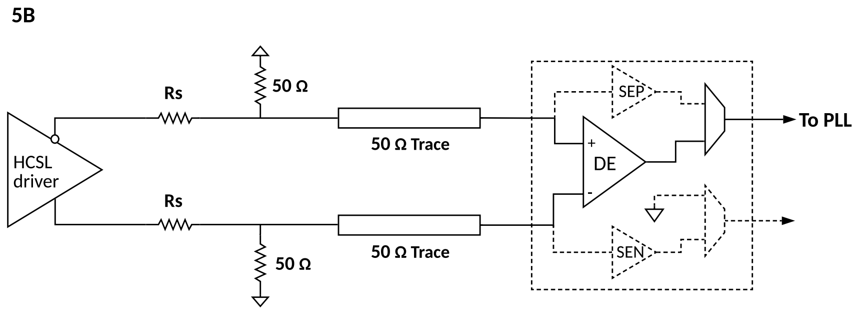


Figure 23. Interface Type V Differential DC Coupled 5B

Clock Monitor Slave Description

Various fault monitoring indicators are available on the chip. The Clock Loss and the Frequency Drift indicators are configurable with the Clock Monitor Slave. The specifications of these fault monitors are indicated in Table 7.

Defect monitoring on any of the clock monitors can be accessed using multiple techniques. The current status of the defect is available as an Active High defect that can be read from the PIF. The "status" is a current indicator of the defect that is high only during the defect (for example during the time that a Clock Loss event is on-going). Additionally, a sticky indicator of the defect called "Notify" can be enabled in the PIF. In this case, the concerned "notify" bit is high the first time the respective defect occurs and stays high till cleared.

There are multiple GPIOs (Flexible IOs) available in the system that can be programmed to monitor individual "notify" signals or a combination of them (as an OR logic). The choice of which fault defect is monitored as an output on the GPIO pin is flexible and can be programmed. Additionally there are selected GPIOs that are hard coded for the information for the clock defects.

Fault Monitoring System

The SiT95314 parts provide an elaborate arrangement of fault monitoring indicators. There are 4 categories of clock monitoring that are necessary for the chip namely: Clock Loss Monitor (CL), Frequency Drift Monitor (FD), Lock Loss Monitor (LL) and XO Clock Loss Monitor (CL_XO).

Clock Loss (CL) monitors loss of input clocks defined as a pre-determined number of consecutive edges missing.

Frequency Drift (FD) monitors frequency drift of a particular clock against a pre-determined Golden Reference.

Lock Loss (LL) monitors the loss of lock in any PLL by monitoring the difference in frequency between the feedback and input clocks.

XO Clock Loss (CL_XO) monitors the loss of the XO reference that is generated from either an external oscillator (XO) or using the on chip XO amplifier that can work with a crystal blank on the PCB.

Each of these categories monitors the health of a particular clock for a certain failure type as illustrated in the name of the clock monitoring category.

For each clock failure observed by the clock monitor block there are two types of indicators provided to the user using the register map:

Live Failure Bit: There is a bit to indicate the live status of a particular failure. [Status]

Sticky Failure Bit: For each live failure bit there is a corresponding sticky bit that is set the first time that corresponding failure is encountered and stays set even if the failure has gone away. Only when the user clears the bit does it clear. [Notify]

The status of these can be either read from the register map or from the pins as a dynamic alarm monitoring arrangement. Additionally, sticky notify registers are available which have sticky status read back from the register map for the various defects. These can be selectively chosen to create an INTRB de-assertion on the INTRB pin as well.

An important point to note is that all of the fault monitoring indicators mentioned above that work with respect to the input clock work on the divided input clock post the DIVN1,k dividers. This implies that the fault monitoring indicators use the frequency fink that is input to the PLL post the DIVN1,k divider translation rather than the external frequencies fin_extk.

Clock Loss Monitors

Each of the inputs are monitored for Clock Loss in terms of missing edges to indicate a loss of input signal. The number of edges used to indicate a clock loss (or recovery from a clock loss) is programmable in the SiT95314 GUI interface allowing for flexibility in choosing these thresholds. In addition there is a programmable "Wait Time" all of which are to be interpreted as follows:

Assertion of Clock Loss-

We declare a CL if "Trigger Edge" number of consecutive edges are missing. The "Trigger Edge" parameter is programmable in the chip GUI.

De-Assertion of Clock Loss-

We declare a ~CL if the clock is back and has less than "Clear Edge" consecutive edges missing. The "Clear Edge" parameter is programmable in the chip GUI.

Wait Time: After the clock is established to have returned, it is ensured that no CL error as defined by the de-assertion threshold occurs for "Val Time" seconds. This valid wait time is programmable using the chip GUI using the "Val Time" parameter which is programmable from the following options: {2m, 128m, 256m, 1, 4 ,32, 64, 128} sec. The use of the this valid wait time ensures that sporadic edges in the input clock (such as ones caused by noise on floating nodes or intermittent unstable clock edges) does not de-assert clock loss and it is established over a user determined period of time that the input clock is available and stable.

Frequency Drift Monitors

Frequency Drift monitors frequency drift of a particular clock against a pre-determined Golden Reference. Any one of the input clocks or the XO clock can be used as the Golden Clock for calculating the frequency drifts of the other clocks. The Golden Clock can be chosen in the GUI and is used as the "0 ppm" Reference Clock for all monitoring.

Fine Frequency Drift has a step size of ±2 ppm.

Fine Frequency Drift has a range of ±2 to ±510 ppm and an independent threshold is programmable for "Set" (for setting the FD monitor) and for "Clear" (for clearing the FD monitor).

Fine Frequency Drift has an implicit hysteresis with resolution of ±2 ppm since the same range is available for the FD assertion and de-assertion. Use of hysteresis prevents unwanted oscillation of the FD monitor output at the decision threshold and is recommended for robust operation. The value of the FD threshold hysteresis is implicit in the choice of the set and clear thresholds.

Assertion of Fine Frequency Drift:

We declare as a Fine FD if Drift in input clock is greater than programmable "set ppm" in the chip GUI.

De-Assertion of Fine Frequency Drift (FD):

Fine FD gets cleared when Drift in input clock over a measurement time is less than programmable "clr ppm" and it should remain less than "clr ppm" threshold until user programmed valid timer times out.

Wait Time: After the clock is established to have returned, it is ensured that no Fine Frequency Drift error as defined by the de-assertion threshold occurs for "Val Time" seconds. This valid wait time is programmable using the chip GUI using the "Val Time" parameter which is programmable from the following options: {2m, 128m, 256m, 1, 4 ,32, 64, 128} sec. The use of the this valid wait timer ensures that input clock is stable and error is less the clr threshold over a user determined period of time.

The Fine Frequency Drift monitors provide precise information for input clock frequency drift. However, since the resolution of the measurement determines time for the measurement- an alternate faster measurement mechanism for drift is needed. This is Coarse Frequency Drift which has coarser measurement but is fast. It is available for cases where the drift is very fast in the input frequency and is programmable from options as shown below.

Coarse Frequency Drift has a step size of ±100 ppm.

Coarse Frequency Drift has a range of ±100 to ±1600 ppm and an independent threshold is programmable for "Set" (for setting the FD monitor) and for "Clear" (for clearing the FD monitor).

Coarse Frequency Drift has an implicit hysteresis with resolution of ±100 ppm since the same range is available for the FD assertion and de-assertion. Use of hysteresis prevents unwanted oscillation of the FD monitor output at the decision threshold and is recommended for robust operation. The value of the FD threshold hysteresis is implicit in the choice of the set and clear thresholds.

Assertion of Coarse Frequency Drift:

We declare as a Coarse FD if Drift in input clock is greater than programmable "set ppm" in the chip GUI.

De-Assertion of Coarse Frequency Drift:

We declare as a ~Coarse FD if Drift in input clock over a measurement time is less than programmable "clr ppm" in the chip GUI. Wait timer is added During this time

Wait Time: After the clock is established to have returned, it is ensured that no Coarse Frequency Drift error as defined by the de-assertion threshold occurs for "Val Time" seconds. This valid wait time is programmable using the chip GUI using the "Val Time" parameter which is programmable from the following options: {2m, 128m, 256m, 1, 4 ,32, 64, 128} sec. The use of the this valid wait timer ensures that input clock is stable and error is less the clr threshold over a user determined period of time.

Important Note regarding the above monitors with respect to clock switch in the PLL:

Normally the CL monitor is used for ascertaining a clock is lost for the PLL to switch to a secondary reference or proceed to Holdover. However, the Fine and/or Coarse FD monitors can also be used in addition to the CL monitor to cause a PLL switch. This implements an "OR" logic for the FD Monitors to be used in addition to the CL monitors for triggering a PLL input clock switch or entry to Holdover. This is programmable as an option in the GUI.

Lock Loss Monitors

Lock loss is programmable for each PLL with lock loss triggered if the frequency of the input reference to the PLL phase detection arrangement and the feedback clock to same PLL are different as per the programmed assertion and de-assertion thresholds.

The Set threshold for asserting the LL monitor is programmable from {±0.05ppb, ±0.1ppb, ±0.5ppb, ±1ppb, ±0.2ppm, ±0.4ppm, ±2ppm, ±4ppm, ±20ppm, ±40ppm, ±200ppm, ±400ppm, ±2000ppm, ±4000ppm} while the Clear threshold for de-asserting the LL monitor is programmable from {±0.05ppb, ±0.1ppb, ±0.5ppb, ±1ppb, ±0.2ppm, ±0.4ppm, ±2ppm, ±4ppm, ±20ppm, ±40ppm, ±200ppm, ±400ppm, ±2000ppm, ±4000ppm}. A pre-determined level of hysteresis is implicit by choosing appropriately the set and clear thresholds for the LL monitor.

Additionally from the point of view of LL de-assertion, there is a delay from the point in time that lower than the specified ppm value is achieved to the point where the actual LL is de-asserted to the user such that LL never asserts during this delay period. The choice of this delay is with a timer that ensures that the delay is in line with the BW of the PLL loop. It is fully programmable from the GUI and is useful to ensure complete settling of the PLL without un-necessary toggling before LL de-assertion.

XO Clock Loss Monitors

The XO Clock Loss Monitor asserts the XO Clock Loss Alarm when the external reference input to the X1 pin (XO) or the internal XO clock generated with the crystal blank is not available.

PLL Slave Description

All settings with respect to each PLLx slave (*x* ϵ {A, B, C, D}) are accessible on the respective Page {A, B, C, D}. The PLL architecture is shown in Figure 24. There are three distinct modes of operation of the PLL: free run mode, synchronized mode and holdover mode. The frequency of the high frequency VCO in the PLL is determined by the specific mode of operation. The VCO frequency is then divided down to get the output frequency on the ODR.

The PLL in the free run mode can be described as a input reference based oscillator where the output frequency is determined by the relation fVCOx = DIVNx*fref. This is the mode of operation before the loop is locked to the selected input clock or the mode of operation for the case none of the input clocks is available. After locking to the chosen input clock, the PLL enters the synchronized mode of operation where the output is now locked to the input frequency with the relation fVCOx = DIVN2x*finx. The PLL Loop that synchronizes (locks) the output to the input clock has a programmable loop bandwidth between 0.09 mHz to 4 KHz and is not affected by static or dynamic drifts in the input reference clock based fref frequency. In case the input clock is lost, the PLL locks to the highest priority spare clock available. If all specified input clocks are lost, the PLL remembers the correction based on historical average of the input clock as specified to enter the Holdover mode of operation.

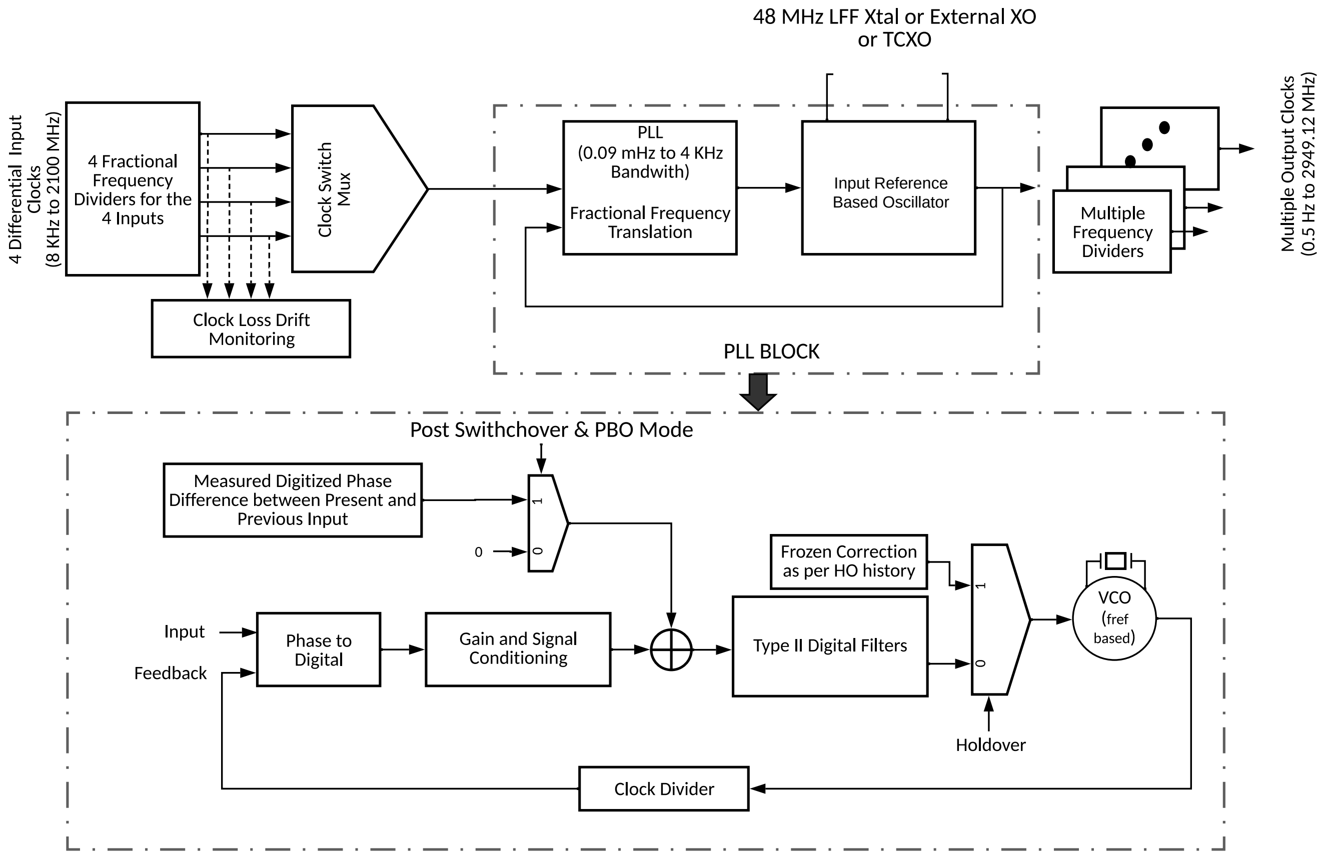

48 MHz LFF Xtal or External XO or TCXO
4 Differential Input Clocks (8 KHz to 2100 MHz)
4 Fractional Frequency Dividers for the 4 Inputs
Clock Switch Mux
Clock Loss Drift Monitoring
PLL (0.09 mHz to 4 KHz Bandwith)
Fractional Frequency Translation
Input Reference Based Oscillator
PLL BLOCK
Multiple Frequency Dividers
Multiple Output Clocks (0.5 Hz to 2949.12 MHz)
Post Swithchover & PBO Mode
Measured Digitized Phase Difference between Present and Previous Input
1
0
0
Input
Feedback
Phase to Digital
Gain and Signal Conditioning
Frozen Correction as per HO history
Type II Digital Filters
1
0
Holdover
VCO (fref based)
Clock Divider

Figure 24. PLL Architecture

The Digital Low Pass Filter (DLPF) receives the phase information from the Phase-to-Digital block, does gain correction and filtering and applies a correction to the Digitally Controlled Oscillator (DCO). The block also does holdover of past correction during clock switching and phase correction after clock switching. There are hooks available to perform external filtering, phase and frequency correction and incremental changes to the fraction applied to the fractional divider. The basic block diagram of the DLPF is shown in Figure 25. The DCO here refers to the input referenced clock loop shown as the “Input Reference (fref) Based Oscillator” in the diagram describing the PLL Internals earlier in this document.

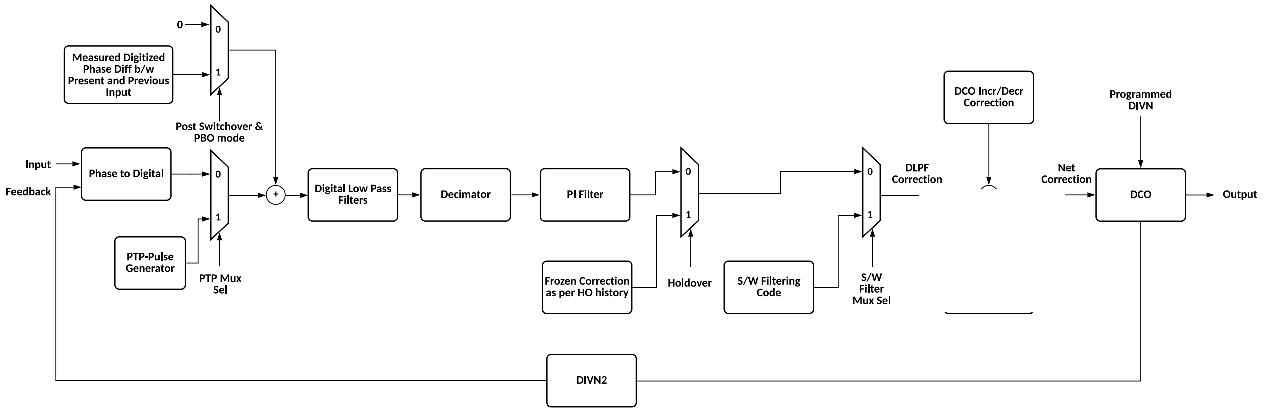


Figure 25. Basic DLPF Block Diagram

Mode of Operations

Basic Mode of Operation

In this mode, the digitized phase information is conditioned by the low-pass filters, decimated and passed through the PI-path filters. Gain correction of the digital word is also done. The conditioned signal is applied as a correction to the programmed division control word (DIVN). This main path is the highlighted in the Figure 26.

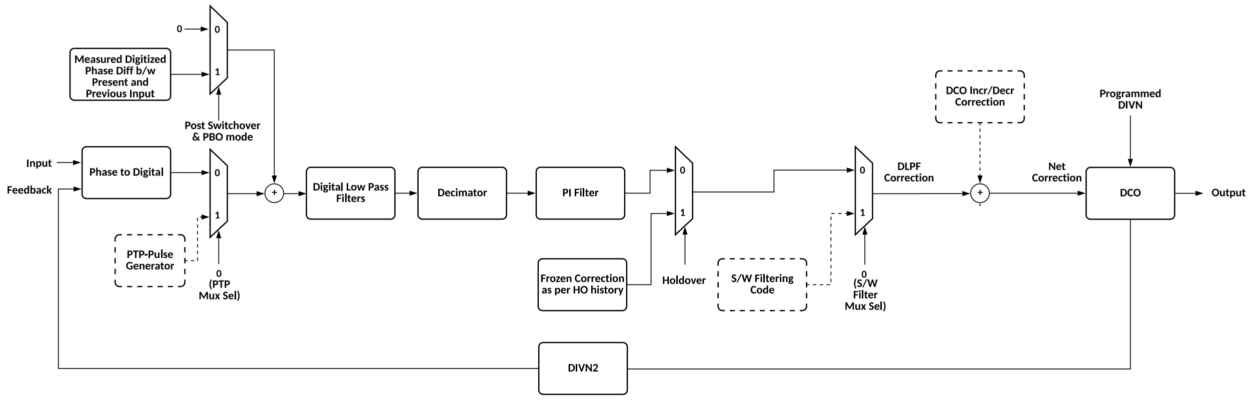


Figure 26. Basic Mode of Operation

Holdover and Phase Build-Out (PBO) modes

During normal mode of operation, a holdover control mode maintains a history of the recent correction applied by the filter. When a clock loss happens, DLPF enters the holdover state. In this mode, the last correction from holdover history is applied at the DLPF output, until another clock is available for switchover. When a new clock is available, DLPF exits the holdover state. If PBO mode is enabled, then DLPF calculates the phase difference between the previous and present inputs and applies this as an offset before exiting the holdover state

.

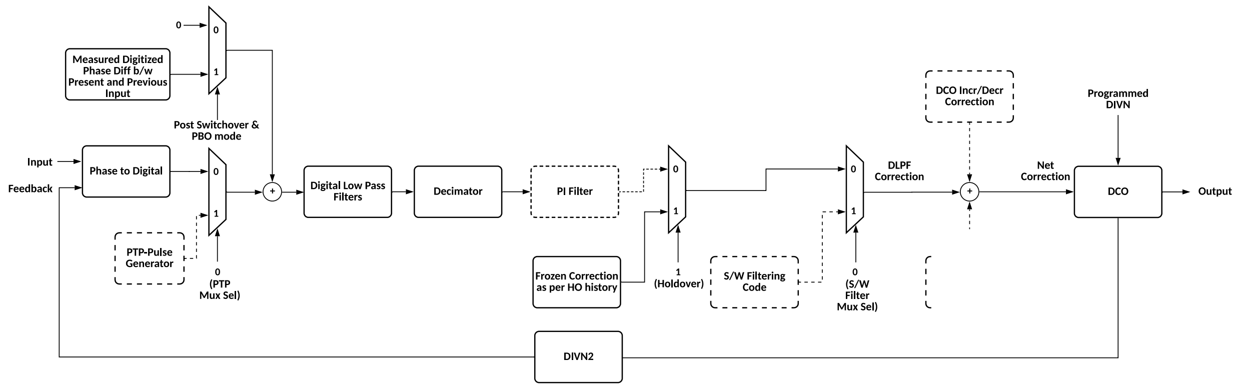


Figure 27. Holdover and Phase Build-Out Modes

Holdover and Phase Propagation (PPG) mode

Phase Propagation mode is used for the case of two input clocks at the same frequency at the PLL input with any phase relationship between them such that the input phase difference is propagated to the output with a programmable slope or PLL bandwidth.

This mode is selected from the GUI with the slope setting given options in GUI within the range of us/s to ns/s settings.

Frequency Ramp Mode

For the input clocks to the PLL that are not exactly the same frequency(plesiochronous clock) but only nominally the same frequency, the frequency ramp feature can be enabled from the GUI, In this case the output frequency will ramp such that is started tracking the new switched input reference with controlled slope ramp rate(ppm/sec) which can also be programmed in GUI.

For the case where the input clocks are only nominally the same but plesiochronous in nature, the frequency ramp feature can be used to control the rate of change of the output frequency

Clock Sel modes

In sync mode the clock on which a PLL is locked to is said to be an active clock. This can be chosen in auto select (can be programmed via GUI) or manual select mode.

In auto select, clock switching happens automatically depending on the clock's availability, At initialization PLL is programmed to use one as the Active clock and the others as Spare clocks.

In manual select it happens on the request from user, controllable via Pin/Registers if GPIO is selected else PLL is always locked to Active clock and stay in holdover in case active clock lost .

There are two mode in Manual active select case when GPIO is selected

<u>Direct Clock Selection:</u> In this user controls the clock for the PLL's set in Manual Select Mode simultaneously via GPIO's or register bits. Direct clock information is given as an input.

<u>Indirect Clock Selection:</u> In this user programs the two clocks for PLLs in registers and then switch between those two via GPIO or register bits

DCO Increment/Decrement Mode

This mode is useful to steer the clock frequency up or down, in small increments or decrements. The DCO increment/decrement path for the inner loop is highlighted in the figure below. A similar path also exists for the outer-loop and will be explained later.

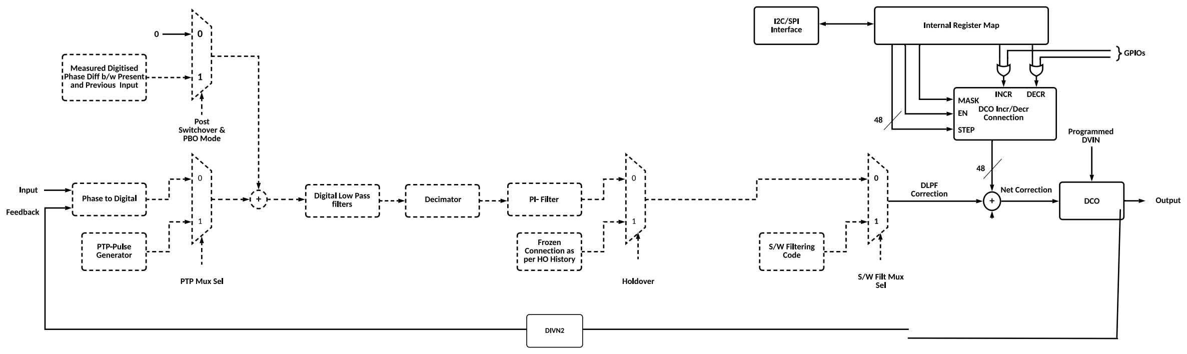


Figure 28. DCO Increment/Decrement path for inner-loop

Inner-Loop DCO increment/decrement path architecture

To provide more insight into the increment/decrement path of the inner-loop, a detailed architecture is presented in Figure 29.

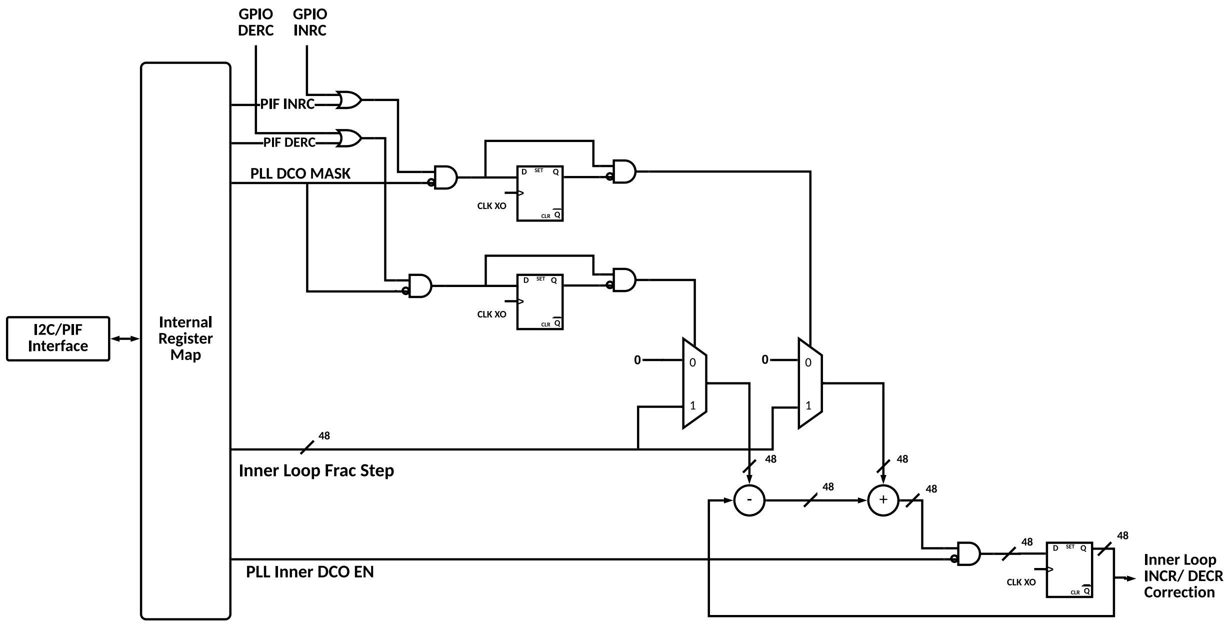


Figure 29. Architecture of the inner-loop increment/decrement path

To change the frequency in steps, a step size has to be programmed. The step size is specified as a 48-bit fraction in the range [-0.5, 0.5). Increments and decrements can be done by applying a pulse either via GPIOs or via the register map. Since the same set of IOs control all PLLs, the DCO mask bit of the PLL of interest whose frequency need to be changed must be written to 0. And the DCO enable bit must be written to 1. The DCO increment/decrement through register bits from the serial interface is unique to each PLL.

With all these settings programmed, increment/decrement pulse can applied. A 0 -> 1 transition of the increment (decrement) causes the fraction step (negative of fraction step) to be accumulated and applied as a correction. The process can be repeated any number of times to obtain the desired net correction. Once this is done, the PLL can be masked and the same process can be repeated for another PLL, if required. The accumulator will continue to remember its state until the DCO enable is cleared.

Outer-Loop DCO increment/decrement path architecture

The outer-loop increment/decrement path has a similar architecture to that of the inner-loop path. While the PLL DCO mask bits are shared between the inner- and outer-loop paths, the paths have dedicated DCO enable bits. In addition to a 48-bit fraction step, outer-loop DCO also has a 24-bit integer step, in order to perform corrections of larger magnitude. Since the correction steps of the inner- and outer-loop DCOs are mapped to different sets of registers in the register map, both the inner- and outer-loop increment/decrement paths can be operated simultaneously.

Zero Delay Mode

There are two independent Zero Delay Mode/Buffer (ZDM/B) are supported on each PLL i.e. External and Internal. Each PLL can be configured for Internal ZDB for very low Input to output delay variation across temperature.

External ZDB is also supported on each PLL for better Input to output delay precision. Feedback option is fully flexible and user selectable. Any output can be fed to any input clock port as per board design and application requirment.

This provides the option to close the feedback loop of the PLL on the PCB and therefore bypasses the internal feedback dividers cancelling therefore the delays introduced by internal dividers and clock distribution pathways. The terminations used for feedback clock would depend on the driver type chosen- the preferred option is to use an LVDS or LVDS boost output ac coupled into a differential 100 Ω termination at the input side.

Output Slave Description

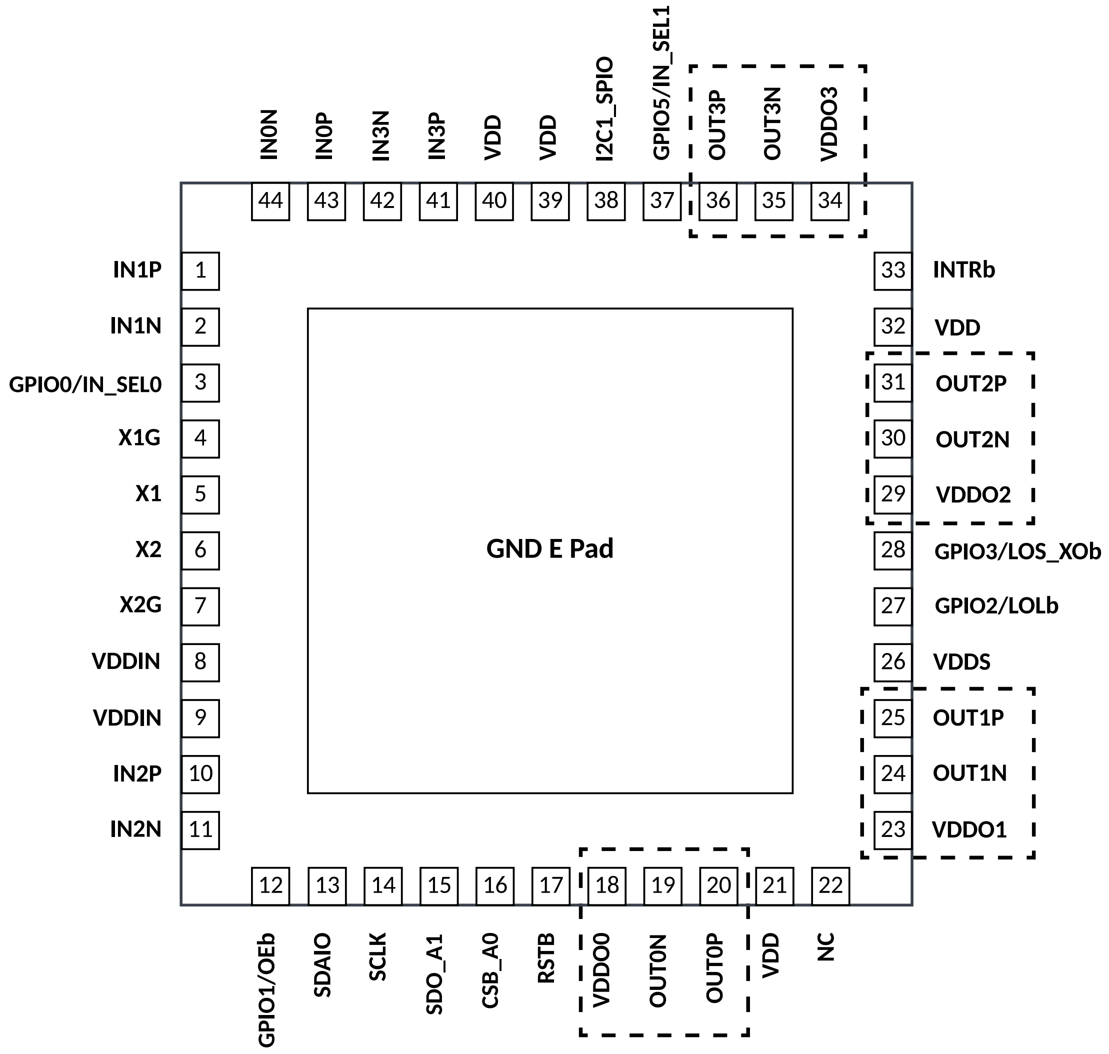


Figure 30. SiT95314 Output Drivers

- Several Terminations are available for the SiT95314 parts Output Driver which serve several industry standards.
- SiT95314 parts offer programmable output swing for the various termination arrangements.
- This section summarises the various termination standards that are supported and the associated programmable swings. The outputs are in Hi-Z state after the chip reset and before the chip configuration. Figure 31 describes the interpretation used for swings.

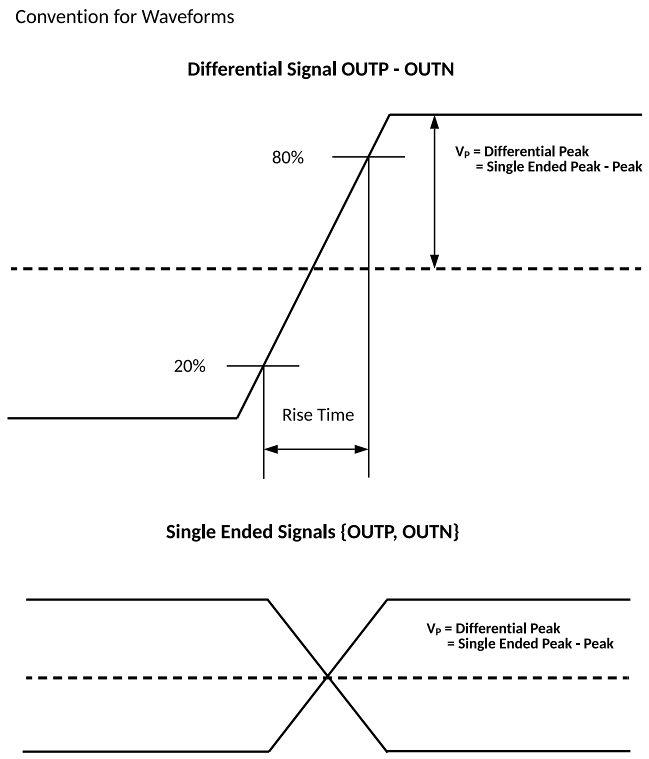

Convention for Waveforms
Differential Signal OUTP - OUTN
80%
V_P = Differential Peak
= Single Ended Peak - Peak
20%
Rise Time
Single Ended Signals {OUTP, OUTN}
V_P = Differential Peak
= Single Ended Peak - Peak

Figure 31. Waveform Convention

Regular High Swing modes

Differential Output Drivers: Regular High Swing Modes

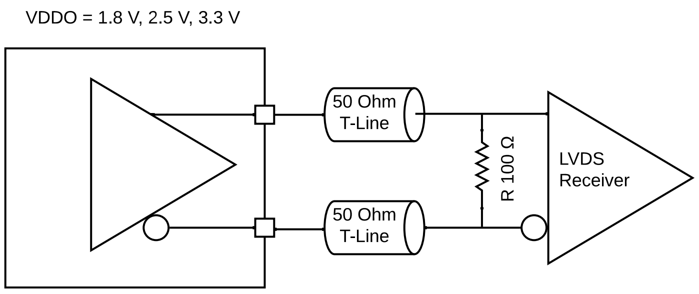


Figure 32. Output Driver used with traditional DC Coupled LVDS receiver

Supported Swing:
Differential Peak Swing → 0.1V to 0.8V
Differential Peak Swing Step Size → 100mV

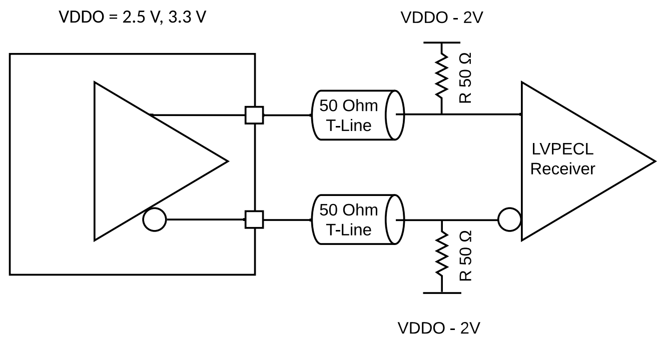


Figure 33. Output Driver used with traditional LVPECL DC Coupled receiver(DC Termination to VDDO - 2V)

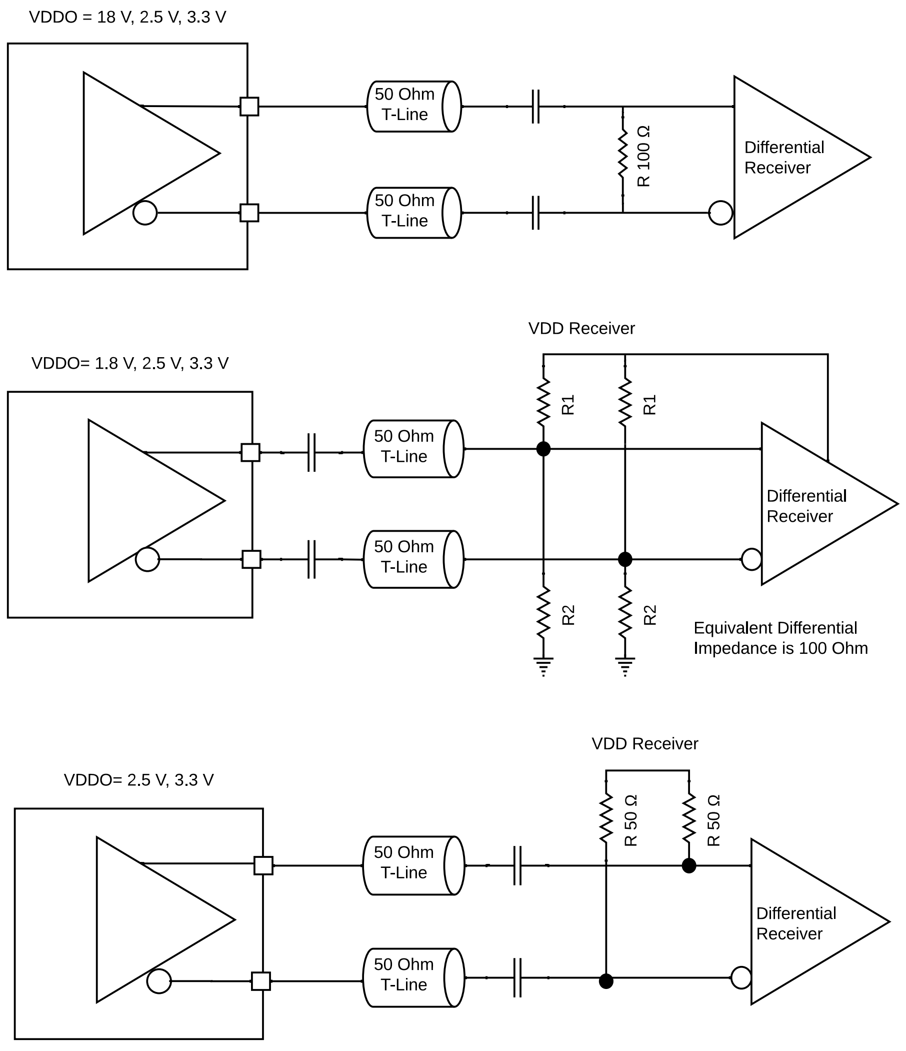


Figure 34. Output Driver used with AC Coupled terminations for various receivers

Supported Swing:	
Differential Peak Swing	→ 0.1V to 0.8V
Differential Peak Swing Step Size	→ 100mV

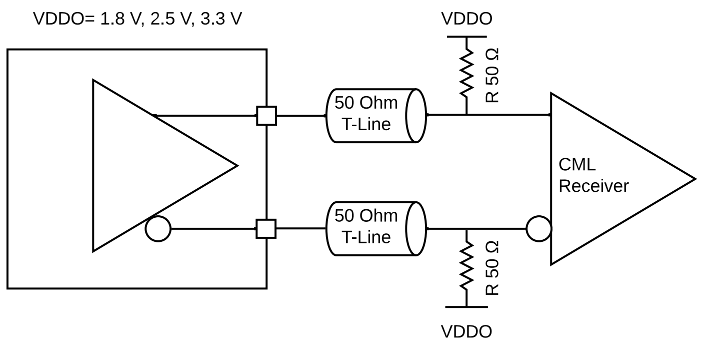
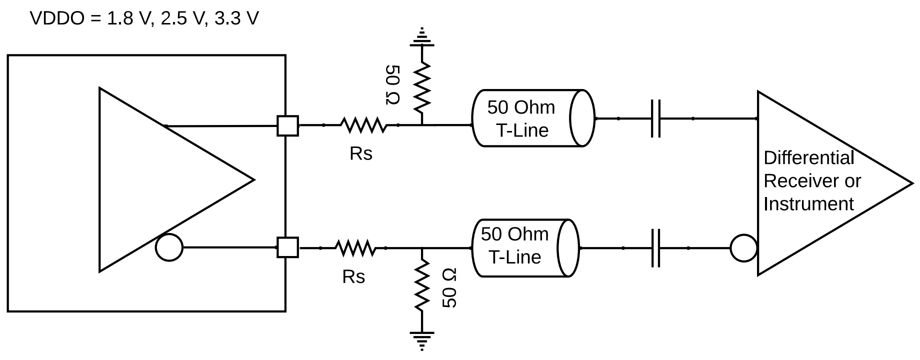


Figure 35. Output Driver used with traditional DC Coupled CML receiver

Supported Swing:	
Differential Peak Swing	→ 0.05V to 0.4V
Differential Peak Swing Step Size	→ 50mV


VDDO = 1.8 V, 2.5 V, 3.3 V
50 Ω
Rs
50 Ohm T-Line
50 Ohm T-Line
Differential Receiver or Instrument
Rs
50 Ω

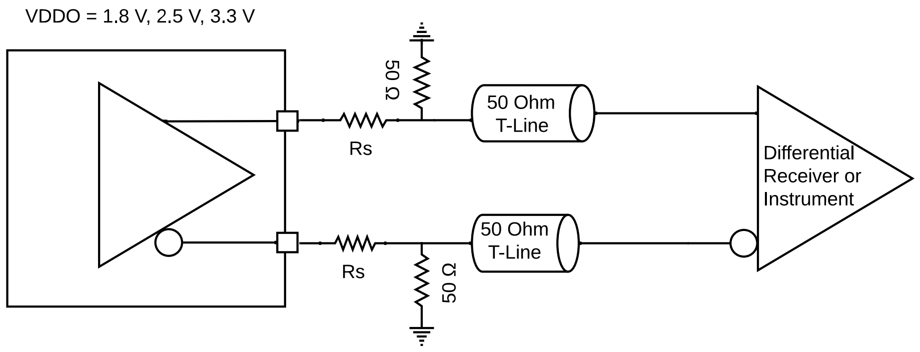

VDDO = 1.8 V, 2.5 V, 3.3 V
50 Ω
Rs
50 Ohm T-Line
50 Ohm T-Line
Differential Receiver or Instrument
Rs
50 Ω

Figure 36. Output Driver used with traditional HCSL receiver

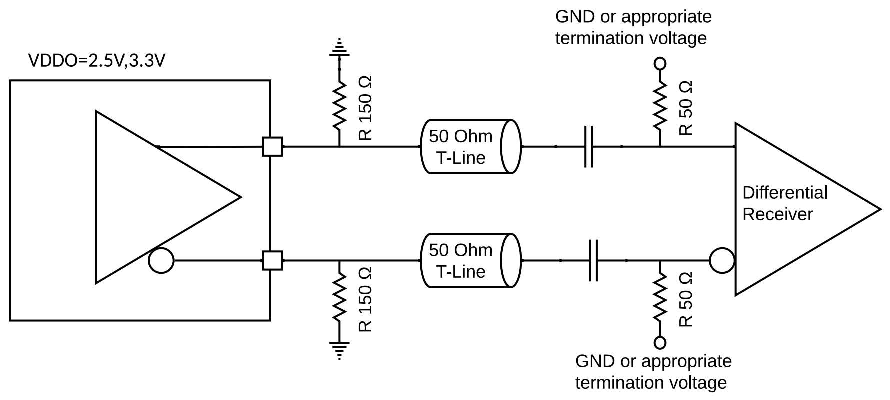


Figure 37. Alternate AC Coupled HCSL with DC Coupled resistors on Chip side

Supported Swing:	
Differential Peak Swing	→ 0.5V to 0.8V
Differential Peak Swing Step Size	→ 50mV

Internal Termination modes

- SiT95314 support differential 100Ω internal termination in LVDS mode.
- If the receiver is correctly terminated, internal termination is not recommended
- The internal termination is recommended for cases where transmission line integrity or far end termination is not robust.

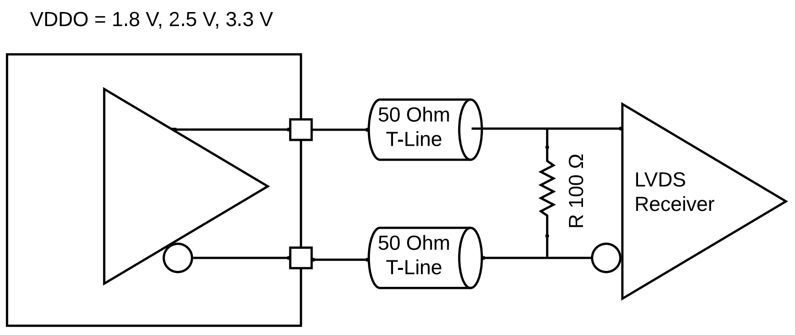


Figure 38 Output Driver used with traditional DC Coupled LVDS receiver

Supported Swing with far end termination:	
Differential Peak Swing	→ 0.05V to 0.4V
Differential Peak Swing Step Size	→ 50mV

Supported Swing without far end termination:	
Differential Peak Swing	→ 0.1V to 0.8V
Differential Peak Swing Step Size	→ 100mV

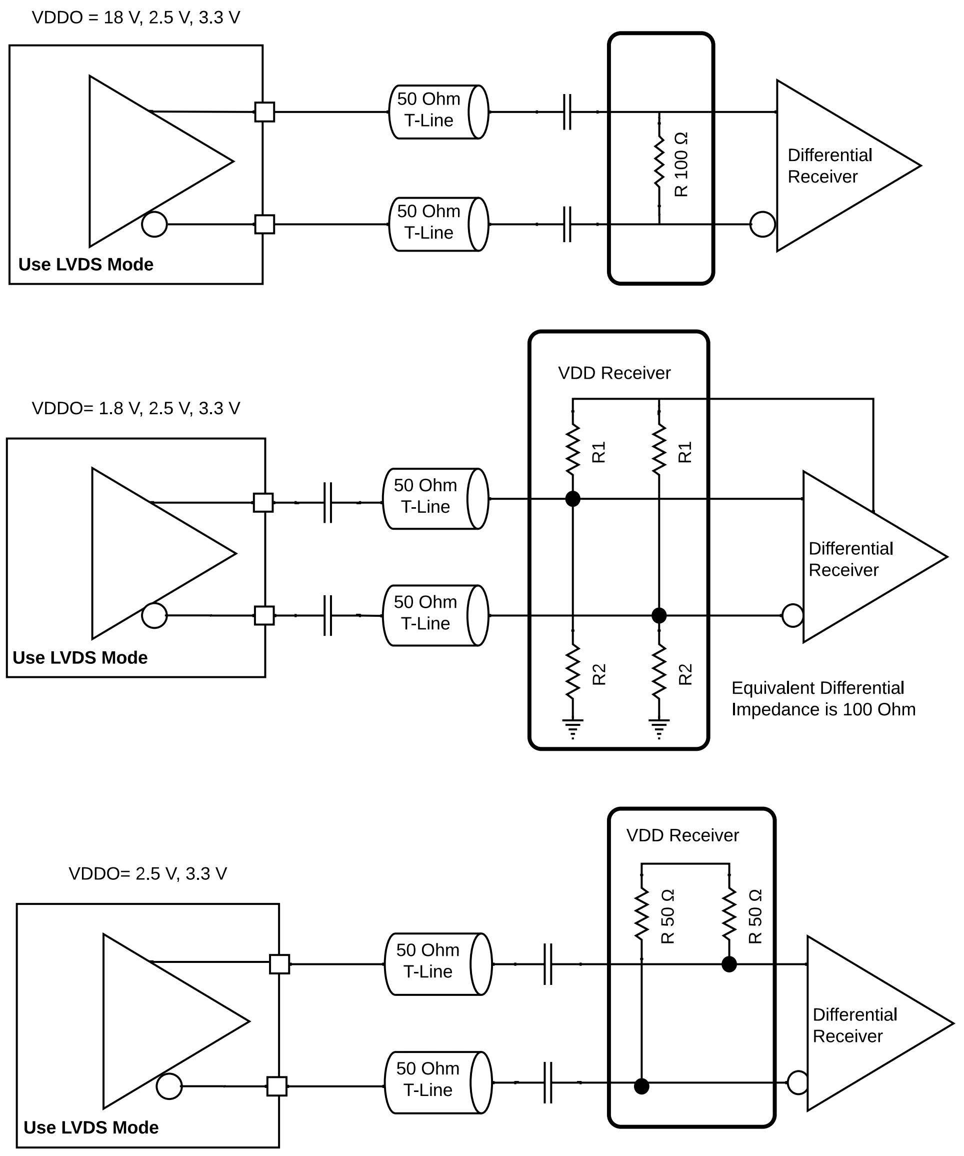


Figure 39. Output Driver used with AC Coupled terminations for various receivers

Supported Swing with far end termination:	
Differential Peak Swing	→ 0.05V to 0.4V
Differential Peak Swing Step Size	→ 50mV

Supported Swing without far end termination:	
Differential Peak Swing	**→ 0.1V to 0.8V**
Differential Peak Swing Step Size	**→ 100mV**

Output Clock Modes

SiT95314 can provide 4 differential clocks or 8 pair of single ended clocks. These output clocks can be programmed differently to serve different applications at system level. This document explains output clock behavior in different functional modes of operations.

Functional Modes

SYSREF mode (JESD204B)

SYSREF mode with Pulser

SYNCB mode

SYSREF Mode

For JESD204B application, SiT95314 can provide up to 2 pairs of DEVICE clock and SYSREF clock. We can choose which output to be programmed as SYSREF clock independently. SYSREF clock can be triggered through 2 ways which are below.

Sysref trigger using PIN

Sysref trigger using REG WRITE (SPI OR I2C operation)

In sysref mode, there is an internal clock which is pre-aligned with known phase offset with respect to device clock. Using sysref_trigger signal, this clock is gated inside in a glitch free manner. Sysref clock (Figure 40) only appears after 2 negative edges after sysref_trigger is asserted high. Similarly, sysref_clock_out is disabled after 2 negative edges of internal clock when sysref_trigger is de-asserted to low.

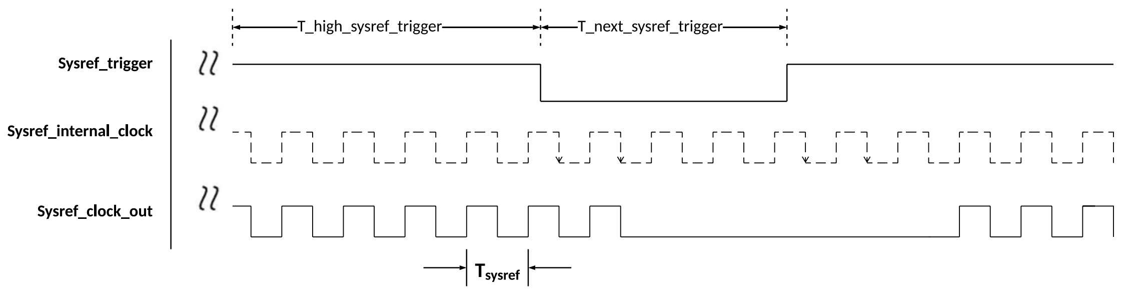


Figure 40. sysref_clock_out behavior with respect to sysref_trigger

For SYSREF to work as shown above, following conditions must be true

$$T\_high\_sysref\_trigger \geq 2 * T_{sysref}$$

$$T\_next\_sysref\_trigger \geq 2 * T_{sysref}$$

SYSREF Mode with Pulser

In few JESD204B applications, sysref clocks are preferred as defined number of pulses rather than a continuous clock. This is done to reduce cross-talk between SYSREF and DEVICE clock. In SiT95314, sysref clocks can be programmed in pulser mode of operation too. There are 8-bits independent to each sysref clock to program number of output pulses from 1 to 255. This mode can also be triggered through below 2 methods.

1. Sysref trigger using PIN
2. Sysref trigger using REG WRITE (SPI OR I2C operation)

In Figure 41, an example of sysref clock in pulser mode of operation is shown. In this example, sysref clocks are programmed for 3 pulses.

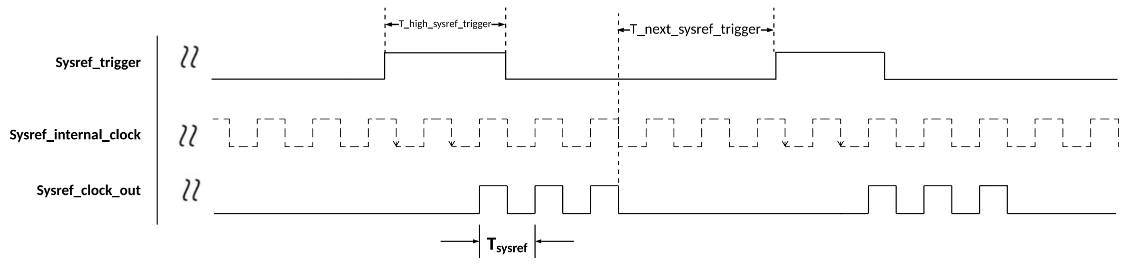


Figure 41. sysref_clock_out behaviour with respect to sysref_trigger in pulser mode

For SYSREF to work in *pulser mode*, following conditions must be true

$$T\_high\_sysref\_trigger \geq 2 * T_{sysref}$$

$$T\_next\_sysref\_trigger \geq T_{sysref}$$

SYNCB Mode

In SYNCB mode oxf operation, any output clock can be independently programmed to react to syncb_trigger signal. In this mode, syncb selected output clock(s) will appear after certain delay which can be programmed with a resolution of 1 clock period of VCO. This mode can also be triggered through below 2 methods.

1. Syncb trigger using PIN
2. Syncb trigger using REG WRITE (SPI OR I2C operation)

In Figure 42 below, an example of syncb trigger is shown. syncb_clock_out appears after T_syncb_delay of time delay when syncb_trigger is asserted high and it is stopped in glitch-free manner after 2 negative edges when syncb_trigger is de-asserted low.

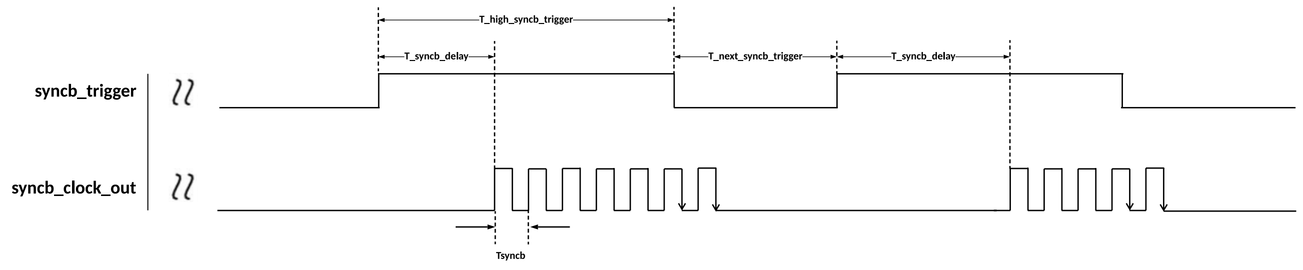


Figure 42 syncb_clock_out behavior with respect to syncb_trigger

Definition:

$\boldsymbol{T_{syncb}}$ is one time period of syncb_clock_out

$\boldsymbol{T_{\_syncb\_delay}}$ is user programmed delay. First edge of syncb_clock_out will appear after this delay when *syncb_trigger* is asserted. It can be programmed with a resolution of 1 TVCO.

$\boldsymbol{T_{\_high\_syncb\_trigger}}$ is minimum duration of *syncb_trigger* staying high for reliable operation as per above definition.

$\boldsymbol{T_{\_next\_syncb\_trigger}}$ is minimum duration of *syncb_trigger* staying low for reliable operation as per above definition and it is directly represented as minimum time duration after which next *syncb_trigger* is asserted)

For SYNCB to work, following conditions must be true

$$T\_high\_syncb\_trigger \geq T\_syncb\_delay + (2 * T_{syncb})$$

$$T\_next\_syncb\_trigger \geq 2 * T_{syncb}$$

Serial Programming Interface Description

The SiT95314 device has a serial programming interface. The serial programming interface supports I2C(Master) protocol to configure the device from the external EEPROM, and I2C(slave) and SPI(slave) serial interface protocols, for reconfiguring the device settings using register read/write.

Serial Interface Pins

Following pins of the device are used as a Serial Interface. These pins are configured for different functionality in different modes.

- SCLK
- SDIO
- SDO
- CSB

Serial Programming Interface Description

I2C protocol

I2C Master

SiT95314 can be configured from external I2C EEPROM. In wakeup sequence master checks for the EEPROM access, if a GPIO is latched as 1 and decoded as EEPROM access disable, and serial interface is configured to I2C mode, SiT95314 tries to read the configuration block from EEPROM.

To read the configuration data from the external I2C EEPROM, this device becomes the I2C Master and initiates the I2C transaction as shown in Figure 43. Serial Interface Pin configuration in I2C master mode is as follows.

- SCLK (SCL)
- SDIO (SDA)
- SDO (A_1)
- CSB (A_0)

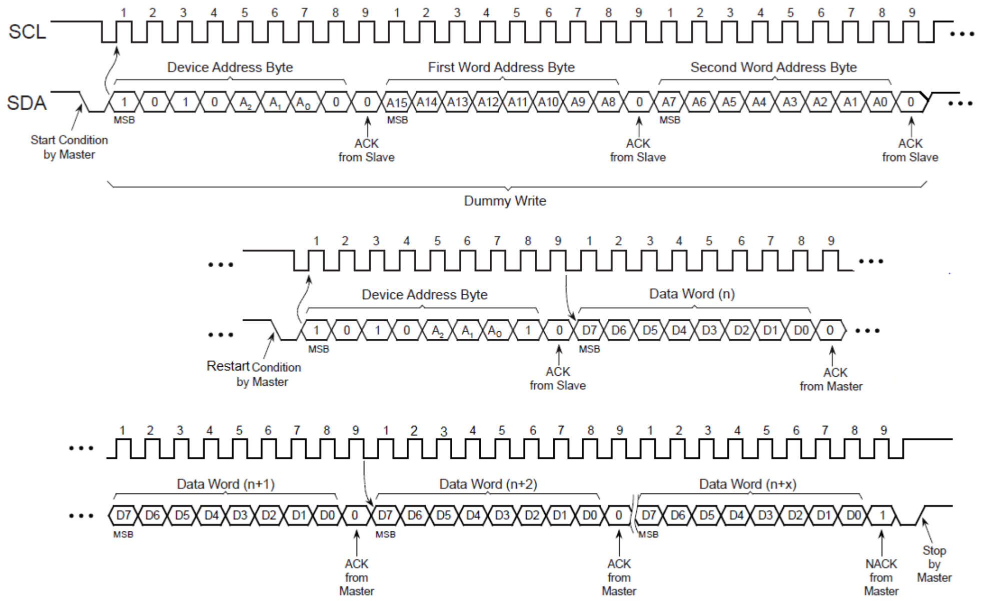


Figure 43. EEPROM read

In I2C master mode device supports following I2C features.

- I2C master is compliant with 100 KHz and 400 KHz I2C interface.
- Supports 7 bit I2C bus address. .
- Clock Synchronization
- Arbitration
- Start Byte

Clock Synchronization

Device supports multi master communication. In the multi master mode the SCL line synchronization is required. In clock synchronization process the SCL's high time is decided by the clock with the shortest high period and the low time is decided by the clock with the longest low time. The SCL line is a WAND, so if any one of the clock is counting its low time, the other clocks are also forced to stay low.

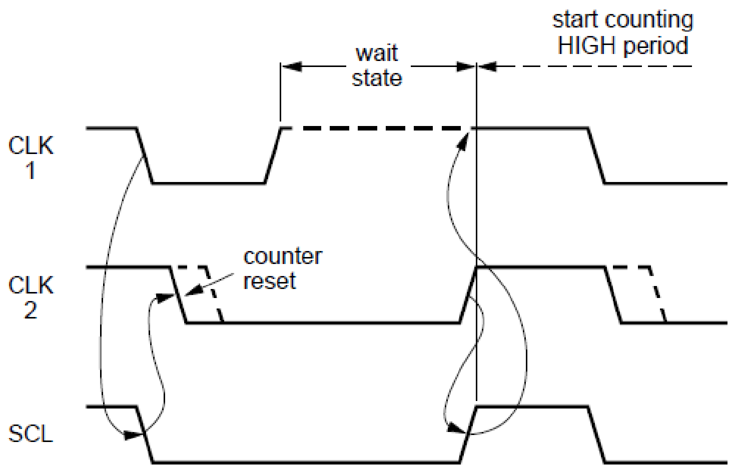


Figure 44. SCL Synchronization

Arbitration

Arbitration refers to a portion of the protocol required only if more than two masters are used in the system. Two masters may generate a START condition together within the hold time of the START condition which results in a valid START condition on the bus. Arbitration is then required to decide which master will complete its transaction. Arbitration proceeds bit by bit. During every bit, while SCL is HIGH, each master checks to see if the SDA level matches what it has sent. This process may take many bits. Two masters can actually complete an entire transaction without error, as long as the transmission is identical. The first time a master tries to send a HIGH, but detects the SDA level is low, the master knows that it has lost arbitration and turns off the transaction.

The arbitration is checked only in a few states. Arbitration is checked only when the master is sending data on the SDA line.

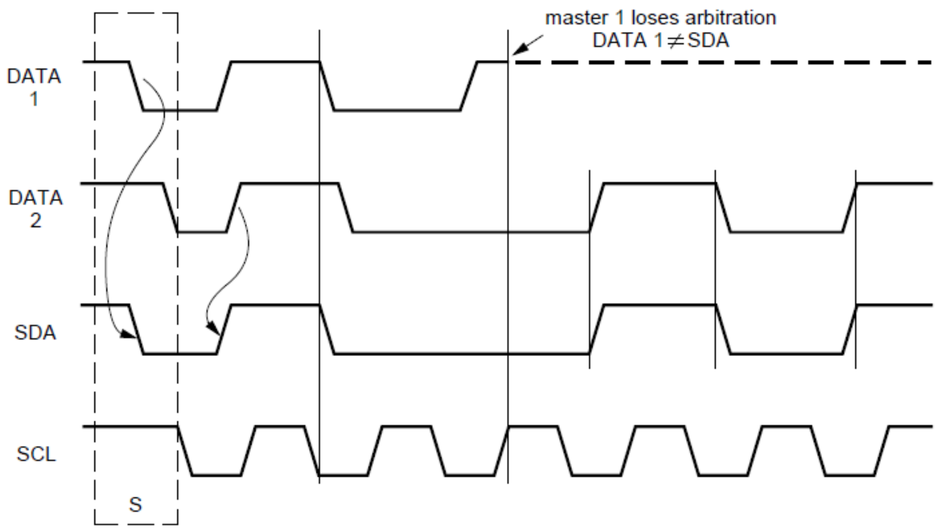


Figure 45. Arbitration

Start Byte

Microcontrollers can be connected to the I2C-bus in two ways. A microcontroller with an on-chip hardware I2C-bus interface can be programmed to be only interrupted by requests from the bus. When the device does not have such an interface, it must constantly monitor the bus via software. There is therefore a speed difference between fast hardware devices and a relatively slow microcontroller which relies on software polling. In this case, data transfer can be preceded by a start procedure which is much longer than normal (please Figure 46). The start procedure consists of:

- A START condition (S)
- A START byte (0000 0001)
- An acknowledge clock pulse (ACK)
- A repeated START condition (Sr).

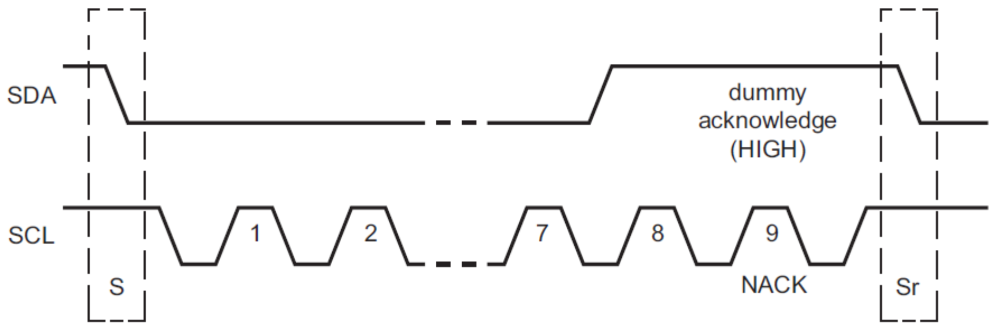


Figure 46. Start Byte Procedure

After the START condition S has been transmitted by a master which requires bus access, the START byte (0000 0001) is transmitted. Another microcontroller can therefore sample the SDA line at a low sampling rate until one of the seven zeros in the START byte is detected. After detection of this LOW level on the SDA line, the microcontroller can switch to a higher sampling rate to find the repeated START condition which is then used for synchronization. A hardware receiver resets upon receipt of the repeated START condition and therefore ignores the START byte. An acknowledge-related clock pulse is generated after the START byte. This is present only to conform with the byte handling format used on the bus. No device is allowed to acknowledge the START byte.

I2C Slave

Pin configuration of serial interface in I2C slave mode is as follows.

- SCLK (SCL)
- SDIO (SDA)
- SDO (A_1)
- CSB (A_0)

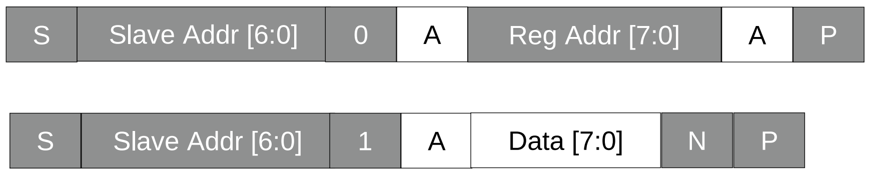
The device uses the SDIO and SCLK pins for a 2-wire serial interface that operates up to 400 Kb/s in Read and Write modes. It complies with the I2C bus standard. The I2C access protocol in device is byte access (random access) only for Write mode and both random and sequential access for Read mode.

The I2C serial interface can operate at either Standard rate (100 Kbps) or Fast rate (400 Kbps).

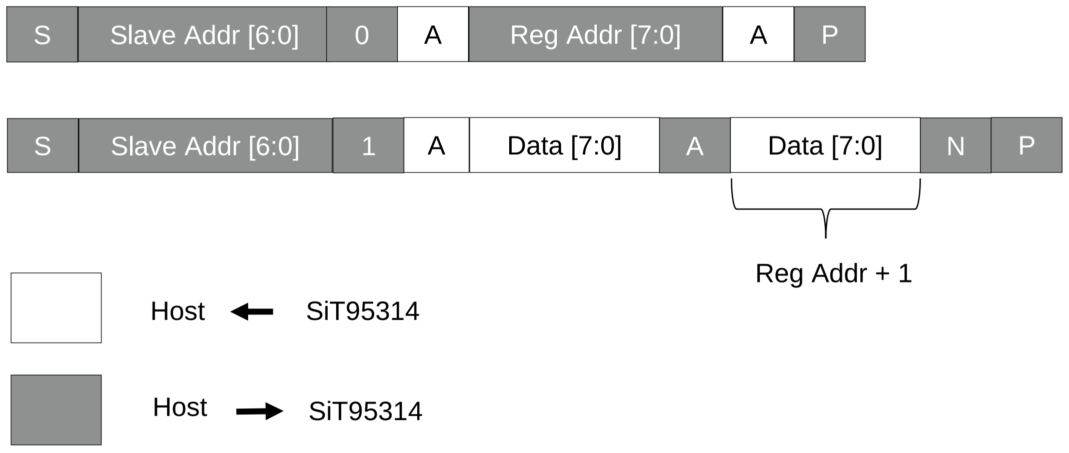
Last two address bits (A1,A0) are pin controlled. Default I2C slave address is 0110,10{SDO}{CSB} where SDO is weak pulled low and CSB is weak pulled high. So default I2C slave address will be 0x69.

Device can support up to 8 device addresses on same I2C bus by configuring A2 address through any of the GPIOs. However, GPIO latching needs to be enabled through a fuse bit to assign any of the GPIOs as A2 during power up. Thus slave address becomes 0110,1{GPIOx}{SDO}{CSB} where GPIOx, SDO and CSB values are controlled by the pins on the device. Slave address remains 0110,10{SDO}{CSB} in case user doesn’t assign A2 from GPIO. A2 is set as ‘0’ by default and 4 devices can be controlled on same I2C lines with 4 uniqe addresses

Read Operation - Single Byte

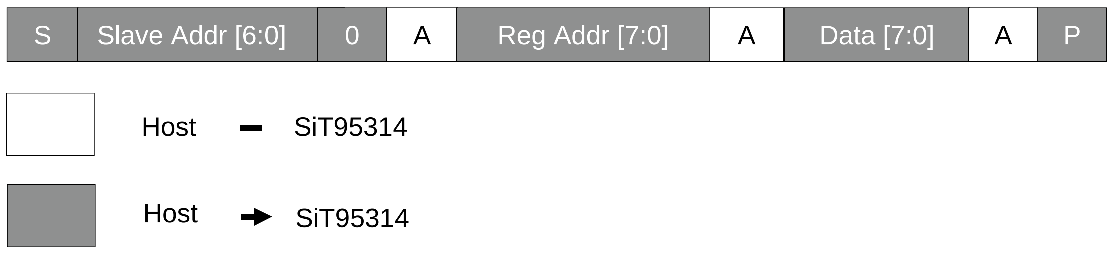


Read Operation - Burst (Auto Address Increment)



1- Read, 0 - Write, A - Acknowledge, N - Not Acknowledge, S - Start Condition, P - Stop Condition

Write Operation - Single Byte



1- Read, 0 - Write, A – Acknowledge (SDA LOW), N - Not Acknowledge (SDA HIGH), S - Start Condition, P - Stop Condition

Single Byte Write

- The master initiates the transaction by issuing a start condition, writes 7-bit slave address and then the read/write bit is written as 0 (write)
- The slave acknowledges by driving zero on the bus
- The master then writes the 8-bit register map address
- The slave acknowledges by driving zero on the bus
- The master then writes the 8-bit data to be written to the register map address specified
- The slave acknowledges by driving zero on the bus
- The master ends the transaction by issuing a stop condition

Single Byte Read

- The master initiates the transaction by issuing a start condition, writes 7-bit slave address and then the read/write bit is written as 0 (write)
- The slave acknowledges by driving zero on the bus
- The master then writes the 8-bit register map address
- The slave acknowledges by driving zero on the bus
- The master ends the transaction by issuing a stop condition
- The master re-initiates the transaction by issuing a start condition, writes 7-bit slave address and then the read/write bit is written as 1 (read)
- The slave then writes the 8-bit data to be written to the register map address specified
- The master does not acknowledge this transaction as the slave may assume a multi-byte read operation and there is a risk of slave holding the bus low
- The master ends the transaction by issuing a stop condition

Multi Byte Read

The multi-byte read mode is used to read a continuous segment of the register map. The multi-byte read is faster than performing multiple single byte reads as the device address and register map address need not be specified for every byte read from the register map

- The master initiates the transaction by issuing a start condition, writes 7-bit slave address and then the read/write bit is written as 0 (write)
- The slave acknowledges by driving zero on the bus
- The master then writes the 8 bit register map address
- The slave acknowledges by driving zero on the bus
- The master ends the transaction by issuing a stop condition
- The master re-initiates the transaction by issuing a start condition, writes 7 bit slave address and then the read/write bit is written as 1 (read)
- The slave then writes the 8 bit data to be written to the register map address specified
- The master acknowledges by driving zero on the bus
- The slave automatically increments the register map address and writes the data in at that address to the bus and the master acknowledges
- When all bytes of data are read, master ends the operation by not acknowledging the last read
- The master then ends the transaction by issuing a stop condition

I2C Bus Timing Specifications

Table 22. I2C bus Timing Specifications

Description	Symbol	Standard Mode		Fast Mode		Units
		Min	Max	Min	Max	
SCLK clock frequency	f_{SCL}	–	100	–	400	kHz
Hold time START condition	$t_{HD:STA}$	0.4	–	0.6	–	µs
Low period of the SCK clock	t_{LOW}	4.7	–	1.3	–	µs
High period of the SCK clock	t_{HIGH}	4.0	–	0.6	–	µs
Setup time for a repeated START condition	$t_{SU:STA}$	4.7	–	0.6	–	µs
Data hold time	$t_{HD:DAT}$	300	–	300	–	ns
Data setup time	$t_{SU:DAT}$	100	–	–	–	ns
Rise time	t_R	–	1000	–	300	ns
Fall time	t_F	–	300	–	300	ns
Setup time for STOP condition	$t_{SU:STO}$	4.0	–	0.6	–	µs
Bus-free time between STOP and START conditions	t_{BUF}	4.7	–	1.3	–	µs
Data valid time	$t_{VD;DAT}$	–	3.45	–	0.9	µs
Data valid acknowledge time	$t_{VD;ACK}$	–	0.9	–	0.9	µs

Note: In I2C mode, the serial data and clock have an on-chip 25 kΩ pull up resistor to VDDIO. Please refer Figure 47 for the nomenclature of these parameters.

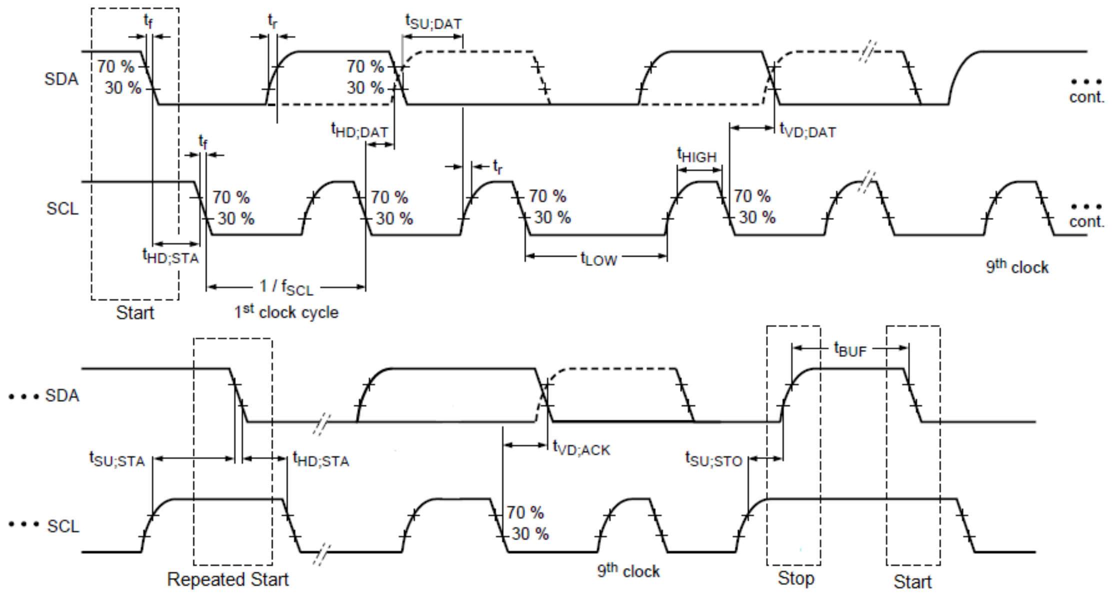


Figure 47. I2C Timing Waveform

SPI Protocol

Pin configuration of serial interface in SPI mode is as shown in Table 23.

Table 23. SPI Pin Configuration

PIN	4 wire mode	3wire mode
SCLK	Serial Clock	Serial Clock
SDIO	Serial Input	Serial Input/Output (Bi directional)
SDO	Serial Output	Not used
CSB	Chip Select	Chip Select

The SPI in a four-pin interface with Chip Select (CSB), Serial Input (SDIO), Serial Output (SDO), and Serial Clock (SCLK) pins. The SPI in a three-pin interface with Chip Select (CSB), Serial Input/Output (SDIO) and Serial Clock (SCLK) pins. SDIO pin acts as an input when receiving data from external SPI master, and acts as an output when transmitting data to external SPI master. The SPI bus on the device can run at speed up to 20 MHz. The SPI is a synchronous serial interface, which uses clock and data pins for serial access. When I2C1_SPI0 pin is Low, a Low on the CSB pin activates the SPI access.

1. The SPI can operate up to 20 MHz for write/read operations.
2. The SPI receives serial data from the external master and provides Wr/rdn, address and data to the register map during the write operation.
3. The SPI receives serial data from the external master and provides Wr/rdn and address to the register map and uses the read data obtained from the register map, serializes the same and transmit to the master.
4. In SiT95314, the total packet size for each SPI single write or single read transaction is 32 bits where the first 8 bits are address instruction byte, the next 8 bits are address and the next 8 bits are data instruction byte and the last 8 bits are data. For incremental write or incremental read operation, the total packet size is 16 bits where the first 8 bits are for instruction byte and the next 8 bits are for data. For burst write or burst read operation, the total packet size is n+2 bytes where first byte is for instruction, second byte is for address and n bytes are for data.

Table 24. SPI Instructions

Function	Value of Instruction Byte
Set Address	000xxxxx
Single Write	010xxxxx
Single Read	100xxxxx
Incremental Write	011xxxxx
Incremental Read	101xxxxx
Burst Write	111xxxxx
Burst Read	001xxxxx

5. In SiT95314 for single write operation, the master assembles the instruction byte, address and data for write operation on the falling edge of the spi clock and the slave in the SiT95314 captures the same on the rising edge of the SPI clock. For incremental write operation the master assembles the instruction byte and data for write operation on the falling edge of the spi clock and the slave in the SiT95314 captures the same on the rising edge of the SPI clock. For the burst write operation, the master assembles the instruction bytes, address byte and the data bytes for write operation on the falling edge of the spi clock and the slave in the SiT95314 captures the same on the rising edge of the SPI clock.
6. In SiT95314 for read operation, the master assembles the instruction byte, address for read operation on the falling edge of the spi clock and the slave in the SiT95314 captures the same on the rising edge of the SPI clock. The falling edge after the 8th rising SPI clock (i.e. the last address bit), is used by the slave to assemble the first read data bit which is captured by the master on the 9th edge of the SPI clock. Subsequent 7 more clocks are used for the 7 remaining data bits. For incremental read operation, the master assembles the instruction byte on the falling edge of the SPI clock and the slave in the SiT95314 captures the same on the rising edge of the SPI clock. The falling edge after the 8th rising edge of SPI clock (i.e. the last instruction bit), is used by the slave to assemble the first read data bit which is captured by the master on the 9th edge of the SPI clock. Subsequent 7 more clocks are used for the 7 remaining data bits. For burst read operation, the master assembles the instruction byte, address for read operation on the falling edge of the spi clock and the slave in the SiT95314 captures the same on the rising edge of the SPI clock. The falling edge after the 16th rising SPI clock (i.e. the last address bit), is used by the slave to assemble the first read data bit which is captured by the master on the 17th edge of the SPI clock. Subsequent clocks are used for the remaining data bits.
7. In SiT95314 the transmitter always sends data on the falling edge of the SPI clock to be captured in the receiver by the rising edge of the SPI clock. The transmitter can be the master for the whole operation of the write and for the control and address portions of the read. The slave is the transmitter during the data portion of the read cycle.
8. In a 4-wire mode of operation, the output data bit slave is controlled only when transmitting the read data bits and is in HiZ for the rest of the times. In 3-wire mode of operation, the data bit is always considered as an input except when the read data is being transimitted by chaning the direction to output.

SPI Timing Specifications

Table 25. SPI Timing Specifications

Description	Symbol	Min	Typ	Max	Units
SCLK clock frequency	f_{SCLK}	-	–	20	MHz
Clock pulse width HIGH	t_{CH}	20			ns
Clock pulse width LOW	t_{CL}	20			ns
CSB HIGH time	t_{CS}	50			ns
CSB setup time	t_{CSS}	25			ns
CSB hold time	t_{CSH}	25			ns
Data in setup time	t_{SD}	10			ns
Data in hold time	t_{HD}	10			ns
Output valid	t_{CO}			10	ns
Output setup time	t_{SO}			10	ns
Output hold time	t_{HO}			10	ns

Note: In SPI mode, the SDIO pin is driven to HiZ when CSB is high for the chip in SPI Mode. Please refer Figure 48 for the nomenclature of these parameters.

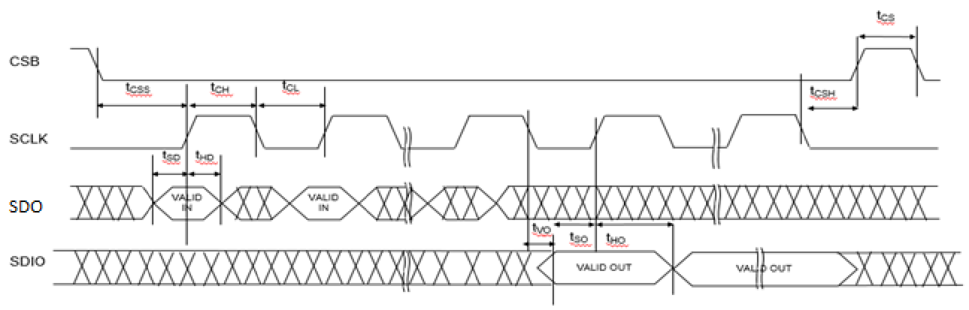


Figure 48. SPI Timing Diagram

SPI Single byte write

- The master initiates the transaction by issuing a start condition by pulling csb_i to active low
- The master assembles the serial data on the falling edge of the SPI clock so the SPI slave can capture the same on the rising edge of the SPI clock
- The first 8 bits are instruction bits for the set address
- The next 8 bits (second byte) are used for the register map address
- After the following posedge csb_i will be toggle and start the Transaction
- The first 8 bits are instruction bits for the single write
- The next 8 bits for the write data
- The 8th rising edge of the SPI clock is used to capture the last instruction bit. The SPI slave then assembles the address, data, enable and wr_rdn to the PIF slave block. The inverted version of the next falling edge of the SPI clock is used by the SPI slave to capture the address, data, enable and wr_rdn to write to the respective registers.
- The CSB is then de-activated (by going high) by the master
- For the next write operation, CSB is held high for at least a duration of two spi clocks following which the entire operation can start again.

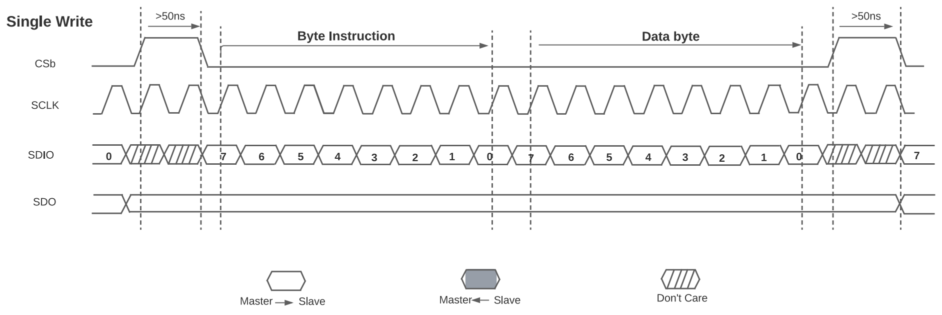


Figure 49. SPI Single Write

SPI Incremental byte write

- The master initiates the transaction by issuing a start condition by pulling csb_i to active low
- The master assembles the serial data on the falling edge of the SPI clock so the SPI slave can capture the same on the rising edge of the SPI clock
- The first 8 bits are instruction bits
- The next 8 bits (second byte) are used for the register map data
- The address is an automatic increment of previous address used. Hence the master should have done a single write transaction before starting the incremental write operation
- The 16th rising edge of the SPI clock is used to capture the last data bit. The SPI slave then assembles the address, data, enable and wr_rdn to the PIF slave block. The inverted version of the next falling edge of the SPI clock is used by the SPI slave to capture the address, data, enable and wr_rdn to write to the respective registers.
- The CSB is then de-activated (by going high) by the master
- For the next write operation, CSB is held high for at least a duration of two spi clocks following which the entire operation can start again

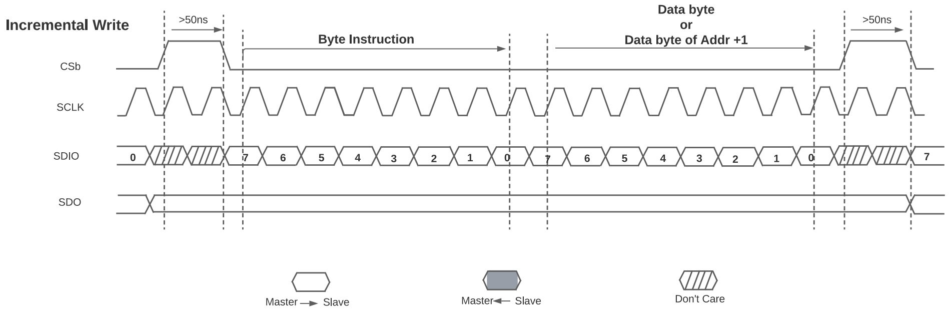


Figure 50. SPI Incremental Write

SPI Burst write

- The master initiates the transaction by issuing a start condition by pulling csb_i to active low
- The master assembles the serial data on the falling edge of the SPI clock so the SPI slave can capture the same on the rising edge of the SPI clock
- The first 8 bits are instruction bits
- The next 8 bits (second byte) are used for the register map address
- The next sets of 8 bits (n bytes) are used for the register map data for n write operations.
- The 24th rising edge of the SPI clock is used to capture the last data bit of the first data byte. The SPI slave then assembles the address, data, enable and wr_rdn to the PIF slave block. The inverted version of the next falling edge of the SPI clock is used by the SPI slave to capture the address, data, enable and wr_rdn to write to the respective registers. Following this after every 8 clocks, a write operation is performed for all the n bytes.
- The CSB is then de-activated (by going high) by the master
- For the next write operation, CSB is held high for at least a duration of two spi clocks following which the entire operation can start again.

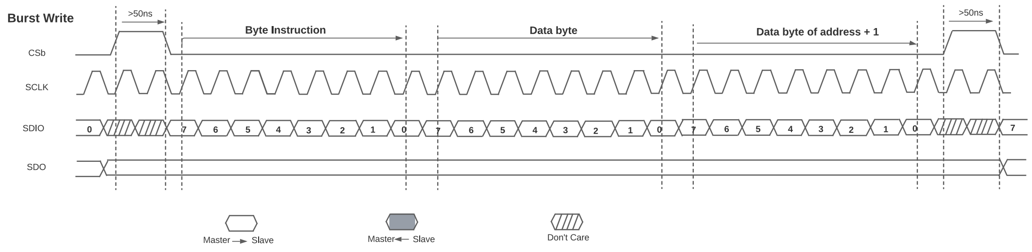


Figure 51. SPI Burst Write

SPI Single byte Read

- The master initiates the transaction by issuing a start condition by pulling csb_i to active low
- The master assembles the serial data on the falling edge of the SPI clock so the SPI slave can capture the same on the rising edge of the SPI clock
- The first 8 bits are instruction bits
- The next 8 bits (second byte) are used for the register map address
- After the following posedge csb_i will be toggle and start the Transaction
- The first 8 bits are instruction bits for the single write
- The 8th rising edge of the SPI clock is used to capture the last instruction bit. The SPI slave then assembles the address, enable and wr_rdn to the PIF slave block. The PIF slave block then returns the 8 bit read data where the first bit is transmitted on the falling edge after the 8th clock to be captured by the master on the 9th clock. After 7 more clocks all the bits of the read data are transmitted.
- The CSB is then de-activated (by going high) by the master
- For the next read operation, CSB is held high for at least a duration of two spi clocks following which the entire operation can start again.

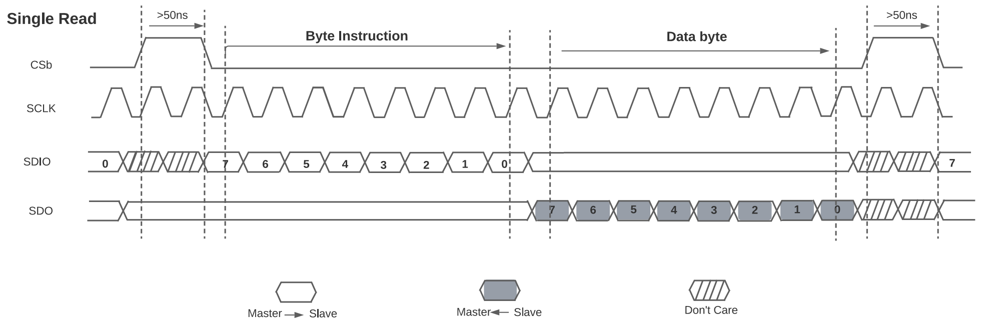


Figure 52. SPI Single Read

SPI Incremental byte Read

- The master initiates the transaction by issuing a start condition by pulling csb_i to active low
- The master assembles the serial data on the falling edge of the SPI clock so the SPI slave can capture the same on the rising edge of the SPI clock
- The first 8 bits are instruction bits
- The devices use an auto-increment of the previous used address for current read operation. The master should have hence done at least one single write/read operation for this increment read operation to work.
- The 8th rising edge of the SPI clock is used to capture the last instruction bit. The SPI slave then assembles the address, enable and wr_rdn to the PIF slave block. The PIF slave block then returns the 8 bit read data where the first bit is transmitted on the falling edge after the 8th clock to be captured by the master on the 9th clock. After 7 more clocks all the bits of the read data are transmitted.
- The CSB is then de-activated (by going high) by the master
- For the next read operation, CSB is held high for at least a duration of two spi clocks following which the entire operation can start again.

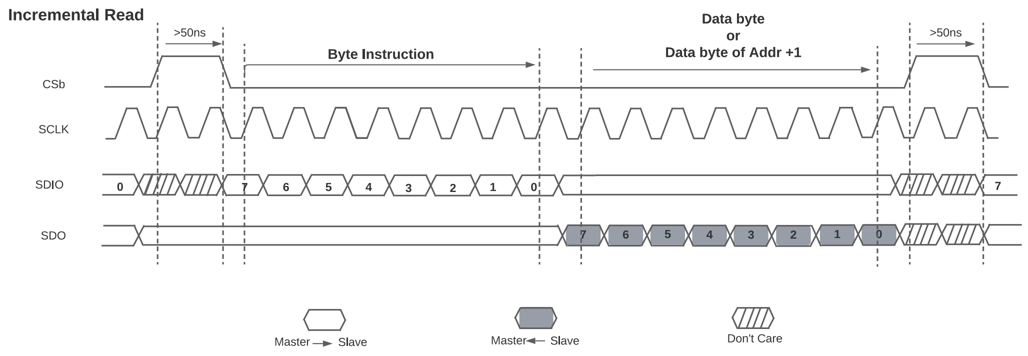


Figure 53. SPI Incremental Read

SPI Burst Read

- The master initiates the transaction by issuing a start condition by pulling csb_i to active low
- The master assembles the serial data on the falling edge of the SPI clock so the SPI slave can capture the same on the rising edge of the SPI clock
- The first 8 bits are instruction bits
- The next 8 bits (second byte) are used for the register map address
- The 16th rising edge of the SPI clock is used to capture the last address bit. The SPI slave then assembles the address, enable and wr_rdn to the PIF slave block. The PIF slave block then returns the 8 bit read data where the first bit is transmitted on the falling edge after the 16th clock to be captured by the master on the 17th clock. After 7 more clocks all the bits of the first read data are transmitted. For subsequent bytes to be read, the address is auto-incremented and 8 clocks are used to transmit for each of the n bytes
- The CSB is then de-activated (by going high) by the master
- For the next read operation, CSB is held high for at least a duration of two spi clocks following which the entire operation can start again.

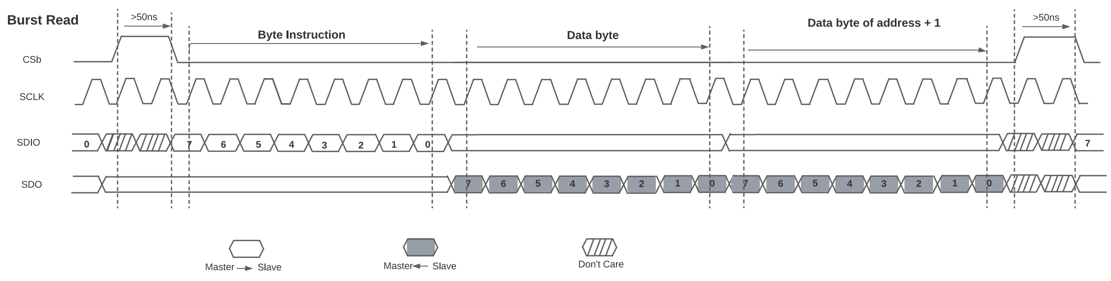


Figure 54. SPI Burst Read

Monitoring through the register map read back: Status and Notify

SiT95314 provides various Status and notify bits that can be accessed from the register map. Below are the details of the procedure to be followed to access the same.

The alarm registers are a set of three types of registers distributed between the various pages as described in Table 26 and illustrated with some examples.

- The Status registers are the current dynamic status of a defect. The live status defects are active high with a '1' indicating the defect is present.
- The Notify registers are the sticky bits for a defect. Sometimes, we may get a very short pulse for the status and it may not be possible for the external user to capture the same. Hence a notify register is provided. The notify register is set to 1 whenever there is a rising edge of the corresponding status register. This is a sticky bit and stays at 1 till the user writes a 1 to that specific bit to clear it.
- Each notify has a masking bit to enable or disable its operation. The notify sticky bit operates only if the corresponding masking bit is set to 1. If the masking register bit is set to 0, notify will not be asserted even when status toggles. The default value for the mask register is 0xff so all the notify signals are enabled. Once the user writes a 1 to clear the notify, the notify bit can again go high on the next rising edge of the status.
- These registers operate on the internal 4 MHz RC clock. When there is a defect (i.e. status) of any bit in register, it gets asserted and de-asserted in a live mode. User can read the corresponding register location to see the current status at any time.
- On the other hand, the default value of the notify and masking register is 0xff – user has to write 0xff to both these registers to clear them at the beginning and use all notifies.It is recommended to clear all notifies after programming a profile.
- The INTRB pin is used as a NOR operation of the selected notifies. The choice of the Notify listing that is used for the INTRB pin is selectable in the GUI when creating the profile. The sticky notifies that are selected and used for INTRB need to be cleared for restoring the INTRB to 1 for further sticky defect monitoring using this pin.

Tabular Listing

Table 26 details the name of the alarm, the page it is located in, the address of status, notify and mask registers in that page and the bit number in the address. In order to access a page of the register map, the particular page number has to be written to address 0xff. For instance, to either write to or read from page 1, first the user needs to write to 0xff a value of 0x01. Following this, any number of write or read operations can be done with page 1.

Table 26. Tabular Listing of Alarm Registers

Sr No	Name of Signal	Description	Pg no	Status Register		Notify Register		Mask Register	
				Register Address	Bit No	Register Address	Bit No	Register Address	Bit No
1	xoclk_loss	XO clock Loss	0	0x9d	4	0x9e	4	0x9f	4
2	cmon_pll_inner_lol	clock mon pll inner loss of lock			5		5		5
3	plla_outer_lol	plla outer loss of lock	0	0x06	0	0x07	0	0x08	0
4	pllb_outer_lol	pllb outer loss of lock			1		1		1
5	pllc_outer_lol	pllc outer loss of lock			2		2		2
6	plld_outer_lol	plld outer loss of lock			3		3		3
9	plla_ho_freeze	plla hold over freeze	0	0x0a	0	0x0b	0	0x0c	0
10	pllb_ho_freeze	pllb hold over freeze			1		1		1
11	pllc_ho_freeze	pllc hold over freeze			2		2		2
12	plld_ho_freeze	plld hold over freeze			3		3		3
15	plla_inner_lol	plla inner loop loss of lock	0	0x92	0	0x93	0	0x94	0
16	pllb_inner_lol	pllb inner loop loss of lock			1		1		1
17	pllc_inner_lol	pllc inner loop loss of lock			2		2		2
18	plld_inner_lol	plld inner loop loss of lock			3		3		3
21	eeprom_ctrl_eeprom_read_do ne	eeprom ctrl eeprom read done	0	0x96	0	0x97	0	0x98	0

Sr No	Name of Signal	Description	Pg no	Status Register		Notify Register		Mask Register	
				Register Address	Bit No	Register Address	Bit No	Register Address	Bit No
22	eeprom_ctrl_crc_deft	eeprom ctrl crc defect			1		1		1
23	eeprom_ctrl_arb_lost_deft	eeprom ctrl arb lost defect			2		2		2
24	eeprom_ctrl_page_id_deft	eeprom ctrl page id defect			3		3		3
25	eeprom_ctrl_dev_id_deft	eeprom ctrl dev id defect			4		4		4
26	eeprom_ctrl_eeprom_size_deft	eeprom ctrl eeprom size defect			5		5		5
27	eeprom_ctrl_word_add_deft	eeprom ctrl word add defect			6		6		6
28	eeprom_ctrl_dev_add_deft	eeprom ctrl dev add defect			7		7		7
29	eeprom_ctrl_start_timeout_deft	eeprom ctrl start timeout defect	0	0x9a	0	0x9b	0	0x9c	0
30	eeprom_ctrl_data_deft	eeprom ctrl data defect			1		1		1
31	eeprom_ctrl_stop_deft	eeprom ctrl stop defect			2		2		2
32	eeprom_ctrl_start_deft	eeprom ctrl start defect			3		3		3
33	eeprom_ctrl_read_done_timeout_deft	eeprom ctrl read done timeout defect			4		4		4
34	plla_phase_loss_of_lock	plla phase loss of lock	0	0x0a	6	0x0b	6	0x0c	6
35	pllb_phase_loss_of_lock	pllb phase loss of lock			7		7		7
36	pllc_phase_loss_of_lock	pllc phase loss of lock	0	0x92	6	0x93	6	0x94	6
37	plld_phase_loss_of_lock	plld phase loss of lock			7		7		7
38	clk0p_freq_fine_drifted	clk0p freq fine drifted	6	0x02	0	0x03	0	0x04	0
39	clk0p_freq_coarse_drifted	clk0p freq coarse drifted			1		1		1
40	clk_in0p_loss	clk in0p loss			2		2		2
41	clk_in0p_loss_with_FD	clk in0p loss with FD			3		3		3
42	clk1p_freq_fine_drifted	clk1p freq fine drifted			4		4		4
43	clk1p_freq_coarse_drifted	clk1p freq coarse drifted			5		5		5
44	clk_in1p_loss	clk in1p loss			6		6		6
45	clk_in1p_loss_with_FD	clk in1p loss with FD			7		7		7
46	clk2p_freq_fine_drifted	clk2p freq fine drifted	6	0x06	0	0x07	0	0x08	0
47	clk2p_freq_coarse_drifted	clk2p freq coarse drifted			1		1		1
48	clk_in2p_loss	clk in2p loss			2		2		2
49	clk_in2p_loss_with_FD	clk in2p loss with FD			3		3		3
50	clk3p_freq_fine_drifted	clk3p freq fine drifted			4		4		4
51	clk3p_freq_coarse_drifted	clk3p freq coarse drifted			5		5		5
52	clk_in3p_loss	clk in3p loss			6		6		6
53	clk_in3p_loss_with_FD	clk in3p loss with FD			7		7		7
54	clk0n_freq_fine_drifted	clk0n freq fine drifted	6	0x92	0	0x93	0	0x94	0
55	clk0n_freq_coarse_drifted	clk0n freq coarse drifted			1		1		1
56	clk_in0n_loss	clk in0n loss			2		2		2
57	clk_in0n_loss_with_FD	clk in0n loss with FD			3		3		3
58	clk1n_freq_fine_drifted	clk1n freq fine drifted	6	0x92	4	0x93	4	0x94	4

Sr No	Name of Signal	Description	Pg no	Status Register		Notify Register		Mask Register	
				Register Address	Bit No	Register Address	Bit No	Register Address	Bit No
59	clk1n_freq_coarse_drifted	clk1n freq coarse drifted			5		5		5
60	clk_in1n_loss	clk in1n loss			6		6		6
61	clk_in1n_loss_with_FD	clk in1n loss with FD			7		7		7
62	clk2n_freq_fine_drifted	clk2n freq fine drifted	6	0x96	0	0x97	0	0x98	0
63	clk2n_freq_coarse_drifted	clk2n freq coarse drifted			1		1		1
64	clk_in2n_loss	clk in2n loss			2		2		2
65	clk_in2n_loss_with_FD	clk in2n loss with FD			3		3		3
66	clk3n_freq_fine_drifted	clk3n freq fine drifted			4		4		4
67	clk3n_freq_coarse_drifted	clk3n freq coarse drifted			5		5		5
68	clk_in3n_loss	clk in3n loss			6		6		6
69	clk_in3n_loss_with_FD	clk in3n loss with FD			7		7		7

Examples for Live Status Read Back

Some examples are presented based on Table 26 for reading the live status of the defects.

In the pseudo code presented below:

wr_cmd(address, data): refers to a “Write Command” where the corresponding data is written in to the specified register address

x= rd_cmd(address): refers to a “Read Command” where the corresponding data is read from the specified register address and stored in the variable ‘x’

y >> x: denotes a bit wise right shift on the number y by x bit locations

y << x:denotes a bit wise left shift on the number y by x bit locations

& is the logical AND operation (bit wise)

Dynamic registers to read the various alarm registers in the RealTime page.

1. Input Clocks CL and FD Related Real Time live status read back:

wr_cmd(0xff, 0x06)   Program the CLKMON_SYS page number

Clock Monitor dynamic status

clock_mon_dyn_status = rd_cmd(0x02) & 0xff

```
(clock_mon_dyn_status >> 0) & 0x01        INP0 Status for FD (fine),        Read bit position [0]
(clock_mon_dyn_status >> 1) & 0x01        INP0 Status for FD (coarse),      Read bit position [1]
(clock_mon_dyn_status >> 2) & 0x01        INP0 Status for CL,               Read bit position [2]
(clock_mon_dyn_status >> 3) & 0x01        INP0 Status for CL with FD,  Read bit position [3]
(clock_mon_dyn_status >> 4) & 0x01        INP1 Status for FD (fine),        Read bit position [4]
(clock_mon_dyn_status >> 5) & 0x01        INP1 Status for FD (coarse), Read bit position [5]
(clock_mon_dyn_status >> 6) & 0x01        INP1 Status for CL,               Read bit position [6]
(clock_mon_dyn_status >> 7) & 0x01        INP1 Status for CL with FD,       Read bit position [7]
```

2. PLL Related Real Time live status read back:

wr_cmd(0xff, 0x00)   Program the MAIN_SYS page number, Page 0

PLL Lock Loss dynamic status

pll_outer_lol_dyn_status = rd_cmd(0x06) & 0xff

(pll_outer_lol_dyn_status >> 0) & 0x01   PLLA Status for Outer LL, Read bit position [0]

(pll_outer_lol_dyn_status >> 1) & 0x01 PLLB Status for Outer LL, Read bit position [1]

(pll_outer_lol_dyn_status >> 2) & 0x01   PLLC Status for Outer LL, Read bit position [2]

(pll_outer_lol_dyn_status >> 3) & 0x01   PLLD Status for Outer LL, Read bit position [3]

Holdover Status

pll_ho_freeze_dyn_status = rd_cmd(0x0a) & 0xff

(pll_ho_freeze_dyn_status >> 0) & 0x01   PLLA Status for HO, Read bit position [0]

(pll_ho_freeze_dyn_status >> 1) & 0x01   PLLB Status for HO, Read bit position [1]

(pll_ho_freeze_dyn_status >> 2) & 0x01   PLLC Status for HO, Read bit position [2]

(pll_ho_freeze_dyn_status >> 3) & 0x01   PLLD Status for HO, Read bit position [3]

3. XO clock loss Related Real Time live status read back:

CLOS_X1X2, XO Clock Loss

wr_cmd(0xff, 0x00)   Program the MAIN_SYS page number, Page 0

clos_x1x2 = rd_cmd(0x9d) & 0x10 // XO CL Status, Read bit position [2]

Examples of Sticky Bit Clearing

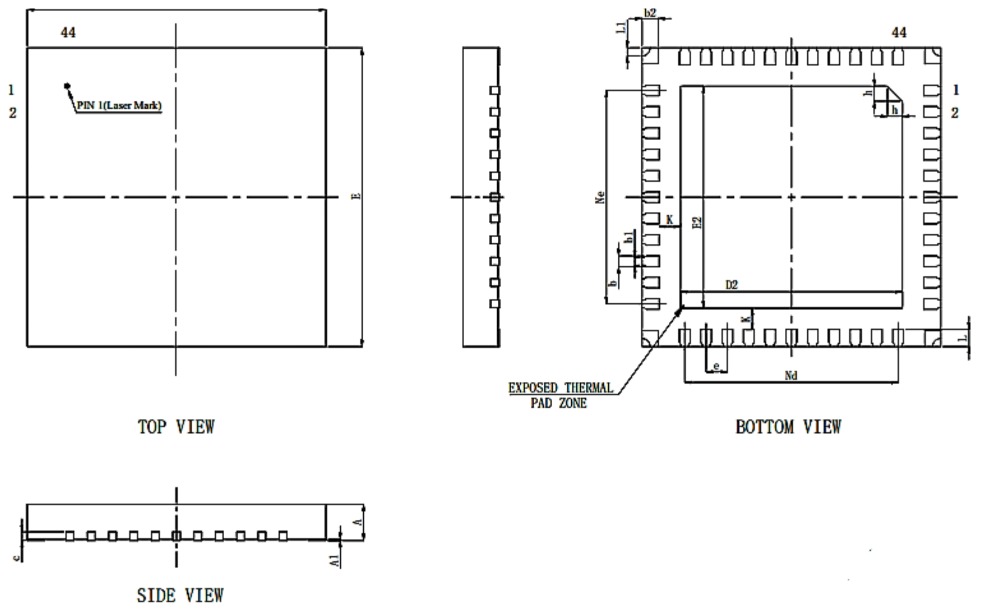
As described earlier, the sticky notify bits are cleared by writing a '1' to the corresponding notify bit itself. The notify bit by itself is enabled by writing a '1' to the corresponding mask bit.

In the pseudo code presented below,

rmw_cmd(addr,bit_loc,no_of_bits,data): denotes the read/modify/write operation where no_of_bits number of bits at bit_loc location (denoted as 7:0) is replaced with the data at address location addr.

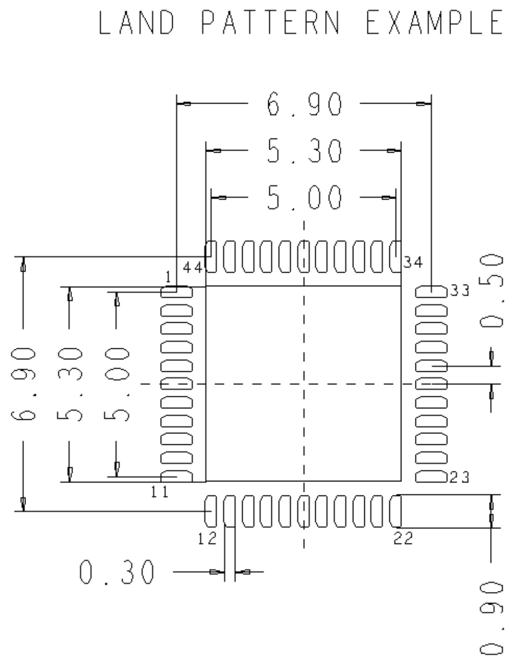
```
def clr_intb_CMON_IN0P():
    This function is used to clear the sticky notify for clear clock mon notify for IN0P
    Write the page number
    wr_cmd(0xff, 0x06)
     Information to clr ** Page6: reg03[3:0]=0x0f **
    addr       = 0x3
    bit_loc    = 3
    no_of_bits = 4
    data       = 0x0f
    rmw_cmd(addr,bit_loc,no_of_bits,data)


def clr_intb_PLL_OUTER_LOL():
    This function is used to clear the sticky notify for outer loss of lock notify for all PLLs
    Write the page number
    wr_cmd(0xff, 0)
     Information to clr ** Page 0: reg07[3:0] = 0x0f  **
    addr       = 0x7
    bit_loc    = 3
    no_of_bits = 4
    data       = 0x0f
    rmw_cmd(addr,bit_loc,no_of_bits,data)


def clr_intb_PLL_HO():
    This function is used to clear the sticky notify for outer loss of lock notify for all PLLs
    Write the page number
    wr_cmd(0xff, 0)
     Information to clr ** Page 0: reg0b[3:0] = 0x0f  **
    addr       = 0x7
    bit_loc    = 3
    no_of_bits = 4
    data       = 0x0f
    rmw_cmd(addr,bit_loc,no_of_bits,data)


def clr_intb_XO_CL():
     This function is used to clear the sticky notify for XO Clock Loss
    Write the page number
    wr_cmd(0xff, 0)
     Information to clr ** Page 0: reg9e[4]=1 **
    addr       = 0x9e
    bit_loc    = 4
    no_of_bits = 1
    data       = 1
    rmw_cmd(addr,bit_loc,no_of_bits,data)


def clr_intrb():
    '''
    This is the main clear function
    which calls the 4 clear functions
    '''
   clr_intb_CMON_IN0P()
   clr_intb_PLL_OUTER_LOL()
   clr_intb_PLL_HO()
   clr_intb_XO_CL()
```

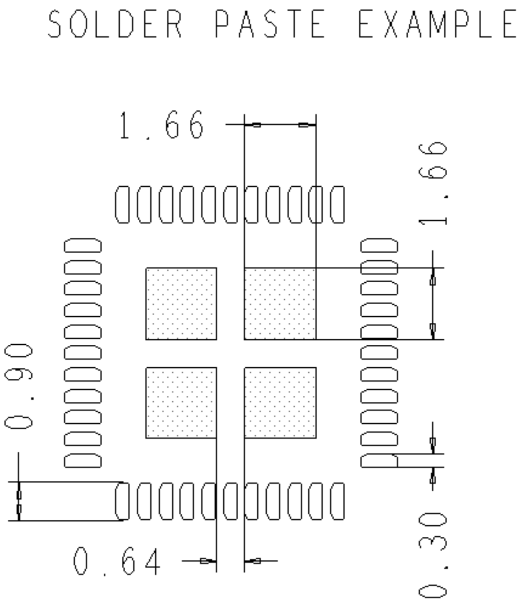
Package Information



SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	0.80	0.85	0.90
A1	0	0.02	0.05
b	0.20	0.25	0.30
b1	0.18REF		
b2	0.325	0.375	0.425
c	0.203REF		
D	6.90	7.00	7.10
D2	5.10	5.20	5.30
e	0.50BSC		
Nd	5.00BSC		
E	6.90	7.00	7.10
E2	5.10	5.20	5.30
Ne	5.00BSC		
L	0.35	0.40	0.45
L1	0.195REF		
K	0.50REF		
h	0.30	0.35	0.40

Figure 55. Package Diagram (44 QFN)





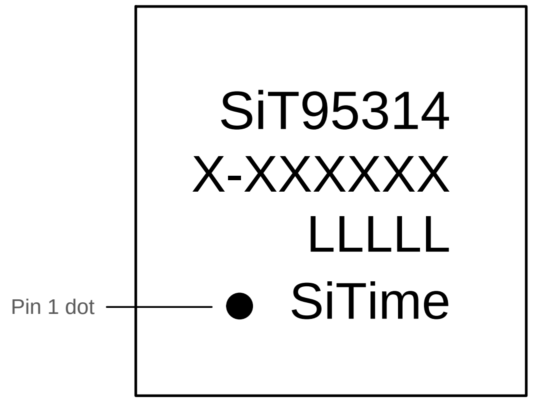
General Notes

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. This Land Pattern Design is based on the IPC-7351 guidelines.
3. All dimensions shown are at Maximum Material Condition (MMC). Least Material Condition is calculated based on a fabrication Allowance of 0.05 mm.

SOLDER MASK DESIGN

4. All metal pads are to be non-solder mask defined (NSMD).

Top Marking



Line 1: “SiT95314”, SiTime part number

Line 2: X-XXXXXX,

- 1st X indicates the device revision code such as rev B, C…etc, empty/blank indicate rev A
- “-XXXXXX” indicates specific custom configuration code, empty for non-programmable devices.

Line 3: “LLLLL”, Lot code from SiTime

Line 4: Pin 1 dot and “SiTime” logo

Table 27. Revision History

Revisions	Release Date	Change Summary
0.5	7-Dec-2023	Initial Release
0.51	15-Apr-2024	Ordering Information update
0.52	24-Apr-2025	Updated Packaging Ordering with options “P” and “N” Updated Top Marking section Formatting updates