Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!


## Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

Register Map: Section 6.2

# ZL30255

## Dual-Channel Any-to-Any Clock Multiplier and Jitter Attenuator

Data Sheet

July 2016

**Ordering Information**

| | | |
|---|---|---|
| ZL30255LFG7 | 64 Pin LGA | Trays |
| ZL30255LFF7 | 64 Pin LGA | Tape and Reel |

Ni Au

Package size: 5 x 10 mm

**-40°C to +85°C**

## Features

- **Two Independent Channels**
- **Three Input Clocks Per Channel**
  - Three inputs, two differential/CMOS, one CMOS
  - Any input frequency from 1kHz to 1250MHz (1kHz to 300MHz for CMOS)
  - Inputs continually monitored for activity and frequency accuracy
  - Automatic or manual reference switching
- **Low-Bandwidth DPLL Per Channel**
  - Programmable bandwidth, 14Hz to 500Hz
  - Attenuates jitter up to several UI
  - Freerun or digital hold on loss of all inputs
  - Digitally controlled phase adjustment
- **Low-Jitter Fractional-N APLL and 3 Outputs Per Channel**
  - Any output frequency from <1Hz to 1035MHz
  - High-resolution fractional frequency conversion with 0ppm error
  - Easy-to-configure, encapsulated design requires no external VCXO or loop filter components
  - Each output has independent dividers
  - Output jitter is typically 0.16 to 0.28ps RMS (12kHz-20MHz integration band)
  - Outputs are CML or 2xCMOS, can interface to LVDS, LVPECL, HSTL, SSTL and HCSL
  - In 2xCMOS mode, the P and N pins can be different frequencies (e.g. 125MHz and 25MHz)
  - Per-output supply pin with CMOS output voltages from 1.5V to 3.3V
  - Precise output alignment circuitry and per-output phase adjustment
  - Per-output enable/disable and glitchless start/stop (stop high or low)
- **General Features**
  - Automatic self-configuration at power-up from internal EEPROM; up to four configurations pin-selectable
  - Numerically controlled oscillator mode
  - Spread-spectrum modulation mode
  - Zero-delay mode with external feedback
  - SPI or $I^2C$ processor Interface
  - Easy-to-use evaluation software

## Applications

- Telecom OTN and SONET/SDH/SyncE cards
- Frequency conversion and jitter attenuation in a wide variety of equipment types

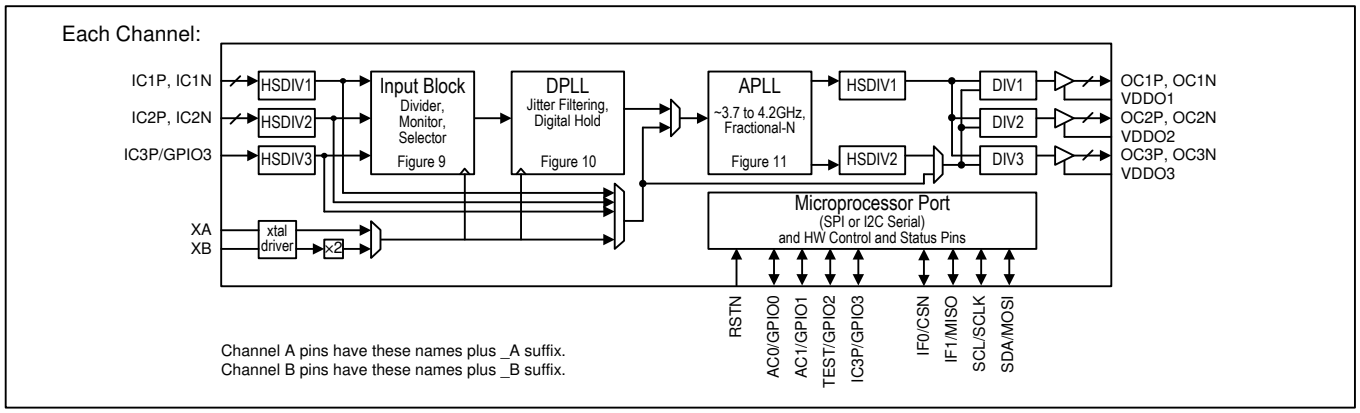


**Figure 1 - Functional Block Diagram**

## Table of Contents

## List of Figures



## List of Tables

# 1. Application Examples

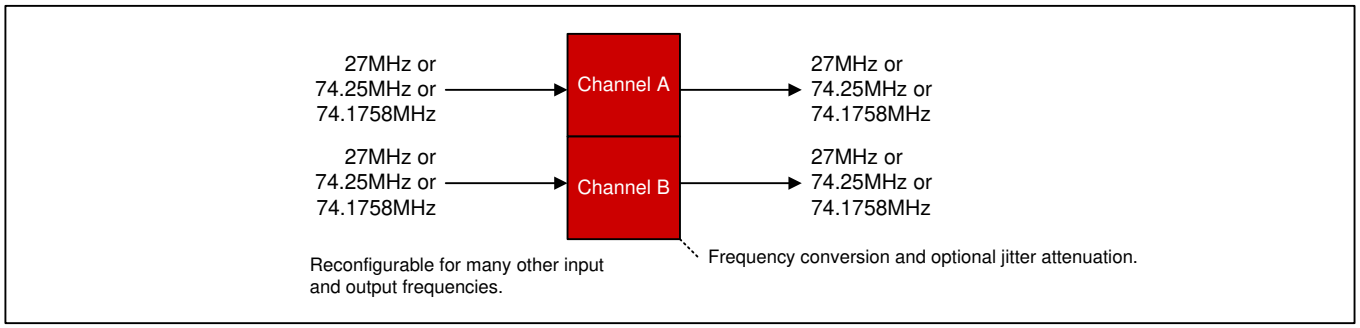


**Figure 2 – Broadcast Video Frequency Conversion**

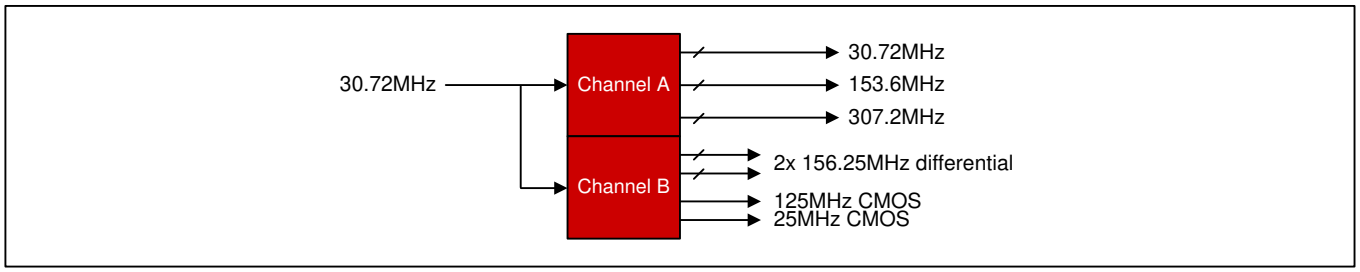


**Figure 3 – Base Station Frequency Conversion with Jitter Attenuation**

# 2. Detailed Features

## 2.1 Input Block Features

- Three input clocks per channel, two differential or single-ended, one single-ended
- Input clocks can be any frequency from 1kHz up to 1250MHz (differential) or 300MHz (single-ended)
- Supported telecom frequencies include PDH, SDH, Synchronous Ethernet, OTN, wireless
- Inputs constantly monitored by programmable activity monitors and frequency monitors
- Fast activity monitor can disqualify the input after a few missing clock cycles
- Frequency measurement and monitoring with 1% resolution
- Optional input clock invalidation on GPIO assertion to react to LOS signals from PHYs

## 2.2 DPLL Features

- One DPLL per channel
- Very high-resolution DPLL architecture
- State machine automatically transitions between tracking and freerun/digital-hold states
- Revertive or nonrevertive reference selection algorithm
- Programmable bandwidth from 14Hz to 500Hz
- Programmable tracking range (i.e. hold-in range)
- Output phase adjustment in 10ps steps
- High-resolution frequency and phase measurement
- Fast detection of input clock failure and transition to digital hold

## 2.3 APLL Features

- APLL with very high-resolution fractional scaling (i.e. non-integer) per channel
- Any-to-any frequency conversion with 0ppm error
- Two high-speed dividers (integers 4 to 15, half divides 4.5 to 7.5)
- Easy-to-configure, completely encapsulated design requires no external VCXO or loop filter components
- Bypass mode supports system testing

## 2.4 Output Clock Features

- Three low-jitter output clocks per channel
- Each output can be one differential output or two CMOS outputs
- Output clocks can be any frequency from 1Hz to 1035MHz (250MHz max for CMOS and HSTL outputs)
- Output jitter is typically 0.16 to 0.28ps RMS (12kHz to 20MHz)
- In CMOS mode, an additional divider allows the OCxN pin to be an integer divisor of the OCxP pin (example: OC3P 125MHz, OC3N 25MHz)
- Outputs easily interface with CML, LVDS, LVPECL, HSTL, SSTL, HCSL and CMOS components
- Supported telecom frequencies include PDH, SDH, Synchronous Ethernet, OTN
- Can produce clock frequencies for microprocessors, ASICs, FPGAs and other components
- Can produce PCIe clocks (PCIe gen. 1, 2 and 3)
- Sophisticated output-to-output phase alignment
- Per-output phase adjustment with high resolution and unlimited range
- Per-output enable/disable
- Per-output glitchless start/stop (stop high or low)

## 2.5 General Features

- SPI or $I^2C$ serial microprocessor interface per channel
- Automatic self-configuration at power-up from internal EEPROM memory; pin control to specify one of four stored configurations
- Each channel can be configured for numerically controlled oscillator (NCO) mode, which allows system software to steer frequency with resolution better than 0.01ppb
- Spread-spectrum modulation mode (meets PCI Express requirements)
- Zero-delay buffer configuration using an external feedback path
- Four general-purpose I/O pins per channel each with many possible status and control options
- Output frame sync signals
- Each channel's local oscillator can be fundamental-mode crystal or low-cost XO
- Internal compensation for local oscillator frequency error

## 2.6 Evaluation Software

- Simple, intuitive Windows-based graphical user interface
- Supports all device features and register fields
- Makes lab evaluation of the ZL30255 quick and easy
- Generates configuration scripts to be stored in internal EEPROM
- Generates full or partial configuration scripts to be run on a system processor
- Works with or without a ZL30255 evaluation board

## 3. Pin Diagram

The device is packaged in a 5x10mm 64-pin LGA.

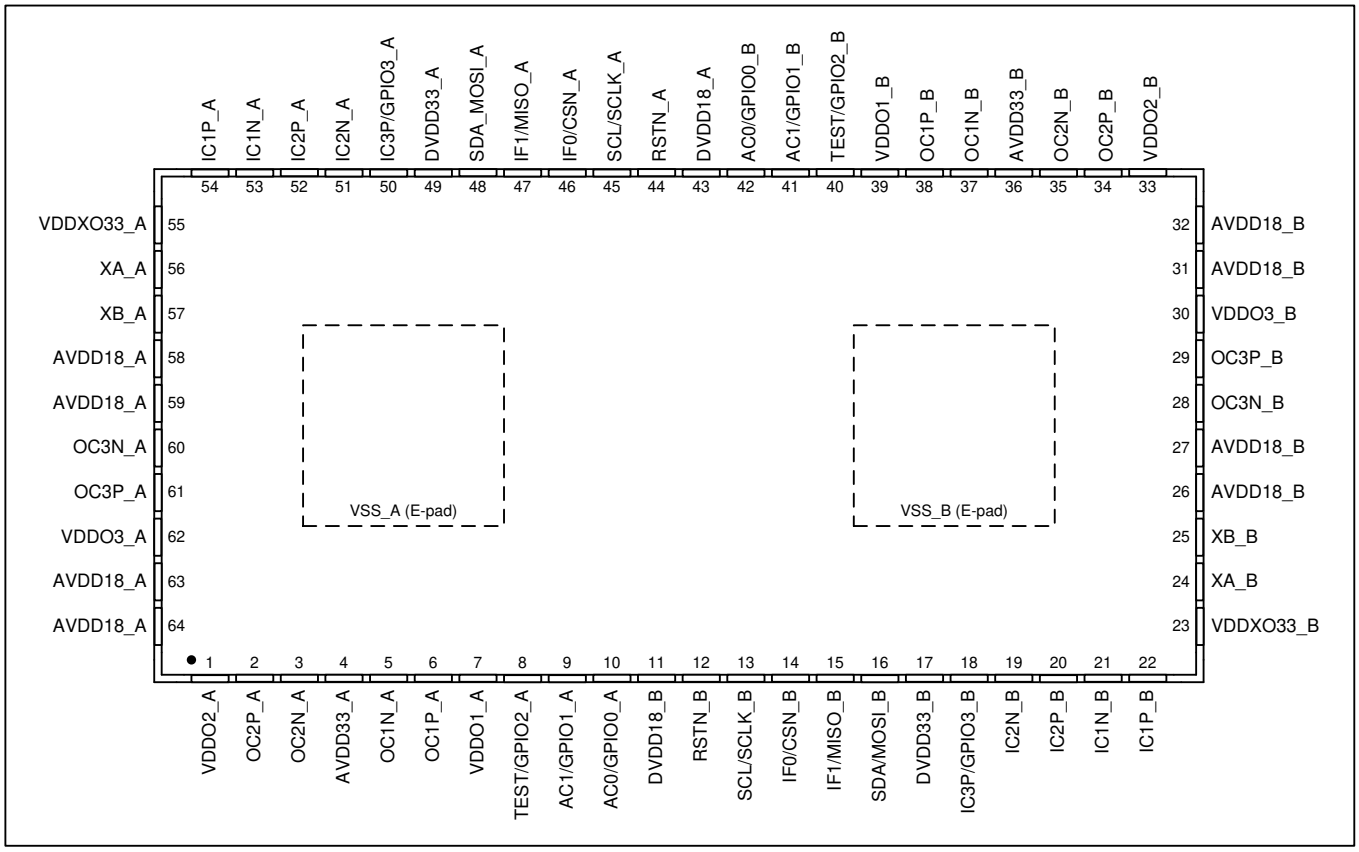


**Figure 4 - Pin Diagram**

## 4. Pin Descriptions

Channel A pins names have "_A" suffix. Channel B pin names have "_B" suffix. All inputs and outputs are LVCMOS unless described otherwise. The Type column uses the following symbols: I – input, $I_{PU}$ – input with 50kΩ internal pullup resistor, O – output, A – analog, P – power supply pin. All GPIO and SPI/$I^2C$ interface pins have Schmitt-trigger inputs and have output drivers that can be disabled (high impedance).

**Table 1 - Pin Descriptions**

| Pin # | Name | Type | Description |
|---|---|---|---|
| 54<br>53<br>52<br>51<br>50<br><br>22<br>21<br>20<br>19<br>18 | IC1P_A<br>IC1N_A<br>IC2P_A<br>IC2N_A<br>IC3P/GPIO3_A<br><br>IC1P_B<br>IC1N_B<br>IC2P_B<br>IC2N_B<br>IC3P/GPIO3_B | I<br>I<br>I<br>I<br>I/O<br><br>I<br>I<br>I<br>I<br>I/O | **Input Clock Pins**<br>Differential or Single-ended signal format. Programmable frequency.<br><br>*Differential:* See Table 10 for electrical specifications, and see Figure 21 for recommended external circuitry for interfacing these differential inputs to LVDS, LVPECL, CML or HSCL output pins on neighboring devices.<br><br>*Single-ended:* For input signal amplitude >2.5V, connect the signal directly to ICxP pin. For input signal amplitude ≤2.5V, AC-coupling the signal to ICxP is recommended. Connect the N pin to a capacitor (0.1μF or 0.01μF) to VSS. As shown in Figure 21, the ICxP and ICxN pins are internally biased to approximately 1.3V. Treat the ICxN pin as a sensitive node; minimize stubs; do not connect to anything else including other ICxN pins.<br><br>*Unused:* Set ICEN.ICxEN=0. The ICxP and ICxN pins can be left floating.<br><br>Note that the IC3N pin is not bonded out. A differential signal can be connected to IC3P by AC-coupling the POS trace to IC3P and terminating the signal on the driver side of the coupling cap. If not needed as an input clock pin, IC3P can behave as general-purpose I/O pin GPIO3, which is configured by GPIOCR2. Its state is indicated in GPIOSR. |
| 56<br>57<br><br>24<br>25 | XA_A<br>XB_A<br><br>XA_B<br>XB_B | A / I | **Crystal or Master Clock Pins**<br>*Crystal:* MCR1.XAB=01. An on-chip crystal driver circuit is designed to work with an external crystal connected to the XA and XB pins. See section 5.5.2 for crystal characteristics and recommended external components.<br><br>*Master Clock:* MCR1.XAB=10. An external local oscillator or clock signal (93-130MHz) can be connected to the XA pin. The XB pin must be left unconnected. |
| 6<br>5<br>2<br>3<br>61<br>60<br><br>38<br>37<br>34<br>35<br>29<br>28 | OC1P_A<br>OC1N_A<br>OC2P_A<br>OC2N_A<br>OC3P_A<br>OC3N_A<br><br>OC1P_B<br>OC1N_B<br>OC2P_B<br>OC2N_B<br>OC3P_B<br>OC3N_B | O | **Output Clock Pins**<br>CML, HSTL or 1 or 2 CMOS. Programmable frequency and drive strength.<br>See Table 11 and Figure 23 for electrical specifications and recommended external circuitry for interfacing to LVDS, LVPECL or CML input pins on neighboring devices.<br>See Table 12 for electrical specifications for interfacing to CMOS and HSTL inputs on neighboring devices.<br>See Figure 24 for recommended external circuitry for interfacing to HCSL inputs on neighboring devices. |
| 44<br><br>12 | RSTN_A<br><br>RSTN_B | $I_{PU}$ | **Reset (Active Low).**<br>When this global asynchronous reset is pulled low, all of the channel's internal circuitry is reset to default values. The channel is held in reset as long as RSTN is low. Minimum low time is 100ns. |

**Table 1 - Pin Descriptions (continued)**

| Pin # | Name | Type | Description |
|---|---|---|---|
| 10<br>9<br><br>42<br>41 | AC0/GPIO0_A<br>AC1/GPIO1_A<br><br>AC0/GPIO0_B<br>AC1/GPIO1_B | I/O | **Auto-Configure [1:0] / General Purpose I/O 0 and 1**<br><br>*Auto Configure:* On the rising edge of RSTN these pins behave as AC[1:0] and specify one of the configurations stored in EEPROM. See section 5.4.<br><br>*General-Purpose I/O:* After reset these pins are GPIO0 and GPIO1. GPIOCR1 configures the pins. Their states are indicated in GPIOSR. |
| 8<br><br>40 | TEST/GPIO2_A<br><br>TEST/GPIO2_B | I/O | **Factory Test / General Purpose I/O 2**<br><br>*Factory Test:* On the rising edge of RSTN the pin behaves as TEST. Factory test mode is enabled when TEST is high. For normal operation TEST must be low on the rising edge of RSTN.<br><br>*General-Purpose I/O:* After reset this pin is GPIO2. GPIOCR2 configures the pin. It state is indicated in GPIOSR. |
| 46<br><br>14 | IF0/CSN_A<br><br>IF0/CSN_B | I/O | **Interface Mode 0 / SPI Chip Select (Active Low)**<br><br>*Interface Mode:* On the rising edge of RSTN the pin behaves as IF0 and, together with IF1, specifies the interface mode for the channel. See section 5.4.<br><br>*SPI Chip Select:* After reset this pin is CSN. When the channel is configured as a SPI slave, an external SPI master must assert (low) CSN to access channel registers. |
| 45<br><br>13 | SCL/SCLK_A<br><br>SCL/SCLK_B | I/O | **$I^2C$ Clock / SPI Clock**<br><br>*$I^2C$ Clock:* When the channel is configured as an $I^2C$ slave, an external $I^2C$ master must provide the $I^2C$ clock signal on the SCL pin.<br><br>*SPI Clock:* When the channel is configured as a SPI slave, an external SPI master must provide the SPI clock signal on SCLK. |
| 47<br><br>15 | IF1/MISO_A<br><br>IF1/MISO_B | I/O | **Interface Mode 1 / SPI Master-In-Slave-Out**<br><br>*Interface Mode:* On the rising edge of RSTN the pin behaves as IF1 and, together with IF0, specifies the interface mode for the channel. See section 5.4.<br><br>*SPI MISO:* After reset this pin is MISO. When the channel is configured as a SPI slave, the channel outputs data to an external SPI master on MISO during SPI read transactions. |
| 48<br><br>16 | SDA/MOSI_A<br><br>SDA/MOSI_B | I/O | **$I^2C$ Data / SPI Master-Out-Slave-In**<br><br>*$I^2C$ Data:* When the channel is configured as an $I^2C$ slave, SDA is the bidirectional data line between the channel and an external $I^2C$ master.<br><br>*SPI MOSI:* When the channel is configured as a SPI slave, an external SPI master sends commands, addresses and data to the channel on MOSI. |

**Table 1 - Pin Descriptions (continued)**

| Pin # | Name | Type | Description |
|---|---|---|---|
| 58, 59, 63, 64<br><br>26, 27, 31, 32 | AVDD18_A<br><br>AVDD18_B | P | **Analog Power Supply.** 1.8V ±5%. |
| 4<br>36 | AVDD33_A<br>AVDD33_B | P | **Analog Power Supply. 3.3V ±5%.** |
| 43<br>11 | DVDD18_A<br>DVDD18_B | P | **Digital Power Supply. 1.8V ±5%.** |
| 49<br>17 | DVDD33_A<br>DVDD33_B | P | **Digital Power Supply. 3.3V ±5%.** |
| 7 | VDDO1_A | P | **Output OC1_A Power Supply. 1.5V to 3.3V ±5%.** |
| 1 | VDDO2_A | P | **Output OC2_A Power Supply. 1.5V to 3.3V ±5%.** |
| 62 | VDDO3_A | P | **Output OC3_A Power Supply. 1.5V to 3.3V ±5%.** |
| 39 | VDDO1_B | P | **Output OC1_B Power Supply. 1.5V to 3.3V ±5%.** |
| 33 | VDDO2_B | P | **Output OC2_B Power Supply. 1.5V to 3.3V ±5%.** |
| 30 | VDDO3_B | P | **Output OC3_B Power Supply. 1.5V to 3.3V ±5%.** |
| 55 | VDDXO33_A | P | **Analog Power Supply for channel A Crystal Driver Circuitry. 3.3V ±5%.** |
| 23 | VDDXO33_B | P | **Analog Power Supply for channel B Crystal Driver Circuitry. 3.3V ±5%.** |
| E-pad | VSS_A | P | **Ground. 0 Volts. For channel A.** |
| E-pad | VSS_B | P | **Ground. 0 Volts. For channel B.** |

# 5. Functional Description

## 5.1 Overview: Two Completely Independent Channels

The device is composed of two identical but completely independent *channels*, A and B. Each channel has its own independent set of pins: input clocks, output clocks, SPI/$I^2C$ interface, GPIO, reset and power supply pins. The two channels have identical pin lists except channel A pin names end with "_A" while channel B pins names end with "_B". Most of this data sheet uses pin names without these "_A" and "_B" endings, for example mentioning the TEST/GPIO2 pin rather than channel A's TEST/GPIO2_A pin or channel B's TEST/GPIO2_B pin. In this document any mention of a pin name without a channel suffix should be understood to apply to that pin in both channels.

The two channels also have identical register maps (shown in section 6.2). Channel A's registers are only accessible through channel A's SPI/$I^2C$ interface, and channel B's registers are only accessible through channel B's SPI/$I^2C$ interface. Each channel's register map starts at address 0.

Since the two channels are completely independent, they can be reset independently and configured independently. For example channel A can be configured for regular DPLL+APLL operation with the DPLL locked to channel A's IC1 input while channel B is configured for NCO mode where its output frequency is dynamically controlled by system software.

## 5.2 Device Identification

The 12-bit read-only ID field and the 4-bit revision field are found in each channel's ID1 and ID2 registers. Contact the factory to interpret the revision value and determine the latest revision.

## 5.3 Top-Level Channel Configuration

Each channel has two fundamental modes of operation: APLL-only and DPLL+APLL.

### 5.3.1 APLL-Only Mode

In APLL-only mode, the channel's input block and DPLL are powered down (PLLEN.DPLLEN=0), and the channel operates as a high-resolution fractional-N APLL. This reduces chip power consumption as shown in Table 7.

The bandwidth of the APLL is approximately 600kHz and therefore in APLL-only mode the channel does not behave as a jitter filter. This means that, in applications where output signals must have sub-ps jitter, the APLL input signal must have sub-ps jitter. In addition, features of the input block and the DPLL including activity monitoring, frequency monitoring and jitter filtering are not available. APLL-only mode is enabled when the APLL input mux is set to select an input other than the DPLL output (i.e. APLLCR3.APLLMUX=0xx).

APLL-only mode has two usage cases. First, the APLL can be locked to an external crystal as shown in Figure 5 for frequency synthesis applications. Second, the APLL can be locked to any of the four input clock signals, as shown in Figure 6 for frequency conversion applications.

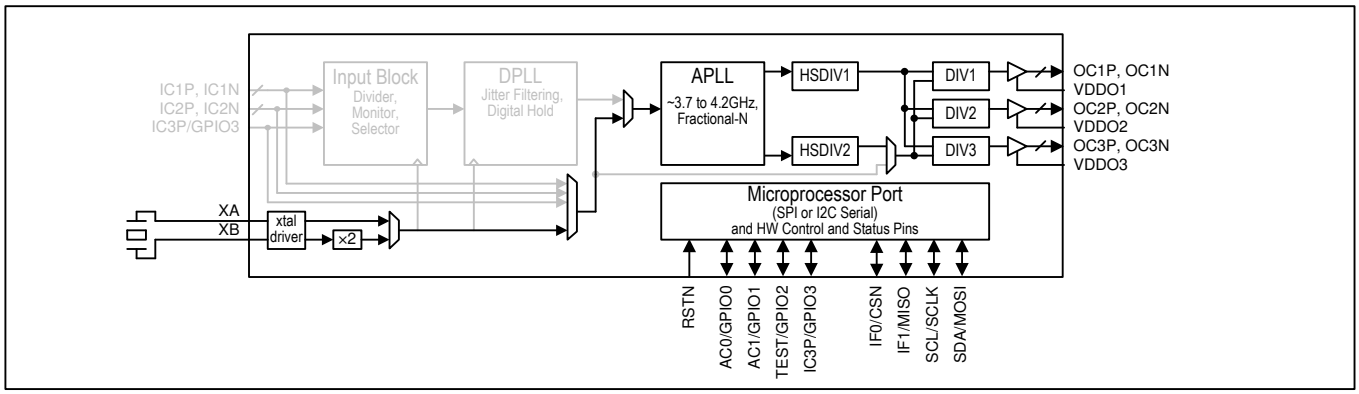


**Figure 5 - APLL-Only Mode: Clock Synthesis from a Crystal**

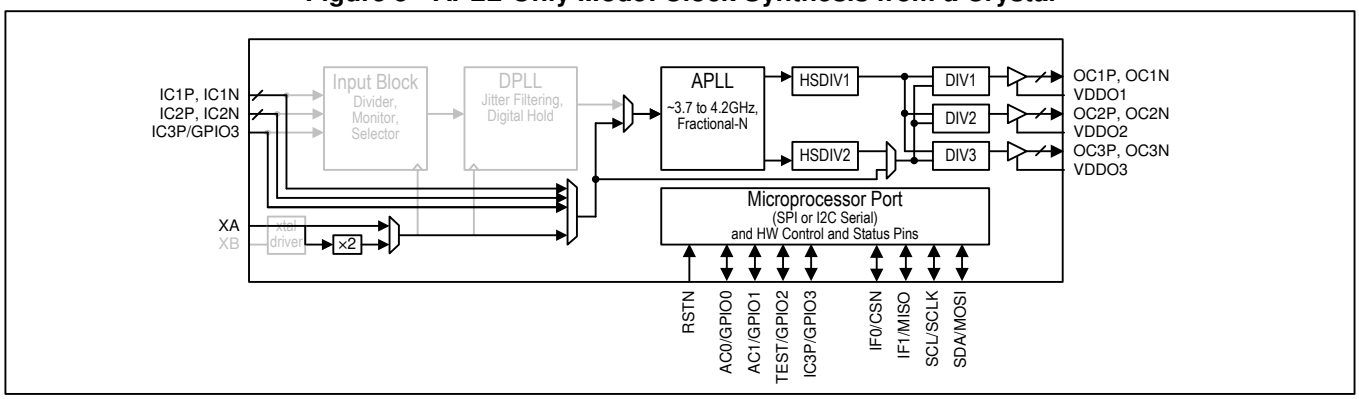


**Figure 6 - APLL-Only Mode: Frequency Conversion from One of Four Input Clocks**

### 5.3.2 DPLL+APLL Mode

In DPLL+APLL mode, the input block and DPLL are enabled and used. In this mode all input block features are available including activity monitoring, frequency monitoring and automatic reference switching. In addition, all DPLL features are available as well, including bandwidths low enough to filter jitter on the input clock signals. Channel power consumption is slightly higher than APLL-only mode.

DPLL+APLL mode is enabled when the APLL input mux is set to select the DPLL output (i.e. APLLCR3.APLLMUX=11x) and the input block and DPLL are enabled (PLLEN.DPLLEN=1).

DPLL+APLL mode includes the following three operating modes:

- Jitter Attenuation mode: The DPLL locks to a jittery input clock signal on IC1, IC2 or IC3 and attenuates (filters) the jitter. The channel outputs low-jitter clocks on its OCx outputs.

- Numerically Controlled Oscillator (NCO) mode: The input block and most of the DPLL are shut down, and system software controls the DPLL's output frequency through register writes.
- Spread-Spectrum mode: The input block and most of the DPLL are shut down, and a spread-spectrum hardware block modulates the DPLL's output frequency at a specified modulation rate over a specified frequency range. This is typically used for as an EMI reduction strategy.

In DPLL+APLL mode the input block and the DPLL must operate from a master clock signal that is approximately 100MHz, 114.285MHz or 125MHz (see MCR2.MCLK for exact frequency ranges). DPLL+APLL mode has two usage cases. First, the master clock can be a clock signal on the XA pin, optionally doubled by the clock doubler, as shown in Option 1 in Figure 7. Second, the master clock can come from an external crystal, the internal crystal driver circuit, and the clock doubler as shown in Option 2 in Figure 7. This second use case requires a crystal frequency between 46.5MHz and 60MHz and the clock doubler in order to get a valid master clock frequency.

Note that the clock doubler can be used with an external XO in NCO and spread-spectrum modes, but the clock doubler generally should not be used with an external XO in jitter attenuation mode. See section 5.5.3 for more details.

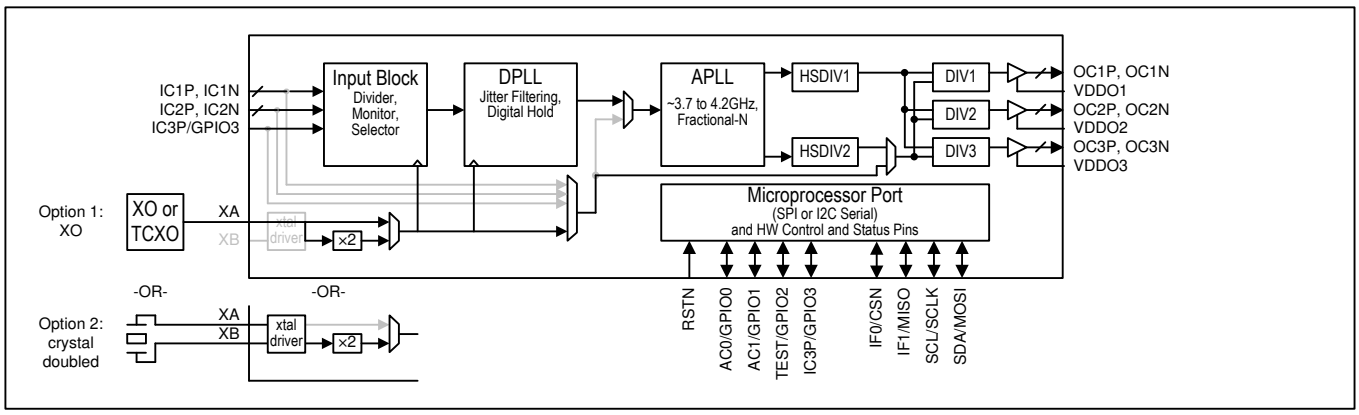


**Figure 7 - DPLL+APLL Mode: Locked to One of Three Inputs, Master Clock from XO or Crystal**

### 5.3.3 Evaluation Software for Device Configuration

Microsemi provides evaluation software that gives the user a simple, intuitive graphical user interface in which to generate complete device configurations. Often customers can generate base device configurations with the evaluation software without learning the device register set in detail. This saves time and money during system development. Use of the evaluation software is required as explained in sections 5.8.1 and 5.7.2.3.

## 5.4 Pin-Controlled Automatic Configuration at Reset

The channel configuration is determined at reset (i.e. on the rising edge of RSTN) by the signal levels on five channel pins: TEST/GPIO2, AC1/GPIO1, AC0/GPIO0, IF1/MISO and IF0/CSN. For these pins, the first name (TEST, AC1, AC0, IF1, IF0) indicates their function when they are sampled by the rising edge of the RSTN pin. The second name refers to their function after reset. The values of these pins are latched into the CFGSR register when RSTN goes high. To ensure the device properly samples the reset values of these pins, the following guidelines should be followed:

1. Any pullup or pulldown resistors used to set the value of these pins at reset should be 1kΩ.
2. RSTN must be asserted at least as long as specified in section 5.13.

The hardware configuration pins are grouped into three sets:

1. TEST - Manufacturing test mode
2. IF[1:0] – Microprocessor interface mode and $I^2C$ address
3. AC[1:0] – Auto-configuration from EEPROM

The TEST pin selects manufacturing test modes when TEST=1 (the AC[1:0] pins specify the test mode). TEST=1 and AC[1:0]=00 configures the part so that production SPI EEPROM programmers can program the internal EEPROM. See section 5.15.2 for more information.

Each channel's IF[1:0] pins specify the processor interface mode and the $I^2C$ address:

| IF1 | IF0 | Processor Interface |
|---|---|---|
| 0 | 0 | $I^2C$, channel A slave address 10000 00<br>channel B slave address 10001 00 |
| 0 | 1 | $I^2C$, channel A slave address 10000 01<br>channel B slave address 10001 01 |
| 1 | 0 | $I^2C$, channel A slave address 10000 10<br>channel B slave address 10001 10 |
| 1 | 1 | SPI Slave |

The AC[1:0] pins specify which of four configurations in the EEPROM to execute after reset:

| AC1 | AC0 | Auto Configuration |
|---|---|---|
| 0 | 0 | Configuration 0 |
| 0 | 1 | Configuration 1 |
| 1 | 0 | Configuration 2 |
| 1 | 1 | Configuration 3 |

For more information about auto-configuration from EEPROM see section 5.15.

## 5.5 Local Oscillator or Crystal

Section 5.3 describes several channel configurations that make use of either an external local oscillator (XO or TCXO) or an external crystal. Section 5.5.1 describes how to connect an external oscillator and the required characteristics of the oscillator. Section 5.5.2 describes how to connect an external crystal to the on-chip crystal driver circuit and the required characteristics of the crystal. The channels can both be provided with the same oscillator signal, but the channels cannot share a crystal.

### 5.5.1 External Oscillator

A signal from an external oscillator can be connected to the XA pin (XB must be left unconnected). Table 9 specifies the range of possible frequencies for the XA input. Several vendors including Vectron, Rakon and TXC offer low-cost, low-jitter XOs with output frequencies in this range. In DPLL+APLL jitter attenuation mode the frequency of the external oscillator must be specified in the MCR2.MCLK field. To minimize jitter, the signal must be properly terminated and must have very short trace length. A poorly terminated single-ended signal can greatly increase output jitter, and long single-ended trace lengths are more susceptible to noise. To connect one oscillator to both channels' XA pins, Microsemi recommends two identical, equal-length routes with identical source-series resistors of 20-30Ω. When MCR1.XAB=10, XA is enabled as a single-ended input.

In DPLL+APLL mode, the stability of the DPLL in freerun or digital hold is equivalent to the stability of the external oscillator. While many applications can make use of a simple XO component, some applications may require the stability of a TCXO. Contact Microsemi timing products technical support for recommended oscillator components.

While the stability of the external oscillator can be important, its absolute frequency accuracy is less important because any known frequency inaccuracy of the oscillator can be compensated. When the channel is configured for DPLL+APLL mode, the DPLL's DFREQZ parameter can be used to compensate for oscillator frequency error. When the channel is configured for APLL-only mode, the APLL's fractional feedback divider value (AFBDIV) can be adjusted by ppb or ppm to compensate for oscillator frequency error.

The jitter on output clock signals depends on the phase noise and frequency of the external oscillator. For a channel to operate with the lowest possible output jitter, the external oscillator should have the following characteristics:

- Phase Jitter: less than 0.1ps RMS over the 12kHz to 5MHz integration band

- Frequency: The higher the better, all else being equal

### 5.5.2 External Crystal and On-Chip Driver Circuit

Each channel's on-chip crystal driver circuit is designed to work with a fundamental mode, AT-cut crystal resonator. See Table 2 for recommended crystal specifications. To enable the crystal driver, set MCR1.XAB=01.

See Figure 8 for the crystal equivalent circuit and the recommended external capacitor connections. To achieve a crystal load ($C_L$) of 10pF, an external 16pF is placed in parallel with the 4pF internal capacitance of the XA pin, and an external 16pF is placed in parallel with the 4pF internal capacitance of the XB pin. The crystal then sees a load of 20pF in series with 20pF, which is 10pF total load. Note that the 16pF capacitance values in Figure 8 include all capacitance on those nodes. If, for example, PCB trace capacitance between crystal pin and IC pin is 2pF then 14pF capacitors should be used to make 16pF total.

The crystal, traces, and two external capacitors should be placed on the board as close as possible to the XA and XB pins to reduce crosstalk of active signals into the oscillator. Also no active signals should be routed under the crystal circuitry.

Note: Crystals have temperature sensitivities that can cause frequency changes in response to ambient temperature changes. In applications where significant temperature changes are expected near the crystal, it is recommended that the crystal be covered with a thermal cap, or an external XO or TCXO should be used instead.

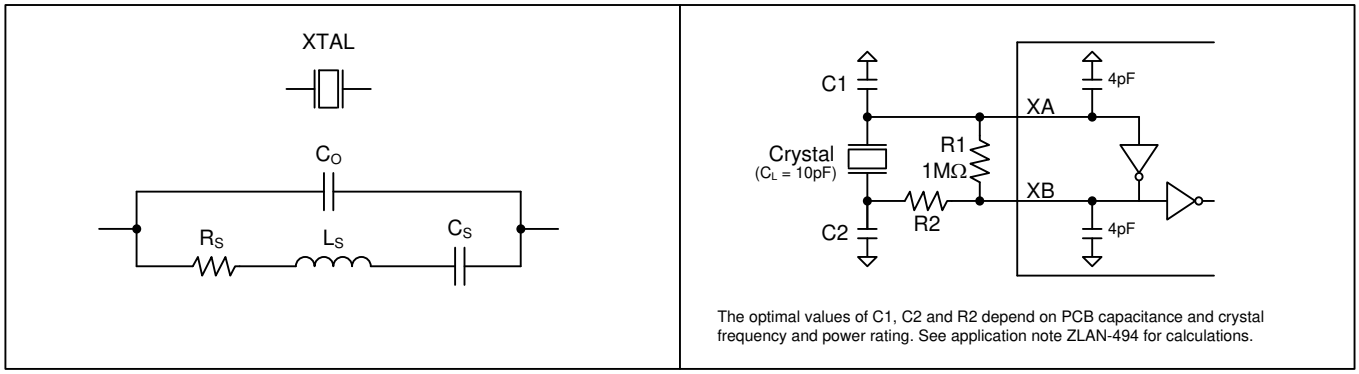


**Figure 8 - Crystal Equivalent Circuit / Recommended Crystal Circuit**

**Table 2 - Crystal Selection Parameters**

| Parameter | | Symbol | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| Crystal oscillation frequency[1] | | $f_{OSC}$ | 25 | | 60 | MHz |
| Shunt capacitance | | $C_O$ | | 2 | 5 | pF |
| Load capacitance | | $C_L$ | | 10 | | pF |
| Equivalent series resistance (ESR)[2] | $f_{OSC}$ < 40MHz | $R_S$ | | | 60 | Ω |
| | $f_{OSC}$ > 40MHz | $R_S$ | | | 50 | Ω |
| Maximum crystal drive level | | | 100 | | | µW |

Note 1: Higher frequencies give lower output jitter, all else being equal.
Note 2: These ESR limits are chosen to constrain crystal drive level to less than 100µW. If the crystal can tolerate a drive level greater than 100µW then proportionally higher ESR is acceptable.

| Parameter | Symbol | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| Crystal Frequency Stability vs. Power Supply | $f_{FVD}$ | | 0.2 | 0.5 | ppm per 10% Δ in VDD |

Any known frequency inaccuracy of the crystal can be compensated in the DPLL or in the APLL. When a channel is configured for DPLL+APLL mode, the DPLL's DFREQZ field can be used to compensate for crystal frequency error. When a channel is configured for APLL-only mode, the APLL's fractional feedback divider value (AFBDIV) can be adjusted by ppb or ppm to compensate for crystal frequency error.

### 5.5.3 Clock Doubler

Figure 1 shows an optional clock doubler ("x2" block) following the crystal driver block. The doubler, which is enabled by setting MCR1.DBL=1, can be used to double the frequency of the internal crystal driver circuit or a clock signal on the XA pin. The following table shows scenarios when the clock doubler can be used.

| Channel Mode | With Crystal | With XO or Clock Signal |
|---|---|---|
| APLL-Only, Integer Multiply | Maybe[1] | Maybe[1] |
| APLL-Only, Fractional Multiply | Yes | Yes |
| DPLL+APLL Jitter Attenuation | Yes | Not Recommended |
| DPLL+APLL NCO | Yes | Yes |
| DPLL+APLL Spread-Spectrum | Yes | Yes |

Note 1: For APLL integer multiplication, use of the doubler is application-dependent. On the positive side, use of the doubler reduces random jitter. On the negative side, the doubler causes a large spur at the XA frequency (but this spur may be outside the band of interest for the application).

### 5.5.4 Ring Oscillator (for System Start-Up)

To ensure that registers can be written immediately after system start-up, in its power-on reset state each channel operates its registers and processor interface from an internal ring oscillator.

When operating a channel in DPLL+APLL mode, as soon as the external oscillator connected to the XA pin has stabilized and is ready to use, the MCR1.MCSEL bit must be set to source the DPLL master clock from XA. If the ring oscillator causes undesirable spurs it can be disabled (powered down) by setting MCR1.ROSCD=1.

When operating the part in APLL-only mode, a master clock signal on the XA pin is not required, and the ring oscillator is left enabled to provide a clock for the processor interface logic and registers.

## 5.6 Input Signal Format Configuration

Each channel's input clocks IC1, IC2 and IC3 are enabled by setting the enable bits in the ICEN register. The power consumed by a differential receiver is shown in Table 7. The electrical specifications for these inputs are listed in Table 10. Each input clock can be configured to accept nearly any differential signal format by using the proper set of external components (see Figure 21). To configure these differential inputs to accept single-ended CMOS signals, connect the single-ended signal to the ICxP pin, and connect the ICxN pin to a capacitor (0.1µF or 0.01µF) to VSS. Each ICxP and ICxN pin is internally biased to approximately 1.3V. If an input is not used, both ICxP and ICxN pins can be left floating. Note that the IC3N pin is not present. A differential signal can be connected to IC3P by AC-coupling the POS trace to IC3P and terminating the signal on the driver side of the coupling cap. If not needed as an input clock pin, IC3P can behave as general-purpose I/O pin GPIO3.

## 5.7 Input Block: Input Divider, Monitor and Selector

The input block performs the following functions:

- Frequency division to a frequency suitable for DPLL locking
- Activity monitoring
- Frequency monitoring
- DPLL input clock selection (automatic or manual)

Figure 9 is a block diagram of the input block. This block requires a master clock as described in section 5.3.2.

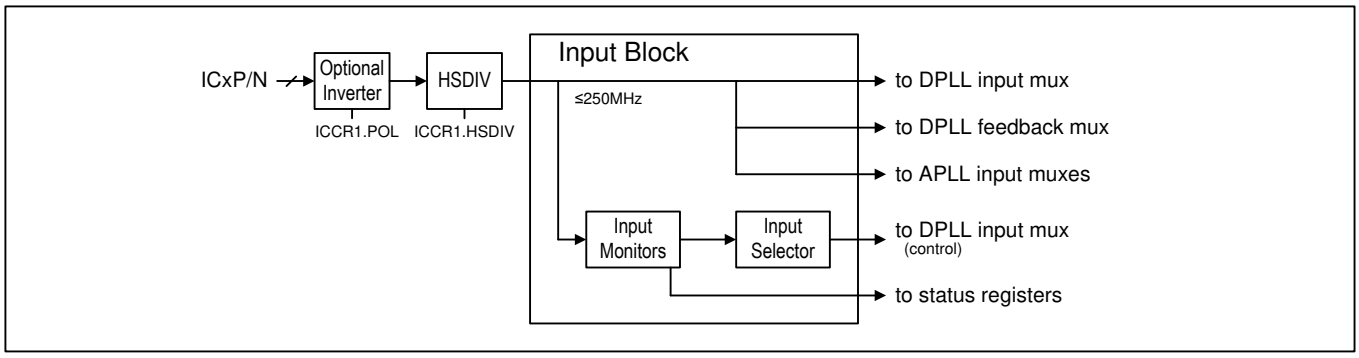


**Figure 9 - Input Block Diagram**

### 5.7.1 Input Clock Inversion and High-Speed Dividers

The input block tolerates a wide range of duty cycles out to a minimum high time or minimum low time of 3ns or 30% of the clock period, whichever is smaller.

Any frequency in the 1kHz to 250MHz range can be accepted by the input block. Important notes about the input block:

- ICxCR1.POL specifies the edge to which the DPLL will lock (by default, the rising edge).
- ICxCR1.HSDIV must be set correctly to reduce the clock frequency below 250MHz.
- In DPLL+APLL mode, the frequencies of all enabled ICx input clocks must divide by integers to a common DPLL phase-frequency detector (PFD) frequency ≥ 1kHz. In addition, the common PFD frequency must be ≥ 20 times the DPLL bandwidth.

### 5.7.2 Input Clock Monitoring

Each ICx input clock is continuously monitored for activity and frequency accuracy.

The activity monitor counts the number of input clock cycles that occur during a configurable interval. This provides the fastest detection when the input clock is stopped or far off frequency. Register bit ICxSR.ACVAL indicates the real-time status of this monitor. The ACVAL bit stays low when the input clock is not toggling or its frequency is grossly too high; ACVAL flickers (i.e. rapidly changes states) when the input clock is toggling but its frequency is grossly too low.

Frequency monitoring is handled by a percent frequency monitor (1% to 20% in 1% steps) Register bits ICxSR.PCVAL indicate the real-time status of the percent monitor.

Any input clock that fails activity monitoring or frequency monitoring is declared invalid. The valid/invalid state of each input clock is reported in the corresponding real-time status bit in the VALSR registers. When the valid/invalid state of a clock changes, the corresponding latched status bit is set in the VALSR registers. Input clocks manually marked invalid in the DPLL's VALCR1 register cannot be automatically selected as the reference for the DPLL.

The activity monitor and frequency monitor can be enabled and disabled using the ACEN and PCEN bits, respectively in the MONxCR2 register.

In addition to the monitors in the input block, the DPLL can also invalidate an input. If the input is the DPLL's selected reference and the DPLL cannot lock within the time specified by the PHLKTO register, the DPLL invalidates the input by setting the ICxSR.LKTO bit.

**Note:** The percent monitor cannot be used with a 1kHz input clock and has a narrower range of settings for input frequencies below 2kHz. See the evaluation software for the exact range of settings available for a particular channel configuration.

#### 5.7.2.1 External Monitoring

Some clock signals come from external components that can monitor the quality of a clock signal or the quality of a signal from which the clock signal is derived. One example is a BITS receiver in telecom equipment. This component receives a DS1, E1 or 2048kHz synchronization signal and recovers a clock from that signal. The BITS receiver monitors the incoming signal and can declare loss of signal (LOS), loss of frame alignment (LOF) and other defects in the incoming signal. Another example is a Synchronous Ethernet PHY, which receives an Ethernet signal and recovers a clock from that signal. This PHY can declare loss of lock, loss of codeword alignment and other defects.

When a neighboring component can detect that the incoming signal or the clock recovered from the signal is somehow out of specification, a loss-of-signal indication from that component can be connected to a GPIO pin to instantly invalidate the input clock. Any of the channel's GPIO pins can be used as a loss-of-signal indicator for any of the IC1, IC2 or IC3 input clocks. ICxSR.LOS indicates the real-time LOS status from the GPIO pin.

**Example**: Configure GPIO1 to be the active-low LOS signal for IC1:

GPIOCR1.GPIO1C=0001 (Configure GPIO1 to be an input with inversion)
MON1CR2.LOSSS=010 (Configure the IC1 monitor's LOS source to be GPIO1)

#### 5.7.2.2 Monitor Priority and Validation Timer

All enabled input monitors must declare an input valid for a configurable duration (which can be zero) before the input clock is validated and considered eligible for selection as the DPLL's selected reference. The monitors have a priority hierarchy in which an invalid declaration by a higher-priority monitor forces an invalid declaration in all lower-priority monitors. When a valid higher-priority monitor declares the input valid, the next lower priority monitor can then initiate its validation process. The monitor hierarchy is as follows:

- Input LOS from a GPIO pin forces all other monitors (activity and frequency) to declare the input invalid.
- When the activity monitor declares invalid, it forces the frequency monitors to declare invalid.

When a monitor is not enabled, it continually declares the input clock valid.

After all monitors declare an input clock valid (ICxSR.VAL=1) the validation timer requires all the monitors to continue to indicate the clock is valid for a configurable validation time before the input is declared valid for use as a DPLL input (ICxSR.VALT=1).

#### 5.7.2.3 Input Monitor Configuration

The device's input monitors are very sophisticated, but the configuration registers for these monitors are, generally speaking, low-level coefficients rather than user concepts such as ppm. As a result most input monitor registers are not documented in this data sheet. Instead, Microsemi provides evaluation software that gives the user a simple, intuitive graphical user interface in which to generate complete device configurations, including all aspects of input monitor behavior. Configuration files from the evaluation software can be stored in internal EEPROM to allow the channel to self-configure at reset. Alternately, system software can perform the register writes listed in the configuration files as needed to configure/reconfigure the channel.

### 5.7.3 Input Clock Priority, Selection and Switching for the DPLL

#### 5.7.3.1 Priority Configuration

During normal operation, the selected reference for the DPLL is chosen automatically based on the priority rankings assigned to the input clocks in the input priority registers (IPR1 and IPR2). The default input clock priorities are shown in Table 3.

Any unused input clock should be given the priority value 0, which disables the clock and marks it as unavailable for selection. Priority 1 is highest.

**Table 3 - Default Input Clock Priorities**

| INPUT CLOCK | DPLL DEFAULT PRIORITY |
|---|---|
| IC1 | 1 |
| IC2 | 2 |
| IC3 | 3 |

### 5.7.3.2 Automatic Selection

When ICSCR1.EXTSW=0, automatic input clock reference selection is used for the DPLL. The input reference selection algorithm chooses the highest-priority valid input clock to be the selected reference. The real-time valid/invalid state of each input clock is maintained in the VALSR registers (see section 5.7.2). The priority of each input clock is set as described in section 5.7.3.1. To select the DPLL input clock based on these criteria, the selection algorithm maintains a priority table of valid inputs. The top entry in this priority table and the selected reference are indicated in the PTAB1 register. The second- and third-priority inputs are indicated in the PTAB2 register.

If two or more input clocks are given the same priority number then those inputs are prioritized among themselves using a fixed circular list. If one equal-priority clock is the selected reference but becomes invalid then the next equal-priority clock in the list becomes the selected reference. If an equal-priority clock that is not the selected reference becomes invalid, it is simply skipped over in the circular list. The selection among equal-priority inputs is inherently nonrevertive, and revertive switching mode (see next paragraph) has no effect in the case where multiple equal-priority inputs have the highest priority.

An important input to the selection algorithm is the REVERT bit in the ICSCR1 register. In revertive mode (REVERT=1), if an input clock with a higher priority than the selected reference becomes valid, the higher priority reference immediately becomes the selected reference. In nonrevertive mode (REVERT=0), the higher priority reference does not immediately become the selected reference but does become the highest priority reference in the priority table (PTAB1.REF1). (The selection algorithm always switches to the highest-priority valid input when the selected reference goes invalid, regardless of the state of the REVERT bit.) For many applications, nonrevertive mode is preferred because it minimizes disturbances on the output clocks due to reference switching.

In nonrevertive mode, planned switchover to a newly-valid higher priority input clock can be done manually under software control. The validation of the new higher priority clock sets the corresponding latched status bit in the VALSR registers, which can drive an interrupt request if needed. System software can then respond to this change of state by briefly enabling revertive mode (toggling REVERT high then back low) to force the switchover to the higher priority clock.

### 5.7.3.3 Manual Selection

The bits of the VALCR1 register can be used to perform manual selection of an input clock. When all input clocks have non-zero priorities in the IPR registers, an input clock can be manually selected by setting the VALCR1 bit for that input clock to 1 and the VALCR1 bits for the other input clocks to 0.

### 5.7.3.4 External Reference Switching Mode

In this mode a GPIO pin controls reference switching between two input clocks. This mode is enabled by setting the ICSCR1.EXTSW=1. In this mode, if the GPIO pin is high, the DPLL is forced to lock to input IC1 (if the priority of IC1 is nonzero in IPR1) or IC3 (if the priority of IC1 is zero) whether or not the selected input has a valid reference signal. If the GPIO pin is low the DPLL is forced to lock to input IC2 whether or not IC2 has a valid reference signal. The GPIO pin is selected by MCR2.EXTSS.

In external reference switching mode the input selector logic behaves as a simple 2:1 mux, and the DPLL is forced to try to lock to the selected reference whether it is valid or not. This mode controls the PTAB1.SELREF field directly and, therefore, is not affected by the state of the ICSCR1.REVERT bit. During external reference switching mode, only PTAB1.SELREF is affected; the PTAB1.REF1 field continues to indicate the highest-priority valid input

chosen by the automatic selection logic. The priorities of IC1, IC2 and IC3 in the IPR registers must be non-zero for proper behavior in external reference switching mode.

## 5.8 DPLL Architecture and Configuration

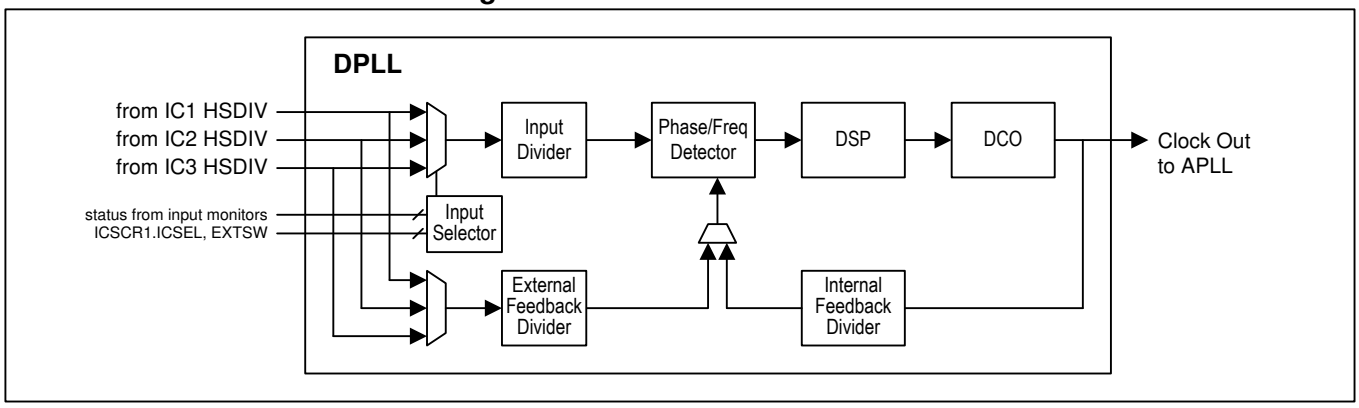


**Figure 10 - DPLL Block Diagram**

Digital PLLs have two key benefits: (1) stable, repeatable performance that is insensitive to process variations, temperature, and voltage; and (2) flexible behavior that is easily configured and reprogrammed. DPLLs use a digitally controlled oscillator (DCO) to generate the DPLL output clock. The DPLL output clock is then provided to an APLL for clock multiplication/frequency conversion.

The DPLL in each channel is configurable for many PLL parameters including bandwidth, input frequency, pull-in/hold-in range, input-to-output phase offset, and more. No knowledge of loop equations or gain parameters is required to configure and operate the device. No external components are required for the DPLL except a local oscillator connected to the XA pin to provide the DPLL's master clock (see section 5.3.2).

### 5.8.1 DPLL Configuration

The DPLL in each channel is very sophisticated, but the configuration registers for the DPLL are, generally speaking, very low-level coefficients rather than user concepts such as bandwidth and pull-in range. As a result most DPLL registers are not documented in this data sheet. Instead, Microsemi provides evaluation software that gives the user a simple, intuitive graphical user interface in which to generate complete device configurations, including all aspects of DPLL behavior. Configuration files from the evaluation software can be stored in internal EEPROM to allow the device to self-configure at reset. Alternately, system software can perform the register writes listed in the configuration files as needed to configure/reconfigure the device. The most frequently used DPLL status register fields and real-time control register fields are documented in section 6.3.6 and discussed in the DPLL sections below.

### 5.8.2 DPLL States

**Tracking (Locked and Unlocked).** When a valid input clock is available, the DPLL is in the tracking state (DSRR1.TRK=1) and is either locked to an input clock (DSRR1.LOL=0) or unlocked (DSRR1.LOL=1).

**Freerun/Digital-Hold.** When all input clocks become invalid, the DPLL enters the freerun/digital-hold state (DSRR1. TRK=0) in which it operates open-loop. In this mode the DPLL can be in freeun (DSRR1.HO=0) in which its output frequency has the same fractional frequency offset as the master clock signal. Or the DPLL can be in digital hold (DSRR1.HO=1) in which the output frequency has the same fractional frequency offset it had previously when the DPLL was locked to an input clock. The DPLL can automatically transition from the freerun/digital-hold state to the tracking state when an input clock is declared valid. The DPLLCR1.HOMODE field controls DPLL freerun/digital-hold behavior.

System software can manually force the DPLL into the freerun/digital-hold state as needed using the DPLLCR1.MODE field.

### 5.8.3 DPLL Capabilities

**Bandwidth.** The DPLL can be configured for any bandwidth from 14Hz to 500Hz.

**Pull-In/Hold-In Range.** The DPLL tracking range is configurable from ±1ppm to ±1000ppm. The DPLL reports when it has reached the limit of the range in the DSRR2.FLIM register bit. The DPLL's hold-in range is the same as its tracking range. The DPLL's pull-in range should be considered to be half the size of the tracking range for reasonable pull-in time. For example, when tracking range is ±1000ppm, pull-in range should be considered to be ±500ppm.

**Programmable Lock Criteria.** The DPLL has configurable criteria for defining when it declares lock. In addition to phase, the DPLL can also be configured to declare loss of lock when its fractional frequency offset exceeds the DPLL's tracking range.

**Programmable Phase Lock Timeout.** When the DPLL fails to lock to the selected input clock within the timeout duration specified by the PHLKTO register, the input is declared invalid by the input block, which sets the ICxSR.LKTO bit.

**Frequency and Phase Reporting.** The DPLL reports in real-time its frequency (i.e. fractional frequency offet in ppb/ppm vs. its nominal frequency) and its phase vs. the input clock signal. DPLL frequency resolution is better than 0.005ppb. DPLL frequency offset is reported in the DFREQ registers. DPLL phase is reported in the DPHASE registers.

**Numerically Controlled Oscillator (NCO) Mode.** In this mode most of the DPLL is shut down and system software controls the DPLL's output frequency using the 40-bit FREQZ field in the the DFREQZ registers. The resolution of frequency control is better than 0.01ppb. See section 5.8.7 for more details.

**Spread-Spectrum Modulation Mode.** For EMI-sensitive applications such as PCI Express, the device can perform spread spectrum modulation (SSM). In SSM the frequency of the output clock is continually varied over a narrow frequency range to spread the energy of the signal and thereby reduce EMI. See section **5.8.8** for more details.

### 5.8.4 Input Wander and Jitter Tolerance

Wander is tolerated up to the point where wander causes an apparent long-term frequency offset larger than the frequency threshold set in the input monitor. In such a situation the input clock would be declared invalid. Jitter can be tolerated up to the point of eye closure. The high-jitter input clock signal should be divided down to a lower frequency by the DPLL's input divider for high jitter tolerance.

### 5.8.5 Jitter and Wander Transfer

The transfer of jitter and wander from the selected reference to the output clocks has a programmable transfer function that is determined by the DPLL bandwidth. The -3dB corner frequency of the jitter transfer function can be set to any value from 14Hz to 500Hz.

During locked mode, the transfer of wander from the local oscillator clock (connected to the XA pin) to the output clocks is not significant as long as the DPLL bandwidth is set high enough to allow the DPLL to quickly compensate for oscillator frequency changes. During freerun/digital-hold, local oscillator wander has a much more significant effect. See section 5.5.1.

### 5.8.6 Output Jitter and Wander

Several factors contribute to jitter and wander on the output clocks, including:

- Jitter and wander amplitude on the selected reference (while in the locked state)
- The jitter/wander transfer characteristic of the channel (while in the locked state)
- The jitter and wander on the local oscillator clock signal (especially wander while in the freerun/digital-hold state)

The DPLL has programmable bandwidth (see Section 5.8.3). With respect to jitter and wander, the DPLL behaves as a low-pass filter with a programmable pole. The bandwidth of the DPLL is normally set low enough to strongly attenuate jitter. The wander and jitter attenuation depends on the DPLL bandwidth chosen.

### 5.8.7 Numerically Controlled Oscillator (NCO) Mode

In this mode of operation most of the DPLL is shut down, and system software controls the DPLL's output frequency using the 40-bit FREQZ field in the the DFREQZ registers. The resolution of frequency control is better than 0.01ppb.

The nominal FREQZ value, hereafter referred to as FREQZ0, is computed by the evaluation software for the desired channel configuration. When the FREQZ field is set to the FREQZ0 value, the channel's output clock frequencies have a fractional frequency offset of zero with respect to the NCO master clock signal applied to the XA pin.

(Fractional frequency offset (FFO) is defined as (actual_frequency – nominal_frequency) / nominal_frequency. FFO is a unitless number but is typically expressed in parts per billion (ppb), parts per million (ppm) or percent.)

To control the NCO, system software first reads the FREQZ0 value. FREQZ0 is a 40-bit unsigned integer.

To change the NCO frequency to a specific FFO (in ppm), system software calculates newFREQZ (a 40-bit unsigned integer) as follows:

newFREQZ = round(FREQZ0 * (1 + FFO/1e6))

System software then writes the newFREQZ value directly to the FREQZ field in the DFREQZ registers.

Note that any subsequent frequency changes are calculated using the same equation from the original FREQZ0 value and are not a function of the previous newFREQZ value. The value of newFREQZ should be kept within ±1000ppm of FREQZ0 and within ±500ppm of the previous newFREQZ value to avoid causing the APLL to lose lock. If spread spectrum modulation is also in use, the total frequency change caused by spread spectrum modulation and NCO control should be kept within ±5000ppm of FREQZ0 to avoid causing the APLL to lose lock.

### 5.8.8 Spread-Spectrum Modulation Mode

For EMI-sensitive applications such as PCI Express, the device can perform spread spectrum modulation (SSM). In SSM the frequency of the output clock is continually varied over a narrow frequency range to spread the energy of the signal and thereby reduce EMI.

This mode is a special case of NCO mode. Most of the DPLL is shut down, and spread-spectrum control circuitry modulates the DPLL's output frequency around the center frequency to perform SSM. The SS circuitry performs triangle-wave center-spread of up to ±0.5% deviation from the center frequency with modulation rate configurable from 25kHz to 55kHz.

Down-spread applications can be supported by converting them into center-spread. This is done by setting the DPLL's center frequency to be the center of the modulation range rather than the high end of range. For example, 100MHz with -1% downspread can be converted into ±0.5% center spread with center frequency of 100MHz/1.005=99.502488MHz.

In PCI Express applications the device can be used as a "point of load" spread-spectrum generator. In such an application, the 100MHz PCI Express clock signal without SSM can be generated centrally and distributed to various points in the system. A device positioned at one of those points can accept the 100MHz signal on its XA pin

and generate multiple 100MHz signals on its outputs. System software can then choose to enable or disable SSM in the device as needed to suit the needs of the application.

## 5.9 APLL Configuration

### 5.9.1 APLL Input Selection and Frequency

#### 5.9.1.1 APLL-Only Mode

In APLL-Only mode (APLLCR3.APLLMUX=0xx) the APLL can lock to any of inputs IC1 through IC3, a clock signal on XA or the crystal driver circuit (optionally clock-doubled) when a crystal is connected to XA and XB. See section 5.3.1 for details and diagrams.

The input to the APLL can be controlled by a GPIO pin or by the APLLCR3.APLLMUX register field. When APLLCR3.EXTSW=0, the APLLCR3.APLLMUX register field controls the APLL input mux.

When APLLCR3.EXTSW=1, a GPIO pin controls the APLL input mux. When the GPIO pin is low, the mux selects the input specified by APLLCR3.APLLMUX. When the GPIO pin is high, the mux selects the input specified by APLLCR3.ALTMUX. MCR2.EXTSS specifies which GPIO pin controls this behavior.

In APLL-only mode, the frequencies of all enabled input clocks (ICx and XA) must divide to a common APLL phase-frequency detector (PFD) frequency from 9.72MHz to 156.25MHz. In this mode the input high-speed dividers (ICxCR1.HSDIV) can be used to divide the ICx frequencies by 1, 2, 4 or 8. The XA pin does not have an internal divider, and, therefore, if XA is an enabled input clock then the XA frequency sets the APLL common PFD frequency. The polarity of an ICx input signal can be inverted by setting ICxCR1.POL.

#### 5.9.1.2 DPLL+APLL Mode

In DPLL+APLL mode (APLLCR3.APLLMUX=11x) the APLL locks to the DPLL output clock signal. Fractional multiplication in the APLL is used to generate the proper output clock frequency.

### 5.9.2 APLL Output Frequency

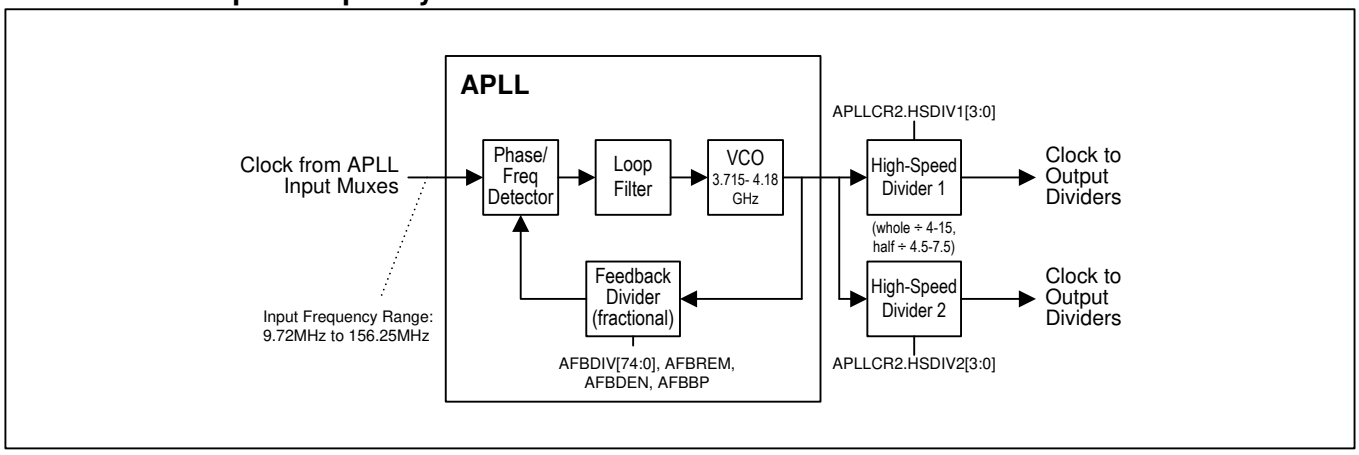


**Figure 11 - APLL Block Diagram**

The APLL is enabled when PLLEN.APLLEN=1. The APLL has a fractional-N architecture and therefore can produce output frequencies that are either integer or non-integer multiples of the input clock frequency. Figure 11 shows a block diagram of the APLL, which is built around an ultra-low-jitter multi-GHz VCO. Register fields AFBDIV, AFBREM, AFBDEN and AFBBP configure the frequency multiplication ratio of the APLL. The APLLCR2.HSDIV1 and HSDIV2 fields specify how the VCO frequency is divided down by the high-speed dividers. Dividing by six is the typical setting to produce 622.08MHz for SDH/SONET or 625MHz for Ethernet applications.

Internally, the exact APLL feedback divider value is expressed in the form AFBDIV + AFBREM / AFBDEN * $2^{-(33-AFBBP)}$. This feedback divider value must be chosen such that APLL_input_frequency * feedback_divider_value is in the operating range of the VCO (as specified in Table 13). The AFBDIV term is a fixed-point number with 9 integer bits in APLL-only mode (7 integer bits in DPLL+APLL mode) and a configurable number of fractional bits (up to 33, as specified by AFBBP). Typically AFBBP is set to 9 to specify that AFBDIV has 33 – 9 = 24 fractional bits. Using more than 24 fractional bits does not yield a detectable benefit. Using less than 12 fractional bits is not recommended.

The following equations show how to calculate the feedback divider values for the situation where the APLL should multiply the APLL input frequency by integer M and also fractionally scale by the ratio of integers N / D. In other words, VCO_frequency = input_frequency * M * N / D. An example of this is multiplying 77.76MHz from the DPLL by M=48 and scaling by N / D = 255 / 237 for forward error correction applications.

$$AFBDIV = trunc(M * N / D * 2^{24}) \quad (1)$$

$$lsb\_fraction = M * N / D * 2^{24} - AFBDIV \quad (2)$$

$$AFBDEN = D \quad (3)$$

$$AFBREM = round(lsb\_fraction * AFBDEN) \quad (4)$$

$$AFBBP = 33 - 24 = 9 \quad (5)$$

The trunc() function returns only the integer portion of the number. The round() function rounds the number to the nearest integer. In Equation (1), AFBDIV is set to the full-precision feedback divider value, M * N / D, truncated after the 24th fractional bit. In Equation (2) the temporary variable 'lsb_fraction' is the fraction that was truncated in Equation (1) and therefore is not represented in the AFBDIV value. In Equation (3), AFBDEN is set to the denominator of the original M * N / D ratio. In Equation (4), AFBREM is calculated as the integer numerator of a fraction (with denominator AFBDEN) that equals the 'lsb_fraction' temporary variable. Finally, in Equation (5) AFBBP is set to 33 – 24 = 9 to correspond with AFBDIV having 24 fractional bits.

When a fractional scaling scenario involves multiplying an integer M times multiple scaling ratios $N_1 / D_1$ through $N_n / D_n$, the equations above can still be used if the numerators are multiplied together to get $N = N_1 \times N_2 \times \ldots \times N_n$ and the denominators are multiplied together to get $D = D_1 \times D_2 \times \ldots \times D_n$.

The easiest way to calculate the exact values to write to the APLL registers is to use the evaluation software, available on the Microsemi website. This software can be used even when no evaluation board is attached to the computer.

Note: After the APLL's feedback divider settings are configured in register fields AFBDIV, AFBREM, AFBDEN and AFBBP, the APLL enable bit PLLEN.APLLEN should be changed from 0 to 1 to cause the APLL to reacquire lock with the new settings. The real-time lock/unlock status of the APLL is indicated by APLLSR.ALK and ALK2.

### 5.9.3 APLL Phase Adjustment

The phase of the APLL's output clock can be incremented or decremented by 1/8th of a VCO cycle. This phase step size is 30ps at maximum VCO frequency of 4180MHz and 33.7ps at minimum VCO frequency of 3715MHz. The APLLCR4.PDSS field specifies the phase decrement control signal, which can be the APLLCR4.DECPH bit or any of the four GPIOs. The APLLCR4.PISS field specifies the phase increment control signal, which can be the APLLCR4.INCPH bit or any of the four GPIOs. Phase is adjusted on every rising edge and every falling edge of the control signal. This phase adjustment affects the output of both high-speed dividers.

## 5.10 Output Clock Configuration

Each channel has three output clock signal pairs. Each output has individual divider, enable and signal format controls. In CMOS mode each signal pair can become two CMOS outputs, allowing a channel to have up to six

output clock signals. Also in CMOS mode, the OCxN pin can have an additional divider allowing the OCxN frequency to be an integer divisor of the OCxP frequency (example: OC3P 125MHz and OC3N 25MHz). The outputs in each channel can be aligned relative to each other and relative to a channel input signal, and the phases of output signals can be adjusted dynamically with high resolution and infinite range.

### 5.10.1 Output Enable, Signal Format, Voltage and Interfacing

To use an output, the output driver must be enabled by setting OCxCR2.OCSF≠0, and the per-output dividers must be enabled by setting the appropriate bit in the OCEN register. The per-output dividers include the medium-speed divider, the low-speed divider and the associated phase adjustment/alignment circuitry and start/stop logic.

Using the OCxCR2.OCSF register field, each output pair can be disabled or configured as a CML output, an HSTL output, or one or two CMOS outputs. When an output is disabled it is high impedance, and the output driver is in a low-power state. In CMOS mode, the OCxN pin can be disabled, in phase or inverted vs. the OCxP pin. In CML mode the normal 800mV $V_{OD}$ differential voltage is available as well as a half-swing 400mV $V_{OD}$. All of these options are specified by OCxCR2.OCSF. The clock to the output driver can inverted by setting OCxCR2.POL=1. The CMOS/HSTL output driver can be set to any of four drive strengths using OCxCR2.DRIVE.

Each output has its own power supply pin to allow CMOS or HSTL signal swing from 1.5V to 3.3V for glueless interfacing to neighboring components. If OCSF is set to HSTL mode then a 1.5V power supply voltage should be used to get a standards-compliant HSTL output. Note that differential (CML) outputs must have a power supply of 3.3V.

The differential outputs can be easily interfaced to LVDS, LVPECL, CML, HCSL, HSTL and other differential inputs on neighboring ICs using a few external passive components. See Figure 23 for examples.

### 5.10.2 Output Frequency Configuration

The frequency of each output is determined by the configuration of the APLL, the high-speed dividers and the per-output dividers. Each output can be connected to either high-speed divider 1 (HSDIV1) or 2 (HSDIV2) using the OCxCR3.DIVSEL field.

Each output has two output dividers, a 7-bit medium-speed divider (OCxCR1.MSDIV) and a 25-bit low-speed output divider (LSDIV field in the OCxDIV registers). These dividers are in series, medium-speed divider first then output divider. These dividers produce signals with 50% duty cycle for all divider values including odd numbers. The low-speed divider can only be used if the medium-speed divider is used (i.e. OCxCR1.MSDIV>0).

Since each output has its own independent dividers, each channel can output families of related frequencies that have an APLL HSDIV output frequency as a common multiple. For example, for Ethernet clocks, a 625MHz HSDIV output clock can be divided by four for one output to get 156.25MHz, divided by five for another output to get 125MHz, and divided by 25 for another output to get 25MHz. Similarly, for SDH/SONET clocks, a 622.08MHz HSDIV output clock can be divided by 4 to get 155.52MHz, by 8 to get 77.76MHz, by 16 to get 38.88MHz or by 32 to get 19.44MHz.

#### Two Different Frequencies in 2xCMOS Mode

When an output is in 2xCMOS mode it can be configured to have the frequency of the OCxN clock be an integer divisor of the frequency of the OCxP clock. Examples of where this can be useful:

- 125MHz on OCxP and 25MHz on OCxN for Ethernet applications
- 77.76MHz on OCxP and 19.44MHz on OCxN for SONET/SDH applications
- 25MHz on OCxP and 1Hz (i.e. 1PPS) on OCxN for telecom applications with Synchronous Ethernet and IEEE1588 timing

An output can be configured to operate like this by setting the LSDIV value in the OCxDIV registers to OCxP_freq / OCxN_freq - 1 and setting OCxCR3.LSSEL=0 and OCxCR3.NEGLSD=1. Here are some notest about this dual-frequency configuration option:

- In this mode only the medium speed divider is used to create the OCxP frequency. The low-speed divider is then used to divide the OCxP frequency down to the OCxN frequency. This means that the lowest OCxP frequency is the high-speed divider output frequency divided by 128.
- An additional constraint is that the medium-speed divider must be configured to divide by 6 or more (i.e. must have OCxCR1.MSDIV≥5).

### 5.10.3 Output Duty Cycle Adjustment

For output frequencies less than or equal to 141.666MHz, the duty cycle of the output clock can be modified using the OCxDC.OCDC register field. This behavior is only available when MSDIV>0 and LSDIV > 1. When OCDC = 0 the output clock is 50%. Otherwise the clock signal is a pulse with a width of OCDC number of MSDIV output clock periods. The range of OCDC can create pulse widths of 1 to 255 MSDIV output clock periods. When OCxCR2.POL=0, the pulse is high and the signal is low the remainder of the cycle. When POL=1, the pulse is low and the signal is high the remainder of the cycle.

Note that duty cycle adjustment is done in the low-speed divider. Therefore when OCxCR3.LSSEL=0 the duty cycle of the output is not affected. Also, when a CMOS output is configured with OCxCR3.LSSEL=0 and OCxCR3.NEGLSD=1, the OCxN pin has duty cycle adjustment but the OCxP pin does not. This allows a higher-speed 50% duty cycle clock signal to be output on the OCxP pin and a lower-speed frame/phase/time pulse (e.g. 2kHz, 8kHz or 1PPS) to be output on the OCxN pin at the same time.

An output configured for CMOS or HSTL signal format should not be configured to have a duty cycle with high time shorter than 2ns or low time shorter than 2ns.

### 5.10.4 Output Phase Adjustment and Phase Alignment

Each channel has flexible, high-resolution tools for managing the phases of the output clocks relative to one another. The key register fields for this are found in the PACR1 and PACR2 global configuration registers and the per-output OCxPH register.

Phase alignment and phase adjustment are done in the medium-speed dividers. Resoution is 0.5 periods (also known as unit intervals or UI) of the high-speed divider (HSDIV) output clock. For example, for an HSDIV output frequency of 800MHz, resolution is 625ps.

#### 5.10.4.1 Phase Adjustment

A phase adjustment is a phase change for an output relative to that output's most recent phase. To cause a channel to perform phase adjustment of an output clock, set PACR1.MODE=1, set OCxCR1.PHEN=1 to enable the output for phase adjustment, and write the phase adjustment amount to the output's OCxPH register. Then an arm/trigger methodology is used to cause the phase adjustment to happen.

The arm step tells the channel that it is enabled to perform the phase adjustment when it sees the trigger stimulus. The source of the arm signal is specified by PACR2.ARMSRC. Options include the 0-to-1 transition of the PACR1.ARM bit, APLL transition from unlocked to locked, DPLL transition from unlocked to locked, or a transition on one of the GPIO pins.

The source of the trigger signal is specified by PACR2.TRGSRC. Options include 0-to-1 transition of the PACR1.TRIG bit, APLL transition from unlocked to locked, DPLL transition from unlocked to locked, a rising edge of the DPLL input clock, or a transition on one of the GPIO pins. The trigger signal can be inverted by setting PACR1.TINV. With TINV=1, the same GPIO signal can arm on one edge and trigger on the opposite edge.