Application Note: AN01006

XS1-L Clock Frequency Control

This application note explains how the clocks for the xCORE Tile, the xCONNECT switch and reference clock are generated from a low frequency external clock and the internal phase locked loop (PLL) on XS1-L devices.

1 Overview

A low frequency external clock is used to drive the internal phase locked loop (PLL) of XS1-L devices and obtain the system clock. A number of system clock dividers are then used on the system clock to derive the clocks for the xCORE tiles, the switch and the reference clock.
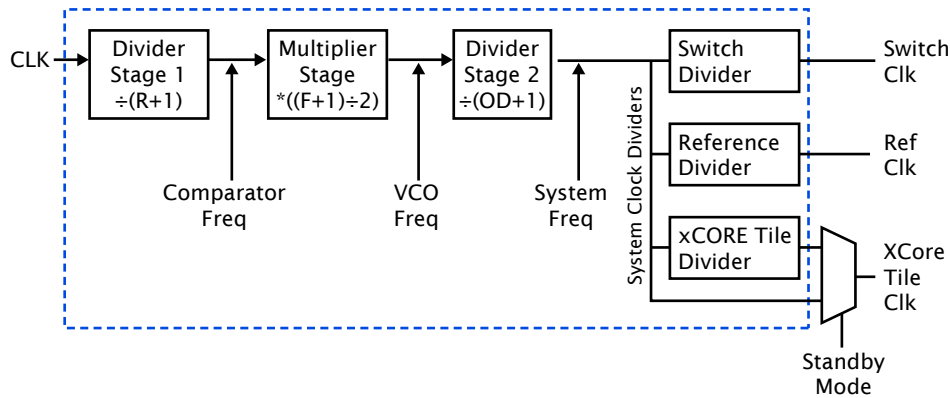

Figure 1: PLL and Clock Dividers

The PLL's initial settings are determined by the state of mode pins on the XS1-L device. The standard configuration allows a 20MHz external clock to be used to operate the xCORE tiles and the switch at 400MHz, and the reference clock at 100MHz. In many applications this configuration will be selected, requiring no reprogramming of the PLL or dividers. If the application requires a different input frequency or system frequency then the PLL must be reprogrammed. This results in the XS1-L device being reset to use the new PLL configuration. The xTIMEcomposer tools can be used to reprogram the PLL automatically by specifying the application's configuration in the XN file.

When the system clock dividers are reprogrammed to change the frequency to the xCORE tiles, the switch or the reference clock, no reset will occur.

The xCORE tile system clock divider can be bypassed so that the system clock is used as the tile clock. This allows the tile to be switched between ACTIVE and STANDBY modes.

2 Constaints

There are a number of constraints on the frequencies of clocks at different points on the XS1-L devices. These constraints must be met for the initial boot sequence, and if the PLL is reprogrammed, for the reprogrammed values too.

Clock	Constraint
CLK	4.22–100.0 MHz
Comparator frequency	1.59 kHz–1300 MHz
VCO frequency	260–1300 MHz
System clock	Maximum operating frequency—see XS1_L device datasheet
xCORE clock	System clock maximum
Switch clock	System clock maximum
Reference clock	System clock maximum

Table 1: Clock Frequency Constraints

2.1 PLL Settings

There are three dividers within the PLL. R divides the input clock down. The next divider divides the output of the voltage controlled oscillator (VCO) stage down to the same frequency as the output of the R divider. Therefore this divider sets the multiplication factor (F) of the PLL. The OD divider divides the output clock of the VCO.

There is a constraint on the frequency of the clock at the comparator—the output of R. There is another constraint placed on the output of the VCO.

3   Configuring the XS1-L Device

The mode pins are used to determine the initial PLL settings used after reset. This configuration must be such that all of the constraints are met for the input clock driven onto CLK.

Mode[1:0]	CLK Range (MHz)	xCORE Clock (MHz)	R	F	OD	System Clock=CLK
00	4.22–13.0	130.0–399.75	0	122	1	30.75
01	21.66–100.0	86.66–400.0	0	23	2	4
10	10.4–48.0	86.66–400.0	0	49	2	8.333
11	4.33–20.0	86.66–400.0	0	119	2	20

Table 2: Mode Pins and Boot Configuration

If a different PLL configuration is required from that used to boot the application, the new settings should be written to the PLL_CTRL register. This will cause the XS1-L device to reset using the new PLL settings, and the same boot code will execute again. It is important, therefore, that the boot code should read the value of the PLL_CTRL register and compare it to the reconfigured value. If there is a difference, then this is the first time the boot code has executed and the new PLL settings should be written to PLL_CTRL, causing a reset. The second time the boot code executes, the value read back from the PLL_CTRL register will be the reconfigured value and the boot process can continue.

The easiest way to reprogram the PLL is to specify the application's frequency requirements in the XN file and use the xTIMEcomposer tools to reprogram the PLL—see Section §6.

The application may require the reprogramming of the system clock dividers. These can be programmed without causing a reset.

3.1   Frequency Control Registers

To access the frequency control registers on the *SSwitch* and *PSwitch*, packets of data must be constructed and communicated to the switches through a channel end.

Global PLL settings are controlled through registers in the System Switch Control (SSCTRL). From C or XC, use the *write_sswitch_reg()* and *read_sswitch_reg()* functions defined in *xs1.h*.

Field	Register	Bitfield	Reset	Description
R	SSCTRL PLL_CTRL (0x6)	[6:0]	Mode Pins	PLL input divider stage = R+1
F	SSCTRL PLL_CTRL (0x6)	[20:8]	Mode Pins	Multiplier stage of the PLL = (F+1)/2
OD	SSCTRL PLL_CTRL (0x6)	[25:23]	Mode Pins	PLL output divider stage = OD+1
SSDIV	SSCTRL SS-WITCH_CLK_DIVIDER (0x7)	[15:0]	0	System switch clock divider = SSDIV+1 Reset value produces 400MHz for a 400MHz system clock
REFDIV	SSCTRL SW_REF_CLK_DIVIDER (0x8)	[15:0]	3	Reference clock divider = REFDIV+1 Reset value produces 100MHz for a 400MHz system clock.

Table 3: SSCTRL Registers

IMPORTANT Writing to the PLL_CTRL register (0x6) resets the xCORE tile. To reset a multi-tile device, make sure that tile 0 is reset first before any other tiles.

Settings on an individual tile basis are controlled through registers in the Processor Switch Control (PSC-TRL). From C or XC, use the *write_pswitch_reg()* and *read_pswitch_reg()* functions defined in *xs1.h*.

Field	Register	Bitfield	Reset	Description
XCDIV	PSCTRL PROC_CONTROL (0x6)	[15:0]	0	xCORE clock divider = XCDIV+1 Reset value produces 400MHz for an 400MHz system clock

Table 4: PSCTRL Registers

4 Example PLL Configurations

4.1 Standard Configuration: 20MHz Oscillator

Use **MODE[1:0] = 11**. The PLL will configure to the standard 400MHz, with the xCORE tile and SSwitch running at 400MHz, with a 100MHz reference clock.

4.2 25MHz Oscillator

Use **MODE[1:0] = 01**. For the initial boot, the system clock will be 100.0MHz, with the xCORE tile also running at 100.0MHz. The following are required: **R** = 0, **F** = 95, **OD** = 2. Write **0x01005F00** to SSCTRL, PLL_CTRL (0x6) register to bring the PLL output up to 400MHz, with code similar to the following:

```
#define PLL_CTRL_REGNUM 6
#define PLL_CTRL_25MHz 0x01005F00
{
  unsigned pllCtrlReadData;
  read_sswitch_reg(get_core_id(), PLL_CTRL_REGNUM, pllCtrlReadData);
  if (pllCtrlReadData != PLL_CTRL_25MHz)
  {
    write_sswitch_reg(get_core_id(), PLL_CTRL_REGNUM, PLL_CTRL_25MHz);
  }
}
```

4.3 15MHz Oscillator

Use **MODE[1:0] = 10**. For the initial boot, the system clock will be 125MHz, with the xCORE tile also running at 125MHz. The following are required: **R** = 0, **F** = 159, **OD** = 2. Write **0x01009F00** to **SSCTRL**, **PLL_CTRL (0x6)** register to bring the PLL output up to 400MHz, with code similar to the following:

```
#define PLL_CTRL_REGNUM 6
#define PLL_CTRL_15MHz 0x01009F00
{
  unsigned pllCtrlReadData;
  read_sswitch_reg(get_core_id(), PLL_CTRL_REGNUM, pllCtrlReadData);
  if (pllCtrlReadData != PLL_CTRL_15MHz)
    {
      write_sswitch_reg(get_core_id(), PLL_CTRL_REGNUM, PLL_CTRL_15MHz);
    }
}
```

5 Example System Clock Divider Configurations

5.1 133MHz Reference Clock

To adjust the Reference Clock to 133MHz with a 400MHz System Clock, set **REFDIV** to **2** using the following code:

```
#define REFDIV_REGNUM 8
write_sswitch_reg(get_core_id(), REFDIV_REGNUM, 0x02);
```

This will adjust all timers and clock-blocks to run at 133.3MHz, and allow ports to be configured at 66MHz, 33MHz and so on.

5.2 Slow Switch Clock

For applications where only a single XS1-L device is used, the SSwitch is only used for configuration purposes. Once the system is configured, the SSwitch clock can be substantially reduced to save on dynamic power. 1MHz is a good option for a low power SSwitch clock because the SSwitch power is dominated by the static power at this frequency.

To reduce the SSwitch clock to 1MHz with a system clock of 400MHz, set **SSDIV** to **399** using the following code:

```
#define SSDIV_REGNUM 7
write_sswitch_reg(get_core_id(), SSDIV_REGNUM, 399);
```

5.3 xCORE Tile Clock 200MHz

If your application does not need to run the xCORE tile at full speed to work, dynamic power can be saved by running the tile at a slower rate and entering STANDBY mode.

To run a tile at 200MHz from a system frequency of 400MHz, set **XCDIV** to **1** and enter STANDBY mode by writing **0x10** to **XCORE_CTRL0**.

```
#define XCDIV_REGNUM 6
write_pswitch_reg(get_core_id(), XCDIV_REGNUM, 1);
#define  PS_XCORE_CTRL0_REGNUM 2
setps(PS_XCORE_CTRL0_REGNUM,0x10);
```

6 Configuring the Clock System Through the XN File

The PLL and the reference clock frequency can be programmed automatically for an application by using the xTIMEcomposer tools. The application's input oscillator frequency, system frequency and reference frequency can be specified in the XN file. When the application code is written to a flash device with XFLASH, the code to reprogram the PLL to the desired system and reference frequencies will be added. When run with XRUN or XGDB the PLL is reprogrammed via JTAG.

Attribute	Description	Default Value
Oscillator	Input frequency on the CLK pin. If this attribute is specified, the system frequency and the reference frequency are programmed using their specified (or default) values. If this attribute is not specified, the boot configuration for the system and reference frequencies are used for the application.	Uses boot configuration
SystemFrequency	The desired system frequency. The *Oscillator* attribute must be specified if this attribute is specified.	400MHz
ReferenceFrequency	The desired reference frequency. The *Oscillator* attribute must be specified if this attribute is specified.	100MHz

Table 5: XN File Frequency Control Attributes

The frequency control attributes should be added to the Node node within the XN file. Frequencies should be specified with their unit of MHz, kHz or Hz, (for example 500MHz, 24576kHz or 6745800Hz). If the frequency control attributes are not specified in the XN file, then the xTIMEcomposer tools will not modify the frequency control registers.

If the target frequency specified in the XN file for either the system or reference frequency cannot be met exactly for the application's input frequency, a frequency close to the target frequency will be selected by the tools and a warning will be issued. XFLASH always issues the warning when it occurs, as does XGDB. XRUN only issues the warning if it has been run with the `--verbose` switch. XGDB issues the warning when the `connect` command is issued. Within xTIMEcomposer Studio, the XFLASH warning is issued to the *Console*, but the XGDB or XRUN warning is not available to the user.

6.1 Example XN file using Frequency Control Attributes

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Network xmlns="http://www.xmos.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.xmos.com http://www.xmos.com">
  <Type>Board</Type>

  <Declarations>
    <Declaration>tile tileref[1]</Declaration>
  </Declarations>

  <Nodes>
    <Node Id="0" Type="XS1-L8A-64-TQ128" Oscillator="20MHz"
        SystemFrequency="500MHz" ReferenceFrequency="250MHz">
      <Tile Number="0" Reference="tileref[0]">
        <Port Location="XS1_PORT_1I" Name="PORT_UART_RX"/>
        <Port Location="XS1_PORT_1J" Name="PORT_UART_TX"/>
      </Tile>
    </Node>
  </Nodes>

  <JTAGChain>
    <JTAGDevice NodeId="0"/>
  </JTAGChain>

</Network>
```