AC458
Application Note
RTG4 CCC Dynamic Configuration

Microsemi
a Microchip company

Microsemi Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Learn more at www.microsemi.com.

51900458. 4.0 12/20

Contents

Figures

Tables

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the current publication.

1.1 Revision 4.0

The following is a summary of the changes made in this revision.

- Added Figure 18, page 14.
- Added Appendix 1: Programming the Device Using FlashPro Express, page 19.
- Added Appendix 2: Running the TCL Script, page 22.
- Removed the references to Libero version numbers.

1.2 Revision 3.0

Update the document for Libero v11.9 SP1 software release.

1.3 Revision 2.0

Updated the document for Libero v11.8 SP2 software release.

1.4 Revision 1.0

The first publication of this document.

2 RTG4 CCC Dynamic Configuration

This document describes how to perform dynamic configuration for both single and dual RTG4™ field programmable gate array (FPGA) clock conditioning circuit (CCC) by changing output clock frequency in a glitch-free way using general purpose divider (GPD).

RTG4 devices have 8 CCC blocks (2 in all 4 corners). Each CCC provides a complete clocking scheme for any logic implemented in the FPGA fabric and base clock for on-chip hard IP blocks like FDDR and SERDESIF. It is possible to configure the CCC parameters dynamically (via an advanced peripheral bus (APB3) interface) without reprogramming the device.

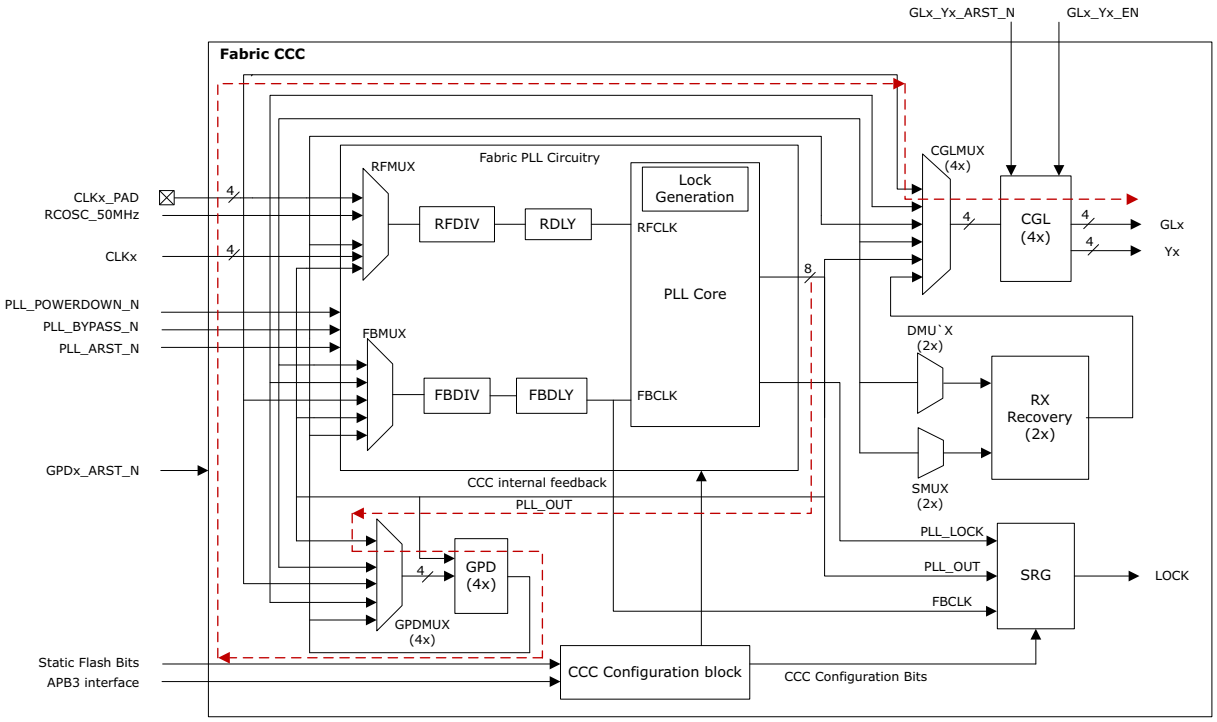
The following figure shows the block diagram of RTG4 Fabric CCC. The dotted line shows the path used when dynamically configuring GPD dividers. Each fabric CCC has four GPDs and CCC outputs can get their source from any of the GPD outputs. For instance, the GPD2 divider can be used for GL0 output.

GPD is used for dividing source clock selected by a general purpose multiplexer (GPMUX) by a factor of 1 to 255 to obtain the necessary clock frequency on the output. In this reference design, the GPD divider value is configured dynamically to different values at run time to obtain different output frequencies. Each GPD has its own GPMUX, which helps in selecting clock source to the GPD circuit from fabric CCC clock sources (external dedicated I/O, RC oscillator, fabric clock, and PLL outputs).

Any of the configuration registers can be accessed dynamically using the APB3 interface. This document does not discuss about all the Fabric CCC registers that can be dynamically configured. It only shows how to dynamically change output clock frequency using the GPD divider.

For more information about dynamic configuration registers, GPD, and GPMUX, refer to *UG0586: RTG4 FPGA Clocking Resources User Guide*.

Figure 1 • **RTG4 Fabric CCC Block Diagram**



2.1 Design Requirements

The following table lists the hardware and software requirements for this design.

Table 1 • **Design Requirements**

Requirement	Version
Hardware	
RTG4 FPGA Development Kit: • USB 2.0 cable • 12 V, 5A AC power adapter and cords	Rev B or later
Host PC or Laptop	64-bit Windows 7 and 10
Software	
FlashPro Express	**Note:** Refer to the `readme.txt` file provided in the design files for the software versions used with this reference design.
Libero® System-on-Chip (SoC)	
Host PC Drivers (included with the design files)	–

Note: Libero SmartDesign and configuration screen shots shown in this guide are for illustration purpose only. Open the Libero design to see the latest updates.

2.2 Prerequisites

Before you start:

1. Download and install Libero SoC (as indicated in the website for this design) on the host PC from the following location: *https://www.microsemi.com/product-directory/design-resources/1750-libero-soc*
2. For demo design files download link:

   http://soc.microsemi.com/download/rsc/?f=rtg4_ac458_df

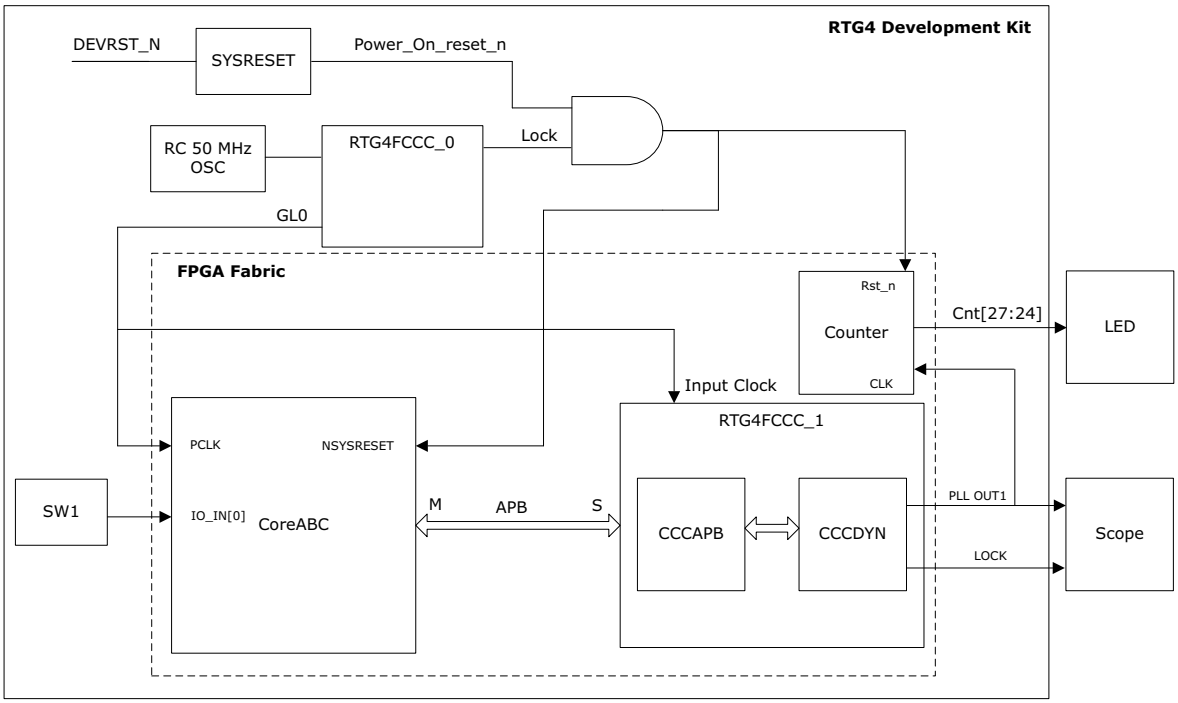
2.3 Design Description

The reference design in this application note contains two use models to demonstrate dynamic configuration for single and dual CCC in the same corner of the RTG4 device. CoreABC provides the APB3 Master interface. CoreABC is a programmable soft processor used to read and write from CCC configuration registers. CoreABC program provided in the design writes into appropriate registers to divide source clock using GPD.

Two separate designs are provided for single and dual CCC dynamic configuration use models. RTG4FCCC_0 is used for generating the APB interface clock in both use models. RTG4FCCC_1 is used for a single CCC dynamic configuration use model and dual_cccdyn_top_0, which contains two fabric CCCs from the same corner is used for dual CCC dynamic configuration use model. In both cases, dynamic frequency reduction in the CCC output clock using the GPD divider is demonstrated by probing CCC outputs in real time using an oscilloscope. The upper nibble of a sequential counter in the fabric drives four on board LED's. Once the clock frequency is dynamically configured using the GPD circuit, the counter slows down which in turn reduces the LED blinking rate.

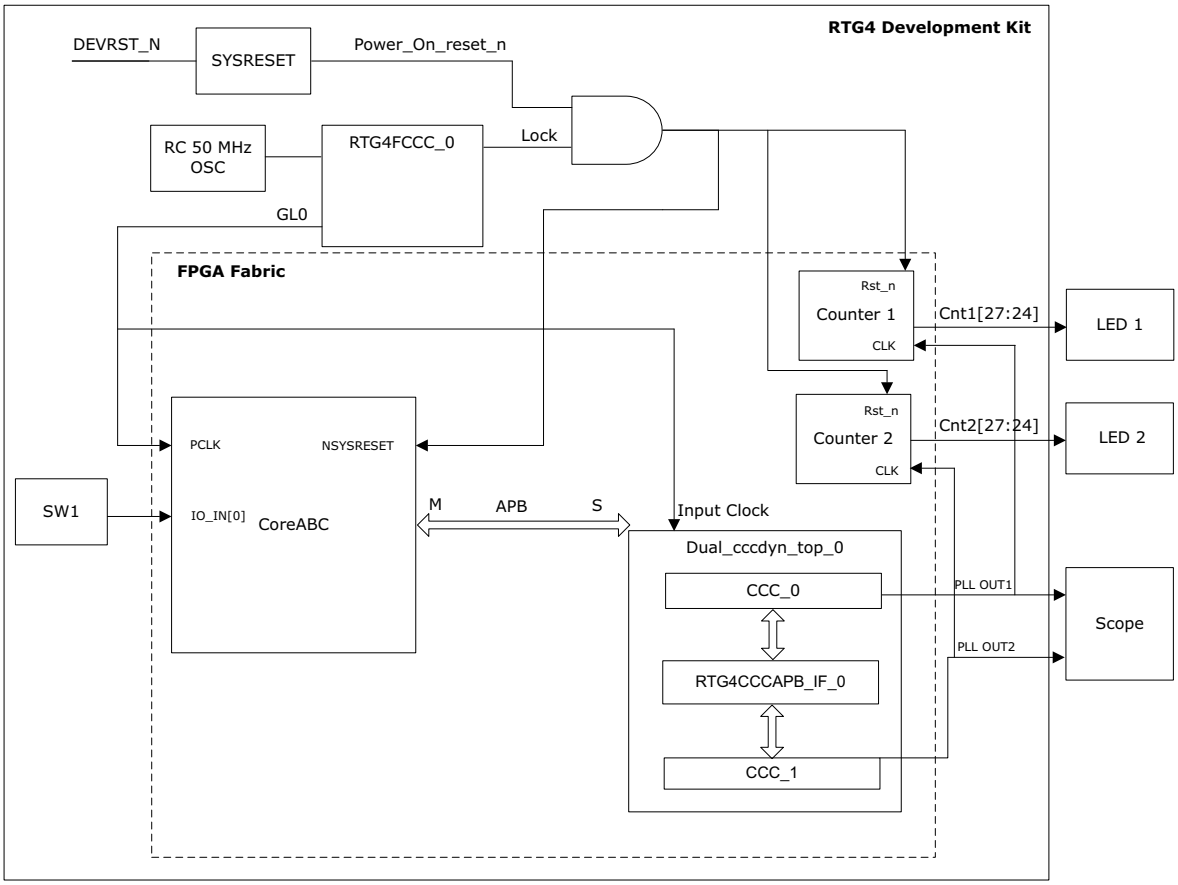
The following figure shows a top-level block diagram of the single CCC use model.

Figure 2 • **Single CCC Use Model Design**



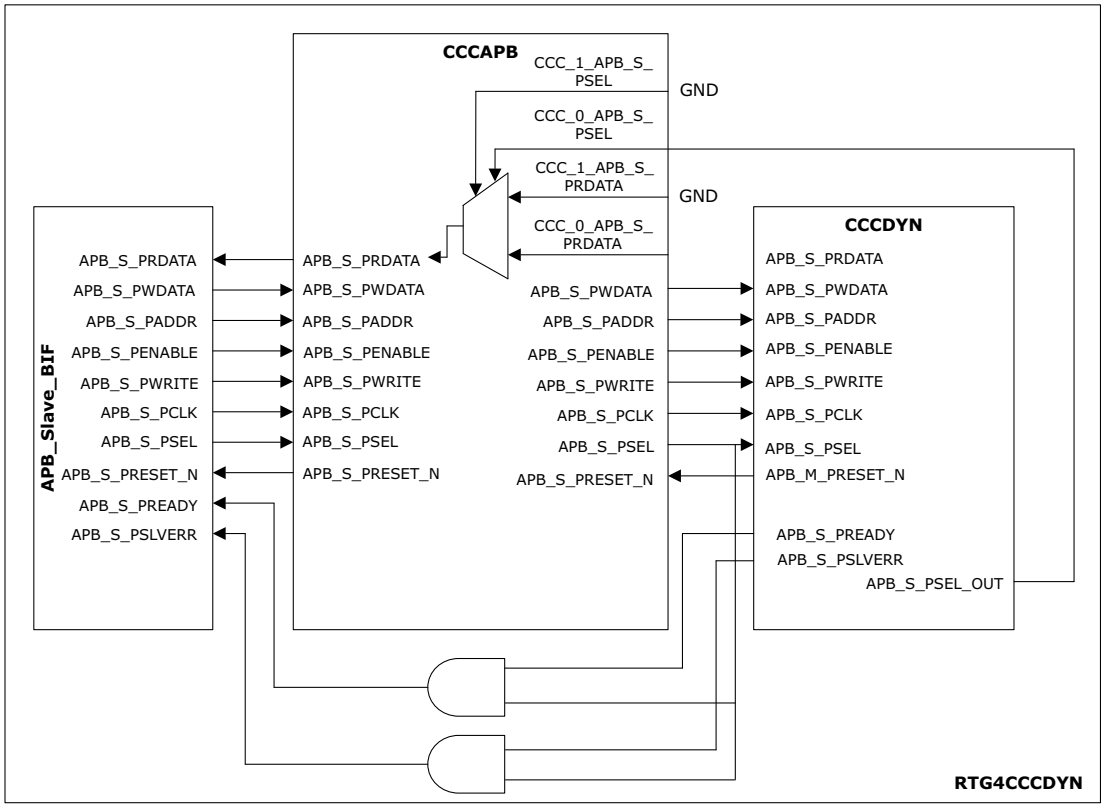
The following figure shows a top-level block diagram of the dual CCC use model.

Figure 3 • **Dual CCC Use Model Design**



The following figure shows connectivity between CCCDYN and CCCAPB macros along with extra soft logic, which is added automatically by the configurator for a single CCC dynamic configuration. For more information about how to perform dynamic configuration for a single CCC, refer to Hardware Implementation, page 7.

Figure 4 • **CCCDYN and CCCAPB Macro Connection for Single CCC Dynamic Configuration**



The new macro CCCDYN (similar to RTG4FCCC macro with dynamic configuration enabled) is used only for the dynamic CCC solution. It has an additional port APB_S_PSEL_OUT, which is a hardwired connection to CCCAPB macro. The APB macro instance is named CCCAPB, which has extra ports CCC_0/1_APB_S_PSEL and CCC_0/1_APB_S_PRDATA to support multiple dynamic CCCs from the same corner.

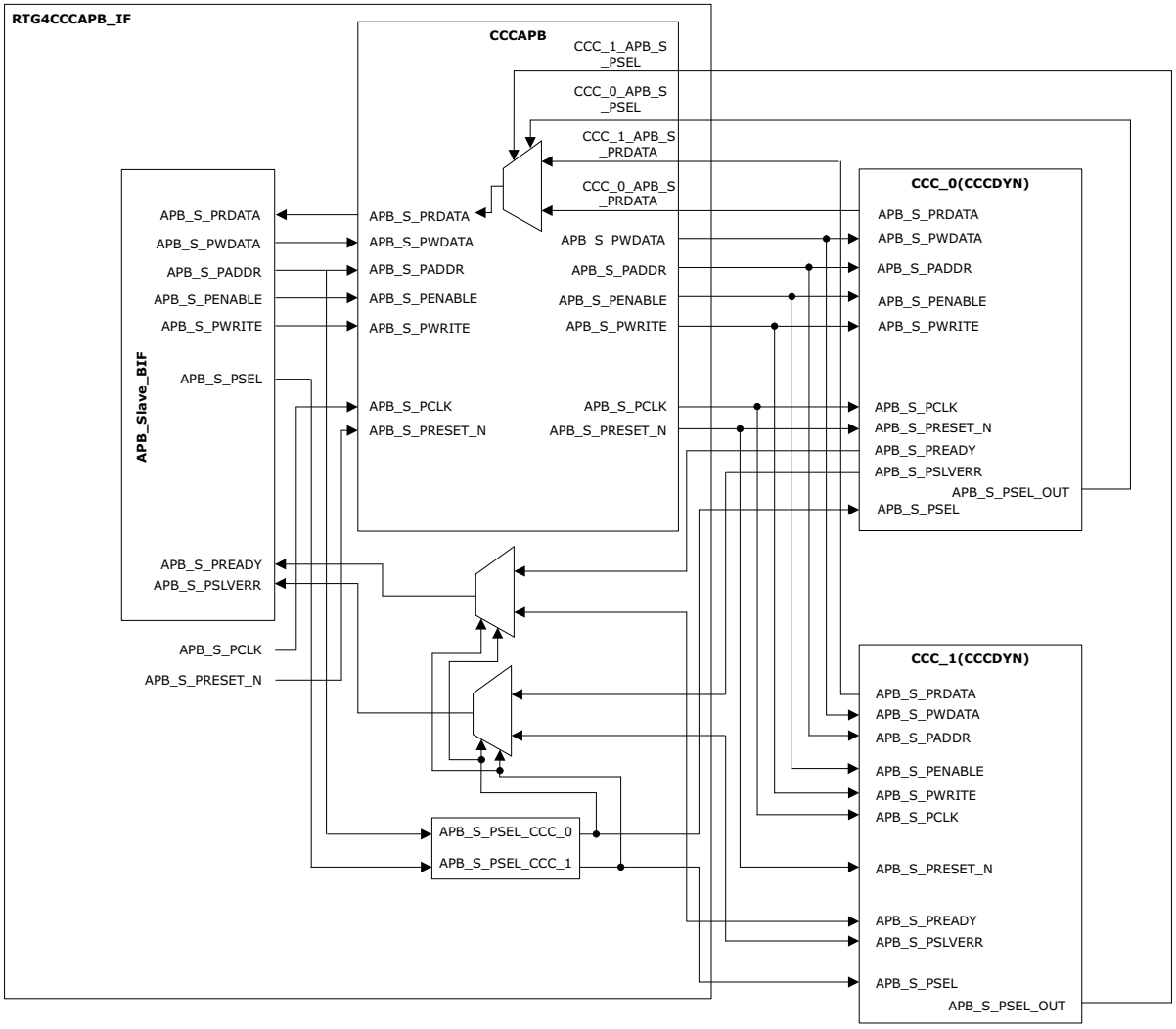
The 0/1 indices refer to two CCCs in the same corner. Only two CCCDYN macros placed in the same corner can be legally connected to the APB macro instance (CCCAPB).

In the preceding figure, CCC_1_APB_S_PSEL and CCC_1_APB_S_PRDATA are connected to GND since we are using only one CCC for a singe CCC dynamic configuration.

The following figure shows connectivity between CCCDYN and CCCAPB macros along with extra soft logic, which is added automatically by the configurator for dual CCC dynamic configuration. The RTG4CCCAPB_IF block delimits the configurator boundaries. Connect manually two CCCDYN macros (CCC_0/CCC_1) with the RTG4CCCAPB_IF module for a dual CCC dynamic solution.

Address bit [8] of APB_S_PADDR is used for decoding, which of the CCC in the same corner is used by CCCAPB macro. If APB_S_PADDR[8] is equal to 1, CCC_1 is selected if APB_S_PADDR[8] is not equal to 1, CCC_0 is selected, as shown in the following figure.

Figure 5 • **CCCDYN and CCCAPB Macro Connection for Dual CCC Dynamic Configuration**



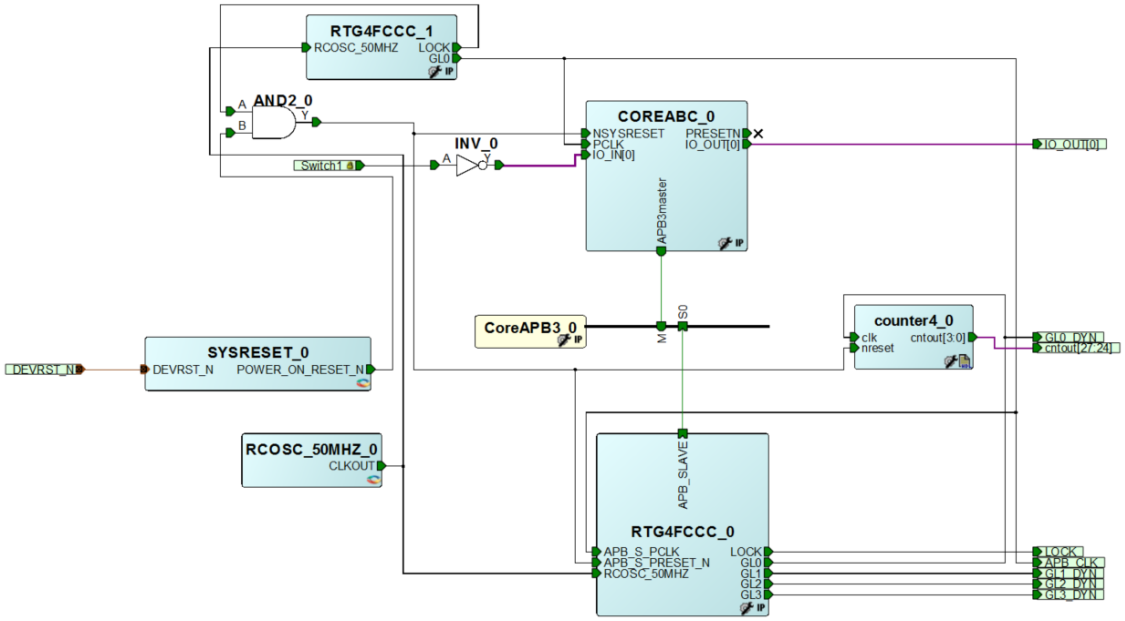
2.4 Hardware Implementation

Hardware implementation involves configuring CCC for dynamic configuration along with the CoreABC program to dynamically read and write into CCC configuration registers. The reference design of both single and dual CCC use model consists of CoreABC, on-chip 50 MHz RC oscillator, Sysreset macro, AND gate, FCCC, and counter module.

In the single CCC use model, RTG4FCCC_0 generates an APB interface clock at 50 MHz frequency and RTG4FCCC_1 generates global output GL0 (PLL OUT1), which is divided dynamically using GPD divider.

The following figure shows the SmartDesign top-level block diagram for a single CCC dynamic configuration.

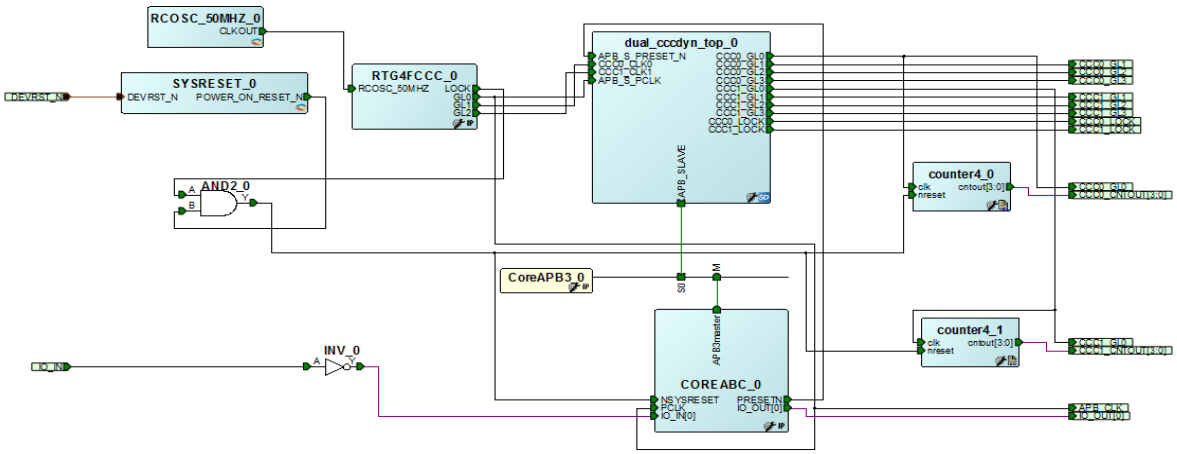
Figure 6 • **SmartDesign Top-level Single CCC Use Model**



In the dual CCC use model, RTG4FCCC_0 generates an APB interface clock at 50 MHz frequency and two 100 MHz clock outputs for sourcing clock to CCC_0 and CCC_1 in dual_cccdyn_top_0 module. CCC_0/CCC_1 generates global output GL0 (PLL OUT0/PLL OUT1), which is divided dynamically using GPD divider.

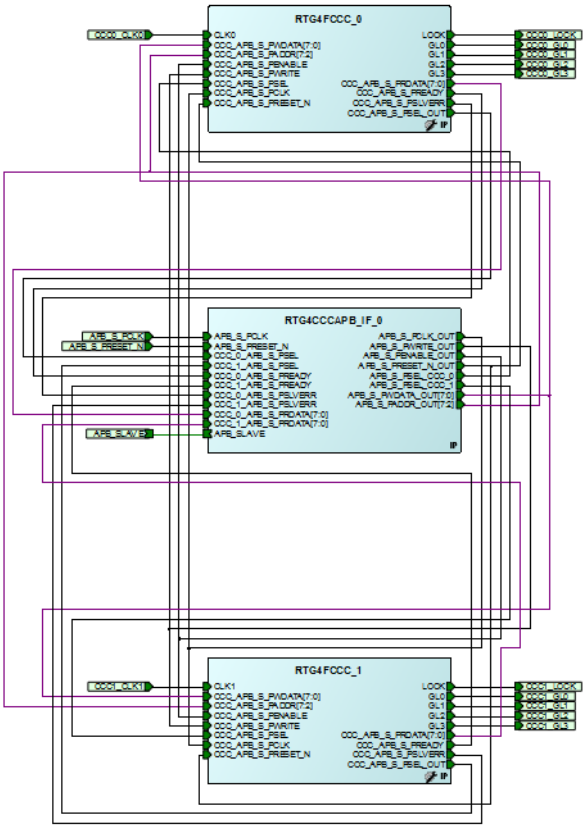
The following figure shows the SmartDesign top-level block diagram for a dual CCC dynamic configuration.

Figure 7 • **SmartDesign Top-level Dual CCC Use Model**



The following figure shows the manual connection between CCCDYN and RTG4CCCAPB_IF macro for a dual CCC dynamic solution.

Figure 8 • **CCCDYN and RTG4CCCAPB_IF Connection**



2.4.1 CoreABC Program

The CoreABC Program describes the register settings required to divide the source input clock using the GPD divider.

Figure 9 • **CoreABC Program for Single CCC Use Model**

```
NOP

$Welcome

$restart

WAIT UNTIL INPUT0

//Selecting PLL output clock as input to GPMUX0
APBWRT DAT 0 0x38 0x07
//Writing GPD0 divider value
APBWRT DAT 0  0x48 0x10
//GPD0 mode
APBWRT DAT 0 0x88 0x03
//Selecting GPD0 output as input to CGLMUX0
APBWRT DAT 0 0x1C 0x08

//Selecting PLL output clock as input to GPMUX1
APBWRT DAT 0 0x3C 0x07
//Writing GPD1 divider value
APBWRT DAT 0  0x4C 0x8
//GPD1 mode
APBWRT DAT 0 0x8C 0x03
//Selecting GPD1 output as input to CGLMUX1
APBWRT DAT 0 0x24 0x09

//Selecting PLL output clock as input to GPMUX2
APBWRT DAT 0 0x40 0x07
//Writing GPD2 divider value
APBWRT DAT 0  0x50 0x2
//GPD2 mode
APBWRT DAT 0 0x90 0x03
//Selecting GPD2 output as input to CGLMUX2
APBWRT DAT 0 0x2C 0x0A

//Selecting PLL output clock as input to GPMUX3
APBWRT DAT 0 0x44 0x07
//Writing GPD3 divider value
APBWRT DAT 0  0x54 0x4
//GPD3 mode
APBWRT DAT 0 0x94 0x03
//Selecting GPD3 output as input to CGLMUX3
APBWRT DAT 0 0x30 0x0B

IOWRT DAT 1

Jump $restart

HALT
```

Setting to divide PLL output with GPD divider value of 16

Setting to divide PLL output with GPD divider value of 8

Setting to divide PLL output with GPD divider value of 2

Setting to divide PLL output with GPD divider value of 4

For the dual CCC use model, the sequence of register settings to be followed for changing output clock frequency using GPD divider is the same as the single CCC use model shown in Figure 9, page 10. To access CCC_0 and CCC_1 separately, APB_S_PADDR[8] address bit must be used, as shown in the following figure.

Figure 10 • **CoreABC Program for Dual CCC Use Model**

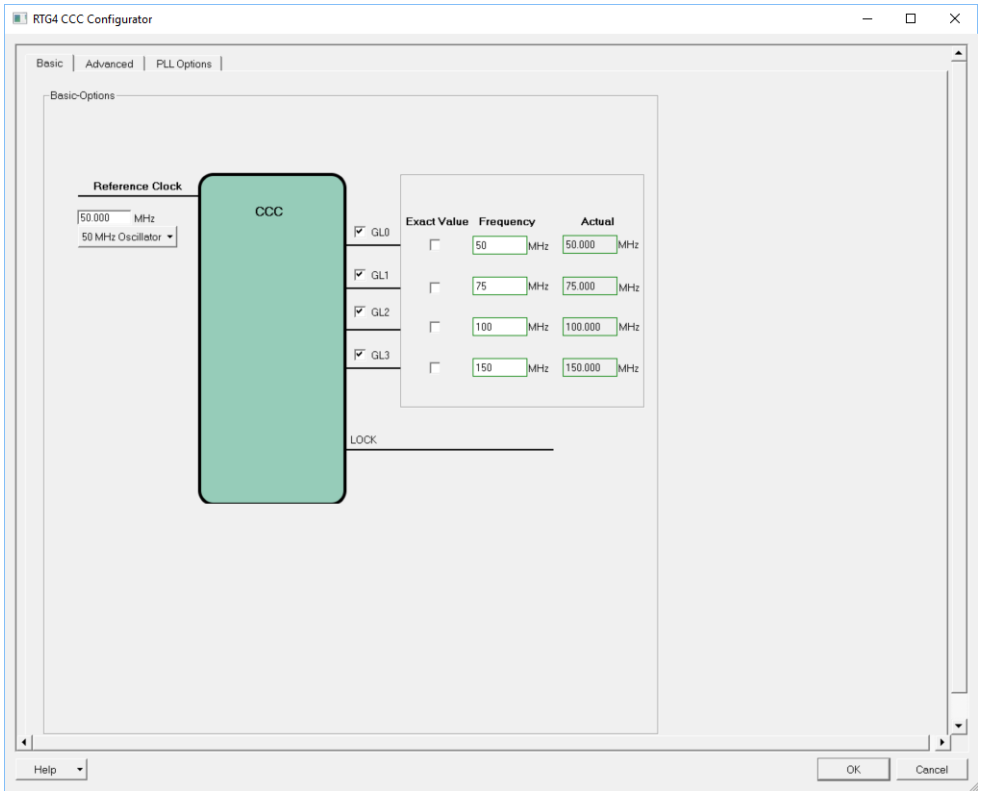
```
//RTG4 CCC0
//Selecting PLL output clock as input to GPMUX0
APBWRT DAT 0 0x038 0x07
//Writing GPD0 divider value
APBWRT DAT 0  0x048 0x08
//GPD0 mode                    -> APB_S_PADDR[8] = 0
APBWRT DAT 0 0x088 0x03
//Selecting GPD0 output as input to CGLMUX0
APBWRT DAT 0 0x01C 0x08

// RTG4 CCC1
//Selecting PLL output clock as input to GPMUX0
APBWRT DAT 0 0x138 0x07
//Writing GPD0 divider value
APBWRT DAT 0  0x148 0x04
//GPD0 mode                    -> APB_S_PADDR[8] = 1
APBWRT DAT 0 0x188 0x03
//Selecting GPD0 output as input to CGLMUX0
APBWRT DAT 0 0x11C 0x08
```

2.4.2 Configuring CCC for Dynamic Configuration

The following steps describe how to configure CCC for dynamic configuration in the Libero SoC software.

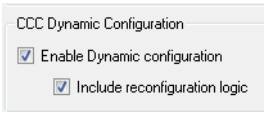
1. Open RTG4 CCC Configurator and then click **Basic** tab.
2. In the **Basic** tab, enter the values and select the checkboxes, as shown in the following figure.

Figure 11 • **PLL Output Clock Settings**



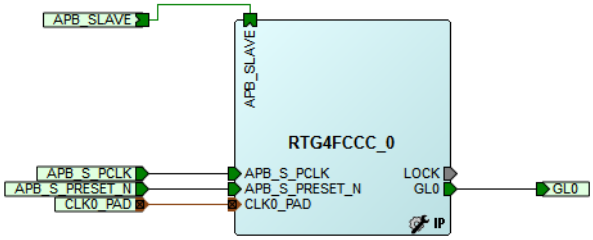
3. In the **PLL Options** tab, selecting **Enable Dynamic configuration** automatically selects **Include reconfiguration logic** for a single CCC use model under **CCC Dynamic Configuration**, as shown in the following figure.

Figure 12 • **Single CCC Dynamic Configuration Setting**



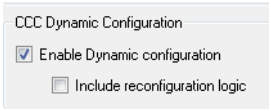
The following figure shows the SmartDesign block of a fabric CCC with a single dynamic configuration enabled.

Figure 13 • **Single CCC Dynamic Configuration SmartDesign**



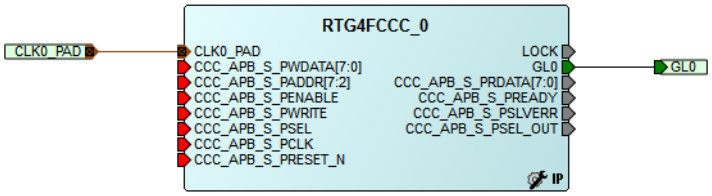
4. In the **PLL Options** tab, select **Enable Dynamic configuration** for dual CCC use model as shown in the following figure.

Figure 14 • **Dual CCC dynamic configuration setting**



The following figure shows the SmartDesign block of a fabric CCC with dual dynamic configuration enabled.

Figure 15 • **Dual CCC Dynamic Configuration SmartDesign**



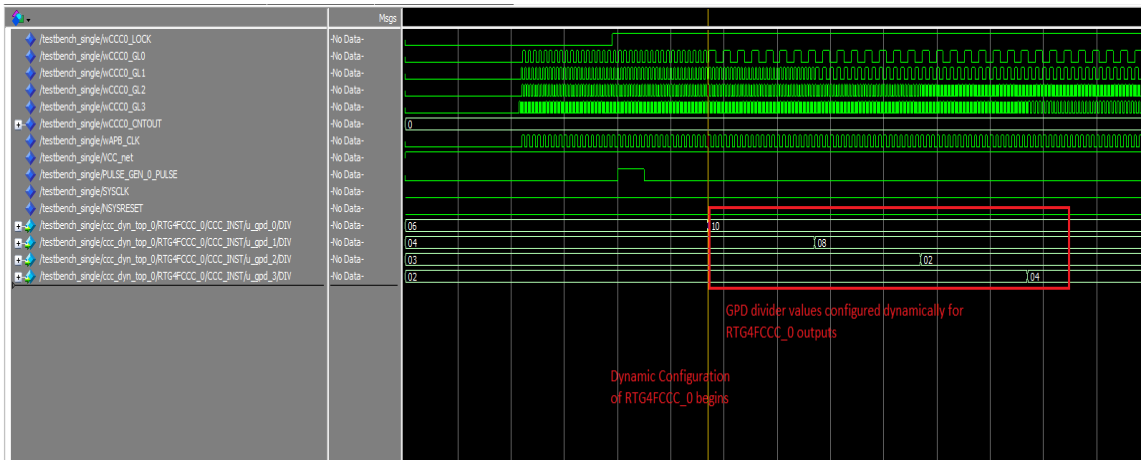
5. Click **Ok** after making the required setting. CCC generation completes with APB slave interface exposed for manual connection, as shown in Figure 8, page 9.

2.5 Simulation Results

Simulation results for single and dual CCC dynamic configuration use model are shown in the following figures.

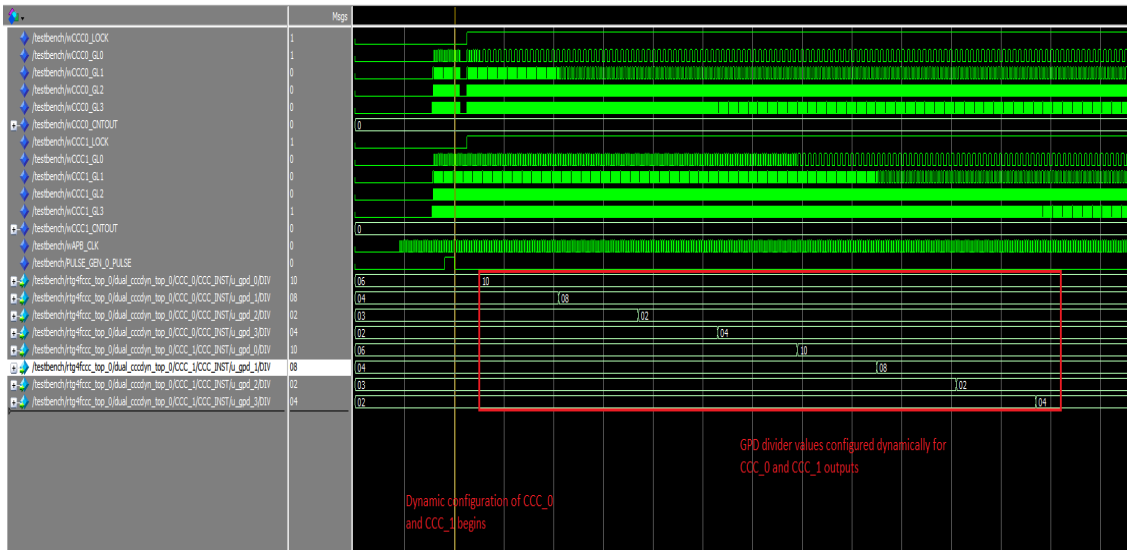
In a single CCC dynamic configuration, GPD divider values for CCC outputs (GL0, GL1, GL2, and GL3) are written dynamically via the APB3 interface as highlighted in the following figure.

Figure 16 • **Single CCC Dynamic Configuration Simulation Window**



In the dual CCC dynamic configuration, GPD divider values for two CCC outputs (GL0, GL1, GL2, and GL3) are written dynamically via the APB3 interface as highlighted in the following figure.

Figure 17 • **Dual CCC Dynamic Configuration Simulation Window**



2.6 Setting Up the Design

The following steps describe how to set up a hardware demo for the RTG4 development kit.

1. Connect the jumpers on the RTG4 development kit as shown in the following table.

Table 2 • **RTG4 Development Kit Jumper Setting**

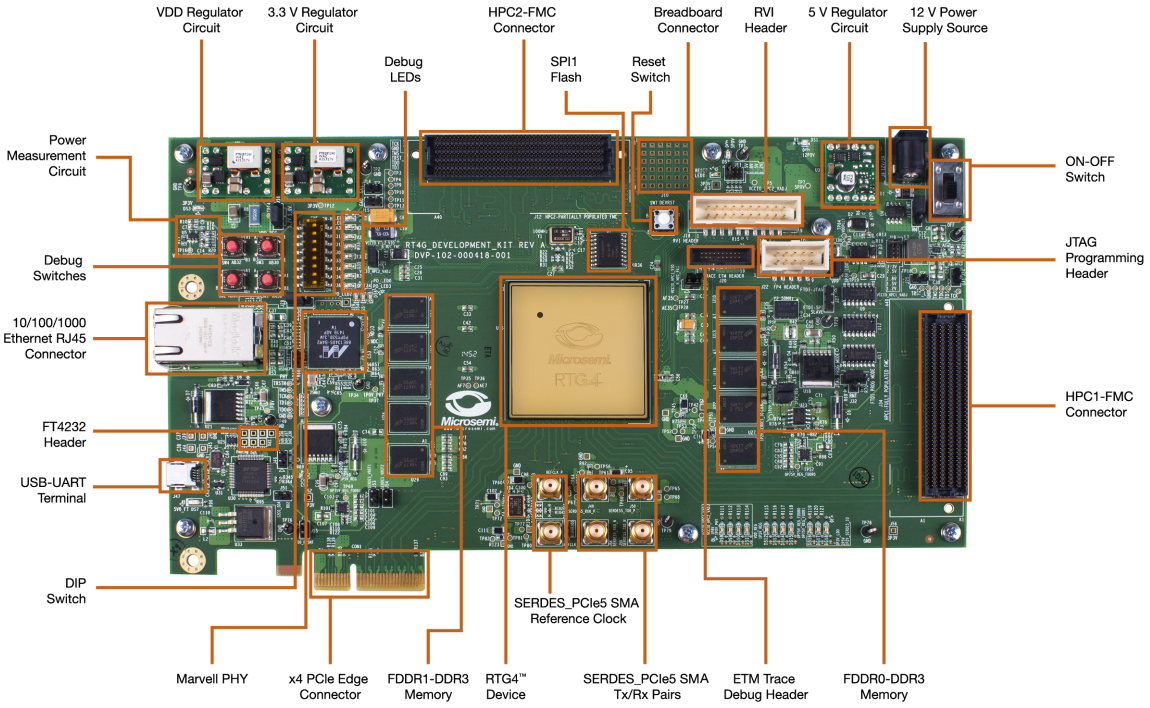
Jumper	Pin From	Pin To	Comments
J11, J17, J19, J23, J26, J21, J32, and J27	1	2	Default
J16	2	3	Default
J33	1 3	2 4	Default

Note: Ensure that the power supply switch **SW6** is switched OFF while connecting jumpers on the RTG4 development kit.

2. Connect the host PC to the **J47** connector using the USB cable.
3. Connect the power supply to the **J9** connector and switch ON the power supply switch, **SW6**.

The following figure shows the RTG4 Development Kit board.

Figure 18 • **RTG4 Development Kit**



2.7 Running the Design

The following are the steps to run the design:

1. To program the RTG4 Development Kit with the job file provided as part of the design files using FlashPro Express software, refer to Appendix 1: Programming the Device Using FlashPro Express, page 19.
2. Fabric counter logic which is connected to the onboard LEDs is run using clock output from CCC in which dynamic configuration is enabled. Once the device is programmed, onboard four LEDs (W35, W34, V30, and W33) for a single CCC use model and eight LEDs (four for each CCC output) for dual CCC use model blinks.
3. Pressing on board switch SW1 initiates CoreABC program to dynamically write GPD divider values using APB3 interface. After dynamic configuration, a slow down in LEDs blinking rate is observed.

Oscilloscope waveform shown in the following figure demonstrates how the CCC output (GL0 and GL1) clock frequency changes dynamically when switch SW1 onboard is pressed. Output clocks are connected to bread board connector J10 for probing.

The following table describes the probe points on single and dual CCC.

Table 3 • **Probe Points**

	Output Signal	J10 Bread Board Pins
Single CCC	APB_CLK	12
	GL0_DYN	6
	GL1_DYN	5
	GL2_DYN	1
	GL3_DYN	4
	LOCK	7
Dual CCC	APB_CLK	6
	CCC0_GL0	1
	CCC0_GL1	2
	CCC0_GL2	5
	CCC0_GL3	8
	CCC0_LOCK	12
	CCC1_GL0	19
	CCC1_GL1	20
	CCC1_GL2	23
	CCC1_GL3	24
	CCC1_LOCK	26

Figure 19 • **CCC Output Clock Frequency Before Dynamic Configuration**

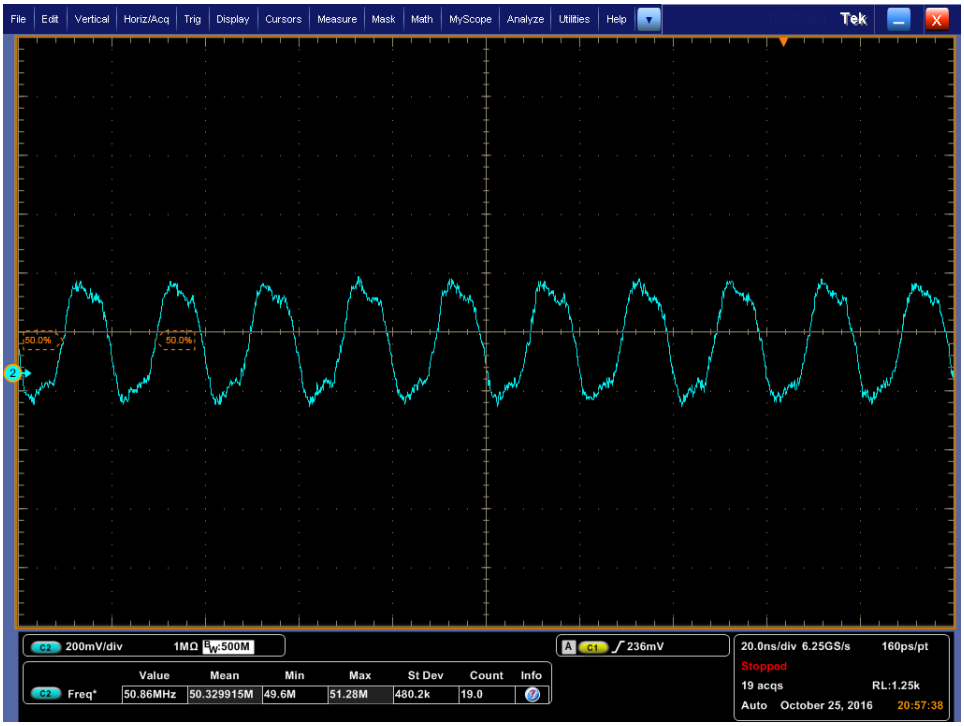


Figure 20 • **CCC Output Clock Frequency After Dynamic Configuration**

Note: The output clock frequency using GPD clock divider is based on: Clkout = Clkin / GPD_DIV[7:0], for GPD0_DIV[7:0] = 1 to 255.

The following figures show GL0 and GL1 dynamic frequency change with the help of trigger at the end of the dynamic configuration.

Figure 21 • **GL0 Dynamic Frequency Change**

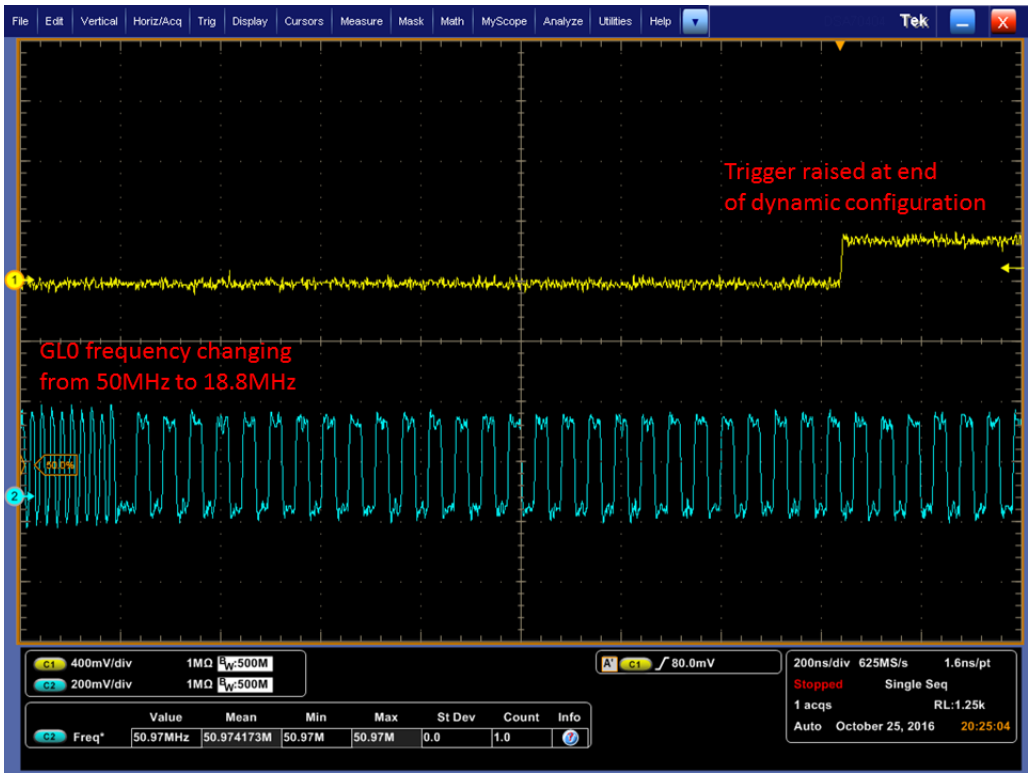
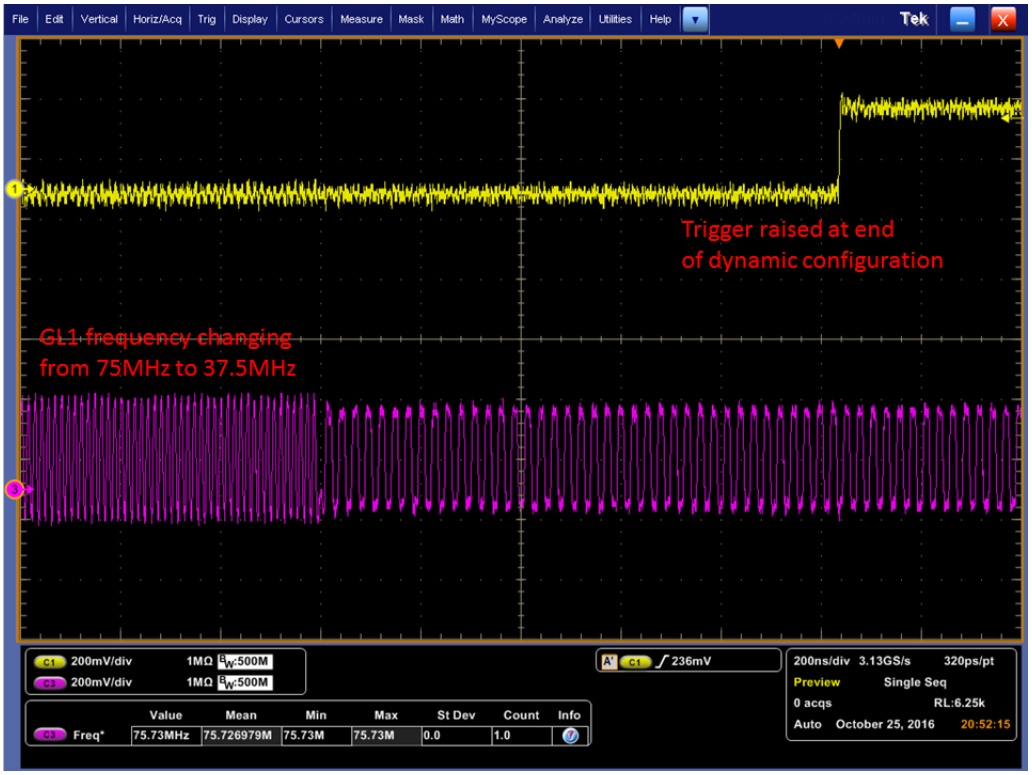


Figure 22 • **GL1 Dynamic Frequency Change**



2.8 Conclusion

This document describes how to perform dynamic configuration via an APB3 interface for both single and dual RTG4 FPGA CCC by changing output clock frequency in a glitchless way using GPD dividers. Results are shown using real time oscilloscope plots of the CCC output after dynamic configuration.

3 Appendix 1: Programming the Device Using FlashPro Express

This section describes how to program the RTG4 device with the programming job file using FlashPro Express.

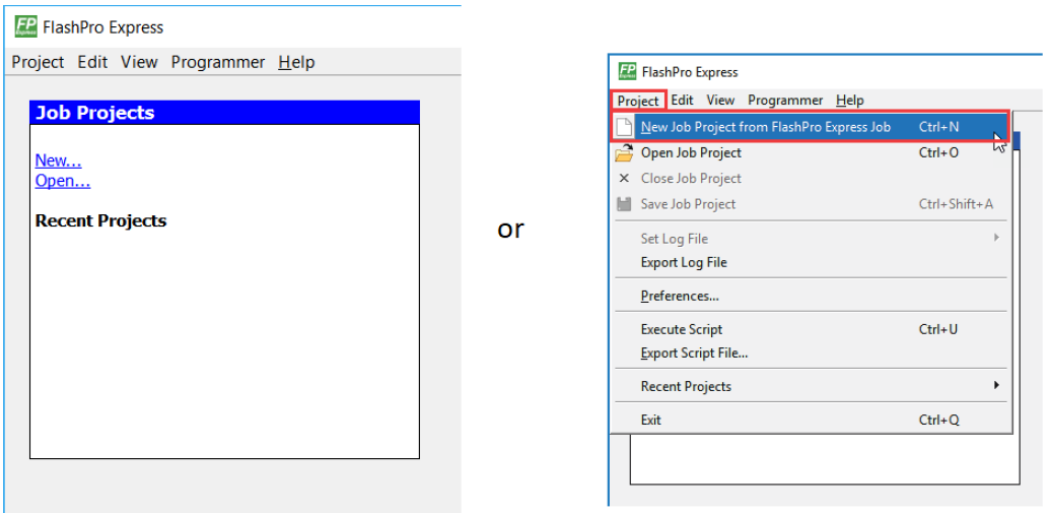
To program the device, perform the following steps:

1. Ensure that the jumper settings on the board are the same as those listed in *Table 3 of UG0617: RTG4 Development Kit User Guide*
2. Optionally, jumper **J32** can be set to connect pins 2-3 when using an external FlashPro4, FlashPro5, or FlashPro6 programmer instead of the default jumper setting to use the embedded FlashPro5.

Note: The power supply switch, **SW6** must be switched **OFF** while making the jumper connections.

3. Connect the power supply cable to the **J9** connector on the board.
4. Power **ON** the power supply switch **SW6**.
5. If using the embedded FlashPro5, connect the USB cable to connector **J47** and the host PC. Alternatively, if using an external programmer, connect the ribbon cable to the JTAG header **J22** and connect the programmer to the host PC.
6. On the host PC, launch the **FlashPro Express** software.
7. Click **New** or select **New Job Project from FlashPro Express Job** from **Project** menu to create a new job project, as shown in the following figure.

Figure 23 • **FlashPro Express Job Project**



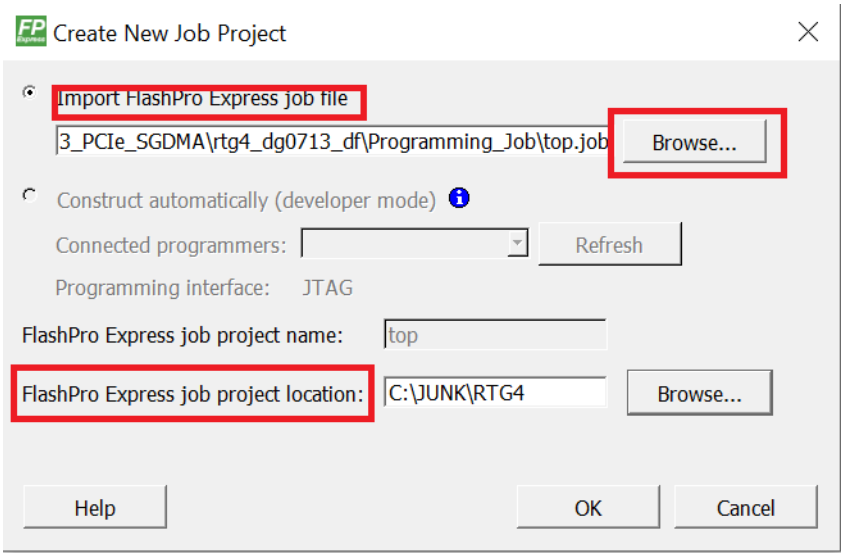
8. Enter the following in the **New Job Project from FlashPro Express Job** dialog box:

- **Programming job file:** Click **Browse**, and navigate to the location where the .job file is located and select the file. The default location is:

  `<download_folder>\rtg4_ac458_df\Programming_Job`

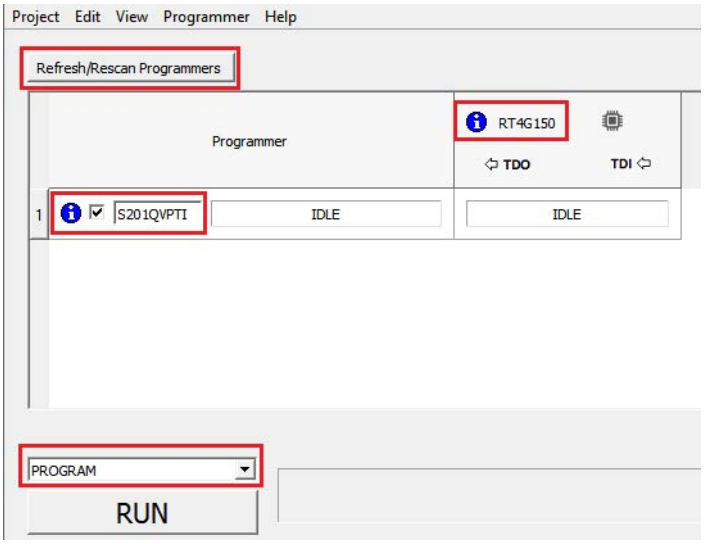
- **FlashPro Express job project location:** Click **Browse** and navigate to the desired FlashPro Express project location.

Figure 24 • **New Job Project from FlashPro Express Job**



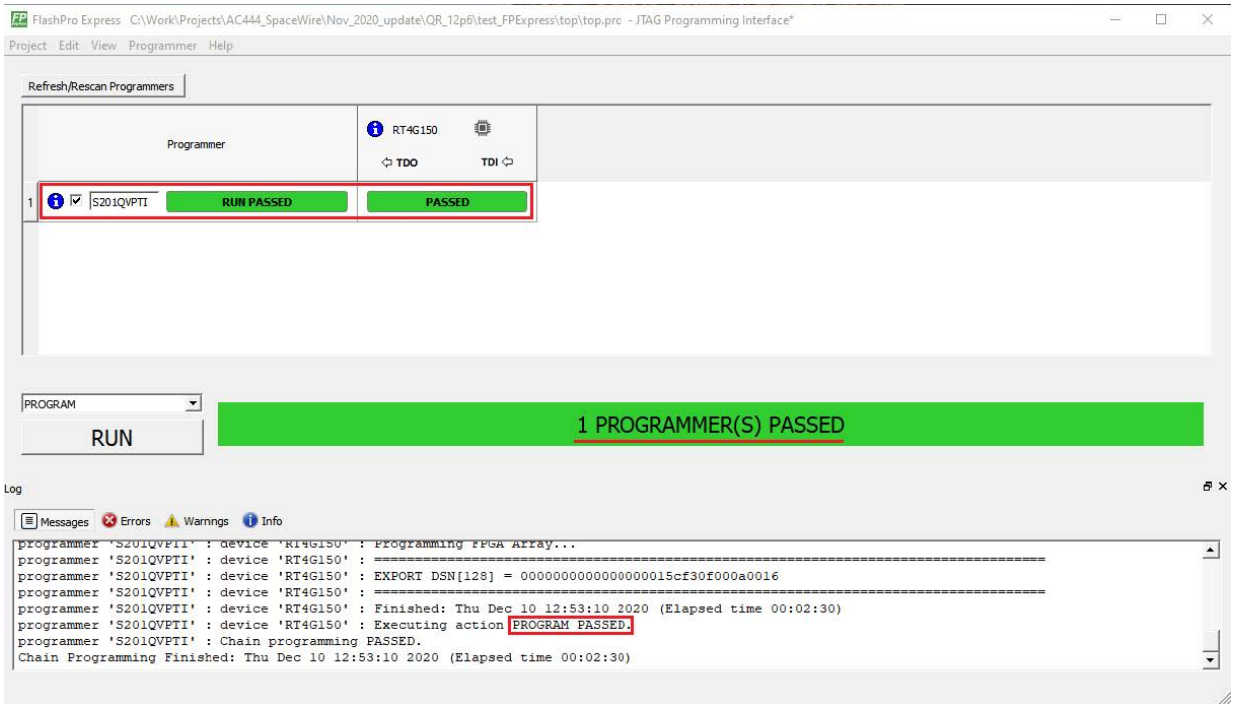
9. Click **OK**. The required programming file is selected and ready to be programmed in the device.
10. The FlashPro Express window appears as shown in the following figure. Confirm that a programmer number appears in the Programmer field. If it does not, confirm the board connections and click **Refresh/Rescan** Programmers.

Figure 25 • **Programming the Device**



11. Click **RUN**. When the device is programmed successfully, a **RUN PASSED** status is displayed as shown in the following figure.

Figure 26 • **FlashPro Express—RUN PASSED**



12. Close **FlashPro Express** or click **Exit** in the Project tab.

4 Appendix 2: Running the TCL Script

TCL scripts are provided in the design files folder under directory TCL_Scripts. If required, the design flow can be reproduced from Design Implementation till generation of job file.

To run the TCL, follow the steps below:

1. Launch the Libero software
2. Select **Project > Execute Script....**
3. Click Browse and select `script.tcl` from the downloaded TCL_Scripts directory.
4. Click **Run**.

After successful execution of TCL script, Libero project is created within TCL_Scripts directory.

For more information about TCL scripts, refer to **rtg4_ac458_df/TCL_Scripts/readme.txt.**

Refer to *Libero® SoC TCL Command Reference Guide* for more details on TCL commands. Contact Technical Support for any queries encountered when running the TCL script.

5 Appendix 3: Design Files

The design files are available for download at:
http://soc.microsemi.com/download/rsc/?f=rtg4_ac458_df

The design files consist of a VHDL version of Libero project folder single and dual CCC use models, programming file (*.job) for RTG4 Development Kit. Refer to the `readme.txt` file included in the design files for the directory structure and description.