


本章介绍了用户模式下的循环冗余检查（CRC）错误检测功能，并对如何从软错误中恢复作了讲解。

 所有的 Cyclone® IV 器件均支持配置错误检测功能，其中包括 Cyclone IV GX 器件、采用 1.0 V 核心电压的 Cyclone IV E 器件和采用 1.2 V 核心电压的 Cyclone IV E 器件。然而，用户模式错误检测功能仅被 Cyclone IV GX 器件以及采用 1.2 V 核心电压的 Cyclone IV E 器件支持。

Cyclone IV 器件中内置的专用电路具有 CRC 错误检测功能，能够有选择性地对单事件干扰（SEU）进行连续自动地检查。

在航空电子设备、通讯、系统控制、医疗和军事等领域所使用的关键应用中，要求能够做到以下两点：


- 确认存储在 FPGA 器件中的配置数据的准确性
- 使系统能够警惕配置错误的发生

在 Cyclone IV 器件中使用 CRC 错误检测功能不会对布线及性能产生影响。

本章涵盖以下几方面内容：

- 配置错误检测（第 9-1 页）
- 用户模式错误检测（第 9-2 页）
- 自动化的 SEU 检测（第 9-3 页）
- CRC\_ERROR 管脚（第 9-3 页）
- 错误检测模块（第 9-4 页）
- 错误检测时序（第 9-5 页）
- 软件支持（第 9-6 页）
- 从 CRC 错误中恢复（第 9-9 页）

## 配置错误检测


 配置错误检测功能可用于所有的 Cyclone IV 器件，包括 Cyclone IV GX 器件、采用 1.0 V 核心电压的 Cyclone IV E 器件和使用 1.2 V 核心电压的 Cyclone IV E 器件。

配置错误检测功能能够确定通过外部存储器件接收的配置数据在配置期间是否损坏。要验证配置数据，Quartus® II 软件使用一个函数来计算每个配置数据帧的 CRC 值，并将基于帧的 CRC 值存储在配置数据中，作为配置比特流的一部分。

配置过程中，Cyclone IV 器件根据接收到的数据帧采用相同的方法来计算 CRC 值，并与数据流中的帧 CRC 值进行比较。配置持续进行，直到器件检测到错误，或者完成了所有值的计算。

除了基于帧的 CRC 值，Quartus II 软件生成一个 32 bit CRC 值，用于整个的配置比特流。这个 32 bit CRC 值在配置的最后被存储在 32-bit 存储寄存器中，并用于“[用户模式错误检测](#)”中所介绍的用户模式错误检测。

## 用户模式错误检测

 用户模式错误检测功能可用于 Cyclone IV GX 和采用 1.2 V 核心电压的 Cyclone IV E 器件。采用 1.0 V 核心电压的 Cyclone IV E 器件不支持用户模式错误检测功能。


软错误是指电离粒子所导致的配置随机存取存储器 (CRAM) 位状态上的变化。Cyclone IV 器件内置错误检测电路，通过 CRAM 单元中的软错误来检测数据的损坏。

这种错误检测功能会根据器件内容不断计算已配置 CRAM 位的 CRC 值，并且与配置最后所得到的预计算出的 CRC 值进行比较。如果 CRC 值匹配，则在当前配置 CRAM 位中没有错误。错误检测过程继续，直到器件被重置（将 nCONFIG 设置为低电平）。

Cyclone IV 器件错误检测功能不检查存储器模块和 I/O 缓冲器。这些器件存储器模块支持奇偶校验位，用于检查存储器模块中的数据错误。I/O 缓冲器在错误检测期间不会被验证，因为配置数据使用抗软错误的触发器作为存储器单元。类似的触发器用于存储预计算的 CRC 值以及其它错误检测电路选项位。

Cyclone IV 器件中的错误检测电路使用 32 bit CRC IEEE 802 标准和一个 32 bit 多项式作为 CRC 发生器。因此，器件只执行一个 32 bit CRC 计算。如果没有出现软错误，则产生的 32 bit 签名值为 0x00000000，这将在 CRC\_ERROR 输出信号上产生一个 0。如果在器件中出现软错误，则产生的签名值是一个非零值，并且 CRC\_ERROR 输出信号为 1。

您可以通过更改 CRC 电路中的 32 bit CRC 存储寄存器，来注入一个软错误。验证引发的故障后，你可以使用相同的指令将 32 bit CRC 值恢复到正确的 CRC 值，并插入正确的值。

 使用已知错误值进行更新前，Altera 建议读出正确的值。

在用户模式中，Cyclone IV 器件支持 CHANGE\_EDREG JTAG 指令，使您能够写入 32 bit 存储寄存器。您可以使用 Jam™ STAPL 文件 (.jam) 来自动化测试及验证过程。只有器件在用户模式下，才能执行这个指令，它是一个强大的设计功能，使您能够动态地验证系统中的 CRC 功能性，而无需重配置器件。您可以接下来使用 CRC 电路来检查由 SEU 引入的真正错误。

表 9-1 说明了 CHANGE\_EDREG JTAG 指令。

表 9-1. CHANGE\_EDREG JTAG Instruction

JTAG 指令	指令代码	说明
CHANGE_EDREG	00 0001 0101	该指令连接 TDI 与 TDO 之间的 32 bit CRC 存储寄存器。所有预计算出的 CRC 值均加载到 CRC 存储寄存器，以测试 CRC_ERROR 管脚上的错误检测 CRC 电路的操作。

 测试完成后，Altera 建议重新启动器件。

## 自动化的 SEU 检测

Cyclone IV 器件提供了片上电路，实现了 SEU 自动检测。那些要求器件无差错运行在高海拔地区或者接近北极或南极地区的应用需要定期检查，以确保持续的数据完整性。由 Quartus II 软件的 **Device and Pin Options** 对话框控制的错误检测循环冗余代码功能使用一个 32 bit 的 CRC 电路，以确保数据的可靠性。同时，这一功能也是减缓 SEU 的最佳选择之一。

您可以使用 Cyclone IV 器件中现有电路来实现错误检测 CRC 功能，而不再需要外部逻辑。在重配置过程中对 CRC 进行计算，然后在常规操作期间，与自动计算出的 CRC 进行对比检查。当配置 CRAM 数据损坏时，CRC\_ERROR 管脚会报告一个软错误。您必须决定是通过选通 nCONFIG 管脚为低电平来重配置 FPGA，还是忽略该错误。


## CRC\_ERROR 管脚

需要一个特定的 CRC\_ERROR 错误检测管脚，以监测用户模式期间错误检测电路的结果。表 9-2 对 CRC\_ERROR 管脚进行了说明。

表 9-2. Cyclone IV 器件 CRC\_ERROR 管脚说明

CRC_ERROR 管脚类型	说明
专用输出或者开漏输出 (可选)	默认情况下，Quartus II 软件将 CRC_ERROR 管脚设置为专用输出管脚。如果 CRC_ERROR 管脚用作专用输出，那么您一定要确保该管脚所在插槽 (bank) 的 V <sub>CCIO</sub> 要符合系统接收信号的输入电压规范。另外，您也可以将该管脚设置为开漏输出管脚，通过在 Quartus II 软件中使能 <b>Device and Pin Options</b> 对话框的 <b>Error Detection CRC</b> 标签中相应选项来实现。将该管脚用作开漏输出，实现了一个在电压电平上的优势。欲将该管脚用作开漏输出管脚，您可以通过一个 10 kΩ 上拉电阻将该管脚置为 bank 1 的 V <sub>CCIO</sub> 。或者，根据系统接收信号的输入电压规范，您也可以将上拉电阻置为不同的上拉电压。

 Altera®网站上的Cyclone IV Devices Pin-Outs对Cyclone IV器件的CRC\_ERROR管脚信息作了详细的报告。

 WYSIWYG(所见即所得)是一种优化技术,在Quartus II软件中对VQM (Verilog Quartus 映射)网表进行优化。

## 错误检测模块

表 9-3 列出了用于检查配置位的 CRC 检测类型。

表 9-3. 用于检查配置位的 CRC 检测类型

第一 CRC 检测类型	第二 CRC 检测类型
<ul style="list-style-type: none"> <li>■ 用户模式期间, CRC_ERROR 管脚所使用的 CRAM 错误检查的能力 (32 bit CRC)。</li> <li>■ 仅有一个 32 bit CRC 值。该值涵盖了所有的 CRAM 数据。</li> </ul>	<ul style="list-style-type: none"> <li>■ 内嵌在每一个配置数据帧中 16 bit CRC。</li> <li>■ 配置期间, 一帧数据加载到器件后, 预计算出的 CRC 被移入 CRC 电路。</li> <li>■ 与此同时, 对被移入数据帧的 CRC 值进行计算。如果预计算出的 CRC 值与计算出的 CRC 值不匹配, 则 nSTATUS 被置低。</li> <li>■ 每个数据帧均有一个 16 bit CRC。因此, 整个配置比特流有多个 16 bit CRC 值。</li> <li>■ 每个器件均有一个不同长度的配置数据帧。</li> </ul>

这一部分重点放在第一类型 — 用户模式期间的 32 bit CRC。

## 错误检测寄存器

错误检测电路中有两组 32 bit 寄存器, 用于存储计算出的 CRC 签名以及预计算出的 CRC 值。签名寄存器中的一个非零值会导致 CRC\_ERROR 管脚被置高。

图 9-1 显示了错误检测模块与两个相关的 32 bit 寄存器 (签名寄存器与存储寄存器) 的结构框图。

图 9-1. 错误检测模块结构框图

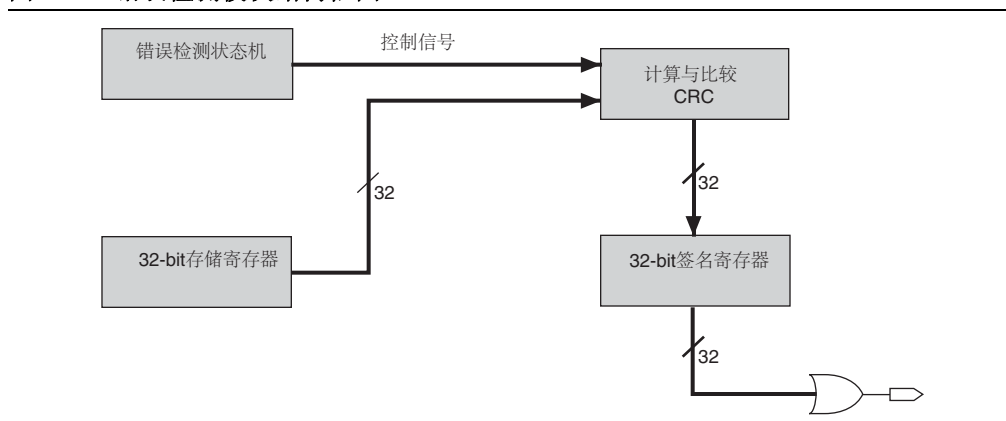


表 9-4 对图 9-1 中所示的寄存器作了定义。

表 9-4. 错误检测寄存器

寄存器	功能
32 bit 签名寄存器	该寄存器包含 CRC 签名。签名寄存器包含用户模式下计算出的 CRC 值与预计算出的 CRC 值的比较结果。如果没有检测到错误，则签名寄存器全部为零。一个非零签名寄存器表明了配置 CRAM 数据中存在错误。 CRC_ERROR 信号的产生取决于该寄存器的数据内容。
32 bit 存储寄存器	在配置阶段的最后，将 32 bit 预计算出的 CRC 签名加载到该寄存器。该签名然后在用户模式期间加载到 32 bit CRC 电路（称为“计算与比较 CRC”模块，如图 9-1 所示），以计算 CRC 错误。该寄存器在 CHANGE_EDREG JTAG 指令执行期间形成了一个 32 bit 扫描链。CHANGE_EDREG JTAG 指令可以更改存储寄存器的内容。因此，通过在操作期间执行这些指令来注入一个错误，从而对错误检测 CRC 电路进行系统检查。当发出 CHANGE_EDREG 指令时，器件的操作没有停止。

## 错误检测时序

通过 Quartus II 软件使能错误检测 CRC 功能时，器件会自动激活 CRC 进程，并在配置和初始化完成后进入用户模式。

在错误检测电路检测到以前的 CRC 计算中的损坏位之前，CRC\_ERROR 管脚一直驱动为低电平。CRC\_ERROR 管脚变为高电平后，在下一个 CRC 计算期间一直保持在高电平。该管脚不记录以前的 CRC 计算。如果新的 CRC 计算不包含任何的损坏位，CRC\_ERROR 管脚被拉低。错误检测继续运行，直到器件被重置。

错误检测电路通过用于设定最大频率的除数，来关闭内部配置振荡器。

表 9-5 列出了最小和最大错误检测频率。

表 9-5. Cyclone IV 器件的最小和最大错误检测频率

错误检测频率	最大错误检测频率	最小错误检测频率	有效除数 (2 <sup>n</sup> )
80 MHz/2 <sup>n</sup>	80 MHz	312.5 kHz	0, 1, 2, 3, 4, 5, 6, 7, 8

通过在 Quartus II 软件中指定一个分频因子，您可以设定一个较低的时钟频率（要了解更多信息，请参考“软件支持”）。该除数是 2 的指数，其中 *n* 介于 0 和 8 之间。除数范围从 1 到 256。请参考公式 1。

公式 1.

$$\text{Error detection frequency} = \frac{80 \text{ MHz}}{2^n}$$

CRC 计算时间取决于器件与错误检测时钟频率。

表 9-6 列出了 Cyclone IV 中，使用最小和最大时钟频率的每个 CRC 计算所需的估算时间。

表 9-6. CRC 计算时间

器件		最短时间 (ms) (1)	最长时间 (s) (2)
Cyclone IV E	EP4CE6 (3)	5	2.29
	EP4CE10 (3)	5	2.29
	EP4CE15 (3)	7	3.17
	EP4CE22 (3)	9	4.51
	EP4CE30 (3)	15	7.48
	EP4CE40 (3)	15	7.48
	EP4CE55 (3)	23	11.77
	EP4CE75 (3)	31	15.81
	EP4CE115 (3)	45	22.67
Cyclone IV GX	EP4CGX15	6	2.93
	EP4CGX22	12	5.95
	EP4CGX30	12	5.95
		34 (4)	17.34 (4)
	EP4CGX50	34	17.34
	EP4CGX75	34	17.34
	EP4CGX110	62	31.27
	EP4CGX150	62	31.27

**表 9-6 注释：**

- (1) 最短时间对应最大错误检测时钟频率，并随着 PVT（工艺、电压和温度）的不同会有相应的变化。
- (2) 最长时间对应最小错误检测时钟频率，并随着 PVT（工艺、电压和温度）的不同会有相应的变化。
- (3) 仅适用于采用 1.2-V 核心电压的器件。
- (4) 仅适用于 F484 器件封装。

## 软件支持


在 Quartus II 软件中使能 CRC 错误检测功能，将生成可选的两用 CRC\_ERROR 管脚上的 CRC\_ERROR 输出。

要通过 CRC

使能错误检测功能，需执行下列步骤：

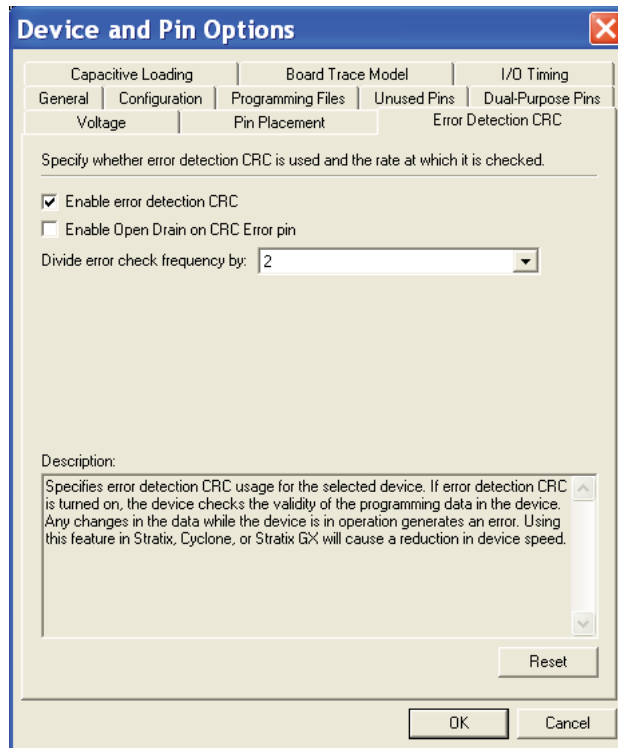
1. 打开 Quartus II 软件，使用 Cyclone IV 器件加载一个工程。
2. 在 Assignments 菜单上，点击 **Settings**。然后出现 **Settings** 对话框。
3. 在 Category 列表中，选择 **Device**。然后出现 **Device** 页面。
4. 点击 **Device and Pin Options**，出现如图 9-2 中所示的 **Device and Pin Options** 对话框。
5. 在 **Device and Pin Options** 对话框中，点击 **Error Detection CRC** 标签。
6. 开启 **Enable error detection CRC**。

7. 在 Divide error check frequency by 设置框中，输入一个有效除数（如第 9-5 页表 9-5 所示）。

 除数值用于配置振荡器输出时钟的分频。该输出时钟用作错误检测进程的时钟源。

8. 点击 OK。

图 9-2. 在 Quartus II 软件中使能错误检测 CRC 功能

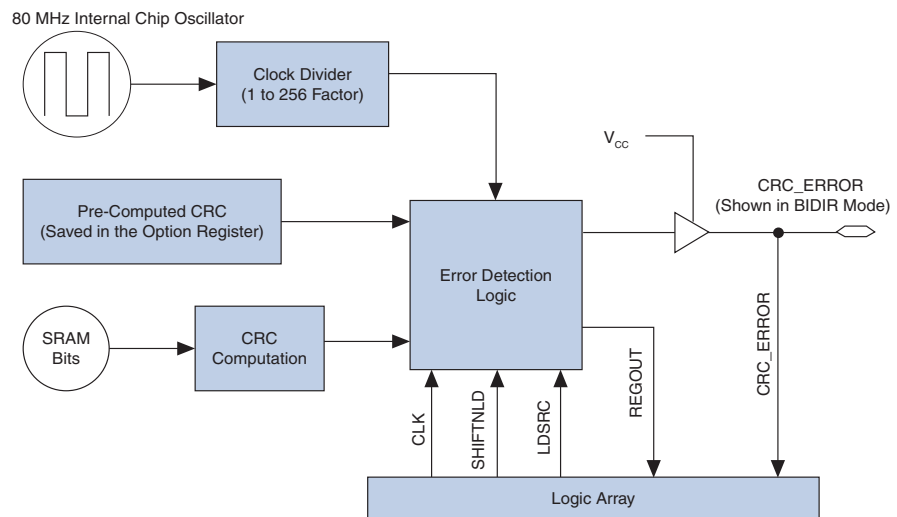



## 通过用户逻辑访问错误检测模块

错误检测电路将计算后的 32 bit CRC 签名存储在 32 bit 寄存器中，内核中的用户逻辑读出这些 32 bit CRC 签名。cycloneiv\_crcblock 原语是一个 WYSIWYG (所用即所得) 组件，用于建立从用户逻辑到错误检测电路的接口。cycloneiv\_crcblock 原语组合电路单元包括输入与输出端口（必须包含在组合电路单元中）。要访问逻辑阵列，cycloneiv\_crcblock WYSIWYG 组合电路单元必须加入到您的设计当中。

图 9-3 显示了 FPGA 器件的错误检测结构框图，以及 WYSIWYG 组合电路单元所使能的设计中的接口。

图 9-3. 错误检测结构框图



 软错误会对用户逻辑产生影响，所以不应该依赖通过 regout 读出 32 bit CRC 签名来检测软错误，而是依赖 CRC\_ERROR 输出信号本身，因为 CRC\_ERROR 输出信号不会被软错误影响。

要使能 cycloneiv\_crcblock WYSIWYG 组合电路单元，您必须相应地对每一个 Cyclone IV 器件命名组合电路单元。

例 9-1 显示了如何定义 Cyclone IV 器件中 WYSIWYG 组合电路单元的输入与输出端口的实例。

#### 例 9-1. 定义 Cyclone IV 器件中 WYSIWYG 组合电路单元的输入与输出端口的实例

```
cycloneiv_crcblock<crcblock_name>
(
  .clk(<clock source>),
  .shiftnld(<shiftnld source>),
  .ldsrc(<ldsrc source>),
  .crcerror(<crcerror out destination>),
  .regout(<output destination>),
);
```

表 9-1 列出了必须包含在组合电路单元中的输入与输出端口。

表 9-1. CRC 模块输入与输出端口

端口	输入 / 输出	定义
<code>&lt;crcblock_name&gt;</code>	输入	CRC 模块的唯一标识符，对于给定描述语言（例如：Verilog HDL、VHDL 和 AHDL），代表合法的标识符名称。此字段是必需的。
<code>.clk(&lt;clock source&gt;</code>	输入	该信号指定该单元的时钟输入。该单元的所有操作都在时钟的上升沿。无论是加载到单元的数据还是来自单元的数据，该信号始终出现在上升沿。此端口是必需的。
<code>.shiftnld (&lt;shiftnld source&gt;)</code>	输入	该信号是到错误检测时钟的一个输入。若 <code>shiftnld=1</code> ，则在每一个 <code>clk</code> 的上升沿数据都会从内部移位寄存器移入 <code>regout</code> 。若 <code>shiftnld=0</code> ，则根据 <code>ldsrc</code> 端口输入，移位寄存器并行加载预先计算出的 CRC 值，或者更新寄存器内容。要做到这一点， <code>shiftnld</code> 必须被驱动为低电平至少两个时钟周期。此端口是必需的。
<code>.ldsrc (&lt;ldsrc source&gt;)</code>	输入	该信号是到错误检测时钟的一个输入。若 <code>ldsrc=0</code> ，则预先计算出的 CRC 寄存器被选择用于 <code>shiftnld=0</code> 时，在 <code>clk</code> 的上升沿加载到 32 bit 移位寄存器。若 <code>ldsrc=1</code> ，则签名寄存器（CRC 计算的结果）被选择用于 <code>shiftnld=0</code> 时，在 <code>clk</code> 的上升沿加载到移位寄存器。当 <code>shiftnld=1</code> 时，该端口被忽略。该端口是必需的。
<code>.crcerror (&lt;crcerror indicator output&gt;)</code>	输出	此信号是单元的输出，同步到器件的内部振荡器（80-MHz 内部振荡器），而不是同步到 <code>clk</code> 端口。如果错误模块检测到 SRAM 位已经翻转，并且对于预先计算出的值，内部 CRC 计算显示了与预先计算出的值的差异，那么此信号被置高。您必须将此信号连接到一个输出管脚或者一个双向管脚。如果连接到一个输出管脚，那么您只能监控 <code>CRC_ERROR</code> 管脚（内核不能够访问此输出）。如果 <code>CRC_ERROR</code> 信号被内核逻辑用于读取错误检测逻辑，那么您必须将此信号连接到 <code>BIDIR</code> 管脚。通过连接 <code>BIDIR</code> 管脚，该信号被直接提供给内核（ <code>BIDIR</code> 管脚具有它本身的，连接到 <code>V<sub>CC</sub></code> 的输出使能端口。请参见第 9-8 页图 9-3）。
<code>.regout (&lt;registered output&gt;)</code>	输出	此信号是错误检测移位寄存器的输出，该错误检测移位寄存器同步到 <code>clk</code> 端口，以使内核逻辑对其进行读取。此信号在每个周期移动一位，所以您应该同步 <code>clk</code> 信号 31 个周期，以读取移位寄存器的 32 个位。

## 从 CRC 错误中恢复

Altera FPGA 所存在的系统必须对器件重配置进行控制。在 `CRC_ERROR` 管脚上检测到错误后，系统可以安全地重配置 FPGA 时，选通 `nCONFIG` 为低电平将指示系统执行重配置。

当通过重配置器件，使用正确的值写入数据位时，器件正常工作。

当软错误在 Altera 器件中屡见不鲜时，某些高可靠性的应用可能需要一个设计来应对这些错误。

## 文档修订历史

表 9-2 显示了本章节的修订历史。

表 9-2. 文档修订历史

日期	版本	修订内容
2010 年 2 月	1.1	针对 Quartus II 9.1 SP1 发布的更新： <ul style="list-style-type: none"><li>■ 更新了“配置错误检测”部分。</li><li>■ 更新了表 9-6。</li><li>■ 在表 9-6 中添加 Cyclone IV E 器件。</li></ul>
2009 年 11 月	1.0	首次发布。