crc MegaCore Function Parameterized CRC Generator / Checker

Solution Brief 30 August 1997, ver. 1

Target Applications:
Communications

Family: FLEX 10K, FLEX 8000 & FLEX 6000

Vendor:

101 Innovation Drive
San Jose, CA 95134
Tel. (408) 544-7000
WWW http://www.altera.com

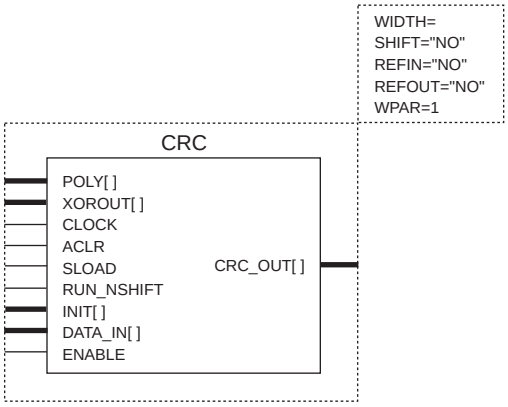
Features

- crc MegaCore™ function, general-purpose cyclic redundancy code (CRC) generator and checker
- Optimized for the FLEX® device architecture
- High-speed operation, over 100 MHz for many configurations
- Fully parameterized, including:
  - Any length generator polynomial
  - Input data width, from 1 bit to the width of the polynomial
  - Any initial value
- Built-in support for:
  - Inverting output data
  - Reflecting (reversing bit order) input and output data
- Risk-free evaluation with the OpenCore™ feature available with MAX+PLUS II

General Description

The crc MegaCore function is a general-purpose CRC generator and checker that validates data frames and ensures that data corruption during transmission is detected. The crc function is fully parameterized, and therefore can be used in virtually any design that requires a CRC checker. See Figure 1.

Figure 1. crc Symbol



Functional Description

The crc function validates data streams via redundant encoding. Similar to parity checking, CRC encoding is a method of generating a code to verify the integrity of the data stream. However, while parity checking uses one bit to indicate even or odd parity, CRC encoding uses multiple bits, and therefore is more likely to catch errors in the data stream.

CRCs are particularly effective for two reasons:

- CRCs provide excellent protection against common errors such as burst errors, in which consecutive bits in a data stream are corrupted during transmission.
- The original data is the first part of the transmission, which makes systems that use CRCs easy to understand and implement.

Because the `crc` function is fully parameterized, virtually any CRC algorithm can be implemented using the parameters available with this function. To maximize flexibility, the `crc` function also allows designers to set port values, e.g., initial register values can be set via the `init[]` input.

The designer can define the size of the `crc` function's generator polynomial to meet design specifications. The larger the CRC polynomial length, the greater the chance of transmission error detection.

A vector generation program, available with the `crc` function, has the same parameters as the `crc` function and generates vector files to verify the operation.

Performance

Table 1 lists sample `crc` function performance and logic cell usage.

Table 1. Sample crc MegaCore Function Performance & Logic Cell Usage

CRC Configuration with FLEX Devices *Note (1)*		Size (LEs) *Note (2)*	Performance (MHz) *Note (2)*	Performance (Mbits/s)
CRC-32 generator polynomial	32-bit wide Input	318	28	896
	8-bit wide input	87	70	560
	1-bit wide input	32	>125	>125
CRC-16/CCITT generator polynomial	16-bit wide input	39	75	1200
	8-bit wide input	24	100	800
	1-bit wide input	16	>125	>125

Notes:
(1) When calculating performance, the fastest speed grade from the FLEX 10K, FLEX 8000, and FLEX 6000 device families was used.
(2) The size and performance of the `crc` function will vary depending on the logic synthesis settings, device fitting, and the chosen polynomial.

Go to the *crc MegaCore Function Parameterized CRC Generator/Checker Data Sheet* for more information.

Reference

Williams, Ross N. *A Painless Guide to CRC Error Detection Algorithms.* Version 3. Hazelwood Park, Australia: Rocksoft PTY Ltd, 1996.

This document explains CRCs and their table-driven implementations, and also provides a generic parameterized model CRC algorithm. For more information, go to the Rocksoft world-wide web site at: **http://www.rocksoft.com.**

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
http://www.altera.com