Mercury Gigabit Transceiver MegaCore Function (M1GXCVR)

November 2001; ver. 1.02 Data Sheet

Introduction

Using an industry standard compliant look-up table (LUT) encoding scheme, the Altera® Mercury™ Gigabit Transceiver MegaCore® Function (M1GXCVR) enables data transfers across a high speed serial link. It improves the transmission quality on a serial link by using 8b10b encoding to ensure a sufficient transition density and limited run length—no more than five consecutive 1s or 0s—simplifying clock recovery and error detection, and eliminating direct current (DC) components from the serial stream. The M1GXCVR comprises customizable receiver (RXM1GXCVR) and transmitter (TXM1GXCVR) variants that can function independently for half-duplex operation, or can be combined for full-duplex operation.

Features

- 1.25 gigabit per second (Gbps) serial high-speed differential interface (HSDI)
- Uses serializer/deserializer (SERDES) and clock & data recovery (CDR) circuits that are built into the Altera Mercury device architecture.
- Configurable for 1 to 18 channels
- Industry compatible special character coding
- Complies with all applicable standards, including:
  - Institute of Electrical and Electronics Engineers, IEEE 802.3z, *Media Access Control (MAC) Parameters, Physical Layer, Repeater and Management Parameters for 1000 Mb/s Operation*, 1998, paragraphs 36.2.4.1 to 36.2.4.6.
- Quartus® II software, and OpenCore® feature allow place-and-route, and static timing analysis of designs prior to licensing
- Secure register transfer level (RTL) simulation models allow simulation with user design in third-party simulators

The M1GXCVR is optimized for the Mercury device architecture, and is dependent on its hardware features. The EP1M350 device can support up to 18 input and 18 output dedicated high-speed channels. The EP1M120 device can support up to 8 input and 8 output dedicated high-speed channels.

For more information on the Altera Mercury device family, refer to the *Mercury Programmable Logic Device Family Data Sheet*.

A-DS-IPM1GXCVR-1.02

Typical Application

Figure 1 shows an example system transmitting 64 bits of data at 125 MHz across eight TXM1GXCVR channels, and being received by eight RXM1GXCVR channels, for an aggregated data rate of 8 Gbps.

Figure 1. Typical Application

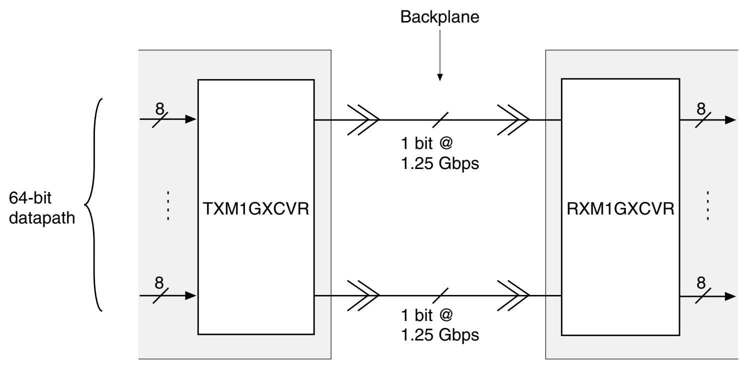
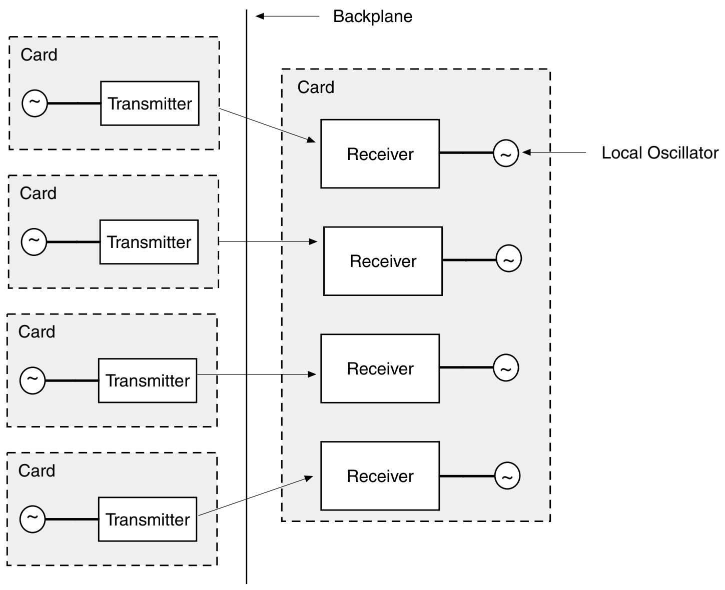


Figure 2 illustrates the system design, including separate local oscillators for each card.

Figure 2. System Design



In system level applications, the transmitted data rate must be less than or equal to the receiver's data handling capacity. This can be accomplished by inserting IDLE characters in the transmit stream. The user must determine whether idle characters are required, depending on the application. For example, if the application uses a star topology with multiple cards running on different voltage controlled oscillators (VCOs) inserting IDLE characters guarantees that the receiver's data capacity exceeds the transmitted data rate. See "Character Codes" on page 4.

Generating Variants

To obtain a customized M1GXCVR variant send a request to telecom@altera.com specifying the required number of channels. Your customized M1GXCVR will be e-mailed to you as an encrypted gate level netlist and secure RTL simulation model, in one business day.

Table 1 shows the optional features available to generate all variants. See Table 5 on page 13 for an estimate of the resources consumed by each variant.

Table 1. Optional Feature

Option	Parameter	Choices
Channel(s)	NCHAN	`1-18`

Functional Description

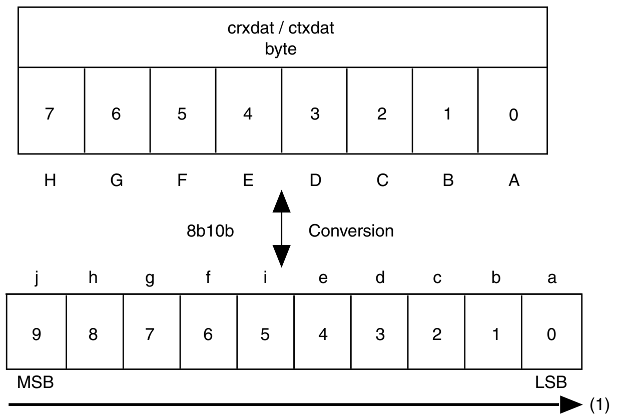
The RXM1GXCVR performs clock and data (CDR) recovery on a received serial bit stream of up to 1.25 Gbps. Then, it finds word alignment and decodes the received 10-bit codes into 8-bit data and control characters.

The TXM1GXCVR encodes a stream of 8-bit data and control characters into a neutral average disparity and run length limited 10-bit code. It then serializes and transmits this code at up to 1.25 Gbps.

The least significant bit (LSB) is always transmitted first, while the most significant bit (MSB) is always transmitted last.

Figure 3 illustrates the bidirectional conversion process.

Figure 3. M1GXCVR Conversion



Note:
(1) The LSB is transmitted/received first; the MSB is transmitted/received last.

Disparity

Disparity is the difference between the number of 1s and 0s in the encoded word.

- Neutral disparity indicates the number of 1s and 0s are equal.
- Positive disparity indicates more 1s than 0s.
- Negative disparity indicates more 0s than 1s.

The M1GXCVR is designed to maintain a neutral average disparity. Average disparity determines the DC component of a serial line. Running disparity is a record of the cumulative disparity of every encoded word. Running disparity is tracked by the encoder. To guarantee neutral average disparity, a positive running disparity must be followed by neutral or negative disparity; a negative running disparity must be followed by neutral or positive disparity. If these conditions are not met, the receiver flags an error.

Character Codes

In addition to the 256 data characters, the stream contains twelve out-of-band indicators, also called special control (K) characters. Special K characters can be used, for example, as packet delimiters, as test data, or as IDLE indicators. The comma character (K28.5) is used for alignment purposes as its 10-bit code is guaranteed not to occur elsewhere in the encoded bit stream, except after K28.7 which is normally only sent during diagnostic.

Table 2 lists the special K codes used by the M1GXCVR.

Table 2. Character Codes

10-Bit Special K Codes	Equivalent 8-Bit Codes
K28.0	8'b000_11100
K28.1	8'b001_11100
K28.2	8'b010_11100
K28.3	8'b011_11100
K28.4	8'b100_11100
K28.5 *(1)*	8'b101_11100
K28.6	8'b110_11100
K28.7	8'b111_11100
K23.7	8'b111_10111
K27.7	8'b111_11011
K29.7	8'b111_11101
K30.7	8'b111_11110

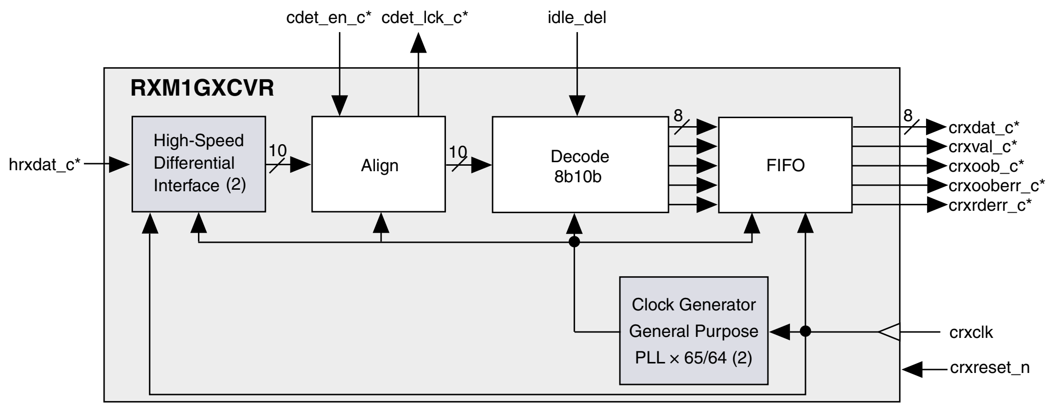
Note:
(1) K28.5 is a comma character used for alignment purposes, and to represent the IDLE code.

Receiver Description

The RXM1GXCVR takes in a serial bit stream and performs clock and data recovery, serial to parallel conversion, and 8b10b decoding. The maximum core frequency is 125 MHz. Figure 4 shows a block diagram of the RXM1GXCVR.

☞ In cases where many channels are used, the RXM1GXCVR does not align the data received on the different channels or compensate for the skew between the channels.

Figure 4. Receiver Block Diagram *Note (1)*



Notes:
(1) The * is equivalent to a number from 0–17, depending on the number of channels chosen as an option, see Table 1. Each channel has a separate external set of signals.
(2) The high-speed differential interface (HSDI) and the general purpose PLL sub-blocks are hard macros, the other sub-blocks are soft macros.

High-Speed Differential Interface (HSDI)

The HSDI is a hard macro for the Mercury devices, it handles serial data transfers of up to 1.25 Gbps per channel. It is instantiated by an ALTCDR megafunction.

The ALTCDR megafunction is a hard macro, preconfigured by the Quartus II development tool, and included as part of the netlist. Each Mercury part allows only one RX ALTCDR macro, therefore a user cannot replicate multiple instances of this function.

The CDR and the deserializer circuits are embedded within the HSDI macro.

Clock & Data Recovery

The CDR recovers a clock from the incoming serial stream, and uses it to resample the data. This eliminates the need to carry a separate clock signal with the data, and prevents any potential skew problems.

Deserializer

The deserializer circuit takes in a serial input stream and deserializes the data into 10-bit words. The LSB is received first.

For more information on these functions, refer to the *CDR in Mercury Devices Application Note #130*.

Clock Generator

The clock generator circuit is external to the HSDI. Using one of the on-device, general purpose phase-locked loops (PLLs) from the Mercury device—independent from the PLLs of the HSDI—the clock generator circuit multiplies the core clock frequency of 125 MHz by 101.5% (PLL ratio of 65/64). This generates a slightly fast clock that runs the majority of the receiver's internal logic, while the 125 MHz core clock is used to read data out of the first-in first-out (FIFO) buffer.

To avoid an overflow condition where data could potentially be lost, data is extracted from the HSDI's internal FIFO buffer at a frequency of 101.5% of the core clock. This feature is included as part of the macro.

☞ The general purpose PLL is a hard macro included as part of the netlist. The PLL is preconfigured by ALTCLKLOCK in the Quartus II development tool.

☞ Each Mercury device has only two dedicated HSDI PLLs, the M1GXCVR uses one for the receiver and one for the transmitter, therefore only one receive clock domain is possible.

Alignment

To achieve character alignment, the function searches for the special IDLE character, the comma character K28.5.

Asserting the `cdet_en` signal initiates the search for the comma character. Once the alignment circuit has locked onto the proper pattern, it asserts the `cdet_lck` signal.

☞ The aligner locks onto the first received comma character and keeps the same alignment until `cdet_en` is cleared to zero and reasserted.

Decoder

Data and identified 10-bit special K codes are converted from 10 bits to 8 bits, see Table 2 for a list of the valid K codes, and Figure 3 for an illustration of the conversion process.

When special 10-bit K codes are received, the special K codes are translated to 8-bit values, and the `crxoob` signal is asserted. The decoder also checks for invalid 10-bit codes, and asserts the `crxooberr` signal when invalid codes are detected.

When the `idle_del` signal is asserted, IDLE characters (K28.5) are automatically deleted and provide for rate adaptation.

When the receiver detects a disparity error, the `crxrderr` signal is asserted.

FIFO Buffer

Since no phase relationship exists between the core clock and the internally generated fast clock, a FIFO buffer is required for data to cross from the fast clock domain to the core clock domain.

The valid signal, `crxval`, is asserted when valid data is available in the FIFO buffer. Two conditions can cause the FIFO buffer to empty:

- IDLE characters are received and deleted because `idle_del` is asserted;
- The serial high-speed clock frequency is slower than ten times the core clock frequency.

Channel Alignment

The receiver only implements character alignment. If a multi-character word is sent across multiple channels, there is no guaranty that the characters corresponding to the same word will be received and put out by the receiver at the same clock cycle. It is up to the communication protocol to provide a way to achieve channel alignment.

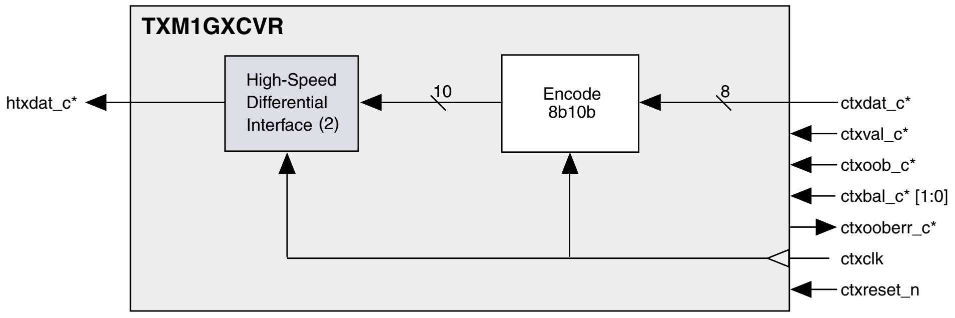
Assuming that the `idle_del` signal is kept asserted, channel alignment can be achieved by transmitting a series of 64 or more IDLE words (IDLE characters on every channel) before resuming the transmission of non-IDLE words (valid non-IDLE data on every channel). Since all the IDLE characters will be deleted, this will completely empty the receiver's FIFO buffers. The first characters to come out of each channel of the receiver will correspond to the first non-IDLE transmitted multi-character word. As long as IDLEs are always transmitted on all the channels at the same time, channel alignment will be maintained until a transmission error occurs.

Channel alignment can also be achieved with less overhead by using one of the non-IDLE control characters as a marker. For example, if control character Kx.y is always sent on all the channels at the same time, the receiving circuit can easily align the channels, and/or detect channel misalignment based on the received Kx.y control characters.

Transmitter Description

The TXM1GXCVR takes a stream of 8-bit bytes, encodes them into 10-bit words, and performs parallel to serial conversion. The converted 10-bit code is output on a serial line. Figure 5 shows a block diagram of the TXM1GXCVR.

Figure 5. Transmitter Block Diagram *Note (1)*



Notes:
(1) The * is equivalent to a number from 0–17, depending on the number of channels chosen as an option, see Table 1. Each channel has a separate external set of signals.
(2) The high-speed differential interface (HSDI) sub-block is a hard macro, the encode sub-block is a soft macro.

High-Speed Differential Interface (HSDI)

The HSDI is a hard macro for the Mercury devices. It handles serial data transfers of up to 1.25 Gbps per channel. The serializer and clock multiplier circuits are embedded within the HSDI macro, along with a seven word FIFO buffer.

Each Mercury device has only two dedicated HSDI PLLs, the M1GXCVR uses one for the receiver and one for the transmitter, therefore only one transmit clock domain is possible.

Serializer

The serializer takes a 10-bit input stream and serializes it. The 125 MHz core clock is boosted by a factor of 10 to be used as the high-speed output clock. Thus, the resulting serial data rate is 1.25 Gbps. The LSB of the encoded word (bit “a” in Figure 3) is transmitted first.

The ALTCDR function is a hard macro, preconfigured by the Quartus II development tool, and included as part of the netlist. Each Mercury part allows only one TX ALTCDR macro, therefore a user cannot replicate multiple instances of this function.

Encoder

The encoder uses a non-partitioned memory-based LUT implementation to convert data and identified out-of-band 8-bit codes from 8 bits to 10 bits. See Figure 3 for an illustration of the conversion process.

To encode an 8-bit word, the 8-bit value must be applied to the `ctxdat` inputs, and the `ctxval` input must be asserted (active high). When `ctxval` is not asserted, IDLE (K28.5) characters are inserted.

When a special 10-bit code is to be inserted, the equivalent out-of-band 8-bit code is placed on the `ctxdat` lines with the `ctxoob` asserted. Error checking is performed to ensure the out-of-band 8-bit code is valid, see Table 2 for a list of the K codes.

Disparity

The running disparity can be forced to positive or negative on the 8-bit word. This allows for a special resynchronization pattern to be inserted by the user. The following codes asserted on inputs `ctxbal_c*` in conjunction with the data—force the 10-bit code to be set to the desired running disparity:

- 2'b00 - neutral disparity
- 2'b01 - positive disparity
- 2'b10 - negative disparity
- 2'b11 - reserved for future use

I/O Signals

Table 3 shows the port list for the RXM1GXCVR function.

Table 3. RXM1GXCVR I/O Signals (Part 1 of 2)

Port	Direction	Description
Receive Interface Signals		
`crxclk`	Input	Receiver clock. This is the receiver's main system clock. It is used to clock data out of the receiver's FIFO buffers, and as reference for the HSSI PLL and the general purpose PLL to generate the high speed serial clock (x 10) and the fast clock (x 65/64 or 101.5%).
`crxreset_n`	Input	Active low synchronous reset. `crxreset_n` must be held low for at least five `crxclk` clock cycles for the receiver to be properly reset.
`crxval_c*`*(1)*	Output	Valid. This signal is asserted when the receiver's FIFO buffer is not empty to indicate that the signals `crxdat_c*`, `crxoob_c*`, `crxooberr_c*` and `crxrderr_c*` are valid.
`crxdat_c*`*(1)* `[7:0]`	Output	Data bus. This bus holds the decoded value of the received data or control character. If `crxooberr_c*` is asserted, the value of `crxdat_c*` is undefined.
`crxoob_c*`*(1)*	Output	Out-of-band. This signal is asserted when a valid control character is available on the `crxdat_c*` data bus. If `crxooberr_c*` is asserted, the value of `crxoob_c*` is undefined.
`crxooberr_c*`*(1)*	Output	Out-of-band error. This signal is asserted when an invalid ten bit code is received. If `crxooberr_c*` is asserted, the value of `crxdat_c*` and `crxoob_c*` are undefined.
`crxrderr_c*`*(1)*	Output	Running disparity error. This signal is asserted when a running disparity error has been detected.
High-Speed Receive Interface Signals		
`hrxdat_c*`*(1)*	Input	Serial data line.
Miscellaneous Signals		
`cdet_en_c*`*(1)*	Input	Comma detect enable. When a low to high transition of this signal is detected, comma detection is initiated. The aligner locks on the following detected comma character (K28.5), and remains locked until this signal is cleared to zero and reasserted and a new comma is detected. This is an asynchronous signal. It is metastable hardened before it is fed to the aligner so there is a delay of three clock cycles of the fast (101.5%) clock before it takes effect. Thus, it should always be kept stable for at least three `crxclk` clock cycles. If `cdet_en_c*` is cleared before a comma character is received, the aligner keeps its current alignment.

Table 3. RXM1GXCVR I/O Signals (Part 2 of 2)

`cdet_lck_c*`*(1)*	Output	Comma detect locked. This signal is asserted when the aligner has found a comma and alignment has been achieved. This signal is synchronous to the internal fast (101.5%) clock and as such should be considered asynchronous to the `crxclk` clock. It should be metastable hardened if it is to be used in the `crxclk` clock domain.
`idle_del`	Input	Idle character delete. When this signal is asserted, the receiver deletes any IDLE character (K28.5) received. This is an asynchronous signal. It is metastable hardened before it is fed to the decoder so there is a delay of three clock cycles of the fast (101.5%) clock before it takes effect. Thus, it should always be kept stable for at least three `crxclk` clock cycles.

Note:
(1) The * is equivalent to a number from 0–17, depending on the number of channels chosen as an option, see Table 1. Each channel has a separate external set of signals.

Table 4 shows the port list for the TXM1GXCVR function.

Table 4. Transmitter Function I/O Signals (Part 1 of 2)

Port	Direction	Description
Transmit Interface Signals		
`ctxclk`	Input	Transmit clock. This is the transmitter's main system clock, it is also used as the reference clock for the HSDI PLL.
`ctxreset_n`	Input	Active low synchronous reset. `ctxreset_n` must be held low for at least five `ctxclk` clock cycles for the transmitter to be properly reset.
`ctxval_c*`*(1)*	Input	Valid signal. This signal must be asserted to indicate that the character defined by the value of `ctxdat_c*` and `ctxoob_c*` should be transmitted. When `ctxval_c*` is not asserted, IDLE characters (K28.5) are transmitted.
`ctxdat_c*`*(1)* [7:0]	Input	Data bus. When `ctxval_c*` is asserted, the value on the `ctxdat_c*` bus determines which character will be transmitted. When `ctxval_c*` is not asserted, the value on the `ctxdat_c*` bus is ignored and IDLE characters (K28.5) are transmitted.
`ctxoob_c*`*(1)*	Input	Out-of-band selector. When `ctxval_c*` is asserted, `ctxoob_c*` selects between control character (`ctxoob_c*` asserted) or data character (`ctxoob_c*` deasserted). When `ctxval_c*` is not asserted, `ctxoob_c*` is ignored and IDLE characters (K28.5) are transmitted.

Table 4. Transmitter Function I/O Signals (Part 2 of 2)

`ctxbal_c*`*(1)* `[1:0]`	Input	Disparity balance bus. This 2-bit input bus is used to specify the disparity of the transmitted characters. If set to 2'b00, the disparity of the transmitted character is based on the running disparity to maintain a neutral average disparity. If set to 2'b10, the transmitted character's negative disparity encoding, if existent, is used for transmission (some characters have a single neutral disparity encoding). If set to 2'b01, the transmitted character's positive disparity encoding, if existent, is used for transmission (some characters have a single neutral disparity encoding). The value 2'b11 is reserved for future use and should not be used.
`ctxooberr_c*`*(1)*	Output	Out-of-band error indicator. This signal is asserted when an attempt to transmit an invalid control character is made, i.e. `ctxval_c*` and `ctxoob_c*` are asserted, and `ctxdat_c*` is not one of the 12 valid codes.
High-Speed Transmit Interface Signals		
`htxdat_c*`*(1)*	Output	Serial data output.

Note:
(1) The * is equivalent to a number from 0–17, depending on the number of channels chosen as an option, see Table 1. Each channel has a separate external set of signals.

Resource Usage

Table 5 shows the estimated resources a single channel M1GXCVR function consumes in a Mercury device.

Table 5. Resources Per Channel Notes (1) (2)

Module	LEs	ESBs	General Purpose PLLs
RXM1GXCVR	387	1/2	1
TXM1GXCVR	53	1/2 *(3)*	0

Notes:
(1) The numbers for the logic elements (LEs) and embedded system blocks (ESBs) are approximate as of November 2001.
(2) A linear relationship exists between the resources and the number of channels. Both the LE and ESB counts are to be multiplied by the number of channels.
(3) One ESB is shared between two transmitters.

Performance

Table 6 shows the speed at which the M1GXCVR function operates in a Mercury device.

Table 6. Performance

	M1GXCVR		HSDI (3)
Module	Internal Frequency (MHz)	Local Frequency (MHz)	f_{MAX} (MHz)
RXM1GXCVR	126.95	125.00(1)	1,250.00
TXM1GXCVR	125.00	125.00(2)	1,250.00

Notes:
(1) As measured on the `crxclk` pins.
(2) As measured on the `ctxclk` pins.
(3) These inputs and outputs need to comply with the frequency range of the Mercury device.

Licensing

A license is not required to perform the following trial operations using your own custom logic:

- Instantiation
- Place-and-route
- Static timing analysis
- Simulation on a third-party simulator

When you are ready to generate programming files, you need to obtain licenses through your local Altera sales representative.

☞ All current M1GXCVR variants use a single license with ordering code: IP–M1GXCVR.

✓ To obtain a customized M1GXCVR variant send a request to telecom@altera.com specifying the required parameter: **NCHAN**.

☞ Remember to include your preferred return e-mail address as your customized M1GXCVR will be e-mailed to you as an encrypted gate level netlist and secure RTL simulation model, in one business day.

Deliverables

The following elements are provided with the package:

- Data Sheet
- Encrypted gate level netlist & ROM file
- Place-and-route constraints (where necessary)
- Secure RTL simulation model
- Demo testbench

☞ The **Readme** file included within the package provides detailed installation instructions, and useful tips for using the M1GXCVR MegaCore function.

ALTERA®

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
http://www.altera.com
Applications Hotline:
(800) 800-EPLD
Customer Marketing:
(408) 544-7104
Literature Services:
lit_req@altera.com