Product Obsolete/Under Obsolescence

XILINX®

XAPP151 (v1.7) October 20, 2004

# Virtex Series Configuration Architecture User Guide

## Summary

The Virtex™ architecture supports powerful new configuration modes, including partial reconfiguration. These mechanisms are designed to give advanced applications access to and manipulation of on-chip data through the configuration interfaces.

This document is an overview of the Virtex architecture, emphasizing data bit location in the configuration bitstream. Knowing bit locations is the basis for accessing and altering on-chip data. FPGA applications can be built that change or examine the functionality of the operating circuit without stopping the circuit loaded in the device. A glossary is included to explain some of the terminology used in this application note.

## Introduction

### CLBs, IOBs, and Configurations

Each Virtex device contains configurable logic blocks (CLBs), input-output blocks (IOBs), block RAMs, clock resources, programmable routing, and configuration circuitry (Figure 1). These logic functions are configurable through the configuration bitstream. Configuration bitstreams contain a mix of commands and data. Configuration bitstreams can be read and written through one of the configuration interfaces on the device.

The Virtex, Virtex-E, and Virtex-E Extended Memory families differ primarily in the amount of block RAM available.

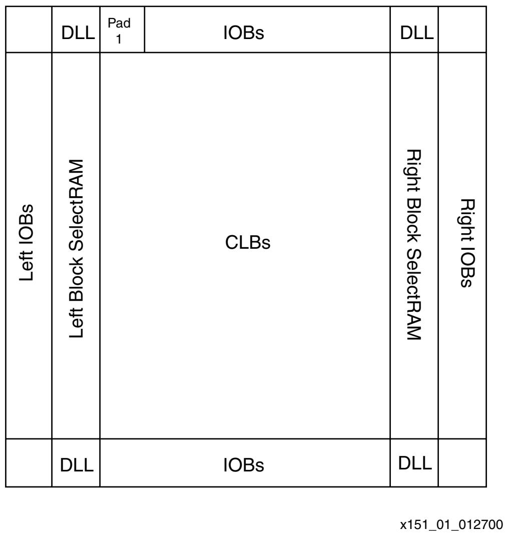

*Figure 1:* **Virtex Architecture Overview**

### Configuring Virtex Devices

Virtex devices can be configured through the SelectMAP™ interface, master/slave serial interfaces, or the Boundary-Scan interface. The collection of configuration bits is called a configuration bitstream. The bitstream is a series of configuration commands and configuration data, as shown in Figure 2. Bitstream architecture is discussed in **Configuration Logic Basics**, page 15.

| | | | | | | |
|---|---|---|---|---|---|---|
| **CMD 1** | **data** | **CMD 2** | **data** | **CMD 3** | **data** | **CMD 1** |

*Figure 2:* **Bitstream Example**

Warning! This document discusses mechanisms for manipulating the configuration bits for the Virtex devices. If portions of the bitstream other than those described here are altered, the device may be damaged. Xilinx is not liable for any consequences of misprogramming a device.

Writing some or all of a configuration is done by issuing configuration commands to the desired interface followed by the configuration data.

The SelectMAP interface is an 8-bit interface on the device with data pins labeled D[7:0]. The configuration bitstream can be written eight bits per clock cycle. Virtex devices can be configured to retain the (D[7:0], BUSY/DOUT, $\overline{\text{INIT}}$, $\overline{\text{WRITE}}$, and $\overline{\text{CS}}$) SelectMAP pins, allowing further re-configuration via those pins. If further re-configuration is not required, those pins can be configured as user I/O.

When the master/slave serial or Boundary Scan interface is used for configuration or re-onfiguration, the configuration bitstream is transmitted one bit per clock cycle.

Timing relationships for the configuration interfaces are discussed in the Virtex series data sheet located at **http://www.xilinx.com/bvdocs/publications/ds003.pdf**.

### Reading Configuration Bits From a Virtex Device

Configuration data can be read using the SelectMAP interface or the Boundary Scan interface. The master/slave serial interfaces can not be used to read a configuration. Reading all or some of a configuration is done by issuing configuration read commands to the desired interface, then reading the data from the same interface. Configuration commands and data formats are discussed in **Configuration Logic Basics**, page 15.

The SelectMAP interface has an 8-bit data port. To use the SelectMAP interface after configuration, all 12 SelectMAP pins must remain as SelectMAP as opposed to user I/O (using the BitGen option: `-g Persist:Yes`).

The Boundary Scan (JTAG) interface allows bit-serial access to the configuration. It is a permanent interface that is always present.

## Configuration Columns

The Virtex configuration memory can be visualized as a rectangular array of bits. The bits are grouped into vertical *frames* that are one-bit wide and extend from the top of the array to the bottom. A frame is the atomic unit of configuration - it is the smallest portion of the configuration memory that can be written to or read from.

Frames are grouped together into larger units called columns. In Virtex, Virtex-E, and Virtex-E extended memory devices, there are several different types of columns:

*Table 1:* **Configuration Column Type**

| Column Type | # of Frames | # per Device |
|---|---|---|
| Center | 8 | 1 |
| CLB | 48 | # of CLB columns |
| IOB | 54 | 2 |
| Block SelectRAM Interconnect | 27 | # of block SelectRAM columns |
| Block SelectRAM Content | 64 | # of block SelectRAM columns |

Each device contains one center column that includes configuration for the four global clock pins. Two IOB columns represent configuration for all of the IOBs on the left and right edges of the device. The majority of columns are CLB columns which each contain one column of CLBs and the two IOBs above and below those CLBs. The remaining two column types involve the block RAM: one for content and the other for interconnect. For each RAM column, one of each type is present (for values for all Virtex devices, see Table 3, "Virtex Series Devices," on page 6).

The columns for an sample Virtex device are shown below:

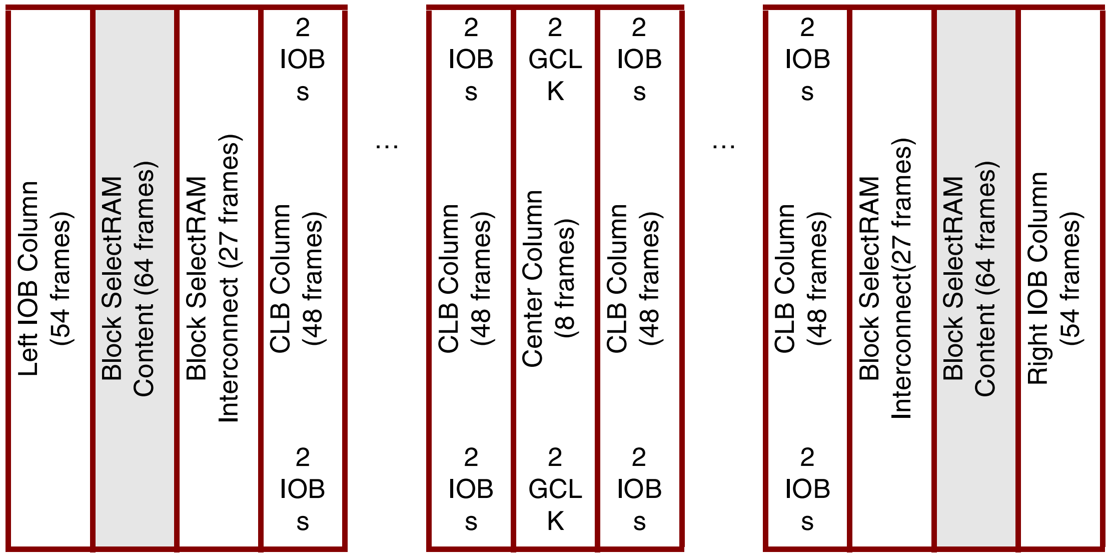
Left IOB Column (54 frames)
Block SelectRAM Content (64 frames)
Block SelectRAM Interconnect (27 frames)
2 IOBs
CLB Column (48 frames)
2 IOBs
...
2 IOBs
CLB Column (48 frames)
2 IOBs
2 GCLK
Center Column (8 frames)
2 GCLK
2 IOBs
CLB Column (48 frames)
2 IOBs
...
2 IOBs
CLB Column (48 frames)
2 IOBs
Block SelectRAM Interconnect(27 frames)
Block SelectRAM Content (64 frames)
Right IOB Column (54 frames)

*Figure 3:* **Configuration Column Example (XCV50)**

## Configuration Addressing

### Block Type, Major Address, Minor Address

The total address space is divided into block types. There are two block types: RAM and CLB. The RAM type contains only the block SelectRAM content columns (not interconnect). The CLB type contains all other column types.

Both the RAM and CLB address spaces are subdivided into major and minor addresses. Each configuration column has a unique major address within the RAM or CLB space. Each configuration frame has a unique minor address within its column.

The numbering schemes for the block type and minor address are identical between Virtex, Virtex-E, and Virtex-E extended memory devices. The major address numbering scheme differs between families and is encapsulated in Table 2. But in both families, even major addresses are on the left half of the device while the odd major addresses are on the right half of the device.

*Table 2:* **Major Addressing Scheme by Family**

| Column Type | Block Type | Virtex | Virtex-E | Virtex-E Extended Memory |
|---|---|---|---|---|
| First MJA | CLB | 0 | 0 | 0 |
| | RAM | 0 | 1 | 1 |
| MJA Order | CLB | 1: Center<br>2: CLB<br>3: IOB<br>4: BRAM Interconnect | 1: Center<br>2: CLB / BRAM Interconnect<br>3: IOB | 1: Center<br>2: CLB<br>3: IOB<br>4: BRAM Interconnect |
| | RAM | BRAM Content | BRAM Content | BRAM Content |

### Virtex Major Addresses

The CLB address space begins with '0' for the center column and alternates between the right and left halves of the device for all the CLB columns, then IOB columns, and finally block SelectRAM interconnect columns.

The RAM address space has '0' for the left block SelectRAM content column and '1' for the right column.

A XCV50 is shown in Figure 4. The shaded columns are in the RAM address space.

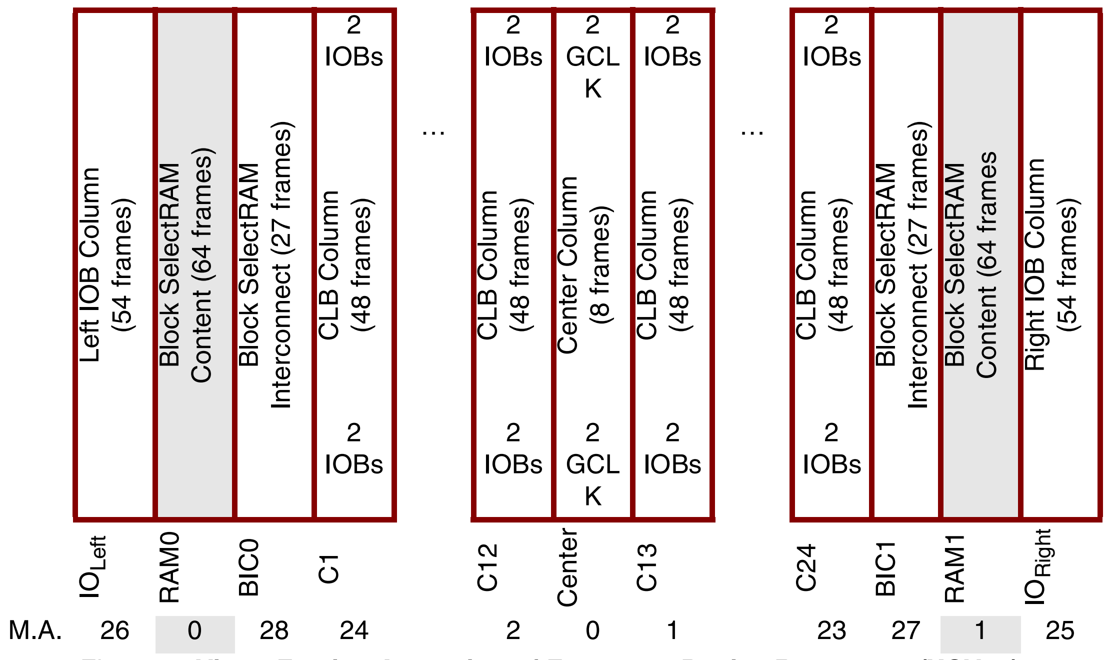

*Figure 4:* **Virtex Family: Allocation of Frames to Device Resources (XCV50)**

### Virtex-E Extended Memory Major Addresses

The Virtex-E extended memory family has the same numbering as the Virtex family but the block SelectRAM content columns begin with "1".

### Virtex-E Major Addresses

The Virtex-E family has block SelectRAM interspersed between CLB columns. Both the CLB and RAM major addressing have been changed to allow for this.

The CLB address space again begins with '0' for the center column and alternates between the right and left sides of the device for CLB columns *and* block SelectRAM interconnect columns and finally IOB columns. The difference is that in Virtex-E devices, the block SelectRAM interconnect appear in the middle of the CLB addresses while in the Virtex and Virtex-E extended memory families, they appear at the very end after the IOB columns. Also, note that the block SelectRAM content columns begin with ‘1’ (Virtex devices begin with ‘0’).

A XCV50E is shown in Figure 5. The shaded columns are in the RAM address space.

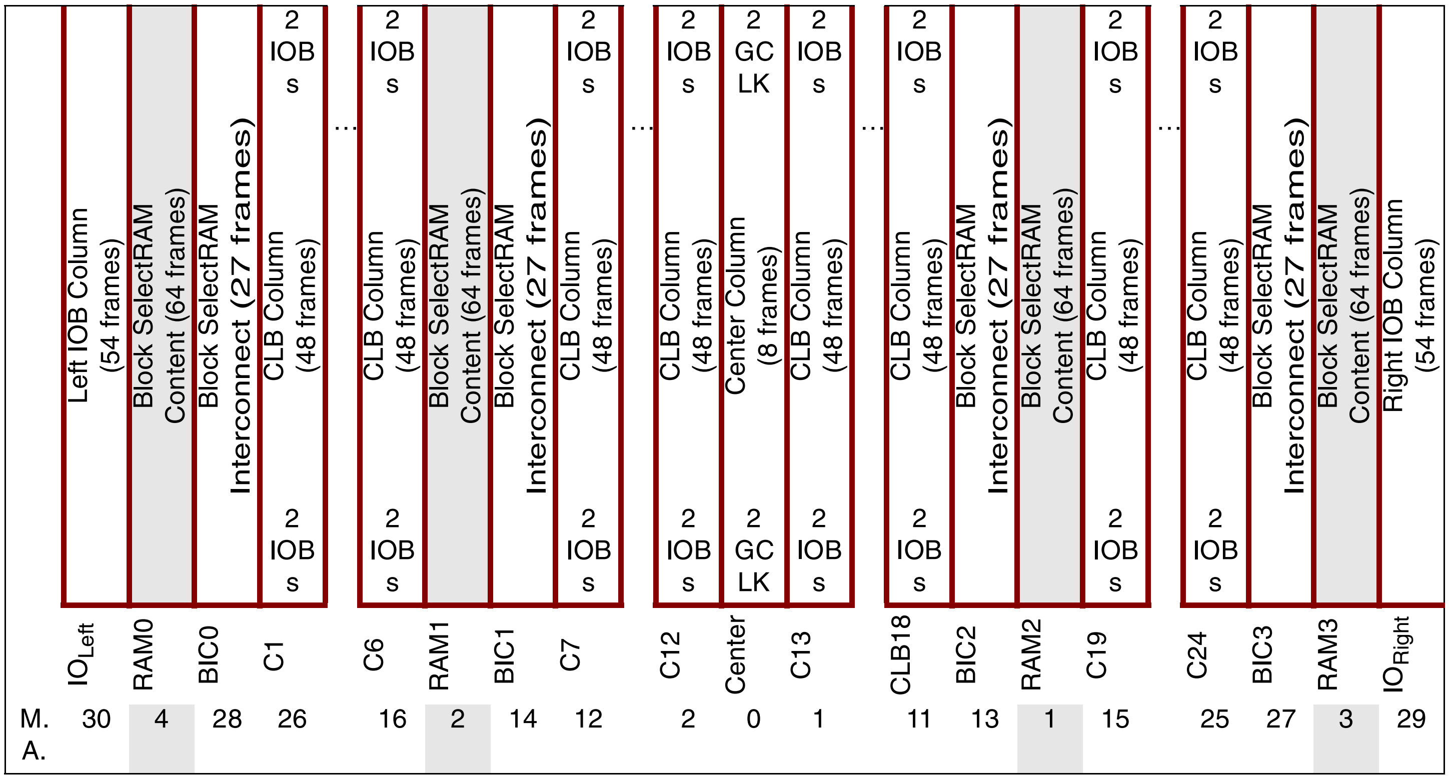

*Figure 5:* **Virtex-E Family: Allocation of Frames to Device Resources (XCV50E)**

## Frames

Frames are read and written sequentially with ascending addresses for each operation. Multiple consecutive frames can be read or written with a single configuration command. The smallest amount of data that can be read or written with a single command is a single frame. The entire CLB array plus the IOBs and block SelectRAM interconnect can be read or written with a single command. Each block SelectRAM Content must be read or written separately.

### Frame Sizes

Frame size depends on the number of rows in the device. The number of configuration bits in a frame is 18 × (# CLB_rows+2) and is padded with zeroes on the right (bottom) to fit in 32-bit words. See "Frame Organization" on page 6. An additional padding word is needed at the end of each frame for pipelining. Table 3 shows the frame sizes for all Virtex devices. This table also shows the size, in words, of the bitstream for the CLB address space and the number of words in each RAM block.

*Table 3:* **Virtex Series Devices**

| | | | | Virtex | | | Virtex-E | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | # of 32-bit Read-back Words [(1)] | | | | # of 32-bit Read-back Words [(1)] | |
| **Device** | **Row × Col.** | **Bits/ Frame** | **Words/ Frame** | **RAM Cols** | **CLB (all)** | **RAM (1 col)** | **RAM Cols** | **RAM Space** | **CLB (all)** | **RAM (1 col)** |
| XCV50 | 16 × 24 | 384 | 12 | 2 | 15,876 | 780 | - | - | - | - |
| XCV50E | 16 × 24 | 384 | 12 | - | - | - | 4 | 6 | 16,524 | 780 |
| XCV100 | 20 × 30 | 448 | 14 | 2 | 22,554 | 910 | - | - | - | - |
| XCV100E | 20 × 30 | 448 | 14 | - | - | - | 4 | 12 | 23,310 | 910 |
| XCV150 | 24 × 36 | 512 | 16 | 2 | 30,384 | 1,040 | - | - | - | - |
| XCV200 | 28 × 42 | 576 | 18 | 2 | 39,366 | 1,170 | - | - | - | - |
| XCV200E | 28 × 42 | 576 | 18 | - | - | - | 4 | 12 | 40,338 | 1,170 |
| XCV300 | 32 × 48 | 672 | 21 | 2 | 51,975 | 1,365 | - | - | - | - |
| XCV300E | 32 × 48 | 672 | 21 | - | - | - | 4 | 12 | 53,109 | 1,365 |
| XCV400 | 40 × 60 | 800 | 25 | 2 | 76,275 | 1,625 | - | - | - | - |
| XCV400E | 40 × 60 | 800 | 25 | - | - | - | 4 | 12 | 77,625 | 1,625 |
| XCV405E | 40 × 60 | 800 | 25 | - | - | - | 14 | 4 | 84,375 | 1,625 |
| XCV600 | 48 × 72 | 960 | 30 | 2 | 108,810 | 1,950 | - | - | - | - |
| XCV600E | 48 × 72 | 960 | 30 | - | - | - | 6 | 12 | 112,050 | 1,950 |
| XCV800 | 56 × 84 | 1088 | 34 | 2 | 142,902 | 2,210 | - | - | - | - |
| XCV812E | 56 × 84 | 1088 | 34 | - | - | - | 20 | 4 | 159,426 | 2,210 |
| XCV1000 | 64 × 96 | 1248 | 39 | 2 | 186,381 | 2,535 | - | - | - | - |
| XCV1000E | 64 × 96 | 1248 | 39 | - | - | - | 6 | 12 | 190,593 | 2,535 |
| XCV1600E | 72 × 108 | 1376 | 43 | - | - | - | 8 | 12 | 237,231 | 2,795 |
| XCV2000E | 80 × 120 | 1536 | 48 | - | - | - | 8 | 12 | 292,464 | 3,120 |
| XCV2600E | 92 × 138 | 1728 | 54 | - | - | - | 8 | 12 | 375,678 | 3,510 |
| XCV3200E | 104 × 156 | 1952 | 61 | - | - | - | 8 | 12 | 477,081 | 3,965 |

**Notes:**
1. Includes pad frame in calculation.

## Frame Organization

Each frame sits vertically in the device, with the “front” of the frame at the top. As shown in Figure 6, it is convenient to consider the frame horizontally when it is viewed as part of a bitstream. The top IOBs are shown on the left followed by the CLBs in the column and the bottom IOBs on the right. Bits in frames are allocated as follows.

For CLB columns, the first 18 bits control the two IOBs at the top of the column; then 18 bits are allocated for each CLB row; finally, the next 18 bits control the two IOBs at the bottom of the CLB column. The frame then contains enough “pad” bits to make it an integral multiple of 32 bits.

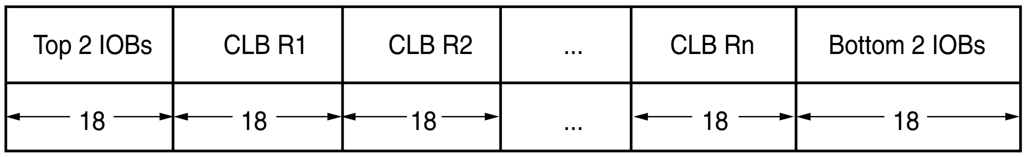

*Figure 6:* **CLB Column Frame Organization**

For IOB columns, the frame allocates 18 bits per three IOB rows, Figure 7. When reading and writing frames, bits are grouped into 32-bit words, starting on the left (corresponding to the top of the device). If the last word does not completely fill a 32-bit word, it is padded on the right with zeroes.

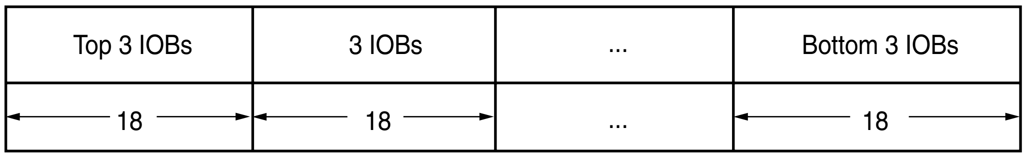

*Figure 7:* **IOB Column Frame Organization**

For block SelectRAM content columns (Figure 8), the first 18 bits are pad bits; then 72 bits are allocated for each RAM row; finally, there are another 18 pad bits. The frame then adds enough "pad" bits to make it an integral multiple of 32 bits.

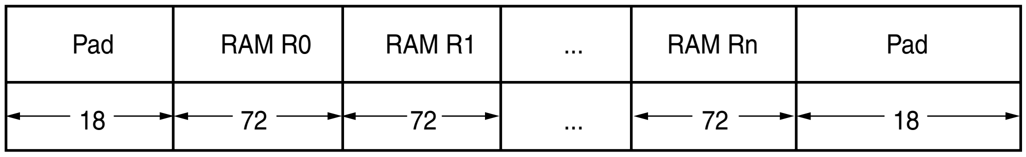

*Figure 8:* **Block SelectRAM Content Column Frame Organization**

## Location of LUT SelectRAM Bits

With respect to the beginning of a configuration frame, relative locations of LUT SelectRAM bits within the bitstream are the same for every CLB slice. The bits of a LUT SelectRAM are spread across 16 consecutive frame Minor Addresses. Each frame Minor Address contains all instances of a single bit index for that column. These 16 frames contain all 16 bits of the LUT SelectRAM for a column of CLB slices. You must read or write the 16 frames containing those bits to read or write the entire LUT SelectRAM.

Note: LUT SelectRAM bits are stored inverted. Flip-Flop data are stored in their true sense. When reading or writing LUT SelectRAM data from the bitstream, it is negated from the logic sense of the data. For example, a 4-input AND gate has the truth table LUT[15:0] = 1000000000000000.

This truth table is stored internally in the LUT SelectRAM as $\overline{\text{LUT}}[15:0]$ = 0111111111111111. Of course, user logic reading the data from the LUT SelectRAM would read the correct logic value, LUT[15:0] = 1000000000000000.

## Configuration Data Operations

### Reading Configuration Data

Configuration information can be read from the Virtex devices if readback is enabled in the current configuration. (See the description of the Control Register field SBITS on page 21.) CLB and IOB configuration data can be read while the device is operating.

When reading data from a Virtex device, a pad frame is read before any of the data frames. This is in addition to the pad word in each data frame. The pad frame must be included in the word count to be read from the device.

Each frame read from the device consists of the number of words shown in Table 3.

For example, the XCV150 has 16-word frames. When reading frames, the first word of a frame is a pad word. Including the pad frame, the XCV150 has 1,899 (30,384/16) frames. (See Figure 9.)

| **Pad Frame (16 words)** | |
|---|---|
| Pad Word | Data Frame 0 (15 words) |
| ⋮ | ⋮ |
| Pad Word | Data Frame n (15 words) |

*Figure 9:* **XCV150 Frame Padding for Device Read**

### Writing Configuration Data

Configuration information can be written to a Virtex device while the device is running if writing is enabled in the current configuration. (See the Control Register field SBITS in Table 23.) Re-writing the same configuration data does not generate any transient signals.

Changing configuration data may generate transient signals, especially if LUT values or signal routing is changed. For this case, all the logic cells and routing can be placed in a non-contentious state by asserting the GHIGH_B signal. See the description for the **Command Register (CMD)**, page 18,

When writing configuration data to the Virtex device, whether from the SelectMAP or JTAG interfaces, the data frames are written to the device with each frame followed by a pad word. After all the data frames are written, a pad frame must be written (to flush internal pipelines). The pad words and pad frame must be included in the number of words to be written to the device. (Table 3.)

For example, the XCV1000 has 39-word frames. When writing frames, the last word of the frame is a pad word. Including the pad frame, the XCV1000 has 4,779 (186,381/39) frames. See Figure 10.)

| | |
|---|---|
| Data Frame 0 (38 words) | Pad Word |
| ⋮ | |
| Data Frame n (38 words) | Pad Word |
| Pad Frame (39 words) | |

*Figure 10:* **XCV1000 Frame Padding for Device Write**

### Altering Configuration Data

Virtex devices support the Read-Modify-Write (RMW) method of changing LUT SelectRAM data. Therefore, it is possible to read or alter the contents of one or more LUT SelectRAMs through the JTAG or SelectMAP interfaces. *When writing data to one or more LUT SelectRAMs, all the bits in the frame* ***must*** *have valid configuration information.* This can be assured by altering valid configurations from bitstream files or from frames read from a properly configured Virtex device. When using the RMW method, it may be expeditious to read the data, ignoring the pad frame and the pad word of only the first frame. This aligns the remaining configuration data in the Device Write format. After modifying the configuration, the altered data can be written to the Virtex device followed by a pad word and a pad frame.

It is not necessary to retain the contents of the pad frame or pad words. However, pad words and pad frames **are** included in the CRC calculation.

Note: Frames span an entire column of CLB slices (or IOBs). Thus, when changing LUT SelectRAM bit 0 for a single CLB slice (*e.g.*, R3C4.S1), LUT SelectRAM bit 0 for *all* slices in that column (*i.e.*, R*C4.S1) is written with the same command. One must ensure that either all other data in the slice is constant or changed externally through partial configurations.

LUT SelectRAMs not configured externally should not lie in the same slice with LUTs or LUT SelectRAMs that are re-configured externally. Such a mixture can cause the unrelated LUT SelectRAM data to be re-configured when the frames are written to the device. Figure 11 shows what can happen if this restriction is not observed.

In this example, it is the objective to perform a Read-Modify-Write on the LUT SelectRAM in R2C3.S1 in column 3, which is shown in the first column in the figure. Row 2 contains a LUT containing the value `AB`. Row 3 contains a SelectRAM containing the value `C3`. Because the Read and Write operations operate on an entire frame, the RAM in R3C3.S1 is also read and written when R2C3.S1 is read and written. Performing the Readback operation reads all the LUT and SelectRAM values for the column. Before the new value for the LUT is written, it is possible for the on-chip circuitry to write a new value, such as `14`, into the SelectRAM. When the new value, `BD`, is written via the configuration interface into LUT R2C3.S1, the value `C3` is also written into RAM R3C3.S1. This may not always be desirable (It is design-dependent).

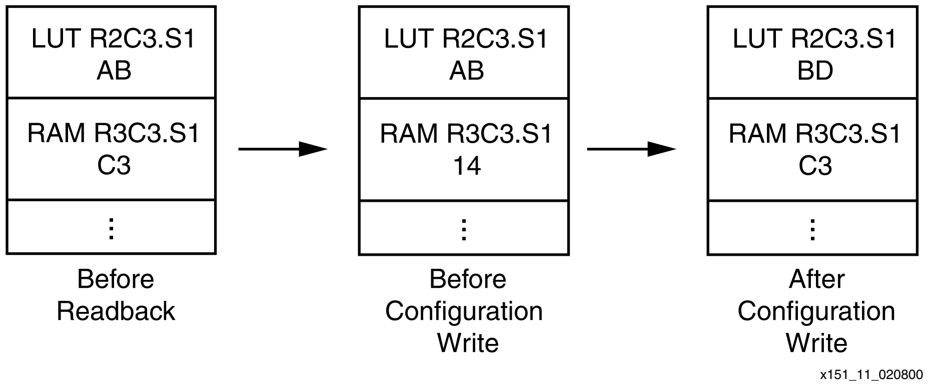

*Figure 11:* **Potential Write Conflicts**

## Definitions

Two sets of variables are defined to determine where a desired bit is in the configuration data. The first is a set of “independent” variables, or design attributes, namely characteristics of the design that are known, i.e., the device size and which CLB and bit(s) within the CLB to locate, Table 4. The second set is a set of “dependent” variables or design variables, namely values that must be calculated to find the bit(s) of interest listed in Table 5.

*Table 4:* **Design Attributes (Independent Variables)**

| Term | Definition |
|---|---|
| Chip_Cols | # of CLB columns on the Virtex device |
| Chip_Rows | # of CLB rows on the Virtex device |
| Chip_Rams | # of block SelectRAM columns on the Virtex device |
| RAM_Space | Spacing of block SelectRAM columns (in terms of CLB columns) |
| FL | # of 32-bit words in the frame. |
| RW | 1 for Read, 0 for Write |
| CLB_Col | Column number of the desired CLB |
| CLB_Row | Row number of the desired CLB |

*Table 4:* **Design Attributes (Independent Variables)** ***(Continued)***

| Term | Definition |
|---|---|
| Slice | 0 or 1 |
| FG | 0 for the F-LUT, 1 for the G-LUT |
| lut_bit | The desired bit from the given LUT. Bits in the LUT are indexed from 0 to 15. |
| XY | 0 for the X Flip-Flop, 1 for the Y Flip-Flop |
| RAM_Col | Column number of the desired block SelectRAM |
| RAM_Row | Row number of the desired block SelectRAM |
| ram_bit | The desired bit from the given block SelectRAM. Bits are indexed from 0 to 4095. |

*Table 5:* **Design Variables (Dependent Variables)**

| Term | Definition |
|---|---|
| MJA | Frame Major Address |
| MNA | Frame Minor Address |
| fm_st_wd | The index of the word within a full configuration segment that corresponds to the starting word of the desired frame. A full configuration segment is defined as the following: 1) for CLB/IOB, all CLB, IOB, and RAM interconnect frames beginning at MJA=0, MNA=0 and 2) for block SelectRAM, all RAM content frames for the given RAM column. Words are numbered starting at 0. |
| fm_wd | The index of the 32-bit word within a frame that contains the desired bit. Words in a frame are numbered starting at 0. |
| fm_wd_bit_idx | The bit index of the desired bit within frame word fm_wd. Words are indexed in “big-endian” style, with bit 31 on the left and bit 0 on the right. |
| fm_bit_idx | Bit index within a frame of the desired bit. Numbered starting with 0 as the left-most (first) bit. Bit numbering within a frame continues across all the words in the frame. |

*Table 6:* **Functions used in Equations**

| Functions | Definition |
|---|---|
| floor(x) | The largest integer not larger than *x*. *E.g.*, floor(3.1) = 3, because the next largest integer, 4, is larger than 3.1, floor(3)=3. |
| ceiling(x) | The smallest integer greater than or equal to *x*. For example, ceiling(3.2) = 4, because 4 is greater than 3.2, and 3 is not, ceiling (4) = 4. |
| % | The modulus operation. 5 % 2 = 1, 5 % 3 = 2, 5 % 5 = 0. |

## CLB LUT SelectRAM Dependent Variables

Table 7 shows equations for the LUT SelectRAM "Dependent Variables" that were defined in Table 5.

*Table 7:* **Virtex Equations for LUT SelectRAM Dependent Variables**

| Term | Definition |
|---|---|
| MJA | if $(CLB\_Col \le Chip\_Cols/2)$,<br>then $Chip\_Cols - CLB\_Col \times 2 + 2$<br>else $2 \times CLB\_Col - Chip\_Cols - 1$ |
| MNA | $lut\_bit + 32 - Slice \times (2 \times lut\_bit + 17)$ |
| fm_bit_idx | $3 + 18 \times CLB\_Row - FG + RW \times 32$ |
| fm_st_wd | $FL \times (8 + (MJA - 1) \times 48 + MNA) + RW \times FL$ |
| fm_wd | floor(fm_bit_idx/32) |
| fm_wd_bit_idx | $31 + 32 \times fm\_wd - fm\_bit\_idx$ |

### Virtex-E CLB LUT SelectRAM Dependent Variables

The additional block SelectRAM columns in Virtex-E devices require an adjustment to the MJA. The following equations calculate the adjustment necessary for each of the affected dependent variables. For each variable, calculate the base value based upon the Virtex equations only and then add the resulting Virtex adjustment.

**Notes:**

- Do NOT input the final results back into the original Virtex equations (e.g. do not use $MJA_{final}$ value to calculate fm_st_wd).
- $MJA_{final} = MJA + MJA\_adj$
- $fm\_st\_wd_{final} = fm\_st\_wd + fm\_st\_wd\_adj$

The left and right sides of the device are treated differently:

*Table 8:* **CLB Location**

| CLB Location | CLB Column |
|---|---|
| Left | $CLB\_Col \le Chip\_Cols/2$ |
| Right | $CLB\_Col > Chip\_Cols/2$ |

*Table 9:* **Virtex-E Families Adjustments for LUT SelectRAM Dependent Variables**

| Term | | Definition |
|---|---|---|
| MJA_adj | Left | RAM_Bound = (Chip_Rams/2 - 1) x RAM_Space<br>MJA_adj = 2 x ceiling((RAM_Bound - CLB_Col + 1) / RAM_Space) |
| | Right | RAM_Bound = Chip_Cols - (Chip_Rams/2 - 1) x RAM_Space + 1<br>MJA_adj = 2 x ceiling((CLB_Col - RAM_Bound + 1) / RAM_Space) |
| fm_st_wd_adj | | $FL \times (27 \times MJA\_adj)$ |

### LUT SelectRAM Examples

See "Examples" on page 28 for several examples of reading and evaluating configuration data. The examples illustrate how to make use of these equations to find the desired data in a bitstream.

## CLB Flip-Flop Dependent Variables

Equations for the CLB flip-flops can be found in Table 10. Their locations are calculated similarly to the LUT SelectRAM locations. Equations for the CLB FF Dependent Variables are defined in Table 5.

*Table 10:* **Virtex Equations for CLB FF Dependent Variables**

| Term | Definition |
|---|---|
| MJA | if (CLB_Col ≤ Chip_Cols/ 2)<br>then Chip_Cols – CLB_Col × 2 + 2<br>else 2 × CLB_Col – Chip_Cols – 1 |
| MNA | Slice × (12 × XY – 43) – 6 × XY + 45 |
| fm_bit_idx | (18 × CLB_Row) + 1 + (32 × RW) |
| fm_st_wd | FL × (8 + (MJA –1) × 48 + MNA) + RW × FL |
| fm_wd | floor(fm_bit_idx/32) |
| fm_wd_bit_idx | 31 + 32 × fm_wd – fm_bit_idx |

### Virtex-E CLB Flip-Flop Dependent Variables

Apply the adjustments given in Table 8 and Table 9.

## IOB Dependent Variables

Each IOB contains four values that can be captured into special registers. These values are:

- I — the input flip-flop
- O — the output flip-flop
- T — the flip-flop for the tri-state control
- P — the value of the I/O pad

These values are captured by utilizing the CAPTURE_VIRTEX symbol in your design. The Libraries Guide has more details on the use of this symbol. The following registers will be read as part of the readback data.

Access to IOB flip-flops is different for the top and bottom IOBs versus the left and right IOBs.

The top and bottom IOBs are part of the CLB column frames. There are two IOBs at the top and bottom of each CLB column.

The left and right IOBs are in columns by themselves. There are three IOBs per CLB row.

IOBs are numbered clockwise around the die. Pad 1 is located at the left side of the top edge, above CLB column 1. The equations for where to find IOB flip-flops in the bitstream are based on the ***pad number*** *which is the same for a given size device****,*** *not the package pin name, which varies from package to package*. The mapping from package pin names to pad numbers can be found in *EPIC* or *fpga_editor*.

Table 11 contains the numeric pad indices for the pads on all four edges of the device in terms of the number of CLB columns and rows on the device.

*Table 11:* **IOB Pad Indices**

| Pad Location | Pad Index i |
|---|---|
| Top | 1 ≤ i ≤ Chip_Cols × 2 |
| Right | Chip_Cols × 2 +1 ≤ i ≤ Chip_Cols × 2 + Chip_Rows × 3 |
| Bottom | Chip_Cols × 2 + Chip_Rows × 3 + 1 ≤ i ≤ Chip_Cols × 4 + Chip_Rows × 3 |
| Left | Chip_Cols × 4 + Chip_Rows × 3 +1 ≤ i ≤ Chip_Cols × 4 + Chip_Rows × 6 |

Table 12 shows the equations for the dependent variables for the IOB flip-flops. The variable ***i*** in this table refers to the index of pad ***i***.

*Table 12:* **Equations for IOB Dependent Variables**

| Term | | | Definition |
|---|---|---|---|
| MJA | Top | | if ($i \leq$ Chip_Cols)<br>then Chip_Cols – ceiling(i/2) $\times$ 2 + 2<br>else 2 $\times$ ceiling (i/2) – Chip_Cols –1 |
| | Right | | Chip_Cols + 1 |
| | Bottom | | if (i > 3 $\times$ (Chip_Cols + Chip_Rows))<br>then 2 $\times$ ceiling((i – 3 $\times$ Chip_Cols – 3 $\times$ Chip_Rows)/2)<br>else Chip_Cols – 2 $\times$ floor((i – 2 $\times$ Chip_Cols – 3 $\times$ Chip_Rows – 1)/2) – 1 |
| | Left | | Chip_Cols + 2 |
| MNA | Top | I | – 25 $\times$ (i%2) + 45 |
| | | O | – 13 $\times$ (i%2) + 39 |
| | | T | – 5 $\times$ (i%2) + 35 |
| | | P | – 4 $\times$ (i%2) + 25 |
| | Right | I | t = (i – 2 $\times$ Chip_Cols) % 3; MNA = 27.5 $\times$ $t^2$ – 57.5 $\times$ t + 32 |
| | | O | t = (i – 2 $\times$ Chip_Cols) % 3; MNA = 21.5 $\times$ $t^2$ – 51.5 $\times$ t + 38 |
| | | T | t = (i – 2 $\times$ Chip_Cols) % 3; MNA = 17.5 $\times$ $t^2$ – 47.5 $\times$ t + 42 |
| | | P | 50 |
| | Bottom | I | 25 $\times$ ((i – 2 x Chip_Cols - 3 x Chip_Rows)%2) + 20 |
| | | O | 13 $\times$ ((i – 2 x Chip_Cols - 3 x Chip_Rows)%2) + 26 |
| | | T | 5 $\times$ ((i – 2 x Chip_Cols - 3 x Chip_Rows)%2) + 30 |
| | | P | 4 $\times$ ((i – 2 x Chip_Cols - 3 x Chip_Rows)%2) + 21 |
| | Left | I | t = (i – 4 x Chip_Cols - 3 x Chip_Rows)%3; MNA = 17.5 $\times$ $t^2$ – 47.5 $\times$ t + 45 |
| | | O | t = (i – 4 x Chip_Cols - 3 x Chip_Rows)%3; MNA = 23.5 $\times$ $t^2$ – 53.5 $\times$ t + 39 |
| | | T | t = (i – 4 x Chip_Cols - 3 x Chip_Rows)%3; MNA = 27.5 $\times$ $t^2$ – 57.5 $\times$ t + 35 |
| | | P | 50 |
| fm_bit_idx | Top | | 32 $\times$ RW |
| | Right | I, O, &T | 18 $\times$ (1 + floor ((i – 2 $\times$ Chip_Cols – 1)/3)) + 32 $\times$ RW |
| | | P | t = (i – 2 $\times$ Chip_Cols) % 3;<br>fm_bit_idx = 18 $\times$ (1 + floor ((i – 2 $\times$ Chip_Cols – 1)/3)) + 6 $\times$ $t^2$ – 17 $\times$ t + 15 + 32 $\times$ RW |
| | Bottom | | 18 $\times$ (Chip_Rows + 1)+ 32 $\times$ RW |
| | Left | I, O, &T | 18 $\times$ (Chip_Rows – floor(<br>(i – 4 $\times$ Chip_Cols – 3 $\times$ Chip_Rows – 1)/3)) + 32 $\times$ RW |
| | | P | t = (i – 4 $\times$ Chip_Cols – 3 $\times$ Chip_Rows)%3;<br>fm_bit_idx = 18 $\times$ (Chip_Rows – floor ((i – 4 $\times$ Chip_Cols – 3 $\times$ Chip_Rows – 1)/3)) – 10.5 $\times$ $t^2$ + 21.5 $\times$ t + 4 + 32 $\times$ RW |
| fm_st_wd | | | if (MJA > Chip_Cols + 1)<br>then FL $\times$ (54 $\times$ MJA – 46 + MNA – 6 $\times$ Chip_Cols) + RW $\times$ FL<br>else FL $\times$ (8 + (MJA –1) $\times$ 48 + MNA) + RW $\times$ FL |
| fm_wd | | | floor (fm_bit_idx/32) |
| fm_wd_bit_idx | | | 31 + 32 $\times$ fm_wd – fm_bit_idx |

### Virtex-E IOB Dependent Variables

Similar to CLB dependent variables, only the variables affected by the major address are different in Virtex-E devices versus Virtex devices. A simple way to account for the change is to determine the CLB column where the pad resides (only true for top and bottom). Then, the adjustment calculated for the CLB column can be applied. The conversion to CLB columns is found in Table 13:

*Table 13:* **Pad -> CLB Column**

| Term | | Definition |
|---|---|---|
| CLB_Col | Top | ceiling(i/2) |
| | Bottom | ceiling((4 x Chip_Cols + 3 x Chip_Rows + 1 - i)/2) |

*Table 14:* **Virtex-E Families Adjustments for IOB Dependent Variables**

| Term | | Definition |
|---|---|---|
| MJA_adj | Top | use Table 8 and Table 9 |
| | Right | Chip_Rams |
| | Bottom | use Table 8 and Table 9 |
| | Left | Chip_Rams |
| fm_st_wd_adj | | $FL \times (27 \times MJA\_adj)$ |

## Block SelectRAM Dependent Variables

The equations for the block SelectRAM dependent variables are given in Table 15. The relationship of a particular memory cell index in the context of a given configuration is described in XAPP130 "Using the Virtex Block SelectRAM+ Resource".

*Table 15:* **Virtex Equations for Block SelectRAM Dependent Variables**

| Term | Definition |
|---|---|
| MJA | if (RAM_Col < Chip_Rams/2)<br>then 2 x (Chip_Rams/2 - 1 - RAM_Col)<br>else 2 x (RAM_Col - Chip_Rams/2) + 1 |
| MNA | 1 x floor(((ram_bit / 64) % 64)/32) + 2 x floor(((ram_bit / 64) % 32)/16)<br>+ 4 x floor(((ram_bit / 64) % 16)/8) + 8 x floor(((ram_bit / 64) % 8)/4)<br>+ 16 x floor(((ram_bit / 64) % 4)/2) + 32 x floor(((ram_bit / 64) % 2)/1)<br>equivalent to MNA = div64[0:5] where div64[5:0] = floor(ram_bit/64) |
| fm_bit_idx | obtain value for bitpos from Table 16<br>fm_bit_idx = 18 + 72 x RAM_Row + bitpos |
| fm_st_wd | $FL \times MNA + RW \times FL$ |
| fm_wd | floor(fm_bit_idx/32) |
| fm_wd_bit_idx | $31 + 32 \times fm\_wd - fm\_bit\_idx$ |

*Table 16:* **Virtex Block SelectRAM Bit Position Within a Given Block SelectRAM**

| ram_bit % 64 | bitpos | ram_bit % 64 | bitpos | ram_bit % 64 | bitpos | ram_bit % 64 | bitpos | ram_bit % 64 | bitpos | ram_bit % 64 | bitpos | ram_bit % 64 | bitpos | ram_bit % 64 | bitpos |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 42 | 8 | 45 | 16 | 29 | 24 | 26 | 32 | 43 | 40 | 44 | 48 | 28 | 56 | 27 |
| 1 | 58 | 9 | 61 | 17 | 13 | 25 | 10 | 33 | 59 | 41 | 60 | 49 | 12 | 57 | 11 |
| 2 | 41 | 10 | 46 | 18 | 30 | 26 | 25 | 34 | 40 | 42 | 47 | 50 | 31 | 58 | 24 |
| 3 | 57 | 11 | 62 | 19 | 14 | 27 | 9 | 35 | 56 | 43 | 63 | 51 | 15 | 59 | 8 |
| 4 | 50 | 12 | 53 | 20 | 21 | 28 | 18 | 36 | 51 | 44 | 52 | 52 | 20 | 60 | 19 |
| 5 | 49 | 13 | 54 | 21 | 22 | 29 | 17 | 37 | 48 | 45 | 55 | 53 | 23 | 61 | 16 |
| 6 | 66 | 14 | 69 | 22 | 5 | 30 | 2 | 38 | 67 | 46 | 68 | 54 | 4 | 62 | 3 |
| 7 | 65 | 15 | 70 | 23 | 6 | 31 | 1 | 39 | 64 | 47 | 71 | 55 | 7 | 63 | 0 |

### Virtex-E Block SelectRAM Dependent Variables

The RAM major address numbering scheme changed slightly such that the lowest MJA on the left side is now "2" instead of "0". As in the case for the CLB equations, the adjustments below should be applied after the Virtex equations.

*Table 17:* **Virtex-E Adjustments for Block SelectRAM Dependent Variables**

| Term | Definition |
|---|---|
| MJA_adj | if (RAM_Col < Chip_Rams/2) MJA_adj = 2 |

## Configuration Logic Basics

### Configuration Data

Configuration data is organized as 32-bit words. There are two major commands that the configuration data can contain; Read and Write. A configuration command is executed when the configuration command is read or written to the appropriate command register.

The OP field contains 01 for a Read operation, and 10 for a Write operation. (See Figure 12).

| OP | CODE |
|---|---|
| Read | 01 |
| Write | 10 |

*Figure 12:* **OP Field Code**

A command is organized as a packet with a header word and optional data words. The header word is the first word written to the appropriate command register for a read or write operation. The header word contains a type field (001), an operand field, a register address field, and a word count field. The format for the command header is shown in Figure 13, page 17.

The Register Address field defines the target of this command, as defined in Table 18, page 17.

The header word count field contains an integer between 0 and 2,047 and indicates the number of words that follow the header. Larger word counts (between 2,048 and 1,048,575 words) are achieved by setting the header word count to 0. The Extension header word has a type field = 010, an OP field that must match the OP field in the preceding Command Header word, and a 20-bit word count, this format is shown Figure 14, page 17.

## Configuration Flow

Virtex devices are configured by presenting configuration data to the SelectMAP or JTAG interfaces in a specific sequence.

### For Initial Configuration

1. Issue one or more pad words (SelectMAP only).
2. Issue Sync word (SelectMAP only).
3. Reset CRC.
4. Set FLR.
5. Set COR.
6. Set MASK.
7. Set CTL. (Set PERSIST if you want to keep SelectMAP active after this configuration.)
8. Issue a SWITCH (switch CCLK frequency) command to the CMD register. This is necessary only in Master Serial configuration mode.

### Reading Configuration

These commands can be issued to read a full configuration or for a partial configuration after the device has been completely configured.

1. Issue a Sync word (SelectMAP only) if the previous configuration command was aborted.
2. Set the FAR to the starting address.
3. Issue a RCFG (read configuration) command to the CMD register.
4. Write the number of words to be read to the FDRO register.
5. Flush the command pipeline with a pad word.
6. Read data frames.

### Writing Configuration

These commands can be issued either as part of an initial configuration or for a partial configuration after the device has been configured.

1. Issue a Sync word (SelectMAP only) if the previous configuration command was aborted.
2. If the frames being written can cause contention, then assert the GHIGH_B signal.
3. Set the FAR to the starting address.
4. Issue a WCFG (write configuration) command to the CMD register.
5. Write the number of words to be written to the FDRI register.
6. Write data frames.
7. If the GHIGH_B signal was asserted, de-assert it by writing the LFRM (Last Frame) command to the CMD register and write one pad frame.

| Type | | | OP | | Register Address | | | | | | | | | | | | | | RSV | | Word Count | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 1 | x | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | x | x | x | 0 | 0 | x | x | x | x | x | x | x | x | x | x | x |

**Notes:**
1. Locations within fields containing a zero or one must have these values. An X in a bit field indicates that the value is variable and must be set.
2. Heavy vertical lines are used to separate fields. Light vertical lines separate nibbles in the word.

*Figure 13:* **Command Header Format**

| Type | | | OP | | Word Count | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 1 | 0 | x | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

**Notes:**
1. Locations within fields containing a zero or one must have these values. An X in a bit field indicates that the value is variable and must be set.
2. Heavy vertical lines are used to separate fields. Light vertical lines separate nibbles in the word.

*Figure 14:* **Large Block Count Header Extension Format**

## Configuration Registers

Configuration logic is accessed and controlled via a collection of 32-bit registers called the configuration registers (see Table 18). Registers are described in the following sections.

*Table 18:* **Configuration Register Addresses**

| Register Name | Mnemonic | R/W | Binary Address |
|---|---|---|---|
| CRC | CRC | R/W | 0000 |
| Frame Address | FAR | R/W | 0001 |
| Frame Data Input | FDRI | W | 0010 |
| Frame Data Output | FDRO | R | 0011 |
| Command | CMD | R/W | 0100 |
| Control | CTL | R/W | 0101 |
| Control Mask | MASK | R/W | 0110 |
| Status | STAT | R | 0111 |
| Legacy Output | LOUT | W | 1000 |
| Configuration Option | COR | R/W | 1001 |
| Reserved | — | — | 1010 |
| Frame Length | FLR | R/W | 1011 |
| Reserved | — | — | 1100 |
| Reserved | — | — | 1101 |
| Reserved | — | — | 1110 |
| Reserved | — | — | 1111 |

## Command Register (CMD)

The content of the Command Register (CMD) is interpreted by the configuration state machine. Configuration commands control the operation of the configuration state machine, the Frame Data Register (FDR), and some of the global signals. The command in the Command Register is executed each time the FAR is loaded with a new value. The effect of each command is defined in Table 19.

*Table 19:* **Configuration Commands**

| Cmd | Code | Description |
|---|---|---|
| Rsvd | 0000 | Reserved |
| WCFG | 0001 | **Write Configuration Data —** Used prior to writing configuration data to the FDRI. It takes the internal configuration state machine through a sequence of states that control the shifting of the FDR and the writing of the configuration memory. (See **Frame Data Input Register (FDRI)**, page 23). |
| Rsvd | 0010 | Reserved |
| LFRM | 0011 | **Last Frame —** This command is loaded prior to writing the last (pad) data frame if the GHIGH_B signal was asserted. This command is not necessary if the GHIGH_B signal was not asserted. This allows overlap of the last frame write with the release of the GHIGH_B signal. |
| RCFG | 0100 | **Read Configuration Data —** Used prior to reading frame data from the FDRO. Similar to the WCFG command in its effect on the FDR (see **Frame Data Output Register (FDRO)**, page 24). |
| START | 0101 | **Begin Startup Sequence —** Starts the startup sequence. This command is also used to start a shutdown sequence prior to partial re-configuration. The Startup Sequence begins with the next successful CRC check (see **Cyclic Redundancy Check (CRC)**, page 21). |
| RCAP | 0110 | **Reset Capture —** Used when performing capture in single-shot mode. This command must be used to reset the capture signal if single-shot capture has been selected. |
| RCRC | 0111 | **Reset CRC —** Used to reset CRC register (see **Cyclic Redundancy Check (CRC)**, page 21). |
| AGHIGH | 1000 | **Assert GHIGH_B Signal —** Used prior to re-configuration to prevent contention while writing new configuration data. All CLB outputs and signals are forced to a one. |
| SWITCH | 1001 | **Switch CCLK Frequency —** Used to change (increase) the frequency of the Master CCLK. The new frequency is specified in Table 21. |
| Rsvd | 1010 | Reserved |
| Rsvd | 1011 | Reserved |
| Rsvd | 1100 | Reserved |
| Rsvd | 1101 | Reserved |
| Rsvd | 1110 | Reserved |
| Rsvd | 1111 | Reserved |

## Configuration Option Register (COR)

The Configuration Option Register (COR) is used to select configuration options that are illustrated in Figure 15 and defined in Table 20. Entries in this table are further explained in the following tables:.

| | DONE_PIPE | DRIVE_DONE | SINGLE | | | | | OSCFSEL | | SSCLKSRC | | | | LOCK_WAIT | SHUTDOWN | | | DONE_CYCLE | | | LCK_CYCLE | | | GTS_CYCLE | | | GWE_CYCLE | | | GSR_CYCLE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

**Notes:**
1. Locations within fields containing a zero or one must have these values. An *X* in a bit field indicates that the value is variable and must be set.
2. Heavy vertical lines are used to separate fields. Light vertical lines separate nibbles in the word.

*Figure 15:* **COR (Configuration of Option Register) Fields**

*Table 20:* **Configuration Option Register Fields**

| Field | Bit Indices | Description | Bitgen Default |
|---|---|---|---|
| DONE_PIPE | 30 | 0: No pipeline stage for DONEIN.<br>1: Add pipeline stage to DONEIN.<br>FPGA waits on DONE that is delayed by one (1) cycle of the StartupClk rather than the pin itself. Use this option when StartupClk is running at high speeds. | 0 |
| DRIVE_DONE | 29 | 0: DONE pin is open drain.<br>1: DONE pin is actively driven high. | 0 |
| SINGLE | 28 | Readback capture is one-shot. | 0 |
| OSCFSEL | 27:22 | Select CCLK frequency in Master Serial configuration mode. | 2 |
| SSCLKSRC | 21:20 | Startup sequence clock source<br>00: Cclk<br>01: UserClk<br>1x: JTAGClk | 0 |
| LOCK_WAIT | 19:16 | 4-bit mask indicating which DLL lock signals to wait for during LCK cycle. The 4 bits (from MSB to LSB) correspond to DLLs TL=3, TR=2, BL=1, BR=0 (top-left, top-right, etc.). In Virtex-E devices where there are 8 DLLs, each mask bit applies to the 2 DLLs in that quadrant of the device. The default is not to wait for any DLL lock signals. | 0 |
| SHUTDOWN | 15 | Indicate whether doing a startup or shutdown sequence.<br>0: Startup<br>1: Shutdown sequence | 0 |
| DONE_CYCLE | 14:12 | Startup phase in which DONE pin is released | 3 |
| LCK_CYCLE | 11:9 | Stall in this startup phase until DLL locks are asserted. | 7 |
| GTS_CYCLE | 8:6 | Startup phase in which I/Os switch from tri-state to user design | 4 |
| GWE_CYCLE | 5:3 | Startup phase in which the global write-enable is asserted | 5 |
| GSR_CYCLE | 2:0 | Startup phase in which the global set/reset is negated | 5 |

Table 21 shows the allowed values for the OSCFSEL field of the COR. Setting OSCFSEL to one of these values will set the Master CCLK frequency to the specified value.

*Table 21:* **OSCFSEL-Specified Master CCLK Frequencies)**

| CCLK (MHz) | OSCFSEL | CCLK (MHz) | OSCFSEL | CCLK (MHz) | OSCFSEL |
|---|---|---|---|---|---|
| 4.3 | 000010 | 13 | 001010 | 41 | 100111 |
| 5.4 | 010001 | 15 | 001101 | 45 | 110011 |
| 6.9 | 000100 | 20 | 010111 | 51 | 101010 |
| 8.1 | 000101 | 26 | 011010 | 55 | 110100 |
| 9.2 | 000110 | 30 | 011101 | 60 | 101101 |
| 10.0 | 000111 | 34 | 110010 | — | — |

**Notes:**
1. These values are accurate to +45%, – 30%.

Table 22 shows the values of the DONE_CYCLE, LCK_CYCLE, GTS_CYCLE, GWE_CYCLE, and GSR_CYCLE fields in COR. This table shows the step in the start-up sequence when each of these signals becomes active.

*Table 22:* **COR Startup Cycle Fields**

| Field Value | DONE_CYCLE (DONE Active) | GTS_CYCLE (GTS_CFG Inactive) | GSR_CYCLE (GSR Inactive) | GWE_CYCLE (GWE Active) | LCK_CYCLE |
|---|---|---|---|---|---|
| 000 | 1 | 1 | 1 | 1 | 0 |
| 001 | 2 | 2 | 2 | 2 | 1 |
| 010 | 3 | 3 | 3 | 3 | 2 |
| 011 | 4 (default) | 4 | 4 | 4 | 3 |
| 100 | 5 | 5 (default) | 5 | 5 | 4 |
| 101 | 6 | 6 | 6 (default) | 6 (default) | 5 |
| 110 | — | DoneIn† | DoneIn† | DoneIn† | 6 |
| 111 | Keep State | Keep State | Keep State | Keep State | Don’t Wait (default) |

**Notes:**
1. † DONE if DonePipe = No, else the delayed version of DONE.

## Control Register (CTL)

The Control Register (CTL) fields are illustrated in Figure 16, page 21 and defined in Table 23.

| | | | | | | | | | | | | | | | | | | | | | | | SBITS | | PERSIST | | | | | | GTS_USR_B |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | x | x | 0 | 0 | 0 | 0 | 0 | x |

**Notes:**
1. Locations within fields containing a zero or one must have these values. An *X* in a bit field indicates that the value is variable and must be set.
2. Heavy vertical lines are used to separate fields. Light vertical lines separate nibbles in the word.

*Figure 16:* **Control Register Fields**

*Table 23:* **Control Register Bits**

| Name | Bit Indices | Description |
|---|---|---|
| SBITS | 8:7 | Security level:<br>0: Read/Write OK (default)<br>1: Readback disabled.<br>2,3: Readback disabled, Writing disabled except CRC register. |
| PERSIST | 6 | Configuration interface remains after configuration.<br>0: No (default)<br>1: Yes |
| GTS_USR_B | 0 | Active-low global tristate I/Os. Turn off pull-ups if GTS_CFG_B is also asserted. |

## Cyclic Redundancy Check (CRC)

A data input error checking mechanism is provided through the Cyclic Redundancy Check (CRC) register. When data is written to any configuration register (except LOUT) a 16-bit CRC value is calculated using both the register data and the address. This value is saved in the CRC register. At the end of any series of writes a pre-calculated CRC block-check value may be written to the CRC register. If the resulting value is non-zero, an error is indicated. The CRC_ERROR bit is accessible through the status register. If a CRC error is detected, configuration logic is put in the ERROR mode. The following section is an algorithm for computing CRC.

### CRC Algorithm

```
/* Initialization */
bcc = 0;
skip_pad = true;
more_words = true;
/* Check for write operation. */
do {
  w = next_word;
  if (w[31:27] == ’00101’) {
    /* A Read OP. Don’t use in CRC*/
    wc = w[10:0];
    if (wc == 0) {
      w = next_word;
      wc = w[19:0];
    }
    while (wc--  > 0) {
      w = next_word;
    }
  }
  elsif (w[31:27] == ’00110’) {
    /* A Write OP. Use in CRC. */
    addr = w[16:13];
    if (addr Œ {0,1,2,4,5,6,9,D,B}) {
      wc = w[10:0];
      if (wc == 0) {
          /* wc is in next word. */
        w = next_word;
        wc = w[19:0];
      }
      while (wc--  > 0) {
        w = next_word;
        sw[35:0] = addr, word;
        for (i=0; i<36; i++) {
          x16 = bcc[15] XOR sw[i];
          x15 = bcc[14] XOR x16;
          x2 = bcc[1] XOR x16;
          bcc[15:0] =
              x15,bcc[13:2],x2,bcc[0],x16;
        }
        /* Note the bit order */
        crc[15:0] = bcc[0:15];
      }
    }
  }
  else {
    /* Pad word - ignore */
  }
} while (more_words)
```

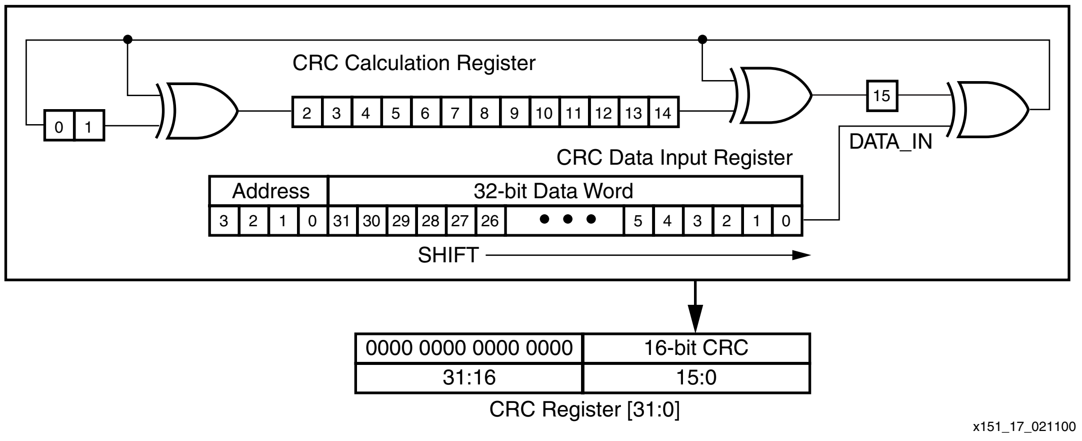

*Figure 17:* **CRC Calculation Register**

## Frame Address Register (FAR)

The Frame Address Register (FAR) holds the address of the current frame. The address is divided into three parts, the block type, the major address, and the minor address. The block type field indicates whether the CLB or block RAM address space is used. The command in the command register is executed each time the FAR is loaded with a new value.

The major address selects the CLB or RAM column, the minor address selects the frame within the column. The minor address is incremented each time a full data frame is read from or written to the frame data register. If the last frame within the CLB column is selected when the increment occurs, the major address is incremented and the minor address is reset to zero, otherwise the minor address is incremented. However, the block RAM major address is not incremented automatically.

Note: the block RAM Major Address is not incremented automatically. To address a different block RAM content column, the FAR must be loaded with the new Major Address.

See Figure 18 for the definitions of valid values for the block type field. The FAR field definitions are shown in Figure 19.

| Type | Codes |
|---|---|
| CLB | 00 |
| RAM | 01 |

*Figure 18:* **Block Type Codes**

| | | | | | Block Type | | Major Address (Column Address) | | | | | | | | Minor Address (Frame Address) | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Notes:**
1. Locations within fields containing a zero or one must have these values. An *X* in a bit field indicates that the value is variable and must be set.
2. Heavy vertical lines are used to separate fields. Light vertical lines separate nibbles in the word.

*Figure 19:* **Frame Address Fields (FAR)**

## Frame Data Input Register (FDRI)

The Frame Data Input Register (FDRI) is used to load configuration frame data into a Virtex device. The Frame Data Register (FDR) is a shift register into which data is loaded prior to transfer to the configuration memory. Configuration data is written to the Virtex device by loading the command register with the WCFG command and then loading the FDR with at least two frames of 32-bit words.

The write operation is pipelined such that the first frame of data is written to the configuration memory while the second frame is being shifted in. The last frame (the pad frame) is always dummy data which is not actually written to the configuration memory. Each frame write must include enough 32 bit data words to load the frame fully. There is one pad word at the end of each frame which is required for the pipelining hardware.

## Frame Data Output Register (FDRO)

The Frame Data Output Register (FDRO) is for reading configuration data or captured data from the Virtex device, a process called readback. Readback is performed by loading the command register with the RCFG command and then addressing the FDRO with a read command.

## Frame Length Register (FLR)

Near the beginning of the configuration bitstream the Frame Length Register (FLR) is written with the length of a frame, as measured in 32-bit words. This length count is used to provide sequencing information for the configuration read and write operations. Note that the FLR must be written before any FDR operation works. It is not necessary to set the FLR more than once. If the number of bits in a frame is not evenly divisible by 32, the length count of the frame must be rounded up to the next highest integer. The values for the FLR for all the current Virtex series devices are given in Table 24.

Note: The FLR contains a value that is one less than the number of words that are read from or written to a given frame. This is because the extra word needed for pipelining is not counted.

*Table 24:* **Frame Length Register Value**

| Device | Row × Col | Frame Length | # Words per Frame | FLR Value |
|---|---|---|---|---|
| XCV50/E | 16 × 24 | 384 | 12 | 11 |
| XCV100/E | 20 × 30 | 448 | 14 | 13 |
| XCV150 | 24 × 36 | 512 | 16 | 15 |
| XCV200/E | 28 × 42 | 576 | 18 | 17 |
| XCV300/E | 32 × 48 | 672 | 21 | 20 |
| XCV400/E | 40 × 60 | 800 | 25 | 24 |
| XCV405E | 40 × 60 | 800 | 25 | 24 |
| XCV600/E | 48 × 72 | 960 | 30 | 29 |
| XCV800 | 56 × 84 | 1088 | 34 | 33 |
| XCV812E | 56 × 84 | 1088 | 34 | 33 |
| XCV1000/E | 64 × 96 | 1248 | 39 | 38 |
| XCV1600E | 72 × 108 | 1376 | 43 | 42 |
| XCV2000E | 80 × 120 | 1536 | 48 | 47 |
| XCV2600E | 92 × 138 | 1728 | 54 | 53 |
| XCV3200E | 104 × 156 | 1952 | 61 | 60 |

## Legacy Output Register (LOUT)

The Legacy Output Register (LOUT) is used for daisy chaining the configuration bitstream to other Xilinx devices. Data written to the LOUT is serialized and appears on the DOUT pin.

## Mask Register (MASK)

The Mask Register (MASK) is a mask register for writes to the CTL register. A "1" in bit N of the mask allows that bit position to be written in the CTL register. The default value of the mask is all "0"s.

## Status Register (STAT)

The Status Register (STAT) is loaded with current values of several control or status signals. The register can be read via the re-configuration block or via JTAG. The fields in the Status register are illustrated in Figure 20. The values of the signals given in Table 25, page 25 can be read from the status register.

| | | | | | | | | | | | | | | | | | DONE | INIT | | | MODE | GHIGH_B | GSR_B | GWE_B | GTS_CFG | IN_ERROR | | | | LOCK | CRC_ERROR |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

**Notes:**
1. An *X* in a bit field indicates that the value is variable.
2. Heavy vertical lines are used to separate fields. Light vertical lines separate nibbles in the word.

*Figure 20:* **Status Register Fields**

## Configuration Interface

*Table 25:* **Status Register Bits**

| Name | Bit Indices | Description |
|---|---|---|
| DONE | 14 | Input from DONE pin |
| INIT | 13 | Value of $\overline{\text{INIT}}$ |
| MODE | 12:10 | Value of M2, M1, M0 mode pins |
| GHIGH_B | 9 | 0 = GHIGH_B asserted |
| GSR_B | 8 | 0 = all flip-flops are Reset/Set |
| GWE_B | 7 | 1 = flip-flops and block RAM are write disabled |
| GTS_CFG | 6 | 0 = I/Os are tri-stated |
| IN_ERROR | 5 | Legacy input error |
| LOCK[3] | 4 | Output from Logical AND of DLL lock signals in bottom-left quadrant.<br>1 = DLL is locked. |
| LOCK[2] | 3 | Output from Logical AND of DLL lock signals in bottom-right quadrant.<br>1 = DLL is locked. |
| LOCK[1] | 2 | Output from Logical AND of DLL lock signals in top-left quadrant.<br>1 = DLL is locked. |
| LOCK[0] | 1 | Output from Logical AND of DLL lock signals in top-left quadrant.<br>1 = DLL is locked. |
| CRC_ERROR | 0 | Indicates that a CRC error has occurred. |

There are two configuration interfaces to the Virtex devices — the bit-serial Boundary Scan interface and the 8-bit byte-serial SelectMAP interface. Conceptually, XCV50 configuration data appears as in Figure 21.

| | |
|---|---|
| Data Frame 0 (11 words) | Pad Word |
| ⋮ | |
| Data Frame *n* (11 words) | Pad Word |
| Pad Frame (12 words) | |

*Figure 21:* **XCV50 Frame Padding for Reads**

Frames and words within frames are written in the same order in both configuration interfaces, starting with Frame 0, word 0 (the left-most in the picture), followed by word 1, etc. Bits within each word are written from left to right (*MSB first*) in the bit-serial configuration interfaces.

Within the SelectMAP interface, data is written a byte at a time. A sample word is shown in Figure 22. The top row indicates the device pin names. The bottom row indicates the bit indices within a configuration word. Byte 0 loads first, followed by byte 1, *et cetera*. The MSB of each byte (*i.e.*, bits 31, 23, 15, and 7) is loaded on pin D0. The LSB of each byte (*i.e.*, bits 24, 16, 8, and 0) is loaded on pin D7.

| ← Byte 0 → | | | | | | | | ← Byte 1 → | | | | | | | | ← Byte 2 → | | | | | | | | ← Byte 3 → | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 |

x151_22_021100

*Figure 22:* **SelectMAP Byte and Bit Ordering**

## Partial Reconfiguration of CLBs

Partial reconfiguration can be performed with and without shutting down the device. If contention occurs it is advised to go through a shutdown sequence prior to loading configuration frames. When shutting down the DONE pin will go active and pull low. If this is not desireable the DONE pin can be prevented from going low by selecting the DONE_CYCLE to Keep State (111). Similarly, to prevent GSR from becoming active and reseting the current state, the GSR_CYCLE can be set to Keep State. When not shutting down, changes occur frame by frame, but is completely done by the end of the Last Frame packet (see Example 1).

The DONE pin will not change in this case. Figure 23 illustrates a typical shutdown, reconfiguration in the CLB address space and restart sequence.

```
FFFF FFFF Dummy Word
AA99 5566 Sync Word. Does not realign an already synchronized SelectMap port
3001 2001 COR
0080 FF2F Shutdown bit set, optionally set bit 29 DRIVE_DONE
          Alternately COR may be read, bit 15 set, and written back
3000 8001 CMD
0000 0005 Start Shutdown
3000 8001 CMD
0000 0007 Reset CRC
0000 0000
0000 0000 Clock shutdown sequence
0000 0000
0000 0000
3000 8001 CMD
0000 0008 Assert GHIGH
3000 8001 CMD
0000 0001 Write Configuration
3000 2001 FAR
0--- ---- Block Type CLB (00); Major/Minor Address
3000 4--- FDRI + Word Count if count is <1024. Otherwise use TypeII Header.

          Write FRAME DATA
          If CLB Frames are not written to non-consecutive addresses repeat
          FAR word, followed by new Major/Minor Address followed by FDRI +
          Word Count and then Frame Data.

3000 0001 Write CRC
---- ---- CRC
3000 8001 CMD
0000 0003 LFRM
3000 4--- FDRI + Word Count

Write PAD FRAME

3001 2001 COR
0080 3F2D Default COR Options
          Alternately COR may be read, bit 15 cleared, and written back
3000 8001 CMD
0000 0005 Begin Startup
3000 0001 Write CRC
---- ---- CRC
0000 0000
0000 0000 4 Dummy Words
0000 0000
0000 0000
```

*Figure 23:* **Typical Shutdown, Reconfiguration in the CLB Address Space, and Restart Sequence**

## Examples

Several examples of reading and evaluating configuration data are provided to illustrate the following.

### Example 1: Read and Write Semaphores in an XCV100 at CLB R1 C1, Slice 0.

Semaphores are a useful communication mechanism documented in XAPP 153, "Status and Control Semaphore Registers using Partial Reconfiguration." Figure 24 shows an abstraction of a microprocessor writing control information to an FPGA and reading status information. One convention for implementing semaphores in Virtex devices is to use two bits of a 16-bit, dual-port RAM, as illustrated in Figure 25. This occupies one CLB slice with the F-LUT implementing the control semaphore and the status semaphore implemented in the G-LUT. Address 15 of the G-LUT is used for on-chip writes to the semaphore and address 14 of the F-LUT is used for on-chip reads. Conversely, the off-chip microprocessor is reading the G-LUT[15], and writing F-LUT[14].

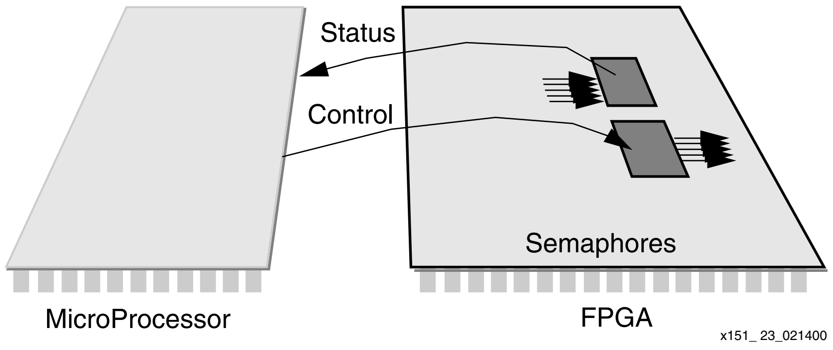

*Figure 24:* **Semaphore Abstraction**

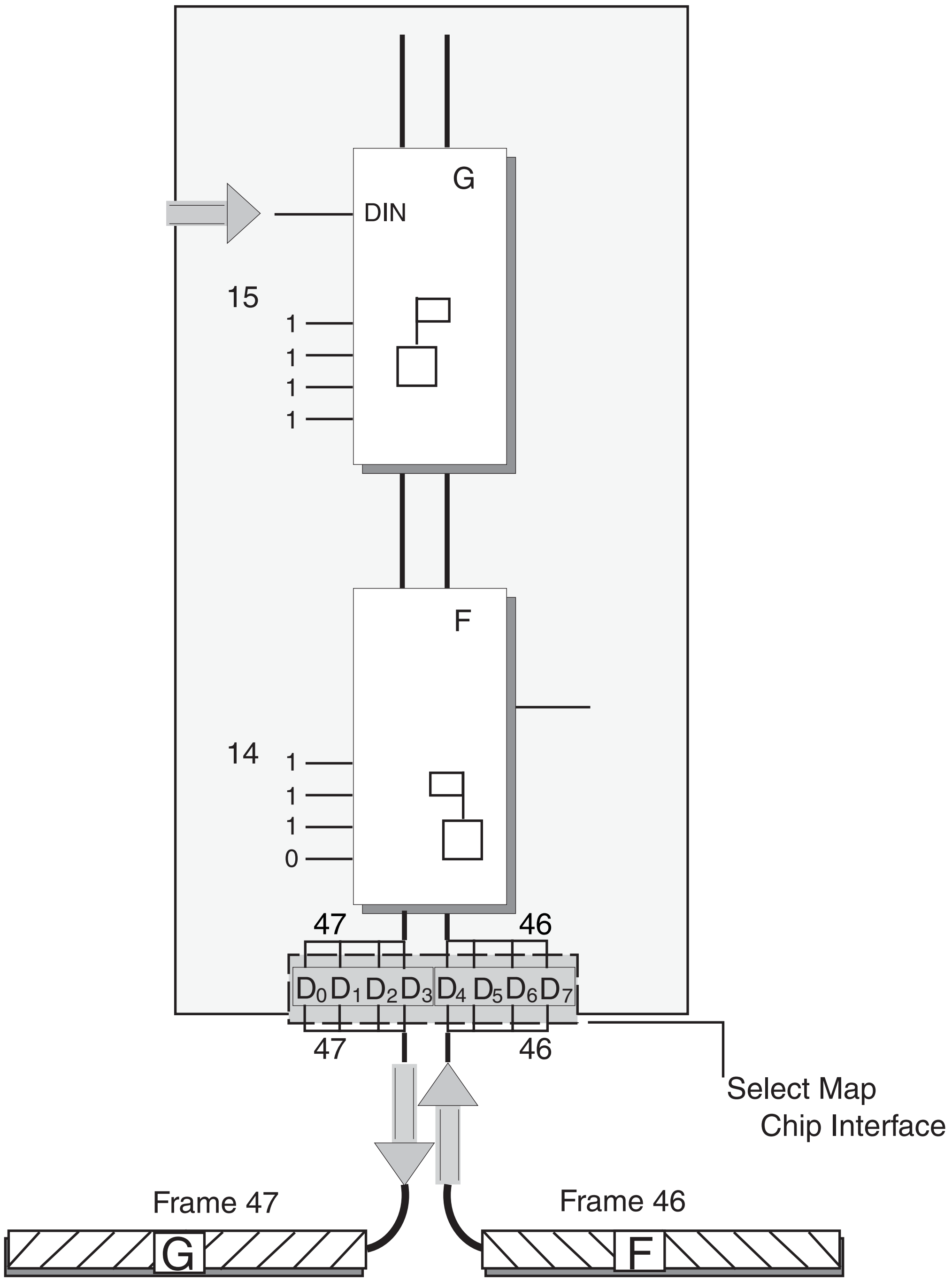

*Figure 25:* **Semaphore Read/Write Implementation**

Using a Semaphore Abstraction dual-port RAM ensures that the LUT SelectRAMs are placed in the CLB in a predictable manner. *When writing data to one or more LUT SelectRAMs or flip-flops on the device, all bits in the frame **must** have valid configuration information.* This is assured by altering valid configurations from bitstream files or from frames read from a properly configured Virtex device. The latter approach is used in this example.

Attributes for this design are summarized in Table 26.

*Table 26:* **Design Attributes for Example 1**

| Attribute | G-LUT [15] | F-LUT [14] | |
|---|---|---|---|
| | **Read** | **Read** | **Write** |
| Chip_Rows | 20 | | |
| Chip_Cols | 30 | | |
| FL | 14 | | |
| CLB_Row | 1 | | |
| CLB_Col | 1 | | |
| Slice | 0 | | |
| FG | 1 | 0 | |
| lut_bit | 15 | 14 | |
| RW | 1 | 1 | 0 |

From the equations in Table 7, page 11, the values shown in Table 27 can be calculated.

*Table 27:* **Semaphore Example Variables, Equations, and Values**

| Variable | Equation | Value(s) | | |
|---|---|---|---|---|
| | | **G-LUT[15]** | **F-LUT[14]** | |
| | | **Read** | **Read** | **Write** |
| MJA | $1 \leq 30/2 \Rightarrow 30 - 1 \times 2 + 2$ | 30 | | |
| MNA | $\text{lut\_bit} + 32 - 0 \times (...)$ | 47 | 46 | |
| fm_bit_idx | $3 + 18 \times 1 - FG + RW \times 32$ | 52 | 53 | 21 |
| fm_st_wd | $14 \times (8 + (30 - 1) \times 48$<br>$+ \{46,47\}) + RW \times (14 + 1)$<br>$= 14 \times (1{,}400 + \{46,47\}) + 14 \times RW$<br>$= 19{,}600 + 14 \times \{46,47\} + 14 \times RW$ | 20,272 | 20,258 | 20,244 |
| fm_wd | floor (20/32) | 1 | 1 | 0 |
| fm_wd_bit_idx | $31 + 32 \times \{1,1,0\} - \{52,53,21\}$ | 11 | 10 | 10 |

From off-chip, for reading the G-LUT[15] bit, read one frame (MNA=47). For writing the F-LUT[14] bit, we show how to read then write one frame (MNA=46), as opposed to modifying data from a bitstream file (both are valid methods). The frames on the XCV100 contain 13 32-bit words and one pad word. Remember that fm_st_wd is calculated assuming the entire configuration has been read. However, only the pad frame and then frames 46 and 47 from Major Address 30 are being read. The desired bit is in Frame 1, word #1, which is word 15.

The commands for reading both frames (and the pad frame) are given in Figure 26.

| Instruction | Data | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Hex | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Sync Word | AA99 5566 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| Write next (1) word to FAR | 3000 2001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| CLB MJA=30, MNA=46 | 003C 5800 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Write next word to CMD | 3000 8001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Register value for RCFG | 0000 0004 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read from FDRO | 2800 602A | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| Flush pipe | 0000 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| (read 42 words) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**Notes:**

1. Binary data is grouped in two ways for ease of interpretation. Thin vertical lines separate nibble boundaries. Heavy vertical lines separate field boundaries.

*Figure 26:* **Commands to Read Two Data Frames**

The 42 words read are shown in Figure 27. F-LUT[14] is in the second frame (frame=1, word=1) at bit 10. The value read is a "1", but because the LUT bits are inverted, the logic value is zero. G-LUT[15] is in word one of the third data frame (frame=2, word=1) at bit 11.The value read is a "1", which is a logic zero.

| Frame | Word | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 3 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 4 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 5 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 6 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 7 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 10 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 11 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 12 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 13 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | F[14] |
| | 2 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 3 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | |
| | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | |
| | 5 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | |
| | 6 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 7 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | |
| | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | |
| | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | |
| | 10 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 11 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 12 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | |
| | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| **2** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | G[15] |
| | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 3 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | |
| | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 5 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | |
| | 6 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 7 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | |
| | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | |
| | 10 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 11 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 12 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | |
| | 13 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

*Figure 27:* **Three Frames Containing the Semaphores**

To write to the F-LUT semaphore, start with the second data frame (frame=1) that was just read. Words 1 - 13 of that frame will become words 0 - 12 of the frame to be written. Word 13 of this new frame is a pad word and can have any value (typically 0 is chosen). In this new frame, set bit 11 of word 0 (zero) to the desired value of the semaphore. A pad frame must follow the data frame. The commands from Figure 28 write these two frames into the device at the proper location.

| Instruction | Data | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Hex | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Sync Word | AA99 5566 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| Write next (1) word to FAR | 3000 2001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| MJA=30, MNA=46 | 003C 5C00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Write next word to CMD | 3000 8001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Register value for WCFG | 0000 0001 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Write 28 words to FDRI | 3000 401C | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| Data Word 0 | A01A EC00 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Data Word 1 | FF0F 3FC0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Data Word 2 | 0FF0 02FC | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| ⋮ | ⋮ | ⋮ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Data Word 27 | 0000 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Notes:**

1. Binary data is grouped in two ways for ease of interpretation. Thin vertical lines separate nibble boundaries. Heavy vertical lines separate field boundaries.

*Figure 28:* **Commands to Write a Semaphore Value**

## Example 2: Reading the Complete Configuration from an XCV50.

Steps:

1. If flip-flop values are needed, clock the on-chip signal, CAPTURE, to capture flip-flop values. See the Xilinx Libraries Guide for use of the CAPTURE_VIRTEX cell.
2. Write the starting frame address (CLB MJA=0 MNA=0) into the FAR.
3. Write the RCFG command to the CMD register.
4. Address the FDRO register with a READ operation and word count equal to the number of 32-bit words in the CLB frames plus one pad frame.
5. Read the data following the timing diagrams in the SelectMAP interface section.
6. Write the address for RAM block 0 to the FAR.
7. Address the FDRO register with a read operation and word count equal to the number of 32-bit words in the RAM block plus one pad frame.
8. Read the data.
9. Write the address for RAM block 1 to the FAR.
10. Address the FDRO register with a read operation and word count equal to the number of 32-bit words in the RAM block plus one pad frame.
11. Read the data.

When using SelectMAP mode to read data words from the Virtex device, de-assert $\overline{CS}$, de-assert $\overline{WRITE}$, assert $\overline{CS}$, then clock the data out. When using JTAG, load the JTAG IR (instruction register) with the CFG_OUT instruction. Then go to the SDR (Shift-DR) state and shift the data out. (See Figure 29).

| Instruction | Data | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Hex | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Sync Word | AA99 5566 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| Write next word to FAR. | 3000 2001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| CLB MJA=0, MNA=0 | 0000 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Write next word to CMD register. | 3000 8001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Register value for RCFG | 0000 0004 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read from FDRO register. | 2800 6000 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15876 words | 4800 3E04 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Flush pipe | 0000 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| (Read 15876 words.) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Write to FAR register. | 3000 2001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| RAM MJA=0, MNA=0 | 0200 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read 780 words from FDRO reg. | 2800 630C | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| Flush pipe | 0000 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| (Read 780 words.) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Write to FAR register. | 3000 2001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| RAM MJA=1, MNA=0 | 0202 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read 780 words from FDRO reg. | 2800 630C | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| Flush pipe. | 0000 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| (Read 780 words.) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

*Figure 29:* **Example 2 - Read Complete Configuration for XCV50**

## Example 3: Read the Slice 0 G-LUT from CLB R1 C1 from the Complete Configuration of an XCV50.

The commands for reading the bitstream from the Virtex device are given in **Example 1: Read and Write Semaphores in an XCV100 at CLB R1 C1, Slice 0.**, page 28. Using an XCV50 device, the independent attributes are show in Table 28:

*Table 28:* **XCV50 Independent Attributes**

| Independent Attributes | Values |
|---|---|
| Chip_Rows | 16 |
| Chip_Cols | 24 |
| FL | 12 |
| CLB_Row | 1 |
| CLB_Col | 1 |
| FG | 1 |
| Slice | 0 |
| RW | 1 |

From the equations given earlier in Table 7, calculating the range of values for fm_st_wd indicates that the word 13740 of the configuration is the starting word of the 12-word (FL=12) frame containing bit 0 of the G-LUT in Slice 0 of CLB R1C1 (Table 29).

*Table 29:* **Variables, Equations and Values for Slice 0 G-LUT**

| Variables | Equations | Values | | | |
|---|---|---|---|---|---|
| MJA | 1 ≤24/2 ⇒24 – 1× 2 + 2 | 24 | | | |
| MNA | 0 × (…) + lut_bit + 32 | 0: 32<br>1: 33<br>2: 34<br>3: 35 | 4: 36<br>5: 37<br>6: 38<br>7: 39 | 8: 40<br>9: 41<br>10: 42<br>11: 43 | 12: 44<br>13: 45<br>14: 46<br>15: 47 |
| fm_bit_idx | 3 + 18 × 1 –1 + RW × 32 | 52 | | | |
| fm_st_wd | 12 × (8+ (24 –1) × 48 + [32:47]) +1 x 12<br>= 12 × (1112 + [32:47]) +12<br>= 13,356 + 12 × 32:47] | 13,740:13,920 | | | |
| fm_wd | floor(52/32) | 1 | | | |
| fm_wd_bit_idx | 31 + 32 – 52 | 11 | | | |

The configuration bits for the given frame are as follows. G-LUT[0] is in fm_wd=0, at bit #11.

| Bitstream Word | Frame 32-bit Word | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | Frame Word |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 13740 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13741 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 13742 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| 13743 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 3 |
| 13744 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 4 |
| 13745 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 5 |
| 13746 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 |
| 13747 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 7 |
| 13748 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 8 |
| 13749 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 9 |
| 13750 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 10 |
| 13751 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 11 |

*Figure 30:* **Configuration Bits for Slice 0, CLB R1C1 G-LUT [0]**

All 16 LUT SelectRAM bits are in the following words at the same bit index, 11.

| Bitstream Word | Frame 32-bit Word | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | LUT Bit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 13741 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13753 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 13765 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| 13777 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
| 13789 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |
| 13801 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 |
| 13813 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 |
| 13825 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 7 |
| 13837 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 8 |
| 13849 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 9 |
| 13861 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 10 |
| 13873 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 11 |
| 13885 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 12 |
| 13897 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13 |
| 13909 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 14 |
| 13921 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15 |

*Figure 31:* **Location of all 16 LUT SelectRAM Bits**

The 16 bits are $\overline{\text{LUT}}$[15:0]=0111111111111111. The LUT SelectRAM bits are inverted from their logic values. The "logical" contents are LUT[15:0]=1000000000000000. Thus, this G-LUT implements a 4-input AND function.

## Example 4: Read the Slice 1 F-LUT from CLB R19 C16 from an XCV100.

Commands for reading the bitstream from the Virtex device are given in Figure 32. Use the following independent attributes to find the given F-LUT.

*Table 30:* **Virtex Bitstream Command Attributes**

| Independent Attributes | Values |
|---|---|
| Chip_Rows | 20 |
| Chip_Cols | 30 |
| FL | 14 |
| CLB_Row | 19 |
| CLB_Col | 16 |
| FG | 0 |
| Slice | 1 |
| RW | 1 |

From the equations in Table 7, page 11, calculating fm_st_wd indicates the starting word with respect to a configuration that starts at MJA=0, MNA=0. Because the frames we are interested in start at MJA=1, MNA=0, which is fm_st_wd = 126, so the first 126 words are not needed (0–125). Therefore, to find the given Slice 1 F-LUT, see Table 31.

*Table 31:* **Variables, Equations, and Values for Slice 1 F-LUT**

| Variables | Equations | Values | | | |
|---|---|---|---|---|---|
| MJA | $16 > 30/2 \Rightarrow 2\times 16 - 30 - 1$ | 1 | | | |
| MNA | lut_bit + 32 – Slice × (2 × lut_bit +17)<br>= [0:15] + 32 – (2 × [0:15] + 17)<br>= 15 – [0:15] | 0: 15<br>1: 14<br>2: 13<br>3: 12 | 4: 11<br>5: 10<br>6: 9<br>7: 8 | 8: 7<br>9: 6<br>10: 5<br>11: 4 | 12: 3<br>13: 2<br>14: 1<br>15: 0 |
| fm_bit_idx | $3 + 18\times 19 - 0 + 32$ | 377 | | | |
| fm_st_wd | 14 × (8 + (1– 1) × 48 + [15:0]) +1 × 14<br>= 14 × (8 + [15:0]) +14<br>= 126 + 14 × [15:0] | = 336:126 | | | |
| fm_wd | floor(377/32) | 11 | | | |
| fm_wd_bit_idx | $31 + 32\times 11 - 377$ | 6 | | | |

Sixteen frames need to be read, one for each bit in the LUT SelectRAM. The bits in LUT SelectRAMs in Slice 1 occur in the opposite order that they do for Slice 0 LUT SelectRAMs.

Frames are read sequentially with ascending addresses. If read in LUT bit order, LUT[0], LUT[1], …, LUT[15]. These are stored in descending addresses which require 16 separate read operations each reading one data frame and one pad frame. However, if read in ascending address order, LUT[15], LUT[14], …, LUT[0], all 16 data frames are read with a single read operation. This requires only one pad frame for all 16 frames. Thus, it takes less time to read ascending frames starting at MJA=1, MNA=0 and finishing with frame MJA=1, MNA=15. The frames in the XCV100 contain 14 32-bit words and a single pad word. Commands for reading only the F-LUT data are given in Figure 32.

| Instruction | Data | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Hex | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Sync Word | AA99 5566 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| Write next (1) word to FAR. | 3000 2001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| CLB MJA = 1, MNA = 0 | 0002 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Write next word to CMD. | 3000 8001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Register value for RCFG | 0000 0004 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read from FDRO. | 2800 60EE | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| Flush pipe. | 0000 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| (Read 238 words.) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

*Figure 32:* **Commands to read Slice 1 F-LUT**

The 377th bit of each frame is the bit in the F-LUT. This LUT SelectRAM bit is in the frame's word index 11, bit index 6.

The configuration bits for the given frame in Figure 33 are as follows: LUT Bit 15 is in MJA=1, MNA=0, fm_wd=11, at bit #6.

| Frame Word | Frame 32-bit Word | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 5 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 6 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 10 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |

*Figure 33:* **Frame Containing F-LUT[15]**

LUT bit 0 is in MJA=1, MNA=15, fm_wd=11, at bit #6.

| Bit stream Word | Frame 32-bit Word | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | FmWd |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 336 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 337 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 338 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| 339 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 3 |
| 340 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 4 |
| 341 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 5 |
| 342 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 |
| 343 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 7 |
| 344 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 8 |
| 345 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 9 |
| 346 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 10 |
| 347 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 11 |
| 348 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 12 |
| 349 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13 |

*Figure 34:* **Frame Containing F-LUT [0]**

For the sake of brevity, here are the sixteen 11th words in the order they appear in the bitstream.

| Bitstream Word | Frame 32-bit Word | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | LUT Bit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 137 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15 |
| 151 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 14 |
| 165 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 13 |
| 179 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 12 |
| 193 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 11 |
| 207 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 10 |
| 221 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 9 |
| 235 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 8 |
| 249 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 7 |
| 263 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 6 |
| 277 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 5 |
| 291 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |
| 305 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
| 319 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| 333 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 347 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

*Figure 35:* **Sixteen Words Containing the F-LUT bit**

The bits are $\overline{\text{LUT}}$[15:0]=`0111111111111111`. The LUT SelectRAM bits are inverted from the logic sense. The logical contents are LUT[15:0]=`1000000000000000`. Thus, this F-LUT implements a 4-input AND gate.

## Example 5: Read All Bits in Slice 0 G-LUTs from CLB C2 and XCV50.

Given the following attributes, the necessary values to find the G-LUT data can be computed.

*Table 32:* **All Bits: Slice 0 G–LUTs from CLB C2 and XCV50**

| Independent Attributes | Values |
|---|---|
| Chip_Rows | 16 |
| Chip_Cols | 24 |
| FL | 12 |
| CLB_Row | 1:16 |
| CLB_Col | 2 |
| FG | 1 |
| Slice | 0 |
| lut_bit | 0:15 |
| RW | 1 |

From the equations in Table 7, the dependent variables can be calculated.

*Table 33:* **Dependent Variables, Equations, and Values**

| Variables | Equations | Values | | | |
|---|---|---|---|---|---|
| MJA | $2 \le 24/2 \Rightarrow 24 - 2\times\ 2 + 2$ | 22 | | | |
| MNA | lut_bit + 32 – Slice $\times$ (2 $\times$ lut_bit + 17) = lut_bit + 32 – 0 $\times$ (2 $\times$ lut_bit + 17) = [0:15] + 32 | 0:32 | 4:36 | 8:40 | 12:44 |
| | | 1:33 | 5:37 | 9:41 | 13:45 |
| | | 2:34 | 6:38 | 10:42 | 14:56 |
| | | 3:35 | 7:39 | 11:43 | 15:47 |
| fm_bit_idx | 3 + 18 $\times$ CLB_Row – 1+ 32 = 34 + 18 $\times$ [1:16] | 1:52 | 5:124 | 9:196 | 13:268 |
| | | 2:70 | 6:142 | 10:214 | 14:286 |
| | | 3:88 | 7:160 | 11:232 | 15:304 |
| | | 4:106 | 8:178 | 12:250 | 16:322 |
| fm_st_wd | 12 $\times$ (8 + (22 –1) $\times$ 48 + MNA) + 1 $\times$ 12 = 12 $\times$ (1,016 + MNA) + 12 = 12,204 + 12 $\times$ MNA | 0: 12,588 | 4: 12,636 | 8: 12,684 | 12: 12,732 |
| | | 1: 12,600 | 5: 12,648 | 9: 12,696 | 13: 12,744 |
| | | 2: 12,612 | 6: 12,660 | 10: 12,708 | 14: 12,756 |
| | | 3: 12,624 | 7: 12,672 | 11: 12,720 | 15: 12,768 |
| fm_wd | floor (fm_bit_idx/32) | 1:1 | 5:3 | 9:6 | 13:8 |
| | | 2:2 | 6:4 | 10:6 | 14:8 |
| | | 3:2 | 7:5 | 11:7 | 15:9 |
| | | 4:3 | 8:5 | 12:7 | 16:10 |
| fm_wd_bit_idx | 31 + 32 $\times$ fm_wd – fm_bit_idx | 1:11 | 5:3 | 9:27 | 13:19 |
| | | 2:25 | 6:17 | 10:9 | 14:1 |
| | | 3:7 | 7: 31 | 11:23 | 15:15 |
| | | 4:21 | 8:13 | 12:5 | 16:29 |

Note that fm_bit_idx, fm_wd, and fm_wd_bit_idx have 16 values, one for each row on the XCV50. Figure 37 shows where the data lies in the first frame, which contains G[0] for the entire column. The process is the same for the other 15 frames. The commands for reading the LUT SelectRAM data are given in Figure 36.

From the calculation for fm_st_wd, Frame 0 would start at word 12,588 reading the whole configuration. The instructions in Figure 36 start at that word, so word 0 (ignoring the 12 words in the pad frame) is the same as word 12,588 of the entire CLB configuration. The 12 words in the frame are shown in Figure 37.

The LUT SelectRAM bits have been shaded for ease of identification. It can be seen from the calculation of fm_bit_idx that the LUT SelectRAM bits are in the order 1:16. For example, from the above calculations for fm_wd and fm_wd_bit_idx, G-LUT[0] in R1C2 is in fm_wd 1, fm_wd_bit_idx 11. This bit is the shaded bit in word 1 at bit index 11. Similarly, the G-LUT[0] for the other 15 CLB rows are also shaded in this table.

| Instruction | Data | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Hex | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Sync Word | AA99 5566 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| Write next (1) word to FAR. | 3000 2001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| CLB MJA=22, MNA=32 | 002C 4000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Write next word to CMD. | 3000 8001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Register value for RCFG | 0000 0004 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read from FDRO. | 2800 60CC | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| Flush pipe. | 0000 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| (Read 204 words.) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

*Figure 36:* **Commands to read R*C2.S0 G-LUT**

| Word | Frame 32-bit Word | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | CLB Row |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 2 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2, 3 |
| 3 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 4, 5 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 6 |
| 5 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 7, 8 |
| 6 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 9, 10 |
| 7 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 11, 12 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 13, 14 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 15 |
| 10 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 16 |
| 11 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

*Figure 37:* **Frame for R*C2.S0 G-LUTs, Bit G[0]**

## Example 6: Read Block SelectRAM index 387 of RAM R2 C0 from an XCV100E.

Commands for reading the bitstream from the Virtex device are given in Figure 38. Use the following independent attributes to find the given F-LUT.

*Table 34:* **Virtex Bitstream Command Attributes**

| Independent Attributes | Values |
|---|---|
| Chip_Rams | 4 |
| FL | 14 |
| RAM_Row | 2 |
| RAM_Col | 0 |
| ram_bit | 387 |
| RW | 1 |

Since we are attempting to read only one bit location, only one frame is necessary to be read. We use the equations of Table 15, Table 16, and Table 17 to determine the dependent variables.

*Table 35:* **Variables, Equations, and Values for Slice 1 F-LUT**

| Variables | Equations | Values |
|---|---|---|
| MJA_virtex | $0 < 4/2 \Rightarrow 2 \times (4/2 - 1 - 0)$ | 2 |
| MJA_adj | $0 < 4/2 \Rightarrow 2$ | 2 |
| MJA | MJA_virtex + MJA_adj | 4 |
| MNA | floor(((387/64)%64)/32)+ 2 x floor(((387/64)%32)/16) + 4 x floor(((387/64)%16)/8) + 8 x floor(((387/64)%8)/4) + 16 x floor(((387/64)%4)/2) + 32 x floor(((387/64)%2)/1) | 24 |
| fm_bit_idx | $3 + 18 \times 19 - 0 + 32$ | 219 |
| fm_st_wd | 14 x 24 + 1 x 14 | 350 |
| fm_wd | floor(219/32) | 6 |
| fm_wd_bit_idx | $31 + 32 \times 6 - 219$ | 4 |

Commands for reading this memory index are given in Figure 38.

| Instruction | Hex | Data | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Sync Word | AA99 5566 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| Write next (1) word to FAR. | 3000 2001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| CLB MJA = 4, MNA = 24 | 0208 3000 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Write next word to CMD. | 3000 8001 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Register value for RCFG | 0000 0004 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read from FDRO. | 2800 601C | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| Flush pipe. | 0000 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| (Read 28 words.) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

*Figure 38:* **Commands to Read Block SelectRAM Index 387**

RAM bit 387 is at fm_wd=6, bit #4.

| Frame Word | Frame 32-bit Word | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 5 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 6 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 10 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |

*Figure 39:* **Frame MNA=24**

## Glossary

**Block SelectRAM Resource**

One of several large, fully-synchronous, dual-port memories in the Virtex FPGAs. Each of these memories contain 4,096 bits. The organization of each memory is configurable. The block SelectRAM resource complements the smaller, distributed, LUT SelectRAMs.

**Boundary Scan Interface**

One of the configuration interfaces on the Virtex device. This is a bit-serial interface. The Boundary Scan interface is also known as the JTAG port. Also see the SelectMAP interface.

**Capture Data**

The flip-flop and pad data saved from the logic cells and I/O blocks into the bitstream. Use the CAPTURE_VIRTEX primitive in your HDL code to specify the trigger and clock for the capture operation.

**Configurable Logic Block (CLB)**

The functional elements for constructing logic circuits. The Virtex CLB is made up of Slices, which contain Logic Cells.

**Configuration Bitstream**

Configuration commands, optionally with configuration data.

**Configuration Commands**

Instructions for the Virtex device. There are two classes of Configuration Command — Major and Minor. The Major Commands read and write data to configuration registers in the Virtex device. The Minor commands instruct the Virtex configuration logic to perform specific functions. See "Command Register (CMD)" on page 18.

**Configuration Data**

Bits that directly define the state of programmable logic. These are written to a Virtex device in a configuration bitstream, and read as Readback Data from a Virtex device.

### Configuration Frame

The configuration bits in a Virtex device are organized in columns. A column of CLBs with the I/O blocks above and below the CLBs contain 48 frames of configuration bits. The smallest number of bits that can be read or written through the configuration interfaces is one frame.

### Configuration Interface

A logical interface on the Virtex device through which configuration commands and data can be read and written. A interface consists of one or more physical Device Pins.

### Configuration Readback

The operation of reading Configuration Data (also known as Readback Data) from a Virtex device.

### Device Pin

One of the electrical connections on the package containing the Virtex device.

### Frame

See Configuration Frame.

### Logic Cell (LC)

The basic building block of the Virtex CLB. An LC includes a 4-input function generator, carry logic, and a storage element.

### LUT SelectRAMs

Shallow RAM structures implemented in CLB Lookup Tables (LUTs). See also block SelectRAM section.

### Pad

Pad bits are extra bits used to make the total number of bits in a frame an integral multiple of 32, the number of bits in a configuration word. A Pad Word is an extra word used at the end of a Configuration Frame for pipelining. A Pad Frame is an extra Configuration Frame used at the beginning of a Configuration Readback and at the end of a Configuration Write for pipelining.

### Readback Data

Configuration data read from a Virtex device. The data is organized as Configuration Frames.

### SelectMAP Interface

One of the configuration interfaces on the Virtex device. This is a byte-serial interface. The pins in the SelectMAP interface may be used as user I/O after configuration has been completed or remain configured as a configuration interface.

### Slice

A subdivision of the Virtex CLB. There are two, vertical, slices in a Virtex CLB. Each slice contains two Logic Cells.

### Sync Word

A 32-bit word with a value that is used to synchronize the configuration logic.

## Revision History

The following table shows the revision history for this document.

| Date | Version | Revision |
|---|---|---|
| 06/17/99 | 1.0 | Initial Release |
| 07/27/99 | 1.1 | Updated the following tables: Table 5: equation for MNA; Table 7: equations for MJA & fm_bit_idx. |
| 09/20/99 | 1.2 | Changes in page 5. Most formulae in Table 7 have been revised. Documented pad capture value P. Corrected bit locations in Examples. |
| 02/22/00 | 1.3 | Updated to include Virtex-E block SelectRAM. Corrected fm_bit_idx (Right IOB) equations in Table 12 and fm_st_wd definition. Corrected CRC algorithm. Various corrections in Examples. Reformatted and edited document and figures. |
| 08/03/00 | 1.4 | Updated to include Virtex-E extended memory devices in Table 3 and Table 24. |
| 09/27/00 | 1.5 | Added Partial Reconfiguration of CLB section and Figure 26. |
| 03/24/03 | 1.6 | Updated to new template with expanded definitions for Lock in Table 25. |
| 10/20/04 | 1.7 | Updated links to data sheets. |