上海沪生电子产品使用说明书

http://www.husoon.com

地址：上海市北京东路 668 号上海赛格电子市场 2C48
电话：021－28304329 13918348251
传真：021－50693558（人工传真）
EMAIL:SALES@HUSOON.COM(销售)
SERVICE@HUSOON.COM

上海沪生电子产品使用说明书

http://www.husoon.com

电话：021-61021969

网站：www.husoon.com

This application note discusses how to configure one or more APEX II, APEX 20K (including APEX 20KE and APEX 20KC), Mercury, FLEX 10K (including FLEX 10KE and FLEX 10KA), and FLEX 6000 device. This application note should be used in conjunction with the following documents:

- *APEX II Programmable Logic Device Family Data Sheet*
- *APEX 20K Programmable Logic Device Family Data Sheet*
- *APEX 20KC Programmable Logic Device Data Sheet*
- *Mercury Programmable Logic Device Family Data Sheet*
- *ACEX 1K Programmable Logic Device Family Data Sheet*
- *FLEX 10K Embedded Programmable Logic Family Data Sheet*
- *FLEX 10KE Embedded Programmable Logic Family Data Sheet*
- *FLEX 6000 Programmable Logic Device Family Data Sheet*
- *Configuration Devices for APEX & FLEX Devices Data Sheet*
- *EPC16 Configuration Device Data Sheet*

If appropriate, illustrations in this application note show devices with generic "APEX 20K", "FLEX 10K", and "FLEX 6000" labels to indicate they are valid for all APEX 20K, FLEX 10K, and FLEX 6000 devices.

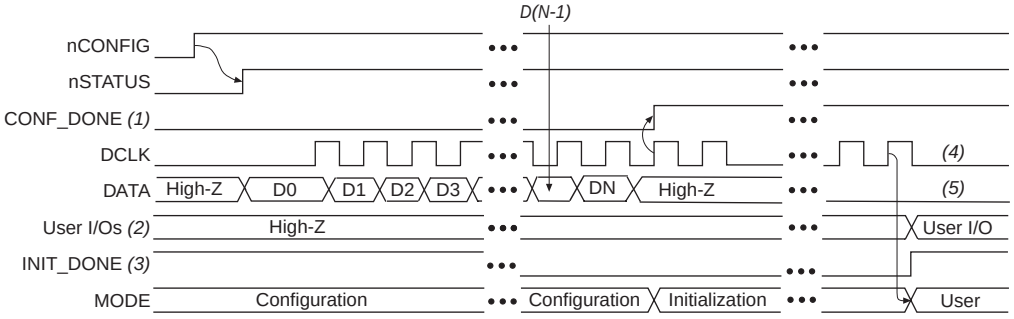
Contents

This application note provides information on the following topics:

Device Configuration Overview

During device operation, APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices store configuration data in SRAM cells. Because SRAM memory is volatile, the SRAM cells must be loaded with configuration data each time the device powers up. After the APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device is configured, its registers and I/O pins must be initialized. After initialization, the device enters user mode for in-system operation. Figure 1 shows the state of the device during the configuration, initialization, and user_mode.

Figure 1. APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K & FLEX 6000 Configuration Cycle



Notes to Figure 1:

(1) During initial power-up and configuration, CONF_DONE is low. After configuration, CONF_DONE goes high. If the device is reconfigured, CONF_DONE goes low after nCONFIG is driven low.

(2) User I/O pins are tri-stated during configuration. APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10KE devices have a weak pull-up resistor on I/O pins during configuration. After initialization, the user I/O pins perform the function assigned in the user's design.

(3) The optional INIT_DONE signal is high when nCONFIG is low before configuration and during approximately the first 40 clock cycles of configuration for APEX II and APEX 20K devices, the first 136 cycles for Mercury devices, or the first 10 cycles for ACEX 1K, FLEX 10K, and FLEX 6000 devices.

(4) DCLK should not be left floating after configuration. It should be driven high or low, whichever is more convenient.

(5) DATA (FLEX 6000 devices) and DATA0 (APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10KE devices) should not be left floating. They should be driven high or low, whichever is more convenient.

The configuration data for APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices can be loaded using an active or passive configuration scheme. When using an active configuration scheme with a configuration device, both the target device and configuration device generate the control and synchronization signals. When both devices are ready to begin configuration, the configuration device sends data to the APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device.

When using any passive configuration scheme, the APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device is incorporated into a system with an intelligent host, such as a microprocessor, that controls the configuration process. The host supplies configuration data from a storage device (e.g., a hard disk, RAM, or other system memory). When using passive configuration, you can change the target device's functionality while the system is in operation by reconfiguring it. You can also perform in-field upgrades by distributing a new programming file to system users.

The APEX II device's configuration scheme is selected by driving its MSEL0 and MSEL1 pins either high or low as shown in Table 2.

Table 2. APEX II Configuration Schemes *Note (1)*

MSEL1	MSEL0	Configuration Scheme
0	0	Configuration device or passive serial.
1	0	Fast passive parallel.
1	1	Passive parallel asynchronous.

Note to Table 2:

(1) The MSEL1 and MSEL0 pins can be used to change configuration modes between configurations. However, they are generally connected to V_{CC} or ground if the application only requires a single configuration mode.

You can select an APEX 20K, Mercury, ACEX 1K, or FLEX 10K device's configuration scheme by driving its MSEL0 and MSEL1 pins either high or low as shown in Table 3.

Table 3. APEX 20K, Mercury, ACEX 1K & FLEX 10K Configuration Schemes *Note (1)*

MSEL1	MSEL0	Configuration Scheme
0	0	Configuration device or passive serial.
1	0	Passive parallel synchronous.
1	1	Passive parallel asynchronous.

Note to Table 3:

(1) The MSEL1 and MSEL0 pins can be used to change configuration modes between configurations. However, they are generally connected to V_{CC} or ground if the application only requires a single configuration mode.

For FLEX 6000 devices, the MSEL pin controls configuration, as shown in Table 4.

Table 4. FLEX 6000 Configuration Schemes *Note (1)*

MSEL	Configuration Scheme
0	Configuration device or passive serial scheme, using the MasterBlaster or ByteBlasterMV cables.
1	Passive serial asynchronous.

Note to Table 4:

(1) You can use the MSEL pin to change configuration modes between configurations. However, it is generally connected to V_{CC} or ground.

Device option bits and device configuration pins are discussed further in "Device Options" on page 84 and "Device Configuration Pins" on page 88, respectively.

Table 5 summarizes the approximate configuration file size required for each APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 device. To calculate the amount of storage space required for multi-device configurations, simply add together the file size of each device.

Table 5. APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K & FLEX 6000 Configuration File Sizes (Part 1 of 2) *Note (1)*

Device	Data Size (Bits)	Data Size (Kbytes)
EP2A90	*(2)*	*(2)*
EP2A70	17,389,000	2,123
EP2A40	9,612,000	1,174
EP2A25	6,275,200	766
EP2A15	4,714,000	576
EP20K1500E	12,011,000	1,467
EP20K1000E, EP20K1000C	8,938,000	1,092
EP20K600E, EP20K600C	5,654,000	691
EP20K400	3,878,000	474
EP20K400E, EP20K400C	3,901,000	477
EP20K300E	2,733,000	334
EP20K200	1,950,000	239
EP20K200E, EP20K200C	1,964,000	240
EP20K160E	1,523,000	186
EP20K100	985,000	121
EP20K100E	1,009,000	124

Table 5. APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K & FLEX 6000 Configuration File Sizes (Part 2 of 2) *Note (1)*

Device	Data Size (Bits)	Data Size (Kbytes)
EP20K60E	641,000	79
EP20K30E	347,000	42
EP1M350	4,383,000	535
EP1M120	1,297,000	159
EP1K100	1,337,000	164
EP1K50	785,000	96
EP1K30	470,000	58
EP1K10	178,000	22
EPF10K250A	3,292,000	402
EPF10K200E	2,740,000	335
EPF10K130E	1,840,000	225
EPF10K130V	1,582,000	194
EPF10K100E	1,336,000	164
EPF10K100, EPF10K100A, EPF10K100B	1,200,000	147
EPF10K70	893,000	110
EPF10K50E	785,000	96
EPF10K50, EPF10K50V	621,000	76
EPF10K40	498,000	61
EPF10K30E	470,000	58
EPF10K30A	402,000	50
EPF10K30	376,000	46
EPF10K20	231,000	29
EPF10K10A	120,000	15
EPF10K10	118,000	15
EPF6024A	398,000	49
EPF6016, EPF6016A	260,000	32
EPF6010A	260,000	32

Notes to Table 5:

(1) Raw Binary Files (**.rbf**) were used to determine these file sizes.

(2) Contact Altera Applications for this information.

The numbers in Table 5 should only be used to estimate the file size before design compilation. The exact file size may vary because different Quartus II or MAX+PLUS II software versions may add a slightly different number of padding bits during programming. However, for any specific version of the Quartus II or MAX+PLUS II software, any design targeted for the same device has the same configuration file size.

Table 6 lists Altera configuration devices that can be used to configure APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices.

Table 6. Configuration Devices

Device	Description
EPC16	16,000,000 × 1-bit device with 3.3-V operation
EPC8	8,000,000 × 1-bit device with 3.3-V operation
EPC2	1,695,680 × 1-bit device with 5.0-V or 3.3-V operation
EPC1	1,046,496 × 1-bit device with 5.0-V or 3.3-V operation
EPC1441	440,800 × 1-bit device with 5.0-V or 3.3-V operation

You can use the data from Tables 5 and 6 to determine the number of configuration devices required to configure your device. For example, to configure one EPF10K100 device, you need two EPC1 devices, but only one EPC2 device. Similarly, one EP20K400 device requires three EPC2 devices, but only one EPC8 device.

Configuration Schemes

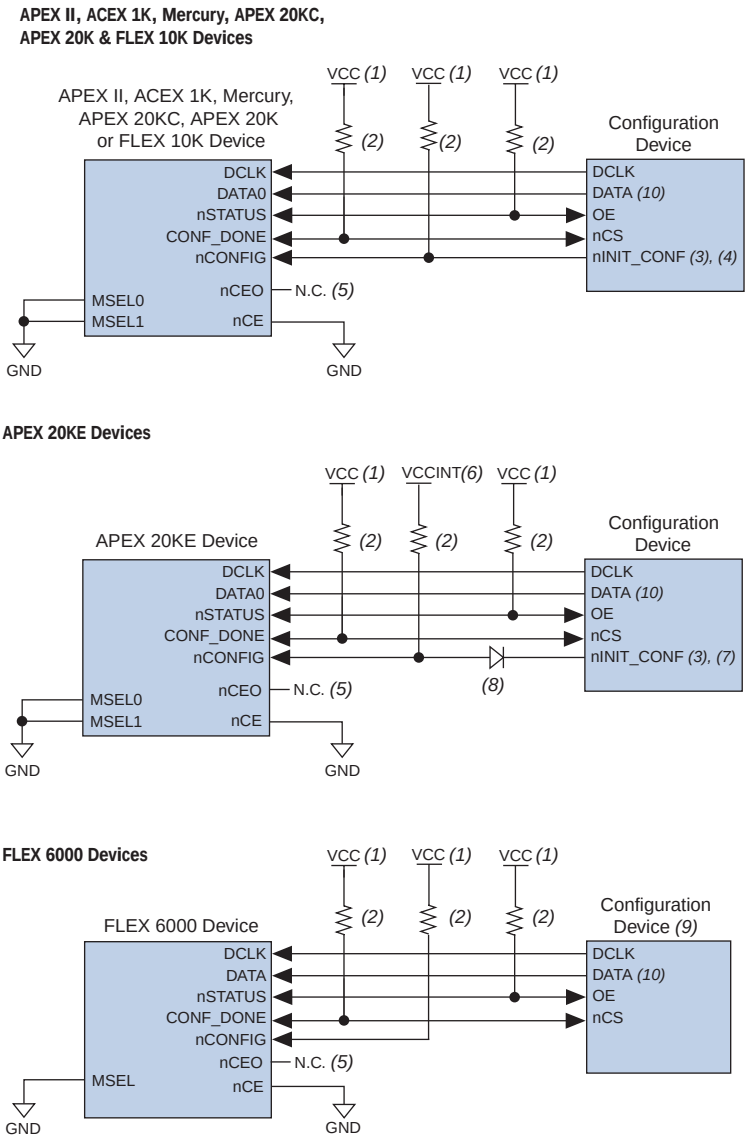
This section describes how to configure APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices with the following configuration schemes:

- Configuration Device
- PS Configuration with a Download Cable
- PS Configuration with a Microprocessor
- PPS Configuration (APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K Devices Only)
- PSA Configuration (FLEX 6000 Devices Only)
- PPA Configuration (APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K Devices Only)
- JTAG Programming and Configuration (APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K Devices Only)
- JTAG Programming and Configuration of Multiple Devices (APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K Devices Only)

Configuration Device

The configuration device scheme uses an Altera-supplied serial configuration device to supply data to the APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device in a serial bitstream. See Figure 2.

Figure 2. Configuration Device Scheme Circuit



Notes to Figure 2:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.
(2) All pull-up resistors are 1 kΩ (10 kΩ for APEX 20KE and APEX 20KC devices). The EPC16, EPC8, and EPC2 devices' OE and nCS pins have internal, user-configurable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the *Disable nCS and OE pull-up resistors on the configuration device* option when generating programming files.
(3) The nINIT_CONF pin is available on EPC2, EPC8, and EPC16 devices only. If nINIT_CONF is not available (i.e., on EPC1 devices) or not used, nCONFIG must be pulled to V_{CC} either directly or through a resistor.
(4) The nINIT_CONF pin has an internal pull-up resistor that is always active in EPC16, EPC8, and EPC2 devices. An external pull-up resistor is not required on the nINIT_CONF pin.
(5) The nCEO pin is left unconnected.
(6) To ensure successful configuration between APEX 20KE and configuration devices in all possible power-up sequences, pull up nCONFIG to V_{CCINT}.
(7) The nINIT_CONF pin has an internal pull-up resistor that is always active in EPC16, EPC8, and EPC2 devices. nCONFIG must be connected to V_{CCINT} through a 10-kΩ resistor.
(8) To isolate the 1.8-V and 3.3-V power supplies when configuring APEX 20KE devices, add a diode between the APEX 20KE device's nCONFIG pin and the configuration device's nINIT_CONF pin. Select a diode with a threshold voltage (V_T) less than or equal to 0.7 V. The diode will make the nINIT_CONF pin an open-drain pin; the pin will only be able to drive low or tri-state.
(9) EPC16, EPC8, and EPC2 devices should not be used to configure FLEX 6000 devices.
(10) The configuration device will drive DATA low after configuration in complete.

In the configuration device scheme, nCONFIG is usually tied to V_{CC} (when using EPC16, EPC8, or EPC2 devices, nCONFIG may be connected to nINIT_CONF). Upon device power-up, the target APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device senses the low-to-high transition on nCONFIG and initiates configuration. The target device then drives the open-drain CONF_DONE pin low, which in-turn drives the configuration device's nCS pin low. When exiting POR, both the target and configuration device release the open-drain nSTATUS pin.

Before configuration begins, the configuration device goes through a POR delay of 200 ms (maximum) to allow the power supply to stabilize; during this time, the configuration device drives its OE pin low. This low signal delays configuration because the OE pin is connected to the target device's nSTATUS pin. When both devices complete POR, they release nSTATUS, which is then pulled high by a pull-up resistor. When configuring multiple devices, configuration does not begin until all devices release their OE or nSTATUS pins. When all devices are ready, the configuration device clocks data out serially to the target devices using an internal oscillator.

After successful configuration, the configuration device starts clocking the target device for initialization. The CONF_DONE pin is released by the target device and then pulled high by a pull-up resistor. When initialization is complete, the configuration device enters user mode.

If an error occurs during configuration, the target device drives its nSTATUS pin low, resetting itself internally and resetting the configuration device. If the *Auto-Restart Configuration on Frame Error* option—available in the MAX+PLUS II **Global Project Device Options** dialog box (Assign menu)—is turned on, the device reconfigures automatically if an error occurs. The Quartus II software provides a similar option for using the **Device & Pin Option** dialog box. To choose this option, select **Compiler Settings** (Processing menu), then click on the **Chips & Devices** tab.

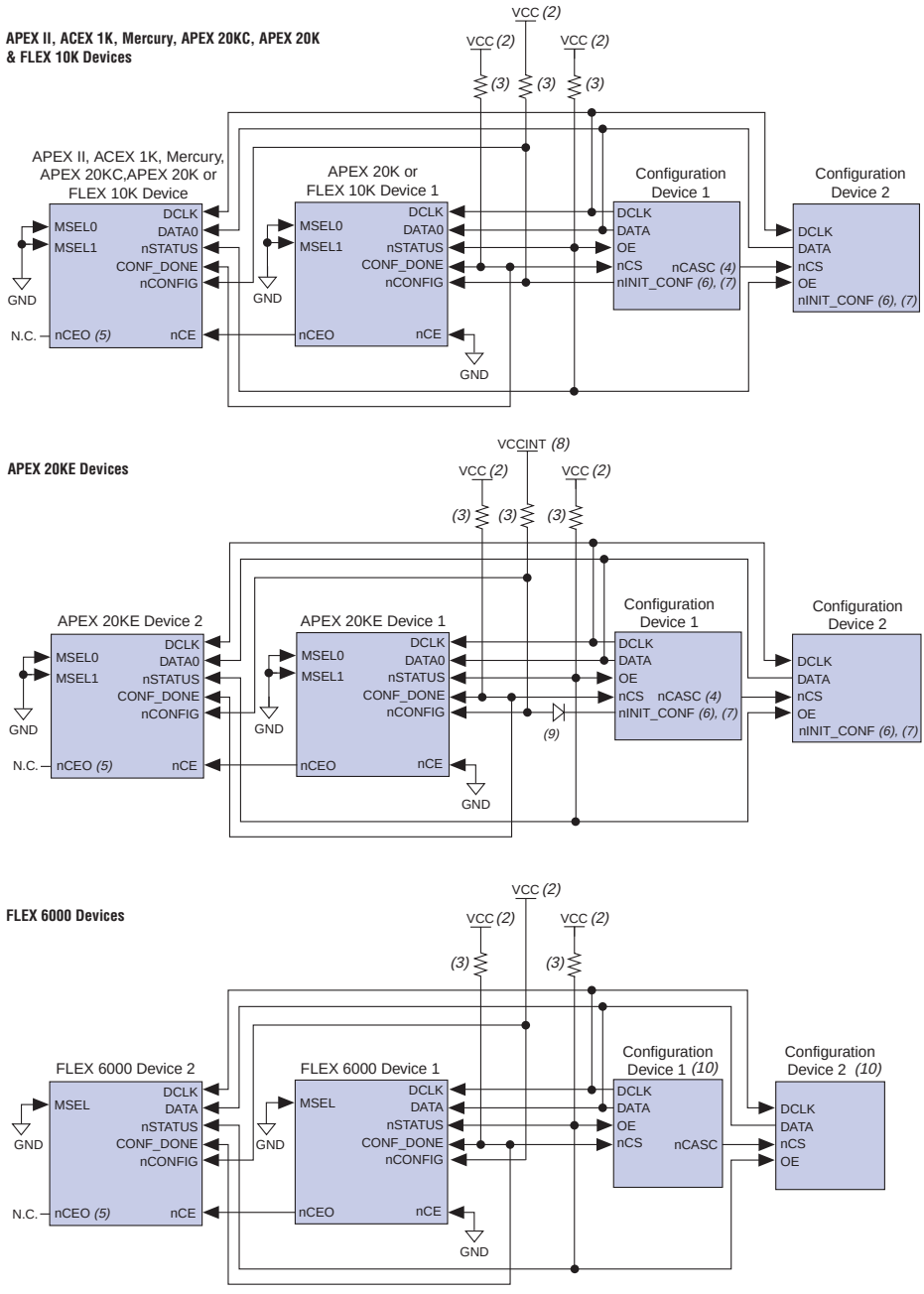
If this option is turned off, the external system must monitor nSTATUS for errors and then pulse nCONFIG low to restart configuration. The external system can pulse nCONFIG if nCONFIG is under system control rather than tied to V_{CC}. When configuration is complete, the target device releases CONF_DONE, which disables the configuration device by driving nCS high. The configuration device drives DCLK low before and after configuration.

In addition, if the configuration device sends all of its data and then detects that CONF_DONE has not gone high, it recognizes that the target device has not configured successfully. In this case, the configuration device pulses its OE pin low for a few microseconds, driving the target device's nSTATUS pin low. If the *Auto-Restart Configuration on Frame Error* option is set in the software, the target device resets and then pulses its nSTATUS pin low. When nSTATUS returns high, the configuration device reconfigures the target device. When configuration is complete, the configuration device drives DCLK low.

When CONF_DONE is driven low after device configuration, the configuration device recognizes that the target device has not configured successfully; therefore, your system should not pull CONF_DONE low to delay initialization. Instead, you should use the Quartus II or MAX+PLUS II software's User-Supplied Start-Up Clock option to synchronize the initialization of multiple devices that are not in the same configuration chain. Devices in the same configuration chain will initialize together. For more information on this option, see "Device Options" on page 84.

Figure 3 shows how to configure multiple devices with a configuration device. This circuit is similar to the configuration device circuit for a single programmable logic device (PLD), except the APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 devices are cascaded for multi-device configuration.

Figure 3. Multi-Device Configuration Circuit *Note (1)*



Notes to Figure 3:

(1) When performing multi-device active serial configuration, you must generate the configuration device's Programmer Object File (**.pof**) from each project's SRAM Object File (**.sof**). You can combine multiple SOFs using the MAX+PLUS II software's **Combine Programming Files** dialog box (File menu). For APEX 20K devices, the Quartus II software provides a similar option in the **Device & Pin Option** dialog box. To choose this option, select **Compiler Settings** (Processing menu), then click the **Chips & Devices** tab. For more information on how to create configuration and programming files, see "Device Configuration Files" on page 92.

(2) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(3) All pull-up resistors are 1 kΩ (10 kΩ for APEX 20KE and APEX 20KC devices). The EPC2 devices' OE and nCS pins have internal, user-configurable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the *Disable nCS and OE pull-up resistors on the configuration device* option when generating programming files.

(4) EPC16 and EPC8 configuration devices cannot be cascaded.

(5) The nCEO pin is left unconnected for the last device in the chain.

(6) The nINIT_CONF pin is available on EPC16, EPC8, and EPC2 devices only. If nINIT_CONF is not available (i.e., on EPC1 devices) or not used, nCONFIG must be pulled to V_{CC} through a resistor.

(7) The nINIT_CONF pin has an internal pull-up resistor that is always active in EPC16, EPC8, and EPC2 devices. These devices do not need an external pull-up resistor on the nINIT_CONF pin.

(8) To ensure successful configuration between APEX 20KE and configuration devices in all possible power-up sequences, pull up nCONFIG to V_{CCINT} through a 10-kΩ resistor.

(9) To isolate the 1.8-V and 3.3-V power supplies when configuring APEX 20KE devices, add a diode between the APEX 20KE device's nCONFIG pin and the configuration device's nINIT_CONF pin. Select a diode with a threshold voltage (V_T) less than or equal to 0.7 V. The diode will make the nINIT_CONF pin an open-drain pin; the pin will only be able to drive low or tri-state.

(10) The EPC16, EPC8, and EPC2 devices should not be used to configure FLEX 6000 devices. EPC1 devices should be used.

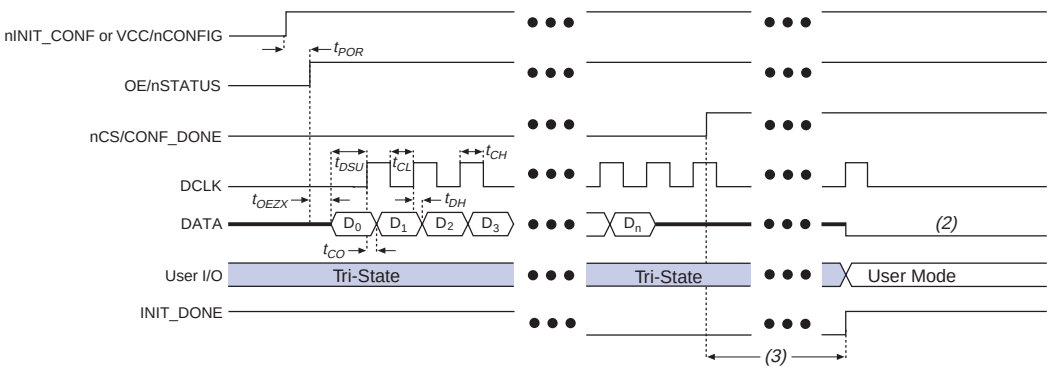
After the first device completes configuration during multi-device configuration, its nCEO pin activates the second device's nCE pin, prompting the second device to begin configuration. Because all device CONF_DONE pins are tied together, all devices initialize and enter user mode at the same time.

In addition, all nSTATUS pins are tied together; thus, if any device (including the configuration devices) detects an error, configuration stops for the entire chain. Also, if the first configuration device does not detect CONF_DONE going high at the end of configuration, it resets the chain by pulsing its OE pin low for a few microseconds. This low pulse drives the OE pin low on the second configuration device and drives nSTATUS low on all APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 devices, causing them to enter an error state. This state is similar to an APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device detecting an error.

If the *Auto-Restart Configuration on Frame Error* option is set in the software, the APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 devices release their nSTATUS pins after a reset time-out period. When the nSTATUS pins are released and pulled high, the configuration devices reconfigure the chain. If the *Auto-Restart Configuration on Frame Error* option is not set, the APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 devices drive nSTATUS low until they are reset with a low pulse on nCONFIG.

You can also cascade several configuration devices to configure multiple APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 devices. When all data from the first configuration device is sent, it drives nCASC low, which in turn drives nCS on the subsequent configuration device. Because a configuration device requires less than one clock cycle to activate a subsequent configuration device, the data stream is uninterrupted. EPC16 and EPC8 configuration devices cannot be cascaded. Figure 4 shows the timing waveform for the configuration device scheme.

Figure 4. Configuration Device Scheme Timing Waveform *Note (1)*



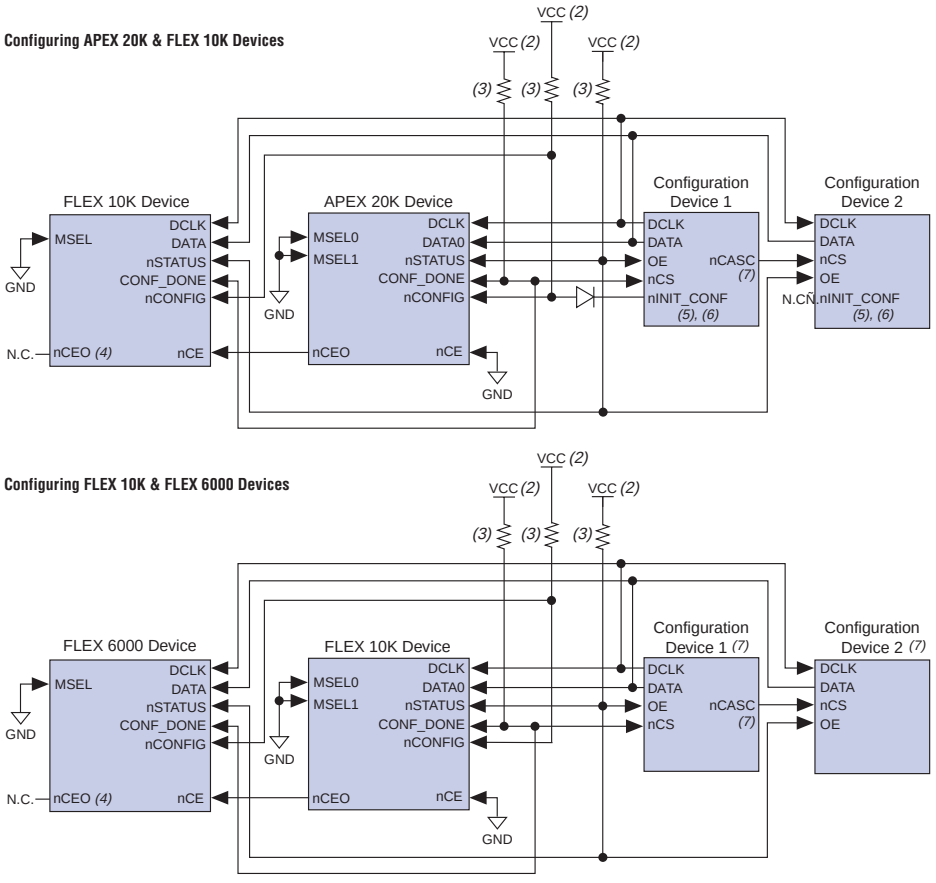
Notes to Figure 4:

(1) For timing information, refer to the *Configuration Devices for APEX & FLEX Devices Data Sheet* or the *EPC16 Configuration Device Data Sheet*.
(2) The configuration device will drive DATA low after configuration.
(3) APEX II and APEX 20K devices enter user mode 40 clock cycles after CONF_DONE goes high. Mercury devices enter user mode 136 clock cycles after CONF_DONE goes high. ACEX 1K, FLEX 10K, and FLEX 6000 devices enter user mode 10 clock cycles after CONF_DONE goes high.

You can use a single configuration chain to configure multiple APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices. In this scheme, the nCEO pin of the first device is connected to the nCE pin of the second device in the chain. To configure properly, all of the device CONF_DONE and nSTATUS pins must be tied together.

Figure 5 shows examples of configuring multiple APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices using a configuration device.

Figure 5. Multi-Device Configuration with APEX 20K, FLEX 10K & FLEX 6000 Devices *Note (1)*



Notes to Figure 5:

(1) For timing information, refer to the *Configuration Devices for APEX & FLEX Devices Data Sheet* or the *EPC16 Configuration Device Data Sheet*.

(2) VCC should be connected to the same supply voltage as the configuration device, except for APEX 20KE and APEX 20KC devices. For APEX 20KE and APEX 20KC devices, use a 10-kΩ resistor to pull nCONFIG up to V_{CCINT}.

(3) All pull-up resistors are 1 kΩ (10 kΩ for APEX 20KE and APEX 20KC devices). The EPC2 device's OE and nCS pins have internal, user-configurable 1-kΩ pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the *Disable nCS and OE pull-up resistors on the configuration device* option when generating programming files.

(4) The nCEO pin is left unconnected for the last device in the chain.

(5) The nINIT_CONF pin is available on EPC16, EPC8, and EPC2 devices only. If nINIT_CONF is not available (i.e., on EPC1 devices) or not used, nCONFIG must be pulled to V_{CC} either directly or through a resistor.

(6) The nINIT_CONF pin has an internal pull-up resistor which is always active in EPC16, EPC8, and EPC2 devices.

(7) Do not use EPC16, EPC8, and EPC2 devices to configure FLEX 6000 devices.

(8) EPC16 and EPC8 configuration devices cannot be cascaded.

For timing information, refer to the *Configuration Devices for APEX & FLEX Devices Data Sheet* or the *EPC16 Configuration Device Data Sheet*.

Table 7 shows the status of the device DATA pins during and after configuration. APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K devices have a DATA[7..0] bus, while FLEX 6000 devices have a DATA pin only.

Table 7. DATA Status During & After Configuration

Pins	APEX II, APEX 20K, Mercury, ACEX 1K or FLEX 10K Device		FLEX 6000 Device	
	During	After	During	After
DATA *(1)*	–	–	Used for configuration	Tri-state
DATA0 *(1)*	Used for configuration	Tri-state	–	–
DATA[7..1] *(2)*	Used in some configuration modes	User defined	–	–
I/O pins	Tri-state	User defined	Tri-state	User defined

Notes to Table 7:

(1) The status shown is for configuration with a configuration device.

(2) The function of these pins depends upon the settings specified in the MAX+PLUS II software's **Global Project Device Options** dialog box. For APEX 20K devices, the Quartus II software provides a similar option using the **Device & Pin Option** dialog box. To choose this option, select **Compiler Settings** (Processing Menu), then click the **Chips & Devices** tab. For more information, refer to MAX+PLUS II or Quartus II Help.

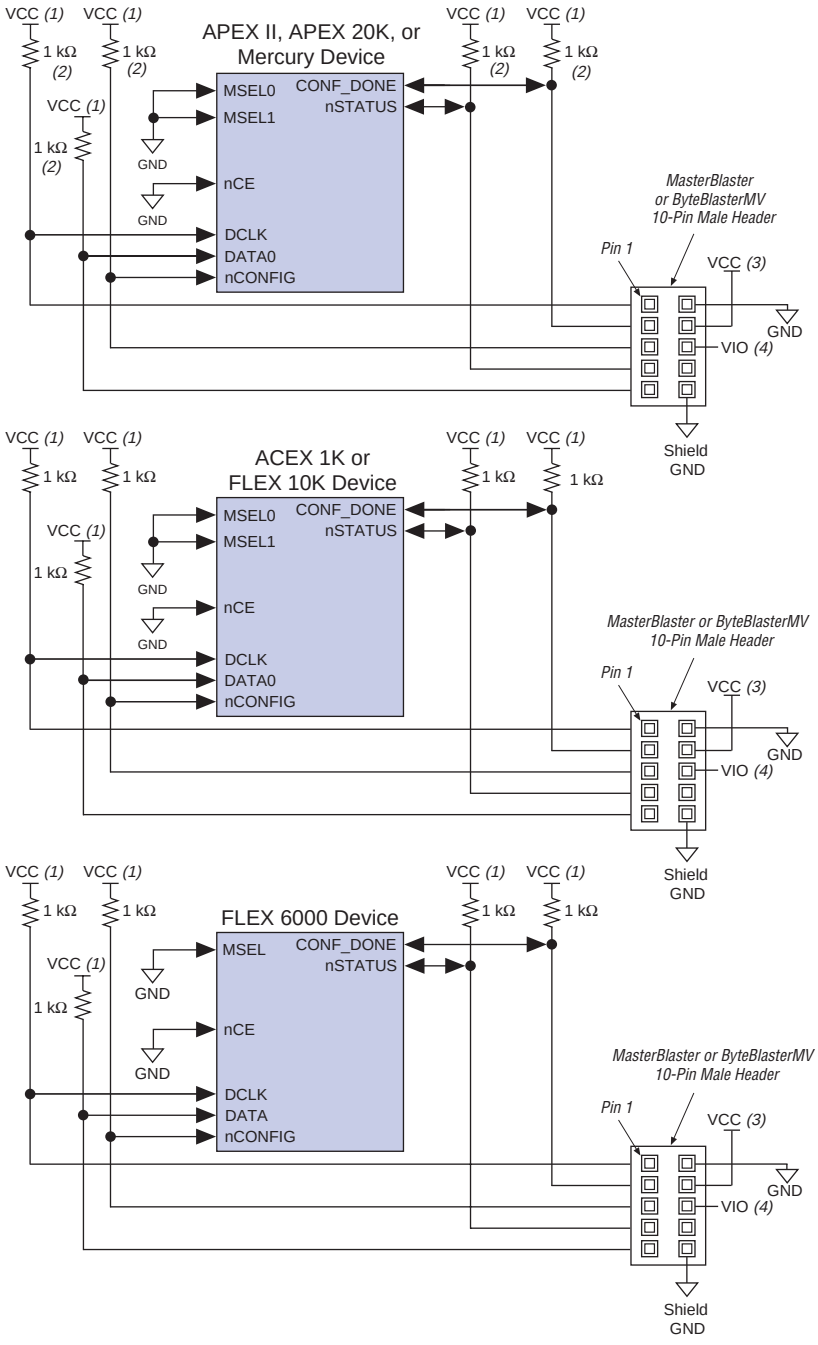
For information on how to create configuration and programming files for this configuration scheme, see "Device Configuration Files" on page 92.

PS Configuration with a Download Cable

In PS configuration with a download cable, an intelligent host transfers data from a storage device to the APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device via the MasterBlaster or ByteBlasterMV cable. To initiate configuration in this scheme, the download cable generates a low-to-high transition on the nCONFIG pin. The programming hardware then places the configuration data one bit at a time on the device's DATA pin (the DATA0 pin in APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K devices, and the DATA pin in FLEX 6000 devices). The data is clocked into the target device until CONF_DONE goes high.

When using programming hardware for APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000, setting the *Auto-Restart Configuration on Frame Error* option does not affect the configuration cycle because the Quartus II or MAX+PLUS II software must restart configuration when an error occurs. Figure 6 shows PS configuration for APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices using a MasterBlaster or ByteBlasterMV cable.

Figure 6. PS Configuration Circuit with MasterBlaster or ByteBlasterMV Cable



Notes to Figure 6:

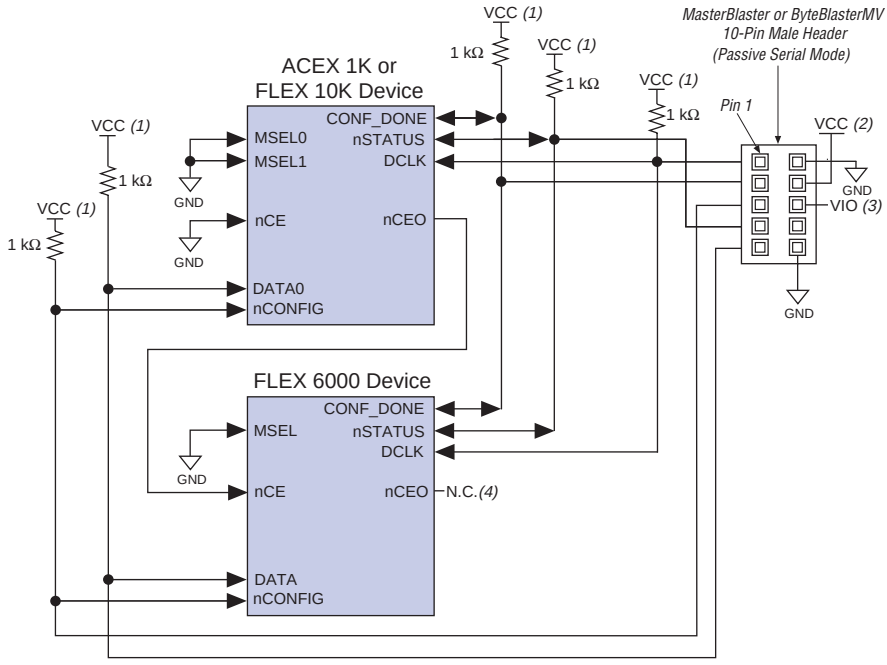
(1) The pull-up resistor should be connected to the same supply voltage as the MasterBlaster (VIO pin) or ByteBlasterMV cable, except for APEX 20KE and APEX 20KC devices. For APEX 20KE and APEX 20KC devices, use a 10-kΩ resistor to pull nCONFIG up to V_{CCINT}.
(2) The pull-up resistor should be 10 kΩ for APEX 20KE and APEX 20KC devices.
(3) The nCEO pin is left unconnected for the last device in the chain.
(4) Power supply voltage: V_{CC} = 3.3 V or 5.0 V for the MasterBlaster cable.
V_{CC} = 3.3 V or 5.0 V for the ByteBlasterMV cable.
(5) Pin 6 of the header is a V_{IO} reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. This pin is a no connect pin for the ByteBlasterMV header.

You can use programming hardware to configure multiple APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 devices by connecting each device's nCEO pin to the subsequent device's nCE pin. All other configuration pins are connected to each device in the chain. Because all CONF_DONE pins are tied together, all devices in the chain initialize and enter user mode at the same time.

In addition, because the nSTATUS pins are tied together, the entire chain halts configuration if any device detects an error. In this situation, the Quartus II or MAX+PLUS II software must restart configuration; the *Auto-Restart Configuration on Frame Error* option does not affect the configuration cycle.

Figure 7 shows how to configure multiple ACEX 1K, FLEX 10K, and FLEX 6000 devices with the MasterBlaster or ByteBlasterMV download cable.

Figure 7. PS Multi-Device Configuration for ACEX 1K, FLEX 10K & FLEX 6000 Devices with a Cable



Notes to Figure 7:

(1) The pull-up resistor should be connected to the same supply voltage as the MasterBlaster (VIO pin) or ByteBlasterMV cable.

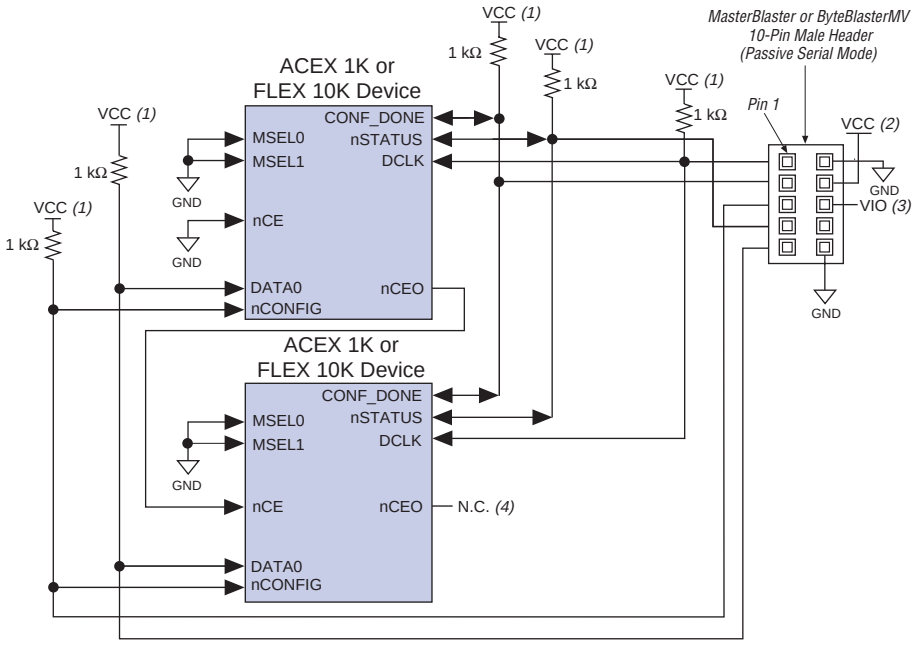
(2) Power supply voltage: V_{CC} = 3.3 V or 5.0 V for the MasterBlaster cable.
V_{CC} = 3.3 V or 5.0 V for the ByteBlasterMV cable.

(3) V_{IO} is a reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

(4) The nCEO pin is left unconnected for the last device in the chain.

Figure 8 shows how to configure multiple ACEX 1K and FLEX 10K devices with the MasterBlaster or ByteBlasterMV download cable.

Figure 8. PS Multi-Device Configuration for Multiple ACEX 1K or FLEX 10K Devices with a Cable



Notes to Figure 8:

(1) The pull-up resistor should be connected to the same supply voltage as the MasterBlaster (VIO pin) or ByteBlasterMV cable.

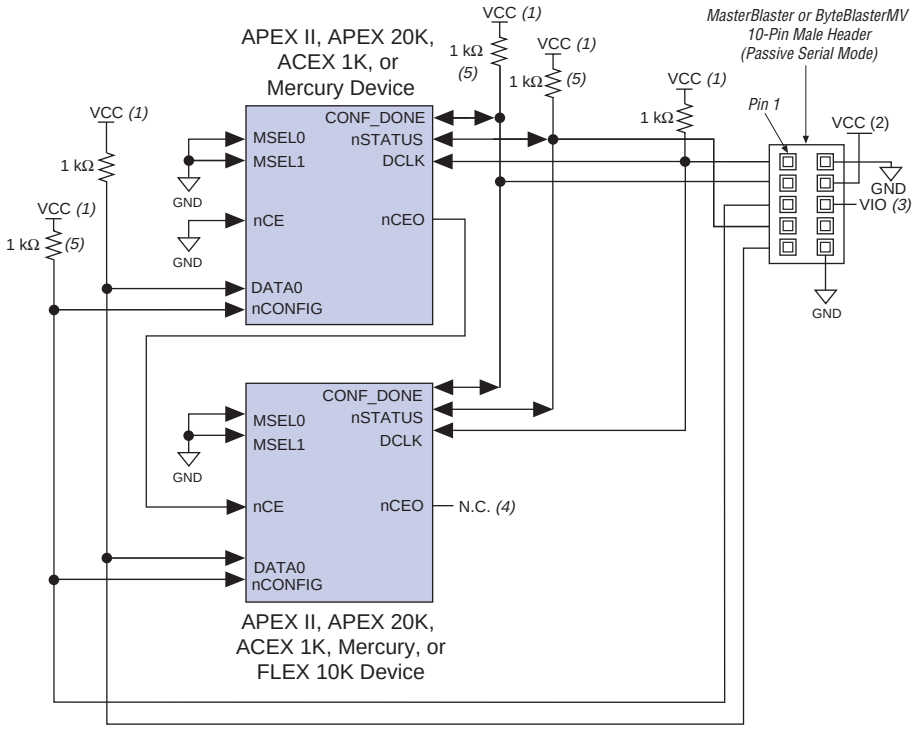
(2) Power supply voltage: V_{CC} = 3.3 V or 5.0 V for the MasterBlaster cable.
V_{CC} = 3.3 V or 5.0 V for the ByteBlasterMV cable.

(3) V_{IO} is a reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

(4) The nCEO pin is left unconnected for the last device in the chain.

Figure 9 shows how to configure multiple APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K devices with the MasterBlaster or ByteBlasterMV cable.

Figure 9. PS Multi-Device Configuration for APEX II, APEX 20K, Mercury, ACEX 1K & FLEX 10K Devices with a Cable



Notes to Figure 9:

(1) The pull-up resistor should be connected to the same supply voltage as the MasterBlaster (VIO pin) or ByteBlasterMV cable, except for APEX 20KE and APEX 20KC devices. For APEX 20KE and APEX 20KC devices, use a 10-kΩ resistor to pull nCONFIG up to V_{CCINT}.

(2) Power supply voltage: V_{CC} = 3.3 V or 5.0 V for the MasterBlaster cable.
V_{CC} = 3.3 V or 5.0 V for the ByteBlasterMV cable.

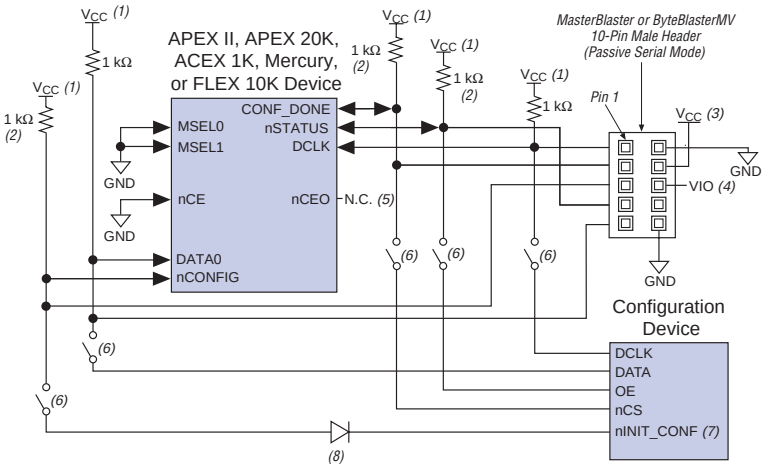
(3) V_{IO} is a reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

(4) The nCEO pin is left unconnected for the last device in the chain.

(5) The resistor value should be 10 kΩ for APEX 20KE and APEX 20KC devices. If there is a combination of APEX 20KE and APEX 20KC devices with other devices, use 10-kΩ pull-up resistors on nSTATUS and CONF_DONE pins.

If you are using a MasterBlaster or ByteBlasterMV cable to configure device(s) on a board that also has configuration devices, you should electrically isolate the configuration device from the target device(s) and cable. One way to isolate the configuration device is to add logic, such as a multiplexer, that can select between the configuration device and the cable. The multiplexer chip should allow bidirectional transfers on the nSTATUS and CONF_DONE signals. Another option is to add switches to the five common signals (i.e., CONF_DONE, nSTATUS, DCLK, nCONFIG, and DATA0) between the cable and the configuration device. The last option is to remove the configuration device from the board when configuring with the cable. Figure 10 shows a combination of a configuration device and a MasterBlaster or ByteBlasterMV cable to configure a PLD.

Figure 10. Configuring with a Combined PS & Configuration Device Scheme



Notes to Figure 10:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device except for APEX 20KE and APEX 20KC devices where nCONFIG should be pulled to V_{CCINT} by a 10-kΩ pull-up resistor.

(2) The resistor value should be 10 kΩ for APEX 20KE and APEX 20KC devices.

(3) Power supply voltage: V_{CC} = 3.3 V or 5.0 V for the MasterBlaster cable.
V_{CC} = 3.3 V or 5.0 V for the ByteBlasterMV cable.

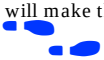
(4) Pin 6 of the header is a V_{IO} reference voltage for the MasterBlaster output driver. V_{IO} should match the target device's V_{CCIO}. This pin is a no connect pin for the ByteBlasterMV header.

(5) The nCEO pin is left unconnected.

(6) You should not attempt configuration with a MasterBlaster or ByteBlasterMV cable while a configuration device is connected to an APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device. Instead, you should either remove the configuration device from its socket when using the download cable or place a switch on the five common signals between the download cable and the configuration device.

(7) The nINIT_CONF pin is available on EPC16, EPC8, and EPC2 devices only. If nINIT_CONF is not available (i.e., on EPC1 devices) or not used, nCONFIG must be pulled to V_{CC} either directly or through a resistor.

(8) Add a diode between the APEX 20KE device's nCONFIG pin and the configuration device's nINIT_CONF pin to isolate the 1.8-V and 3.3-V power supplies. Select a diode with a threshold voltage (VT) less than or equal to 0.7 V. The diode will make the nINIT_CONF pin an open-drain pin. These pins will only break to drive low or tri-state.



For more information on how to use the MasterBlaster or ByteBlasterMV cables, see the following documents:

- *MasterBlaster Serial/USB Communications Cable Data Sheet*
- *ByteBlasterMV Parallel Port Download Cable Data Sheet*

For information on how to create configuration and programming files for this configuration scheme, see "Device Configuration Files" on page 92.

PS Configuration with a Microprocessor

In PS configuration with a microprocessor, a microprocessor transfers data from a storage device to the target APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device. To initiate configuration in this scheme, the microprocessor must generate a low-to-high transition on the nCONFIG pin and the target device must release nSTATUS. The microprocessor or programming hardware then places the configuration data one bit at a time on the DATA pin of the target device (the DATA0 pin for APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K devices, and the DATA pin for FLEX 6000 devices). The least significant bit (LSB) of each data byte must be presented first. Data is clocked continuously into the target device until CONF_DONE goes high.

After all data is transferred, DCLK must be clocked an additional 10 times for ACEX 1K, FLEX 10K, and FLEX 6000 devices, an additional 40 times for APEX II and APEX 20K devices, or an additional 136 times for Mercury devices to initialize the device. The device's CONF_DONE pin goes high to show successful configuration and to start initialization. The configuration files created by the Quartus II or MAX+PLUS II software incorporate extra bits for initialization. Driving DCLK to the device after configuration does not affect device operation. Therefore, sending the entire configuration file to the device is sufficient to configure and initialize it.

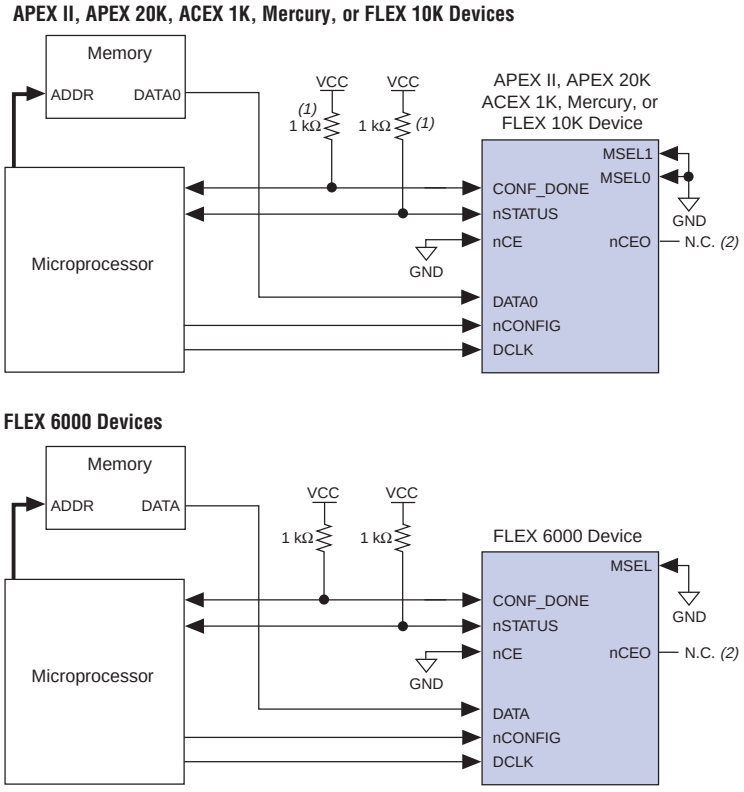
Handshaking signals are not used in PS configuration modes. Therefore, the configuration clock speed must be below the specified frequency to ensure correct configuration. No maximum DCLK period exists. You can pause configuration by halting DCLK for an indefinite amount of time.

If the target device detects an error during configuration, it drives its nSTATUS pin low to alert the microprocessor. The microprocessor can then pulse nCONFIG low to restart the configuration process. Alternatively, if the *Auto-Restart Configuration on Frame Error* option is set in the Quartus II or MAX+PLUS II software, the target device releases nSTATUS after a reset time-out period. After nSTATUS is released, the microprocessor can reconfigure the target device without needing to pulse nCONFIG low.

The microprocessor can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. If the microprocessor sends all data and the initialization clock starts but CONF_DONE & INIT_DONE have not gone high, it must reconfigure the target device.

Figure 11 shows the circuit for PS configuration with a microprocessor.

Figure 11. PS Configuration Circuit with Microprocessor



Notes to Figure 11:

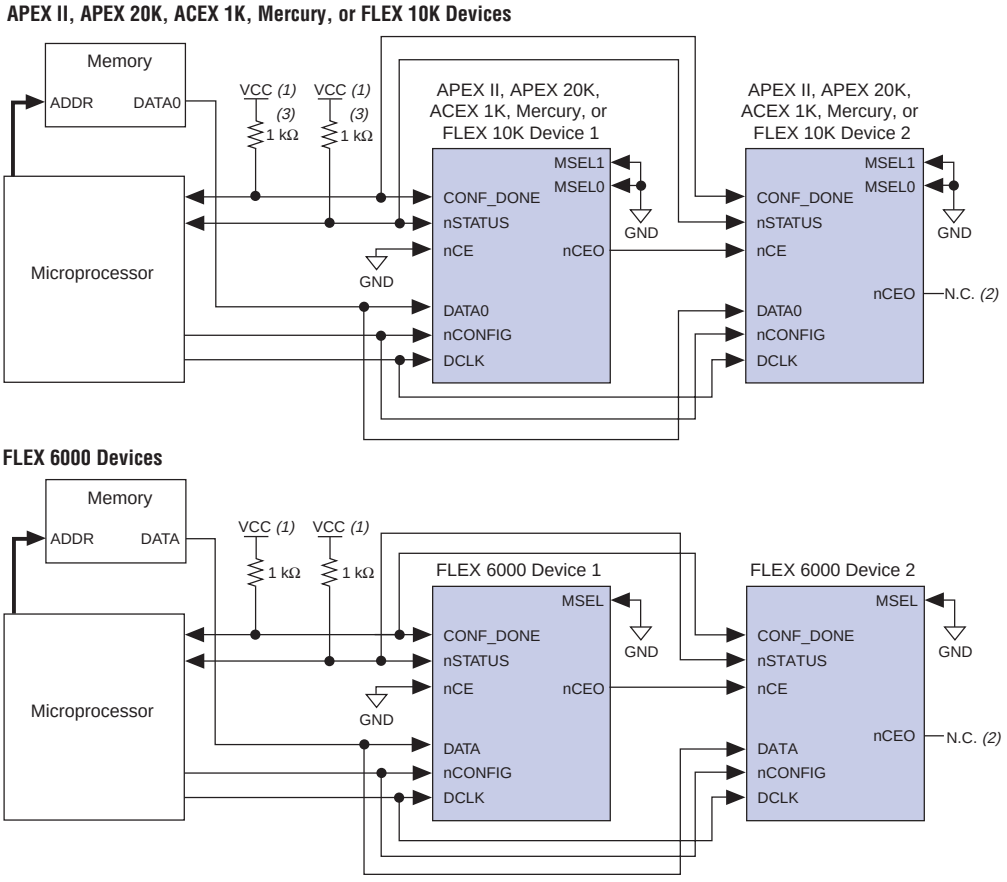
(1) The pull-up resistor should be 10 kΩ for APEX 20KE and APEX 20KC devices.

(2) The nCEO pin is left unconnected for single device configuration.

For multi-device PS configuration with a microprocessor, the first device's nCEO pin is cascaded to the second device's nCE pin. The second device in the chain begins configuration within one clock cycle; therefore, the transfer of data destinations is transparent to the microprocessor. Because all device CONF_DONE pins are tied together, all devices initialize and enter user mode at the same time.

In addition, the nSTATUS pins are tied together. Thus, if any device detects an error, the entire chain halts configuration and drives nSTATUS low. The microprocessor can then pulse nCONFIG low to restart the configuration process. Alternatively, if the *Auto-Restart Configuration on Frame Error* option is set in the Quartus II or MAX+PLUS II software, the target devices release nSTATUS after a reset time-out period. After nSTATUS is released, the microprocessor can reconfigure the target devices. Figure 12 shows multi-device configuration with a microprocessor.

Figure 12. PS Multi-Device Configuration with a Microprocessor



Notes to Figure 12:

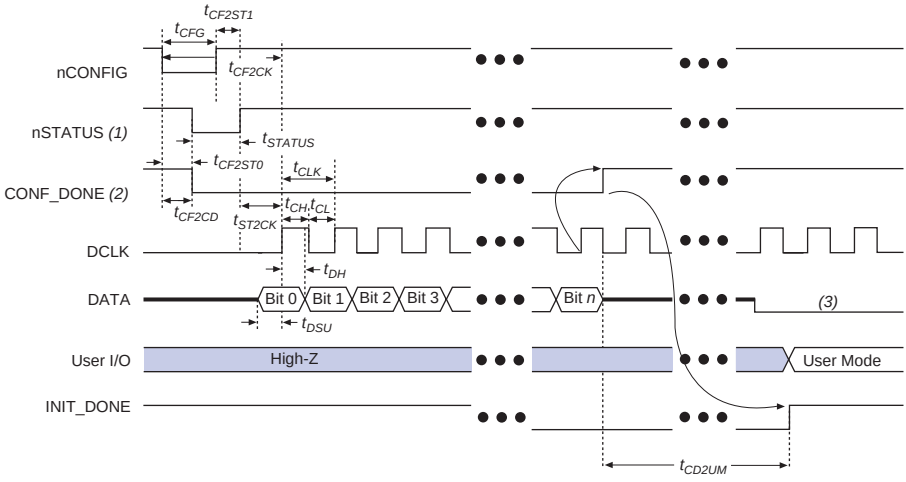
(1) The pull-up resistor should be connected to a supply that provides an acceptable input signal for all devices in the chain. For example, when a device chain contains a mixture of 5.0-V FLEX 10K devices and 2.5-V FLEX 10KE devices, the pull-up resistor should be connected to 5.0 V. You should use 5.0 V in this scenario because FLEX 10KE I/O pins are 5.0-V tolerant.

(2) The nCEO pin is left unconnected for the last device in the chain.

(3) The pull up resistor should be 10 kΩ for APEX 20KE and APEX 20KC devices. If there is a combination of APEX 20KE or APEX 20KC devices along with other devices, use a 10-kΩ pull-up resistor on the nSTATUS and CONF_DONE pins.

Figure 13 shows the PS configuration timing waveform for APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices.

Figure 13. PS Timing Waveform for APEX 20K, FLEX 10K & FLEX 6000 Devices



Notes to Figure 13:

(1) Upon power-up, the APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device holds nSTATUS low for not more than 5 µs after V_{CC} reaches its minimum requirement.
(2) Upon power-up and before configuration, CONF_DONE is low.
(3) DATA should not be left floating after configuration. It should be driven high or low, whichever is more convenient.

Tables 8 and 9 contain timing information for APEX II and APEX 20K devices.

Table 8. PS Timing Parameters for APEX 20K Devices

Symbol	Parameter	Min	Max	Units
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		1 *(3)*	µs
t_{CFG}	nCONFIG low pulse width *(1)*	8		µs
t_{STATUS}	nSTATUS low pulse width	10	40	µs
t_{CF2CK}	nCONFIG low to first rising edge on DCLK	40		µs
t_{ST2CK}	nSTATUS high to first rising edge on DCLK	1		µs
t_{DSU}	Data setup time before rising edge on DCLK	10		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH}	DCLK high time	15		ns
t_{CL}	DCLK low time	15		ns
t_{CLK}	DCLK period	30		ns
f_{MAX}	DCLK maximum frequency		33.3	MHz
t_{CD2UM}	CONF_DONE high to user mode *(2)*	2	8	µs

Table 9. PS Timing Parameters for APEX II, APEX 20KE & APEX 20KC Devices *Note (4)*

Symbol	Parameter	Min	Max	Units
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		1 *(3)*	µs
t_{CFG}	nCONFIG low pulse width *(1)*	8		µs
t_{STATUS}	nSTATUS low pulse width	10	40	µs
t_{CF2CK}	nCONFIG low to first rising edge on DCLK	40		µs
t_{ST2CK}	nSTATUS high to first rising edge on DCLK	1		µs
t_{DSU}	Data setup time before rising edge on DCLK	10		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH}	DCLK high time	8.75		ns
t_{CL}	DCLK low time	8.75		ns
t_{CLK}	DCLK period	17.5		ns
f_{MAX}	DCLK maximum frequency		57	MHz
t_{CD2UM}	CONF_DONE high to user mode *(2)*	2	8	µs

Tables 10 and 11 contain timing information for Mercury and ACEX 1K devices.

Table 10. PS Timing Parameters for Mercury Devices *Note (4)*

Symbol	Parameter	Min	Max	Units
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		1 *(3)*	µs
t_{CFG}	nCONFIG low pulse width *(1)*	21		µs
t_{STATUS}	nSTATUS low pulse width	10	40	µs
t_{CF2CK}	nCONFIG low to first rising edge on DCLK	45		µs
t_{ST2CK}	nSTATUS high to first rising edge on DCLK	1		µs
t_{DSU}	Data setup time before rising edge on DCLK	10		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH}	DCLK high time	10		ns
t_{CL}	DCLK low time	10		ns
t_{CLK}	DCLK period	20		ns
f_{MAX}	DCLK maximum frequency		50	MHz
t_{CD2UM}	CONF_DONE high to user mode *(2)*	6	28	µs

Table 11. PS Timing Parameters for ACEX 1K Devices

Symbol	Parameter	Min	Max	Units
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		4 *(3)*	µs
t_{CFG}	nCONFIG low pulse width *(1)*	2		µs
t_{STATUS}	nSTATUS low pulse width	1		µs
t_{CF2CK}	nCONFIG low to first rising edge on DCLK	5		µs
t_{ST2CK}	nSTATUS high to first rising edge on DCLK	1		µs
t_{DSU}	Data setup time before rising edge on DCLK	10		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH}	DCLK high time	15		ns
t_{CL}	DCLK low time	15		ns
t_{CLK}	DCLK period	30		ns
f_{MAX}	DCLK maximum frequency		33.3	MHz
t_{CD2UM}	CONF_DONE high to user mode *(2)*	0.6	2	µs

Tables 12 and 13 contain timing information for FLEX 10KE, FLEX 10K, and FLEX 6000 devices.

Table 12. PS Timing Parameters for FLEX 10KE Devices

Symbol	Parameter	Min	Max	Units
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		4	µs
t_{CFG}	nCONFIG low pulse width *(1)*	2		µs
t_{STATUS}	nSTATUS low pulse width	1		µs
t_{CF2CK}	nCONFIG low to first rising edge on DCLK	5		µs
t_{ST2CK}	nSTATUS high to first rising edge on DCLK	1		µs
t_{DSU}	Data setup time before rising edge on DCLK	10		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH}	DCLK high time	15		ns
t_{CL}	DCLK low time	15		ns
t_{CLK}	DCLK period	30		ns
f_{MAX}	DCLK maximum frequency		33.3	MHz
t_{CD2UM}	CONF_DONE high to user mode *(2)*	0.6	2	µs

Table 13. PS Timing Parameters for FLEX 10K & FLEX 6000 Devices

Symbol	Parameter	Min	Max	Units
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		4	µs
t_{CFG}	nCONFIG low pulse width *(1)*	2		µs
t_{STATUS}	nSTATUS low pulse width	1		µs
t_{CF2CK}	nCONFIG low to first rising edge on DCLK	5		µs
t_{ST2CK}	nSTATUS high to first rising edge on DCLK	1		µs
t_{DSU}	Data setup time before rising edge on DCLK	10		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH}	DCLK high time	30		ns
t_{CL}	DCLK low time	30		ns
t_{CLK}	DCLK period	60		ns
f_{MAX}	DCLK maximum frequency		16.7	MHz
t_{CD2UM}	CONF_DONE high to user mode *(2)*	0.6	2	µs

Notes to Tables 8 – 13:

(1) If configuration is stopped and reinitiated before CONF_DONE goes high, this value applies when the internal oscillator is selected as the clock source. If configuration is stopped and reinitiated before CONF_DONE goes high and the clock source is CLKUSR or DCLK, multiply the clock period by 40 for APEX II or APEX 20K devices, 136 for Mercury devices, or 10 for FLEX 10K and FLEX 6000 devices to determine this value.

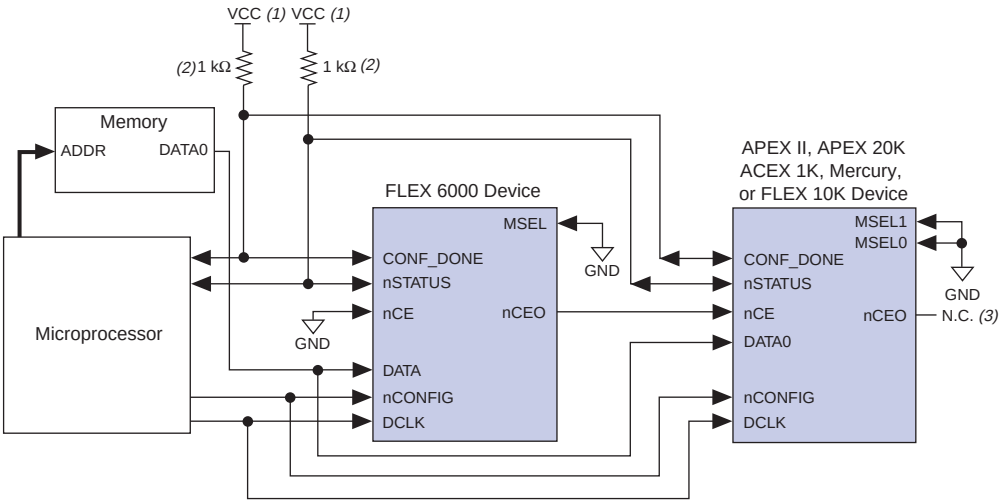
(2) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 40 for APEX II and APEX 20K devices, 136 for Mercury devices, or 10 for ACEX 1K, FLEX 10K, and FLEX 6000 devices to obtain this value.

(3) This value is obtainable if users do not delay configuration by extending the nSTATUS low pulse width.

(4) APEX II, APEX 20KC, and Mercury device timing parameters are preliminary.

Figure 14 shows an example of PS multi-device configuration circuit with APEX II, APEX 20K, ACEX 1K, Mercury, FLEX 10K, and FLEX 6000 devices using a microprocessor.

Figure 14. PS Multi-Device Configuration of APEX II, APEX 20K, ACEX 1K, Mercury, FLEX 10K & FLEX 6000 Devices with a Microprocessor



Notes to Figure 14:

(1) The pull-up resistor should be connected to a supply that provides an acceptable input signal for all devices in the chain. For example, when a device chain contains a mixture of 5.0-V FLEX 10K devices and 2.5-V FLEX 10KE devices, the pull-up resistor should be connected to 5.0 V. You should use 5.0 V in this scenario because FLEX 10KE I/O pins are 5.0-V tolerant.

(2) The pull-up resistor should be 10 kΩ for APEX 20KE and APEX 20KC devices. If there is a combination of APEX 20KE or APEX 20KC devices with other devices, use a 10-kΩ pull-up resistor on nSTATUS and CONF_DONE pins.

(3) The nCEO pin is left unconnected for the last device in the chain.

For information on how to create configuration and programming files for this configuration scheme, see "Device Configuration Files" on page 92.

PPS Configuration (APEX 20K, Mercury, ACEX 1K & FLEX 10K Devices Only)

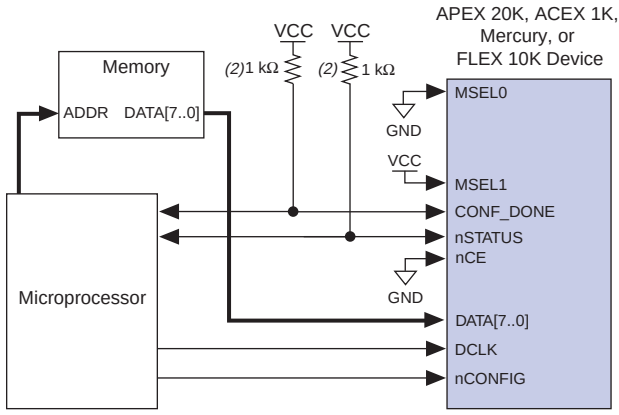
In a passive parallel synchronous (PPS) configuration scheme, an intelligent host drives the target APEX 20K, Mercury, ACEX 1K, or FLEX 10K device. The host system outputs parallel data and the serializing clock to the device. The target device latches the byte-wide data on the DATA[7..0] pins, and serializes it internally.

The DCLK, CONF_DONE, nCONFIG, nSTATUS, and DATA[7..0] pins are connected to a port on the intelligent host, such as a microprocessor. To begin configuration, nCONFIG is given a low-to-high transition and the host places an 8-bit configuration word on the target device's data inputs. The host clocks the target device; new data should be presented by the host and latched by the target device every eight clock cycles.

On the first rising clock edge, a byte of configuration data is latched into the target device; the subsequent eight falling clock edges serialize the data in the device. On the ninth rising clock edge, the next byte of configuration data is latched and serialized into the target device. You can pause configuration by halting DCLK. DCLK may be halted for an indefinite amount of time. A status pin (RDYnBSY) on the target device indicates when it is serializing data and when it is ready to accept the next data byte. If an error occurs during configuration, the nSTATUS pin drives low. The host senses this low signal and begins reconfiguration or issues an error.

Once the target device configures successfully, it releases the CONF_DONE pin. When CONF_DONE goes high, it indicates that configuration is complete. After the last data byte, the DCLK pin must be clocked 40 times for APEX 20K devices, 136 times for Mercury devices, and 10 times for FLEX 10K devices to release CONF_DONE and initialize the device. See Figure 15.

Figure 15. PPS Configuration Circuit *Note (1)*



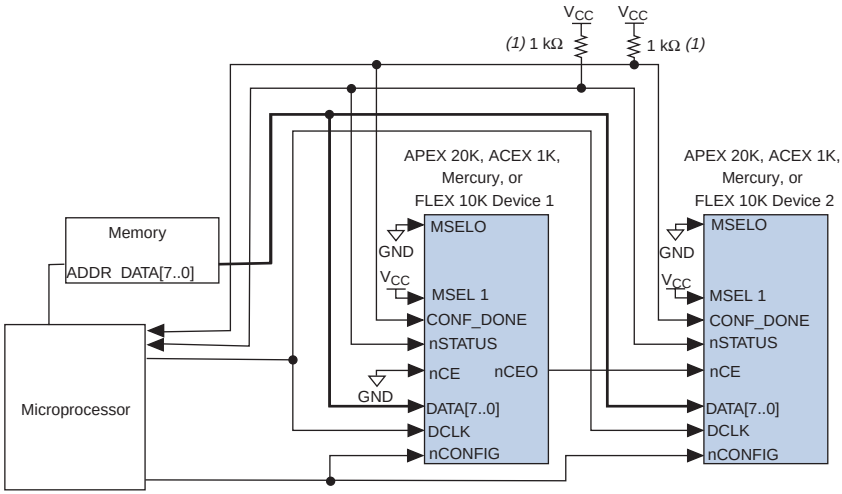
Notes to Figure 15:

(1) For APEX 20K, Mercury, ACEX 1K, or FLEX 10K devices, the configuration word can be in Tabular Text File (**.ttf**), Raw Binary File (**.rbf**), or Hexadecimal (**.hex**) format. For information on how to create configuration and programming files, see "Device Configuration Files" on page 92.

(2) The pull-up resistor should be 10 kΩ for APEX 20KE and APEX 20KC devices.

You can configure multiple APEX 20K, Mercury, ACEX 1K, or FLEX 10K devices in PPS mode by cascading the devices. Once the first device is configured, it drives its nCEO pin low, driving the second device's nCE pin low. The second device begins configuration within one clock cycle. Because all device CONF_DONE pins are tied together, all devices initialize and enter user mode at the same time. In addition, all nSTATUS pins are tied together; thus, if any device detects an error, the entire chain is reset for automatic reconfiguration. See Figure 16.

Figure 16. PPS Multi-Device Configuration Circuit

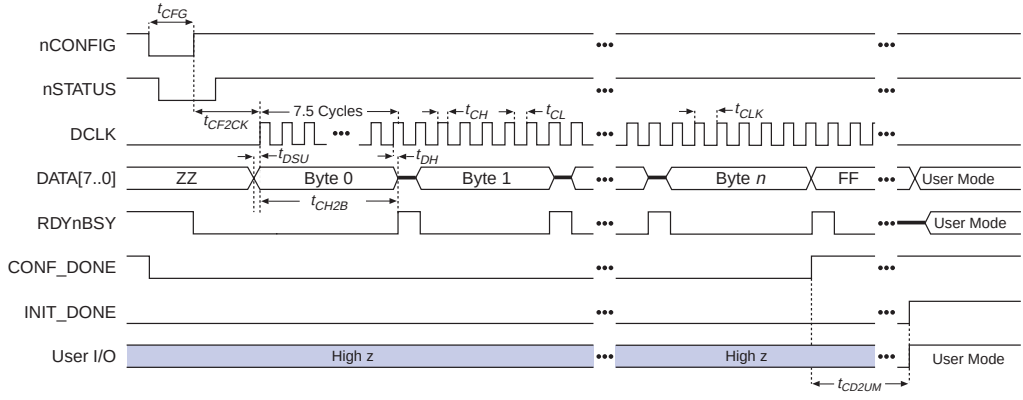


Note to Figure 16:

(1) The pull-up resistor should be 10 kΩ for APEX 20KE and APEX 20KC devices. If APEX 20KE or APEX 20KC devices are combined with other devices, use a 10-kΩ pull-up resistor on nSTATUS and CONF_DONE pins.

Figure 17 shows the timing waveforms for PPS configuration for APEX 20K, Mercury, ACEX 1K, and FLEX 10K devices.

Figure 17. PPS Timing Waveform for APEX 20K, Mercury, ACEX 1K & FLEX 10K Devices



Tables 14 and 17 define the timing parameters for PPS configuration in APEX 20K, Mercury, ACEX 1K, and FLEX 10K devices.

Table 14. PPS Timing Parameters for APEX 20K Devices Note (1)				
Symbol	**Parameter**	**Min**	**Max**	**Units**
t_{CF2CK}	nCONFIG low to first rising edge on DCLK	40		µs
t_{DSU}	Data setup time before rising edge on DCLK	10		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH2B}	First rising DCLK to first rising RDYnBSY *(2)*	0.75		µs
t_{CFG}	nCONFIG low pulse width *(3)*	8		µs
t_{CH}	DCLK high time	30		ns
t_{CL}	DCLK low time	30		ns
t_{CLK}	DCLK period	60		ns
f_{MAX}	DCLK frequency		16.7	MHz
t_{CD2UM}	CONF_DONE high to user mode *(4)*	2	8	µs
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		1 *(5)*	µs
t_{Status}	nSTATUS low pulse width	10	40	µs
t_{ST2CK}	nSTATUS high to first rising edge of DCLK	1		µs

Notes to Table 14:

(1) This information is preliminary.

(2) This parameter depends on the DCLK frequency. The RDYnBSY signal goes high 7.5 clock cycles after the rising edge of DCLK. This value was calculated with a DCLK frequency of 10 MHz.

(3) This value applies only if the internal oscillator is selected as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 40 to obtain this value.

(4) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 40 to obtain this value.

(5) This value is obtainable if users do not delay configuration by extending the nSTATUS low pulse width.

Table 15. PPS Timing Parameters for Mercury Devices *Note (1)*

Symbol	Parameter	Min	Max	Units
t_{CF2CK}	nCONFIG low to first rising edge on DCLK	40		µs
t_{DSU}	Data setup time before rising edge on DCLK	10		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH2B}	First rising DCLK to first rising RDYnBSY *(2)*	0.75		µs
t_{CFG}	nCONFIG low pulse width *(3)*	8		µs
t_{CH}	DCLK high time	30		ns
t_{CL}	DCLK low time	30		ns
t_{CLK}	DCLK period	60		ns
f_{MAX}	DCLK frequency		16.7	MHz
t_{CD2UM}	CONF_DONE high to user mode *(4)*	2	8	µs
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		1 *(5)*	µs
t_{Status}	nSTATUS low pulse width	10	40	µs
t_{ST2CK}	nSTATUS high to first rising edge of DCLK	1		µs

Notes to Table 15:

(1) This information is preliminary.

(2) This parameter depends on the DCLK frequency. The RDYnBSY signal goes high 7.5 clock cycles after the rising edge of DCLK. This value was calculated with a DCLK frequency of 10 MHz.

(3) This value applies only if the internal oscillator is selected as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 136 to obtain this value.

(4) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 136 to obtain this value.

(5) This value is obtainable if users do not delay configuration by extending the nSTATUS low pulse width.

Table 16. PPS Timing Parameters for ACEX 1K Devices

Symbol	Parameter	Min	Max	Units
t_{CF2CK}	nCONFIG low to first rising edge on DCLK	40		µs
t_{DSU}	Data setup time before rising edge on DCLK	10		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH2B}	First rising DCLK to first rising RDYnBSY *(1)*	0.75		µs
t_{CFG}	nCONFIG low pulse width *(2)*	8		µs
t_{CH}	DCLK high time	30		ns
t_{CL}	DCLK low time	30		ns
t_{CLK}	DCLK period	60		ns
f_{MAX}	DCLK frequency		16.7	MHz
t_{CD2UM}	CONF_DONE high to user mode *(3)*	2	8	µs
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		1	µs
t_{Status}	nSTATUS low pulse width	10	40	µs
t_{ST2CK}	nSTATUS high to first rising edge of DCLK	1		µs

Notes to Table 16:

(1) This parameter depends on the DCLK frequency. The RDYnBSY signal goes high 7.5 clock cycles after the rising edge of DCLK. This value was calculated with a DCLK frequency of 10 MHz.

(2) This value applies only if the internal oscillator is selected as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 10 to obtain this value.

(3) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 10 to obtain this value.

Table 17. PPS Timing Parameters for FLEX 10K Devices

Symbol	Parameter	Min	Max	Units
t_{CF2CK}	nCONFIG low to first rising edge on DCLK	5		µs
t_{DSU}	Data setup time before rising edge on DCLK	10		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH2B}	First rising DCLK to first rising RDYnBSY *(1)*	0.75		µs
t_{CFG}	nCONFIG low pulse width *(2)*	2		µs
t_{CH}	DCLK high time	30		ns
t_{CL}	DCLK low time	30		ns
t_{CLK}	DCLK period	60		ns
f_{MAX}	DCLK frequency		66	MHz
t_{CD2UM}	CONF_DONE high to user mode *(3)*	0.6	2	µs
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		4	µs
t_{Status}	nSTATUS low pulse width	1		µs
t_{ST2CK}	nSTATUS high to first rising edge on DCLK	1		µs

Notes to Table 17:

(1) This parameter depends on the DCLK frequency. The RDYnBSY signal goes high 7.5 clock cycles after the rising edge of DCLK. This value was calculated with a DCLK frequency of 10 MHz.

(2) This value applies only if the internal oscillator is selected as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 10 to obtain this value.

(3) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 10 to obtain this number.

For information on how to create configuration and programming files for this configuration scheme, see "Device Configuration Files" on page 92.

Fast Passive Parallel Configuration with APEX II Devices

Parallel configuration in APEX II devices is designed to meet the continuously increasing demand for faster configuration times. APEX II devices are designed with the capability of receiving byte-wide configuration data per clock cycle, and guarantee a configuration time of less than 100 ms with a 66-MHz configuration clock. Designers can take advantage of the parallel configuration feature using a microprocessor, an EPC8, or an EPC16 device.

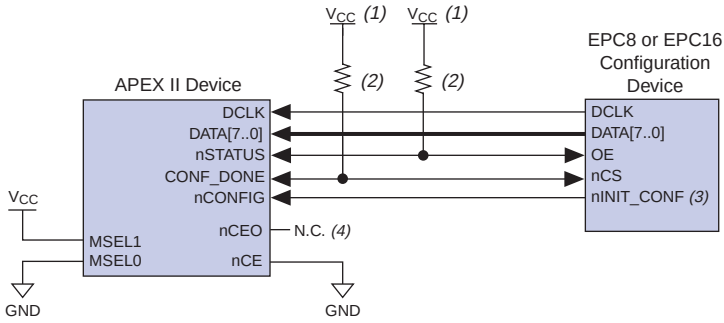
This section discusses the following schemes for parallel configuration in APEX II devices:

- Parallel configuration using a configuration device
- Parallel configuration using a microprocessor

Parallel Configuration Using a Configuration Device

The configuration device scheme in a parallel configuration uses either an EPC8 or an EPC16 device to supply data in a byte-wide fashion to the APEX II device. See Figure 18.

Figure 18. APEX II Fast Passive Parallel Configuration Using Configuration Device



Notes to Figure 18:

(1) The pull-up resistors should be connected to the same supply voltage as the EPC16 or the EPC8 device.

(2) All pull-up resistors are 1 kΩ. The OE, nCS, and nINIT_CONF pins on the EPC16 or EPC8 devices have user-configurable internal pull-up resistors. The internal pull-up resistor on nINIT_CONF pin is always active.The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the *Disable nCS and OE pull-up resistors on the configuration device* option when generating programming files.

(3) If nINIT_CONF is not used, nCONFIG must be pulled to V_{CC} either directly or through a 1-kΩ resistor.

(4) The nCEO pin is left unconnected for the last device on the chain.

In the configuration device scheme, nCONFIG is tied to nINIT_CONF. On power up, the target APEX II device senses the low-to-high transition on nCONFIG and initiates configuration. The target APEX II device then drives the open-drain CONF_DONE pin low, which in-turn drives the EPC8 or EPC16 device's nCS pin low.

Before configuration starts, there is a POR delay of 2 ms if the PORSEL pin is connected to V_{CC} in the EPC16 or the EPC8 device. If the PORSEL pin is connected to GND, the POR delay is 100 ms. When each device determines that its power is stable, it will release its nSTATUS or OE pin. Since the EPC16 or EPC8 device's OE pin is connected to the target APEX II device's nSTATUS pin, configuration is delayed until both the nSTATUS and OE pins are released by each device, when each devices the signal will be pulled up by the resistor. When configuring multiple devices, connect the nSTATUS pins together to ensure configuration only happens when all devices release their OE or nSTATUS pins. The EPC16 or the EPC8 device then clocks data out in parallel to the APEX II device using a 66-MHz internal oscillator.

If there is an error during configuration, the APEX II device drives the nSTATUS pin low, resetting itself internally and resetting the configuration device. The Quartus II software provides an Auto-restart configuration after error option that automatically initiates the reconfiguration whenever an error occurs. Use the following instructions to turn this option on or off:

1. Go to **Compiler Settings** (Processing menu)

2. Choose the **Chips & Devices** tab

3. Click **Device & Pin Options**

4. In the **General** tab, check or uncheck the Auto-restart configuration after error option

5. Click OK twice

If this option is turned off, nSTATUS must be monitored to check for errors. To initiate reconfiguration, pulse nCONFIG low. The external system can pulse nCONFIG if it is under system control rather than tied to V_{CC}. Therefore, nCONFIG must be connected to nINIT_CONF if you need to reprogram the APEX II device on the fly.

When configuration is complete, the APEX II device releases the CONF_DONE pin, which is then pulled up by a resistor. This disables the EPC16 or the EPC8 device as nCS is driven high. When initialization is complete, the APEX II device enters user mode. The EPC16 or the EPC8 device drives DCLK low before and after configuration.

If, after sending out all of its data, the EPC8 or the EPC16 device does not detect CONF_DONE going high, it recognizes that the APEX II device has not configured successfully. The EPC8 or EPC16 device pulses its OE pin low for a few microseconds, driving the nSTATUS pin on APEX II device low. If the Auto-restart configuration after error option is on, the APEX II device resets and then pulses its nSTATUS low. When nSTATUS returns high, reconfiguration is restarted.

Do not drive CONF_DONE low after device configuration to delay initialization. Instead, use the Enable user-supplied start-up clock option in the same Quartus II dialog box as the Auto-restart configuration after error option. Figure 19 shows the dialog box to enable or disable this option. This option can be used to synchronize the initialization of multiple devices that are not in the same configuration chain. Devices in the same configuration chain will initialize together.

Figure 19. Configuration Options Dialog Box

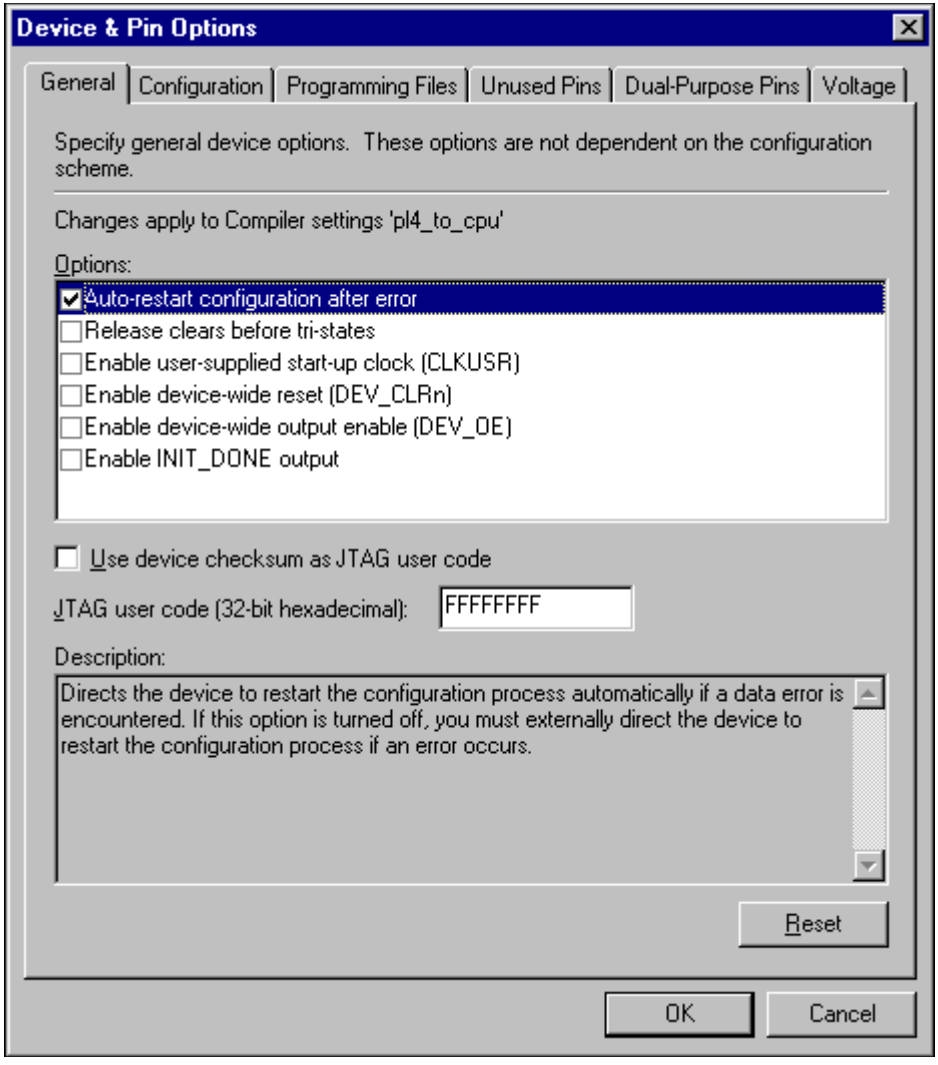
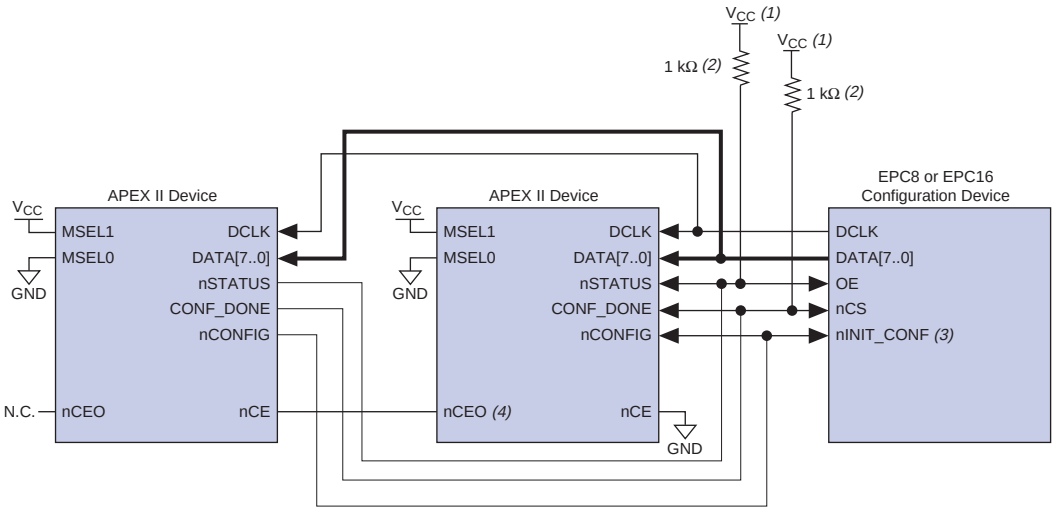


Figure 20 shows the configuration of multiple APEX II devices with an EPC16 or EPC8 device. This circuit is similar to Figure 18 (configuring a single APEX II device with an EPC16 or EPC8 configuration device), except that the APEX II devices are cascaded for multi-device configuration. The EPC16 or EPC8 device itself cannot be cascaded.

Figure 20. Fast Passive Parallel Configuration with Multiple APEX II Devices & EPC8 or EPC16 Configuration Device



Notes to Figure 20:

(1) The pull-up resistors should be connected to the same supply voltage as the EPC16 or the EPC8 device.

(2) All pull-up resistors are 1 kΩ. The OE, nCS, and nINIT_CONF pins on the EPC16 or the EPC8 devices have internal pull-up resistors. The internal pull-up resistor on nINIT_CONF pin is always active. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the *Disable nCS and OE pull-up resistors on the configuration device* option when generating programming files.

(3) If nINIT_CONF is not used, nCONFIG must be pulled to V_{CC} either directly or through a 1-kΩ resistor.

(4) The nCEO pin is left unconnected for the last device on the chain.

After the first APEX II device completes configuration during multi-device configuration, its nCEO pin activates the second APEX II device's nCE pin, prompting the second device to begin configuration. All devices initialize and enter user mode at the same time since CONF_DONE pins are tied together. Since nSTATUS are tied together, configuration stops for the whole chain if any device (including the EPC16 or the EPC8) detects an error. Also, if the EPC16 or the EPC8 device does not detect a high on CONF_DONE at the end of configuration, it will pulse its OE low for a few microseconds to reset the chain. The low OE pulse will drive nSTATUS low on all APEX II devices, causing them to enter an error state. This state is similar to an APEX II device detecting an error.

If the Auto-restart configuration after error option is on, the APEX II devices release their nSTATUS pins after a reset time-out period. When the nSTATUS pins are released and pulled high, the configuration devices reconfigure the chain. If the Auto-restart configuration after error option is not on, nSTATUS will stay low until the APEX II devices are reset with a low pulse on nCONFIG.

Parallel Configuration with a Microprocessor

When using a microprocessor for parallel configuration, the microprocessor transfers data from a storage device to the APEX II device via configuration hardware. To initiate configuration, the microprocessor needs to generate a low-to-high transition on the nCONFIG pin and the APEX II device must release nSTATUS. The microprocessor then places the configuration data to the DATA[7:0] pins of the APEX II device 8 bits at a time. Data is clocked continuously into the APEX device until CONF_DONE goes high.

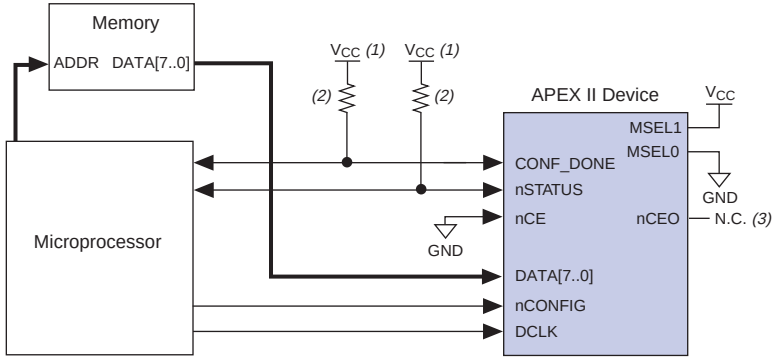
After all data is transferred, DCLK must be clocked an additional 40 times for the APEX II device to initialize. The APEX II device's CONF_DONE pin goes high to show successful configuration and to start initialization. The configuration files created by the Quartus II software incorporate extra bits for initialization and driving DCLK to the device after configuration. These extra bits do not affect device operation. Therefore, sending the entire configuration file to the device is sufficient to configure and initialize it. Configuration can be paused by halting DCLK. Although there is no maximum DCLK period, the configuration clock speed should be below 66 MHz to ensure correct configuration.

If the APEX II device detects an error during configuration, it drives nSTATUS low to alert the microprocessor. The microprocessor can then pulse nCONFIG low to restart configuration error. With the Auto-restart configuration after error option on, the APEX II device releases nSTATUS after a reset time-out period. After nSTATUS is released, the microprocessor can reconfigure the APEX II device without pulsing nCONFIG low.

The microprocessor can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. If the microprocessor sends all the data and the initialization clock starts, but CONF_DONE and INIT_DONE have not gone high, it must reconfigure the APEX II device.

Figure 21 shows the circuit for parallel configuration of an APEX II device using a microprocessor.

Figure 21. APEX II Parallel Configuration Using a Microprocessor



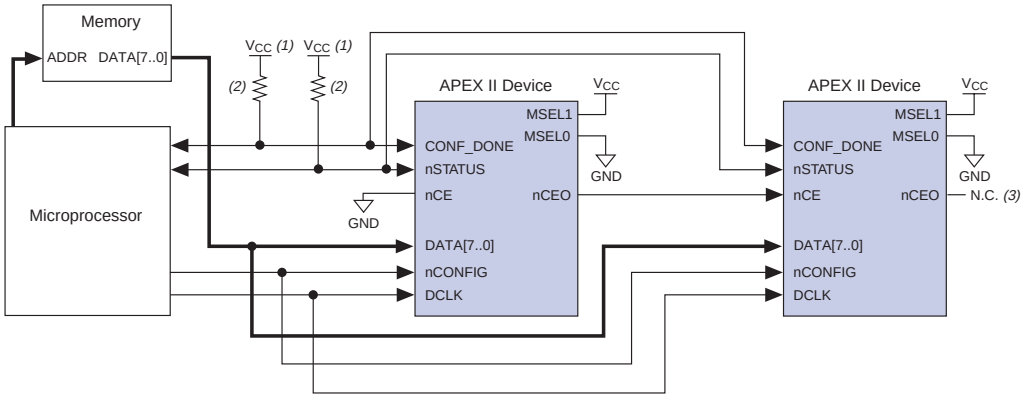
Notes to Figure 21:

(1) The pull-up resistors should be connected to any V_{CC} that meets the APEX II high-level input voltage (V_{IH}) specification.
(2) All pull-up resistors are 1 kΩ.
(3) The nCEO pin is left unconnected.

For multi-device parallel configuration with a microprocessor, the nCEO pin of the first APEX II device is cascaded to the second device's nCE pin. The second device in the chain begins configuration within one clock cycle; therefore, the transfer of data destinations is transparent to the microprocessor. Because the CONF_DONE pins of the devices are connected together, all devices initialize and enter user mode at the same time.

Since the nSTATUS pins are also tied together, if any of the devices detects an error, the entire chain halts configuration and drives nSTATUS low. The microprocessor can then pulse nCONFIG low to restart configuration. If the Auto-restart configuration after error option is on in the Quartus II software, the APEX II devices release nSTATUS after a reset time-out period. The microprocessor can then reconfigure the devices once nSTATUS is released. Figure 22 shows multi-device configuration using a microprocessor. Figure 23 shows multi-device configuration when both APEX devices are getting the same data.

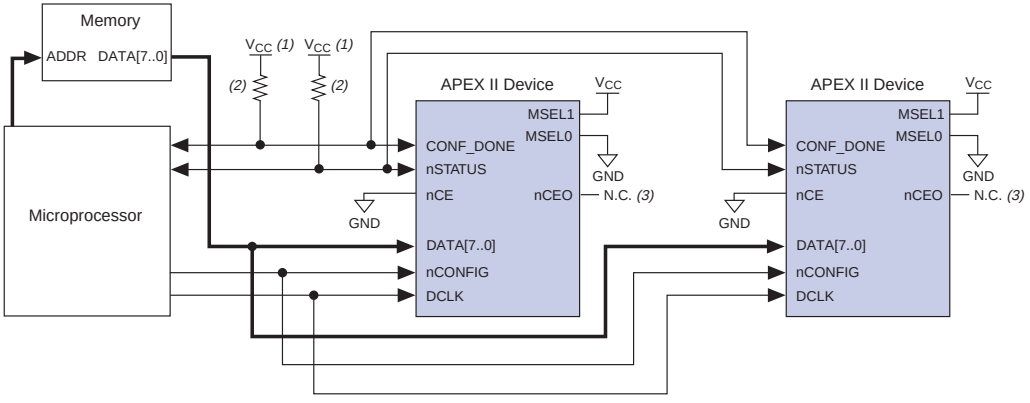
Figure 22. Multiple APEX II Device Parallel Data Transfer in a Serial Configuration with a Microprocessor



Notes to Figure 22:

(1) The pull-up resistors should be connected to any V_{CC} that meets the APEX II high-level input voltage (V_{IH}) specification.
(2) All pull up resistors are 1 kΩ.
(3) The nCEO pin of the last device is left unconnected.

Figure 23. Multiple APEX II Device Parallel Configuration with the Same Data Using a Microprocessor



Notes to Figure 23:

(1) The pull-up resistors should be connected to any V_{CC} that meets the APEX II high-level input voltage (V_{IH}) specification.
(2) All pull up resistors are 1 kΩ.
(3) The nCEO pins are left unconnected when configuring the same data into multiple APEX II devices.

Figure 24 shows the FPP timing waveform for APEX II devices. Table 18 shows the FPP timing parameters for APEX II devices.

Figure 24. FPP Timing Waveform for APEX II Devices

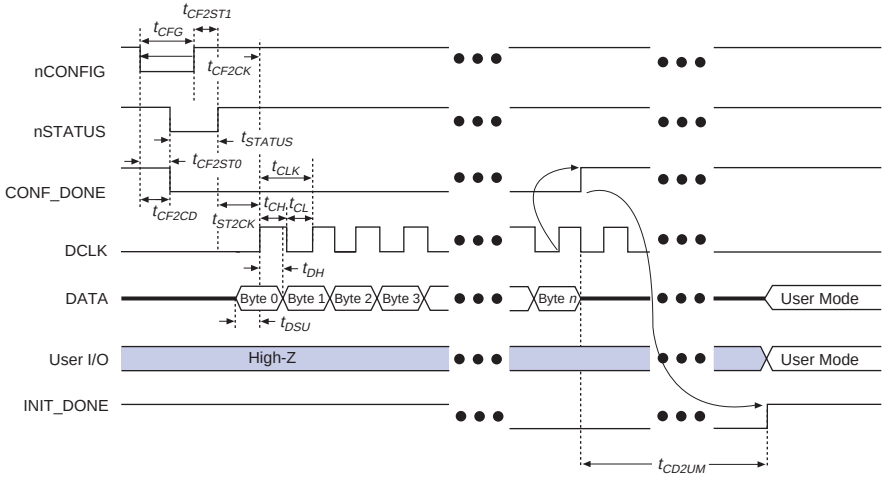


Table 18. FPP Timing Parameters for APEX II Devices *Note (1)*

Symbol	Parameter	Min	Max	Units
t_{CF2CK}	nCONFIG low to first rising edge on DCLK	40		µs
t_{DSU}	Data setup time before rising edge on DCLK	10		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH2B}	First rising DCLK to first rising RDYnBSY	0.75		µs
t_{CFG}	nCONFIG low pulse width *(2)*	8		µs
t_{CH}	DCLK high time	30		ns
t_{CL}	DCLK low time	30		ns
t_{CLK}	DCLK period	60		ns
f_{MAX}	DCLK frequency		16.7	MHz
t_{CD2UM}	CONF_DONE high to user mode *(3)*	2	8	µs
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		1 *(4)*	µs
t_{Status}	nSTATUS low pulse width	10	40	µs
t_{ST2CK}	nSTATUS high to first rising edge of DCLK	1		µs

Notes to Table 18:

(1) This information is preliminary.

(2) This value applies only if the internal oscillator is selected as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 40 to obtain this value.

(3) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 40 to obtain this value.

(4) This value is obtainable if users do not delay configuration by extending the nSTATUS low pulse width.

PSA Configuration (FLEX 6000 Devices Only)

In passive serial asynchronous (PSA) configuration, a microprocessor drives data to a FLEX 6000 device. When in PSA mode, you should pull the DCLK pin high using a 1-kΩ pull-up resistor to prevent unused configuration pins from floating.

To begin configuration, the microprocessor drives nCONFIG high and then pulls the FLEX 6000 device's nCS pin low and CS pin high. The microprocessor places a configuration bit on the FLEX 6000 device's DATA input and pulses nWS low to write data to the FLEX 6000 device. On the next rising edge of nWS, the FLEX 6000 device latches a bit of configuration data. Next, the FLEX 6000 device drives the RDYnBSY signal low, indicating that it is processing the configuration data. The microprocessor can then perform other system functions while the FLEX 6000 device is processing the data bit.

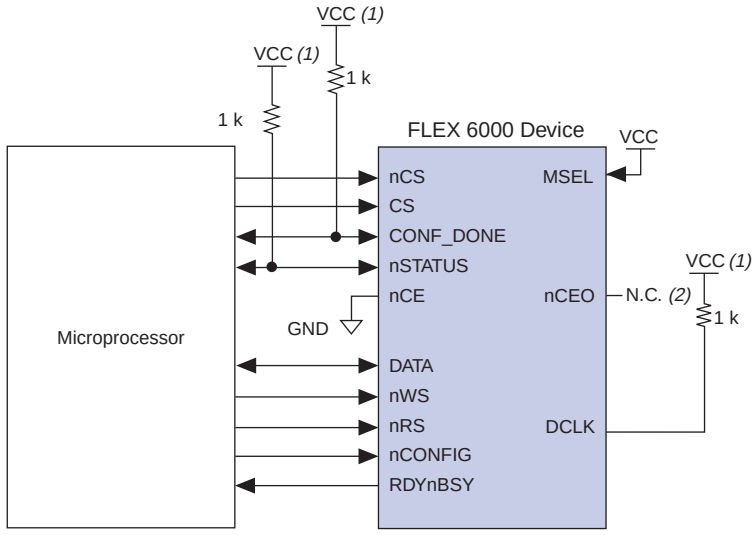
Afterward, the microprocessor checks nSTATUS and CONF_DONE. If the device asserts nSTATUS low, it has encountered an error and the microprocessor should restart configuration. If nSTATUS is not low and all configuration data has been received, the FLEX 6000 device is ready for initialization. At the beginning of initialization, CONF_DONE goes high to indicate that configuration is complete. If both nSTATUS and CONF_DONE are not low, the microprocessor sends the next data bit.

The microprocessor can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. If the microprocessor has sent all configuration data and has started initialization but CONF_DONE is not high, the microprocessor must reconfigure the FLEX 6000 device.

The MAX+PLUS II- or Quartus II-generated programming files include the extra bits required to initialize the device in PSA configuration. However, in PSA configuration, the FLEX 6000 device can initialize itself. Therefore, the FLEX 6000 device asserts CONF_DONE high and initializes itself before all data is sent. The microprocessor can stop sending configuration data when CONF_DONE is asserted high.

The FLEX 6000 device's nCS or CS pins can be toggled during PSA configuration if the design meets the specifications set for t_{CSSU}, t_{WSP}, and t_{CSH} in Table 19 on page 57. Figure 25 shows PSA configuration for FLEX 6000 devices.

Figure 25. PSA Configuration Circuit for FLEX 6000 Devices



Notes to Figure 25:

(1) The pull-up resistor should be connected to the same supply voltage as the FLEX 6000 device.

(2) The nCEO pin is left unconnected.

An optional address decoder can control the device's nCS and CS pins. This decoder allows the microprocessor to select the FLEX 6000 device by accessing a particular address, simplifying the configuration process. The microprocessor can also control the nCS and CS signals directly. You can tie one of the nCS or CS signals to its active state (i.e., nCS can be tied low) and the other signal can be toggled to control configuration.

The FLEX 6000 device can process data internally without the microprocessor. When the device is ready for the next bit of configuration data, it pulls RDYnBSY high, causing the microprocessor to strobe the next bit of configuration data into the FLEX 6000 device. Alternatively, the nRS signal can be strobed low, causing the RDYnBSY signal to appear on DATA. To simplify configuration, the microprocessor can wait for the total time of $t_{BUSY(Max)} + t_{RDY2WS} + t_{W2SB}$ before sending the next data bit.

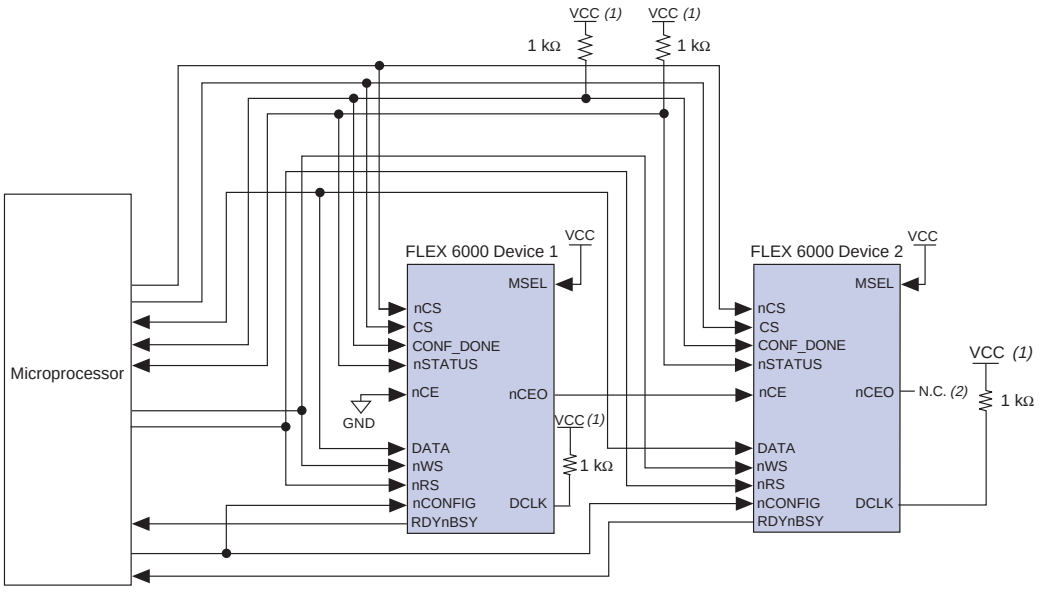
Because RDYnBSY does not need to be monitored, strobing nRS to read the state of the configuration data saves one system I/O port. You should not drive data onto the DATA pin while nRS is low because it causes contention. If the nRS pin is not used to monitor configuration, it should be tied high.

After configuration, the nCS, CS, nRS, nWS, and RDYnBSY pins act as user I/O pins. However, when using a PSA scheme, as a default these pins are tri-stated in user mode and should be driven by the microprocessor. The PSA scheme default can be changed in MAX+PLUS II software under "Global Project Device Option".

If the FLEX 6000 device detects an error during configuration, it drives nSTATUS low to alert the microprocessor. The microprocessor can then pulse nCONFIG low to restart the configuration process. Alternatively, if the *Auto-Restart Configuration on Frame Error* option is set in the Quartus II or MAX+PLUS II software, the FLEX 6000 device releases nSTATUS after a reset time-out period. After nSTATUS is released, the microprocessor can reconfigure the FLEX 6000 device. At this point, the microprocessor does not need to pulse nCONFIG low.

PSA mode can also be used to configure multiple FLEX 6000 devices. Multi-device PSA configuration is similar to single-device PSA configuration, except that the FLEX 6000 devices are cascaded. After the first FLEX 6000 device is configured, nCEO is asserted low, which asserts the second device's nCE pin low, causing it to begin configuration. The second FLEX 6000 device begins configuration within one write cycle of the first device; therefore, the transfer of data destinations is transparent to the microprocessor. All FLEX 6000 device CONF_DONE pins are tied together, so all FLEX 6000 devices initialize and enter user mode at the same time. If more than five FLEX 6000 devices are used, use a buffer to split the fan-out on the nWS signal. See Figure 26.

Figure 26. PSA Multi-Device Configuration Circuit for FLEX 6000 Devices



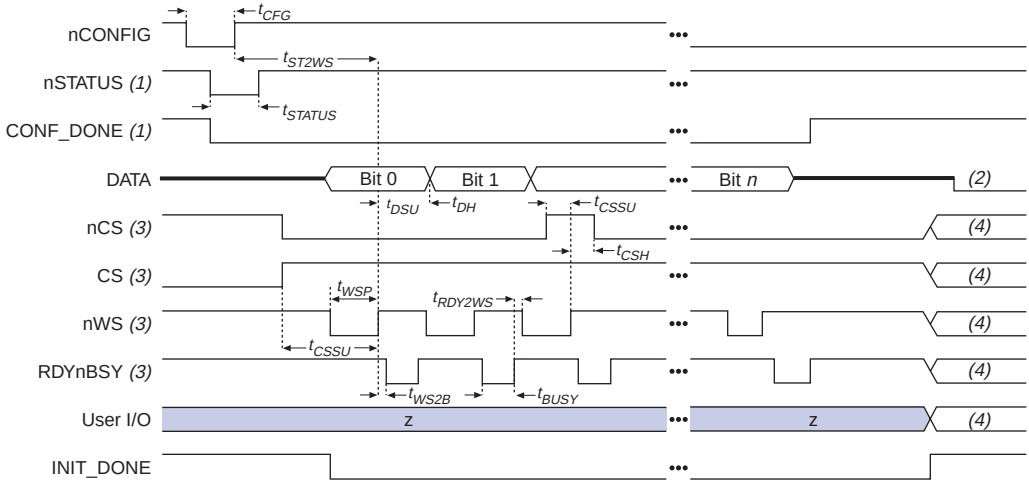
Notes to Figure 26:

(1) The pull-up resistor should be connected to the same supply voltage as the FLEX 6000 device.

(2) The nCEO pin is left unconnected for the last device in the chain.

Figure 27 shows the FLEX 6000 timing waveforms for PSA configuration.

Figure 27. PSA Timing Waveforms in FLEX 6000 Devices

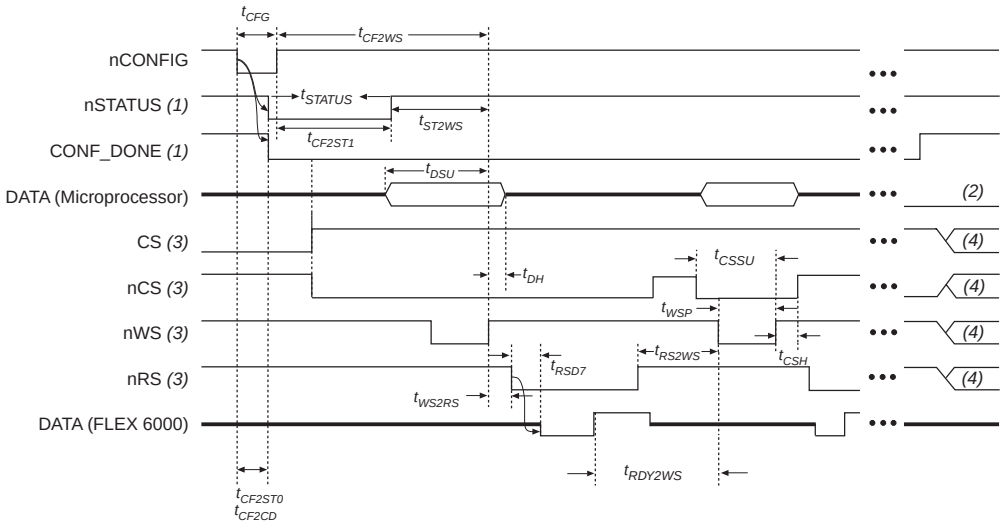


Notes to Figure 27:

(1) Upon power-up, nSTATUS is held low not more than 5 µs when V_{CC} reaches its minimum requirement.
(2) DATA should not be left floating. It should be driven high or low, whichever is more convenient.
(3) After configuration, the state of CS, nCS, nWS, and RDYnBSY depends on design programming in the FLEX 6000 device.
(4) Device I/O pins are in user mode.

Figure 28 shows the FLEX 6000 timing waveforms when using a strobed nRS and nWS signal.

Figure 28. PSA Timing Waveforms Using nRS & nWS in FLEX 6000 Devices



Notes to Figure 28:

(1) Upon power-up, nSTATUS is held low not more than 5 µs when V_{CC} reaches its minimum requirement.
(2) DATA should not be left floating. It should be driven high or low, whichever is more convenient.
(3) After configuration, the state of CS, nCS, nWS, nRS, and RDYnBSY depends on design programming in the FLEX 6000 device.
(4) Device I/O pins are in user mode.

Table 19 summarizes the timing parameters for PSA configuration.

Table 19. PSA Timing Parameters for FLEX 6000 Devices

Symbol	Parameter	Min	Max	Units
t_{CFG}	nCONFIG low pulse width *(1)*	2		µs
t_{STATUS}	nSTATUS low pulse width	2.5		µs
t_{CF2ST1}	nCONFIG high to nSTATUS high		4	µs
t_{ST2WS}	nSTATUS high to first rising edge on nWS	1		µs
t_{CF2WS}	nCONFIG high to first rising edge on nWS	5		µs
t_{DSU}	Data setup time before rising edge on nWS	20		ns
t_{DH}	Data hold time after rising edge on nWS	0		ns
t_{CSSU}	Chip select setup time before rising edge on nWS	20		ns
t_{CSH}	Chip select hold time after rising edge on nWS	5		ns
t_{WSP}	nWS low pulse width	50		ns
t_{WS2B}	nWS rising edge to RDYnBSY low		50	ns
t_{BUSY}	RDYnBSY low pulse width		200	ns
t_{RDY2WS}	RDYnBSY rising edge to nWS falling edge	50		ns
t_{WS2RS}	nWS rising edge to nRS falling edge	200		ns
t_{RS2WS}	nRS rising edge to nWS falling edge	200		ns
t_{RSD7}	nRS falling edge to DATA7 valid with RDYnBSY signal		50	ns
t_{CF2CD}	nCONFIG low to CONF_DONE low		1	µs
t_{CF2ST0}	nCONFIG low to nSTATUS low		1	µs

Note to Table 19:

(1) This value applies only if the internal oscillator is selected as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 10 to obtain this value.

For information on how to create configuration and programming files for this configuration scheme, see "Device Configuration Files" on page 92.

PPA Configuration (APEX II, APEX 20K, Mercury, ACEX 1K & FLEX 10K Devices Only)

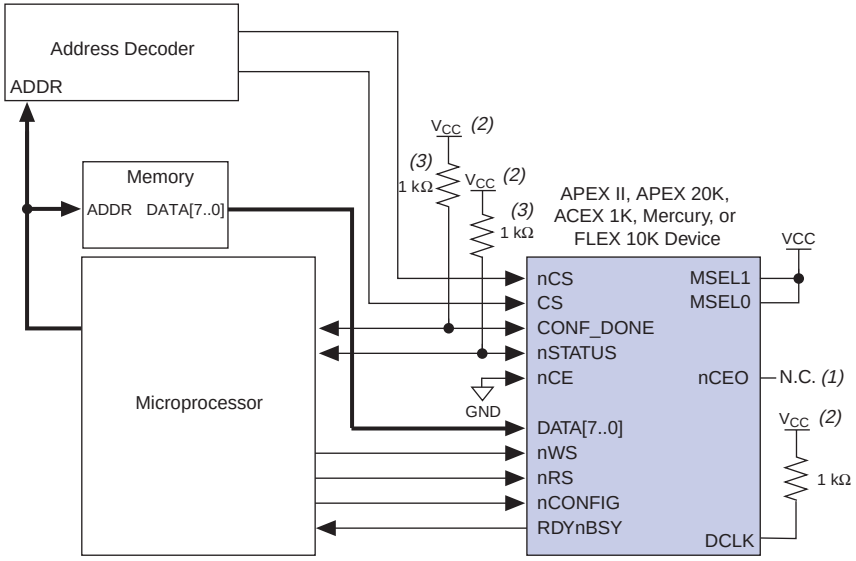
In passive parallel asynchronous (PPA) schemes, a microprocessor drives data to the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device via a download cable. When using a PPA scheme, you should pull the DCLK pin high through a 1-kΩ pull-up resistor to prevent unused configuration pins from floating.

To begin configuration, the microprocessor drives nCONFIG high and then asserts the target device's nCS pin low and CS pin high. Next, the microprocessor places an 8-bit configuration word on the target device's data inputs and pulses nWS low. On the rising edge of nWS, the target device latches a byte of configuration data and then drives its RDYnBSY signal low, indicating that it is processing the byte of configuration data. The microprocessor can then perform other system functions while the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device is processing the byte of configuration data.

Next, the microprocessor checks nSTATUS and CONF_DONE. If nSTATUS is not low and CONF_DONE is not released and pulled high, the microprocessor sends the next data byte. If nSTATUS is low, the device is signaling an error and the microprocessor should restart configuration. However, if nSTATUS is not low and all the configuration data has been received, the device is ready for initialization. At the beginning of initialization, CONF_DONE goes high to indicate that configuration is complete.

Figure 29 shows the PPA configuration circuit. An optional address decoder controls the device nCS and CS pins. This decoder allows the microprocessor to select the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device by accessing a particular address, simplifying the configuration process.

Figure 29. PPA Configuration Circuit in APEX II, APEX 20K, Mercury, ACEX 1K & FLEX 10K Devices



Notes to Figure 29:

(1) The nCEO pin is left unconnected.

(2) The pull up resistor should be connected to the same supply voltage as the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device.

(3) The pull up resistor should be 10 kΩ for APEX 20KE and APEX 20KC devices.

The device's nCS or CS pins can be toggled during PPA configuration if the design meets the specifications set for t_{CSSU}, t_{WSP}, and t_{CSH} in Tables 20 and 22 on page 64. The nCS and CS signals can also be controlled directly by the microprocessor. You can tie one of the nCS or CS signals to its active state (i.e., nCS may be tied low) and the other signal can be toggled to control configuration.

APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K devices can serialize data internally without the microprocessor. When the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device is ready for the next byte of configuration data, it drives RDYnBSY high. If the microprocessor senses a high signal when it polls RDYnBSY, the microprocessor strobes the next byte of configuration data into the device. Alternatively, the nRS signal can be strobed, causing the RDYnBSY signal to appear on DATA7. Because RDYnBSY does not need to be monitored, reading the state of the configuration data by strobing nRS low saves a system I/O port. Data should not be driven onto the data bus while nRS is low because it causes contention on DATA7. If the nRS pin is not used to monitor configuration, it should be tied high. To simplify configuration, the microprocessor can wait for the total time of t_{BUSY}(max) + t_{RDY2WS} + t_{W2SB} before sending the next data bit.

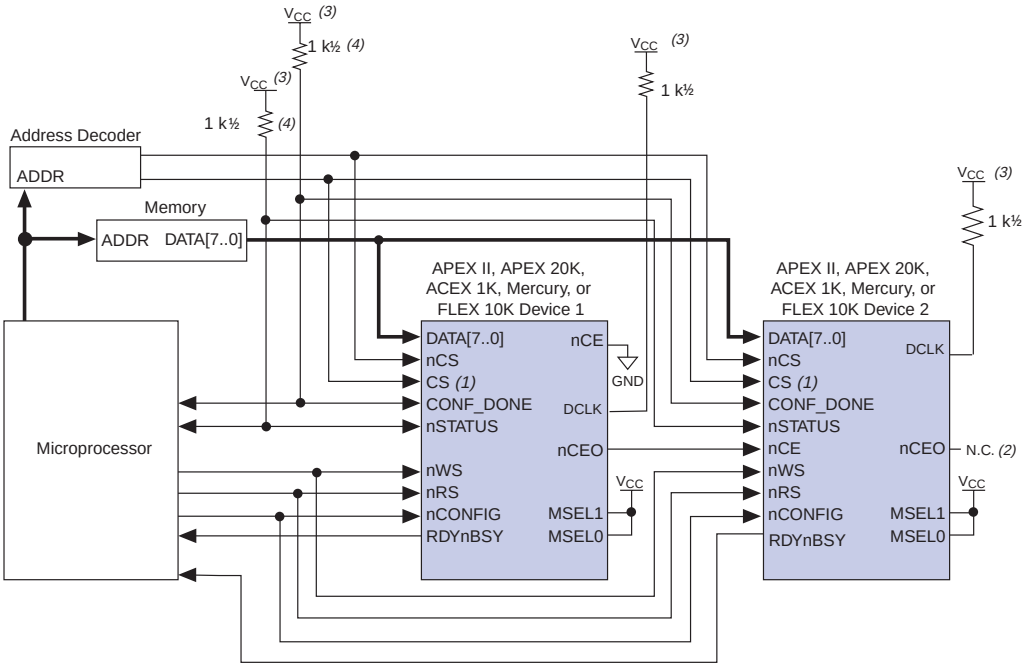
After configuration, the nCS, CS, nRS, nWS, and RDYnBSY pins act as user I/O pins. However, when the PPA scheme is chosen in the Quartus II or MAX+PLUS II software, as a default these I/O pins are tri-stated in user mode and should be driven by the microprocessor. To change this default option in the MAX+PLUS II software, select the **Global Project Device Option** dialog box or in the Quartus II software, select the **Device & Pin Option** dialog box in the Compiler Setting menu.

If the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device detects an error during configuration, it drives nSTATUS low to alert the microprocessor. The microprocessor can then pulse nCONFIG low to restart the configuration process. Alternatively, if the *Auto-Restart Configuration on Frame Error* option has been set in the Quartus II or MAX+PLUS II software, the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device releases nSTATUS after a reset time-out period. After nSTATUS is released, the microprocessor can reconfigure the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K. At this point, the microprocessor does not need to pulse nCONFIG low.

The microprocessor can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. The CONF_DONE pin must be monitored by the microprocessor to detect errors and determine when programming completes. If the microprocessor sends all configuration data and starts initialization but CONF_DONE is not asserted, the microprocessor must reconfigure the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device.

PPA mode can also be used to configure multiple APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K devices. Multi-device PPA configuration is similar to single-device PPA configuration, except that the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K devices are cascaded. After the first APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device is configured, nCEO is asserted, which asserts the nCE pin on the second device, causing it to begin configuration. The second APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device begins configuration within one write cycle of the first device; therefore, the transfer of data destinations is transparent to the microprocessor. All APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device CONF_DONE pins are tied together, so all devices initialize and enter user mode at the same time. See Figure 30.

Figure 30. PPA Multi-Device Configuration Circuit for APEX II, APEX 20K, Mercury, ACEX 1K & FLEX 10K Devices

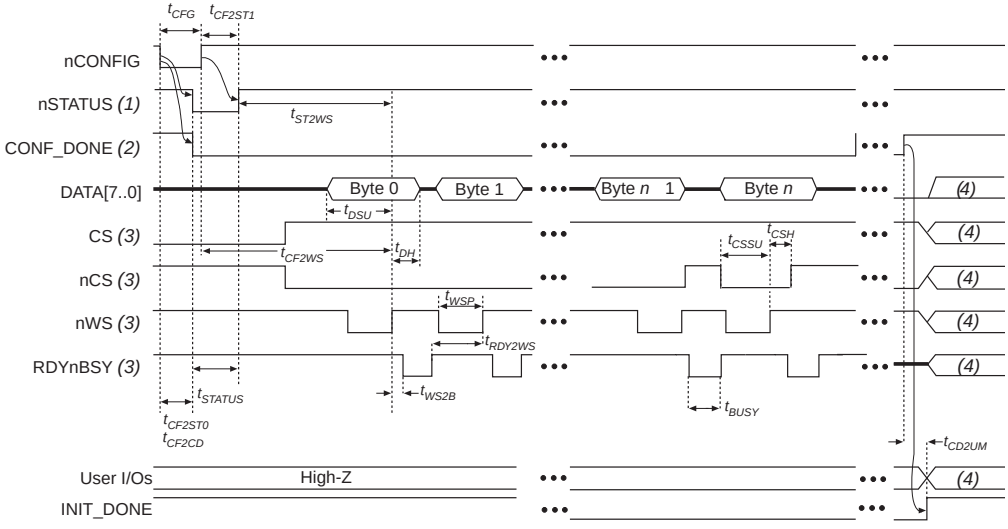


Notes to Figure 30:

(1) If not used, the CS pin can be connected to V_{CC} directly.
(2) The nCEO pin is left unconnected for the last device in the chain.
(3) The pull-up resistor should be connected to the same supply voltage as the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device.
(4) The pull-up resistor should be 10 kΩ for APEX 20KE and APEX 20KC devices.

Figure 31 shows the APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K timing waveforms for PPA configuration.

Figure 31. PPA Timing Waveforms for APEX 20K & FLEX 10K Devices

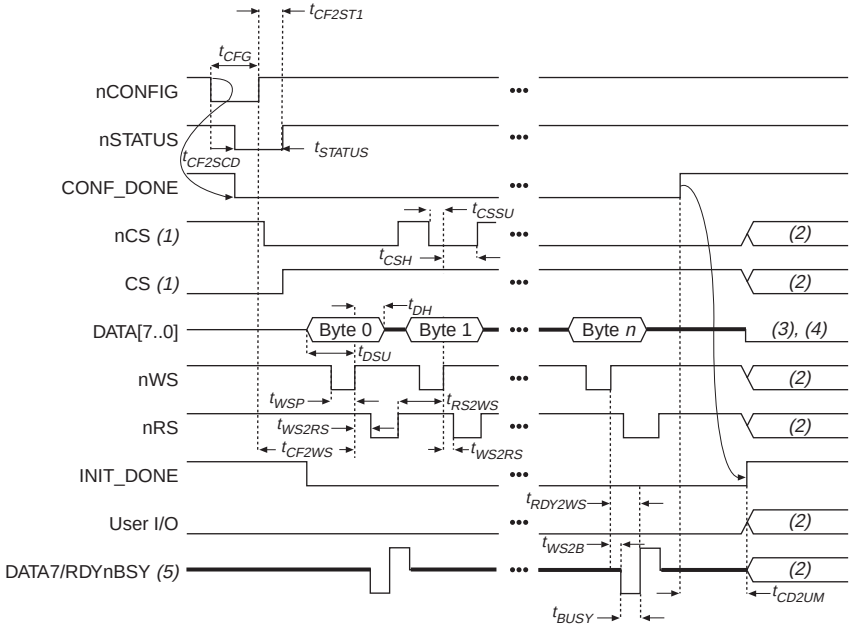


Notes to Figure 31:

(1) Upon power-up, nSTATUS is held low not more than 5 µs when V_{CC} reaches its minimum requirement.
(2) Upon power-up, CONF_DONE is low.
(3) After configuration, the state of CS, nCS, nWS, and RDYnBSY depends on the design programmed into the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device.
(4) Device I/O pins are in user mode.

Figure 32 shows the APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K timing waveforms when using a strobed nRS and nWS signal.

Figure 32. PPA Timing Waveforms Using nRS & nWS



Notes to Figure 32:

(1) The user can toggle nCS or CS during configuration if the design meets the specification for t_{CSSU}, t_{WSP}, and t_{CSH}.
(2) Device I/O pins are in user mode.
(3) DATA0 should not be left floating. It should be driven high or low, whichever is more convenient.
(4) Only the DATA[7..1] pins are I/O pins during user mode. DATA0 is only input in user mode.
(5) DATA7 is a bidirectional pin. It is input for data input, but it is output to show the status of RDYnBSY.

Tables 20 through 23 define the APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K timing parameters for PPA configuration.

Table 20. PPA Timing Parameters for APEX II & APEX 20K Devices *Note (1)*

Symbol	Parameter	Min	Max	Units
t_{CF2WS}	nCONFIG high to first rising edge on nWS	40		µs
t_{DSU}	Data setup time before rising edge on nWS	10		ns
t_{DH}	Data hold time after rising edge on nWS	0		ns
t_{CSSU}	Chip select setup time before rising edge on nWS	10		ns
t_{CSH}	Chip select hold time after rising edge on nWS	0		ns
t_{WSP}	nWS low pulse width	200		ns
t_{CFG}	nCONFIG low pulse width *(2)*	8		µs
t_{WS2B}	nWS rising edge to RDYnBSY low		50	ns
t_{BUSY}	RDYnBSY low pulse width	0.4	1.6	µs
t_{RDY2WS}	RDYnBSY rising edge to nWS rising edge	50		ns
t_{WS2RS}	nWS rising edge to nRS falling edge	200		ns
t_{RS2WS}	nRS rising edge to nWS rising edge	200		ns
t_{RSD7}	nRS falling edge to DATA7 valid with RDYnBSY signal		50	ns
t_{CD2UM}	CONF_DONE high to user mode *(3)*	2	8	µs
t_{STATUS}	nSTATUS low pulse width	10	40	µs
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		1 *(4)*	µs

Table 21. PPA Timing Parameters for Mercury Devices *Note (1)*				
Symbol	**Parameter**	**Min**	**Max**	**Units**
t_{CF2WS}	nCONFIG high to first rising edge on nWS	40		µs
t_{DSU}	Data setup time before rising edge on nWS	10		ns
t_{DH}	Data hold time after rising edge on nWS	0		ns
t_{CSSU}	Chip select setup time before rising edge on nWS	10		ns
t_{CSH}	Chip select hold time after rising edge on nWS	0		ns
t_{WSP}	nWS low pulse width	200		ns
t_{CFG}	nCONFIG low pulse width *(2)*	21		µs
t_{WS2B}	nWS rising edge to RDYnBSY low		50	ns
t_{BUSY}	RDYnBSY low pulse width	0.4	1.6	µs
t_{RDY2WS}	RDYnBSY rising edge to nWS rising edge	50		ns
t_{WS2RS}	nWS rising edge to nRS falling edge	200		ns
t_{RS2WS}	nRS rising edge to nWS rising edge	200		ns
t_{RSD7}	nRS falling edge to DATA7 valid with RDYnBSY signal		50	ns
t_{CD2UM}	CONF_DONE high to user mode *(3)*	6	28	µs
t_{STATUS}	nSTATUS low pulse width	10	40	µs
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		1 *(4)*	µs

Table 22. PPA Timing Parameters for ACEX 1K Devices

Symbol	Parameter	Min	Max	Units
t_{CF2WS}	nCONFIG high to first rising edge on nWS	5		µs
t_{DSU}	Data setup time before rising edge on nWS	20		ns
t_{DH}	Data hold time after rising edge on nWS	0		ns
t_{CSSU}	Chip select setup time before rising edge on nWS	20		ns
t_{CSH}	Chip select hold time after rising edge on nWS	10		ns
t_{WSP}	nWS low pulse width	200		ns
t_{CFG}	nCONFIG low pulse width *(5)*	2		µs
t_{WS2B}	nWS rising edge to RDYnBSY low		50	ns
t_{BUSY}	RDYnBSY low pulse width	0.4	1.6	µs
t_{RDY2WS}	RDYnBSY rising edge to nWS rising edge	50		ns
t_{WS2RS}	nWS rising edge to nRS falling edge	200		ns
t_{RS2WS}	nRS rising edge to nWS rising edge	200		ns
t_{RSD7}	nRS falling edge to DATA7 valid with RDYnBSY signal		50	ns
t_{CD2UM}	CONF_DONE high to user mode *(6)*	0.6	2	µs
t_{STATUS}	nSTATUS low pulse width	1		µs
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		4	µs

Table 23. PPA Timing Parameters for FLEX 10K Devices

Symbol	Parameter	Min	Max	Units
t_{CF2WS}	nCONFIG high to first rising edge on nWS	5		µs
t_{DSU}	Data setup time before rising edge on nWS	20		ns
t_{DH}	Data hold time after rising edge on nWS	0		ns
t_{CSSU}	Chip select setup time before rising edge on nWS	20		ns
t_{CSH}	Chip select hold time after rising edge on nWS	10 *(7)* 15 *(8)*		ns
t_{WSP}	nWS low pulse width	200		ns
t_{CFG}	nCONFIG low pulse width *(5)*	2		µs
t_{WS2B}	nWS rising edge to RDYnBSY low		50	ns
t_{BUSY}	RDYnBSY low pulse width	0.4	1.6	µs
t_{RDY2WS}	RDYnBSY rising edge to nWS rising edge	50		ns
t_{WS2RS}	nWS rising edge to nRS falling edge	200		ns
t_{RS2WS}	nRS rising edge to nWS rising edge	200		ns
t_{RSD7}	nRS falling edge to DATA7 valid with RDYnBSY signal		50	ns
t_{CD2UM}	CONF_DONE high to user mode *(6)*	0.6	2	µs
t_{STATUS}	nSTATUS low pulse width	1		µs
t_{CF2CD}	nCONFIG low to CONF_DONE low		200	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		4	µs

Notes to Tables 20 – 23:

(1) Timing information for APEX II and Mercury devices is preliminary.
(2) This value applies only if the internal oscillator is selected as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 40 to obtain this value.
(3) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device. If the clock source is CLKUSR or DCLK, multiply the clock period by 40 for APEX II and APEX 20K devices or 136 for Mercury devices to obtain this value.
(4) This value is obtainable if users do not delay configuration by extending the nSTATUS low pulse width.
(5) This value only applies if the internal oscillator is selected as the clock source for starting up the device. If the clock source is CLKUSR or DCLK multiply the clock period by 10 to obtain this value.
(6) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device. If the clock source is CLKUSR, multiply the clock period by 10 to obtain this value.
(7) This parameter value applies to EPF10K10, EPF10K20, EPF10K40, EPF10K50, all FLEX 10KA, and FLEX 10KE devices.
(8) This parameter value applies to EPF10K70 and EPF10K100 devices only.

For information on how to create configuration and programming files for this configuration scheme, see "Device Configuration Files" on page 92.

JTAG Programming & Configuration (APEX II, APEX 20K, Mercury, ACEX 1K & FLEX 10K Devices)

The Joint Test Action Group (JTAG) has developed a specification for boundary-scan testing. This boundary-scan test (BST) architecture offers the capability to efficiently test components on PCBs with tight lead spacing. The BST architecture can test pin connections without using physical test probes and capture functional data while a device is operating normally. The JTAG circuitry can also be used to shift configuration data into the device.

For more information on JTAG boundary-scan testing, see *Application Note 39 (IEEE 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices)*.

A device operating in JTAG mode uses four required pins, TDI, TDO, TMS, and TCK, and one optional pin, TRST. All other pins are tri-stated during JTAG configuration. You should not begin JTAG configuration until all other configuration is complete. Table 24 shows each JTAG pin's function.

Table 24. JTAG Pin Descriptions

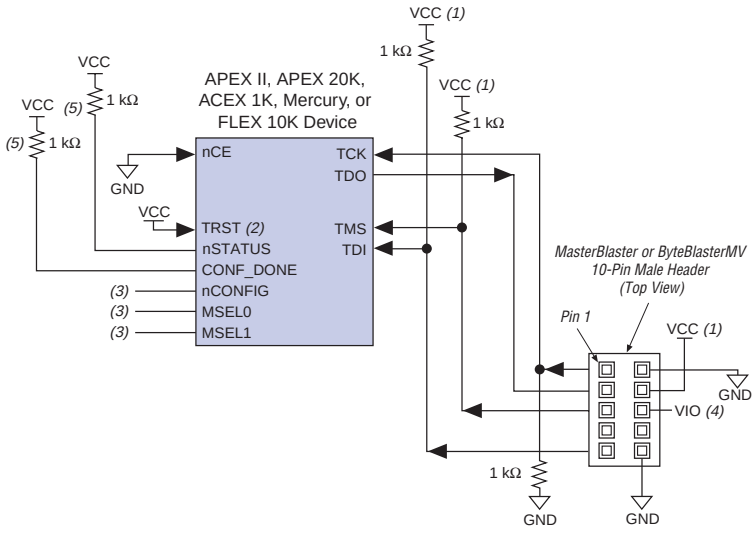
Pin	Description	Function
TDI	Test data input	Serial input pin for instructions as well as test and programming data. Data is shifted in on the rising edge of TCK.
TDO	Test data output	Serial data output pin for instructions as well as test and programming data. Data is shifted out on the falling edge of TCK. The pin is tri-stated if data is not being shifted out of the device.
TMS	Test mode select	Input pin that provides the control signal to determine the transitions of the TAP controller state machine. Transitions within the state machine occur on the rising edge of TCK. Therefore, TMS must be set up before the rising edge of TCK. TMS is evaluated on the rising edge of TCK.
TCK	Test clock input	The clock input to the BST circuitry. Some operations occur at the rising edge, while others occur at the falling edge.
TRST *(1)*	Test reset input (optional)	Active-low input to asynchronously reset the boundary-scan circuit. The TRST pin is optional according to IEEE Std. 1149.1.

Note to Table 24:

(1) FLEX 10K devices in 144-pin thin quad flat pack (TQFP) packages do not have a TRST pin. Therefore, the TRST pin can be ignored when using these devices.

During JTAG configuration, data is downloaded to the device on the PCB through the MasterBlaster or ByteBlasterMV header. Configuring devices through a cable is similar to programming devices in-system, except the TRST pin should be connected to V_{CC}; this connection ensures that the TAP controller is not reset. See Figure 33.

Figure 33. JTAG Configuration of a Single APEX II, APEX 20K, Mercury, ACEX 1K or FLEX 10K Device



Notes to Figure 33:

(1) The pull-up resistor should be connected to the same supply voltage as the download cable.

(2) FLEX 10K devices in 144-pin TQFP packages do not have a TRST pin. Therefore, the TRST pin can be ignored when configuring FLEX 10K devices in 144-pin TQFP packages.

(3) The nCONFIG, MSEL0, and MSEL1 pins should be connected to support a non JTAG configuration scheme. If only JTAG configuration is used, connect nCONFIG to V_{CC}, and MSEL0 and MSEL1 to ground.

(4) V_{IO} is a reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

(5) The pull-up resistor should be 10 kΩ for APEX 20KE and APEX 20KC devices.

To configure a single device in a JTAG chain, the programming software places all other devices in BYPASS mode. In BYPASS mode, devices pass programming data from the TDI pin to the TDO pin through a single bypass register without being affected internally. This scheme enables the programming software to program or verify the target device. Configuration data driven into the device appears on the TDO pin one clock cycle later.

APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K devices have dedicated JTAG pins that always function as JTAG pins. JTAG testing can be performed on APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices both before and after configuration, but not during configuration. The chip-wide reset and output enable pins on APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices do not affect JTAG boundary-scan or programming operations. Toggling these pins does not affect JTAG operations (other than the usual boundary-scan operation).

When designing a board for JTAG configuration of APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K devices, the regular configuration pins should be considered. Table 25 shows how these pins should be connected during JTAG configuration.

Table 25. Pin Descriptions

Signal	Description
nCE	On all APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K devices in the chain should be driven low by connecting nCE to ground, pulling it down via a resistor, or driving it by some control circuitry.
nSTATUS	Pulled to V_{CC} via a 1-kΩ or 10-kΩ resistor. When configuring multiple devices in the same JTAG chain, each nSTATUS pin should be pulled up to V_{CC} individually. *(1)*
CONF_DONE	Pulled to V_{CC} via a 1-kΩ or 10-kΩ resistor. When configuring multiple devices in the same JTAG chain, each CONF_DONE pin should be pulled up to V_{CC} individually. *(1)*
nCONFIG	Driven high by connecting to V_{CC}, pulling up via a resistor, or driven by some control circuitry.
MSEL0, MSEL1	These pins must not be left floating. These pins support whichever non-JTAG configuration is used in production. If only JTAG configuration is used, you should tie both pins to ground.
DCLK	Should not be left floating. Drive low or high, whichever is more convenient.
DATA0	Should not be left floating. Drive low or high, whichever is more convenient.
TRST	This JTAG pin is not connected to the download cable. It should be driven to logic high.

Note to Table 25:

(1) nSTATUS pulling low in the middle of JTAG configuration indicates that an error has occurred. CONF_DONE pulling high at the end of JTAG configuration indicates successful configuration.

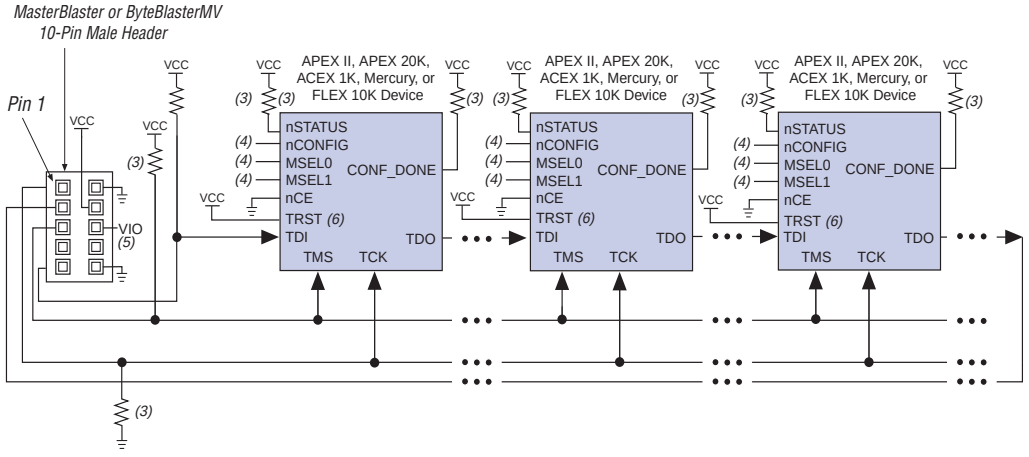
For information on how to create configuration and programming files for this configuration scheme, see "Device Configuration Files" on page 92.

JTAG Programming & Configuration of Multiple Devices (APEX II, APEX 20K, Mercury, ACEX 1K & FLEX 10K Devices)

When programming a JTAG device chain, one JTAG-compatible header, such as the ByteBlasterMV header, is connected to several devices. The number of devices in the JTAG chain is limited only by the drive capability of the download cable. However, when more than five devices are connected in a JTAG chain, Altera recommends buffering the TCK, TDI, and TMS pins with an on-board buffer.

JTAG-chain device programming is ideal when the PCB contains multiple devices, or when testing the PCB using JTAG BST circuitry. Figure 34 shows configuration.

Figure 34. Multi-Device JTAG Configuration *Notes (1), (2)*



Notes to Figure 34:

(1) APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and MAX devices can be placed within the same JTAG chain for device programming and configuration.

(2) For more information on all configuration pins connected in this mode, refer to Table 25 on page 70.

(3) All pull-up / pull-down resistors are 1 kΩ. For APEX 20KE and APEX 20KC devices, the pull up resistors on nSTATUS and CONF_DONE are 10 kΩ.

(4) The nCONFIG, MSEL0, and MSEL1 pins should be connected to support a non-JTAG configuration scheme. If only JTAG configuration is used, connect nCONFIG to V_{CC}, and MSEL0 and MSEL1 to ground.

(5) VIO is a reference voltage for the MasterBlaster output driver. VIO should match the device's V_{CCIO}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

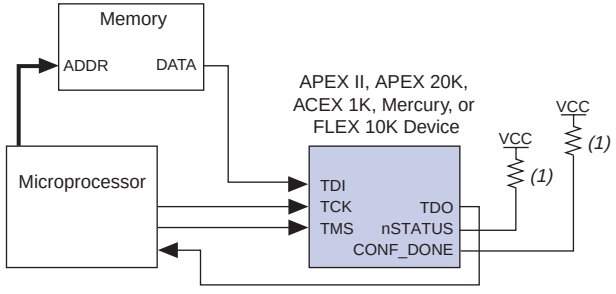
(6) The TRST pin is only available on APEX II, APEX 20K, Mercury, and ACEX 1K devices, and on all FLEX 10K devices except in the TQFP 144-pin package.

Successful JTAG configuration is verified automatically by the Quartus II or MAX+PLUS II software at the end of JTAG configuration. The software checks the state of CONF_DONE through the JTAG port at the end of configuration. If CONF_DONE is not in the correct state, the Quartus II or MAX+PLUS II software indicates that configuration has failed. If CONF_DONE is in the correct state, the software indicates that configuration was successful.

When using the JTAG pins for configuration, if V_{CCIO} is tied to 3.3 V, both the I/O pins and JTAG TDO port will drive at 3.3-V levels.

JTAG and non-JTAG configuration should not be attempted simultaneously. When configuring via JTAG, allow any non-JTAG configuration to complete first. Figure 35 shows the JTAG configuration of an APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device with a microprocessor.

Figure 35. JTAG Configuration of APEX II, APEX 20K, Mercury, ACEX 1K or FLEX 10K Device with a Microprocessor



Note to Figure 35:

(1) All pull-up resistors are 1kΩ except for APEX 20KE and APEX 20KC devices, which are 10 kΩ

Jam STAPL Programming & Test Language

In-circuit configuration via an embedded processor enables easy design prototyping, streamlines production, and allows quick and efficient in-field upgrades. The Jam™ Standard Test and Programming Language (STAPL) programming and test language, a standard file format using the IEEE Std. 1149.1 (JTAG) interface, further simplifies in-circuit configuration by providing small file sizes and increased flexibility. Jam Files and Jam Byte-Code Files (**.jbc**) contain both the programming algorithm and data required to upgrade one or more devices. The Jam language is supported by MAX+PLUS II software versions 8.0 and higher and the Quartus II software.

You can estimate the JBC size using the following equation:

$$\text{JBC Size} = Alg + \sum_{k=1}^{N} Data$$

Where: Alg = Space used by the algorithm (see Table 26)
$Data$ = Space used by compressed programming data (see Table 27)
k = Index representing family type(s) being targeted
N = Number of target devices in the chain

Table 26. Algorithm Constants

Device	Typical JBC File Algorithm Size (Kbytes)
APEX 20K	14
APEX 20KE	14
ACEX 1K	15
FLEX 10K	15
FLEX 10KE	15
FLEX 10KA	15

Table 27. Data Constants

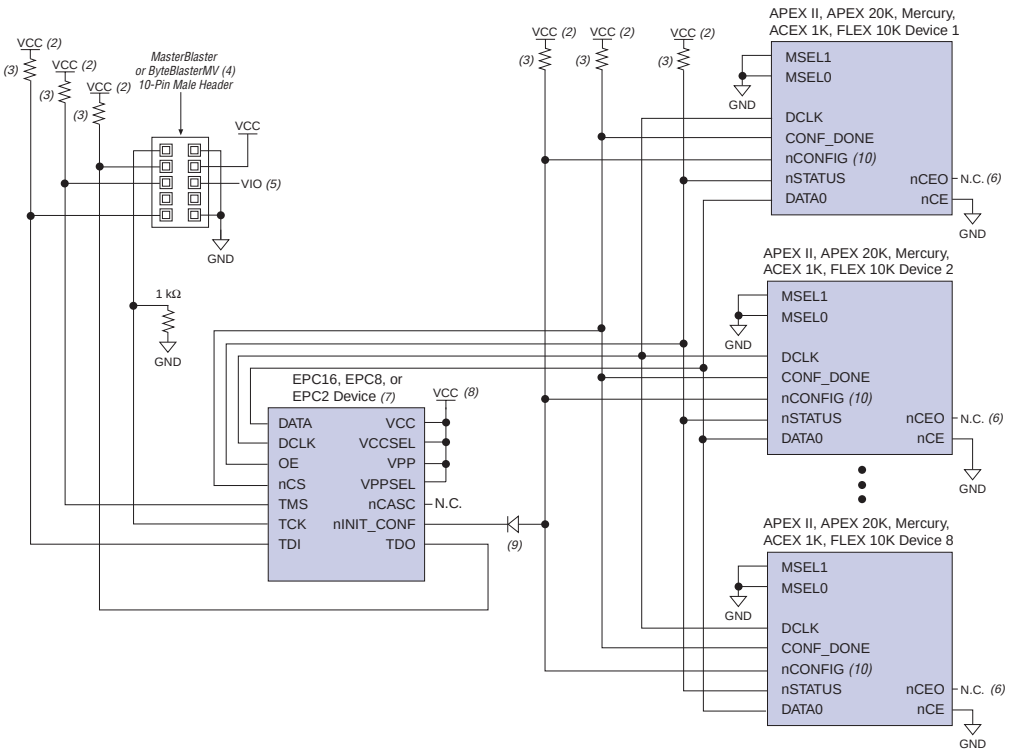
Device	Typical Jam STAPL Byte-Code Data Size (Kbytes)	
	Compressed	Uncompressed
EPF10K10, EPF10K10A	12	15
EPF10K20	21	29
EPF10K30	33	47
EPF10K30A	36	51
EPF10K30E	36	59
EPF10K40	37	62
EPF10K10K50, EPF10K50V	50	78
EPF10K50E	52	98
EPF10K70	76	112
EPF10K100, EPF10K100A, EPF10K100B	95	149
EPF10K100E	102	167
EPF10130E	140	230
EPF10K130V	136	199
EPF10K200E	205	345
EPF10K250A	235	413
EP1K10		
EP1K30	36	59
EP1K50	50	78
EP1K100	95	149
EP20K100	128	244
EP20K200	249	475
EP20K400	619	1,180

For more information on how to configure devices using the Jam STAPL programming and test language, see *Application Note 122 (Using Jam STAPL for ISP & ICR via an Embedded Processor)*.

Combining Different Configuration Schemes

This section shows you how to configure APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices using multiple configuration schemes on the same board. Figure 36 shows the configuration of an APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device using a download cable and a configuration device. In Figure 36, multiple PLDs are configured simultaneously with the same data.

Figure 36. Device Configuration with a Download Cable & Configuration Device *Note (1)*



Notes to Figure 36:

(1) In this Figure 36, eight PLDs are simultaneously configured with the same data.

(2) V_{CC} should be connected to the same supply voltage as the configuration device.

(3) All pull-up resistors are 1 kΩ. On APEX 20KE and APEX 20KC devices, pull-up resistors on nSTATUS and CONF_DONE pins are 10 kΩ. The OE, nCS and nINIT_CONF pins on EPC16, EPC8, and EPC2 devices have internal user configurable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins.

(4) The download cable programs the configuration device (EPC16, EPC8, or EPC2 device).

(5) V_{IO} is a reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

(6) The nCEO pin is left unconnected for single device configuration.

(7) If a 3.3-V supply voltage is used, the VCC, VCCSEL, VPP, and VPPSEL pins should be connected to a 3.3-V supply. If a 5.0-V supply voltage is used, the VCC and VPP pins should be connected to a 5.0-V supply, and the VCCSEL and VPPSEL pins should be connected to ground. To improve in-system programming times, you can connect VPP to 5.0 V, VCC to 3.3 V, and VPPSEL to ground. For more information on these pins, see Table 29 on page 88.

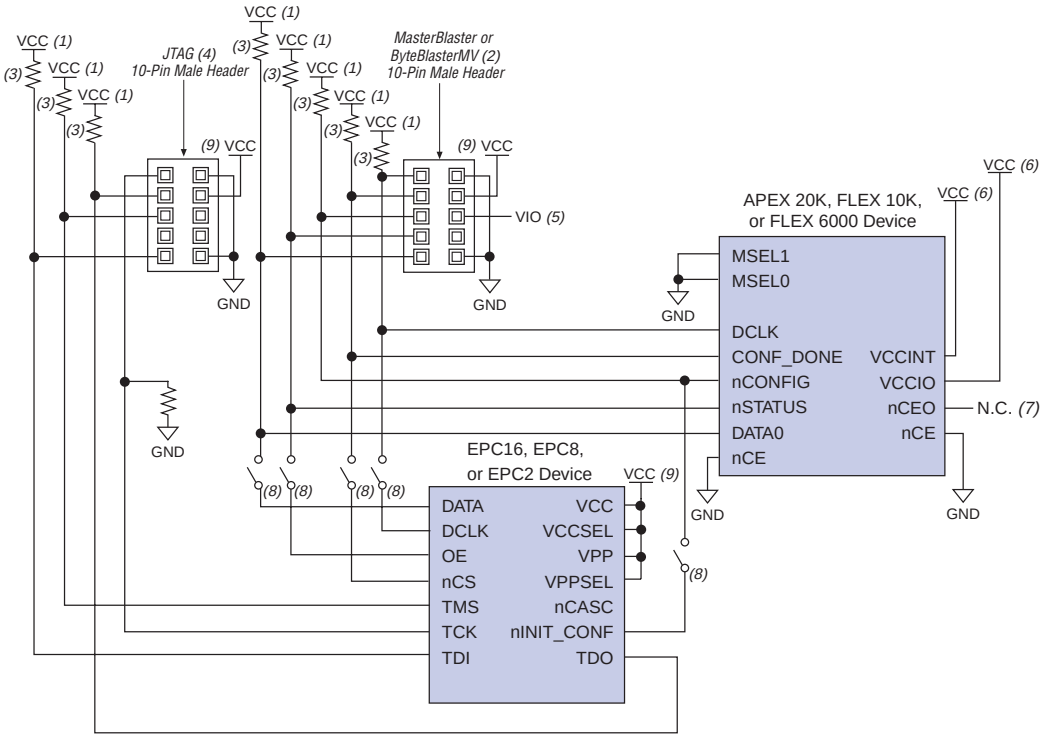
(8) The configuration device configures the APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device. Figure 36 shows the pin connections for an EPC16, EPC8, or EPC2 configuration device. For any other configuration device, connect the pins appropriately.

(9) Add a diode between the APEX 20KE device's nCONFIG pin and the configuration device's nINIT_CONF pin to isolate the 1.8-V and 3.3-V power supplies. Select a diode with a threshold voltage (VT) less than or equal to 0.7 V. The diode will make the nINIT_CONF pin an open-drain pin. These pins will only break to drive low or tri-state.

(10) To ensure that the configuration device successfully configures the APEX 20KE and APEX 20KC device in all possible power-up sequences, use a resistor to pull nCONFIG up to V_{CCINT}.

Figure 37 shows a schematic for configuring APEX 20K, FLEX 10K, or FLEX 6000 devices using two download cables and an EPC2 device.

Figure 37. Device Configuration with a Download Cable & an EPC2, EPC8, or EPC16 Device



Notes to Figure 37:

(1) V_{CC} should be connected to the same supply voltage as the configuration device.

(2) The target APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device can be configured by either the configuration device or the download cable.

(3) All pull-up resistors are 1 kΩ. On APEX 20KE and APEX 20KC devices, pull-up resistors for nSTATUS and CONF_DONE pins should be 10 kΩ.

(4) The download cable programs the configuration device through the JTAG circuitry.

(5) V_{IO} is a reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

(6) V_{CCINT} and V_{CCIO} should be applied according to the target device's V_{CCINT} and V_{CCIO}.

(7) The nCEO pin is left unconnected.

(8) You should not attempt configuration with a download cable while a configuration device is connected to an APEX II, APEX 20K, Mercury, ACEX 1K, or FLEX 10K device. To perform this operation, you should either remove the configuration device from its socket when using the download cable, or place a switch on the five common signals between the download cable and the configuration device.

(9) If a 3.3-V supply voltage is used, the VCC, VCCSEL, VPP, and VPPSEL pins should be connected to a 3.3-V supply voltage. If a 5.0-V supply voltage is used, the VCC and VPP pins should be connected to a 5.0-V supply voltage, and the VCCSEL and VPPSEL pins should be connected to ground. To improve in-system programming times, you can connect VPP to 5.0 V, VCC to 3.3 V, and VPPSEL to ground. For more information on these pins, see Table 29 on page 88.

Using Flash Memory to Configure PLDs

As Altera introduces higher-density PLDs, the configuration bit stream size also increases. As a result, designs require more configuration devices to store the data and configure these devices. The disadvantage of having multiple configuration devices is not only that they take up valuable board space, but multiple devices also increase board complexity.

Flash memory can be used to store configuration data. A Flash memory controller is required to read and write to the Flash memory and perform configuration. You can use a MAX 3000 or MAX 7000 device to implement the Flash memory controller.

There are separate design files available for both MAX 7000 and MAX 3000A devices on the Altera web site (**http://www.altera.com**). You can download these files from the application notes literature page underneath *AN 116: Configuring SRAM-Based LUT Devices*.

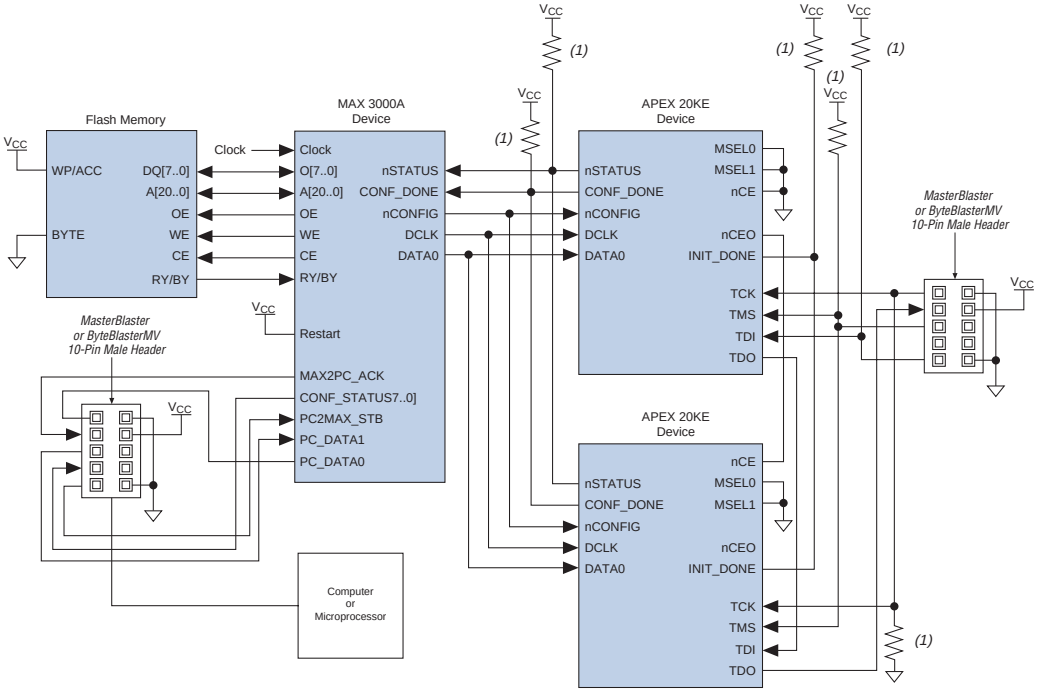
Device Configuration Using Flash Memory & MAX 3000A Devices

The Flash memory controller can interface with a PC or microprocessor to receive configuration data via parallel port (Figure 38). The controller generates a programming command sequence to program the Flash memory and extract configuration data to configure PLDs.

The Flash memory controller supports various commands such as:

- Programming the Flash memory
- Configuring PLDs
- Verifying the content of Flash memory

Figure 38. Configuring PLD through Flash Memory & MAX 3000A Controller



Note to Figure 38:

(1) All resistors are 1 kΩ. For APEX 20KE and APEX 20KC devices, resistors are 10 kΩ for the nSTATUS and CONF_DONE pins.

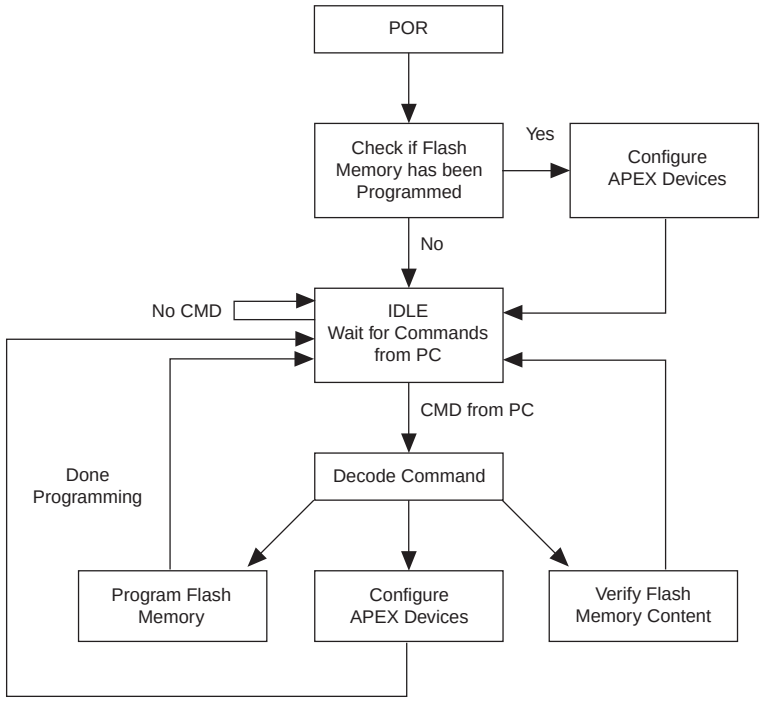
Flash Memory Controller Design Specification

The controller will check to see if the Flash memory is programmed successfully after the board powers up. If the Flash memory is programmed successfully, then the controller configures the PLDs. If Flash memory is not programmed successfully, then the controller waits for commands from the PC or microprocessor. The receiver decodes the commands it receives from the PC or microprocessor as one of the following:

- Program Flash memory
- Configure PLD
- Verify Flash memory programming

After a command is executed, the controller returns to idle mode and waits for the next command. Figure 39 shows the controller state machine.

Figure 39. Flash Memory Controller State Machine



Flash Memory Controller Functionality

The controller writes a byte to a special location in the Flash memory when it programs the memory. After POR, the controller checks this special location in the Flash memory to see if the byte is written there or not.

If the byte is written, then the Flash memory has been programmed and the controller can proceed to configuring the PLDs by reading data from the Flash memory. If this byte is not there or the value is not as expected, the controller will go idle and wait to be programmed by the PC or microprocessor.

Getting Data from the PC or Microprocessor

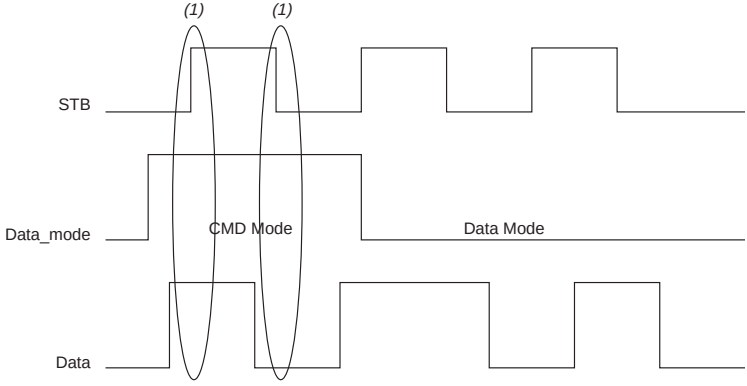
The PC or microprocessor uses the parallel port to interface with the controller. There are two types of signals involved in this connection (see Figure 40), a 3-bit input signal from the PC or microprocessor to the controller, and a 2-bit output signal from the controller to the PC or microprocessor. The input signal includes the following three signals:

- STB: Strobe signal from the PC or microprocessor to indicate that the PC or microprocessor's data is valid.
- `data_mode`: Indicates whether the controller is in command mode or data mode. When `data_mode` is high, the controller is in command mode; when `data_mode` is low, the controller is in data mode.
- `data`: Content of this signal depends on `data_mode`. It can be data for command mode or data mode.

The output signal contains the following two signals:

- ACK: Acknowledge signal is a handshaking signal from the controller to the PC or microprocessor.
- `conf_status`: Indicates configuration status.

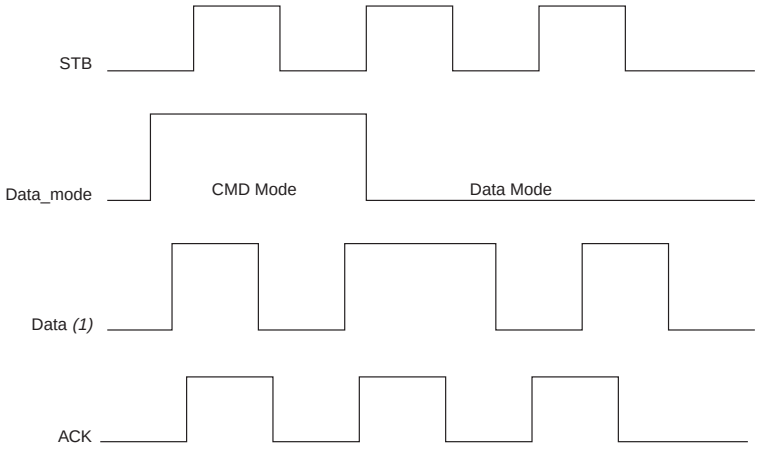
Figure 40. Getting Data from PC or Microprocessor



Note to Figure 40:
(1) Data is sent on both positive and negative edges of the STB signal.

The controller receives a bit of data or command from PC or microprocessor on the rising and falling edges of the STB signal. After receiving this data, controller will send an acknowledgement signal to the PC or microprocessor to initiate sending of the next bit of data. The acknowledge signal (ACK) should be the same logic level as the last received STB signal. By de-asserting ACK, the controller can stop the PC or microprocessor from sending data. Figure 41 shows the STB and ACK relationship.

Figure 41. Sending Acknowledge Signal (ACK) to PC or Microprocessor



Note to Figure 41:
(1) One bit of data is received at each STB signal edge (both positive and negative).

Programming Flash Memory

After receiving a command from the PC or microprocessor, the controller first erases and then starts programming the Flash memory. A separate state machine is required to generate a programming command sequence and programming pulse width.

While programming the Flash memory, the controller must check if a command (`data_mode=1`) has been received or not. A command indicates the end of data from the PC or microprocessor, and the controller will exit Program_Flash_memory state and go into idle mode.

Another state machine is required to read and serialize byte data from the Flash memory and generate DCLK and DATA0. The controller needs to monitor CONF_DONE signals from the PLDs to determine if configuration is complete. When configuration is done, the controller exits the configure state and goes back to idle mode.

Flash Memory Content Verification

There are two types of Flash memory controllers. One works with AMD / Fujitsu Flash memory, and the other works with Intel's Flash memory. Verification is not supported with AMD / Fujitsu Flash Memory, but is supported with Intel's.

To verify that the content of the Intel Flash memory is the same as the configuration file, a verifying command is necessary. Once it issues the verifying command, the MAX device will read back from the Flash memory. Every time the MAX device reads a byte, it will send each bit of data from that byte to the PC or microprocessor through the ByteBlasterMV cable. The software running on the PC realigns each bit into a byte and then reconstructs the TTF file based on the byte data read back from the MAX device. During reconstruction, the software adds in required commas and line breaks into the file so that it will have exactly the same format as the TTF file the Quartus II software created.

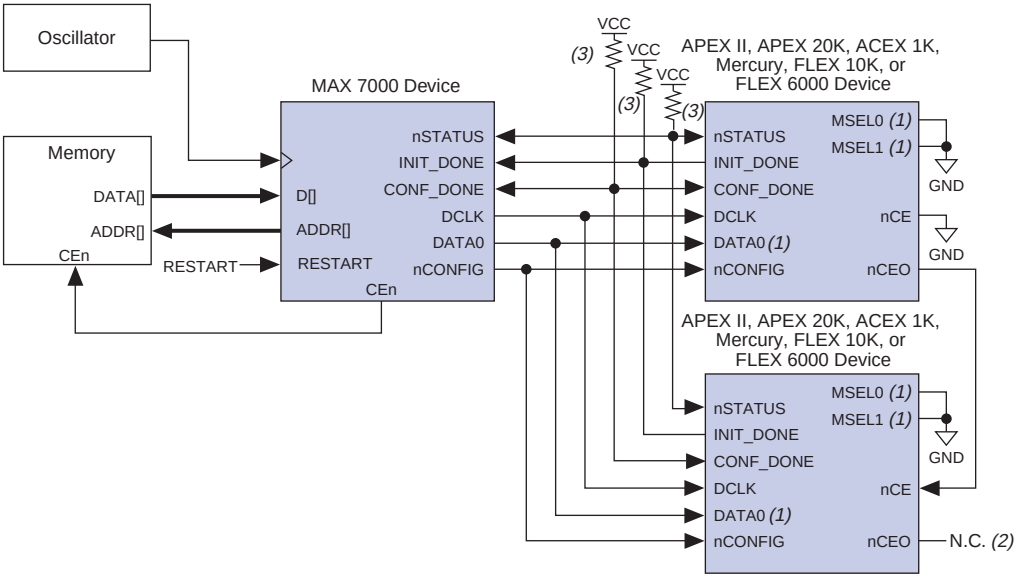
The whole process stops when the MAX device finishes reading back from the Flash memory. At this time, the software is finished constructing the TTF file. At this point, the software will compare the reconstructed TTF file with original TTF to check if they are the same. The program will output a message stating the results of the comparison.

If an error is found during verification, the controller will set the CONF_STATUS signal to low to inform the PC or microprocessor that there is a verification error, and returns to the idle state immediately. During verification, controller still needs to check if a command (data_mode=1) has been received or not. This command indicates the end of data from PC or microprocessor. Following this, controller will exit Program_Flash_Memory state and go into idle mode.

Device Configuration Using Flash Memory & MAX 7000 Devices

Figure 42 shows the schematic for this configuration scheme with a MAX 7000 device. Two sample design files for the MAX 7000 device (*Figure 31 Design File for Configuring APEX 20K Devices and Figure 31 Design File for Configuring FLEX 10K and FLEX 6000 Devices*) are available on the Altera web site at **http://www.altera.com/html/literature/lan.html** under *AN 116: Configuring APEX 20K, FLEX 10K & FLEX 6000 Devices.*

Figure 42. Device Configuration Using External Memory & a MAX 7000 Device

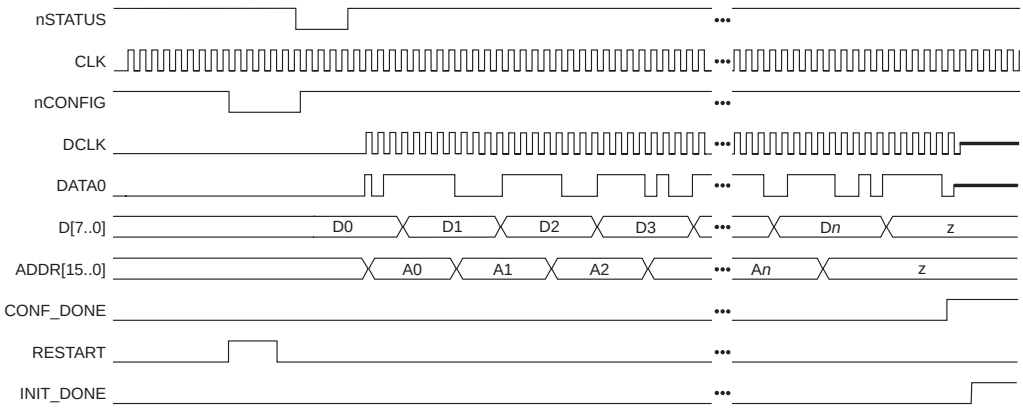


Notes to Figure 42:

(1) FLEX 6000 devices have a single MSEL pin, which is tied to ground, and its DATA0 pin is renamed DATA.
(2) The nCEO pin is left unconnected for the last device in the chain.
(3) All pull-up resistors are 1 kΩ. On APEX 20KE and APEX 20KC devices, pull-up resistors for nSTATUS, CONF_DONE, and INIT_DONE pins should be 10 kΩ.

Figure 43 shows the timing waveform for configuring an APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 device using external memory and a MAX 7000 device.

Figure 43. Timing Waveform for Configuration Using External Memory & a MAX 7000 Device



Device Options

You can set ACEX 1K, FLEX 10K, and FLEX 6000 device options in Altera's MAX+PLUS II development software by choosing **Global Project Device Options** (Assign menu). You can also set device options in the Quartus II software using the **Device & Pin Option** dialog box. To choose this option, select the Processing menu, choose Compiler Settings, then click the **Chips & Devices** tab. Table 28 summarizes each of these options.

Table 28. APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K & FLEX 6000 Configuration Option Bits (Part 1 of 3)

Device Option	Option Usage	Default Configuration (Option Off)	Modified Configuration (Option On)
User-supplied start-up clock (ACEX 1K, FLEX 10K, and FLEX 6000 devices only.)	To complete initialization, the device must be clocked 10 times after all data is transferred. CONF_DONE goes high after the initialization process has begun. You can select which clock source to use for initialization by choosing CLKUSR rather than DCLK.	In the PPA and PSA configuration schemes, the device's internal oscillator supplies the initialization clock. In the configuration device, PS, and PPS schemes, the internal oscillator is disabled. Thus, external circuitry, such as a configuration device or download cable, must provide the initialization clock on the DCLK pin. In the configuration device scheme, the configuration device supplies the clock; in PS and PPS schemes, the microprocessor supplies the clock.	The user provides the clock on the CLKUSR pin. This clock can synchronize the initialization of multiple devices. The clock should be supplied when the last data byte is transferred. Supplying the clock during configuration will not affect the configuration process. The operation of the CLKUSR pin during user mode is selected in the software.
User-supplied start-up clock (APEX II, APEX 20K, and Mercury devices only).	To begin initialization, the device must be clocked 40 times for APEX II or APEX 20K devices or 136 times for Mercury devices after all data is transferred. CONF_DONE goes high after the initialization process begins. The device can be initialized by the internal oscillator or by external clocks provided on DCLK or CLKUSR.	The device's internal oscillator supplies the start-up clock.	The user provides the clock on the CLKUSR or DCLK pin. The Quartus II software specifies which pin is used. This clock synchronizes the initialization of multiple devices. The clock should be supplied when the last data byte is transferred. Supplying a clock on CLKUSR will not affect the configuration process. The Quartus II software specifies the CLKUSR pin's operation mode. When using a configuration device, you can use CLKUSR to synchronize the initialization; DCLK can be used in passive configuration only.

Table 28. APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K & FLEX 6000 Configuration Option Bits (Part 2 of 3)

Device Option	Option Usage	Default Configuration (Option Off)	Modified Configuration (Option On)
Auto-restart configuration on frame error	If a data error occurs during configuration, you can choose how to restart configuration.	The configuration process stops until you direct the device to restart configuration. The nSTATUS pin is driven low when an error occurs. When nCONFIG is pulled low and then high, the device begins to reconfigure.	The configuration process restarts automatically. The nSTATUS pin drives low and releases. The nSTATUS pin is then pulled to V_{CC} by the pull-up resistor, indicating that configuration can restart. In the configuration device scheme, if the target device's nSTATUS pin is tied to the configuration device's OE pin, the nSTATUS reset pulse resets the configuration device automatically. The configuration device releases its OE pin (which is pulled high) and reconfiguration begins. If an error occurs during passive configuration, the device can be reconfigured without the system having to pulse nCONFIG. After nSTATUS goes high, reconfiguration can begin.
Release clears before tri-states	During configuration, the device I/O pins are tri-stated. During initialization, you choose the order for releasing the tri-states and clearing the registers.	The device releases the tri-states on its I/O pins before releasing the clear signal on its registers.	The device releases the clear signals on its registers before releasing the tri-states. You can use this option to allow the design to operate before it drives out, so all outputs do not start up low.
Enable chip-wide reset	Enables a single pin to reset all device registers.	Chip-wide reset is not enabled. The DEV_CLRn pin is available as a user I/O pin.	Chip-wide reset is enabled for all registers in the device. All registers are cleared when the DEV_CLRn pin is driven low.

Table 28. APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K & FLEX 6000 Configuration Option Bits (Part 3 of 3)

Device Option	Option Usage	Default Configuration (Option Off)	Modified Configuration (Option On)
Enable chip-wide output enable	Enables a single pin to control all device tri-states.	Chip-wide output enable is not enabled. The DEV_OE pin is available as a user I/O pin.	Chip-wide output enable is enabled for all device tri-states. After configuration, all user I/O pins are tri-stated when DEV_OE is low.
Enable INIT_DONE output	Enables a pin to drive out a signal when the initialization process is complete and the device has entered user mode.	The INIT_DONE signal is not available. The INIT_DONE pin is available as a user I/O pin.	The INIT_DONE signal is available on the open-drain INIT_DONE pin. This pin drives low during configuration. After initialization, it is released and pulled high externally. The INIT_DONE pin must be connected to a 1-kΩ pull-up resistor. If the INIT_DONE output is used, the INIT_DONE pin cannot be used as a user I/O pin.
Enable JTAG support (FLEX 6000 devices only. In APEX II, APEX 20K, Mercury, and FLEX 10K devices, JTAG support is always enabled.)	Enables post-configuration JTAG boundary-scan testing support in FLEX 6000 devices.	JTAG boundary-scan testing can be performed before configuration; however, it cannot be performed during or after configuration. During JTAG boundary-scan testing, nCONFIG must be held low.	JTAG boundary-scan testing can be performed before or after device configuration via the four JTAG pins (TDI, TDO, TMS, and TCK); however, it cannot be performed during configuration. When JTAG boundary-scan testing is performed before device configuration, nCONFIG must be held low.

Device Configuration Pins

Table 29 summarizes the APEX 20K, FLEX 10K, and FLEX 6000 device configuration pins.

Table 29. Pin Functions (Part 1 of 5)

Pin Name	Device Family	User Mode	Configuration Scheme	Pin Type	Description
MSEL0 MSEL1	APEX II APEX 20K Mercury ACEX 1K FLEX 10K	–	All	Input	2-bit configuration input. Sets the APEX 20K and FLEX 10K device configuration scheme. See Table 2 on page 4.
MSEL	FLEX 6000	–	All	Input	1-bit configuration input. Sets the FLEX 6000 device configuration scheme. See Table 4 on page 5.
nSTATUS	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	–	All	Bidirectional open-drain	The device drives nSTATUS low immediately after power-up and releases it within 5 µs. (When using a configuration device, the configuration device holds nSTATUS low for up to 200 ms.) The nSTATUS pin must be pulled up to V_{CC} with a 1-kΩ resistor (10-kΩ for APEX 20KE or APEX 20KC devices). If an error occurs during configuration, nSTATUS is pulled low by the target device. If an external source drives the nSTATUS pin low during configuration or initialization, the target device enters an error state. Driving nSTATUS low after configuration and initialization does not affect the configured device. However, if a configuration device is used, driving nSTATUS low will cause that device to attempt to configure the APEX or FLEX device.
nCONFIG	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	–	All	Input	Configuration control input. A low transition resets the target device; a low-to-high transition begins configuration. All I/Os go tri-state when setting nConfig low.

Table 29. Pin Functions (Part 2 of 5)

Pin Name	Device Family	User Mode	Configuration Scheme	Pin Type	Description
CONF_DONE	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	–	All	Bidirectional open-drain	Status output. The target device drives the CONF_DONE pin low before and during configuration. Once all configuration data is received without error and the initialization clock cycle starts, the target device releases CONF_DONE. Status input. After all data is received and CONF_DONE goes high, the target device initializes and enters user mode. The CONF_DONE pin must be pulled to V_{CC} with a 1-kΩ resistor (10-kΩ for APEX 20KE or APEX 20KC devices). An external source can drive this pin low to delay the initialization process, except when configuring with a configuration device. Driving CONF_DONE low after configuration and initialization does not affect the configured device.
DCLK	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	–	Configuration device PS PPS	Input	Clock input used to clock data from an external source into the target device. In PSA or PPA schemes, DCLK should be held high to prevent this pin from floating.
nCE	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	–	All	Input	Active-low chip enables. The nCE pin activates the device with a low signal to allow configuration and should be tied low for single device configuration. The nCE pin must be held low during configuration, initialization, and user mode.
nCEO	APEX II APEX 20K Mercury ACEX 1K FLEX 10K	–	Multi-device	Output	Output that drives low when device configuration is complete. During multi-device configuration, this pin feeds a subsequent device's nCE pin.
	FLEX 6000	I/O			

Table 29. Pin Functions (Part 3 of 5)

Pin Name	Device Family	User Mode	Configuration Scheme	Pin Type	Description
nWS	APEX II APEX 20K Mercury ACEX 1K FLEX 10K	I/O	PPA	Input	Write strobe input. For APEX 20K and FLEX 10K devices, a low-to-high transition causes the device to latch a byte of data on the DATA[7..0] pins. For FLEX 6000 devices, a low-to-high transition causes the device to latch a bit of data on the DATA pin.
	FLEX 6000	I/O	PSA		
nRS	APEX II APEX 20K Mercury ACEX 1K FLEX 10K	I/O	PPA	Input	Read strobe input. For APEX 20K and FLEX 10K devices, a low input directs the device to drive the RDYnBSY signal on the DATA7 pin. For FLEX 6000 devices, a low input directs the device to drive the RDYnBSY signal on the DATA pin. If the nRS pin is not used, it should be tied high.
	FLEX 6000	I/O	PSA		
RDYnBSY	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	I/O	PPA PSA	Output	Ready output. A high output indicates that the target device is ready to accept another data byte. A low output indicates that the target device is not ready to receive another data byte.
nCS CS	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	I/O	PPA PSA	Input	Chip-select inputs. A low on nCS and a high on CS select the target device for configuration. If only one chip-select input is used, the other must be tied to the active value (e.g., nCS can be tied to ground if CS is used). The nCS and CS pins must be held active during configuration and initialization.
CLKUSR	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	I/O	All	Input	Optional user-supplied clock input. Synchronizes the initialization of one or more devices.
DATA	FLEX 6000	–	Configuration device PS PSA	Input	Data inputs. Bit-wide configuration data is presented to the FLEX 6000 device on the DATA pin. In the PSA configuration scheme, the DATA pin presents the RDYnBSY signal after the nRS signal has been strobed, which may be more convenient for microprocessors than using the RDYnBSY pin.

Table 29. Pin Functions (Part 4 of 5)

Pin Name	Device Family	User Mode	Configuration Scheme	Pin Type	Description
DATA[7..1]	APEX II APEX 20K Mercury ACEX 1K FLEX 10K	I/O	PPS PPA	Inputs	Data inputs. Byte-wide configuration data is presented to the target device on DATA[7..0].
DATA0	APEX II APEX 20K Mercury ACEX 1K FLEX 10K	–	Configuration device PS PPA PPS	Input	Data input. In serial configuration modes, bit-wide configuration data is presented to the target device on the DATA0 pin.
DATA7	APEX II APEX 20K Mercury ACEX 1K FLEX 10K	I/O	PPA	Output	In the PPA configuration scheme, the DATA7 pin presents the RDYnBSY signal after the nRS signal has been strobed, which may be more convenient for microprocessors than using the RDYnBSY pin.
INIT_DONE	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	I/O	All	Output open-drain	Status pin. Can be used to indicate when the device has initialized and is in user mode. The INIT_DONE pin drives low during configuration. Before and after configuration, the INIT_DONE pin is released and is pulled to V_{CC} by an external pull-up resistor. Because INIT_DONE is tri-stated before configuration, it is pulled high by the external pull-up resistor. Thus, the monitoring circuitry must be able to detect a low-to-high transition. This option is set in the MAX+PLUS II or Quartus II software.
DEV_OE	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	I/O	All	Input	Optional pin that allows the user to override all tri-states on the device. When this pin is driven low, all I/Os are tri-stated; when this pin is driven high, all I/Os behave as programmed. This option is set in the Quartus II or MAX+PLUS II software.
DEV_CLRn	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	I/O	All	Input	Optional pin that allows you to override all clears on all device registers. When this pin is driven low, all registers are cleared; when this pin is driven high, all registers behave as programmed. This option is set in the Quartus II or MAX+PLUS II software.

Table 29. Pin Functions (Part 5 of 5)

Pin Name	Device Family	User Mode	Configuration Scheme	Pin Type	Description
TDI	APEX II APEX 20K Mercury ACEX 1K FLEX 10K FLEX 6000	I/O or JTAG pins	All	Input	JTAG pins. When used as user I/O pins, JTAG pins must be kept stable before and during configuration. JTAG pin stability prevents accidental loading of JTAG instructions.
TDO				Output	
TMS				Input	
TCK				Input	

Device Configuration Files

Altera's Quartus II and MAX+PLUS II development tools can create one or more configuration and programming files to support the configuration schemes discussed in this application note. This section describes these files.

SRAM Object File (.sof)

You should use an SRAM Object File (**.sof**) during PS configuration when the data is downloaded directly from the Altera programming hardware with a MasterBlaster or ByteBlasterMV cable. For APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices, the Quartus II or MAX+PLUS II Compiler's Assembler module automatically creates the SOF for each device in your design. The Quartus II or MAX+PLUS II software controls the configuration sequence and automatically inserts the appropriate headers into the configuration data stream. All other configuration files are created from the SOF.

Programmer Object File (.pof)

A Programmer Object File (**.pof**) is used by the Altera programming hardware to program a configuration device. A POF is generated automatically when an APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, or FLEX 6000 project is compiled. For smaller FLEX devices (e.g., EPF10K20 devices), multiple POFs can fit into one configuration device; for larger devices (e.g., APEX 20K devices), multiple configuration devices may be required to hold the configuration data.

Raw Binary File (.rbf)

The Raw Binary File (**.rbf**) is a binary file, e.g., one byte of RBF data is 8 configured bits `10000101` (85 Hex), containing the configuration data. Data must be stored so that the least significant bit (LSB) of each data byte is loaded first. The converted image can be stored on a mass storage device. The microprocessor can then read data from the binary file and load it into the device. You can also use the microprocessor to perform real-time conversion during configuration. In the PPA and PPS configuration schemes, the target device receives its information in parallel from the data bus, a data port on the microprocessor, or some other byte-wide channel. In PS and PSA configuration schemes, the data is shifted in serially, LSB first.

The following steps explain how to create an RBF file for ACEX 1K, FLEX 10K or FLEX 6000 devices using the MAX+PLUS II software. You can follow a similar procedure in the Quartus II software to generate RBFs for APEX II, APEX 20K, and Mercury devices.

1. In the MAX+PLUS II Compiler or Programmer, choose the **Convert SRAM Object Files** command (File menu.)

2. In the **Convert SRAM Object Files** dialog box, specify which SOF files to combine and then select.*rbf (Sequential)* in the *File Format* box. Click **OK**.

For more information on creating RBFs, search for "RBF" in Quartus II or MAX+PLUS II Help.

Hexadecimal (Intel-Format) File (.hex)

A Hex File is an ASCII file in the Intel Hex format. This file is used by third-party programmers to program Altera's serial configuration devices. Hex Files are also used to program parallel configuration devices with third-party programming hardware. You can use parallel configuration devices in the PPS, PPA, or PSA configuration schemes, in which a microprocessor uses the parallel configuration device as the data source.

The following steps explain how to create a Hex file for ACEX 1K, FLEX 10K, or FLEX 6000 devices using the MAX+PLUS II software. You can follow a similar procedure in the Quartus II software to generate Hex files for APEX II, APEX 20K, or Mercury devices.

1. In the MAX+PLUS II Programmer or Compiler, choose the **Convert SRAM Object Files** command (File menu).

2. In the **Convert SRAM Object Files** dialog box, specify which SOF files to combine and then select.*hex* in the *File Format* box. Click **OK**.

For more information on creating Hex Files, search for "Hex File" in Quartus II or MAX+PLUS II Help.

Tabular Text File (.ttf)

The Tabular Text File (**.ttf**) is a tabular ASCII file that provides a comma-separated version of the configuration data for the PPA, PPS, PSA, and bit-wide PS configuration schemes. In some applications, the storage device containing the FLEX 10K or FLEX 6000 configuration data is neither dedicated to nor connected directly to the target device. For example, a configuration device can also contain executable code for a system (e.g., BIOS routines) and other data. The TTF allows you to include the configuration data as part of the microprocessor's source code using the `include` or `source` commands. The microprocessor can access this data from a configuration device or mass-storage device and load it into the target device. A TTF can be imported into nearly any assembly language or high-level language compiler.

The following steps explain how to create a TTF file for FLEX 10K or FLEX 6000 devices using the MAX+PLUS II software.

1. In the MAX+PLUS II Programmer or Compiler, choose the **Convert SRAM Object Files** command (File menu).

2. In the **Convert SRAM Object Files** dialog box, specify which SOF files to combine and then select *.ttf (Sequential)* in the *File Format* box. Click **OK**.

For more information on creating TTFs, search for "TTF" in Quartus II or MAX+PLUS II Help.

Serial Bitstream File (.sbf)

A Serial Bitstream File (**.sbf**) is used in PS schemes to configure FLEX 10K and FLEX 6000 devices in-system with the BitBlaster cable.

SBFs are supported by the MAX+PLUS II software only.

The following steps explain how to create an SBF file for FLEX 10K or FLEX 6000 devices using the MAX+PLUS II software.

1. In the MAX+PLUS II Programmer or Compiler, choose the **Convert SRAM Object Files** command (File menu).

2. In the **Convert SRAM Object Files** dialog box, specify which SOF files to combine and then select *.sbf (Sequential)* in the *File Format* box. Click **OK**.

For more information on creating SBFs, search for "SBF" in MAX+PLUS II Help.

Jam File (.jam)

A Jam File is an ASCII text file in the Jam device programming language that stores device programming information. These files are used to program, verify, and blank-check one or more devices in the Quartus II or MAX+PLUS II Programmer or in an embedded processor environment.

Jam Byte-Code File (.jbc)

A Jam Byte-Code File (**.jbc**) is a binary file of a Jam File in a byte-code representation. JBC files store device programming information used to program, verify, and blank-check one or more devices in the MAX+PLUS II Programmer or in an embedded processor environment.

Programming Configuration Devices

You can configure APEX 20K, FLEX 10K, and FLEX 6000 devices using data stored in either a configuration device or the Quartus II or MAX+PLUS II software.

Configuration with a Configuration Device

You program configuration devices using the Quartus II or MAX+PLUS II software, the Master Programming Unit (MPU), and the appropriate configuration device programming adapter. Table 30 shows which programming adapter to use with each configuration device.

Table 30. Programming Adapters

Device	Package	Adapter
EPC16	88-pin Ultra FineLine BGA 100-pin PQFP	–
EPC8	100-pin PQFP	–
EPC2	20-pin J-Lead 32-pin TQFP	PLMJ1213 PLMT1213
EPC1	8-pin DIP 20-pin J-Lead	PLMJ1213 PLMJ1213
EPC1441	8-pin DIP 20-pin J-Lead 32-pin TQFP	PLMJ1213 PLMJ1213 PLMT1064

The following steps explain how to program Altera configuration devices using the MAX+PLUS II software:

1. Open the MAX+PLUS II Programmer.
2. Load the appropriate POF using the **Select Programming File** dialog box (File menu). By default, the Programmer loads the current project's POF. The *Device* field displays the appropriate device for the current programming file.
3. Insert a blank configuration device into the programming adapter's socket.
4. Click the **Program** button.

After successful programming, you can place the configuration device on the PCB to configure an APEX 20K, FLEX 10K, or FLEX 6000 device.

For more information on configuration devices, see the *Configuration Devices for APEX & FLEX Devices Data Sheet*.

APEX 20KE Power Sequencing

Many Altera devices include the MultiVolt™ feature, where there are separate power supplies for powering the logic array and the I/O banks. For APEX II, APEX 20KC, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices, the power supplies can be powered up in either order. However, for APEX 20KE devices, the following guidelines explain how to manage device power sequencing. These guidelines apply to all configuration schemes.

The APEX 20KE logic array and I/O logic can operate on different power supplies. V_{CCINT} powers the logic array, and each I/O bank has a separate V_{CCIO} supply.

These guidelines will allow you to configure APEX 20KE devices upon power-up as well as recover from power brown-out conditions. Specifically, V_{CCINT} can decrease to any voltage, and the device will successfully reconfigure when power is restored. During the brown-out condition, APEX 20KE devices may lose configuration if V_{CCINT} falls below the minimum V_{CCINT} device specification. If V_{CCINT} drops below the specified operating range, the APEX 20KE device resets and drives nSTATUS low. When V_{CCINT} is in the specified operating range, nSTATUS will be released, initiating configuration.

Use different connections between APEX 20KE devices and configuration devices, depending on the order that the power supplies are powered up.

Power Sequencing Considerations

Use the guidelines in one of the following sections to ensure a successful power-up and configuration.

- V_{CCINT} is powered before V_{CCIO}.
- V_{CCINT} is powered during or after the configuration device's power on reset (POR) time.

Altera has enhanced the APEX II and APEX 20KC devices, so you do not need to follow these guidelines for those devices. However, a system designed for an APEX 20KE device can successfully configure an APEX II or APEX 20KC device.

V_{CCINT} is Powered Before V_{CCIO}

If V_{CCINT} is powered before V_{CCIO}, the nCONFIG signal must be held low for the entire time that the two supplies are powering up to their specified operating ranges. The recommendations in this section should also be followed for applications where the sequence of the power supplies is unknown (e.g., hot-socketing applications).

For more details on APEX 20KE and configuration device voltage supply specifications, refer to the *APEX 20K Programmable Logic Device Family Data Sheet* and the *Configuration Devices for APEX & FLEX Devices Data Sheet*, respectively.

Use one of the following methods to hold nCONFIG low:

Use a power-monitoring circuit on the board. This circuit will drive nCONFIG low when V_{CCINT} and V_{CCIO} are out of range, and then release nCONFIG. nCONFIG can then be externally pulled up when V_{CCINT} returns to a normal operating level. National Semiconductor offers a small power-on reset circuit (part number LP3470) for a power monitor.

Many voltage regulators offer a power-good signal that can be used to hold nCONFIG low during power-up. If there are multiple regulators, use the power-good signal on the regulator that powers up last. If the power sequence is unknown, the power-good signals can be ANDed in a discrete device.

A microcontroller or intelligent host can externally drive nCONFIG low while both the V_{CCINT} and V_{CCIO} supplies are being powered.

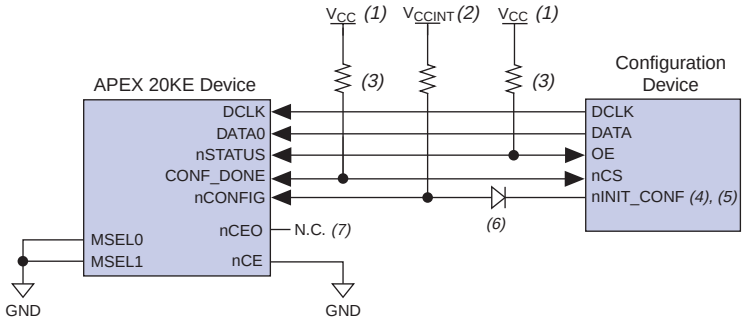
V_{CCINT} is Powered During or After Configuration Device's POR Time (V_{CCINT} is powered after V_{CCIO})

Use this configuration method when V_{CCINT} is powered up after the configuration device has exited POR (released nSTATUS/OE signal). The configuration device will exit POR 200 ms after power-up. This configuration method applies to Altera configuration devices. If the design uses another configuration device, ensure that the configuration controller toggles nCONFIG at the beginning of configuration.

If the configuration device is powered by the same 3.3-V supply as V_{CCIO}, and V_{CCINT} is powered up after V_{CCIO}, then the APEX 20KE device will successfully power-up and configure.

Figure 44 shows the board connections used to support the initiate configuration instruction with the nINIT_CONF pin. The nINIT_CONF pin is open-drain and has an internal pull-up resistor (typically 1 kΩ) internally connected to the 3.3-V supply. This resistor is always enabled, so a diode is necessary to isolate the 1.8-V and 3.3-V supplies.

Figure 44. Powering 3.3-V Configuration Device First When Configuring an APEX 20KE Device



Notes to Figure 44:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(2) To ensure a successful configuration between APEX 20KE and configuration devices in all possible power-up sequences, pull nCONFIG up to V_{CCINT}.

(3) All pull-up resistors are 10 kΩ. The configuration device's OE and nCS pins have internal user-configurable pull-up resistors. If internal pull-up resistors are used, do not use external pull-up resistors on these pins.

(4) The nINIT_CONF pin is available on EPC2, EPC8, and EPC16 devices only. If nINIT_CONF is not available (i.e., EPC1 and EPC1441 devices) or not used, nCONFIG must be pulled to V_{CCINT} through a 10-kΩ resistor.

(5) The nINIT_CONF pin has an internal pull-up resistor which is always active in EPC16, EPC8, and EPC2 devices. nCONFIG must be connected to V_{CCINT} through a 10-kΩ resistor.

(6) To isolate the 1.8-V and 3.3-V power supplies when configuring APEX 20KE devices, add a diode between the APEX 20KE device's nCONFIG pin and the configuration device's nINIT_CONF pin. Select a diode with a threshold voltage (V_T) less than or equal to 0.7 V. The diode will make the nINIT_CONF pin an open-drain pin; the pin will only be able to drive low or tri-state.

(7) The nCEO pin is left unconnected for single-device configuration.

V_{CCINT} is Powered After V_{CCIO}

Nothing special is required if V_{CCINT} is powered after V_{CCIO}.

Configuration Reliability

The APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 architectures have been designed to minimize the effects of power supply and data noise in a system, and to ensure that the configuration data is not corrupted during configuration or normal user-mode operation. A number of circuit design features are provided to ensure the highest possible level of reliability from this SRAM technology.

Cyclic redundancy code (CRC) circuitry is used to validate each data frame (i.e., sequence of data bits) as it is loaded into the target device. If the CRC generated by the device does not match the data stored in the data stream, the configuration process is halted, and the nSTATUS pin is pulled and held low to indicate an error condition. CRC circuitry ensures that noisy systems will not cause errors that yield an incorrect or incomplete configuration.

The device architectures also provide a very high level of reliability in low-voltage brown-out conditions. These device's SRAM cells require a certain V_{CC} level to maintain accurate data. This voltage threshold is significantly lower than the voltage required to activate the device's POR circuitry. Therefore, the target device stops operating if the V_{CC} starts to fail, and indicates an operation error by pulling and holding the nSTATUS pin low. The device must then be reconfigured before it can resume operation as a logic device. In active configuration schemes in which nCONFIG is tied to V_{CC}, reconfiguration begins as soon as V_{CC} returns to an acceptable level. The low pulse on nSTATUS resets the configuration device by driving OE low. In passive configuration schemes, the host system starts the reconfiguration process.

These device features ensure that APEX II, APEX 20K, Mercury, ACEX 1K, FLEX 10K, and FLEX 6000 devices have the highest possible reliability in a wide variety of environments, and provide the same high level of system reliability that exists in other Altera PLDs.

Board Layout Tips

Even though the DCLK signal (used in configuration device, PS, and PPS configuration schemes) is fairly low-frequency, it drives edge-triggered pins on the APEX II, APEX 20K, Mercury, FLEX 10K, or FLEX 6000 device. Therefore, any overshoot, undershoot, ringing, or other noise can affect configuration. When designing the board, lay out the DCLK trace using the same techniques as laying out a clock line, including appropriate buffering. If more than five devices are used, Altera recommends using buffers to split the fan-out on the DCLK signal.

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