AP-398

APPLICATION NOTE

Designing with the 28F016XS

KEN MCKEE
TECHNICAL MARKETING
ENGINEER

April 1995

Order Number: 292147-003

Additional copies of this document or other Intel literature may be obtained from:

Intel Corporation
Literature Sales
P.O. Box 7641
Mt. Prospect, IL 60056-7641

or call 1-800-879-4683

 CG-041493

CONTENTS

ADVANCE INFORMATION

1.0 INTRODUCTION

The interfacing concepts discussed in this document are based on preliminary 28F016XS specifications. Please contact your Intel or distribution sales office for up-to-date information. Do not finalize a design based on the specifications in this document.

The 28F016XS is an extremely high performance 16-Mbit memory component, organized as either 2 Mbytes or 1 Mword. The 28F016XS contains sixteen 128-Kbyte (64 Kword) blocks. Each block is separately erasable, lockable and capable of 1 million write/erase cycles by providing wear-leveling algorithms and graceful block retirement.

28F016XS's enhancements over first generation 16-Mbit flash memories include:

- Synchronous pipelined read interface providing significantly improved read performance
- SmartVoltage technology

The 28F016XS's synchronous pipelined read interface delivers highest read performance when interfaced to a microprocessor with a burst or pipelined bus, such as the i960® and Intel486™ microprocessors. The 28F016XS delivers equivalent or better read performance than DRAM, given an optimized interface.

The enhanced read performance capability of the 28F016XS eliminates the need of code shadowing from nonvolatile memory (ROM, EPROM, etc.) to DRAM for increased system performance. The 28F016XS eliminates the need for this multi-component memory model, enabling direct code execution out of nonvolatile memory with its high read performance.

This application note will discuss the concepts involved in interfacing to the 28F016XS, illustrating that it requires minimal glue logic and is compatible with a wide range of processors and buses.

2.0 OPERATIONAL FUNDAMENTALS OF THE 28F016XS

The 28F016XS, in read mode, is a fully synchronous component with a maximum operating frequency of 66 MHz at 5V V_{CC}. Write operations to the 28F016XS are asynchronous, similar to traditional flash memories such as the 28F016SA. Flash memory is read, erased and written in system via the local processor.

2.1 28F016XS Pinout Comparison to the 28F016SA

The 28F016XS's pinout is very similar to the pinouts of the 28F016SA/SV (see Figure 1). All devices use the 56-Lead TSOP package.

The 28F016SA uses the 3/5# pin (pin 1) to configure the flash memory for operation at 3.3V or 5.0V V_{CC}. The 28F016XS and 28F016SV use an internal detector connected to V_{CC} to accomplish this same function. The 3/5# pin is no longer needed on the 28F016XS and 28F016SV, and pin 1 has been renamed NC ("no connect"). This pin may be driven to V_{IL} or V_{IH}, or may be left unconnected.

The 28F016XS also incorporates two new pins in comparison to the 28F016SA/SV: CLK (pin 29) and ADV# (pin 30). These two pins control the 28F016XS's synchronous pipelined read interface.

CLK

CLK provides the fundamental timing and internal operating read frequency for the 28F016XS (maximum frequency of 66 MHz at 5.0V ± 0.5V V_{CC} and 50 MHz at 3.3V ± 0.3V V_{CC}). CLK latches input addresses on its rising edge in conjunction with ADV#, times out the SFI Configuration (configurable via the Configure Device Command), and synchronizes device outputs. CLK can be slowed or stopped without loss of data synchronization. CLK is ignored during write operations.

ADVANCE INFORMATION

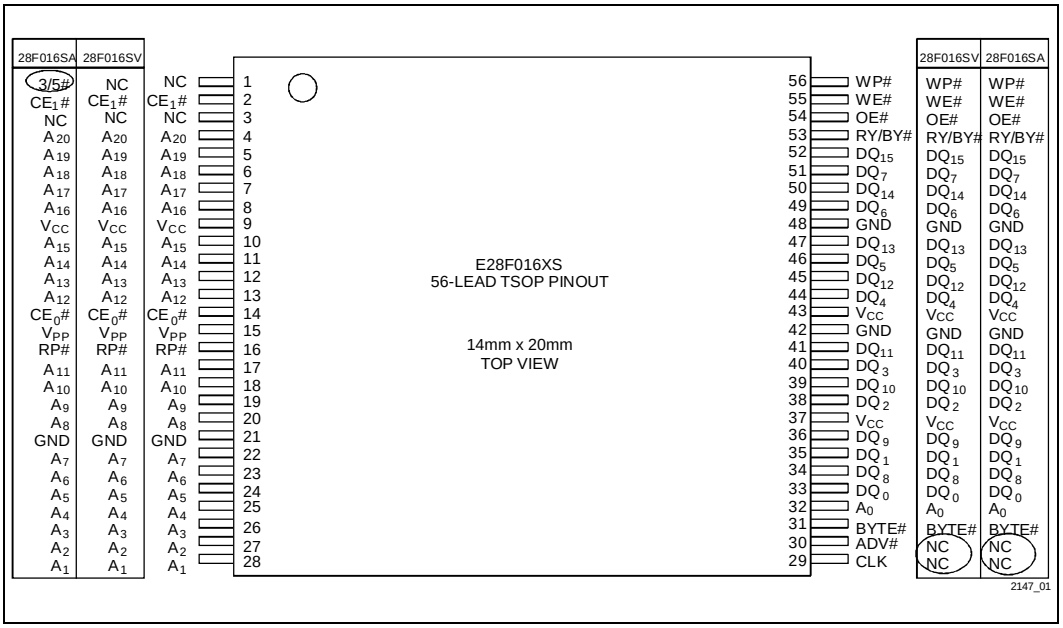


Figure 1. 28F016XS Pinout Configuration Compared to the 28F016SA/SV

ADV#

Address Valid (ADV#) indicates that a valid address is present on the 28F016XS's address pins. ADV# sensed active low on the rising edge of CLK latches the address into the 28F016XS, which initiates a read access. ADV# is ignored by the 28F016XS during write cycles.

2.2 Enhanced Read Capability

Given an optimized interface, the 28F016XS's read performance is up to 2x higher than traditional asynchronous flash memory. The 28F016XS's synchronous pipelined read interface is capable of executing multiple read accesses in parallel. This parallel execution capability more than doubles read performance.

Memory Address Decoding

Figure 2 illustrates the 28F016XS's memory address decoding. Addresses A_{20-1} select a 16-bit location within the 28F016XS's memory array. Address A_1 makes the bank selection, even or odd bank. Byte selection, in x8 mode (BYTE# = V_{IL}), is based on the value of address A_0. In x16 mode (BYTE# = V_{IH}), the 28F016XS does not use A_0.

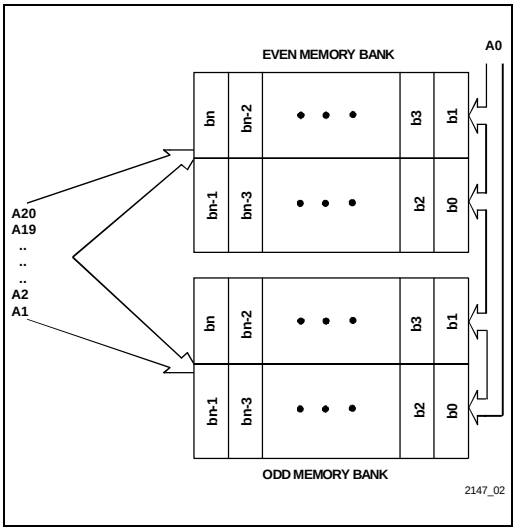
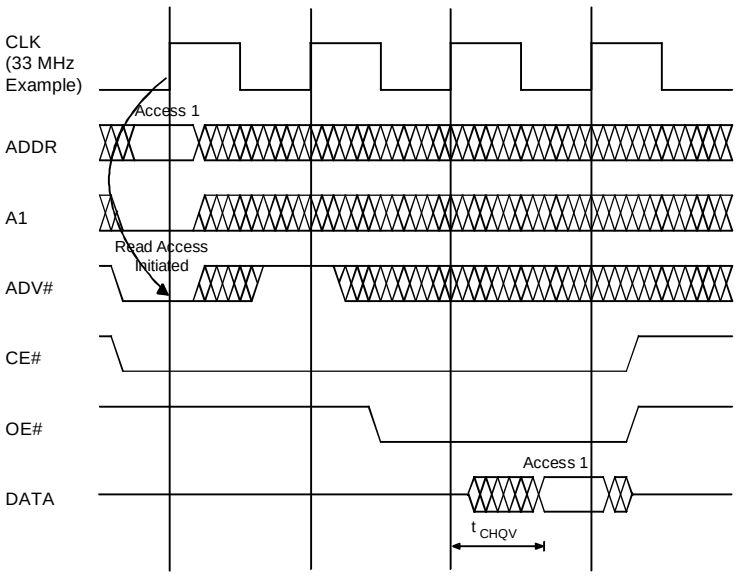


Figure 2. Memory Address Decoding Illustrating A_0 Selecting the Byte Location and A_1 Selecting the Even or Odd Bank

ADVANCE INFORMATION



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 3. Initiating a Single Read Access Waveform (28F016XS-15, SFI Configuration = 2)

Executing a Read Access Cycle

In read mode, ADV# and CLK together initiate read accesses when the device is selected (Figure 3). A read cycle is initiated and addresses are latched when the 28F016XS senses ADV# low on a rising CLK edge. After a read cycle begins and the SFI Configuration value has elapsed, data is latched and begins driving on the output pins. Valid data is guaranteed t_{CHQV} after the elapse of the SFI Configuration value.

Refer to Section 2.3 for further information about the SFI Configuration.

Consecutive Read Accesses

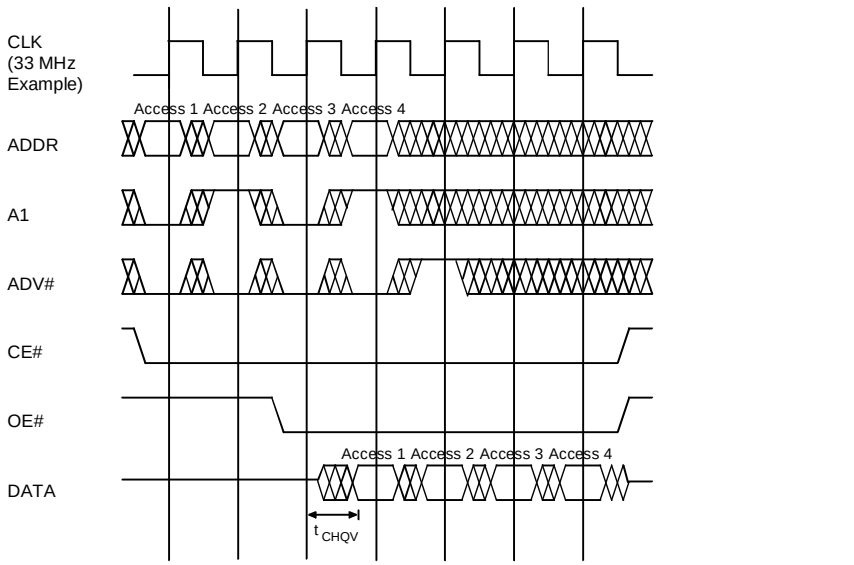
Consecutive accesses to the 28F016XS, from the processor perspective, occur in either an *Alternating-A_1* or a *Same-A_1* pattern. Of the two sequences, consecutive *Alternating-A_1* accesses offer highest read performance.

Consecutive *Alternating-A_1* accesses change the value of address A_1 between consecutive read cycles. As Figure 4 illustrates, up to three accesses can be initiated before data from the first access is valid on the output pins. Consecutive *Alternating-A_1* accesses allow multiple accesses to the 28F016XS to occur in parallel, effectively filling the 28F016XS's internal pipeline.

Consecutive *Same-A_1* accesses retain the same address A_1 value between consecutive read cycles. This pattern allows up to two accesses to be initiated before data from the first accesses is read. The SFI Configuration value must elapse before a second access can begin (see Section 2.3).

Refer to Sections 4.2 and 4.3, the *Alternating-A_1* and *Same-A_1* access rules, for minimum delays between consecutive read accesses. The *Alternating-A_1* and *Same-A_1* access rules both apply to consecutive *Alternating-A_1* accesses, while only the *Same-A_1* rule applies to consecutive *Same-A_1* accesses.

ADVANCE INFORMATION



NOTE:
Refer to 28F016XS datasheet for timing specifications.

Figure 4. Synchronous Pipelined Read Waveform Example (28F016XS-15, SFI Configuration = 2, Consecutive *Alternating-A$_1$* Access)

2.3 SFI Configuration

The SFI Configuration (Table 1) optimizes the 28F016XS for a wide range of input CLK frequencies. After a read access begins, the 28F016XS will begin latch data on its output pins after a CLK count corresponding to the SFI Configuration has elapsed.

The 28F016XS default SFI Configuration is 4 after power-up or return from deep power-down mode (RP# = V_{IL}), allowing system boot from the 28F016XS, if desired, at any CLK frequency up to 66 MHz at 5.0V ± 0.5V V_{CC} (50 MHz at 3.3V ± 0.3V V_{CC}). SFI Configurations are retained if the 28F016XS is put in sleep mode via a Sleep or Abort command. An optimized SFI Configuration for a given frequency (Table 1) enables the highest read performance. The SFI Configuration is updated via the Device Configuration command. SFI Configurations can range from 1 to 4.

The 28F016XS is fully synchronous. CLK can be slowed or stopped at any time within a series of accesses without loss of data synchronization.

Table 1. SFI Configuration Correspondence to CLK Frequency

28F016XS-15	
Frequency (MHz)	SFI Configuration
66 (and below)	4
50 (and below)	3
33 (and below)	2
16.7 (and below)	1

28F016XS-20	
Frequency (MHz)	SFI Configuration
50 (and below)	4
37.5 (and below)	3
25 (and below)	2
12.5 (and below)	1

ADVANCE INFORMATION

Table 1. SFI Configuration Correspondence to CLK Frequency (Continued)

28F016XS-25	
Frequency (MHz)	SFI Configuration
40 (and below)	4
30 (and below)	3
20 (and below)	2
10 (and below)	1

NOTE:
SFI Configurations other than those shown in Table 1 produce spurious results and should not be used.

3.0 CPU/BUS COMPATIBILITY WITH 28F016XS

The 28F016XS's synchronous read interface pipelines up to three accesses at time. This multiple execution capability makes the 28F016XS highly compatible with CPUs and buses that employ an Intel or linear burst, or pipelined bus.

Burst Bus

A burst bus executes a series of accesses in retrieving data from the memory subsystem. Accessing data consecutively, a processor or bus employing an Intel or linear burst order increments the lower address lines, effectively executing consecutive *Alternating-*A_1 accesses. Tables 2 and 3 illustrate the Intel and linear burst order, for a 4-access burst.

Table 2. Intel Burst Order Address A_{3-2} on a 32-Bit Processor

Intel Burst Order			
First Address	**Second Address**	**Third Address**	**Fourth Address**
00	01	10	11
01	00	11	10
10	11	00	01
11	10	01	00

Table 3. Linear Burst Order Address A_{3-2} on a 32-Bit Processor

Linear Burst Order			
First Address	**Second Address**	**Third Address**	**Fourth Address**
00	01	10	11
01	10	11	00
10	11	00	01
11	00	01	10

During a burst cycle, some CPUs themselves increment addresses, while others only supply the initial address, requiring peripheral logic to increment the address. CPUs that increment addresses, however, may not provide addresses quick enough to take full advantage of the 28F016XS's synchronous pipelined interface. Some processors wait for the completion of the first access before incrementing the address for the next access. Generating addresses within the interfacing logic, the 28F016XS is not forced to wait for the CPU. Therefore, several accesses can be initiated before the 28F016XS completes the initial access.

Figure 5 shows a block diagram of a four double-word burst CPU interface to the 28F016XS. This interface executes a 4 double word burst. Notice that addresses A_{2-1} are controlled by the interface. A multi-bit counter generates the addresses for the burst cycle.

Using the interface to increment addresses, the 28F016XS achieves highest read performance executing multiple *Alternating-A_1* accesses at the same time. For example, the 28F016XS-15, at 33 MHz and 5.0V V_{CC}, can deliver effective zero wait-state performance interfacing to a burst bus, after the initial pipeline fill.

Pipelined Bus

A pipelined bus activates the address and control signals for the next cycle before completing the current cycle. Pipelined buses have no defined access order; therefore, *Alternating-A_1* accesses are not guaranteed. The interface must guard against a possible mixture of consecutive *Alternating-A_1* and *Same-A_1* accesses. Figure 6 illustrates a pipelined bus interface to the 28F016XS.

In a pipelined interface, the system logic does not increment the 28F016XS's lower addresses. The system logic instead latches address A_1 and compares it to A_1 of the following cycle. Comparing A_1, the interface logic can identify *Alternating-A_1* and *Same-A_1* accesses, which directly informs the interface logic when it can initiate a read access to the 28F016XS. The *Alternating-A_1* and *Same-A_1* access rules define the minimum delay between consecutive accesses (see Section 4.2 and 4.3).

In the past, external latches were required to latch the next address and control signals. The 28F016XS eliminates this extra system overhead, latching the next address internally and initiating the next cycle prior to completing the current cycle. The 28F016XS's synchronous pipelined interface takes full advantage of a pipelined bus.

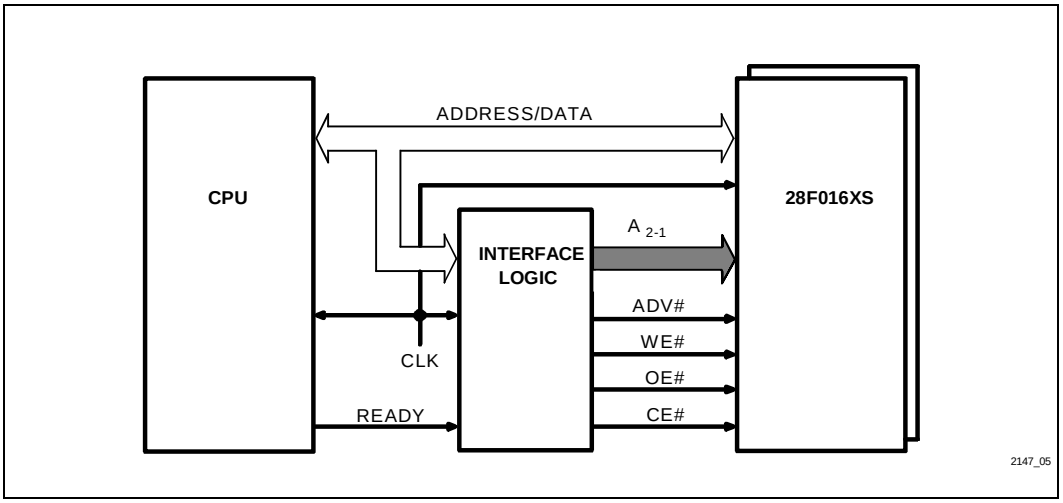


Figure 5. Burst Address Generation and Wait-State Control When Interfacing the 28F016XS to a Burst Processor

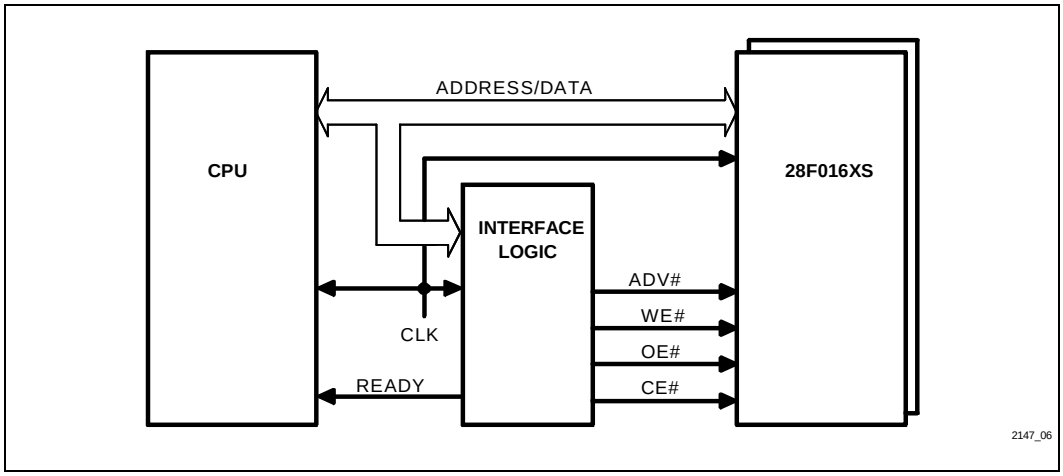


Figure 6. Comparing Past Address with Current Address to Determine Whether an Alternating-A_1 or Same-A_1 Access Occurs When Interfacing the 28F016XS to a Pipelined Bus Processor

4.0 INTERFACING TO THE 28F016XS

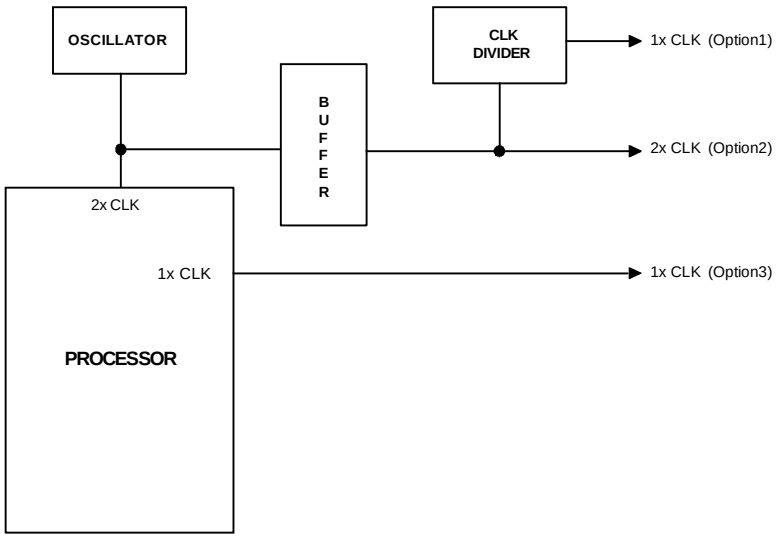
The 28F016XS can interface to a wide range of CPUs and bus architectures. Glue logic is minimal and the performance enhancements are significant. Below are key considerations to keep in mind when interfacing to the 28F016XS:

- Clocking Options
- *Alternating-A_1* Access Rule
- *Same-A_1* Access Rule
- Optimizing Read Performance in x8 Mode
- Consecutive Accesses across Bank Boundaries
- Handling Asynchronous Write Cycles
- System Boot-Up out of the 28F016XS

4.1 Clocking Options

In choosing a CLK option, keep in mind that the 28F016XS operates at optimum performance with a CLK frequency at an upper SFI Configuration boundary (50 MHz and 33 MHz are two of four SFI Configuration upper boundaries for the 28F016XS-15 at 5.0V V_{CC}). See Section 2.3 for information about the SFI Configuration.

For example, a processor running at 25 MHz with a 2x CLK input of 50 MHz provides both 25 MHz and 50 MHz CLK options (Figure 7). Using the 25 MHz CLK, the 28F016XS-15/-20 will begin driving data to the output pins two clocks (or 80 ns) after initiating a read cycle (SFI Configuration = 2 at 5.0 ± 0.5V V_{CC}). Using the 2x CLK, the 28F016XS-15 will begin driving data to the output pins in three clocks, or 60 ns (SFI Configuration = 3 at 5.0 ± 0.5V V_{CC}). The CLK frequency and SFI Configuration relationship affects the 28F016XS's read performance.



NOTE:
The buffer in Figure 7 eliminates loading effects on the oscillator

Figure 7. Different CLK Options when Interfacing to a Processor that Requires a 2x CLK Input

2x vs. 1x CLK

Figure 7 illustrates three possible CLK options when interfacing to a processor requiring a 2x CLK input (processors such as the i960 processor and Intel386™ processor are two examples). Depending on the processor frequency, the 2x CLK may be closer to an upper SFI Configuration boundary, thereby reducing the 28F016XS's access time. The higher frequency of the 2x CLK however, reduces the amount of available time for interface logic to meet the 28F016XS's set-up time (ADV#, CE_X# and OE# to rising CLK edge). A 1x CLK, on the other hand, has a longer period, relaxing the demands on the interfacing logic to meet the 28F016XS set-up time.

A general *"rule of thumb"* when choosing a CLK option: The higher the CLK frequency the higher the 28F016XS read performance but the faster the interfacing logic required.

Clock Generation and Synchronization

External clock generation and synchronization may be required when the interfacing processor uses a 2x CLK input and does not provide external access to the internally synchronized 1x CLK. Generating and synchronizing a 1x CLK from a 2x or 4x CLK can be accomplished with simple flip-flop logic. (Many processors, such as the Intel386 processor, synchronize their internal 1x CLK with the trailing edge of RESET.) Example PLD equations that generate and synchronize a 1x system output CLK are located in Appendix A.

Clock skew can arise depending upon the approach taken to derive the 1x CLK. Figure 8 (A) will have a maximum skew between the 2x and 1x clock of t_{CO1}. Figure 8 (B) will have minimal-to-no skew because both the 2x and 1x CLK outputs will have the same delay, t_{CO1}, through the EPLD. Figure 9 graphically illustrates the amount of skew the two different CLK generation circuits produce.

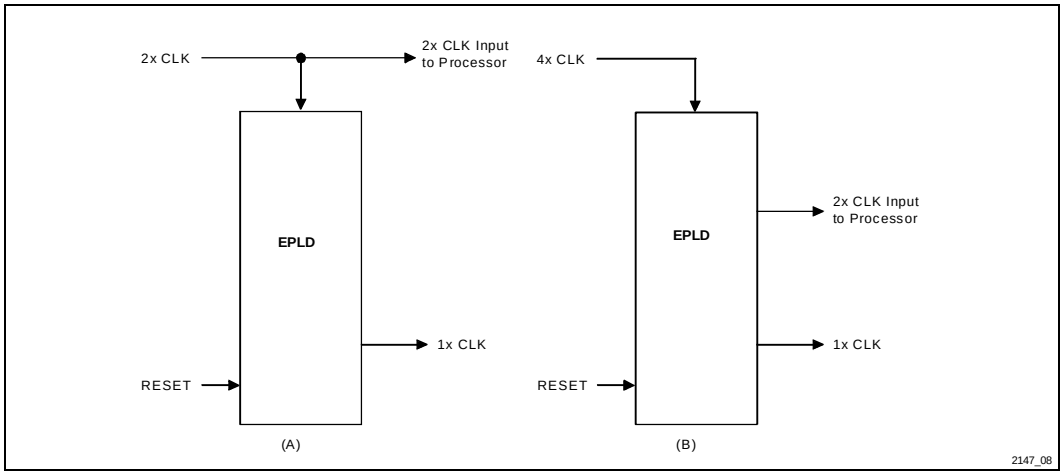


Figure 8. 1x CLK Configurations Using RESET for CLK Synchronization

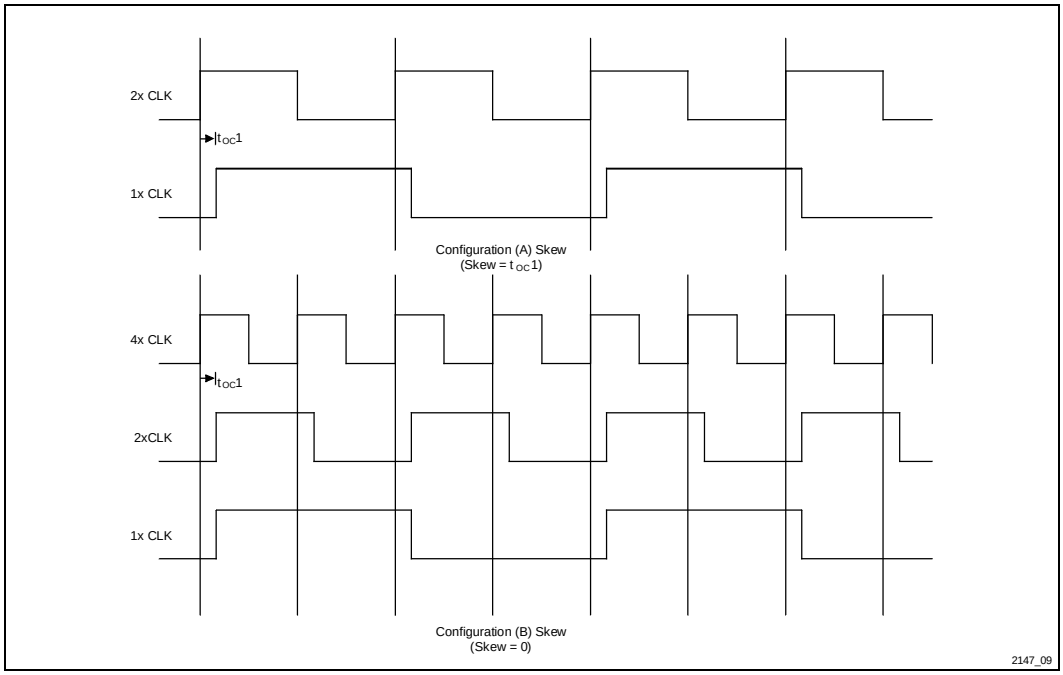
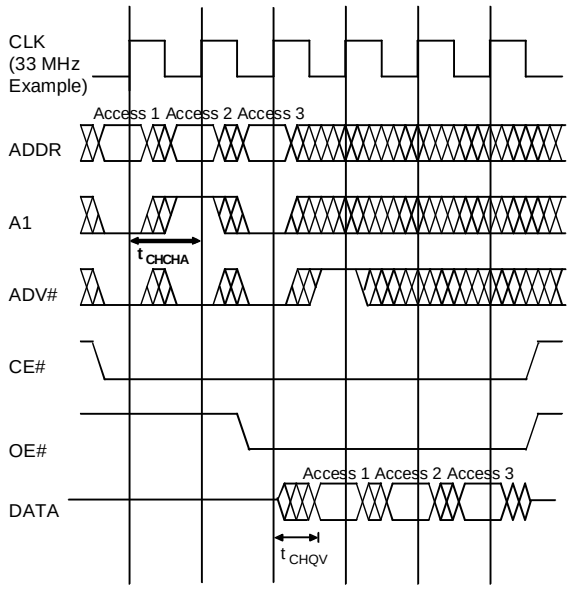


Figure 9. CLK Skew Produced by 1x CLK Configurations in Figure 8

ADVANCE INFORMATION



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 10. Alternating Access Rule, Illustrating t_{CHCHA} (28F016XS-15, SFI Configuration = 2, Consecutive *Alternating-A_1* Accesses)

4.2 *Alternating-A_1* Access Rule

The *Alternating-A_1* access rule (Figure 10) defines the minimum time required between consecutive accesses with different A_1 values. The *Alternating-A_1* access rule is:

$$t_{CHCHA} >= t_{CHQV} + (\text{System Propagation Delay}) + (\text{CPU Set-up Requirement})$$

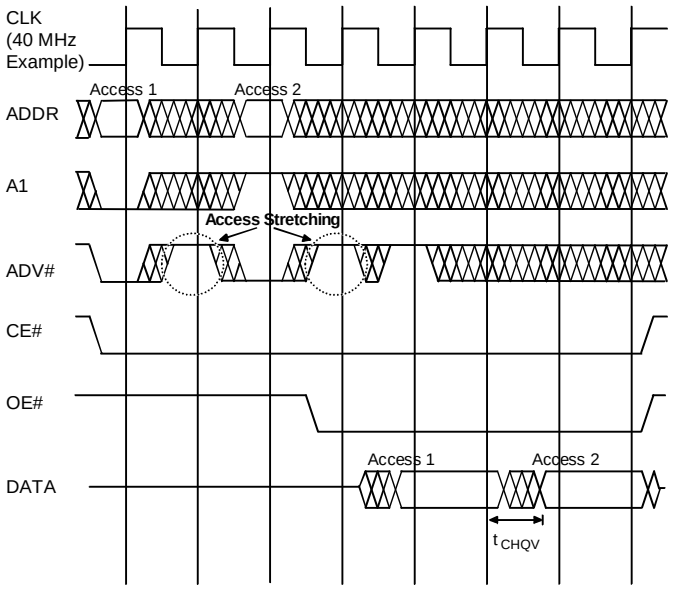
t_{CHCHA} is then rounded up to the nearest CLK period. For example, if the summation of t_{CHQV}, system delay and CPU set-up requirement equals 24 ns, and the CLK (33 MHz) period equals 30 ns, t_{CHCHA} is rounded up to 30 ns (1 CLK period).

When t_{CHCHA} equal one CLK period, an *Alternating-A_1* access can execute on the next rising CLK edge. In some designs, however, the sum of t_{CHQV}, system delay and CPU set-up requirement may exceed the CLK period, requiring access stretching.

Section 4.4 describes several interface examples demonstrating the *Alternating-A_1* access rule.

Access Stretching

Access stretching (Figure 11) extends the amount of time data is held between consecutive accesses. Intermediate cycles with ADV# disabled between initial access and subsequent access will extend the number of CLKs that the 28F016XS holds data. Intermediate cycles with ADV# disabled are required in all cases when t_{CHQV} exceeds the CLK period.



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 11. Access Stretching Example at 40 MHz (28F016XS-15, SFI Configuration = 3, Consecutive *Alternating-A_1* Accesses)

ADVANCE INFORMATION

4.3 *Same-A_1* Access Rule

The *Same-A_1* access rule (Figure 12) defines the minimum time required between accesses with the same address A_1 value. The *Same-A_1* access rule is:

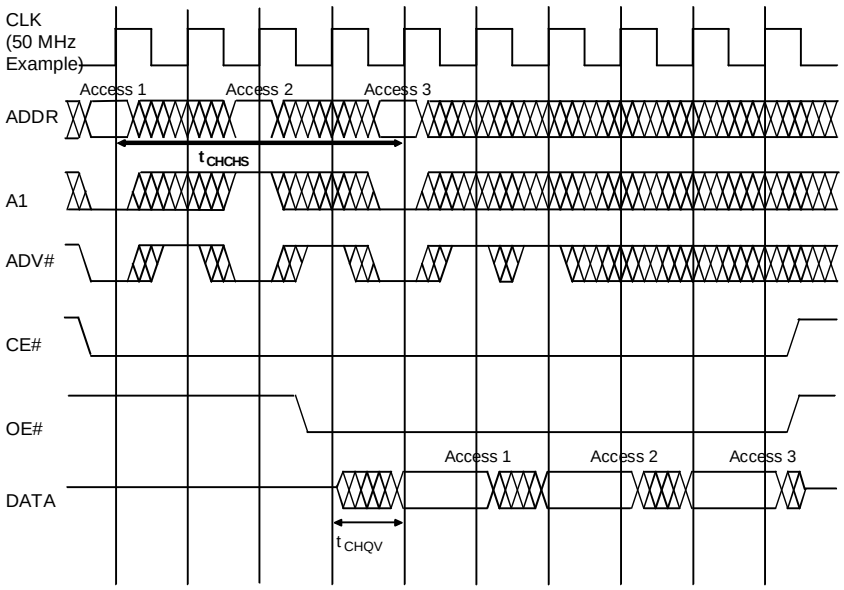
$$t_{CHCHS} >= \text{SFI Configuration}$$
or
$$t_{CHCHS} >= 2 \times t_{CHCHA}$$

t_{CHCHS} equals the largest value derived from the two above equations. For instance, a system operating at 50 MHz (20 ns period) with a SFI Configuration value set to 3 and a t_{CHCHA} equaling 40 ns

$$\text{SFI Configuration (3)} = 60 \text{ ns}$$
$$2 \times t_{CHCHA} = 80 \text{ ns}$$

will have a t_{CHCHS} equal to 80 ns.

Section 4.4 describes several interface examples that show the *Same-A_1* access rule.



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 12. *Same-A_1* Access Rule, Illustrating t_{CHCHS} (28F016XS-15, SFI Configuration = 3, Consecutive *Alternating-A_1* Accesses)

4.4 Interfacing Examples

The following waveform examples illustrate the *Alternating-A_1* and *Same-A_1* access rules.

Consecutive *Alternating-A_1* Accesses at 33 MHz

Figure 13 illustrates consecutive *Alternating-A_1* accesses at 33 MHz, interfacing to a burst bus. The system logic controls ADV#, CE_X# and OE# to initiate read accesses to the 28F016XS. t_{CHCHA} and t_{CHCHS} (delay between *Alternating-A_1* and *Same-A_1* accesses) define the rate at which the system logic can initiate read accesses to the 28F016XS.

t_{CHCHA} calculation:

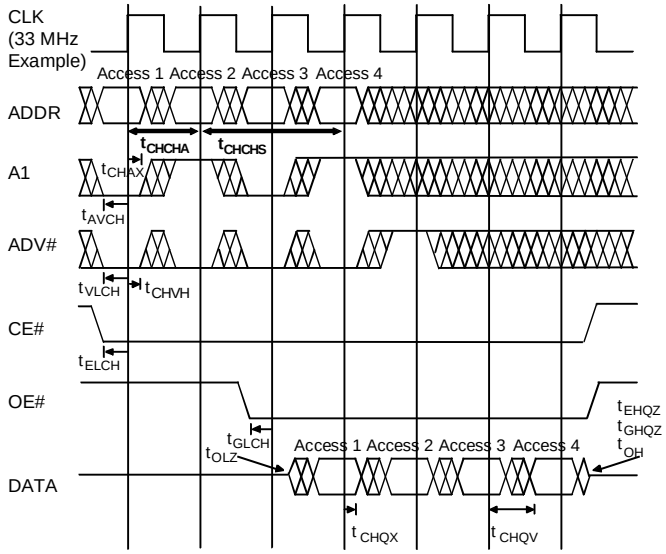
	t_{CHQV}	= 20 ns
	System Propagation Delay	= 0 ns
	CPU Set-up Requirement	= 4 ns
	Summation	= 24 ns
∴	t_{CHCHA}	= 30 ns

Access 2 can begin on the next rising CLK edge after access 1 because t_{CHCHA} equals one CLK period.

t_{CHCHS} calculation:

	SFI Configuration (2)	= 60 ns
	or	
	2 × t_{CHCHA}	= 60 ns
∴	t_{CHCHS}	= 60 ns

t_{CHCHS} is equal to two CLK periods. A second *Same-A_1* access can occur two clocks after the initial access. This burst bus interface delivers 1-0-0-0 wait-state read performance at 33 MHz, excluding system-level overhead.



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 13. 33 MHz Read Timing Waveform Example
(28F016XS-15, SFI Configuration = 2, Consecutive *Alternating-A_1* Accesses)

ADVANCE INFORMATION

Consecutive *Same-A_1* Accesses at 33 MHz

Figure 14 illustrates consecutive *Same-A_1* accesses at 33 MHz. This situation could occur when interfacing to a pipelined bus. Pipelined buses do not guarantee consecutive *Alternating-A_1* accesses.

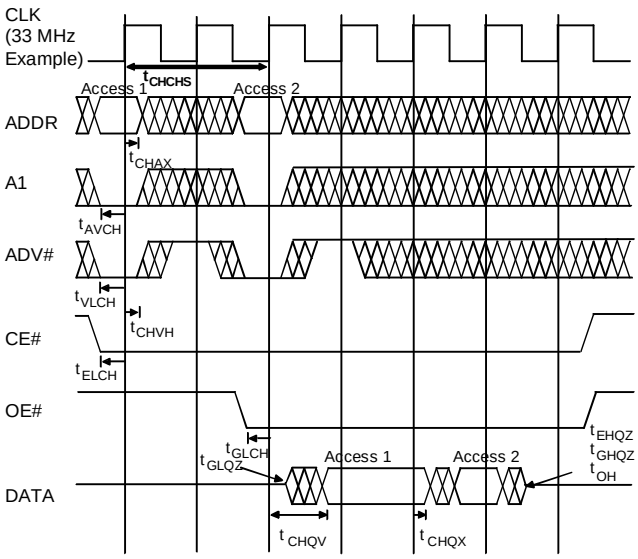
Consecutive *Same-A_1* accesses, as shown in this example, retain the same address A_1 value between accesses. Therefore, the t_{CHCHA} calculation is not required because no *Alternating-A_1* accesses occur.

t_{CHCHS} calculation:

	SFI Configuration (2)	= 60 ns
∴	t_{CHCHS}	= 60 ns

t_{CHCHS} equals two CLK periods. Hence, access 2 can begin two CLK periods after access 1.

This example read wait-state performance is 1-1 at 33 MHz, excluding system-level overhead.



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 14. 33 MHz Read Timing Waveform Example (28F016XS-15, SFI Configuration = 2, Consecutive *Same-A_1* Access)

ADVANCE INFORMATION

Consecutive *Alternating-A_1* Accesses at 40 MHz, Example 1

Figure 15 illustrates consecutive *Alternating-A_1* accesses at 40 MHz, interfacing to a burst bus.

t_{CHCHA} calculation:

t_{CHQV}	= 20 ns
System Propagation Delay	= 0 ns
CPU Set-up Requirement	= 4 ns
Summation	= 24 ns
∴ t_{CHCHA}	= 25 ns

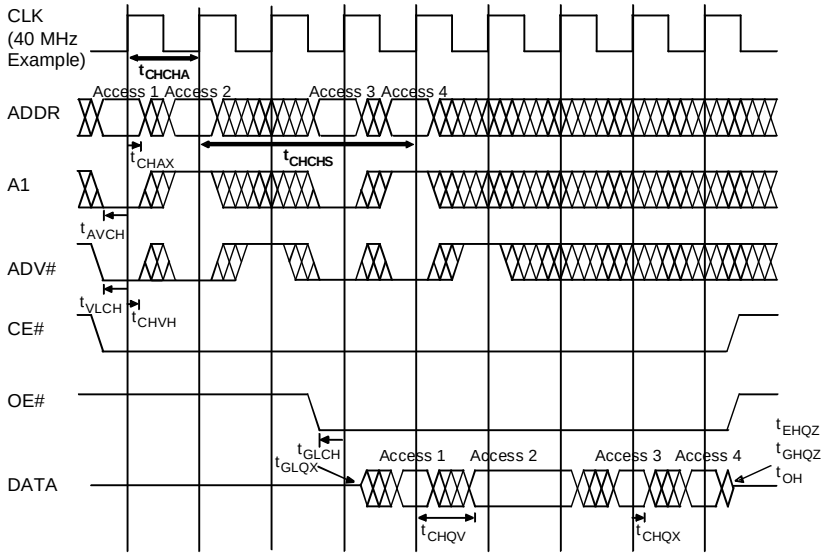
t_{CHCHA} equals one CLK period (25 ns). Therefore, an *Alternating-A_1* can begin on the next rising CLK edge.

t_{CHCHS} calculation:

SFI Configuration (3)	= 75 ns
or	
2 × t_{CHCHA}	= 50 ns
∴ t_{CHCHS}	= 75 ns

t_{CHCHS} is equal to three CLK periods. A second *Same-A_1* access, therefore, can start three clocks after the initial access.

This interface delivers 2-0-1-0 wait-state read performance at 40 MHz, excluding system-level overhead.



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 15. 40 MHz Read Timing Waveform Example 1 (28F016XS-15, SFI Configuration = 3, Consecutive *Alternating-A_1* Accesses)

ADVANCE INFORMATION

Consecutive *Alternating-A$_1$* Accesses at 40 MHz, Example 2

Figure 16 also illustrates consecutive *Alternating-A$_1$* accesses at 40 MHz, interfacing to a burst bus. In this example, a transceiver between the 28F016XS's outputs and the CPU inputs add a system propagation delay to the t_{CHCHA} calculation.

t_{CHCHA} calculation:

	t_{CHQV}	= 20 ns
	System Propagation Delay	= 5 ns
	CPU Set-up Requirement	= 4 ns
	Summation	= 29 ns
∴	t_{CHCHA}	= 50 ns

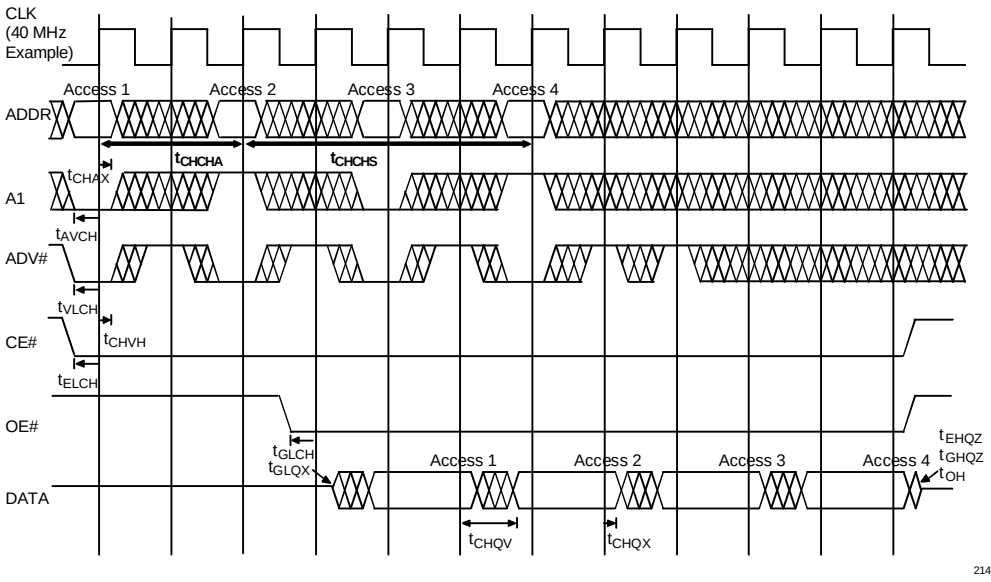
t_{CHCHA} equals two CLK periods (25 ns). Therefore, access stretching is required between *Alternating-A1* accesses.

t_{CHCHS} calculation:

	SFI Configuration (3)	= 75 ns
	or	
	2 × t_{CHCHA}	= 100 ns
∴	t_{CHCHS}	= 100 ns

t_{CHCHS} is equal to four CLK periods. A second *Same-A$_1$* access can begin four clocks after the initial access.

This burst bus interface to the 28F016XS gives 2-1-1-1 wait-state read performance at 40 MHz, excluding system-level overhead.



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 16. 40 MHz Read Timing Waveform Example 2 (28F016XS-15, SFI Configuration = 3, Consecutive *Alternating-A$_1$* Accesses, Access Stretching)

ADVANCE INFORMATION

Consecutive *Alternating-A_1* Accesses at 66 MHz

Figure 17 illustrates consecutive *Alternating-A_1* accesses at 66 MHz, interfacing to a burst bus.

t_{CHCHA} calculation:

t_{CHQV}	= 20 ns
System Propagation Delay	= 0 ns
CPU Set-up Requirement	= 4 ns
Summation	= 24 ns
$\therefore$ t_{CHCHA}	= 30 ns

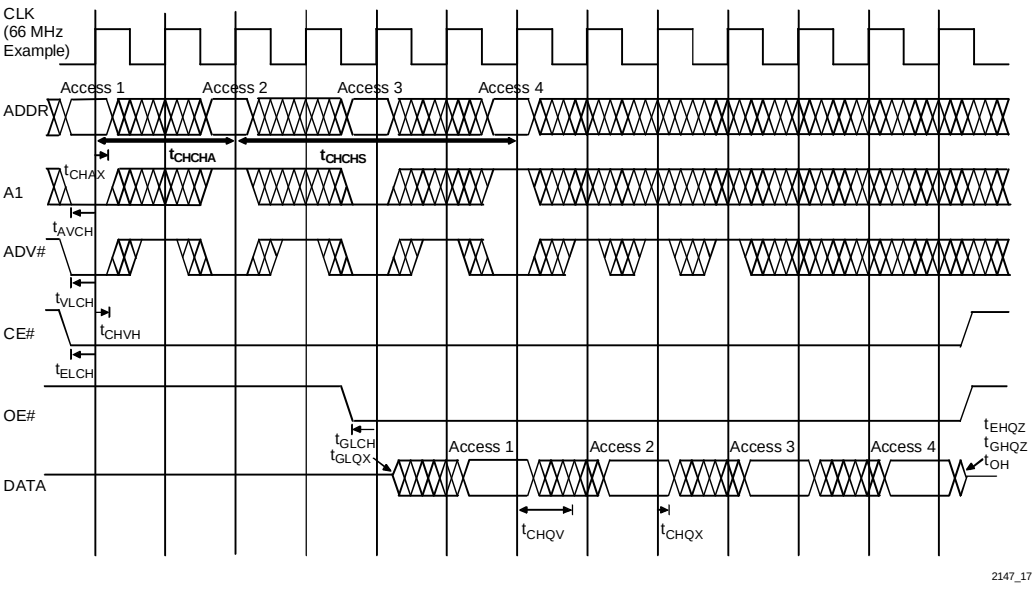
t_{CHCHA} equals two CLK periods or 30 ns. Therefore, access stretching is required between *Alternating-A_1* accesses.

t_{CHCHS} calculation:

SFI Configuration (4)	= 60 ns
or	
2 × t_{CHCHA}	= 60 ns
$\therefore$ t_{CHCHS}	= 60 ns

t_{CHCHS} is equal to four CLK periods. Hence, a second *Same-A_1* access can begin four clocks after the initial access.

This burst bus interface to the 28F016XS delivers 3-1-1-1 wait-state read performance at 66 MHz, excluding system-level overhead.



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 17. 66 MHz Read Timing Waveform Example (28F016XS-15, SFI Configuration = 3, Consecutive *Alternating-A1* Accesses, Access Stretching)

ADVANCE INFORMATION

4.5 Optimizing Read Performance in x8 Mode

When using the 28F016XS as a x8 device, the system interface to flash memory should be slightly modified to ensure highest read performance. The interface shown in Figure 18 ensures that, for a series of sequential accesses, address A_1 will alternate between "0" and "1." This configuration takes advantage of consecutive *Alternating-A_1* accesses and enables highest the 28F016XS read performance.

The examples below show the A_0/A_1 address sequence that will be presented to the 28F016XS when either the conventional or optimized A_0 and A_1 interface configurations are used.

Conventional A_0 and A_1 Configuration

A_0	A_1	Access
0	0	Initial Access
0	1	Subsequent *Same-A_1* Access
1	0	Subsequent *Alternating-A_1* Access
1	1	Subsequent *Same-A_1* Access

Optimized A_0 and A_1 Configuration

A_0	A_1	Access
0	0	Initial Access
1	0	Subsequent *Alternating-A_1* Access
0	1	Subsequent *Alternating-A_1* Access
1	1	Subsequent *Alternating-A_1* Access

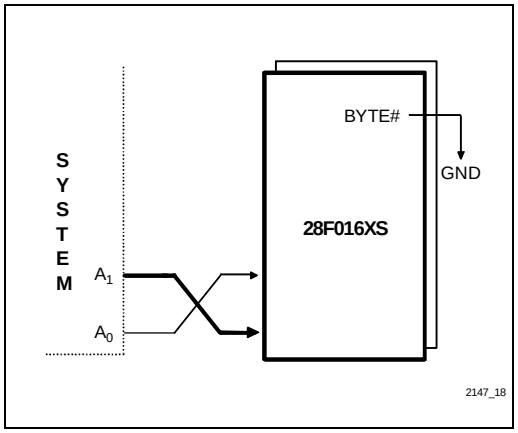


Figure 18. A_0/A_1 Optimized Interface for x8 Mode Operation

Implementing this hardware solution requires that code and data be stored to the 28F016XS in a modified sequence when using a PROM programmer. This allows expected data to be read in-system with addresses A_0 and A_1 swapped. On-board or in-system data write and erase "proceed as normal;" the address translation is handled automatically by hardware.

Figure 19 shows the code/data pattern that should be used when storing information to the 28F016XS using a PROM programmer. Figure 20 shows how that same data pattern appears to the system bus, after the A_0/A_1 address swap.

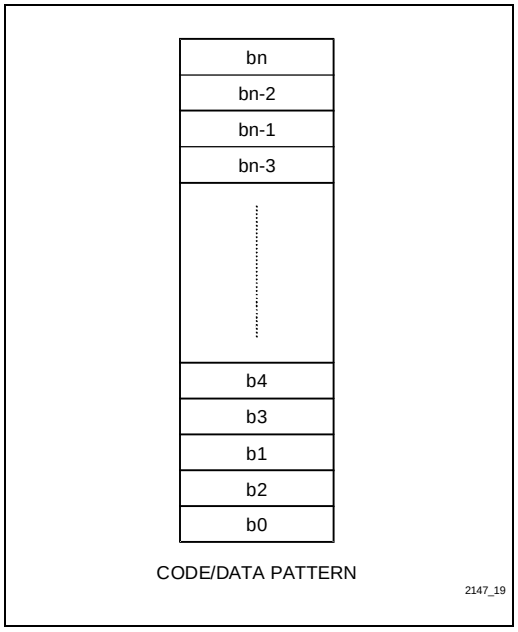


Figure 19. Code/Data Structure When Performing Off-Board Programming

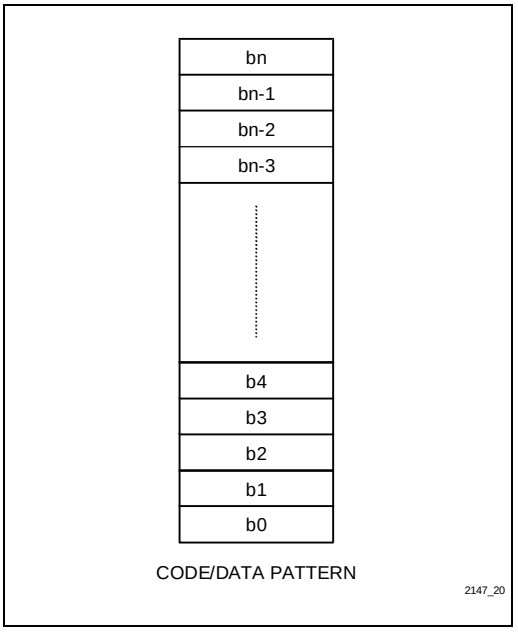


Figure 20. Code/Date Structure Seen by the Processor

4.6 Consecutive Accesses across Memory Bank Boundaries

Figure 21 illustrates a multi-bank 28F016XS configuration. CE_0# and CE_1# make the particular memory bank selection.

In this configuration, a sequence of possible accesses can potentially begin and end in different memory banks, which could potentially cause bus contention if not properly handled. All pipelined accesses to the initial bank must complete before activating the output buffer to the next memory bank. Both burst and pipelined buses are susceptible to this occurrence because of their sequential accessing nature.

ADVANCE INFORMATION

Burst Bus

A burst bus will only cross memory bank boundaries when the length of the burst transfer exceeds the size of the individual banks. Refer to Tables 2 and 3 for the Intel and linear burst orders.

Pipelined Bus

Pipelined buses do not guarantee linear sequential accesses. Therefore, consecutive accesses can potentially bounce back and forth between different memory banks, crossing memory bank boundaries.

Handling Consecutive Accesses That Cross Memory Bank Boundaries

When consecutive accesses cross memory bank boundaries, it is important to guard against enabling the output buffers for both banks. All pipeline accesses to the initial bank must complete before activating the OE# to the next bank. Enabling the buffers for both banks at the same time will cause bus contention.

System logic can handle consecutive accesses that cross bank boundaries in one of two ways:

1. Complete all accesses initiated in the first memory bank before addressing a new bank (Figure 22). This configuration requires only one OE# (Figure 21).
2. Before finishing all accesses to the initial memory bank, read cycles targeting a new bank can start as long as its OE# is deactivated. All accesses to the initial bank must finish before activating the OE# to the next bank. The OE# for the next bank is activated t_{GHQZ} after the OE# to the initial bank has been deactivated (Figure 23). This configuration requires an OE# for each bank.

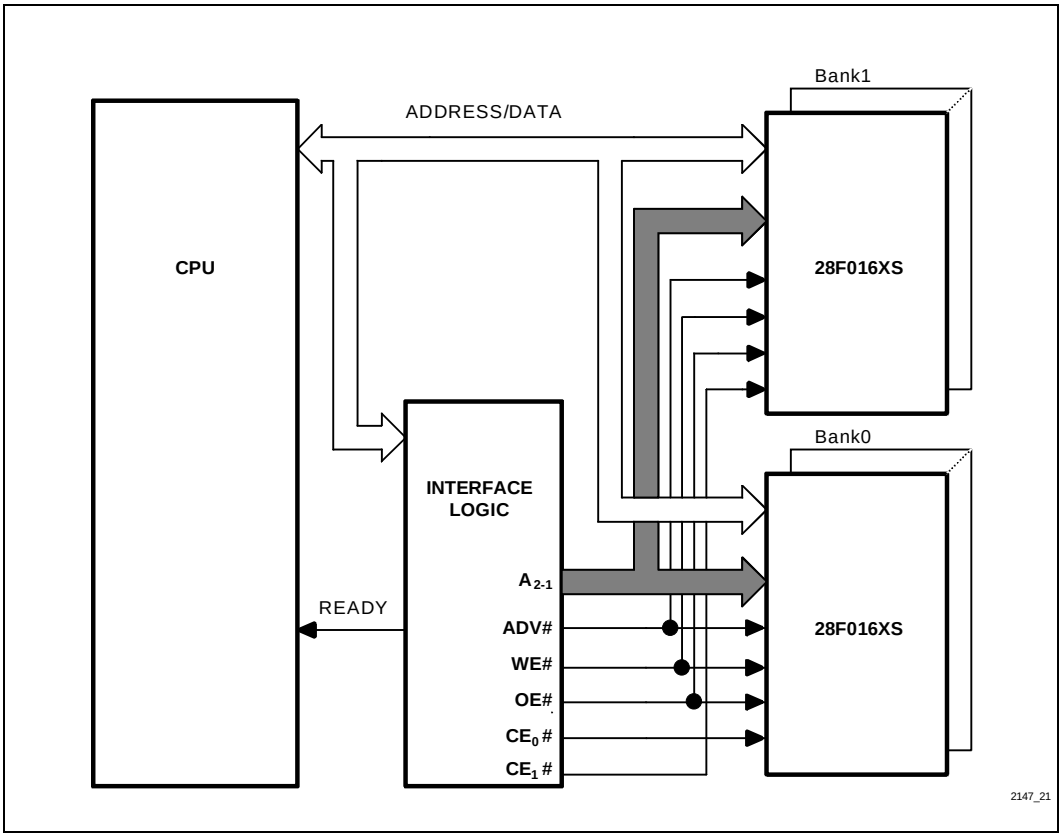
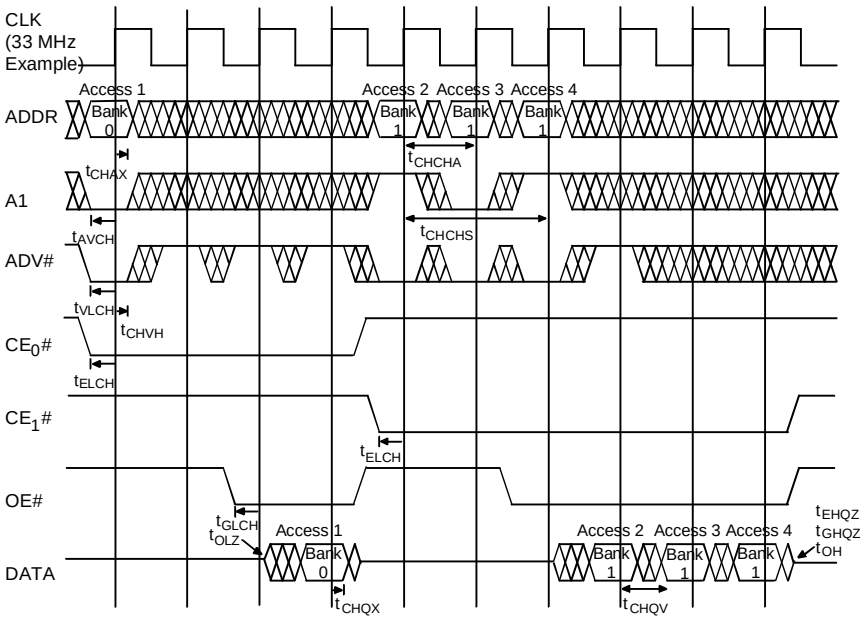


Figure 21. Two-Bank 28F016XS Combination, Burst Bus Interface, Single OE#

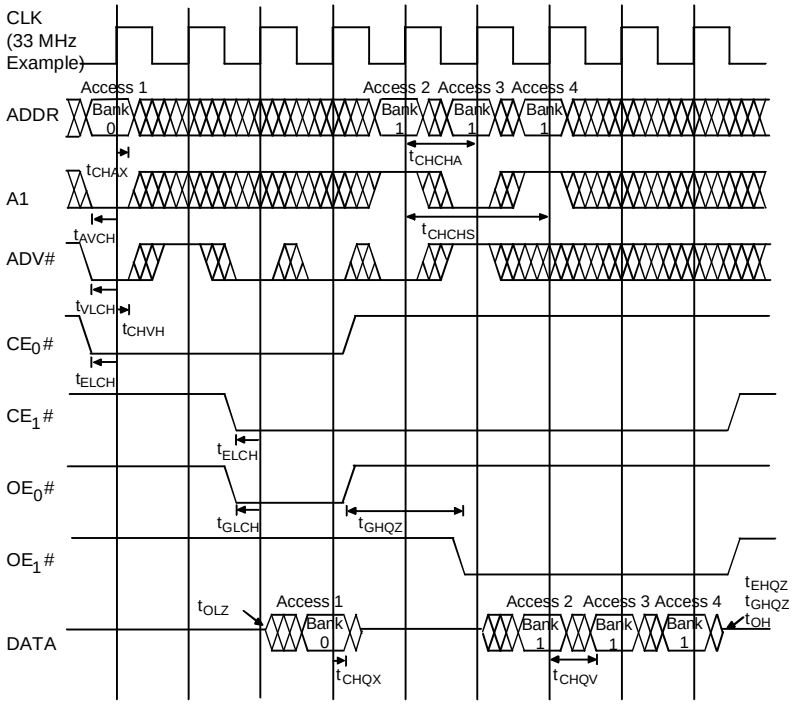
ADVANCE INFORMATION



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 22. Consecutive *Alternating-A$_1$* Accesses Crossing Bank Boundaries, One OE# (28F016XS-15, SFI Configuration = 2)

ADVANCE INFORMATION



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 23. Consecutive *Alternating-A$_1$* Accesses Crossing Bank Boundaries, One OE# per Bank (28F016XS-15, SFI Configuration = 2)

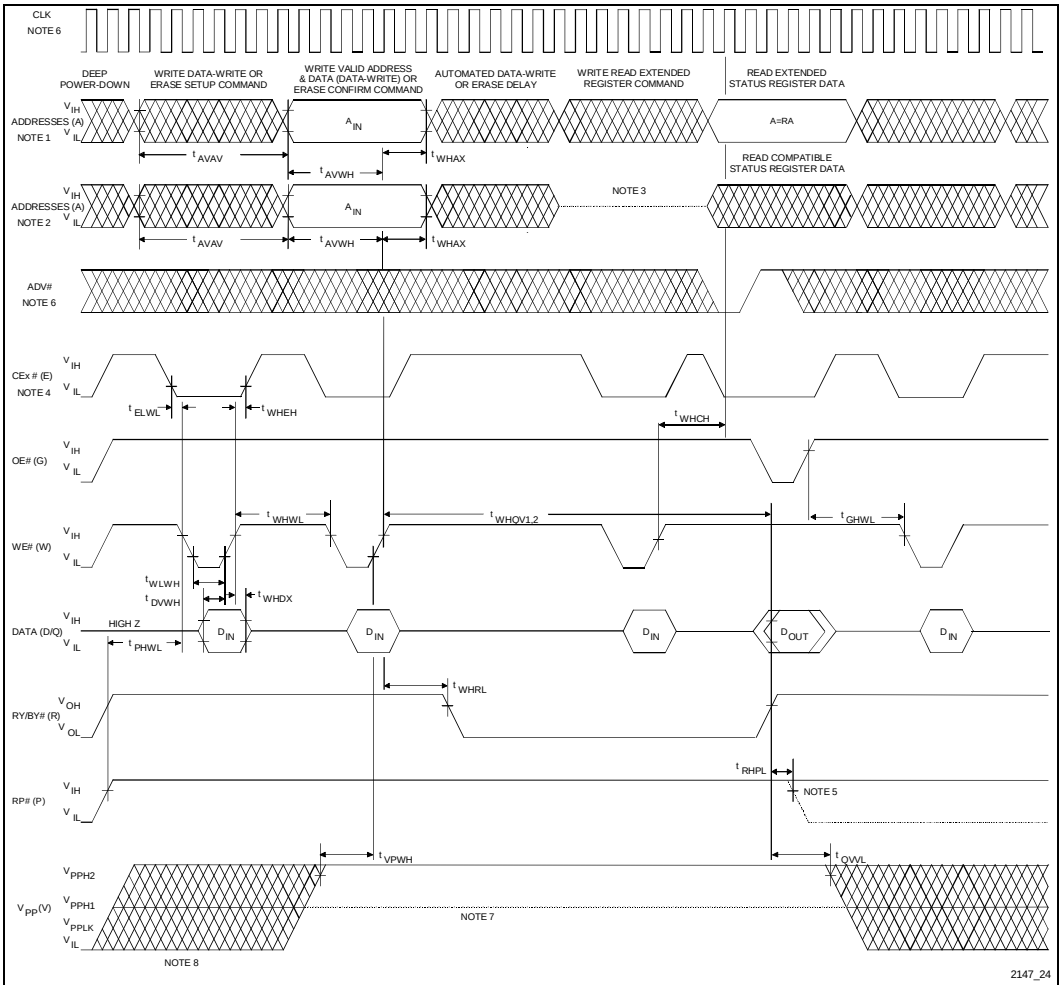
4.7 Handling Asynchronous Writes

The 28F016XS write interface is asynchronous, similar to other Intel flash memories. The 28F016XS's write interface is not pipelined, therefore a write cycle must complete before another begins.

When the interfacing CPU can execute a burst write cycle, the system logic needs to manage the write cycle, allowing only one write cycle to the 28F016XS at a time. Write-back caches, for example, support burst write cycles.

When the interfacing CPU does not support burst writes, the 28F016XS's asynchronous write interface is not an issue. The processor will only execute one write cycle at a time.

When executing a write cycle, the interfacing processor or bus will drive a write signal informing the system of the desired operation. Monitoring this signal, the interfacing logic can correctly transition into an appropriate state machine sequence. In this situation, the interfacing logic directly controls the 28F016XS's write control signals (Figure 24), just as with asynchronous Intel flash memories.



NOTE:
Refer to the 28F016XS datasheet for timing specifications.

Figure 24. Write Timing Waveform with V_{PP} at 5V or 12V

4.8 System Boot from 28F016XS

When booting from the 28F016XS, the interfacing logic must first support the default SFI Configuration value, 4. Depending on the actual CLK frequency however, the SFI Configuration (Section 2.3) may require adjustment to achieve maximum read performance. To achieve this flexibility, the interfacing logic must:

- Support the default SFI Configuration
- Be aware of changes to the SFI Configuration
- Alter the interface accordingly to handle optimized configuration

If a change is made to the SFI Configuration, the interface must be informed so that it can adjust accordingly. If the interface does not adjust, it will

continue performing read cycles assuming a default SFI Configuration.

Figure 25 illustrates a possible way of informing the interface of a change to the SFI Configuration, using a general purpose input/output (GPIO). On system power up or reset, the GPIO transitions to a specific logic level (V_{IL} or V_{IH}), informing the interface of a default SFI Configuration. After altering the SFI Configuration, software changes the GPIO value. This input informs the interface of changes to the SFI Configuration.

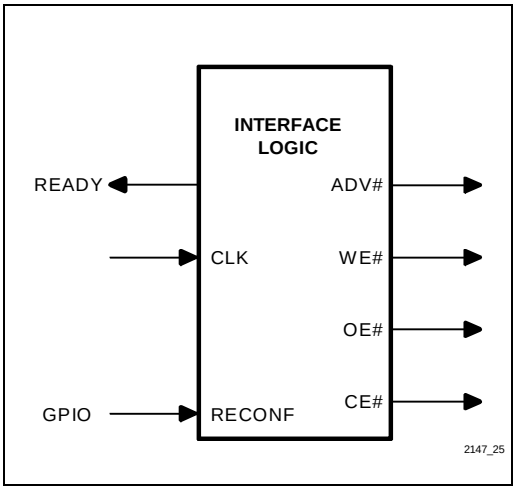


Figure 25. Informing Interface of a SFI Configuration Change

Figure 26 illustrates the internal state machine. Depending upon the value of the GPIO (RECONFIG), the state machine will take one of two paths. One path is capable of handling the default SFI Configuration, while the second path supports the optimized SFI Configuration. Supporting the default configuration, the interface logic initiates accesses to the 28F016XS corresponding to a SFI Configuration of 4. In the optimized SFI Configuration, the interface can initiate read accesses to the 28F016XS at a faster rate if the SFI Configuration is less than 4. The rate at which the system logic can start read cycles is related to the SFI Configuration value. Refer to Section 4.2 and 4.3 for the *Alternating-A*$_1$ and *Same-A*$_1$ access rules.

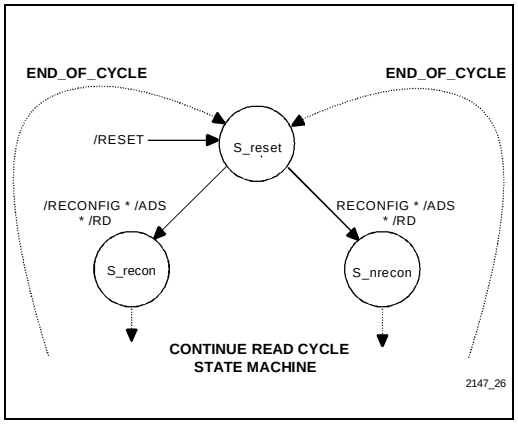


Figure 26. State Machine Handling of Boot-Up and Optimized SFI Configuration

5.0 DESIGNING A FLEXIBLE INTERFACE FOR THE 28F016XS AND 28F016SA/SV

Designers using the 28F016SA/SV now who are planning a future upgrade to the 28F016XS can design system logic capable of interfacing to both components.

Pinout similarities between the 28F016XS and 28F016SA/SV support a single footprint suitable for both components (see Section 2.1).

Supporting both components, the interface incorporates a state machine designed to control the 28F016SA and a second one to control the 28F016XS enhanced read interface.

The device identifiers can be used to communicate whether the 28F016SA/SV or enhanced 28F016XS device is in the system. The 28F016SA/SV and 28F016XS have distinct device IDs. A jumper configuration can also be used to communicate 28F016SA/SV or 28F016XS presence. Figure 27 illustrates the jumper identification method. The status of the jumper controls state machines within interface logic and/or can be read by system software to set wait-state registers within the CPU.

Device ID or jumper identification can also be used to enable system software usage of 28F016XS's Status Register enhancements and the Device Configuration Code.

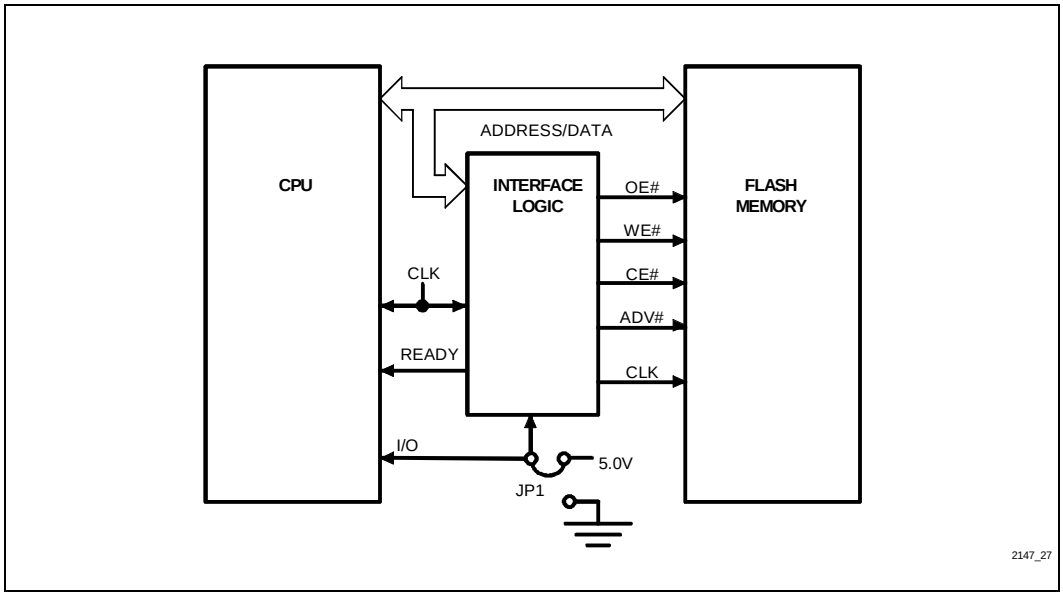


Figure 27. Jumper Identification of 28F016SA/SV or 28F016XS Presence for Wait-State Control

ADVANCE INFORMATION

6.0 CONCLUSION

The read performance of the 28F016XS far exceeds that of traditional flash memories. Up to three accesses at a time can execute in parallel. The 28F016XS's synchronous interface makes it highly compatible with CPU and bus architectures that implement an Intel or linear burst or pipelined bus. This application note has provided the fundamental knowledge that will enable designers to easily interface to the 28F016XS. For further information about the 28F016XS, consult reference documentation for a more comprehensive understanding of device capabilities and design techniques.

ADDITIONAL INFORMATION

Order Number	Document/Tool
290532	28F016XS Datasheet
297500	"Interfacing the 28F016XS to the i960 ® Microprocessor Family"
297504	"Interfacing the 28F016XS to the Intel486™ Microprocessor Family"
292146	AP-600, "Performance Benefits and Power/Energy Savings of 28F016XS Based System Designs"
292126	AP-377, "16-Mbit Flash Product Family Software Drivers, 28F016SA/28F016SV/28F016XS/28F016XD"
297372	16-Mbit Flash Product Family User's Manual
287508	FLASHBuilder Utility
Contact Intel/Distribution Sales Office	28F016XS Benchmark Utility
Contact Intel/Distribution Sales Office	28F016XS iBIS Models
Contact Intel/Distribution Sales Office	28F016XS VHDL/Verilog Models
Contact Intel/Distribution Sales Office	28F016XS Timing Designer Library Files
Contact Intel/Distribution Sales Office	28F016XS Orcad and ViewLogic Schematic Symbols

REVISION HISTORY

Number	Description
001	Original Version
002	Revised Section 4.7, "Handling Asynchronous Writes," removing Page Buffer reference

APPENDIX A
PLD EQUATIONS FOR CLOCK GENERATION AND SYNCHRONIZATION

Clock Generation and Synchronization Using a 2x Clock Input

```
Title                1x Clock Generations & Synchronization
Pattern              PDS
Revision             1
Author               Example
Company Name         Intel
Date                 5/26/94

CHIP    Clock_Generation_Circuit    85C220

; Input assignments
PIN          CLK       ; 2x CLK input frequency
PIN          RESET     ; System RESET
PIN          SYSCLK    ; Synchronized 1x CLK output

EQUATIONS
SYSCLK := /SYSCLK * /RESET
SYSCLK.CLKF = CLK
```

Clock Generation and Synchronization Using a 4x Clock Input

```
Title                1x Clock Generation and Synchronization
Pattern              PDS
Revision             1
Author               Example
Company Name         Intel
Date                 5/26/94

CHIP    Clock_Generation_Circuit    85C220

; Input assignments
PIN          CLK       ; 4x CLK input frequency
PIN          RESET     ; System RESET
PIN          CLK2      ; 2x CLK output
PIN          SYSCLK    ; Synchronized 1x CLK output

EQUATIONS
CLK2 := /CLK2
CLK2.CLKF = CLK
SYSCLK :=  CLK2 *  SYSCLK * /RESET
      + /CLK2 * /SYSCLK * /RESET
SYSCLK.CLKF = CLK
```