Technical Note

Comparing Z8 Encore! XP® F640x and Z8 Encore! XP® F64xx Flash Loaders

TN003003-0308

Introduction

This Technical Note describes the differences in procedure when loading Flash memory on Zilog's Z8F640x and Z8F64xx microcontroller units (MCU). The Z8F64xx MCUs, which evolved from the Z8F640x MCUs, offer better Flash protection and flexibility features by comparison.

General Overview of Z8 Encore! XP® Flash Memory

The Z8F640x and Z8F64xx MCU families each feature 64 KB (65,536 bytes) of non-volatile Flash memory with read, write, and erase capability. Flash memory can be programmed and erased in-circuit by either user code or through the On-Chip Debugger. The Flash memory array is arranged as 512 bytes per page. A 512 bytes page is the minimum Flash block size that can be erased.

Flash Information Area

The Z8F64xx MCUs feature a 512 bytes Information Area, which is accessed by setting bit 7 of the Flash Page Select Register to 1. When access is enabled, the Information Area is mapped into Program Memory and overlays the 512 bytes in the address range FE00h–FFFFh. When access to the Information Area is enabled, a code instruction returns data from the Information Area. CPU instructions are always fetched from Program Memory regardless of the setting of the Information Area access bit. Access to the Information Area is Read-Only.

Table 1 describes the functions of each range of Program Memory addresses in the Z8F64xx MCU's Information Area.

Table 1. Z8F64xx Information Area Map

Program Memory Address (Hex)	Function
FE00h–FE3Fh	Reserved.
FE40h–FE53h	Part Number—20 character ASCII alphanumeric code; left-justified and filled with zeroes.
FE54h–FFFFh	Reserved.

Note: *The Z8F640x MCUs do not feature an Information Area.*

Table 2 compares the Flash memory spaces of the Z8F640x MCU and the Z8F64xx MCU.

Table 2. Flash Memory Configurations for Z8F640x and Z8F64xx MCUs

Z8 Encore! XP MCU	Total Size of Flash Memory	Number of Pages	Program Memory Addresses	Number of Sectors	Pages per Sector	Size of Each Sector
Z8F640x	64 KB (65,536)	128	0000h–FFFFh	1	16	8 KB (8192)
Z8F64xx	64 KB (65,536)	128	0000h–FFFFh	8	16	8 KB (8192)

Table 3 compares the Flash address ranges of the Z8F640x MCU and the Z8F64xx MCU by sector.

Table 3. Flash Sector Address Ranges

Sector Number	Z8F640x MCUs	Z8F64xx MCUs
0	NA	0000h–1FFFh
1	NA	2000h–3FFFh
2	NA	4000h–5FFFh
3	NA	6000h–7FFFh
4	NA	8000h–9FFFh
5	NA	A000h–BFFFh
6	NA	C000h–DFFFh
7	E000h–FFFFh (High Sector)	E000h–FFFFh

Z8 Encore! XP® Flash Control Register Descriptions

There are a number of registers available in the Z8 Encore! XP MCU that provide access to Flash memory. They are briefly described in this section.

Flash Control Register

The Flash Control Register (FCTL) unlocks the Flash Controller for programming and erase operations. It also selects the Flash Sector Protect register on the Z8F64xx MCU. The Write-Only Flash Control Register shares its register file address space with the Read-Only Flash Status Register.

Flash Controller Behavior in DEBUG Mode

The following changes in the functionality of the Z8F64xx MCU Flash Controller occur when the Flash Controller is accessed using the On-Chip Debugger:

- The Flash Write Protect option bit is ignored.
- The Flash Sector Protect Register is ignored for programming and erase operations.
- Programming operations are not limited to the page selected in the Flash Page Select Register.
- Bits in the Flash Sector Protect Register can be written to with values one or zero.
- The second Write of the Flash Page Select Register to unlock the Flash Controller is not necessary.
- The Flash Page Select Register can be written when the Flash Controller is unlocked.
- The Mass Erase command is enabled.

At Reset, the status is as listed below:

- The Flash Controller is locked on both the Z8F640x MCU and the Z8F64xx MCU.
- On the Z8F64xx MCU, none of the pages, including the Information Area, are selected and all sectors are rendered unprotected.

Flash Controller Bypass

The Flash Controller is bypassed and the control signals for Flash memory brought out to the GPIO pins in both the Z8F64xx and Z8F640x MCUs. Bypassing the Flash Controller allows faster row programming algorithms by controlling the Flash programming signals directly. Row programming is recommended for gang programming applications and large volume customers who do not require in-circuit initial programming of Flash memory. Mass Erase and Page Erase operations are also supported when the Flash Controller is bypassed.

Flash Controller Bypass mode is enabled by writing three bytes to the On-Chip Debugger via the DBG interface:

0x80—Initiates an autobaud calculation of the DGB interface data and clock rate.

0xF0—An OCD Write Test Mode Register command.

0x04—Data to be written to the Test Mode Register.

For details about the On-Chip Debugger, refer to the appropriate Z8 Encore! XP product specification for your device.

Flash Status Register

Values in the Flash Status Register (FSTAT) indicate the current state of the Flash Controller. This register can be read at any time. The Read-Only Flash Status Register shares its register file address space with the Write-Only Flash Control Register. Table 4 on page 4 compares the values associated with the FSTAT register for both the Z8F640x and Z8F64xx MCUs.

Table 4. FSTAT Register Values for Z8F640x and Z8F64xx MCUs

Operations	Z8F640x MCU FSTAT Register Values	Z8F64xx MCU FSTAT Register Values
Flash Controller Locked	000000	00_0000
First unlock command received	000001	00_0001
Second unlock command received	NA	00_0010
Flash Controller unlocked	000010	00_0011
Flash Sector Protect register selected	NA	00_0100
Program operation in progress	001xxx	00_1xxx
Page erase operation in progress	010xxx	01_0xxx
Mass erase operation in progress	100xxx	10_0xxx

Note: *Register values for FSTAT[7:6] are reserved on both Z8F640x and Z8F64xx MCUs, and must be 0.*

Flash Page Select Register

The Flash Page Select Register (FPS) is used to select one of the 128 available Flash memory pages to be erased in a Page Erase operation. On a Z8F64xx MCU, the FPS register is also used to select the page to be programmed. During a Page Erase operation, all Flash memory locations containing addresses with the most significant 7 bits provided by FPS[6:0] are erased. An erased Flash byte contains all ones (FFh). Other pages are not affected by this operation.

Note: *On a Z8F64xx MCU, the Flash Page Select register shares its register file address space with the Flash Sector Protect Register. The Flash Page Select register cannot be accessed when the Flash Sector Protect Register is enabled.*

Flash Frequency Byte Registers

The Flash Frequency High (FFREQH) and Low Byte (FFREQL) registers combine to form a 16 bit value, FFREQ, to control timing for Flash program and erase operations. The 16 bit Flash Frequency registers should be written with the system clock frequency in kHz for Program and Erase operations.

Flash Sector Protect Register

Z8F64xx MCUs feature an additional Flash Control Register—the Flash Sector Protect Register (FPROT). This register protects sectors of Flash memory from being programmed or erased by user code. The Flash Sector Protect Register shares its Register File address space with the Flash Page Select register. The Flash Sector Protect Register is accessed only after writing 5Eh to the Flash Control Register.

Reset values and descriptions for each control register are indicated in Table 5 on page 5.

Table 5. Control Register Values

Control Register	Value at Reset	Description
FCTL	00h	Flash Controller locked.
FSTAT	00h	Flash Controller locked.
FPS	00h	None of the pages are selected.
FFREQH	00h	—
FFREQL	00h	—
FPROT	00h	All sectors become unprotected.

Note: *The registers in Table 5 are defined in the header file* `<ez8.h>`. *To use these registers, it is necessary to include this header file in the C program.*

Operational Comparisons

The following sections explain how to use and set up different registers for programming Flash memory on the Z8F640x and Z8F64xx MCUs.

Setting the Timing of Flash Operation

Before performing either a program or an erase operation in Flash memory, you must first configure the Flash Frequency High Byte and Low Byte registers. These Flash Frequency registers allow programming and erasure of Flash memory with system clock frequencies ranging from 32 kHz (32768 Hz) to 20 MHz in Z8F640x MCUs and with system clock frequencies ranging from 20 kHz to 20 MHz in Z8F64xx MCUs (the valid range is limited to each device's operating frequencies).

The 16 bit register FFREQ must contain the system clock frequency in kHz. This value is calculated using the following equation:

$$\text{FFREQ[15:0]} = \frac{\text{System Clock Frequency (Hz)}}{1000}$$

For example, the system clock frequency for both Z8F640x MCUs and Z8F64xx MCUs is 18432000 Hz. Using this value in the equation, the values for the Flash Frequency High and Low Byte registers are calculated as follows:

FFREQ[15:0] = 18432000 ÷ 1000

FFREQ[15:0] = 18432d

FFREQ[15:0] = 4800h

Therefore, the Flash Frequency High Byte register, FFREQH, yields a value of `48h` and the Flash Frequency Low Byte register, FFREQL, yields a value of `00h`.

Read Protection of Flash User Code

The user code contained within Flash memory is protected from external read access. Programming the Flash Read Protect Option Bit (RP) prevents the reading of user code by the On-Chip Debugger in both the Z8F640x and the Z8F64xx MCUs. Additionally, in Z8F64xx devices, Flash Controller Bypass mode is used to prevent the reading of Flash memory. In the case of a Read request on a Z8F64xx MCU, a default value of FFh is returned.

Write/Erase Protection of Flash User Code

Z8F640x MCUs provide several levels of protection against accidental program and erasure of the contents of Flash memory. This protection is provided by a combination of the option bits—Flash Write Protect (FWP) and Flash High Sector Write Protect (FHWP)—and the locking mechanism of the Flash Controller.

On Z8F64xx MCUs, this protection is provided by the Flash Controller unlock mechanism, the Flash Sector Protect register (FPROT), and the Flash Write Protect (FWP) option bit. A description of the unlock mechanism for each of the two types of MCUs follows.

Flash Controller Unlock Mechanism in Z8F640x Devices

At Reset, the Flash Controller locks to prevent the accidental programming or erasing of Flash memory. To program or erase Flash memory, the Flash Controller must be unlocked. To unlock the Flash Controller, a particular sequence of instructions is provided to it. Any value written by the user code out of sequence to the Flash Control Register (FCTL) on a Z8F640x MCU locks the Flash Controller. However, it can be unlocked by following the instructions below:

1. Write the first unlock command, 73h, to the Flash Control Register.
2. Write the second unlock command, 8Ch, to the Flash Control Register.

After unlocking the Flash Controller, Flash memory can be programmed or erased. When the Flash Controller is unlocked, any value written to the Flash Control Register locks the Flash Controller. Writing the Mass Erase or Page Erase commands executes either function before locking the Flash Controller. The following sample code can be used to unlock the Flash Controller on a Z8F640x MCU.

```
unsigned int unlock_flash(void)       // for Z8F640x MCUs
{
     if((FSTAT & 0x3F) == 0x00)       // Flash Controller Locked state
   {
     FCTL = 0x73;                     // First unlock command
     if ((FSTAT & 0x3F) == 0x01)      // First unlock Command received
     {
        FCTL = 0x8C;                  // Second unlock command
        if ((FSTAT & 0x3F) == 0x02)   // Unlocked programming state
         return 0x00;
        else
         return 0xFF;                 // Return error code
        else
```

```
         return 0xFF;
   }
     else
        return 0xFF;
}
```

Note: *It is important that the instructions are written in the order presented.*

To lock the Flash Controller in a Z8F640x MCU, use the following code:

```
void lock_flash(void)                  // for Z8F640X MCUS revision
{
     FCTL=0xFF;                        // lock command
}
```

Flash Controller Unlock Mechanism in Z8F64xx Devices

Like the Z8F640x MCU, the Flash Controller locks at Reset to prevent the accidental programming or erasing of Flash memory. To program or erase Flash memory, the Flash Controller must be unlocked. To unlock the Flash Controller, a particular sequence of instructions is provided to it. Any value written by the user code out of sequence to the Flash Control Register (FCTL) and the Flash Page Select Register (FPS) on Z8F64xx MCUs locks the Flash Controller. However, it is unlocked by following the instructions below:

1. Write 00h to the Flash Control Register (FCTL) to reset the Flash Controller.
2. Write the page to be programmed or erased to the Flash Page Select Register (FPS).
3. Write the first unlock command, 73h, to the Flash Control Register (FCTL).
4. Write the second unlock command, 8Ch, to the Flash Control Register (FCTL).
5. Rewrite the page written in Step 2 to the Flash Page Select Register (FPS).

After unlocking the Flash Controller, Flash memory can be programmed or erased. The following sample code is used to unlock the Z8F64xx Flash Controller.

```
Unsigned int unlock_flash(unsigned int pagenum) // for Z8F642x MCUs
                                          // pagenum is the page to
                                          // be unlocked.
{
  if( (FSTAT & 0x3F) == 0x00 )        // Flash Controller Locked state
  {
     FCTL = 0x00;
     FPS = pagenum;
     FCTL = 0x73;                     // First unlock command
     if ((FSTAT & 0x3F) == 0x01)      // First unlock Command received
     {
        FCTL = 0x8C;                  // Second unlock command
```

```
        if ((FSTAT & 0x3F) == 0x02)   // Second unlock command received
        {
         FPS = pagenum;
         if((FSTAT & 0x3F) == 0x03)   // Flash Controller unlocked
          return 0x00;
         else
          return 0xFF;                // Return error code
        }
        else
         return 0xFF;
     }
     else
        return 0xFF;
  }
   else
     return 0xFF;
}
```

Note: *It is important that the instructions are written in the order presented above.*

The following code is used to lock the Flash Controller in Z8F64xx MCUs.

```
void lock_flash(void)                 // for Z8F642X MCU
{
 FCTL = 0x00;                         // Lock command
}
```

Protecting the Flash Sector User Code

The Z8F64xx MCU's Flash memory space is divided into eight sectors. It supports Flash Sector Protection. The Flash Sector Protect Register is configured to prevent sectors from being programmed or erased inadvertently. When a sector is protected, it cannot be unprotected by user code. The Flash Sector Protect Register is cleared after Reset; as a result, any previously-written protection values are lost. User code must include this register in the initialization routine to enable sector protection. The Mass Erase command overrides the settings in this register.

Follow the steps below for setting up the Flash Sector Protect Register from user code:

1. Write `00h` to the Flash Control Register to reset the Flash Controller.
2. Write `5Eh` to the Flash Control Register to select the Flash Sector Protect Register.
3. Read and/or write the Flash Sector Protect Register, which now resides at Register File address `FF9h`.
4. Write `00h` to the Flash Control Register to return the Flash Controller to its reset state.

The following code segment describes how to set up the Flash Sector Protect Register, with the assumption that Sector 5 is to be protected from either a Program or an Erase operation.

```
Unsigned int flash_sector_protect (void)
{ . . .

 FCTL = 0x00;
 FCTL = 0x5E;                          // selects FPROT register
 if( (FSTAT & 0x3F) == 0x04)           // FPROT register is selected
 {
     FPROT = 0x20;                     // protects Sector 5
     return 0x00;
  }
 else
 return 0xFF;                          // return error code
 . . .
 }
```

Note: *User code can only write a 1 to the bits in this register; bits cannot be cleared to 0 by user code.*

Byte Programming

When a Z8 Encore! XP Flash Controller is unlocked, writing to Program Memory programs a byte into Flash only if the address is located in an unlocked page. An erased Flash byte contains all ones (FFh). The programming operation is specifically used to change bits from ones to zeroes. To change a Flash bit (or multiple bits) from zero to one requires a Page Erase or Mass Erase operation.

Byte programming is accomplished using the On-Chip Debugger's Write Memory command or by executing an eZ8 CPU code instruction.

Byte Programming in Z8F640x Devices

The following code segment is used for byte programming.

```
************************************************************************
Char RamByte must be declared as a global variable. In the write_flash()
function, the parameter ‘location’ represents the absolute Flash memory
address and the parameter ‘value’ is the data to be written at that
address.
************************************************************************
Char RamByte;                         // declared as a global variable
void write_flash(unsigned int location, char value)
{
 RamByte = (char)(location >> 0x08);  // get the higher byte of
                                      // absolute Flash address
 asm("LDX R8, _RamByte");             // Load the higher byte into
                                      // R8 register
```

```
 RamByte = (char)(location & 0x00FF); // get the lower byte of
                                      // absolute Flash address
 asm("LDX R9, _RamByte");             // Load the lower byte into R9
                                      // register
 RamByte = value;                     // get the data to be written
                                      // into Flash memory
 asm("LDX R10, _RamByte");            // Load the data into R10 register
 asm("LDC @RR8, R10");                // Load Byte into Flash
                                      // location pointed by R8 & R9

 asm("INC R9");                       // Increment R9 to point to
                                      // the next location
 asm("LDC @RR8, R10");                // Write to the next Flash location
 asm("INC R9");                       // Again increment R9 to point
                                      // to the next location
 asm("LDC @RR8, R10");                // Write to the next Flash location
}
```

On Z8F640x MCUs, to exit programming mode and lock the Flash Controller, write any value other than the Mass Erase or Page Erase commands to the Flash Control Register.

Using the Write Memory Command for Byte Programming

On Z8F640x MCUs, the On-Chip Debugger's Write Memory command can also be used for byte programming. This Write Program Memory command, 0Ah, writes data to Program Memory. This command is equivalent to the LDC and LDCI instructions.

Data can be written 1 to 65536 bytes at a time (65536 bytes is written by setting the size to zero). The on-chip Flash Controller is written to and unlocked for the programming operation to occur. If the Flash Controller is not unlocked, the data is discarded. Data is also discarded if the device is not in DEBUG mode or if the Read Protect Option Bit is enabled.

```
DBG ← 0Ah //indicates data sent from host to the On_Chip Debugger
DBG ← Program Memory Address[15:8]
DBG ← Program Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG ← 1-65536 data bytes
```

DBG is the On-Chip Debugger.

Note: *The Write Memory command is not accessible.*

Byte Programming in Z8F64xx Devices

Follow the steps for programming Z8F64xx Flash memory from user code:

1. Write 00h to the Flash Control Register to reset the Flash Controller.
2. Write the page of memory to be programmed to the Flash Page Select Register.
3. Write the first unlock command, 73h, to the Flash Control Register.
4. Write the second unlock command, 8Ch, to the Flash Control Register.
5. Rewrite the page written in Step 2 to the Flash Page Select Register.
6. Write to Program Memory using the LDC or LDCI Flash programming instructions.
7. Repeat Step 6 to program additional memory locations on the same page.
8. Write 00h to the Flash Control Register to lock the Flash Controller.

The following code describes how to program a byte to Z8F64xx Flash memory.

```
************************************************************************
Char near RamByte must be declared as a global variable. In the
write_flash() function, the parameter ‘location’ represents the absolute
Flash memory address, the parameter ‘pagenum’ is the page to be programmed,
and the parameter ‘value’ is the data to be written at that address.
************************************************************************
Char near RamByte;                    // Declared as a global variable.
void write_flash(unsigned int location,unsigned int pagenum,char value)
{
  FCTL = 0x00;
  FPS = pagenum;                      // Page to be unlocked and written
  FCTL = 0x73;                        // First unlock command
  FCTL = 0x8C;                        // Second unlock command
  FPS = pagenum;                      // Page to be unlocked and written

  RamByte = (char)(location >> 0x08); // Get the higher byte of the
                                      // absolute Flash address
  asm("LD R8, _RamByte");             // Load the higher byte into R8
                                      // register
  RamByte = (char)(location & 0x00FF);// Get the lower byte of the
                                      // absolute Flash address
  asm("LD R9, _RamByte");             // Load the lower byte into R9
                                      // register
  RamByte = value;                    // Get the data to be written
                                      // into Flash memory
  asm("LD R10, _RamByte");            // Load the data into R10 register
  asm("LDC @RR8,R10");                // Load Byte into Flash location
                                      // pointed by R8 & R9
  asm("INC R9");                      // Increment R9 to point to the
                                      // next location
  asm("LDC @RR8,R10");                // Write to the next Flash
```

```
                                        // location
  asm("INC R9");                        // Again increment R9 to point to
                                        // the next location
  asm("LDC @RR8,R10");                  // Write to the next Flash
                                        // location
  FCTL = 0x00;                          // Exit from programming mode
                                        // and lock the Flash Controller
}
```

To exit programming mode and lock the Z8F64xx Flash Controller, write 00h to the Flash Control Register.

Page Erase Operation

Flash memory is erased one page (512 bytes) at a time. Erasing a page in Flash memory sets all bytes on that page to the value FFh. The Flash Page Select Register identifies the page to be erased. When a Page Erase is complete, the Flash Controller returns to its locked state.

Page Erase on Z8F640x Devices

With the Flash Controller unlocked, follow the steps below to perform a Page Erase operation on an Z8F640x MCU:

1. Write the page to be erased to the Flash Page Select Register.
2. Write the Page Erase command 95h to the Flash Control Register.

The following code describes how to erase a page from the Z8F640x Flash memory.

```
*********************************************************************
In the erase_flash() function, the parameter 'addr' is the absolute
Flash memory address containing the written data.
*********************************************************************
void erase_flash(unsigned int addr)   // For Z8F640x MCUs
{
  unsigned int pagenum;               // 'pagenum' is the page to be
                                      // erased
  pagenum = (addr >> 9);              // Get the page to be erased from
                                      // absolute Flash address
  FPS=pagenum;                        // Pageno. of page to be erased is
                                      // loaded into FPS register
  FCTL=0x95;                          // Page erase command
}
```

Note: *The Page Erase operation can also be performed through the On-Chip Debugger, wherein the Flash Status Register must be polled to determine when the Page Erase operation is complete.*

Page Erase in Z8F64xx Devices

With the Flash Controller unlocked, follow the steps below to perform a Page Erase operation on an Z8F64xx MCU:

1. Write 00h to the Flash Control Register to reset the Flash Controller.
2. Write the page to be erased to the Flash Page Select Register.
3. Write the first unlock command, 73h, to the Flash Control Register.
4. Write the second unlock command, 8Ch, to the Flash Control Register.
5. Rewrite the page number written in Step 2 to the Flash Page Select Register.
6. Write the Page Erase command, 95h, to the Flash Control Register.

Note: *Only pages located in unprotected (unlocked) sectors can be erased.*

```
**********************************************************************
In the erase_flash() function, the parameter ‘pagenum’ is the page to be
erased.
**********************************************************************

void erase_flash(unsigned int pagenum) // For Z8F642x MCUs
{
  FCTL = 0x00;
  FPS = pagenum;                       // Page to be erased
  FCTL = 0x73;                         // First unlock command
  FCTL = 0x8C;                         // Second unlock command
  FPS = pagenum;                       // Page to be erased
  FCTL = 0x95;                         // Page erase command
}
```

Mass Erase Operation

The Mass Erase operation is possible only on Z8F640x MCUs, where a Mass Erase operation erases the entire contents of Flash memory using the Flash Controller. Flash memory cannot be mass-erased by user code on the Z8F64xx MCUs unless the Flash Controller is accessed using the On-Chip Debugger. For more details, see Z8 Encore! XP® Flash Control Register Descriptions on page 2.

Mass-erasing Flash memory on Z8F640x MCUs sets all bytes to the value FFh. Flash memory can be mass erased by user program code. However, after a Mass Erase operation, the user program code is also completely erased. When the Mass Erase operation is complete, the Flash Controller returns to its locked state.

To Mass Erase Flash memory on Z8F640x MCUs, unlock the Flash Controller and write 63h to the Flash Control Register.

Summary

Flash memory on Z8 Encore! XP MCUs is accessed via the Flash Control Registers. The older Z8F640x MCUs were improved up on and an enhanced Flash protection scheme is currently available in the newer Z8F64xx MCUs. Modifications made in Z8F64xx MCUs achieve a higher granularity of protection and flexibility. Application-level Flash Erase and Flash Program routines are impacted on Z8F64xx MCUs. The following list contains the highlights of the changes on Z8F64xx MCUs as compared to the Z8F640x MCUs:

- Flash memory is divided into eight sectors, each of which is individually protected. User software can select different Flash sectors for protection.
- Page-level division of Flash memory is maintained. Only pages of an unprotected sector can be programmed or erased. For an added level of protection, user software must reconfirm any page number selected for unlocking.
- Page-level protection is enhanced to confine byte-programming only within an unlocked page.
- A mass erase operation is permitted only through the debug port. Application software cannot perform a mass erase.

Reference

The document associated with Z8 Encore! XP available on www.zilog.com is provided below:

- Z8 Encore! XP® F64xx Series Flash Microcontrollers Product Specification (PS0199). This document discusses the Z8F64xx, Z8F482x, Z8F322x, Z8F242x, and Z8F162x devices.

Warning: DO NOT USE IN LIFE SUPPORT

LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

Document Disclaimer