

# Technical Note

## Password Protecting Flash Memory Blocks

---

### Introduction

Protecting the contents of Flash memory helps protect both intellectual property and service revenue. Ensuring that code is not inadvertently or maliciously changed and preventing users from modifying how the device is provisioned can reduce service costs and returns and can protect your business from fraud and theft.

### Protection Methods

Several methods can be used to protect Flash memory, one-time-programmable memory, block locking, and passwords are among the most common methods used. This document focuses on password protecting blocks, which is supported by a variety of Micron NOR Flash devices, including M29EW and some M29W family devices. However, because available methods for block password protection vary on each device, check the data sheet for your device to verify which methods are supported.

NOTE: This document focuses on password protecting blocks using Micron's Axcell™ M29EW device as an example. All sample code, steps, and configuration settings apply to the M29EW device. Check the data sheet for your device for specific configuration settings.

### Password Protection

Legacy Flash memory often has the capability to protect blocks using block locking. Block locking acts like a switch to enable or disable modifications. However, anyone could issue a command to unlock the blocks. Password protection takes protection a step further by requiring a password to lock or unlock the blocks. In this document we use the Micron's M29EW device to illustrate the password protection mechanism. Sample Software Functions on page 6 contains sample C code for password protection commands. A general familiarity with C programming is assumed.

Enabling password protection requires several steps, but once enabled the device will always power up with protected blocks locked. To enable password protection:

1. Program the 64-bit password.
2. Set the nonvolatile protection bit (NVPB) for each block to be locked.
3. Set the password protection mode to power the device with password protection enabled.

### Setting the Password Configuration

The first step in configuring the password is to enter the password protection command mode. This mode allows the execution of device commands related to password protection.

**Table 1: Block Protection Commands, 8-Bit Mode<sup>1, 2, 3</sup>**

Command	Length	Bus Operations																					
		1st		2nd		3rd		4th		5th		6th		7th		8th		9th		10th		11th	
		Ad	Data	Ad	Data	Ad	Data	Ad	Data	Ad	Data	Ad	Data	Ad	Data	Ad	Data	Ad	Data	Ad	Data	Ad	Data
ENTER PASSWORD PROTECTION COMMAND SET <sup>4</sup>	3	AAA	AA	555	55	AAA	60	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
PASSWORD PROGRAM <sup>5, 6</sup>	2	X	A0	PW An	PWD n	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
PASSWORD READ	8	00	PWD 0	01	PWD 1	02	PWD 2	03	PWD 3	04	PWD 4	06	PWD 5	06	PWD 6	07	PWD 7	–	–	–	–	–	–
PASSWORD UNLOCK <sup>6</sup>	11	00	25	00	03	00	PWD 0	01	PWD 1	02	PWD 2	03	PWD 3	04	PWD 4	05	PWD 5	06	PWD 6	07	PWD 7	00	29

**Table 2: Block Protection Commands, 16-Bit Mode<sup>1, 2, 3</sup>**

Command	Length	Bus Operations													
		1st		2nd		3rd		4th		5th		6th		7th	
		Ad	Data	Ad	Data	Ad	Data	Ad	Data	Ad	Data	Ad	Data	Ad	Data
ENTER PASSWORD PROTECTION COMMAND SET <sup>4</sup>	3	555	AA	2AA	55	555	60	–	–	–	–	–	–	–	–
PASSWORD PROGRAM <sup>5, 6</sup>	2	X	A0	PWAn	PWDn	–	–	–	–	–	–	–	–	–	–
PASSWORD READ	4	00	PWD0	01	PWD1	02	PWD2	03	PWD3	–	–	–	–	–	–
PASSWORD UNLOCK <sup>6</sup>	7	00	25	00	03	00	PWD0	01	PWD1	02	PWD2	03	PWD3	00	29

- Notes:**
1. Ad = Address; Dat = Data; BAd = Any address in the block; RD = Read data; PWDn = Password byte 0–3; PWAn = Password address (n = 0–3); X = “Don’t Care.” All values in the table are in hexadecimal.
  2. Grey cells represent READ cycles, the other cells are WRITE cycles.
  3. DQ[15:8] are “Don’t Care” during unlock and command cycles. A[MAX:16] are “Don’t Care” during unlock and command cycles unless an address is required.
  4. An enter command sequence must be issued prior to any operation. It disables READ and WRITE operations from and to block 0. READ and WRITE operations from any other block are allowed.
  5. Only one portion of the password can be programmed or read by each PASSWORD PROGRAM command.
  6. The password portion can be entered or read in any order as long as the entire 64-bit password is entered or read.

As described in the command sequence in the tables, entering the password protection mode is a three-step command:

```
FlashWrite( ConvAddr(0x00555), CMD(0x00AA) ); /* 1st Cycle */
FlashWrite( ConvAddr(0x002AA), CMD(0x0055) ); /* 2nd Cycle */
FlashWrite( ConvAddr(0x00555), CMD(0x0040) ); /* 3rd Cycle */
```

### Program the 64-Bit Password

In the previous example, the first parameter is the address and the second parameter is the command to place on the data bus. Once the three commands have been issued, the M29EW device will accept commands to set the password and enable the NVPB for the desired blocks.

Once the command mode has been entered, the password can be set and the NVPB set for the blocks that are to be password protected. For more details, see FlashSetPassword-Protection on page 6.

The password program is a multistep command. Once the device is in password protection mode, the password can be set by issuing an A0 command followed by the data. For devices in x16 mode, this sequence would be repeated four times (64 bits). For devices in x8 mode, it would be repeated eight times. The following example is for devices in x8 mode:

```
for(i=0;i<8;i++)
{
    FlashWrite( ANY_ADDR, CMD(0x00A0) );
    FlashWrite( i , CMD(*(ucpPWD+i)) );
}
```

The previous example assumes that the PWD is referenced by a character (8-bit) pointer (ucpPWD), which is indexed each pass through for the loop to write the password to the device. For more details, see FlashPasswordProgram page 8.

### Set the NVPB for Each Block to be Locked

The next step in enabling password protection is programming the NVPB for each block to be protected.

```
/* Program the VPB */
FlashWrite( ANY_ADDR, CMD(0x00A0) ); /* 1st Cycle */
FlashWrite( BlockOffset(ublBlockNr), CMD(0x0000) ); /* 2nd
Cycle */
```

Programming the NVPB for each block is again a two-cycle command. First, the command 0xA0 is issued to the device. Then, the command 0x00 is issued to each block to be protected. As with standard programming operations, the device status register should be polled until the operation completes, and then the status register checked for results. For more details, see FlashSetBlockNVPB on page 9. The NVPB for each block can be modified, unless the global NVPB bit is set. The following command sequence sets the global protection bit to prevent changes:

```
/* Program the Non-volatile Protection Bit*/
FlashWrite( ANY_ADDR, CMD(0x00A0) ); /* 1st Cycle */
```

```
FlashWrite( BlockOffset(ublBlockNr) , CMD(0x0000) ); /*  
2nd Cycle */
```

Again, the device must be polled until the operation completes. For more details, see FlashSetNVPBLockBit on page 11.

## Enable the Password Protection Mode

To complete configuration, the password protect mode must be enabled. This is to ensure the device is locked when it powers up:

```
/* Program the Password mode lock bit*/  
FlashWrite( ANY_ADDR, CMD(0x00A0) ); /* 1st Cycle */  
FlashWrite( ANY_ADDR,  
CMD(ucProtStatus&(~PASSWORD_MODE_LOCKBIT)) );  
/* 2nd Cycle */
```

After configuration is complete, you must exit the Flash protection configuration mode. This is a simple command sequence that returns the device to normal Flash read array mode.

```
/* Send the Exit Protection command to the device */  
FlashWrite( ANY_ADDR, (uCPUBusType)CMD(0x0090) ); /* 1st Cycle  
*/  
FlashWrite( ANY_ADDR, (uCPUBusType)CMD(0x0000) ); /* 2nd Cycle  
*/
```

## Unlocking Blocks

After the password protect mode has been enabled, protected blocks cannot be modified without entering the password. Entering the password removes protection from all NVPB bits. As with configuring the password protection, the first step is entering the password protect command mode, then issuing the password unlock command:

```
/* Write password unlock command (4 words) */  
FlashWrite( 0x00000, CMD(0x0025) ); /* 1st Cycle */  
FlashWrite( 0x00000, CMD(0x0003) ); /* 2nd Cycle */
```

This is followed by the password (again, four or eight writes will be required depending on x16 or x8 mode). Assuming x8 mode:

```
for(i=0;i<8;i++)  
{  
FlashWrite( i, CMD(*ucpPWD+i));  
}
```

Once the password has been entered, the UNLOCK command must be confirmed:

```
FlashWrite( 0x00000, CMD(0x0029) );
```

Again, as with PROGRAM or ERASE commands, it is important to poll the status register until the UNLOCK is completed and the PROTECTION command set can be exited (as before) to return to read array mode.

### Conclusion

Protecting Flash blocks is an effective method for protecting devices from modification. Using passwords improves protection when compared to a simple block lock toggle. Reference code for a complete driver that implements support for the M29EW and its protection features is available for download: <http://www.micron.com/products/nor-flash/nor-flash-software>.

For more information or additional support, contact your Micron representative.

## Sample Software Functions

### ReturnType FlashSetPasswordProtection( void ) Function

```
/******
```

```
Function:      ReturnType FlashSetPasswordProtection( void )
```

```
Arguments:     None
```

```
Return Values: The function returns the following conditions:
```

```
Flash_Success,
```

```
Flash_SpecificError,
```

```
Description:   This function set the device into Password Protection mode
```

```
Pseudo Code:
```

```
Step 1: Send Enter Lock Register Command Set command
```

```
Step 2: Read Lock Register
```

```
Step 3: Judge the Lock Register, if in Password mode or NVP mode, then return
```

```
Step 4: program the password mode lock bit
```

```
Step 5: Follow Data Toggle Flow Chart until Program/Erase Controller completes
```

```
Step 6: Return to Read Array mode
```

```
Step 7: Verify the program
```

```
*****/
```

```
ReturnType FlashSetPasswordProtectionMode( void ) {
```

```
ReturnType rRetVal; /* Holds the return value */
```

```
uCPUBusType ucProtStatus;
```

```
/* Step 1: Send Enter Lock Register Command Set command */
```

```
FlashWrite( ConvAddr(0x00555), CMD(0x00AA) ); /* 1st Cycle */
```

```
FlashWrite( ConvAddr(0x002AA), CMD(0x0055) ); /* 2nd Cycle */
```

```
FlashWrite( ConvAddr(0x00555), CMD(0x0040) ); /* 3rd Cycle */
```

```
/* Step 2: Read Lock Register*/
```

```
ucProtStatus = FlashRead( ANY_ADDR ) ;
```

```
/* Step 3a: Judge the Lock Register, if in Password mode, then return */
```

```
if((ucProtStatus&PASSWORD_MODE_LOCKBIT)==0)
```

```
{
```

```
    rRetVal = Flash_Password_Protection_Mode;
```

```
    FlashExitProtection(); /*exit protection command set*/
```

```
    return rRetVal;
```

```
    }

/* Step 3b: Judge the Lock Register, if already in NVP mode, then return*/
    if((ucProtStatus&NVP_MODE_LOCKBIT)==0)
    {
        rRetVal = Flash_NV_Protection_Mode;

        FlashExitProtection(); /*exit protection command set*/

    return rRetVal;
    }

/* Step 4: program the Password mode lock bit*/
    FlashWrite( ANY_ADDR, CMD(0x00A0) ); /* 1st Cycle */
    FlashWrite( ANY_ADDR, CMD(ucProtStatus&(~PASSWORD_MODE_LOCKBIT))); /* 2nd
        Cycle */

/* Step 5: Follow Data Toggle Flow Chart until Program/Erase Controller
completes */
if( FlashDataToggle() != Flash_Success ) {
    /* Return to Read mode (if an error occurred) */
        FlashWrite( ANY_ADDR, (uCPUBusType)CMD(0x00F0) ); /* Use single
instruction cycle method */
        rRetVal=Flash_ProgramFailed;
    }

/* Step 6: Exit Protection Command Set and return to Read Array mode */
    FlashExitProtection();

/* Step 7: Verify the program */
    rRetVal = FlashCheckProtectionMode();
    if(rRetVal == Flash_Password_Protection_Mode)
    {
        rRetVal = Flash_Success;
    }
    else
        rRetVal = Flash_ProgramFailed;
```

```

    return rRetVal;
} /* EndFunction FlashSetPasswordProtection */

```

## ReturnType FlashPasswordProgram( uCPUBusType \*ucpPWD ) Function

```

/*****

```

Function:            ReturnType FlashPasswordProgram( uCPUBusType \*ucpPWD )

Arguments:           uwPWD = Password to program

Return Values:      The function returns the following conditions:

    Flash\_Success

    Flash\_ProgramFailed

Description:        This function is used to set the password(64 bit) for password protection mode.

### Pseudo Code:

- Step 1: Send Enter Password Command Set command
- Step 2: Write password (1 word)
- Step 3: Wait until Program/Erase Controller has completed
- Step 4: Return to Read Array mode

```

*****

```

```

ReturnType FlashPasswordProgram( uCPUBusType *ucpPWD ) {

```

```

    ReturnType rRetVal = Flash_Success; /* Holds the return value */
    udword i,data;

```

```

/* Step 1: Send Enter Password Command Set command */

```

```

/*

```

#### Note:

For 2-Gbit (1-Gbit/1-Gbit) device, all the set-up command should be re-issued to the device when different die is selected.

```

*/

```

```

FlashWrite( ConvAddr(0x00555), CMD(0x00AA) ); /* 1st Cycle */

```

```

FlashWrite( ConvAddr(0x002AA), CMD(0x0055) ); /* 2nd Cycle */

```

```

FlashWrite( ConvAddr(0x00555), CMD(0x0060) ); /* 3rd Cycle */

```

```

/* Step 2: Write password (1 word) */

```

```

#if defined (USE_16BIT_FLASH)

```

```
        for(i=0;i<4;i++)
#endif
#if defined (USE_8BIT_FLASH)
        for(i=0;i<8;i++)
#endif
    {
        FlashWrite( ANY_ADDR, CMD(0x00A0) );
        FlashWrite( i , CMD(*(ucpPWD+i)) );
        FlashPause(2);
    }
    /* Step 4: Exit Protection Command Set and return to Read Array mode */
    FlashExitProtection();

    return rRetVal;

} /* EndFunction FlashPasswordProgram */
```

## ReturnType FlashSetBlockNVPB( uBlockType ublBlockNr ) Function

/\*\*\*\*\*\*

Function:            ReturnType FlashSetBlockNVPB( uBlockType ublBlockNr )

Arguments:           ublBlockNr = block number to be set

Note:                The first block is Block 0

Return Values:      The function returns the following conditions:

Flash\_BlockNrInvalid  
Flash\_Success  
Flash\_ProgramFailed  
Flash\_NonVolatile\_Unprotected  
Flash\_NonVolatile\_Unclear

Description:        This function is used to set the Non-Volatile Modify  
                    Protection bit of a block.

Pseudo Code:

- Step 1: Check Range of Block Number Parameter
- Step 2: verify the Non-volatile Protection Bit, if already set , then exit
- Step 3: Send Enter Non-volatile Protection command
- Step 4: Program the Non-volatile Protection Bit
- Step 5: Follow Data Toggle Flow Chart until Program/Erase Controller completes
- Step 6: Exit Protection Command Set and return to Read Array mode

Step 7: verify the NVPB state

\*\*\*\*\*/

```
ReturnType FlashSetBlockNVPB( uBlockType ublBlockNr ) {
    ReturnType rRetVal; /* Holds the return value */

    /* Step 1: Check that the block number exists */
    if ( ublBlockNr >= ublNumBlocks )
        return Flash_BlockNrInvalid;

    /* Step 2: Check the NVPB lok bit*/
    rRetVal = FlashCheckNVPBLockBit();

    if(rRetVal == Flash_NVPB_Unlocked)
    {
    /* Step 3: verify the Non-volatile Protection Bit*/
        rRetVal = FlashCheckBlockNVPB(ublBlockNr);

        if(rRetVal == Flash_NonVolatile_Unprotected)
        {
        /* Step 4: Send Enter Non-volatile Protection command */
            FlashWrite( ConvAddr(0x00555), CMD(0x00AA) ); /* 1st Cycle */
            FlashWrite( ConvAddr(0x002AA), CMD(0x0055) ); /* 2nd Cycle */
            FlashWrite( ConvAddr(0x00555), CMD(0x00C0) ); /* 3rd Cycle */

            /* Step 5: Program the Non-volatile Protection Bit*/
            FlashWrite( ANY_ADDR, CMD(0x00A0) ); /* 1st Cycle */
            FlashWrite( BlockOffset(ublBlockNr) , CMD(0x0000) ); /* 2nd Cycle */

            /* Step 6: Follow Data Toggle Flow Chart until Program/Erase Controller
            completes */
            if( FlashDataToggle() != Flash_Success ) {
                /* Return to Read mode (if an error occurred) */
                FlashWrite( ANY_ADDR, (uCPUBusType)CMD(0x00F0) ); /* Use single
                instruction cycle method */
                rRetVal=Flash_ProgramFailed;
            }

            /* Step 7: Exit Protection Command Set and return to Read Array mode */
        }
    }
}
```

```
FlashExitProtection();

/* Step 8: verify the NVPB state*/
rRetVal = FlashCheckBlockNVPB(ublBlockNr);
if(rRetVal == Flash_NonVolatile_Protected)
rRetVal = Flash_Success;
else
rRetVal = Flash_ProgramFailed;
}
}
return rRetVal;
} /* EndFunction FlashSetBlockNVPB */
```

### ReturnType FlashSetNVPBLockBit( void ) Function

```
/******
```

Function:            ReturnType FlashSetNVPBLockBit( void )

Arguments:           none

Return Values:      The function returns the following conditions:

Flash\_Success

Flash\_NVPB\_Locked

Flash\_NVPB\_Unclear

Flash\_ProgramFailed

Description:        This function used to set the NVPB Lock Bit.

Note that:

- 1) there is only one NVPB Lock Bit per device which is volatile.
- 2) There is no software way to clear(erase to '1') of this bit unless in Password Protection Mode
- 3) It can be clear(erase to '1') by hardware means in Nonvolatile Protection mode, like a power up or a hardware reset.

Pseudo Code:

- Step 1: check NVPB Lock Bit, if locked then exit
- Step 2 : Send Enter NVPB Lock Bit Command Set command
- Step 3: Send Program NVPB Lock Bit command
- Step 4: Follow Data Toggle Flow Chart until Program/Erase Controller completes
- Step 5: Exit Protection Command Set and return to Read Array mode
- Step 6: Verify NVPB Lock Bit

```
*****/
```

```
ReturnType FlashSetNVPBLockBit( void) {
```

```
    ReturnVal = Flash_Success; /* Holds the return value */

/* Step 1 : check NVPB Lock Bit*/
    rRetVal = FlashCheckNVPBLockBit();
    if(rRetVal == Flash_NVPB_Unlocked)
    {
/* Step 2: Send Enter NVPB Lock Bit Command Set command */
        FlashWrite( ConvAddr(0x00555), CMD(0x00AA) ); /* 1st Cycle */
        FlashWrite( ConvAddr(0x002AA), CMD(0x0055) ); /* 2nd Cycle */
        FlashWrite( ConvAddr(0x00555), CMD(0x0050) ); /* 3rd Cycle */

/* Step 3: Send Program NVPB Lock Bit command*/
        FlashWrite( ANY_ADDR, CMD(0x00A0) ); /* 1st Cycle */
        FlashWrite( ANY_ADDR, CMD(0x0000) ); /* 2nd Cycle */

/* Step 4: Follow Data Toggle Flow Chart until Program/Erase Controller completes
           */
        if( FlashDataToggle() != Flash_Success ) {
            /* Return to Read mode (if an error occurred) */
            FlashWrite( ANY_ADDR, (uCPUBusType)CMD(0x00F0) ); /* Use single instruction
                cycle method */
            rRetVal=Flash_ProgramFailed;
        }

/* Step 5: Exit Protection Command Set and return to Read Array mode */
        FlashExitProtection();

/* Step 6: Verify NVPB Lock Bit*/
        rRetVal = FlashCheckNVPBLockBit();
        if(rRetVal == Flash_NVPB_Locked)
            rRetVal = Flash_Success;
        else
            rRetVal = Flash_ProgramFailed;
    }

    return rRetVal;

} /* EndFunction FlashSetNVPBLockBit */
```

## ReturnType FlashSetPasswordProtection( void ) Function

/\*\*\*\*\*\*

Function: ReturnType FlashSetPasswordProtection( void )

Arguments: None

Return Values: The function returns the following conditions:

Flash\_Success,  
Flash\_SpecificError,

Description: This function set the device into Password Protection mode

Pseudo Code:

Step 1: Send Enter Lock Register Command Set command

Step 2: Read Lock Register

Step 3: Judge the Lock Register, if in Password mode or NVP mode, then return

Step 4: program the password mode lock bit

Step 5: Follow Data Toggle Flow Chart until Program/Erase Controller completes

Step 6: Return to Read Array mode

Step 7: Verify the program

\*\*\*\*\*/

```
ReturnType FlashSetPasswordProtectionMode( void ) {
```

```
    ReturnType rRetVal; /* Holds the return value */
```

```
    uCPUBusType ucProtStatus;
```

```
/* Step 1: Send Enter Lock Register Command Set command */
```

```
    FlashWrite( ConvAddr(0x00555), CMD(0x00AA) ); /* 1st Cycle */
```

```
    FlashWrite( ConvAddr(0x002AA), CMD(0x0055) ); /* 2nd Cycle */
```

```
    FlashWrite( ConvAddr(0x00555), CMD(0x0040) ); /* 3rd Cycle */
```

```
/* Step 2: Read Lock Register*/
```

```
    ucProtStatus = FlashRead( ANY_ADDR );
```

```
/* Step 3a: Judge the Lock Register, if in Password mode, then return */
```

```
    if( (ucProtStatus & PASSWORD_MODE_LOCKBIT) == 0 )
```

```
    {
```

```
        rRetVal = Flash_Password_Protection_Mode;
```

```
        FlashExitProtection(); /*exit protection command set*/
```

```
    return rRetVal;
```

```
}

/* Step 3b: Judge the Lock Register, if already in NVP mode, then return*/
if((ucProtStatus&NVP_MODE_LOCKBIT)==0)
{
    rRetVal = Flash_NV_Protection_Mode;

    FlashExitProtection(); /*exit protection command set*/

    return rRetVal;
}

/* Step 4: program the Password mode lock bit*/
FlashWrite( ANY_ADDR, CMD(0x00A0) ); /* 1st Cycle */
FlashWrite( ANY_ADDR, CMD(ucProtStatus&(~PASSWORD_MODE_LOCKBIT))); /* 2nd
Cycle */

/* Step 5: Follow Data Toggle Flow Chart until Program/Erase Controller completes
*/
if( FlashDataToggle() != Flash_Success ) {
/ * Return to Read mode (if an error occurred) */
FlashWrite( ANY_ADDR, (uCPUBusType)CMD(0x00F0) ); /* Use single instruction
cycle method */
rRetVal=Flash_ProgramFailed;
}

/* Step 6: Exit Protection Command Set and return to Read Array mode */
FlashExitProtection();

/* Step 7: Verify the program */
rRetVal = FlashCheckProtectionMode();
if(rRetVal == Flash_Password_Protection_Mode)
{
    rRetVal = Flash_Success;
}
else
rRetVal = Flash_ProgramFailed;
```

```
return rRetVal;
```

```
} /* EndFunction FlashSetPasswordProtection */
```

## Void FlashExitProtection (void) Function

```
/******
```

Function:           void FlashExitProtection (void);

Arguments:         None

Return Value:      None

Description:       This function is used to send the Exit Protection command to the device

Pseudo Code:

    Step 1: Send the Exit Protection command to the device

```
*****/
```

```
void FlashExitProtection( void ){
```

```
    /* Step 1: Send the Exit Protection command to the device */
```

```
        FlashWrite( ANY_ADDR, (uCPUBusType)CMD(0x0090) ); /* 1st Cycle */
```

```
        FlashWrite( ANY_ADDR, (uCPUBusType)CMD(0x0000) ); /* 2nd Cycle */
```

```
    } /* EndFunction FlashExitProtection */
```

## ReturnType FlashPasswordProtectionUnlock ( uCPUBusType \*ucpPWD ) Function

/\*\*\*\*\*\*

Function:            ReturnType FlashPasswordProtectionUnlock( uCPUBusType \*ucpPWD)

Arguments:          \*uwPWD = Password to write

Return Values:     The function always returns :

    Flash\_Success

Description:        This function is used to clear the NVPB Lock Bit under Password Protect mode.

    This operation will unprotect all Non-volatile Modify Protection bits when the device is in

    Password Protection mode.

Pseudo Code:

    Step 1: Send unlock cycle and issue program command

    Step 2: Write password (4 words)

    Step 3: Follow Data Toggle Flow Chart until Program/Erase Controller completes

    Step 4: Exit Protection Command Set

\*\*\*\*\*/

```
ReturnType FlashPasswordProtectionUnlock( uCPUBusType *ucpPWD ) {
```

```
    ReturnType rRetVal;
```

```
    ubyte i = 0;
```

```
    /* Step 1a: Check Input parameters */
```

```
if (NULL == ucpPWD)
```

```
    return Flash_ResponseUnclear;
```

```
/* Step 1: Send Enter Password Command Set command */
```

```
    FlashWrite( ConvAddr(0x00555), CMD(0x00AA) ); /* 1st Cycle */
```

```
    FlashWrite( ConvAddr(0x002AA), CMD(0x0055) ); /* 2nd Cycle */
```

```
    FlashWrite( ConvAddr(0x00555), CMD(0x0060) ); /* 3rd Cycle */
```

```
/* Step 2: Write password (4 words) */
```

```
    FlashWrite( 0x00000, CMD(0x0025) ); /* 1st Cycle */
```

```
    FlashWrite( 0x00000, CMD(0x0003) ); /* 2nd Cycle */
```

```
    #if defined (USE_16BIT_FLASH)
```

```
        for(i=0;i<4;i++)
```

```
#endif
#if defined (USE_8BIT_FLASH)
    for(i=0;i<8;i++)
#endif
{
    FlashWrite( i, CMD(*(ucpPWD+i)));
}

FlashPause(10);

FlashWrite( 0x00000, CMD(0x0029) ); /*7th Cycle*/

/* Step 3: Follow Data Toggle Flow Chart until Program/Erase Controller completes
*/
if( FlashDataToggle() != Flash_Success ) {
    /* Return to Read mode (if an error occurred) */
    FlashWrite( ANY_ADDR, (uCPUBusType)CMD(0x00F0) ); /* Use single instruction
        cycle method */
    rRetVal=Flash_ProgramFailed;
}

/* Step 4: Exit Protection Command Set */
FlashExitProtection();

return Flash_Success;

} /* EndFunction FlashPasswordProtectionUnlock */
```

8000 S. Federal Way, P.O. Box 6, Boise, ID 83707-0006, Tel: 208-368-3900  
www.micron.com/productsupport Customer Comment Line: 800-932-4992

Micron and the Micron logo are trademarks of Micron Technology, Inc. All other trademarks are the property of their respective owners.



## **Revision History**

<b>Rev. C</b> .....	<b>.01/12</b>
<ul style="list-style-type: none"><li>• Updated links to <a href="http://micron.com">micron.com</a></li></ul>	
<b>Rev. B</b> .....	<b>.05/11</b>
<ul style="list-style-type: none"><li>• Added description of supported devices</li><li>• Updated links to <a href="http://micron.com">micron.com</a></li></ul>	
<b>Rev. A</b> .....	<b>.03/11</b>
<ul style="list-style-type: none"><li>• Initial release</li></ul>	