AP-475

APPLICATION NOTE

Using the 8XC196NT

RICHARD N. EVANS
CHRISTINE NEFFENGER
APPLICATIONS ENGINEERS

January 1994

Order Number: 272315-001

Information in this document is provided in connection with Intel products. Intel assumes no liability whatsoever, including infringement of any patent or copyright, for sale and use of Intel products except as provided in Intel's Terms and Conditions of Sale for such products.

Intel retains the right to make changes to these specifications at any time, without notice. Microcomputer Products may have minor variations to this specification known as errata.

*Other brands and names are the property of their respective owners.

†Since publication of documents referenced in this document, registration of the Pentium, OverDrive and iCOMP trademarks has been issued to Intel Corporation.

Contact your local Intel sales office or your distributor to obtain the latest specifications before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature, may be obtained from:

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

or call 1-800-879-4683

Using the 8XC196NT

CONTENTS

PAGE

CONTENTS

PAGE

CONTENTS

PAGE

CONTENTS

PAGE

1.0 INTRODUCTION

The 8XC196NT can address 1 Mbyte of linear address space which is beyond the standard MCS®-96 64 Kbyte address space. Many applications require much memory whether it be from large code sizes due to high level language compilation or from large data tables. The standard MCS-96 family member has 16 address lines which allow linear access to 64 Kbytes of address space. To accommodate a growing need for address space the 8XC196NT was designed with 20 external address lines. The 8XC196NT is an upgrade from the standard MCS-96 family member. There are many similarities and a few differences that must be noted. The same peripheral set exists on the 8XC196NT as on the 8XC196KR. The peripherals on the 8XC196NT and the 8XC196KR are more advanced than the peripherals on the 8X96BH, 8XC196KB, 8XC196KC and 8XC196KD. The instruction set is a superset of the MCS-96 family containing special extended addressing instructions. The bus controller on the 8XC196NT has new modes which allow for slower memories to be utilized resulting in cost savings. An extended address port (EPORT) adds 4 more address lines to the external bus. The EPORT can also be used as an I/O port if the extended address lines are not needed.

2.0 ARCHITECTURAL OVERVIEW

The architecture for the 8XC196NT is consistent with the previous members (see Figure 2-1). The program counter has been extended to 24 bits to accommodate the extended addressing capabilities. The ALU is tied directly to the register RAM which creates a register-register architecture. The 8XC196NT has 1000 bytes of register RAM. Hence, one thousand accumulators (each a byte wide) are possible. This many accumulators minimizes the number of load/store operations. Notice that there are 32 Kbytes of on-chip (EP)ROM available and 512 bytes of internal RAM. The internal RAM can be used just like external RAM to execute code or hold data.

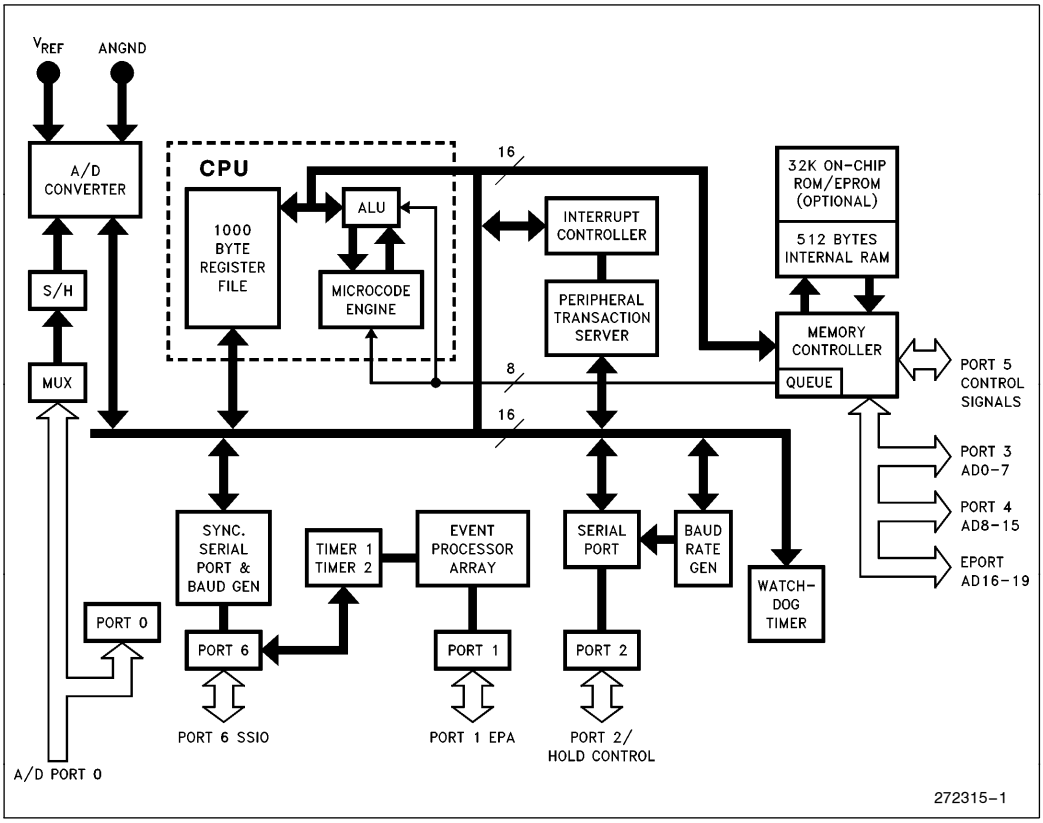


Figure 2-1. 8XC196NT Block Diagram

The peripheral set on the 8XC196NT includes the following:

- Serial Port (3 asynchronous modes, 1 synchronous mode)
- SSIO (Synchronous Serial I/O Port with bi-directional clocking with two separate data and two separate clock channels)
- Slave Port
- EPA (Event Processor Array—high speed input capture and output compare) 10 channels
- PTS (Peripheral Transaction Server—microcoded interrupt service routine)
- A/D converter (4 channel 8/10 bit resolution with programmable sample and convert times)
- Two 16-bit timers
- Watchdog Timer
- Dedicated 15-bit Baud Rate Generator
- Extended Address Port (EPORT)

See pin out diagram in Figure 2-2.

For more information on the peripherals consult the 8XC196NT or 8XC196KX user's guides. The next sections go into more detail on how to use the EPORT, the memory map and the bus controller improvements which allow for greater flexibility in choosing memories. Also, code examples are given where needed as well as memory interface diagrams.

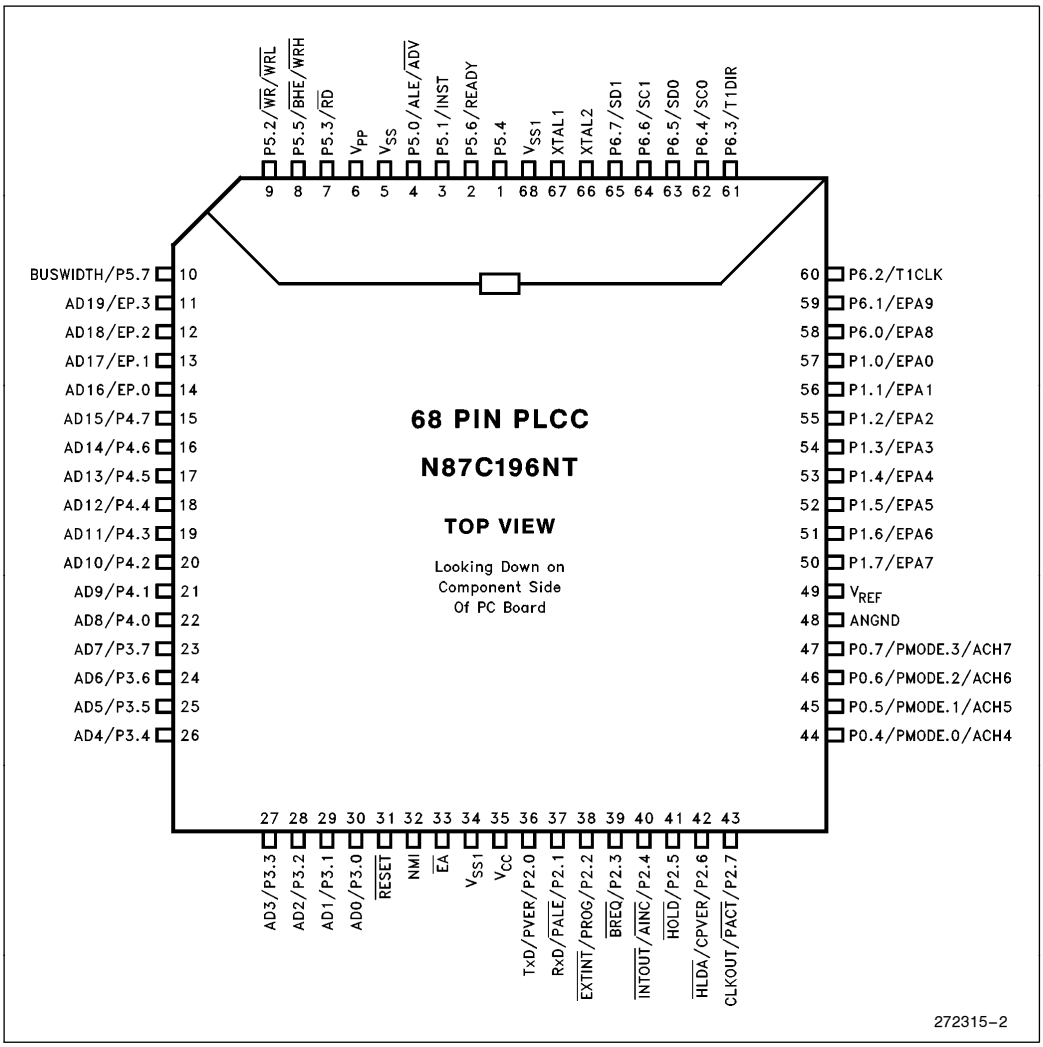


Figure 2-2. 8XC196NT Pin Out Diagram

3.0 8XC196NT NEW INSTRUCTIONS

There are a few new instructions for accessing the 1 Mbyte address space. They are listed below and explained in detail in the following pages. When executing in 24-bit mode, instructions which push or pop the PC onto the stack will decrement or increment the stack pointer by four. These instructions include LCALL, SCALL, and RET. RET is explained in the new instruction section.

New Instructions:

EBMOVI—Interruptable Extended Block Move

EBR—Extended Branch, Indexed

ECALL—Extended Subroutine Call

EJMP—Extended Jump

ELD/ELDB—Extended Load Word/Byte, Indexed or Indirect

EST/ESTB—Extended Store Word/Byte, Indexed or Indirect

RET—24-Bit Mode Return from Subroutine

3.1 ELD/ELDB—Extended Load Word/Byte

Operation:

The word/byte contents of the source—indirect, indirect auto-increment, or index addressed—are copied into the register indicated by the destination. The source is located at a 24-bit address.

Assembly Language:

```
     DST   SRC
ELD wreg, waopi
```

where:

wreg = a word register in the internal register file.

waopi = a word operand 24-bit indirect or indexed addressed. Even address boundary, so LSB = 0 if indirect and LSB = 1 if indirect auto-increment.

Object Code Format:

Instruction	->	Object code format
ELD wreg, [wiop]	->	[E8][wiop 0][wreg]
ELD wreg, [wiop]+	->	[E8][wiop 1][wreg]
ELD wreg, xx[wiop]	->	[E9][wiop][xx][wreg]
ELDB breg, [biop]	->	[EA][biop 0][breg]
ELDB breg, [biop]+	->	[EA][biop 1][breg]
ELDB breg, xx[biop]	->	[EB][biop][xx][breg]

Example:

```
rseg at 1CH
A:     dsb  1
B:     dsw  2

cseg at 0FF2080h

ld     B,#01FE6h  ; load low word of B with low word of 24-bit address
ldbze  B+2,#00h   ; load high word of B with highest byte of 24-bit address
eldb   A,[B]      ; (A) <= (001FE6h), which reads EP_PIN register
```

3.2 EST/ESTB—Extended Store Word/Byte

Operation:

The word/byte contents of the source—indirect, indirect auto-increment*, or index addressed—are copied into the register indicated by the destination. The source is located at a 24-bit address.

Assembly Language:

```
     SRC   DST
EST wreg, wiop
```

where:

wreg = a word register in the internal register file

wiop = a word operand 24-bit indirect or indexed addressed modes. Even address boundary, so LSB = 0 for indirect and LSB = 1 for indirect auto-increment.

Object Code Format:

```
Instruction            ->  Object code format
EST  wreg, [wiop]      ->  [1C][wiop][wreg]
EST  wreg, xx[wiop]    ->  [1D][wiop 0][xx][wreg]
EST  wreg, xx[wiop]+   ->  [1D][wiop 1][xx][wreg]
ESTB breg, [biop]      ->  [1E][biop 0][breg]
ESTB breg, [biop]+     ->  [1E][biop 1][breg]
ESTB breg, xx[biop]    ->  [1F][biop][xx][breg]
```

Example:

```
rseg at 1CH
A:     dsb  1
B:     dsw  2

cseg at 0FF2080h
ldb    A,#055h    ; load A with value
ld     B,#01FFCh  ; load low word of B with low word of extended address
ldbze  B+2,#00h   ; load high word of B with highest byte of extended address
estb   A,[B]      ; (001FFCh) <= 055h, which writes to P3_REG, a location
                  ; unavailable with windowing or with 'stb' if EP_REG not
                  ; set to 00h
```

3.3 EBR—Extended Branch

Operation:

Execution continues at the extended address specified in the operand register.

Assembly Language:
`EBR [dwreg]`

where:

dwreg = double word register containing 24-bit address of the branch location

Object Code Format:
`[E3][dwreg]`

Example:

```
rseg at 1CH
B:     dsw  2

cseg at 0FF2080h

.
.
ld     B,#2080h
ldbze  B+2,#00h  ; load B with external address for execution
ebr    [B]       ; load PC with 24-bit address in B (002080h)
```

3.4 EBMOVI—Interruptable Extended Block Move

Operation:

This instruction is used to move a block of word data from one location in extended memory to another and is interruptable. The source and destination registers are calculated using the indirect auto-increment addressing modes. A long register addresses the source and destination pointers which are stored in adjacent double word registers. The number of transfers is specified in the word register. The blocks of data can reside anywhere in memory, but should not overlap.

Assembly Language:
`BMOVI  qwreg,wreg`

where:

qwreg = a quad-word register

Object Code Format:
`[E4][wreg][dLreg]`

Example:

```
rseg at 1CH
ptrs:  dsl  2
count: dsw  1

cseg at 0FF2080h

ld      count,#1000h
ld      ptrs,#4000h
ld      ptrs+2,#0003h
ld      ptrs+4,#3000h
ld      ptrs+6,#0005h
ebmovi ptrs,count      ; moves 1000h words of data from 034000h through
                       ;035000h to 053000h through 054000h
```

3.5 ECALL—Extended Subroutine call

Operation:

The contents of the program counter (the return address) are pushed onto the stack*. Then the distance from the end of the instruction to the target label is added to the program counter, effecting the call. The offset from the end of the instruction to the call must be in the range of −8,388,608 to +8,388,607 inclusive, which is a 24-bit offset.

Assembly Language:
```
ECALL  label
```

Object Code Format:
```
[F1][24-bit offset]
```

NOTE:
*The PC is pushed onto the stack as 4 bytes or 32-bits.

3.6 RET—24-Bit Mode Return

Operation:

The PC is popped off the stack.

Assembly Language:
```
RET
```

Object Code Format: `[F0]`

NOTE:
Since 32-bits were pushed onto the stack at call of subroutine, 32-bits are popped. Therefore, RET in 24-bit mode will execute the following:
```
PC <= SP
SP = SP+4
```

3.7 EJMP—Extended Jump

Operation:

The distance from the end of this instruction to the target label is added to the program counter, effecting the jump. The operand may be any address in the entire address space.

Assembly Language:
```
EJMP cadd
```

Object Code Format:
```
[E6][24-bit displacement]
```

4.0 EPORT

4.1 EPORT Gives 20-Bit External Address

The main feature of the 8XC196NT is its ability to address 1 Mbyte of external memory plus over 32 Kbytes of internal memory on the device. This is accomplished using the extended address port, which is the EPORT. The EPORT can be configured as four additional address lines. Therefore, operating the device in 24-bit mode, external memory is accessed with a **20-bit address** (pins: EP.3 → EP.0 and AD15 → AD0). The memory map in Figure 5-4 shows the address range accessible using the 20-bit external memory address. The EPORT replaces four of the A/D channels, therefore the 8XC196NT has only four remaining A/D channels. The pin diagram of the 8XC196NT 68-pin PLCC is shown in Figure 2-2.

4.2 EPORT is Address Lines or I/O Pins

The EPORT can be used as either additional **address lines or I/O pins**. In addition, the port can function as any combination of additional address lines or I/O pins. For example, two of the EPORT pins can be used as address lines for two additional address bits and the other lines for two additional I/O pins. The user must be aware that this configuration as address or I/O can be changed during normal execution using the EPORT control registers. Therefore, caution should be used when changing the control registers of the EPORT.

4.3 EPORT SFRs

The function, direction, and data of the EPORT are controlled by four special function registers (SFRs) which are: **EP__MODE (1FE1h)**, **EP__DIR (1FE3h)**, **EP__REG (1FE5h)**, and **EP__PIN (1FE7h)**. Table 4-1 shows the SFR control of the pins.

Table 4-1. EPORT SFRs Control of EPORT Pins

EP__Mode	EP__Dir	EP__Reg	Pull-Up	Pull-Down	Pin Function
x	0	0	Off	On	Output 0
x	0	1	On	Off	Output 1
x	1	0	Off	On	Open-Drain 0
x	1	1	Off	Off	Open-Drain 1 (Input)

4.3.1 EP__MODE REGISTER

The **EP__MODE register** determines the function of the EPORT pins. This register is written as a byte. Each bit of the EP__MODE indicates whether the pin will be a I/O port pin (EP__MODE.x = 0) or an address line pin (EP__MODE.x = 1).

4.3.2 EP__DIR REGISTER

The **EP__DIR register** indicates whether the pin will be an input or open-drain output (EP__DIR.x = 1) or a complementary output (EP__DIR.x = 0). This register is written as a byte.

4.3.3 EP__REG REGISTER

The **EP__REG register** can be used in two different ways. If the pin is configured as an I/O pin, the EP__REG is written with the data to be placed on the pin. If the pin is configured as an address line, the EP__REG will supply the extended address on the EPORT when a 16-bit instruction is executed in 24-bit mode. It should be noted that, although the EP__REG is an 8-bit register, only the lower 4-bits can place data onto the pins while all 8-bits can place an extended address onto the internal extended address bus. This register is written as a byte. If EP__MODE has configured the pins as standard I/O, the value written to the EP__REG will appear immediately on the pins.

For example see code 4-1, if the contents of the EP__REG = 0FFh and a 16-bit instruction is used, the 16-bit address is concatenated to the contents of the EP__REG to form the 24-bit address.

But extended address instructions will drive the extended address specified in the instruction and the EPORT pins will hold these values throughout the bus cycle.

```
Ax   EQU  lCh
STB  Ax,lFFEh ;if EP_REG = OFFh (reset value of register) and device in 24-bit mode
              ;   OFFlFFEh <= contents of lCh
```

Code 4-1. Writing to the EPORT Register with 16-Bit Addressing

4.3.4 EP__PIN REGISTER

The **EP__PIN register** contains the data that is currently on the port pins. This register is read as a byte.

When the EPORT SFRS are to be used as address lines, they should be initialized in the following order:

1. Write to the EP__REG
   — since output address line, write the expected value.
2. Write to the EP__DIR
   — 0 for complementary output (without external pull-up)
   — 1 for open-drain output.
3. Write to the EP__MODE (write FFh since EPORT used for address lines).

4.3.5 RESET VALUES OF EPORT SFRs

The **EPORT registers reset** to the following values: EP__MODE = 0FFh, EP__DIR = 0FFh, EP__REG = 0FFh, EP__PIN = xxh, and the EPORT pins are tri-stated after reset and are weakly pulled high during reset. Also, during the CCB fetch, the EPORT is forced to 0FFh since the CCBs are located in 0FF20xx.

4.3.6 ACCESSING EPORT SFRs

Changing the **values of the EPORT SFRs** is not possible using the windowing feature of the MCS-96 devices. The EPORT, Ports 3, 4, and 5, and the Slave Port SFRs, which are located in the 1FE0h–1FFFh range, are only accessed using 24-bit instructions. All the other SFR registers, which are located in the 1F00h–1FDF range, can be addressed using 8-bit addressing and the windowing feature.

4.4 EPORT

A **block diagram of the EPORT** is shown in Figure 4-1. The extended address either accesses data using the contents of the extended data address register (EDAR) or executes code using the address in the extended slave program counter (ESPC). If the EP__MODE and EP__DIR registers are configured for the addressing function and code is to be executed or data retrieved from an extended address, this extended address will be placed onto the EPORT pins. If a 16-bit instruction is executed and the device is in 24-bit mode, the contents of the EP__REG will determine the highest byte of the extended address. Therefore, the contents of the EP__REG should contain the correct extended address for the registers in the 16-bit instruction. If the device is in 16-bit mode and the port is configured for the addressing function, the EPORT pins will default to 0Fh. If the EP__MODE and the EP__DIR registers are configured for the I/O function, the contents of the EP__REG register is placed on the EPORT pins and the contents of the EP__PIN register is updated to the current value on the pins.

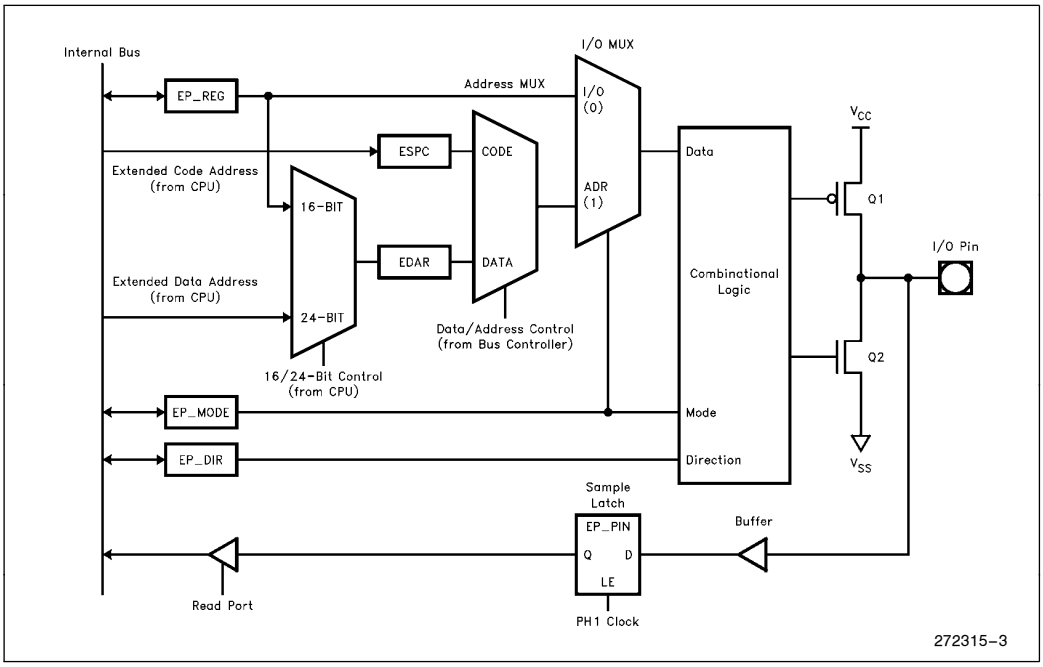


Figure 4-1. EPORT Block Diagram

5.0 MEMORY MAP

5.1 Memory Layout

The memory addressing capability of the 8XC196NT is 1 Mbyte of linear address space. Two bits in the chip configuration bytes configure the memory map four different ways. It is conceptually convenient to think of the address space as sixteen 64-Kbyte pages of address space (see Figure 5-5). The lower 8 Kbyte (in page 00h) is specific purpose memory. The specific purpose memory contains the CPU SFRs, the register file, 512 bytes of internal RAM and the peripheral SFRs. The upper 64 Kbyte (page FFh) is where the internal (EP)ROM is located. After RESET, the CCBs are fetched from FF2018h, FF201Ah and FF201CH. Then the program counter is set to FF2080h where execution begins. Like the previous MCS-96 devices, if $\overline{EA}$ is tied low accesses to locations FF2000h–FF9FFFh go external and access to the internal (EP)ROM is not available. The 512 bytes of internal RAM are mapped to both pages FFh and 00h in locations FF0400h–FF05FFh (and 000400h–0005FFh). In all memory configurations the CPU SFRs (0000h–0017h), peripheral SFRs (1F00h–1FDFh) and Register RAM (0018h–03FFh) are mapped to all pages. Hence, 16-bit loads/stores can be used to access them from any page. When referencing these locations (except page 00h) with extended loads/stores memory access goes external. Note however that the "memory mapped" peripheral SFRs (1FE0h–1FFFh) are only mapped in page 00h (this includes P3, P4, P5, EPORT and the Slave Port; see Figure 5-5, General Memory Map of the 8XC196NT).

5.2 How the EPORT Affects Memory Accesses

The standard 16-bit address/data bus of the MCS-96 family is extended by the extended addressing port (EPORT). The EPORT is 8 bits wide, however only four bits are bonded out to make the extra four address lines. The EPORT can be used for I/O as discussed in the previous EPORT section of this application note.

Understanding how the EPORT is loaded is important. The EPORT block diagram is shown in Figure 4-1. First, the extended data address register (EDAR) is 8 bits wide and concatenated with the 16-bit data address register (DAR) to make a 24-bit data address register. Notice in the EPORT block diagram that the EDAR is loaded from either the EP__REG or the CPU data address bus. When using 16-bit instructions (LD, ST), EP__REG is the source for the EDAR. When using the extended addressing instructions (ELD, EST) the CPU data address bus is the source. When using the 16-bit addressing instructions, the 1 Mbyte of memory can be viewed as sixteen 64 Kbyte memory sections. When it is desired to use the 16-bit addressing instructions on a particular page, use the extended store instruction to initialize EP__REG to that page (see Code 5-1).

The code in Code 5-1 changes the EPORT to 23h. So, when 16-bit instructions are used such as LD, ST, CALL, and JMPs they all refer to the 64 Kbytes of memory in page 23h.

Now suppose that instead of the previous code example, the code in Code 5-2 had been used to change the EPORT.

```
;************* Initializing the EPORT *************

EP_REG    equ   lfe5h

          ldb page,#23h     ;initialize so 16-bit instructions will access page 23h
          estb page,EP_REG  ;use extended store to write to SFR 001fe5h
```

Code 5-1. Initializing the EPORT

```
EP_REG    equ   lfe5h

    ldb page,#23h     ;initialize so 16-bit instructions will access page 23h
    stb page EP_REG   ;use 16-bit store to write to SFR xxlfe5h
```

Code 5-2. Erroneously Initializing the EPORT

If the code in Code 5-2 is executed, where will the register "page" be stored? Refer to the 8XC196NT general memory map (Figure 5-5) and look at locations 001FE0h–001FFFh. These SFRs are ONLY mapped to page 00h. So in the code (Code 5-2) it might not be known what value the EP__REG contained before the "stb" is executed. If the EP__REG contained a "01h", the "page" register would have been stored in location "011FE5h" which is external memory. Therefore, when accessing a peripheral SFR in locations 001FE0h–001FFFh, use the extended store instruction.

It is important to note that the 8XC196NT has 20 address lines. The EPORT register is an 8-bit register. Future devices may bond out the upper four bits of the EPORT register to make a total of 24 address lines. Since internally the (EP)ROM is located in page FFh, you must assemble your (EP)ROM code starting at location FF2080h (where execution begins after reset).

It is a good rule of thumb to set the EP__REG (when the EPORT is configured for address) **to 00h and leave it alone. Hence, all 16-bit loads and stores thereafter will occur in page 00h.** Changing the page via EP__REG should be done with attentiveness.

5.2.1 THE iC COMPILER

The iC compiler assumes that the current page is 00h. So, to remain compatible with the iC compiler, it is recommended to keep the page (via EP__REG) set to 00h.

5.2.2. WARNING ABOUT CHANGING PAGES WITH EP__REG WHEN THE STACK IS IN EXTERNAL MEMORY

If it is desired to use non-extended instructions such as LD, ST, etc. in any page, then the EP__REG can be loaded with the appropriate page value and non-extended instructions will operate within that page. This also applies to stack operations. For example, lets say you set the stack pointer to 0800H in page 00H (external memory). Then you change to page 01H by loading EP__REG with 01H. Now, any stack operation such as PUSH, POP, CALL, ECALL will use the stack pointer in page 01H. So the new stack location will be 010800H. Therefore, if you enter a subroutine while in page 00H, then once in the subroutine you change the EP__REG to any other value and then return from the subroutine, you will return to some unknown location. This is because the stack pointer is now operating in a page other than 00H. Remember the following three rules when coding and when EPORT is configured as address.

1. You must pay careful attention when changing EP__REG when the stack is located in external memory.
2. Before doing any stack operation, make sure EP__REG is loaded with the same page as the stack is in.
3. If the stack is in register RAM then there is no problem since register RAM is mapped to all pages.

5.3 The Different Memory Maps

There are many possible memory map configurations with the 8XC196NT (see Figures 5-6 through 5-9). Two bits in chip configuration byte 2 (CCB2) control the memory map configurations and the number of EPORT lines decoded. These bits are named MODE16 (CCB2.1) and REMAP (CCB2.2). First, these two configuration bits are discussed, then the memory map configurations are explained.

5.3.1 MODE 16

When MODE16 is set to a 1 the extended slave program counter (ESPC) is forced to FFh (see the EPORT block diagram Figure 4-1). The extended slave program counter is also 8 bits wide and is concatenated with the 16-bit program counter (PC) to create a 24-bit wide program counter. The extended program counter can be thought of as holding the page location value of the currently executing code. Hence, code fetches are limited to the 64 Kbyte region in page FFh when MODE16 is set to a 1. **In this mode extended branching instructions (EBR, EJMP, ECALL) do not work and must not be used.**

If MODE16 is set to a 0, then 24-bit mode is entered (see Figures 5-8 and 5-9). Now the extended program counter can be any page value. A simple extended jump instruction across pages changes the extended program counter value to the destination page. For example, if code is executing out of internal (EP)ROM in page FFh, the code can branch to external memory location 003000h by the following instruction (see Code 5-3):

```
FF2090:     ld temp,#12h      ;some code executing in page FFh
            st temp,portl
            ebr     003000h   ;24 bit instruction jump to external location
                              ;003000h (page 00h)
                  .
                  .
                  .
003000:     add temp,#50h
                  .
                  .
                  .
```

Code 5-3. Using the Extended Branching Instruction

5.3.2 REMAP

The REMAP bit, when set to a 1, maps the internal (EP)ROM to both pages 00h and FFh. See Figures 5-6 and 5-7 for the memory maps with REMAP = 1. The disadvantage of having the (EP)ROM mapped in both pages 00h and FFh is that a 32 Kbyte piece of page 00h is used. However, this may be useful if data tables are stored in (EP)ROM. When the REMAP bit is set to a 0, the internal (EP)ROM is only mapped to page FFh. But, if executing externally ($\overline{EA}$ low) the REMAP bit is a don't care. Hence, the REMAP bit is ONLY effective when $\overline{EA}$ is high.

5.3.3 64K COMPATIBLE MODE

Now that the two CCB2 bits MODE16 and REMAP have been defined, we can discuss the different memory configurations mentioned earlier. The first configuration is called 64K compatible mode and its memory map is pictured in Figure 5-6. This mode makes the 8XC196NT completely compatible with the 8XC196KR (EP__REG should be loaded with 00h if complete compatibility is desired). In this mode, MODE16 is set to a 1 so the program counter is limited to a 64K address space and the REMAP bit is set to a 1 so the (EP)ROM is mapped in both page FFh and page 00h. A program written for a 64K MCS-96 device such as the 8XC196KR can be ported over to the 8XC196NT easily. The lower 64 Kbytes of memory looks similar to the 64K MCS-96 devices. If desired, the EPORT lines can be configured as address allowing access to 1 Mbyte of data. Setting the EP__REG to a page value allows use of the 16-bit addressing instructions (i.e. LD, ST) in any page 01h through 0Eh.

5.3.4 EXAMPLE: USING THE "64K COMPATIBLE MODE" WITH 96 KBYTES OF DATA

Figure 5-1 shows the 87C196NT running from internal (EP)ROM in 64K compatible mode. External data accesses are allowed using extended addressing instructions. The first 32 Kbyte RAM is mapped to the locations in page 00h that are not mapped for a specific purpose (see general memory map Figure 5-5). The general purpose spaces in page 00h amount to 32 Kbytes of memory. The 64 Kbyte RAM is mapped to page 01h. The 16-bit instructions (LD, ST) access RAM #1 (if the EP__REG was set to 00h) and the extended instructions access both RAMs. The 16-bit instructions can access RAM #2 if the EP__REG is loaded with 01h.

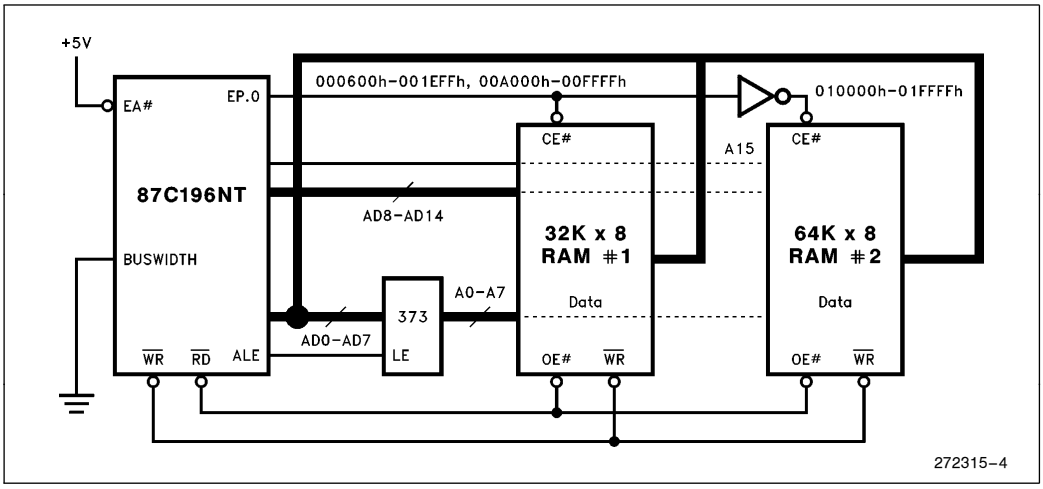


Figure 5-1. 64K Compatible Mode with 96 Kbytes of External Data Space

The code below applies to Figure 5-1 above. There are two subroutines (A and B). In subroutine A (see Code 5-4) page 01h is accessed using 16-bit stores. However, first the EP__REG has to be initialized to page 01h. In addition the subroutine has to restore the page value to EP__REG upon exiting the subroutine. So there is some overhead involved with using 16-bit loads/stores in different pages. Although if many data accesses are done in subroutine A then 16-bit loads/stores would be advantageous since they are faster. The extended loads/stores require an extra bus cycle since the internal bus is 16-bits wide.

In subroutine B (see Code 5-5) the extended instructions are used. Notice the overhead from subroutine A has been eliminated. Subroutine B takes less memory space and is more "readable" since fewer instructions are needed. If time is not critical and subroutine B is not called often to do data accesses then it would be justifiable to use extended instructions. Only in time critical applications where many loads and stores are done would it be advantageous to use 16-bit non-extended instructions.

```
                ;* Accessing data in page 01h using 16-bit instructions
EP_REG          equ 1FE5h
                rseg at 1ch
     temp:      dsw 1
     page:      dsb 1
     current_page: dsb 1
     result:    dsw 1

                cseg at 0FF2080h
                .                           ;Some code
                .
                .
SubA:           pusha                       ;save flags, disable interrupts
                eldb current_page,EP_REG    ;save the current setting of the
                                            ;page
                ldb page,#01h               ;all 16-bit accesses go to page 01h
                est page,EP_REG             ;initialize EP_REG for data access
                                            ;in page 01h
                ld temp,#1234h
                st temp,600h                ;value in "temp" is stored in
                                            ;location 010600h
                add result,temp,#4000h      ;do something with registers
                st result,602h              ;store result in 010602h
                                            ;more ld/st instructions in page
                                            ;01h
                .
                .
                .
                estb current_page,EP_REG    ;restore page value for calling
                                            ;program
                popa                        ;restore flags and interrupts
                ret
                .                           ;The rest of the code
                .
                .
done:           br done
                end
```

Code 5-4: Accessing Page 01h in "64K Compatible Mode" Using 16-Bit instructions

```
                ;* Accessing data in page 01h using extended instructions
EP_REG          equ 1FE5h
                rseg at 1ch
     temp:      dsw 1
     result:    dsw 1
                cseg at 0FF2080h
                .                           ;Somecode
                .
                .
SubB:           pusha                       ;save flags, disable interrupts
                ld temp,#1234h
                est temp,010600h            ;value in "temp" is stored in
                                            ;location 010600h
                add result,temp,#4000h      ;do something with registers
                est result, 010602h         ;store result in 010602h
                .                           ;more eld/est instructions
                .
                .
                popa                        ;restore flags and interrupts
                ret
                .                           ;The rest of the code
                .
                .
done:           brdone
                end
```

Code 5-5: Accessing Page 01h in "64K Compatible Mode" Using Extended Instructions

5.3.5 64K COMPATIBLE MODE WITH PAGE 00h FREE

The next memory map is pictured in Figure 5-7. This is the same as the previous mode except that the REMAP bit is set to 0. The program counter is still forced to page FFh since MODE16 = 1. So any jumps beyond the page will stay within page FFh. Page 00h is now available with the exception of some of the lower memory (see the general memory map Figure 5-5). This mode can be used for a program that was written for a 64 Kbyte MCS-96 device, which needs more memory for data.

5.3.6 EXAMPLE: USING THE 64K COMPATIBLE MODE WITH 128K OF EXTERNAL DATA

Since page 00h is free to use (except for SFRs, etc.), this next memory configuration takes advantage of page 00h in 64K compatible mode (see Figure 5-2). The program counter is limited to 64 Kbyte. But, the (EP)ROM is only mapped in page FFh and the 32 Kbyte memory space in page 00h is now free. This allows use of a 64 Kbyte RAM instead of the 32 Kbyte RAM shown in Figure 5-1. Hence, there is a total of 128 Kbytes of external data available to access when using only one EPORT address line.

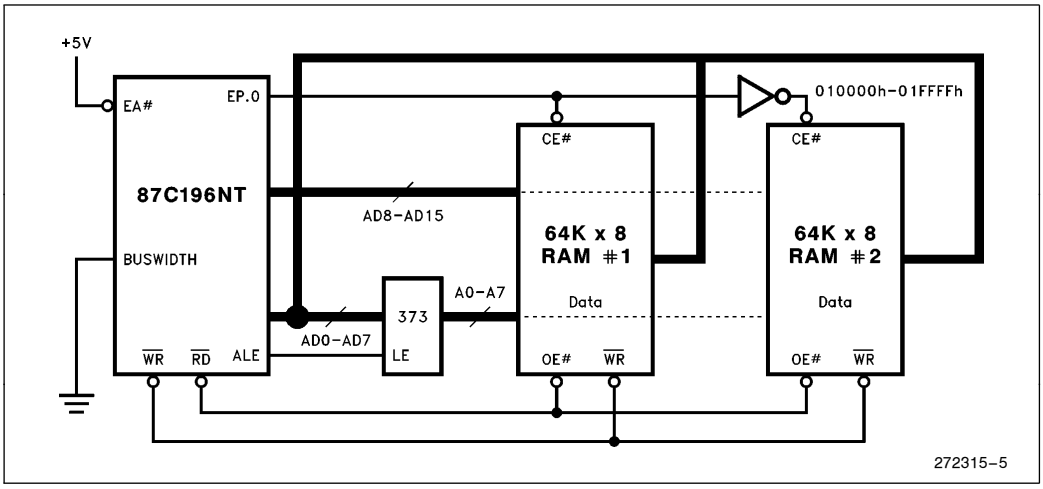


Figure 5-2. 64K Compatible Mode with 128 Kbytes of External Data

5.3.7 24-BIT MODE WITH (EP)ROM REMAPPED

Figure 5-8 shows the third memory map. Here the (EP)ROM is remapped to page 00h as well as page FFh (REMAP = 1) but the program counter is 24 bits wide (MODE16 = 0). Hence, code can be executed from anywhere in the memory space. Remember that 32 Kbytes of memory space in page 00h are dedicated to the internal (EP)ROM which would typically be used to store look-up tables or constants. The ELD/EST instructions work in all modes and always allow access to 1 Mbyte of data.

5.3.8 EXAMPLE: USING 24-BIT MODE WITH (EP)ROM REMAPPED

The third memory configuration is extended addressing with the EPROM remapped (Figure 5-3). In this configuration, MODE 16 = 0 so the program counter is 24 bits wide. This memory configuration could be used when a program was written for the 87C196KR but requires more memory space for code. Notice in Figure 5-3 that there are three pages of external memory used. The RAM (page 01h) is used for external data storage. The external EPROM (pages 02h and 03h) is used for 128 Kbytes of code. So in the example below there are 160 Kbytes of code space (32 Kbytes internal EPROM + 128 Kbytes external EPROM) and 64 Kbytes of data space. Note this time a 16-bit buswidth was used.

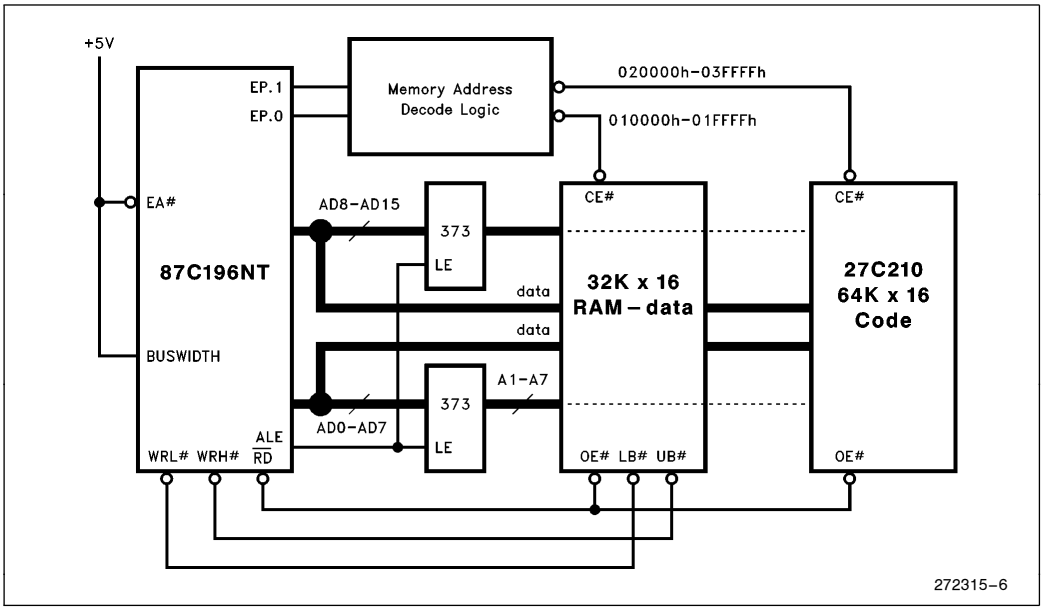


Figure 5-3. Extended Addressing with 64 Kbytes of External EPROM and 32 Kbytes of External RAM (16-Bit Buswidth)

5.3.9 24-BIT MODE

The 24-bit mode is the most liberal of the memory map configurations (see Figure 5-9). The program counter is 24 bits wide and the internal (EP)ROM is only mapped to page FFh. In this mode, code and data can be accessed almost anywhere in memory.

5.3.10 EXAMPLE: USING 24-BIT MODE

Finally, the normal extended addressing mode is shown in Figure 5-4. Here we are in 24-bit mode and the REMAP bit is a don't care since all code accesses go external. Code begins execution in page FFh. The first 64 Kbyte EPROM is used for page FFh. The next 64 Kbyte EPROM is used for page 00h. The 64 Kbyte RAM can be used for data storage; in addition, code can be executed from the RAM. Only two of the EPORT pins need to be decoded. If more memory is needed, then it is easy to decode the additional EPORT pins.

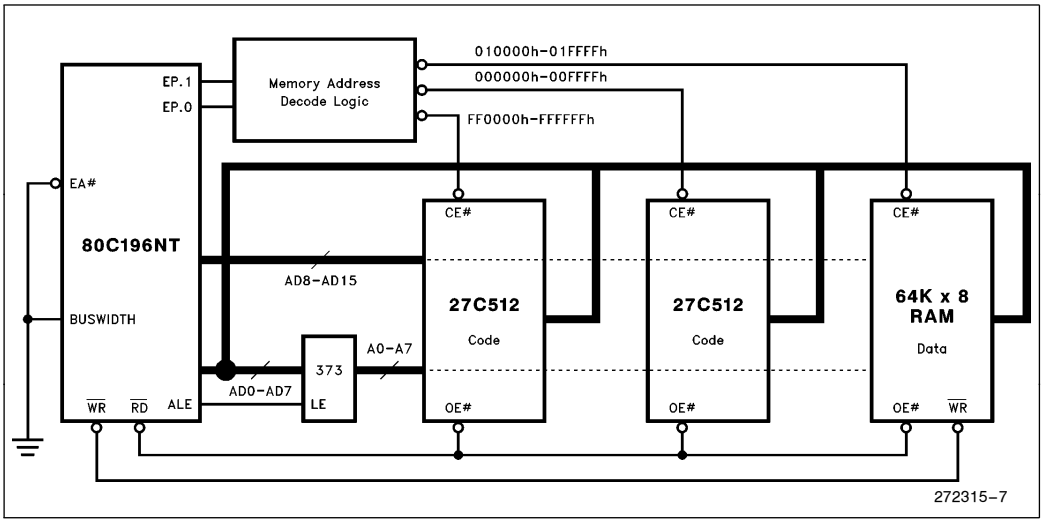


Figure 5-4. Normal Extended Addressing with 128 Kbytes of Code, 64 Kbytes of External Data

5.4 Internal RAM

The 8XC196NT has 512 bytes of internal RAM. This RAM is mapped in both pages FFh and 00h (see Figure 5-5: General Memory Map). The internal RAM is like external RAM that has been brought inside the chip. Hence, the internal RAM can be used to execute code or store data. To access the internal RAM as data, only indexed or indirect addressing can be used. Code can be placed in internal RAM in many ways. For example, if a board executes different code each time it powers up then the boot code can be downloaded via the serial port to the internal RAM.

5.4.1 READING THE EA PIN AND REDIRECTING INTERNAL RAM ACCESSES

It is possible to read the logic level on the $\overline{EA}$ pin via the internal RAM register (IRAM__REG). The format of the IRAM__REG (1FE0h) is shown below:

7	6	5	4	3	2	1	0
EA__STAT	IRAM	0	0	0	0	0	0

EA__STAT: This bit is the complement of the logic level on the $\overline{EA}$ pin.

= 0 means $\overline{EA}$ = 1 and code accesses in the range FF2000h–FF9FFFh are directed to internal EPROM or ROM.

= 1 means $\overline{EA}$ = 0 and code accesses in the range FF2000h–FF9FFFh are directed to external memory.

IRAM: direct the internal RAM accesses to external memory

= 0 means all internal RAM accesses from 400h–5FFh go INTERNAL

= 1 means all internal RAM accesses from 400h–5FFh go EXTERNAL

Bits 5–0: reserved and should be set to 0

The IRAM__REG is useful when in the design stage and the internal RAM code must be interrogated with a logic analyzer. This can't be accomplished unless all internal RAM accesses are directed to external memory where a logic analyzer can be hooked up. The following initialization code (Code 5-6) can be placed at the beginning of your program to check for internal execution or external execution and configure internal RAM accesses accordingly:

```
IRAM_REG equ 1FE0h

rseg at 1Ch
temp: dsb 1

cseg at 0FF2080h
eldb temp,IRAM_REG[0]          ;put complement of EA# pin in bit 7 of temp
shrb temp,#1                   ;put complement of EA# in bit 6 of temp

estb temp,IRAM_REG[0]          ;Allow internal RAM accesses to go either
                               ;INTERNAL(IRAM_REG.6=0) or EXTERNAL
                               ;(IRAMREG.6=1)
```

Code 5-6: Changing Access of Internal RAM with the IRAM__REG

5.5 Wraparound

Given the memory configuration of Figure 5-2 suppose the following instruction is executed (see code 5-7):

```
eld temp, 023000h[0]
```

Code 5-7: An Example of Wraparound

Since EP.1 is not decoded the 02h part of the address is ignored and "temp" gets loaded with the value at location 003000h. Wraparound occurs when referencing a memory location that requires greater than 20 address lines (if all four EPORT lines are used) since internally, extended addresses are 24 bits wide. There is no negative effect of wraparound, just keep it in mind when coding.

FFFFFF FFA000	External Memory
FF9FFF FF2000	Internal (EP)ROM or External Memory
FF1FFF FF0600	External Memory
FF05FF FF0400	Internal RAM (mapped to 000400 to 0005FF also)
FF03FF FF0100	External Memory
FF00FF FF0000	Reserved for ICE
FEFFFF 010000	896K of External Memory
00FFFF 00A000	External Memory
009FFF 002000	External Memory or Internal (EP)ROM (depends on remap bit)
001FFF 001FE0	Peripheral Special Function Registers
001FDF 001F00	Peripheral Special Function Registers*
001EFF 000600	External Memory
0005FF 000400	Internal RAM (mapped to FF0400 to FF05FF also)
0003FF 000018	Register RAM*
000017 000000	CPU Special Function Registers*

NOTES:
Code accesses to locations 00000h to 003FFFh go external.
*Accessible in every page using 16-bit addressing (LD, ST). External memory in every page (except 00h) when using extended addressing instructions.

Figure 5-5. General Memory Map of the 8XC196NT

FFFFFF FF0000	• External code and data • 32K Internal (EP)ROM • 512 Bytes of internal RAM (mapped from page 00h)
FEFFFF 010000	• External data ONLY
00FFFF 000000	• External data • Remapped internal (EP)ROM • 512 Bytes of internal RAM • SFRs • Register File

Figure 5-6. Memory Map: 64K Compatible Mode

MODE 16 = 1 program counter limited to 64K; no limit to data access
REMAP = 1 internal (EP)ROM mapped in both pages 00h and FFh

FFFFFF FF0000	• External code and data • 32K Internal (EP)ROM • 512 Bytes of internal RAM (mapped from page 00h)
FEFFFF 010000	• External data ONLY
00FFFF 000000	• External data • 512 Bytes of internal RAM • SFRs • Register File

Figure 5-7. Memory Map: 64K Compatible Mode with Page 00h Available

MODE 16 = 1 program counter limited to 64K; no limit to data access
REMAP = 0 internal (EP)ROM mapped in page FFh ONLY

NOTES:
When $\overline{EA}$ is high accesses from FF2000h to FF9FFFh go INTERNAL
When $\overline{EA}$ is low accesses from FF2000h to FF9FFFh go EXTERNAL

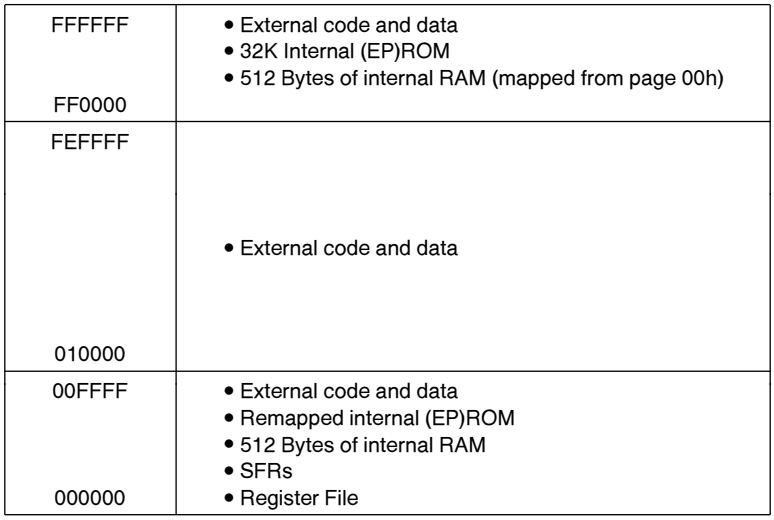


Figure 5-8. Memory Map: 24-Bit Mode with (EP)ROM Remapped

MODE 16 = 0 program counter is 24 bits wide; no limit to data access
REMAP = 1 internal (EP)ROM mapped in both pages 00h and FFh

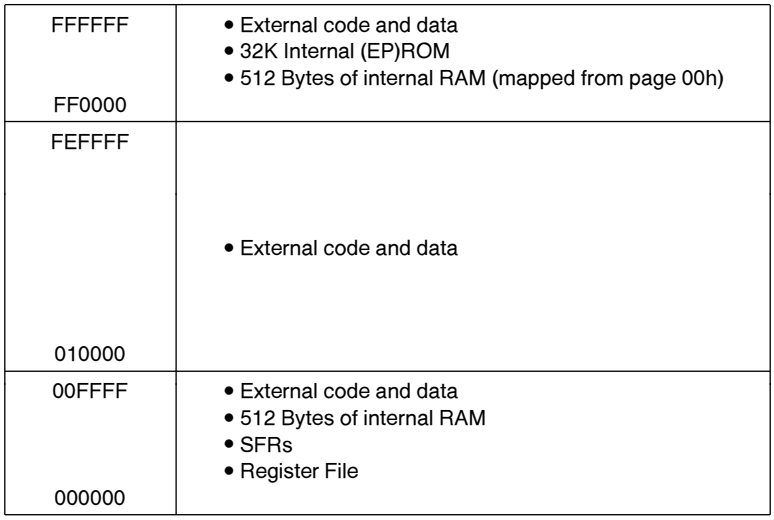


Figure 5-9. Memory Map: 24-Bit Mode

MODE 16 = 0 program counter is 24 bits wide; no limit to data access
REMAP = 0 internal (EP)ROM mapped in page FFh ONLY

NOTES:
When $\overline{EA}$ is high accesses from FF2000h to FF9FFFh go INTERNAL
When $\overline{EA}$ is low accesses from FF2000h to FF9FFFh go EXTERNAL

6.0 TIMING IMPROVEMENTS

The 8XC196NT has an enhanced bus controller. The timing improvements as a result of the enhanced bus controller allow the microcontroller to run with no wait states using slower and less expensive memories on the external bus. This section discusses the improvements to the bus controller that allows this flexibility.

6.1 Signals

First, an explanation of the A.C. timing symbols is necessary. Consult the chart below when studying timing diagrams:

Timing Name: T_{WXYZ}

The signals are **W** and **Y**.

The conditions are **X** and **Z**.

A — address
B — bus control signals ($\overline{BHE}$, INST)
BR — $\overline{BREQ}$ (bus request)
C — CLKOUT
D — data in
G — buswidth
H — $\overline{HOLD}$
HA — $\overline{HLDA}$ (hold acknowledge)
L — ALE/$\overline{ADV}$
Q — data out
R — $\overline{RD}$ (read)
W — $\overline{WR}$/$\overline{WRH}$/$\overline{WRL}$
X — XTAL
Y — READY

6.2 Conditions

H — high
L — low
V — valid
X — no longer valid
Z — floating

6.3 Critical Memory Timings

Before discussing the value of the improved bus controller timings, it is necessary to explain the critical memory device timings. The diagram (see Figure 6-1) below illustrates the important memory device READ timing relationships.

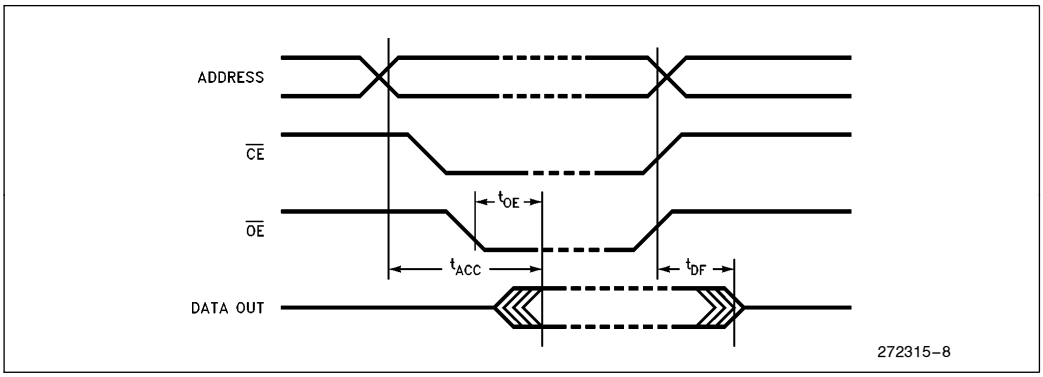


Figure 6-1. Critical Memory Device Timings

t_{ACC} — delay time from when the address is valid to when the data output from the memory device is valid.

t_{OE} — delay time from when the output enable of the memory device is asserted to when the data output from the device is valid.

t_{DF} — delay time from when the output enable is deasserted until when the data output from the memory device floats. The smaller this spec is the better because then the microcontroller can drive the next address on the bus sooner.

t_{WP} — the write pulse width

The corresponding MCS-96 BUS READ signals and timings are pictured in Figure 6-2.

T_{AVDV} — address valid to data input valid

T_{RLDV} — read low to data valid

T_{RHDZ} — read high to data float

T_{WLWH} — the write low pulse width (this is the only critical write timing spec)

T_{OSC} — The period of the oscillator or ½ the period of CLOCKOUT. $T_{OSC} = 1/F_{XTAL1}$

Hence, given the above timings, the following relations can be derived:

1. (t_{ACC} + latch delay) must be < T_{AVDV}
2. t_{OE} must be < T_{RLDV}
3. t_{DF} must be < T_{RHDZ}
4. t_{WP} < T_{WLWH}

6.4 What Can be Done to Use Less Expensive Memories and Run at Maximum Speed with Zero Wait States?

With the standard bus timings it is difficult for inexpensive memory devices to meet the critical timings of the microcontroller. Therefore, either the microcontroller has to run at a slower clock frequency or more expensive, faster memories must be used.

The new timing modes on the 8XC196NT solve the problem of meeting critical memory timings. The 8XC196NT new bus timing modes allow the microcontroller to run at its maximum specified frequency and use standard (slower) memory devices. The method of solving the problem centers around adjusting the microcontroller's critical bus timings to accommodate slower memory devices. Refer to the critical timings diagrams (Figure 6-1 and 6-2) to conceptualize the following explanation: typically the limiting factor with external memories is the t_{OE} spec (corresponding to T_{RLDV}). If T_{RLDV} is increased enough to meet the t_{OE} spec then the next limiting spec is t_{ACC} (corresponding to T_{AVDV}). However, when T_{AVDV} is relaxed (i.e. increased) the T_{RHDZ} becomes worse (i.e. decreases). Consequently t_{DF} now becomes the limiting factor with the memory device. Hence, there is a tradeoff associated with relaxing the T_{AVDV} spec.

6.5 New Timing Modes

There are four different timing modes available on the 8XC196NT. The modes are: standard timing, standard timing with one wait state, long read/write with advanced ALE, long read/write with advanced address. These modes are selectable by the chip configuration bytes. The chip configuration bytes are loaded into the chip configuration registers after RESET goes high. These bytes configure the microcontroller (see Figure 7-1 for the chip configuration byte definitions). Basically there are two bits in CCB1: MSEL0 and MSEL1 (see Table 6-1) that define the timing modes. A pictorial comparison of these modes is shown in Figure 6-3.

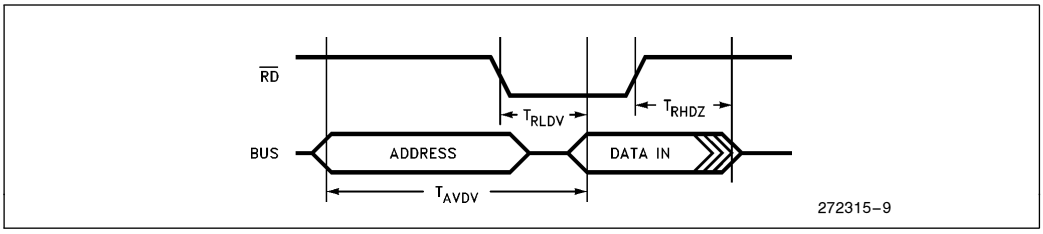


Figure 6-2. Critical 8XC196NT Bus Timings (Read Cycle)

Table 6-1. 8XC196NT Bus Timing Mode Selections

MSEL0 (CCB1.6)	MSEL1 (CCB1.7)	Mode	Description
0	0	0	**Standard Timing with One Wait State** inserted into the bus cycle. Also known as Slow External Memory (SEM) mode.
0	1	1	**Long R/W:** advances $\overline{RD}/\overline{WR}$ by 1 T_{OSC} creating a 2 T_{OSC} $\overline{RD}/\overline{WR}$ low time. ALE is also advanced by 0.5 T_{OSC} but ALE high time remains 1 T_{OSC}.
1	0	2	**Early Address:** Same as mode 1 but the address is put on the bus 0.5 T_{OSC} earlier in the bus cycle.
1	1	3	**Standard Timing Mode**

6.5.1 STANDARD TIMING (MODE 3)

The *standard timing* mode configures the bus controller to operate with timings similiar to the 8XC196KR. This mode is the default bus timing mode after RESET. See Figure 6-3 for a pictorial representation of mode 3 compared to modes 1 and 2.

6.5.1.1 Mode 3 Timing Specs

Memory Device Spec	Corresponding 8XC196NT Spec	Time in ns
t_{ACC}	T_{AVDV}	3 T_{OSC} − 55 (Max)
t_{DF}	T_{RHDZ}	T_{OSC} (Max)
t_{OE}	T_{RLDV}	T_{OSC} − 30 (Max)
t_{WP}	T_{WLWH}	T_{OSC} − 30 (Min)

NOTE:
For latest specs consult the current datasheet for the 8XC196NT.

6.5.2 STANDARD TIMING WITH ONE WAIT STATE (MODE 0)

Mode 0 is the same as the *Standard Timing* mode except that one wait state is inserted into the bus cycle. A wait state is 2 T_{OSC}, which must be added to the relevant "no wait state" spec to get the MODE 0 spec. This mode is also referred to as the *Slow External Memory* (SEM) mode.

6.5.2.1 Mode 0 Timing Specs

Memory Device Spec	Corresponding 8XC196NT Spec	Time in ns
t_{ACC}	T_{AVDV}	5 T_{OSC} − 55 (Max)
t_{DF}	T_{RHDZ}	T_{OSC} (Max)
t_{OE}	T_{RLDV}	3 T_{OSC} − 30 (Max)
t_{WP}	T_{WLWH}	3 T_{OSC} − 30 (Min)

NOTE:
For latest specs consult the current datasheet for the 8XC196NT.

6.5.3 LONG READ/WRITE WITH ADVANCED ALE (MODE 1)

Mode 1 improves the t_{OE} spec. Since t_{OE} corresponds to T_{RLDV}, it follows that lengthening the READ (or WRITE) pulse also lengthens T_{RLDV}. Consequently, the memory device has more time to place the data onto the bus. In other words, the t_{OE} spec of the memory device can now be larger which corresponds to a less expensive device. Hence, MODE 1 is called the Long READ/WRITE mode since the $\overline{RD}/\overline{WR}$ signals are advanced by one T_{OSC} (see Figure 6-3, comparison of the bus timing modes). The time the address is driven on the bus is shortened by one T_{OSC} from two T_{OSC}s (see Figure 6-3). ALE is advanced by one-half a T_{OSC}, but still has a one T_{OSC} high time. Advancing ALE guarantees that the address will be valid when ALE goes low (shown in Figure 6-3).

6.5.3.1 Mode 1 Timing Specs

Memory Device Spec	Corresponding 8XC196NT Spec	Time in ns
t_{ACC}	T_{AVDV}	3 T_{OSC} − 60 (Max)
t_{DF}	T_{RHDZ}	T_{OSC} (Max)
t_{OE}	T_{RLDV}	2 T_{OSC} − 44 (Max)
t_{WP}	T_{WLWH}	2 T_{OSC} − 20 (Min)

NOTE:
For latest specs consult the current datasheet for the 8XC196NT.

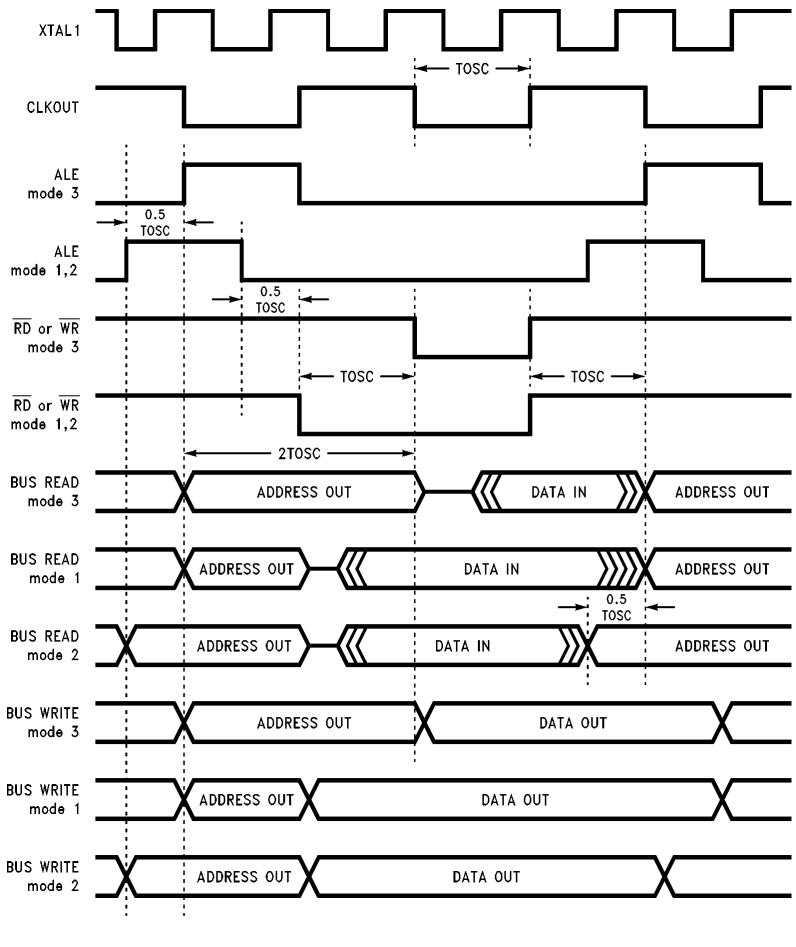
6.5.4 LONG READ/WRITE WITH ADVANCED ALE AND EARLY ADDRESS (MODE 2)

This mode improves the t_{ACC} spec by lengthening T_{AVDV}. As shown in Figure 6-3, the address is placed on the bus one-half T_{OSC} earlier than the other modes. Therefore, the address is driven for one and a half T_{OSC}s. Since the address is placed on the bus one-half T_{OSC} earlier, the previous data has to be taken off the bus faster. Hence, the T_{RHDZ} spec is shortened (i.e., worse than other modes). In Figure 6-3, the mode 2 A/D bus drawing shows the data output being floated earlier due to the early address. The tightening of T_{RHDZ} (t_{DF}) is a necessary compromise that results from relaxing T_{AVDV}.

6.5.4.1 Mode 2 Timing Specs

Memory Device Spec	Corresponding 8XC196NT Spec	Time in ns
t_{ACC}	T_{AVDV}	3.5 T_{OSC} − 55 (Max)
t_{DF}	T_{RHDZ}	0.5 T_{OSC} (Max)
t_{OE}	T_{RLDV}	2 T_{OSC} − 44 (Max)
t_{WP}	T_{WLWH}	2 T_{OSC} − 20 (Min)

NOTE:
For latest specs consult the current datasheet for the 8XC196NT.



NOTE:
Mode 0 is the same as Mode 3 but with one wait state inserted into the bus cycle.

Figure 6-3. Comparison of 8XC196NT Bus Timing Modes

6.6 Given These New Timing Modes, Which Memory Devices Can Be Used with No Wait States?

The timing improvements discussed above allow the memory devices to be used at higher frequencies with no wait states. Included in this application note are four spreadsheets that show the maximum frequency the 8XC196NT can be run at with the given memory device in a given mode. In different modes, the maximum frequency is contingent upon one of the three critical timing specs (i.e., t_{ACC}, t_{OE} or t_{DF}). The spreadsheets show the calculations for each critical timing. For example, it calculates the maximum frequency assuming that t_{ACC} is the limiting spec. Then the spreadsheets calculate the maximum frequency assuming t_{OE} is the limiting spec and so forth. After the maximum frequency is calculated the corresponding 8XC196NT bus timings are calculated based on the maximum frequency just found. For example, in spreadsheet 2, Mode 1 calculations are shown. Three different groups of calculations are done based on either t_{ACC}, t_{OE} and t_{DF}. Notice that the shaded maximum frequency column is under t_{ACC}. That means that t_{ACC} was the limiting spec. Also, the timings t_{ACC}, t_{OE} and t_{DF} are within the limits of the calculated T_{AVDV}, T_{RLDV} and T_{RHDZ} respectively. Notice in the other two columns (t_{OE} and t_{DF}), that the specs t_{ACC}, t_{OE} and t_{DF} do not all meet T_{AVDV}, T_{RLDV} or T_{RHDZ}. The flash memory parts have the same analysis except just t_{OE} and t_{WP} are considered. Also, mode 0, the standard timing mode with one wait state is included for completeness. The last spreadsheet (#5) is a summary of the previous four.

6.7 Changes to the Bus Control Timings

The main control signals affected by the enhanced bus controller are ALE, $\overline{RD}$, and $\overline{WR}$. But, some other control signals changed as well. For example, in modes 1 and 2, $\overline{ADV}$, BHE and INST occur one-half a T_{OSC} earlier to remain consistent with ALE and $\overline{RD}/\overline{WR}$. Also, the $\overline{WRL}/\overline{WRH}$ lengthened the same as $\overline{WR}$.

6.8 8-Bit Bus in Modes 1 and 2

It is required to latch address lines 0 through 15 when running in 8-bit mode when operating in modes 1 (long read/write) and 2 (early address). The reason for the latch is because the upper address lines (A8–A15) are not driven with the address during the write or read portion of the bus cycle. Instead A8–A15 are driven with the data from AD0–AD7 during the write or read cycle.

A latch is not required on the upper address lines A8–A15, when operating in Modes 0 and 3. In these modes the address is driven on A8–A15 during the entire bus cycle. Only a latch on the lower address lines (AD0–AD7) is required.

				Mode 0											
				Based on t_{ACC}				Based on t_{OE}				Based on t_{DF}			
EPROMs	t_{ACC}	t_{OE}	t_{DF}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}
27C256-200	200	75	55	18.66	200.00	130.80	53.60	28.57	107.00	75.00	35.00	18.18	207.00	135.00	55.00
27C256-150	150	60	50	22.94	150.00	100.80	43.60	33.33	82.00	60.00	30.00	20.00	182.00	120.00	50.00
27C256-120	120	55	30	26.60	120.00	82.80	37.60	35.29	73.67	55.00	28.33	33.33	82.00	60.00	30.00
27C512-200	200	70	60	18.66	200.00	130.80	53.60	30.00	98.67	70.00	33.33	16.67	232.00	150.00	60.00
27C512-150	150	60	50	22.94	150.00	100.80	43.60	33.33	82.00	60.00	30.00	20.00	182.00	120.00	50.00
27C512-120	120	55	30	26.60	120.00	82.80	37.60	35.29	73.67	55.00	28.33	33.33	82.00	60.00	30.00
27C010-200	200	70	60	18.66	200.00	130.80	53.60	30.00	98.67	70.00	33.33	16.67	232.00	150.00	60.00
27C010-150	150	60	50	22.94	150.00	100.80	43.60	33.33	82.00	60.00	30.00	20.00	182.00	120.00	50.00
27C010-120	120	55	30	26.60	120.00	82.80	37.60	35.29	73.67	55.00	28.33	33.33	82.00	60.00	30.00
27C020-200	200	70	60	18.66	200.00	130.80	53.60	30.00	98.67	70.00	33.33	16.67	232.00	150.00	60.00
27C020-150	150	60	50	22.94	150.00	100.80	43.60	33.33	82.00	60.00	30.00	20.00	182.00	120.00	50.00
27C040-200	200	70	60	18.66	200.00	130.80	53.60	30.00	98.67	70.00	33.33	16.67	232.00	150.00	60.00
27C040-150	150	60	50	22.94	150.00	100.80	43.60	33.33	82.00	60.00	30.00	20.00	182.00	120.00	50.00

Flash	t_{ACC}	t_{OE}	t_{DF}	t_{WP}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	T_{WLWH}
28F256A-150	150	55	35	60	22.94	150.00	100.80	43.60	100.8
28F256A-120	120	50	30	60	26.60	120.00	82.80	37.60	82.8
28F512-150	150	55	35	60	22.94	150.00	100.80	43.60	100.8
28F512-120	120	50	30	60	26.60	120.00	82.80	37.60	82.8
28F010-150	150	55	35	60	22.94	150.00	100.80	43.60	100.8
28F010-120	120	50	30	60	26.60	120.00	82.80	37.60	82.8
28F020-200	200	60	40	60	18.66	200.00	130.80	53.60	130.8
28F020-150	150	55	35	60	22.94	150.00	100.80	43.60	100.8
28F001BX-150	150	55	35	50	22.94	150.00	100.80	43.60	100.8
28F001BX-120	120	50	30	50	26.60	120.00	82.80	37.60	82.8

Spreadsheet 1. Mode 0 Matched Memory Devices

NOTE:
A latch delay of 13 ns is included in the calculations.

				Mode 1											
				Based on t_{ACC}				Based on t_{OE}				Based on t_{DF}			
EPROMs	t_{ACC}	t_{OE}	t_{DF}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}
27C256-200	200	75	55	10.99	200.00	138.00	91.00	16.81	105.50	75.00	59.50	18.18	92.00	66.00	55.00
27C256-150	150	60	50	13.45	150.00	104.67	74.33	19.23	83.00	60.00	52.00	20.00	77.00	56.00	50.00
27C256-120	120	55	30	15.54	120.00	84.67	64.33	20.20	75.50	55.00	49.50	33.33	17.00	16.00	30.00
27C512-200	200	70	60	10.99	200.00	138.00	91.00	17.54	98.00	70.00	57.00	16.67	107.00	76.00	60.00
27C512-150	150	60	50	13.45	150.00	104.67	74.33	19.23	83.00	60.00	52.00	20.00	77.00	56.00	50.00
27C512-120	120	55	30	15.54	120.00	84.67	64.33	20.20	75.50	55.00	49.50	33.33	17.00	16.00	30.00
27C010-200	200	70	60	10.99	200.00	138.00	91.00	17.54	98.00	70.00	57.00	16.67	107.00	76.00	60.00
27C010-150	150	60	50	13.45	150.00	104.67	74.33	19.23	83.00	60.00	52.00	20.00	77.00	56.00	50.00
27C010-120	120	55	30	15.54	120.00	84.67	64.33	20.20	75.50	55.00	49.50	33.33	17.00	16.00	30.00
27C020-200	200	70	60	10.99	200.00	138.00	91.00	17.54	98.00	70.00	57.00	16.67	107.00	76.00	60.00
27C020-150	150	60	50	13.45	150.00	104.67	74.33	19.23	83.00	60.00	52.00	20.00	77.00	56.00	50.00
27C040-200	200	70	60	10.99	200.00	138.00	91.00	17.54	98.00	70.00	57.00	16.67	107.00	76.00	60.00
27C040-150	150	60	50	13.45	150.00	104.67	74.33	19.23	83.00	60.00	52.00	20.00	77.00	56.00	50.00

Flash	t_{ACC}	t_{OE}	t_{DF}	t_{WP}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	T_{WLWH}
28F256A-150	150	55	35	60	13.45	150.00	104.67	74.33	128.67
28F256A-120	120	50	30	60	15.54	120.00	84.67	64.33	108.67
28F512-150	150	55	35	60	13.45	150.00	104.67	74.33	128.67
28F512-120	120	50	30	60	15.54	120.00	84.67	64.33	108.67
28F010-150	150	55	35	60	13.45	150.00	104.67	74.33	128.67
28F010-120	120	50	30	60	15.54	120.00	84.67	64.33	108.67
28F020-200	200	60	40	60	10.99	200.00	138.00	91.00	162.00
28F020-150	150	55	35	60	13.45	150.00	104.67	74.33	128.67
28F001BX-150	150	55	35	50	13.45	150.00	104.67	74.33	128.67
28F001BX-120	120	50	30	50	15.54	120.00	84.67	64.33	108.67

Spreadsheet 2. Mode 1 Matched Memory Devices

NOTE:
A latch delay of 13 ns is included in the calculations.

				Mode 2											
				Based on t_{ACC}				Based on t_{OE}				Based on t_{DF}			
EPROMs	t_{ACC}	t_{OE}	t_{DF}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}
27C256-200	200	75	55	13.06	200.00	109.14	38.29	16.81	140.25	75.00	29.75	9.09	317.00	176.00	55.00
27C256-150	150	60	50	16.06	150.00	80.57	31.14	19.23	114..00	60.00	26.00	10.00	282.00	156.00	50.00
27C256-120	120	55	30	18.62	120.00	63.43	26.86	20.20	105.25	55.00	24.75	16.67	142.00	76.00	30.00
27C512-200	200	70	60	13.06	200.00	109.14	38.29	17.54	131.50	70.00	28.50	8.33	352.00	196.00	60.00
27C512-150	150	60	50	16.06	150.00	80.57	31.14	19.23	114.00	60.00	26.00	10.00	282.00	156.00	50.00
27C512-120	120	55	30	18.62	120.00	63.43	26.86	20.20	105.25	55.00	24.75	16.67	142.00	76.00	30.00
27C010-200	200	70	60	13.06	200.00	109.14	38.29	17.54	131.50	70.00	28.50	8.33	352.00	196.00	60.00
27C010-150	150	60	50	16.06	150.00	80.57	31.14	19.23	114.00	60.00	26.00	10.00	282.00	156.00	50.00
27C010-120	120	55	30	18.62	120.00	63.43	26.86	20.20	105.25	55.00	24.75	16.67	142.00	76.00	30.00
27C020-200	200	70	60	13.06	200.00	109.14	38.29	17.54	131.50	70.00	28.50	8.33	352.00	196.00	60.00
27C020-150	150	60	50	16.06	150.00	80.57	31.14	19.23	114.00	60.00	26.00	10.00	282.00	156.00	50.00
27C040-200	200	70	60	13.06	200.00	109.14	38.29	17.54	131.50	70.00	28.50	8.33	352.00	196.00	60.00
27C040-150	150	60	50	16.06	150.00	80.57	31.14	19.23	114.00	60.00	26.00	10.00	282.00	156.00	50.00

Flash	t_{ACC}	t_{OE}	t_{DF}	t_{WP}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	T_{WLWH}
28F256A-150	150	55	35	60	14.29	177.00	96.00	35.00	120.00
28F256A-120	120	50	30	60	16.67	142.00	76.00	30.00	100.00
28F512-150	150	55	35	60	14.29	177.00	96.00	35.00	120.00
26F512-120	120	50	30	60	16.67	142.00	76.00	30.00	100.00
28F010-150	150	55	35	60	14.29	177.00	96.00	35.00	120.00
28F010-120	120	50	30	60	16.67	142.00	76.00	30.00	100.00
28F020-200	200	60	40	60	12.50	212.00	116.00	40.00	140.00
28F020-150	150	55	35	60	14.29	177.00	96.00	35.00	120.00
28F001BX-150	150	55	35	50	14.29	177.00	96.00	35.00	120.00
28F001BX-120	120	50	30	50	16.67	142.00	76.00	30.00	100.00

NOTE:
A latch delay of 13 ns is included in the calculations.

Spreadsheet 3. Mode 2 Matched Memory Devices

				Mode 3											
				Based on t_{ACC}				Based on t_{OE}				Based on t_{DF}			
EPROMs	t_{ACC}	t_{OE}	t_{DF}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}
27C256-200	200	75	55	11.19	200.00	59.33	89.33	9.52	247.00	75.00	105.00	18.18	97.00	25.00	55.00
27C256-150	150	60	50	13.76	150.00	42.67	72.67	11.11	202.00	60.00	90.00	20.00	82.00	20.00	50.00
27C256-120	120	55	30	15.96	120.00	32.67	62.67	11.76	187.00	55.00	85.00	33.33	22.00	0.00	30.00
27C512-200	200	70	60	11.19	200.00	59.33	89.33	10.00	232.00	70.00	100.00	16.67	112.00	30.00	60.00
27C512-150	150	60	50	13.76	150.00	42.67	72.67	11.11	202.00	60.00	90.00	20.00	82.00	20.00	50.00
27C512-120	120	55	30	15.96	120.00	32.67	62.67	11.76	187.00	55.00	85.00	33.33	22.00	0.00	30.00
27C010-200	200	70	60	11.19	200.00	59.33	89.33	10.00	232.00	70.00	100.00	16.67	112.00	30.00	60.00
27C010-150	150	60	50	13.76	150.00	42.67	72.67	11.11	202.00	60.00	90.00	20.00	82.00	20.00	50.00
27C010-120	120	55	30	15.96	120.00	32.67	62.67	11.76	187.00	55.00	85.00	33.33	22.00	0.00	30.00
27C020-200	200	70	60	11.19	200.00	59.33	89.33	10.00	232.00	70.00	100.00	16.67	112.00	30.00	60.00
27C020-150	150	60	50	13.76	150.00	42.67	72.67	11.11	202.00	60.00	90.00	20.00	82.00	20.00	50.00
27C040-200	200	70	60	11.19	200.00	59.33	89.33	10.00	232.00	70.00	100.00	16.67	112.00	30.00	60.00
27C040-150	150	60	50	13.76	150.00	42.67	72.67	11.11	202.00	60.00	90.00	20.00	82.00	20.00	50.00

Flash	t_{ACC}	t_{OE}	t_{DF}	t_{WP}	Max Freq	T_{AVDV}	T_{RLDV}	T_{RHDZ}	T_{WLWH}
28F256A-150	150	55	35	60	11.11	202.00	60.00	90.00	60.00
28F256A-120	120	50	30	60	11.11	202.00	60.00	90.00	60.00
28F512-150	150	55	35	60	11.11	202.00	60.00	90.00	60.00
28F512-120	120	50	30	60	11.11	202.00	60.00	90.00	60.00
28F010-150	150	55	35	60	11.11	202.00	60.00	90.00	60.00
28F010-120	120	50	30	60	11.11	202.00	60.00	90.00	60.00
28F020-200	200	60	40	60	11.11	202.00	60.00	90.00	60.00
28F020-150	150	55	35	60	11.11	202.00	60.00	90.00	60.00
28F001BX-150	150	55	35	50	12.50	172.00	50.00	80.00	50.00
28F001BX-120	120	50	30	50	12.50	172.00	50.00	80.00	50.00

Spreadsheet 4. Mode 3 Matched Memory Devices

NOTE:
A latch delay of 13 ns is included in the calculations.

Summary of Maximum Frequencies at Different Modes

EPROMs	t_{ACC}	t_{OE}	t_{DF}	t_{WP}	Mode 0	Mode 1	Mode 2	Mode 3
27C256-200	200	75	55		18.18	10.99	9.09	9.52
27C256-150	150	60	50		20.00	13.45	10.00	11.11
27C256-120	120	55	30		26.60	15.54	16.67	11.76
27C512-200	200	70	60		16.67	10.99	8.33	10.00
27C512-150	150	60	50		20.00	13.45	10.00	11.11
27C512-120	120	55	30		26.60	15.54	16.67	11.76
27C010-200	200	70	60		16.67	10.99	8.33	10.00
27C010-150	150	60	50		20.00	13.45	10.00	11.11
27C010-120	120	55	30		26.60	15.54	16.67	11.76
27C020-200	200	70	60		16.67	10.99	8.33	10.00
27C020-150	150	60	50		20.00	13.45	10.00	11.11
27C040-200	200	70	60		16.67	10.99	8.33	10.00
27C040-150	150	60	50		20.00	13.45	10.00	11.11
FLASH								
28F256A-150	150	55	35	60	22.94	13.45	14.29	11.11
28F256A-120	120	50	30	60	26.60	15.54	16.67	11.11
28F512-150	150	55	35	60	22.94	13.45	14.29	11.11
28F512-120	120	50	30	60	26.60	15.54	16.67	11.11
28F010-150	150	55	35	60	22.94	13.45	14.29	11.11
28F010-120	120	50	30	60	26.60	15.54	16.67	11.11
28F020-200	200	60	40	60	18.66	10.99	12.50	11.11
28F020-150	150	55	35	60	22.94	13.45	14.29	11.11
28F001BX-150	150	55	35	50	22.94	13.45	14.29	12.50
28F001BX-120	120	50	30	50	26.60	15.54	16.67	12.50

Spreadsheet 5. Compilation of All Modes at Maximum Speeds with Matched Memories

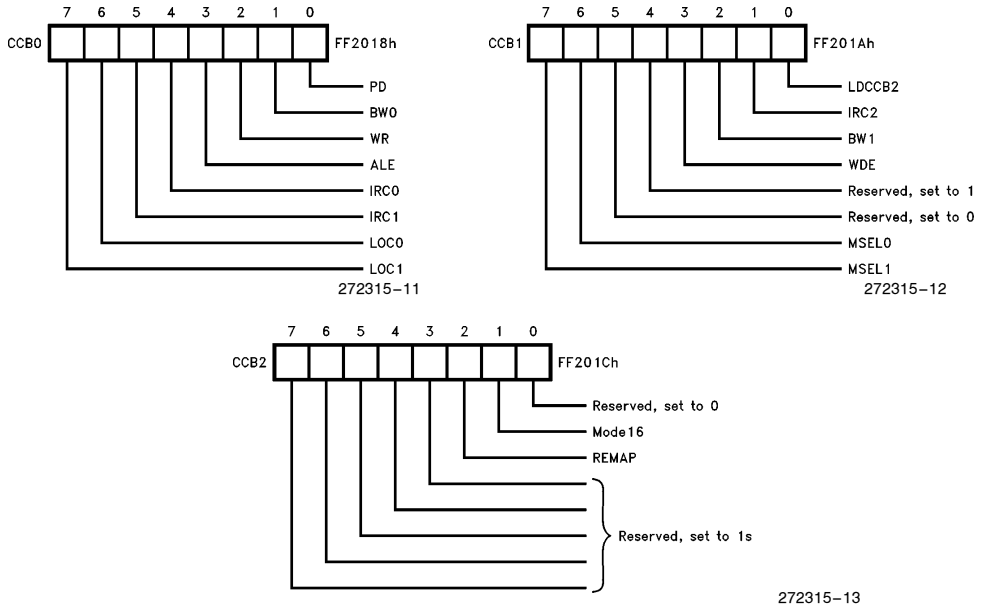
NOTE:
A latch delay of 13 ns is included in the calculations.

7.0 CHIP CONFIGURATION BYTES

7.1 CCBs

There are three CCBs on the 8XC196NT: CCB0, CCB1, and CCB2. Like the other devices in the MCS-96 family, following RESET a CCB fetch will occur. If $\overline{EA}$ = 1 at the end of RESET, the CCBs are fetched from internal memory. If $\overline{EA}$ = 0 at the end of RESET, the CCBs are fetched from external memory. But regardless of the value of $\overline{EA}$, the extended address bus is forced to 0FFh during the CCB fetch. Therefore, the CCBs must be located at 0FF2018h, 0FF201Ah, and 0FF201Ch, respectively, in internal memory ($\overline{EA}$ = 1). Or if the EPORT is used as an extended address port and $\overline{EA}$ = 0, the CCBs must be located at xF2018h, xF201Ah, and xF201Ch, respectively, in external memory. As in the 8XC196KR device, after RESET, the 8XC196NT is configured to work in 16-bit mode, independent of the BUSWIDTH input. However, weak holding latches on Port 4 (AD8–AD15) retain the high order address byte on the bus. Therefore, the CCBs are still fetched in 8-bit external systems provided the high byte address of the CCB is 20h. After the CCBs are read and written to the Chip Configuration Registers (CCRs), the bus is configured as either 8-bit, 16-bit, or BUSWIDTH-controlled based on the CCBs. The CCRs are written only during the reset sequence. The device must be reset to change the values of the CCRs.

The bits of the CCBs correspond to the bits of the CCRs. The CCBs and their bits' functions are shown in Figure 7-1. Except for some changes to CCB1 and the addition of CCB2, the functions of CCB0 and CCB1 remain the same as the KR device so the 8XC196KR User's Manual can be referenced. CCB1 has the addition of load CCB2 (LDCCB2) and bus timing mode select (MSEL0–MSEL1). CCB2 provides the means of selecting the mode of the device and the memory mapping which are described in the memory map section.



IRC2	IRC1	IRC0	Max Wait States
0	0	0	Zero Wait States
1	0	0	1 Wait State
1	0	1	2 Wait States
1	1	0	3 Wait States
1	1	1	Infinite Wait States

MSEL1	MSEL0	Bus Timing Mode
0	0	Mode 0 (Standard with 1 Wait State)
0	1	Mode 1 (Long R/W—Address Enabled Sooner)
1	0	Mode 2 (Early Address—ALE Rising Edge Occurs Sooner)
1	1	Mode 3 (Standard)

LOC1	LOC0	Function
0	0	Read and Write Protected
0	1	Write Protected Only
1	0	Read Protected Only
1	1	No Protection

BW1	BW0	Bus Width
0	0	ILLEGAL
0	1	16-Bit Only
1	0	8-Bit Only
1	1	BW Pin Controlled

PD = 1 to Enable Powerdown
= 0 to enable Powerdown

WR = 1 for $\overline{WR}/\overline{BHE}$ signals
= 0 for $\overline{WRL}/\overline{WRH}$ signals

ALE = 1 for $\overline{ALE}$ signal
= 0 for $\overline{ADV}$ signal

REMAP = 1 for remap of internal (EP)R0M to 0FF2000h only
= 0 for remap of internal (EP)R0M to 002000h and 0FF2000h

LDCCB2 = 1 for CCB2 Fetch
= 0 for no CCB2 Fetch

WDE = 0 for WDT always enabled
= 1 for S/W or RST enable WDT

Mode16 = 1 for 16-bit mode
= 0 for 24-bit mode

Figure 7-1. CCB Definitions

8.0 QUESTIONS AND ANSWERS

Q: Where are the interrupt vectors located?

A: The interrupt vectors are located at 0FF2000h to 0FF2013h and 0FF2030h to 0FF203Fh.

Q: If the device is operating in 24-bit mode can the EPORT be used for I/O?

A: Yes, the EP__REG drives a value on the EPORT if the output option is selected by EP__MODE and EP__DIR for a given pin.

Q: Will access to all SFRs always default to page 00h, thus accessible using LD/ST instructions?

A: No, SFRs for the EPORT, Port 3, 4, and 5, and Slave Port can only be accessed with ELD/EST instructions if the current value on the extended address bus is a value other than 00h. All other SFRs can be accessed using the LD/ST instructions and windowing.

Q: Can access to the Register File, located at 00–3FFh, be accomplished using non-extended addressing instructions regardless of current page in EP__REG?

A: Yes, access to Register File always defaults to page 00h.

Q: Are there any special considerations when operating in 24-bit mode and using the stack?

A: Yes, a subroutine call will decrement the SP by 4 and push a 4-byte return address onto the stack. Similarly, the return instruction pops a 4-byte return address and increments SP by 4.

Q: Why is the data being written to 0A000h with a 16-bit instruction actually written to 0FFA000h?

A: The contents of EP__REG is 0FFh. EP__REG must be cleared to 00h or an extended address instruction must be used to write to memory regions in page 00h.

Q: What page does the stack pointer (SP) point to?

A: The value in the EP__REG determines the page that the SP points to. So if SP = 1000h and EP__REG = 01h, the operations on the stack will be performed starting at 011000h.