intel®

AB-33

APPLICATION BRIEF

Memory Expansion for the 8096

DOUG YODER
ECO APPLICATIONS ENGINEER

April 1989

Order Number: 270522-001

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

MEMORY EXPANSION FOR THE 8096

CONTENTS

This Application Brief presents two examples of a paging scheme for the 8096, allowing either 256K bytes of total memory, or 544K bytes of total memory. Both systems utilize PORT1 as the output for the upper address lines. Because Interrupt vectors, and other critical sections of code must always be present, addresses 0-7FFFH always refer to the same main page. The PORT1 upper addresses only affect addresses 8000-FFFFH, by slapping several 32K pages in and out.

THE 256K SYSTEM

Hardware

The hardware for the 256K system (see Figures 4 & 5, an example with 128K ROM and 128K RAM) utilizes a 74LS157 quad 2 to 1 multiplexer. The enable pin of the 74LS157 is tied to the inverted A15 signal, which is the latched addr/data 15 (AD15) signal from the 96. In this way, when A15 is low, the 74LS157 is disabled and all its outputs are low. Particularly, MA17 is low, which selects the 27512 and deselects the rams. Also, MA15 and MA16 are low, which guarantee that addresses 0-7FFFH of the 27512 are accessed.

When A15 is high, the 74LS157 is enabled to pass MA15 - MA17 values. The bank select pin of the 74LS157 is connected to the INST pin of the 96. When the INST pin is high, for a code access, INSTA15 - INSTA17 (PORT1.0 - PORT1.2) are used. When INST is low, for a data read or write, DATAA15 - DATAA17 (PORT1.3 - PORT 1.5) are used. This allows for the use of separate pages for code and data without having to change the upper address lines each time. Also, it is possible to select a ROM page for a data table, or load a RAM page with executable code downloaded from another source. PORT1.6 and P0RT1.7 can still be used as I/O ports. If a -90 part were used, the INST pin would need to be latched since it is only valid during the address output on the bus pins.

This system was designed to get the maximum amount of memory with a minimum amount of hardware. The amount of ROM and RAM was picked arbitrarily, and could be reconfigured in various ways, however, this may require slight modifications or additions to the decoder circuitry. This setup has a main page at addresses 0-7FFFH, and upper pages 1-7 at addresses 8000-FFFFH. Note that upper page 0 is the same as the main page. The WRL and WRH feature of the BH part was used to allow for byte writes to RAM. If the -90 part were to be used, additional logic would be necessary to generate these signals from WR and BHE.

The RAM chips utilized were NEC uPD43256-15 32K x 8 static rams with an access time of 150ns. The ROMs were Intel 27512 64K x 8 EPROMs with an access time of 200ns. The decoder circuitry used was entirely LS TTL. Using an 8097BH running at 10MHz, there was ample time for address decoding and memory access. Timing analysis showed that 12MHz operation would also be accommodated easily. If slower memories are used, further analysis would be necessary. Also, it would be possible to switch to S TTL to greatly decrease the decoding response time.

Software

When using this system there are several things to keep in mind when preparing the software.

Since ASM96 will only allow addresses from 0-FFFFH, it is necessary to generate each page of code in a separate file. These pages should not be linked together, but rather should each be used to program the proper section of the EPROM associated with that page. The main page routine should be coded with addresses from 0-7FFFH, and each of the upper pages should be coded with addresses from 8000-FFFH. Because linking is not possible, each module should contain a table of constants which defines the symbols used in other modules. These values are easily obtained from the listing file, which can be created using zeros in the table the first time. The addresses of the pages in a 27512 after splitting low and high bytes into 2 EPROMs are shown in Figure 1.

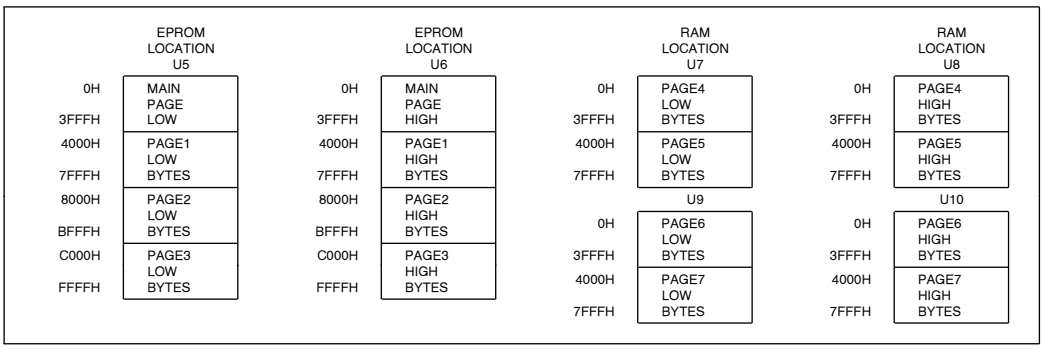


Figure 1. The Current System

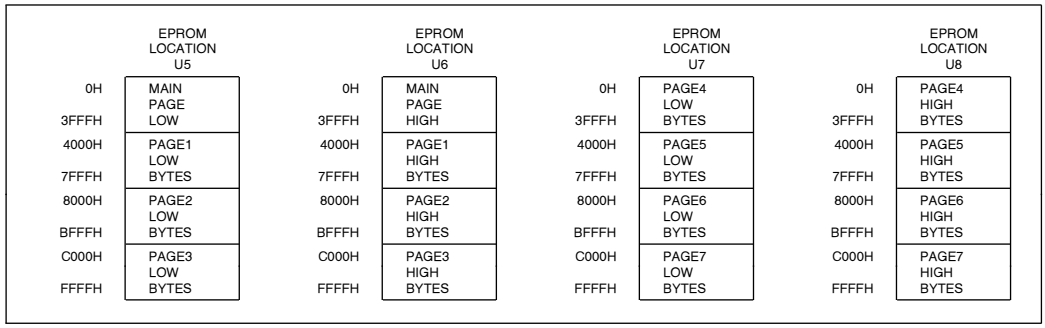


Figure 2. A System Using all EPROMS and no RAM

All changes to the upper instruction addresses of PORT1 must be made by code located in the main page. A listing of subroutines for use in the main page, and a listing of macros for use in all pages is provided. By invoking one of these macros the programmer can easily transfer from one page to another, or select a new data page. The subroutines should not be called directly, they should be entered by using the appropriate macro. The subroutines should be located at the addresses specified, otherwise the macros must be changed as they are written to call an absolute address in the main page. Also, any hardware changes may render the software inoperative.

Because the WRL-WRH feature of the 96BH is used, the correct Chip Configuration Register value of 0FBH must be loaded into the ROMs at address 2018H. This is done in the main code file with the following statements:

```
CSEG AT 2018H

CCR: DCB 0FBH   ;VALUE FOR CHIP
                 CONFIGURATION REGISTER
```

Finally, it is necessary to initialize the DATA address at the start of the program this can be done using the NEW_DATA_PAGE MACRO.

THE 544K SYSTEM

Hardware

The hardware for the 544K system (see Figures 6 & 7, an example with 288K ROM and 256K RAM) has some slight changes from the 256K system.

First, all pins of PORT1 are now in use as address lines. This allows for PORT1 to select 16 pages of memory, with a different address for instructions or data.

Second, 27128 16K x 8 EPROMS have been added for use as the main code page. In this system, the main page is physically separate from upper page 0. The 27128's are selected by A15 being low. The upper pages of memory are selected when A15 is high which enables the 74LS155 demultiplexer which is used for address decoding. When the 74LS155 is disabled, its outputs are all high, which disables all upper memories. The 74LS157 is enabled all the time, to speed up address decoding, as its outputs do not matter when the 74LS155 is disabled.

Software

All rules for the 256K system apply to the 544K system, except that the main page no longer overlaps page 0. However, because all of PORT1 is now in use, different macros and subroutines must now be used. These have been included also.

THE INST PIN

The instruction pin has been verified to work correctly on the 8X9X- 90, 8X9XBH, and the 80C196. The functionality of the INST pin is as follows.

Instruction Fetches

The INST pin is high during an external memory read indicating the read is an instruction fetch. This includes immediate data reads since the data is embedded in the code.

Data Reads and Writes

The INST is low during an external memory read or write indicating the bus cycle is a data cycle. This would be indirect and indexed instructions which are directed at external memory.

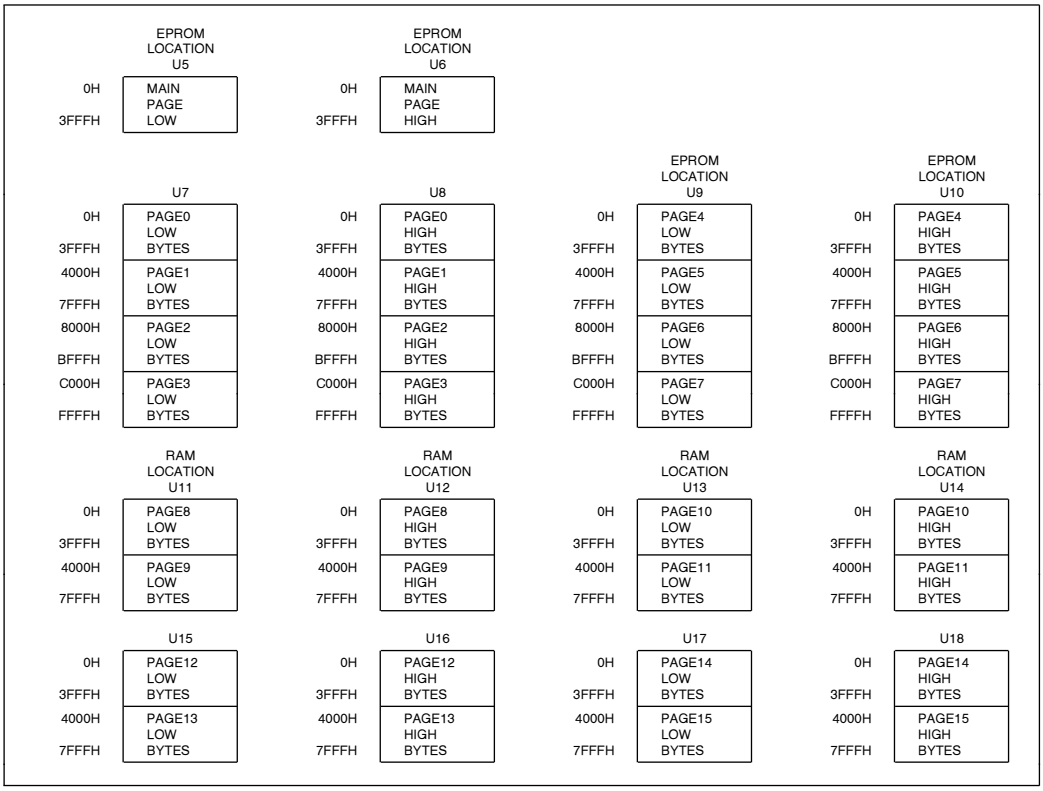
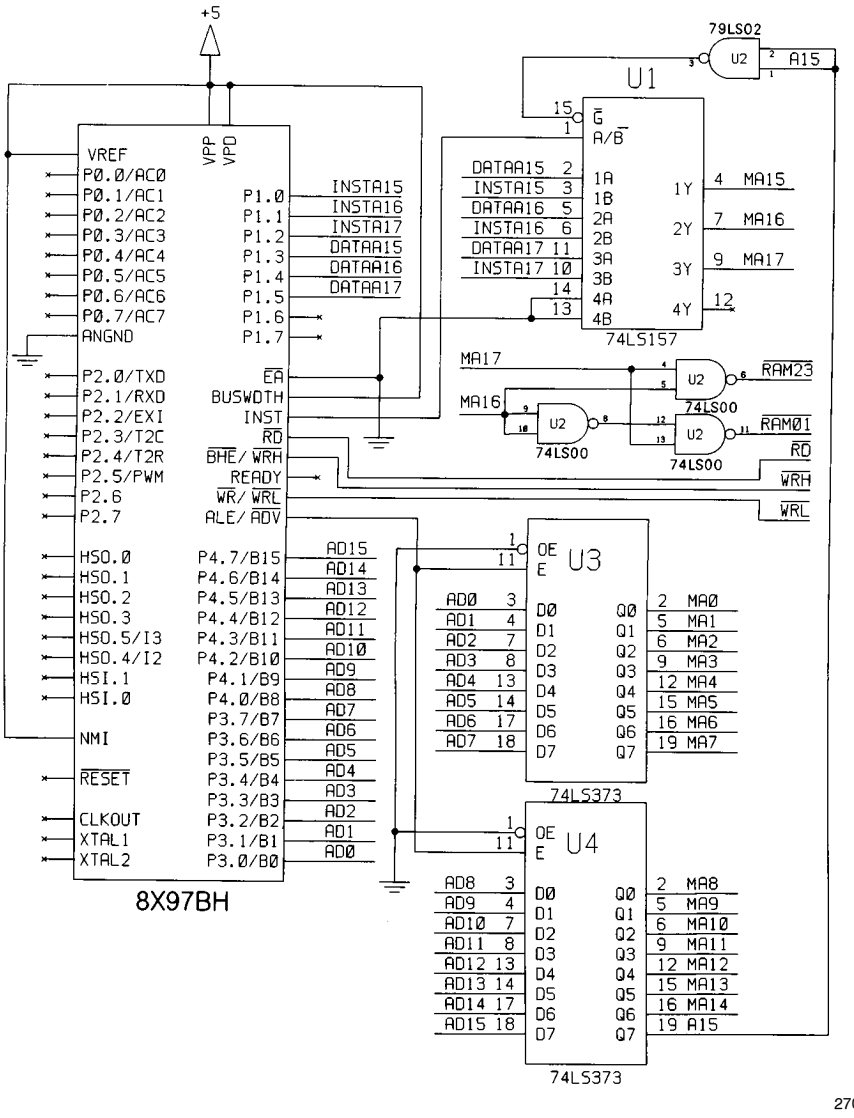


Figure 3. The 544K Memory Map



NOTE:
All other connections should be made as specified in current Embedded Controller Handbook.

Figure 4. 128K ROM + 128K RAM Memory

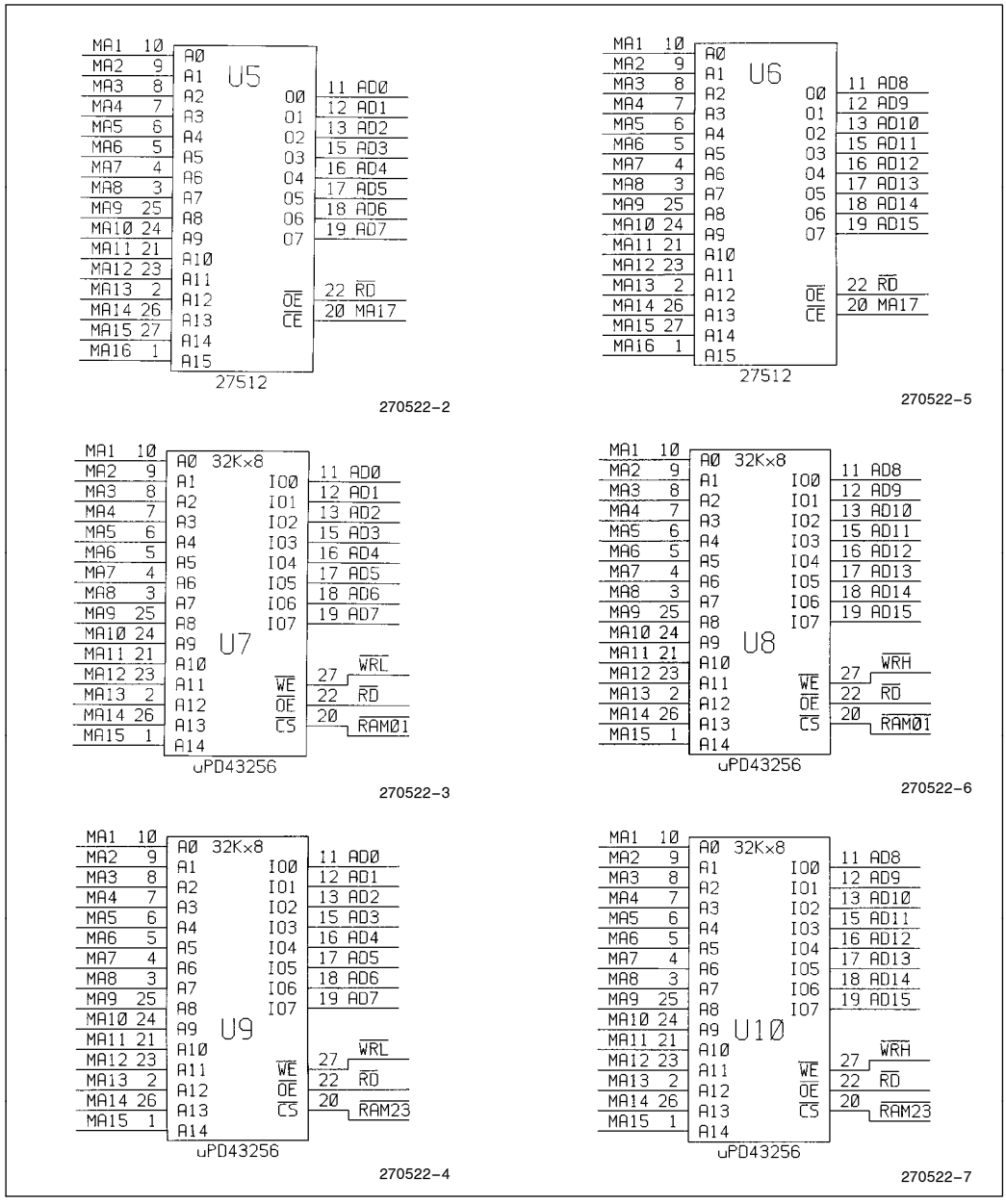
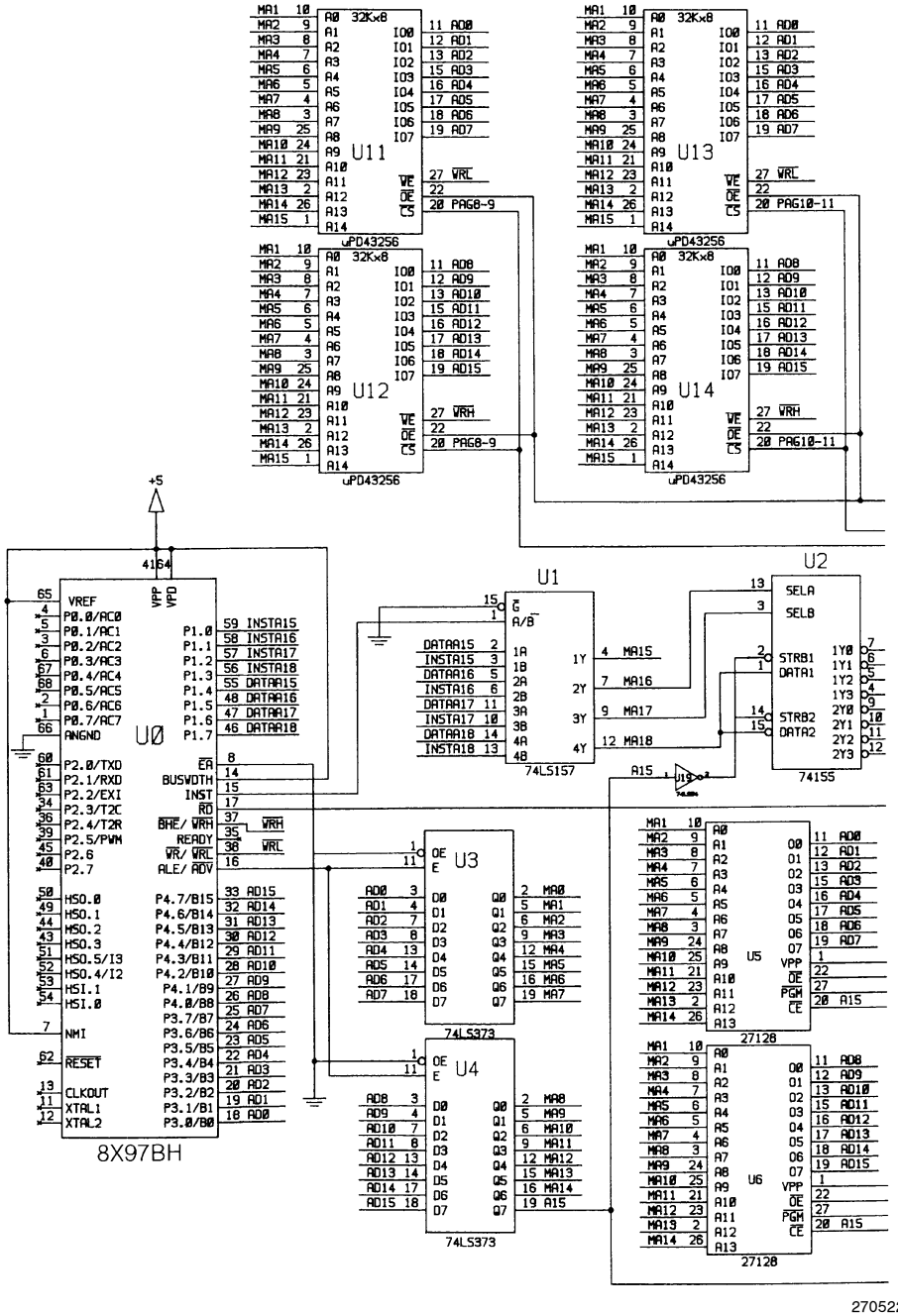


Figure 5. 128K ROM + 128K RAM Memory

```
;MACROS FOR 256K SYSTEM

;LONG_BRANCH IS INVOKED TO BRANCH FROM ONE PAGE TO ANOTHER.
       ;ADDRESS MUST HAVE A VALUE FROM 8000H TO FFFFH.
       ;NEW_PAGE CAN BE AN IMMEDIATE NUMBER OR A REGISTER NUMBER.

LONG_BRANCH    MACRO  ADDRESS, NEW_PAGE
               LD     CODE_ADDRESS, #ADDRESS ;SET UP CODE_ADDRESS REGISTER
               LDB    NEW_PAGE_NO, NEW_PAGE  ;SET UP NEW_PAGE_NO REGISTER
               BR     7FF0H                  ;BRANCH TO I_P_BRANCH
               ENDM


;LONG_CALL IS INVOKED TO CALL A SUBROUTINE IN ANOTHER PAGE.
       ;ADDRESS MUST HAVE A VALUE FROM 8000H TO FFFFH.
       ;NEW_PAGE CAN BE AN IMMEDIATE NUMBER OR A REGISTER NUMBER.

LONG_CALL      MACRO  ADDRESS, NEW_PAGE
               LD     CODE_ADDRESS, #ADDRESS ;SET UP CODE_ADDRESS REGISTER
               LDB    NEW_PAGE_NO, NEW_PAGE  ;SET UP NEW_PAGE_NO REGISTER
               CALL   7FC0H                  ;CALL I_P_CALL
               ENDM

;PUSH_OLD_DATAPAGE IS INVOKED TO INSTALL A NEW DATA PAGE AND SAVE
;THE OLD VALUE ON THE SYSTEM STACK.
       ;NEW_PAGE CAN BE AN IMMEDIATE NUMBER OR A REGISTER NUMBER.

PUSH_OLD_DAPAG MACRO  NEW_PAGE
               LDB    AL, PORT1              ;GET OLD PAGE NUMBER...
               PUSH   AX                     ;STORE IT ON THE STACK
               LDB    AL, NEW_PAGE           ;GET NEW DATA PAGE NUMBER...
               ANDB   AL, #00000111B         ;MASK IT...
               SHLB   AL, #3                 ;SHIFT IT TO PROPER POSITION...
               ANDB   PORT1, #11000111B      ;CLEAR THE OLD ONE...
               ORB    PORT1, AL              ;AND LOAD IN NEW ONE
               ENDM

;POP_OLD_DATAPAGE IS INVOKED TO REINSTALL AN OLD DATA PAGE THAT WAS SAVED
;ON THE SYSTEM STACK BY PUSH_OLD_DATAPAGE.

POP_OLD_DAPAG  MACRO
               POP    AX                     ;RECALL OLD PAGE NUMBER...
               ANDB   AL, #00111000B         ;MASK OLD ONE FOR DATA PAGE...
               ANDB   PORT1, #11000111B      ;CLEAR NEW DATA PAGE...
               ORB    PORT1, AL              ;AND LOAD IN OLD ONE
               ENDM

;NEW_DATA_PAGE IS INVOKED TO INSTALL A NEW DATA PAGE.
       ;NEW_PAGE CAN BE AN IMMEDIATE NUMBER OR A REGISTER NUMBER.

NEW_DATA_PAGE  MACRO  NEW_PAGE
               LDB    AL, NEW_PAGE           ;GET NEW DATA PAGE NUMBER...
               ANDB   AL, #00000111B         ;MASK IT...
               SHLB   AL #3                  ;SHIFT IT TO PROPER POSITION...
               ANDB   PORT1, #11000111B      ;CLEAR THE OLD ONE...
               ORB    PORT1, AL              ;AND LOAD IN NEW ONE
               ENDM
```

```
;SUBROUTINES FOR 256K SYSTEM

CSEG AT 7FC0H

;SUBROUTINE:   I_P_CALL
;        THIS SUBROUTINE ALLOWS FOR THE CALLING OF SUBROUTINES LOCATED IN
;        A DIFFERENT PAGE OF MEMORY.
;
;        PARAMETERS:     CODE_ADDRESS, NEW_PAGE_NO
;        SUBROUTINES:    ANY THAT ARE REQUESTED.
;
I_P_CALL:       LDB     AL, PORT1               ;GET OLD PAGE NUMBER...
                PUSH    AX                      ;STORE IT ON THE STACK
                ANDB    PORT1, #11111000B       ;CLEAR OLD INST PAGE...
                ANDB    NEW_PAGE_NO, #00000111B ;MASK NEW ONE...
                ORB     PORT1, NEW_PAGE_NO      ;AND LOAD IT IN
                PUSH    #I_P_RETURN             ;SAVE RETURN ADDRESS...
                BR      [CODE_ADDRESS]          ;CALL REQUESTED ROUTINE

I_P_RETURN:     POP     AX                      ;RECALL OLD PAGE NUMBER...
                ANDB    PORT1, #11111000B       ;CLEAR NEW INST PAGE...
                ANDB    AL, #00000111B          ;MASK OLD ONE...
                ORB     PORT1, AL               ;AND LOAD IT IN
                RET                             ;RETURN TO CALLING ROUTINE

CSEG AT 7FF0H

;SUBROUTINE:    I_P_BRANCH
;        THIS SUBROUTINE ALLOWS FOR BRANCHING TO LOCATIONS IN A DIFFERENT
;        PAGE OF MEMORY.
;
;        PARAMETERS:     CODE_ADDRESS, NEW_PAGE_NO
;        SUBROUTINES:    NONE
;
I_P_BRANCH:     ANDB    PORT1, #11111000B       ;CLEAR OLD INST PAGE...
                ANDB    NEW_PAGE_NO #00000111B  ;MASK NEW ONE...
                ORB     PORT1, NEW_PAGE_NO      ;AND LOAD IT IN
                BR      [CODE_ADDRESS]          ;BRANCH TO REQUESTED
ROUTINE
```



NOTE:
All other connections should be made as specified in current Embedded Controller Handbook.

Figure 6. 288K ROM + 256K RAM Memory

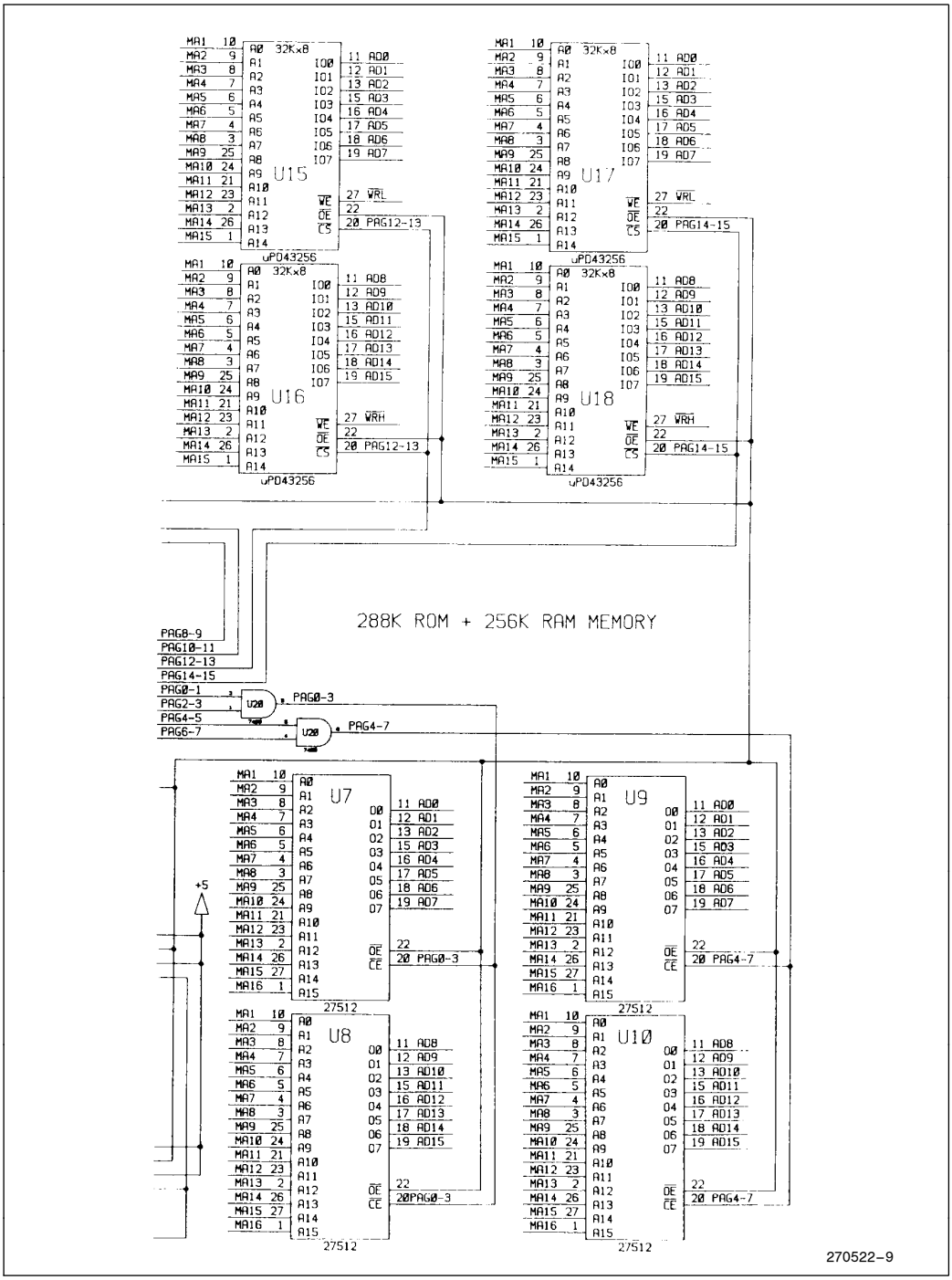


Figure 7. 288K ROM + 256K RAM Memory

```
;MACROS FOR 544K SYSTEM

;LONG_BRANCH IS INVOKED TO BRANCH FROM ONE PAGE TO ANOTHER.
       ;ADDRESS MUST HAVE A VALUE FROM 8000H TO FFFFH.
       ;NEW_PAGE CAN BE AN IMMEDIATE NUMBER OR A REGISTER NUMBER.

LONG_BRANCH    MACRO  ADDRESS, NEW_PAGE
               LD     CODE_ADDRESS, #ADDRESS ;SET UP CODE_ADDRESS REGISTER
               LDB    NEW_PAGE_NO, NEW_PAGE  ;SET UP NEW_PAGE_NO REGISTER
               BR     7FF0H                  ;BRANCH TO I_P_BRANCH
               ENDM

LONG_CALL IS INVOKED TO CALL A SUBROUTINE IN ANOTHER PAGE.
       ;ADDRESS MUST HAVE A VALUE FROM 8000H TO FFFFH.
       ;NEW_PAGE CAN BE AN IMMEDIATE NUMBER OR A REGISTER NUMBER.

LONG_CALL      MACRO  ADDRESS, NEW_PAGE
               LD     CODE_ADDRESS, #ADDRESS ;SET UP CODE_ADDRESS REGISTER
               LDB    NEW_PAGE_NO, NEW_PAGE  ;SET UP NEW_PAGE_NO REGISTER
               CALL   7FC0H                  ;CALL I_P_CALL
               ENDM

;PUSH_OLD_DATAPAGE IS INVOKED TO INSTALL A NEW DATA PAGE AND SAVE THE OLD
;VALUE ON THE SYSTEM STACK.
       ;NEW_PAGE CAN BE AN IMMEDIATE NUMBER OR A REGISTER NUMBER.

PUSH_OLD_DAPAG MACRO  NEW_PAGE
               LDB    AL, PORT1              ;GET OLD PAGE NUMBER...
               PUSH   AX                     ;STORE IT ON THE STACK
               LDB    AL, NEW_PAGE           ;GET NEW DATA PAGE NUMBER:..
               SHLB   AL, #4                 ;SHIFT IT TO PROPER POSITION...
               ANDB   PORT1, #00001111B      ;CLEAR THE OLD ONE...
               ORB    PORT1, AL              ;AND LOAD IN NEW ONE
               ENDM

;POP_OLD_DATAPAGE IS INVOKED TO REINSTALL AN OLD DATA PAGE THAT WAS SAVED
;ON THE SYSTEM STACK BY PUSH_OLD_DATAPAGE.

POP_OLD_DAPAG  MACRO
               POP    AX                     ;RECALL OLD PAGE NUMBER...
               ANDB   AL, #11110000B         ;MASK OLD ONE FOR DATA PAGE...
               ANDB   PORT1, #00001111B      ;CLEAR NEW DATA PAGE...
               ORB    PORT1, AL              ;AND LOAD IN OLD ONE
               ENDM

;NEW_DATA_PAGE IS INVOKED TO INSTALL A NEW DATA PAGE.
       ;NEW_PAGE CAN BE AN IMMEDIATE NUMBER OR A REGISTER NUMBER.

NEW_DATA_PAGE  MACRO  NEW_PAGE
               LDB    AL, NEW_PAGE           ;GET NEW DATA PAGE NUMBER...
               SHLB   AL, #4                 ;SHIFT IT TO PROPER POSITION...
               ANDB   PORT1, #00001111B      ;CLEAR THE OLD ONE...
               ORB    PORT1, AL              ;AND LOAD IN NEW ONE
               ENDM
```

```
;SUBROUTINES FOR 544K SYSTEM

CSEG AT 7FC0H

;SUBROUTINE:   I_P_CALL
;       THIS SUBROUTINE ALLOWS FOR THE CALLING OF SUBROUTINES LOCATED IN
;       A DIFFERENT PAGE OF MEMORY.
;
;       PARAMETERS:     CODE_ADDRESS, NEW_PAGE_NO
;       SUBROUTINES:    ANY THAT ARE REQUESTED.
;
I_P_CALL:       LDB     AL, PORT1               ;GET OLD PAGE NUMBER...
                PUSH    AX                      ;STORE IT ON THE STACK
                ANDB    PORT1, #11110000B       ;CLEAR OLD INST PAGE...
                ANDB    NEW_PAGE_NO, #00001111B ;MASK NEW ONE...
                ORB     PORT1, NEW_PAGE_NO      ;AND LOAD IT IN
                PUSH    #I_P_RETURN             ;SAVE RETURN ADDRESS...
                BR      [CODE_ADDRESS]          ;CALL REQUESTED ROUTINE

I_P_RETURN:     POP     AX                      ;RECALL OLD PAGE NUMBER...
                ANDB    PORT1, #11110000B       ;CLEAR NEW INST PAGE...
                ANDB    AL, #00001111B          :MASK OLD ONE...
                ORB     PORT1, AL               ;AND LOAD IT IN
                RET                             ;RETURN TO CALLING ROUTINE

CSEG AT 7FF0H

;SUBROUTINE:    I_P_BRANCH
;       THIS SUBROUTINE ALLOWS FOR BRANCHING TO LOCATIONS IN A DIFFERENT
;       PAGE OF MEMORY.
;
;       PARAMETERS:     CODE_ADDRESS, NEW_PAGE_NO
;       SUBROUTINES:    NONE
;
I_P_BRANCH:     ANDB    PORT1, #11110000B       ;CLEAR OLD INST PAGE...
                ANDB    NEW_PAGE_NO, #00001111B ;MASK NEW ONE...
                ORB     PORT1, NEW_PAGE_NO      ;AND LOAD IT IN
                BR      [CODE_ADDRESS]          ;BRANCH TO REQUESTED ROUTINE
```