As you are now the owner of this document which should have come to you for free, please consider making a donation of £1 or more for the upkeep of the (Radar) website which holds this document. I give my time for free, but it costs me money to bring this document to you. You can donate here https://blunham.com/Misc/Texas

Many thanks.

Please do not upload this copyright pdf document to any other website. Breach of copyright may result in a criminal conviction.

This Acrobat document was generated by me, Colin Hinson, from a document held by me. I requested permission to publish this from Texas Instruments (twice) but received no reply. It is presented here (for free) and this pdf version of the document is my copyright in much the same way as a photograph would be. If you believe the document to be under other copyright, please contact me.

The document should have been downloaded from my website https://blunham.com/, or any mirror site named on that site. If you downloaded it from elsewhere, please let me know (particularly if you were charged for it). You can contact me via my Genuki email page: https://www.genuki.org.uk/big/eng/YKS/various?recipient=colin

You may not copy the file for onward transmission of the data nor attempt to make monetary gain by the use of these files. If you want someone else to have a copy of the file, point them at the website. (https://blunham.com/Misc/Texas). **Please do not point them at the file itself as it may move or the site may be updated.**

It should be noted that most of the pages are identifiable as having been processed by me.

I put a lot of time into producing these files which is why you are met with this page when you open the file.

If you find missing pages, pages in the wrong order, anything else wrong with the file or simply want to make a comment, please drop me a line (see above).

It is my hope that you find the file of use to you.

Colin Hinson
In the village of Blunham, Bedfordshire.

TEXAS INSTRUMENTS

AMPL

TMAM6083 Emulator

For TMS9995

MICROPROCESSOR SERIES™

User's Guide

TMAM 6083 EVM/EMULATOR MODULE

TABLE OF CONTENTS

SECTION 1: INTRODUCTION

SECTION 2: INSTALLATION AND SYSTEM CHECKOUT

SECTION 3: EMUBUG INTERACTIVE MONITOR

SECTION 4: THEORY OF OPERATION

SECTION 5: MODIFICATIONS TO THE TMAM 6095 EVM

LIST OF ILLUSTRATIONS

LIST OF TABLES

SECTION 1

OVERVIEW

1.1 GENERAL

The TMAM 6083 is a stand-alone in-circuit emulator for the TMS9995 microcomputer. The Emulator assembly is a dual-board system designed to enable the user to test hardware and software of TMS 9995 systems. A minimal system consists of the following components:

- A TMAM 6083 Emulator unit, consisting of:
  - An Emulator Board, part number 1603212
  - A modified TMAM 6095 Evaluation Board, part number 1603152-2
  - A target connector assembly, part number 1603352
- An RS-232-compatible CRT terminal (user supplied)
- Power supply unit, part number TM990/518 (user supplied)

Note:- The target system clock option is not supported on the Target connector Assembly 1603352-0001

No host system is required.

Its features include:

- Full-speed, no-wait-state in-circuit emulation for TMS 9995-12 (12 MHz).
- Complete access to user target system memory and I/O spaces, i.e., no restriction on target system usage of these spaces.
- 256-state trace of the 16-bit address bus and the 8-bit external data bus.
- 8K bytes of user memory
- Second EIA serial port for download of software from an optional TI FS/990, DS/990, or TMAM 9000 host system.
- An interactive DEBUG monitor (EMUBUG).
- A versatile hardware breakpoint capability on any memory access. Qualifiers are user selectable to breakpoint on read, write, instruction fetch, or any combination thereof.

- Symbolic assembler.
- Reverse assembler.

1.2 TERMINOLOGY

Throughout this document, the following terminology will be used:

Emulator: The TMAM 6083 two-board assembly.

EVM or EVM Board: The modified TMAM 6095 Evaluation Module for the TMS 9995 microcomputer. (Detailed information concerning this board is given in the TMAM 6095 Evaluation Module For The TMS9995 User's Guide; hardware is referenced in Section 5.)

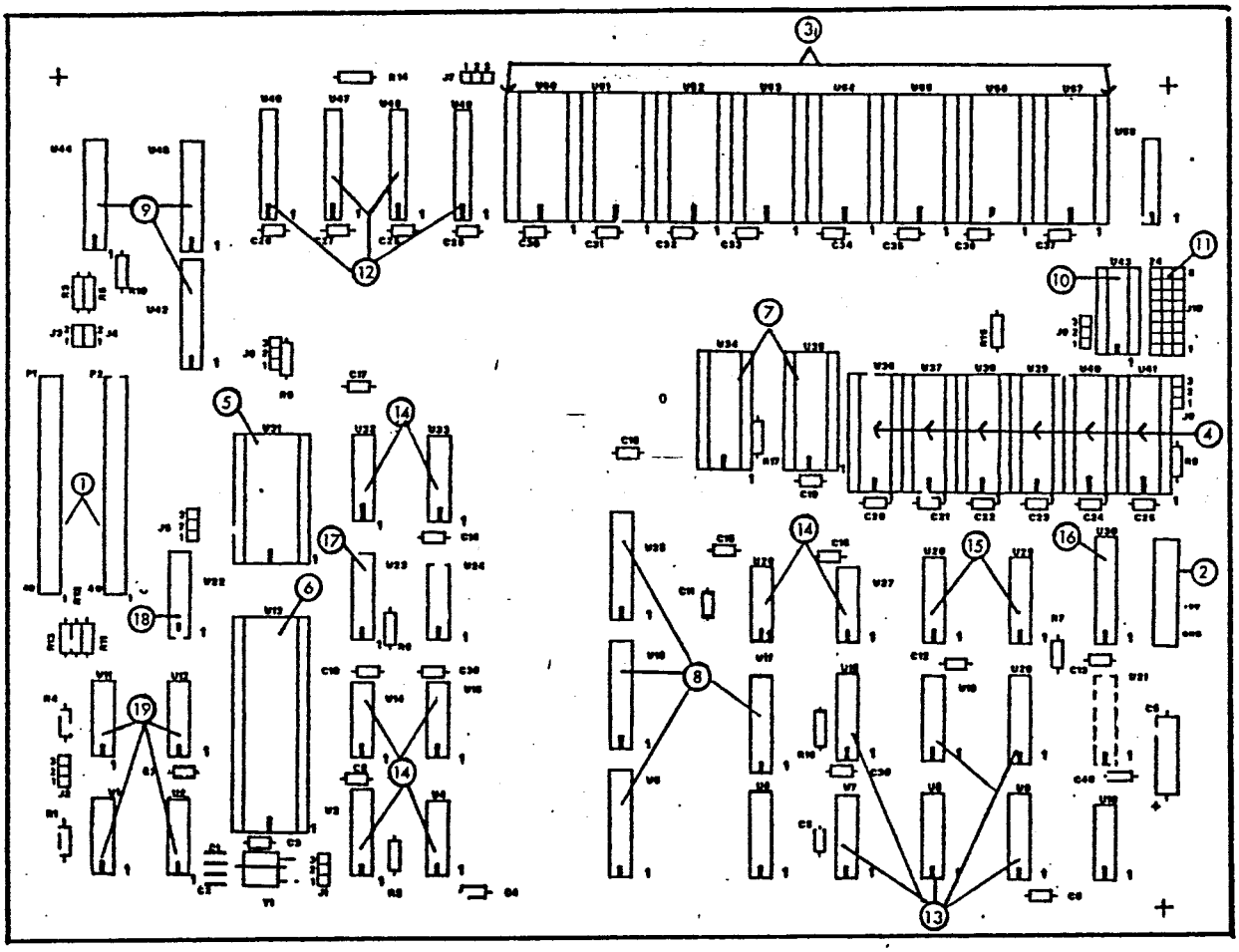
EMU or EMU Board: The second board of the Emulator assembly.

Target Connector or Target Connector Assembly: The actual interface mechanism between the Emulator and the Target System.

1.3 CONFIGURATION

The EVM and EMU boards are joined, component sides out, by a 38-pin connector located approximately in the center of the boards. The terminal for the Emulator is connected to EIA Port 1 of the EVM board; the optional host system is connected to Port 2. The EVM controls the Emulator, but functions completely asynchronous to the system.

Figure 1-1 illustrates the Emulator board portion of the Emulator System.



1. Target Connector Cable Sockets (P1, P2)
2. Power Cable Socket
3. User RAM Chips (U50-U57)
4. Trace Memory (U36-U41)
5. Virtual RAM (U31)
6. TMS 9995 Socket (U13)
7. Breakpoint and Trace Address Memories (U34, U35)
8. Interface Buffers TO EVM Board (U5, U16, U17, U25)
9. Interface Buffers To Target System (U42, U44, U45)
10. Mapping PROM (U43)
11. Map-select Jumpers (J10)
12. Breakpoint (U46, U48) and IAQ (U47, U49) Address Registers
13. CRU Decoders (U6-U9, U18-U20)
14. Breakpoint and Trace Decode Logic (U26/U27/U32/U33/U14/U15/U3/U4)
15. Trace State Counters (U28, U29)
16. Address Buffer For Reading Trace Memory (U30)
17. Virtual RAM Control Multiplexor (U23)
18. Target System Buffer (U22)
19. TMS 9995 Ancillary Decode Logic

FIGURE 1-1. TMAM 6083 EMULATOR BOARD.

1.4 REFERENCE MATERIAL

The following documents will provide support information:

- TMAM 6095 Evaluation Module For TMS9995 User's Guide, MP 401 (NOTE: in this document, this manual is refered to as the EVM User's Guide)
- TMS9995 Microcomputer Data Manual, MP021

SECTION 2

INSTALLATION

2.1 GENERAL

This section provides instructions for installation of the TMAM 6083 Emulator. The following paragraphs supply information for unpacking, set-up, and powering up the system, as well as providing parameters for power, space and environmental requirements.

2.2 REQUIRED EQUIPMENT

The following components are required for the TMAM 6083 Emulator:

- The two-board Emulator assembly
- Target connector assembly
- Two target connector/emulator interconnect cables
- Power supply cable
- TM990/518 power supply unit, or equivalent
- Terminal: EIA RS-232, or 20ma current loop-compatible TTY.

2.3 SPACE AND ENVIRONMENTAL REQUIREMENTS

The Emulator setup requires adequate space on a flat, non-conductive horizontal surface. The space must allow room at the rear for EIA cable connections to the EVM board, and the placement of the power supply. Space to the side is needed for the target system and interface cables. Clearance at the front is necessary for user access to both the Emulator and the terminal. Additional space for a target system terminal may also be necessary, and, if desired, workspace for an oscilloscope. All space provided should be free of any material that could block the ventilation louvers on the underside of the interface terminal(s).

Environmental requirements are the same as for any microprocessor system: a reasonably open, air conditioned area. Air temperature should not exceed 80 degrees Farenheit; humidity, 80 percent.

2.4 POWER SUPPLY

The TM990/518 power supply unit is plugged into a standard AC wall

outlet. The power connect cable connects the power supply unit to the EVM (bottom) board power buses located in the center of the right margin of the prototyping area, and the EMU (top) board power buses located in the center of the right margin of the emulator board, as shown in Figure 2-1. The connections on the EVM and Emulator ends of the power supply cable are positively keyed to prohibit misconnection to the power supply. Care should be exercised to insure proper connection at the power supply end (See 2.6, below).

2.5 UNPACKING

Lift the TMAM 6083 Emulator from its carton and remove the protective wrapping. Check for shipping damage; if any is found, notify your TI distributor. Verify that the following components are included:

- Two-board TMAM 6083 Emulator
- Target connector assembly
- Two target connector/emulator interconnect cables
- Power connector cable

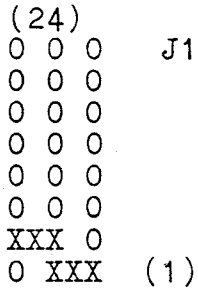
Insure that all jumpers are in correct positions as follows:

ON THE EVM BOARD:
J1 joins location 2 and 3
J2 joins location 2 and 3
J3 joins location 2 and 3
J4 joins location 1 and 2
J5 joins location 1 and 2
J6 joins location 1 and 2
J7 joins location 2 and 3 (Auto First Wait State ON)
J8 joins location 2 and 3

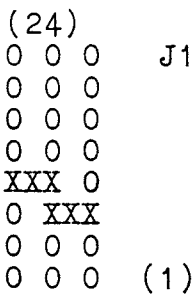
ON THE EMU BOARD:
J1 joins location 1 and 2
J2 joins location 1 and 2
J3 joins 1 and 2
J4 joins 1 and 2
J5 joins location 1 and 2
J6 joins location 2 and 3
J7 joins location 2 and 3
J8 joins location 2 and 3
J9 not used

Jumper J10 is used in conjunction with the memory decoding PROM (U43) to map the 8K bytes of emulator expansion memory into the desired address space. The decoding PROM (U43) provided with the emulator allows the user to select one of four separate memory maps by simply changing the jumper positions of J10.

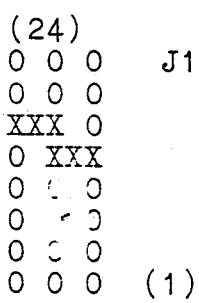
J10 consists of eight rows with three pins in each row. In order to select a particular memory map the jumpers should be positioned as shown below:



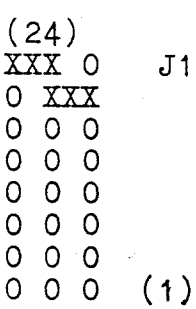
J10 JUMPER CONFIGURATION NUMBER 1 FOR 4K BYTES OF RAM MAPPED AT LOCATION >0000 AND 4K BYTES OF RAM MAPPED AT LOCATION >1000



J10 JUMPER CONFIGURATION NUMBER 2 FOR 4K BYTES OF RAM MAPPED AT LOCATION >0000 AND 4K BYTES OF RAM MAPPED AT LOCATION >8000



J10 JUMPER CONFIGURATION NUMBER 3 FOR 4K BYTES OF RAM MAPPED AT LOCATION >0000 AND 4K BYTES OF RAM MAPPED AT LOCATION >E000



J10 JUMPER CONFIGURATION NUMBER 4 FOR 4K BYTES OF MEMORY MAPPED AT LOCATION >D000 AND 4K BYTES OF MEMORY MAPPED AT LOCATION >E000

See Section 4.3.1 for details concerning how to change the memory map if one different from those given above is needed.

2.6 HOOKUP

1. Attach power-connect cable to both boards of the Emulator, as shown in Figure 2-1.

   NOTE: If using a power supply other than the TM990/518, remove the 1/4 inch Faston terminals from the cable and attach the proper connector or plugs for the power supply being used. The power cable conductors are color-coded as follows:

+5V	Red
+12V	White
-12V	Green
Ground	Black

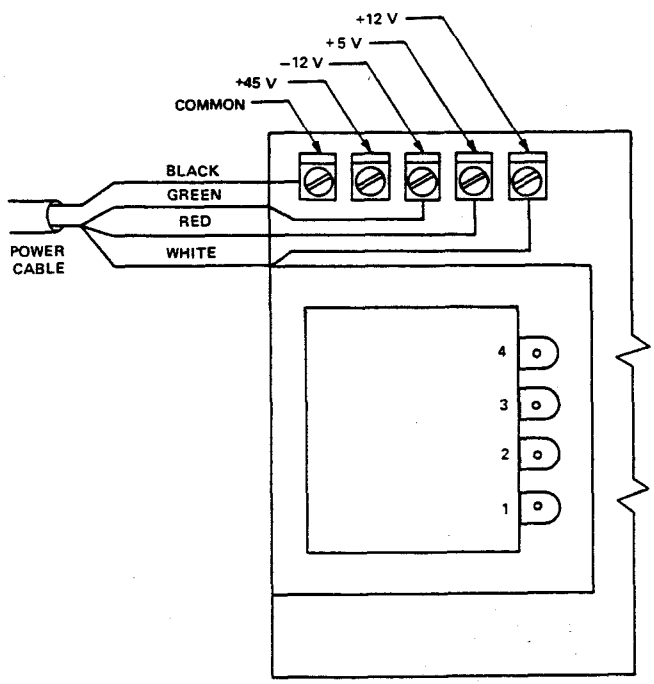


FIGURE 2-1. POWER CABLE CONNECTION TO THE TM990/518 POWER SUPPLY UNIT.

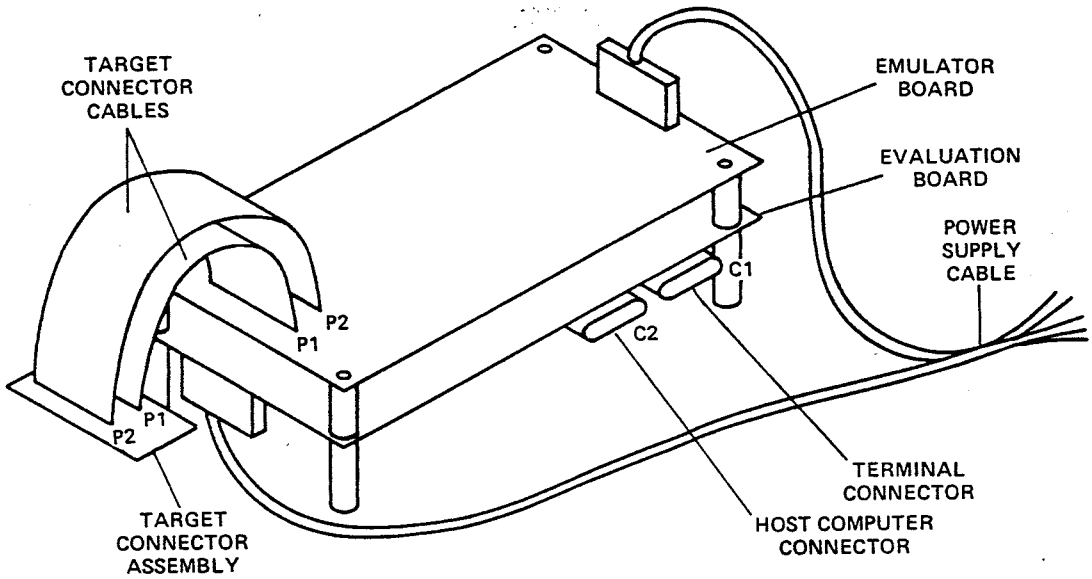


FIGURE 2-2. TMAM 6083 EMULATOR POWER SUPPLY AND INTERCONNECT CABLE CONNECTIONS.

2. Connect target connector assembly to the Emulator as follows:

   - Target connector cable from P1 on the EMU board to P1 on the target connector assembly.

   - Target connector cable from P2 on the EMU board to P2 on the target connector assembly.

3. Connect the terminal cable to Port 1 - EVM board. The following signals must be present:

Pin Number	
2	Serial Data In
3	Serial Data Out
~~8~~20	Data ~~Set~~ TERMINAL Ready
7	Ground

IF DTR IS TO BE USED OR IS UNAVAILABLE Pin 20 should be jumpered to +12V (Pin 6)

If DSR is not to be used or is unavailable, pin 6 should be jumpered to ground (pin 7).

4. In order to utilize the download option, connect a host computer to Port 2 - EVM board. See Section 7 of the EVM User's Guide for information on the EIA communications link. In addition to the cable connection shown in the EVM manual, the Data ~~Set~~ TERMINAL Ready (D~~S~~R) signal must be provided (pin2~~0~~ to pin2~~0~~).

5. Plug the power supply unit line into any properly grounded AC wall outlet.

> CAUTION: Be extremely careful to apply correct voltage levels to the module. Texas Instruments assumes no responsibility for damage caused by improper wiring or voltage applications by the user.

2.7 INITIAL SYSTEM CHECKOUT

This section contains a system check-out procedure to verify that the Emulator is operational.

2.7.1 Verification

Verify that all jumpers are in the correct positions and that the following conditions apply:

- Power is connected to correct pins.
- All board connections as described in paragraph 2.6 are correct.
- The baud rate and communications mode are correctly set at the terminal and that the terminal is ON LINE. (NOTE: baud rate should be set at 2400 or lower.)

2.7.2 Power Up/Reset

1. Apply power to the Emulator and the terminal.
2. Activate the RESET Switch on the EVM.
3. Press the 'A' key on the terminal. EMUBUG measures the time of the start bit and determines the baud rate. To account for different terminals used, a carriage return time of 200 ms is provided for all baud rates at or slower than 1200 baud.
4. EMUBUG prints the message:

```
9995 EMULATOR 16013596  REV 1.n (n = version number)

MON ?
```

This is a request to input a command to the scanner. Commands are descrbed in detail in Section 3 of this manual. The Instruction Set for the EVM assembler is defined in Section 6 of the EVM User's Guide.

2.7.3 Sample Program

The following sample program may be used to test the Emulator:

```
9995 EMULATOR 1603596 REV 1.0

MON? XA 80
0080 0300      LIMI 0                          MASK ALL INTERRUPTS
0082 0000
0084 02E0      LWPI WP                         LOAD WORKSPACE POINTER
0086R0000
0088 04C0      CLR R0                          INITIALIZE OLDEST VALUE
008A 0201      LI R1,1                         INITIALIZE MOST RECENT VALUE
008C 0001
008E A001 LP   A R1,R0                         COMPUTE NEXT VALUE
0090R18FF      JOC OVR                         IF CARRY, THEN OVERFLOW OCCURRED
0092 C080      MOV R0,R2                       SAVE NEXT VALUE
0094 C001      MOV R1,R0                       UPDATE OLDEST VALUE
0096 C042      MOV R2,R1                       UPDATE MOST RECENT VALUE
0098 10FA      JMP LP                          VALUE TOO LARGE, STOP
009A 0340 OVR IDLE
0090*1804
009C 10FE      JMP OVR                         CONTINUE
009E       WP  EQU $
0086*009E
009E           END   0000
MON? XRA 80 9C
0080 0300 LIMI >0000
0084 02E0 LWPI >009E
0088 04C0 CLR  R0
008A 0201 LI   R1,>0001
008E A001 A    R1,R0
0090 1804 JOC  >009A
0092 C080 MOV  R0,R2
0094 C001 MOV  R1,R0
0096 C042 MOV  R2,R1
0098 10FA JMP  >008E
009A 0340 IDLE
009C 10FE JMP  >009A
MON? IR
W=BFD6
P=F084  80
S=0000
MON? BP
MODE (IN,OUT,1,2) = I ? 2
ADDRESS 1 = 0000 ? 92
ADDRESS 2 = 0000 ? 9A
QUAL (XARW) = 0000 ? 9A
MON? EXB
BREAKPOINT ENCOUNTERED AT 0092    009E   0092   C000    C080
MON? IWR

R0=0001  R1=0001  R2=5AFE  R3=DAFE  R4=52F4  R5=A100  R6=78FC  R7=7ABE
R8=50FF  R9=4B0C  RA=A5FE  RB=A5FB  RC=A5FD  RD=5B0C  RE=E5FA  RF=AFFB
MON? EXB
BREAKPOINT ENCOUNTERED AT 0092    009E   0092   C000    C080
MON? IWR
R0=0002
```

```
MON?EXB
BREAKPOINT ENCOUNTERED AT 0092    009E   0092   C000    C080
MON? IWR
R0=0003
MON? EXB
BREAKPOINT ECOUNTERED AT  0092    009E   0092  C000    C080
MON? IWR
R0=0005
MON? BP
MODE (IN,OUT,1,2) = 2 ? 1
ADDRESS 1 = 0092 ? 9A
QUAL (XARW) = 1000 ?
MON? TR
MODE (IN,OUT,1,2) = I
ADDRESS 1 = 0000 ? A2
ADDRESS 2 = 0000 ? FFFF
QUAL (XARW = 0000 ? 0001
BP ON FULL TRACE = NO ? Y
TRACE COUNT LIMIT = 01 ? 20
MON? EXB
TRACE BUFFER FULL    009E   0096   C000    C042
MON? DT 20
0020    00A2  00    0005
001F    00A3  05
001E    00A2  00    0008
001D    00A3  08
001C    00A2  00    000D
001B    00A3  0D
001A    00A2  00    0015
0019    00A3  15
0018    00A2  00    0022
0017    00A3  22
0016    00A2  00    0037
0015    00A3  37
0014    00A2  00    0059
0013    00A3  59
0012    00A2  00    0090
0011    00A3  90
0010    00A2  00    00E9
000F    00A3  E9
000E    00A2  01    0179
000D    00A3  79
000C    00A2  02    0262
000B    00A3  62
000A    00A2  03    03DB
0009    00A3  DB
0008    00A2  06    063D
0008    00A3  3D
0006    00A2  0A    0A18
0005    00A3  18
0004    00A2  10    1055
0003    00A3  55
0002    00A2  1A    1A6D
0001    00A3  6D
```

```
MON? TR
MODE (IN,OUT,1,2) = I ?
ADDRESS 1 = 00A2 ?
ADDRESS 2 = FFFF ?
QUAL (XARW) = 0001 ?
BP ON FULL TRACE = YES ? N
MON? EXB
BREAKPOINT ENCOUNTERED AT  009A    009E    009A    D000    0340
MON? DT 10
0010    00A2  06    063D
000F    00A3  3D
000E    00A2  0A    0A18
000D    00A3  18
000C    00A2  10    1055
000B    00A3  55
000A    00A2  1A    1A6D
0009    00A3  6D
0008    00A2  2A    2AC2
0007    00A3  C2
0006    00A2  45    452F
0005    00A3  2F
0004    00A2  6F    6FF1
0003    00A3  F1
0002    00A2  B5    B520
0001    00A3  20
MON? IR
W=009E
P=009A
S=D000
MON?
```

SECTION 3

EMUBUG INTERACTIVE DEBUG MONITOR

3.1 GENERAL

This section provides a description of the commands and subroutines available in the TMAM 6083 Emulator Debug Monitor (EMUBUG), including syntax conventions and EMUBUG error messages.

EMUBUG is a debug monitor which provides the interactive interface between the user and the TMAM 6083 Emulator. It is contained in three 2532-35 EPROMs. (NOTE: For the purpose of the EMUBUG monitor, all references to address space, memory contents, and CRU addresses specifically refer to user memory and CRU in which the Emulator operates.)

3.2 USER MEMORY

User memory is provided on the Emulator as "expansion RAM" (ERAM). Figure 3-1 illustrates the memory map.

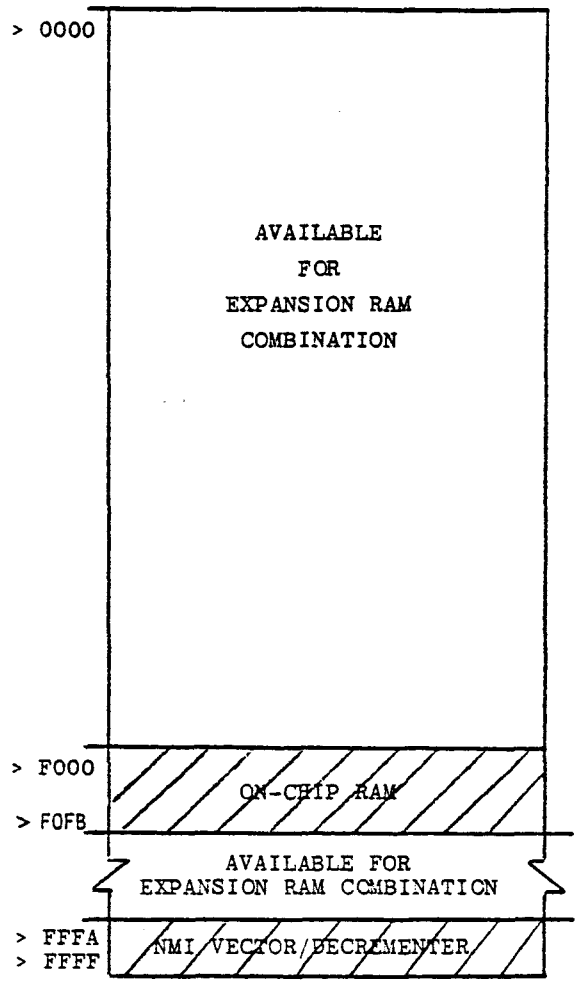


FIGURE 3-1. SYSTEM MEMORY MAP

The memory maps available to the user can be organized into one of four combinations, each consisting of two 4K combinations:

COMBINATION	ADDRESS LOCATION
0 and 4K	>0000 - >1FFF
0 and 32K	>0000 - >0FFF and >8000 - >8FFF
0 and 56K	>0000 - >0FFF and >E000 - >EFFF
52K and 56K	>D000 - >EFFF

3.3 EMUBUG COMMANDS

The Emulator software maintains the same user interface as EVMBUG in the TMS9995 Evaluation Module. The following commands are identical to the corresponding EVMBUG commands, except that they now deal with target memory:

```
IM   (Inspect/Change Memory)
DM   (Dump Memory)
FD   (Find Data In Memory)
IC   (Inspect/Change CRU)
IR   (Inspect/Change Hardware Register: WP/PC/ST)
EX   (Execute User Program)
XA   (Execute Assembler With New Symbol Table)
IWR  (Inspect/Change User Workspace Registers)
LMC  (Load Memory From Digital Cassette)
DMC  (Dump Memory To Digital Cassette)
HEX  (Hex Arithmetic)
TNF  (Toggle Null Flag)
XAE  (Execute Assembler With Existing Symbol Table)
XRA  (Execute Reverse Assembly)
XCL  (Execute Communications Link)
```

Two of the EVMBUG commands have been modified:

```
EXB  (Execute With Breakpoints and Trace)
SS   (Execute In Single-Step Mode)
```

Ten new commands have been added which are unique to the Emulator:

```
BP   (Set Breakpoint)
TR   (Set Trace)
HLT  (Halt)
DT   (DumpTrace)
STA  (Status)
RST  (Enable Target Reset)
NMI  (Enable Non-Maskable Interrupt)
AFW  (automatic First Wait State)
RDY  (Enable Target READY Control of ERAM)
OP   (Display Options)
```

The EMUBUG commands are described in the following paragraphs . Table 3-1 presents the syntax conventions used in the command definitions.

TABLE 3-1. COMMAND SNYTAX CONVENTIONS.

SYMBOL	EXPLANATION
WP	Current User Workspace Pointer Contents
PC	Current User Program Counter Contents
ST	Current User Status Register Contents
Caps	Other items in capital letters are to be entered literally
< >	Items to be supplied by the user. The term within the angle brackets is a generic term.
[ ]	Optional item. May be included or omitted at the user's discretion.
{ }	One of several optional items shown inside the brackets must be chosen.
(CR)	Carriage Return
/\	Space
(LF)	Line Feed
RO - R15	Registers zero to fifteen

NOTE

Except where otherwise indicated, all numeric output is assumed to be hexidecimal; the last four digits inut will be the value used. Thus, an erroneous numerical input can be corrected simply by making the last four digits the correct value. If fewer than four digits are input, they are right-justified.

3.3.1 EMUBUG Commands Identical To EVMBUG Commands

3.3.1.1 Inspect/Change CRU (IC). This command reads the number of bits specified by "count" (beginning at the specified CRU address), and displays them, right-justified, in a 16-bit hex number. "CRU address" is a 16-bit number stored in Register 12; up to 16 CRU bits may be displayed.

SYNTAX: IC[<^,><CRU address>[<^,><count>]]<(CR)>

The corresponding CRU output bits may be altered following input bit display by keying in desired hex data, right-justified.

NOTE: the effective software CRU address is double the hardware CRU bit address.

EXAMPLES:

(1) Check the contents of the TMS 9995 Flag Register (Flag 0-15).

```
MON? IC 1EE0
1EE0=7FE0
```

(2) Using the CRU, configure the TMS 9995 Decrementer as an Event Counter and start decrementing.

```
MON? IC 1EE0=7FE0
1EE0=003
MON?
```

(3) Check the contents of the MID Flag register on the TMS 9995.

```
MON? IC 1FDA
1FDA=FFFE
MON?
```

3.3.1.2 Dump Memory (DM). Invoking DM causes memory to be displayed, beginning and ending at the start and stop addresses respectively, if specified. ~~If no addresses are given, EMUBUG displays the contents of location >0000 and then returns control to EMUBUG.~~

SYNTAX: ~~DM [<start address>[<^,><stop address>]]~~
DM [<^,><START ADDRESS>[<^,><STOP ADDRESS>]]<(CR)>

Each line of output begins with the address of the first memory word displayed on the line. Eight memory words follow on each line.

If a start address, but no stop address is supplied, all memory locations from the start address to the end of memory will be output before control returns to EMUBUG.

Memory dump can be terminated at any time by typing any character on the keyboard.

3.3.1.3 Dump Memory To Digtl. Cassette/Paper Tape (DMC). This command causes computer memory to be copied to digital cassette or paper tape.

SYNTAX: DMC[/\,]<start address>[/\,]<stop address>
[/\,]<entry address>]]]<(CR)>

The memory image is stored in non-relocatable 990 object format. (Object record format is explained in Appendix A of the EVM User's Guide.) The block of memory stored begins at start address and ends at stop address. The entry address parameter is for use by the LMC command to initialize the program counter when the memory block is restored from cassette or paper tape to computer memory. Once these parameters are entered, the monitor will display the prompt: IDT . The user then enters an IDT (program identifier) of up to eight charcters, followed by a space or carriage return:

IDT=<program name>]<(CR)>

After the IDT prompt is answered, the monitor will display the prompt: READY Y/N . When the cassette or paper tape punch is ready, answer the prompt with a Y.

See the EVM User's Guide, Section 5, paragraph 3.4.4, for details on preparing an ASR or paper tape punch.

3.3.1.4 Execute (EX). The EX command causes task execution to begin at current values in the Workspace Pointer and Program Counter.

SYNTAX: EX<^,(CR)>

Execution, once started, will continue until interrupted by the HLT command. ~~upon encountering a breakpoint, or by software return to the monitor normally provided at the end of a user program.~~ Breakpoint + trace are both disabled when the processor is released with this command.

3.3.1.5 Find Data (FD). The contents of memory locations from start address to stop address are compared to value for equality. The memory addresses whose contents equal value are printed out.

SYNTAX: FD[/\,]<start address>[/\,]<stop address>[/\,]<value>]]]<(CR)>

If the termination character of <value> is a minus sign, the search will print the addresses of all bytes from start address to stop address whose contents are the rightmost byte in <value>. If the temination character is a carriage return (CR), then the search will print the addresses of all words from start address to stop address whose contents are equal to <value>.

EXAMPLES:

```
MON? FD E00,EF0 400
E040
0E74

(or)
MON? FD E00,EF0 4-
0E01
0E12
0E30
0E32
0E40
0E5A
0E5C
0E74
0EBC
0EC6
0ED2
0ED6
0EEC
MON?
```

3.3.1.6 Hexadecimal Arithmetic HEX). The sum and difference of two hexadecimal numbers are output.

SYNTAX: HEX[/\,]<number 1>[/\,]<number 2>]<(CR)>

EXAMPLE:

```
MON? HEX 200,100
H1+H2=0300 H1-H2=0100
MON?
```

3.3.1.7 Load Memory From Cassette Or Paper Tape (LMC). Data in 990 object record format is loaded from paper tape or cassette into memory. <Bias> is the relocation bias (starting address in RAM). Its default is >0. However, object code saved using the DMC command is restored using the relocation bias (starting address) specified for that command. Both relocatable and absolute data may be loaded into memory with the LMC command. After data is loaded, the module identifier (See Appendix A, Tag 0, in EVM User's Guide) is printed on the next line.

SYNTAX: LMC[/\,]<bias>]<(CR)>

3.3.1.8 Inspect/Change Memory (IM). Memory Inspect/Change opens a memory location, displays it, and gives the option of changing the data in the location.

SYNTAX: IM[/\,]<start address>]<(CR)>

The IC memory address directs a display of memory contents from the

start address each time the space bar is pressed. Each line of output consists of the address of the data word, followed by the data word itself. A termination character causes the following:

(1) If a carriage return, control is returned to the command scanner.

(2) If a space, the next memory location is opened and displayed.

(3) If a minus sign (-), the previous location is opened and displayed.

If a hexadecimal value is entered before the termination character, the displayed memory location is updated to the value entered.

EXAMPLE:

```
ED00=02E0
ED02=EEA4
ED04=0200
ED06=000A
MON?
```

3.3.1.9 Inspect/Change User WP/PC/ST/Registers (IR)> The user Workspace Pointer (WP), Program Counter (PC), and Status Register (ST) are inspected and changed with the IR command.

SYNTAX: IR<^,(CR)}

The output letters WP/PC/ST identify the values of the three hardware registers passed to the TMS9995 microcomputer when an EXB, EX, or SS command is entered. WP points to the workspace register area, PC points to the next instruction to be executed, and ST gives the Status Register contents.

A termination character causes the following:

(1) A carriage return causes control to return to the command scanner.

(2) A space causes the next register to be opened.

The order of display is: WP, PC, ST

EXAMPLES:

(1) MON? IR

```
W=EC16  100
P=02E2  D00
MON?
```

(2) MON? IR

```
W=EC16
P=02E2
S=D600
MON?
```

3.3.1.10 Toggle Null Flag (TNF). The TNF command is used to alert EMUBUG that the terminal being used is a 1200 baud terminal which is not a Texas Instruments' 733 ASR.

SYNTAX: TNF<^,(CR)}

TNF is used ONLY when operating with a true 1200 baud peripheral device. TNF is NEVER used when operating at other baud rates.

To revoke the TNF command, enter it again.

3.3.1.11 Inspect/Change User Workspace Registers IWR). The IWR command is used to display the contents of all workspace registers or to display one register at a time, while allowing the user to change the register contents. The workspace begins at the address in the worspace pointer.

SYNTAX: IWR[</\,}<register number>]<(CR)>

IWR, followed by a carriage return, causes the contents of the entire workspace to be printed. control is then passed to the command scanner.

IWR, followed by a register number in hex and a carriage return, causes display of the specified register's contents. The user may then enter a new value into the register. The follwoing are valid termination characters:

(1) A space causes display of the next register.

(2) A minus sign causes display of the previous register.

(3) A carriage return gives control to the command scanner.

EXAMPLES:

(1) MON? IWR

```
R0=0000  R1=0000  R2=0AF5  R3=0000  R4=4AE9  R5=0A00  R6=0006
R7=EC0E  R8=0001  R9=0142  RA=4AE9  RB=04AC  RC=0000  RD=EC00
RE=0E7A  RF=9000  2
MON?
```

(2)
```
MON? IWR 2
R2=EC6  3456
R3=02E2  100
R4=CA01
R5=EC38  800F
R6=02A3  0
MON?
```

3.3.1.12 Execute Assembler With New Assembler Table (XA). The XA command clears the existing symbol table and allows the user to establish a new symbol table.

SYNTAX: XA[/\,]<assembly address>]<(CR)>

EXAMPLE:

```
MON? XA ED00
ED00
```

See note below

3.3.1.13 Execute Assembler W/Existing Symbol Table (XAE). The XAE command assembles using the existing symbol table.

SYNTAX: XAE[/\,]<assembly address>]<(CR)>

EXAMPLE:

```
MON? XAE ED00
ED00
```

See note below

3.3.1.14 Execute Reverse Assembler (XRA). This command allows the user to inspect any EVM memory location and see the mnemonic representation of its contents. The program recreates a source listing from the object code stored in memory by printing the memory address, memory data, instruction mnemonic, and operands.

SYNTAX: XRA[/\,]<start address>[/\,]<end address>]]<(CR)>

See note 1 on page 3-19

Note Symbols used by the assembler are 2 characters long. Longer symbols may be entered, but they are truncated by the assembler.

EXAMPLE:

```
EMUBUG  R1.0
MON? XRA EDOO EDO4
EDOO F6BB SOCB *R11+,R10
EDO2 9AA3 CB   @>02A3(R3),@>FD7D(R10)
MON?
```

3.3.1.15 Execute Communications Link XCL).

SYNTAX: XCL<^,(CR)}

This command transfers control from the monitor to the EIA communications module. (See Section 7 of the EVM User's Guide.)

3.3.2 MODIFIED COMMANDS

The following EVMBUG commands have been modified:

3.3.2.1 Execute With Breakpoints And Trace (EXB). This command is used to execute instructions until a breakpoint occurs. Program execution begins at the address in the PC (set up by using the IR command). Execution terminates at the address specified in the EXB command, and a banner is output showing the contents of the hardware WP, PC, and ST registers, in that order. It is similar to the EVMBUG EXB command, except that no parameters are accepted. The necessary parameters are entered with the BP and TR commands (See paragraph 3.3.3).

SYNTAX: EXB<^,(CR)}

If a breakpoint is encountered, it is displayed in the same manner as an EVMBUG command, i.e., execution terminates at the address specified and a banner is ouput showing the contents of the hardware WP, PC, and ST registers, in that order.

If no breakpoint is encountered, or if the users wishes to enter another command (such as Halt), the user may press any character and the monitor will be invoked.

EXAMPLE:

```
MON? EXB                          (WP)   (PC)   (ST)   (op code)
BREAKPOINT ENCOUNTERED AT 002A    0000   002A   8000   1000
```

See note on page 3-18 ! & Page 3-19 Note 3

3.3.2.2 Execute Single-step Mode (SS)>
Similar to the corresponding EVMBUG command, except that the system will now accept a step count. If no count is entered, one step is performed.

SYNTAX: SS[<^,}<count>]<(CR)>

See Note 1 on page 3-19

EXAMPLE:

```
MON? SS,8

       (WP)   (PC)   (ST)   (op code)
0001   0000   0020   8000   0200
0002   0000   0024   8000   0201
0003   0000   0038   8000   C440
0004   0000   002A   8000   1000
0005   0000   002C   8000   10FF
0006   0000   002C   8000   10FF
0007   0000   002C   8000   10FF
0008   0000   002C   8000   10FF
```

3.3.3 New Commands

The following commands are unique to the Emulator:

3.3.3.1 Set Breakpoint (BP). This command is used to specify breakpoint(s).

SYNTAX: BP<^,(CR)}

When invoked, the following prompts are displayed:

```
BP

MODE (IN,OUT,1,2)= # ?
ADDRESS 1 = #### ?
ADDRESS 2 = #### ?
QUAL (XARW) = #### ?
```

Where: # represents the last value entered.

To select the default, press the space bar or enter a carriage return.

The response to the Mode command must be an I, O, 1 or 2. These represent in range (inclusive), out of range (exclusive), one unique address, or two unique addresses. The addresses 1 and 2 set the boundaries of the range to be searched, i.e., betweeen Address 1 and Address 2 (I mode), or exclusive of Address 1 thru Address 2 (O mode).

The addresses are specified in the address prompts. The qualifier(s) to be used are selected by entering four characters which are either ones or zeros. A one selects the respective qualifier; a zero disables it. The qualifiers are:

X Execution (Instruction fetch only)

A All (ALL memory references)

See note on Page 3-20 1/2

R Read (ANY memory Read operation)

W Write (ANY memory Write operation)

EXAMPLES:

(1)
```
MON? BP
MODE (IN,OUT,1,2) = 1 ?
ADDRESS 1 = 0000 ? 002A
QUAL (XARW) = 0000 ? 1000
```
COMMENTS: Specifying one breakpoint

(2)
```
MON? BP
MODE (IN,OUT,1,2) = 2 ?
ADDRESS 1 = 0000 ? 002A
ADDRESS 2 = 0000   0201
QUAL (XARW) = 0100
```
COMMENTS: Specifying two breakpoints

See Note on Page 3-18

3.3.3.2 Trace) (TR). When a specified trace event (IAQ, Memory Read/Write, etc.) occurs between two specific addresses, the data is stored in the trace memory, which can be displayed when the user issues a Dump Trace command.

SYNTAX: TR<^, (CR)}

When invoked, the following prompts are displayed:

```
TR

MODE (IN,OUT,1,2)= # ?
ADDRESS 1 = #### ?
ADDRESS 2 = #### ?
QUAL (XARW) = #### ?
BP ON FULL TRACE = ### ?
TRACE COUNT LIMIT = ## ?
```

Where: # represents the last value entered.

To select the default, press the space bar or enter a carriage return.

The response to the Mode command must be an I, O, 1 or 2. These represent in range (inclusive), out of range (exclusive), one unique address, or two unique addresses.

The addresses are specified in the address prompts. The qualifier(s) to be used are selected by entering four characters which are either ones or zeros. A one selects the respective qualifier; a zero disables it. The qualifiers are:

See note on Page 3-22

X Execution (Instruction fetch only)

A All (ALL memory references)

R Read (ANY memory Read operation)

W Write (ANY memory Write operation)

If the full trace prompt is answered with a Y, a breakpoint will occur when the number of trace events specified in the trace count limit prompt have occurred; the number must be 1 through 255 (in Hex).

EXAMPLE:

```
MON? TR
MODE (IN,OUT,1,2) = I ?
ADDRESS 1 = 0000 ?
ADDRESS 2 = FFFF ?
QUAL (XARW) = 0100 ?
BP ON FULL TRACE = YES ? N
TRACE COUNT LIMIT = 01 ? 4
```

3.3.3.3 Halt (HLT) This command halts the emulator and returns control to the monitor. The Emulator must be in a Halt state before any command that utilizes Emulator address space can be executed.

SYNTAX: HLT<^,(CR)>

EXAMPLE:

```
MON? HLT
MON? STA
EMULATOR IS HALTED
```

See note on page 3-18

3.3.3.4 Dump Trace (DT)> This command dumps the contents of the trace memory, from the specified number of trace events up to the most recent event.

SYNTAX: DT[<^,>]<parameter>]<(CR)>

The command accepts one parameter. If no parameter is entered, 255 steps will be output.

See note 12 on page 3-19

EXAMPLE:

```
MON? DT 7
                (contents in
    (address)   bytes)
007     F2FF    FF
006     0E45    40      (op codes)
005     F6FB    FF
004     0022    FF      FF44
```

```
003     0023  44
002     0024  02      0201
001     0025  01
```

3.3.3.5 Status (STA) This command outputs the status of the emulator

SYNTAX: STA<^,(CR)}

EXAMPLE:

```
MON? STA
EMULATOR IS RUNNING
```

3.3.3.6 Enable Target RESET (RST). This command permits the target system to control RESET.

SYNTAX: RST[^/\,}<value>]<(CR)>

If a non-zero value is entered with this command, the reset signal from the target is allowed to cause a reset. If a zero or no value is entered, target resets are ignored.

EXAMPLE:

```
MON? RST 1
MON?
```

3.3.3.7 Enable Non-Maskable Interrupt (NMI).
Triggers the LREX external instruction on the EVM board to perform single-stepping. (See paragraph 4.11 in the EVM User's Guide)

SYNTAX: NMI [<^,}<VALUE>]<(CR)>

3.3.3.8 Automatic FirstWait State (AFW). If a non-zero value is entered with this command, the emulator will run with an Automatic First Wait State.

SYNTAX: AFW [<^,}<VALUE>]<(CR)>

3.3.3.9 Enable Target READY Control Of ERAM (RDY). If a non-zero value is entered with this command, the READY signal from the target system will control the timing for the Emulator's expansion RAM.

SYNTAX: RDY [<^,}<VALUE>]<(CR)>

3.3.3.10 Display Options (OP). Evoking this command will cause a display of the states of the RST, NMI, AFW, and RDY flags.

SYNTAX: OP <^, (CR)>

EXAMPLE:

```
MON? OP

TARGET RESET DISABLED
AUTOMATIC FIRST WAIT STATE DISABLED
TARGET MEMORY CONTROL OF READY LOGIC DISABLED
MON? RST 1
MON? OP
TARGET RESET ENABLED
AUTOMATIC FIRST WAIT STATE DISABLED
TARGET NMI DISABLED
TARGET MEMORY CONTROL OF READY LOGIC DISABLED
```

3.4 ERROR MESSAGES

Error messages have been provided in the EMUBUG monitor. In the event of an error, the word ERROR is output, followed by a single digit indicating the error condition. Table 3-2 describes the possible error conditions.

TABLE 3-2. EMUBUG ERROR MESSAGES.

ERROR	CONDITION
0	INVALID TAG DETECTED BY THE LOADER
1	CHECKSUM ERROR DETECTED BY LOADER
2	INVALID TERMINATION CHARACTER DETECTED
3	NULL INPUT FIELD DETECTED BY DUMP ROUTINE
4	INVALID COMMAND ENTERED

POSSIBLE CAUSES AND CORRECTION PROCEDURES

ERROR 0/1: The program load process is terminated.

If the program is being input from a 733 ASR, possible causes of the error are a faulty cassette tape or dirty READ heads in the tape transport.

If the terminal device is an ASR 33, chaf may be caught in a punched hole in the paper tape.

TO CORRECT: In either case, repeat the load procedure.

ERROR 2: Invalid Termination Character. Command is terminated.

TO CORRECT: Reissue the command and parameters with a valid termination character.

ERROR 3: Incorrect input to DUMP command. Dump command is terminated.

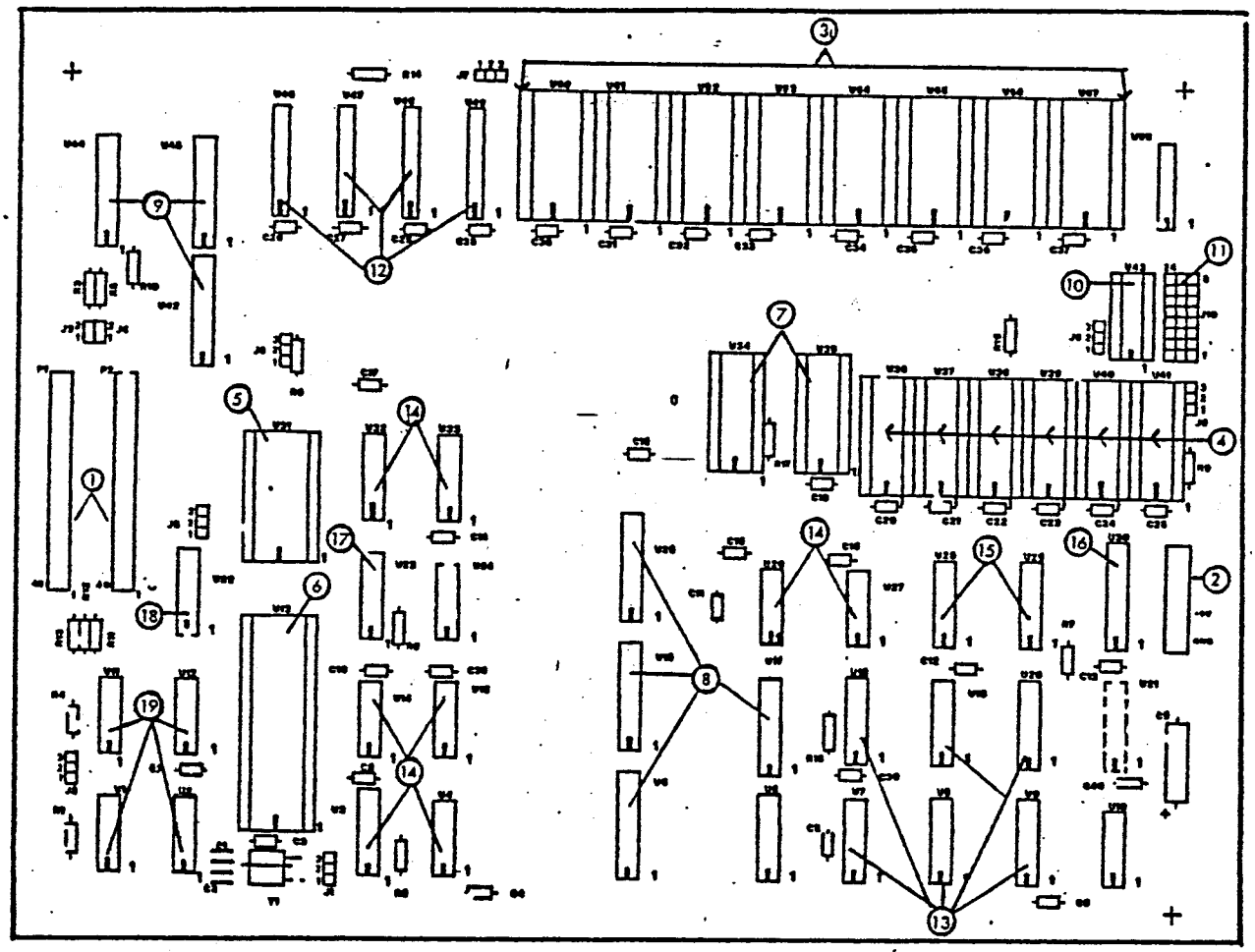
User either input a null field for start address, stop address, or the entry address to to dump routine, and/or,

Ending address is less than the beginning address.

TO CORRECT: Reissue the Dump command and input all necessary parameters.

ERROR 4: Self explanatory.

TO CORRECT: Enter a valid command.



1. Target Connector Cable Sockets (P1, P2)
2. Power Cable Socket
3. User RAM Chips (U50-U57)
4. Trace Memory (U36-U41)
5. Virtual RAM (U31)
6. TMS 9995 Socket (U13)
7. Breakpoint and Trace Address Memories (U34, U35)
8. Interface Buffers TO EVM Board (U5, U16, U17, U25)
9. Interface Buffers To Target System (U42, U44, U45)
10. Mapping PROM (U43)
11. Map-select Jumpers (J10)
12. Breakpoint (U46, U48) and IAQ (U47, U49) Address Registers
13. CRU Decoders (U6-U9, U18-U20)
14. Breakpoint and Trace Decode Logic (U26/U27/U32/U33/U14/U15/U3/U4)
15. Trace State Counters (U28, U29)
16. Address Buffer For Reading Trace Memory (U30)
17. Virtual RAM Control Multiplexor (U23)
18. Target System Buffer (U22)
19. TMS 9995 Ancillary Decode Logic

FIGURE 4-1. TMAM 6083 EMULATOR BOARD.

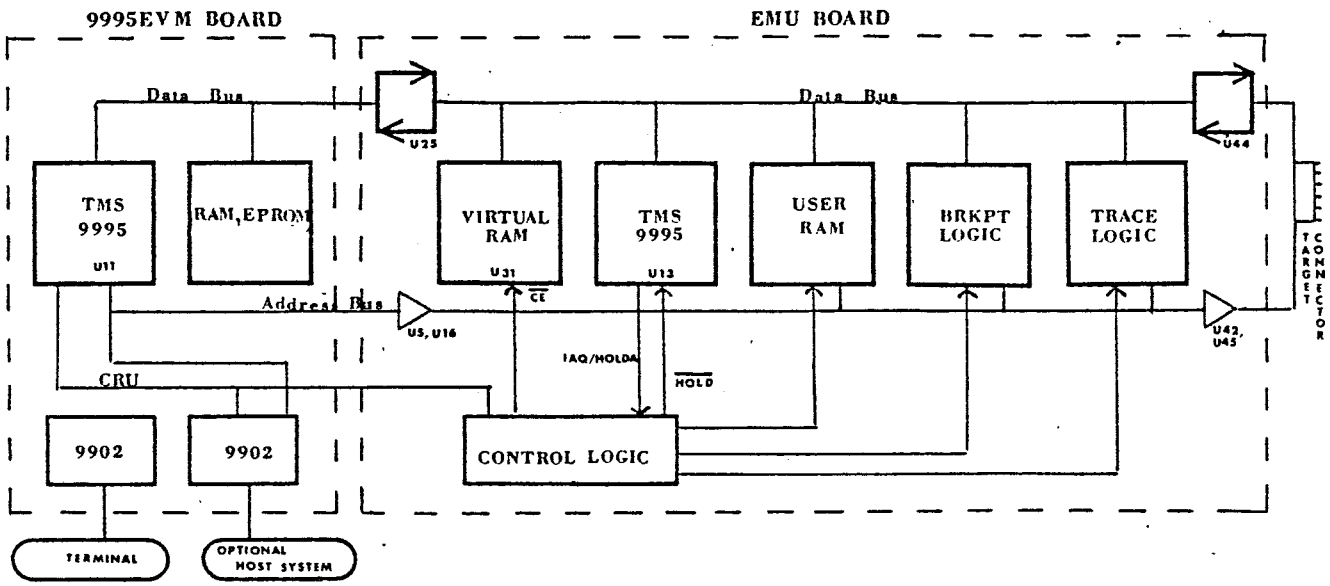


FIGURE 4-2. TMAM 6083 EMULATOR SYSTEM BLOCK DIAGRAM.

4.[illegible] Interface Between The EVM and EMU

Emulation is performed by a TMS 9995 on the EMU board. Control of the Emulator TMS 9995 as well as all other functions of the EMU board is performed by the CRU of the TMS 9995 on the EVM board. Virtual RAM (U31 at 1K bytes) is used to pass information between the EVM and EMU 9995s. When it is necessary to read or set user memory, set breakpoints, or read trace memory, the EVM CRU places the EMU in Hold and loads a small program into the virtual RAM. The EMU then executes the program, performs the required task and places the results in virtual RAM. At this point, a breakpoint set by the EVM is encountered, and the data is read from the virtual RAM by the EVM.

8K bytes of RAM (U50-U57) are available to the user. Jumper (J10) settings allow user memory to be mapped in several configurations within the EMU memory map (See Figure 4-3).

Breakpoint logic on the EMU determines when a specified address is accessed. It is possible to set various types of breakpoints, such as instruction execution, read/write data addresses combined with exact

address, and in-bounds/out-of-bounds addresses.

Trace logic can trace both addresses and data, with the same control qualifications as breakpoints. Trace memory for address is 256 x 16, and for data, a second 256 x 8 memory is used.

Control logic for the EMU is controlled by CRU signals from the EVM. Output signals from the EVM CRU are held in addressable latches. These control functions such as bus directions, Enables and Holds on the EMU.

4.2.2 CRU Control

The EMU board is controlled by the CRU bits on the EVM board. Table 4-2 lists the functions of the 24 CRU bits which output signals from the EVM to the EMU; Table 4-3 lists the functions of the 6 input signals from EMU to EVM.

NOTE: the user can no longer change the CRU map on the EVM; this does not affect the CRU map of the target or Emulator.

TABLE 4-2. MODIFIED EVM CRU MAP: OUTPUTS FROM EVM TO EMU.

CONTROL BASE ADDRESS = <3850. Offsets are relative to this address.

ASSIGNMENT	OFFSET	BIT NO	DEFINITION
NMIINH	15	1	INHIBIT NMI FROM USER 1 = INHIBITED 0 = ENABLED
RDYT	2	2	ENABLE READY FROM OFFBOARD TARGET 1 = READY ENABLED 0 = READY NOT ALLOWED
RDYE	7	3	ENABLE READY FROM EXPANSION RAM 1 = EXPANSION RAM READY 0 = EXPANSION RAM IGNORED
RSTINH	11	4	ENABLE RESET FROM OFFBOARD TARGET 1 = INHIBITED 0 = ENABLED
RSTREQ	6	5	REQUEST RESET FROM EVM 1 = RESET REQUESTED 0 = NO RESET REQUESTED
CLRBPT	12	6	CLEAR BREAKPOINT/INHIBIT BKPT 1 = BREAKPOINTS ENABLED 0 = BKPTS CLEARED/INHIBITED
HLDREQ	8	7	REQUEST EMULATOR TO HOLD 1 = NO HOLD REQUEST MADE 0 = HOLD REQUESTED
IDLFLP	4	8	IDLE FLIP FLOP CLEAR 1 = CLEAR 0 = NO ACTION
XXXXXX	0	9	(UNUSED)
WEDBIN	1	10	ENABLE TARGET WE AND DBIN 1 = ON 0 = OFF (TRISTATE)
HSTMEM	9	11	EVM ACCESS TO MEMORY 1 = MEMORY ACCESSIBLE 0 = NO ACCESS

(Continued)

TABLE 4-2. MODIFIED EVM CRU MAP: OUTPUTS FROM EVM TO EMU (Cont).

CONTROL BASE ADDRESS = <3850. Offsets are relative to this address.

ASSIGNMENT	OFFSET	BIT NO	DEFINITION
VRMENA	23	12	VIRTUAL RAM ENABLE 1 = VRAM ON 0 = VRAM OFF
VRAMSL	13	13	VIRTUAL RAM MODE SELECT 1 = HOST 0 = EMULATOR
MEMRED	10	14	MISC. MEMORY READ ENABLE 1 = ENABLE 0 = DISABLE
MEMWRT	22	15	MISC. MEMORY WRITE ENABLE 1 = ENABLE 0 = DISABLE
TARGET	5	16	TARGET (OFFBD) MEMORY ENABLE 1 = OFF 0 = ON
BPTRNG	17	17	BREAKPOINT RANGE SELECT 1 = RANGE ENABLED 0 = UNIQUE ADDRESSES
TRCFUL	19	18	TRACE MEMORY FULL BP ENABLE 1 = BP WILL OCCUR 0 = BP WILL NOT OCCUR
TRCCLR	14	19	CLEAR TRACE COUNTER 1 = CLEAR 0 = DO NOT CLEAR
TRCRNG	21	20	TRACE RANGE SELECT 1 = RANGE ENABLED 0 = UNIQUE ADDRESSES
TRCDIR	3	21	TRACE MEMORY MODE 1 = READ TRACE 0 = WRITE TRACE
MSCAD0	18	22	MSB OF MISC. MEMORY SELECT
MSCAD1	20	23	MID OF MISC. MEMORY SELECT
MSCAD2	16	24	LSB OF MISC. MEMORY SELECT

NOTE: The I/O ports retain the same CRU addresses, i.e., >0000 and >400.

TABLE 4-3. MODIFIED EVM CRU MAP: INPUTS FROM EMU TO EVM.

CONTROL BASE ADDRESS = >3800. Offsets are relative to this address.

ASSIGNMENT	OFFSET	BIT #	DEFINITION
TGTVCC	2+CORRCT	1	TARGET VCC PRESENT
BPREQ	1+CORRCT	2	BREAKPOINT REQUEST (SEE NOTE)
		3	UNUSED
FULTRC	3+CORRCT	4	TRACE FULL FLAG
BPRSNT	4+CORRCT	5	BREAKPOINT PRESENT (SEE NOTE)
HLDACK	5+CORRCT	6	HOLD ACKNOWLEDGEMENT

NOTE: THE RELATIONSHIP BETWEEN BPREQ AND BPRSNT IS AS FOLLOWS:

BPREQ: indicates that a breakpoint is being requested by the breakpoint hardware.

BPRSNT: indicates that one IAQ has occurred since BPREQ appeared. This may fail if an IDLE instruction was executed after the breakpoint operation. A failure such as this would occur if breakpoint was set to occur after all memory writes and the following code was executed:

```
MOV  R0,R1        (BP occurs on Write to R1)
IDLE              (Prefetched before Write)
NOP               (Will not be fetched)
```

4.3 BREAKPOINT OPERATION

The breakpoint logic is very flexible , allowing for many variations. Two 4x4 register files (U32, U33) are used to hold breakpoint configuring data. They are written to in the same manner as any normal memory device. Outputs from U32 and U33 are used to determine when breakpoints or traces will occur. The Read addresses are generated by WE, DBIN, IAQ, and MEMEN.

The Compare function is performed by two 256x4 RAMs (U34, U35). The RAMs are loaded with the address locations desired. Input lines connected to the Address Bus provide the current address of the EMU

for comparison. Output from U34 and U35 is fed to gates U27 and U15 to determine whether or not to set a breakpoint. If a breakpoint is desired, Flip-flop U3 is set. (NOTE: breakpoints occur only on IAQs. The breakpoint signal is synchronized on the IAQ signal.) Triggering the flip-flop for a breakpoint places the EMU in a Hold condition. The EVM detects this condition by reading U9, which contains status information.

When the breakpoint is requested, the current address is latched into two octal flip-flops (U46, U48), and the address of the last instruction to be executed is latched into U47 and U49.

4.4 TRACE OPERATION

Much of the trace logic is shared with the breakpoint logic. In addition, two register files, U32 and U33, also hold information on what to trace. RAMs U34 and U35 are used to determine trace addresses. Output from U34 and U35 is gated through U27 and U15 to determine when to store trace information.

Both data and addresses are held in the trace memory (U37, U38, U40, U41), 256 x 4 RAMs. This allows tracing of 256 sixteen-bit addresses. The two RAMs, U36 and U39, are used to trace the data, permiting tracing of 256 eight-bit data values. Addresses for the trace memory are generated by two counters, U28 and U29. When the trace memory becomes full, the Emulator is placed in a Hold condition. To read the trace memory, outputs from the RAM chips are placed on the data bus and the desired address is loaded into the counter by the EVM.

4.5 USER MEMORY

User memory supplied on the EMU consists of 8 2K by 1 RAM chips, U50 - U57). Memory mapping on the EMU is highly flexible, permitting the user to select virtually any configuration of target memory needed. A SN745288 PROM (U43) and a set of jumpers (J10) are used to configure user memory.

The four basic configurations possible using the PROM provided with the board are given in Section 2. If a different mapping configuration is desired, the decode PROM in U43 must be replaced. Figure 4-3 illustrates the memory map shipped with the EMU board.

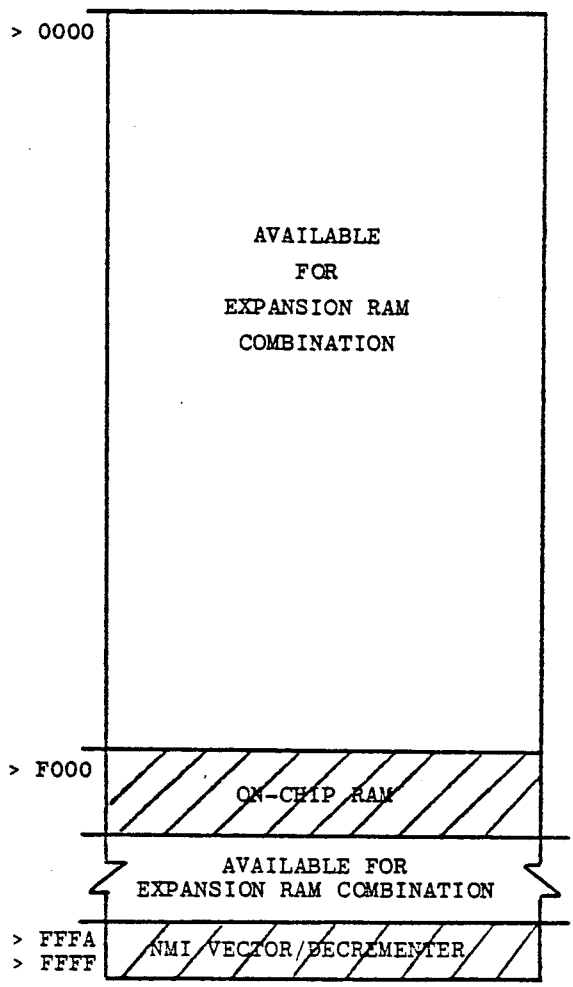


FIGURE 4-3. EMULATOR MEMORY MAP.

4.5.1 Memory Map Changes

In order to run the emulator with a memory map different from one of the four shown in Section 2, it is necessary to program a new decoding PROM for U43 (a 74S288 Programmable ROM). The contents of the decoding PROM shipped with the emulator follow:

PROM ADDRESS	PROM DATA
	QQQQQQQQ
EDCBA	87654321
00000	00010101
00001	00000011
00010	00000000
00011	00000000
00100	00000000
00101	00000000
00110	00000000
00111	00000000
01000	00001100
01001	00000000
01010	00000000
01011	00000000
01100	00000000
01101	01000000
01110	11110000
01111	00000000
10000	00000000
10001	00000000
10010	00000000
10011	00000000
10100	00000000
10101	00000000
10110	00000000
10111	00000000
11000	00000000
11001	00000000
11010	00000000
11011	00000000
11100	00000000
11101	00000000
11110	00000000
11111	00000000

To program a new PROM, first identify one PROM bit to signify which 4K address block(s) is to be overlaid with expansion RAM, then identify a second PROM bit to act as a select line to the 74LS138 RAM chip select decoding device (U58).

Of the four basic configurations possible with the PROM supplied:

Configuration 1 is associated with PROM bits Q1 AND Q2.

Configuration 2 is associated with PROM bits Q3 AND Q4.

Configuration 3 is associated with PROM bits Q5 AND Q6.

Configuration 4 is associated with PROM bits Q7 AND Q8.

4.5.2 Reading From Target Memory

The EVM CRU places the EMU in Hold. The EVM inputs a program to EMU virtual RAM. EMU executes the program and copies the routine into U13 (on-chip RAM), then branches on-chip. A breakpoint is encountered and the EMU holds. The EVM sets the EMU memory map into the target map configuration. Another breakpoint is encountered, the EMU program continues and initiates a transfer of data from target memory into on-chip memory. A breakpoint is encountered and the EMU holds while the EVM switches the EMU memory map back to virtual RAM configuration. The EMU program then continues, moving the data from on-chip memory into virtual RAM. A final breakpoint is encountered, halting the EMU. The EVM may then access the data from the EMU's virtual RAM.

4.5.3 Writing To Target Memory

The EVM places the EMU in Hold. The EVM inputs a program to EMU virtual RAM. EMU executes the program and copies the routine into U13 (on-chip RAM), then branches on-chip. A breakpoint is encountered and the EMU holds. The EVM sets the EMU memory map into the target map configuration. Another breakpoint is encountered, the EMU program continues and initiates a transfer of data from on-chip memory into target memory. Data is then written to the target system.

SECTION 5

MODIFICATIONS TO THE TMAM 6095 EVM

5.1 GENERAL

Certain modifications have been made to the EVM board to permit using it in conjunction with the Emulator board. Those changes are as follows:

- A 38-pin connector has been added to the back side of the EVM board. This connector is used to pass all required signals to the emulator section of the system.

- A jumper wire has been added at the line of signal connectors at the left edge of the prototyping area, connecting CRUIN and the connector just above SEL8. The EVM prototype area is no longer available to the user; all those signals have been switched to the EMU board.

- Address decoding PROM U16 has been modified to permit the use of three 2532 only EPROMs (The Personality Plugs U4, U5, U6 are now Type-2 only). The EVM software has been removed and a new software package has been installed in the three EPROMs located at U8, U9, and U10 (EMUBUG).

- Jumpers J2 and J3 have been repositioned from locations 1 and 2 to locations 2 and 3.

- A socket has been installed under the integrated circuit at U19. Pin 2 of the circuit was cut and soldered to Pin 1. SELECT-ENABLE is now a function of DBIN only.

- U14 has been reprogrammed to comply with the new SELECT ADDRESS assignments (the CRU map). A listing of the new CRU map is presented in Section 4, Tables 4-2 and 4-3. The user can no longer change the CRU map assignments.

- The EVM memory map shown in the EVM User's Guide (Figure 4-5) is no longer valid. The map presented in Figure 5-1 following, reflects the modified status of EVM memory.

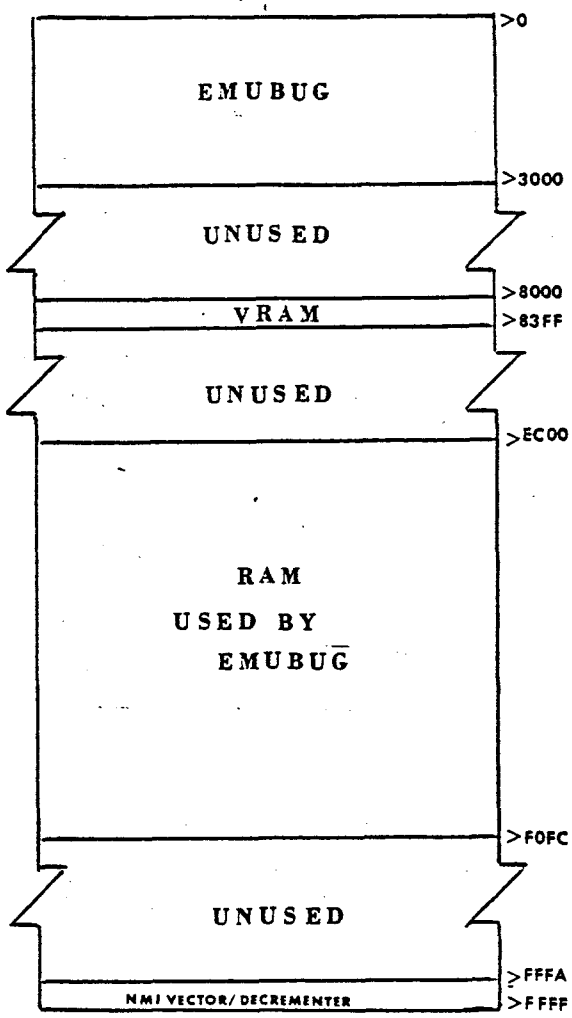


FIGURE 5-1. MODIFIED EVM BOARD MEMORY MAP.

In addition to the above modifications, the Auto First Wait State should be selected ON,i.e., Jumper J7 set at locations 2 and 3.

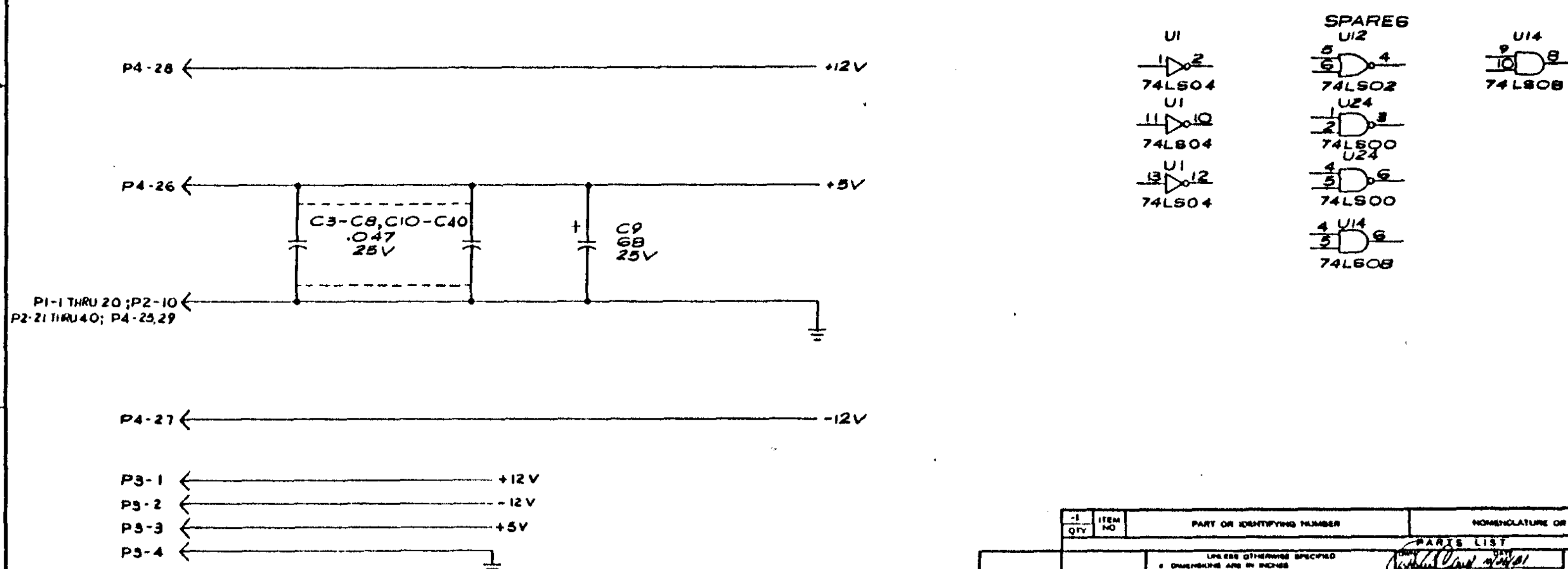
NOTES: UNLESS OTHERWISE SPECIFIED:

1. ALL CAPACITANCE VALUES ARE IN MICROFARADS
2. ALL RESISTANCE VALUES ARE IN OHMS
3. ALL RESISTORS ARE .25W, 5%

JUMPER CONFIGURATION	
JUMPER	LOCATION
J1	1,2
J2	1,2
J3	1,2
J4	1,2
J5	1,2
J6	2,3
J7	2,3
J8	2,3
J10	1,9; 10,17

REFERENCE DESIGNATORS	
USED	NOT USED
C1 - C40	
J1 - J10	J9
P1 - P4	
R1 - R18	
U1 - U58	
Y1	

REVISIONS			
REV	DESCRIPTION	DATE	APPROVED
A	INF ENG/DFTG UPDATE	2-12-82	
B	CN494928	4/20/82	



PARTS LIST

TEXAS INSTRUMENTS

DIAGRAM, LOGIC
TMS9995 EVM-EMULATOR

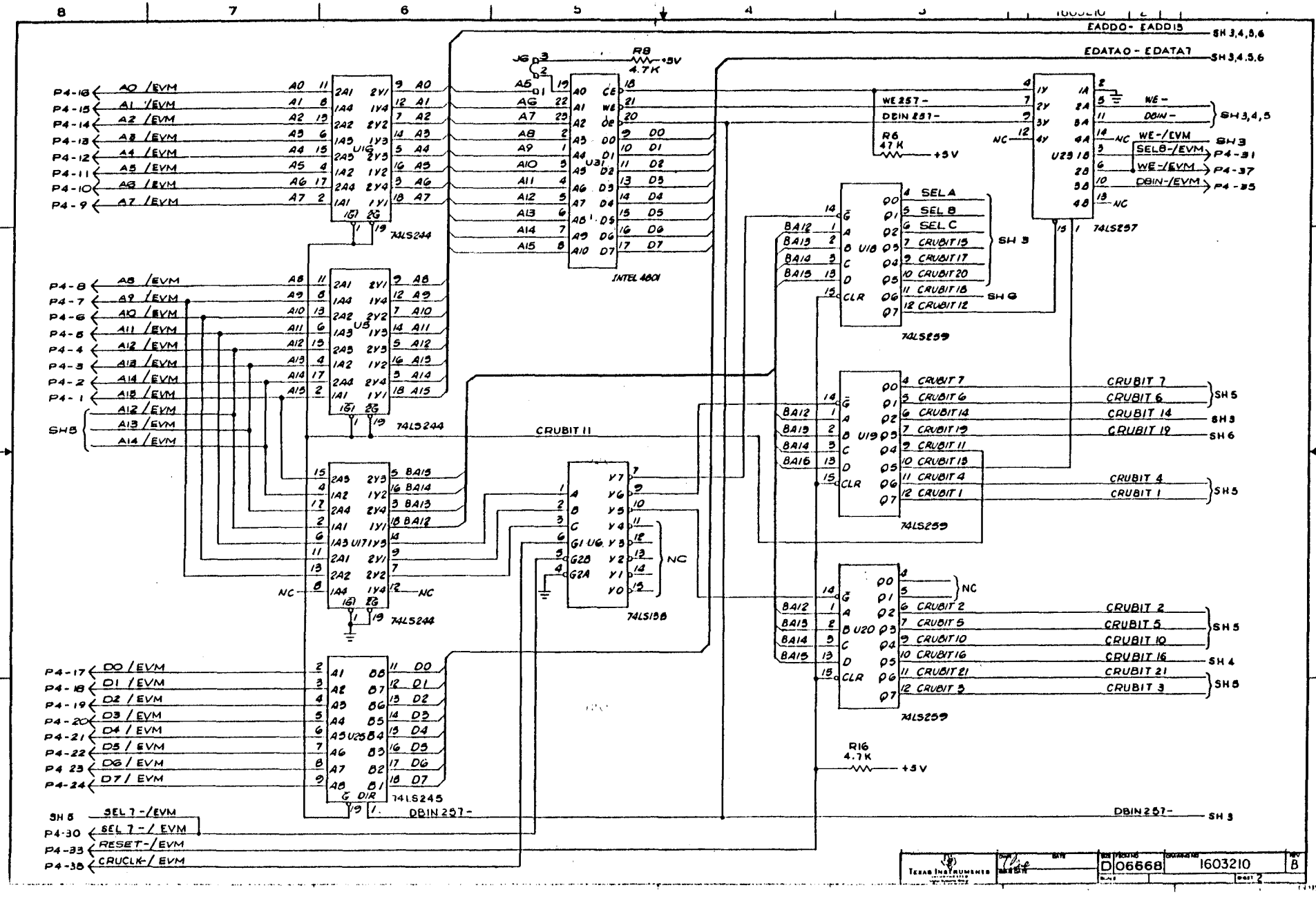
NEXT ASSY	USED ON
1603212	7041
APPLICATION	

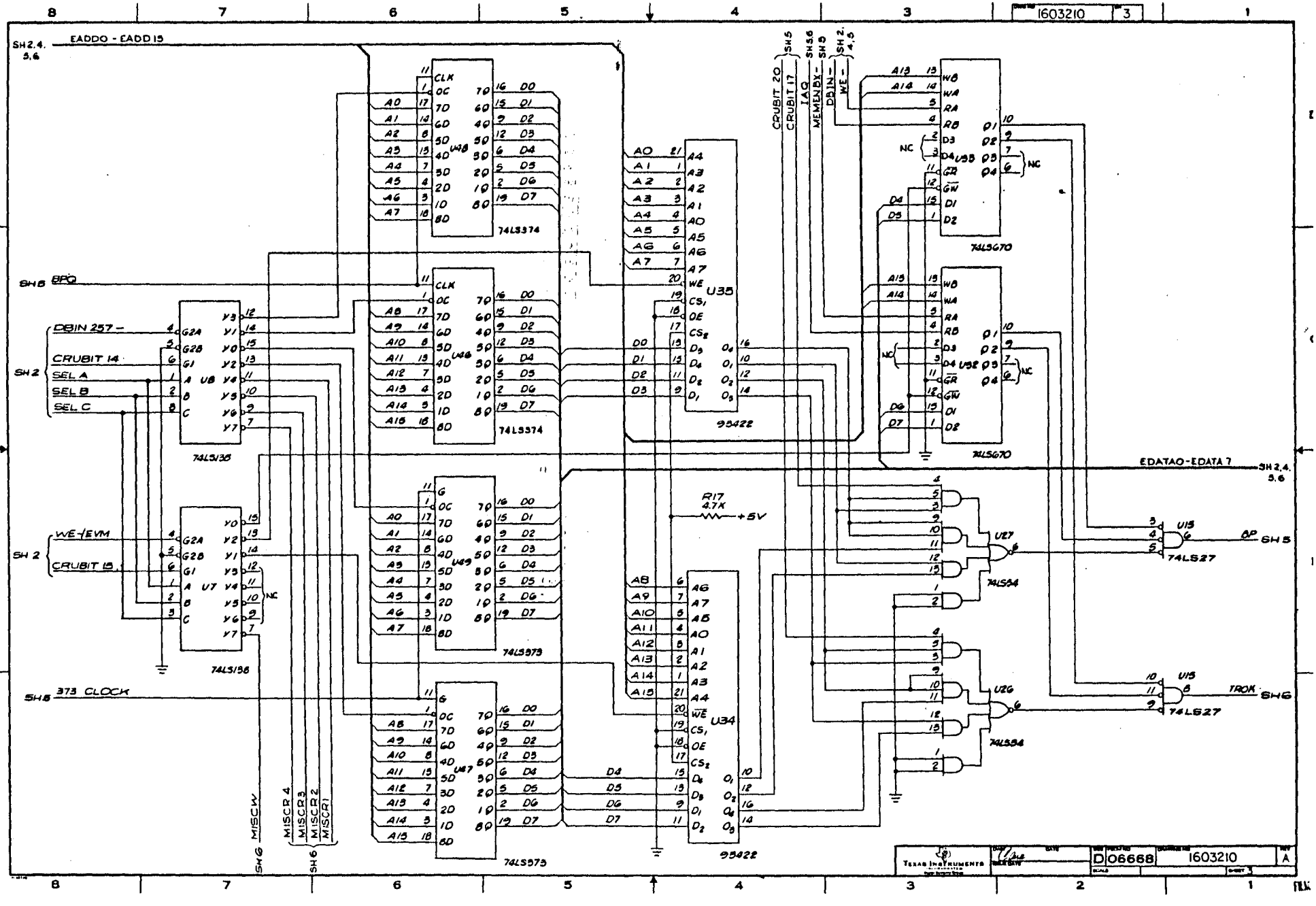
SIZE D FSCM NO 96214 DWG NO 1603210

SCALE NONE SHEET 2 OF 6

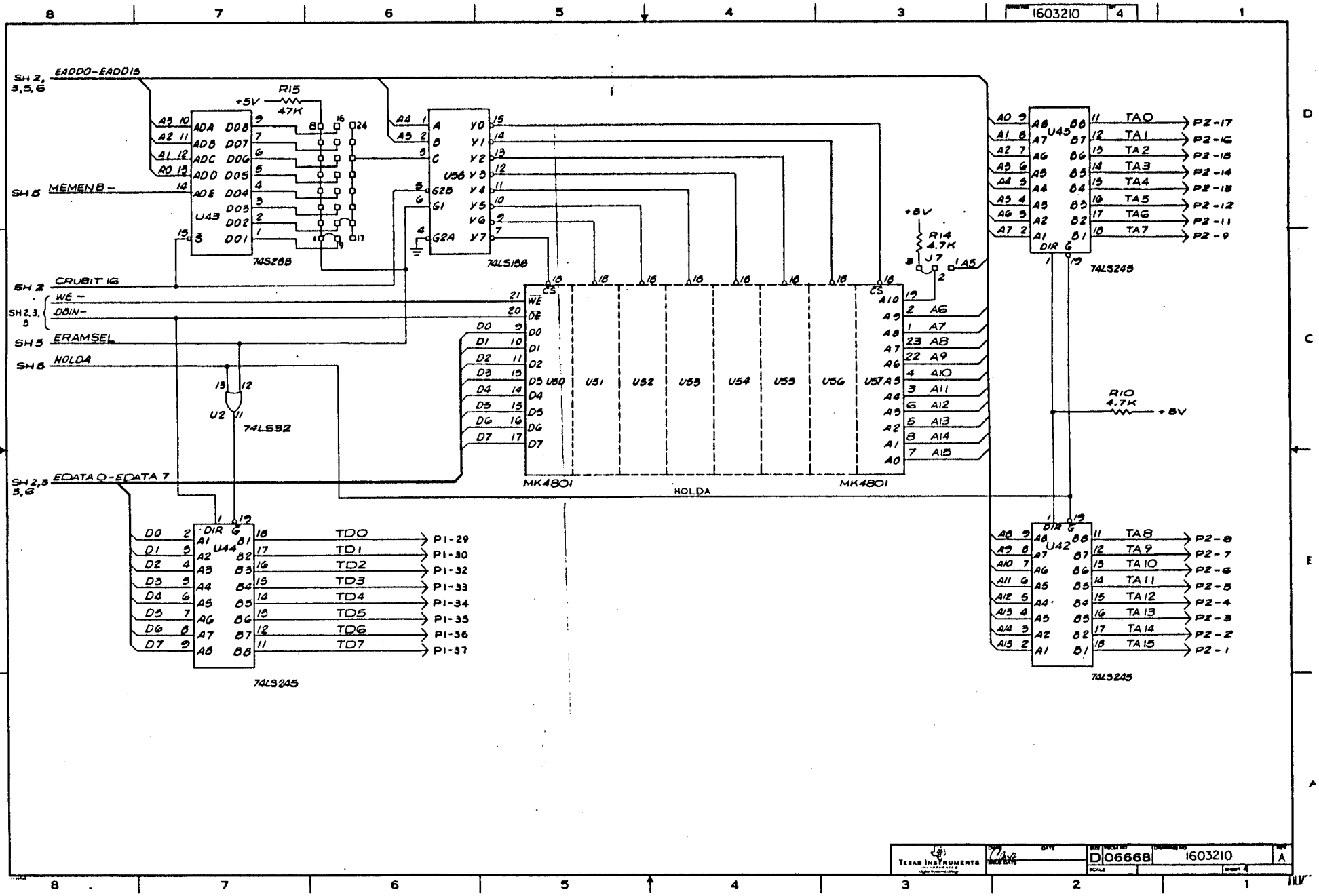
REV STATUS OF SHEETS: REV B B A A A A; SH 1 2 3 4 5 6

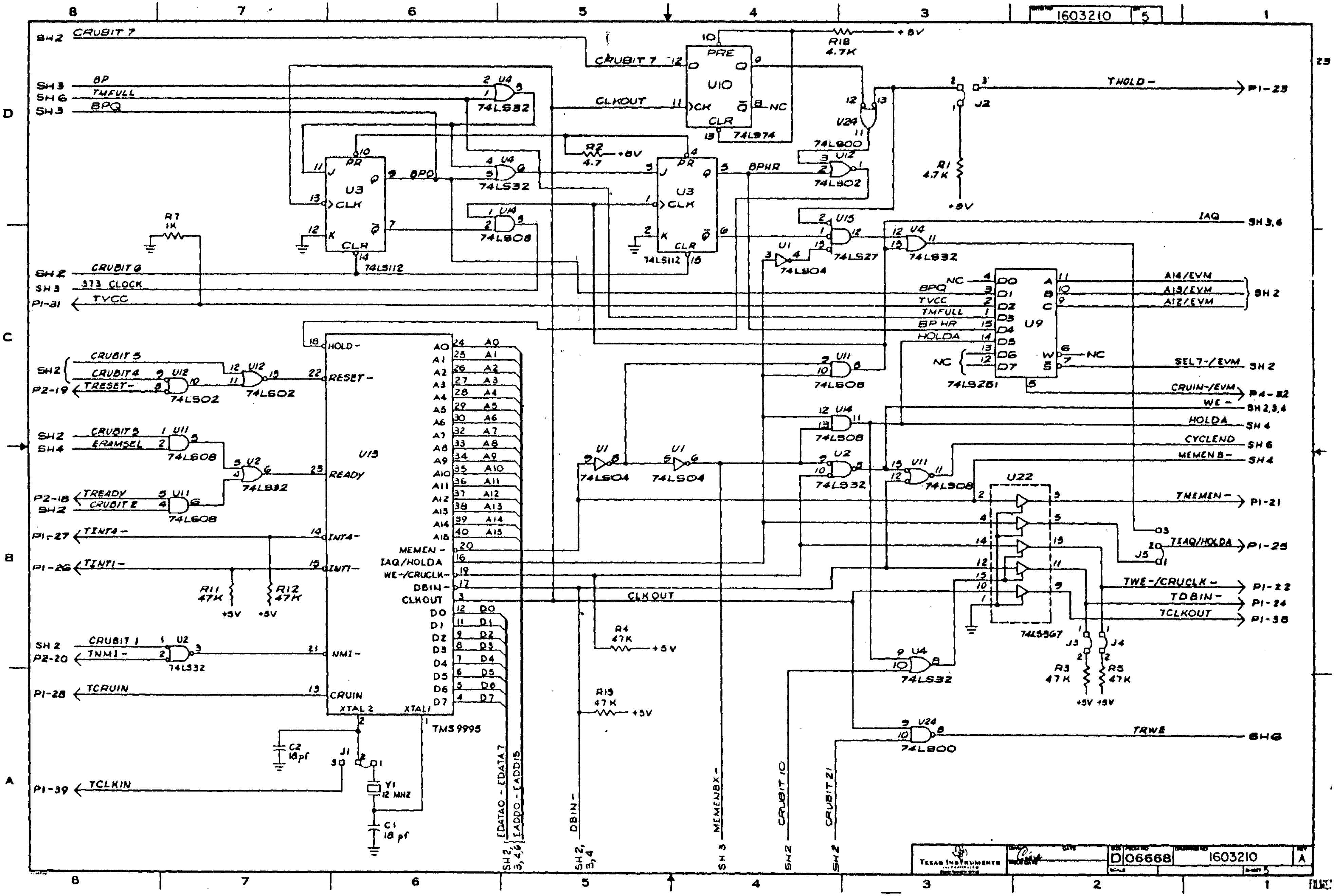
PROCESS CLASSIFICATION


EADD0 - EADD15
SH 3,4,5,6
EDATA0 - EDATA7
SH 3,4,5,6
P4-16 A0 /EVM
P4-15 A1 /EVM
P4-14 A2 /EVM
P4-13 A3 /EVM
P4-12 A4 /EVM
P4-11 A5 /EVM
P4-10 A6 /EVM
P4-9 A7 /EVM
U16
74LS244
P4-8 A8 /EVM
P4-7 A9 /EVM
P4-6 A10 /EVM
P4-5 A11 /EVM
P4-4 A12 /EVM
P4-3 A13 /EVM
P4-2 A14 /EVM
P4-1 A15 /EVM
SH8 A12 /EVM
A13 /EVM
A14 /EVM
U5
74LS244
CRUBIT 11
NC
74LS244
P4-17 D0 /EVM
P4-18 D1 /EVM
P4-19 D2 /EVM
P4-20 D3 /EVM
P4-21 D4 /EVM
P4-22 D5 /EVM
P4 23 D6 /EVM
P4-24 D7 /EVM
U25
74LS245
DBIN 257-
SH 8 SEL 7 -/EVM
P4-30 SEL 7 - / EVM
P4-33 RESET-/EVM
P4-38 CRUCLK-/ EVM
JG
R8 4.7K +5V
U31
INTEL 4801
WE 257 -
DBIN 257-
R6 47K +5V
U23
74LS257
WE -
DBIN -
SH 3,4,5
NC WE-/EVM SH 3
SELB-/EVM P4-31
WE-/EVM P4-37
DBIN-/EVM P4-35
U18
74LS259
SEL A
SEL B
SEL C
CRUBIT 15
CRUBIT 17
CRUBIT 20
CRUBIT 18
CRUBIT 12
SH 3
SH 6
U19
74LS259
CRUBIT 7
CRUBIT 6
CRUBIT 14
CRUBIT 19
CRUBIT 11
CRUBIT 13
CRUBIT 4
CRUBIT 1
SH 5
SH 3
SH 6
U20
74LS259
CRUBIT 2
CRUBIT 5
CRUBIT 10
CRUBIT 16
CRUBIT 21
CRUBIT 3
SH 5
SH 4
SH 5
R16 4.7K +5V
DBIN 257-
SH 3
TEXAS INSTRUMENTS
D 06668
1603210
B
SHEET 2

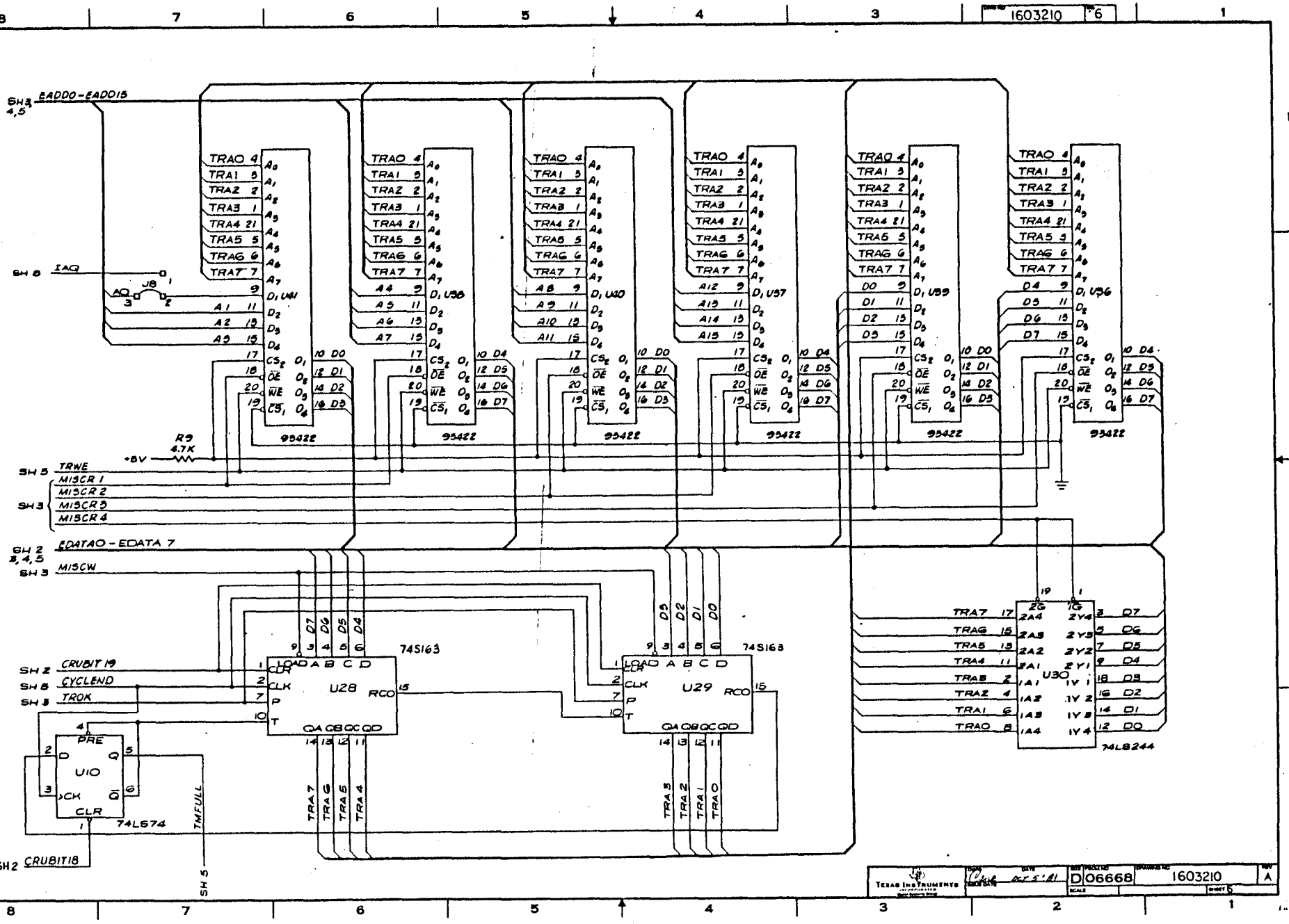

1603210
3
EADDO - EADD15
SH2,4,5,6
U48
U46
U49
U47
74LS374
U35
U33
U32
74LS670
95422
74LS138
U7
U8
74LS373
U34
DBIN 257 -
CRUBIT 14
SEL A
SEL B
SEL C
SH2
SH6 BPQ
WE -/EVM
CRUBIT 15
SH6 373 CLOCK
SH6 MISCW
MISCR 4
MISCR3
MISCR 2
MISCR1
SH6
CRUBIT 20
CRUBIT 17
IAQ
MEMDBX -
DBIN -
WE -
SH5
SH5,6
SH5
SH 2, 4, 5
R17
4.7K
+5V
EDATAO-EDATA 7
SH 2,4, 5,6
U27
U26
U15
74LS27
74LS54
BP
SH 5
TROK
SH6
D06668
1603210
A
TEXAS INSTRUMENTS

4 OF 7


1603210
4
EADD0-EADD15
SH 2, 3, 5, 6
R15
+5V
47K
MEMENB-
SH 6
CRUBIT IG
SH 2
WE-
DBIN-
SH 2, 3, 5
ERAMSEL
SH 5
HOLDA
SH 6
U43
74LS688
U56
74LS138
U2
74LS32
R14
4.7K
+5V
J7
1A5
MK4801
HOLDA
MK4801
U45
74LS245
R10
4.7K
+5V
EDATA0-EDATA7
SH 2, 3, 5, 6
U44
74LS245
U42
74LS245
TD0
TD1
TD2
TD3
TD4
TD5
TD6
TD7
P1-29
P1-30
P1-32
P1-33
P1-34
P1-35
P1-36
P1-37
TA0
TA1
TA2
TA3
TA4
TA5
TA6
TA7
TA8
TA9
TA10
TA11
TA12
TA13
TA14
TA15
P2-17
P2-16
P2-15
P2-14
P2-13
P2-12
P2-11
P2-9
P2-8
P2-7
P2-6
P2-5
P2-4
P2-3
P2-2
P2-1
TEXAS INSTRUMENTS
D06668
1603210
A


1603210
5
CRUBIT 7
BP
TMFULL
BPQ
CLKOUT
U10
PRE
CLR
74LS74
R18
4.7K
+5V
THOLD-
P1-23
J2
R1
4.7K
U24
74LS00
U4
74LS32
U3
74LS112
R2
4.7
BPO
BPHR
U12
74LS02
U14
74LS08
U15
74LS27
U1
74LS04
IAQ
SH 3,6
R7
1K
CRUBIT 6
373 CLOCK
TVCC
P1-31
U9
74LS251
HOLDA
NC
A14/EVM
A13/EVM
A12/EVM
SEL7-/EVM
CRUIN-/EVM
P4-32
WE-
SH 2,3,4
CYCLEND
MEMENB-
CRUBIT 5
CRUBIT 4
TRESET-
P2-19
HOLD-
RESET-
READY
U13
TMS 9995
CRUBIT 3
ERAMSEL
TREADY
P2-18
CRUBIT 2
U11
U2
TINT4-
P1-27
TINT1-
P1-26
INT4-
INT1-
R11
47K
R12
47K
MEMEN-
IAQ/HOLDA
WE-/CRUCLK-
DBIN-
CLKOUT
U22
74LS367
TMEMEN-
P1-21
TIAQ/HOLDA
P1-25
J5
TWE-/CRUCLK-
P1-22
TDBIN-
P1-24
TCLKOUT
P1-38
J3
J4
R3
47K
R5
47K
R4
47K
R13
47K
CRUBIT 1
TNMI-
P2-20
NMI-
TCRUIN
P1-28
CRUIN
XTAL 2
XTAL1
C2
18pf
J1
Y1
12 MHZ
C1
18 pf
TCLKIN
P1-39
TRWE
SH6
EDATA0 - EDATA7
EADD0 - EADD15
SH 2, 3, 4, 6
DBIN-
SH 2, 3, 4
MEMENBX-
SH 3
CRUBIT 10
SH 2
CRUBIT 21
SH 2
TEXAS INSTRUMENTS
D 06668
1603210
A

6 OF 7


1603210
6
EADD0-EADD15
IAQ
J8
R9
4.7K
+5V
TRWE
MISCR1
MISCR2
MISCR3
MISCR4
EDATA0-EDATA7
MISCW
CRUBIT19
CYCLEND
TROK
CRUBIT18
TMFULL
U10
74LS74
U28
U29
74S163
U30
74LS244
95422
D06668
1603210
A

Manual Update

MANUAL TITLE: TMAM6083 EMULATOR FOR TMS9995 USER'S GUIDE

REVISION CHANGE: B TO C MP NUMBER: 50 P/N: 1603772-9701

PRINTING DATE: NOVEMBER 1981 DATE OF CHANGE: APRIL 8, 1982

ECN NUMBER: 486614 MANUAL UPDATE NUMBER: 95EMU-*-3

CHANGES/ADDITIONS REQUESTED

Page 2-5, Item 3: Change under pin number "6 Data Set Ready" to "20 Data Terminal Ready". Change last sentence of Item 3: to read "If DTR is to be used or is unavailable, pin 20 should be jumpered to +12 (pin 6).

Change the last sentence of the first paragraph of Item 4, page 2-5 to read as follows: "In addition to the cable connection shown in the EVM manual, the Data Terminal Ready (DTR) signal must be provided (pin 20 to pin 20).

Page 3-11, Paragraph 3.3.2.1 and Paragraph 3.3.2.2; and Page 3-14, Paragraph 3.3.3.3: Immediately following the examples in each of the above mentioned paragraphs add the following note:

NOTE: When the emulator halts, the TMS9995 processor used in the emulator is reset. This results in:

- The rising-edge triggered latches for INT1- and INT4 are cleared.
- FLAG0 and FLAG1 are set to zero. This configures the decrementer as an interval timer and disables interrupts from the decrementer. This also configures INT4- as a level 4 interrupt request.
- The MTD flag (CRU address >1FDA) is cleared.

If these results are contrary to the needs of the user's application program, a small routine should be added to the user's program during debug that can be invoked when restarting the emulator to restore these, or the application system can be restarted from the user's reset vector.

Page 1-1, Paragraph 1.1 GENERAL: After item Power supply unit,...(user supplied), add the following paragraph:

NOTE: The target system clock option is not supported on the Target Connector Assy 1603352-0001.

The following six pages of schematics are to be included to the user's guide as Appendix A.

Page 3-10, Section 3.3.1.14 Reverse Assembler
Page 3-11, Section 3.3.2.2 Single Step Execution
Page 3-14, Section 3.3.3.4 Dump Trace
Add the following paragraph to each of the sections:

Note 1

The output of this command may be paused by pressing any character except a "Q" or "esc" (escape key). Pressing either a "Q" or "esc" will terminate the output and terminate the processing of this command.

NOTE 2

Page 3-14, Section 3.3.3.4 Dump Trace
Add the following paragraph:

The output of this command is of the form:

nnnn aaaa dd xxxx

where:
- nnnn = event counter from 255 (>FF) to 1 where 1 is the last event traced
- aaaa = the address traced
- dd = the data byte traced
- xxxx (may not appear) = the data word traced. This data is output if the next address in the buffer is aaaa+1. This usually represents a word access which caused two successive trace events.

If aaaa is an onchip address (>F000 through >F0FB or >FFFA though >FFFF), the data value cannot be traced and will be represented as "??". When a word access is performed, only the first byte of the access will appear in the trace.

Page 3-11, Section 3.3.2.1 Execute With Breakpoints And Trace
Add the following paragraph:

If the user wishes to disable breakpoints or trace during the execution of the emulator, the appropriate qualifier can be set to all zeros. If both are to be disabled, the EX command can be used.

Page 3-12, Section 3.3.3.1 Set Breakpoint
Add the following paragraphs:

Breakpoints and trace are enabled only when the processor is executed with the EXB command. Use of the EX command disables both features. If the trace is to be used and the breakpoint feature is not desired, it can be disabled by specifying a qualifier of "0000".

Breakpoints will halt the processor only on instruction boundaries. Thus, the processor may halt with a PC value after the instruction that caused the breakpoint to occur. In general, the processor will halt either just before the instruction is executed, before the next instruction, or after the next instruction. The next section is presented as an aid in predicting exactly where the processor will halt. The TMS9995 performs instruction prefetch, which results in instruction cycles of the following form:

```
        <instruction read  of  opcode n>
        <destination write for opcode n-1>
        <source      read  for opcode n>
        <destination read  for opcode n>
(the destination write for opcode n is in the next cycle:)
        <instruction read  of  opcode n+1>
        <destination write for opcode n>
```

See the TMS9995 data manual for a more thorough description of the TMS9995 instruction cycles. The breakpoint hardware halts the processor by asserting a HOLD signal at the end of an IAQ (instruction aquisition) memory cycle. If the preceeding instruction requires a destination write, the processor will complete the write before honoring the HOLD signal. As an example of the possible cases, the results of various breakpoints upon the instruction A R0,R1 will be examined.
Assume that the following code fragment is being executed:

```
Address   Data      Instruction
  0100    0201 0005 LI   R1,5
  0104    A040      A    R0,R1
  0106    04C2      CLR  R2
  0108    0703      SETO R3
(assume WP=0080, R0=000A)
```

The following is a list of the pertinant memory cycles:

R/W	IAQ	Cycle	Addr	Data	
R	Y	1	0100	0201	LI R1, instruction
R	N	2	0102	0005	Reading value to be loaded
R	Y	3	0104	A040	A R0,R1 instruction
W	N	4	0082	0005	Writing 5 to R1
R	N	5	0080	000A	Reading R0 for add
R	N	6	0082	0005	Reading R1 for add
R	Y	7	0106	04C2	CLR R2 instruction
W	N	8	0082	000F	Writing R1 result from add
R	Y	9	0108	0703	SETO R3 instruction
W	N	10	0084	0000	Clear of R2
R	Y	11	010A	xxxx	Prefetch of next opcode
W	N	12	0086	FFFF	Set to ones of R3

A breakpoint caused by the A R0,R1 instruction's execution, could occur on any of the following memory cycles:

1 - The fetch of the opcode, cycle 3.
2 - The read of R0's value, cycle 5.
3 - The read of R1's value, cycle 6.
4 - The write of the result to R1, cycle 8.

In case 1, the emulator will halt before the execution of the add instruction, after cycle 4.

In cases 2 and 3, the emulator will complete the add and halt before the clear instruction is executed, after cycle 8.

In case 4, the emulator will complete the clear instruction and halt before the set to ones instruction is executed, after cycle 10. In all of the cases, the final write of the instruction in progress when the HOLD is asserted will be performed before the processor halts.

The general rule is:
- Breakpoints on instruction fetches will halt the emulator before the instruction fetched will execute.
- Breakpoints on read operations performed in the course of performing an operation will allow the opcode to complete before the emulator will halt.
- Breakpoints on write operations performed in the course of performing an operation will allow the NEXT opcode to execute completely before the emulator will halt.

Care should be taken that the byte nature of the TMS9995 is taken into account when defining events. For example, an event defined to be any access outside the range >0020 to >0040 will generate an event when a word access is made at location >0040 since the bytes >0040 and >0041 are accessed. The second access, to address >0041, is outside the specified range and will trigger an event.

Page 3-13, Section 3.3.3.2 Trace
Add the following paragraph:

The trace feature is enabled only when the EXB command is used to run the emulator. If EX is used, trace is disabled. Since the EXB command is necessary to use breakpoints, the trace qualifier can be set to 0000 to selectively disable the trace when breakpoints must still be used.

Care should be taken that the byte nature of the TMS9995 is taken into account when defining events. For example, an event defined to be any access outside the range >0020 to >0040 will generate an event when a word access is made at location >0040 since the bytes >0040 and >0041 are accessed. The second access, to address >0041, is outside the specified range and will trigger an event.

The following six pages of schematics are to be added to the user's guide as Appendix A.

SECTION 4

THEORY OF OPERATION

4.1 GENERAL

This section presents the theory of operation of the TMAM 6083 Emulator Module. Discussion in this section will cover a general system description, breakpoint and trace operations, and EMU memory.

4.2 SYSTEM DESCRIPTION

The TMAM 6083 Emulator consists of the TMAM 6095 Evaluation Board for the TMS 9995 microcomputer and the TMAM Emulator Board joined, component sides out, by a 38-pin connector. Table 4-1 lists the signals on connector.

TABLE 4-1. EVM/EMU 38-PIN CONNECTOR SIGNALS.

PIN NO.	SIGNAL	PIN NO.	SIGNAL
1	A15	20	D4
2	A14	21	D3
3	A13	22	D2
4	A12	23	D1
5	A11	24	DO
6	A10	25	GND
7	A9	26	+5V
8	A8	27	-12V
9	A7	28	+12V
10	A6	29	GND
11	A5	30	SEL7_
12	A4	31	SEL8_
13	A3	32	CRUIN
14	A2	33	RESET_
15	A1	34	MEMENBUF_
16	AO	35	DBIN_
17	D7	36	IAQ
18	D6	37	WE
19	D5	38	CRUCLK

The EVM Board functions as the controlling unit by way of bidirectional address and data buffers on the EMU board which provide EVM access to the main address and data buses on the EMU. The EVM maintains control of the EMU by using the EVM CRU bus to set control lines on the EMU.

Detailed information concerning EVM signals, buses, trace logic, on-chip RAM, user RAM, the Decrementer (timer/event counter), flags, and interrupt control is presented in Section 4 of the EVM User's Guide. Section 5 of this manual presents information on modifications made to the EVM board which will alter/void some segments of information in the EVM User's Guide.

Figure 4-1 identifies the various components of the Emulator Board. Figure 4-2 provides a simple block diagram of the Emulator System data flow.

Manual Update

MANUAL TITLE: TMAM 6083 EMULATOR FOR TMS9995 USER'S GUIDE

REVISION CHANGE: C TO D MP NUMBER: 50 P/N: 1603772-9701

PRINTING DATE: NOVEMBER 1981 DATE OF CHANGE: MAY 13, 1982

ECN NUMBER: 486626 MANUAL UPDATE NUMBER: 95EMU-*-4

CHANGES/ADDITIONS REQUESTED

Page 2-5, Item 3: Change under pin number "6 Data Set Ready" to "20 Data Terminal Ready". Change last sentence of Item 3 to read: "If DTR is unavailable, pin 20 should be jumpered to +12 (pin 6).

Change the last sentence of the first paragraph of Item 4, page 2-5 to read as follows: "In addition to the cable connection shown in the EVM manual, the Data Terminal Ready (DTR) must be provided (pin 20 to pin 20).

Page 3-11, Sections 3.3.2.1 and 3.3.2.2, and Page 3-14, Section 3.3.3.3: Immediately following the examples in each of the above mentioned sections add the following note:

NOTE: When the emulator halts, the TMS9995 processor used in the emulator is reset. This results in:

- o The rising-edge triggered latches for INT1- and INT4- are cleared.
- o FLAG0 and FLAG1 are set to zero. This configures the decrementer as an interval timer and disables interrupts from the decrementer. This also configures INT4- as a level 4 interrupt request.
- o The MID flag (CRU address >1FDA) is cleared.

If these results are contrary to the needs of the user's application program, a small routine should be added to the user's program during debug that can be invoked when restarting the emulator to restore these, of the application system can be restarted from the user's reset vector.

Page 1-1, Section 1.1 GENERAL: After item Power supply unit,...(user supplied), add the following paragraph:

NOTE: The target system clock option is not supported on the Target Connector Assembly 1603352-0001.

Page 3-5, Section 3.3.1.1 Inspect/Change CRU
Change syntax to: IC[<∧,}<cru address>[<∧,}<count>]]<(CR)>

Page 3-5, Section 3.3.1.2 Dump Memory
Change syntax to: DM[<∧,}<start address>[<∧,}<stop address>]]<(CR)>

Page 3-6, Section 3.3.1.3 Dump Memory To Digital Cassette/Paper Tape
Change syntax to: DMC[<∧,}<start address>[<∧,}<stop address>
[<∧,}<entry address>]]]<(CR)>
IDT prompt syntax: IDT=<program name><(CR)>

Page 3-6, Section 3.3.1.4 Execute
Change syntax to: EX<∧,(CR)}

Page 3-6, Section 3.3.1.5 Find Data
Change syntax to: FD[<∧,}<start address>[<∧,}<stop address>
[<∧,}<value>]]]<(CR)>

Page 3-7, Section 3.3.1.6 Hexadecimal Arithmetic

Change syntax to: HEX[<^,}<number 1>[<^,}<number 2>]]<(CR)>

Page 3-7, Section 3.3.1.7 Load Memory From Cassette Or Paper Tape
Change syntax to: LMC[<^,}<bias>]<(CR)>

Page 3-7, Section 3.3.1.8 Inspect/Change Memory
Change syntax to: IM[<^,}<start address>]<(CR)>

Page 3-8, Section 3.3.1.9 Inspect/Change User WP/PC/ST Registers
Change syntax to: IR<^,(CR)}

Page 3-9, Section 3.3.1.10 Toggle Null Flag
Change syntax to: TNF<^,(CR)}

Page 3-9, Section 3.3.1.11 Inspect/Change User Workspace Registers
Change syntax to: IWR[<^,}<register number>]<(CR)>

Page 3-10, Section 3.3.1.12 Execute Assembler With New Assembler Table
Change syntax to: XA[<^,}<assembly address>]<(CR)>

Page 3-10, Section 3.3.1.13 Execute Assembler With Existing Symbol Table
Change syntax to: XAE[<^,}<assembly address>]<(CR)>

Page 3-10, Section 3.3.1.14 Execute Reverse Assembler
Change syntax to: XRA[<^,}<start address>[<^,}<end address>]]<(CR)>

Page 3-11, Section 3.3.1.15 Execute Communications Link
Change syntax to: XCL<^,(CR)}

Page 3-11, Section 3.3.2.1 Execute With Breakpoints And Trace
Change syntax to: EXB<^,(CR)}

Page 3-11, Section 3.3.2.2 Execute Single-step Mode
Change syntax to: SS[<^,}<count>]<(CR)>

Page 3-12, Section 3.3.3.1 Set Breakpoint
Change syntax to: BP<^,(CR)}

Page 3-13, Section 3.3.3.2 Trace
Change syntax to: TR<^,(CR)}

Page 3-14, Section 3.3.3.3 Halt
Change syntax to: HLT<^,(CR)}

Page 3-14, Section 3.3.3.4 Dump Trace
Change syntax to: DT[<^,}<parameter>]<(CR)>

Page 3-15, Section 3.3.3.5 Status
Change syntax to: STA<^,(CR)}

Page 3-15, Section 3.3.3.6 Enable Target Reset
Change syntax to: RST[<^,}<value>]<(CR)>

Page 3-15, Section 3.3.3.7 Enable Non-Maskable Interrupt
Change syntax to: NMI[</\,}<value>]<(CR)>

Page 3-15, Section 3.3.3.8 Automatic First Wait State
Change syntax to: AFW[</\,}<value>]<(CR)>

Page 3-15, Section 3.3.3.9 Enable Target READY Control Of ERAM
Change syntax to: RDY[</\,}<value>]<(CR)>

Page 3-16, Section 3.3.3.10 Display Options
Change syntax to: OP</\,(CR)}

Page 3-5, Section 3.3.1.2 Dump Memory
Delete the last sentence in the first paragraph: "If no addresses ... control to EMUBUG."

Page 3-6, Section 3.3.1.4 Execute
The last paragraph ("Execution, once started, ... a user program.") should be replaced with:

> Execution, once started, will continue until interrupted by the HLT command. Breakpoint and trace are both disabled when the processor is released with this command.

Page 3-10, Section 3.3.1.12 Execute Assembler With New Assembler Table
Page 3-10, Section 3.3.1.13 Execute Assembler With Existing Symbol Table
Add the following paragraph to each of the sections:

> Symbols used by the assembler are two characters long. Longer symbols may be entered, but they will be truncated to the first two characters by the assembler.

Page 3-10, Section 3.3.1.14 Reverse Assembler
Page 3-11, Section 3.3.2.2 Single Step Execution
Page 3-14, Section 3.3.3.4 Dump Trace
Add the following paragraph to each of the sections:

The output of this command may be paused by pressing any character except a "Q" or "esc" (escape key). Pressing either a "Q" or "esc" will terminate the output and terminate the processing of this command.

Page 3-14, Section 3.3.3.4 Dump Trace
Add the following paragraph:

The output of this command is of the form:

nnnn aaaa dd xxxx

where:
- nnnn = event counter from 255 (>FF) to 1 where 1 is the last event traced
- aaaa = the address traced
- dd = the data byte traced
- xxxx (may not appear) = the data word traced. This data is output if the next address in the buffer is aaaa+1. This usually represents a word access which caused two successive trace events.

If aaaa is an onchip address (>F000 through >F0FB or >FFFA though >FFFF), the data value cannot be traced and will be represented as "??". When a word access is performed, only the first byte of the access will appear in the trace.

Page 3-11, Section 3.3.2.1 Execute With Breakpoints And Trace
Add the following paragraph:

If the user wishes to disable breakpoints or trace during the execution of the emulator, the appropriate qualifier can be set to all zeros. If both are to be disabled, the EX command can be used.

Page 3-12, Section 3.3.3.1 Set Breakpoint
Add the following paragraphs:

Breakpoints and trace are enabled only when the processor is executed with the EXB command. Use of the EX command disables both features. If the trace is to be used and the breakpoint feature is not desired, it can be disabled by specifying a qualifier of "0000".

Breakpoints will halt the processor only on instruction boundaries. Thus, the processor may halt with a PC value after the instruction that caused the breakpoint to occur. In general, the processor will halt either just before the instruction is executed, before the next instruction, or after the next instruction. The next section is presented as an aid in predicting exactly where the processor will halt. The TMS9995 performs instruction prefetch, which results in instruction cycles of the following form:

```
        <instruction read  of  opcode n>
        <destination write for opcode n-1>
        <source      read  for opcode n>
        <destination read  for opcode n>
(the destination write for opcode n is in the next cycle:)
        <instruction read  of  opcode n+1>
        <destination write for opcode n>
```

See the TMS9995 data manual for a more thorough description of the TMS9995 instruction cycles. The breakpoint hardware halts the processor by asserting a HOLD signal at the end of an IAQ (instruction aquisition) memory cycle. If the preceeding instruction requires a destination write, the processor will complete the write before honoring the HOLD signal. As an example of the possible cases, the results of various breakpoints upon the instruction A R0,R1 will be examined.
Assume that the following code fragment is being executed:

```
Address   Data      Instruction
  0100    0201 0005 LI   R1,5
  0104    A040      A    R0,R1
  0106    04C2      CLR  R2
  0108    0703      SETO R3
(assume WP=0080, R0=000A)
```

The following is a list of the pertinant memory cycles:

R/W	IAQ	Cycle	Addr	Data	
R	Y	1	0100	0201	LI R1, instruction
R	N	2	0102	0005	Reading value to be loaded
R	Y	3	0104	A040	A R0,R1 instruction
W	N	4	0082	0005	Writing 5 to R1
R	N	5	0080	000A	Reading R0 for add
R	N	6	0082	0005	Reading R1 for add
R	Y	7	0106	04C2	CLR R2 instruction
W	N	8	0082	000F	Writing R1 result from add
R	Y	9	0108	0703	SETO R3 instruction
W	N	10	0084	0000	Clear of R2
R	Y	11	010A	xxxx	Prefetch of next opcode
W	N	12	0086	FFFF	Set to ones of R3

A breakpoint caused by the A R0,R1 instruction's execution, could occur on any of the following memory cycles:

1 - The fetch of the opcode, cycle 3.
2 - The read of R0's value, cycle 5.
3 - The read of R1's value, cycle 6.
4 - The write of the result to R1, cycle 8.

In case 1, the emulator will halt before the execution of the add instruction, after cycle 4.

In cases 2 and 3, the emulator will complete the add and halt before the clear instruction is executed, after cycle 8.

In case 4, the emulator will complete the clear instruction and halt before the set to ones instruction is executed, after cycle 10. In all of the cases, the final write of the instruction in progress when the HOLD is asserted will be performed before the processor halts.

The general rule is:
- Breakpoints on instruction fetches will halt the emulator before the instruction fetched will execute.
- Breakpoints on read operations performed in the course of performing an operation will allow the opcode to complete before the emulator will halt.
- Breakpoints on write operations performed in the course of performing an operation will allow the NEXT opcode to execute completely before the emulator will halt.

Care should be taken that the byte nature of the TMS9995 is taken into account when defining events. For example, an event defined to be any access outside the range >0020 to >0040 will generate an event when a word access is made at location >0040 since the bytes >0040 and >0041 are accessed. The second access, to address >0041, is outside the specified range and will trigger an event.

Page 3-13, Section 3.3.3.2 Trace
Add the following paragraph:

The trace feature is enabled only when the EXB command is used to run the emulator. If EX is used, trace is disabled. Since the EXB command is necessary to use breakpoints, the trace qualifier can be set to 0000 to selectively disable the trace when breakpoints must still be used.

Care should be taken that the byte nature of the TMS9995 is taken into account when defining events. For example, an event defined to be any access outside the range >0020 to >0040 will generate an event when a word access is made at location >0040 since the bytes >0040 and >0041 are accessed. The second access, to address >0041, is outside the specified range and will trigger an event.

The following six pages of schematics are to be added to the user's guide as Appendix A.

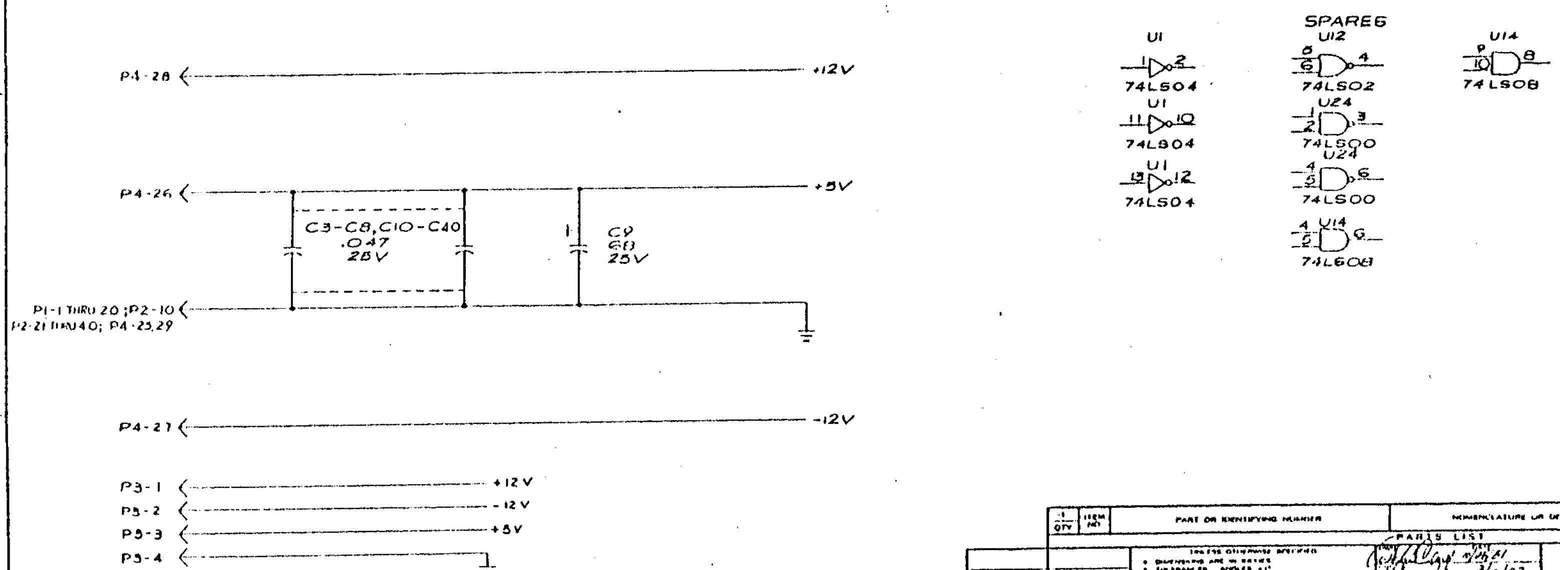
NOTES: UNLESS OTHERWISE SPECIFIED:

1. ALL CAPACITANCE VALUES ARE IN MICROFARADS
2. ALL RESISTANCE VALUES ARE IN OHMS
3. ALL RESISTORS ARE .25W, 5%

JUMPER CONFIGURATION	
JUMPER	LOCATION
J1	1,2
J2	1,2
J3	1,2
J4	1,2
J5	1,2
J6	2,3
J7	2,3
J8	2,3
J10	1,9; 10,17

REFERENCE DESIGNATORS	
USED	NOT USED
C1 – C40	
J1 – J10	J9
P1 – P4	
R1 – R18	
U1 – U58	
Y1	

REVISIONS			
REV	DESCRIPTION	DATE	APPROVED
A	INF ENG/DFTG UPDATE	2-12-82	[illegible]
B	CH494928 [illegible]	4/20/82	[illegible]



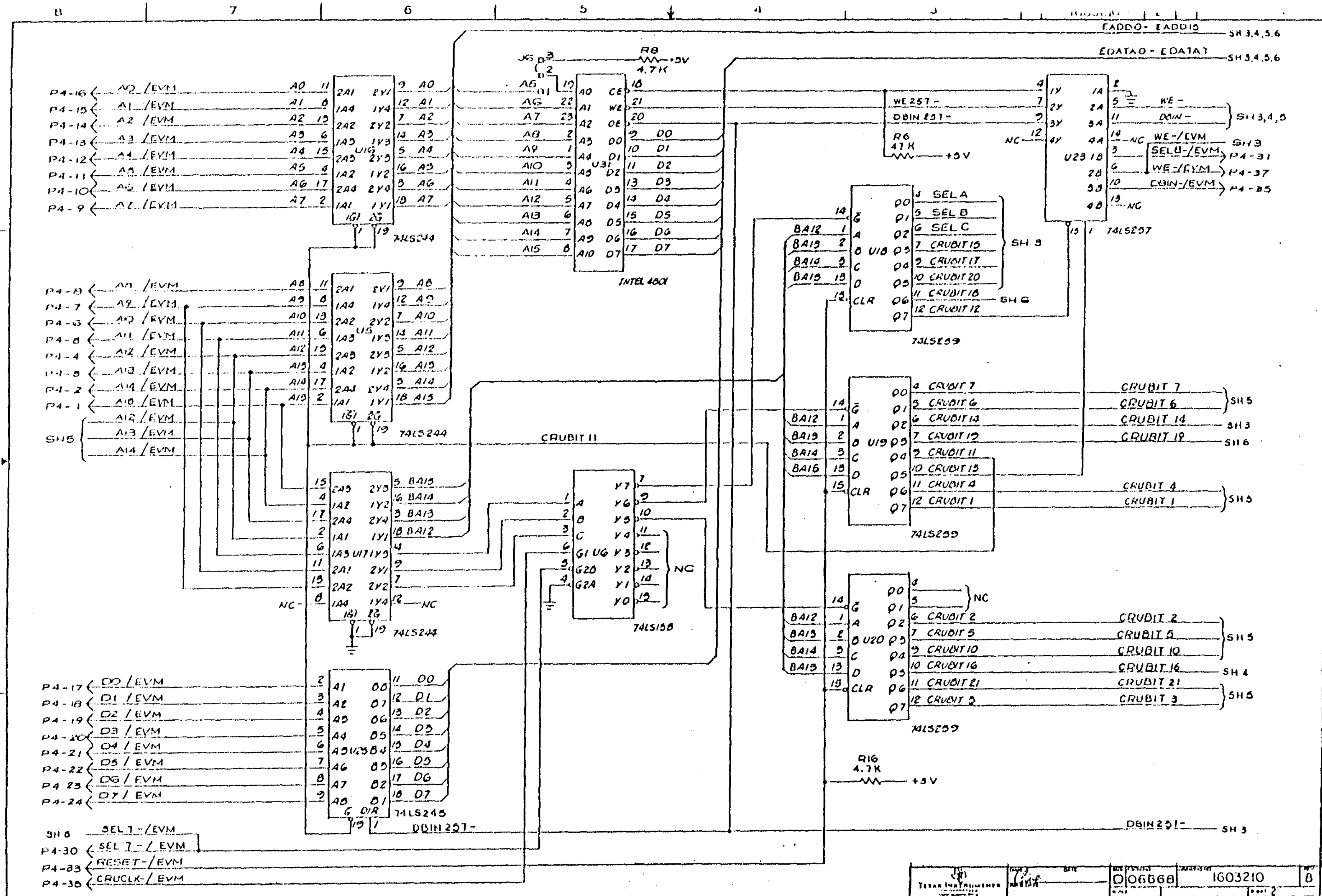
1603212	7041
NEXT ASSY	USED ON
APPLICATION	

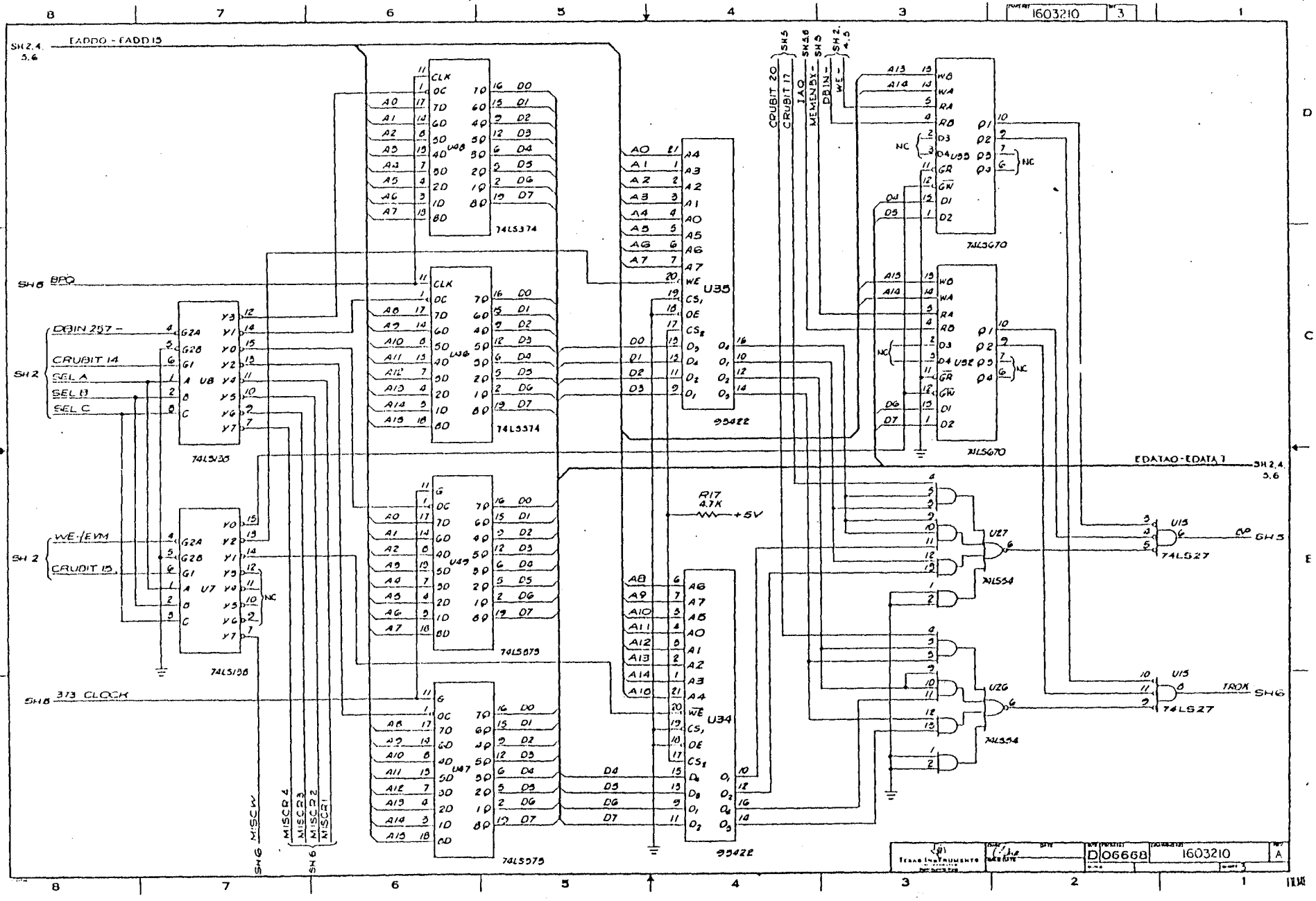
PARTS LIST

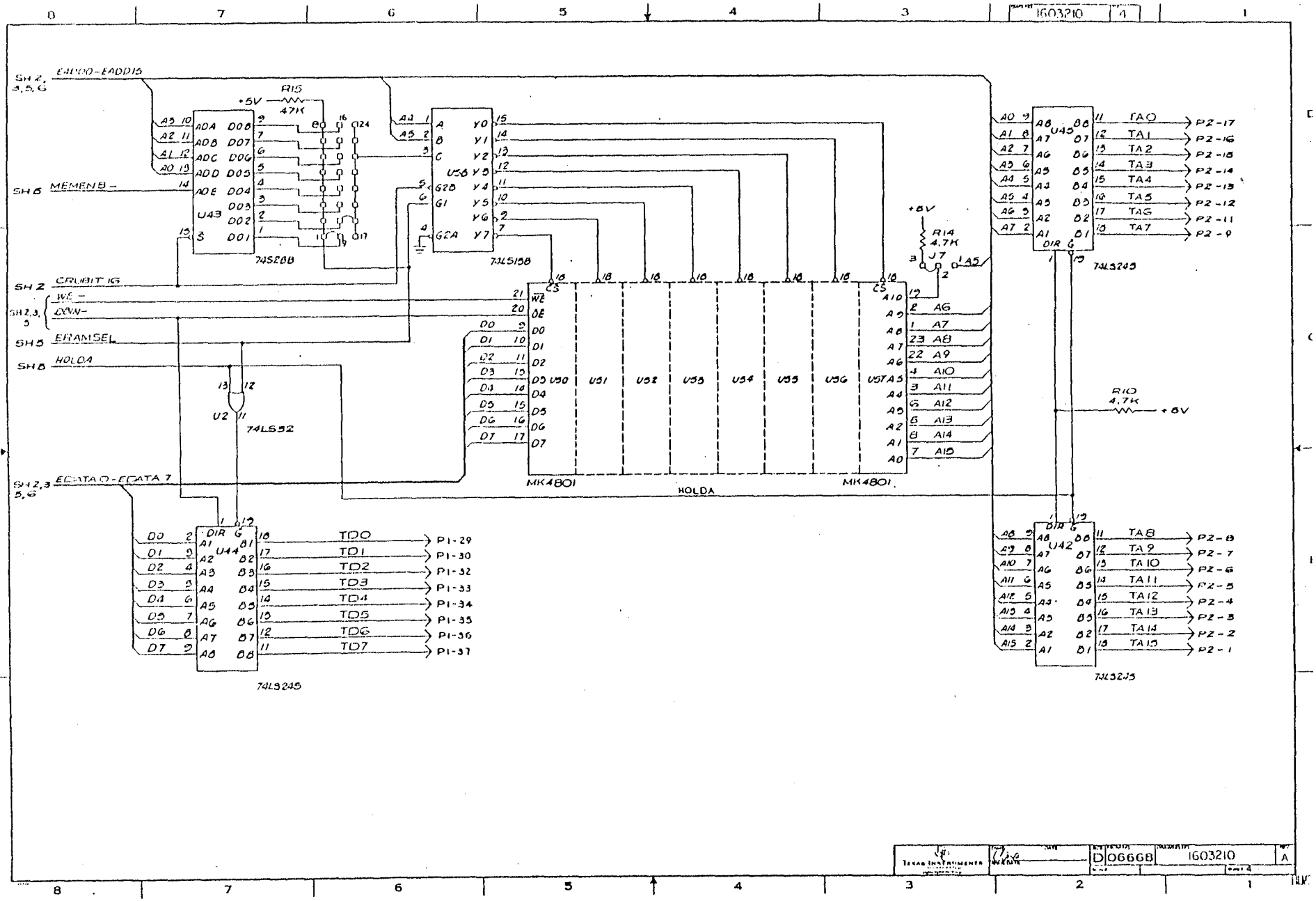
TEXAS INSTRUMENTS

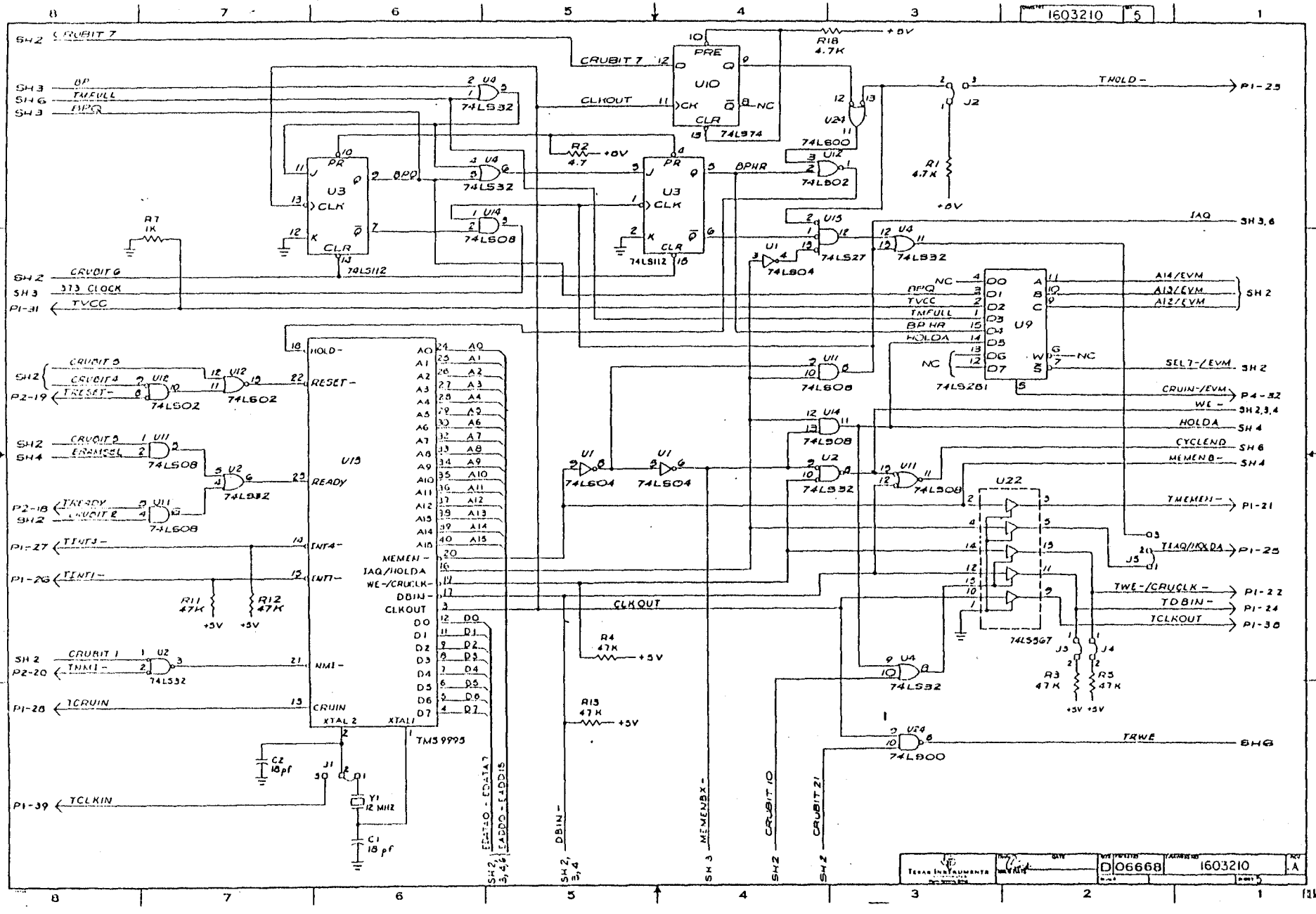
DIAGRAM, LOGIC
TMS9995 EVM - EMULATOR

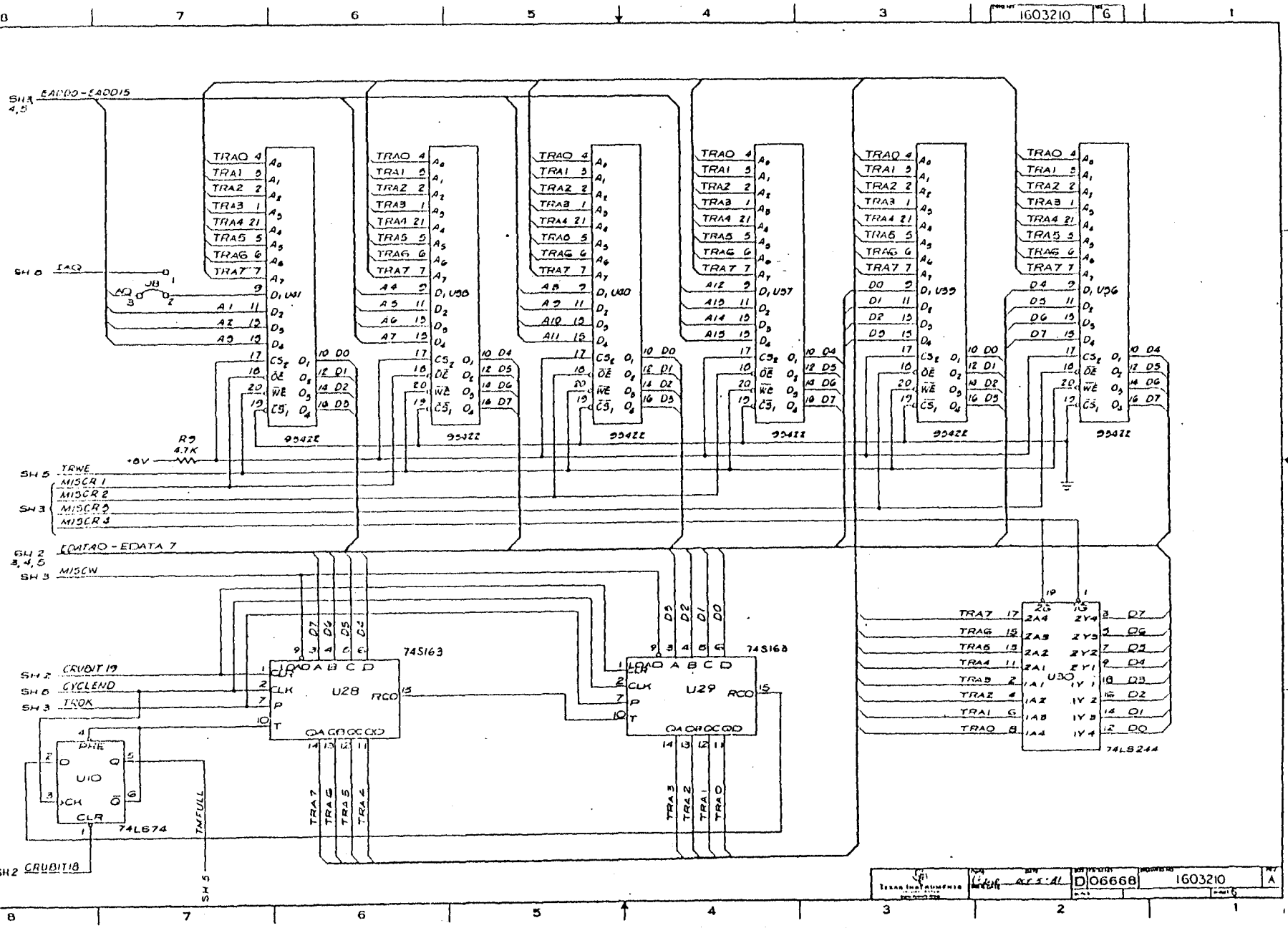
D 96214 1603210


EADDO- EADD15
SH 3,4,5,6
EDATA0 - EDATA7
SH 3,4,5,6
R8
4.7K
+5V
74LS244
INTEL 4801
74LS257
WE257-
DBIN 257-
R6
47K
+5V
WE-
DBIN-
SH 3,4,5
WE-/EVM
SH 3
SELB-/EVM
P4-31
WE-/EVM
P4-37
DBIN-/EVM
P4-35
SEL A
SEL B
SEL C
CRUBIT 15
CRUBIT 17
CRUBIT 20
CRUBIT 18
CRUBIT 12
SH 3
SH 6
74LS259
CRUBIT 7
CRUBIT 6
CRUBIT 14
CRUBIT 19
CRUBIT 11
CRUBIT 13
CRUBIT 4
CRUBIT 1
SH 5
SH 3
SH 6
74LS259
CRUBIT 2
CRUBIT 5
CRUBIT 10
CRUBIT 16
CRUBIT 21
CRUBIT 3
SH 5
SH 4
74LS259
NC
74LS138
R16
4.7K
+5V
74LS245
DBIN 257-
SH 3
P4-16 A0 /EVM
P4-15 A1 /EVM
P4-14 A2 /EVM
P4-13 A3 /EVM
P4-12 A4 /EVM
P4-11 A5 /EVM
P4-10 A6 /EVM
P4-9 A7 /EVM
P4-8 A8 /EVM
P4-7 A9 /EVM
P4-6 A10 /EVM
P4-5 A11 /EVM
P4-4 A12 /EVM
P4-3 A13 /EVM
P4-2 A14 /EVM
P4-1 A15 /EVM
SH5
A12 /EVM
A13 /EVM
A14 /EVM
P4-17 D0 /EVM
P4-18 D1 /EVM
P4-19 D2 /EVM
P4-20 D3 /EVM
P4-21 D4 /EVM
P4-22 D5 /EVM
P4-23 D6 /EVM
P4-24 D7 /EVM
SH 6 SEL 7-/EVM
P4-30 SEL 7-/EVM
P4-33 RESET-/EVM
P4-36 CRUCLK-/EVM
TEXAS INSTRUMENTS
D 06668
1603210
B


1603210
3
SH2,4,5,6 EADDO - EADD15
SH6 BRQ
SH2 DBIN 257 -
CRUBIT 14
SEL A
SEL B
SEL C
74LS138
WE-/EVM
CRUBIT 15
SH6 373 CLOCK
MISCW
MISCR4
MISCR3
MISCR2
MISCR1
74LS374
74LS373
CRUBIT 20
CRUBIT 17
IAO
MEMENBX-
DBIN-
WE-
SH5
SH5,6
SH5
SH2,4,5
U35
U34
93422
74LS670
EDATAO - EDATA7
SH2,4,5,6
R17
4.7K
+5V
74LS27
74LS54
GH3
TROX
SH6
Texas Instruments
D06668
1603210
A


1603210
SH 2, 4, 5, 6
EADDO-EADD15
R15
+5V
47K
SH 6
MEMENB-
74S288
74LS158
SH 2
CRUBIT IG
WE-
SH 2, 3, 5
DIN-
SH 5
ERAMSEL
SH 6
HOLDA
U2
74LS32
MK4801
HOLDA
MK4801
+5V
R14
4.7K
J7
1-A5
74LS245
R10
4.7K
+5V
SH 2, 3, 5, 6
EDATA0-EDATA 7
74LS245
74LS245
TD0
TD1
TD2
TD3
TD4
TD5
TD6
TD7
P1-29
P1-30
P1-32
P1-33
P1-34
P1-35
P1-36
P1-37
TA0
TA1
TA2
TA3
TA4
TA5
TA6
TA7
TA8
TA9
TA10
TA11
TA12
TA13
TA14
TA15
P2-17
P2-16
P2-15
P2-14
P2-13
P2-12
P2-11
P2-9
P2-8
P2-7
P2-6
P2-5
P2-4
P2-3
P2-2
P2-1
D06668
1603210
A


1603210
5
CRUBIT 7
CLKOUT
BPHR
BPPQ
TMFULL
CRUBIT 6
CRUBIT 5
CRUBIT 4
CRUBIT 3
CRUBIT 1
TVCC
TCLKIN
TCRUIN
TNMI-
TINT3-
TINT1-
TRESET-
TAREADY
TMEMEN-
TDBIN-
TCLKOUT
TWE-/CRUCLK-
TIAQ/HOLDA
TRWE
THOLD-
IAQ
HOLDA
CYCLEND
MEMEN8-
WE-
SEL7-/EVM
CRUIN-/EVM
A14/EVM
A13/EVM
A12/EVM
U10
U3
U13
U9
U22
74LS74
74LS32
74LS08
74LS02
74LS04
74LS00
74LS27
74LS112
74LS261
74LS367
TMS9995
R1 4.7K
R2 4.7
R3 47K
R4 47K
R5 47K
R7 1K
R11 47K
R12 47K
R13 47K
R16 4.7K
C1 18 pf
C2 18pf
Y1 12 MHz
+5V
D06668
1603210
A


1603210
6
SH8 EADD0-EADD15
4,5
SH 6 IAQ
JB
95422
R9
4.7K
+5V
SH 5 TRWE
MISCR 1
MISCR 2
SH 3 MISCR 3
MISCR 4
SH 2 EDATA0-EDATA 7
3,4,5
SH 3 MISCW
SH 2 CRUBIT 19
SH 6 CYCLEND
SH 3 TROK
74S163
U28
74S168
U29
U30
74LS244
U10
74L674
PRE
CLR
TXFULL
SH 2 CRUBIT 18
SH 5
D06668
1603210
A

Manual Update

MANUAL TITLE: TMAM6083 EMULATOR FOR TMS9995 USER'S GUIDE

REVISION CHANGE: B TO C MP NUMBER: 50 P/N: 1603772-9701

PRINTING DATE: NOVEMBER 1981 DATE OF CHANGE: APRIL 8, 1982

ECN NUMBER: 486614 MANUAL UPDATE NUMBER: 95EMU-*-3

CHANGES/ADDITIONS REQUESTED

Page 2-5, Item 3: Change under pin number "6 Data Set Ready" to "20 Data Terminal Ready". Change last sentence of Item 3: to read "If DTR is to be used or is unavailable, pin 20 should be jumpered to +12 (pin 6).

Change the last sentence of the first paragraph of Item 4, page 2-5 to read as follows: "In addition to the cable connection shown in the EVM manual, the Data Terminal Ready (DTR) signal must be provided (pin 20 to pin 20).

Page 3-11, Paragraph 3.3.2.1 and Paragraph 3.3.2.2; and Page 3-14, Paragraph 3.3.3.3: Immediately following the examples in each of the above mentioned paragraphs add the following note:

NOTE: When the emulator halts, the TMS9995 processor used in the emulator is reset. This results in:

- The rising-edge triggered latches for INT1- and INT4 are cleared.
- FLAG0 and FLAG1 are set to zero. This configures the decrementer as an interval timer and disables interrupts from the decrementer. This also configures INT4- as a level 4 interrupt request.
- The MTD flag (CRU address >1FDA) is cleared.

If these results are contrary to the needs of the user's application program, a small routine should be added to the user's program during debug that can be invoked when restarting the emulator to restore these, or the application system can be restarted from the user's reset vector.

Page 1-1, Paragraph 1.1 GENERAL: After item Power supply unit,...(user supplied), add the following paragraph:

NOTE: The target system clock option is not supported on the Target Connector Assy 1603352-0001.

The following six pages of schematics are to be included to the user's guide as Appendix A.

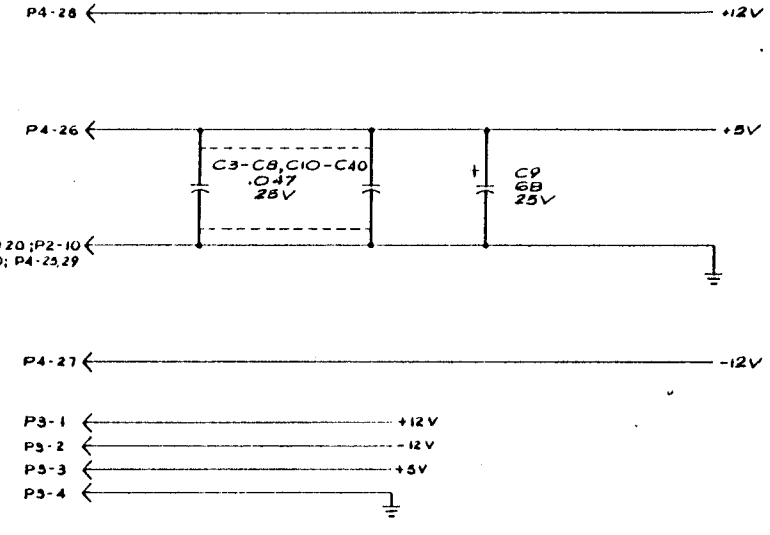
NOTES: UNLESS OTHERWISE SPECIFIED:

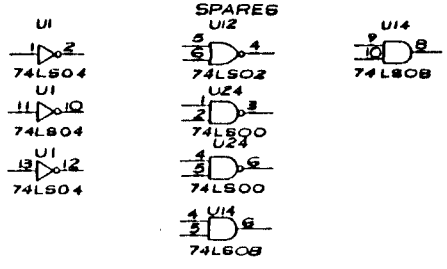
1. ALL CAPACITANCE VALUES ARE IN MICROFARADS
2. ALL RESISTANCE VALUES ARE IN OHMS
3. ALL REGISTORS ARE .25W, 5%

JUMPER CONFIGURATION	
JUMPER	LOCATION
J1	1,2
J2	1,2
J3	1,2
J4	1,2
J5	1,2
J6	2,3
J7	2,3
J8	2,3
J10	1,9 ; 10,17

REFERENCE DESIGNATORS	
USED	NOT USED
C1 - C40	
J1 - J10	J9
P1 - P4	
R1 - R18	
U1 - U58	
Y1	

REVISIONS			
REV	DESCRIPTION	DATE	APPROVED
A	INF ENG/DFTG UPDATE	2-12-82	
B	CN494928 D Keenan 4/19/82	4/20/82	





QTY	ITEM NO	PART OR IDENTIFYING NUMBER	NOMENCLATURE OR DESCRIPTION	PROCUREMENT SPECIFICATION	NOTES
		1603212	7041		

NEXT ASSY | USED ON APPLICATION

PARTS LIST

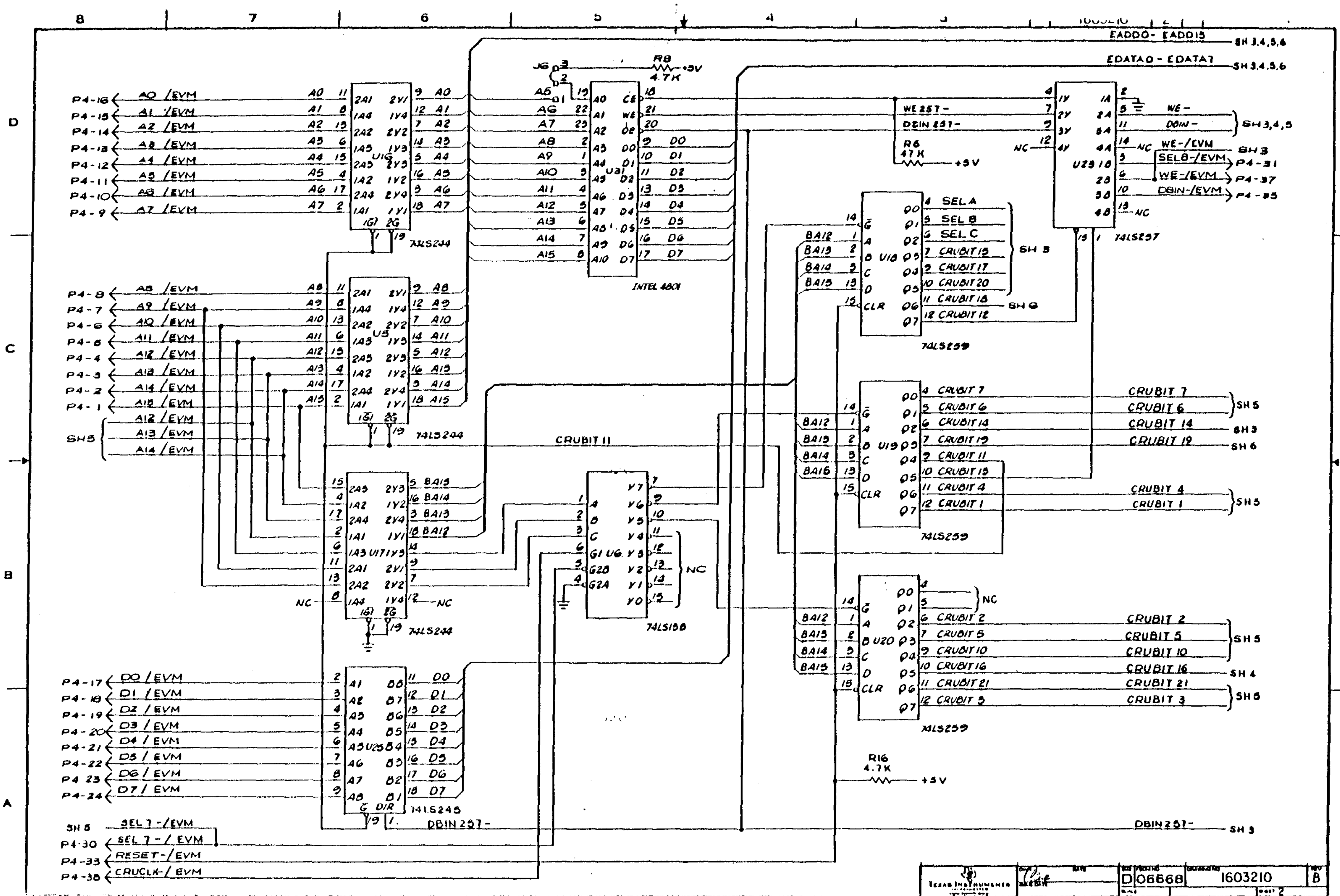
TEXAS INSTRUMENTS

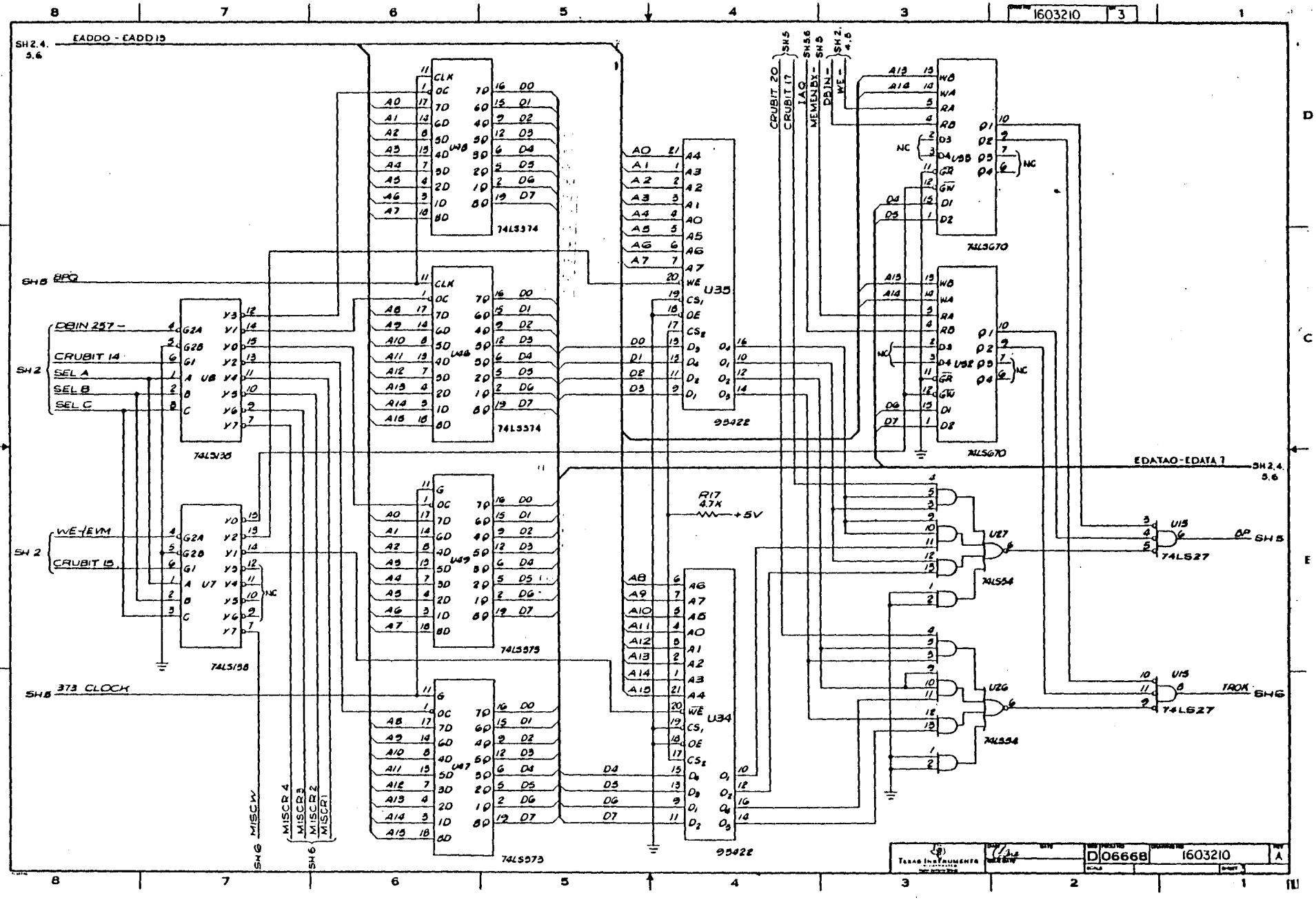
DIAGRAM, LOGIC
TMS9995 EVM-EMULATOR

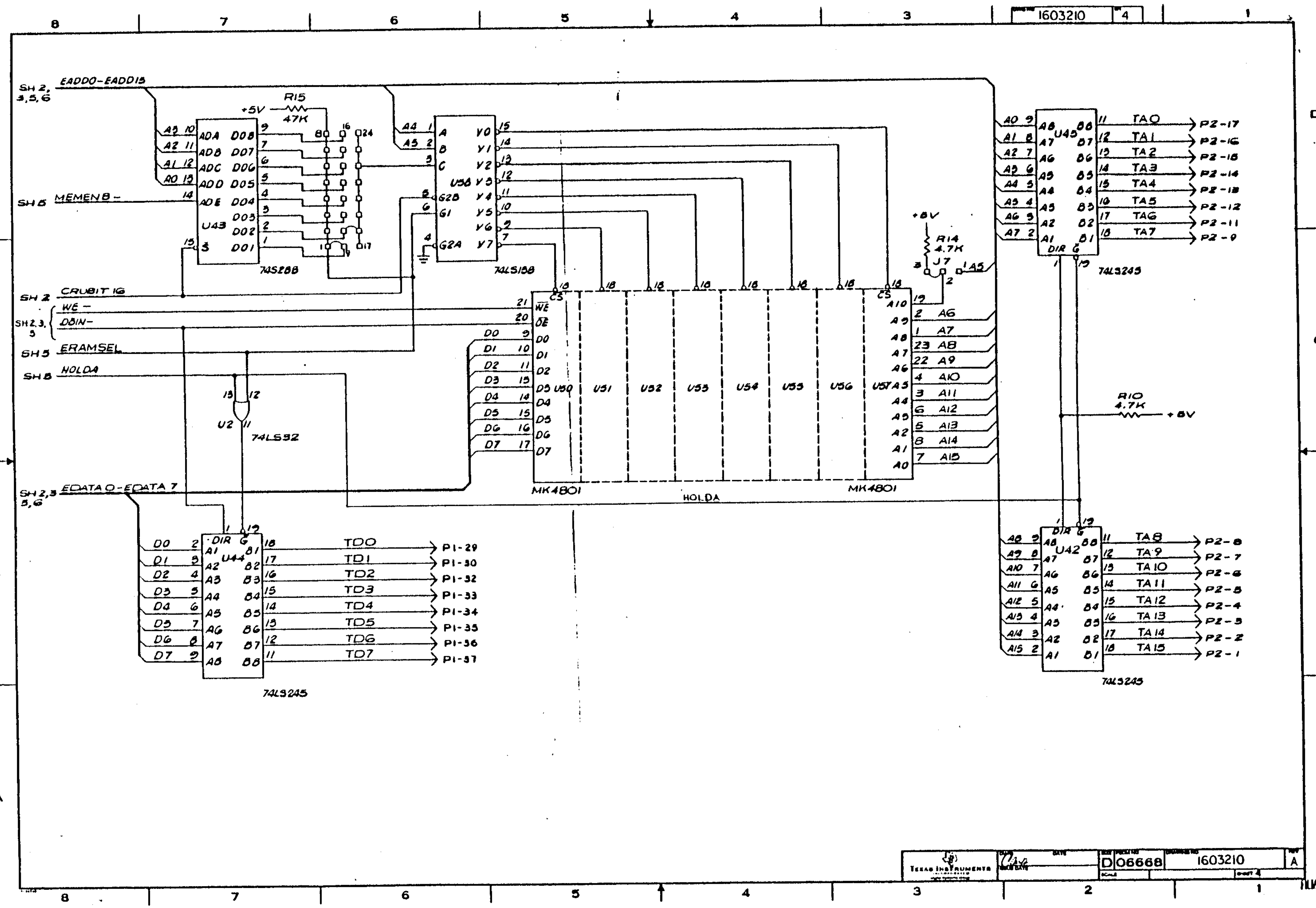
D | 96214 | 1603210

SHEET 2 OF 6

2 OF 7

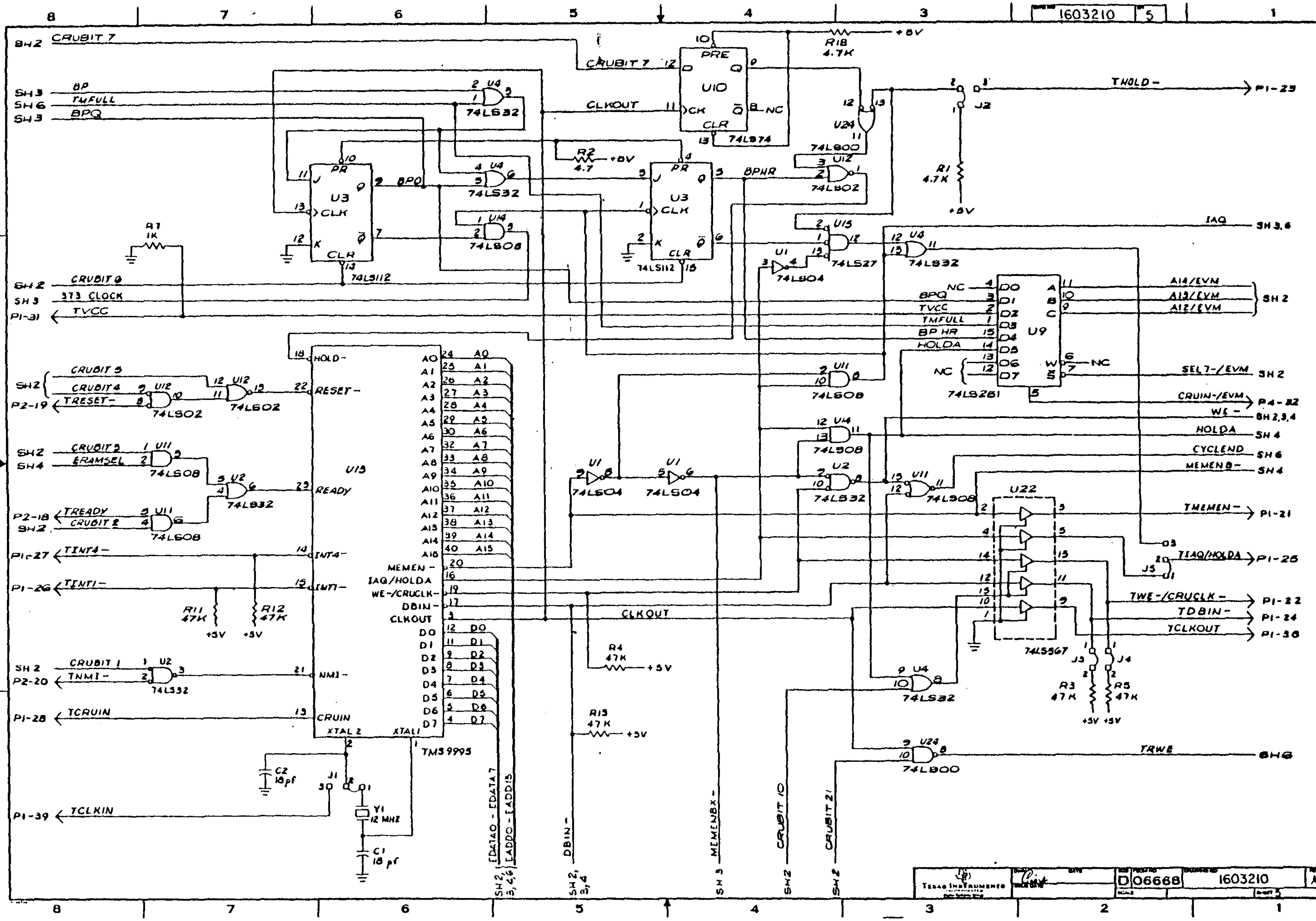

EADDO- EADD15
SH 3,4,5,6
EDATAO - EDATA7
SH 3,4,5,6
J6
R8
4.7K
+5V
P4-16
A0 /EVM
P4-15
A1 /EVM
P4-14
A2 /EVM
P4-13
A3 /EVM
P4-12
A4 /EVM
P4-11
A5 /EVM
P4-10
A6 /EVM
P4-9
A7 /EVM
U16
74LS244
INTEL 4801
U31
WE 257-
DBIN 257-
R6
47K
+5V
U23
74LS257
WE-
DBIN-
SH 3,4,5
WE-/EVM
SH 3
SEL8-/EVM
P4-31
WE-/EVM
P4-37
DBIN-/EVM
P4-35
NC
SEL A
SEL B
SEL C
CRUBIT 15
CRUBIT 17
CRUBIT 20
SH 5
CRUBIT 18
SH 6
CRUBIT 12
U18
74LS259
P4-8
A8 /EVM
P4-7
A9 /EVM
P4-6
A10 /EVM
P4-5
A11 /EVM
P4-4
A12 /EVM
P4-3
A13 /EVM
P4-2
A14 /EVM
P4-1
A15 /EVM
SH5
A12 /EVM
A13 /EVM
A14 /EVM
U5
74LS244
CRUBIT 11
U19
74LS259
CRUBIT 7
CRUBIT 6
SH 5
CRUBIT 14
SH 3
CRUBIT 19
SH 6
CRUBIT 13
CRUBIT 4
CRUBIT 1
SH 5
U17
74LS244
BA15
BA14
BA13
BA12
U6
74LS138
NC
U20
74LS259
CRUBIT 2
CRUBIT 5
CRUBIT 10
SH 5
CRUBIT 16
SH 4
CRUBIT 21
CRUBIT 3
SH 6
P4-17
D0 /EVM
P4-18
D1 /EVM
P4-19
D2 /EVM
P4-20
D3 /EVM
P4-21
D4 /EVM
P4-22
D5 /EVM
P4 23
D6 / EVM
P4-24
D7 / EVM
U25
74LS245
R16
4.7K
+5V
SH 6
SEL 7-/EVM
P4-30
SEL 7- / EVM
P4-33
RESET-/EVM
P4-38
CRUCLK-/ EVM
DBIN 257-
SH 3
TEXAS INSTRUMENTS
D
06668
1603210
B

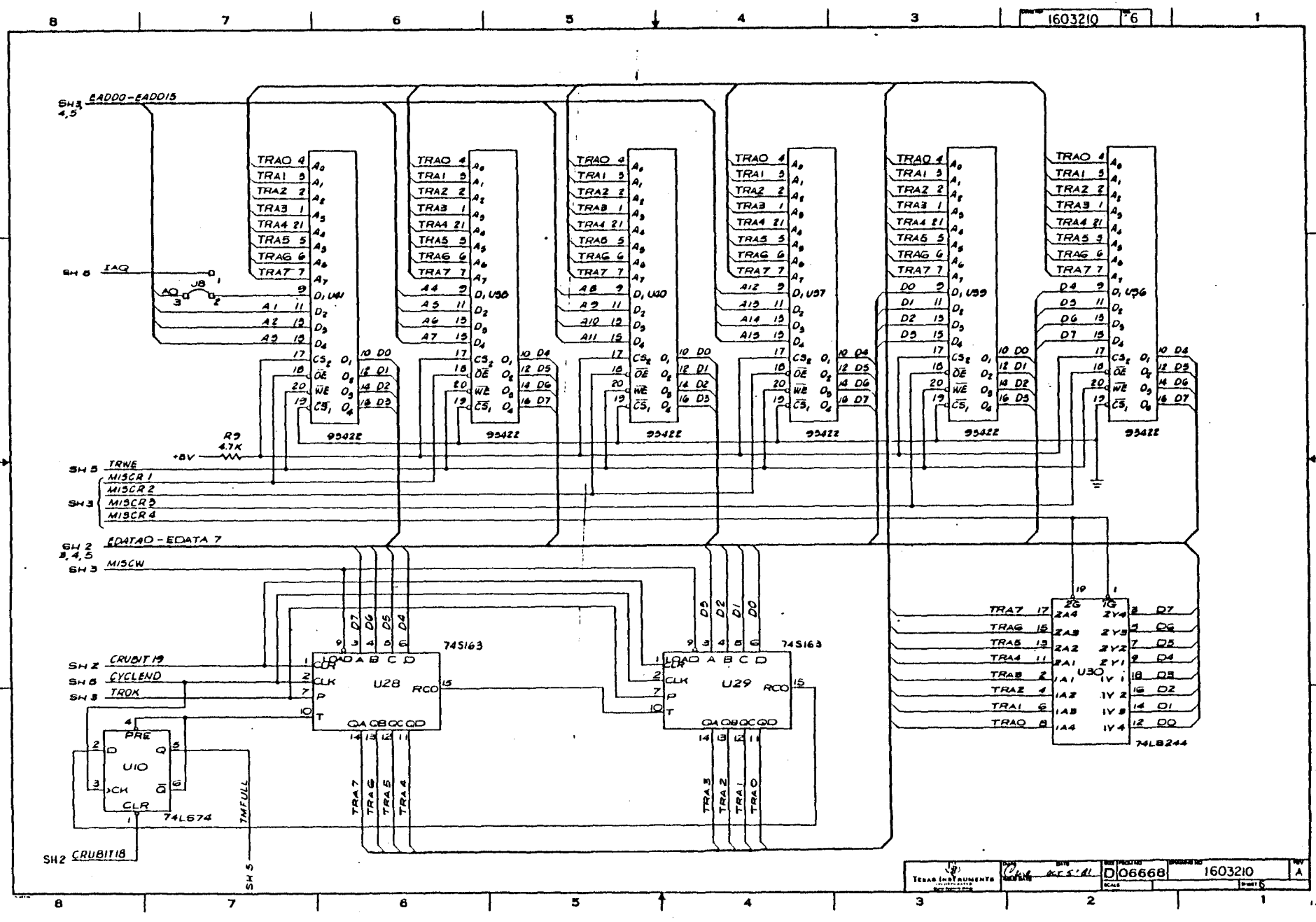

1603210
EADDO - EADD15
SH2,4,5,6
SH8 BRQ
SH8 373 CLOCK
DBIN 257 -
CRUBIT 14
SEL A
SEL B
SEL C
SH2
WE-/EVM
CRUBIT 15
74LS374
74LS138
74LS573
74LS158
95422
74LS670
74LS27
74LS54
74LS34
U34
U35
U36
U37
U38
U39
U7
U8
U13
U26
U27
R17 4.7K
+5V
NC
MISCW
MISCR4
MISCR3
MISCR2
MISCR1
SH6
CRUBIT 20
CRUBIT 17
IAQ
MEMENBX-
DBIN-
WE-
EDATAO-EDATA7
DP SH5
TROK SH6
D06668
1603210
A


1603210
4
EADDO-EADD15
SH 2, 3, 5, 6
R15 47K
+5V
U43
74S288
MEMEN8-
SH 6
U58
74LS158
CRUBIT 16
SH 2
WE-
DBIN-
SH 2, 3, 5
ERAMSEL
SH 5
HOLDA
SH 8
U2
74LS32
U50
U51
U52
U53
U54
U55
U56
U57
MK4801
HOLDA
+5V
R14 4.7K
J7
U45
74LS245
TA0
TA1
TA2
TA3
TA4
TA5
TA6
TA7
P2-17
P2-16
P2-15
P2-14
P2-13
P2-12
P2-11
P2-9
R10 4.7K
EDATA 0-EDATA 7
SH 2, 3, 5, 6
U44
74LS245
TD0
TD1
TD2
TD3
TD4
TD5
TD6
TD7
P1-29
P1-30
P1-32
P1-33
P1-34
P1-35
P1-36
P1-37
U42
74LS245
TA8
TA9
TA10
TA11
TA12
TA13
TA14
TA15
P2-8
P2-7
P2-6
P2-5
P2-4
P2-3
P2-2
P2-1
TEXAS INSTRUMENTS
D 06668
1603210
A

5 OF 7

6 OF 7


1603210
5
CRUBIT 7
BP
TMFULL
BPQ
CLKOUT
U10
74LS74
R18
4.7K
+5V
THOLD-
P1-25
J2
R1
4.7K
U3
74LS112
U4
74LS32
R2
4.7
BPHR
U12
74LS02
74LS00
U14
74LS08
U15
74LS27
U1
74LS04
IAQ
SH 3,6
R7
1K
CRUBIT 6
373 CLOCK
TVCC
P1-31
U9
74LS251
A14/EVM
A13/EVM
A12/EVM
SH 2
TMFULL
HOLDA
SEL 7-/EVM
CRUIN-/EVM
P4-32
WE-
SH 2,3,4
SH 4
CYCLEND
SH 6
MEMENB-
CRUBIT 5
CRUBIT 4
TRESET-
P2-19
RESET-
HOLD-
ERAMSEL
READY
TREADY
P2-18
CRUBIT 2
U13
TINT4-
P1-27
INT4-
TINT1-
P1-26
INT1-
R11
47K
R12
47K
MEMEN-
IAQ/HOLDA
WE-/CRUCLK-
DBIN-
CLKOUT
U22
74LS367
TMEMEN-
P1-21
J5
TIAQ/HOLDA
TWE-/CRUCLK-
P1-22
TDBIN-
P1-24
TCLKOUT
P1-36
R4
47K
J3
J4
R3
47K
R5
47K
CRUBIT 1
TNMI-
P2-20
NMI-
TCRUIN
P1-28
CRUIN
R13
47K
XTAL 2
XTAL 1
TMS 9995
TRWE
SH 6
C2
18 pf
J1
Y1
12 MHZ
C1
18 pf
TCLKIN
P1-39
EDATA0 - EDATA7
EADD0 - EADD15
DBIN-
MEMENBX-
CRUBIT 10
CRUBIT 21
TEXAS INSTRUMENTS
D 06668
1603210
A


1603210
SH3 4,5 EADDO-EADDI5
SH 6 IAQ
J8
R9 4.7K
+8V
SH 5 TRWE
MISCR 1
MISCR 2
SH3 MISCR 3
MISCR 4
SH 2 3,4,5 EDATAO - EDATA 7
SH 3 MISCW
SH 2 CRUBIT 19
SH 6 CYCLEND
SH 3 TROK
SH2 CRUBIT 18
SH 5 TMFULL
U28
U29
U30
U10
745163
74LS74
74LS244
95422
TRAO
TRA1
TRA2
TRA3
TRA4
TRA5
TRA6
TRA7
D0
D1
D2
D3
D4
D5
D6
D7
TEXAS INSTRUMENTS
D06668
1603210
A