TEXAS INSTRUMENTS

TMS320C54x Evaluation Module

Technical Reference

1995 Microprocessor Development Systems

Printed in U.S.A., October 1995
D425002-9741 revision*

SPRU135

TMS320C54x Evaluation Module Technical Reference

SPRU135
October 1995

Printed on Recycled Paper

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

Preface

Read This First

About This Manual

This document describes the board-level operation of the TMS320C54x evaluation module (EVM).

The TMS320C54x EVM is a PC-AT plug-in card that lets you evaluate certain characteristics of the TMS320C54x digital signal processor (DSP) to determine if the DSP meets your application requirements. You can also create your software to run on board or expand the system in a variety of ways.

Notational Conventions

This document uses the following conventions.

- The TMS320C541 and TMS320C542 processors are referred to collectively as the ’C54x.
- Program listings, program examples, and interactive displays are shown in a special typeface similar to a typewriter’s typeface. Here is a sample program listing:

```
@radix 16;
equations
    !usrsel_ = !haen_ & !bhe_ & !ha0 &
      ( ((map==0) & (addr >= 0a40) & (addr < 0a60))
      # ((map==1) & (addr >= 0a80) & (addr < 0aa0))
      # ((map==2) & (addr >= 0b20) & (addr < 0b40))
      # ((map==3) & (addr >= 0b40) & (addr < 0b60)) );
```

Information About Cautions

This book may contain cautions.

This is an example of a caution statement.

A caution statement describes a situation that could potentially damage your software or equipment.

The information in a caution is provided for your protection. Please read each caution carefully.

Related Documentation From Texas Instruments

To obtain a copy of any of these TI documents, call the Texas Instruments Literature Response Center at (800) 477–8924. When ordering, please identify the document by its title and literature number.

TMS320C54x User's Guide (literature number SPRU131) describes the TMS320C54x 16-bit, fixed-point, general-purpose digital signal processors. It describes pin assignments, architecture, internal register structure, instruction set, pipeline, specifications, DMA, and I/O ports. Software and hardware applications are covered in dedicated chapters.

TMS320C54x Assembly Language Tools User's Guide (literature number SPRU102) describes the assembly language tools (assembler, linker, and other tools used to develop assembly language code), assembler directives, macros, common object file format, and symbolic debugging directives for the 'C54x generation of devices.

TMS320C54x Optimizing C Compiler User's Guide (literature number SPRU103) describes the 'C54x C compiler. This C compiler accepts ANSI standard C source code and produces TMS320 assembly language source code for the 'C54x generation of devices.

TMS320C5xx C Source Debugger User's Guide (literature number SPRU099) tells you how to invoke the 'C54x emulator, EVM, and simulator versions of the C source debugger interface. This book discusses various aspects of the debugger interface, including window management, command entry, code execution, data management, and breakpoints. It also includes a tutorial that introduces basic debugger functionality.

If You Need Assistance. . .

If you want to. . .	Do this. . .
Request more information about Texas Instruments Digital Signal Processing (DSP) products	Write to: Texas Instruments Incorporated Market Communications Manager, MS 736 P.O. Box 1443 Houston, Texas 77251–1443
Ask questions about product operation or report suspected problems	Contact the electronic bulletin board: **(713) 274–2323**
Order Texas Instruments documentation	Call the TI Literature Response Center: **(800) 477–8924**
Ask questions about product operation or report suspected problems	Call the DSP hotline: **(713) 274–2320** **FAX: (713) 274–2324**
Report mistakes in this document or any other TI documentation	Fill out and return the reader response card at the end of this book, or send your comments to: Texas Instruments Incorporated Technical Publications Manager, MS 702 P.O. Box 1443 Houston, Texas 77251–1443 Electronic mail: **comments@books.sc.ti.com**

FCC Warning

This equipment is intended for use in a laboratory test environment only. It generates, uses, and can radiate radio frequency energy and has not been tested for compliance with the limits of computing devices pursuant to subpart J of part 15 of FCC rules, which are designed to provide reasonable protection against radio frequency interference. Operation of this equipment in other environments may cause interference with radio communications, in which case the user at his own expense will be required to take whatever measures may be required to correct this interference.

Trademarks

MS-DOS and MS-Windows are registered trademarks of Microsoft Corp.

PC-DOS is a trademark of International Business Machines Corp.

Contents

Figures

Tables

Chapter 1

Overview of the TMS320C54x EVM

The TMS320C54x evaluation module (EVM) is a PC/AT plug-in card that lets you evaluate certain characteristics of the ’C54x digital signal processor (DSP) to see if the DSP meets your application requirements. You can also create your software to run on board or expand the system in a variety of ways.

The ’C54x EVM carries a ’C541 DSP on board to allow full-speed verification of ’C54x code. The ’C541 has 5K bytes of on-chip program/data RAM, 28K bytes of on-chip ROM, two serial ports, a timer, access to 64K bytes each of external program and data RAM, and an external analog interface for evaluation of the ’C54x family of devices for a given application.

To simplify code development and shorten debugging time, a graphical, window-oriented debugger is also included. Its friendly, window-, mouse-, and menu-oriented interface reduces learning time and eliminates the need to memorize complex commands. For more information about the debugger, refer to the *TMS320C5xx C Source Debugger User’s Guide*.

Topic **Page**

1.1 Key Features of the TMS320C54x EVM

The ’C54x EVM has the following features:

- ’C541 operating at 40 MIPS with 128K words of zero wait-state memory
- Voice-quality analog interface to line I/O or speaker/microphone (user-selectable) via standard RCA connectors
- External serial port
- Parallel I/O expansion bus
- Two 16-bit bidirectional host/target communication channels; one channel contains 64 words of buffering
- Embedded emulation support based on the IEEE 1149.1 standard
- A single 16-bit ISA half card, mappable to one of four I/O locations

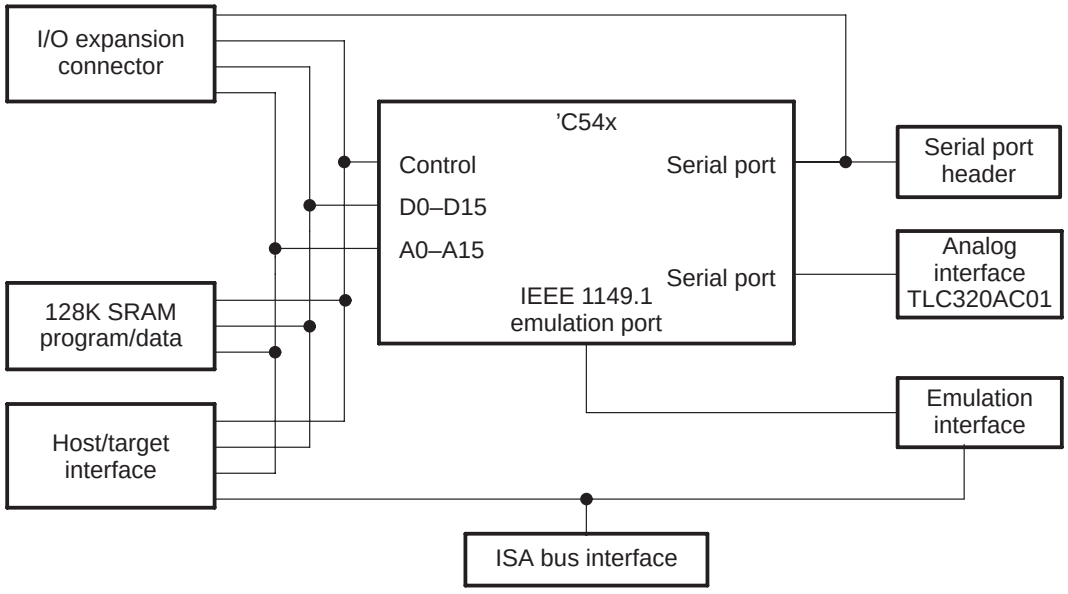
1.2 Functional Overview of the TMS320C54x EVM

Figure 1–1 shows the basic configuration and interconnects of the ’C54x EVM, including the host interface, target memory, analog interface, and emulation interface. The ’C54x EVM supports a 16-bit PC/AT-host interface.

The ’C54x interfaces to 128K words of zero wait-state static memory. An external I/O interface supports 16 parallel I/O ports, a serial port, and other I/O features. The I/O connector is a standard, 64-pin DIN connector. The TLC320AC01 analog interface circuit interfaces to the ’C54x serial port. Two RCA connectors provide analog input and output.

A 74ACT8990 supports embedded emulation for the C source debugger and loading of code. There is no on-board boot PROM/EPROM or direct access into the ’C54x memory. User applications communicate to the ’C54x through the host communications port.

Figure 1–1. ’C54x EVM Connectivity



Chapter 2

TMS320C54x EVM Host Requirements

This chapter describes PC host requirements for the ’C54x evaluation module (EVM).

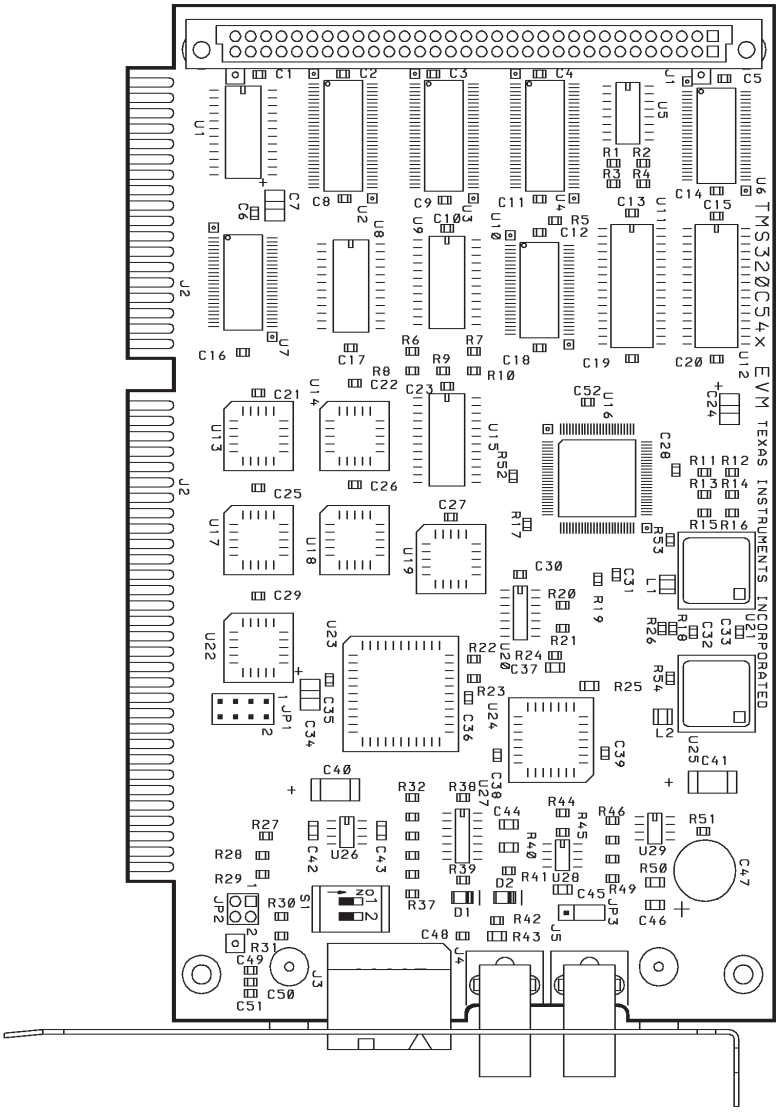
Topic **Page**

2.1 The TMS320C54x EVM Board

The ’C54x EVM board is a PC/AT half-length board that installs in a vacant 16-bit slot in your PC. Figure 2–1 shows the layout of the ’C54x EVM.

For more information about installing the ’C54x EVM, refer to the *TMS320C54x Evaluation Module Installation Guide*.

Figure 2–1. ’C54x EVM Board



2.2 Host Requirements

To install the EVM board and test and run applications successfully, you need at least a minimum hardware configuration. These are the minimum host system requirements for the ’C54x EVM:

- **Host.** An IBM PC/AT or 100%-compatible ISA/EISA-based PC
- **Expansion slot.** One 16-bit slot
- **I/O bus clock speed.** 8MHz or less
- **Memory.** At least 640K bytes of RAM
- **I/O space.** 96 bytes (mapped into three, 32-byte pages)
- **Power requirements.**
  - 1.5 amps at 5 volts
  - 0.1 amps at –5 volts
  - 0.5 amps at +12 volts
  - 0.1 amps at –12 volts

If you use a daughter card, you might exceed the above current ratings. However, make sure that you do not to exceed the host computer’s recommended power ratings.

For more information about installing the ’C54x EVM, refer to the *TMS320C54x Evaluation Module Installation Guide*.

2.3 Host I/O Map Requirements

The ’C54x EVM resides in the host I/O address space. It requires three 32-byte pages, for a total of 96 bytes. Pages are spaced 1K bytes apart. Since PC/AT-compatible machines decode only the first 1K of I/O space, the three pages appear to be mapped on top of one another.

You can map the EVM into one of four I/O base address ranges by setting switches 1 and 2, as shown in Table 2–1. Check your PC system documentation to make sure that the I/O space you have selected does not conflict with other I/O devices such as disk controllers, local area network controllers, etc.

Table 2–1. ’C54x EVM Switch Settings for I/O Address Space

I/O Address Space	Switch #1	Switch #2	
0x0240-0x025F page 0	on	on	default setting
0x0640-0x065F page 1			
0x0A40-0x0A5F page 2			
0x0280-0x029F page 0	on	off	
0x0680-0x069F page 1			
0x0A80–0x0A9F page 2			
0x0320-0x033F page 0	off	on	
0x0720-0x073F page 1			
0x0B20-0x0B3F page 2			
0x0340-0x035F page 0	off	off	
0x0740-0x075F page 1			
0x0B40-0x0B5F page 2			

Chapter 3

TMS320C54x EVM Operation

This chapter describes the ’C54x evaluation module (EVM), its key components, and how they operate. It also provides additional information on the EVM’s various interfaces.

The ’C54x EVM consists of seven major blocks of logic:

- ’C54x memory interface
- PC/AT host interface
- ’C54x I/O interface
- Host/target communications interface
- External serial port
- Analog interface
- Emulation interface

Topic **Page**

3.1 TMS320C54x Memory Interface

The EVM includes 64K words of zero wait-state program memory and 64K words of zero wait-state data memory, providing a total of 128K words of external memory. Two important conditions affect memory usage:

- **Memory precedence.** Internal memory, when enabled, takes precedence over external memory. If you want to use external memory at locations that correspond to internal memory locations, you must disable those on-chip memories at run time. Refer to the *TMS320C54x User's Guide* for more information.
- **Wait states.** External memory is subject to automatic wait-state insertion. If you want to access external memory without wait states, you must set the on-chip wait-state generator to zero wait states for program and data memory. Refer to the *TMS320C54x User's Guide* for more information.
  - For the 'C54x memory, no wait states or program/data bank switching delay is necessary.
  - For the 'C54x host interface I/O, two or more wait states are required. However, external I/O devices of your own may require more wait states. Refer to the *TMS320C54x User's Guide* for more information about the software-programmable wait-state generator.

3.2 PC/AT Host Interface

The PC/AT interface provides the buffering, host I/O decode, and access control. The PC/AT host communicates to the ’C54x EVM via 38 16-bit I/O locations. Each I/O location is defined by the I/O page 0 address described by an offset. The first 32 I/O locations support emulation. The remaining six locations support host/target communications and control. The I/O offsets are defined in Table 3–1.

Table 3–1. ’C54x EVM Host Interface Register Offsets

I/O Offset From Base Address	Register	Size	Register Type
0x0000–0x01F	Test bus controller	16	Read/write
0x0400–0x41F	Test bus controller	16	Read/write
0x0800	Channel A	16	Read/write
0x0802	Reserved	16	Read/write
0x0804	Channel B	16	Read/write
0x0806	Channel B	16	Read/write
0x0808	Status/control	16	Read/write
0x080A	Status/control	16	Read/write

Note: All locations are defined as 16-bit words. The EVM hardware ignores byte accesses.

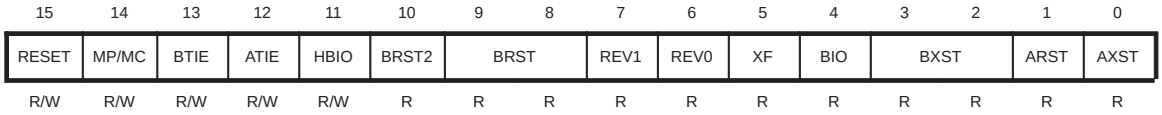
Register usage is summarized in Table 3–2.

Table 3–2. ’C54x EVM Host Interface Register Usage

Register	Description
Test bus controller	Used for emulation
Channel A	A single, bidirectional register for transferring commands and data ❑ Host writes to channel A (offset 0x0800) overwrite the current value and generate an INT1 to the target, set the AXST bit (bit 0) in the host control register, and set the ARST bit (bit 1) in the target control register. ❑ Host reads to the same location clear the ARST bit (bit 1) in the host control register and clear the AXST bit (bit 0) in the target control register.
Channel B	A 64-deep, bidirectional first-in first-out (FIFO) register for transferring commands and data ❑ Host writes to either location (0x0804 or 0x0806) are buffered by the FIFO. Data is ignored if the channel is full. ❑ Host writes to offset 0x0806 have the side effect of generating an INT2 on the target. A transmit-FIFO-full signal also generates an INT2 on the target.
Status/control	A register that provides system-level control and status for the EVM. Host writes to offsets 0x0808 and 0x080A are identical except that writes to offset 0x080A reset the ARST, AXST, BRST, and BXST flags for the host and target and the BRST2 flag for the host. This should be done by the host at the start of the host/target communications. Note that the debugger resets (via the RESET bit) the target and emulation only.

The host control register is shown in Figure 3–1; the bit definitions are shown in Table 3–3. See Section 3.4 on page 3-9 for details of host/target communications.

Figure 3–1. Host Control Register



Note: Writes to read-only bits have no effect.

Table 3–3. Host Control Register Bit Definitions

Bit	Name	Description
0	AXST	Channel A transmit status. If AXST = 1, the host has written to its channel A register. The AXST flag is cleared when the target reads channel A.
1	ARST	Channel A receive status. If ARST = 1, the target has written to its channel A register. The ARST flag is cleared when the host reads channel A.
3–2	BXST	Channel B transmit status:
		Bit 3 **Bit 2** **Channel B Transmit Status**
		0 0 buffer empty
		0 1 buffer less than half full
		1 1 buffer half or more full
		1 0 buffer full
4	BIO	$\overline{\text{BIO}}$ input to target processor
5	XF	External flag from target processor (status)
6	REV0	Card revision status bit 0
7	REV1	Card revision status bit 1
9–8	BRST	Channel B receive status:
		Bit 9 **Bit 8** **Channel B Receive Status**
		0 0 buffer empty
		0 1 buffer less than half full
		1 1 buffer half or more full
		1 0 buffer full
10	BRST2	Channel B receive status bit 2. If BRST2 = 1, the target has written to channel B, forcing an interrupt. The BRST2 flag is cleared when the host reads its channel B.
11	HBIO	Host $\overline{\text{BIO}}$ input to target processor
12	ATIE	Channel A target interrupt enable. If ATIE = 1, channel A receive conditions generate a host interrupt.
13	BTIE	Channel B target interrupt enable. If BTIE = 1, channel B receive conditions generate a host interrupt.
14	MP/MC	Microprocessor/microcomputer mode select. The EVM powers up in microcomputer mode.
15	RESET	Software reset. If RESET = 1, the target processor and emulation logic are reset, but the host/target communication flags are not reset.

The target processor can interrupt the host under certain conditions. For more details, see Section 3.4, *Host/Target Communications*, on page 3-9. You can select from four host interrupts: IRQ5, IRQ7, IRQ10, and IRQ11. Configure jumper block JP1 to select the appropriate interrupt. JP1 jumper settings are shown in Table 3–4.

Table 3–4. Host Interrupt Selection

Host Interrupt	JP1 Jumper Setting
IRQ5	Pins 1–2
IRQ7	Pins 3–4
IRQ10	Pins 5–6
IRQ11	Pins 7–8

Note: The EVM is shipped without a jumper installed on JP1.

3.3 TMS320C54x I/O Interface

The EVM supports two communications channels configured as six I/O ports for host/target communication and sixteen I/O ports for user expansion.

Port usage is summarized as follows:

- Sixteen parallel I/O ports (**PIO**) are for applications and are available on the parallel expansion connector J1, an industry standard 64-pin female DIN connector. The I/O port address, control, and data signals are buffered. You can extend the I/O wait states with the READY signal.
- The host/target message communication system provides a simple means to pass data between the target and host while maintaining real-time operation. The message protocol is user-defined. The system can be interrupt-driven, polled, or a mixture of the two. Channels A and B can be used to pass information between the host and target:
  - **Channel A** is a single 16-bit bidirectional register mapped into two I/O port locations. Both locations are identical for reads. Target writes to location 0x0010 generate a host interrupt *only* if the host control bit ATIE is set. Target writes to location 0x0011 generate a host interrupt regardless of the value of ATIE. The only exception to this is when both ATIE and BTIE have been zero since the last interrupt was serviced. In this case, it is necessary to cycle either ATIE or BTIE high and then low in order to use this feature (target write generating a host interrupt).

- **Channel B** is a single, bidirectional, 64-deep, first-in first-out (FIFO) buffer that is also mapped into two I/O port locations. Both locations are identical for reads. If the host control bit BTIE is set, writes to either location that fill the buffer generate a host interrupt. If BTIE is set, writes to port 0x0013 generate a host interrupt whether the FIFO buffer is full or not.

  You can use channels A and B independently or together.

- A **status** I/O port is for target control and provides general-purpose control, status, and discrete bit I/O. USRBIN0 and USRBIN1, and USRBOT0, USRBOT1, and USRBOT2 are discrete TTL-compatible inputs and outputs, respectively, that are available for use on the expansion connectors. Inputs are pulled high with 4.7K resistors.

All I/O accesses require at least two wait states; use the ’C54x software-programmable wait-state generator to provide these wait states. Refer to the *TMS320C54x User’s Guide* for information about the software-programmable wait-state generator.

Table 3–5 shows target I/O port usage.

Table 3–5. ’C54x Port Usage

Port Address	Name	Usage
0x0000–0x000F	PIO	User expansion
0x0010	Channel A	Communications
0x0011	Channel A	Communications
0x0012	Channel B	Communications
0x0013	Channel B	Communications
0x0014	Status	Target status/control
0x0015	Reserved	Reserved
0x0016–0xFFFF	Reserved	Reserved

The target control register is shown in Figure 3–2; the bit definitions are shown in Table 3–6. See Section 3.4 on page 3-9 for details of host/target communications.

Figure 3–2. Target Control Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AIC-RST	USR-BOT2	USR-BOT1	USR-BOT0	Reserved				USR-BIN1	USR-BIN0	BRST		BXST		ARST	AXST
R/W	R/W	R/W	R/W					R	R	R	R	R	R	R	R

Note: Writes to read-only bits have no effect.

Table 3–6. Target Control Register Definitions

Bit	Name	Description		
0	AXST	Channel A transmit status. If AXST = 1, the target has written to its channel A register. The AXST flag is cleared when the host reads channel A.		
1	ARST	Channel A receive status. If ARST = 1, the host has written to its channel A register. The ARST flag is cleared when the target reads channel A.		
3–2	BXST	Channel B transmit status:		
		Bit 3	**Bit2**	**Channel B Transmit Status**
		0	0	buffer empty
		0	1	buffer less than half full
		1	1	buffer half or more full
		1	0	buffer full
5–4	BRST	Channel B receive status:		
		Bit 5	**Bit 4**	**Channel B Receive Status**
		0	0	buffer empty
		0	1	buffer less than half full
		1	1	buffer half or more full
		1	0	buffer full
6	USRBIN0	User discrete input bit 0		
7	USRBIN1	User discrete input bit 1		
8	—	Reserved		
9	—	Reserved		
10	—	Reserved		
11	—	Reserved		
12	USRBOT0	User discrete output bit 0		
13	USRBOT1	User discrete output bit 1		
14	USRBOT2	User discrete output bit 2		
15	AICRST	If AICRST = 0, the analog interface circuit is reset.		

Table 3–7 shows the I/O expansion connector J1 pin assignments.

Table 3–7. I/O Expansion Bus Signals

Row	Column A	Column B	Row	Column A	Column B
1	+5	+5	17	ID14	ID15
2	GND	XCLKOUT	18	GND	GND
3	GND	GND	19	$\overline{\text{INT0}}$	$\overline{\text{INT3}}$
4	XR/$\overline{\text{W}}$	$\overline{\text{XIOSTRB}}$	20	$\overline{\text{NMI}}$	IACK
5	$\overline{\text{XIS}}$	READY	21	$\overline{\text{X1BIO}}$	XF
6	XA0	XA1	22	TOUT	USRBIN0
7	XA2	XA3	23	USRBOT2	USRBIN1
8	GND	GND	24	USRBOT0	USRBOT1
9	ID0	ID1	25	GND	GND
10	ID2	ID3	26	CLKR0	CLKX0
11	ID4	ID5	27	DR0	DX0
12	ID6	ID7	28	$\overline{\text{FSR0}}$	$\overline{\text{FSX0}}$
13	GND	GND	29	GND	GND
14	ID8	ID9	30	+5	+5
15	ID10	ID11	31	–5	$\overline{\text{RESET}}$
16	ID12	ID13	32	+12	–12

3.4 Host/Target Communications

The ’C54x EVM has two independent communication channels, A and B, through which the host and target can communicate. Features of host and target communications include:

- Two-way, independent transfer. Both channels are bidirectional and fully independent.
- Equal status for host and target. There is no master/slave relationship between the host and the target. Both parties are equal under the specification.
- Polled or interrupt-driven channels. Both channels can be polled or interrupt-driven.

- Mutual status flag availability. In general, status flags for both the host and target sides are available to both the host and the target (the exception is BRST2, which is only available on the host side). See Table 3–3 on page 3-5 and Table 3–6 on page 3-8.

- Individually maskable interrupts on the target side. Target interrupts use separate interrupts: $\overline{\text{INT1}}$ for channel A and $\overline{\text{INT2}}$ for channel B. Both interrupts are individually maskable in the IMR register. Refer to the *TMS320C54x User's Guide* for more details.

- Individually enabled channels on the host side. Both channels must share a single host interrupt (selected by JP1; see Table 3–4 on page 3-6) but are individually enabled by the host by setting the ATIE bits for channel A or the BTIE bits for channel B in the host control register. See Table 3–3 on page 3-5.

- Unqueued interrupts. Interrupts are not queued. Both the host and target interrupts stay active until the host or target reads from the channel that caused the interrupt.

- Handshake-free. Handshaking between the host and target is not required to pass information. When the corresponding read/write operations are complete, the appropriate flags are changed automatically.

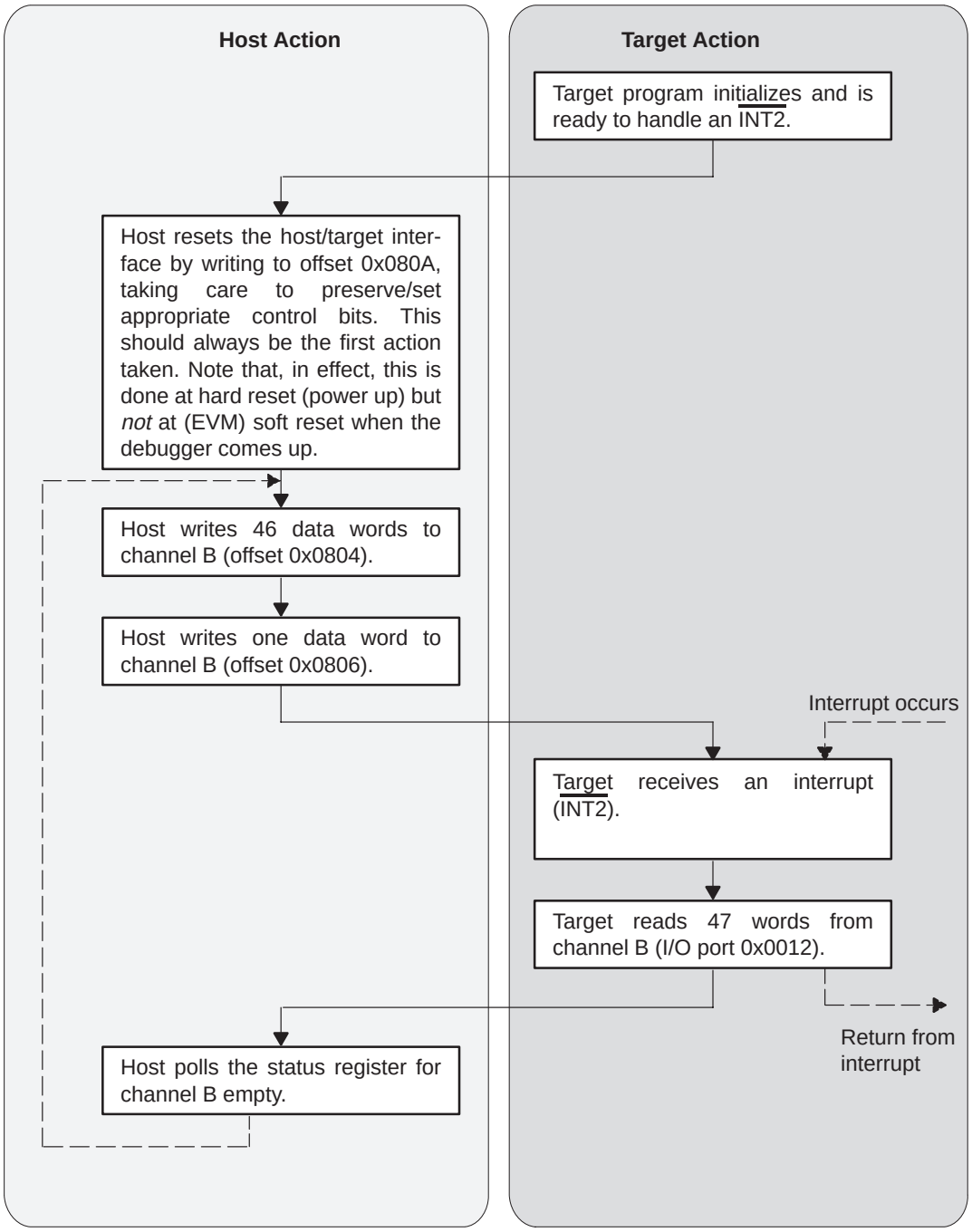
- Automatic overwrites. Writes to channel A, a single-word buffer, overwrite the previous contents of the buffer, regardless of whether the data has been read by the other side.

- Immediate data availability. Data written to channel B, a 64-word buffer, is available immediately to the other side on a first-in first-out (FIFO) basis. Data written when the buffer is full will be lost. Once the receiving side reads a word, that space is then available. Essentially, channel B is a bidirectional queue implemented as circular lists of length 64.

Example 3–1 shows the transmission of 47 16-bit words of data and commands from the host to the target. Example 3–2 illustrates the transfer of a continuous data stream from the host to the target; the results are sent back to the host.

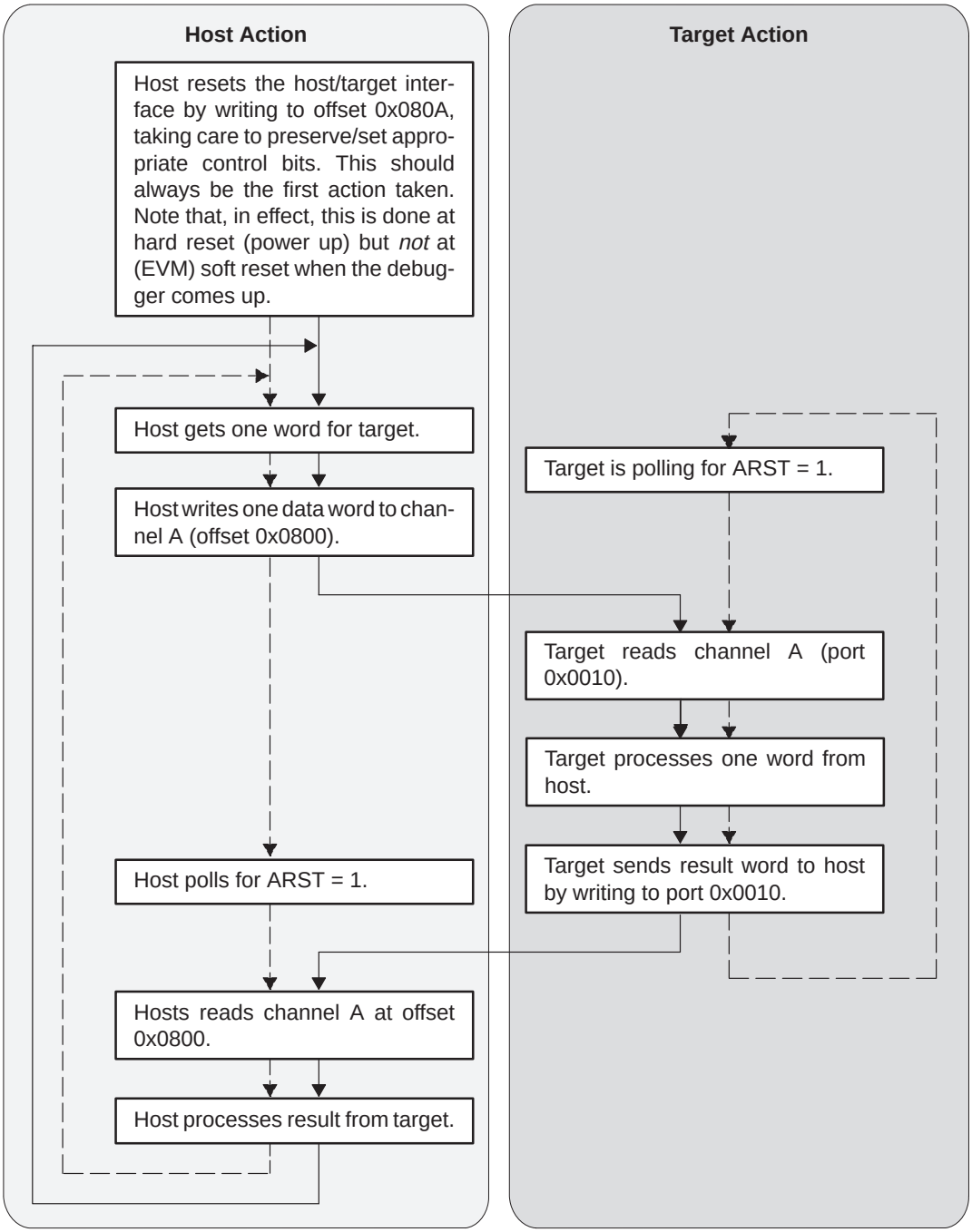
Since software development is constantly changing and growing, you can keep up with the latest developments by accessing the TI 24-hour electronic bulletin boards (BBS) with your PC and modem (in the United States and Canada). The TI DSP BBS phone number is (713) 274–2323. Protocol for the BBS is eight data bits, one stop bit, and no parity.

Example 3–1. Sending 47 16-Bit Words of Data and Commands From Host to Target Using Channel B



Note: The solid lines represent the data flow, and the dashed lines represent the program flow.

Example 3–2. Transmitting a Continuous Data Stream From Host to Target, Then Sending Results Back to Host Using Channel A



Note: The solid lines represent the data flow, and the dashed lines represent the program flow.

3.5 External Serial Port

The EVM provides one external serial port interface by means of a 10-pin header accessible at the rear of the PC. The serial port interface is connected directly to ’C541 serial port 0. Table 3–8 shows the pin assignments for the serial port interface.

Table 3–8. External Serial Port Signals

Pin	Signal
1	GND
2	$\overline{\text{FSR0}}$
3	CLKX0
4	DR0
5	DX0
6	$\overline{\text{X2BIO}}$
7	$\overline{\text{FSX0}}$
8	XF
9	CLKR0
10	GND

3.6 Analog Interface

The ’C54x EVM provides a single channel of voice-band analog input and output with programmable filtering, scaling, and sampling based on the TLC320AC01 analog interface circuit.

Features of the analog interface circuit include:

- 14-bit analog-to-digital/digital-to-analog (with 16 bits of dynamic range)
- Programmable sample rate (up to 43.2kHz)
- Programmable ±3V full scale input
- Programmable ±3V full scale output into 300 ohms

  or

  Can drive an 8-ohm speaker directly (jumper-selectable)
- Programmable anti-aliasing and reconstruction filters

The analog interface circuit is hard-wired in master/standalone mode with $\overline{\text{PWRDWN}}$ pulled high. The target control register (see Table 3–6 on page 3-8) controls the analog interface circuit reset and powers up low.

For applications that use the analog interface circuit, you must set the $\overline{\text{AICRST}}$ signal high before the analog interface circuit will operate. The analog interface circuit interfaces directly to the ’C54x via serial port 1 and generates $\overline{\text{FS}}$ and CLK for both transmit and receive. Clock signals for serial interface and filters are derived from the MCLK input, which comes from a 10.368-MHz crystal oscillator.

Analog input

The analog input for the analog interface circuit is taken from the external connector J4, prefiltered, buffered, referred to ADCMID and fed to the analog interface circuit IN differential inputs.

The analog interface circuit has two sets of inputs, IN and AUXIN, which are software-selectable. The IN signals are the default and are DC-coupled to the analog interface circuit with a gain of one. The AUXIN channel is AC-coupled with a gain of ten, which is useful for small signals such as those taken directly from a microphone.

On either channel, you can increase input sensitivity two or four times through software. For complete instructions on how to do this, refer to the *TLC320AC01 Data Manual.*

Analog output

Analog output (J5) is jumper-selectable (JP3) between one of two sources.

JP3 in the lower position (pins 2 and 3) selects ground-referenced analog interface circuit output. This output will drive ±3V full scale into a minimum of 300 ohms. You should find this more than adequate to drive most line-level audio component inputs.

You may reduce full-scale output two or four times through software. This measure is recommended when driving stereophonic equipment. For more complete instructions, refer to the *TLC320AC01 Data Manual*.

To drive speakers as low as 4 ohms directly, put JP3 in the upper position (pins 1 and 2) to select amplified output from a LM386 audio power amplifier.

3.7 Emulation Interface

The 74ACT8990 test bus controller (U23) provides the EVM emulation interface. The test bus controller connects directly to the ’C54x JTAG emulation port. The host interfaces to the test bus controller through 32 16-bit I/O mapped registers. The JTAG test clock (TCK) to the test bus controller is taken from the 10.368 crystal oscillator (U25) that drives the analog interface circuit.

The emulation interface and host/target communications interface can operate independently. The emulation interface features include:

- Emulation supported without external cabling, monitor software, or consumption of user resources
- Easy access to the ’C54x high-level language debuggers, factory testing, field diagnostics, etc.
- No system boot-ROM requirement. The host can download all necessary program files or data through the emulation port.

The EVM was designed so that you can debug a ’C54x application in conjunction with a host application. The only resource shared between the ’C54x debugger and host/target communications interface is the RESET bit of the host control register.

When you invoke a reset command within the debugger, the RESET bit is toggled high and then low to reset the EVM.

Do not reset the EVM outside of the debugger once you have invoked the debugger. To do so will take the ’C54x out of emulation mode.

3.8 Applications Considerations

The ’C54x EVM provides your application with numerous resources, some of which are shared.

Shared resources

In order to function properly, the application must coordinate use of these resources:

- **Host control register.** Your application and C source debugger share the RESET bit in this register. The reset command in the debugger cycles this bit high and then low, resetting the EVM. The application program on the host should not reset the EVM when the debugger is running. This will interfere with debugger operation.

- **$\overline{\text{BIO}}$ signal.** Three sources are gated together to form the ’C54x $\overline{\text{BIO}}$ signal: $\overline{\text{X1BIO}}$ located on connector J1, $\overline{\text{X2BIO}}$ located on connector J3, and $\overline{\text{HBIO}}$ located in the host control register. If any of these signals are low, the processor $\overline{\text{BIO}}$ signal is low, effectively disabling the other signals.

- **XF signal.** The XF signal from the ’C54x is routed to both J1 and J3 unbuffered.

- **Serial port.** The EVM provides one external serial port interface by means of a 10-pin header accessible at the rear of the PC. The ’C54x serial port 0 is routed to both J1 and J3.

Analog output

You can use the analog output connector, J5, to drive a low-impedance load such as a speaker (JP3 in upper position, pins 1 and 2) or a high-impedance load (300 ohms or more) such as a stereo amplifier.

When set to drive a high-impedance load (JP3 in lower position, pins 2 and 3), the full-scale output from the EVM is that of the analog interface circuit. This, by default, is $\pm$3V, which is well above what most audio components will handle without distortion or clipping.

To avoid potential distortion, reduce the full scale analog interface circuit output to $\pm$0.75V and ensure that JP3 is set to match the load being driven.

Parallel I/O

The parallel I/O available on J1 is buffered such that it requires at least two wait states in order to meet system timing requirements. External parallel I/O devices connected to J1 may require extra wait states or use of the READY signal. Be sure to observe system timing requirements when connecting external parallel I/O devices. For more information on timing and automatic wait-state generation, refer to the *TMS320C54x User's Guide*.

Appendix A

TMS320C54x EVM PAL Equations

This appendix contains the programmable logic source for the ’C54x EVM PALS. These PAL equations were compiled with DATA I/O ABEL version 4.0 with per-pin autopolarity optimization.

Topic **Page**

A.1 Host Decode

```
module Host_Address_Decode

title ’
DWG. NAME   TMS320C5xx EVM
ASSY #      D600100–0001
PAL #       U17, 100*
COMPANY     TEXAS INSTRUMENTS INCORPORATED
ENGR        GREG LOREK
DATE        12/21/94’

    u17     device  ’P16L8’;

    ha0     pin     1;  ”inputs
    ha5     pin     2;
    ha6     pin     3;
    ha7     pin     4;
    ha8     pin     5;
    ha9     pin     6;
    ha10    pin     7;
    ha11    pin     8;
    bhe_    pin     9;
    haen_   pin     11;
    mapsel0 pin     13;
    mapsel1 pin     14;

    io16_   pin     12; ”outputs
    usrsel_ pin     16;
    emusel_ pin     17;
    card_   pin     18;

    addr    = [ha11,ha10,ha9,ha8,ha7,ha6,ha5,.x.,.x.,.x.,.x.,.x.];
    map     = [mapsel0,mapsel1];

@radix 16;
equations
    !usrsel_ = !haen_ & !bhe_ & !ha0 &
      ( ((map==0) & (addr >= 0a40) & (addr < 0a60))
      # ((map==1) & (addr >= 0a80) & (addr < 0aa0))
      # ((map==2) & (addr >= 0b20) & (addr < 0b40))
      # ((map==3) & (addr >= 0b40) & (addr < 0b60)) );

    !emusel_ = !haen_ & !bhe_ & !ha0 &
      ( ((map==0) & (addr >= 0240) & (addr < 0260))
      # ((map==0) & (addr >= 0640) & (addr < 0660))
      # ((map==1) & (addr >= 0280) & (addr < 02a0))
      # ((map==1) & (addr >= 0680) & (addr < 06a0))
      # ((map==2) & (addr >= 0320) & (addr < 0340))
      # ((map==2) & (addr >= 0720) & (addr < 0740))
      # ((map==3) & (addr >= 0340) & (addr < 0360))
      # ((map==3) & (addr >= 0740) & (addr < 0760)) );

    !card_ = !usrsel_ # !emusel_;

    io16_ = 0;
    io16_.oe = !card_;
```

```
test_vectors
    ([haen_, bhe_, map, addr, ha0] -> [usrsel_, emusel_, card_, io16_])
     [  1  ,  0  ,  0 , 0a40,  0 ] -> [  1    ,   1    ,   1  ,  .z. ];
     [  0  ,  1  ,  0 , 0a40,  0 ] -> [  1    ,   1    ,   1  ,  .z. ];
     [  0  ,  0  ,  0 , 0a41,  1 ] -> [  1    ,   1    ,   1  ,  .z. ];

     [  0  ,  0  ,  0 , 0a40,  0 ] -> [  0    ,   1    ,   0  ,   0  ];
     [  0  ,  0  ,  1 , 0a80,  0 ] -> [  0    ,   1    ,   0  ,   0  ];
     [  0  ,  0  ,  2 , 0b20,  0 ] -> [  0    ,   1    ,   0  ,   0  ];
     [  0  ,  0  ,  3 , 0b40,  0 ] -> [  0    ,   1    ,   0  ,   0  ];

     [  0  ,  0  ,  0 , 0240,  0 ] -> [  1    ,   0    ,   0  ,   0  ];
     [  0  ,  0  ,  0 , 0640,  0 ] -> [  1    ,   0    ,   0  ,   0  ];
     [  0  ,  0  ,  1 , 0280,  0 ] -> [  1    ,   0    ,   0  ,   0  ];
     [  0  ,  0  ,  1 , 0680,  0 ] -> [  1    ,   0    ,   0  ,   0  ];
     [  0  ,  0  ,  2 , 0320,  0 ] -> [  1    ,   0    ,   0  ,   0  ];
     [  0  ,  0  ,  2 , 0720,  0 ] -> [  1    ,   0    ,   0  ,   0  ];
     [  0  ,  0  ,  3 , 0340,  0 ] -> [  1    ,   0    ,   0  ,   0  ];
     [  0  ,  0  ,  3 , 0740,  0 ] -> [  1    ,   0    ,   0  ,   0  ];
end
```

A.2 Host Control Strobes

```
module Host_Control_Strobes

title ’
DWG. NAME   TMS320C5xx EVM
ASSY #      D600100–0001
PAL #       U13, 100*
COMPANY     TEXAS INSTRUMENTS INCORPORATED
ENGR        GREG LOREK
DATE        12/21/94’

    u13     device  ’P16L8’;

    ha1     pin     1;  ”inputs
    ha2     pin     2;
    ha3     pin     3;
    ha4     pin     4;
    usrsel_ pin     5;
    iord_   pin     6;
    iowr_   pin     7;
    hreset_ pin     8;

    hard_   pin     12; ”outputs
    hawr_   pin     13;
    hbrd_   pin     14;
    hbwr_   pin     15;
    hbwrs_  pin     16;
    hstrd_  pin     17;
    hstwr_  pin     18;
    hstwrs_ pin     19;

    addr = [ha4,ha3,ha2,ha1,.x.];   ”shorten for test vectors
    ard = hard_;
    awr = hawr_;
    brd = hbrd_;
    bwr = hbwr_;
    bws = hbwrs_;
    srd = hstrd_;
    swr = hstwr_;
    sws = hstwrs_;

@radix 16;
equations
    !hard_  = !usrsel_ & !iord_ &  (addr == 00);
    !hawr_  = !usrsel_ & !iowr_ &  (addr == 00);
    !hbrd_  = !usrsel_ & !iord_ & ((addr == 04) # (addr == 06));
    !hbwr_  = !usrsel_ & !iowr_ & ((addr == 04) # (addr == 06));
    !hbwrs_ = !usrsel_ & !iowr_ &  (addr == 06);
    !hstrd_ = !usrsel_ & !iord_ & ((addr == 08) # (addr == 0a));
    !hstwr_ = !usrsel_ & !iowr_ & ((addr == 08) # (addr == 0a));
    !hstwrs_= !usrsel_ & !iowr_ &  (addr == 0a) # !hreset_;
```

```
test_vectors
    ([usrsel_,iord_,iowr_,addr,hreset_] -> [ard,awr,brd,bwr,bws,srd,swr,sws])
     [  1    ,  0  ,  0  ,  0 ,   1   ] -> [ 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 ];
     [  0    ,  1  ,  1  ,  0 ,   1   ] -> [ 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 ];

     [  0    ,  0  ,  1  ,  0 ,   1   ] -> [ 0 , 1 , 1 , 1 , 1 , 1 , 1 , 1 ];
     [  0    ,  1  ,  0  ,  0 ,   1   ] -> [ 1 , 0 , 1 , 1 , 1 , 1 , 1 , 1 ];
     [  0    ,  0  ,  0  ,  2 ,   1   ] -> [ 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 ];
     [  0    ,  0  ,  1  ,  4 ,   1   ] -> [ 1 , 1 , 0 , 1 , 1 , 1 , 1 , 1 ];
     [  0    ,  1  ,  0  ,  4 ,   1   ] -> [ 1 , 1 , 1 , 0 , 1 , 1 , 1 , 1 ];
     [  0    ,  0  ,  1  ,  6 ,   1   ] -> [ 1 , 1 , 0 , 1 , 1 , 1 , 1 , 1 ];
     [  0    ,  1  ,  0  ,  6 ,   1   ] -> [ 1 , 1 , 1 , 0 , 0 , 1 , 1 , 1 ];
     [  0    ,  0  ,  1  ,  8 ,   1   ] -> [ 1 , 1 , 1 , 1 , 1 , 0 , 1 , 1 ];
     [  0    ,  1  ,  0  ,  8 ,   1   ] -> [ 1 , 1 , 1 , 1 , 1 , 1 , 0 , 1 ];
     [  0    ,  0  ,  1  , 0a ,   1   ] -> [ 1 , 1 , 1 , 1 , 1 , 0 , 1 , 1 ];
     [  0    ,  1  ,  0  , 0a ,   1   ] -> [ 1 , 1 , 1 , 1 , 1 , 1 , 0 , 0 ];
     [  1    ,  1  ,  1  ,  0 ,   0   ] -> [ 1 , 1 , 1 , 1 , 1 , 1 , 1 , 0 ];
end
```

A.3 Target Control Strobes

```
module Target_Control_Strobes

title ’
DWG. NAME   TMS320C5xx EVM
ASSY #      D600100–0001
PAL #       U19, 100*
COMPANY     TEXAS INSTRUMENTS INCORPORATED
ENGR        GREG LOREK
DATE        12/21/94’

    u19     device  ’P16L8’;

    ta0     pin     1;  ”inputs
    ta1     pin     2;
    ta2     pin     3;
    ta3     pin     4;
    ta4     pin     5;
    iostb_  pin     6;
    is_     pin     7;
    rw_     pin     8;

    tard    pin     12; ”outputs
    tawr_   pin     13;
    tawrs_  pin     14;
    tbrd_   pin     15;
    tbwr_   pin     16;
    tbwrs_  pin     17;
    tstrd_  pin     18;
    tstwr_  pin     19;

    addr = [ta4,ta3,ta2,ta1,ta0];   ”shorten names for test vectors
    ard = tard;
    awr = tawr_;
    aws = tawrs_;
    brd = tbrd_;
    bwr = tbwr_;
    bws = tbwrs_;
    srd = tstrd_;
    swr = tstwr_;

@radix 16;
equations
     tard   = !iostb_ &  rw_ & ((addr == 10) # (addr == 11));
    !tawr_  = !iostb_ & !rw_ & ((addr == 10) # (addr == 11));
    !tawrs_ = !iostb_ & !rw_ & (               (addr == 11));
    !tbrd_  = !iostb_ &  rw_ & ((addr == 12) # (addr == 13));
    !tbwr_  = !iostb_ & !rw_ & ((addr == 12) # (addr == 13));
    !tbwrs_ = !iostb_ & !rw_ & (               (addr == 13));
    !tstrd_ = !iostb_ &  rw_ & ((addr == 14));
    !tstwr_ = !iostb_ & !rw_ & ((addr == 14));
```

```
test_vectors
    ([iostb_,rw_, addr] -> [ard,awr,aws,brd,bwr,bws,srd,swr])
     [  1   ,.x.,  .x.] -> [ 0 , 1 , 1 , 1 , 1 , 1 , 1 , 1 ];

     [  0   , 1 ,  10 ] -> [ 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 ];
     [  0   , 1 ,  11 ] -> [ 1 , 1 , 1 , 1 , 1 , 1 , 1 , 1 ];
     [  0   , 0 ,  10 ] -> [ 0 , 0 , 1 , 1 , 1 , 1 , 1 , 1 ];
     [  0   , 0 ,  11 ] -> [ 0 , 0 , 0 , 1 , 1 , 1 , 1 , 1 ];

     [  0   , 1 ,  12 ] -> [ 0 , 1 , 1 , 0 , 1 , 1 , 1 , 1 ];
     [  0   , 1 ,  13 ] -> [ 0 , 1 , 1 , 0 , 1 , 1 , 1 , 1 ];
     [  0   , 0 ,  12 ] -> [ 0 , 1 , 1 , 1 , 0 , 1 , 1 , 1 ];
     [  0   , 0 ,  13 ] -> [ 0 , 1 , 1 , 1 , 0 , 0 , 1 , 1 ];

     [  0   , 1 ,  14 ] -> [ 0 , 1 , 1 , 1 , 1 , 1 , 0 , 1 ];
     [  0   , 0 ,  14 ] -> [ 0 , 1 , 1 , 1 , 1 , 1 , 1 , 0 ];

end
```

A.4 Miscellaneous Logic

```
module Misc_Logic

title ’
DWG. NAME   TMS320C5xx EVM
ASSY #      D600100–0001
PAL #       U22, 100*
COMPANY     TEXAS INSTRUMENTS INCORPORATED
ENGR        GREG LOREK
DATE        12/21/94’

    u22     device  ’P16L8’;

    iord_   pin     1;  ”inputs
    iowr_   pin     2;
    emusel_ pin     3;
    card_   pin     4;
    tbcevt3 pin     5;
    x1bio_  pin     6;
    hbio_   pin     7;
    x2bio_  pin     8;
    hreset  pin     9;

    tbcrd_  pin     12; ”outputs
    tbcwr_  pin     13;
    dira    pin     14;
    dirb_   pin     15;
    ttrst_  pin     16;
    bio_    pin     17;
    hreset_ pin     19;

equations
    !tbcrd_ = !iord_ & !emusel_;
    !tbcwr_ = !iowr_ & !emusel_;
     dira   = !iord_ & !card_;
    !dirb_  = !iowr_ & !card_ # card_;

    !ttrst_ = tbcevt3;
    !bio_   = !x1bio_ # !x2bio_ # !hbio_;

    hreset_ = !hreset;
```

```
test_vectors
    ([iord_,iowr_,emusel_,card_] -> [tbcrd_,tbcwr_,dira,dirb_])
     [  1  ,  1  ,   1   ,  1  ] -> [  1   ,  1   ,  0 ,  0  ];
     [  0  ,  1  ,   1   ,  1  ] -> [  1   ,  1   ,  0 ,  0  ];
     [  1  ,  0  ,   1   ,  1  ] -> [  1   ,  1   ,  0 ,  0  ];

     [  1  ,  1  ,   1   ,  0  ] -> [  1   ,  1   ,  0 ,  1  ];
     [  0  ,  1  ,   1   ,  0  ] -> [  1   ,  1   ,  1 ,  1  ];
     [  1  ,  0  ,   1   ,  0  ] -> [  1   ,  1   ,  0 ,  0  ];

     [  1  ,  1  ,   0   ,  0  ] -> [  1   ,  1   ,  0 ,  1  ];
     [  0  ,  1  ,   0   ,  0  ] -> [  0   ,  1   ,  1 ,  1  ];
     [  1  ,  0  ,   0   ,  0  ] -> [  1   ,  0   ,  0 ,  0  ];

test_vectors
    ([tbcevt3] -> [ttrst_])
     [   1   ] -> [   0  ];
     [   0   ] -> [   1  ];

test_vectors
    ([x1bio_,x2bio_,hbio_] -> [bio_])
     [  1   ,  1   ,  1  ] -> [ 1  ];
     [  0   ,  1   ,  1  ] -> [ 0  ];
     [  1   ,  0   ,  1  ] -> [ 0  ];
     [  1   ,  1   ,  0  ] -> [ 0  ];

test_vectors
    ([hreset] -> [hreset_])
     [   1  ] -> [   0   ];
     [   0  ] -> [   1   ];

end
```

A.5 Host Interrupt Logic

```
module Host_Interrupt

title ’
DWG. NAME   TMS320C5xx EVM
ASSY #      D600100–0001
PAL #       U18, 100*
COMPANY     TEXAS INSTRUMENTS INCORPORATED
ENGR        GREG LOREK
DATE        12/21/94’

    u18     device  ’P16R4’;

    clk     pin     1;      ”inputs
    atie    pin     2;
    btie    pin     3;
    tard    pin     4;
    tawr_   pin     5;
    tawrs_  pin     6;
    tbwrs_  pin     7;
    hard_   pin     8;
    hbrd_   pin     9;
    oe_     pin     11;
    hstwrs_ pin     12;
    thfull_ pin     13;

    q0      pin     14;     ”outputs
    harst   pin     15;
    q2      pin     16;
    hintq   pin     17;
    hint    pin     18;
    tardf_  pin     19;

    flag_clear  =   ^b00;   ”states
    write       =   ^b01;
    flag_set    =   ^b11;
    read        =   ^b10;

    idle        =   ^b00;
    aint        =   ^b01;
    wait        =   ^b11;
    bint        =   ^b10;

equations
    [q0,harst,q2,hintq].c = clk;
    [q0,harst,q2,hintq].oe = !oe_;

    !tardf_ = tard # !hstwrs_;

test_vectors
    ([tard, hstwrs_] -> [tardf_])
     [  1  ,   0   ] -> [  0   ]; ”1
     [  1  ,   1   ] -> [  0   ];
     [  0  ,   0   ] -> [  0   ];
     [  0  ,   1   ] -> [  1   ];
```

```
state_diagram [harst,q0]
    State flag_clear:
        if (!hstwrs_) then goto flag_clear else
        if (!tawr_) then goto write
                    else goto flag_clear;

    State write:
        if (!hstwrs_) then goto flag_clear else
        if (!tawr_) then goto write
                     else goto flag_set;

    State flag_set:
        if (!hstwrs_) then goto flag_clear else
        if (!hard_) then goto read
                    else goto flag_set;

    State read:
        if (!hstwrs_) then goto flag_clear else
        if (!hard_) then goto read
                     else goto flag_clear;

test_vectors
    ([clk,oe_,hstwrs_,tawr_,hard_] -> [harst,q0])
     [.c., 0 ,   0   ,  1  ,  1  ] -> flag_clear;       "5
     [.c., 0 ,   1   ,  1  ,  1  ] -> flag_clear;
     [.c., 0 ,   1   ,  0  ,  1  ] -> write;
     [.c., 0 ,   1   ,  0  ,  1  ] -> write;
     [.c., 0 ,   1   ,  1  ,  1  ] -> flag_set;
     [.c., 0 ,   1   ,  1  ,  1  ] -> flag_set;
     [.c., 0 ,   1   ,  1  ,  0  ] -> read;
     [.c., 0 ,   1   ,  1  ,  0  ] -> read;
     [.c., 0 ,   1   ,  1  ,  1  ] -> flag_clear;

     [.c., 0 ,   0   ,  0  ,  1  ] -> flag_clear;       "14
     [.c., 0 ,   1   ,  1  ,  1  ] -> flag_clear;
     [.c., 0 ,   1   ,  0  ,  1  ] -> write;
     [.c., 0 ,   0   ,  0  ,  1  ] -> flag_clear;
     [.c., 0 ,   0   ,  0  ,  1  ] -> flag_clear;

     [.c., 0 ,   1   ,  0  ,  1  ] -> write;           "19
     [.c., 0 ,   1   ,  1  ,  1  ] -> flag_set;
     [.c., 0 ,   1   ,  1  ,  1  ] -> flag_set;
     [.c., 0 ,   0   ,  1  ,  0  ] -> flag_clear;

     [.c., 0 ,   1   ,  0  ,  1  ] -> write;           "23
     [.c., 0 ,   1   ,  1  ,  1  ] -> flag_set;
     [.c., 0 ,   0   ,  1  ,  1  ] -> flag_clear;

     [.c., 0 ,   1   ,  0  ,  1  ] -> write;           "26
     [.c., 0 ,   1   ,  0  ,  1  ] -> write;
     [.c., 0 ,   1   ,  1  ,  1  ] -> flag_set;
     [.c., 0 ,   1   ,  1  ,  1  ] -> flag_set;
     [.c., 0 ,   1   ,  1  ,  0  ] -> read;
     [.c., 0 ,   0   ,  1  ,  0  ] -> flag_clear;
```

```
state_diagram [hintq,q2]
    State idle:
        hint.oe = 1;
        hint = 0;
        if (!hstwrs_) then goto idle else
        if (atie & !tawr_ # !tawrs_) then goto aint else
        if (btie & (!tbwrs_ # !thfull_)) then goto bint else
        goto idle;

    State aint:
        hint.oe = 0;
        hint = 1;
        if (!hstwrs_) then goto wait else
        if (!hard_) then goto wait else
        goto aint;

    State bint:
        hint.oe = 0;
        hint = 1;
        if (!hstwrs_) then goto wait else
        if (!hbrd_) then goto wait else
        goto bint;

    State wait:
        hint.oe = 0;
        hint = 1;
        if (!hstwrs_) then goto wait else
        if (!hard_ # !hbrd_) then goto wait else
        if ( (atie # btie) & !(btie & !thfull_) ) then goto idle else
        goto wait
```

```
test_vectors
    ([clk,oe_,hstwrs_,atie,tawr_,tawrs_,hard_,hbrd_] -> [[hintq,q2],hint])
     [.c., 0 ,   0   ,  1 ,  1  ,  1   ,  1  ,  1  ] -> [ .x.,.x.];       "32
     [.c., 0 ,   1   ,  1 ,  1  ,  1   ,  1  ,  1  ] -> [idle, 0 ];

     [.c., 0 ,   0   ,  1 ,  1  ,  1   ,  1  ,  1  ] -> [idle, 0 ];

     [.c., 0 ,   1   ,  0 ,  1  ,  0   ,  0  ,  1  ] -> [aint,.z.];
     [.c., 0 ,   1   ,  0 ,  1  ,  1   ,  1  ,  1  ] -> [aint,.z.];
     [.c., 0 ,   1   ,  0 ,  1  ,  1   ,  0  ,  1  ] -> [wait,.z.];
     [.c., 0 ,   1   ,  0 ,  1  ,  1   ,  0  ,  1  ] -> [wait,.z.];
     [.c., 0 ,   1   ,  0 ,  1  ,  1   ,  1  ,  0  ] -> [wait,.z.];
     [.c., 0 ,   1   ,  1 ,  1  ,  1   ,  1  ,  1  ] -> [idle, 0 ];

     [.c., 0 ,   1   ,  0 ,  0  ,  1   ,  1  ,  1  ] -> [idle, 0 ];       "41
     [.c., 0 ,   1   ,  1 ,  0  ,  1   ,  0  ,  1  ] -> [aint,.z.];
     [.c., 0 ,   1   ,  1 ,  0  ,  1   ,  1  ,  1  ] -> [aint,.z.];
     [.c., 0 ,   1   ,  1 ,  1  ,  1   ,  0  ,  1  ] -> [wait,.z.];
     [.c., 0 ,   1   ,  1 ,  1  ,  1   ,  1  ,  1  ] -> [idle, 0 ];

     [.c., 0 ,   1   ,  1 ,  0  ,  1   ,  1  ,  1  ] -> [aint,.z.];
     [.c., 0 ,   0   ,  1 ,  1  ,  1   ,  1  ,  1  ] -> [wait,.z.];
     [.c., 0 ,   0   ,  1 ,  1  ,  1   ,  1  ,  1  ] -> [wait,.z.];
     [.c., 0 ,   1   ,  1 ,  1  ,  1   ,  1  ,  1  ] -> [idle, 0 ];

test_vectors
([clk,oe_,hstwrs_,btie,tbwrs_,thfull_,tawrs_,hard_,hbrd_] -> [[hintq,q2],hint])
 [.c., 0 ,   1   ,  0 ,  1   ,   1   ,  1   ,  1  ,  1  ] -> [idle, 0 ];   "50
 [.c., 0 ,   1   ,  1 ,  0   ,   1   ,  1   ,  1  ,  1  ] -> [bint,.z.];
 [.c., 0 ,   1   ,  1 ,  1   ,   1   ,  1   ,  1  ,  1  ] -> [bint,.z.];
 [.c., 0 ,   1   ,  1 ,  1   ,   1   ,  1   ,  1  ,  0  ] -> [wait,.z.];
 [.c., 0 ,   1   ,  1 ,  1   ,   1   ,  1   ,  1  ,  0  ] -> [wait,.z.];
 [.c., 0 ,   1   ,  1 ,  1   ,   1   ,  1   ,  1  ,  1  ] -> [idle, 0 ];

 [.c., 0 ,   1   ,  0 ,  1   ,   0   ,  1   ,  1  ,  1  ] -> [idle, 0 ];   "56
 [.c., 0 ,   1   ,  1 ,  1   ,   0   ,  1   ,  1  ,  1  ] -> [bint,.z.];
 [.c., 0 ,   0   ,  1 ,  1   ,   1   ,  1   ,  1  ,  1  ] -> [wait,.z.];
 [.c., 0 ,   0   ,  1 ,  1   ,   1   ,  1   ,  1  ,  1  ] -> [wait,.z.];
 [.c., 0 ,   1   ,  1 ,  1   ,   0   ,  1   ,  1  ,  0  ] -> [wait,.z.];
 [.c., 0 ,   1   ,  1 ,  1   ,   0   ,  1   ,  1  ,  1  ] -> [wait,.z.];
 [.c., 0 ,   1   ,  1 ,  1   ,   1   ,  1   ,  1  ,  1  ] -> [idle, 0 ];

end
```

A.6 Target Interrupt Logic

```
module Target_Interrupt

title ’
DWG. NAME   TMS320C5xx EVM
ASSY #      D600100–0001
PAL #       U14, 100*
COMPANY     TEXAS INSTRUMENTS INCORPORATED
ENGR        GREG LOREK
DATE        12/21/94’

    u14     device  ’P16R4’;

    clk     pin     1;      ”inputs
    tbwrs_  pin     2;
    hbrd_   pin     3;
    hstwrs_ pin     4;
    hbwrs_  pin     5;
    tbrd_   pin     6;
    htfull_ pin     7;
    htepty_ pin     8;
    thfull_ pin     9;
    oe_     pin     11;
    thepty_ pin     12;

    int2_   pin     14;     ”outputs
    q1      pin     15;
    q2      pin     16;
    hbrst2  pin     17;

    hbxst0  pin     18;
    tbxst0  pin     19;

    idle        =   ^b11;   ”states
    write       =   ^b01;
    int         =   ^b00;
    full        =   ^b10;

    flag_clear  =   ^b00;
    flag_set    =   ^b01;
    wait        =   ^b11;
    read        =   ^b10;

equations
    [int2_,q1,q2,hbrst2].c  = clk;
    [int2_,q1,q2,hbrst2].oe = !oe_;

    tbxst0 = thfull_ & thepty_;
    hbxst0 = htfull_ & htepty_;

test_vectors
    ([thfull_,thepty_] -> [tbxst0])
     [   1   ,   1   ] -> [   1  ];       ”1
     [   0   ,   1   ] -> [   0  ];
     [   1   ,   0   ] -> [   0  ];
     [   0   ,   0   ] -> [   0  ];
```

```
test_vectors
    ([htfull_,htepty_] -> [hbxst0])
     [   1   ,   1   ] -> [   1  ];        "5
     [   0   ,   1   ] -> [   0  ];
     [   1   ,   0   ] -> [   0  ];
     [   0   ,   0   ] -> [   0  ];

state_diagram [hbrst2,q2]
    State flag_clear:
        if (!hstwrs_) then goto flag_clear else
        if (!tbwrs_) then goto flag_set else
        goto flag_clear;

    State flag_set:
        if (!hstwrs_) then goto flag_clear else
        if (!tbwrs_) then goto flag_set else
        if ( hbrd_) then goto wait else
        goto flag_set;

    State wait:
        if (!hstwrs_) then goto flag_clear else
        if (!hbrd_) then goto read else
        goto wait;

    State read:
        if (!hstwrs_) then goto flag_clear else
        if (!hbrd_) then goto read else
        goto flag_clear;

test_vectors
    ([clk,oe_,hstwrs_,tbwrs_,hbrd_] -> [hbrst2,q2])
     [.c., 0 ,  0    ,  1   ,  1  ] -> flag_clear;  "9
     [.c., 0 ,  1    ,  1   ,  1  ] -> flag_clear;
     [.c., 0 ,  1    ,  0   ,  1  ] -> flag_set;
     [.c., 0 ,  1    ,  0   ,  1  ] -> flag_set;
     [.c., 0 ,  1    ,  1   ,  0  ] -> flag_set;
     [.c., 0 ,  1    ,  1   ,  1  ] -> wait;
     [.c., 0 ,  1    ,  1   ,  1  ] -> wait;
     [.c., 0 ,  1    ,  1   ,  0  ] -> read;
     [.c., 0 ,  1    ,  1   ,  0  ] -> read;
     [.c., 0 ,  1    ,  1   ,  1  ] -> flag_clear;

     [.c., 0 ,  1    ,  0   ,  1  ] -> flag_set;   "19
     [.c., 0 ,  0    ,  0   ,  1  ] -> flag_clear;

     [.c., 0 ,  1    ,  0   ,  1  ] -> flag_set;
     [.c., 0 ,  1    ,  1   ,  1  ] -> wait;
     [.c., 0 ,  0    ,  0   ,  1  ] -> flag_clear;

     [.c., 0 ,  1    ,  0   ,  1  ] -> flag_set;
     [.c., 0 ,  1    ,  1   ,  1  ] -> wait;
     [.c., 0 ,  0    ,  0   ,  0  ] -> flag_clear;

     [.c., 0 ,  1    ,  0   ,  1  ] -> flag_set;
     [.c., 0 ,  1    ,  1   ,  1  ] -> wait;
     [.c., 0 ,  1    ,  1   ,  0  ] -> read;
     [.c., 0 ,  0    ,  1   ,  0  ] -> flag_clear;
```

```
state_diagram [q1,int2_]
    State idle:
        if (!hstwrs_) then goto idle else
        if (!htfull_) then goto full else
        if (!hbwrs_) then goto write else
        goto idle;

    State write:
        if (!hstwrs_) then goto idle else
        if (!hbwrs_) then goto write else
        goto int;

    State int:
        if (!hstwrs_) then goto idle else
        if (!htfull_) then goto full else
        if (!tbrd_) then goto idle else
        goto int;

    State full:
        if (!htfull_) then goto full else
"        if (!hstwrs_) then goto idle else    !hstwrs_ => htfull_
        goto idle;

test_vectors
    ([clk,oe_,hstwrs_,htfull_,hbwrs_,tbrd_] -> [q1,int2_])
     [.c., 0 ,   0   ,   1   ,   1  ,  1  ] -> idle;        "31
     [.c., 0 ,   1   ,   1   ,   1  ,  1  ] -> idle;
     [.c., 0 ,   1   ,   1   ,   0  ,  1  ] -> write;
     [.c., 0 ,   1   ,   1   ,   0  ,  1  ] -> write;
     [.c., 0 ,   1   ,   1   ,   1  ,  1  ] -> int;
     [.c., 0 ,   1   ,   1   ,   1  ,  1  ] -> int;
     [.c., 0 ,   1   ,   1   ,   1  ,  0  ] -> idle;

     [.c., 0 ,   1   ,   1   ,   0  ,  1  ] -> write;       "38
     [.c., 0 ,   1   ,   1   ,   1  ,  0  ] -> int;
     [.c., 0 ,   1   ,   0   ,   1  ,  0  ] -> full;
     [.c., 0 ,   1   ,   0   ,   1  ,  1  ] -> full;
     [.c., 0 ,   1   ,   1   ,   1  ,  1  ] -> idle;

     [.c., 0 ,   1   ,   0   ,   1  ,  1  ] -> full;        "43
     [.c., 0 ,   1   ,   1   ,   1  ,  1  ] -> idle;

     [.c., 0 ,   1   ,   1   ,   0  ,  1  ] -> write;
     [.c., 0 ,   1   ,   1   ,   1  ,  0  ] -> int;
     [.c., 0 ,   0   ,   1   ,   1  ,  1  ] -> idle;

     [.c., 0 ,   1   ,   1   ,   0  ,  1  ] -> write;
     [.c., 0 ,   0   ,   1   ,   1  ,  1  ] -> idle;

     [.c., 0 ,   1   ,   0   ,   1  ,  1  ] -> full;        "50
     [.c., 0 ,   0   ,   0   ,   1  ,  1  ] -> full;
     [.c., 0 ,   0   ,   1   ,   1  ,  1  ] -> idle;

end
```

Appendix B

TMS320C54x EVM Schematics

This appendix contains the schematics for the TMS320C54x EVM.

NOTES, UNLESS OTHERWISE SPECIFIED:

1. RESISTANCE VALUES ARE IN OHMS.

2. CAPACITANCE VALUES ARE IN MICROFARADS.

3. HIGHEST REFERENCE DESIGNATOR USED:

A. CAPACITORS	C51
B. RESISTORS	R54
C. INDUCTORS	L2
D. DIODES	D2
E. IC'S	U29
F. CONNECTORS	J5
G. JUMPERS	JP3
H. SWITCHES	S1

4. SWITCH S1 SELECTION

1	2	I/O BASE ADDRESS SELECTED
ON	ON	0x0240-0x025F
ON	OFF	0x0280-0x029F
OFF	ON	0x0320-0x033F
OFF	OFF	0x0340-0x035F

5. JP1 SELECTS HOST INTERRUPT SIGNAL

1-2 SELECTS INTERRUPT IRQ5
3-4 SELECTS INTERRUPT IRQ7
5-6 SELECTS INTERRUPT IRQ10
7-8 SELECTS INTERRUPT IRQ11

6. JP2 DETERMINES MAJOR HARDWARE REVISIONS.

SEE ASSEMBLY DRAWINGS FOR REVISION INFORMATION.

7. JP3 SELECTS AIC OUTPUT MODE

1-2 SELECTS 8 OHM SPEAKER COMPATIBLE SIGNAL
2-3 SELECTS 600 OHM LINE COMPATIBLE SIGNAL

8. R11, R12, R13, R14, R15, AND R16 SELECT CLOCK MODE.
INSTALL ONLY R11, R14 AND R16 FOR 40 MIPS OPERATION.

REVISIONS

REV	DESCRIPTION	DATE	APPROVED
A	ECN563502(E) M.DAWKINS 6-23-95	8-31-95	J. CLARK
B	ECN563626(E) M. DAWKINS 8-31-95		J. CLARK
	FORMAL RELEASE		

REVISION STATUS OF SHEETS

REV									
SH									
REV	*	B	*	A	*	A			
SH	8	9	10	11	12	13		D600100	
REV	B	A	B	*	*	B	A	NEXT ASSY	USED ON
SH	1	2	3	4	5	6	7	APPLICATION	

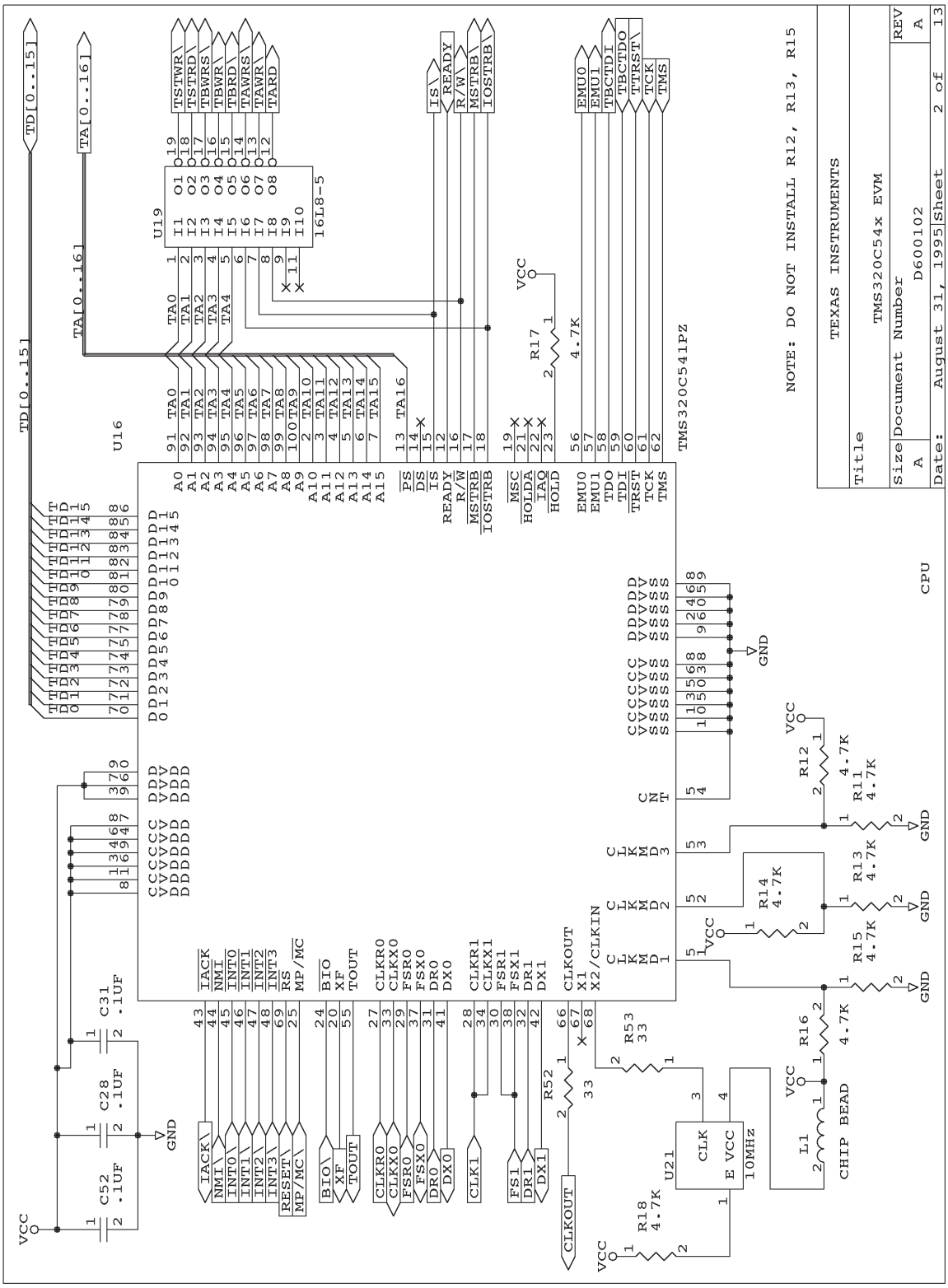
DWN	G. LOREK	DATE 3-3-95
CHK	M. DAWKINS	DATE
ENGR	G. LOREK	DATE
ENGR-MGR	T. COOMES	DATE
QA	J.J. NANEZ	DATE
MFG	M. JACKSON	DATE
RLSE	J. CLARK	DATE

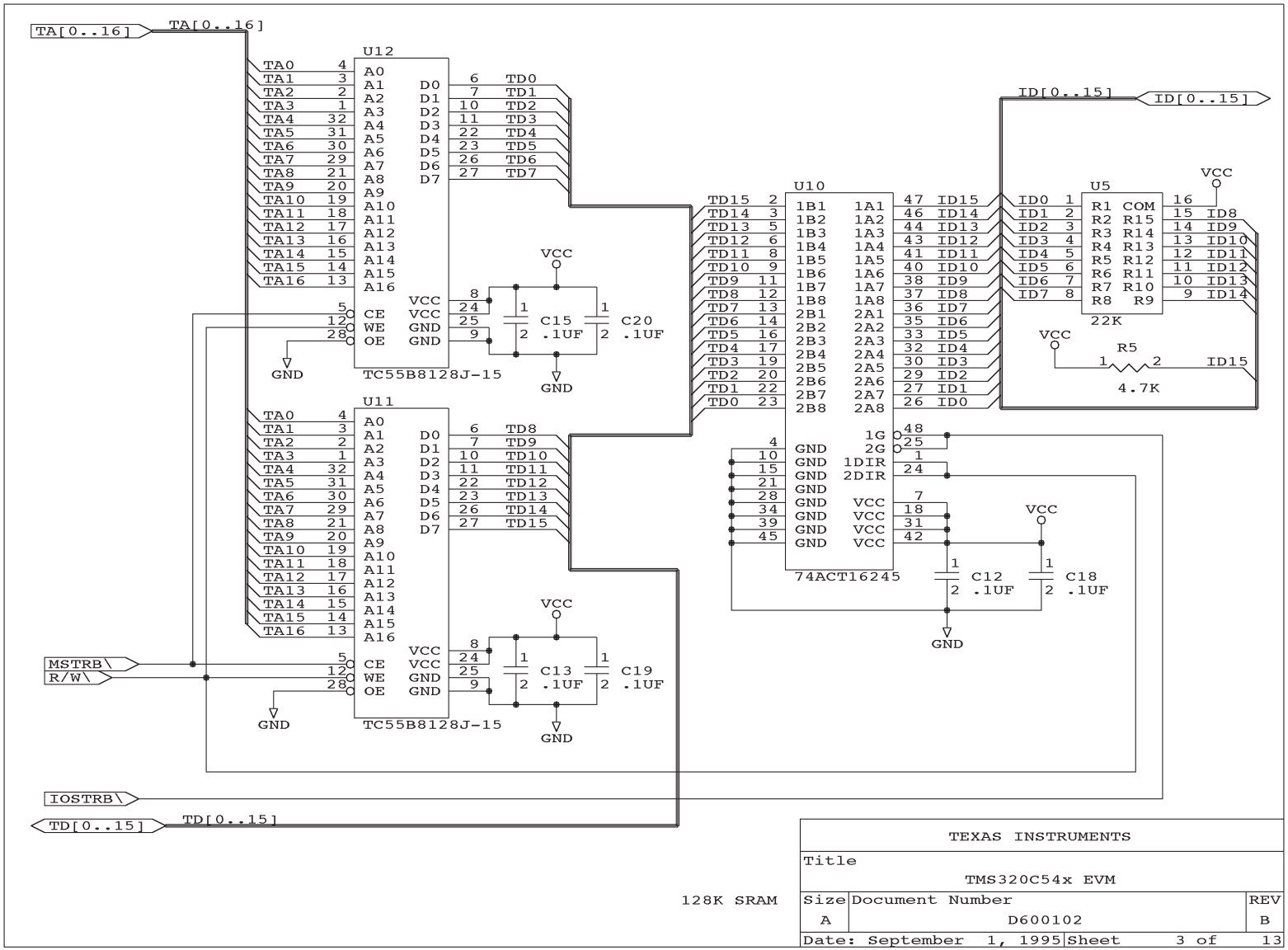
TEXAS INSTRUMENTS
SOFTWARE DEVELOPMENT SYSTEMS
SEMICONDUCTOR GROUP
HOUSTON, TEXAS

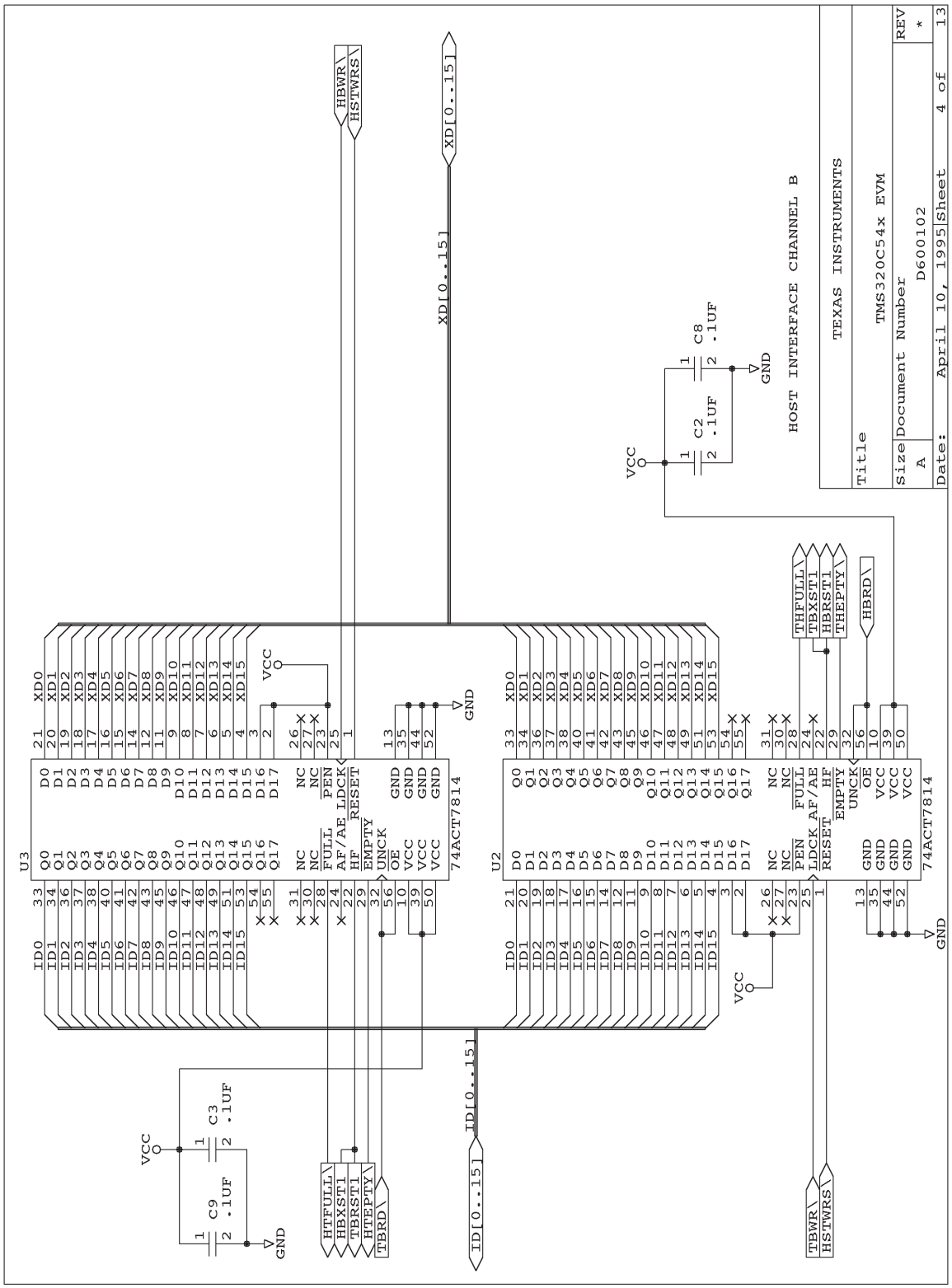
Title: TMS320C54x EVM

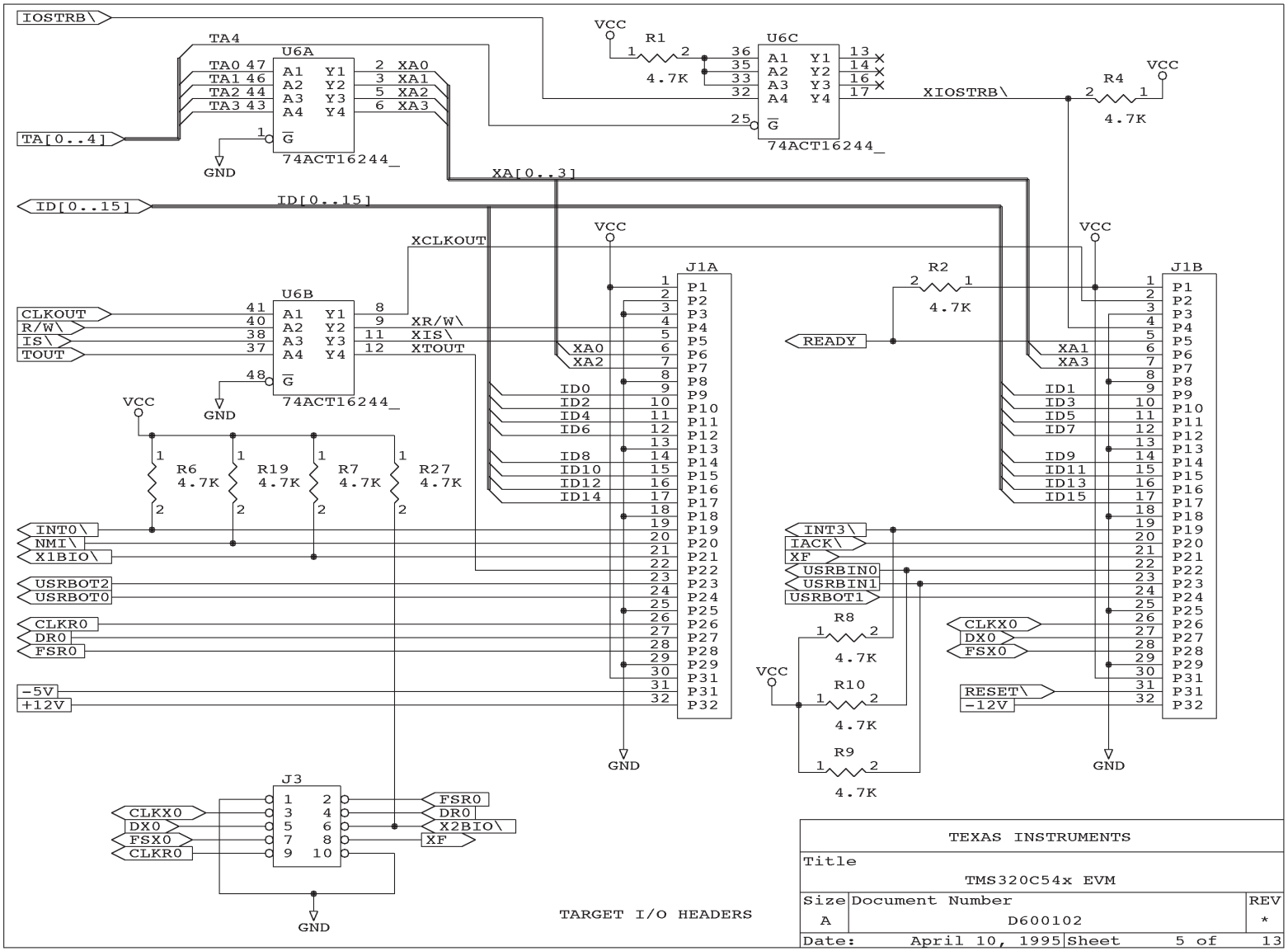
Size	Document Number	REV
A	D600102	B

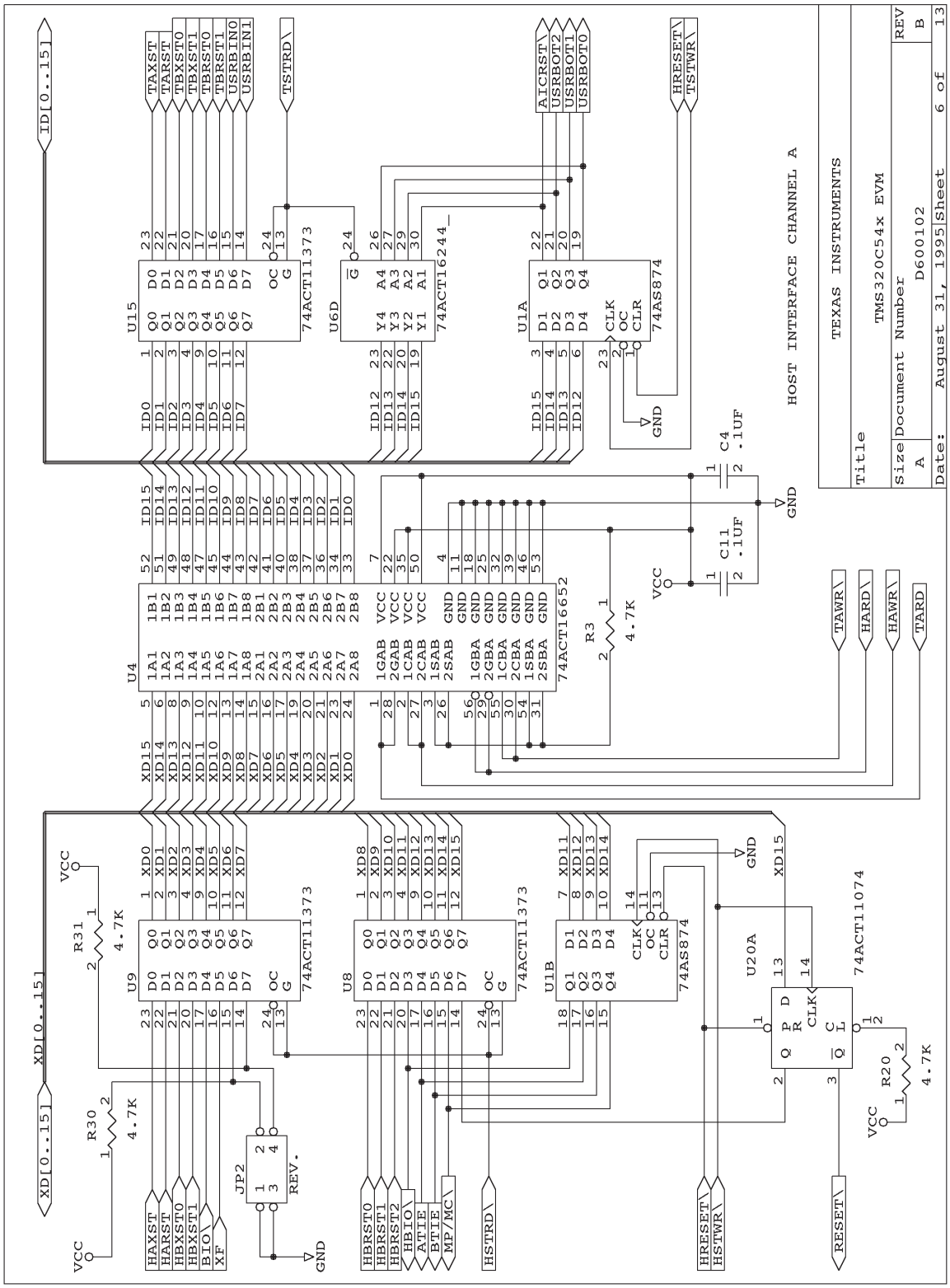
Date: September 1, 1995 | Sheet 1 of 13

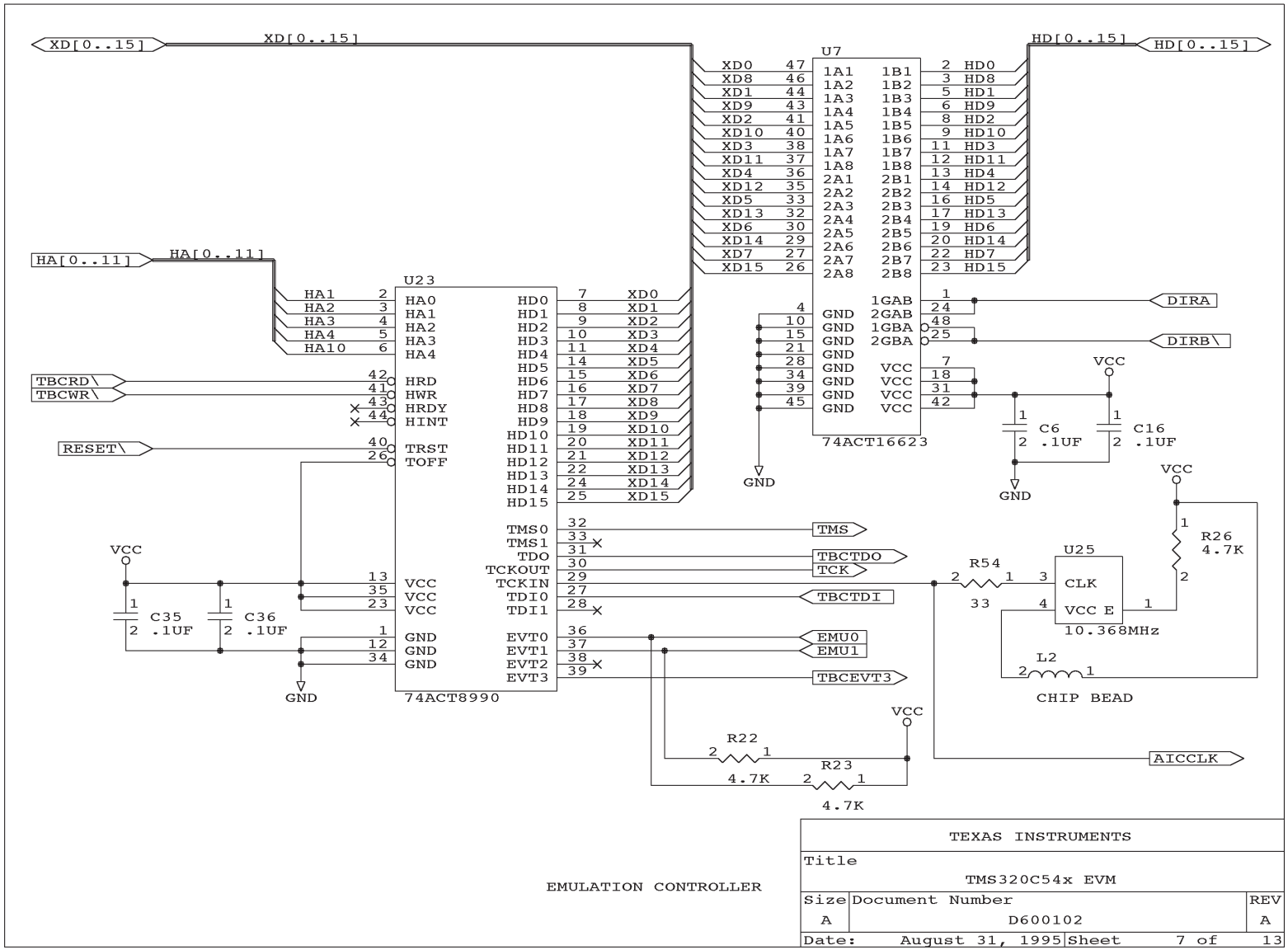

TD[0..15]
TA[0..16]
U16
TMS320C541PZ
U19
16L8-5
TSTWR\
TSTRD\
TBWRS\
TBWR\
TBRD\
TAWRS\
TAWR\
TARD
IS\
READY
R/W\
MSTRB\
IOSTRB\
EMU0
EMU1
TBCTDI
TBCTDO
TTRST\
TCK
TMS
VCC
R17
4.7K
NOTE: DO NOT INSTALL R12, R13, R15
TEXAS INSTRUMENTS
Title
TMS320C54x EVM
Size
A
Document Number
D600102
REV
A
Date: August 31, 1995
Sheet 2 of 13
CPU
IACK\
NMI\
INT0\
INT1\
INT2\
INT3\
RESET\
MP/MC\
BIO\
XF
TOUT
CLKR0
CLKX0
FSR0
FSX0
DR0
DX0
CLK1
CLKR1
CLKX1
FSR1
FSX1
DR1
DX1
CLKOUT
X1
X2/CLKIN
C31 .1UF
C28 .1UF
C52 .1UF
R52 33
R53 33
R18 4.7K
R14 4.7K
R15 4.7K
R16 4.7K
R13 4.7K
R11 4.7K
R12 4.7K
U21
CLK
E VCC
10MHz
L1
CHIP BEAD
GND

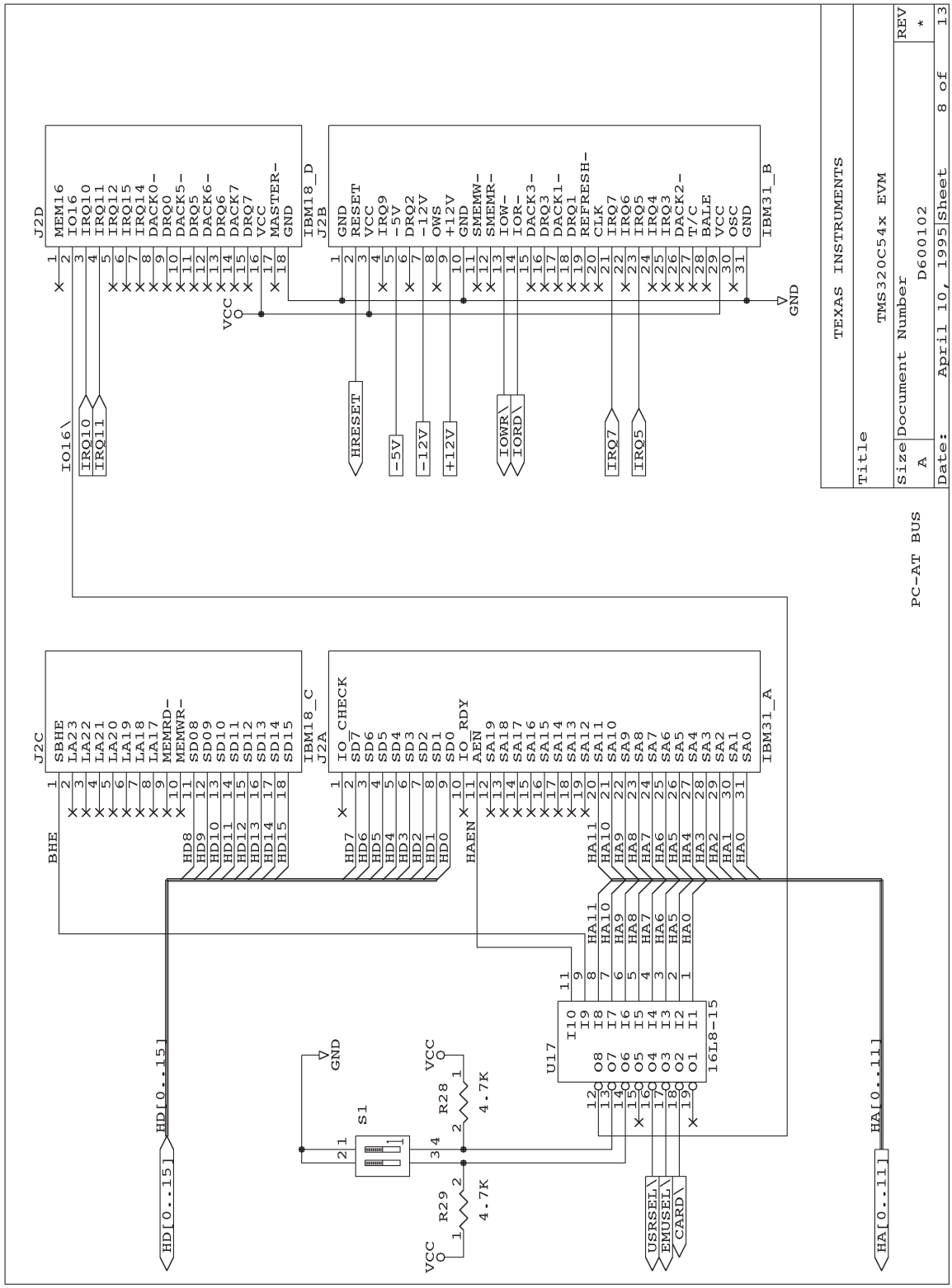

TA[0..16]
TA[0..16]
U12
TC55B8128J-15
U11
TC55B8128J-15
U10
74ACT16245
U5
22K
ID[0..15]
ID[0..15]
VCC
GND
C15 .1UF
C20 .1UF
C13 .1UF
C19 .1UF
C12 .1UF
C18 .1UF
R5
4.7K
MSTRB\
R/W\
IOSTRB\
TD[0..15]
TD[0..15]
128K SRAM
TEXAS INSTRUMENTS
Title
TMS320C54x EVM
Size A
Document Number D600102
REV B
Date: September 1, 1995
Sheet 3 of 13

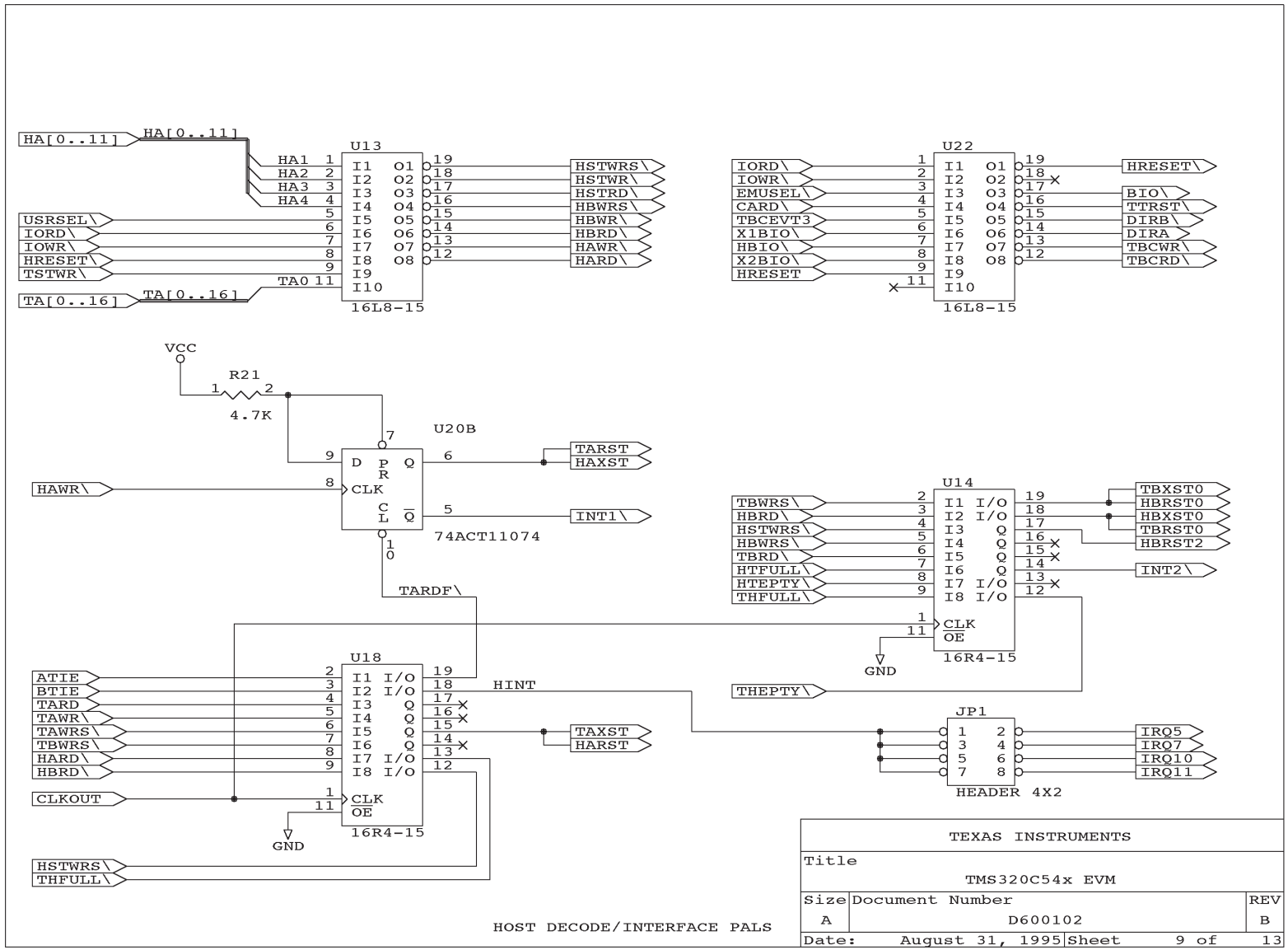

HOST INTERFACE CHANNEL B
TEXAS INSTRUMENTS
Title
TMS320C54x EVM
Size Document Number REV
A D600102 *
Date: April 10, 1995 Sheet 4 of 13
U3
74ACT7814
U2
74ACT7814
XD[0..15]
ID[0..15]
HBWR\
HSTWRS\
TBWR\
HSTWRS\
HTFULL\
HBXST1
TBRST1
HTEPTY\
TBRD\
THFULL\
TBXST1
HBRST1
THEPTY\
HBRD\
VCC
GND
C2 .1UF
C3 .1UF
C8 .1UF
C9 .1UF

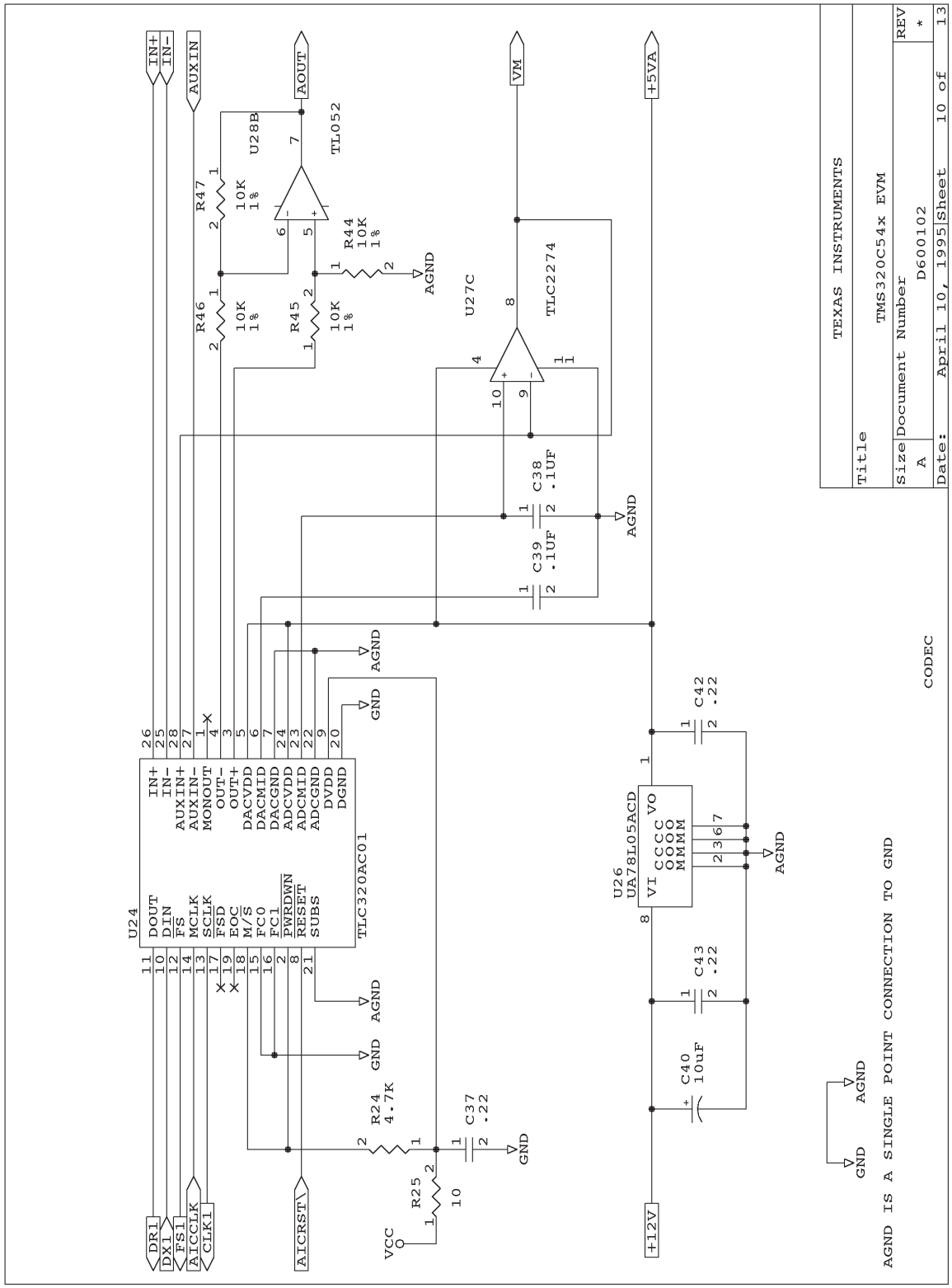

IOSTRB\
TA4
U6A
TA0 47
TA1 46
TA2 44
TA3 43
A1 Y1
A2 Y2
A3 Y3
A4 Y4
2 XA0
3 XA1
5 XA2
6 XA3
1 G
74ACT16244_
TA[0..4]
GND
VCC
R1
4.7K
U6C
36 A1 Y1 13
35 A2 Y2 14
33 A3 Y3 16
32 A4 Y4 17
25 G
74ACT16244_
XIOSTRB\
R4
4.7K
VCC
XA[0..3]
ID[0..15]
ID[0..15]
XCLKOUT
VCC
VCC
J1A
J1B
U6B
41 CLKOUT
40 R/W\
38 IS\
37 TOUT
A1 Y1
A2 Y2
A3 Y3
A4 Y4
8
9 XR/W\
11 XIS\
12 XTOUT
48 G
74ACT16244_
GND
R2
4.7K
READY
XA0
XA2
XA1
XA3
ID0
ID2
ID4
ID6
ID8
ID10
ID12
ID14
ID1
ID3
ID5
ID7
ID9
ID11
ID13
ID15
P1
P2
P3
P4
P5
P6
P7
P8
P9
P10
P11
P12
P13
P14
P15
P16
P17
P18
P19
P20
P21
P22
P23
P24
P25
P26
P27
P28
P29
P30
P31
P32
VCC
R6 4.7K
R19 4.7K
R7 4.7K
R27 4.7K
INT0\
NMI\
X1BIO\
USRBOT2
USRBOT0
CLKR0
DR0
FSR0
-5V
+12V
INT3\
IACK\
XF
USRBIN0
USRBIN1
USRBOT1
R8
4.7K
R10
4.7K
R9
4.7K
VCC
CLKX0
DX0
FSX0
RESET\
-12V
GND
GND
J3
1 2
3 4
5 6
7 8
9 10
CLKX0
DX0
FSX0
CLKR0
FSR0
DR0
X2BIO\
XF
GND
TARGET I/O HEADERS
TEXAS INSTRUMENTS
Title
TMS320C54x EVM
Size
A
Document Number
D600102
REV
*
Date: April 10, 1995
Sheet 5 of 13

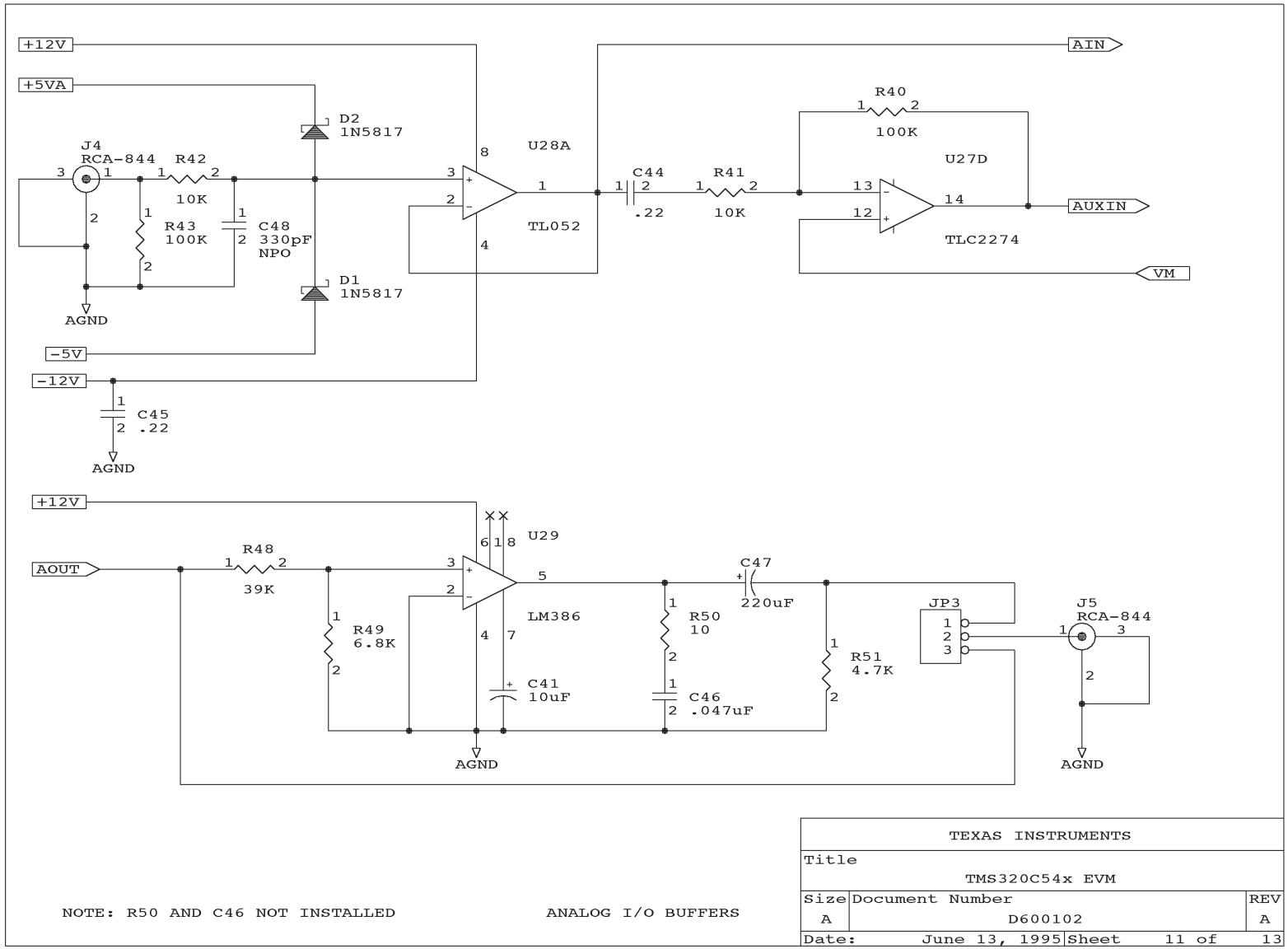

HOST INTERFACE CHANNEL A
TEXAS INSTRUMENTS
Title
TMS320C54x EVM
Size
A
Document Number
D600102
REV
B
Date: August 31, 1995
Sheet 6 of 13
ID[0..15]
XD[0..15]
U15
74ACT11373
U6D
74ACT16244
U1A
74AS874
U4
74ACT16652
U9
74ACT11373
U8
74ACT11373
U1B
74AS874
U20A
74ACT11074
TAXST
TARST
TBXST0
TBXST1
TBRST0
TBRST1
USRBIN0
USRBIN1
TSTRD\
AICRST\
USRBOT2
USRBOT1
USRBOT0
HRESET\
TSTWR\
TAWR\
HARD\
HAWR\
TARD
R3
4.7K
C4
.1UF
C11
.1UF
R31
4.7K
R30
4.7K
R20
4.7K
JP2
REV.
HAXST
HARST
HBXST0
HBXST1
BIO\
XF
HBRST0
HBRST1
HBRST2
HBIO\
ATIE
BTIE
MP/MC\
HSTRD\
HRESET\
HSTWR\
RESET\
VCC
GND


XD[0..15]
HD[0..15]
HA[0..11]
U7
74ACT16623
U23
74ACT8990
DIRA
DIRB\
TBCRD\
TBCWR\
RESET\
HRD
HWR
HRDY
HINT
TRST
TOFF
TMS
TBCTDO
TCK
TBCTDI
EMU0
EMU1
TBCEVT3
AICCLK
VCC
GND
C6 .1UF
C16 .1UF
C35 .1UF
C36 .1UF
R54
33
R26 4.7K
R22 4.7K
R23 4.7K
U25
CLK
VCC E
10.368MHz
L2
CHIP BEAD
EMULATION CONTROLLER
TEXAS INSTRUMENTS
Title
TMS320C54x EVM
Size A
Document Number D600102
REV A
Date: August 31, 1995
Sheet 7 of 13


TEXAS INSTRUMENTS
Title
TMS320C54x EVM
Size
A
Document Number
D600102
REV
*
Date: April 10, 1995
Sheet 8 of 13
PC-AT BUS
J2D
MEM16
IO16
IRQ10
IRQ11
IRQ12
IRQ15
IRQ14
DACK0-
DRQ0
DACK5-
DRQ5
DACK6-
DRQ6
DACK7
DRQ7
VCC
MASTER-
GND
IBM18_D
J2B
GND
RESET
VCC
IRQ9
-5V
DRQ2
-12V
OWS
+12V
GND
SMEMW-
SMEMR-
IOW-
IOR-
DACK3-
DRQ3
DACK1-
DRQ1
REFRESH-
CLK
IRQ7
IRQ6
IRQ5
IRQ4
IRQ3
DACK2-
T/C
BALE
VCC
OSC
GND
IBM31_B
J2C
SBHE
LA23
LA22
LA21
LA20
LA19
LA18
LA17
MEMRD-
MEMWR-
SD08
SD09
SD10
SD11
SD12
SD13
SD14
SD15
IBM18_C
J2A
IO_CHECK
SD7
SD6
SD5
SD4
SD3
SD2
SD1
SD0
IO_RDY
AEN
SA19
SA18
SA17
SA16
SA15
SA14
SA13
SA12
SA11
SA10
SA9
SA8
SA7
SA6
SA5
SA4
SA3
SA2
SA1
SA0
IBM31_A
IO16\
IRQ10
IRQ11
HRESET
-5V
-12V
+12V
IOWR\
IORD\
IRQ7
IRQ5
VCC
GND
HD[0..15]
HA[0..11]
BHE
HAEN
HD8
HD9
HD10
HD11
HD12
HD13
HD14
HD15
HD7
HD6
HD5
HD4
HD3
HD2
HD1
HD0
HA11
HA10
HA9
HA8
HA7
HA6
HA5
HA4
HA3
HA2
HA1
HA0
U17
16L8-15
I10
I9
I8
I7
I6
I5
I4
I3
I2
I1
O8
O7
O6
O5
O4
O3
O2
O1
S1
R28
4.7K
R29
4.7K
USRSEL\
EMUSEL\
CARD\


HA[0..11]
HA[0..11]
U13
HA1
HA2
HA3
HA4
USRSEL\
IORD\
IOWR\
HRESET\
TSTWR\
TA0
TA[0..16]
TA[0..16]
16L8-15
HSTWRS\
HSTWR\
HSTRD\
HBWRS\
HBWR\
HBRD\
HAWR\
HARD\
U22
IORD\
IOWR\
EMUSEL\
CARD\
TBCEVT3
X1BIO\
HBIO\
X2BIO\
HRESET
16L8-15
HRESET\
BIO\
TTRST\
DIRB\
DIRA\
TBCWR\
TBCRD\
VCC
R21
4.7K
U20B
HAWR\
TARST
HAXST
INT1\
74ACT11074
TARDF\
U14
TBWRS\
HBRD\
HSTWRS\
HBWRS\
TBRD\
HTFULL\
HTEPTY\
THFULL\
CLK
OE
16R4-15
GND
TBXST0
HBRST0
HBXST0
TBRST0
HBRST2
INT2\
THEPTY\
U18
ATIE
BTIE
TARD
TAWR\
TAWRS\
TBWRS\
HARD\
HBRD\
CLKOUT
HINT
TAXST
HARST
16R4-15
GND
HSTWRS\
THFULL\
JP1
HEADER 4X2
IRQ5
IRQ7
IRQ10
IRQ11
HOST DECODE/INTERFACE PALS
TEXAS INSTRUMENTS
Title
TMS320C54x EVM
Size
A
Document Number
D600102
REV
B
Date: August 31, 1995
Sheet 9 of 13

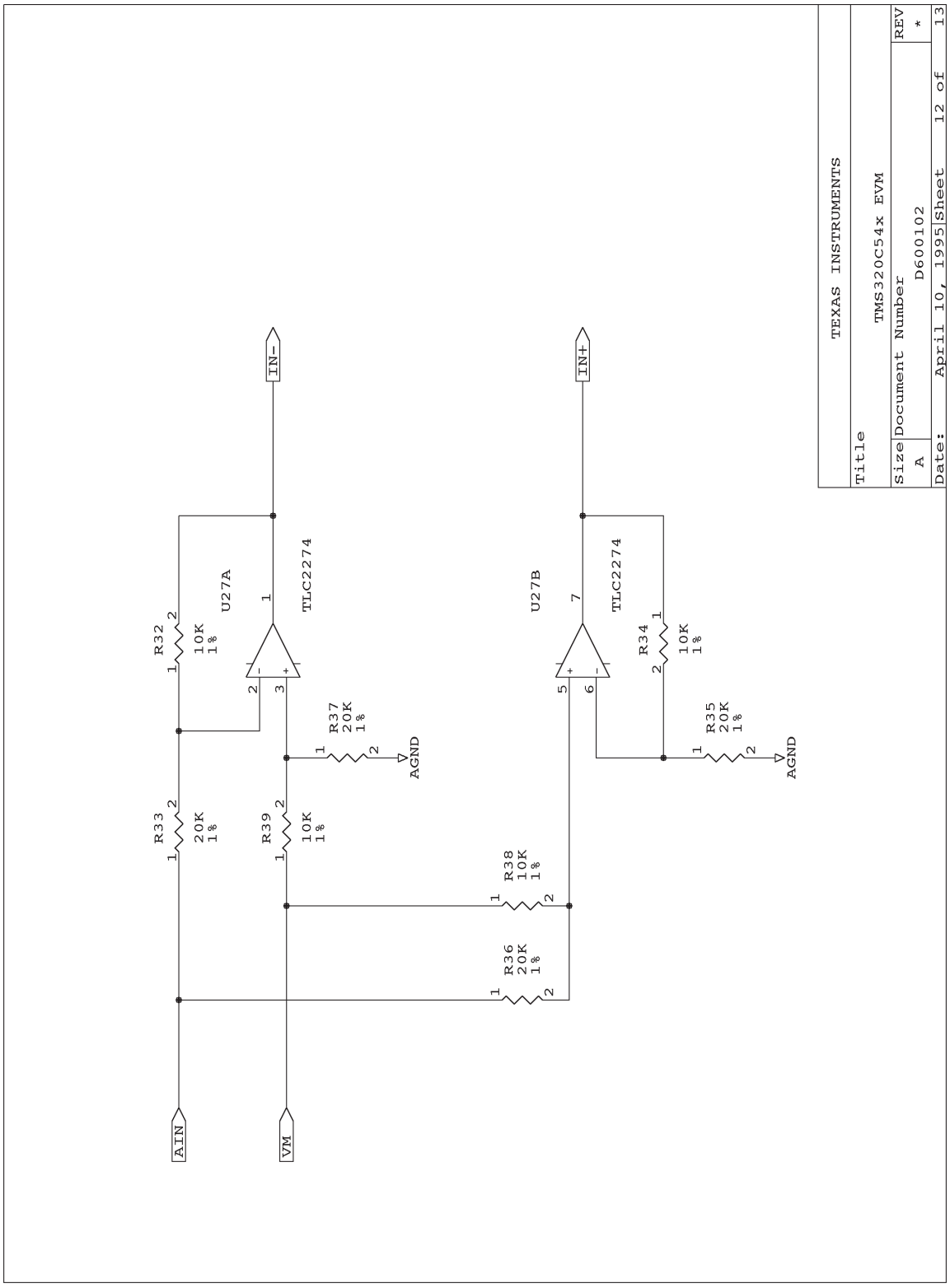

TEXAS INSTRUMENTS
Title
TMS320C54x EVM
Size Document Number REV
A D600102 *
Date: April 10, 1995 Sheet 10 of 13
CODEC
AGND IS A SINGLE POINT CONNECTION TO GND
U24
TLC320AC01
U26
UA78L05ACD
U27C
TLC2274
U28B
TL052
R24 4.7K
R25 10
R44 10K 1%
R45 10K 1%
R46 10K 1%
R47 10K 1%
C37 .22
C38 .1UF
C39 .1UF
C40 10uF
C42 .22
C43 .22
IN+
IN-
AUXIN
AOUT
VM
+5VA
+12V
VCC
DR1
DX1
FS1
AICCLK
CLK1
AICRST\
AGND
GND



NOTE: R50 AND C46 NOT INSTALLED

ANALOG I/O BUFFERS

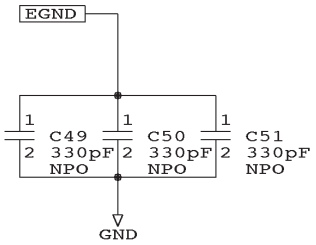
TEXAS INSTRUMENTS		
Title: TMS320C54x EVM		
Size: A	Document Number: D600102	REV: A
Date: June 13, 1995	Sheet 11 of 13	

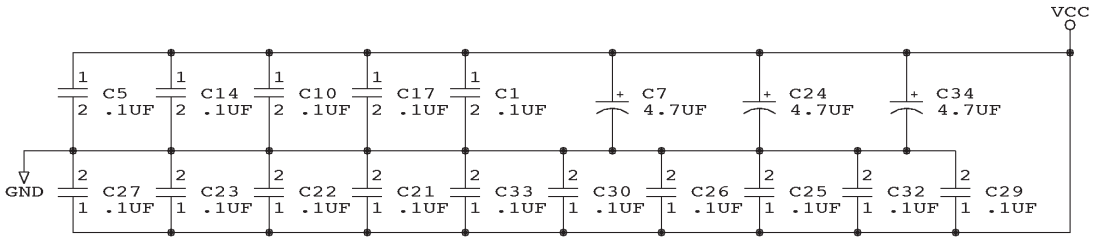

TEXAS INSTRUMENTS
Title
TMS320C54x EVM
Size
A
Document Number
D600102
REV
*
Date:
April 10, 1995
Sheet
12 of
13
IN-
IN+
AIN
VM
U27A
U27B
TLC2274
TLC2274
R32
10K
1%
R33
20K
1%
R34
10K
1%
R35
20K
1%
R36
20K
1%
R37
20K
1%
R38
10K
1%
R39
10K
1%
AGND
AGND

POWER CONNECTIONS:

IC	VCC	GND
TMS320C5xxPZ	8,11,36,39,49,64,76,87,90	1,9,10,26,35,40,50, 63,65,88,89
74ACT8990FN	13,23,35	1,12,34
TC55B8128J	8,24	9,25
74ACT7814DL	10,39,50	13,35,44,52
74ACT16245DL	7,18,31,42	4,10,15,21,28,34,39,45
74ACT16623DL	7,18,31,42	4,10,15,21,28,34,39,45
74ACT16244DL	7,18,31,42	4,10,15,21,28,34,39,45
74ACT16652DL	7,22,35,50	4,11,18,25,32,39,46,53
74ACT11373DW	18,19	5,6,7,8
74ACT11074D	11	4
74AS874DW	24	12
PAL16L8FN	20	10
PAL16R4FN	20	10
OSC 10MHz	4	2
OSC 10.368MHz	4	2
RES	16	

IC	+5VA	GND	AGND	+12	-12
320TLC320AC01FN	5,24	20	7,22		
TLC2274CD	4		11		
TL052CD				8	4
LM386M-1			4	6	
uA78L05ACD	1		2,3, 6,7	8	





CAPS AND POWER

TEXAS INSTRUMENTS		
Title: TMS320C54x EVM		
Size: A	Document Number: D600102	REV: A
Date: June 14, 1995	Sheet 13 of 13	

Index

J

K

M

O

P

S

T

W