

DOS Stamp™ User's Manual

by Ivan Baggett

Rev. 1.0 05/25/99

Rev. 1.1 06/30/99

Rev. 1.2 08/30/99

Rev. 1.3 09/17/99

Rev. 1.4 11/26/99

Rev. 1.5 02/16/01

Copyright 1999-2001 Bagotronix Inc.

All rights reserved

This page intentionally left blank

The material presented in this document is the intellectual property of Bagotronix Inc., except for material furnished by others where indicated. No part of this publication may be reproduced or distributed in any form or by any means, or stored in a data base or retrieval system, without the prior written permission of Bagotronix Inc.

DOS Stamp™ is a trademark of Bagotronix Inc.

DiskOnChip® 2000 is a registered trademark of M-Systems Inc.

Bagotronix Inc.
2900-1 Crescent Drive
Tallahassee, FL 32301
850-942-7905

This page intentionally left blank

Chapter 1: Introduction

This page intentionally left blank

Chapter 1: Introduction

What is the DOS Stamp™?

The DOS Stamp™ is a DOS-based miniaturized single board computer for embedded control applications. Measuring just 2.600 x 2.000 x 0.625 inches and consuming only about 1 W of power, the DOS Stamp™ incorporates I/O, storage, processing, communications, and a DOS operating system. Your application software may be programmed in familiar DOS-based software tools such as Borland C/C++, Microsoft C/Visual C++, compiled Quickbasic, PowerBasic, MASM, TASM, etc.

DOS Stamp™ features:

- AM188ES CPU at 40 MHz
- 512K SRAM
- 256K flash (organized as 128K BIOS and DOS, 128K flash disk)
- Optional DiskOnChip 2000 auxiliary flash disk up to 144 MB
- Real time clock with timed power-up
- Power monitor with power-on reset and watchdog timer
- Simple Bus Interface (SBI) expansion bus
- Power requirement: 5V only, 200 mA typical at 40 MHz
- 2 asynchronous serial communication ports (2 RS-232 or 1 RS-232 / 1 RS-485)
- 16 digital I/O lines, each programmable as input or output, with opto rack interface
- Optional 8-channel 12-bit ADC
- 1 uncommitted timer/counter
- 6-layer FR-4 PCB

Why BIOS and DOS?

The purpose of having BIOS and DOS in an embedded computer is to facilitate the development of application software. The task of software development is eased by the use of low cost, familiar software tools such as Borland C/C++, Microsoft C/Visual C++, QuickBasic, or PowerBasic. These compilers, and many others, produce an executable (EXE) output program. In order to run an EXE, a file system and some type of file storage medium

Chapter 1: Introduction

are required. The file system is provided by DOS-ROM and the storage medium is provided by BIOS flash disk support.

Why does an EXE require BIOS and DOS?

An EXE is a relocatable program. The final output program from the compiler is not fixed to run at a particular address in memory. When the program is loaded, it is the job of DOS to choose where in memory to load the program. Then DOS must “fix up” the code as it is loaded to run at the address DOS has chosen. Once the program is loaded, it may make use of hardware, such as input from a keyboard, output to a screen, reading and writing to disk drives, etc. To do this, the program will interact with DOS and/or BIOS. The BIOS provides the program with a software interface to the low-level hardware, isolating your program from variations in hardware design. For example, to set the time of the real-time clock, you would call BIOS interrupt 1Ah, function 03h. Using this BIOS function, your program would function identically on a DOS Stamp™ as on a PC. Another reason BIOS and DOS are needed is that your compiler’s library has many functions which make calls to BIOS and DOS, such as console I/O, file operations, date/time functions, etc.

What languages can I use with the DOS Stamp™?

You can use any programming language which meets all the following criteria:

- Is a compiled language that produces a stand-alone EXE that runs under DOS
- Produces 16-bit code for 8086 or 80186
- Has I/O read, I/O write, and call interrupt functions
- Is able to link in external object modules written in other languages

All DOS-based C/C++ compilers, Turbo Pascal, and compiled QuickBasic qualify. The low-level I/O driver source for the DOS Stamp™ is in C/ASM, therefore native access to the hardware is feasible only in C or ASM. However, by linking of object files, other languages may call these drivers for an efficient software interface to the hardware.

Chapter 1: Introduction

How do I get Technical Support?

To help you get the most out of the DOS Stamp™, Bagotronix offers application assistance at no charge. Contact information:

voice: (850)-942-7905

fax: (850)-942-7905

e-mail: techsupport@bagotronix.com

Internet: www.bagotronix.com

If you require design assistance for other aspects of your embedded application, Bagotronix consulting design services are available for a nominal fee.

Organization

The organization of the DOS Stamp™ User's Manual is logically divided into several chapters. Procedures for writing, compiling, and debugging embedded application software are covered in Chapter 4: Software Development. Interfacing to the SBI bus, ADC, serial, and I/O ports is discussed in Chapter 3: Hardware Reference. Technical information about the DOS Stamp™ is in the Appendix section.

Throughout this manual, it is assumed that you have some experience in C or Basic programming, 8086 assembly language, and PC usage. The development process involves these steps:

- 1) Design or acquire your application's I/O hardware (sensors, switches, relays, solenoids, etc.)
- 2) Connect the DOS Stamp™ to your application's I/O hardware and the PC serial port
Write, compile, download, and debug your application.

This page intentionally left blank

Chapter 2: Setup

This page intentionally left blank

Chapter 2: Setup

Overview

This chapter tells you how to set up the DOS Stamp™ for development activities. You are probably eager to get started on your project, but please take time to read this chapter first.

What you Need

To develop your embedded application software you need at least the following items:

- 1) DOS Stamp™
- 2) DOS Stamp™ software library on floppy disk
- 3) DB9F to DB9F null-modem serial adapter (crossover cable)
- 4) JP1 cable for DOS Stamp™
- 5) 5V DC power supply
- 6) Your DOS-based compiler
- 7) PC with an available serial port

Items 1 - 5 are included with the DOS Stamp™ Starter Kit.

How to Connect It All

- 1) Place the DOS Stamp™ on the table. Verify that standoffs are installed on the bottom side of the DOS Stamp™ to prevent direct contact of the circuit board with the table surface.
- 2) Connect the JP2 cable to the JP2 header as shown in Figure 2-1. The polarizing key should point toward the middle of the PCB.
- 3) Connect the JP2 cable DB9M connector to one end of the serial crossover cable as shown in Figure 2-1.
- 4) Plug the 5 V power supply into an AC outlet.
- 5) Connect the other end of the serial crossover cable to the DB9M serial port connector on the back of your PC.
- 6) Connect the power supply plug to the JP2 cable power jack. The DOS Stamp™ is now powered up. Go to the section “Checking it Out”.

Chapter 2: Setup

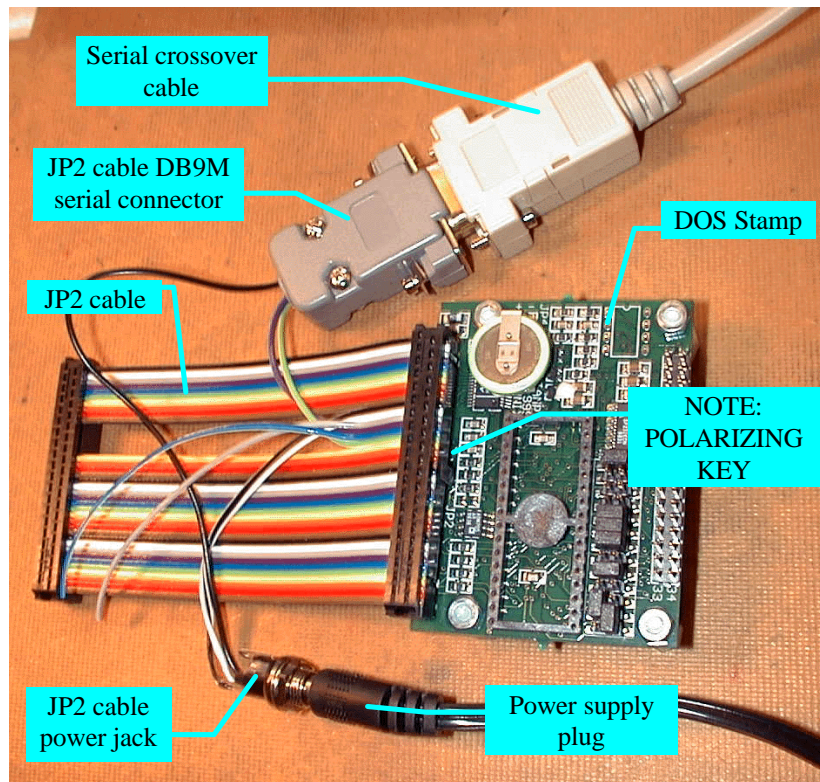


Fig. 2-1: DOS Stamp™ Development Setup

Checking it Out

It is assumed that the compiler you will use for software development is already installed on your PC. If this is not so, install it now.

1) To check the serial link between your PC and the DOS Stamp™, start a terminal emulator program. XLTERM is a terminal emulator program which is designed for easy use with the DOS Stamp™. If you would rather use a different terminal emulator (such as Procomm, HyperTerminal, etc.) on your PC, set it to 9600 baud, 8 data bits, no parity, 1 stop bit. When the DOS Stamp™ powers up, the sign-on message should appear on the terminal window as follows:

```
General Software 80186 Embedded BIOS (tm) Version 4.0
Copyright (C) 1992-1997 General Software, Inc.
DOS Stamp
...
00512 KB OK
Wait.....
FLASH DISK: 000DE000
...
(C) 1997 General Software, Inc.
80186-4.0-4040-0000
```

Chapter 2: Setup

```
...  
B>
```

If this does not work, recheck your power and serial cable connections and try again. If this still does not work, call Bagotronix technical support.

2) Insert the example programs floppy disk in your PC, create a directory \DOSSTAMP\SRC on your PC's hard disk and copy the files into them.

3) Try interacting with the DOS by typing some commands, such as DIR, TIME, DATE, etc.:

```
B>DIR  
RAMDISK  SYS           6224  4-May-99  3:01p  
TRANSFER EXE        15948  4-May-99  3:01p  
CONFIG   SYS           35   4-May-99  3:02p  
AUTOEXEC BAT         11 26-May-99 12:57p  
4 File(s)      87552 bytes free
```

```
B>
```

If your attempt to get started was unsuccessful, go back and examine each step to see if it was done correctly. It is important to get this test working, since you will not be able to develop your application software unless the DOS Stamp™ and your PC cooperate. Serial ports are frequently troublesome. If your PC has a non-standard serial port, you are in for some detective work.

This page intentionally left blank

Chapter 3: Hardware Reference

This page intentionally left blank

Chapter 3: Hardware Reference

Overview

This chapter contains information on how to use the DOS Stamp™ features. It is assumed that you are familiar with 8086 architecture and possess a working knowledge of electronics.

CPU

The DOS Stamp™ uses the AM188ES CPU, which is instruction set compatible with the 80186, which is a superset of the 8086. The AM188ES has a memory address range of 1 MB. The I/O address range is 64 KB. The on-chip peripherals include two asynchronous serial channels, programmable I/O, power-save unit, interrupt controller, timer/counter, chip select unit, and DMA controller.

Memory Map

DOS Stamp™ memory map:

00000h - 7FFFFh	SRAM
80000h - BFFFFh	Auxiliary flash disk socket
C0000h - FFFFFh	Boot flash (BIOS, DOS, and flash disk)

A DiskOnChip 2000 flash disk may be inserted into the auxiliary flash disk socket for additional storage needs.

The flash disk, BIOS, and DOS are all contained in a single flash chip. This saves space and cost, but has the limitation that reads from the flash are not permitted while a write to the flash is in progress. For this reason, interrupts are disabled during writes. A flash write takes about 5 ms to complete. Reading from files does not disable interrupts. If your application requires interrupts to be serviced during file writes, you must write to either a RAM disk or to another flash disk installed in the auxiliary flash disk socket.

The flash disk occupies 128K of the flash chip. The BIOS has an integral flash driver which handles flash erasure and wear-leveling transparently. Some of the flash disk is consumed by the File Allocation Table (FAT) and a spare block area. This results in a usable area of 105,984 bytes. Your program may use some or all of this space.

Chapter 3: Hardware Reference

The BIOS and DOS-ROM occupy the top 128K of flash space. Since they are in flash memory, the BIOS and DOS-ROM can be updated to customize, add new features, fix bugs, etc. This requires a special utility program.

I/O Map

DOS Stamp™ I/O map:

0800h - 08FFh	-PCS0 (SBI bus)
0900h - 09FFh	-PCS1 (SBI bus)
0B00h - 0BFFh	-PCS3
0D00h - 0DFFh	Real Time Clock
0E00h - 0EFFh	ADC
FF00h - FFFFh	AM188ES Peripheral Control Block

See the [AM188ES User's Manual](#) for more details on the AM188ES Peripheral Control Block.

Simple Bus Interface (SBI):

If your application needs more than the built-in I/O on the DOS Stamp™, you need to use the SBI bus. The SBI bus signals are listed in Table 3-1. Active low signals are prefixed by (-). When -PCS0 and -PCS1 are used as chip selects, they can not be used as general purpose I/O. The -PCS0 and -PCS1 signals must be enabled for the chip select function **before** any circuitry connected to them is accessed. To do this for -PCS0, clear bit 0 of register PIO1_MODE, then clear bit 0 of register PIO1_DIR. To do this for -PCS1, clear bit 1 of register PIO1_MODE, then clear bit 1 of register PIO1_DIR.

Chapter 3: Hardware Reference

Pin Number	JP2	Description
1	AD0	Address/Data bit 0 (3-state)
2	ALE	Address Latch Enable (output)
3	AD1	Address/Data bit 1 (3-state)
4	-RD	Read (active low)
5	AD2	Address/Data bit 2 (3-state)
6	-WR	Write (active low)
7	AD3	Address/Data bit 3 (3-state)
8	A0	Address bit 0 (output)
9	AD4	Address/Data bit 4 (3-state)
10	A1	Address bit 1 (output)
11	AD5	Address/Data bit 5 (3-state)
12	A2	Address bit 2 (output)
13	AD6	Address/Data bit 6 (3-state)
14	INT0	Interrupt 0 (input, rising edge or high level trigger)
15	AD7	Address/Data bit 7 (3-state)
16	INT1	Interrupt 1 (input, rising edge or high level trigger)
17	-RESET	Reset (output, active low)
20	-PCS0/GPIO9	Peripheral Chip Select 0 (output, active low, multiplexed with General Purpose I/O 9)
21	-PCS1/GPIO10	Peripheral Chip Select 1 (output, active low, multiplexed with General Purpose I/O 10)
29	GND	Ground (0 V)
30	VCC	System power (+5 V)

Table 3-1: SBI Bus Signals

As a minimum, interfacing to the SBI bus requires transfer of data. To transfer data, data lines (AD7..AD0), a chip select line (-PCS0 or -PCS1), and -RD (for input) or -WR (for output) signals must be used. Figure 3-1 shows how to connect more output lines to the SBI bus using a TPIC6273 (8-bit register with active low N-MOSFET outputs). This circuit is good for driving relays, solenoids, incandescent bulbs, small stepper motors, etc. If you want regular logic level outputs, substitute a 74HC273 instead. Figure 3-1 also shows how to connect more input lines to the SBI bus using a 74HC373 (8-bit transparent latch with output enable). If more than two I/O devices are desired, it is necessary to generate more chip selects. This can be done by attaching a 74HC138 decoder to the address lines A2..A0 as shown in Figure 3-2.

Circuitry connected to the SBI bus may also generate interrupts on INT0 and INT1. Interrupts can operate in two different modes: edge-sensitive and level-sensitive. In edge-sensitive mode, interrupt requests are made on the rising edge (low-to-high transition) of the request pulse. In

the level-sensitive mode, interrupt requests may occur as long as the interrupt request line remains high. Two or more devices may share the same interrupt in level-sensitive mode. The AM188ES interrupt controller must be programmed for the desired mode and the interrupt input unmasked before interrupts can actually occur on that input. For information on programming the AM188ES interrupt controller, refer to the [AM188ES User's Manual](#).

The SBI bus does not support DMA requests from hardware, since there are no DMA request

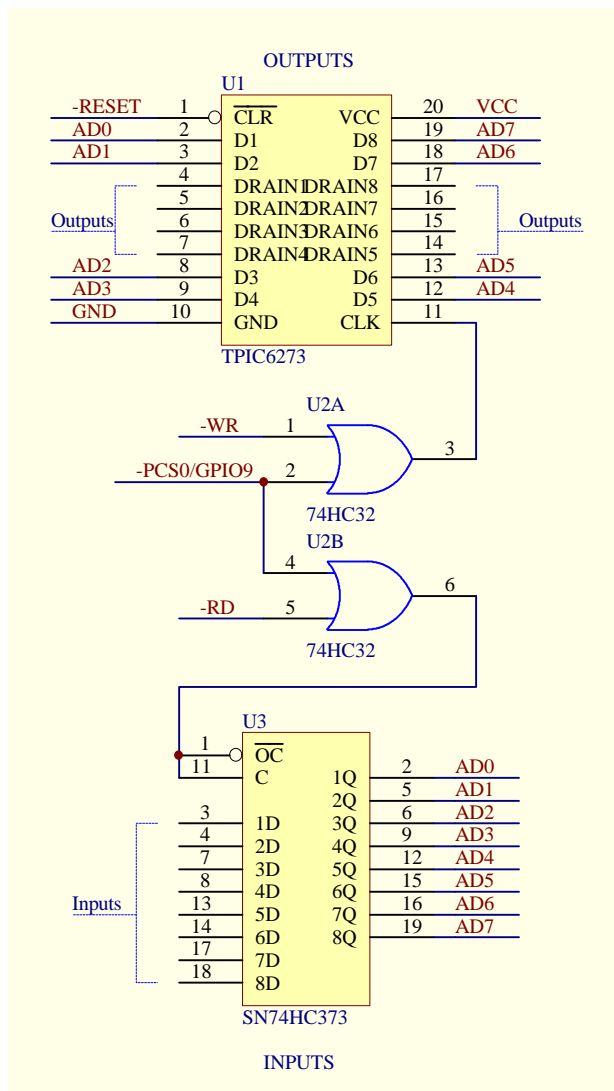


Figure 3-1

lines on the SBI bus. However, the GPIO1 and GPIO2 lines on the JP1 connector may be used as hardware DMA request signals when programmed for their alternate function. DMA transfers are also possible when requested through software. See the [AM188ES User's Manual](#) for further information on how to use DMA.

On-Board Peripherals

A summary of the DOS Stamp™ on-board peripherals follows. For connector pinouts see Appendix A.

RS-232 Port: the primary asynchronous serial port (COM1) has RS-232 transceivers (MAX232) for transmit data (TX), receive data (RX), request to send (RTS), and clear to send (CTS). To use RTS and CTS functions, JP3 jumpers

must be configured properly. Baud rates range from 50 to 1250K, with full or half duplex

operation. RS-232 voltages are generated on-board. COM1 is at I/O address FF80h and uses interrupt vector 14h. Due to the non-standard I/O address, interrupt vector and register set, languages with built-in serial port support (i.e. QuickBasic) will not work in the standard manner. Such programs should call assembly language functions for serial port usage.

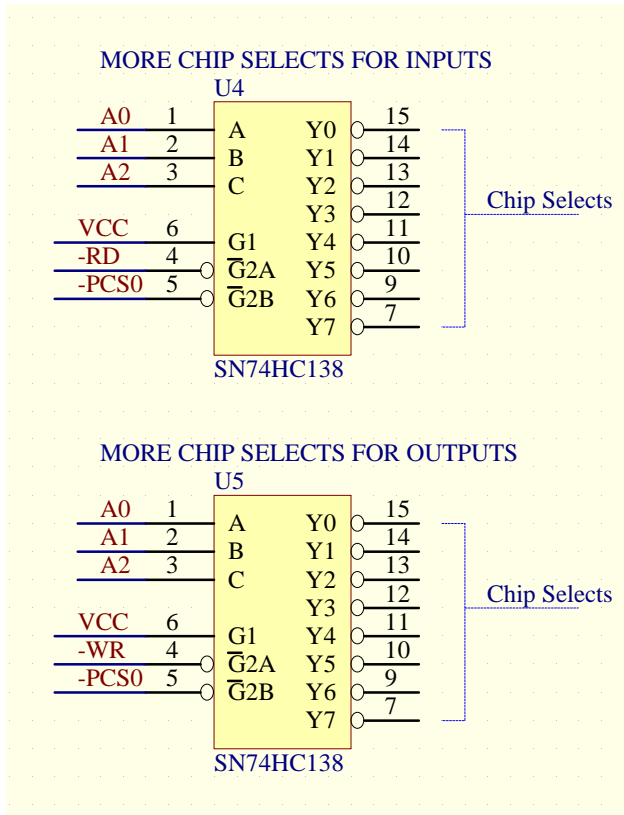


Figure 3-2

RS-232/485 Port: the secondary asynchronous serial port (COM2) may be configured for either RS-232 or RS-485. RS-232 transceivers support TX and RX only. Baud rates range from 50 to 1250K, with full or half duplex operation. RS-232 voltages are generated on-board. Alternatively, an RS-485 transceiver (75176 type) may be used, with half duplex operation. Jumper (JP3) selectable 100 ohm AC termination is provided for RS-485 network connection. COM2 is at I/O address FF10h and uses interrupt vector 11h. Due to the non-standard I/O address, interrupt vector and register

set, languages with built-in serial port support (i.e. QuickBasic) will not work in the standard manner. Such programs should call assembly language functions for serial port usage.

General Purpose I/O: there are 16 general purpose I/O (GPIO) pins on the JP1 connector. Each GPIO can be programmed as an input with pull-up/down, an input without pull-up, an output, an open-drain output, or a peripheral function. The value of the pull-up/down resistor is about 10K. The default configuration of a GPIO is an input with pull-up, except for GPIO5 and GPIO7, which are inputs with pull-down. Refer to [the AM188ES User's Manual](#) for

Chapter 3: Hardware Reference

details on how to use the Programmable I/O. The GPIOs are mapped to the AM188ES PIO pins and the JP1 connector as shown in Table 3-2.

GPIO	PIO	JP1 Pin	Alternate Function
0	30	31	INT4
1	12	29	DRQ0/INT5
2	13	27	DRQ1/INT6
3	20	9	-RTS0
4	11	25	N/A
5	10	23	N/A
6	0	21	TMRIN1
7	1	19	TMROUT1
8	31	17	INT2
9	16	15	-PCS0
10	17	13	-PCS1
11	18	5	-EOC (when ADC option is installed)
12	19	7	-PCS3
13	27	3	TXD1 (when configured by JP3 jumpers)
14	28	1	RXD1 (when configured by JP3 jumpers)
15	21	11	-CTS0 (when configured by JP3 jumpers)

Table 3-2: GPIO Mapping

The GPIOs can source or sink small currents (i.e. 10 mA), such as for driving LEDs directly. Be sure to use a series resistor to limit the LED current. For larger output currents, use a solid state relay or Darlington transistor with a resistor to limit the input current or base current, respectively.

ADC: if installed, a MAX197 ADC allows interfacing of up to eight channels of analog circuitry, sensors, measuring devices, etc. with 12-bit resolution. The sample-hold is built into the ADC. The reference voltage (VREF) is built into the ADC and is available to your interface circuitry on JP2 pin 32. Do not load VREF by more than 0.5 mA, or ADC accuracy may suffer. Also, do not change VREF loading during a conversion. It is best to buffer VREF with an op-amp if you need the reference voltage for your application circuitry. Unused ADC channels should be connected to ground. The range of each ADC channel may be

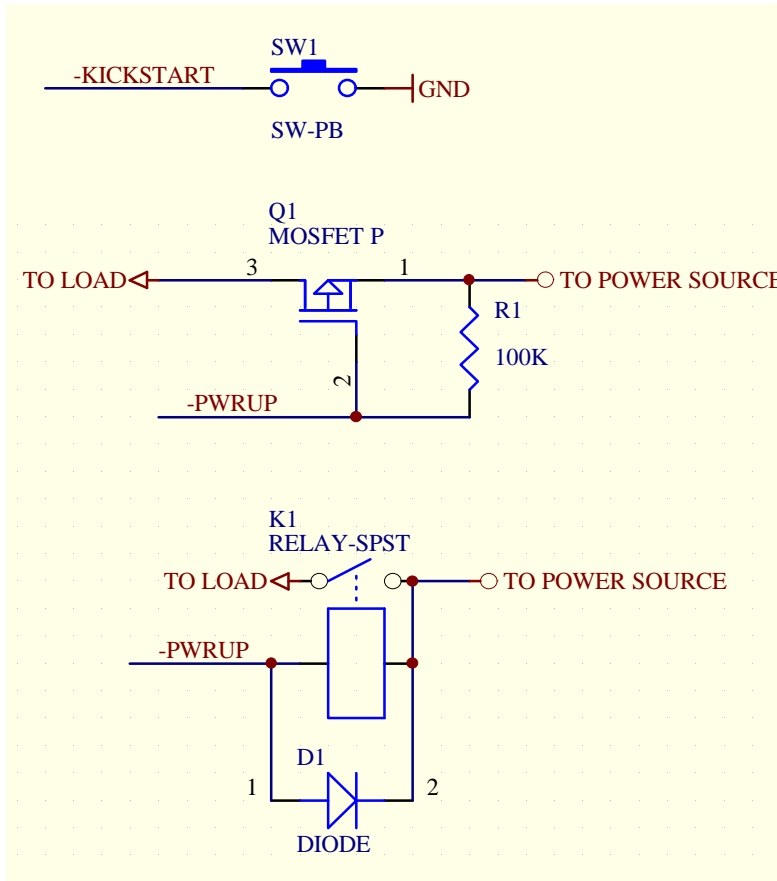


Figure 3-3

effective way to eliminate 60 Hz power line noise from ADC conversions is to sample many times over a 60 Hz period (16.67 ms) and average all the samples together. This has the effect of integrating the AC noise over the entire cycle to zero.

Real Time Clock (RTC): a Dallas 1689 keeps date and time in the absence of VCC power. The RTC may be accessed through the BIOS or directly as I/O reads and writes. The RTC is capable of powering up the DOS Stamp™ and other equipment connected to the -PWRUP pin (JP2 pin 22). This feature can be used to let the system turn itself on at a previously programmed time and date, do some processing, set the time and date for the next power up, and then turn itself off. No standby power is required. To use this feature, a manual pushbutton and a relay or transistor is required. Connect these items as shown in Fig. 3-3. The

programmed individually for 0 to +5, 0 to +10, -5 to 5, and -10 to +10 V. See the MAX197 data sheet for more details.

There are many hardware and software based techniques to eliminate noise in ADC conversions. Hardware techniques include the use of shielded wire to carry signals, RC filtering, active filtering, etc. Software based techniques include oversampling, averaging, digital filtering, etc. One

Chapter 3: Hardware Reference

manual pushbutton forces the RTC to power up when the pushbutton is pressed. The -PWRUP output of the DOS Stamp™ goes low in response to the RTC alarm time match or by pressing the -KICKSTART pushbutton. When this happens, the P-MOSFET transistor is turned on, allowing power to flow from the source to the load. In the case of a battery operated embedded application, the power source is a battery and the load is a voltage regulator which supplies regulated voltage to the DOS Stamp™ and associated circuitry. The relay circuit works the same way as the P-MOSFET circuit, except a relay is used instead of a MOSFET. DO NOT OMIT THE DIODE! It is needed to protect the -PWRUP output from the back-EMF produced by the relay coil when it is de-energized.

Watchdog Timer Reset: the AM188ES has a built-in watchdog timer. The BIOS disables the watchdog timer during boot. This is necessary because your application may take more time to load and start running than the timeout period of the watchdog timer. Unfortunately, the watchdog timer cannot be re-enabled after being disabled. Refer to the AM188ES User's Manual for more information. If your application must have a watchdog timer, you may use an off-board watchdog timer chip connected to the -EXTRESET input and toggled by a PIO pin configured as output.

Power Monitor: the +5 V supply is monitored to make sure it is at least 4.75 V. If not, the CPU will be held in reset. This prevents improper program execution from occurring due to marginal supply voltage.

Auxiliary Flash Disk Socket

The auxiliary flash disk socket can take either a flash chip or a DiskOnChip, depending on the jumper configuration of JP3 pins 1, 2, 3, and 4 (see Appendix C). The pinout of the auxiliary flash disk socket is shown in Table 3-3. The socket is mapped into memory address space from 80000h - BFFFFh. To access a 512K byte flash chip, a bank select signal is present on A18 (pin 1). This signal is controlled by PIO25 of the AM188ES CPU. After reset, PIO25 defaults to an input with pull-up, therefore the upper 256K of the flash chip will be selected.

Chapter 3: Hardware Reference

To access the lower 256K of the flash chip, PIO25 must be configured as an output by software, and then driven low.

Pin	Function
1	A18 (bank select to PIO25)
2	A16
3	A15
4	A12
5	A7
6	A6
7	A5
8	A4
9	A3
10	A2
11	A1
12	A0
13	D0
14	D1
15	D2
16	GND
17	D3
18	D4
19	D5
20	D6
21	D7
22	CE#
23	A10
24	OE#
25	A11
26	A9
27	A8
28	A13
29	A14
30	A17 (if JP3 3-4) or VCC (if JP3 1-2)
31	WE#
32	VCC

Table 3-3: Auxiliary Flash Disk Socket Pinout

This page intentionally left blank

Chapter 4: Software Development

This page intentionally left blank

Chapter 4: Software Development

Overview

This chapter contains information on how to develop application software for the DOS Stamp™. It is assumed that you have already connected your PC to the DOS Stamp™ and that the serial link is working properly.

Application programs are written and compiled on the PC. The resulting EXE is transferred to the DOS Stamp™ via serial port download. Either the TRANSFER.EXE program or the Manufacturing Link can be used for serial transfers. Debugging can be accomplished by placing “print” statements throughout the program to display the state of variables.

The software you need:

- Your compiler (C, Basic, etc.)
- A terminal emulator program (XLTERM or other)
- DOS Stamp™ distribution disk: Embedded DOS-ROM and Utilities (included)
- DOS Stamp™ distribution disk: software libraries, examples, and source (included)

DOS Mini-command Interpreter

The resident part of DOS which handles commands such as “dir”, “copy”, and others, is typically contained in the file COMMAND.COM in a disk-based PC. On the DOS Stamp, COMMAND.COM is built into the flash BIOS and does not appear in a directory listing of files. This is called the “mini-command interpreter”, and it provides a subset of the features of a typical COMMAND.COM. The limitations of the mini-command interpreter are as follows:

- 1) Does not handle wildcards. You cannot use the command “copy *.* b:” because of the wildcard “*.*”. You must type each filename to be copied individually.
- 2) Does not infer destination filenames. You cannot use the command “copy somefile.txt b:” because the destination filename is inferred. You must type “copy somefile.txt b:somefile.txt”.

Chapter 4: Software Development

If you cannot live with the limitations of the mini-command interpreter, you can use the TRANSFER program to transfer the full version of the command interpreter, COMMAND.COM, over to the DOS Stamp. Once the transfer is complete, type “COMMAND” at the DOS prompt. The full COMMAND.COM will then load into RAM and allow you full usage of wildcards and inferred filenames.

Transferring Files to/from a PC

The TRANSFER program is used to transfer files to and from a PC using the XMODEM file transfer protocol. Be advised that XMODEM rounds file sizes up to the nearest 128 bytes, so when you view a directory listing of transferred files, the file sizes may be slightly larger than the original files. XMODEM also pads the end of the file with extra non-printing characters to round out the file size. These padding characters can cause problems with some software. For example, batch files (AUTOEXEC.BAT, etc.), edited on a PC and transferred to the DOS Stamp will have padding characters after the last line. These characters can cause problems with the DOS mini-command interpreter. It is best to create batch files on the DOS Stamp directly by typing “copy con batchfil.bat” at the DOS prompt. Then type in the lines of text you need. When finished, hit <CTRL>-Z to close the file.

Floating Point Math

If you intend to use floating point math in your embedded application software, you may need to take special steps, depending on your compiler. The DOS Stamp does not have an 8087 math coprocessor, so instructions that are specific to the 8087 must be emulated. Most compilers have built-in support for 8087 emulation.

Some compilers have automatic detection of the 8087 during program startup. In the case of Borland C compilers and the PowerBasic compiler, the 8087 detection tests do not work on the DOS Stamp and will cause it to hang. To overcome this problem, place the command “SET 87=NO” in the AUTOEXEC.BAT file before your application is started. This will tell Borland C and the PowerBasic

Chapter 4: Software Development

startup code in explicit terms that there is no 8087 present. In the case of Microsoft C, link in the alternate floating point support library.

This page intentionally left blank

Appendix B: Interrupt Map

This page intentionally left blank

Appendix B: Interrupt Map

Interrupt Map

Consult the [AM188ES User's Manual](#) regarding the use of any interrupts not listed here. This listing is only for hardware interrupts used in the DOS Stamp™.

Vector # (hex)	Vector Stored At (hex)	Description
08	00020	Timer 0 (used by BIOS for 18.2 Hz timer)
09	00024	Reserved for AMD use
0A	00028	DMA0 interrupt / INT5
0B	0002C	DMA1 interrupt / INT6
0C	00030	INT0 (available on SBI bus)
0D	00034	INT1 (available on SBI bus)
0E	00038	INT2 (available at GPIO8)
0F	0003C	INT3 (used by real time clock)
10	00040	INT4 (available at GPIO0)
11	00044	COM2 (also used by BIOS Equipment Check function)
12	00048	Timer 1 (also used by BIOS Memory Size function)
13	0004C	Timer 2 (not usable for timer; used by BIOS Disk Services function)
14	00050	COM1 (also used by BIOS Serial Services function)

Table B-1: Interrupt Map

This page intentionally left blank

Appendix C: Connections

This page intentionally left blank

Appendix C: Connections

The following header pin tables are referenced to the top side of the PCB. Dual-row header pins are odd numbered (1, 3, 5, 7...) in the row with the square pad (pin 1), and even numbered (2, 4, 6, 8...) in the other row.

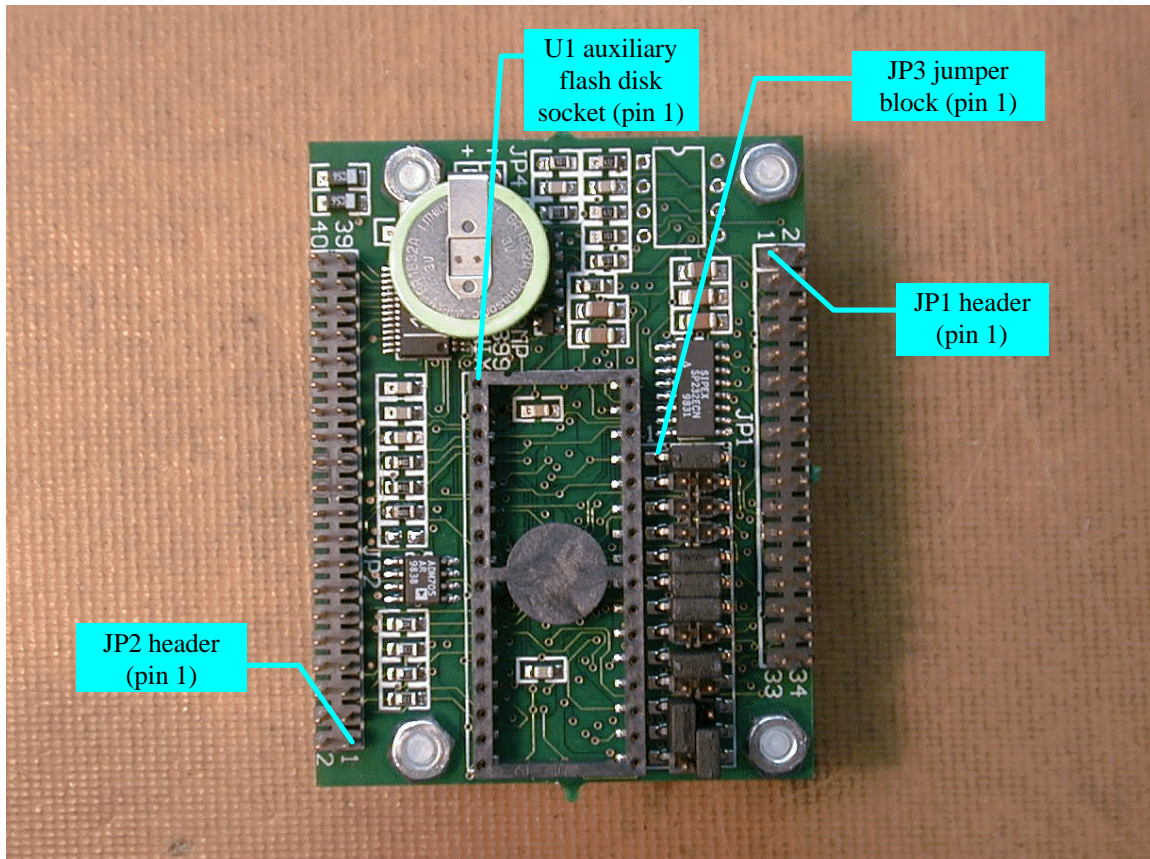


Figure C-1: DOS Stamp™ Header Locations

Appendix C: Connections

JP1 Header

JP1 is designed for easy connection to I/O module mounting boards (opto racks). When the DOS Stamp™ powers up, each pin defaults to the Primary Function as an input with pull-up, or pull-down as noted. Pull-ups and pull-downs are approximately 10K. It is up to your application software to reprogram the function of each pin as appropriate.

JP1 Pin	Primary Function / Alternate Function
1	PIO28 (General Purpose I/O 14) / COM2 Receive Data (RX) TTL level
all even	Ground
3	PIO27 (General Purpose I/O 13) / COM2 Transmit Data (TX) TTL level
5	PIO18 (General Purpose I/O 11) / ADC End of Conversion. Do not drive with external signal when ADC option is installed.
7	PIO19 (General Purpose I/O 12) / COM2 Request To Send (RTS) TTL level
9	PIO20 (General Purpose I/O 3) / COM1 Request To Send (RTS) TTL level
11	PIO21 (General Purpose I/O 15) / COM1 Clear To Send (CTS) TTL level
13	PIO17 (General Purpose I/O 10) / -PCS1 SBI Bus Chip Select 1
15	PIO16 (General Purpose I/O 9) / -PCS0 SBI Bus Chip Select 0
17	PIO31 (General Purpose I/O 8) / INT2 Interrupt Request Input
19	PIO1 (General Purpose I/O 7) / Timer 1 Output (PULLED DOWN)
21	PIO0 (General Purpose I/O 6) / Timer 1 Input
23	PIO10 (General Purpose I/O 5) / Timer 0 Output (PULLED DOWN)
25	PIO11 (General Purpose I/O 4)
27	PIO13 (General Purpose I/O 2) / DMA Channel 1 Request Input or INT6 Interrupt Request Input
29	PIO12 (General Purpose I/O 1) / DMA Channel 0 Request Input or INT5 Interrupt Request Input
31	PIO30 (General Purpose I/O 0) / INT4 Interrupt Request Input
33	+5 V

Table C-1: JP1 Header Pin Functions

Appendix C: Connections

JP2 Header

JP2 is a multipurpose header which has signals for the ADC, serial ports, timed powerup, and SBI bus.

JP2 Pin	Function
1	AD0 (Address/Data bit 0)
2	ALE (Address Latch Enable)
3	AD1 (Address/Data bit 1)
4	-RD (Read Strobe)
5	AD2 (Address/Data bit 2)
6	-WR (Write Strobe)
7	AD3 (Address/Data bit 3)
8	A0 (Address bit 0)
9	AD4 (Address/Data bit 4)
10	A1 (Address bit 1)
11	AD5 (Address/Data bit 5)
12	A2 (Address bit 2)
13	AD6 (Address/Data bit 6)
14	INT0 (Interrupt Request input 0)
15	AD7 (Address/Data bit 7)
16	INT1 (Interrupt Request input 1)
17	-RESET (Reset output, active low)
18	-EXTRESET (External Reset input, active low)
19	-KICKSTART (external startup signal, active low)
20	-PCS0 (Peripheral Chip Select 0 output, active low)
21	-PCS1 (Peripheral Chip Select 1 output, active low)
22	-PWRUP (Powerup output, active low, open drain)
23	TXRX1+ (COM2 RS-485 Transmit/Receive +, input/output)
24	TXRX1- (COM2 RS-485 Transmit/Receive -, input/output)
25	RX0 (COM1 RS-232 Receive input)
26	RTS0 / TX1 (COM1 RS-232 Request To Send output / COM2 RS-232 Transmit output, function depends on JP3 jumper selection)
27	TX0 (COM1 RS-232 Transmit output)
28	CTS0 / RX1 (COM1 RS-232 Clear To Send input / COM2 RS-232 Receive input, function depends on JP3 jumper selection)
29	Ground
30	+5 V
31	Ground
32	VREF (ADC Voltage Reference output, 4.095 V)
33	AI7 (ADC Input 7)
34	AI6 (ADC Input 6)

Appendix C: Connections

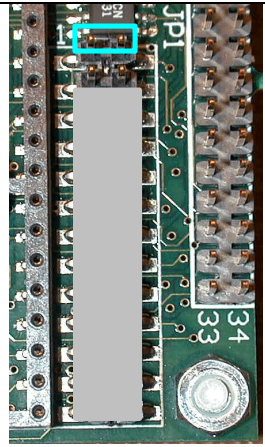
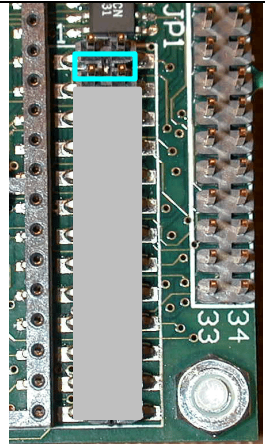
35	AI5 (ADC Input 5)
36	AI4 (ADC Input 4)
37	AI3 (ADC Input 3)
38	AI2 (ADC Input 2)
39	AI1 (ADC Input 1)
40	AI0 (ADC Input 0)

Table C-2: JP2 Header Pin Functions

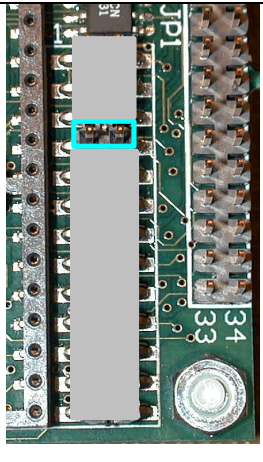
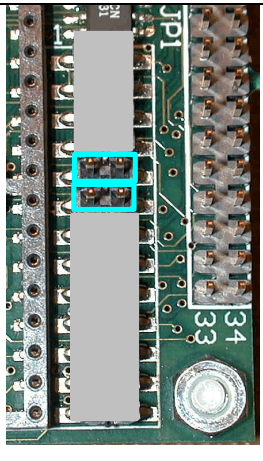
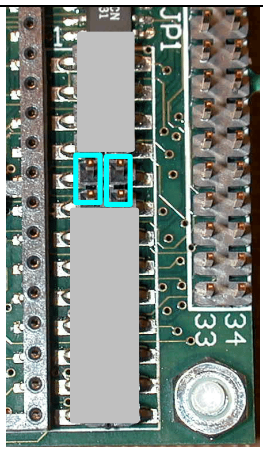
Appendix C: Connections

JP3 Header

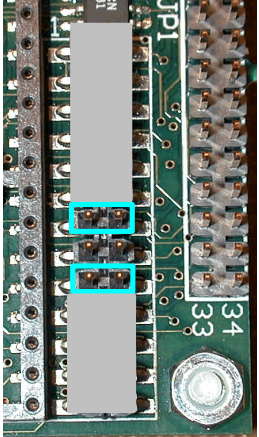
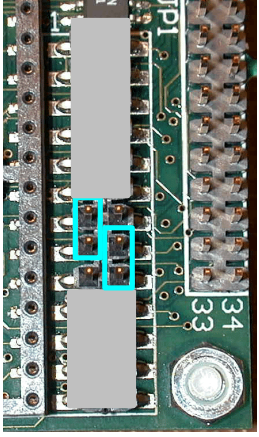
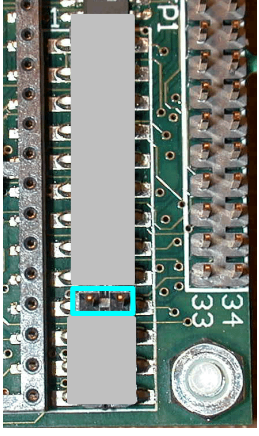
JP3 is a jumper selection header. Jumpers are used to configure the DOS Stamp™ in various ways. An installed jumper is indicated by a blue box outline enclosing the pins which are to be jumpered together. Unrelated jumpers are grayed out.

	<p>U1 Auxiliary Memory Device Selection:</p> <p>Install jumper from 1-2 for DiskOnChip 2000.</p> <p>Remove jumper from 3-4.</p>
	<p>U1 Auxiliary Memory Device Selection:</p> <p>Install jumper from 3-4 for flash chip.</p> <p>Remove jumper from 1-2.</p>

Appendix C: Connections

	<p>Boot Mode:</p> <p>To make the Setup Menu appear when booting, install jumper from 7-8. To bypass the Setup Menu when booting, remove jumper from 7-8.</p>
	<p>Boot Device Selection:</p> <p>Normal: install jumpers from 9-10 and 11-12</p>
	<p>Boot Device Selection:</p> <p>U1: install jumpers from 9-11 and 10-12</p> <p>This configuration is normally used only at the factory for using U1 as the boot device. Use of this mode also requires jumper 3-4 to be installed and 1-2 to be removed.</p>

Appendix C: Connections

	<p>Serial Port Configuration:</p> <p>To configure COM2 as RS-232 (TX and RX only), install jumpers from 13-14 and 17-18. Remove jumper from 15-16. COM1 RTS and CTS not available.</p> <p>-OR-</p> <p>To use GPIO13, remove jumper from 13-14.</p> <p>To use GPIO14, remove jumper from 17-18.</p>
	<p>Serial Port Configuration:</p> <p>To configure COM1 RS-232 with RTS and CTS, install jumpers from 13-15 and 16-18. COM2 RS-232 TX and RX not available, but COM2 may be used as RS-485 by installing a RS-485 transceiver chip (75176 type) in the U7 socket.</p> <p>-OR-</p> <p>To use GPIO3, remove jumper from 13-15.</p> <p>To use GPIO15, remove jumper from 16-18.</p>
	<p>RS-485 Bus Termination:</p> <p>To configure the COM2 RS-485 port with 100 ohm bus termination, install jumper from 19-20.</p>

Appendix C: Connections

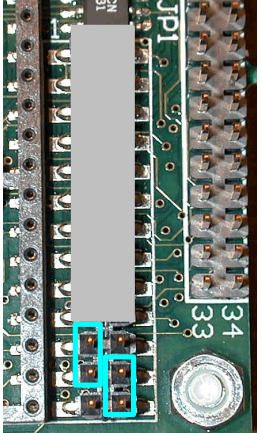
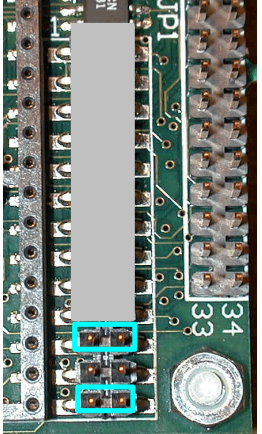
	<p>Volatile SRAM Configuration:</p> <p>If there is no need to retain data in SRAM when powered down, install jumpers from 21-23 and 24-26. This is the default configuration as shipped from the factory.</p>
	<p>Non-volatile SRAM Configuration:</p> <p>If there is a need to retain data in SRAM when powered down, install jumpers from 21-22 and 25-26. Remove jumper from 23-24. The SRAM will be powered from the on-board lithium battery when the DOS Stamp™ is unpowered. This will shorten the life of the lithium battery due to the standby retention current of the SRAM.</p>

Table C-3: JP3 Configuration Jumpers