# RTC-320

RTC series Microcontroller
using the DALLAS 80C320

**Technical Manual**

Release 1.0
8/3/95



**Micromint, Inc.**
902 Waterway Place
Longwood, FL 32750
www.micromint.com

## Conditions of Sale and Product Warranty

Micromint Inc. and the Buyer agree to the following terms and conditions of Sale and Purchase:

**1.** Micromint Inc. extends the following warranty: a factory-manufactured circuit board or assembly carries with it a one-year warranty covering both parts and labor. Any unit found to have a defect in materials or workmanship within this period will, at the discretion of Micromint Inc., be repaired or replaced.

**2.** Products distributed, but not manufactured by Micromint Inc. carry the full original manufacturer's warranty, usually 90 days. Such products include, but are not limited to, power supplies, sensors, I/O modules, LCD displays, battery-backed RAM modules, and disk drives.

**3.** A minimum inspection fee must be prepaid for the repair of units no longer under warranty. Call Micromint Inc. for information on current minimum charges.

**4.** Micromint Inc. will not be responsible for the repair or replacement of any unit damaged by user modification, negligence, abuse and mishandling, or improper installation.

**5.** Micromint Inc. is not responsible to the Buyer for any loss or claim of special or consequential damages.

**6.** All units returned for repair must first receive prior authorization from Micromint Inc. A return authorization number may be obtained by phone or letter. Please retain a record of this number, because most subsequent correspondence will refer to this authorization. Under no circumstances should any product be returned to Micromint Inc. without this authorization. Micromint Inc. assumes no responsibility for returns unaccompanied by an authorization number. All returns must be shipped prepaid. Insurance is recommended because losses by a shipping carrier are not the responsibility of Micromint. Repaired units will be returned postage and insurance paid.

**7.** Micromint Inc. reserves the right to alter any feature or specification at any time. This right extends to fees, charges, and any other conditions or warranties contained herein.

**REV. 5/93**

## COPYRIGHT



**Micromint, Inc.**
902 Waterway Place
Longwood, FL 32750
www.micromint.com

## DISCLAIMER

Devices sold by Micromint are covered by the warranty and patent indemnification provisions in its Terms of Sale only. Micromint makes no warranty, express, statutory, implied, or by description regarding the information set forth herein or regarding the freedom of the described devices from patent infringement. Micromint makes no warranty of merchantability or fitness for any purposes. Micromint reserves the right to discontinue production and change specifications and process any time and without notice. This product is intended for use in normal commercial applications. Applications requiring extended temperature and unusual environmental requirements, or applications requiring high reliability applications, such as military, medical life support or life-sustaining equipment, are specifically NOT recommended without processing by Micromint for such application.

Occasionally in this manual we refer to other manufacturers' products. Such references do not constitute an endorsement of these products, but are included for the purpose of illustration or clarification. We do not intend such technical information and interface data to supersede information provided by individual manufacturers.

## Table of Contents

## MICROCONTROLLER EVOLUTION

The 8031 microcontroller is probably the most widely used core for embedded processing in the industry. It's price/performance ratio as a general purpose controller helps to keep it a popular choice for even new designs. It is not surprising that manufacturers have based many of today's newer processors on the 8031 core. While adding bells and whistles may make the newer parts more desirable, many designers are looking for faster processing. Dallas Semiconductor has made the first real improvement to the 8031 core by reducing the number of clock cycles necessary per instruction cycle to four (vs. twelve).

This factor of 3 increase in speed using the same crystal allows the user's original object code to execute up to 150% faster. (Some instructions execute at less than 3 times as fast but none fewer than 1.5 times.) The maximum crystal frequency has been elevated as well. The 33MHz upper limit is triple again over 11MHz standard. Fast programs will now literally scream.

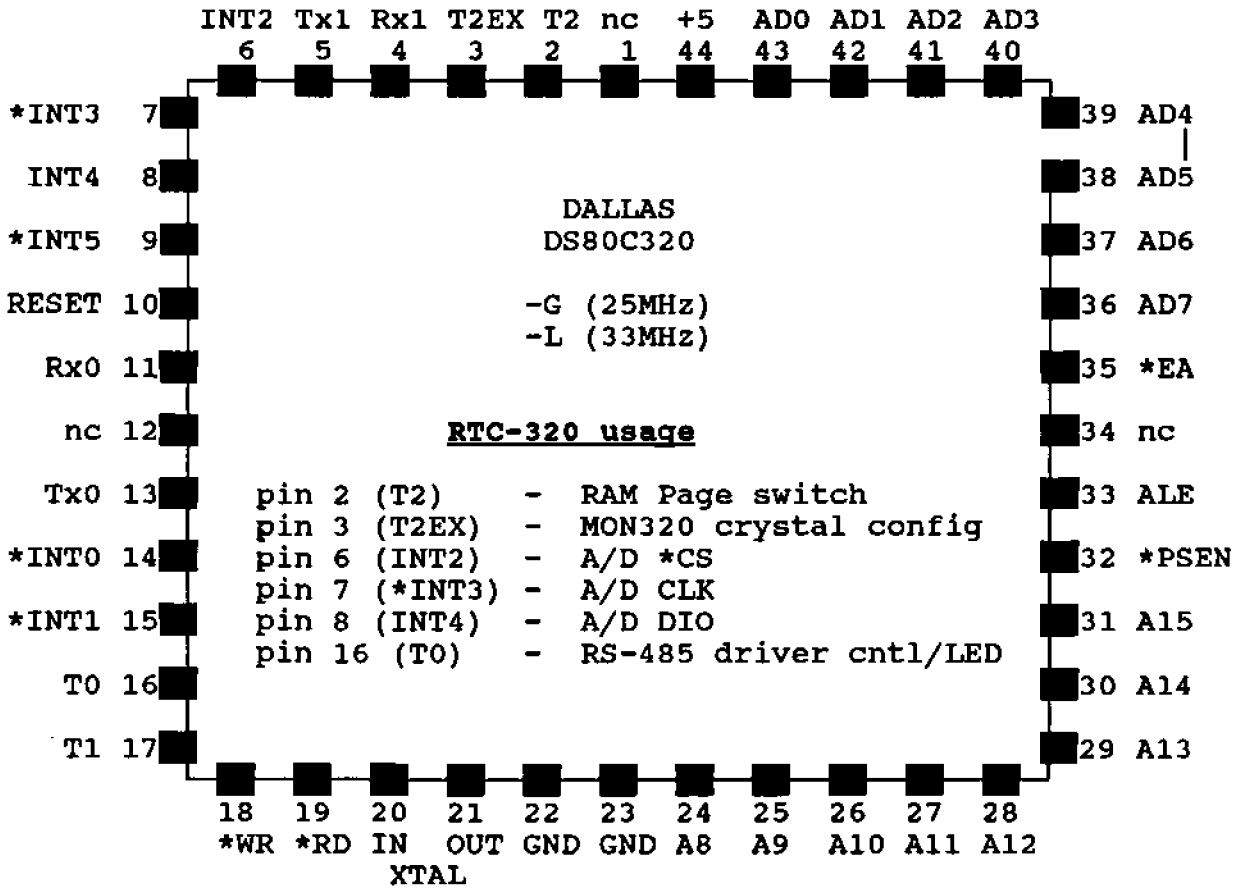
What does this do to the I/O devices connected? As far as external CODE space is concerned, installed devices (EPROMs) must comply with the timing specifications based on the crystal speed and glue logic delays. DATA space devices (SRAMs and peripheral chips) need NOT be upgraded if the user selects the appropriate number of 'Stretch Cycles' (WAIT states) which are added to the access times of the MOVX instructions. The number of stretch cycles are dynamically alterable to allowing the user complete control.

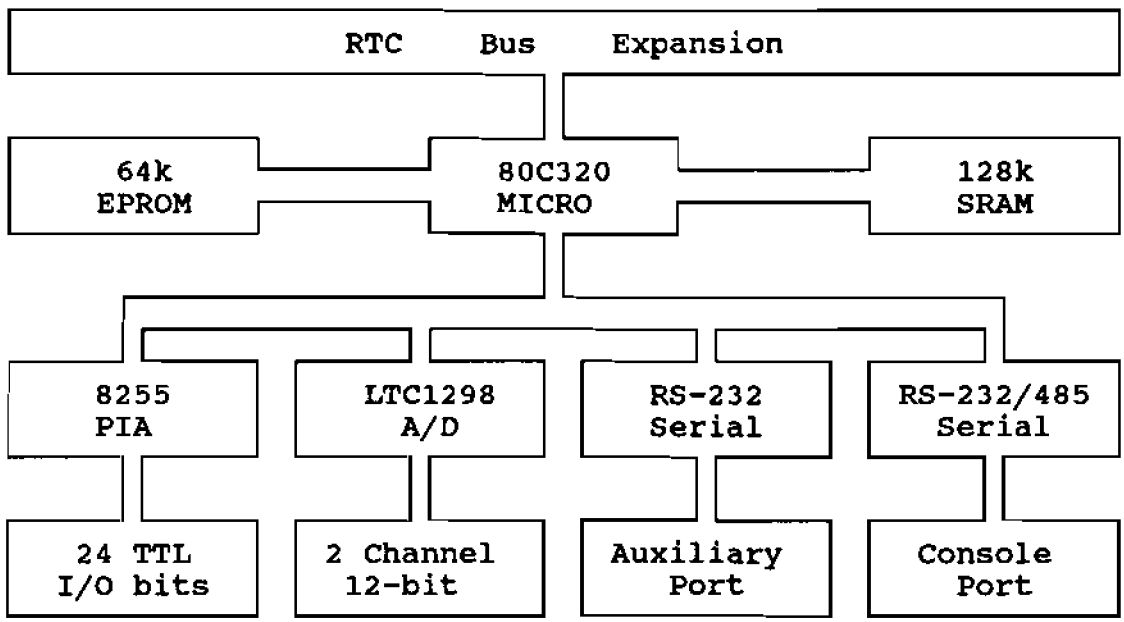
| Crystal speed | CODE space | DATA space (default) |
|---|---|---|
| 22.1184MHz | 100nS | 150nS |
| 33.1776MHz | 55nS | 100nS |

The RTC-320 is a plain-brown-wrapper 80C320 controller that is optimized with the most requested I/O on-board for minimal configuration applications. While the RTC-320 may be perfect for most single board applications, it does contain the RTC expansion bus for adding on those special bits of I/O which might be necessary for your specific application. If that special I/O isn't available as a standard expansion board, it can be easily added to an RTC prototying expansion board.

The RTC system measures only 3.5 inches square and uses vertical stacking connectors for I/O expansion. The RTC-320 processor board contains the 80C320 processor, EPROM and RAM memory, address decoding and buffering, 24-bits of parallel I/O , a 2-channel 12-bit A/D convertor, and two RS-232 serial ports (one alternately available as RS-485. Each vertically stacked expansion board only increases the system height by 3/4 inch.

## 80C320 PIN DESCRIPTION



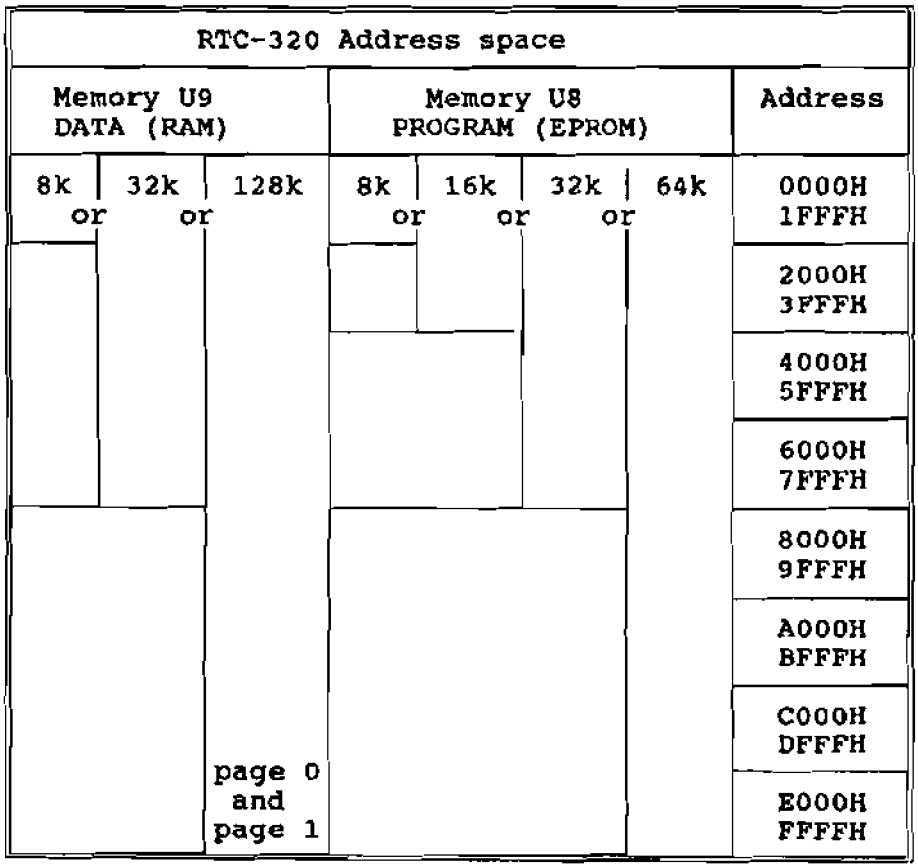

**RTC-320 Block Diagram**

## RTC320 EXTERNAL ADDRESSING SPACE

The RTC-320 microcontroller can directly address 64k of external memory. That is, 64K of DATA memory and 64K of PROGRAM memory. The *RD and *WR lines control DATA memory (READ/WRITE and I/O) and the *PSEN line controls PROGRAM memory (READ-only memory). Overlapped space occurs when the *RD and *PSEN lines are combined, a combined DATA/PROGRAM space is useful when CODE must be executed out of RAM for various reasons.

Many combinations of 8k through 128k RAMs and 8k through 64k EPROMs are possible on the RTC-320 board. **The first 16k of address space (0000H-3FFFH) is always separated into DATA space for RAM and CODE space for EPROM. The next 16k of address space (4000H-7FFFH) can be separated or combined spaces. The last 32k of address space (8000H-FFFFH) can also be assigned as either separate or combined DATA and CODE space.**

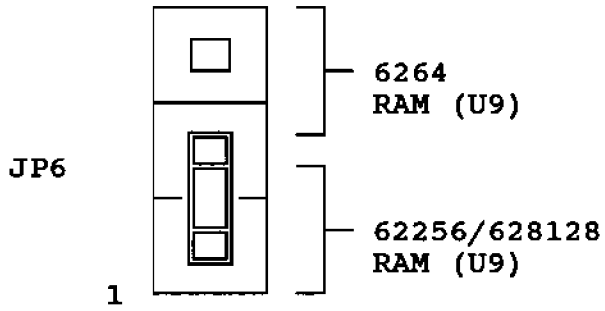
The following chart shows conventional addressing for the memories used on the RTC-320:



## RTC-320 Memory Device Configuration

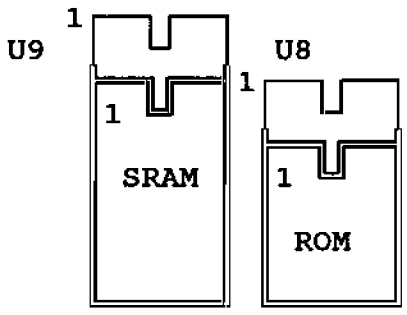
### SRAM SIZE

The RTC-320 provides IC sockets for two memory devices. Socket U9 has been designated as the SRAM position and U8 as the EPROM position. SRAMs from 8k to 128k by 8 may be used in the RTC-320, however the user must know how much memory is installed and access only that area because the RAM will wrap around and be duplicated in multiple address blocks. Only one jumper needs to be placed to configure the SRAM socket U9 for your SRAM chip, JP6.



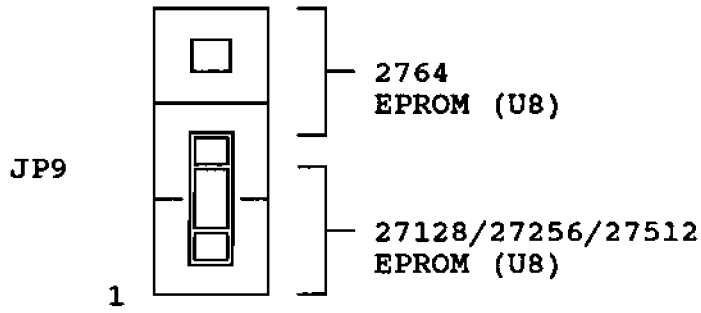

SRAM size configuration jumper JP6 is shown configured for 628128 (128kx8) SRAM.

NOTE: smaller memory devices should be lower justified in IC sockets.

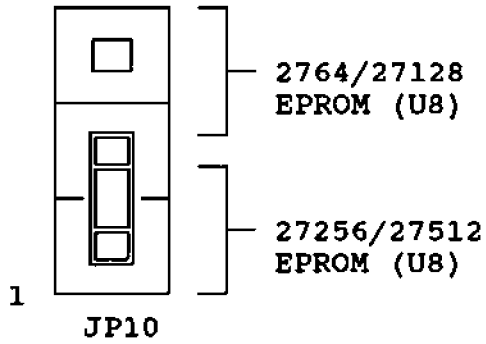


## EPROM SIZE

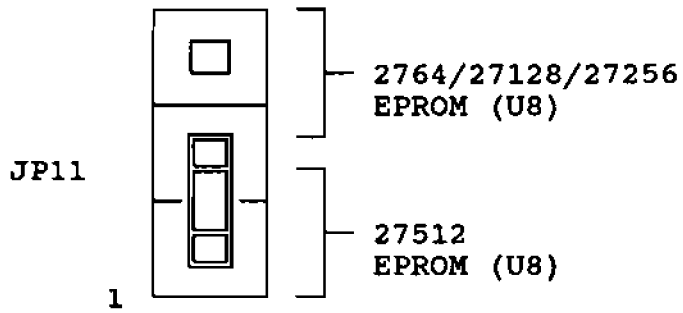
Four different size EPROMs may be used in the RTC-320 ROM socket U8. As with the RAM, EPROMs smaller than 64k will continually wrap around and be duplicated in multiple address blocks. Three jumpers need to be placed to configure the EPROM socket U8 for your EPROM chip; JP9, JP10, and JP11.




The EPROM configuration jumper JP9 is shown configured for a 27512.




The EPROM configuration jumper JP10 is shown configured for a 27512.



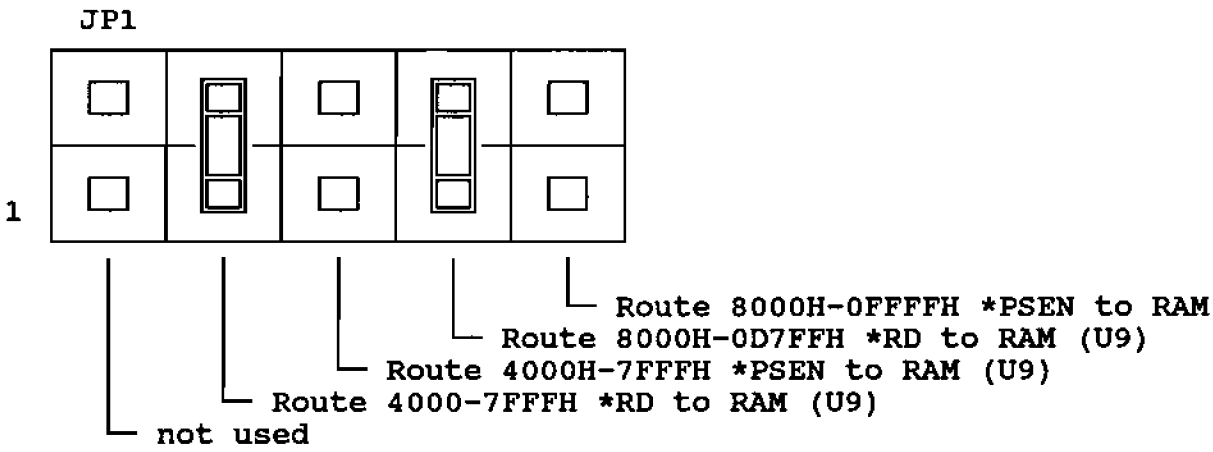

The EPROM configuration jumper JP11 is shown configured for a 27512.

## Defining DATA and CODE Space

The 64k address space is divided into three areas: 0000H-3FFFH, 4000H-7FFFH, and 8000H-FFFFH. The first area (first quadrant) is permanently separated into non-overlapping DATA and CODE spaces. Any access to CODE space (*PSEN 0000H-3FFFH) goes to the U8 EPROM socket and any access to DATA space (*RD *WR 0000H-3FFFH) goes to the SRAM socket.

The second area 4000H-7FFFH (second quadrant) can be configured as either overlapping or non-overlapping DATA and CODE spaces. The *PSEN (CODE read) line can be directed toward either the SRAM or the EPROM allowing a code fetch for execution from either device. The *RD (DATA read) line can be directed toward either device as well (although, this may not make logical sense to you).

The third area 8000H-FFFFH (third and fourth quadrants) can be configured as either overlapping or non-overlapping DATA and CODE spaces. The *PSEN (CODE read) line can be directed toward either the SRAM or the EPROM allowing a code fetch for execution from either device. The *RD (DATA read) line can be directed toward either device as well (although again, this may not make logical sense to you).

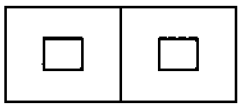



JP1 is shown configuring all *RD signals to RAM and all *PSEN signals to EPROM. This separates the DATA and CODE spaces providing non-overlapping spaces (64k [potential] DATA space and 64k [potential] CODE space).

## EXTERNAL CODE SELECTION

The 80C320 microcontroller requires *EA (pin 31 on the microcontroller) to be pulled down to a logic low level. This instructs the processor to start executing machine language code starting at address 0000H (CODE space).

JP2

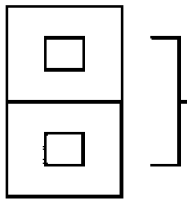


Enable External EPROM CODE at 0000H

**JP2 shows the microcontroller enabled for external code execution**

## RESETTING THE RTC-320

Power-ON reset of the RTC-320 board occurs whenever power reaches approximately 4 volts. Manual reset is accomplished by momentarily shorting the pins of J2 together or connecting a normally open push button switch to J2 and pressing it.



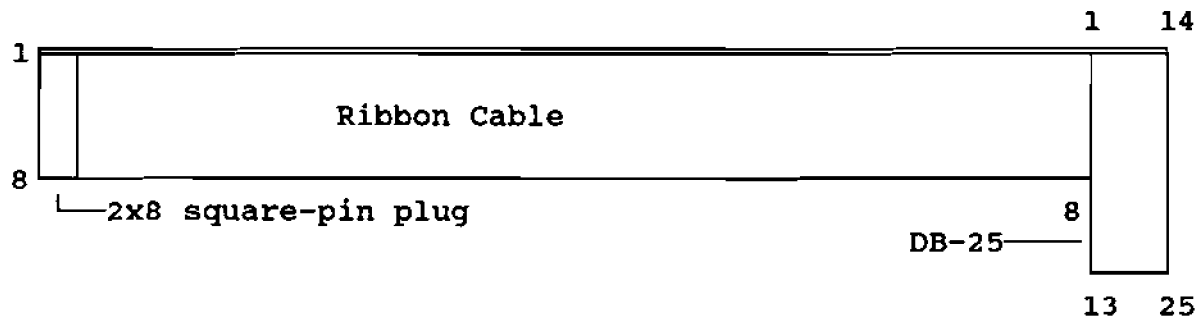
Momentarily 'short' pins to RESET board

J2

**Use J2 for connecting a normally open push button switch as an external system RESET.**

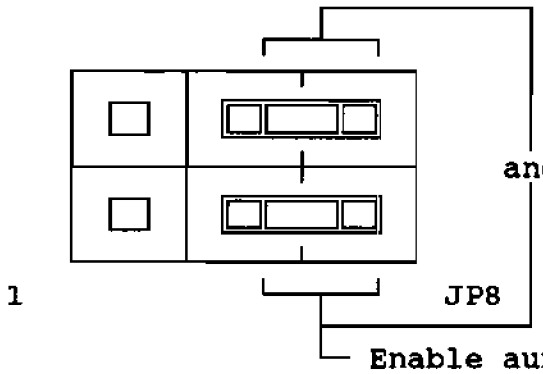
## RS-232 COMMUNICATIONS

The 80C320 contains two full-duplex serial channels. TTL-level serial signals are converted to ±10-volt RS-232-compatible signals by U6, the MAX232 device. A 16-lead flat ribbon cable made with a DB-25 at one end (for connection to DTE terminal device and a 16-pin dual row plug header (connection to the RTC-320) will permit serial communication. The console port is available through JP4 and the auxiliary port is available through J3.



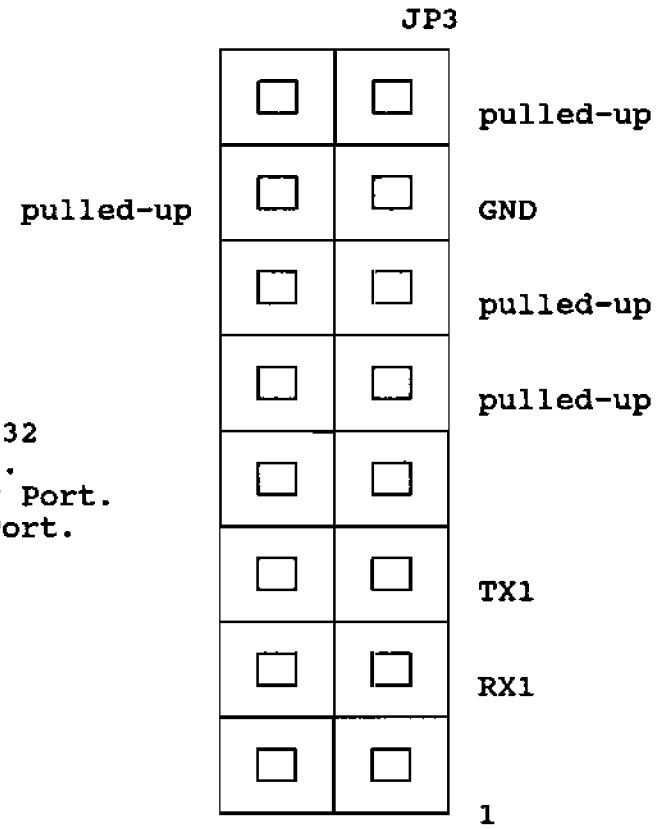

**Cable required for RS-232 communications**

| To eliminate unwanted noise on the RX input to the processor, remove the unused 75176 line driver chip. (U4) |
|---|

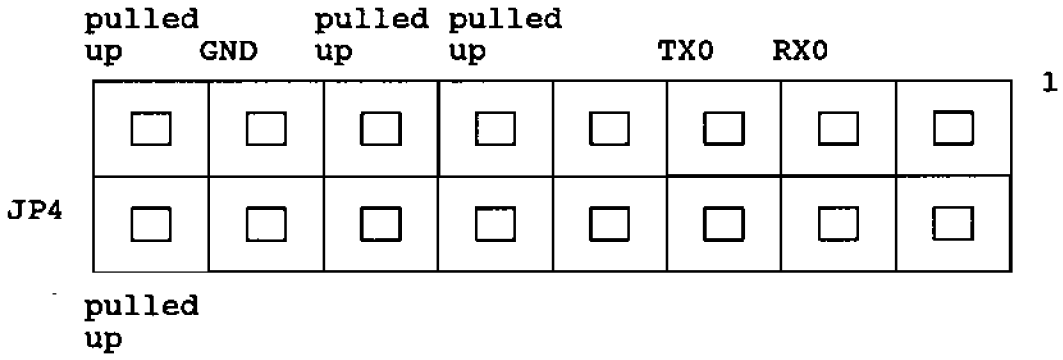



JP8 must have both jumpers installed as shown to enable the auxiliary serial port.

Enable auxiliary RS-232 for 80C320



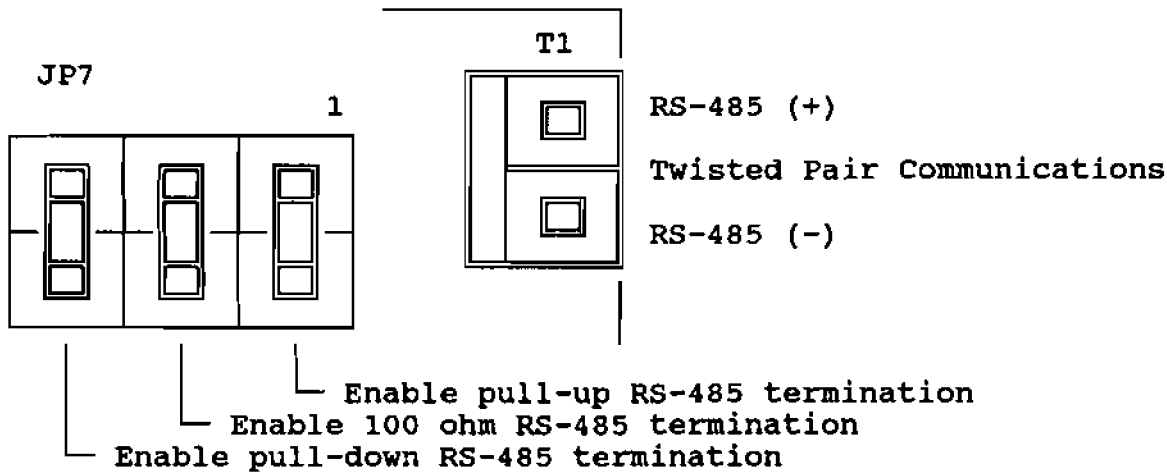

JP3 and JP4 are RS-232 interface connectors.
JP3 is the Auxiliary Port.
JP4 is the Console Port.



## RS-485 COMMUNICATIONS

RS-485 communications over a single twisted pair can include multiple (up to 32) devices. Since each device can transmit and receive, certain protocols must be adhered to to prevent message collision. The simplest being "listen to the line and transmit only if free". (The protocol you use will depend on the application and is beyond the scope of this manual.) JP7 enables a termination resister across the twisted pair and should be installed only on the microcontrollers located at the extremes of the twisted pair (one at each end.) If low power operation is of great concern and the RS-485 is not being used, current consumption can be reduced by removing the 75176. (Actually the 75176 should be removed whenever RS-485 is NOT being used.)

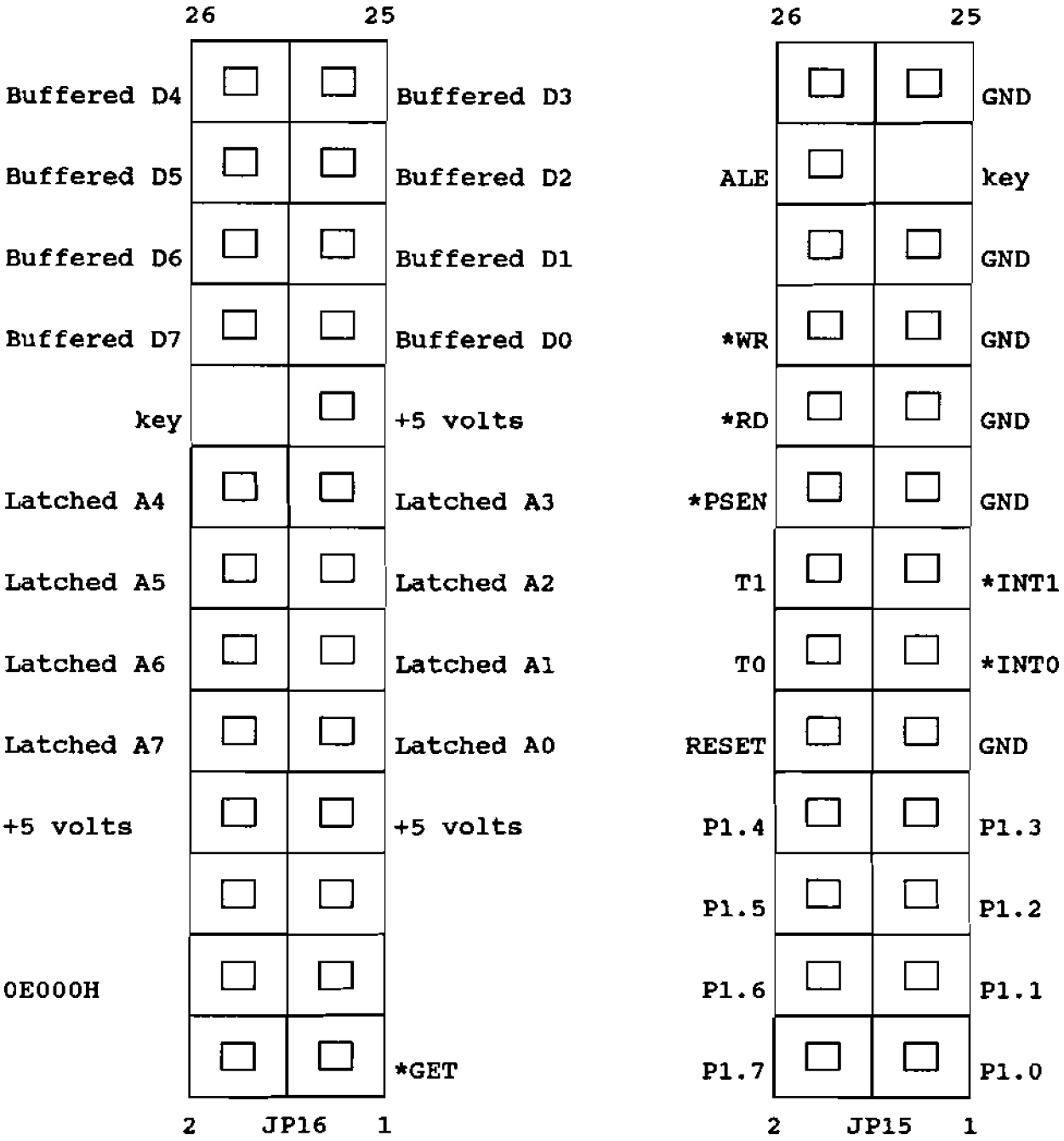
A set of screw terminal blocks are provided for RS-485 twisted pair communication.




JP7 shows termination of the RS-485 lines enabled

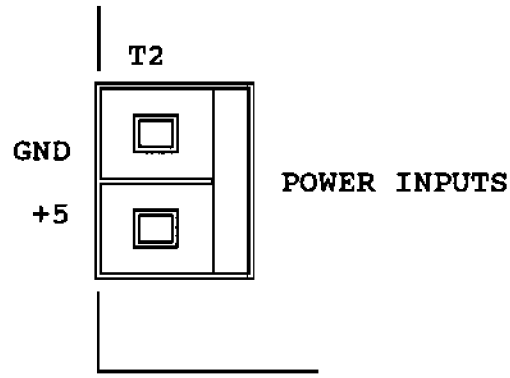
## VERTICAL-STACKING EXPANSION HEADER

The small size of the RTC-320 mirocontroller board is not compromised by expanding I/O through the expansion connector. The footprint remains the same as each I/O board only adds 3/4 of an inch to the height of the system. I/O expansion is obtained through a vertical header system making a backplane unnecessary. The data bus and latched low-order address bus passed are through the expansion header along with control lines and power. In place of the upper address bus, the upper 8K block is decoded and passed through the expansion header as a block select.



## RTC-320 POWER REQUIREMENTS

A set of screw terminal blocks are provided for +5 volt power and ground. 200mA are required for 22 MHz operation. 250mA are required for 33Mhz operation.

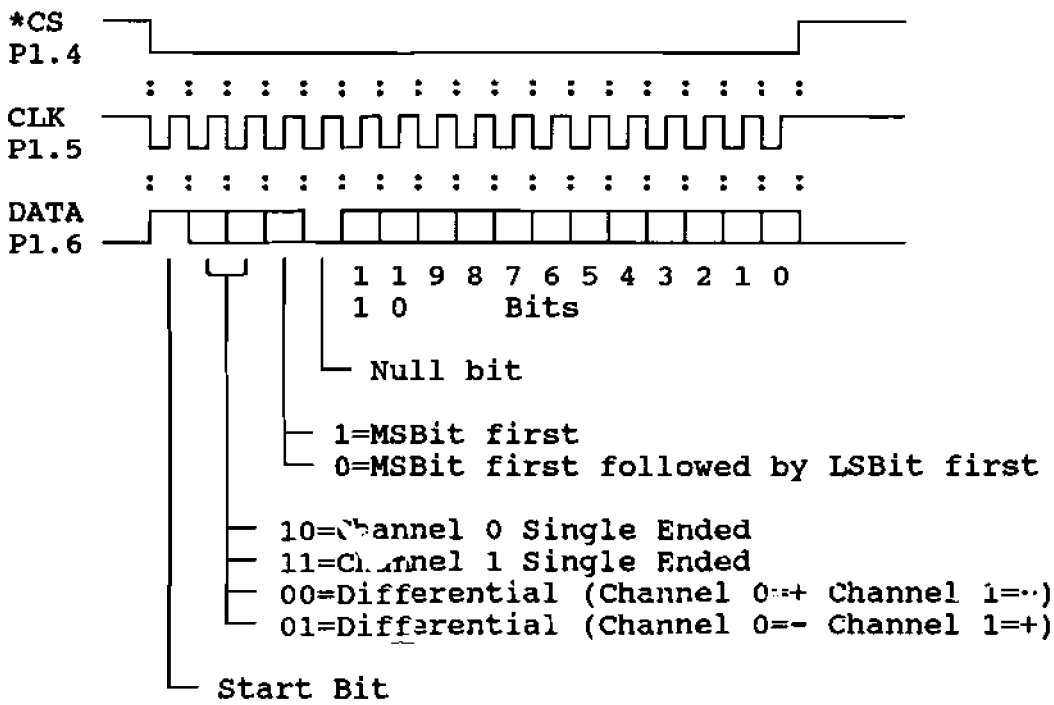


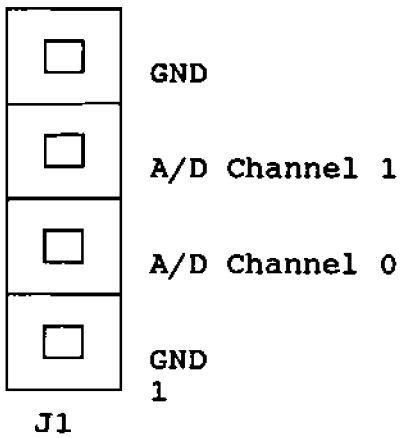

## STAND-ALONE I/O CONNECTIONS

In an attempt to allow the RTC-320 to be more widely used in a standalone mode, the most popular I/O was added. Additional I/O includes 24-bits of digital I/O plus a two channel 12-bit A/D converter.

## A/D Convertor

A three wire serial A/D (LTC1298) is used to provide the RTC-320 with two channels of 12-bit analog to digital conversion. The conversion is based on the clock speed, the maximum clock speed is 2.5µS high and 2.5µS low. Input impedance looks like 500Ω in series with 20pF. At maximum speed (about 90µS cycle time) the DC input current is about 1.56µA. A 750Ω source impedance will cause about 1 bit of full-scale error. If the source resistance can not be small, then the clocking speed can be reduced by a factor of 10 or even 100.




Analog input channels are available on a 1x4 square pin header.

This skeletal program suggests one possible routine for reading the A/D convertor. The user needs to set the MODE and provide two registers for the 12-bit result.

```
CLK         EQU  P1.5
DIO         EQU  P1.6
CS          EQU  P1.4
MODE        EQU  0BH          ; Single Ended Channel 0
;               0FH           ; Single Ended Channel 1
;               09H           ; Differential +=Chan0, -=Chan1
;               0DH           ; Differential +=Chan1, -=Chan0
A_D_MSB     EQU  ???          ; Register to hold upper 4 bits
A_D_LSB     EQU  ???          ; Register to hold lower 8 bits
```

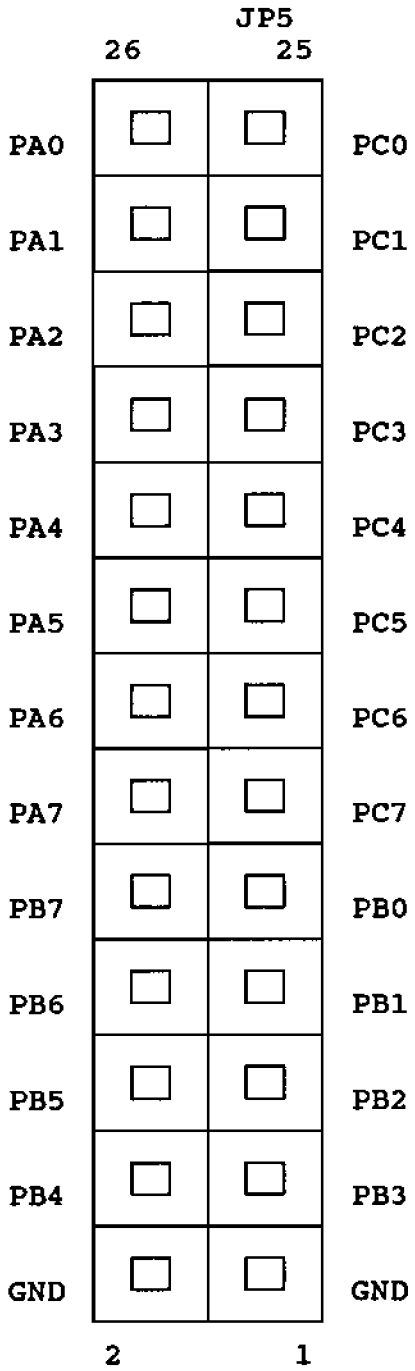
The actual routine start here:

```
LTC:        SETB CS
            SETB CLK
            MOV  B,#04H
            MOV  A,MODE
            CLR  CS

D_OUT:      CLR  CLK
            RRC  A
            MOV  DIO,C        ; 1 or more as necessary dependent
;           NOP               ; on execution speed, min 2.5µs
            SETB CLK          ; between clock edges
;           NOP
            DJNZ B,D_OUT

            CLR  A
            MOV  B,#05H
D_IN_M:     CLR  CLK
;           NOP
            MOV  C,DIO
            RLC
            SETB CLK
;           NOP
            DJNZ B,D_IN_M

            MOV  A_D_MSB,A
            MOV  B,#08H
D_IN_L:     CLR  CLK
;           NOP
            MOV  C,DIO
            RLC
            SETB CLK
;           NOP
            DJNZ B,D_IN_L

            MOV  A_D_LSB,A
            SETB CS
```

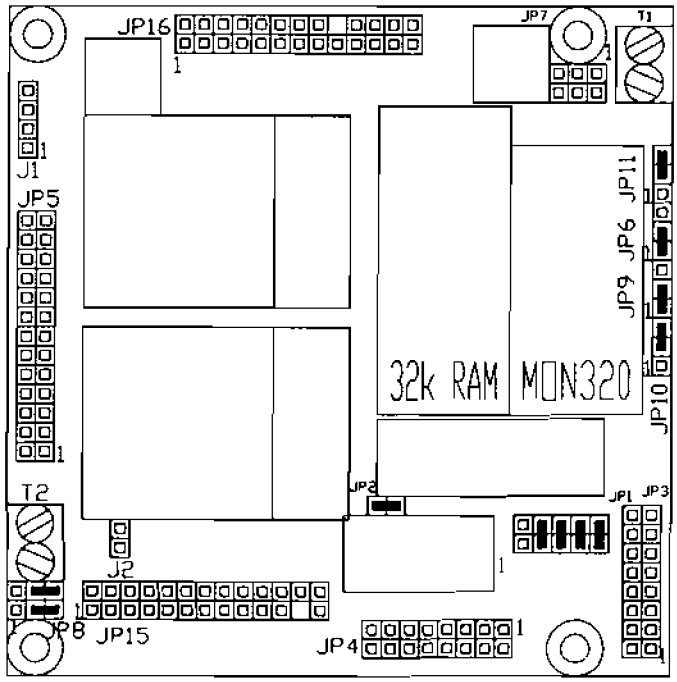
## 8255 PIA

The 8255 is user programable through the MODE port at address 0DC03H. The I/O is split into three Ports. Port A is available through address 0DC00H. Port B is available through address 0DC01H. Port C is available through address 0DC02H. **Upon power-up the 8255 PIA is configured as all input bits.** The user can change the configuration at any time by writing to the MODE Port. NOTE: Any change to the MODE Port resets all outputs to logic low.

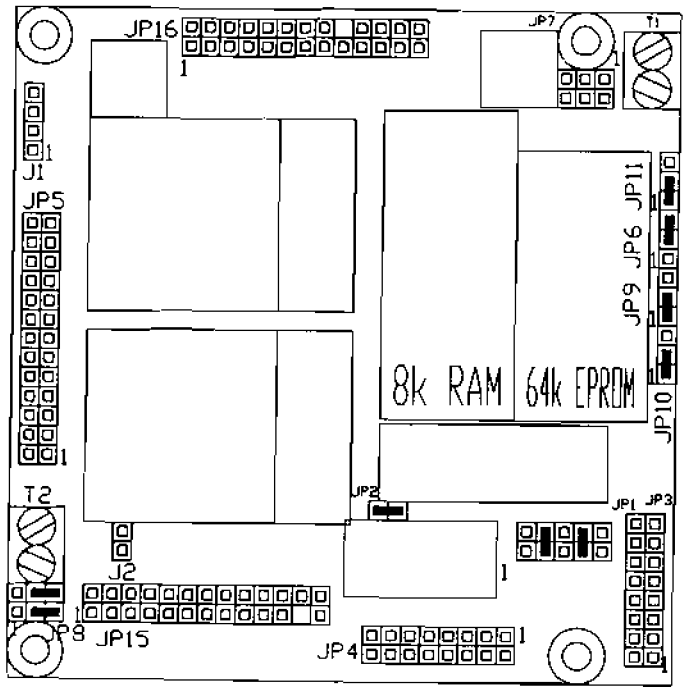
| 8255 Configuration | | | | |
|---|---|---|---|---|
| Port A | Port B | Port C upper nybble | Port C lower nybble | MODE Value |
| IN | IN | IN | IN | 09BH |
| IN | IN | IN | OUT | 09AH |
| IN | IN | OUT | IN | 093H |
| IN | IN | OUT | OUT | 092H |
| IN | OUT | IN | IN | 099H |
| IN | OUT | IN | OUT | 098H |
| IN | OUT | OUT | IN | 091H |
| IN | OUT | OUT | OUT | 090H |
| OUT | IN | IN | IN | 08BH |
| OUT | IN | IN | OUT | 08AH |
| OUT | IN | OUT | IN | 083H |
| OUT | IN | OUT | OUT | 082H |
| OUT | OUT | IN | IN | 089H |
| OUT | OUT | IN | OUT | 088H |
| OUT | OUT | OUT | IN | 081H |
| OUT | OUT | OUT | OUT | 080H |




Digital I/O connection are available on a 2x13 sqaure pin header.




Suggested Jumper Configuration for 32k RAM (62256) and MON320 (27128).




Suggested Jumper Configuration for 8k RAM (6264) and USER EPROM (27512).

To eliminate unwanted noise on the RX input to the processor, remove the 75176 line driver chip (U4) if not using RS-485.

## GETTING STARTED

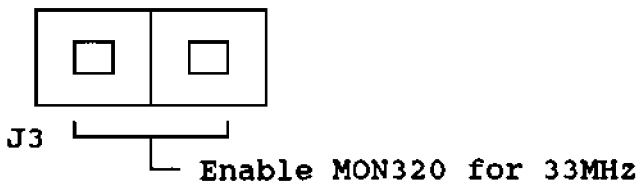
### RTC-320

The RTC-320 does not have any internal program, so the user must program an EPROM with machine language code which will start execution at 0000H. The following code can be programmed into a blank EPROM and will turn the LED on and off, verifying the ability to execute code. Refer to DALLAS's High-Speed Microcontroller Data Book for the particulars on the internal registers of the 80C320.

```
0000H  15H  F0H          ;DEC  B
0002H  D5H  F0H  F5H     ;DJNZ B,0000H
0005H  14H               ;DEC  A
0006H  70H  F8H          ;JNZ  0000H
0008H  B2H  B4H          ;CPL  T0
000AH  01H  00H          ;AJMP 0000H
```

## MON320

When using the MON-320 (monitor EPROM), the code looks at P1.1 on the processor to determine what crystal is connected. The user must configure J3 correctly or the sign-on BAUD rate will be incorrectly configured.




J3 is shown configured for 22MHz (or 11MHz). NOTE: You can use this jumper in your own application once MON320 has been remove.
It can be examined through P1.1 (high=no jumper low=jumper).