Note: there is nothing to hand in for this assignment. In this homework assignment, you will explore:

- Timers/Counters
- Interrupt System
- Atmel AT89C51RC2 Web Site and Resources
- Final Project

The assigned reading will be available on the course web site in PDF format.

1. Read section 6.2 of the C501 User's Manual regarding the timers and counters.
2. What is the purpose of the TR0 and TR1 bits in the TCON register?
3. Given an 11.0592 MHz CPU clock, what is the maximum time you can measure using a single continuously running C501 8-bit timer without the timer rolling over from FFh to 00h?
4. Read section 7 of the C501 User's Manual regarding the interrupt system.
5. What is the difference between a level triggered interrupt and an edge triggered interrupt?
6. Suppose the IE register contains 85h, the TCON register contains 05h, and the IP register contains 01h.
   (a) What happens if pin INT0* goes low, followed 12 microseconds later by pin INT1*?
   (b) What happens if pin INT1* goes low, followed 12 microseconds later by pin INT0*?
7. Some processors have an NMI (non-maskable interrupt) pin. Why is this type of interrupt important in embedded systems? Note that the 8051 does not have an NMI.
8. Start exploring the Atmel web site for the Atmel AT89C51RC2 processor.

http://www.atmel.com/dyn/products/product_card.asp?family_id=604&family_name=8051+Architecture&part_id=2854

   Review the available support files, including documentation and tools and software.

9. Thoroughly read the requirements document for the final project and final project PDR.
10. Brainstorm some ideas for your final project. Give some thought about whether you want to work alone or on a team. Each person in the class will need to discuss their initial ideas for a minute or two during class in week 8 or 9, in preparation for the final project preliminary design review (PDR) around week 10.

    Now is a good time to start ordering long lead-time parts which you will need for your final project work next month. Don't wait, as it can sometimes take several weeks to get parts. Be sure to order through-hole parts, unless surface mount components are all that are available. If using surface mount parts, have a plan for building your own PCB or obtaining small adapter boards that convert surface mount components into through hole components.

    Do not use surface mount parts which are designed exclusively for solder reflow ovens. Packages such as LGAs or BGAs have all their connections underneath the chip, and you cannot solder underneath the package with a soldering iron.

    When choosing ICs for your final project, make sure you choose chips that have good data sheets and application notes. Poor documentation can result in project failure.

Note: there is nothing to hand in for this assignment. In this homework assignment, you will explore:

- Final project ideas
- Serial communication (RS-232, RS-423, RS-422, RS-485, USB)
- Atmel AT89C51RC2, Bootloader and In System Programming (ISP)

The assigned reading will be available on the course web site in PDF format.

---

1. Think about your final project from hardware, firmware, and testing perspectives.
2. Read the sections of the C501 and/or the 80C51/Atmel AT89C51RC2 manuals which refer to serial communication. Focus the majority of your attention on serial mode 1 and learn how the RI and TI flags in the SCON register may be polled or may be used to generate a serial interrupt.
3. Which timer(s) can be used to generate the baud rate for serial port transmissions?
4. Suppose the serial port interrupt is enabled. What will happen if the serial port ISR firmware does not clear the RI/TI flags when a serial interrupt occurs?
5. What is the difference between synchronous serial data transmission and asynchronous serial data transmission? You may want to look on the web or in a text for more information regarding this topic.
6. What mode of the 8051 would you use to transmit asynchronous serial data with eight data bits, one start bit, and one stop bit? What is the purpose of the start bit? What is the purpose of the stop bit?
7. When the serial port on the 8051 is receiving data, what algorithm does it use to determine if the bit it is receiving is a zero or a one? Does it use the same algorithm for both start bits and data bits?
8. Write initialization code in assembly which will configure the microcontroller to receive and transmit data at 9600 baud using serial mode 1. Assume a clock frequency of 11.0592 MHz and use Timer 1 to generate the baud rate. Write an infinite loop which constantly transmits the character 'U'.
9. MIDI (Musical Instrument Digital Interface) is an asynchronous serial communication protocol. The official MIDI specification requires a 31.25KHz baud rate (+/- 1%) and one start bit, eight data bits, and one stop bit. Choose an oscillator frequency and serial mode for the 80C51 (Note: the 80C51 does not have Timer 2) which will allow MIDI transmission. Can you use these same exact oscillator frequency and serial mode values if you want to transmit via RS-232 at 19.2Kbps? Explain.
10. Obtain and read the following document:
    - Dallas Semiconductor Application Note AN-83 (app83.pdf) "Fundamentals of RS-232 Serial Communications". For reference, the Dallas web site is: http://www.dallassemiconductor.com (now part of Maxim, at http://www.maxim-ic.com).
11. Obtain and read the following document: (at http://www.maxim-ic.com/)
    - Data sheet for the MAX232 RS-232 interface chip
12. Determine how the charge pump circuitry in the MAX232 provides a +10V and –10V output supply voltage when its supply is only +5V (handout available from instructor that shows charge pump). Briefly describe how the charge pump works. What values should be used for the charge pump capacitors? How is the output ripple related to the size of the output capacitors and why?
13. Read pages 86-94 and pages 21-25 of the AT89C51RC2 data sheet regarding the flash memory, bootloader process (page 94), processor memory map, and the internal XRAM. Read the data sheet section regarding the Atmel Bootloader and the application note regarding the C51 Bootloader and In System Programming.

(continued on next page)

14. **[Optional]** Obtain and read the following documents:
    - National Application Note AN-216 "Summary of Well Known Interface Standards".
    - Data sheets for the National DS1488 and DS1489 analog interface chips.
    - Data sheet for the Maxim MAX485E RS-485 interface chip

    Note that the 1488 and 1489 are often used in systems which have both positive and negative voltage supplies, while the MAX232 chips are often used in single voltage supply systems. For reference, the National Semiconductor web site is at: http://www.national.com.

15. **[Optional]** Describe the advantages and disadvantages of using RS-485 instead of RS-232. Include a description of the benefits that differential transmission has over single-ended transmission. Make sure you understand at a technical level how and why each transmission standard works. Why does differential transmission offer such high data rates as compared to single-ended transmission?

16. **[Optional]** Obtain and read an application note that describes an overview of USB. For example, https://www.intel.com/content/www/us/en/io/universal-serial-bus/universal-serial-bus.html

17. **[Optional]** Read about USB 2.0 at the USB developer's web site: http://www.usb.org/developers. While at that site, review some of the presentations and overview material about USB. Visit other USB web sites, such as http://www.lvr.com/usb.htm and http://www.usb-by-example.com.
    - What are some of the advantages of USB over RS-232 and the standard parallel port?
    - Why does the USB connector have longer power pins and shorter data pins?
    - What is the rated bandwidth of low speed, full speed, and high speed USB devices?
    - Why isn't there a clock signal defined on the USB connector?
    - How much current can a low power USB device draw from the bus?
    - What is USB On-The-Go? What does dual-role mean?

18. **[Optional]** What is an eye diagram and why is it important in serial bus design?

Note: there is nothing to hand in for this assignment. In this homework assignment, you will explore:

- Monitor Programs (PAULMON2) [also called ROM monitors or debug monitors]
- C Programming
- Make and makefiles
- Integrated development environments (e.g. Code::Blocks or other IDE)
- C Compilers (e.g. SDCC)

Note: Students may try the freely available SDCC compiler. In addition to the documentation distributed with SDCC, there are additional notes posted on the course web site for students in this course. Use the correct version of SDCC as indicated in the course materials.

1) Read about the PAULMON2 monitor program. Read the PAULMON2 manual (link available on the course web site). Familiarize yourself with the PAULMON2 commands.
   [Optional] Download the paulmon21.asm and extra.asm files, and configure and assemble them using the AS31 assembler (executable and documentation available on the course web site).
2) Refresh your memory regarding C programming:
   - What is the format/syntax for a simple program in C?
   - How do you use C preprocessor directives (e.g. #include, #define, etc.)?
   - What data types are supported in C?
   - What is a function? How do you call a function in C?
   - What is a "return type"?
   - What is a parameter?
   - What is "passing by value"? What is "passing by reference"?
   - How do you declare a variable?
   - What arithmetic operators are supported in C?
   - What logical operators are supported in C? Remember to consider bit-wise logical operators.
   - What assignment operators are supported in C? How do you assign a value to a variable?
   - What precedence do the operators have in C?
   - What is the syntax for a "for" statement in C?
   - What is the syntax for a "switch" statement in C?
   - What is a pointer?
   - What is type casting? (e.g. cast a pointer)
   - What is the "scope of a variable"?
   - What does the "static" keyword do? Consider local variables, global variables, and functions.
   - What does the "volatile" keyword do?
   - What does the "extern" keyword do?
3) Read about the make utility (GNU make) and examine the makefile examples available on the course web site. GNU make is recommended.

   Note: You can obtain the free GNU make utility from the course web site.

   Note: A make tutorial is available at: http://web.eng.hawaii.edu/Tutor/Make/

   Note: If you want a UNIX-like command line development environment and tool set for Windows, check out the Cygwin web site, http://www.cygwin.com
4) Read about an IDE for C code development (e.g. Code::Blocks, Eclipse or another of your choice).

   Code::Blocks can be found at http://www.codeblocks.org/.

(continued)

5) **Read the "Notes on SDCC" document, available from the course web site**. This document should save you significant effort in this course.
6) Explore the SDCC resources available on the web, as described in the "Notes on SDCC" document.
7) Read the web site regarding the SDCC memory map.

   If not using the computers in the ECEE 1B28 lab: Download SDCC from the sdcc.sourceforge.net web site and install it on your system. Review the SDCC manual that is included with the download. Use SDCC 2.6 or the latest 3.x revision. You can use the make utility or an IDE like Code::Blocks. SDCC version 2.6 and others can be found at: http://sourceforge.net/projects/sdcc/ http://sourceforge.net/projects/sdcc/files/sdcc-win32/2.6.0/
8) How do you configure the system so that the compiler knows where ROM and RAM are located in the memory map, and how much space is available?
9) Determine how you will write the `putchar()` and `getchar()` functions for SDCC. These functions must be obtained in order for you to use serial input and output. These are very simple functions that are responsible for moving a character into or out of the UART on the 8051. For best performance, you can implement an interrupt-based serial port driver. For lower performance but easier implementation, you can implement a polled serial port driver, similar to the routines below.

```
void putchar(char c)
{
      while(!TI);       // wait for transmitter to be ready
      TI = 0;           // reset transmit flag
      SBUF = c;         // write character to transmit buffer
}

char getchar(void)
{
      while(!RI);       // wait for character to be received
      RI = 0;           // reset receive flag
      return SBUF;      // retrieve character and return
}
```

   You might want to consider disabling interrupts around the two statements that clear the flag and access the SBUF register.

   Note that when outputting a newline character from your C routines, you may find that you need to output both a newline \n and a carriage return \r.
10. Be able to explain the use and effect of the 'register' modifier when defining variables.
11. In which order are parameters to a function stored on the stack and why?
12. Suppose you've downloaded the machine code for a function via the serial port and you've stored this function in RAM starting at address 0xD100. Give at least two ways that you can call this function from within another C module.
13. What does the 'static' modifier do when applied to global variables? Local variables?
14. Understand how SDCC handles structures and in-line assembly code.
15. Write a program in C which toggles pin P1.0 of Port 1 continuously.
16. Is getchar() a blocking or non-blocking function call? What happens if you call getchar() and no character has yet been received by the serial port?

Note: there is nothing to hand in for this assignment. In this homework assignment, you will explore:

- Serial EEPROMs
- I^2C

The required reading for this assignment is available on the course web site.

1. Read pages 1-8, 12, 13 of Microchip Application Note AN536 "Basic Serial EEPROM Operation" and skim the rest.

2. Read Microchip Application Note AN551 "Serial EEPROM Solutions vs. Parallel Solutions".

3. Read the first 4 pages of Fairchild Application Note AN-794 "Using an EEPROM - I^2C Interface".

4. Read the data sheet for the NM24C16 Serial EEPROM.
   (a) Why is a dummy write required prior to a random read?
   (b) Why is acknowledge polling (ACK polling, or busy polling) used with serial EEPROMs?
   (c) During a page write operation, what happens if more than 16 bytes of data are sent to the EEPROM from the master?
   (d) During a page write operation, what happens if 16 bytes of data are sent to the EEPROM from the master, but the transfer starts at an address in the middle of a page?
   (e) During a sequential read from an NM24C16, what happens if 2050 bytes are read starting at address 0?

5. Read Microchip Application Note AN709 "System Level Design Considerations When Using I^2C Serial EEPROM Devices".
   (a) How can an internal EEPROM reset be forced by using software?
   (b) Why would this be important?

6. Determine what I^2C software you will use for your development environment. If you are using SDCC, you can use I^2C code from the web - **refer to the Notes on SDCC document for more information on code libraries previously used in this course**.

7. [Optional] Skim pages 1-20, 39 of the I^2C Specification, available on the course web site.
   (a) What data rates are supported by I^2C?
   (b) If two devices start communicating simultaneously, is the transmission lost due to bus contention? Briefly describe how bus arbitration works.
   (c) What determines the minimum and maximum values of the pull-up resistors used on SCL and SDA?

8. [Optional] Review the Maxim application note "Comparing the I2C Bus to the SMBus".
   - What are the major differences between I^2C and SMBus?

9. Review the data sheets and application notes related to the PCF8574 I^2C I/O Expander.

Note: there is nothing to hand in for this assignment. In this homework assignment, you will explore:

- EEPROMs
- LCDs

The reading for this assignment is available on the course web site.

1. [Optional] Review the technology links available at: **http://ecee.colorado.edu/~mcclurel/misc.html**
2. Review the new links and documents available on the course web site. The data sheets for the HD44780 and SED1278F LCD controllers are available there, along with the data sheet for the LCD module (e.g., the Optrex DMC20434/DMC16433). A few LCD handouts will be distributed in class.
3. Read the document "Adding an NM24C04 (or NM24C16) EEPROM to your board".
4. Wire up your EEPROM interface.
5. Read the document "Adding an LCD (with an HD44780 LCD controller) to your board".
6. Wire up your LCD interface and connect your LCD to your board. The holes in the four corners of the LCD module are the right size for #2 machine screws. **However, it is easiest just to use 30 AWG wire wrap wire or some stripped 22 AWG wire to connect the four corners of the LCD module to your board instead of drilling holes. Drilling holes may add risk for short circuits in the PCB.**

Note: A pin strip header/socket combination makes a nice way to interface your LCD to your board. If your LCD module has a female socket already attached, you can wire wrap the pin strip header into your board and then plug in your LCD. Your LCD then can be removed from your board easily. If your LCD module has no female socket and only has holes in the PCB, you can do one of the following:

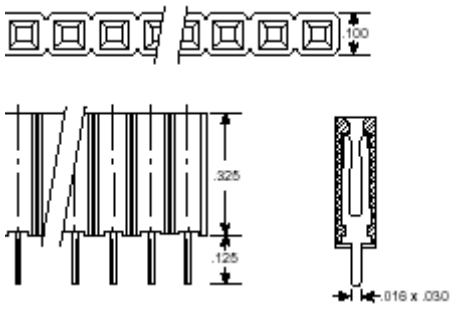
1. solder a female pin strip socket to the LCD module, and then wire wrap the male pin strip header to your board. This method allows you to remove the LCD module from your board.
2. solder in the male pin strip header to the LCD module, and then wire wrap the modified LCD module to the board. With this method, you cannot easily remove your LCD module from your board.

One good type of male header is 3M part number 929834-03-36 (mating length 0.235", tail length 0.410"), available from vendors such as DigiKey and Mouser Electronics. It is similar to the picture below to the left (36 pin header trimmed down to 14 pins). Mill-Max also makes a similar type of header.

One good type of female socket is 3M part number 929974-01-14 (Mouser number 517-974-01-14). It is similar to the drawing below to the right. AMP also makes a similar type of socket (AMP 1-535541-2).

Male pin strip header

Female pin strip socket



Note: there is nothing to hand in for this assignment. In this homework assignment, you will explore:

- Interrupts in C

---

1. Examine the use of interrupts in SDCC. Review the **Notes on SDCC** document and the SDCC manual.

2. [SDCC] Learn what the __critical and __naked key words do relative to interrupts.

3. [SDCC] Learn what the "volatile", "__reentrant", and "#pragma nooverlay" key words do relative to interrupts.

4. Why do you need to be careful about using global variables in an ISR?

5. Look at the interrupt re-vectoring code in PAULMON21.ASM.

   a) What will happen at runtime if you do the following?
   - Set the value of the variable 'vector' to 0x8000 in PAULMON21.ASM
   - Link the code with SDCC linker option "--code-loc 0x8000".

   b) What will happen at runtime if you do the following?
   - Set the value of the variable 'vector' to 0x2000 in PAULMON21.ASM
   - Link the code with SDCC linker option "--code-loc 0x8000".

   c) What will happen at runtime if you do the following?
   - Set the value of the variable 'vector' to 0x8000 in PAULMON21.ASM
   - Link the code with SDCC linker option "--code-loc 0x8100".

6. Analyze the code in the **interrupt exercise posted on the course web site** and handed out in class. Come to class with your analysis of whether there are any potential problems in the code. Assume there are no syntax errors in the code.