

EE345M Laboratory Manual Spring 2008



Jonathan W. Valvano
Electrical and Computer Engineering
University of Texas at Austin

email: valvano@mail.utexas.edu
web: <http://users.ece.utexas.edu/~valvano>
how to program: <http://users.ece.utexas.edu/~valvano/embed/toc1.htm>

This laboratory manual accompanies the book, Embedded Microcomputer Systems: Real Time Interfacing, Second edition, by Jonathan W. Valvano, published by Thomson, copyright © 2006.

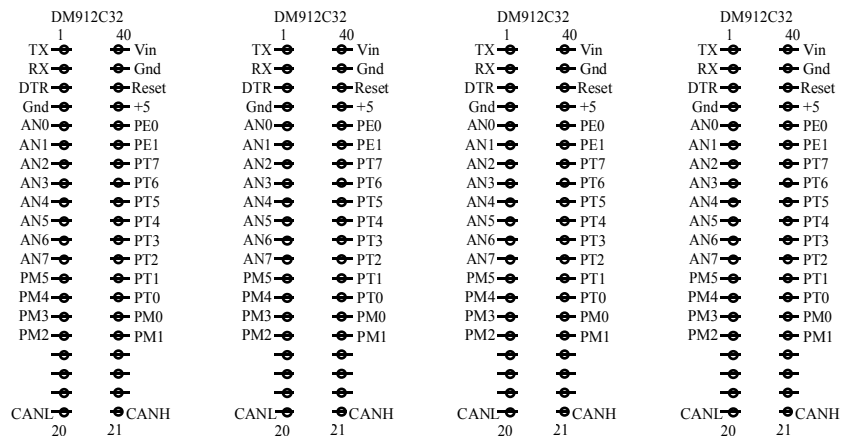
Table of Contents

- Lab introduction (rules and procedures) (Labintro.pdf)
- How to Program the Tech Arts 9S12C32 board using Metrowerks (Howtobuild_S12C32.pdf)
- Logic analyzer instructions (logic.pdf)
- M68DKIT912C32DMSCH data sheet (M68DKIT912C32DM.pdf)
- 25b. Solid state disk, SPI, address translation, layered software, file system
- 22b Audio recording and playback
- 10e. Distributed data acquisition using a controller area network (CAN)
- 16. Digital scope and spectrum analyzer using a USB interface
- 18. Real-time operating system
- Competitive Soccer Robot (teams of 3 or 4)
 - 30A. Motor interface and PID control
 - 30B. Steering, and sensor design and interfacing
 - 30C. Control algorithm, system performance analysis

Parts list

- 1- 25LC640 8k by 8-bit EEPROM
- 1- TLC2274 quad rail-to-rail op amp
- 1- 2.5V Electret microphone
- 1- MC34119 audio amp
- 1- 32 ohm speaker
- 1- Microchip MCP2551
- 2- L293 or L293D stepper motor driver
- 2- IRF522 or IRF540 MOSFETs
- 8- 1N914 snubber diodes
- 1- Red LED
- 1- Yellow LED
- 1- Green LED
- 1- 7406 or 7405
- 1- LCD display (long term checkout)
- 1 – UM245R USB-fifo module with USB AB cable

Cut out a template, and place it between your 9S12C32 board and the protoboard.



I. Introduction to EE345M Laboratory

I.1. Grading Policies

Groups will consist of exactly two students. Lab partners have separate checkout grades⁺, but share the preparation^{*}, report^{*}, software quality^{*}, and late penalty grades^{*}. For each lab assignment, there are a number of preparation tasks that must be completed before the laboratory period. The following activities occur in this order at the beginning of lab, you turn in your preparation, the TA will give a short lab lecture, the TA will check your preparation, and if possible the TA will return the preparation to you.

<u>Responsibility</u>	<u>grade</u>
Preparation	20 [*]
Preparation includes a printout of your software, which is due at the start of lab. The software should be typed into the computer and compiled with no syntax errors. Preparation does not usually involve running and debugging, but it should not be handwritten. Preparation also includes hardware circuit diagrams, which are also due at the start of lab. Handwritten diagrams are acceptable for the prep, but diagrams in the report must be generated using a CAD package. Hardware diagrams include chip numbers, pin numbers, resistor/capacitor types and tolerances, connections to computer. You are responsible for the procurement of all necessary parts before lab starts. As part of the preparation, hardware circuits but not necessarily built or debugged.	
Software Quality (see the section on software style guidelines later on in the lab manual)	10 [*]
Documentation, comments, choice of good variable and function names	
Proper style, organization, modular structure, ease of understanding	
Report (10) Final hardware circuit diagrams (must be generated using a CAD program)	20 [*]
(10) Results printout, performance/data graphs (handwritten drawings are OK)	
Checkout, demonstration to TA	
Performance, correctness of the program function	25 [*]
Interface to the human operator, menus, error messages	
Oral understanding of engineering tradeoffs	25 ⁺
Penalty 85% multiplier if checkout on Friday by appointment with TA,	
60% multiplier if checkout during first lab period the following week	
Late Report	<u>-5/day[*]</u>
Total	100 ⁺

If a TA other than yours checks out your lab, please email your TA specifying the time, date, lab number and the other TA's name. Also, please follow up with both TAs to make sure you got credit for the lab. Please include the following information at the beginning of each of your software files:

- 1) **Students' names**
- 2) **TA name**
- 3) **Date of last change**
- 4) **Lab assignment number**
- 5) **Purpose of the software module**
- 6) **Hardware configuration**

I.2. SAFETY REGULATIONS:

IN CASE OF EMERGENCY DIAL 911 or 9-911

Since there will be times when students will work other than the regularly scheduled lab sections, it is necessary that certain regulations be observed for the convenience and safety of all. Since the possibility of lethal shock exists in those circuits utilizing low potentials, the following should always be observed:

1. Working alone in a lab room is not permitted.
2. Working after regular hours without written permission is not permitted.
3. Work benches must be clear of all coats, knapsacks and extraneous materials. Coat racks are desired for those desiring this convenience. Otherwise all materials must be stored under the work area or out of the way.
4. Shoes must be worn in the lab at all times. Shoes represent a significant protection against electrical shock.
5. Smoking, food and beverages (e.g., coffee) are not permitted anywhere in the lab area.

IN CASE OF INJURY OR SHOCK:

Turn off power, do not move the injured. Start artificial respiration if breathing has stopped. Have someone else call 911 or 9-911 if CPR is needed.

IN CASE OF FIRE:

Turn off the power, call 911 or 9-911, fight fire with available extinguisher, have someone clear the building.

I.3. LAB PROCEDURES AND POLICIES

WEEK OF January 22-24: Go to the regularly scheduled lab during the first full week of classes. During this time, you will be introduced to the lab equipment. You will also be instructed on lab procedures and grading policy. If you missed your regularly scheduled lab, attend one of the other lab periods. Note that attending a lab session for which you are not registered is not permitted except during the first week of classes.

LAB PARTNERS: Every student is required to have a lab partner. You will perform all labs with a partner. Students choose their own lab partners during the first week.

LAB EQUIPMENT USAGE: Lab hours are posted in the laboratory. There are no sign-up sheets, but cooperation is expected. If you start debugging on a station, you may stay as long as you like, with three exceptions:

- You must leave when the second floor labs are closed for the day;
- You must leave during the first half-hour of the other regularly scheduled lab periods;
- You may not leave the station unattended for more than 15 minutes.

If you would like to use a station that has been left unattended for more than 15 minutes:

- 1) Carefully disconnect the hardware and eject any floppy disks;
- 2) Do not save any software files;
- 3) Return all materials (hardware, disks, paper) to the front desk;
- 4) Leave a note on the station with your name and time;
- 5) Write a note to the TA describing exact times listing what you turned in.

LAB LECTURE: The purpose of the lab lecture is to provide necessary information to complete the lab. The scope of the lectures will be material relevant to the lab. The lecture will be conducted during the first 15 minutes of each lab session.

LAB PREPARATION: Lab preparation must be performed prior to the regularly scheduled lab period. All software must be written, edited and designed before coming to the lab. Hardware must be designed down to the pin numbers. Label all resistance and capacitance values and types. For example, 1k Ω 5% carbon, or 0.01 μ F 5% ceramic. In this way, the lab period may be spent in debugging your system with the TA's help. The preparation is due at the start of the lab. Preparation includes gathering all the physical components required to perform the lab.

LAB REPORT: Each lab report is bound separately. There is a **Deliverables** section that details the specific components required for that lab report. Lab report typically include the following items:

- A) Objectives (1/2 page maximum)
- B) Hardware Design. Detailed hardware designs with pin numbers.
 - Generated using a CAD program like **ExpressSCH**
 - Include all external devices used (chips, R's C's values and types)
 - Show connections to the 6812 board.
- C) Software Design (no software printout in the report)
 - Draw figures illustrating the major data structures used,
 - A call-graph illustrating the modularity of the software components
 - Draw data-flow graph showing how data is processed
- D) Measurement Data
 - Whenever appropriate, use the printer to get a hardcopy listing of your results. Include graphs and figures as specified in the assignment.
- E) Analysis and Discussion (1 page maximum)

CHECKOUT: A rough draft of your hardware diagrams and hard copy printout of your source code listings must be given to the TA before you demonstrate. If your experiment works, you will be assigned a good score on the performance part. The TA will ask the partners oral questions that test your “understanding” of the computer engineering concepts of the lab. The partners will answer separate questions and receive separate “understanding” grades. You must get your rough draft software listings signed and dated by a TA to prove that the lab was completed in a satisfactory manner. Late checkouts will result in lost points. **Your software files will be copied onto the TA’s web space during checkout.**

I.4. Variables In C

Consider the following simple C program. Assume this code is located in `file.c`.

```
short A1; int A2;
short B1; short static B2;
short C1; short volatile C2;
short D1=5; short const D2=5;
short F1(short n){ return n+1;}
short static f2(short n){ return n+1;}
short FreezingPoint1=32; short FreezingPoint2=0x20; // degrees F
void program(void){
    short e1; short static e2;
}
short G1(short m1){ return m1+1;}
short G2(const short m2){ return m2+1;}
```

a) *What is the difference between **A1** and **A2**?* **short** is always 16 bits, while **int** is compiler-dependent. On Metrowerks CodeWarrior, they are both 16 bits.

b) *What is the difference between **B1** and **B2**?* **B1** is public, and can be accessed from anywhere in the software system. In particular, any file can define **B1** as `extern short B1;` and access this variable. The linker will resolve the address uncertainty. **B2** is private, and can be only be accessed from software in this particular file.

c) *What is the difference between **C1** and **C2**?* **C2** will not be optimized by the compiler (**C1** can be optimized). In particular, it will not keep a copy of **C2** in a register, rather it will reload a new value each time it is needed. It assumes **C2** can be changed by operations other than direct software action. Two good applications of **volatile** are I/O ports and global variables shared by two or more threads. Assume **C1** is incremented by a background interrupt and `wait` is called from the foreground. Because **C1** is not **volatile** then the compiler could take this

```
void wait(void){
    while (C1<100){};
}
```

and create the following "optimized" assembly (Metrowerks does not optimize this way, but some compilers might)

```
wait:: ldd C1 ; get a copy of C1
loop: cpd #100 ; assume RegD has count, check for greater than 100
      blo loop
      rts
```

d) *What is the difference between **D1** and **D2**?* Formally, **const** means can't be changed. On an embedded system, it means **D2** is stored in ROM, and **D1** is stored in RAM. There will be two copies of **D1**.

e) *What is the difference between **e1** and **e2**?* When used inside a function **static** has a different meaning than when used outside a function. In this situation, it means **e2** is statically allocated in permanent RAM and the values persist from one function call to the next. **e1** is dynamically allocated on the stack, used inside the function, and deallocated at the end of the function. The values of **e1** do not persist from one function call to the next.

f) *What is the difference between **F1** and **f2**?* In this context, **static** is used in a similar manner to the way static is used for a variable outside a function. **F1** is public, and can be called from anywhere in the software system. In particular, any file can define **F1** as

```
extern short F1(short n);
```

and call this function. The linker will resolve the address uncertainty. **f2** is private, and can be only be called from software in this particular file.

g) *What is the difference between **FreezingPoint1** and **FreezingPoint2**?* The only difference is style. 32 is much better style (easy to read) than 0x20 in this context.

h) *What is the difference between **m1** and **m2**?* In this context, **const** does not modify where the parameter is allocated. Rather, **const** prevents the programmer from modifying the parameter within the body of the function.

I.5. Web sites

see 6812 products	http://www.technologicalarts.com
electronic parts	http://www.bgmicro.com
electronic parts	http://www.jameco.com/
electronic parts	http://www.digikey.com/
electronic parts	http://www.mouser.com/
mechanical parts	http://www.allelectronics.com
robot parts	http://www.robotstore.com/
robot sensors	http://www.parallax.com/
robot parts	http://www.towerhobbies.com/
enclosures	http://www.pactecenclosures.com/
Writing in C for 6812	http://users.ece.utexas.edu/~valvano/embed/toc1.htm
Freescale	http://www.freescale.com/
Other links	http://users.ece.utexas.edu/~valvano/book.htm

I.6. Legal Stuff

The opinions expressed in these notes do not necessarily reflect the opinions of the University, its management or its big time financial donors. Also, there shall be no bologna, Bevis, mustard, chewing the cables, free lunch, sob stories, running & screaming, whining, hitting, spitting, kicking, biting, or tag backs. Quit it or we're telling. (Enjoy the course.)