# Mechatronics I Laboratory Exercise: Introduction of Microcontrollers

The purpose of this lab is to give you experience with microcontrollers. Their small size, cheap price, and ever increasing speed makes them perfect for control applications where a large computer isn't necessary or the space isn't available. They are now used in such areas as control of automobile engines, microwave ovens, robotics, and many other applications. The microcontroller we will use is the Handy Board. The information presented in this lab only introduces the Handy Board. More detailed information can be found at http://-el.www.media.mit.edu/groups/el/Projects/hb/index.html or in "The Mechatronics Robot Handbook", by Kam Leang.

The Handy Board is a small experimental board developed by Fred Martin of the MIT Media Laboratories. The main features of the Handy Board are a Motorola 68HC11 microprocessor chip, four DC motor outputs, seven 8-bit analog inputs, nine digital inputs, and a 16x2 LCD screen, as well as other features. It is easily programmable using Interactive C (a C programming language) developed by Randy Sargent of MIT. A diagram of the Handy Board with labeled features is shown below in Figure 1.

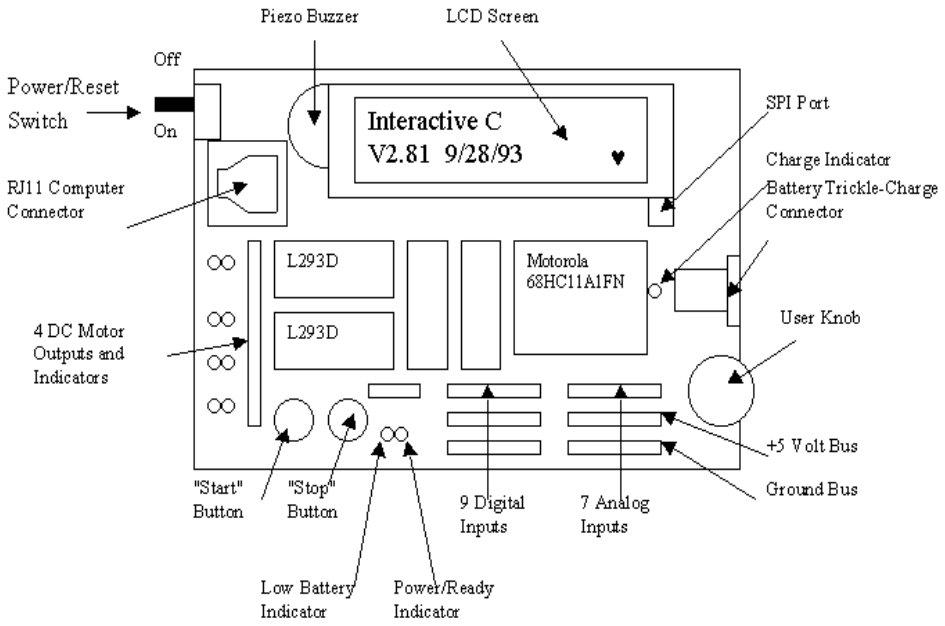


Figure 1: Handy Board Labeled Features

**Step 1: Starting the Handy Board**

Before starting the Handy Board, make sure the following are done:

1. The Handy Board is connected to the computer.
2. One end of the RJ11 modular cable (looks like a telephone cable) should be plugged into the computer connector port on the Handy Board (see Figure 1).
   (a) The other end of the RJ11 modular cable should be plugged into the Serial Interface & Battery Charger Board (A separate small interface board).
   (b) From the Serial Interface Board, a serial cable should be connected to the serial port of the computer.
   (c) Make sure the computer is in MS DOS mode. To put the computer into DOS mode, activate the "start" button and select shutdown/ Restart in MSDOS mode.

(d) Make sure that your working directory is C:\handy. Type "cd c:\handy" to change to the "handy" directory. All future work should be performed in this directory.

3. Insert the plug-in transformer into a wall outlet. Insert the other end into the receptacle on the Handy Board battery trickle-charge connector (see Figure 1). This step will keep the batteries on the Handy Board charged while you are working on it, but is not necessary if the batteries are fully charged.

The Handy Board is now ready to be turned on. Turn the power/reset switch on the Handy Board on (see Figure 1). The Handy Board screen should read:

Interactive C
V 2.81 9/28/93

The heart should be beating. If both do not occur, then the memory has been corrupted and the operating system code needs to be reloaded. To reload the operating system code (p_code), do the following:

1. Type "dl pcode_hb.s19" at the C:\HANDY prompt and press <enter>.
2. Turn the Handy Board off and turn it back on while you hold down the "stop" button. This procedure places the Handy Board in "download mode". It is critical that the Handy Board be in this mode for operating system code reloading.
3. Follow the on screen instructions
4. Once the download process is complete, restart the Handy Board and you should see:

Interactive C
V 2.81 9/28/93

If the download process is unsuccessful, then ask your TA for help.

**Step 2: Running the Handy Board directly from the PC (Interactive Mode)**

Now turn on the computer (if not already on) and go the "C:\handy" directory ( in DOS mode). Once in the handy directory, type "ic<enter>" at the Dos command prompt. You should see a "C>" prompt appear on the computer screen. You are now in the Interactive Mode for the Handy Board. The Handy Board is now ready to use.

Commands to the Handy Board can be sent directly to it from the host computer in Interactive Mode. For example, at the "C>" prompt, try typing: `printf("Hello World!\n");` <enter> The screen on the Handy Board should read, Hello World! Yippy!!! Pretty cool huh???

Now type beep();<enter>. You should hear a short tone coming from the piezo buzzer (see Figure 1). Try typing tone(800.,5.);<enter>. You should hear a tone at 800 Hz for 5 seconds. Other commands are included at the end of this lab handout. Explore them later.

To quit the interactive mode, type one of the following: quit, bye, or exit <enter>.

**Step 3: Downloading Programs to the Handy Board**

We are now ready to write a program to download to the Handy Board. From any text editor (such as DOS edit or Windows Notepad), type in the following program and name it sample.c. Place the file in "C:\handy" directory. Type the following in your program:

```
void main(){
    printf("You now know how to download\n");
    }
```

Save your file, exit the editor, and start up interactive mode IC as in step 3, above.

At the "C>" prompt type load sample.c <enter>. If the download was successful, you should not see any error messages. Now to start the program, simply turn the Handy Board off and then back on. The screen should read, You now know how to download. Each time the board is turned off and then back on, the program will be executed.

**Remark:** The program can be named any name allowable by Dos (up to 8 characters with certain characters being restricted) but must have a ".c" extension. Multiple programs can be loaded (up to 32K) onto the Handy Board but only one can have the function main(). It will be this program that is run on start up of the Handy Board. The other programs are callable either from the program containing the function

main(), other functions, or from the Interactive Mode.

**Step 4: Handy Board Functions**

There are many commands that can be used to control the Handy Board. You have seen three of them, printf(), beep(), and tone(). These commands can be given directly in the Interactive Mode (as in step 3) or used as function calls in a C program (as in step 4) and downloaded to the Handy Board. We will be mainly writing programs and downloading them to the Handy Board.

A brief description of the more useful functions and how to use them are given below. For a more complete listing of the functions, see the Interactive C Manual or "The Mechatronics Robot Handbook", by Kam Leang.

**printf(format-string,[arg-1], . . . , [arg-N])**

Prints to the LCD screen. This is best illustrated by some examples. Example 1: Printing a message to the screen. `printf("Hello World\n");` The character "\n" at the end of the string signifies end-of-line. When an end-of-line character is printed, the LCD screen will be cleared when a subsequent character is printed. Example 2: Printing a number. `printf(``Value is %d\n",x);` The special form

**Table I. Printing Formats**

| Format Command | Data Type | Description |
|---|---|---|
| %d | int | decimal number |
| %x | int | hexadecimal number |
| %f | floating | floating point number |
| %s | char | character array (string) |

**motor(int m, int p)**

Turns on motor port m at power level p. There are four motor ports on the Handy Board, labeled Motor-0, Motor-1, Motor-2, Motor-3 (see Figure 1 and 2 for location). The power level ranges from 100 for full on forward to -100 for full on backwards. Although the range of power levels goes from -100 to 100 in increments of 1, there are actually only 15 motor speeds (See Table II below).

**alloff() or ao()**

Turns off all motors.

**off(int m)**

Turns off motor m.

**Table II. Motor Levels**

| Power Level | Forward Motor Speed |
|---|---|
| 0-10 | off |
| 11-24 | 1 |
| 25-38 | 2 |
| 39-52 | 3 |
| 53-66 | 4 |
| 67-80 | 5 |
| 81-94 | 6 |
| 95-100 | 7 |

*A negative power level corresponds to the appropriate Motor Speed, but in reverse.

The motor driver outputs (see Figure 1 for location of motor outputs on the Handy Board) are arranged as shown below in Figure 2:

**Remark:** The middle slot for each motor is used just for spacing and doesn't carry any signal. The motor drivers chips (L293D on the Handy Boards) are capable of putting out up to 9.6 volts and 600 mAmps. If higher current demand is required for DC motors, then an additional motor driver circuit must be used. Refer to "The Mechatronics Robot Project Guide" for additional information on external motor driver circuits.

The motor drivers use pulse width modulation (PWM) to control the voltage output. This means that full voltage is switched on and off at high frequency for varying lengths of time. The effective voltage seen by a motor (which has a slow response time compared to the frequency of the PWM) is proportional to the average time spent in the high state (full voltage). Note that if you measure the output with a oscilloscope,

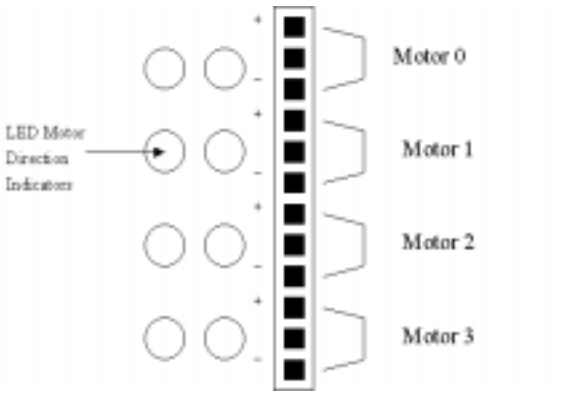


Figure 2: Arrangements of Motor Output Ports

you will see a high frequency signal.

**digital(int p)**
Returns the value of the digital sensor in digital sensor port (7-15). If zero volts are applied to the port, it returns ture (1). If over 2.5 volts are applied, the value returned is false (0).

**analog(int p)**
Returns the value of the sensor in analog sensor port (0-6). The value returned is between 0 and 255. Zero applied voltage returns a value of 0, while 5 volts returns a value 255. A diagram of the analog and digital sensor port is given below in Figure 3 (see Figure 3 for the location on the Handy Board). The

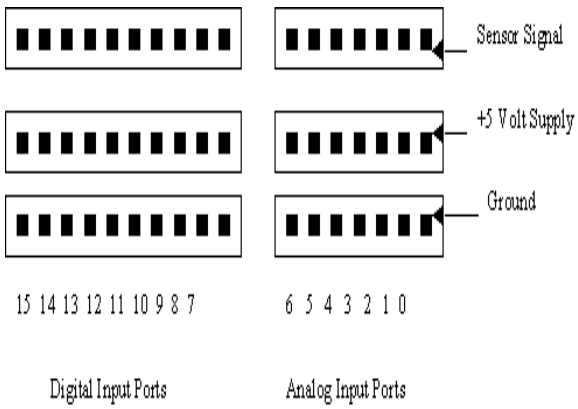


Figure 3: Analog and Digital Input Ports

bottom two strips are ground and +5 supply voltage that can be used to power sensors. The top strip is for the sensor signal (this is the value that will be read by the analog and digital functions).
**knob()**
Returns the position of the user know (see Figure 1) as a value between 0 and 255.

**start_button()**
Returns the value of the button labeled "start" (see Figure 1). Returns 1 if pressed and 0 if released.

**stop_button()**
Returns the value of the button labeled "stop" (see Figure 1). Returns 1 if pressed and 0 if released. Example: `/*Wait until start button is pressed*/ while(!start_button()){}`

**Other Functions and Additional Information**

There are many other functions that might be useful in programming the Handy Board. A complete listing is given on the following page. For a complete description of these functions, see The Interactive C Manual for the Handy Board or The Mechatronics Robot Project Guide, by Kam Leang. These references also contain a basic C programming tutorial for those who aren't familiar with C programming.

Further information about The Handy Board, ranging from schematics, modifications, new functions, and much more is available on the World Wide Web at http://el.www.media.mit.edu/groups/el/Projects/-hb/index.html

**List of Handy Board Functions**

**LCD Screen Printing**

printf(format-string, [arg-1], . . . , [arg-N])

**DC Motors**

void fd(int m)
void bk(int m)
void off(int m)
void alloff()
void ao()
void motor(int m, int p)

**Servo Motors**

void servo_a7_init(n)
int servo_a7_pulse
void servo_a5_inti(n)
int servo_a5_pulse

**Sensor Input**

int digital(int p)
int analog(int p)

**User Buttons and Knob**

int stop_button()
int start_button()
void stop_press()
void start_press()
int knob()

**Infrared Subsystem**

int sony_init(1)
int sony_init(0)
int ir_data(int dummy)

**Time Commands**

void_reset_system_time()
long mseconds()
float seconds()
float sleep(float sec)
void msleep()

**Tone Functions**

void beep()
void tone(float frequency, float length)
void set_beeper_pitch(float frequency)
void beeper_on()

void beeper_off()

**Multi-Tasking**

start_process(function-call(. . .), [TICKS],[STACK-SIZE])
int kill_process(int pid)

**Process Management**

void hog_processor()
void defer()

**Arithmetic**

float sin(float angle)
float cos(float angle)
float tan(float angle)
float atan(float angle)
float sqrt(float num)
float log10(float num)
float log(float num)
float exp10(float num)
float exp(float num)
(float) a^(float) b

**Memory Access**

int peek(int loc)
int peekword(int loc)
void poke(int loc, int byte)
void pokeword(int loc, int word)
void bit_set(int loc, int mask)
void bit_clear(int loc, int mask)

**Laboratory Exercise**

1. Read through the microcontroller introduction handout in lab and learn how to use the Handy Board.
2. Write a short C program to control a small DC motor through the Handy Boards DC motor output ports using the knob feature on the Handy Board. In other words, the DC motor should rotate counter clockwise when the knob is completely turned counter clockwise and should rotate clockwise when the knob is completely turned clockwise-with a gradual transition for intermediate positions of the knob. Demonstrate to your TA that your code works.

**Post-Lab Report**

Explain your motor program and include a copy of it in your report.