

Flash Innovation 2003
Design contest using the Motorola HC08
Spa Controller
Project F173

Project description

This project uses the Motorola 68HC908QT4 micro controller as the brains for a temperature controller. The temperature controller will replace an analog timer and bimetal temperature sensor. The controller will switch the relays already installed to operate and maintain the spa. This project is exciting because it incorporates several interfaces. We have an analog temperature sensor. There is an operational amplifier to boost the analog temperature signal. An opto-isolator driving a triac AC switch. And an I2C interface to control the I/O. There is a debug/download port available. This is very useful and should be used to its fullest extent. This design has the circuitry for this circuit on a separate board that plugs in via 4-pin connector. This way the CPU can be soldered in place and still be able to flash a new program.

Circuit Description

The temperature sensor is a LM35. It outputs .01 volts per degree centigrade. The HC08 8-bit DAC precision is 0.0196 volts per bit. That is equivalent to 3.6 degree Fahrenheit (F). That's not acceptable for a hot tub. We will add a 10x amplifier to give us .36 degree F per bit accuracy. The LM358 OPAMP is a good choice. It provides amplification up to $VCC - 1.5$. This limits us to a maximum of 3.5 volts. At 104 degrees F the output of the LM35 will be .40 volts. This would be too much for the LM358 to provide for because the maximum is 3.5 volts. So to solve this problem we will add a voltage divider to the input of the LM358.

For I/O control we will use an I2C I/O expander chip PCF8574AN. This I/O port will control the heater, the filter pump and the LCD display. To limit the current on the I2C bus 30k resistors are added in series. During input the internal pull-up resistors are enabled.

The LCD alphanumeric display is a standard 20 character by 2 line display. There are only 6 control lines available. So we will be using the display in 4 bit mode. And it can only be written to. Because of this we will need to have software delays after writing to the LCD port.

The AC control circuit uses an optoisolator to drive a triac. Optoisolators provide an easy way for a microcontroller to interface to a triac. The MOC3062 optoisolator has a zero crossing (ZC) circuit. The ZC circuit limits the gate voltage to 20 volts. This means we can use this circuit for AC voltages 20-240 RMS volts with out having to change the resistor value. The triac - Q401 can drive 1 amp at 400 volts peak.

For downloading and debugging - a separate circuit using a RS232 level translator is plugged into the controller board. This reduces the part count on the main board.

Software Description

Main module

The main module performs the temperature control and menu functions. When the temperature is less than the preset temperature then the heater is enabled.

I2C module

To communication with the 8574 I/O expander chip we will use a bit banded I2C interface.

Global functions:

```
bool i2c_init(void);  
bool i2c_writeb(unsigned char add, unsigned char data);
```

LCD module

The routines in this module communicate with the LCD panel using the I2C interface.

Global functions:

```
void lcd_pstr(char *str);  
void lcd_setcur(char line, char col);  
bool lcd_init(void);  
void lcd_clr(void);
```

Time module

All time functions are in this module.

Global functions:

```
void time_tick(void);  
char * time_gets(char * buf);  
void time_set(void);  
void time_disp_lcd(void);
```

Button module

There are three buttons on the controller. Enter, Up, and Down. This module has the routine for reading the buttons.

Global functions:

```
unsigned char button_wait(void);  
unsigned char button_nowait(void);
```

String module

The string functions in the C compiler use too much code space for our use. So here we will define some specialized string functions suited for our needs.

Global functions:

```
char * ftos(float fl, char * buf);  
char * uctos(unsigned char i, char * buf);
```

Embedded C

Programming microcontrollers is easy when using embedded C. It is important to understand the architecture of the microcontroller. The keywords “static”, “const”, “bool”, and “volatile” are important to understand and use.

Code Warrior IDE and beans

The development kit includes the CodeWarrior integrated development environment (IDE). After installing the IDE, you will need to register the software. Then you will receive e-mail on how to download the license file. Download the file to “C:\Program Files\Metrowerks\CodeWarrior CW08_V2.1\license.dat”.

Note: The CodeWarrior allocates 32 bytes of RAM to ZPAGE memory. I recommend that after your first build you change the settings in the PRM file and allocate all the ZPAGE memory to RAM memory. Otherwise you will run out of stack space. You will also need to modify the CPU bean to not generate a PRM file.

Downloading Code to Spa Controller

Downloading new code to the HC08 is easy using the built in ROM monitor called MON8. Plug the serial cable into the DB9 connector. Then press the button connected to PTA(2) (pin 5) down while powering the circuit on. The processor is now ready to receive the new code.



