Digital Control of a Mobile Air Conditioning System

James Solberg

ME 313 project
Professor N. R. Miller
5/12/98

Abstract

When automotive air conditioners are started up on a hot day, they are known to create whistling and hissing noises. These noises have resulted in customer complaints. It has been shown at the University of Illinois that the acoustic noise results from high speed, single phase vapor refrigerant flowing through the evaporator. It was then proposed to digitally control the A/C system to resolve this and other problems. It has also been proven at the University of Illinois that closing the expansion device can reduce the torque imposed on the compressor at startup. These two remedies along with a PID control algorithm where combined to formulate a control routine for a mobile air conditioning system. The algorithm is implemented by Dan Block's DSP, and controls a mobile A/C system that uses an electric expansion valve (EEV) as its expansion device.

Background

Expansion Valves

A schematic flow diagram showing the basic components of the vapor compression refrigeration system is shown in figure 1. Some typical temperatures for air conditioning applications are indicated. Refrigerant fluid circulates through the piping and equipment in the direction shown.

Process 1 - 2. At point (1), the refrigerant is in the liquid state at a relatively high pressure and high temperature. It flows to (2) through a restriction, called the flow control device or expansion device. The refrigerant loses pressure going though the restriction. The pressure at (2) is so low that a small portion of the refrigerant flashes (vaporizes) into a gas. But in order to vaporize it must gain heat (which it takes from that portion of the refrigerant that did not vaporize), thus cooling the mixture and resulting in a low temperature at (2).

Process 2 - 3. This is when the refrigerant passes through the evaporator. During this time, the refrigerant vaporizes, taking heat from the environment to do so. This is how an air conditioner cools air.

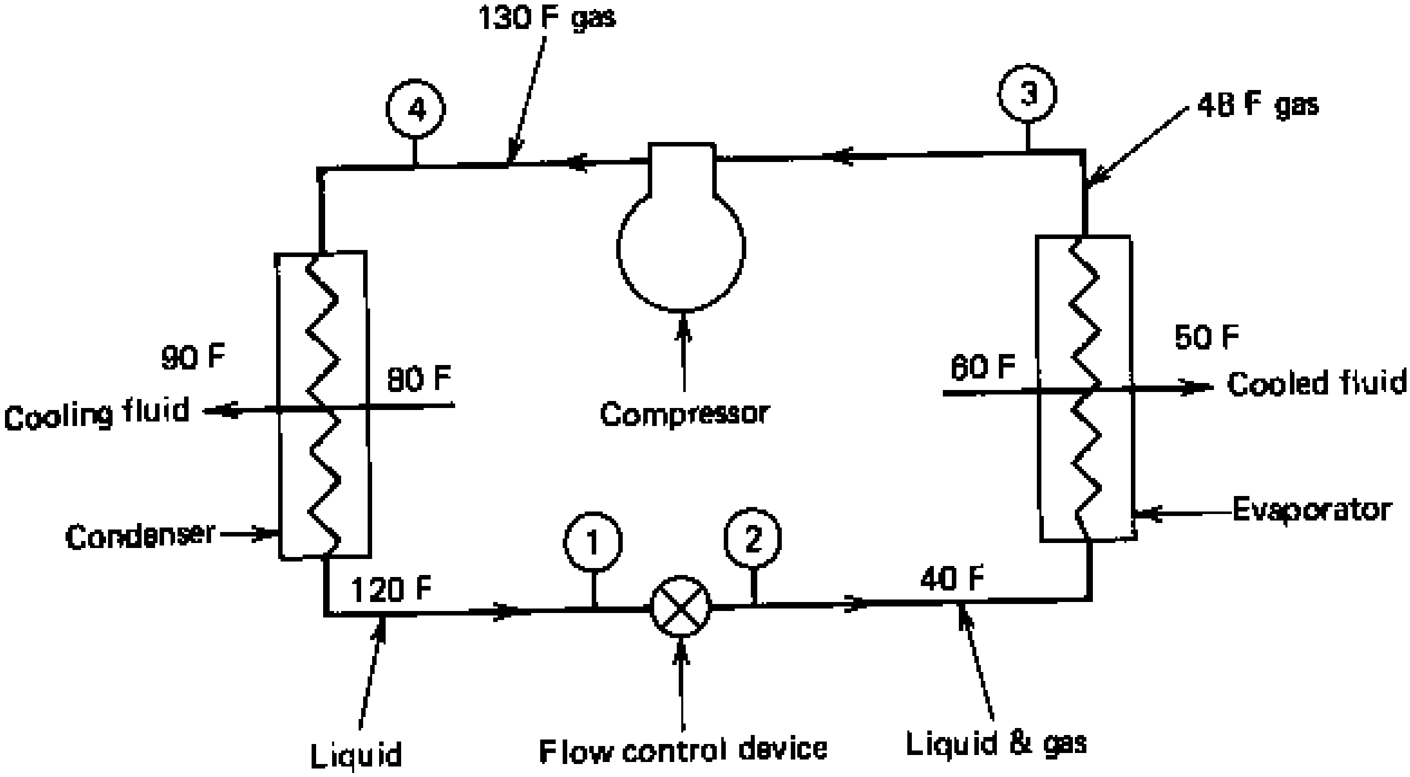


Figure 1. The vapor compression refrigeration system.

The expansion device can play a dual role. The most obvious purpose of an expansion device is to expand some of the liquid into gas by imposing a pressure drop. The expansion device can also be responsible for regulating the refrigerant flow according to the load.

One of the two most common traditional methods of refrigerant control in automobiles is the orifice tube. The orifice tube is only able to exhibit a fixed restriction, and is not able to actively control the flow of refrigerant. The orifice tube has been traditionally used because of its low cost and high reliability due to the fact of no moving parts.

The second of the two most common traditional methods of refrigerant control in automobiles is the thermal expansion valve. The thermal expansion valve is able to actively change the flow of refrigerant as needed by the system. A bulb filled with a fluid is strapped to the outlet of the evaporator, and thus, senses the temperature at this point. This bulb is connected to the valve by a tube in a manner so that the pressure of the fluid in the bulb tends to open the valve more, against a closing spring pressure. If the load in the system increases, then the refrigerant in the evaporator picks up more heat, and the temperature and pressure of the fluid in the bulb increases. This action is able to open the valve more to handle the increased load. Their disadvantage is they are less reliable than orifice tubes due to their moving parts and their relatively fragile capillary tubes.

Electronic expansion valves have not been traditionally used for automotive applications due to the expense on such a system. An electronic expansion valve (EEV) system requires a computer to monitor the temperatures at the inlet and the outlet of the evaporator. From this feedback, the computer actively controls the valve setting. The versatility and efficiency of an EEV make it ideal for an automotive air conditioner, but, the cost is simply too great.

Discussion

DSP / AC Interfacing

Dan Block's DSP unit was chosen to initially control the mobile AC unit. The DSP is equipped with four digital to analog and to digital converters. Each has twelve bits of resolution and operates at a voltage range of -10 volts to +10 volts. There are also 32 digital I/O lines, where 16 are input and 16 are output. These lines operate at +5 volts for logic high, and 0 volts for logic low.

At the AC unit a thermocouple will be placed that the inlet of the evaporator and one will be placed at the outlet of the evaporator. An Omega linearized, isolated thermocouple input will be used to change -100^{o}C to 400^{o}C temperature to a 0 volt to 5 volt signal. But, the greatest range that the thermocouples would experience is from -10^{o}C to 30^{o}C. This corresponds to a 0.9 volt to a 1.3 volt range. That is only 2% of the resolution of the DSP. It would be ideal to have -10°C correspond to -10 volts and 30°C correspond to a +10 volts.

With constraints that -10°C correspond to -10 volts and 30°C correspond to a +10 volts, the relationship between the input of the desired circuit to the output of the desired circuit is found to be:

$V_{out} = 50*V_{in} - 55$

This would correspond to the following circuit

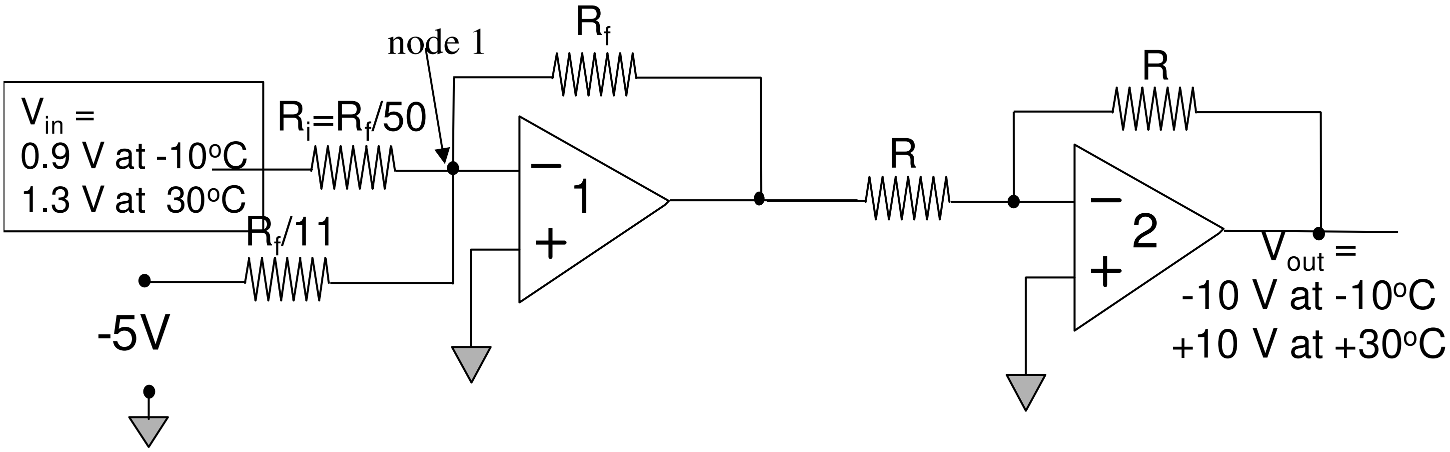


Figure 2. *Op-Amp circuit used to increase the resolution of the thermocouples.*

This circuit uses two inverting op-amps. The first op-amp has a gain of -50. Recall that an inverting op-amp has a gain equal to $-R_f/R_i$. So, R_i is $R_f/50$. But, while 0.9 volts to 1.3 volts is centered around 1.1 volts, -10 volts to +10 volts is centered around 0 volts. So, some sort of off-set must be used to compensate for this. -5 volts is a common supply voltage, so that will be used for this circuit. By summing the currents into node 1 (see figure 2), the following constraint is developed:

$$\frac{50V_{in}}{R_f} - \frac{5}{R_1} = \frac{V_{out}}{R_f}$$

R_1 is the resistor at the -5 volt supply. By applying the design constraints set previously, R_1 is equal to $R_f/11$. So, the complete circuit will look like figure 3.

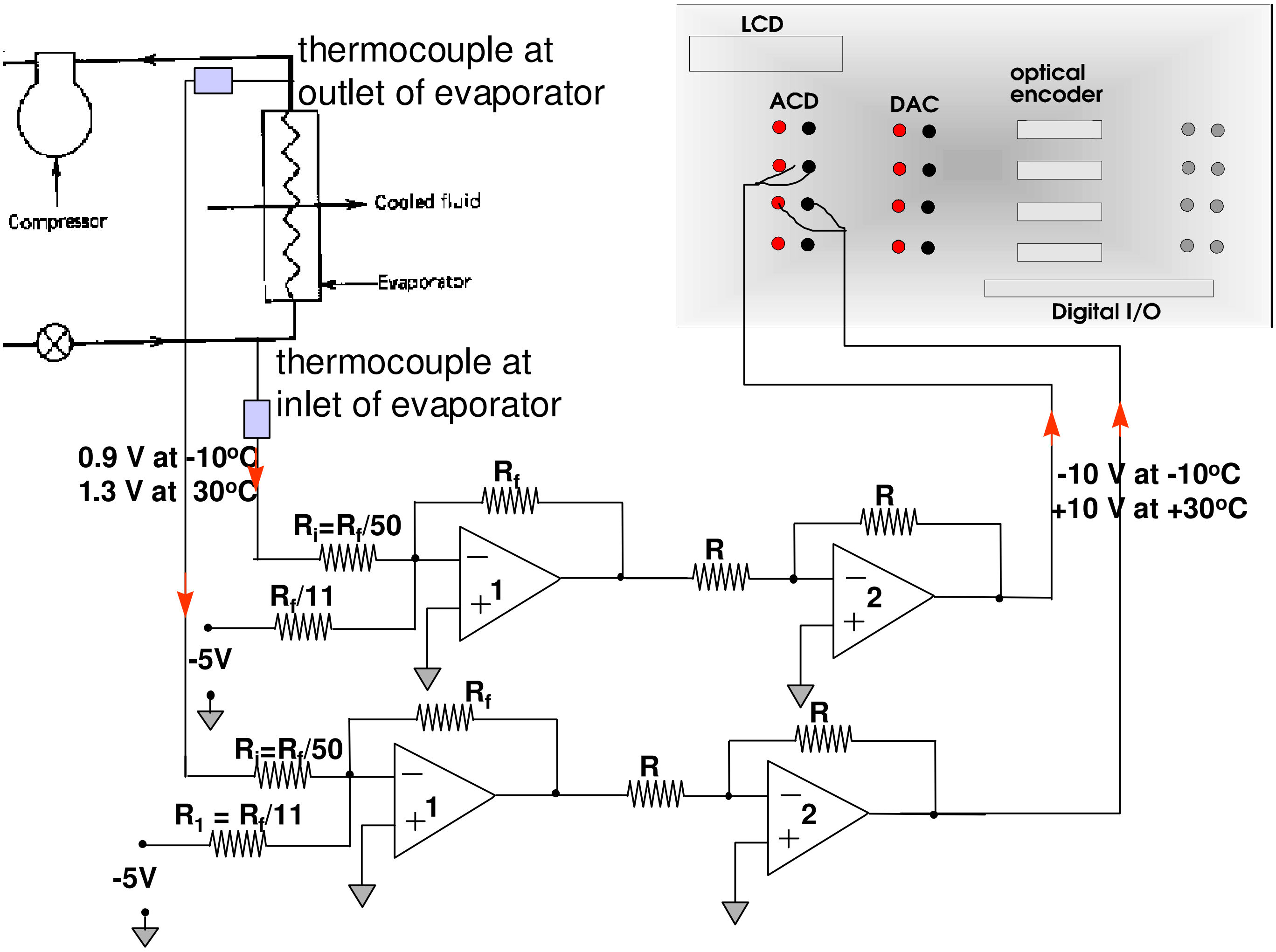


Figure 3. *The circuit used for the DSP to read the feedback from the thermocouples.*

The next problem was to find a way to control the EEV. A twelve volt power supply is needed to control the stepper motor that regulates the opening of the valve. This is satisfied by the Sporlan Universal Electronic Interface Board. This interface board takes in a 4 mA to 20 mA current signal, and outputs the proper valve setting. Signals of 4 mA and 20 mA correspond to valve openings of 0.0% and 100% respectively.

The valve is now controlled by a current source, but the DAC's use -10 volt to +10 volt voltage for control. Omega sells an isolated current output that will convert a -5 volt to +5 volt signal to a 4 mA to 20 mA signal. Although it only uses half the resolution of the DSP, it will have to do. So, at the DAC on the DSP, -5 volts means the valve is fully closed, and +5 volts means the valve is fully open. Figure 4 shows the complete circuit.

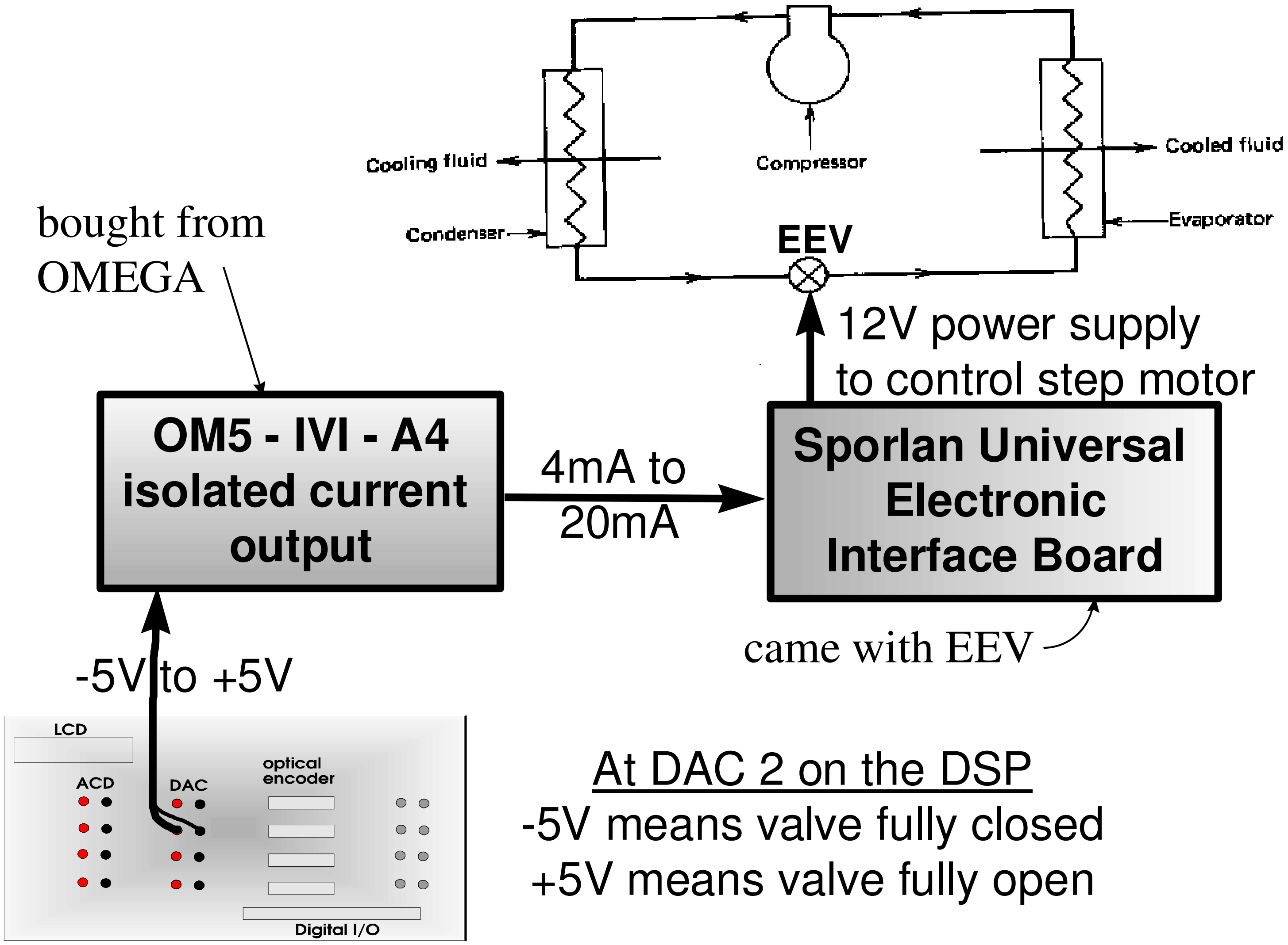


Figure 4. *This circuit used to control the EEV using the DSP.*

The clutch of the compressor would also like to be controlled. The compressor pulley "free wheels" with the drive until the electromagnetic clutch on the compressor is engaged. 12 volts applied to the clutch engages the compressor clutch. The best way to control the clutch is by using one of the digital output lines from the DSP. The digital output would feed into a relay circuit where logic high (+5 volts) would engage the clutch and a low signal (0 volts) would disengage the clutch. The final interfacing between the A/C unit and the DSP is shown in figure 5.

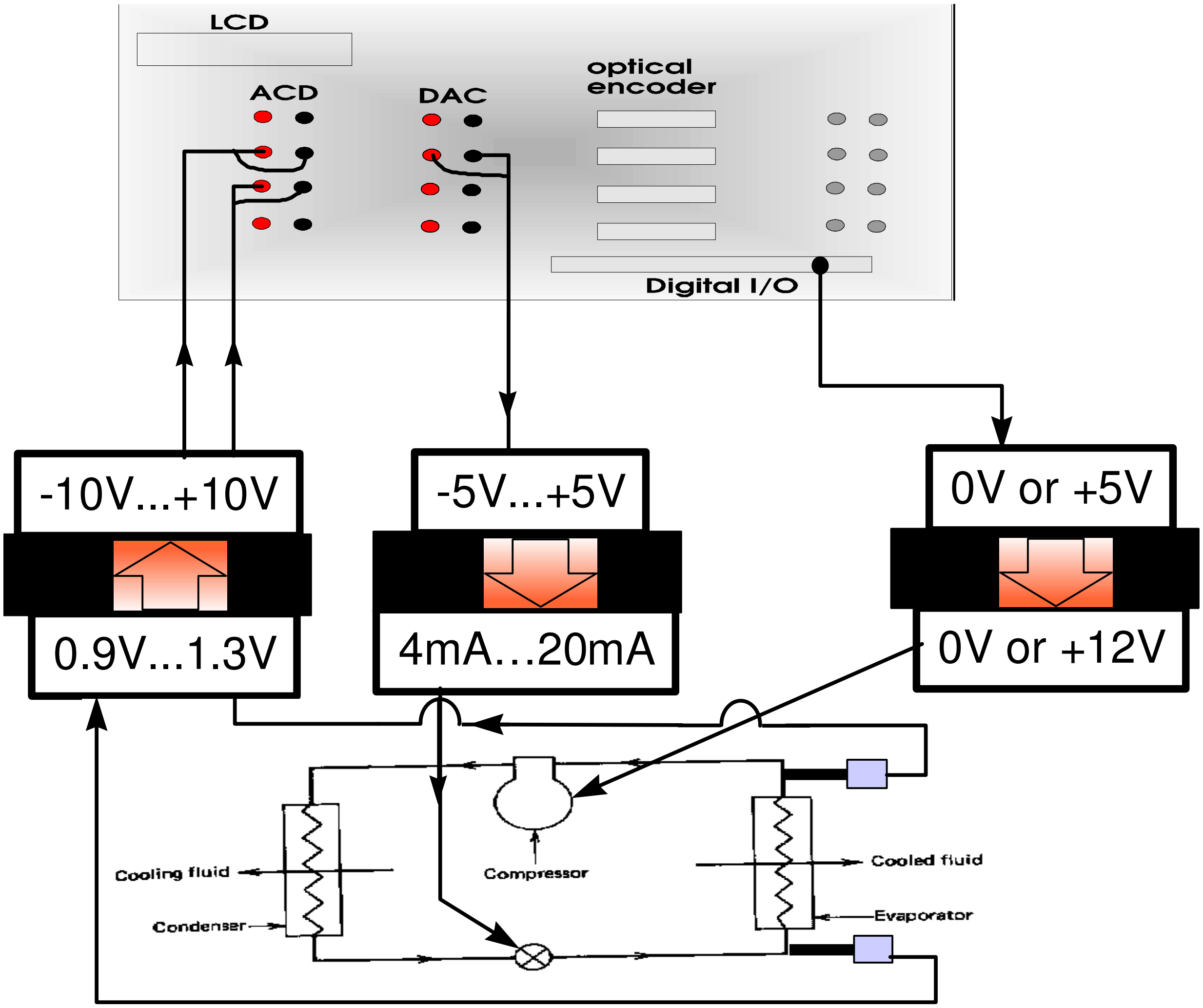


Figure 5. Interfacing diagram between the DSP and A/C unit.

Control Algorithm

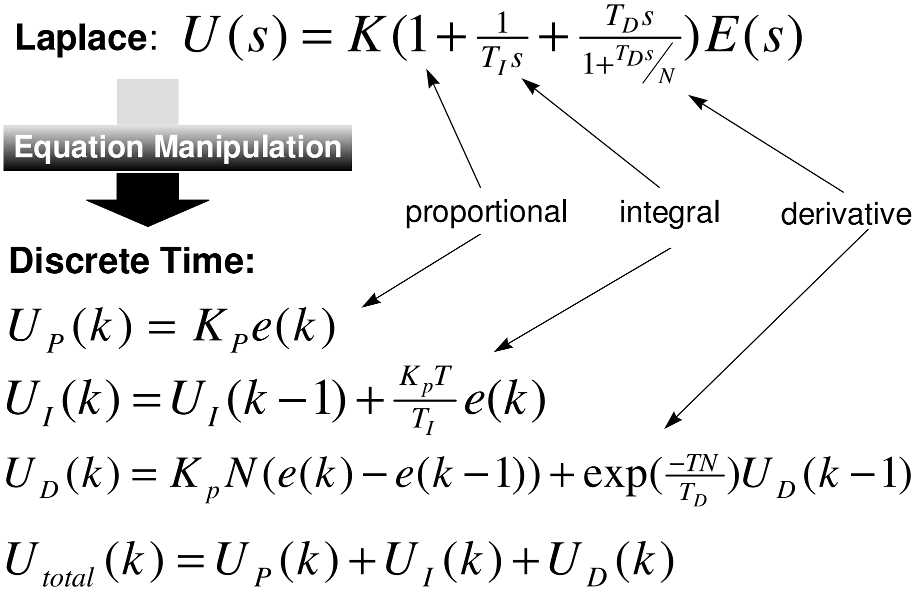
The algorithm to control a mobile A/C system that is presented in this paper will address three issues:

- reduce compressor wear at startup
- reduce noise caused by high speed vapor through the evaporator
- maintain efficiency at steady-state

Eric Wandell [1997] had shown that closing the EEV at startup can reduce compressor torque. So, by having the EEV closed when the clutch is thrown, torque on the compressor can be reduced. If the torque is reduced, then the compressor experiences less wear. Wandell's findings were only relevant for the first 12 seconds after startup.

Once the valve is open, and if the system is hot enough, high speed vapor will come whistling through the evaporator. Simply by constricting the flow, it is believed that less noise will be present in the system. This can be accomplished by only opening the EEV some small amount, initially. Once two-phase refrigerant is detected in the lines, the valve can process to a standard control algorithm. Two-phase refrigerant can be detected when the temperature of the refrigerant drops below some known value. This is when the refrigerant has some liquid to vaporize, thus, cooling the refrigerant.

Once the two-phase is detected, a PID control algorithm will be executed. In the Laplace domain, the PID equation will take the form:



Laplace: $U(s) = K\left(1 + \frac{1}{T_I s} + \frac{T_D s}{1 + T_D s / N}\right)E(s)$

Discrete Time:

$$U_P(k) = K_P e(k)$$

$$U_I(k) = U_I(k-1) + \frac{K_p T}{T_I} e(k)$$

$$U_D(k) = K_p N(e(k) - e(k-1)) + \exp\left(\frac{-TN}{T_D}\right) U_D(k-1)$$

$$U_{total}(k) = U_P(k) + U_I(k) + U_D(k)$$

Where:
$U(s)$ and $E(s)$ are the Laplace output and error respectively;
K is the proportional gain;
T_I is the integral, or reset, time;
T_D is the derivative time;
N is a constant for the filter in the derivative portion

The error term is defined as the desired superheat minus the actual superheat. Because there are only temperature transducers to measure the state of the system, the superheat is defined as the temperature at the outlet of the evaporator minus the temperature at the inlet of the evaporator. Notice that the error term is the only term changing with time. All of the other parameters are constant. These constant parameters are used as tuning parameters that are unique to each system. These parameters are usually found empirically, and are used to optimize the system.

The algorithm will be composed of three distinct modes: a delay mode, an open mode, and a PID mode. The delay mode is when the EEV is fully closed, and the clutch is engaged. This is to reduce the torque on the compressor during startup. Once the delay mode is over, the open mode begins. This is when the EEV opens some predetermined amount. The high-speed refrigerant vapor races through the evaporator, but, hopefully, the noise will be attenuated due to the throttling by the EEV. The open mode will continue until the temperature at the inlet of the evaporator drops below some known value. At this time there is now two-phase refrigerant. Finally, the PID mode is reached. Basically, the DSP will read in the two temperatures, compute the error term, then compute the PID value to be sent to EEV. Of course for the system to work efficiently, the tuning parameters must be correct.

Next, the algorithm had to be loaded into the DSP so that it can control the A/C system. A compiler is available that allows one to write a program in C, and then download it to the DSP so the DSP can encode it. The DSK board of the DSP interfaces with the host PC through the parallel port of the PC. The program “dskhost.exe” was written than allows one to interface with the DSK. Its only purpose is to download the code and then upload the data the DSP saved.

The algorithm must be coded so this specific DSP can execute the control. Figure 6 shows a flow chart for the routine.

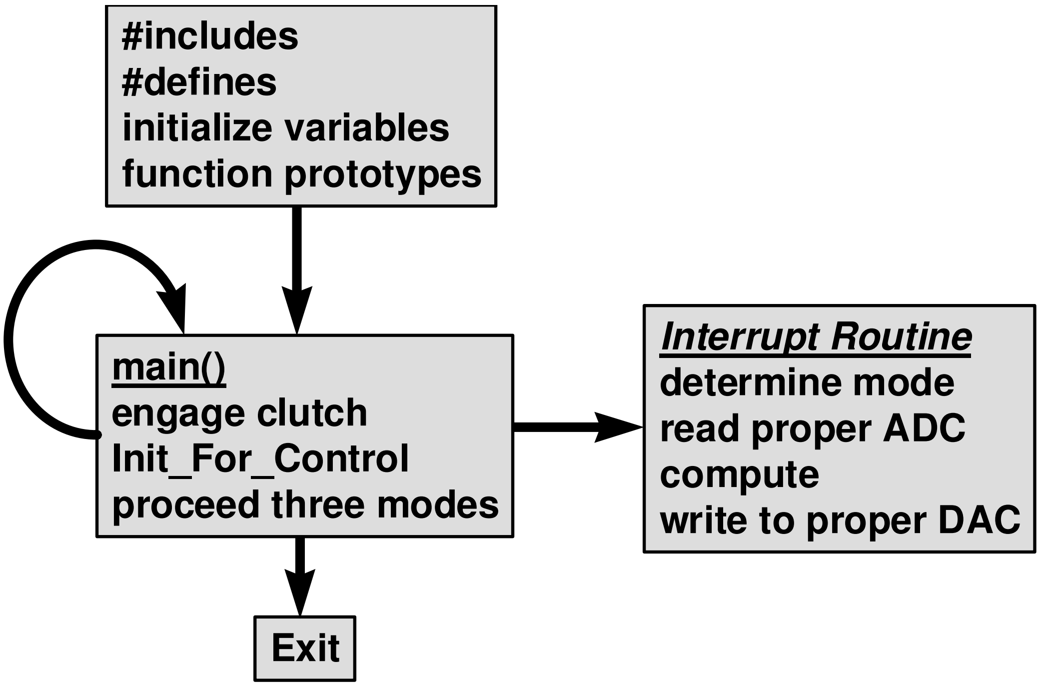


Figure 6. *This is a flow chart for the control algorithm.*

First, the includes and defines are at the top of the code. Includes tell the compiler what other routine to include when compiling. This is very important because much code was written that specifically tells the DSP how to operate. This code is in files such as dac.c (tells how to write to a DAC), io.c (interfacing with I/O), and dsk.c (describes to the DSP how to perform basic operations such as memory allocation). The defines are also important because they define global constants that are used as the tuning parameters for the specific system.

Then the routine moves to the main function. The EEV is immediately closed, the clutch is engaged, and the mode is set to the delay mode. Then, a function called `Init_For_Control` is called. This function stands for "interrupt for control" because it interrupts the main function every sample period to execute some type of control. In this case the interrupt first determines what mode it is in, reads the temperatures, determine what needs to be done, and then sends the correct control signals to the DAC's. Meanwhile, the main function is still active and proceeding through the proper modes.

Translated into some kind on computer pseudocode, the routine might take the form:

```
#includes
#define SAMPLE, DELAY_COUNT, INITIAL_OPEN, GAS_TEMP
        TARGET_SUPERHEAT, KP, TI, TD, N;
/* these are global constants, and are used to tune the routine
    for optimal control (as defined by the user) */
```

```
begin main();
  close EEV;
  mode = DELAY_MODE;
    clutch on;
    do /*nothing*/
      while int_count < DELAY_COUNT;
  mode = OPEN_MODE;
    openEEV( INITIAL_OPEN);
    do /*nothing */
      while inlet_temp > GAS_TEMP;
  mode = PID_MODE;
```

```
begin interrupt;
  if mode == DELAY_MODE
    then int_count = int_count + 1;
  if mode == OPEN_MODE
    then check inlet_temp;
  if mode == PID_MODE
    then read inlet_temp;
    read outlet_temp;
    superheat = outlet_temp - inlet_temp;
    error = TARGET_SUPERHEAT -
             superheat;
    Utotal = f(KP, TI, TD, N, SAMPLE,
               error);
    send Utotal to EEV;
```

A complete version of the actual C code used is found in Appendix A.

The routine begins with all of the relevant include files. Then, the global constants are defined. `SAMPLE` is the sample rate in seconds. `DELAY_COUNT` is the number of sample periods that the algorithm will delay before the `EEV` opens its initial amount. `INITIAL_OPEN` is the amount that the EEV will open initially defined as a fraction of fully open. `GAS_TEMP` is the temperature at he inlet of the evaporator that determines when to stop the open mode and begin the PID mode. `TARGET_SUPERHEAT` is the desired superheat at the outlet of the evaporator. `KP` is the proportional constant. `TI` is a tuning parameter for the integral portion. `TD` is a tuning parameter for the derivative portion. `N` is another tuning parameter associated with the derivative portion.

Immediately after the defines, the function prototypes are established. Functions such as `void clutch_on()` and `void clutch_off()` will turn the clutch on and off respectively. The function `void openEEV( float fractionOpen )` will tell the DSP to open the EEV. When `fractionOpen` is 1.0, EEV is fully open; When `fractionOpen` is 0.0, EEV is fully closed. Refer to Appendix A for all of the function prototypes.

After the prototypes are defined, the global variables are initialized. `int_mode` (or `mode` in the pseudocode) is the mode that the control routine is in (e.g. delay, open, or PID). `int_count` is the number of times that the interrupt routine has been executed. `inlet_temp` is the temperature at the inlet of the evaporator. `outltet_temp` is the temperature at the outlet of the evaporator. `superheat` is the amount of superheat, that is coming out of the evaporator. `ek` (`error` in the pseudocode) is the current error defined as the difference between the actual superheat and the target superheat.

Now the routine is ready for the main function. First the EEV is closed. Then, the `mode` is set to `DELAY_MODE`. Next, the clutch is turned on, and then a `do-while` loop is entered. Meanwhile, every sample period the interrupt routine is executed. In the interrupt routine while the `mode` is equal to `DELAY_MODE`, the variable `int_count` will be incremented by 1. Back in the main function, the `do` loop is doing nothing `while int_count` is less than `DELAY_COUNT`. But, once `int_count` has been incremented to `DELAY_COUNT`, `mode` is set to `OPEN_MODE`. Meanwhile at the interrupt routine, the DSP is reading `inlet_temp` from the ADC. Back at the main function, the `do` loop is doing nothing `while inlet_temp` is greater than `GAS_TEMP`. But once `inlet_temp` is equal to `GAS_TEMP, mode` is set to `PID_MODE`. Now, at every sample period the DSP will read in `inlet_temp` and `outlet_temp`, compute `superheat`, compute `error`, and find `Utotal`, which is the numerical solution to the PID equation. `Utotal` will then be sent to the EEV.

Conclusions and Future Work

Automobile consumers complaining about noisy air conditioners began this investigation for better control of such an A/C system. Besides reducing noise at startup, the new control algorithm reduces torque imposed on the compressor at startup, and recommends an algorithm for steady-state. The algorithm would be coded in C and downloaded to a DSP designed by Mr. Dan Block. Once encoded into the DSP, the routine would proceed through its three stages. First, the EEV would close, the clutch would engage, and then it would wait until the mode was over. This stage was responsible for reducing torque at startup. Next, the EEV would open some initial amount until two-phase refrigerant was detected. This stage was implemented to reduce the noise by restricting the flow. During the final state a PID routine is executed to control the system once two-phase refrigerant is detected.

The code written for this project was done so to be as general as possible. This was because the controller used for the actual experimentation will not be the one described in this paper. Although the new controller will use a C compiler to download its instructions, many of the commands will be different.

The new controller and the circuitry described earlier will be setup in the Mobile A/C Lab in room 115K MEL. Once everything is properly installed, the code will be tuned to that specific system. Experiments will be conducted to verify this proposed solution.