Closed-Loop Feedback Controller of DC Motor's RPM

The project objective was to design and prototype a software controlled DC motor closed-loop system, utilizing the 80515 microcontroller development board. The primary goals were to model the entire system theoretically and in MATLAB Simulink to determine an appropriate controller design, to design the hardware interface for the microcontroller, joystick, and motor, and to develop assembly code for the user interface and for the microcontroller to control the motor's RPM with both open-loop and closed-loop feedback control. Key goals were to meet design specifications, to design interrupt driven software, and to choose appropriate control criteria and to optimize these criteria. Also, a working prototype was to be built and tested in order to give a demonstration of the design to Dr. Dempsey.

The system's analysis and design tasks were divided between Adam Olson and Simon Benik. Adam Olson's tasks were first to analyze and evaluate the system's Simulink model, and then to design, build, and test the hardware interface for the various subsystems, create and document flow charts and schematics, and then finally to assist in the software design, specifically the error control and closed-loop feedback modules. Simon Benik's tasks were to create the vast majority of code for the microcontroller including modification of user interface code from a previous project, timer 0 and timer 2 code design for timing required in product specification, PWM signal generation, RPM calculation, and I/O interfacing, particularly in joystick switch scanning.

The product specifications required the LCD display to be 2-lines in normal display mode. Upon power up, the display is "Mini-Project 2006". After 0.75 seconds, the normal mode is displayed which is line 1: "RPM=WWW" and line2: "PWM=YYY.Y%", where the increments in PWM are to be at most 0.2%. When the 'C' key is pressed on the keypad, the display should enter keypad entry mode, which is line 1: "A=HOME D=GAIN E=P/PI" and line 2: "F=JOYSTICK SCAN". When 'A' is pressed, the display should return to normal mode. When 'D' is pressed, the display should switch to only line 1 reading "Enter Kp:000-255", and if an invalid entry is entered, an error message should be displayed. If 'E' is pressed, the display should switch to only line 1 reading "Enter 1=PRO,2=PI", and if an invalid entry is entered, an error message is displayed. If 'F' is pressed, the display should switch to a message of only line 1 reading "Enter ms:005-250" to enter the scan rate, and again, if an invalid entry is entered, an error message is displayed. The LCD refresh rate must be 0.5 seconds.

The power supply provided is 20VDC, and is used for the motor and external circuitry. The PWM signal must have a fixed period of 1ms, and a variable duty cycle from 0% to 100% in 0.2% increments or less. The display accuracy of the RPM reading must be ± 40 RPM at the most and this RPM reading must be displayed every 0.5 seconds.

The joystick scan period must be every 40ms by default, and be able to vary from 5 to 250ms by user selection. The joystick must be powered by the EMAC's 5VDC power regulator. Button D2 and D3 of the joystick increase and decrease the RPM respectively when the trigger button is pressed. Button D4 activates the motor brake and exits from the closed-loop control when pressed alone. If D4 is pressed while the trigger is active, the closed-loop control is initiated and enabled.

During closed-loop operation, the center (bias) velocity is set via the joystick buttons. If the error signal exceeds the software limits (overflow), the controller must revert back to the original duty cycle at closed-loop initiation and remain in closed-loop mode. The proportional controller gain must be obtained for maximum motor supply voltage range, and velocity accuracy versus supply voltage must be recorded. Extra credit was given for the D/A channel to monitor and display the error signal as 2.5V (0 error) ± 2.5V error.

The last requirements of the product specifications are that the product operates at a temperature range of 0 to 40 degrees C. Also, An LED must be updated at the LCD refresh rate on port 4.0, port 4.1 must be toggled to show 1ms timing, port 4.2 must be toggled to show the joystick scan rate, and port 4.3 must show the 1ms interrupt execution time. All of the software must be interrupt driven and designed in a modular programming method.

The system controls the RPM of a DC Pittman motor. A functional block diagram of this system is shown in figure 2-1.

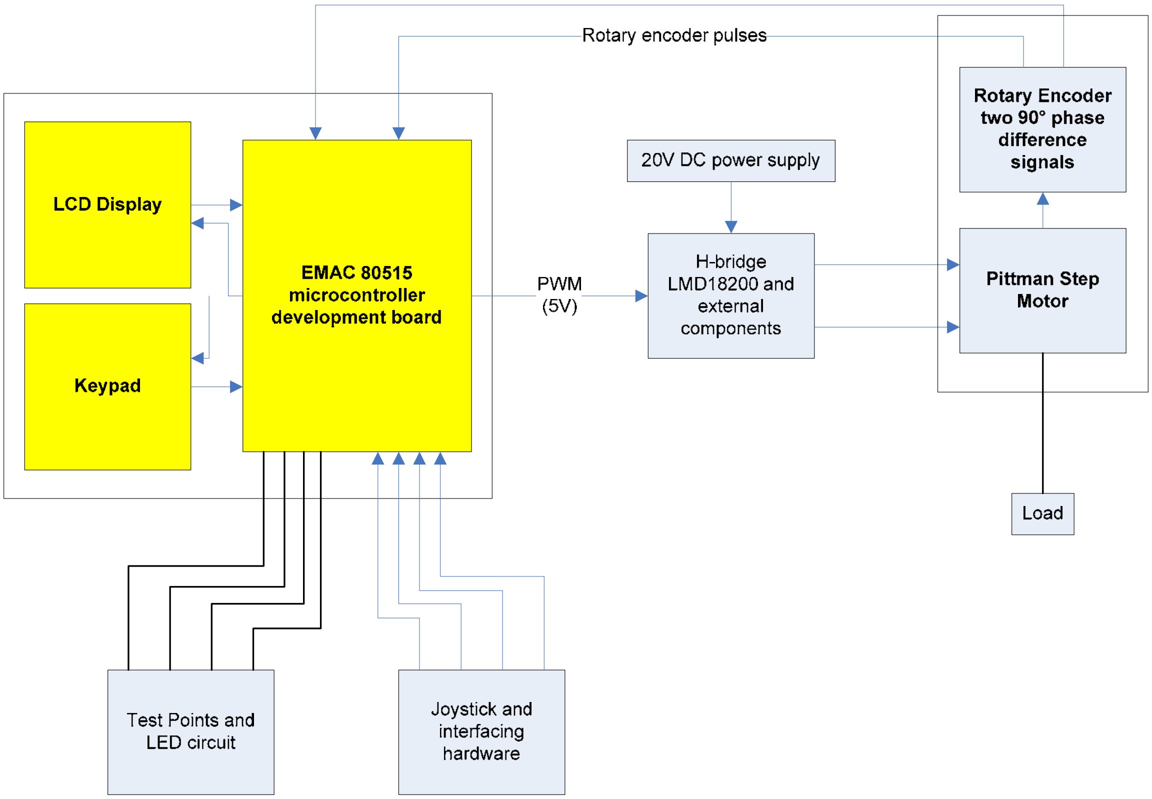

Rotary encoder pulses
LCD Display
Keypad
EMAC 80515 microcontroller development board
PWM (5V)
20V DC power supply
H-bridge LMD18200 and external components
Rotary Encoder two 90° phase difference signals
Pittman Step Motor
Load
Test Points and LED circuit
Joystick and interfacing hardware

Fig. 2-1: System functional block diagram with subsystems

The 80515 microcontroller development board's LCD and keypad are used to interface with the user of the system. The high level flowchart of the software subsystem is shown in figure 3-1.

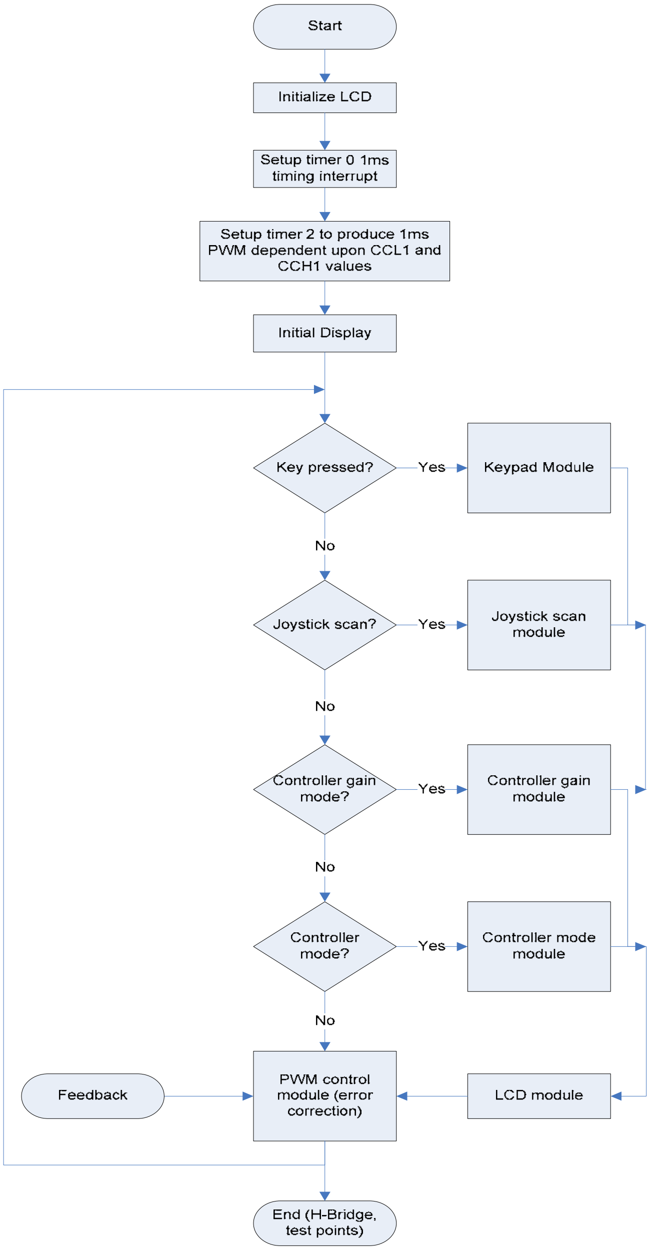


Fig. 3-1: High level flowchart of software

The keypad routine is interrupt driven, and upon its service routine, the ASCII character in the accumulator is compared to selected ASCII values to determine what character was pressed and thereupon executing the appropriate action for the character. The keypad flowchart is shown in figure 10-1. The LCD flowchart is shown in figure 11-1. The LCD function is called, where the LCD status registers are read as external memory locations to determine the status of the LCD. If the LCD is not busy, different cases will be checked to determine what value is written to the LCD.

The system begins in open-loop control, where the user controls the PWM signal generated by the microcontroller which corresponds to a specific RPM value displayed. The flowchart for this part of the code is shown in figure 12-1. The RPM value is read from two pulses out of the rotary encoder of the DC motor. The two pulses have a 90 degrees phase difference, which allows for two pulses rather than one pulse to be counted, thereby doubling the resolution for RPM calculations. These pulses drive two pins that generate interrupts, where the pulse value is incremented. Every 1ms, timed by timer 0, the pulse count is checked and cleared, and a corresponding RPM is found and displayed using a look up table in memory. Timer 2 is setup for generating a PWM signal with the 1ms period of timer 0. A value of FFFFh – 922d is the minimum value loaded into timer 2, corresponding to a 100% duty cycle, and a maximum value of FFFFh – 1h corresponds to a 0% duty cycle. The value is incremented and decremented when either the D2 or D3 switches of the joystick are scanned as high in conjunction with the trigger switch scanned as high. This gives a resolution of (100%/922) = 0.108% duty cycle increments, which is nearly half of the required spec of 0.2%. When a new value is initiated by the joystick scans, the incremented value is stored in CCL1 and CCH1 to be reloaded into timer 2 for its overflow interrupt timing, which toggles P1.1 for PWM generation. The reload value is also used to look up a corresponding value in a look up table in memory to display the duty cycle on the LCD.

The PWM signal is signaled to external circuitry which is used to interface the 80515 microcontroller development board with the Pittman motor. The signal is used as an output via a pull up resistor to toggle the PWM input pin 5 of the H-bridge. The H-bridge circuitry is shown in figure 4-1.

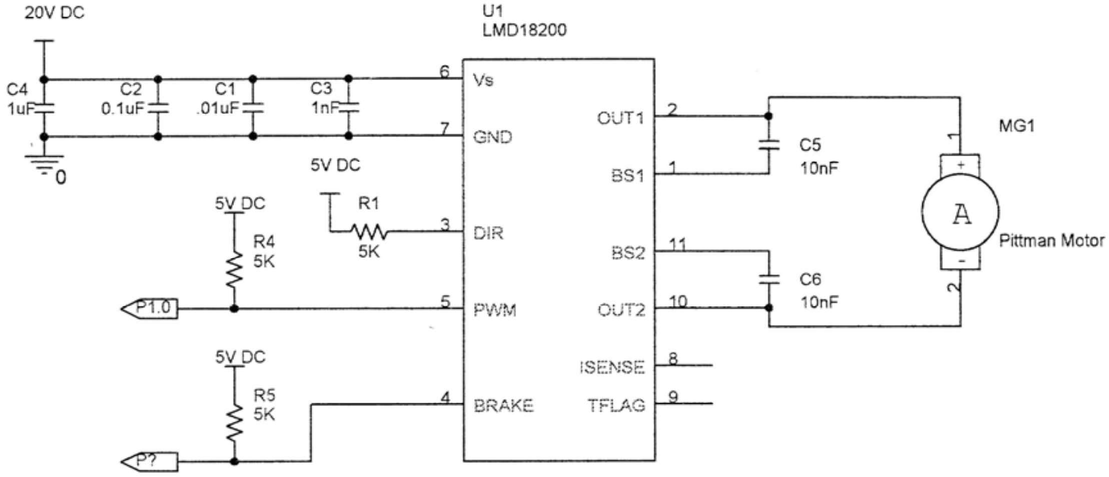
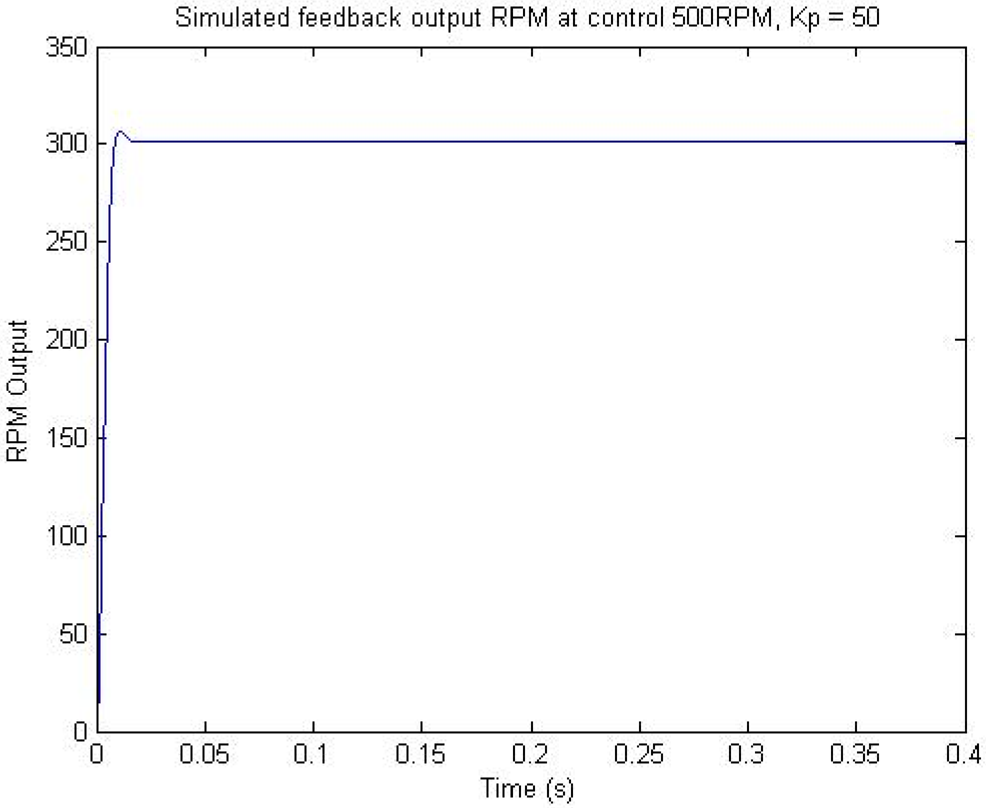


Fig. 4-1: H-bridge circuitry used to control the motor and interface with the microcontroller.

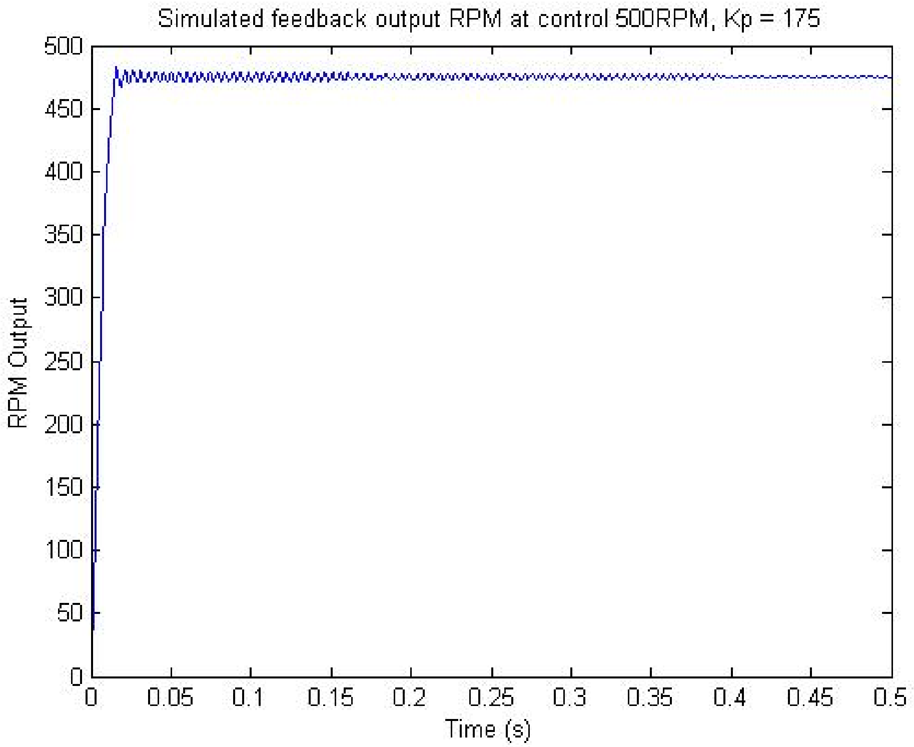
The direction control is hardwired rather than software controlled. The brake pin is connected to P3.7 of the microcontroller which, when pulled low by software, will allow the motor to run, and when pulled high, will cause the motor to stop. Pins 1 and 11 are connected to the two output pins through 10nF capacitors. These bootstrap pins allow the output pins to pull extra energy from the capacitors when necessary. Depending on the direction set, one output pin is always at 20V, which is the supply voltage, while the other pin pulses to ground corresponding to the input PWM from the microcontroller. Using pull-up resistors of 5KΩ with 5V sources allowed a maximum current flow through the microcontroller ports to be 1mA which is within the safe current limits of the 80515 data sheets. The LED required a 250Ω resistor to limit current within safe levels.

The H-bridge drew about 100mA of current, so at its maximum operating voltage of 20VDC, the power dissipation is 2.1316W (calculated in lab book pages 87-89) which, at room temperature, is within the limits of the H-bridge safe operation. Total power dissipation on average is 2.3076W. The entire system circuitry design is shown in the schematic of figure 14-1.

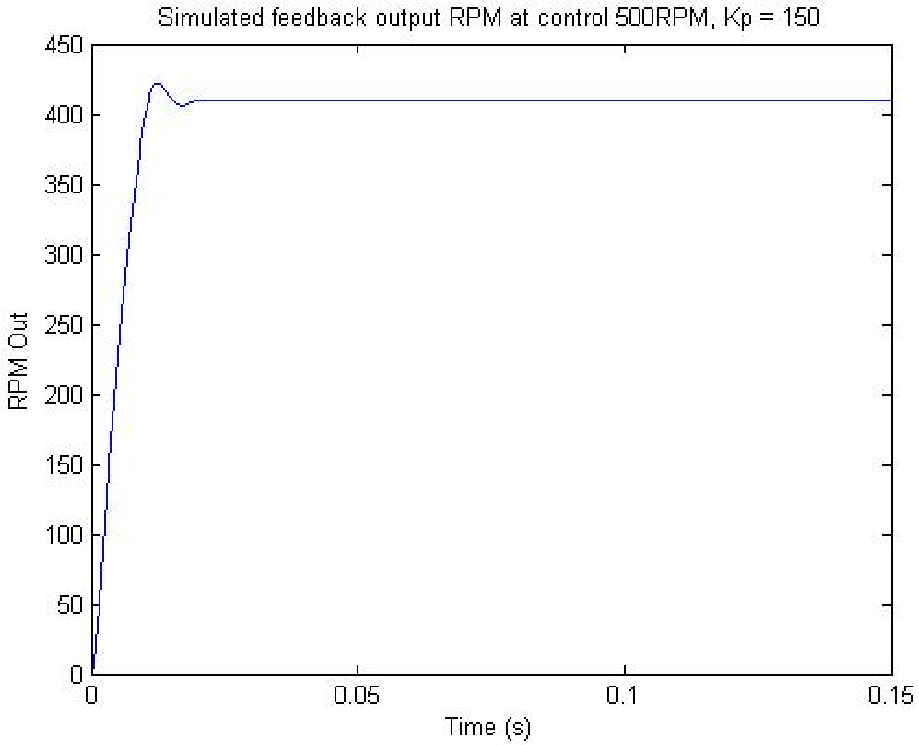
The closed-loop control code was designed to operate upon user activation, where the current pulses counted and the timer 2 load values were stored. The optimum proportional gain was determined to be 150 in Simulink because at lower kp values, for instance in figure 5-1, the steady state RPM was considerably lower than the control RPM, and at higher than 150 kp values, in figure 6-1, the steady state RPM became unstable and oscillated. Overshoot and rise time were taken into account as well, as discussed in the lab book. The compromise was a kp value of 150, whose transient and steady state properties are apparent in figure 6-2.



<u>Fig. 5-1</u>: Simulated RPM with kp of 50.



<u>Fig. 6-1</u>: Simulated RPM with kp of 175.



<u>Fig. 6-2</u>: Simulated RPM with kp of 150.

The algorithm that manipulated the closed-loop control of the PWM signal is shown in the flowchart of figure X.X. The kp value was multiplied by the error and then added or subtracted, depending on the sign of the error. If at anytime the error adjustment exceeded the bounds of FFFFh-1h or FFFFh-922d, the timer 2 reload values would revert back to the original values at the initiation of the closed-loop. Closed-loop control was never achieved in lab.

The experimental results of the open-loop RPM at given average voltages (duty cycle of 20V) is compared to the simulated values to give a comparison of how the system performs. Table 7-1 shows the RPM data comparisons. The prototype is least accurate at very low RPM and then greatly improves accuracy, giving the system an average RPM percentage error from the simulated system of 11.34%.

Table 7-1: Experimental RPM of prototype compared to simulated RPM

Average Voltage	RPM of simulated system	RPM of prototype	percent error
1V	30	20	33%
5V	145	140	3.40%
10V	295	260	11.80%
15V	440	410	6.80%
20V	590	580	1.70%
		Average Error:	**11.34%**

The average difference in RPM from simulated to experimental, however, is only 18 RPM, which is less than the required resolution of RPM display in the specifications, so this is a better representation of how the experimental prototype matches the simulated RPM. The system open-loop mode RPM values are accurate and also, the PWM value displayed on the LCD was accurate within 0.1% duty cycle when compared to measured duty cycle values using the oscilloscope.

A bill of material for the external circuitry only is shown in table 7-2.

Table 7-2: Bill of materials for external circuitry

Component	Quantity	Value	Manufacturer	Cost per unit	Total Cost
C1, C5, C6	3	0.01uF	Kemet	$0.06	$0.18
C2	1	0.1uF	Kemet	$0.06	$0.06
C3	1	1nF	Kemet	$0.06	$0.06
C4	1	1uF	Kemet	$0.06	$0.06
R1-R8	8	5KΩ	Panasonic	$0.04	$0.32
R9	1	250Ω	Panasonic	$0.04	$0.04
U1	1	LMD18200	National	$11.70	$11.70
Single unit cost					**$12.42**
10,000 units cost					**$124,200.00**

The values were obtained from www.digikey.com and only reflect an estimate of what large volume production costs may be for these components. A total list of parts is shown in table 8-1. The microcontroller development kit is used rather than the individual components of the kit, such that components on the kit may be unnecessary in the scope of the system design. Also, the HP DC power supply is included as the voltage regulator for the prototype, although this is impractical for most market products.

Table 8-1: Final parts list

Component	Quantity	Value/Part #	Manufacturer
Vsupply	1	20VDC	HP
EMAC 80515development kit	1	Micropac 535	Keil
DC Motor	1	GM9000	Pittman
Joystick	1	Wingman	Logitech
C1, C5, C6	3	0.01uF	Kemet
C2	1	0.1uF	Kemet
C3	1	1nF	Kemet
C4	1	1uF	Kemet
R1-R8	8	5KΩ	Panasonic
R9	1	250Ω	Panasonic
U1	1	LMD18200	National

Much success was achieved for the project, however, not all of the goals were completed. Closed-loop control was never implemented correctly. As discussed in the lab book, when the closed-loop control was activated, the PWM signal was no longer generated, even while the PWM reading on the LCD was changing, and the timer 2 reload registers during debugging were changing. Another partially non-functional minor problem was that the display would not display down to absolute 0% PWM. Also, although the keypad entry mode was implemented for the kp value entry, the value was never adjusted by the user, it remained a constant 150. All open-loop code worked almost flawlessly, and with better performance than the product specifications in many cases. The hardware was interfaced correctly between all subsystems and worked in the prototype as designed. Adam Olson received a 92% for the demo grade, and most of the penalization came in the closed-loop functionality. Simon Benik received a 97% for the demo grade by being penalized for the incomplete PWM display.

Product improvements could include an actual user selectable kp value as mentioned above. Also, various other control designs could be utilized, primarily the integrator that was never designed in code. The integrator would improve the steady state RPM greatly with minor transient setbacks. Also, direction control could be implemented because one of the H-bridge's primary features is the ability to switch polarity of the source voltage to the motor. The H-bridge consists of four switches in an H circuit configuration, and these switches can be manipulated with the direction pin, which was tied to ground in our prototype. If this pin were to be toggled with the microcontroller, the motor could be controlled in both directions. Also, the H-bridge's thermal flag could be implemented in the design to allow for safety protection to the circuitry by monitoring the temperature level with the microcontroller, where duty cycle could be decreased or completely set to 0% when the temperature level is too high. Additional design features could be to use the potentiometer of the joystick to give a voltage level to the microcontroller's A/D converter where the position of the joystick could be monitored. Also, the D/A converter could be utilized as suggested in the product specifications to generate a voltage level between 0V and 5V that is proportional to the error value used by the closed-loop feedback control.

Project design hours for Adam Olson were approximately 32 hours, with almost equal time being spent in hardware design, Simulink analysis, and closed-loop code design. Simon Benik spent approximately 34 hours, with all time being spent in all aspects of microcontroller system design and programming.

The hardware design time was approximately 11 hours, where the final design took 6 hours to reach, debugging and building took approximately 3.5 hours over the course of the project including reassembly of the circuits each lab day. Testing took approximately 30 minutes, and one hour was spent creating schematics.

The software design time was approximately 55 hours. 30 hours were spent designing and coding. About 10 hours were required for debugging and testing of the software designed. Five hours was spent for software documentation, primarily in flowcharts and lab notebook data. Simulink modeling and analysis took 10 hours and was used in the software design.

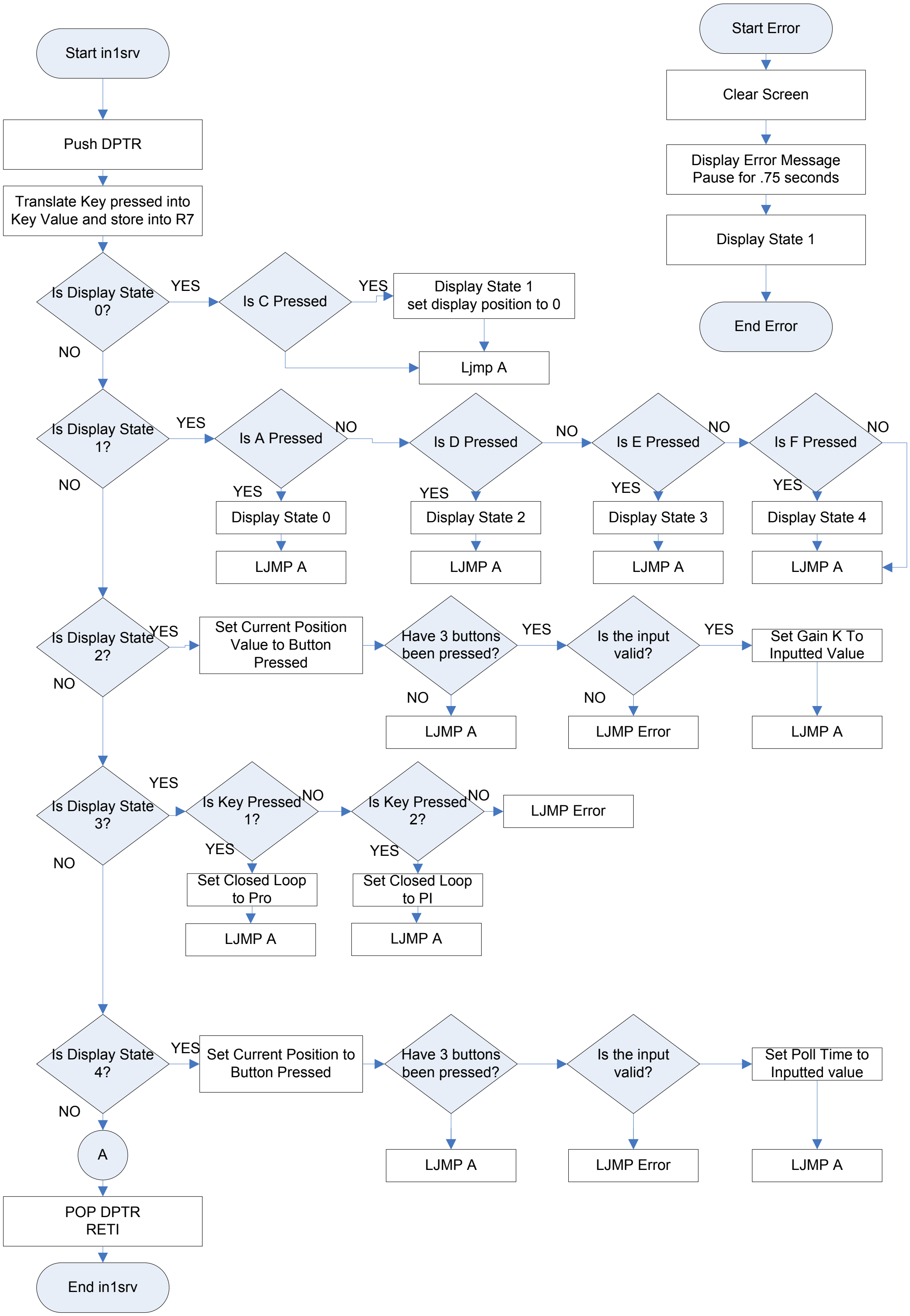


Fig. 10-1: Keypad subsystem flow chart.

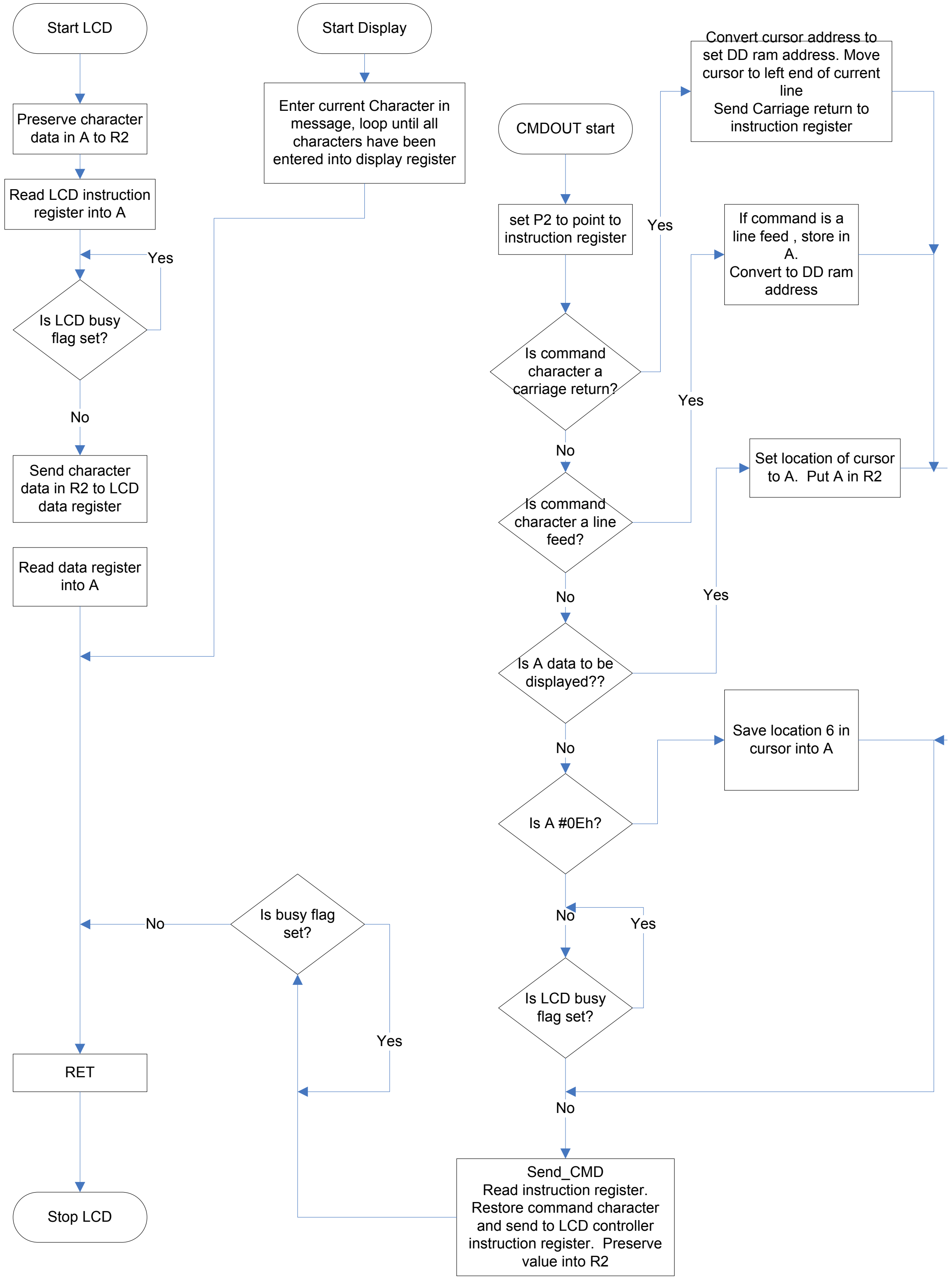

Start LCD
Preserve character data in A to R2
Read LCD instruction register into A
Yes
Is LCD busy flag set?
No
Send character data in R2 to LCD data register
Read data register into A
RET
Stop LCD
Start Display
Enter current Character in message, loop until all characters have been entered into display register
No
Is busy flag set?
Yes
CMDOUT start
set P2 to point to instruction register
Is command character a carriage return?
Yes
Convert cursor address to set DD ram address. Move cursor to left end of current line
Send Carriage return to instruction register
No
Is command character a line feed?
Yes
If command is a line feed , store in A.
Convert to DD ram address
No
Is A data to be displayed??
Yes
Set location of cursor to A. Put A in R2
No
Is A #0Eh?
Save location 6 in cursor into A
No
Yes
Is LCD busy flag set?
No
Send_CMD
Read instruction register.
Restore command character and send to LCD controller instruction register. Preserve value into R2

Fig. 11-1: LCD subsystem flow chart.

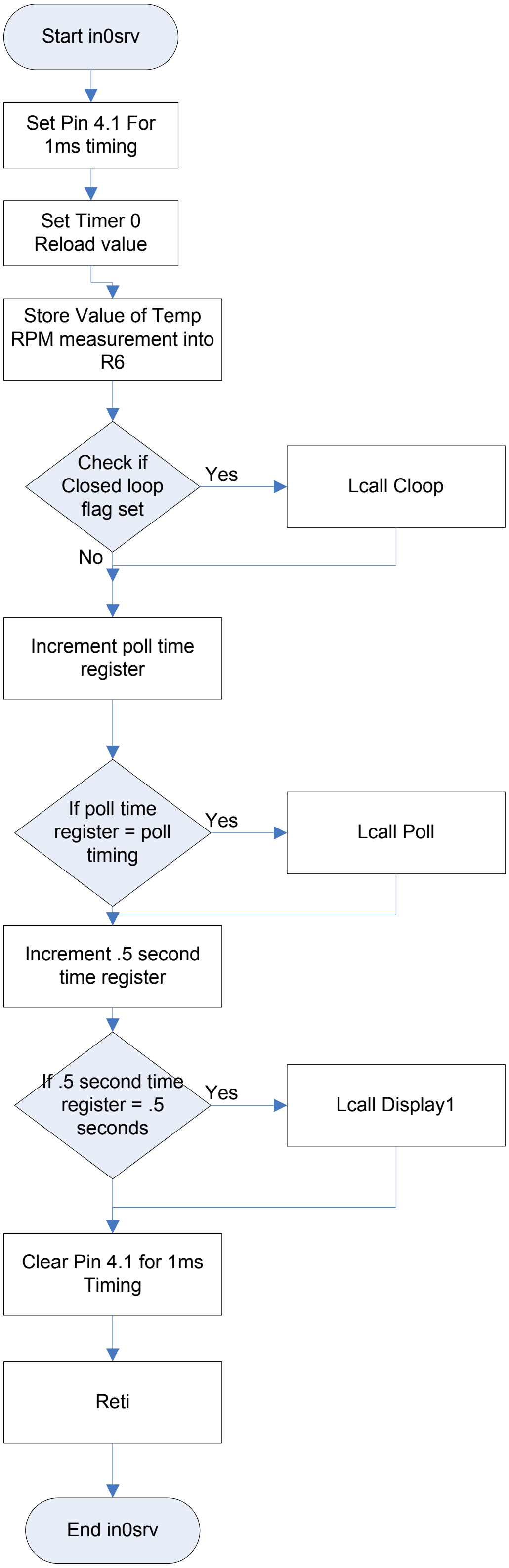

Start in0srv
Set Pin 4.1 For 1ms timing
Set Timer 0 Reload value
Store Value of Temp RPM measurement into R6
Check if Closed loop flag set
Yes
Lcall Cloop
No
Increment poll time register
If poll time register = poll timing
Yes
Lcall Poll
Increment .5 second time register
If .5 second time register = .5 seconds
Yes
Lcall Display1
Clear Pin 4.1 for 1ms Timing
Reti
End in0srv

Fig. 12-1: Interrupt service routine for Timer 0.

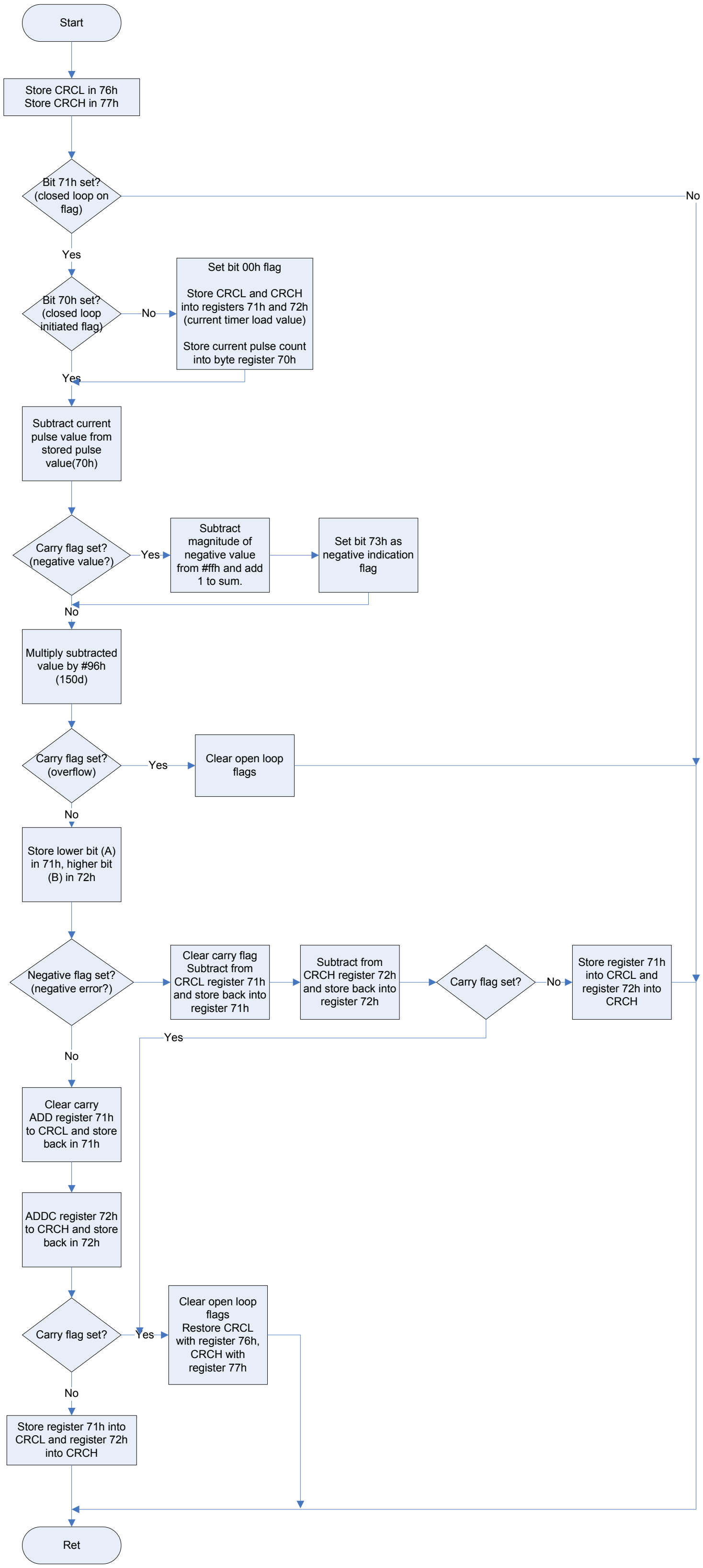


Fig 13-1: Closed-loop control subsystem flow chart.