

## **Table of Contents:**

Introduction	2
Main Loop and Opener	3
Barcode Function	5
Digital Lock	6
Sprayer Function	7
AddFluid Function	8
Conclusion	9
Assembly	
Main.asm	
C Code	
AddFluid.h	
AddFluid.c	
barcode.h	
barcode.c	
barcode_tables	
main	
Opener.h	
Opener.c	
password.h	
password.c	
Sprayer.h	
Sprayer.c	

## **Introduction:**

Smart Farming is a user friendly interface combined with an intelligent choice making system that implements a crop sprayer. The system can adapt to both crop conditions and user input through the PowerPC controls.

After a brief animated introduction and a mandatory password protected entry, the user is catapulted into an enormous array of options. This stockpile of selections has been cleverly integrated to provide the farmer with the information, resources, and intelligence to farm well.

A help screen informs the farmer which functions he can call. By calling the password function the farmer can reset his password or lock up his system. Calling the barcode function lets the farmer 'scan' the barcode of the current crop he is treating. The program does some error checking and records the information received for later use.

Before the farmer can spray a field, he must fill up his tank with fluid. AddFluid performs this task admirably, allowing the farmer to control the speed of the pump and fill exactly the amount he/she needs. A bargraph graphically displays the fluid in the tank and the percentage filled is displayed on an LCD screen.

Next, the farmer can spray his field by calling Sprayer. This function computes the exact spray pattern that will best help his/her crops. Again, the farmer is allowed a hands on approach to farming. The farmer can increase the pressure and cycle speed of the spraying and watch the fluid level remaining on an LCD panel.

If the farmer forgets his options he/she is always free to check the handy help screen.

During the entire process, the farmer is in direct communication with the home base, utilizing a stylized messaging service. This allows the farmer to seek advice, make plans, and complete the job in record time.

Smart Farming is just that. Smart. Sophisticated. Intelligent. Farming at its best.



“Tractor:” and “Base:” were added by hard coding them into the first 8 and 5 spaces of the PC and QT buffers. Also added to these buffers was formatting for the HyperTerminal exchange. By always resetting the write pointers to zero, but setting the read pointer to 9 and 6 past the beginning of the array, the program will print who sent the message and will never overwrite these labels. These efforts transform the unwieldy lab 8 protocol into something more refined.

Also added was the ability to call all the functions and utilize the help screen from this main program loop. Since the Qterm keyboard only has 5 keys, and there are many more functions than that, something additional was needed. By adding functionality to the shift key, the available options were doubled. A register holds a value signifying whether the shift key had been pressed, and several branch statements control which C functions can be called.

The relevant data for spraying each field are held in registers r30 and r31. as iCurrentSeedType, which signifies what kind of crop is being worked with, and iSprayerFluid, which stores how much sprayer fluid the tank holds. Some of the functions one variable, and sprayer.c needs both variables, so the mains.asm loop puts the required variables into r3 and r4 before calling each function, then takes the result out and reassigns it to the proper registers.

The ASCII animation is an interesting addition. Making ASCII art is not difficult, but takes a great deal of time since everything must fit within the particular screen that will be used. By making full use of the member functions, the cursor can be moved back to the top left-hand corner, which causes less flicker in the screen. By using a variety of loops and assignment operators, a great deal of laborious recopying is eliminated. The QTerm LCD screen is 20 characters wide by 4 rows deep. You can make a message 100 characters wide and 4 deep; then just assign the characters i to i+20 to be printed to the screen. The msleep function is well suited to make animated screen art and once the programmer grows used to the setup, the number of designs available is only limited to the programmers’ imagination.

# Barcode Function:

by Wendell Cotton

## Project Functionality:

④

The barcode function is used to determine which spray pattern to use for which crop. This function was based on the code submitted for Lab 4. It is broken up into three different parts. The first part is the header file for the entire function. The header contains all of the function prototypes that will be used for the other functions that make up the entire Barcode Function. The second part is the Barcode\_tables.c function. This function contains all of the tables used to determine the values for the various digits in the barcode. It also contains all the functions used to determine if the barcode is finished transmitting or not. The third and final function is the function that actually does all of the executions for the program.

When the program is executed the user will first see the instructions for the program. The user will then be prompted to enter a value from the table below for the barcode. After they enter the value the user must then use bit 7 of either Dip Switch 1 or Dip Switch 2 to determine what type of crop to use, beans or corn. Once the user has selected what crop to use they will see the complete barcode. If the user is satisfied, they must terminate the program by setting bit 7 of both Dip Switch 1 and Dip Switch 2 to low.

DIGIT	LEFT-HAND ENCODING	RIGHT-HAND ENCODING
	ALL CHARACTERS	ALL CHARACTERS
0	0001101	1110010
1	0011001	1100110
2	0010011	1101100
3	0111101	1000010
4	0100011	1011100
5	0110001	1001110
6	0101111	1010000
7	0111011	1000100
8	0110111	1001000
9	0001011	1110100

## Project Design:

The barcode function is designed to take a specific value in through the Dip switches, convert that number to a decimal number 0-9, calculates a check digit, and uses that check digit to determine which spray pattern is to be used by the sprayer. It accomplishes this through various modules in the program. The program will not calculate a check digit until bit 7 of either Dip Switch 1 or Dip Switch 2 is set to high and will terminate and send the last calculated value of the check digit to the program that is used to determine the spray pattern.



# Digital Lock:

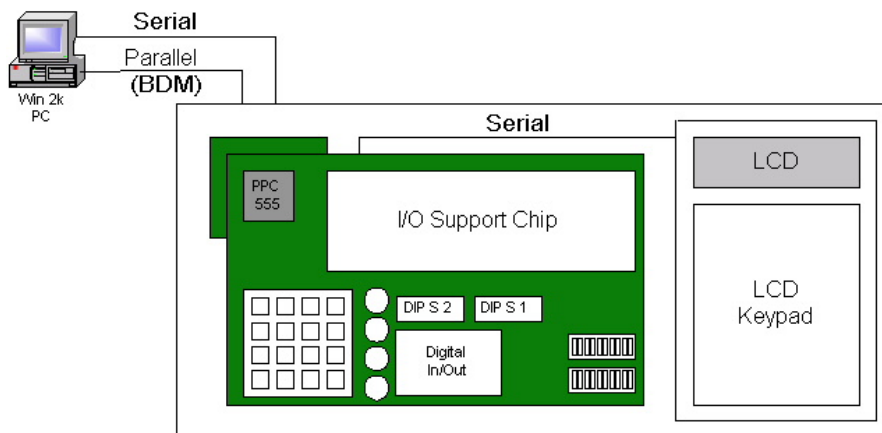
By Michael Fong

③

## Project Functionality:

This program provides the user a more secure system for the sprayer tractor. The Password system not only allows checking of the password, but also allows the user to modify his/her password freely. Moreover, a second confirmation has been added when the owner tries to change a new password.

The program requires the user to key in the password through the keypad, and displays this on the LCD screen, which is shown as the diagram below:



The keypad has four rows and four columns of keys, where the character # is defined to be the terminal character of password.

1	2	3	A
4	5	6	B
7	8	9	C
*	0	#	D

## Project Design:

The code is build on a body function called 'pwd\_Main()', where all other helper functions are called.

First of all, the program checks to see if this is the first-time login for the owner. If it is the first time, the program allows the user to create his/her own password in order to get rid of the default password 'NULL'.

Due to the internal keypad problem, the Power Box might not grab the correct key. Hence, a patch feature was added to the program. Each time a new password is modified, the user will be asked to confirm the password.

After the user logs on, the program allows the user to change to a new password or keep the old one.

The maximum length of the password is 256 characters.

.

# Sprayer Function:

By Piyush Patel

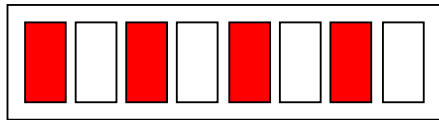
## Project Functionality

⑤

Sprayer function uses the analog knobs to change the pressure and cycle speed of the tractor sprayer and displays the fluid left, the pressure, and the cycle speed on the LCD screen. For creativity, the pattern of the sprayer and fluid level is viewed on the LED Bargraphs.



Sprayer function is called using the smart farming menu through Q-Term. Q-Term LCD will display how much fluid is in the tank, the pressure, and the cycle speed



of the sprayer. LED Bargraph 1 will show the pattern that is used to spray the fluid. LED Bargraph 2 shows the fluid level; if bits 0-7 are lit up, then it means the fluid tank is full or almost full

and if only bit 0 is lit up, it means the fluid tank is getting close to being empty. The main functionality of the sprayer function is the analog knobs. Analog knob 0 is used to change the pressure of the sprayer and analog knob 1 is used to change the cycle speed of the sprayer. Increasing the analog knob 0 will increase the pressure, therefore the sprayer will use slightly more fluid than before. Similarly, increasing analog knob 1 will increase the speed of how fast the sprayer sprays. This will use even more fluid because the sprayer will spray faster. Due to that, LED Bargraph 1 will display the pattern faster and the fluid level will go down faster on LED Bargraph 2. To manually exit the sprayer function, turn on bit 0 of DIP Switch 1. Otherwise the function will automatically exit when the fluid left reaches 0.

## Project Design

Sprayer function is completely programmed in C with a variety of new features added to Lab 5. The main part of the function is the while loop where all the user controlled features are handled. Depending on the value passed in for `iCurrentSeedType`, a particular pattern function is called to print the pattern on LED Bargraph 1. To print on the LED Bargraphs, a variable is declared to hold the value to be printed, and then accordingly a bargraph function is called with the variable as the parameter to print. “`msleep(1000-AV1*30)`” is an algorithm used to increase or decrease how fast the pattern is printed on bargraph depending on the value of AV1, which is the value read in from analog knob 1 and converted so it is between 0-100. I used one more random algorithm to decrease the fluid depending on input from analog knob 0 and analog knob 1. All of the LED Bargraphs are reset before exiting. If the fluid goes down to 0 at the end, a message will be displayed on the LCD screen to inform the user.

# AddFluid Function:

*By Ankur Tandon*

## **Project Functionality**

⑤

AddFluid function, as the name suggests, adds life (fluid) to the Smart Tractor. With the use of analog knobs, the speed of refueling can be adjusted at any time. The percentage of the tank filled is displayed on the LCD screen. The features that make this function and the tractor overall 'smart' is the pattern displayed on the LED Bargraphs outputting the level of fluid in the tank and the handy feature that allows the smart farmer to exit refueling any time.

With the right selection from the smart farming command options on the Q-Term, AddFluid function can be called by the farmer any time he wants to add more life for better performance of his tractor. Q-Term LCD displays how much fluid is in the tank.



LED Bargraph 2 shows the fluid level; if bits 0-7 are lit up, then it means the fluid tank is full or almost full and if only bit 0 is lit up, it means the fluid tank is close to being empty. The refill speed, another important aspect of refueling the tank, can be adjusted by using the analog knobs. Increasing the analog knob 0 will increase the refill speed, thereby filling the tank at a faster rate than before. In order to make the tractor smart, Bit 0 of Dip Switch 1 has been left so at his/her discretion, the farmer can manually exit the refueling process. Turning on bit 0 stops the refueling process and returns the exact amount of fluid to the main function. Thus a farmer can fill the tank any time according to his own interest.

## **Project Design**

AddFluid function is completely programmed in C. A variety of features were added to Lab 5 to make this function more farmer-friendly. The main part of the program is the recursive while loop which continually checks for the farmer's need through out the function. Depending on the fluid level passed from the main function, AddFluid adds more to it continuously regulating the speed as desired by the smart farmer. Continuous level increase inside the tank is displayed on the LED Bargraph 2. To print on the LED Bargraphs, a variable is declared to hold the value to be printed, and then a bargraph function is called with the variable as the parameter to print. All of the LED Bargraphs are reset before exiting. If at any time the user sets bit 0 of Dip Switch 1, the current value of fluid level is returned to the main function. As a courtesy, it informs the user of the exact value of fluid level before exiting the loop.

## **Conclusion:**

The ASCII animation in `opener.c` works well and makes use of the wide range of art that can be used using timers and the available characters. The `main.asm` program contained some pitfalls. Different machines use different 'shift' key values, and some shift keys on the Qterm keyboards do not work at all. Because of this, a great deal of unnecessary grief came to the team. Integrating the bits of software proved challenging, because relevant variables had to be assigned and the list of who could modify what had to be made strict. An interesting challenge came from doing the loop in assembly. Memory concerns had to be taken very seriously, as only a limited number of registers were available and nothing could afford to be overwritten. The programmers became even more familiar with the restrictions of EABI and appreciated the large number of nonvolatile registers. In the main loop, the most interesting functionality was the coding of labels for the function. By simply modifying all the read and write reset functions, names could be assigned to the keyboard typing. This neat bit of programming really beautified the system.

The barcode function does exactly what is requested of it. Unfortunately, the programmer was unable to implement any software to check and make sure that the values entered in through the Dip Switches correspond to values in the table. This could cause some serious problems with the spray pattern if the user does not enter the correct value for the Dip Switches. In addition if the user terminates the program at the exact second that the program is checking to see if the barcode is ready to be read in or not, the program might possibly not return any value at all and the program may need to be re-run to get the value for the program. Fortunately the use of a real barcode scanner should avoid these problems.

The new digital lock offers a lot more flexibility and security than the original, plus a user-friendly interface. This allows users to easily get familiar with it

The sprayer function usage of analog controls tested our knowledge of analog to digital conversions, yet eventually worked like a charm. Generating the precise algorithms needed to make the function work perfectly took time and refinement.

The `AddFluid` function proved challenging with difficulties like making the counter for percentage full to go till 100 in case it was completely full. By remembering the different class types for variables this problem was conquered.

Overall this lab project was exciting. The material was interesting and debugging was fun. We had many problems, but in fixing them we deepened our knowledge of the system and strengthened team skills. Despite busy schedules, the effort put in by project members was amazing. Working in a team really helped as group partners always came up with new ideas to improve each person's function. Working on this project was a truly memorable experience because it helped fix small yet but important details missed earlier in CprE 211, and allowed us to run with deeper, unexplored options on the PowerPC.