EE 474 Lab 4
Interrupts and Real Time

Jonathan Lockwood
Ahmed Boustani
David Anderson

INTRODUCTION:

In this lab, we further extend our knowledge and functionality of the Beaglebone through the use of the motors. The the previous lab, we programmed the modules in kernel mode; however, for this lab, we programmed the system using user space C code. The purpose of using user space is that it is easier to implement, easier to debug, and it is more widely used in the real world than kernel mode. In this lab, we used two different kinds of interrupt that execute handler functions. The timer interrupt relies on a present frequency and the general interrupt relies on some user-defined conditions. To make the tank move, we implemented an H-bridge motor driver to handle the different directions each motor moves. We also incorporated four sensors to act as the "eyes" of the tank so it can avoid bumping into other objects. Finally, we learned how to use user Inter-process communication (IPC). IPC allows us to be able to run two processes at once where the first process creates a special file and the second process writes to that file.

OVERVIEW:

Our first task was to incorporate the four distance sensors onto the tank. Each sensor had three wires coming out from it. The red wire connects to the power line, the black wire connects to ground, and the white wire connects to one of the ADC pins on the X69 header. The voltage that the white wire gets starts at 0V when there is nothing in front of the sensor. As the tank moves and objects get closer to the sensor, the voltage in the white wire will increase. The ADC module converts the analog data into digital to be read by the programmer. On our tank, we chose to place two sensors in the front of the tank and two in the back so they are symmetrical around the tank. We use the timer interrupt to read the converted digital data from each sensor by requesting the data at a certain frequency. To make the system real-time, we request the ADC data a certain amount of time per second to be able to see how the tank's distance to an object changes over time. The more samples per second, the more accurate the data.

Once the four sensors of the tank were all set up, we began to implement the H-bridge to the system to be able to control the functionality of the motors. The H-bridge we used was the TB6612FNG. The schematic of the H-bridge can be seen in Figure 1 below. There are 24 pins in the device that connect to 16 physical pins that can be plugged into the breadboard.

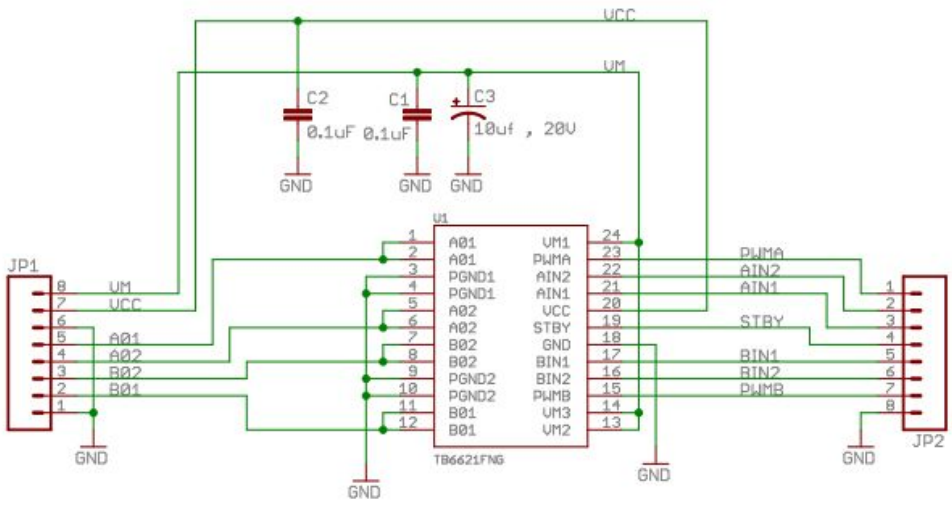

Figure 1: H-bridge schematic

The description of each of the 24 pins can be seen in Table 1 below. The input/output functionality is also specified.

Table 1: H-bridge pin description

Pin NO.	Symbol	I/O	Remarks
1	AO1	O	chA output1
2	AO1		
3	PGND1	—	Power GND 1
4	PGND1		
5	AO2	O	chA output2
6	AO2		
7	BO2	O	chB output2
8	BO2		
9	PGND2	—	Power GND 2
10	PGND2		
11	BO1	O	chB output1
12	BO1		
13	VM2	—	Motor supply(2.5V～13.5V)
14	VM3		
15	PWMB	I	chB PWM input / 200kΩ pull-down at internal
16	BIN2	I	chB input2 / 200kΩ pull-down at internal
17	BIN1	I	chB input1 / 200kΩ pull-down at internal
18	GND	—	Small signal GND
19	STBY	I	"L"=standby / 200kΩ pull-down at internal
20	Vcc	—	Small signal supply (2.7V～5.5V)
21	AIN1	I	chA input1 / 200kΩ pull-down at internal
22	AIN2	I	chA input2 / 200kΩ pull-down at internal
23	PWMA	I	chA PWM input / 200kΩ pull-down at internal
24	VM1	—	Motor supply(2.5V～13.5V)

To determine the control of the motor, pulse width modulation (PWM) signals are used. We used two PWM pins on the Beaglebone (one for each motor). Based on the different combinations of inputs 1 and 2, as well as the PWM and standby signals, the output of the motor can be determined. This controls whether the motor turns clockwise, counter-clockwise, brakes, or coasts. The whole functionality table can be seen in Table 2 below.

Table 2: H-bridge control

Input				Output		
IN1	IN2	PWM	STBY	OUT1	OUT2	Mode
H	H	H/L	H	L	L	Short brake
L	H	H	H	L	H	CCW
		L	H	L	L	Short brake
H	L	H	H	H	L	CW
		L	H	L	L	Short brake
L	L	H	H	OFF (High impedance)		Stop
H/L	H/L	H/L	L	OFF (High impedance)		Standby

The H-bridge allows us to control two different motors at the same time due to the number of pins provided.

FUNCTIONS:

There are two main functions in our lab. The first is the timer interrupt that takes in data samples every 300 nanoseconds from each pair of sensors. We then took the average of each pair and displayed the distance. We then had the command function which gives commands to the H-bridge so send to each motor.

RESULTS:

We were able to successfully hook up the distance sensors and convert the analog data to digital to be used for mobility purposes. At first we had some initial trouble trying to program the system because we were attempting to do it in kernel mode. Once we realized that we needed to program in user-space we were able to correctly convert the data from the sensors. When trying to set up the H-bridge, the first problem we had was that we didn't have enough space to fit it in with the LCD display still attached. We initially thought we could just attach a longer breadboard onto the tank so we could fit everything on. However, we ended up just removing the LCD display to free up much more space on the breadboard. Wiring the H-bridge was a little

confusing at first. We were confused by the fact that there were 24 pins but only 16 physical pins to plug into the breadboard. We were also initially confused by the fact that there were four inputs but the motor control only used two data inputs. We soon figured out that the H-bridge was designed to be able to control two different motors at once (one for motor A and one for motor B). Once the H-bridge was properly wired to the Beaglebone and the two motors, we were able to successfully program the device to move around and change direction.

CONCLUSION:

In conclusion, we were able to successfully use the four distance sensors and make the tank move. The distance sensors each took in distance data and converted it to be displayed on the console when we prompted it. The only problems we had with the distance sensors was initially trying to program it in kernel mode instead of user-space. Once we began programming in user-space we were able to have full functionality of the distance sensors. Setting up the H-bridge was initially a little tricky; however, once we understood the full design of the device, we were able to successfully wire it to the Beaglebone and the motors. We ended up removing the LCD display from our tank because it was taking up too much space and didn't allow enough room for the H-bridge to be fully set up. Once we removed the LCD display, and got the H-bridge fully wired up, we were able to successfully program the device. We were able to make each motor move clockwise, counter-clockwise, and brake. After that we were able to fully integrate the system to move and sense correct distance values to different objects.