Design of a Tetherless Remotely Operated Underwater Vehicle

by

Harish Ramaswamy

A thesis submitted in Partial Fulfillment of the Requirements for the Degree of

MASTER OF APPLIED SCIENCE

In the Department of Mechanical Engineering

We accept this thesis as conforming to the requested standard

[redacted]

Dr. Colin Bradley, Supervisor (Department of Mechanical Engineering)

[redacted]

Dr. Joanne L. Wegner, Member (Department of Mechanical Engineering)

[redacted]

Dr. James S. Collins, Outside Member (Department of Electrical Engineering)

[redacted]

Dr. Vijay K. Bhargava, External Examiner (Department of Electrical Engineering)

University of Victoria

Abstract

The oceans have a wealth of resources useful to mankind, and the unpredictable and hazardous environments make it difficult to tap these resources directly. Remote access to the resources from land could avoid human presence in these difficult environments, and the use of remotely operated underwater vehicles is a viable means. Operating underwater vehicles with tethers requires tethers running 4 kilometers and longer. The storage of tethers with their requirement for large spools and drums increases operating costs. Further, the use of a tether decreases the efficiency of the underwater vehicle, due to the drag imposed by the tether.

The thesis presents a new class of small underwater vehicles that operate without a tether and are controlled remotely from the land. The underwater vehicle is designed for inspection of leakages in pipes, oil rigs, construction and other electromechanical devices. Vehicles that fall into this class are named "Tetherless Remotely Operated Vehicles" (TLROV). The main objective of this thesis was to develop a control and communication system between the TLROV and the operator. Ultrasonic sound signals have been used to develop the communication system. The system has been practically tested for its performance and the results are promising.

Examining Committee:

[redacted]

Dr. Colin Bradley, Supervisor (Department of Mechanical Engineering)

[redacted]

Dr. Joanne L. Wegner, Member (Department of Mechanical Engineering)

[redacted]

Dr. James S. Collins, Outside Member (Department of Electrical Engineering)

[redacted]

Dr. Vijay K. Bhargava, External Examiner (Department of Electrical Engineering)

Table of contents

List of Figures

Acknowledgements

My thanks go first and foremost to my supervisor Dr. Colin Bradley. I am very grateful to him for his support, motivation and guidance throughout my degree program, especially in preparing this manuscript. There are no words that can express my thanks to him for giving me many opportunities in the different fields of engineering over the past two years. These opportunities have helped me understand and work better to achieve my desired goal.

I would like to take this opportunity to thank Mr. Kevin Jones for his patience and for devoting his valuable time to help me understand various concepts. This effort would not have been possible without his help.

I would also like to thank Mr. Jeff Kennedy and many others (Graduate students researchers and faculty) who have supported me through out my degree program.

Chapter 1

1 Introduction

The world's oceans have been almost impenetrable to human exploration until the very recent development of remote underwater vehicles. These underwater vehicles have begun to allow humans to overcome the extreme pressure exerted deep below the surface of the water and the lack of natural light available at these depths. Exploration of the oceans is important for a number of reasons. These include, locating and acquiring of the large quantities of natural resources, inspection of structures, pipelines and sea floor mapping. These requirements lead to the study, design and use of underwater vehicles. Currently manned underwater vehicles are limited in their application by the very high costs, operator fatigue, safety issues and environments under which they can operate. Highly efficient underwater vehicles available today are operated through the use of a physical link (tether) connecting the underwater vehicle and the operator. This greatly limits the extent to which the underwater vehicle can be operated due to fact that the physical link is of a fixed length. Although there are implementations that have overcome this barrier, by using ultrasonic, acoustic and sonar waves as means to establish communication between the underwater vehicle and the operator the cost of these underwater vehicles is too high and not as efficient as compared to the underwater vehicles operated with a tether. "The continuing problems facing further commercial use of such systems include accurate underwater navigation and communication links. Further research in low-cost navigational accuracy and communications links is needed." [17]

Based on the implementation, manned underwater vehicles are broadly classified into two categories- Remotely Operated Vehicles and Autonomously Operated Vehicles. There seems to be a very thin line in differentiating ROV's from AUV's and often they seem to be used interchangeably.

The ROV is controlled by an operator on the surface. A tether is used as a link between the ROV and the operator. The tether is used to both transmit power to the thrusters, lights, cameras, and any other onboard device and receive video signals. A tether allows the operator to enjoy real-time control and the use of a tether does simplify the design of the ROV, but there are inherent drawbacks associated with its use. The tether of any practical length (~100 feet) provides a significant amount of drag that hinders the performance of the ROV. Tethers are also prone to snagging on obstacles or on the ROV itself. Electrical problems can also arise with the use of a tether.

The AUV is self-guiding and therefore requires no tether. It is preprogrammed with a specific mission and then sent on its way. A typical AUV mission might include traveling to a destination, scanning back and forth over a predefined area collecting data or video, and returning to the mother ship. To achieve this, the AUV is equipped with various sensors that provide information on velocity, acceleration, heading, and depth. Using this information the AUV can calculate where it is relative to an initial datum point. This method is called dead reckoning and can be reasonably accurate when water current is negligible. However, this is rarely the case. To cope with position error, induced by heavy currents, the AUV surfaces periodically to take a GPS reading and recalibrate its current position. The period between GPS readings is a function of current strength and required position accuracy. An effective replacement for the tether has been based on ultrasonic communication signals, which are used for both sending control signals and receiving status and sensor data from the ROV. This requires sophisticated data acquisition and navigation systems. AUV's operating with reasonable accuracies are available at very high cost and lot of research is being carried out to enhance the accuracy and reduce cost of AUV's.

1.1 History of Autonomous Underwater Vehicles

AUV's have been in development since the 1960's. Vehicles like the Rebikoff's SEA SPOOK developed in the early 1960's were large, inefficient, expensive, or a combination of all three [5]. Massachusetts Institute of Technology has been a major contributor to the design and development of AUV's since the early 1990's [6]. With the increase in commercial interest, which includes areas like the offshore oil industry, there have been remarkable breakthroughs in the design of AUV's. During 1991 and 1992 a revolutionary new autonomous underwater vehicle (AUV) was developed at the MIT Sea Grant College Program AUV Laboratory [6]. This vehicle, called Odyssey, was designed to provide marine scientists with economical access to the ocean which brought about a huge leap in the development of AUV's [6]. The positive results seen from the test with Odyssey, led to the creation of a second-generation vehicle, Odyssey II [6]. Later, with the advancement in technology, Odyssey II was followed by Odyssey IIb which was essentially an improvement of Odyssey II [6].

With the above history in mind, AUV research continues in hopes of creating more efficient and cost effective underwater vehicles.

1.2 Categories of underwater vehicles

Modern underwater vehicles are classified based on their size, depth capability, onboard horsepower and on whether they are all electrical or electro-hydraulic. The following are the classes of ROV's:

- **Small ROV**

  Commercially available vehicles that fall in this category include the LCROV from Linkquest shown in Figure 1.1 and the Hyball from Hydravision shown in Figure 1.2. These vehicles are capable of operating at depths less than 300 meters and are limited in system power to 5 Hp [5]. The cost ranges from $10000 to $100000 dollars. These vehicles are used for inspection and observation of leakage in pipes and oil rigs. They use a tether to communicate with the operator, and they are electrically powered by batteries.

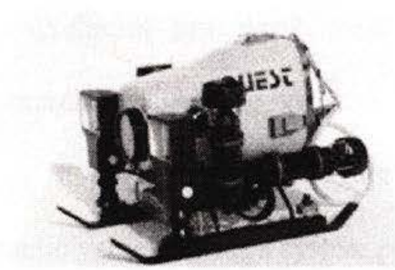


Figure 1.1 Linkquest LCROV

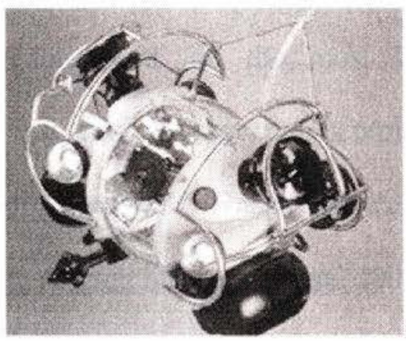


Figure 1.2 Hydrovision Hyball

- **High capacity electric ROV**

  The Perry Tritech Voyager shown in Figure 1.3 and the MaxRover shown in Figure 1.4 are high capacity electric underwater vehicles. Unlike the small vehicles which are low in cost, vehicles in this category reach the $500000 mark. They are capable of working to greater depths (6000 meters) and are limited in the total system power to 10 Hp [5]. Like the small vehicles, the high capacity electric underwater vehicles are not capable of doing heavy work due to their lack of electro- hydraulic manipulators. These vehicles are interfaced to computers with sophisticated communication and imaging systems. They use tethers to communicate with the operator, and they are electrically powered by batteries.

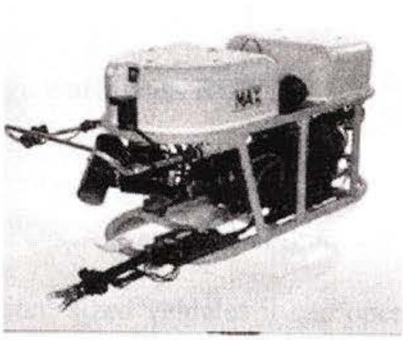


Figure 1.3 Perry Tritech Voyager

Figure 1.4 MaxRover

- **Medium sized ROV**

These vehicles are work class vehicles and are the most widely used. Commercially available vehicles in this class include the Scorpio and Viper shown in Figure 1.5 and Figure 1.6 respectively. These vehicles are electro hydraulic vehicles with a system power ranging between 20 and 100 hp [5]. They range in weight from 1000 to 2200 kg with typical payload capacities in the 100 to 200 kg range. This class was developed to perform work, carrying one or two manipulators, in high current conditions and to work at depths of 1000 meters. Typical tasks (work) for this class are drilling support, construction support and pipeline inspection.

Figure 1.5 Scorpio

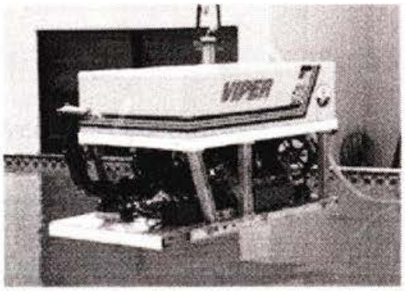


Figure 1.6 Viper

- **Large work class ROV**

  There are fewer underwater vehicles in this category due to lower demand. The TRITON XL, shown in Figure 1.7 is available commercially and compared to the medium sized vehicles it can operate in deepwater (2500 meters) with a system ranging from 100 to 250 hp [5]. These vehicles are specifically designed to perform heavy work over prolonged periods of time without diver intervention. The only difference with the ultra-deep vehicles is the added cost due to the requirement of additional power and tools for heavy work.

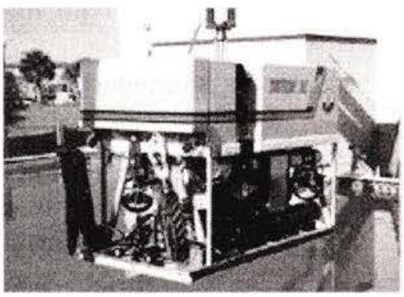


Figure 1.7 TRITON XL

The use of a tether as a link between the operator and the underwater vehicle for transmitting control signals and receiving sensor information imposes severe limitations on the control of an underwater vehicle. The reasons for these limitations are:

- The tether, which is usually circular in its shape, becomes a source of drag for the vehicle in motion.
- The depth to which the underwater vehicle can be operated is limited by the length of the tether.
- The operational cost of the underwater vehicles with tethers is high. The cost of operation per day ranges between \$50000 and \$100000. The use of tethers requires spools and drums for storage of tethers that can run up to 4 km or even longer. This leads to a requirement for large deck space and hence for larger boats.

This thesis focuses on the design of a new class of small underwater vehicles that operate without tethers and receive control signals from an operator on land for navigation. Underwater vehicles that fall into this class are named "Tetherless Remotely Operated Vehicles (TLROV)". The TLROV differ from the autonomous underwater vehicles (AUV) in their method of operation. AUVs are widely used in sea floor mapping and have a preprogrammed path for their navigation. A TLROV is better than an AUV for deep water inspections during which, the operator might want to have better control over the vehicle. The chapters that follow discuss and present various aspects involved in TLROV design.

1.3 Outline of the thesis

The purpose of this work is to develop an ultrasonic communication system to retrieve sensor information of scientific data and to control a remotely operated vehicle. This project is also to serve as a test bed for designing a navigation system and an image acquisition system for future work. The chapters that follow address:

- Factors influencing design of TLROV
- Design of a software / hardware interface to control and monitor the underwater vehicle
- Use of ultrasonic transducers to design a communication system based on the RS232 protocol
- Error correction and synchronization in software to achieve reliability in communication of control signals and sensor data

Chapter 2

2 Mechanical design

The TLROV is a small electrically operated vehicle designed to operate in low current conditions (less than 1 knot) and to a depth of 200 meters. The size is selected to accommodate the sensors, batteries and thruster control electronics. Figure 2.1 shows the exploded view of the TLROV. It is used as a shallow water inspection (tunnel, ship hull, pipes, oil and gas) vehicle and has a top speed of 1 knot and an endurance of 10 hours at the top speed. There are several factors that influence TLROV design and the following sections discuss each of the factors considered in detail.

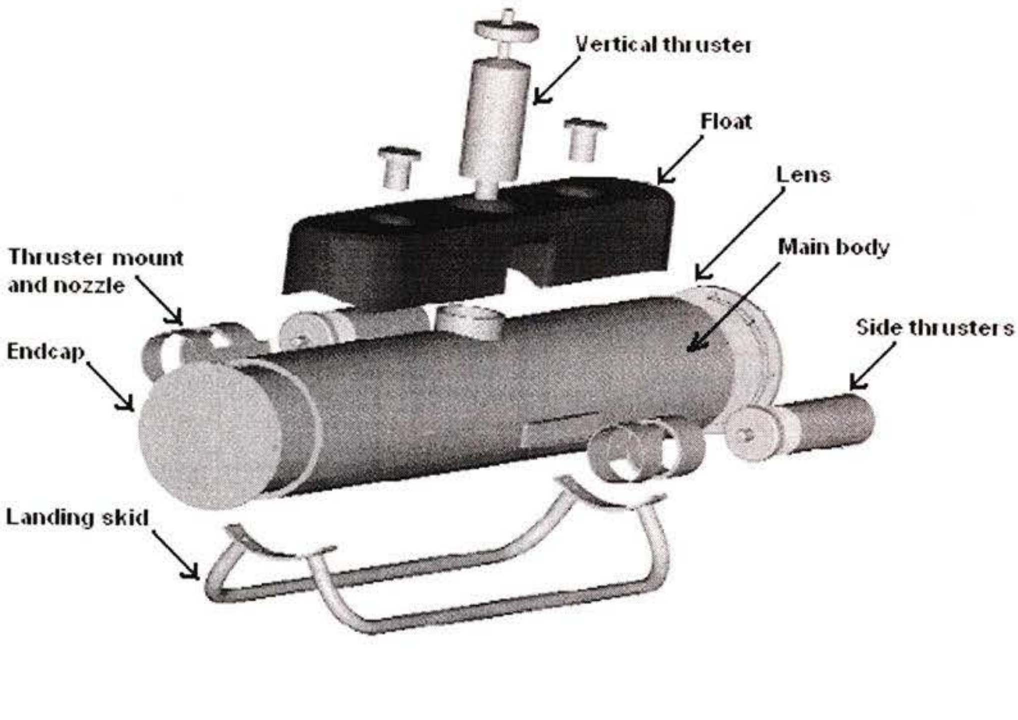


Figure 2.1 TLROV exploded view

2.1 Vehicle description

The vehicle weighs 6.5 kg and measures 110 mm meters wide and 500 mm long. The hull is constructed of 6 mm thick 6061 aluminum and forms the pressure housing containing electronics and batteries (see Figure 2.2). Aluminum has a low density and weighs only one third the weight of steel and hence provides a balance in vehicle design. The low weight of aluminum combined with its corrosion resistance and non-combustible nature has made it an appropriate building material for the TLROV. The hull is anodized to protect it, by making the surface much harder than natural aluminum. Aluminum oxide is grown out of the surface during anodizing, and the extremely hard and the porous nature of the anodized layer allows it to be dyed in any color that is required for good finish. In operation the TLROV will impact the bottom of a pool, rocks, or other abrasive and uneven surfaces. The skid bars offer a steady and robust landing platform for the TLROV.

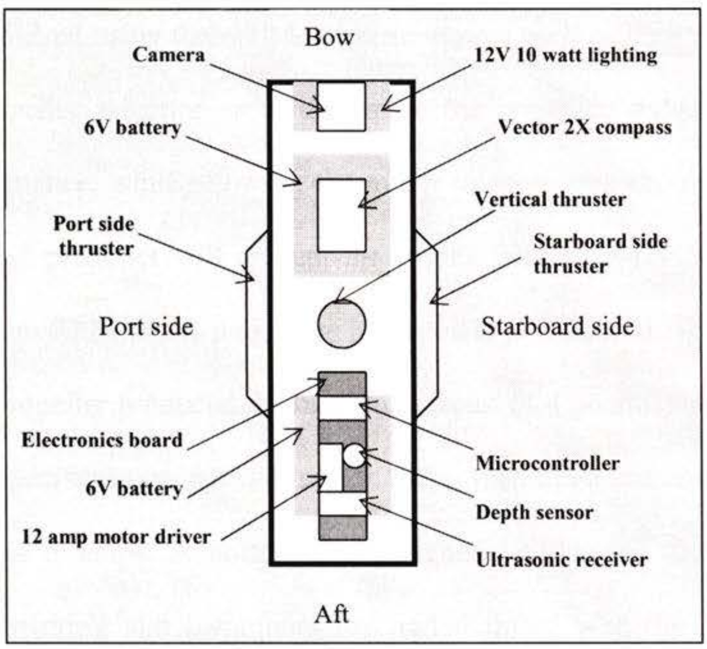


Figure 2.2 TLROV component layout view

Figure 2.2 shows the compact layout design used in the TLROV to distribute component weights equally over the entire length of the hull. A compass module used to determine heading is placed on the bow side isolating it from the magnetic fields due to electronics placed at the aft. A single CCD camera with a halogen light is mounted on the bow of the TLROV and enclosed in the pressure housing with a plexiglass and an o-ring. The aft section of the TLROV houses the batteries and electronics and is sealed in the pressure housing with an end-cap and o-ring. O-Rings are torus-shaped objects made from elastomeric compounds such as natural or synthetic rubber, and are used to seal mechanical parts against fluid movement. O-Rings perform their sealing action by deforming to take the shape of their cavity, after being oversized to guarantee an interference fit. A depth sensor, microcontroller, motor-driver and an ultrasonic transceiver are placed on a single circuit board at the aft over the battery (see Figure 2.2).

2.2 Propulsion system

Propulsion is achieved using three 12 V electric motors each turning a 50 mm propeller. The goal in propeller selection is to determine the propeller style and size that will maximize performance, while allowing the motor to operate in the recommended RPM range. The correct propeller will prevent the engine from over-revving; yet allow it to reach the minimum RPM where maximum horsepower is produced. Testing the motors, a 50 mm cupped propeller produced the maximum thrust (1.4 pounds) at 2200 RPM which is within the specifications of the motor. The maximum current draw with this configuration was 3 amps. A nozzle was designed to increase the efficiency of the thrusters by redirecting and combining lost radial thrust with the axial thrust. As a result, this new combination provided 1.4 pounds of thrust at 12 V, but with a current draw of only 2.5 amps. Figure 2.3 shows the thruster layout system which is used to control the TLROV. Two thrusters located on the starboard side and port side respectively work in tandem to achieve motion along the x and y direction (yaw), while the third thruster is used to provide vertical thrust (pitch). The three thrusters are centered horizontally along the x-axis and z-axis, to distribute the weight evenly and increase stability of the TLROV.

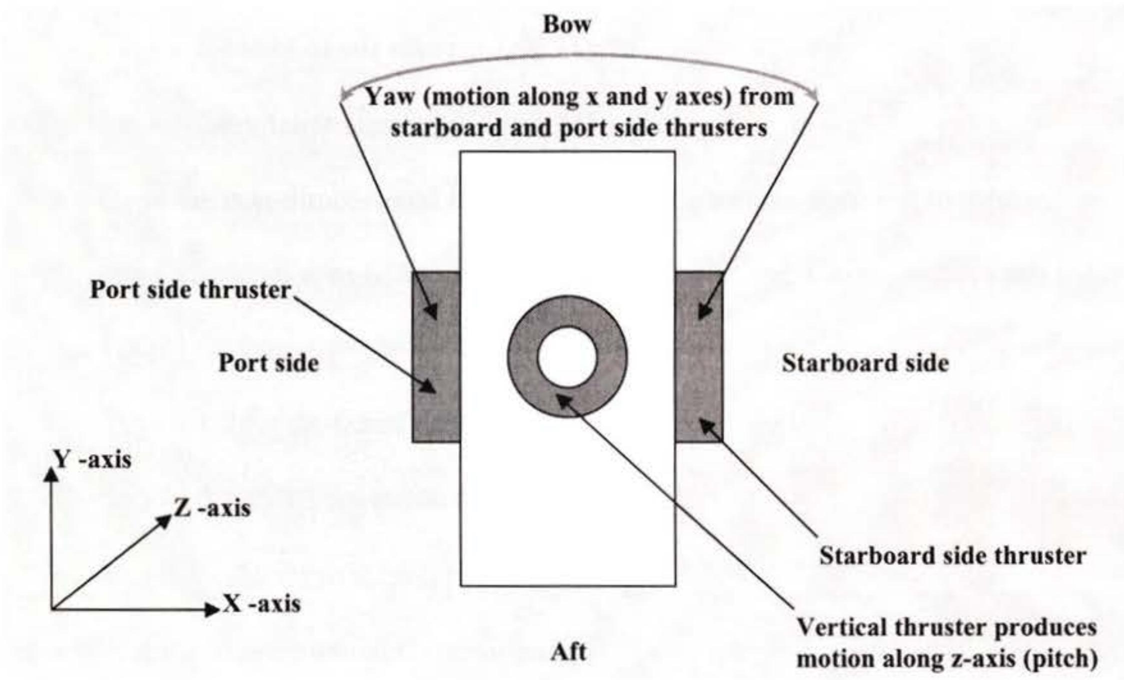


Figure 2.3 Propulsion system

2.3 Drag

Drag is a mechanical force and its generation requires:

- Interaction and contact of a solid body with a fluid (liquid or gas)
- A difference in velocity between the solid object and the fluid.

Drag acts in a direction that opposes the motion. An aerodynamic design reduces drag to a great extent resulting in higher speeds and efficiency. In tethered underwater vehicles, the tether is responsible for very high drag due to its tubular dimensions. The speed that a vehicle can attain depends on the power that is available and the drag imposed by the vehicle and the tether. Equation 1 [5], is used to calculate drag on the TLROV.

$$\text{Drag} = \frac{1}{2} \times S \times A \times V \times 2 \times C_d \tag{1}$$

Where,

- S is the density of sea water / gravitational acceleration
  - Density of sea water = 1,025 kg/m^3
  - Gravitational acceleration = 9.8 m/sec^2
- C_d is the non-dimensional drag coefficient. C_d ranges from 0.8 to 1 based on the cross sectional area of the vehicle [5]. For the tether, C_d ranges between 0.1 and 1.2 [5].
  - 1.2 for un-faired cables [5].
  - 0.1 to 0.2 for faired cables [5].
- V is the velocity in meters per second.
- A is the area on which C_d is normalized.
  - For the TLROV, the area under consideration is usually the cross sectional area of the front
  - For the tether the area under consideration is the diameter of the cable in meters times one 12th the length perpendicular to the flow [5].

The total drag on a vehicle is the sum of the drag due to the vehicle geometry and the drag due to the tether. On calculation of the respective drags it is seen that the drag due to the geometry of the vehicle is lower than the drag due to the tether. Thus, it is clear that drag on TLROV will be much less than that for a similarly sized ROV. Drag calculation for the TLROV is shown in Appendix.

2.4 Buoyancy and stability

Buoyancy is typically described as a measure of an object's tendency to float in a given fluid. The buoyant force (F_b) is described as the upward force exerted on an object when submerged in a fluid. This force is equal to the mass of the fluid displaced multiplied by the gravitational field. The mass of the fluid displaced is equal to its density (ρ) multiplied by the displaced volume (V). For a TLROV to remain stable (neutrally buoyant) while submerged in water, the weight of the displaced water must equal the weight of the TLROV. Equation (2) is used to calculate the buoyant force acting on the TLROV and to distribute the weight so as to make the system stable.

$$F_b = \rho \times V \qquad (2)$$

Stability requires that, when disturbed slightly from its equilibrium position, the TLROV will tend to return to the equilibrium position. The stability of the TLROV when submerged in water is affected by the center of gravity and buoyancy. With reference to Figure 2.3 (B represents center of buoyancy and G represents center of gravity):

- The center of buoyancy and the center of gravity of the vessel must lie on a common vertical line. If they lie on different vertical lines, this will produce a torque, rotating the vessel [5].
- The center of gravity of the vessel must be below the center of buoyancy. Dense objects sink in fluids with lower densities. If the center of gravity of the vessel is

below its center of buoyancy, then the majority of the weight rests below the center of buoyancy. This implies that the volume of the vessel below this point is denser than the same volume of displaced water, and the volume above this level is less dense than the same volume of displaced water. Thus, the lower portion of the vessel will tend to sink while the upper portion will tend to rise, preventing the vessel from rolling [5].

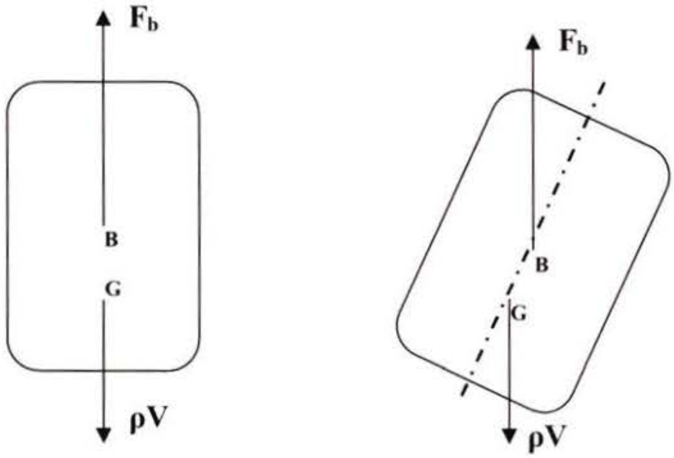


Figure 2.4 Center of buoyancy and center of gravity

If the weight of the TLROV is less than the weight of water it displaces then, it is said to be positively buoyant and the TLROV will rise to the water surface. On the other hand, if the weight of the TLROV is greater than the weight of water it displaces it is negatively buoyant and the TLROV will tend to sink to the bottom.

The TLROV is designed to be positively buoyant for the following reasons:

- On failure of either the control system or thrusters it will return to the surface.
- With the TLROV positively buoyant, it provides for near bottom maneuvering without thrusting up, forcing water down, thus stirring up sediment.

A float made of 6.8 kg (6.8 kg/0.02832 m_3) polyurethane foam coated with a rubberized vinyl coating is used to design the TLROV positively buoyant. To allow for the circuit weight and any additional electronics that may be added in the future, 1.2 kg of excess buoyancy was added and this equates to a total float volume of approximately 100 in^3. Buoyancy calculations showing the TLROV to be positively buoyant, is shown in Appendix.

2.5 Power system

The TLROV requires a reliable source of power to run the thrusters, electronic devices and sensors. With the absence of a tether to deliver power to the TLROV form the land, batteries that are able to last for a long time and charge quickly are required. There has been a great amount of research done on fuel cells as a source of low cost power, and they could be a viable solution in future. Currently the fuel cells available are, large in size and with limited power capabilities. Capabilities such as autonomous recharging are being implemented, and thus, the underwater vehicle is capable of automatically homing in and docking onto a recharging system when the power level reduces to a specified level.

The power source is selected based on the speed that the TLROV needs to attain and the duration of operation. A wide range of rechargeable batteries are available and they include lithium primary, lead acid, Ni-Cad, Ni-Zn, Li-ion, lithium polymer and silver zinc [18]. Of these Ni-Cad and lead acid batteries are common and widely used. The lead acid batteries tend to leak hydrogen. The hydrogen is developed during the final stages of charging and tend to leak out over period of days. Ni-cad batteries offer only a very small added advantage over lead acid batteries in terms of energy density but do not leak unless they are overcharged. Energy density is defined as the energy obtainable per unit weight (gravimetric energy density) or per unit volume (volumetric energy density). The above factor and the low cost have lead to the use of lead acid batteries on the TLROV. Two six volt batteries are connected in series to provide a total system voltage of 12 volts. Each of the six volt batteries are rated at 3.4Ah and together should run the TLROV at 1 knot for 10 hours.

Chapter 3

3 TLROV electronic hardware

Efficient TLROV thruster control for guidance and positioning requires sensor feedback to the operator. Therefore, the TLROV is equipped with a suite of sensors to provide relevant feedback information. The sensors include a depth sensor, compass and CCD camera for calculating depth and heading and real time images respectively.

Feedback and control signals for the thruster have to be communicated in real time. The above goals are met in the TLROV design using electronics. The electronics used include sensors, microcontrollers and interface circuits. Sensors are electromechanical devices that replicate human senses for identification, measurement and feel. Microcontrollers act as processing and control unit to coordinate the actions between external devices and the operator. Electronic interface circuits are required to interface external devices such as the sensors and the thrusters to the microcontroller. The use of sensors to replace human senses, although an interesting solution, may not produce accurate results if used incorrectly. Sensors are designed for specific applications and operational environments. Their implementation usually requires a good working knowledge and circuit designs to interface with the microcontroller. This can be a challenge, to achieve accurate results. The electronic components used on the TLROV include:

- Depth sensor for feedback of TLROV depth
- Compass module for feedback of heading
- Microcontroller for control and processing

- H-Bridge interface circuit to control thruster from the microcontroller.
- CCD sensor to view underwater images in real time

The combination of the above hardware enables a low cost, small sized and remotely operated underwater vehicle which can perform effectively in a range of underwater environments.

This chapter discusses the performance and the working principle of each sensor and also presents the electronic circuits implemented to interface the thrusters and the sensors with the microcontroller.

3.1 Microcontroller selection

Microcontrollers, as the name suggests, are small semiconductor devices. They are like single chip computers that are often embedded into other systems to function as a processing and controlling unit. Microcontrollers are used in a wide variety of applications such as automation, data monitoring and data recording. The connection of sensors and thruster to a microcontroller requires interface circuitry that is responsible for providing:

- A common protocol for communication between the external device and the microcontroller.
- Isolation of the microcontroller from voltage/current surges, since microcontrollers operate at very low voltages and with the capability of sinking and sourcing currents only in the order of milliamps.

Several types of microcontrollers were initially considered for use in the TLROV. After careful evaluation, the Basic Stamp and Motorola 68HC11 were incorporated into the design. These microcontrollers were evaluated based on their functionality, response and ease of development. The tests indicated that the Basic Stamp was limited by its processing speed and response time compared to the Motorola 68HC11. The Basic Stamp has several built in high level commands for pulse width modulation (pwm) and serial input/output to communicate with external devices. The RS-232 protocol is also available on the I/O ports, which adds flexibility for circuit design. A major disadvantage of the Basic Stamp is the lack of hardware interrupt. Without this feature, the Basic Stamp is limited for applications where communication between one or more microcontrollers is a requirement.

Table 1 summarizes the evaluation results between the different microcontrollers. The Motorola microcontrollers are available in a wide variety, each suitable for different applications. The 68HC11 are 8 bit micro-controllers, i.e. they process 8 bits of data at a time. The 68HC12 and 68HC16 are 16 bit micro-controllers and are useful for high accuracy applications. The 16 bit microcontrollers provide additional features such as, digital signal processing (DSP) capabilities and large data and program space (1 MB). The design of TLROV involves the use of a compass and depth sensor for measurements and the accuracy of the devices is limited to 8 bits. Furthermore, the program space required for the implementation is less than 256 Kb. The use of either the 68HC12 or 68HC16 increases the cost for the TLROV design with no added benefits.

The Motorola 68HC11 with interrupt capability meets the TLROV design requirement, keeping cost and development time within acceptable limits, and hence it has been the choice in the final implementation.

Company	Chip	Pros	Cons
Motorola	68HC11	• Features: PWM, timer, SPI, A/D, D/A, RS232, Output Compares, Input Capture, Interrupts • Proven development environment already available	• Poor A/D, D/A resolution • RS232 available on only Port D
	68HC12	• Builds on HC11 (source compatible) • 16-bit • Low power and voltage • 20-bit ALU • Fast (8 MHz) • Flash EEPROM • Debug instructions • Large address space	• Expensive development environment would need to be acquired
	68HC16	• 16-bit • Similar to HC11 instruction set • DSP opcodes • Very large (1 MB) data and program spaces	• Not as familiar as HC11
Parallax	BASIC Stamp	• Several built-in high level functions (PWM, serial I/O) • RS232 available on any of the ports • Speeds development	• Limited program memory (2 kB) • Slow (2k-4k ops/s) • Expensive ($35 chip)

Table 1: Comparison between microcontrollers

3.2 Thruster electronics

Three thrusters are mounted on the TLROV. Two of the three thrusters work in tandem to provide for thrust and direction along the horizontal axis. The third thruster is used to provide vertical thrust. For effectively controlling the TLROV, both the speed and direction of the thrusters must be variable. Pulse width modulation (PWM) is used to change the speed of the thrusters. For change in direction of rotation of the thrusters, polarity of voltage supplied must be changed. Electronically, this change in polarity is achieved by using an H-Bridge circuit.

3.2.1 Method of operation

The H-Bridge circuit (Figure 3.1) consists of four transistors operated as switches. By selectively enabling each of the four transistors the motor is either operated in reverse or forward mode.

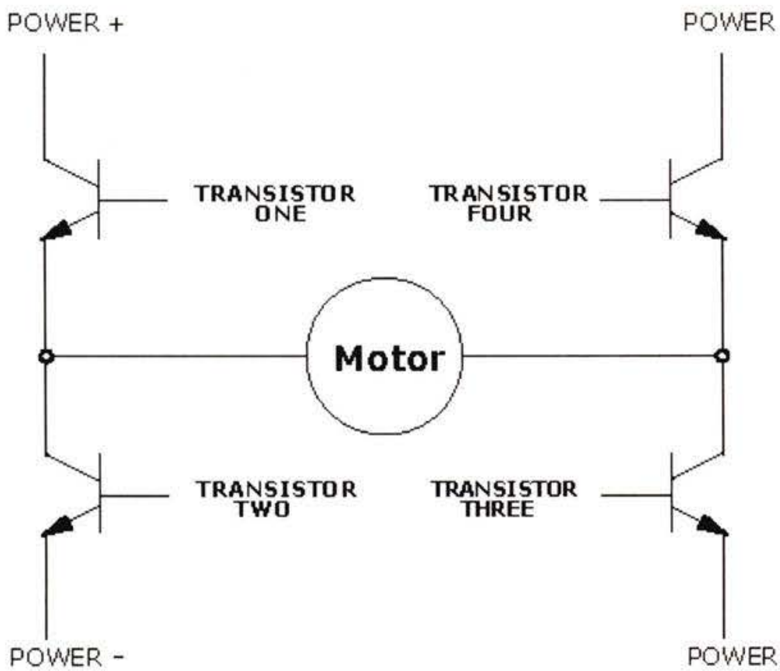


Figure 3.1 Thruster H-Bridge circuit

Enabling the opposite pair of transistors (transistor one and transistor three), allows current to flow through the motor in the forward direction. The other pair (transistor two and transistor four) is disabled at this point. Similarly enabling transistor two and transistor four, allows current to flow through the motor in the reverse direction. Transistor one and transistor three are disabled at this time. Enabling transistor one and two or transistor three and four simultaneously would cause current to flow from 'Power +' to 'Power –' through the transistors, and not the motors, at the maximum current-handling capacity of either the power supply or the transistors. This usually results in failure of the H-Bridge. With reference to Figure 1, Table 2 shows transistors that need switched on to operate the thruster in forward and reverse modes.

Direction	**Transistor 1**	**Transistor 2**	**Transistor 3**	**Transistor 4**
Forward	On	Off	On	Off
Reverse	Off	On	Off	On

Table 2 Transistor switching sequence

3.2.2 Thruster interface electronics

In the initial TLROV design an H-Bridge circuit was implemented using the Basic Stamp microcontroller and the Motor Mind H-Bridge circuit. The Basic Stamp has a built-in instruction set for PWM that produces the PWM output. The PWM output produced by the basic stamp is not very accurate as it uses the time required to charge and discharge a capacitor as a reference to turn on or off a particular pin. This method of using the capacitor charge and discharge as a time of reference is not very accurate. To generate accurate PWM signals on the basic stamps it is necessary to use an external PWM circuitry thus making the overall circuitry complex. The Motor Mind H-Bridge drivers used were capable of providing continuous currents of 1 amp which is well below the current required for the thrusters to run at full speed.

In the final design, the H-Bridge circuit used is a TPIC0107 from Texas Instruments. This semiconductor device is capable of running thrusters with a constant current draw of up to 3 amps and peak current draw of up to 5 amps. Figure 3.2 shows the H-Bridge interfaced to the microcontroller.

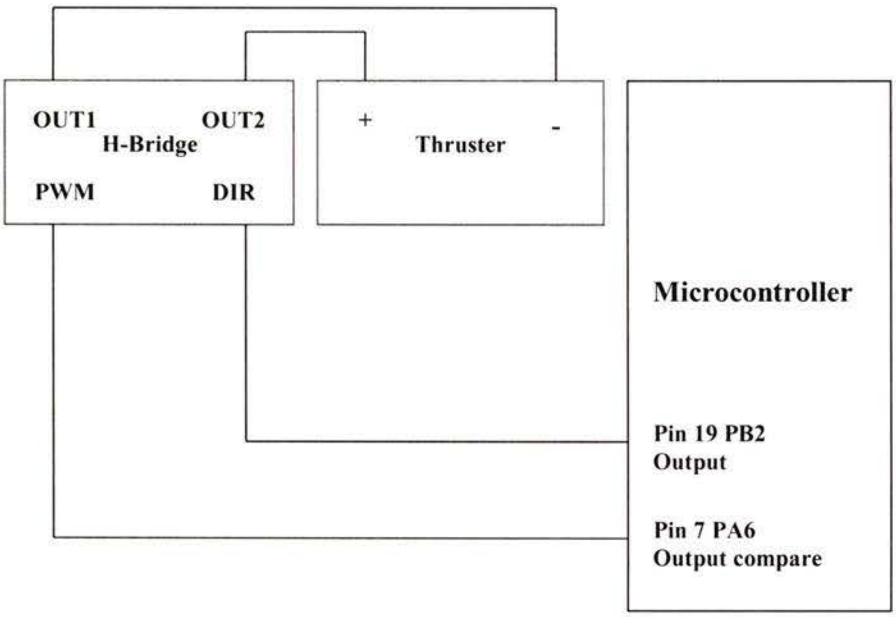


Figure 3.2 H-Bridge interfaced to the microcontroller

With reference to Figure 2 it can be seen that, two pins on the microcontroller are used to provide for PWM and direction control. The PWM pin on H-Bridge circuit is connected to PA6 (Pin 7) of micro controller. PA6 is capable of performing output compares which automatically sets Pin 7 high or low at specified time intervals. The time interval at which Pin 7 should be switched on or off is written to the output compare register of port A in software. This effectively results in pulse width modulated signal, which is fed to the PWM pin of the H-Bridge. For direction control the DIR pin on the H-Bridge is connected to Pin19 on the micro controller. The direction of the rotation of thruster is changed by setting this pin high or low. Three such circuits are used in the final design to control three thrusters individually for an effective thruster control system.

3.3 Compass module

The Vector 2X compass module is used on the TLROV to determine heading. Heading information provides a sense for direction to maneuver the TLROV. The compass module is based on the magneto inductive (MI) sensor technology [8]. This technology is patented by Precision Navigation. The MI technology is superior to that of the flux gate and magneto resistive technologies. The MI sensors show a change in the inductive value for varied magnetic field strengths. This variable inductance property has been used to create a very sensitive magnetic sensor that is low in cost and power [8]. These advantages have made MI sensors the choice for use in a variety of applications including compasses, automobiles, Polaris jet skis, boats, and a new generation of Timex compass watches.

3.3.1 Method of operation

The MI sensors, that employ a single solenoid winding for each axis, consume less power than conventional fluxgate or magneto-resistive technologies. With reference to Figure 3.3, the sensor coil serves as the inductive element with its effective inductance being influenced by the ambient magnetic field component running parallel to the coil axis. The frequency of the square wave changes with applied magnetic field, and the output from the opamp is fed directly into a microcontroller.

Three models of MI sensors are available: SEN-T, SEN-W, and SEN-M. SEN-T MI sensors are designed for applications that require high resolution [8]. SEN-W and SEN-M MI sensors have a large dynamic range and are well suited for environments with large magnetic fields. SEN-M MI sensors are miniaturized, making them ideal for limited-size applications where a large dynamic range is needed.

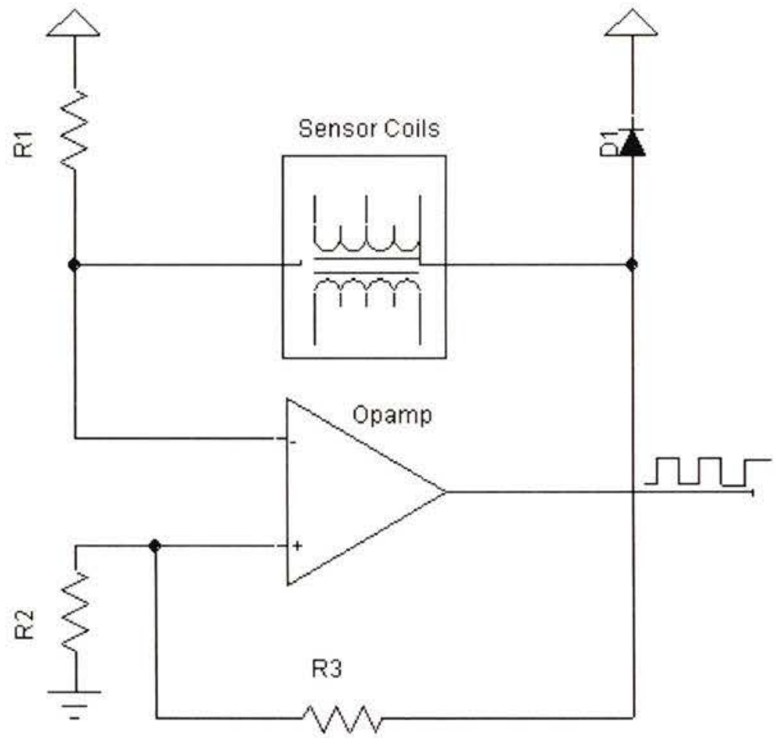


Figure 3.3 Compass interface electronics

The compass module supports two synchronous communication standards: serial peripheral interface (Motorola) and micro wire (National) which facilitates for interfacing to different microcontrollers. The compass module can be operated either as a slave or a master. In the TLROV, the compass module is interfaced to the microcontroller through the serial peripheral interface, and the compass module is configured as a slave. Since the compass module is configured as a slave, the master (microcontroller) is responsible for providing a clock with a frequency of less than 1 MHz for clocking out the data from the compass. The compass module outputs a 2 byte (16 bit) heading in sixteen complete clock cycles on the leading edge of the clock. It is important to note that the clocking provided by the master should be acceptable by the slave in terms of polarity and frequency. The heading value has an accuracy of +/- 1 degree.

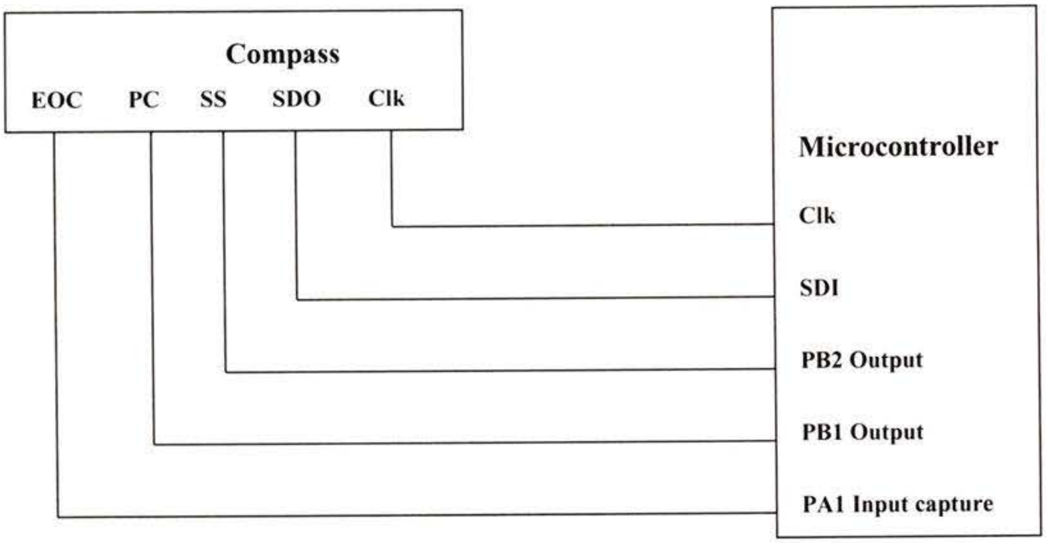


Figure 3.4 Compass module interfaced with the microcontroller

With reference to Figure 3.4, for the compass to output data, it has to be initiated to calculate the heading. For this both PC and SS on the compass module are pulled low and held low for about 10 ms. Once the EOC pin on the compass module goes high the data is available from the SDO pin on the compass module for 16 complete clock cycles. On completion of acquiring data from the compass module, both PC and SS are brought high and the compass module goes into sleep mode until the next poll. In sleep mode, the compass module consumes less power in the order of 2 milliamps. This method of acquiring data from the compass, by polling for data although not very efficient saves much power, which is an important factor to consider in the design of a TLROV.

3.4 Depth sensor

The TLROV is designed to operate at depths up to 200 meters. Operating the TLROV at depths greater than 200 meters would result in failure due to weakening of the thruster seals. Thus, it is important to know the current operating depth for scientific measurements and providing operator feedback.

The pressure transducer used to measure depth is the Motorola MPX5700, a piezo-resistive monolithic silicon pressure sensor. The pressure sensor is designed for a wide range of applications but particularly those incorporating a microcontroller or microprocessor with analog to digital inputs. This single element transducer combines advanced micromachining techniques to provide an accurate, high level analog output signal that is proportional to the applied pressure. The pressure range for the sensor is between 0 to 700 kpa. Thus the pressure sensor can be used to measure depths of up to 70 meters with an accuracy of + / - 2.5 %.

3.4.1 Method of operation

Pressure sensors operate by the detection of a physical force on a diaphragm which rises due to pressure. The force is directed from the high pressure region to the low-pressure region. To achieve maximum deflection for a small pressure force, the diaphragm is maintained to be as thin as possible, which is dictated by the manufacturing limits. A pressure change causes the diaphragm to flex, inducing a stress or strain in the diaphragm and the resistors that are etched onto the silicon diaphragm. The resistors value change in proportion to the applied stress and produce an electrical output.

3.4.2 Depth sensor interface electronics

The sensor outputs an analog voltage between 0 and 5 volts. The analog voltage is input to PE0 (pin 22) on the microcontroller. Pin 22 has a built in 8 bit analog to digital converter, and hence, the digital value ranges between 0 and 255. The analog to digital converter on the 68HC11 microcontroller is an 8 channel 8 bit successive approximation converter. A successive approximation ADC employs a digital-to-analog converter (DAC) and a single comparator. It effectively makes a binomial search by beginning with an output of zero. It provisionally sets each bit of the DAC, beginning with the most significant bit. The search compares the output of the DAC to the voltage being measured. If setting a bit to one, causes the DAC output to rise above the input voltage, that bit is set to zero. Conversion rates over 200 kHz are common. Successive approximation is relatively inexpensive to implement. The final digital value from the DAC representing pressure is calibrated and used to determine depth.

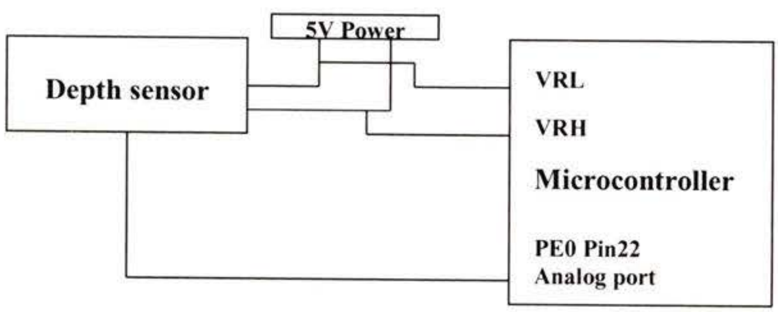


Figure 3.5 Depth sensor interfaced to the microcontroller

3.5 Electronics for controlling light intensity

With proper lighting objects that are few meters from a camera can be clearly imaged in ocean water, but, unlike air, even in the best ocean water the clarity is sharply reduced for distances as small as 5 to 10 m. Since the amount of light available at any instant, constantly changes, it is often required to adjust intensity of light for good quality images. The TLROV is equipped with a 12 volt 10 watt halogen light for clear under water images. The light is switched on or off from the graphical user interface on a laptop. The light intensity can also be varied from the graphical user interface using pulse width modulation. Since the halogen light operates at 12 volts and 10 watts of power it draws 0.88 amps of current for its operation. The microcontroller used to generate the pulse width modulated signal, to change the intensity of light, is capable of sourcing or sinking currents of up to 20 milliamps, which is much less than the current required by the halogen lamp used. To overcome this limitation a transistor or mosfet is used as a switch. Transistors are current operated devices; whereas; mosfets are voltage operated. Compared to the transistors, mosfets are semiconductors with high input impedance. This is due to the fact that the mosfets have a dielectric material at the gate which impedes the current draw at the gate. Thus for very small input currents they are capable of providing very high output currents with reduced heat generation when compared to transistors. The above factors have lead to the use of a mosfet as an interface between the halogen lamp and the microcontroller for varying light intensity. The mosfet used on the TLROV is IRFZ40 which is capable of operating at a maximum drain voltage of 50 volts and max drain source current of 50 amps.

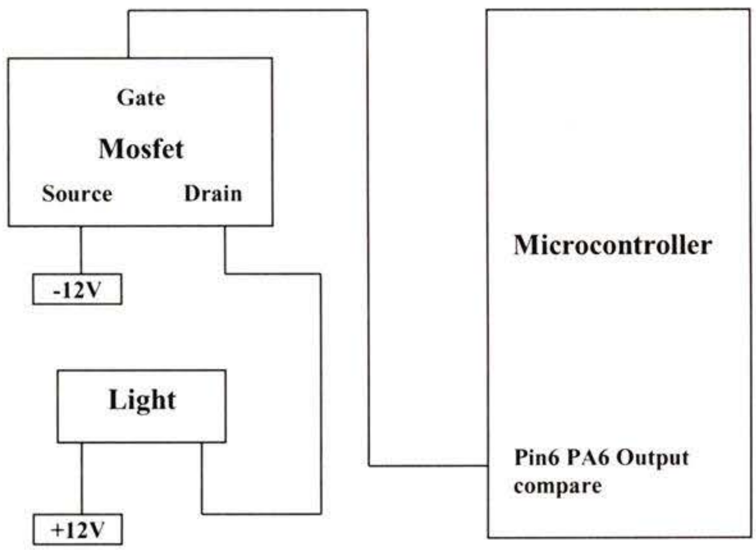


Figure 3.6 Lighting interface circuit

With the circuit in Figure 3.6, it is easy to turn the light on and off effectively from software running on the microcontroller. The circuit also allows the operator to control the intensity of light using pulse width modulation. The gate of the mosfet is connected to PA7 (Pin 6) of microcontroller. PA7 is capable of performing output compares and thus, turning Pin 6 high and low at specified time intervals. The time interval at which Pin 6 should be switched on or off is written on the output compare register of port A. This effectively results in pulse width modulation and changes the intensity of light. The use of pulse width modulation to control light reduces the power consumed by the light and thus provides a longer battery life.

3.6 Real-time image capture

A camera system is required to assist in maneuvering the TLROV and performing inspections. There is no single system available that can be used for all underwater applications. For the TLROV, the camera used to capture images from under water in real time is a single chip color CCD sensor. The sensor outputs an analog signal at 30 frames per second which is the NTSC standard for analog video signals. The analog signal is fed into the USB frame grabber connected to a laptop. The frame grabber uses the entire bandwidth available on the USB channel for its transmission. The USB 1.0 standard has transfer rates of up to 12 Mbps and the USB 2.0 standard has transfer rates of up to 480 Mbps. With an image capture resolution of 640/480 and three bytes used to represent a single pixel value, a single frame would consist of 900 kilo bytes of data. For 30 frames per second the size of data is 26 Mb. Clearly, the band width on the USB 1.0 standard is not sufficient to transfer images in real time at 30 frames per second. The USB 2.0 or the Fire wire standard which transfers data at the rate of 1600 Mbps is a better choice for real time video transfers. For this research project the video signal is transferred to the laptop through an USB 1.0 interface, and a laptop with a fire wire port would be an ideal choice. Due to the limited bandwidth available with the ultrasonic communication system, capture of video in real time is made available only with the use of a tether.

Chapter 4

4 TLROV software development

The TLROV employs two software systems for remote operation; namely, embedded software and the laptop GUI application. The software systems perform diverse tasks such as retrieving sensor data (depth and heading), transmitting thruster control signals and providing a graphical user interface (GUI). The software routines implemented on the microcontroller and the laptop must be in synchronization with each other to achieve real time control. This chapter presents the software developed for control and communication with the TLROV. The software development process was subdivided into the following areas:

- Embedded software
- Graphical user interface (GUI)

4.1 Low level system control - Embedded software

A microcontroller programmed with a reduced instruction set (RISC) is termed embedded software. The TLROV embedded software instructs the microcontroller to communicate with sensors and electronic devices to perform the following tasks:

- Retrieve data from the pressure sensor, compass and battery level indicator.
- Control thruster speed and direction via the H-Bridge circuit.
- Communicate data to and from the operator employing a GUI and RS-232 data transmission protocol

The embedded programming environment for the Motorola 68HC11 microcontroller is a combination of Assembly and C programming languages. The use of assembly language provides optimized code as opposed to programs written in C. With limited memory available on a microcontroller, sections of the program which require memory optimization are written in assembly language.

4.2 High level system control - GUI

The GUI provides an intuitive interface to control and monitor the TLROV. The information displayed on the GUI is (see Figure 1):

- A joystick simulation for thruster control (bottom right).
- Depth and heading information (top right).
- A pulse width modulated signal representation which provides a sense of thruster speed to the operator (bottom left).
- Video images from the CCD camera mounted on the TLROV (top left).

The GUI and its associated software also act as a link between the embedded low level software and the TLROV operator. The GUI interacts with the microcontroller to perform system verification on initialization, receive sensor information and communicate thruster control information.

The programming environment for the GUI is a combination of Visual Basic and Visual C++ programming languages. Visual Basic is a user friendly programming environment which reduces development time but does not provide support for accessing hardware devices. Visual C++ is used to access the hardware devices through an application programming interface and to represent the data in a format suitable for access in Visual Basic. The following sections discuss each of the above topics in detail.

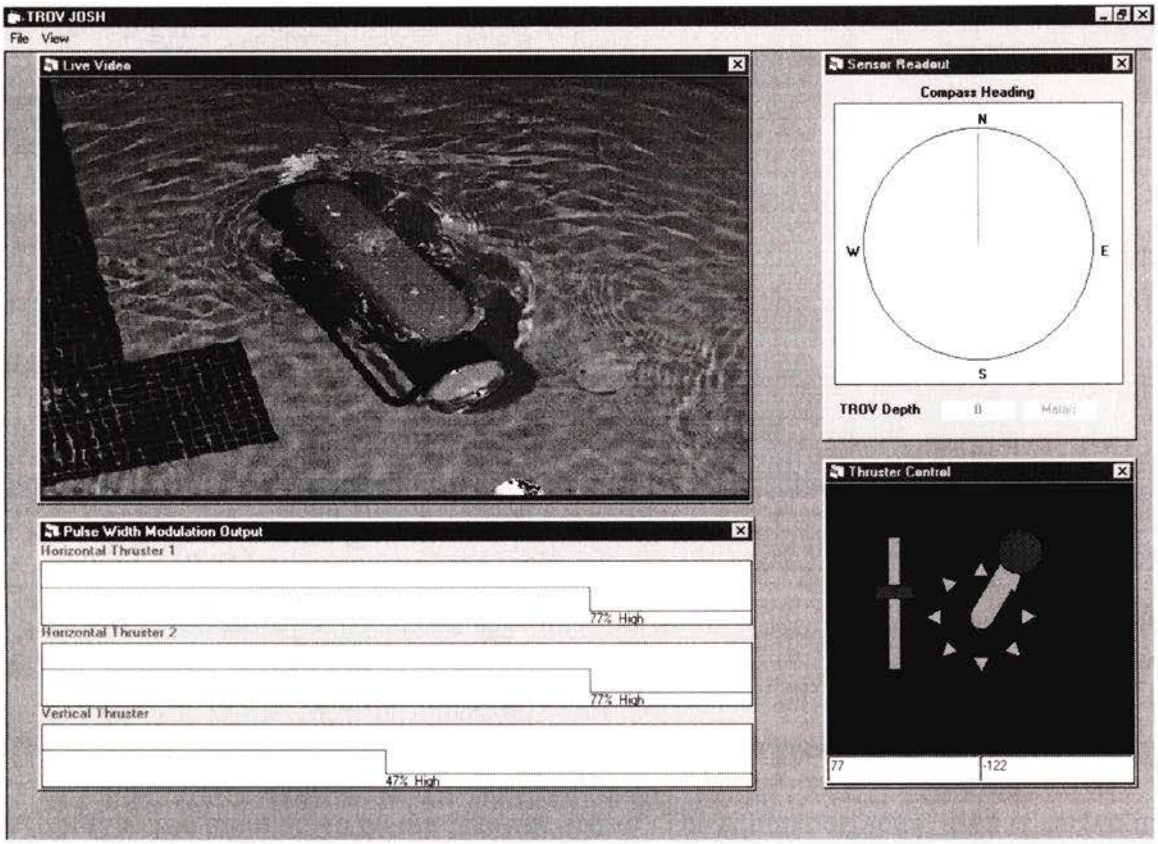


Figure 4.1 TLROV graphical user interface

4.3 Depth sensor

The depth sensor was designed using a piezo-resistive monolithic silicon pressure sensor produced by Motorola. The Motorola MPX 5700DP has several suitable features and is capable of sensing pressure ranging between 0 and 700 Kpa. The sensor is particularly useful in applications where it has to be interfaced to a microcontroller or microprocessor. The pressure sensor outputs an analog signal varying between 0.2 volts and 4.7 volts for the corresponding pressure. This analog signal is input to the microcontroller on an analog port. The microcontroller converts the analog signal to 8 bit digital value between 0 and 255. The digital value is calibrated to represent pressure. From a known value of pressure it is possible to determine the depth using Equation (4).

$$P = \rho g h \tag{4}$$

Where,

- P is pressure in Pascal
- ρ is density of salt water in kg/m3
- g is acceleration due to gravity in m/sec^2
- h is depth in meters

Equation (4) four is rearranged resulting in Equation (5) to solve for depth in meters.

$$h = P / (\rho g) \tag{5}$$

4.3.1 Sensor calibration

The sensor is calibrated by connecting a tube carrying high pressure air from the pressure cylinder to the input valve on the pressure sensor. The calibration setup used is shown in Figure 4.3. A pressure gauge is mounted to the high pressure cylinder. Pressure is increased to 700 Kpa and the corresponding value from the microcontroller is read on a computer screen. Pressure is decreased in steps of 100 until it falls to zero Kpa. The digital value corresponding to pressure at each step is noted, and a graph of pressure versus digital values is shown in Figure 4.2. The graph is linear over a wide range, and this confirms use of the pressure sensor for measuring depth of TLROV.

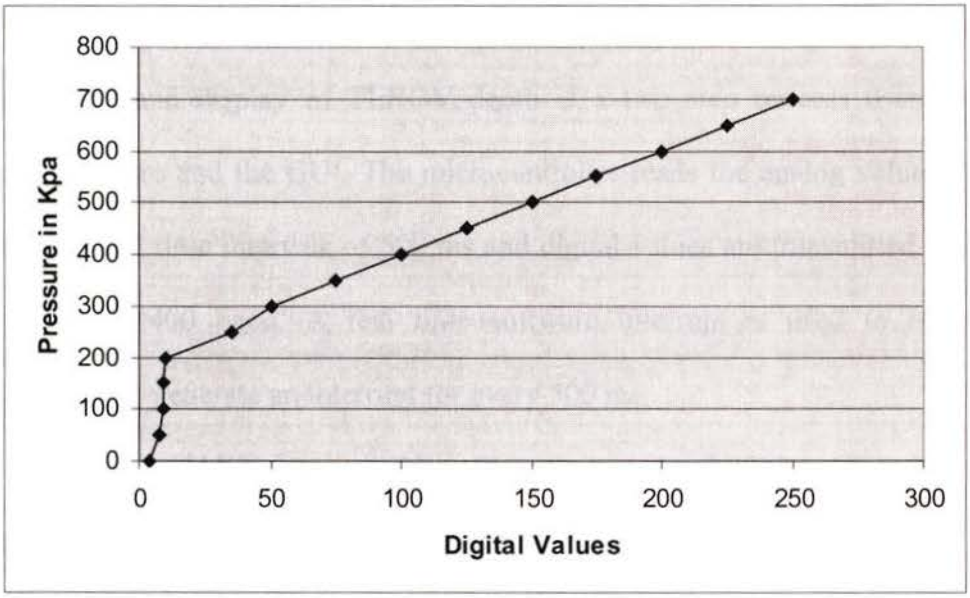


Figure 4.2 Pressure calibration graph

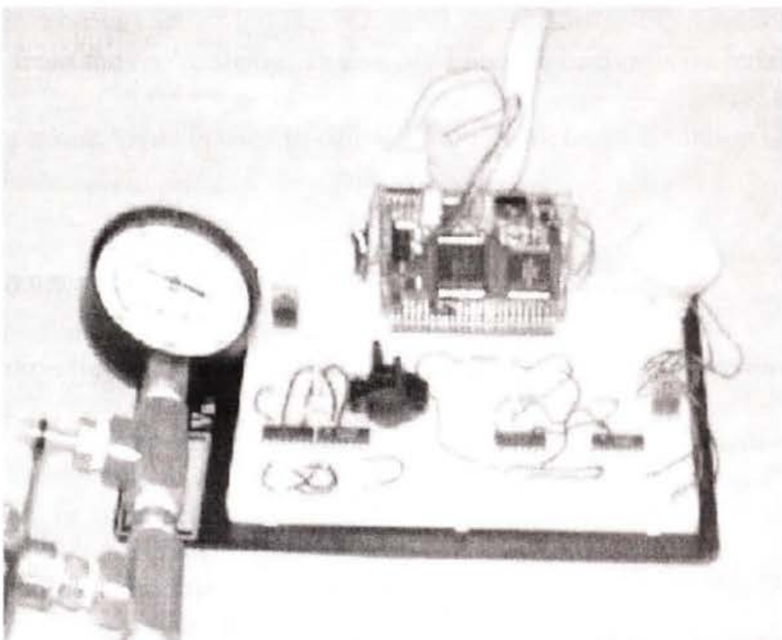


Figure 4.3 Pressure sensor calibration apparatus

4.3.2 Depth sensor software implementation

The calculation and display of TLROV depth is a two step process using both the embedded software and the GUI. The microcontroller reads the analog values from the pressure sensor at time intervals of 500 ms and digital values are transmitted via RS232 to the GUI at 2400 baud. A real time software interrupt is used to instruct the microcontroller to generate an interrupt for every 500 ms.

On the laptop the UART receives the information at the RS232 port and stores the information in a buffer. The size of the buffer is set to 1024 bytes in software. On the GUI, a timer control is employed to read the buffer at 1000 ms intervals, and the contents of the buffer are stored in a variable. Since the variable consists of values that were stored for the past 1000 ms, a string filter algorithm is used to filter and retrieve the most recent pressure value from the entire string. The String Filter algorithm is included in Appendix. The obtained pressure value is used to calculate the depth using Equation (2).

4.4 Compass module

The TLROV direction is determined using the Vector 2X compass module. It has a resolution of 1 degree and an accuracy of 2 degrees. It communicates with microcontrollers, or other electronic components, using the Serial Peripheral Interface protocol. The compass module can be operated either as a master or a slave to retrieve the compass heading. In the master mode, the compass module is responsible for providing the clock required to transmit heading information to the slave. To receive data (heading information), 16 complete clock cycles are required. In the slave mode, the microcontroller interfaced with compass module is responsible for providing the clock signals required to retrieve data at a maximum frequency of 5 Hz.

4.4.1 Compass calibration

The compass module does not have non-volatile memory. Therefore, when the power is turned off the calibration settings are lost. This can be a problem in situations of power failure. To avoid repeated calibration on power up of the host system (microcontroller), the host system calculates the calibration values. The calibrated values are saved in the non-volatile memory of the host system. With the host system calculating the calibration values it is required that the host system also calculate the heading. The following steps are used to calibrate the compass module:

- The compass module is placed in the TLROV on a level plane. To measure the earth's magnetic field the compass module must be on a level plane.
- In software the compass module is configured to operate as the slave while the host system (Microcontroller) is configured to operate as the master.
- It is programmed to operate in RAW mode. In the RAW mode the output values related directly to the two sensors on the compass
- The TLROV is positioned at a known direction 50 degrees and the raw outputs(X and Y) from the compass module are read for sixteen complete clock cycles.
- The TLROV is rotated by 180 degrees and the raw outputs(X and Y) from the compass module are read for 16 complete clock cycles.

The above procedure gives two readings in directions that are 180 degrees apart. These values are stored on the microcontroller and are used to calculate the heading.

4.4.2 Calculation of compass heading

A configuration similar to that used during calibration is maintained between the microcontroller and the compass module for communication, wherein microcontroller is the master and the compass module is the slave. The raw sensor data is retrieved over 16 clock cycles by polling the compass module at a frequency of 38 Hz. The raw sensor data is converted to a heading in software using the following calculations:

- The earth's magnetic field must be considered in order to calculate heading after eliminating magnetic fields from other sources in the system. From the calibration procedure we have sensor values for two axes (X and Y) at two directions 180 degrees apart. This is represented by X1, Y1, X2 and Y2 respectively. Heading is calculated using equation 6.

  $$Ae = \tan^{-1}(Xe / Ye) \quad (6)$$

  Where,

  Ae represents heading.

  Xe is the calculated earth's magnetic field in the X-direction.

  Ye is the calculated earth's magnetic field in the Y-direction.

- Then, Xe and Ye are given by equations (7) and (8):

  $$Xe = Xn - Xo \quad (7)$$

  $$Ye = Yn - Yo \quad (8)$$

  Where,

  Xn and Yn are the output from the two sensors on the compass under normal operation.

Xo and Yo are the are the constant offset which results from local magnetic fields of the system in which the compass module is located.

- The hard iron offsets (Xo, Yo) are the same at both (X1, Y1) and (X2, Y2). The earth's magnetic field at (X1, Y1) is opposite the earth's magnetic field at (X2, Y2) because they are 180 degrees apart. (Xo, Yo) are calculated using equations (9) and (10).

$$Xo = (X1 + X2) / 2 \tag{9}$$

$$Yo = (Y1 + Y2) / 2 \tag{10}$$

Equations (6), (7), (8), (9) and (10) are implemented in software to calculate the heading and the source code is presented in Appendix.

4.5 Video Image Display

The analog signal from the camera on the TLROV is fed to the Universal Serial Bus (USB) video capture device. The USB video capture device converts the analog signals into digital values and presents it to the operating system as streaming media. Microsoft Direct Show is used in Visual C++ to create a filter graph. Use of Direct Show to render video and audio in real time facilitates simple capture, rendering and playback of video streams. The basic building block of Direct Show is a software component called a filter [1]. A filter performs a single operation on a multimedia stream. To perform a given task, an application connects several filters so that the output from one filter becomes the input for another. A collection of filters are connected together to form a logical structure. The logical structure is termed as Filter Graph [1]. To access the components related to a Filter Graph in Visual Basic, the filter graph implemented in VC++ is a dynamic link library which:

- Exposes the components through an interface to Visual Basic. An interface is a group of related functions that provide access to components from the outside world.
- Stores the image frames as a matrix in a SAFEARRAY format. The fundamental building block of any array is the memory block that contains the array's data. A VB array variable is not a pointer to the array's data block. Instead, the array variable points to a structure that describes the contents of the array. This structure is known as an array descriptor. By describing arrays as SAFEARAY structure, VB can leverage a wide variety of array-manipulation routines and interoperate easily with external components. Figure 4.4 shows the filter graph used to render video in real time.

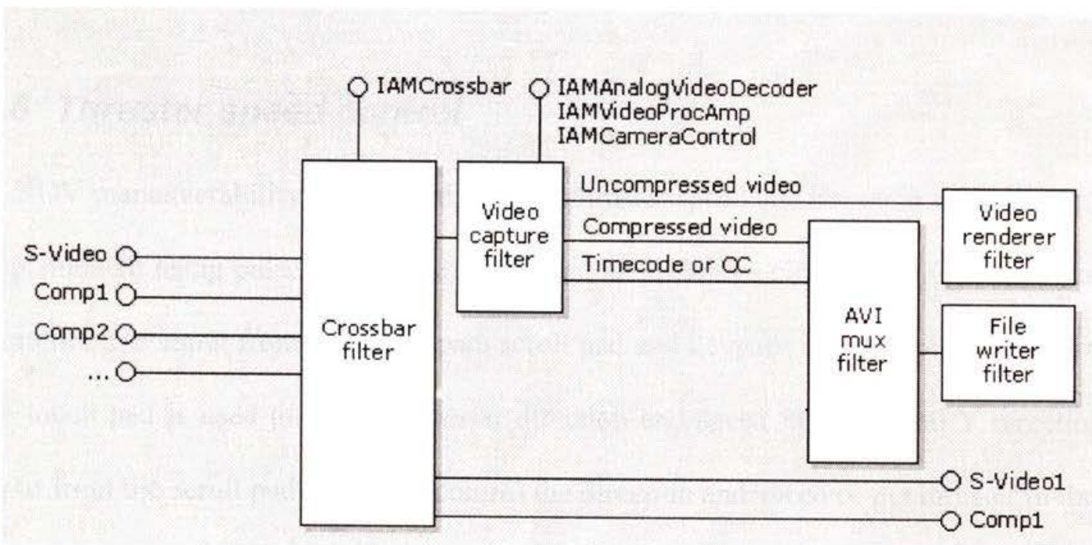


Figure 4.4 Direct show filter graph

With reference to Figure 4.4, each component is treated as a filter and the filter graph consists of the following five filters:

- Crossbar Filter
- Video Capture Filter
- Video Renderer Filter
- File Write Filter
- AVI Mux Filter

The streaming video from the TLROV is received on the S-Video input of the crossbar filter. It is then routed to the input pin on the video capture filter. Since the video is not in a compressed format the output from the video capture filter is directly routed to either the video render filter or the file writer filter based on the input from the user on the graphical user interface. The source code implemented for generation of the filter graph and its execution to render video in real time is attached to Appendix.

4.6 Thruster speed control

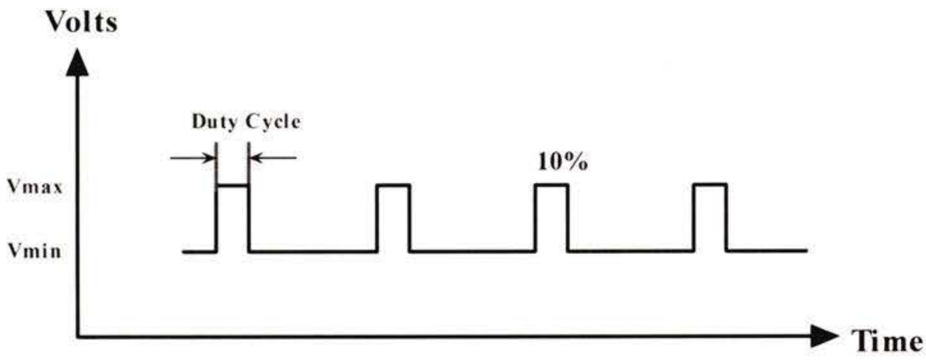
TLROV maneuverability is achieved through thruster speed and direction control. This is implemented using pulse width modulation and an H-Bridge circuit. The GUI constantly monitors user input from the touch pad, scroll pad and keypads on the laptop. Input from the touch pad is used to control thruster direction and speed in the X and Y directions. Input from the scroll pad is used to control the direction and speed of the thruster in the Z direction (vertical thrust). The microcontroller on the TLROV receives the speed and direction for each thruster from the GUI in the RS232 format, and the values received are used to generate pulse width modulation and direction signals for output to the motor drivers. For a change in thruster direction of rotation, the supply voltage polarity is reversed. To achieve the change in polarity, based on digital logic outputs from the microcontroller, an H-Bridge circuit is required. The implementation of an H-Bridge circuit is presented in Chapter 3. 2.

4.6.1 Pulse width modulation

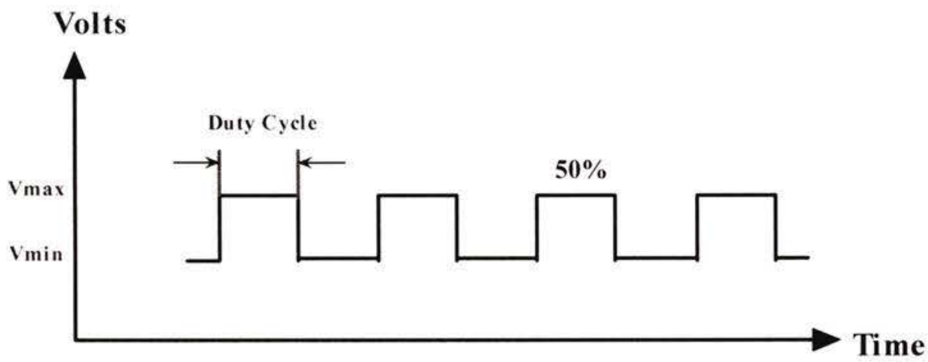
By controlling analog circuits digitally, system costs and power consumption can be drastically reduced. PWM is a way of digitally encoding analog signal levels. Through the use of high-resolution counters, the duty cycle of a square wave is modulated to encode a specific analog signal level. The PWM signal is still digital because, at any given instant of time, the full DC supply is either fully on or fully off. The voltage or current source is supplied to the analog load by means of a repeating series of on and off pulses. The on-time is the time during which the DC supply is applied to the load, and the off-time is the period during which, that supply is switched off. Given sufficient bandwidth, any analog value can be encoded with PWM.

Figure 4.5 shows three different PWM signals. Figure 5(a) shows a PWM output at a 10% duty cycle. That is, the signal is on for 10% of the period and off the other 90%.Figures 5b and 5c show PWM outputs at 50% and 90% duty cycles respectively. These three PWM outputs encode three different analog signal values, at 10%, 50%, and 90% of the full strength. If, for example, the supply is 9V and the duty cycle is 10%, a 0.9V analog signal results.

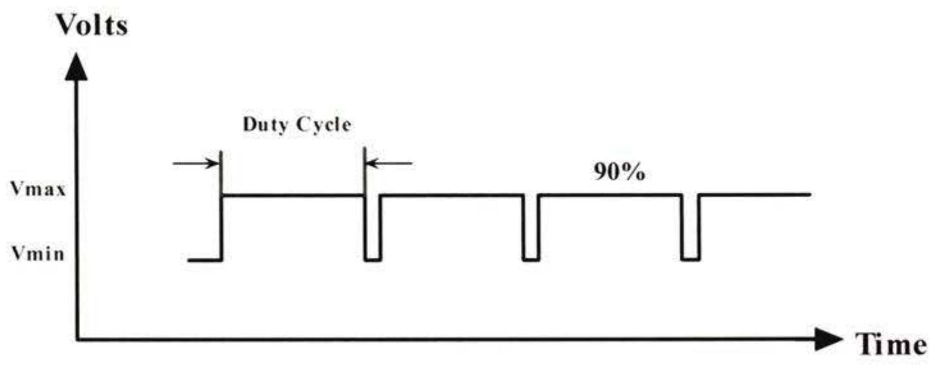
This concept of PWM has been used in the implementation of the TLROV to control speed of motors and the brightness of light. The source code for generation of a pulse width modulated output based on the user input for control of speed and intensity of light is included in Appendix.



5(a)



5(b)



5(c)

Figure 4.5 Pulse width modulation

4.7 System startup procedure

The following sequence of startup checks are performed in software to check for the availability of communication channels and the proper functionality of all devices:

- A check is made to determine the availability of serial ports on the laptop for communication with the microcontroller on the TLROV. If no serial communication ports are available the application is terminated
- If a serial communication port is available, an 8 bit test string is sent to the microcontroller on the TLROV and the GUI waits for an 8 bit reply. On receiving a reply from the microcontroller a check is performed for the validity of data received. If the data received from the microcontroller is not valid or if no data is received, the following assumptions are made:
  - The selected port for communication port on the laptop is incorrect. To correct this problem the user has to switch the cable from the currently connected port to the next available serial port.
  - There is a fault in the communication channel established between the graphical user interface and the microcontroller on the TLROV. This could be due to improper connection of cables or lack of power supply to the microcontrollers.

If the data received is valid, all devices related to the TLROV are functional and the GUI tests for communication with the USB frame grabber, initializing the device driver related to the USB capture device. If 30 frames of valid data are received from the USB device, the camera and the capture device are functional and the application is started. If there is no valid data from the USB capture device the application is started with a warning to the operator, indicating that the live video may not be available. With the above sequence of operations debugging faults related to communication and devices is made easy for the operator of the TLROV. Source code implementation related to the startup procedure is included in Appendix.

Chapter 5

5 TLROV ultrasonic communication system

The use of a tether as a link between the operator and the underwater vehicle for transmitting control signals and receiving sensor information poses severe limitations on the control of an underwater vehicle. In order to circumvent the problems, outlined in Chapter 1, inherent with the use of a tether, a wireless communication system has been implemented.

Sound waves are a suitable medium for communication underwater with a TLROV due to improved propagation in water over air. This is due to the increased density of water relative to air. Higher water density 1024 kg /m^3, gives a corresponding wave velocity 1525 m/ sec. How sound waves propagate under water largely depends on the frequency. At higher frequencies sound waves attenuate faster and hence the distance traveled is greatly reduced. This is attributed to increased absorption of sound waves by water molecules at higher frequencies. At low frequencies sound waves are capable of traveling to greater distances with reduced attenuation. Moreover the bandwidth available to transmit information is greatly reduced at low frequencies. This poses a limitation in transmitting video images, for which, megahertz bandwidth is required.

The typical range of frequencies that humans can hear is between 20 Hz and 20 KHz. Frequencies above 20 kHz are termed as ultrasonic. This chapter discusses and presents the implementation of an ultrasonic communication system used in the TLROV for receiving control signals and transmitting sensor data.

5.1 The communication system

Communication involves the following operations to transmit a signal form the source to the destination:

- The generation of a message signal.
- The description of the message signal with the desired level of accuracy.
- Encoding the symbols in a format suitable for transmission.
- The transmission of the encoded symbols.
- Decoding and reproduction of the encoded symbols at the receiving end.

With reference to Figure 5.1, the fundamental components of a communication system are the transmitter, channel, and receiver. The communication system designed for the TLROV consists of a transceiver interfaced to a laptop (through the RS-232) port and another transceiver interfaced to the microcontroller on the TLROV. Both transceivers are capable of transmitting and receiving ultrasonic signals at a maximum frequency of f_{pizeo}= 40 kHz in the half duplex mode.

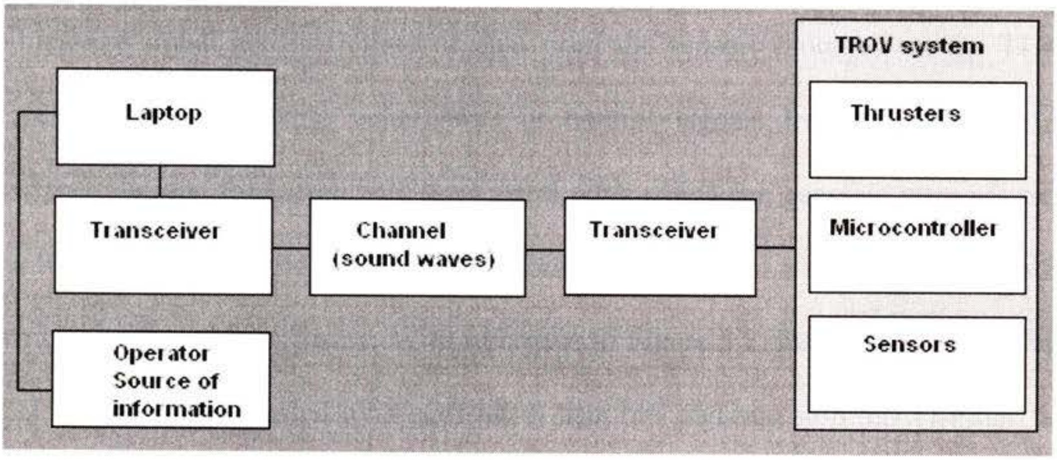


Figure 5.1 Proposed underwater communication system

Testing the transducer underwater, for its communication capabilities has confirmed that the transducer is resonant and is sensitive to frequencies that are equal to f_{pizeo} or $f_{pizeo/}n$. Sensitivity of the receiver directly relates to the capability of the transceiver diaphragm to oscillate. Thus, for the receiver to be sensitive to weak signals over longer distances the carrier frequency should be equal to f_{pizeo}. The frequency of sound that operates best in water and propagates over a long distance with low attenuation has been determined to range between 18 kHz and 27 kHz [16]. This leads to setting the carrier frequency for the communication system in TLROV to 20 kHz.

The mode of communication can be either a broadcast or point-to-point. If the communication mode is broadcast, then there is only one transmitter and several receivers. If the mode of communication is point-to-point, then the communication process takes place between a transmitter and receiver. In this case, there is usually a bidirectional flow of information.

5.2 Generation of message signal

The message signal consists either of data from the sensors mounted on the TLROV (compass heading, pressure value, etc.) or control signals from the operator. The microcontroller and the laptop both have serial ports which can generate message signals in the RS232 format. Signals in the RS232 format are output at a baud rate of up to 2400 which can be varied in software. With reference to Figure 5.2, the message signal in the RS232 format is represented by a start bit, 8 data bits and one stop bit. The start bit is responsible for indicating to the receiver that new data is ready to be sent. The 8 data bits consist of sensor information and a maximum value of 2^8 (256) can be sent at a time. The stop bit is used to indicate to the receiver, the end of data transmission. The message signal transmitted over a channel has no guarantee of accuracy at the receiver. This is due to the noise in the system. In order to check for the correctness of the data at the receiving end, instead of using 8 bits for data, it is common to use only 7 bits for data and the remaining bit is used for parity. This method of error detection will limit the value that can be sent to a maximum value of 2^7(128). Thus, the number of data bits is retained at 8 in this implementation and error correction is achieved by using a parity check in software, which is discussed in Chapter 5.6.

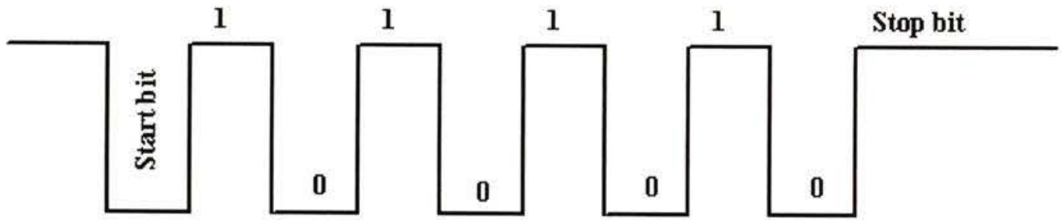


Figure 5.2 RS232 data format

5.3 Signal modulation

Modulation essentially involves encoding the symbols in a format that can be understood by the receiver and suitable for transmission over the channel. The receiver recreates the original symbols removing noise that has been added over the channel during transmission. The type of modulation scheme used dictates the quantity of noise that can be removed before the original message is retrieved [4]. The modulation process used in the TLROV communication system is pulse code amplitude modulation. The carrier wave is a series of rectangular pulses (see Figure 5.3) at a frequency of 20 KHz generated by the oscillator.

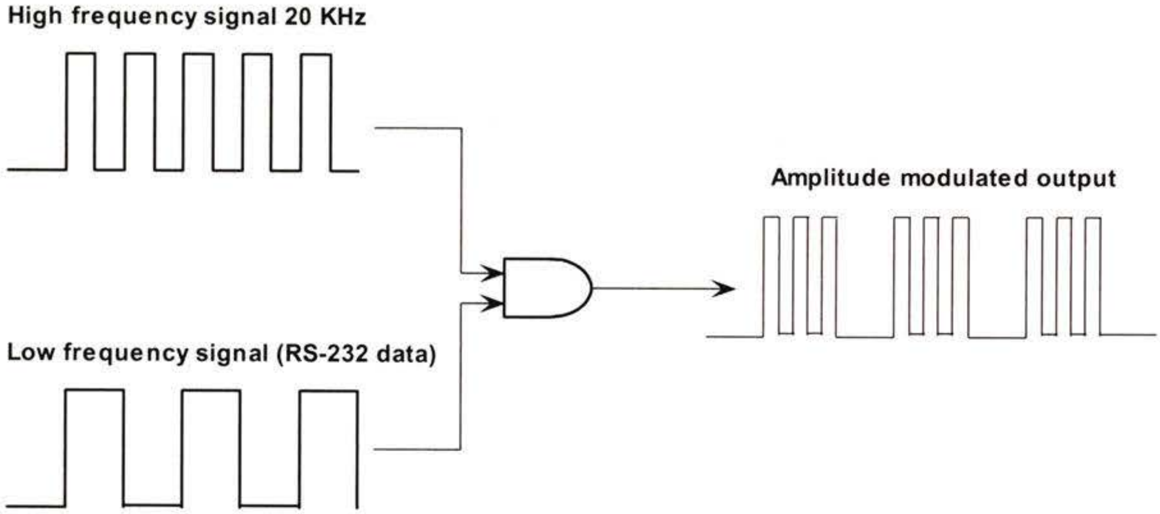


Figure 5.3 Amplitude modulation

The message signal is a low frequency signal in the RS232 format. Both the signals (carrier and message) are input to an AND gate and the output from the AND gate is a pulse code amplitude modulated signal. Pulse code amplitude modulation refers to varying the amplitude of a high frequency periodic digital signal in accordance with the message signal [4]. The transmitter circuit used to communicate with the TLROV uses pulse code amplitude modulation scheme to represent the message signal in a format suitable to transmit over the channel. Amplitude modulation is preferred over frequency modulation as the transceivers are resonant and are sensitive to frequencies equal to F_{pizeo} or F_{pizeo}/n. A frequency modulated signal would have worked much better with a non-resonant piezo. This is due to the complexity required to assure that the resonant frequency is targeted. Overshoot or thermal drift, for example, would cause the data to be misrepresented in a resonant design. This frequency scale is very small and a number of component issues come into effect. For this reason, amplitude modulation was chosen. If a non-resonant piezo had been available, frequency modulation would have been attempted, to further combat noise issues.

5.4 Transmitting the modulated signal

The schematic of the transmitter used in the TLROV is shown in Figure 5.4, and has the following major features:

- Transistors Q2 and Q1

  They are used in combination to form an AND gate. The message signal is input to Q2 in the RS232 format. The carrier signal is generated using an oscillator which is a series of periodic pulses at a frequency of 20 KHz and input to Q1. Switching transistor Q2 on with a 5 volts signal at the base resistor, results in current flowing from collector to emitter on Q2. This prevents transistor Q1 from conducting and the piezo X1 sees a low (zero volts). Switching Q2 off results in transistor Q1 conducting and the piezo X1 sees the high frequency signal (5 volts).

- Zener diode D1

  The carrier signal which has a square wave form, is generated by a 0 to 5 volts output by an oscillator or function generator. If the output voltage from the generator is too high, or biased incorrectly by a user, the zener will protect the transmitter by keeping the voltage between +5V and -0.7V.

- Resistor R1 and capacitor C1

  Used in combination as a low pass filter. The use of a low pass filter removes some high frequency components from the carrier signal and minimizes shock to the piezo that might occur from a square wave response (it keeps piezo from shattering).

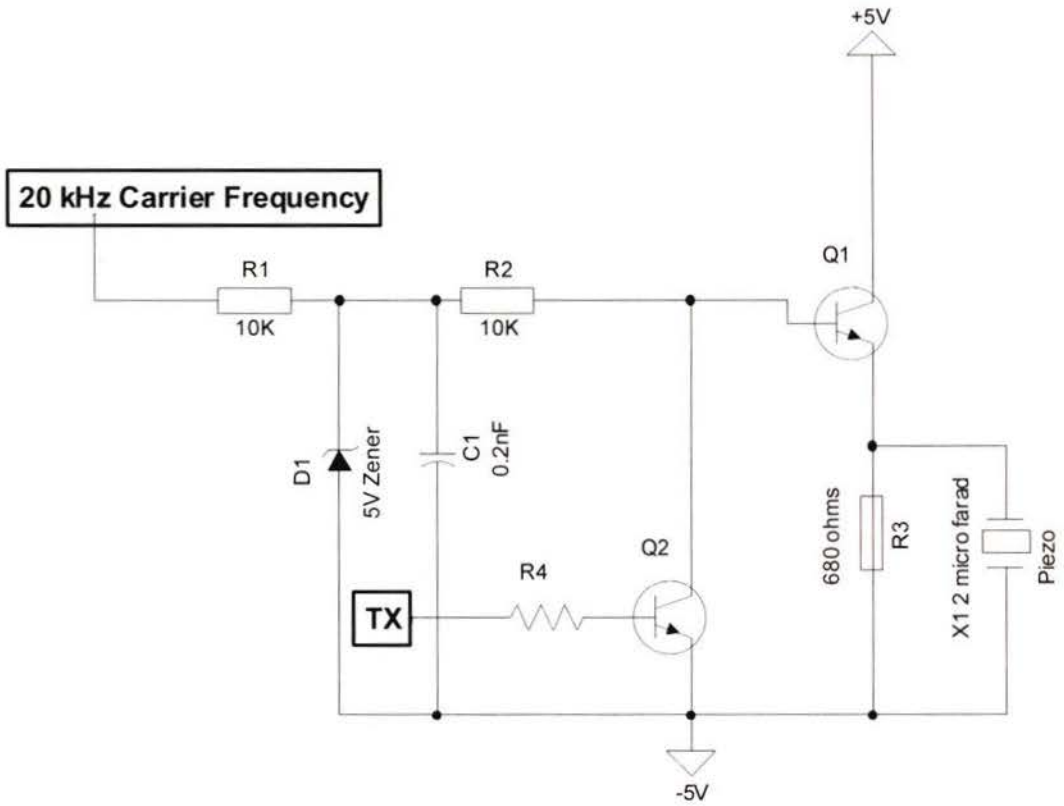


Figure 5.4 Transmitter circuit

- Resistor R2

  R2 is used to set the gain of the transistor Q1 and limit current to both the transistors Q1 and Q2 avoiding damage to the transistors.

- Resistor R3 and Transducer X1

  The amplitude modulated signal before being transmitted is approximated to a sine wave, limiting frequencies higher than 40 KHz. The piezo X1 and resistor R3 in combination can be modeled as a low pass filter. By limiting frequency, the piezo (X1) is further protected from shattering. The square wave is input to the transducer. Input signal to the transducer pulses the diaphragm high and low which results in the generation of sound waves. Figure 5.5 is used to show the effect of R3 and X1 in combination acting as a low pass filter to the square wave input.

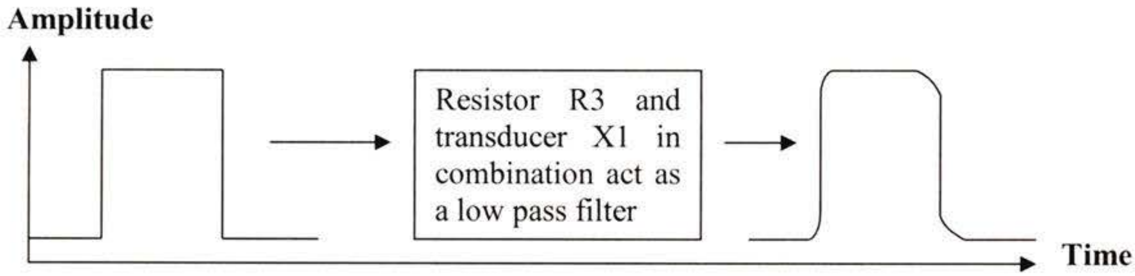


Figure 5.5 Effect of low pass filter on square wave

5.5 Signal demodulation

The received signal is weak and requires amplification before demodulation. The weak signal is amplified in three stages as shown in Figure 5.6. The output from the opamp at each stage is set to produce a gain of 30 units. U1, U2 and U3 represent the three operational amplifiers, producing an over all gain of 900 units. The output from U1 is an amplified signal that has low frequency components (60 Hz) which has to be filtered. A high pass filter is used to remove the low frequency noise (mostly 60 Hz) from the signal. Resistor R1 and capacitor C1 are used in combination to form a single stage high pass filter. The capacitor C1 is an insulator to 60 Hz signal and the capacitance of the capacitor used determines the cut off frequency. Equation 1 is used to determine the values for capacitor C1 and resistor R1.

$$F = 1 / (2 \pi R C) \qquad (1)$$

Where,

- F is the cutoff frequency
- π is a constant = 3.142

- R is the value of resistance
- C is the value of capacitance

The cutoff frequency, for which the high pass filter (R1 and C1 used in combination) is designed, does not produce the ideal brick wall response as shown in Figure 5.7. This is due to the attenuation in the filter. An ideal filter will have no attenuation and the response of the filter will resemble the brick wall. In practice the response of the system is improved by cascading two or more high-pass filters. Resistor R2 and capacitor C2 in combination represent a second stage high pass filter, which are cascaded with the first stage high pass filter as shown in Figure 5.6. Resistors R3-R4, R5-R6 and R7-R8 are used to tune the gain in the operational amplifiers U1, U2 and U3 respectively.

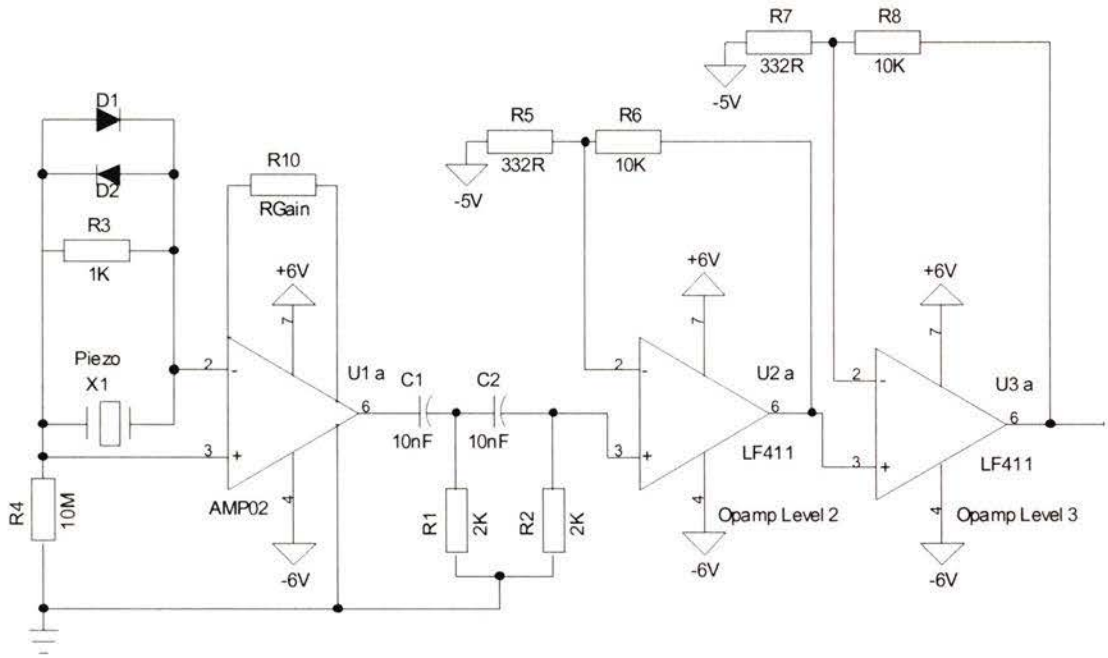


Figure 5.6 Receiver Circuit stage 1

The amplified signal from first stage is converted to RS232 format for input to the microcontroller in stage 2 of the receiver circuit as shown in Figure 5.8. The 5V input to the capacitor (C3) charges the capacitor. When diode D3 is not conducting, the output is a high (5 Volts). At 0.7 volts the diode D3 starts conducting and the output is a low (0 volts). The rounded edges of the signal are trimmed using a Schmitt trigger (U2c in Figure 5.8) to represent the signal in a TTL level RS-232 data format.

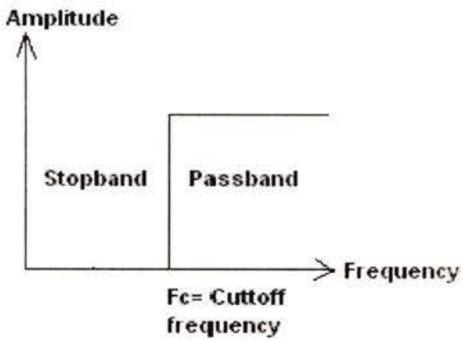


Figure 5.7 Ideal high pass filter

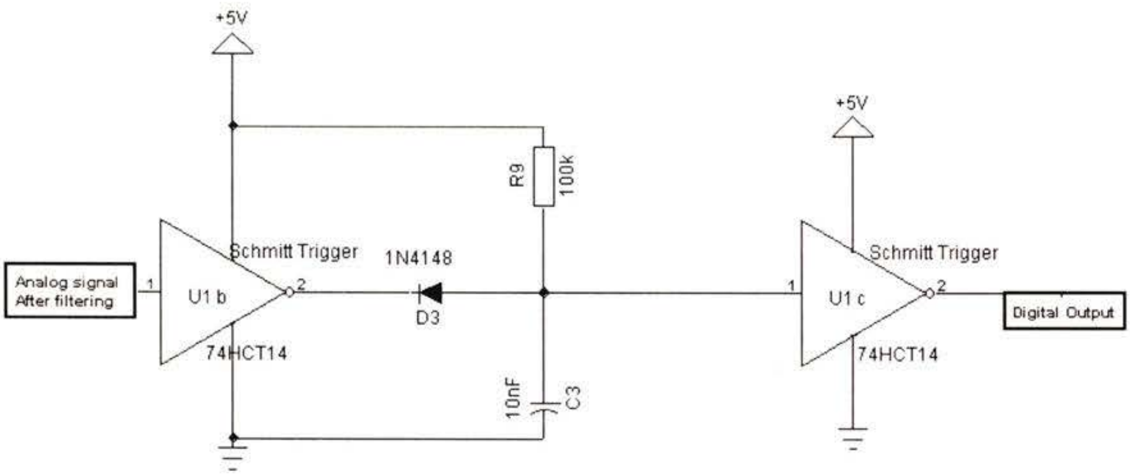


Figure 5.8 Receiver circuit stage 2

5.6 Error correction

With sophisticated hardware it is possible to generate the original signal quite reliably. However, there are many times when the received signal is not accurate and it is important to determine the accuracy of the signal received. For this reason, it is required to implement error correction either in hardware or in software. For the communication system used in the TLROV, an error correction algorithm is implemented in software using a parity check in software.

5.6.1 Calculation of checksum

To implement error correction in software the microcontroller creates a value which is function of the data to be transmitted. The value created is termed as checksum. Before transmitting the data the checksum value is appended to data. At the receiver, the checksum value is separated form the data and performs the same arithmetic that the transmitter had performed to calculate a checksum. The checksum received from transmitter is compared with the calculated checksum at the receiver. If they are equal the data received is most likely error free and the data is retained. If the checksum values are not equal, the data received is corrupted and it is discarded.

Message	**Checksum (Sum of all high bits)**	**Message after transmission**
Decimal value 6	2	Byte1 = 20 (Data) Byte 2 = 2 (Checksum)
Decimal value 20	2	Byte 1 = 6 (Data) Byte 2 = 2 (Checksum)

Table 3: Checksum example

Table 3 is used to show an example of a checksum calculation. The data to be transmitted is the decimal value 6. The checksum value (decimal 2) calculated for the data, is the sum of all the high bits from the 8 bit binary representation (00000110). The data received by the receiver is the decimal value 20 followed by decimal 2 due to the noise in the channel. The receiver computes a new checksum for the data received which turns out to be decimal value 2. Although the data received is a decimal 20 in place of the decimal 6, both the transmitted checksum and the calculated checksum at the receiver are equal and this leads to retaining the decimal value 20 which is incorrect. The checksum algorithm used in the example shown in Table 3 is very simple and does not affect every bit of the data, which leads to ambiguities. To increase the effectiveness of the error correction algorithm other techniques need to be used [21]. There are many complex algorithms available for error correction that change the original data and calculate a checksum. For example, the use of binary division as an alternative to binary addition increases the accuracy. Binary division affects every bit on the data, increases the complexity of the checksum algorithm and reduces the ambiguity evident in the example shown in Table 3. For calculating the checksum using binary division, a divisor and dividend are required. Data represents the dividend and the divisor is a constant 8 bit value. The remainder during the binary division process is altered at every step and the remainder at the end of calculation represents the checksum value. The simple error correction algorithm used in the TLROV communication system is an even parity check implemented in software. Parity check determines whether the number of ones or zeros in a byte is odd or even. If the number of zero's present in the byte is used to calculate parity, it is termed as even parity and if number of one's present in the byte is used to calculate parity it is termed as odd parity.

5.7 Communication system performance results

The ultrasonic communication system performance has been evaluated underwater up to distances of 150 feet. The distance at which the receiver can sense the transmitted signal is affected by the beam angle of the transducers. It defines how much the beam will spread with distance. Beam angle is largely determined by the frequency of the sound waves. A high frequency transducer produces a narrow beam. A low frequency transducer produces a wider beam. Beam angle can be shaped to some extent in the physical design of a transducer. A wide beam angle for the transducers would spread the sound wave over a larger surface area in water and hence the signal is available for long distances. The use of wide beam angle weakens the signal due to dispersion of sound waves and requires the use of highly sensitive receivers. A transducer with narrow beam angle can be used to achieve greater depths of communication with reduced distance due to the sound waves being focused over a small surface area. The graphs shown in Figure 5.9 and Figure 5.10, obtained from test results are used to characterize the communication system. The ultrasonic communication system was tested at a depth of 10 feet over a range of 80 feet. With increase in distance the signal strength dropped (see Figure 5.10) and at 84 feet the signal received was noise affected along with a drop in signal strength to 0.8 Vac RMS. With reference to Figure 5.9, the signal reception was good to an angle of up to +90 and -90 degrees and with only a small reduction in signal strength between 90 and 120 degrees. This confirms that the transducers used have wide beam angle. The reduction in signal strength and the presence of noise in the signal are attributed to the presence of noise in the area surrounding the test platform.

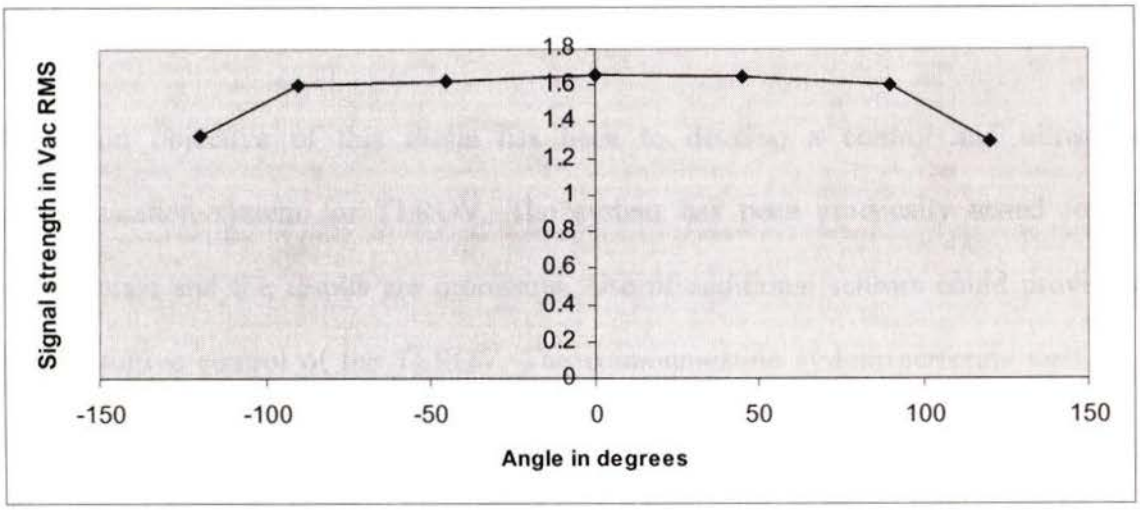


Figure 5.9 Angle versus signal strength performance

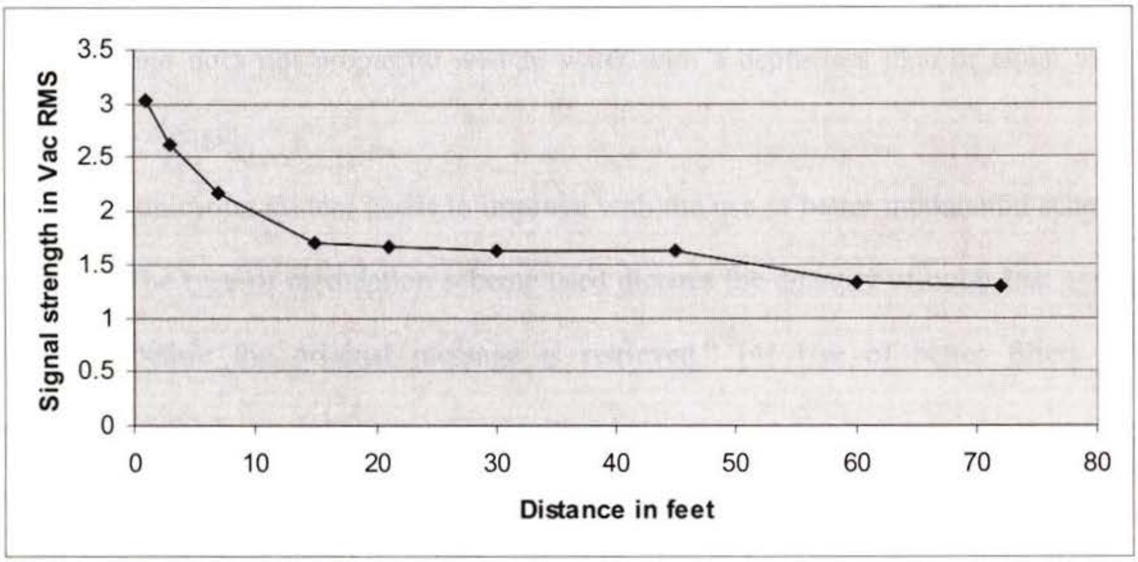


Figure 5.10 Distance versus signal strength performance

Chapter 6

6 Conclusion

The main objective of this thesis has been to develop a control and ultrasonic communication system for TLROV. The system has been practically tested for its performance and the results are promising. Use of additional sensors could provide a more intuitive control of the TLROV. The communication system performs well to a distance of 80 feet without large degradation in signal quality. Distances beyond this have not been tested, but the system is expected to perform better in deeper waters and over longer distances due to the following reasons:

- With increase in depth noise in the 20 kHz frequency range is greatly reduced.
- Sound does not propagate well in water with a depth less than or equal to it's wavelength.

The communication system needs to improve with the use of better modulation schemes because "The type of modulation scheme used dictates the quantity of noise that can be removed before the original message is retrieved." [4] Use of better filters and transducers which are not resonant can further help improve the communication system. The promising results, seen from the test runs, have further increased interest to study and develop the TLROV in fields related to communication and control.

6.1 Future work

There are several aspects that can be improved for the TLROV system. An aerodynamic design of the TLROV's physical structure would improve performance by reducing drag on the vehicle. A modular design in structure can facilitate upgrading the TLROV for the addition of sensors, actuators and electronics as required.

The communication system developed for the thesis does not take into consideration the effects of multi-path signals. The signals that bounce off objects and surfaces underwater would cause interference for both communication and the navigation system. The communication system has to be further improved to overcome this limitation.

The error correction currently implemented in software to detect errors does not try to reproduce the original information on detection of error. Instead the algorithm basically rejects the message and requests for a resend to the transmitter. This method of error-correction increases traffic in the communication channel. To overcome this, better error correction algorithms, which will reconstruct the original message on detection of error need to be implemented.

Finally, the TLROV system requires reliable navigation and image acquisition systems to determine its position underwater at an instant and transfer video in real time. A navigation system which is based on the principle of triangulation is currently under development. An image acquisition system has many challenges due to the multi-path correction required and the limited bandwidth. The topic of image acquisition in real time can be treated as a topic on its own for a MSc or PhD thesis.

6.2 References

[1] Microsoft.
"Introduction to DirectShow"
http://msdn.microsoft.com/

[2] Microsoft.
"Visual Studio and Languages"
http://msdn.microsoft.com/

[3] Albert Paul Malvino.
"Electronic Principles"
Mc-Graw Hill Publications, 6th edition

[4] Simon Haykin
"Communication Systems"
Wiley Publications, 4th edition

[5] Drew Michel
"ROV Educational Materials"
http://www.rov.org/education.html

[6] MIT
"AUV History"
http://auvlab.mit.edu/

[7] Motorola
"Motor Operation Principles"
http://e-www.motorola.com/collateral/MOTORDCTUT.html

[8] Peter H. Anderson and Vance Foster
"Interfacing with A Precision Navigation Vector 2X Compass Module"
http://www.phanderson.com/printer/compass/compass.html

[9] Motorola.
"68HC11 Reference Manual"
http://e-www.motorola.com/brdata/PDFDB/docs/M68HC11RM.pdf

[10] Parallax.
"Basic Stamp Manual"
http://www.parallaxinc.com/downloads/Documentation/Basic Stamps/BASIC Stamp Manual v2.0.pdf

[11] Peter Wild and Kevin Jones
"Mechatronic Systems Design"
Department of Mechanical Engineering
University of Victoria

[12] Horowitz and Hill
"The Art of Electronics"
Cambridge publication, 2nd edition

[13] Sea Technology
"Diving, Underwater Vehicles and Imaging"

[14] Sea Technology
"Diving, Underwater Vehicles and Imaging"
Volume 41, No.12

[15] Sea technology
"Offshore Technology Ocean engineering"
Volume 41, No.4

[16] James C. Preisig and Mark P.Johnson
"Signal Detection for Communications in the Underwater Acoustic Environment"
Oceanic Engineering, Volume 25, no.4, 2001

[17] David Bryan Marco and Anthony J.Healey
"Command Control and Navigation Experimental Results with the NPS AIERIES AUV", Oceanic Engineering, Volume 25, no.4, 2001

[18] Albert M.Bradley, Micahel D. Feezor, Hanuman Singh and F.Yates Sorrell
"Power Systems for Autonomous Underwater Vehicles", Oceanic Engineering, Volume 25, no.4, 2001

[19] Charles C. Eriksen and T. James Osse
"Seaglider: A Long-Range Autonomous Underwater Vehicle for Oceanographic Research", Oceanic Engineering, Volume 25, no.4, 2001

[20] Gianluca Antonelli and Stefano Chiaverini
"Real-Time Path Planning and Obstacle Avoidance for RAIS: An Autonomous Underwater Vehicle", Oceanic Engineering, Volume 25, no.4, 2001

[21] Ross N. Williams
"A painless guide to crc-error correction algorithm"
http://www.geocities.com/SiliconValley/Pines/8659/crc.htm

6.3 Appendix

Drag calculation:

Drag = 0.5 (density of sea water / gravitational acceleration) x (area) x

(Velocity) ^{2} x (non-dimensional drag coefficient)

Density of sea water = 1025 kg/m3

Gravitational acceleration = 9.8 m/sec^2

Area of front face of vehicle = 3.14 x 0.063 x 0.063 = 0.01142 m^2

Area of tether = (0.00534 /12) x 100 = 0.0445 m^2

Velocity = 1 knot (0.51 m/sec)

Non-dimensional drag coefficient for vehicle = 0.8

Non-dimensional drag coefficient for tether = 1.2

Vehicle drag = ½ x (1025 / 9.81) x (0.011426) x (0.51)^2 x (0.8) = 0.124

Umbilical drag = ½ x (1025 / 9.81) x (0.0445) x (0.51)^2 x (1.2) = 0.73 kg

<u>Buoyancy calculation results:</u>

Density (pounds per cubic inch):

- Aluminum: 0.1
- Acrylic: 0.042
- Water: 0.036
- Foam: 0.009

Part	Volume of material in cubic inches	Mass of material (pounds)	Volume of water displaced (cubic inches)	Mass of displaced water (pounds)	Net underwater mass (pounds):
Housing	61.68	6.17	306	11	-4.8
Vertical tube	3.74	0.37	3.74	0.13	0.24
Lens	8.82	0.37	3.9	0.14	0.23
End cap	6.65	0.67	1.7	0.06	0.61
Thrusters (3)	8 each	0.77 each	8 each	0.29 each	0.48 each
Thruster mounts (2)	1.29 each	0.13 each	1.29 each	0.05 each	0.08 each
Thruster mount spacers (2)	0.38 each	0.038 each	0.38 each	0.014 each	0.024 each
Stand	8.81	0.88	8.81	0.317	0.56
Foam	101	0.91	101	3.64	-3.64
Batteries		1.37 each			2.74

Distance from center of housing (inches):

- Lens: 9.94
- Endcap: 9.87
- Thrusters: 3.5
- Thruster mounts: 5.03
- Batteries: 3.6

String filter algorithm:

```
Private Sub Timer1_Timer()

// Variable declarations
Dim Buffer As Variant
Dim st1 As Variant
Dim st2 As Variant
//Clear text area for display
Text1.Text = " "

//Read all data from serial port
Buffer = MSComm1.Input

//store the data in a temporary location after conversion to visual basic format
temp = Text1.Text + StrConv(Buffer, vbUnicode)

//flush the serial port buffer
MSComm1.InBufferCount = 0

//extract the most recent data form temporary storage and display information to user
st1 = temp
st2 = ""
While Left(st1, 1) = Chr(32)
 st1 = Right(st1, Len(st1) - 1)
Wend
While Left(st1, 1) <> Chr(32)
```

st2 = st2 & Left(st1, 1)

st1 = Right(st1, Len(st1) - 1)

Wend

Text1.Text = st2

End Sub

<u>Read Compass heading:</u>

```
void readcompass(void)
{
      double bit;
      PORTB = 0x07;           //Make Portb pins 0(Reset), 1(SS) and
                              //2(Clck) High

      delay();                //Generate a small delay

      PORTB = 0x05;           //Make PC and SS pin 1 low
      delay();                //wait for campass to caluclate
                              //heading

      delay();                //wait for campass to caluclate
                              //heading

      PORTB = 0x07;           //Make Portb pins 0(Reset), 1(SS) and
                              //2(Clck) High

      PORTB = 0x05;           //make pc/ss low

      /* clock 7 times to clock out seven bits which aren't
     used. */

     for (bit = 1;bit <= 7;bit ++)
     {

          PORTB = 0x02;      //make clck low
          PORTB = 0x06;      //make clck high

     }

     /* get the next bit */
     PORTB = 0x02;           //make clck low
```

```
    PORTB = 0x06;          //make clck high

    if (ADR1 == 1)
    {
          heading=256;

    }



    /* clock out the next 8 bits */
    delay();
    for (bit = 7;bit > 0;bit --)
    {
          PORTB = 0x02;     //make clck low
          PORTB = 0x06;     //make clck high
          if (ADR1 != 0)
          {
                heading +=(int)((2^(int)(bit))+1);
          }

    }

    /*clock out final bit (bit 0)*/

    PORTB = 0x02;          //make clck low
    PORTB = 0x06;          //make clck high
    if (ADR1 != 0)
    {
          heading +=1;

    }
    PORTB = 0x02;          //make clck low
    PORTB = 0x07;          //Make Portb pins 0(Reset), 1(SS) and
                           2(Clck) High

    PORTB = 0x06;          //make clck high
}
```

Filter graph generation:

```
Dim gGraph As IMediaControl
Dim gRegFilters As Object
Dim gCapStill As VBGrabber

Private Declare Function CreateCompatibleDC Lib "GDI32" _
   (ByVal hdc As Long) As Long
Private Declare Function SelectObject Lib "GDI32" _
   (ByVal hdc As Long, ByVal hbitmap As Long) As Long
Private Declare Function BitBlt Lib "GDI32" _
   (ByVal hdc As Long, ByVal x As Long, ByVal y As Long, _
   ByVal width As Long, ByVal height As Long, _
   ByVal hdcSrc As Long, ByVal xSrc As Long, ByVal ySrc As Long, _
   ByVal mode As Long) _
   As Long
Private Declare Sub DeleteDC Lib "GDI32" _
   (ByVal hdc As Long)

Public Sub ShowVideo()
Set gGraph = New FilgraphManager
Set gRegFilters = gGraph.RegFilterCollection
View
End Sub

Private Sub View()
' make a new graph
   Set gGraph = Nothing
   Set gCapStill = Nothing
   Set gGraph = New FilgraphManager
   Set gRegFilters = gGraph.RegFilterCollection
   Dim pinOut As IPinInfo
   Dim filter As IRegFilterInfo
```

```
 Dim fGrab As IFilterInfo
 Dim fSrc As IFilterInfo
 Dim xbar As CrossbarInfo
 Dim pinIn As IPinInfo
Dim pin As String
For Each filter In gRegFilters
   If filter.Name = "SampleGrabber" Then
      filter.filter fGrab
      Set gCapStill = New VBGrabber
      gCapStill.FilterInfo = fGrab
      Exit For
 End If
Next filter

For Each filter In gRegFilters
   If filter.Name = "ATI Rage Theater Video Capture" Then
      filter.filter fSrc
      Exit For
   End If
Next filter

Set xbar = New CrossbarInfo
On Error GoTo NoXBar
xbar.SetFilter fSrc

Dim idx As Long
For idx = 0 To xbar.Inputs - 1
        pin = xbar.Name(True, idx)
Next idx

xbar.Standard = AnalogVideo_NTSC_M_J
```

```
NoXBar:
   On Error Resume Next

   For Each pinOut In fSrc.Pins
      If pinOut.Direction = 1 Then
         Exit For
      End If
   Next pinOut

   For Each pinIn In fGrab.Pins
      If pinIn.Direction = 0 Then
         pinOut.Connect pinIn
         Exit For
      End If
   Next pinIn

   For Each pinOut In fGrab.Pins
      If pinOut.Direction = 1 Then
         pinOut.Render
         Exit For
      End If
   Next pinOut

   gGraph.Run
End Sub
```

Embedded software source code (pulse width modulation, startup, analog to digital conversion) in C language:

```
#include <hc11.h>
#include <stdio.h>

#define TDRE 0x80
#define RDRF 0x20
#pragma interrupt_handler SCI_Handler
//#pragma interrupt_handler TOC2_Handler
//#pragma interrupt_handler TOC3_Handler
#pragma interrupt_handler RTI_Handler
#define DUMMY_ENTRY   (void (*)(void))0xFFFF

extern void _start(void);  /* entry point in crt.s */

void opensci(void);
void outsci(unsigned char Data);
void HexToAscii(unsigned char Data);
void readsci(void);
void init(void);
void delay(void);
void reset(void);
unsigned char Data;
int ascii1, ascii2, ascii3, hex;
int Receive;
static int Speed;
static int Forward;
static int Reverse;
static int Right;
static int Left;
static int Light;
static int Up;
static int Down;
static int Begin;
int pot;

void main()
{
      init();
      while(1)
      {
            pot = ADR1;
            HexToAscii(pot); //Seperate the Hexadecimal value
                             //and convert to Ascii value for SCI

                             //transmit
                                        outsci(0);

                                        Receive = '';
```

```
}

void init(void)
{

      INTR_ON();
      opensci();                  //Initilaize SCI

      OPTION = 0x80;              //Enable A/D Converter
      ADCTL = 0x30;               //Set A/D Converter Control
                                  //Parameteres

      OC1M = 0xF8;                //Initialize PWM
      OC1D = 0xF8;                //set PA6,PA5,PA4 high
      TOC1 = 0x0;
      TCTL1 = 0xAA;               //Set PA6,PA5,PA4 Low
      PORTB = 0x0;                //Set port B Parameters
      TOC2 = 1;
      TOC3 = 1;
      TOC4 = 1;
      TOC5 = 1;

}

void reset(void)
{

      TOC1 = 1;
      TOC2 = 1;
      TOC3 = 1;
      TOC4 = 1;
      TOC5 = 1;
      TCTL1 = 0x0;                //Set PA6,PA5,PA4 Low
      OC1M = 0x0;                 //Initialize PWM
      OC1D = 0x0;                 //set PA6,PA5,PA4 high
      PORTA = 0x0;


}

void opensci(void)
{
      setbaud(BAUD9600);         //Set Baud = 9600
      SCCR2 = 0x2C;              //Set SCI Options
    PORTD = 0x02;                //Set Data Direction for PortD SCI
}
```

```
void outsci(unsigned char Data)
{
     if(Data != 0)
     {
          while((SCSR & TDRE) == 0); //wait till TDRE bit is set
          SCDR = Data;                 //write to SCI
          return;

     }

     if(hex == 1)
     {
          while((SCSR & TDRE) == 0);//wait till TDRE bit is set
          SCDR = ' ';                //write to SCI blank ASCII
          while((SCSR & TDRE) == 0);//wait till TDRE bit is set
          if(ascii3 != '0')          //do not send the value if
                                     //the value is 0
          {
               SCDR = ascii3;        //write to SCI 100th
                                     //postion ASCII value

          }
          if(ascii2 == '0')       //if the ascii2 is 0 check to
                                  //see if ascii3 is zero
          {

               if(ascii3 != '0') //if ascii three is not zero
                                 //then send zero
               {

                    while((SCSR & TDRE) == 0);//wait here till
                                              //TDRE bit is set

                    SCDR = ascii2;    //write to SCI 10th postion
               }
          }
          if(ascii2 != '0') //if ascii2 is not zero send
          {
           while((SCSR & TDRE) == 0);//wait till TDRE bit is set
           CDR = ascii2;             //write to SCI 10th postion
          }

          while((SCSR & TDRE) == 0);//wait till TDRE bit is set
          SCDR = ascii1;          //write to SCI ones postion
```

```
                    hex = 2;

                }

            }
//Seperate the Hexadecimal value
//and convert to Ascii value for SCI transmit
void HexToAscii(unsigned char Data)
{
     int temp;
     int deci1, deci2, deci3;

     //Divide by 10 and store the remainder

     deci1 = Data % 10;

     //this will give the ones value
     temp = (Data-deci1) / 10; //Get the tens value
     deci2 = (temp % 10);
     temp = (temp-deci2) / 10; //get the hundreds value
     deci3 = temp % 10;

     ascii1 = deci1 + '0';          //add ascii 0 to the decimal

                                    //value to convert to

                                    //AScii value for SCi transmit
     ascii2 = deci2 + '0';
     ascii3 = deci3 + '0';
     hex = 1;                       //set the status to indicate

                                    //that a hex value is waiting

                                    //to be sent
}


void SCI_Handler(void)
{
      while((SCSR & TDRE)==0); //wait here till RDRF bit is set
     Receive = SCDR;

      if (Receive == '1')
      {
           Begin = 1;

      }
```

```
      if (Receive == '2')
 {

     Begin  = 2;
 }
 if (Receive == 'f')
 {

     //PORTB |= 0x40;
     //PORTB |= 0x80;
     PORTB = 0xFF;
while((SCSR & RDRF)==0); //wait here till RDRF bit is set
     Forward = SCDR;
     TOC2 = Forward * 254;
     TOC3 = Forward * 254;
     Receive = 0;


 }

 if (Receive == 'b')
 {
     //PORTB &= ~ 0x40;
     //PORTB &= ~ 0x80;
     PORTB = 0x0;
while((SCSR & RDRF)==0); //wait here till RDRF bit is set
     Reverse = SCDR;
     TOC2 = Reverse * 254;
     TOC3 = Reverse * 254;
     Receive = 0;


 }

 if (Receive == 'r')
 {

     PORTB |= 0x40;
     PORTB &= ~0x80;
while((SCSR & RDRF)==0); //wait here till RDRF bit is set
     Right = SCDR;
     TOC2 = Right * 254;
     TOC3 = Right * 254;
     Receive = 0;


 }

 if (Receive == 'l')
 {

     PORTB &= ~0x40;
```

```
        PORTB |= 0x80;
while((SCSR & RDRF)==0); //wait here till RDRF bit is set
        Left = SCDR;
        TOC2 = Left * 254;
        TOC3 = Left * 254;
        Receive = 0;


  }

  if (Receive == 'u')
  {

        //PORTB |= 0x20;
        PORTB = 0x0;
while((SCSR & RDRF)==0); //wait here till RDRF bit is set
        Up = SCDR;
        TOC4 = Up * 254;
        Receive = 0;

  }

  if (Receive == 'd')
  {

        //PORTB |= ~0x10;
        PORTB = 0xFF;
while((SCSR & RDRF)==0); //wait here till RDRF bit is set
        Down = SCDR;
        TOC4 = Down * 254;
        Receive = 0;

  }

  if (Receive ==  'p')
  {

        //PORTB &= ~0x10;
        PORTB = 0x0;
while((SCSR & RDRF)==0); //wait here till RDRF bit is set
        Light = SCDR;
        TOC5 = Light * 254;
        Receive = 0;



  }

}
```

```
void RTI_Handler(void)
{

      while((SCSR & TDRE) == 0);//wait here till TDRE bit is set
    SCDR = '1';                  //write to SCI ones postion
}
void delay(void)
{
      int temp;
      for (i =0; i<10; i++)
      {
        temp = temp + 1;

      }
}
//read SCI input
void readsci(void)
{
      while((SCSR & RDRF)==0); //wait till RDRF bit is set
      Receive = SCDR;
}


#pragma abs_address:0xffd6

void (*interrupt_vectors[])(void) =
      {
      SCI_Handler,     /* SCI */
      DUMMY_ENTRY,     /* SPI */
      DUMMY_ENTRY,     /* PAIE */
      DUMMY_ENTRY,     /* PAO */
      DUMMY_ENTRY,     /* TOF */
      DUMMY_ENTRY,     /* TOC5 */
      DUMMY_ENTRY,     /* TOC4 */
      DUMMY_ENTRY,     /* TOC3 */
      DUMMY_ENTRY,     /* TOC2 */
      DUMMY_ENTRY,     /* TOC1 */
      DUMMY_ENTRY,     /* TIC3 */
      DUMMY_ENTRY,     /* TIC2 */
      DUMMY_ENTRY,     /* TIC1 */
      RTI_Handler,     /* RTI */
      DUMMY_ENTRY,     /* IRQ */
      DUMMY_ENTRY,     /* XIRQ */
      DUMMY_ENTRY,     /* SWI */
      DUMMY_ENTRY,     /* ILLOP */
      DUMMY_ENTRY,     /* COP */
      DUMMY_ENTRY,     /* CLM */
      _start     /* RESET */
      };
 #pragma end_abs_address
```

Graphical user interface source code listing:

Thruster control.frm

```
Private Sub Form_Unload(Cancel As Integer)
ClipCursor ByVal 0&
End Sub

Private Sub Joystick1_JoyMove()
Text1.Text = Int(Joystick1.XPos)
Text2.Text = Int(Joystick1.YPos)
If Joystick1.XPos > Joystick1.YPos Then
perx = Abs(Int(Joystick1.XPos)) / 100
pery = Abs(Int(Joystick1.XPos)) / 100
Else
perx = Abs(Int(Joystick1.YPos)) / 100
pery = Abs(Int(Joystick1.YPos)) / 100
End If
perz = Abs(Int(Joystick1.ZPos)) / 100
PWM.draw
End Sub

Private Sub Joystick1_MouseDown(Button As Integer, Shift As Integer, x As Single, y As Single)
If Button = 2 Then
   ClipCursor ByVal 0&
End If
If Button = 1 Then
   Dim lcClip As RECT
   GetWindowRect ThrusterControl.hwnd, lcClip
   ClipCursor lcClip
End If
End Sub
```

```
Private Sub Joystick1_MouseUp(Button As Integer, Shift As Integer, x As Single, y As Single)
Joystick1.Center 1
Joystick1.Center 2
Text1.Text = Joystick1.XPos
Text2.Text = Joystick1.YPos
End Sub
```

MDIfrm.mdi

```
Private Sub Exit_Click(Index As Integer)
Unload Me
End Sub

Private Sub MDIForm_Load()
PWM.Show
PWM.Left = 400
PWM.Top = MDIForm1.Width * 0.4
Sensor.Show
Sensor.Left = MDIForm1.Height * 0.97
Sensor.Top = 0
ThrusterControl.Show
ThrusterControl.Top = MDIForm1.Width * 0.35
ThrusterControl.Left = MDIForm1.Height * 0.97
Video.Show
Video.Left = 400
Video.Top = 0
clip
End Sub
```

```
Private Sub Pwmout_Click(Index As Integer)
PWM.Show
PWM.Left = 400
PWM.Top = MDIForm1.Width * 0.4
End Sub

Private Sub Sensorout_Click(Index As Integer)
Sensor.Show
Sensor.Left = MDIForm1.Height * 0.97
Sensor.Top = 0
End Sub

Private Sub Thrusterout_Click(Index As Integer)
ThrusterControl.Show
ThrusterControl.Top = MDIForm1.Width * 0.35
ThrusterControl.Left = MDIForm1.Height * 0.97
End Sub

Private Sub Videoout_Click(Index As Integer)
Video.Show
Video.Left = 400
Video.Top = 0
End Sub

Function clip()
Dim rcClip As RECT
GetWindowRect ThrusterControl.hwnd, rcClip
SetCursorPos ThrusterControl.ScaleWidth / 2, ThrusterControl.ScaleHeight / 2
ClipCursor rcClip
ThrusterControl.MousePointer = 2
End Function
```

PWM.frm

```
Private Sub Form_Load()
c = 0
perx = 0
pery = 0
perz = 0
End Sub

Private Sub HScroll1_Scroll()
per = HScroll1.Value / 100
Picture1.Cls
Picture2.Cls
Picture3.Cls
draw
End Sub

Private Sub Timer1_Timer()
c = Not c
If c = 0 Then
draw
Timer1.Interval = Timer1.Interval / 2
Else
Timer1.Interval = 1000
'Picture1.Cls
End If
End Sub

Function draw()
PWM.Picture1.Cls
PWM.Picture2.Cls
PWM.Picture3.Cls
```

```
Picture1.Line (0, 100)-(255 * perx, 100), RGB(0, 0, 255)
Picture1.Line (255 * perx, 200)-(255 * 1, 200), RGB(0, 0, 255)
Picture1.Line (255 * perx, 100)-(255 * perx, 200), RGB(0, 0, 255)
Picture1.Print Int(perx * 100) & "%" & "  High"

Picture2.Line (0, 100)-(255 * pery, 100), RGB(0, 0, 255)
Picture2.Line (255 * pery, 200)-(255 * 1, 200), RGB(0, 0, 255)
Picture2.Line (255 * pery, 100)-(255 * pery, 200), RGB(0, 0, 255)
Picture2.Print Int(pery * 100) & "%" & "  High"

Picture3.Line (0, 100)-(255 * perz, 100), RGB(0, 0, 255)
Picture3.Line (255 * perz, 200)-(255 * 1, 200), RGB(0, 0, 255)
Picture3.Line (255 * perz, 100)-(255 * perz, 200), RGB(0, 0, 255)
Picture3.Print Int(perz * 100) & "%" & "  High"

End Function
```

Sesor.frm

```
Private Sub Form_Activate()
DrawCompass
End Sub

Function DrawCompass()
Picture1.Cls
cx = Picture1.ScaleWidth / 2
cy = Picture1.ScaleHeight / 2
Picture1.CurrentX = 0
Picture1.CurrentY = cy

cx = Picture1.ScaleWidth / 2
cy = Picture1.ScaleHeight / 2
```

```
Picture1.Circle (cx, cy), 4
Picture1.CurrentX = cx + 4.25
Picture1.CurrentY = cy
Picture1.FontBold = True
Picture1.Print ("E")
Picture1.CurrentX = cx - 4.6
Picture1.CurrentY = cy
Picture1.Print ("W")
Picture1.CurrentX = cx
Picture1.CurrentY = cy - 4.7
Picture1.Print ("N")
Picture1.CurrentX = cx
Picture1.CurrentY = cy + 4.4
Picture1.Print ("S")
Picture1.FontBold = False
Picture1.Line (cx, cy)-(cx, 1), RGB(255, 0, 0)
End Function
```

VITA

Surname: Ramaswamy Given Name: Harish

Place of Birth: Tirunelveli, India Date of Birth: 21-09-1975

Certifications Attained:

Microsoft Certified Programmer
Microsoft Certified Systems Engineer
Cisco Trained Network Associate

Degree Awarded:

B. Eng.	Bangalore University	September 1998

PARTIAL COPYWRITE LICENSE

I hereby grant the right to lend my thesis (or dissertation) to users of the University of Victoria Library, and to make single copies only for such users or in response to a request from the Library of any other university, or similar institution, on its behalf or for one of its users. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by me or a member of the University designated by me. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Title of Thesis/Dissertation: Design of Tetherless Remotely Operated Underwater Vehicle.

Author: [redacted]
(Signature)

Harish Ramaswamy
(Name)

28/09/2002
(Date)