CODEN:LUTEDX/(TEIE-5265)/1-39/(2009)

Industrial Electrical Engineering and Automation

Wireless Control via Bluetooth

Ásta Andrésdóttir

Division of Industrial Electrical Engineering and Automation
Faculty of Engineering, Lund University

Abstract

A personal transporter is under development at LTH, Industrial Electrical Engineering and Automation, IEA. A wireless link is desired to develop control remotely for the transporter. The development of the control becomes easier without wires and if the transporter can move freely within the range of the link.

A link is built, using Bluetooth devices and a microprocessor from Microchip. This link is able to carry an eight bit signal back and forth. The development of the transporter has not reached a state in which it can receive or transmit signals. Instead the link is tested with a signal generator and an oscilloscope at first and then for DC-motor control.

A test run is carried out with a signal generator and an oscilloscope. A signal is created with the signal generator, which is fed to the microcontroller that forwards the signal by a Bluetooth transceiver. Another Bluetooth transceiver receives the signal at the remote site. The signal is forwarded to a computer hosted control system, dSPACE, and a control signal is produced. The control signal is sent back to the microprocessor by the Bluetooth transceivers and forwarded to an oscilloscope via a D/A converter.

The link is furthermore used to control a DC motor. A PWM signal is created using the microcontroller and forwarded to the motor via an H-bridge, an electronic circuit that controls the direction and velocity of the DC motor. A potentiometer is used to sense the location of the motor. The voltage from the potentiometer is forwarded to the microcontroller that converts it to a digital signal. The digital signal for the measurement is sent through the Bluetooth transceivers to dSPACE where the desired control signal is calculated. The control signal is sent to the microcontroller that generates the corresponding PWM signal.

An 8 bit signal can be sent and received with the wireless Bluetooth link. Some problems occurred with buffer overflows, which were eliminated by tuning the sampling of signals. The range of the wireless link is close to 40 m and should be enough to develop control for the transporter. Depending on the amount of signals, that affect the control of the transporter, a multichannel link might be desirable.

Table of Contents

Contents

List of Figures

Preface

The work in this thesis was carried out at LTH, Industrial Electrical Engineering and Automation, IEA, under the guidance of Gunnar Lindstedt. The work began in the fall 2008, a maternity leave was taken during the following spring and summer and then the work was completed in November 2009.
I would like to thank...

- Gunnar Lindstedt for feedback and advice
- Getachew Darge for his help
- Johan Björnstedt for the introduction to dSPACE
- Ámundi and our girls for inspiration
- Family and friends for support and good advice when needed

1 Introduction

1.1 Background

In 2007, two students at LTH, IEA began building a personal transporter (Sandberg, 2007). The project of building and controlling the transporter has not been completed, there are some tasks yet to be fulfilled. One of these tasks is to set up wireless transmission between the transporter and a controlling computer, which is the project described in this report. The wireless system is a development tool for control algorithms and will not be a part of the final vehicle.

1.2 Objectives

The objective is to create a wireless link between the transporter and a computer. A microcontroller is connected to the transporter, which forwards measurements from sensors to dSPACE with the help of Bluetooth transceivers. Control signals are calculated within the dSPACE control system and sent back the same way to control the transporter.

It is desirable to develop control for the transporter with dSPACE and then create code so that it can be run from a microcontroller located at the transporter.

1.3 Scope

The scope of this project is to set up a Bluetooth link between a computer with dSPACE and the transporter's electronic part. The project includes programming a microcontroller and setting up Bluetooth chips, adapters and converters. A simple control algorithm is implemented and used to calculate control signals. The communication link between dSPACE and the transporter is however the main task.

The project is limited because the transporter is not fully operational and the control can therefore not be evaluated as it should be. The transmissions back and forth can be verified and tested using a signal generator and an oscilloscope. Control is applied to a DC motor to verify that wireless control can be carried out via the link.

2 Theory

2.1 Personal Transporters

2.1.1 The Segway

The Segway personal transporter, the first self-balancing transportation device, was first introduced to the public in December 2001. The Segway company was founded by the inventor Dean Kamen nearly three years earlier with the goal of creating a transporter. The name of the transporter, and the company, comes from the word *segue* which means "to transition smoothly from one state to another" (Segway, 2008c).

Figure 1: A Segway personal transporter, model i2 (Segway, 2008a).

A Segway personal transporter has two wheels, a footplate, shaft and handle as seen in Figure 1. The driver stands on the footplate and steers the transporter with the movements of his/her own body. To move backwards or forwards the driver just leans in the direction that he/she wants to travel and to slow down the driver moves back to the vertical position and then eventually comes to a complete stop. To turn left or right the steer frame is moved in that direction (Segway, 2008b).

The Segway is equipped with five gyroscopes and two accelerometers that sense the terrain traveled on and the position of the driver's body at a frequency of 100 times per second (Segway, 2008d).

2.1.2 Our Transporter

At IEA the building of a personal transporter is in progress. The hardware has been constructed but the electronic part is not ready and control has to be implemented. Wireless control is one of the task yet to be carried out. This project will therefore contribute to the ongoing work. When the work on the transporter is

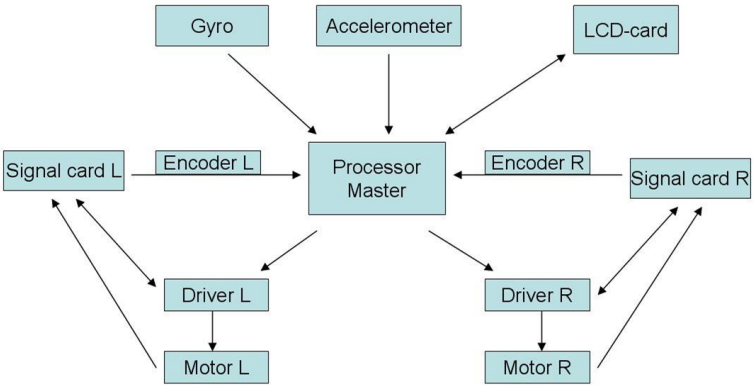


Figure 2: The signal flow in the transporter (Sandberg 2007 p. 5)

completed it will be self sufficient and maintain control using a master processor located on board. The signal flow in the transporter is shown in Figure 2. The two sensors, gyroscope and accelerometer, measure angular momentum and angular velocity, respectively, and the signal cards provide feedback from the motors that the controller uses to calculate the next control signal. Before this control is implemented it will be tested and developed from a remote computer with the help of wireless communication. The wireless communication will, however, not be used in the final version of the transporter because an on board processor will handle the control.

2.2 Wireless Transmissions

Wireless transmissions can for example be based on radio waves, microwaves or infrared waves. The choice should depend on the characteristics of the methods and how they fit the intended purpose of the application in question.

2.2.1 Radio Waves

Radio waves are electromagnetic waves ranging in frequency between 3 kHz and 1 GHz. Radio waves are mostly omnidirectional, that is they propagate evenly in all directions. The sender and the receiver do not have to be aligned in order to exchange signals. A sender can transmit to one or more receivers and the signals can be received by antennas that are built for the same wavelength and are located within the transmission range. A disadvantage is that other signals sent on the same frequency can easily interfere with the transmitted signal. Radio waves can travel through substances and therefore line-of-sight is not needed for communication. Permission is needed when signals are sent with radio waves. Radio waves are for example used when broadcasting audio and sound, such as in television and radio (Forouzan, 2003 (p. 186-188)).

2.2.2 Microwaves

Microwaves operate in the frequency interval between 1 and 300 GHz. Microwaves are unidirectional, they do not propagate evenly in all directions. The sending and receiving antennas have to be aligned in order to exchange signals. This means that there is less chance of disturbance from other sources but also that line-of-sight is needed. Therefore the antennas are often located on the top of buildings or high towers. The higher frequency band of microwaves can not travel through substances. A certain portion of the band requires permission from authorities before it is used. Microwaves are for example used in cellular phones, satellite networks and wireless LANs (Forouzan, 2003 (p. 188-189)).

2.2.3 Infrared

Infrared signals, with frequencies from 300 GHz to 400 THz can be used for short range communication. Infrared signals can not travel through substances and they require line-of-sight connection in order to operate. Infrared waves are used for short-range communication using line-of-sight propagation, for example in remote controls (Forouzan, 2003 (p. 189-190)).

2.3 Protocols and Standards

2.3.1 Bluetooth

Bluetooth was invented in 1994 by Sven Mattisson at L. M. Ericsson of Sweden. It is named after Harald Blaatand II who was the king of Denmark in the years 940-981. A special interest group for Bluetooth, SIG, was formed in 1998. Some major companies in the telecommunications branch, such as Ericsson, IBM, Intel, Nokia and Toshiba, founded the group and later many other companies joined. The group developed an open specification for short-range wireless connectivity (Sairam, 2002 (p. 90)).

Bluetooth operates at the unlicensed 2.4 GHz industrial, scientific and medical frequency band, ISM. Household appliances, such as microwave-ovens, baby monitors and cordless phones, use the same band and therefore some disturbance might be expected. Spread-spectrum frequency hopping, SSFH, is used to minimize disturbances. The transmitters change frequency 1600 times per second, which means that each time slot is 625 micro seconds (McDermott-Wells, 2004 (p. 34)). This reduces the risk of interference and if an interference occurs the disturbance time is short.

The operating range of Bluetooth equipment is usually around 10 meters, which can be increased by amplifiers. Bluetooth operates at radio frequency and the waves can travel through substances, which means that line-of-sight is not needed when Bluetooth technology is utilized. The transmission rate is about 780 kb/s, less if used symmetrically (Sairam, 2002 (p. 91)).

2.3.2 RS232

The standard is named EIA/TIA-232E, developed by the Electronic Industry Association and the Telecommunications Industry Association, but it is often referred to as RS232 where RS stands for recommended standard.

The original standard was defined in 1962 and has been updated several times. A high voltage level is defined as being +5 to +15 volts and a low level is defined as being between -5 and -15 volts at the transmitting side. At the receiving side a 2 volt noise margin is provided and the levels become +3 to +15 volts and -3 to -15 volts. The low level is defined as a logic one and referred to as marking and the high level is defined as logic null and referred to as spacing.

Originally the length of the cable carrying the signal was limited to 15 meters, but the standard has been changed and instead the capacitance of the cable is now limited to 2500 pF. The maximum cable length can be calculated from the capacitance per unit length of the cable (Dallas, 1998 (p. 1-8)).

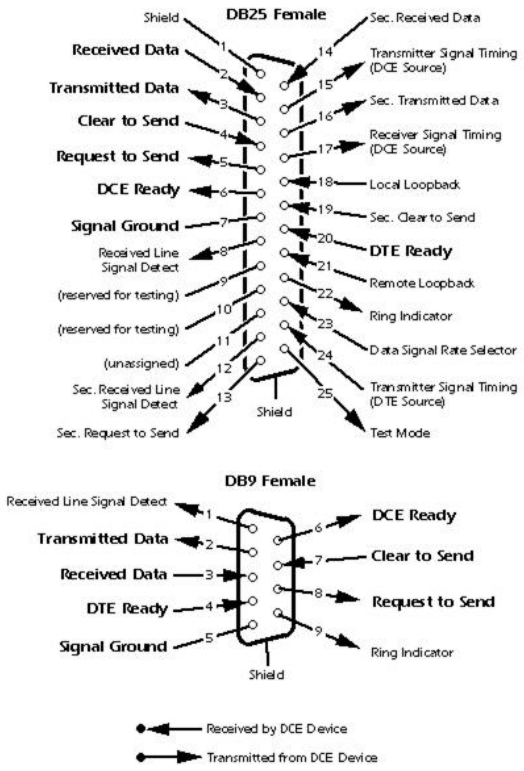


Figure 3: Pinout for 25 and 9 pins connectors (Strangio, 2006).

The RS232 standard specifies a 25 pin connector, but because many applications do not use all of the signals specified, versions with 15 or 9 pins can be used instead. The 9 pin connector is enough to transmit and receive the necessary signals for modem applications. A comparison of signals sent in a 25 and 9 pin connector is shown in Figure 3. The 9 pin connector is used in this project.

2.3.3 AT Commands

AT comes from the word attention. The original AT commands were defined by Dennis Hayes, his commands all began with the letters AT and therefore the naming. These were used with modems and made it possible to use only one communication channel by shifting between data mode and command mode, otherwise two channels were needed. The command language has grown and now includes commands for data, fax, voice and SMS communications (Bies, 2008b).

Even though the command set used for the Bluetooth chip in this project is similar to other AT command sets, it is implemented especially for EZURiOs Bluetooth devices (Ezurio, 2006 (p. 4)).

2.3.4 UART

UART stands for Universal Asynchronous Receiver/Transmitter. With UART transmitters, bytes of data are changed from parallel data to serial data that can be sent on a communication line. A receiving UART collects bits of serial data and outputs a whole byte of parallel data. UART handles tasks like timing and parity checking. The communication speed can vary; the maximum allowed communication speed is 115200 bauds, symbols per second. To change the speed this value can be decreased by dividing it by a programmable value (Bies, 2008a).

Parameters that can be altered when configuring a UART transmission are the communication speed, number of data bits, the type of parity check and the number of stop bits sent.

The PIC, the Bluetooth chips and dSPACE are all equipped with a UART.

2.4 The Microcontroller

The controller used in this project is PIC16F73 from Microchip referred to as PIC in this report. The PIC has the operating frequency of 20MHz. It can be programmed using the C programming language - the instruction set in the data sheet is however given in assembly language that can also be used. Various development tools are available that can be used with the PIC, such as a compilers and editors.

The UART port on the PIC is used to connect to the Bluetooth chip. Incoming measurements arrive through the on-chip A/D converter and digital ports are used to forward control signals to the transporter via a D/A converter. When the DC-motor control is applied the PWM output is used instead of the digital output and the D/A converter.

2.4.1 Communication with Bluetooth

The communication with the Bluetooth transceiver is done through the UART port on the PIC. The UART communication is in full duplex mode; that is information is sent and received through the UART port on the PIC.

The UART consists of four elements; a baud rate generator, a sampling circuit, an asynchronous transmitter and an asynchronous receiver. An on-chip 8 bit baud rate generator is used to derive standard baud rate frequency from an oscillator. The sampling circuit samples the input three times and a majority check determines if the value should be high or low (Microchip, 2002 (p. 71-73)).

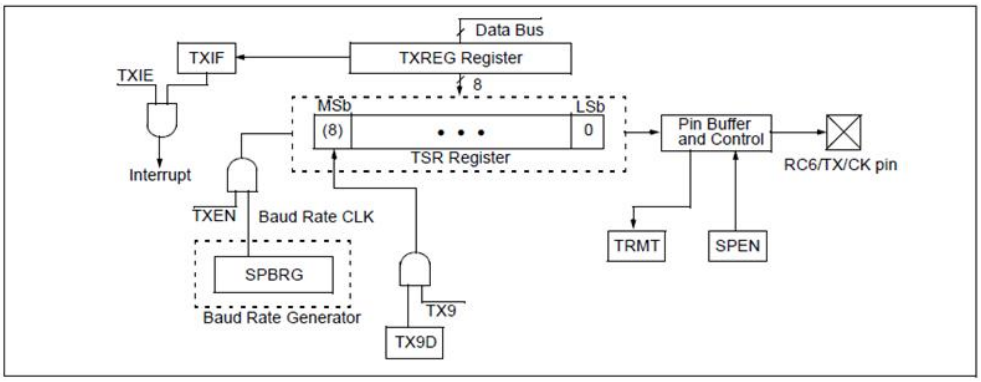


Figure 4: A block diagram of the transmitter (Microchip, 2002 (p. 73)).

The operation of the asynchronous transmitter is shown in Figure 4. A transmit buffer, TXREG, is used to insert data from software. The data to be sent is forwarded from TXREG and loaded in the transmit shift register, TSR. During

transmission the data is further forwarded to the TX output-pin with the least significant bit first. When the last bit, the stop bit, has been transmitted the new content of TXREG is loaded into the TSR. One instruction cycle later a flag, TXIF, is set and TXREG can be reloaded. The flag is cleared when TXREG has been loaded again. The flag can be used to determine whether the data buffer is empty or full. Another flag, TMRT, shows the status of the TSR. To enable transmission the TXEN bit has to be set. Transmission takes place after TXEN has been set, TXREG has been loaded and a shift clock has been produced by the baud rate generator. Usually the TSR is empty when transmissions begin (Microchip, 2002 (p. 73-74)).

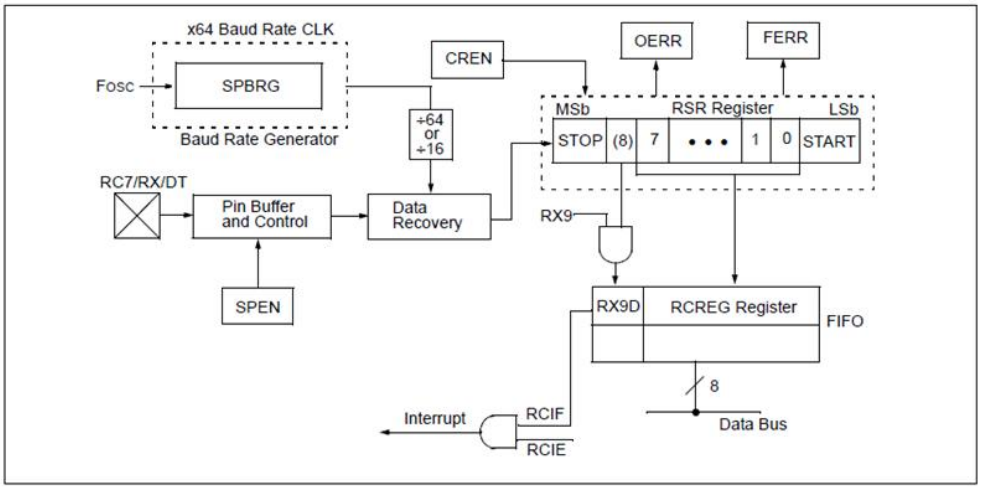


Figure 5: A block diagram of the receiver (Microchip, 2002 (p. 75)).

The operation of the asynchronous receiver is shown in Figure 5. The data is received through the RX input-pin. The data recovery block operates at 64 times the baud rate but the receive shift register, RSR, works at baud rate. After the stop bit arrives at the RSR the data is sent to the receive register, RCREG, and the flag bit, RCIF, is set. When RCREG has been read and emptied RCIF is cleared. The RCREG can carry two bytes of data and the third byte can begin to shift in the RSR. When the third byte is sent from the RSR to RCREG an overrun will occur. A flag bit, OERR, is used to detect overrun errors. A framing error, FERR, is detected when the stop bit is clear instead of set. To clear overrun and framing errors the enable bit, CREN, is cleared and then set again. CREN has to be set to enable reception (Microchip, 2002 (p. 75-76)).

2.4.2 Analog Input

The PIC is equipped with 5 analog inputs and one 8 bit A/D converter as shown in Figure 6. The on-chip A/D converter works with one input at a time, the input channel is selected before conversions.

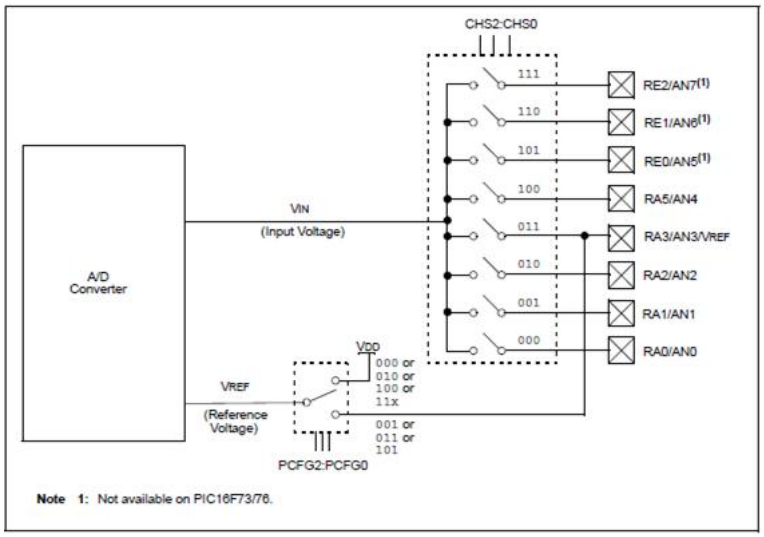


Figure 6: A block diagram of the analog input (Microchip, 2002 (p. 85)).

A capacitor is charged to the voltage level of the input channel. It is important to give the capacitor sufficient time to charge so that the analog input will be accurate. When the enable bit, ADGO, is set the A/D conversion begins. When the conversion is complete the ADGO bit is cleared, the ADIF flag is set and the 8 bit digital number can be found in the result register, ADRES. The ADGO bit can be set again. The reference voltage can be the same as the supply voltage for the PIC or set on a defined input pin that is otherwise used as an analog input (Microchip, 2002 (p. 83-87)).

2.4.3 Analog Output

The PIC does not have an analog output, instead a digital output and a D/A converter are used. The D/A converter used is AD7303 from Analog Devices. The D/A converter receives three signals from the PIC; one for data, another one for a clock signal and the third one to enable data input.

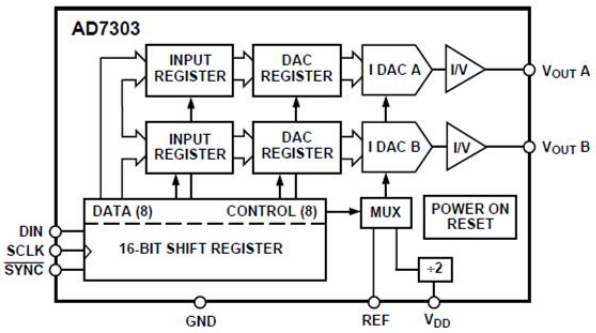


Figure 7: A block diagram of the D/A converter (Analog, 1997 (p. 1)).

The function of the converter is shown in Figure 7. The input shift register is enabled when SYNC is cleared, then data bits are transferred via the data pin, DIN, on the rising edges of the serial clock, SCLK. The shift register contains 16 bits, 8 bits are for the control of the D/A converter and the other 8 bits are the data bits that will be converted. The voltage reference can be the same as used for the chip or set at an input, REF. The output channel is chosen with the control bits, either A or B is used (Analog, 1997 (p. 1-4)).

2.4.4 Pulse Width Modulation

The PIC has a pulse width modulation, PWM, module. The module is used to create output when controlling a DC motor.

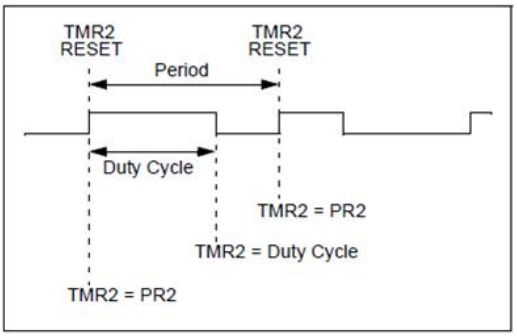


Figure 8: PWM signal (Microchip, 2002 (p. 57)).

A timing diagram of a PWM signal is shown in Figure 8. The duty cycle can vary between 0% and 100% of the period. The full voltage is seen at 100% and none at 0%. The PMW frequency is the reciprocal of the period.

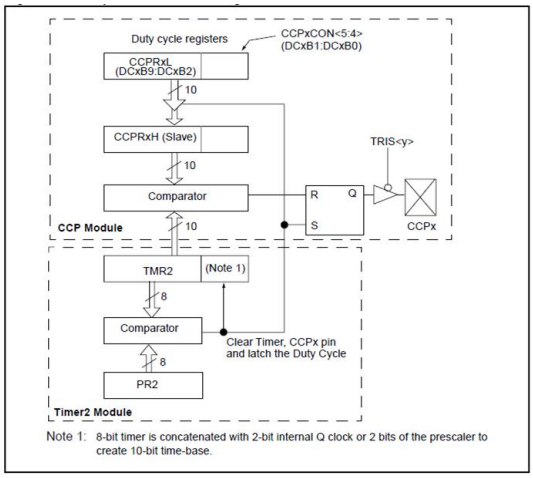


Figure 9: PWM block diagram (Microchip, 1997 (p. 14-8)).

The operation of the PWM module is shown in Figure 9. One of the timers in the PIC, TMR2, is used together with the PWM module. The desired PWM period is written to register PR2. The value for the duty cycle is written to the CCPR1L register and bits 4 and 5 of the CCP1CON register, 10 bits in total. It is forwarded to CCPR1H and a 2-bit internal latch when PR2 and TMR2 become equal. At this time TMR2 is cleared and the CCP1 output-pin is set. When TMR2 is equal to the contents of CCPR1H and the 2-bit latch the CCP1 output-pin is cleared (Microchip, 1997 (p. 14-8–14-11)).

A multifunction PIC board that is available at the department is used in this project. This board consists of a PIC, D/A converters, LCD screen and other integrated circuits that can be used with the PIC. A diagram for this board is shown in Appendix C.

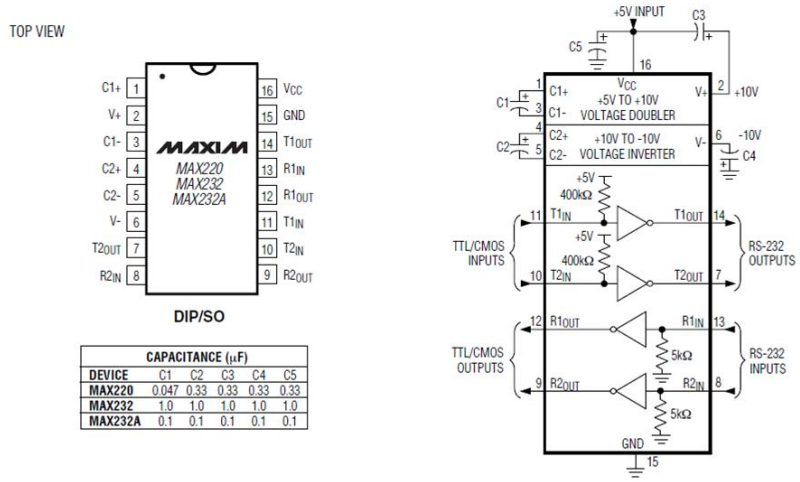
2.5 The Bluetooth Chip

The chips used for Bluetooth communication are the BISM2 Bluetooth Version 2.0 Serial Modules from EZURiO. The range of the chip is listed as 300 m, and the transmission speed up to 300 kb/s (Ezurio, 2009). One chip is located at the computer and another one at the transporter. Each Bluetooth chip has a unique address, a 12 digit hexadecimal string that is used to identify it. The chip is set up and controlled with AT-commands. The chips can operate in three modes; *data mode* to send information, *local command mode* to set up and control the chip or *remote command mode* to receive commands through a Bluetooth connection (Ezurio, 2006 (p. 4)).

This chip is used because it has been successfully used in another wireless project at the department.

2.6 MAX232

Maxim's MAX232 chip is used as a link between the Bluetooth chip and the RS232 interface. The RS232 interface operates with voltage in the range of -10 to 10 volts but the Bluetooth chip operates at 0 to 5 volts. The MAX232 chip translates the signals between the Bluetooth chip and the RS232 communication interface. The definition of high and low is reversed and the voltage is altered.



CAPACITANCE (μF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	0.047	0.33	0.33	0.33	0.33
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Figure 10: Pinout and functional diagram for MAX232 (Maxim, 2006 (p. 17)).

The MAX232 consists of three sections: dual charge-pump DC-DC voltage converters, RS232 drivers and RS232 receivers (Maxim, 2006 (p. 14)). The pinout and the function of MAX232 can be viewed in Figure 10.

2.7 dSPACE

dSPACE is a development tool, which consists of both a hardware and a software part. The software used in this project is ControlDesk where signals can be viewed and parameters changed. The hardware, DS1104 R&D controller board, is a digital signal processor that translates incoming and outgoing signals in real time. The board is connected to the PC via a PCI bus. Systems can be implemented in C and run in real time (dSPACE, 2009 (p. 332)).

A model of the system is constructed in Simulink with the help of Real-Time Interface, RTI, and then translated into C programming code for the ControlDesk in dSPACE. All improvements of the controller are carried out in Simulink.

2.8 Other Tools

Some other tools are used in the project and are mentioned in the report. A short explanation follows for these.

- Hyper Terminal is a terminal emulation program that is run at the computer. It is used to connect to devices via COM ports.
- All-11 is a programming device from Hi Lo systems. It is used to translate incoming code to voltages so that the PIC can be programmed.
- Waccess is the software used for programming the PIC. It communicates with All-11.
- MPLAB IDE, software from Microchip. It is used to write and compile C code for the PIC.
- Simulink is a program within Matlab. It is used to design the system using existing customizable blocks.
- Real-Time Interface, RTI, supplies Simulink with input and output blocks that can be used with dSPACE.
- DC-motor, a 12 V brushed direct current motor is used in the project.
- H-bridge, a circuit to regulate and amplify the control signals for the DC-motor.
- Gyroscope, used to measure angular momentum.
- Accelerometer, used to measure angular velocity.

3 Implementation

3.1 Connections

The connection of the equipment is shown in Figure 11. The Bluetooth transceiver is connected to a dSPACE controller board via a serial connection. The serial RS232 connector has to be translated to be able to communicate with the Bluetooth transceiver. A MAX232 adapter is used to do this. The information travels through air to the other Bluetooth transceiver and onwards to a PIC-microcontroller via a MAX232 adapter. The measurements from the transporter are received via an analog input of the PIC and a D/A converter is used to translate the digital control signal to an analog signal for the transporter.

Figure 11: The connection between dSPACE and the transporter

3.1.1 Verification of Connections

The work begins with the setup of the MAX232 adapter chip. The signal conversion is verified with measurements using a voltmeter; these are compared to the data sheet and proved to be right. Afterwards the chip is connected to a computer via the serial interface. Characters are sent from the Hyper Terminal at the computer and the translation is verified by viewing bits on an oscilloscope and comparing them to the ASCI code for the sent character.

The next step is to connect the Bluetooth chip to the remote side of the MAX232 chip. Transmission are still controlled by the Hyper Terminal but now AT commands are sent to the Bluetooth chip. The Bluetooth chip is made visible and connectable. For verification a connection with a Bluetooth enabled mobile phone is made, this connection is not used in the project.

The PIC is now configured and connected. The PIC board is equipped with an LCD screen, which is used to display characters. This screen is very useful when verifying the operation of the PIC. At first the connection between the computer and the PIC is verified, characters sent from the computer are displayed at the

screen successfully. Later on the computer is unplugged and instead the Bluetooth chip is connected to the PIC via the MAX232 adapter. Preprogrammed AT commands are sent from the PIC to the Bluetooth chip that becomes visible and connectible. For further verification a character is displayed at the LCD screen at certain steps in the programming.

A connection is then set up between the two Bluetooth transceivers. This turned out to be quite straight forward and the chips are configured so that they automatically connect at power up.

The final part of the connection process is to communicate with dSPACE. One of the Bluetooth chips is connected to dSPACE and after some tuning of parameters communication is established with dSPACE.

3.2 The Bluetooth Chips

The Bluetooth chips are controlled by AT commands. Hyper Terminal is used to communicate with the chips. After a command is given at Hyper Terminal, the Bluetooth chip replies with some preprogrammed feedback like OK or ERROR. The commands are easily understood and well explained in a listing provided on EZURiO's website. The chips are programmed to connect at start up.

One of the chips is the master and the other one a slave. In this setup it does not matter which transceiver is the slave and which one is the master. By default the master is connectable but not discoverable, that is it can only be connected to by equipment that already have its address. The same settings were chosen for the slave. When the Bluetooth chips are turned on the master looks for the slave and a connection is made. If the connection can not be made the master tries to connect to the slave again after a waiting period of 2 sec. This time is set by AT commands. When a connection has been made all the information that is received on the transceivers UART ports is forwarded to the other transceiver. This is the same for both the chips.

Explanation of AT commands can be found in Appendix A.

3.3 The PIC

The PIC is programmed using the C programming language. The C code is compiled and the output is stored in a HEX file that can be translated with Waccess software and then programmed to the PIC with an All-11 programmer.

The code is divided in five separate C-files. One file includes the main function, the second file contains functions for the UART port, the third file functions for the A/D and D/A conversion, the fourth file functions for PWM and the fifth file functions for the LCD screen. The main function is run when the PIC is turned on and other functions are activated from within the main function. The code could have been written in one file, however, the division is done to have

better overview of the programming.

In the main function the setup of different ports and values is carried out and then a while loop is repeated until the equipment is turned off. In this loop the interrupt flags for incoming UART signal and incoming analog signal are checked.

If a UART signal is detected, two functions are started. The first function retrieves the incoming control signal. If the output should be an analog signal the second function forwards a digital output and conversion control values to the D/A converter. If the output should be a PWM signal the second function determines the direction of which the motor should rotate and forwards a value to the PMW module. The analog conversion (ADGO) is set at the end of this loop.

If an analog signal is detected, the first function retrieves the signal and the second one forwards it to the outgoing UART port. The sampling of the analog signal can be continuous, but in order to limit the sampling speed to the transmission speed the A/D conversion is turned on after a UART signal has been detected.

The code for the PIC can be found in Appendix B.

3.4 Working with dSPACE

The Control Desk in dSPACE is used to view the signals. A model is built in Matlab Simulink and translated to C-code to work with dSPACE.

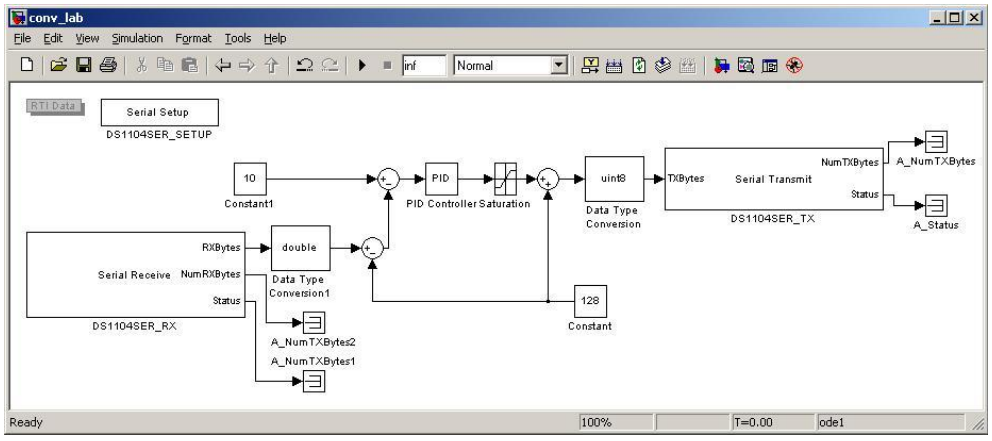


Figure 12: The layout in Simulink.

The layout for the motor control in Simulink is shown in Figure 12. Most of the blocks are ordinary Simulink blocks but three of the blocks come from the RTI. These are the blocks that represent the UART. One represents the received signal,

one the transmitted signal and the third one is used to control the transmission speed and other settings for UART. These blocks require an input of the form unsigned 8 bit integer, uint8.

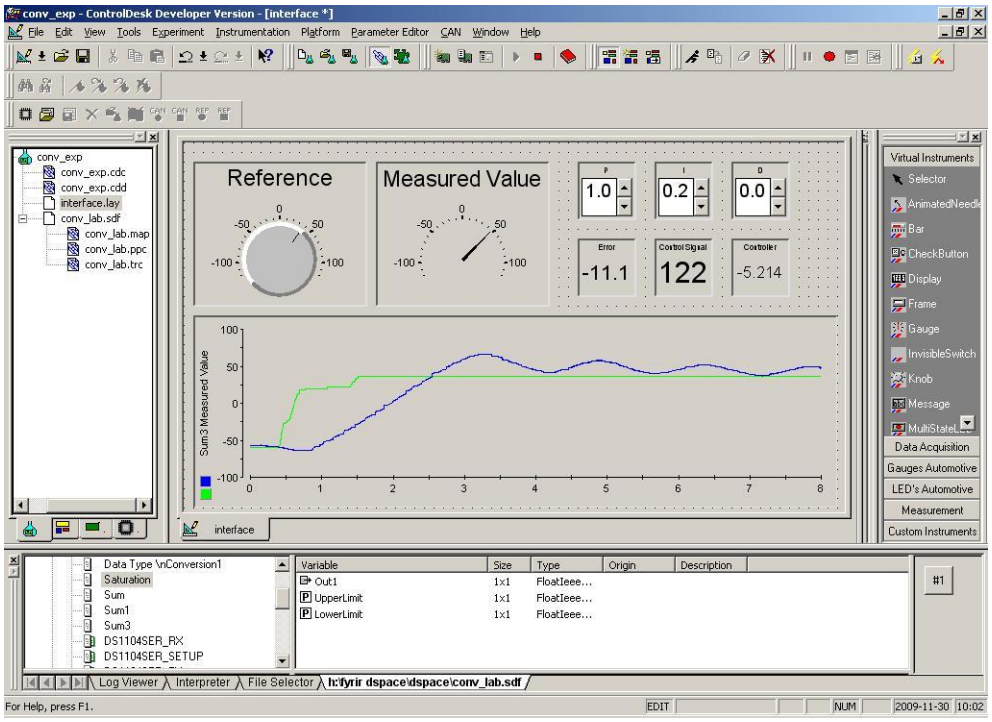


Figure 13: The layout in dSPACE.

The setup for the motor control in dSPACE is shown in Figure 13. The reference is set with the button at the upper left corner of the layout. The measurement from the motor is viewed at the meter that is located besides the button. The parameters for the PID controller can be changed and viewed in the upper right corner. The error value, the output from the PID controller and the control signal are shown below the PID parameters. The graph displays the reference value and the measurements from the motor.

Below the layout figure a listing of variables is found. Parameters marked with P can be changed at any time in dSPACE. Other variables are incoming signals or outcomes of functions that can be viewed.

4 Conclusions

The trial of the system is twofold. To begin with the settings that will be used for the transporter are tested with the help of a signal generator and a oscilloscope, furthermore the control abilities of the system are tested by running a motor. Figure 14 displays the set up for both options.

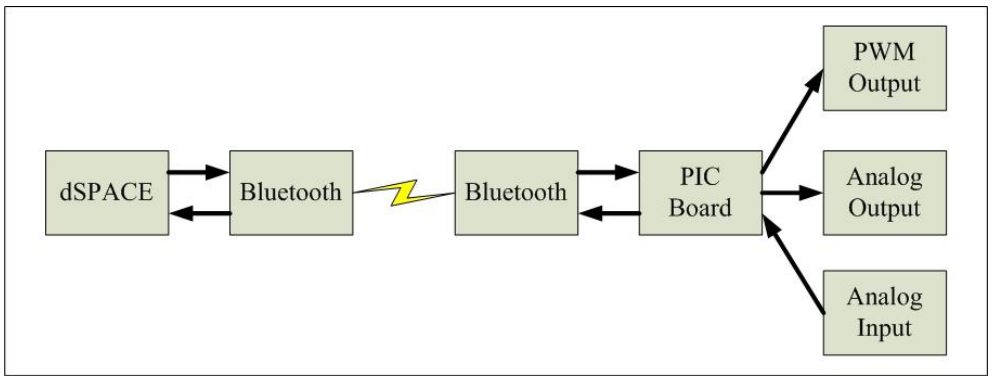


Figure 14: The set up of the system

4.1 Signal Generator and Oscilloscope

The link is tested with a signal generator and an oscilloscope. The signal generator is fed to an analog input of the PIC. The PIC translates the signal and sends an 8 bit digital signal to the Bluetooth transceiver that is attached to the PIC. The transceiver forwards the signal to the other transceiver that is connected to the dSPACE board. The signal can be viewed in dSPACE. A control signal is calculated in dSPACE and sent back to the PIC via the Bluetooth transceivers. At the PIC the signal is sent to a D/A converter and displayed at the oscilloscope.

A delay was seen in the system. The gathering of the analog signal was too fast and resulted in buffer overflow at one of the Bluetooth transceivers. The PIC samples as soon as it can forward a signal, this sampling is faster than the transmission from the Bluetooth transceiver and the extra samples are put in a buffer. When the buffer is full the sampling stops, but when the buffer has a free space again a new sample is taken. The buffer is large enough to cause a delay of around 10 seconds. This problem was fixed by synchronizing the sampling of the analog signal to the receival of the control signal from dSPACE.

4.2 Motor Control

A DC motor is controlled with the Bluetooth link. The input to the PIC remains the same, the output is now a PWM signal. The PWM signal is forwarded to a H-bridge and then to the DC motor. A variable resistance is used to measure the velocity of the DC motor and that signal is fed to the analog input on the PIC. The communication with dSPACE remains unchanged.

A different delay was seen with this setup. The control signal is delayed on the way to the PIC. The sample time in dSPACE is too high and the control signal is put in a buffer on the way to the PIC. This is fixed by adjusting the sample time in Simulink.

4.3 Measurements

The transmission time from the input of one Bluetooth chip to the output of the other is 24 ms, the frequency is 41.6 Hz. The measurements were made with an oscilloscope.

The range of the chips was also tested and connection was lost at about 40 m, the connection could be reset at around 20 m.

5 Future Work

The link can be used for communication, and is sufficient for the motor control. Future work includes more channels for communication, improved set up of the circuits and to add a portable power supply.

Future work for the Segway task includes the following:

- Improved control
- Connection to the transporter

When a transporter is operated various security issues will have to be dealt with. How should it behave when the battery runs out? What should be done when the maximum speed is reached? How fast should it accelerate and slow down? What happens if the driver falls off when the transporter is moving? These and many more issues will have to be configured in the control.

6 References

Analog Devices (1997), *+2.7 V to +5.5 V, Serial Input, Dual Voltage Output 8-Bit DAC*, One Technology Way, Massachusetts, USA.

Bies, L. (April 2008a), *Serial UART, an in depth tutorial*, http://lammertbies.nl-/comm/info/serial-uart.html, 21. October 2009.

Bies, L. (April 2008b), *Hayes modem AT commands*, http://lammertbies.nl/comm-/info/hayes-at-commands.html, 21. October 2009.

Dallas Semiconductor (1998), *Application Note 83, Fundamentals of RS-232 Serial Communications.*

dSPACE (2009), *Catalog 2009*, dSPACE, Paderborn, Germany.

EZURiO Limited (2006), *EZURiO, AT command set.*

EZURiO Limited (2009), *EZURiO, Bluetooth Intelligent Serial Module*, http://-www.ezurio.com/products/bism/, 13. October 2009.

Forouzan, B. A. (2003), *Data Communications and Networking third edition*, McGraw-Hill, New York.

Maxim Integrated Products (2006), *Maxim, +5V-Powered, Multichannel RS-232 Drivers/Receivers*, Maxim Integrated Products, California.

McDermott-Wells, Patricia (2004), *What is Bluetooth?*, IEEE Potentials, December2004/January2005, p. 33-35.

Microchip Technology Inc (2002), *PIC16F7X, Data Sheet*, U.S.A.

Microchip Technology Inc (1997), *PICmicro, Mid-Range MCU Family, Reference Manual*, U.S.A.

Sairam, K.V.S.S.S.S.- Gunasekaran, N. - Rama Reddy, S. (2002), *Bluetooth in Wireless Communication*, IEEE Communications Magazine, June 2002, p. 90-96.

Sandberg, Anders - Westman, Bengt (2007), *Report Balancing scooter project*, IEA, Sweden.

Segway Inc. (2008a), *Explore Models*, http://www.segway.com/individual/models/-index.php, 29. September 2008.

Segway Inc. (2008b), How the Segway PT Works, http://www.segway.com/individual/learn-how-works.php, 22. October 2009.

Segway Inc. (2008c), *Segway Milestones*, http://www.segway.com/about-segway/-segway-milestones.php, 29. September 2008.

Segway Inc. (2008d), *Simply Moving*, Segway Product Brochure.

Strangio, Christopher E. (2006), *The RS232 Standard*, CAMI Research Inc., Massachusetts, http://www.camiresearch.com/Data_Com_Basics/RS232_-standard.html#anchor341671, 9. Oktober 2009.

7 Appendix

A AT Commands

This list contains some of the commands used in the project and a short explanation. Full documentation on the AT commands can be found on EZURiO's website.

ath	drop an existing connection.
ati4	returns a 12 digit hexadecimal Bluetooth address
ats0	number of RING indication before automatically answering an incoming connection
ats504	suppresses messages arising from connections or pairing
ats507	configure how connection is dropped
ats512	specifies power up state
ats520	change to a standard baud rate
ats530	reconnect delay when configured as master in pure-cable-replacement mode
at&f*	clear non-volatile memory
at&w	write to non-volatile memory
at+btr	set outgoing peer address
at+btt	add trusted device

B PIC Programming Code

B.1 main

```
#include <pic.h>
#include "header.h"
#include "delay.h"
#include "forLCD.c"
#include "forUART.c"
#include "forAD.c"
#include "forPWM.c"

__CONFIG(0x5a); //PIC16F73

void DelayMs(unsigned char ccnt)
{
unsigned char i, cnt;
cnt = ccnt;
do {
i = 4;
do {
DelayUs(250);
} while(--i);
} while(--cnt);
}

void DelayS(unsigned char ccnt)
{
unsigned char i, cnt;
cnt = ccnt;
do {
i = 4;
do {
DelayMs(250);
} while(--i);
} while(--cnt);
}

void adrPort(int adr){
switch(adr){
case 4:
RC0=0;
RC1=0;
```

```
RA3=1;
break;
case 2:
RC0=0;
RC1=1;
RA3=0;
break;
case 0:
RC0=0;
RC1=0;
RA3=0;
break;
}
}

// Define function of the pins I/O
void setupPorts()
{
TRISA = 0xef;   //1110 0111
TRISB = 0; //0000 0000
TRISC = 0x80;   //1000 0000
}

void main()
{
char ctrlSignal, measuredValue;
// Define ports
setupPorts();
// Setup the LCD display
setupLCD();
// Setup analog digital conversion
setupAD();
// Setup PWM
setupPWM();
// Display message, mostly used to verify updates
//msgLCD();
msgOK();
// Set up the UART connection
setupUART();
DelayS(4);
ADGO=1;

while(1)
```

```
{
// If RCIF, data on UART forwarded to D/A
if(RCIF)
{
ctrlSignal=rxUART();
// Output as analog signal
//sendDA(ctrlSignal);
// Output as PWM
setPWM(ctrlSignal);
ADGO=1;
}
// If ADIF, data on A/D forwarded to UART
if(ADIF & TXIF)
{
measuredValue=getAD();
txUART(measuredValue);
}
}
}
```

B.2 forLCD

```
// Send control commands to the screen
void LCDControl(char ctrl)  //Control
{
LCDbus = ctrl;
RS = 0;
//AdrPort=LCDadr;
adrPort(0);
AdrEnable = 1;
DelayUs(50);
AdrEnable = 0;
DelayUs(4);
}

// Configure the LCD screen
void setupLCD()
{
DelayMs(20);
LCDControl(FunctionSet);
DelayMs(10);
LCDControl(FunctionSet);
DelayMs(5);
```

```
LCDControl(FunctionSet);
DelayMs(5);
LCDControl(FunctionSet);
DelayMs(5);
LCDControl(EntryModeSet);
DelayMs(5);
LCDControl(DisplayOff);
DelayMs(5);
LCDControl(DisplayClear);
DelayMs(10);
LCDControl(DisplayOn);
DelayMs(20);
}

// Display a character on LCD screen
void LCDWrite(char write)
{
LCDbus = write;
RS = 1;
adrPort(LCDadr);
DelayUs(40);
AdrEnable = 1;
DelayUs(40);
AdrEnable = 0;
DelayUs(4);
}

void msgLCD()
{
LCDWrite('A');
LCDWrite('A');
//LCDWrite('');
}

void msgOK()
{
// Write on the lower line of the display
LCDControl( LowerLine );
LCDWrite('O');
LCDWrite('K');
    //Return the cursor on the upper line
LCDControl( UpperLine );
}
```

B.3 forUART

```
// Configure UART on PIC
void setupUART()
{
SPBRG=129; //Valid for high speed, 20MHz and 9600 Baud
BRGH=1; //1 for High speed, 0 for Low speed
SYNC=0; //1 for synchronous, 0 for asynchronous
SPEN=1; //Enable the pins for the serial ports
TXIE=0; //Transmission interrupts. 0 for disable, 1 for enable
RCIE=0; //Reception interrupts. 0 for disable, 1 for enable
TX9=0; //Transmission : 1 for nine bits, 0 for eight
RX9=0; //Reception : 1 for nine bits, 0 for eight
TXEN=1; //Enable transmission
CREN=1; //Enable reception
}

// Send via Bluetooth (UART)
void txUART(char prufa)
{
//int j=0;
while(!TXIF)
{
//Do nothing, just wait
}
TXREG=prufa;
}

// Receive via Bluetooth (UART)
char rxUART()
{
char temp;
// Check for framing error and reset
if(FERR)
{
LCDWrite('F');
CREN=0;
CREN=1;
}
// Check for overfull error and reset
if(OERR)
{
LCDWrite('O');
```

```
CREN=0;
CREN=1;
}
temp=RCREG;
return temp;
}
```

B.4 forAD

```
// Configure A/D converter on PIC
void setupAD()
{
ADCON1=0x04; // U U U U U PCFG2 PCFG1 PCFG0
ADCON0=0x81;  // ADCS1 ADCS0 CHS2 CHS1 CHS0 GO/DONE U ADON
ADIF=0; //Flag bit for incoming A/D
}

// Get A/D input
char getAD()
{
char temp;
temp=ADRES;
ADIF=0;
return temp;
}

// Forward bits to D/A chip
void outDA(char out)
{
char i, sr;
sr=out;
for (i=0; i<8; i++)
{
// Send the MSB and then shift
if ((sr & 0x80) == 0)
DAdat=0;
else
DAdat=1;
// Bits are loaded on clocks rising edge
DAclk=1;
sr = sr<<1;
DAclk=0;
}
```

```
}

// Send D/A output, data and control signal for inverter
void sendDA(char DA)
{
char ControlBits=0x03;  // Channel A/B - 3/7
char DataBits=DA;
DAdat=0;
DAclk=0;
//AdrPort=DA0adr;
adrPort(4);  // D/A chip U107/U109 - 4/5
AdrEnable = 1;
outDA(ControlBits);
outDA(DataBits);
AdrEnable = 0;
}
```

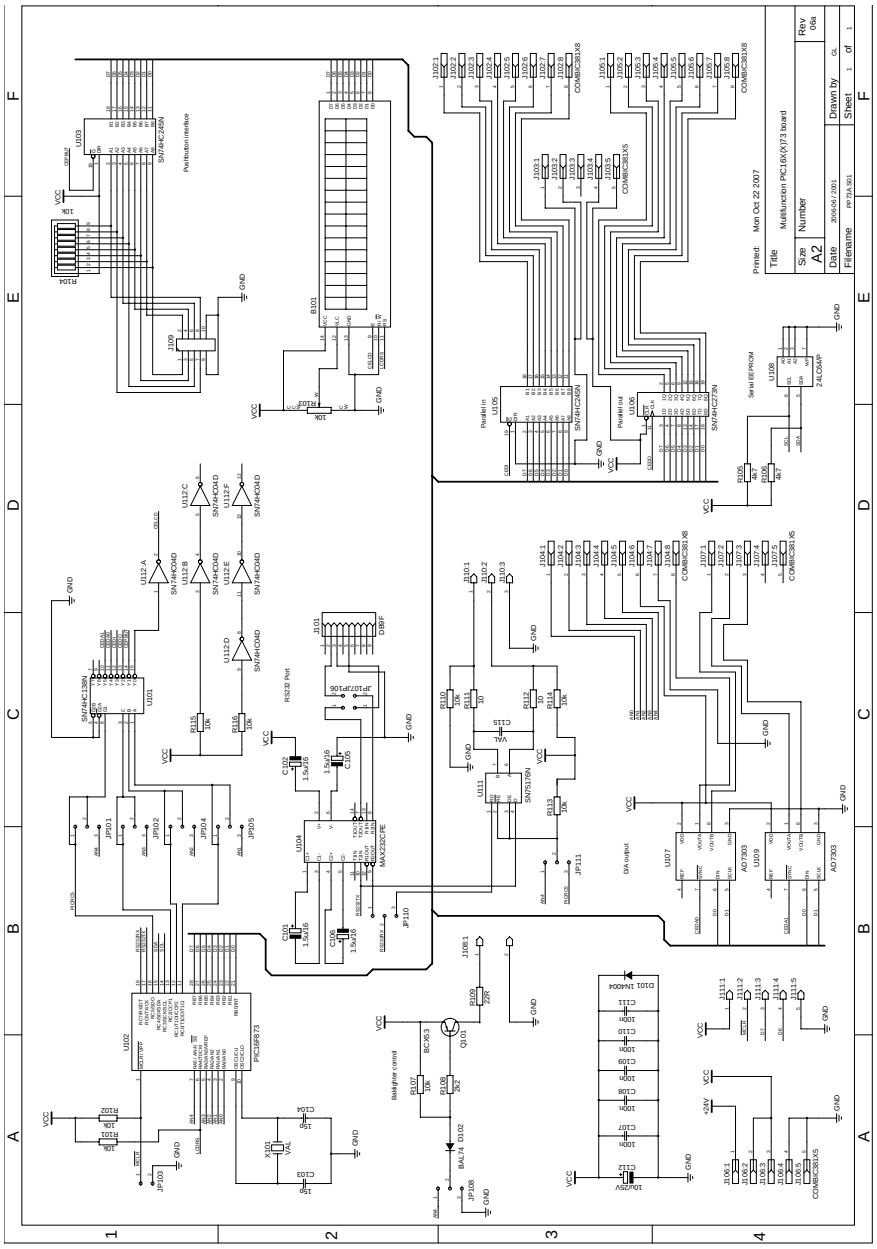
B.5 forPWM

```
// Configure PWM on PIC
void setupPWM()
{
PR2=0xC7; //PWM frequency 50 kHz, 20 uS
CCPR1L=0x80;  //Begin with half duty cycle
CCP1CON=0x0C; //0000 1100 PWM mode, the two lsb=0
T2CON=0x04; //Timer prescaler =1, timer2on
}

// Set the PWM duty cycle
void setPWM(char value)
{
char enable;
char pwm = value;  // 0<=value<=255
// Set direction and
if((pwm & 0x80) == 0)  // 0<=value<=127
{
enable=0x80-pwm; //1<=enable<=128
DIN1=0;
DIN2=1;
if((pwm & 0x80) == 0)
{
enable=0x7F;
}
```

```
}
else  // 128<=value<=255
{
enable=pwm-0x80; // 0<=enable<=127
DIN2=0;
DIN1=1;
}
//AdrPort=DOUTadr;
adrPort(2);
AdrEnable=1;
enable=enable<<1;  // 0<=enable<=254 (step 2)
CCPR1L=enable;
AdrEnable=0;
}
```

C The Multifunction PIC Board



The board contains a microcontroller, LCD screen and other integrated circuits that are frequently used together with the controller. This board is very good to have under the development phase of a project.