ÇANKAYA UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
COMPUTER ENGINEERING

MASTER THESIS

AUTOMATIC GUITAR TUNER

SAİT MUTLU

FEBRUARY 2013

Title of the Thesis : Automatic Guitar Tuner

Submitted by Sait MUTLU

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University

Prof. Dr. Taner Altunok
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Assist.Prof. Dr. Murat Saran
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree Master of Science.

Assist.Prof. Dr. Abdül Kadir Görür
Supervisor

Examination Date : 07.02.2013

Examining Committee Members

Asst. Prof. Dr. Abdul Kadir Görür (Çankaya Univ.)

Asst. Prof. Dr. Bülent Gürsel Emiroğlu (Başkent Univ.)

Asst. Prof. Dr. Fahd Jarad (Çankaya Univ.)

STATEMENT OF NON-PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Sait MUTLU

Signature :

Date : 07.02.2013

ABSTRACT

AUTOMATIC GUITAR TUNER

MUTLU, Sait

M.S.c., Department of Computer Engineering

Supervisor: Prof. Dr. Abdül Kadir GÖRÜR

Co-Supervisor: Dr. Ali Rıza AŞKUN

February 2013, 93 pages

This thesis presents the Automatic Guitar Tuner, a new device to help musicians to tune guitars effectively. The device is composed of two main parts: Frequence Analysing and String Adjusting. Frequence analysing job can be done by any tuner on the market. It is very easy to find a tuner to understand the notes of strings. Thus the most important point is string adjusting part. This part consists of a combination of mechanical components and electronic circuits. It is not necessary to make any circuit to analyse frequency in this project. Cherub wst-550c metro-tuner is chosen for this job because of its price and performance.

String adjusting is highly complicated job because of its mechanic part and integration of its components. Construction phase and all used equipments are described in 3rd Chapter. test phase an and test results are presented in 5th Chapter. Test results show that the device has enough accuracy and using this device makes easier tuning job.

Keywords: Microcontroller, PIC, Tuner, FFT, CCS C,

ÖZ

OTOMATİK GİTAR AKORT CİHAZI

MUTLU, Sait

Yüksek Lisans, Bilgisayar Mühendisliği Bölümü

Danışman: Prof. Dr. Abdül Kadir GÖRÜR

Eş-Danışman: Dr. Ali Rıza AŞKUN

Şubat 2013, 93 Sayfa

Bu tezde, müzisyenlerin gitarlarını etkili şekilde akort etmelerini sağlayan Otomatik Gitar Tuner sunulmaktadır. Bu cihaz iki ana parçadan oluşmaktadır: Frekans Analizi ve Tel Ayarlama. Frekans Analizi piyasadaki herhangi bir tüner ile yapılabilmektedir. Bir tünerle gitar tellerinin notasını anlamak çok kolaydır. Bu yüzden en önemli nokta Tel Ayarlama kısmıdır. Bu kısım, mekanik parçalardan ve elektronik devrelerden oluşmaktadır. Frekans Analiz kısmı ise bu projede gerekli görülmemiştir. Bu iş için fiyatı ve performansından dolayı Cherub WST-550C Metro Tuner seçilmiştir.

Otomatik tel ayarlama, mekanik kısımları ve elektronik parçalarının entegrasyonundan ötürü çok karışık bir iş olarak görülmektedir. Yapım aşaması, ve kullanılan tüm ekipmanlar 3. Bölümde, Test fazı ve sonuçları 5. bölümde sunulmaktadır. Test sonuçları bu cihazın yeterli doğrulukta ve akort işlemini kolaylaştırdığını göstermektedir.

Anahtar Kelimeler: Mikrodenetleyici, PIC, Akort Cihazı, FFT, CCS C

ACKNOWLEDGMENTS

First and foremost, I wish to express my gratitude to Abdül Kadir GÖRÜR for his guidance, advice, comments and encouragements throughout the research.

I would also like to thank Fatih Genç for his suggestions and technical assistance on assembling and testing electronic circuits.

Finally, I wish to thank my parents for understanding the long periods I have been working and not setting aside time for them.

TABLE OF CONTENTS

CHAPTERS

LIST OF TABLES

TABLE OF FIGURES

INTRODUCTION

The body and neck which are main two parts of any guitar are made of wood. Wood is a natural material so that air humidity and temperature changes cause expansion or contraction. Tensions and also pitches of the strings are affected by expansion and contraction of the guitar. Therefore tuning operation is common work for guitar musicians. Tuning of a guitar takes average 5 -15 minutes after setting up new strings and average 3 - 5 minutes before every performance. Furthermore, guitar teachers have to tune their student's musical instruments which results in a considerable amount of waste time.

Tuning is a common process for all guitar musicians using conventional tuners. The simplest way is to plug the electric guitar directly into an electronic tuner. This project is completed by simple steps. Firstly, a string is plugged and tuner is checked to see if the note is sharp, flat or in tune. According to the feedback from the tuner, string tension is adjusted by turning the pegs clokwise or counterclockwise. Subsequently this process is repeated until the string is in tune. Then, all each string is tuned with this method. So it is strongly needed to reduce the time spent on tuning each string on a guitar.

Another popular method is to tune a guitar manually according to a reference voice or pitch pipe. This requires a good musical ear, but it is required to find a quiet place to make the tuning operation and it will also take considerable time . So there is a strong need to minimize the amount of time spent on manually tuning every string on a musical instrument.

Background

Musicians tune their instruments without any tool or by using tuner device. After plucking the string, the tuner shows whether the note is sharp or flat. Then the musician wraps and loosen the string to increase and decrease the string tension. This tuning style can take too much time and be inefficient under adverse or quickly changeable weather conditions. This thesis describes the design and operational processes of the Automatic Guitar Tuner. This concept device is designed to help reducing the tuning time by using an electromechanical part (Geared DC Motor) to wrap or loosen the strings to attain the preset frequencies. This device also provides increasing the productivity in the guitar courses by saving time for the tuning process.

CHAPTER II

THEORY AND LITERATURE REVIEW

1.1 BASIC DEFINITIONS AND CONCEPTS

Before proceeding detailed explanations about tuning, some terms as “sound”, “note”, “pitch”, “frequency” and relationship between notes and frequencies must be already known. Because these terms frequently will be used on remaining parts of the thesis.

“**Sound** is a mechanical wave that is an oscillation of pressure transmitted through a solid, liquid, or gas, composed of frequencies within the range of hearing and of a level sufficiently strong to be heard, or the sensation stimulated in organs of hearing by such vibrations.”[1]

“A **sound wave** is a type of pressure wave caused by the vibration of an object in a conductive medium such as air. When the object vibrates, it sends out a series of waves which can be interpreted as sound. For example, when someone hits a drum, it causes the membrane of the drum to vibrate, and the vibration is transmitted through the air, where it can reach the ear of a listener.” [2]

“The **wavelength** is the distance travelled by the pressure wave during one cycle. The frequency (f), which is the number of pressure variation cycles in the medium per unit time, or simply, the number of cycles per second, and is expressed in **Hertz** (Hz). Noise is usually composed of many frequencies combined together. The relation between wavelength and frequency can be seen in Figure 2.1.”[3]

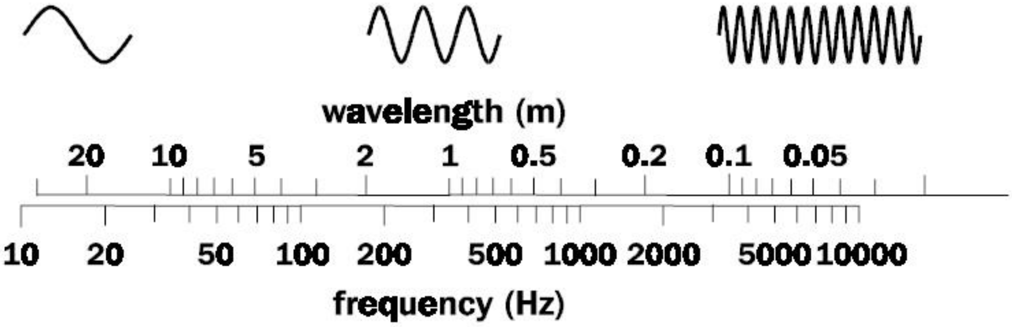


Figure 2.1:"Wavelength in air versus frequency under normal conditions."[3]

Hertz (Hz)=cycles / second

While a sound wave is moving through particles in the medium, it enables each particle to vibrate at the same frequency. For instance, a guitar string which is vibrating at 440 Hz will set all the air particles that carries a sound signal to the ear of a person in the medium to the same frequency and 440 Hz sound wave is detected anywhere in the medium.

Pitch is the sensation of a frequency that describes highness or lowness of a sound. Actually notes don't have a standard preset frequency. "With a frequency around 261.626 Hz, middle C is designated C_4 in scientific pitch notation because of the note's position as the fourth C key on a standard 88-key piano keyboard"[4] and each of the other notes has its own frequency according to middle C. For example the **A** above middle C is usually set at 440 Hz. Some higher and lower frequencies are respectively the higher and lower pitches of A note.

"The perception of sound in any organism is limited to a certain interval of frequencies. For humans, hearing is normally limited to frequencies between about 20 Hz and 20,000 Hz (20 kHz).The upper limit generally decreases in older ages.The other species have different ranges of hearing. For example, dogs can perceive audio frequencies up to 60,000 Hz."[5] Any sound with a frequency less than 20 Hz is in audible range and known as an *infrasound*. Any sound which has a frequency above the audible range is known as an *ultrasound*.

2.1.1 Frequency Measurement

“**A fast Fourier transform** (FFT) is an algorithm to compute the discrete Fourier transform (DFT) and its inverse. There are many different FFT algorithms involving a wide range of mathematics, from simple complex-number arithmetic to group theory and number theory. The DFT is obtained by decomposing a sequence of values into components of different frequencies. This operation is useful in many fields (see discrete Fourier transform for properties and applications of the transform) but computing it directly from the definition is often too slow to be practical. An FFT is a way to compute the same result more quickly: computing the DFT of *N* points in the naive way, using the definition, takes O(N^2) arithmetical operations, while an FFT can compute the same DFT in only O(*N* log *N*) operations. The difference in speed can be substantial, especially for long data sets where *N* may be in the thousands or millions. In practice, the computation time can be reduced by several orders of magnitude in such cases, and the improvement is roughly proportional to *N* / log(*N*). This huge improvement made the calculation of the DFT practical; FFTs are of great importance to a wide variety of applications, from digital signal processing and solving partial differential equations to algorithms for quick multiplication of large integers.”[6]

“The best-known FFT algorithms depend upon the factorization of *N*, but there are FFTs with O(*N*log(*N*)) complexity for all *N*, even for prime *N*. Many FFT algorithms only depend on the fact that $e^{-\frac{2\pi i}{N}}$is an *N*-th primitive root of unity, and thus can be applied to analogous transforms over any finite field, such as number-theoretic transforms. Since the inverse DFT is the same as the DFT, but with the opposite sign in the exponent and a 1/*N* factor, any FFT algorithm can easily be adapted for it. The FFT has been described as "the most important numerical algorithm of our lifetime.”[6]

Cherub WST-550c tuner device can capture live sound using its own microphone that is sticked to the body of the guitar by additional cable. This device analyses the sound signal and measures the frequency of the data coming from the device's microphone by a complex scientific algorithm using Fast Fourier Transform (FFT).

2.2 GUITAR TUNING

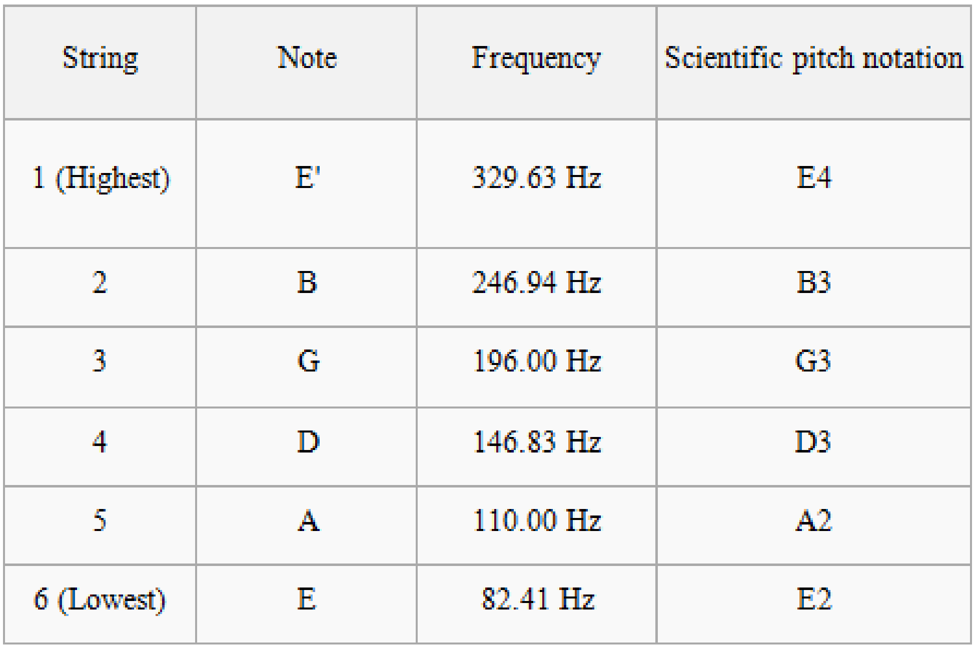
A standard guitar has six strings which are tuned between 82.41 Hz for the lowest string and 329.63 Hz for the highest one. The other strings are tuned to specific frequencies between these two values.

Guitar tuning is a phrase which refers to the adjusting the pitches of the strings to their desired frequency. But desired pitches are changes according to the tuning method, because there are more than one tuning methods. One of them which is called standard tuning is the most used method all over the World. The standard tuning is adjusting pitches of all the strings on the guitar from lowest (low E) to highest (high E) as E, A, D, G, B, and E.

There are three main method for guitar tuning; with a reference of pitch pipe or an other instrument (such as an electronic keyboard), tuning by ear and tuning with an electronic or software (using an interface on a computer or a mobile device) tuner. Regardless of having help from a tool or not, ears must be always used during the tuning job. Instruments should be tuned in the same ambiance as where it will be played. Because most instruments which are made of wood willl be detuned depending on the changing air temperature and humidity. The wood contraction and expansion may affect the string tension and the pitch.

2.2.1 Standard Tuning

Guitar tuning methods have improved for many years. Standard guitar tuning is recently the most popular tuning style and also it is assumed as default tuning style in the music industry. if a specific tuning style is not mentioned. It comprises the following notes:



String	Note	Frequency	Scientific pitch notation
1 (Highest)	E'	329.63 Hz	E4
2	B	246.94 Hz	B3
3	G	196.00 Hz	G3
4	D	146.83 Hz	D3
5	A	110.00 Hz	A2
6 (Lowest)	E	82.41 Hz	E2

Table 1.1: Frequencies and scientific notations of notes

2.2.2 Relative tuning by using Unison method (4th/5th fret method)

There are several popular methods commonly used to tune guitars. Tuning with a reference instrument, with an electronic tuner, tuning in octaves, tuning with harmonics and unison method are most commonly used ones.

Here are basic instructions how to tune a guitar by **unison method**. Using a reference or tuner, the uppermost string is tuned to E (instrument's lowest string) as accurate as possible. Because this string will be used as reference when tuning the other strings. That is, if this string is tuned wrongly, then all other strings will not be in tune.

To tune next string a finger must be placed on the 5th fret of the E-string, which is an A note. After placing a finger on sixth string (E String), Fifth string (A- String) must be adjusted until the two strings are in tune.

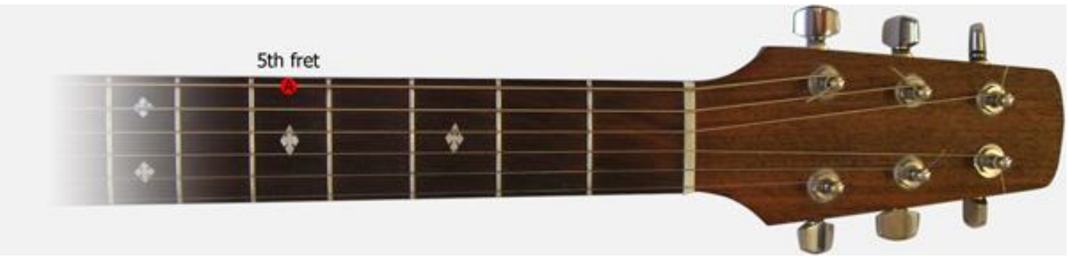


Figure 2.2: Tuning 5th String

When the two upper strings are in tune, Tuning work continues with the D string.After placing a finger on the 5th fret of the A string(D note),fourth string must be adjusted until it is in tune with the reference D note from the A string,

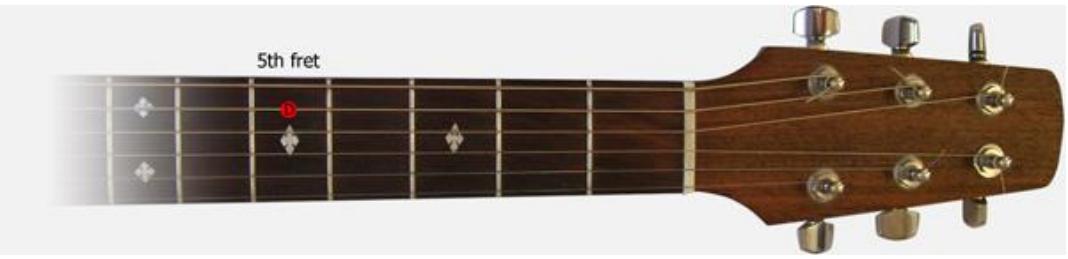


Figure 2.3: Tuning 4th string

5th fret of the D string is a G note. After placing a finger on the 5th fret of the D String, 3rd string must be adjusted until the two strings are in tune.

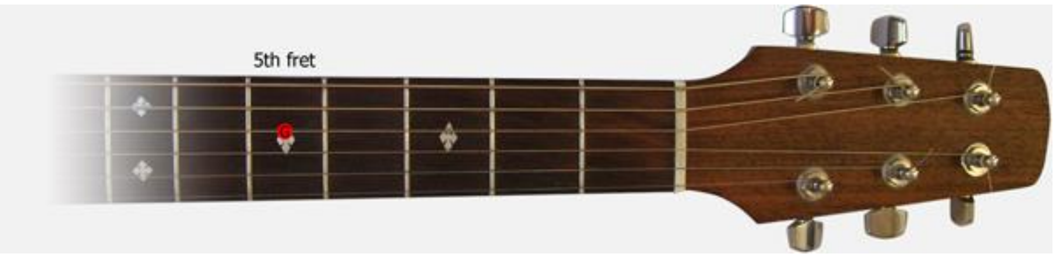


Figure 2.4: Tuning 3rd string

To get a B note from the G string a finger must be putted on the 4th fret. After putting a finger, the 2th string must be adjusted until it is in tune with the reference B note from the G string.

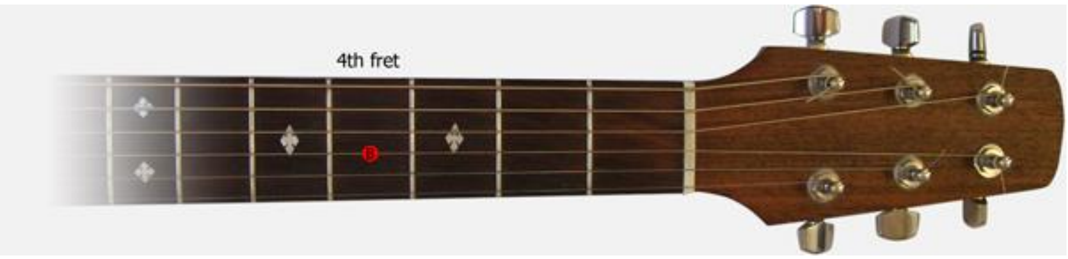


Figure 2.5: Tuning 2nd string

The 5th fret of the B string is an E note.After placing a finger on 5th fret of the B string, The last string must be adjusted untl it is in tune with the reference E note from the B string. After all this instructions, 6th and 1st Strings also must be in tune. According to standard guitar tuning , 6th string is an octave above the 1st string.

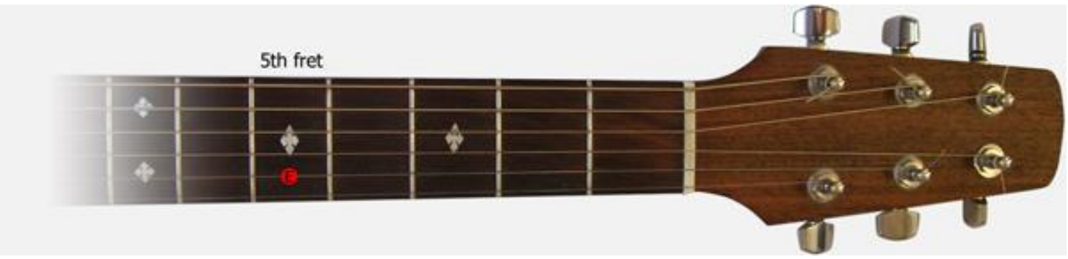


Figure 2.6: Tuning first string

The Figure 2.7 shows name of the pitches for first 5 frets of a guitar when standard guitar tuning is used.

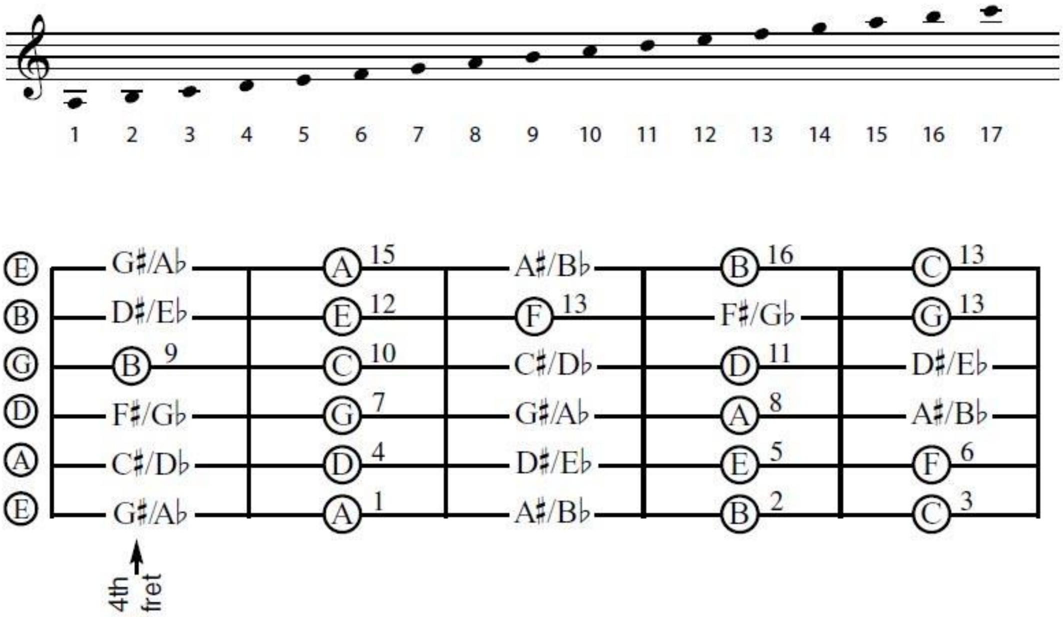


Figure 2.7: The layout of first 5 frets[7]

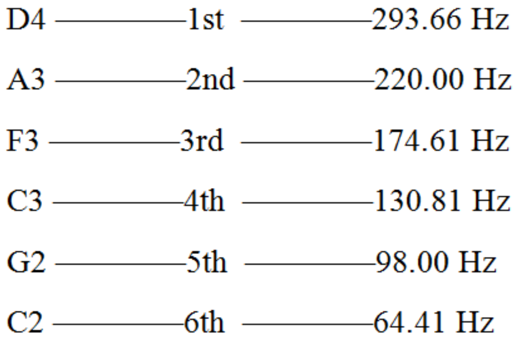
2.2.3 Other Tuning Styles

it is useful to tune the guitar in a different way in some conditions to enable different playing styles and give the instrument another sound character. After some upgrades, Automatic Guitar Tuner can be used to tune the guitar to alternative tunings.

The following tuning is called “Drop D”. The reason why it is called dropped is the sixth string is tuned down from E to D, thus dropped from E to D. The strings are numbered 1-6 starting from the thinnest one.

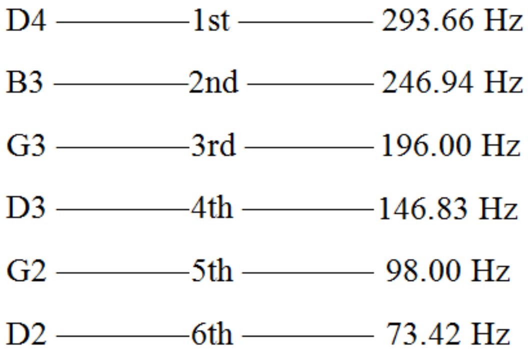
E4	1st	329.63Hz
B3	2nd	246.94 Hz
G3	3rd	196.00 Hz
D3	4th	146.83 Hz
A2	5th	110 Hz
D2	6th	73.42 Hz

Figure 2.8: Drop D Tuning



D4	1st	293.66 Hz
A3	2nd	220.00 Hz
F3	3rd	174.61 Hz
C3	4th	130.81 Hz
G2	5th	98.00 Hz
C2	6th	64.41 Hz

Figure 2.9: Drop C tuning



D4	1st	293.66 Hz
B3	2nd	246.94 Hz
G3	3rd	196.00 Hz
D3	4th	146.83 Hz
G2	5th	98.00 Hz
D2	6th	73.42 Hz

Figure2.10: Open G tuning

2.3 PREVIOUS STUDIES

2.3.1 Tronical Power Tune System

This system is used in some guitar models of two brands. It Works as a function of guitar. Without the need to pluck, the guitar can tune itself in 2 second by using the stretches of strings.

It does not use FFT algorithms to analysis frequency of the string because of taking more time, thanks to special alogithm of this device, tuning work of all strings only takes 2 second.

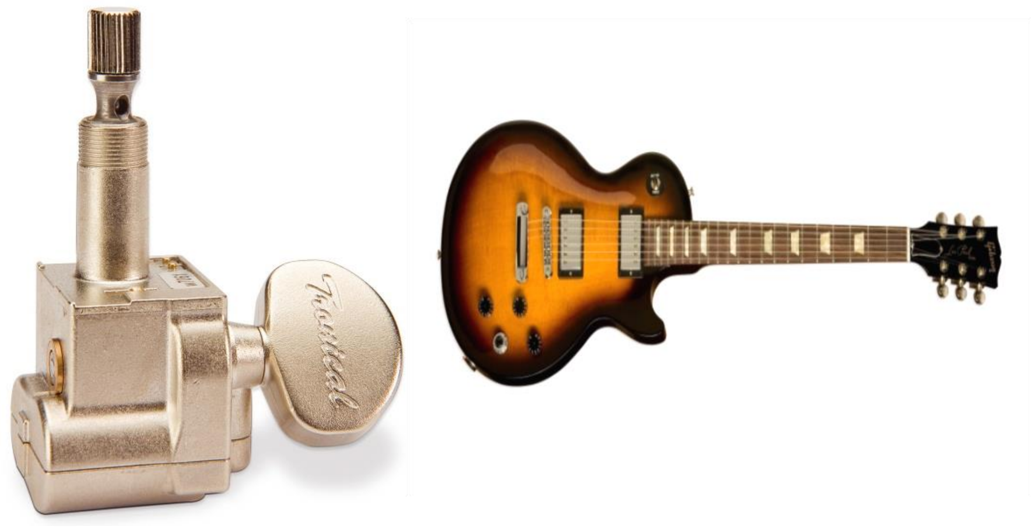


Figure 2.11: Tronical Power Tune System

The device comprises a high-performance servo motor, indicator leds and note selecting button. The power supply of the system comprises two 1.2V NiMh AA rechargeable batteries.
System is integrated with the guitar and can not be used as a separate device. To use this device, it is necessary to buy a Gibson guitar that have Tronical Power Tune System. It can be expensive way to use a automatic tuning system.

PowerTune™ system kits are ready for Fender Telecaster-series and Stratocaster-series beside Gibson SG-series, Les Paul-series, and Flying V.

2.3.2 String MasterTunermatic Guitar Tuner

It is a handheld tuning device released by Actiontuners brand. It is used for tuning guitar strings one by one and no modification to the guitar is necessary to use this device. It uses a servo motor but there is no information about the power or torque of the motor on the internet or any source.

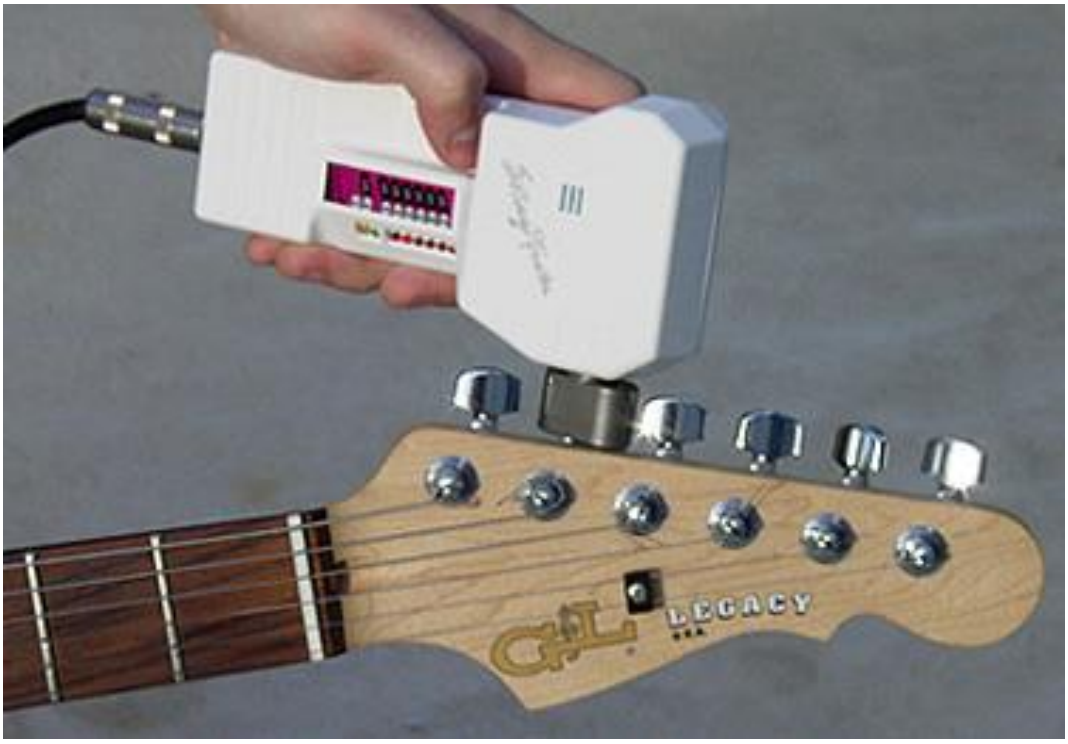


Figure 2.12: String MasterTunermatic Guitar Tuner

“Robotic Guitar tuner operates on one new 9-volt Alkaline Battery. Works on any six string electric or acoustic guitar with an output for a guitar cord. Acoustic guitars without built in pick-ups require acoustic guitar adapter.”[8]

“Features:

- Powerful gear motor
- Works with any pickup-equipped 6-string
- Automatic or manual tuning modes
- Winds or unwinds strings at the push of a button
- Beeps to let you know the string is in perfect pitch
- Fast, easy, and incredibly accurate
- One-year parts and labor warranty”[8]

2.3.3 Automatic guitar tuner by Ryan Freeberg,Darren Pocci, Tiffany Kasettratut

After selecting tuning method with pressing push buttons, it automatically plucks the strings and turns tuning pegs to adjust string tension. It detects frequency of a plucked string and compares predefined tuning set the user specifies. It is an efficient but also a huge and complicated device.

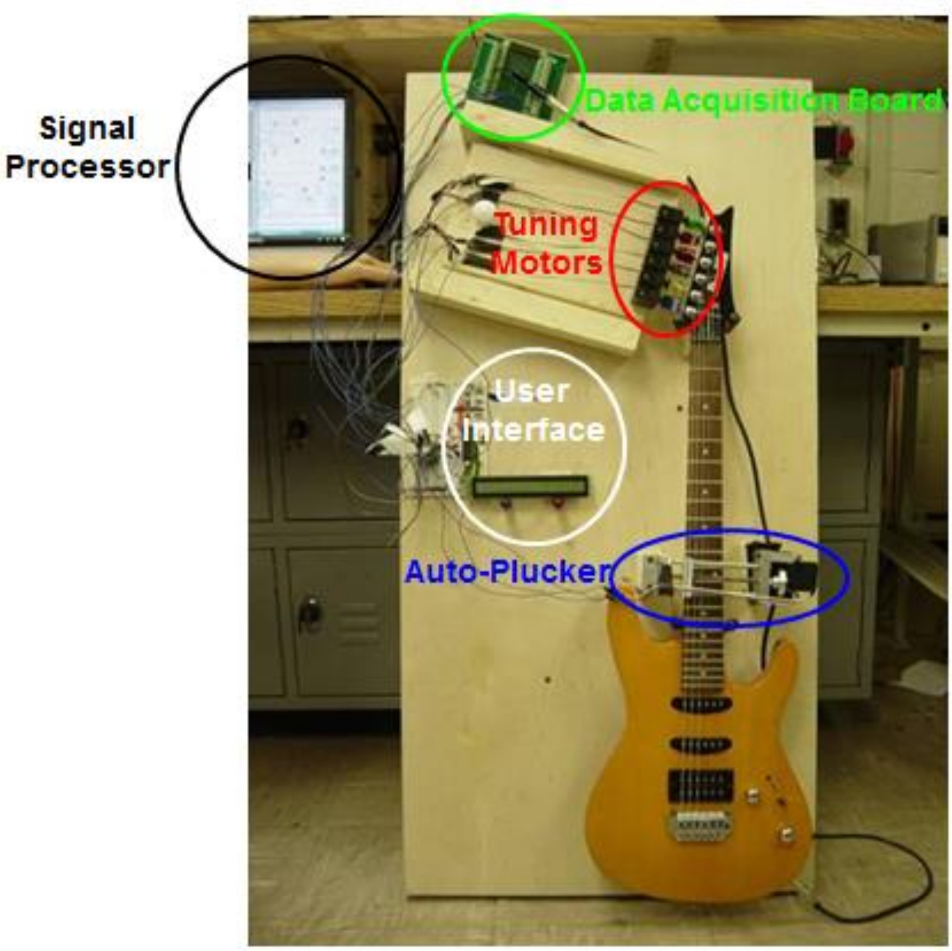


Figure 2.13: Automatic guitar tuner by Ryan Freeberg,Darren Pocci, Tiffany Kasettratut

2.4 MAİN FACTORS FOR A SUCCESFUL TUNER DEVICE

2.4.1 Speed

“CCS C is the one of the most preferred development tool for PIC programming. Because CCS provides a complete, integrated tool suite for developing and debugging embedded applications running on Microchip PIC® MCUs and dsPIC® DSCs. Development tools offered by CCS include an optimized C compiler, in-circuit programmers/debuggers, production programmers and complete development kits that contain all hardware, software and accessories needed to jump start your product development.”[9]

To be able to use computer engineering skills, it is also important to use CCS C on this Project. Basic C codes can be used with this tool. Therefore we can easily generate C output files beside hex files.

16F877A is preferred as main controller. The 16F877A is one of the most popular PIC microcontrollers. It has 40 pin and many internal peripherals.

The 40 pins make it easier to use the peripherals as the functions are administered over all above the pins. So this is easy to decide what kind of external devices to attach without worrying too much if there enough pins for the job.

2.4.2 Continuity and Maintenance

Material quality is the most important factor for the longevity of the product. It is the one of the most important things for consumer relations and satisfaction after the mass production. Durability of a product depends on the weakest part of this device. Even if a device has a perfect algorithm or design, quality of its materials is important as other factors.

Thus, the parts are selected that have been used for a long time in market. 16F877 PIC is preferred because of being one of the most popular PIC since 1980’s and

Brushless DC Motor is preferred because of their longevity. The other electronic equipment is also specially chosen.

Even if the material quality is higher, due to the some unpredictible problem s or bad designed algorithm, device can be producted in a poor quality. To prevent this situation, CCS C and pcw compiling interface is preferred to program this device by using computer engineering background.

At first, a basic algorithm is designed. According to hardware deficiencies, algorithm and the code has been revised. Some difficulties can cause deep changes in any project.

2.4.3 Producibility

To make device suitable for mass production, readily availible parts are selected. complex structures should be avoided. So more basic design is preferred.

According to this principle, 6 String tuning system idea fails because complexity and size is more than expected. Finally a system is designed with only one DC motor. It means all the strings can be tuned one by one by using this device. Despite this disadvantage, its bulk is more lightweight and smaller.

2.4.4 Ergonomics

Device must be ergonomically perfect and should be designed to be kept with one hand to facilitate the tuning process.

Dimensions should allow user to carry easily and should be small enough to fit in the pocket of the guitar case. At the same time, it should have a strong motor to turn 7 kg from 1cm distance.

2.4.5 Performance and Accuracy

System should work properly and posible errors should be handled before present to customers. For example, 1. String and 6. String are both E. But 1. String must be 329.63 hz and 6. String must be 82.41 hz. It is important to handle which string must be at which frequency. Now the device runs without evaluating the ovtave of any note. This is an important handicap but the problem will be fixed with removing particular parts and upgrading the code. On the other hand, test results shows us this device tune guitars with enough accuracy.

CHAPTER II

3. THE OPERATION OF THE DEVICE

The Automatic Guitar Tuner Hardware is shown in figure 3.1. This figure shows the operation of the system briefly.

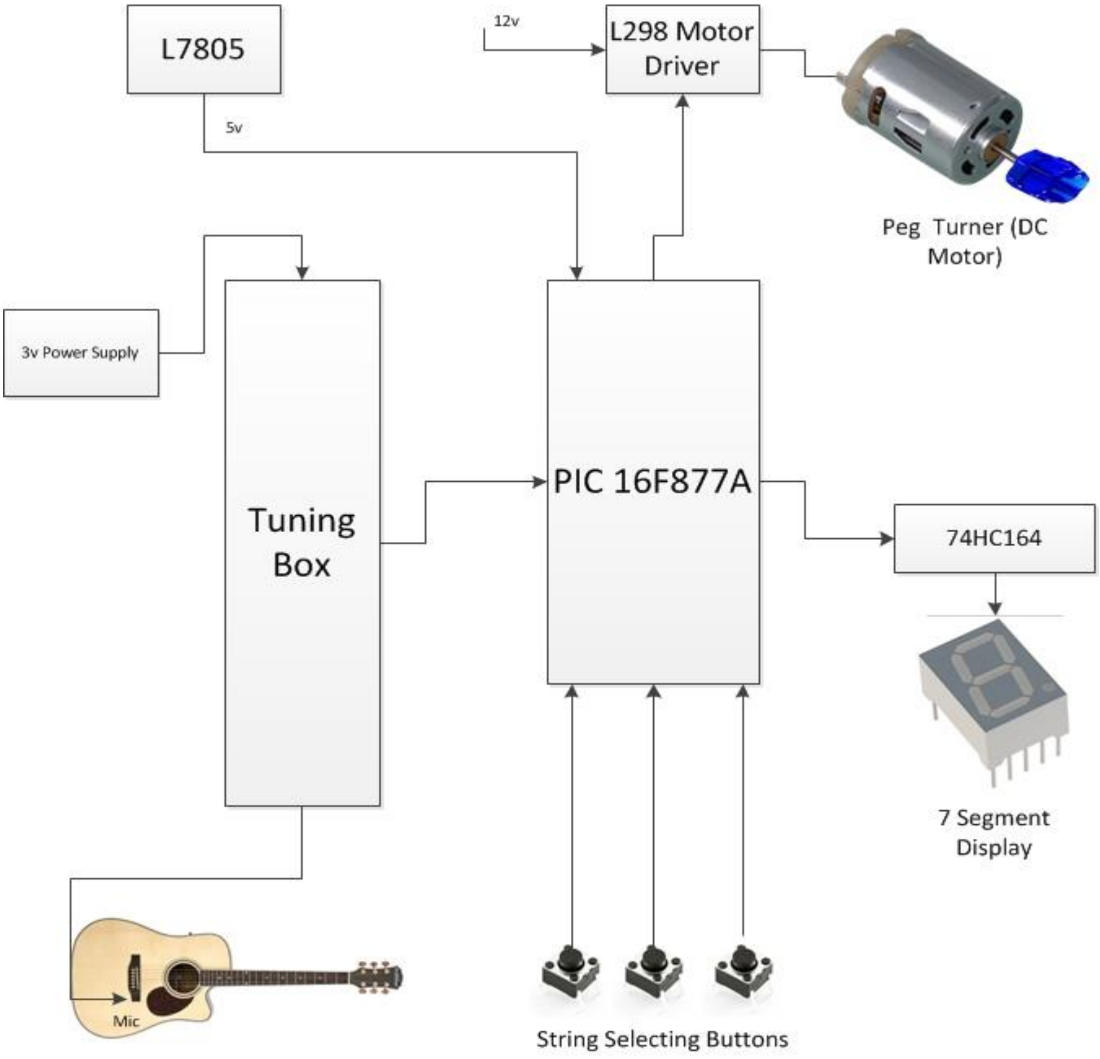


Figure 3.1: Block Diagram of the Automatic Guitar Tuner

Firstly, the string which is wanted to tune must be selected with up/down buttons. By pressing trigger button, tuner device waits for signal. By plucking string, tuner starts to analyse the frequency. After analysing frequency, it sends the data to 16f877 through A port. According to coming information, pic rotetes the motor left or right for a specified period of time. Until making true adjustment, 2,3 and 4. steps are repeated. After finishing the adjustment, system resets itself and comes back ready position for new string.

User Instructions

- Choose the string by up/down buttons
- Place the device to true tuning peg.
- Press trigger button
- Pluck the true string manually by other hand.
- Keep plucking until the string has been tuned.
- Do same work with other strings.

All the components are mounted on 5 connected breadboards except Tuning Box. Tuning Box analyses the incoming signal and sends the data to PIC through leds and photodiodes.

PIC is the brain of the system. It compares the data which is sent from Tuning Box and selected string. After comparation, PIC gives the order of turning direction and time to Motor Driver. Motor driver turns the motor according to coming order. Detailed information of all the components will be given in this chapter.

3.1 HARDWARE IMPLEMENTATION

There are 3 different power input. L7805 chip converts 12v to 5v to supply PIC, and other chips. Also 12v is needed for DC Motor to get enough torque. Tuner device works with 3v. So there are 2 different power supply as 12v and 3v.

16F877A PIC, L298 Motor Driver, Cherub WST-550C Tuner Device, 60W DC Motor, 74HC164 Shift Register, 3mm leds, BPW34 Photodiodes, 7-Segment

Display, USBURN Brenner 8 USB Pic Programmer, L7805 Voltage Regulator, push buttons, UF5408 diodes, 4Mhz Crystal Oscillator and different kind of capacitors and resistors are used in this Project. Figure 3.2 shows the circuit design.

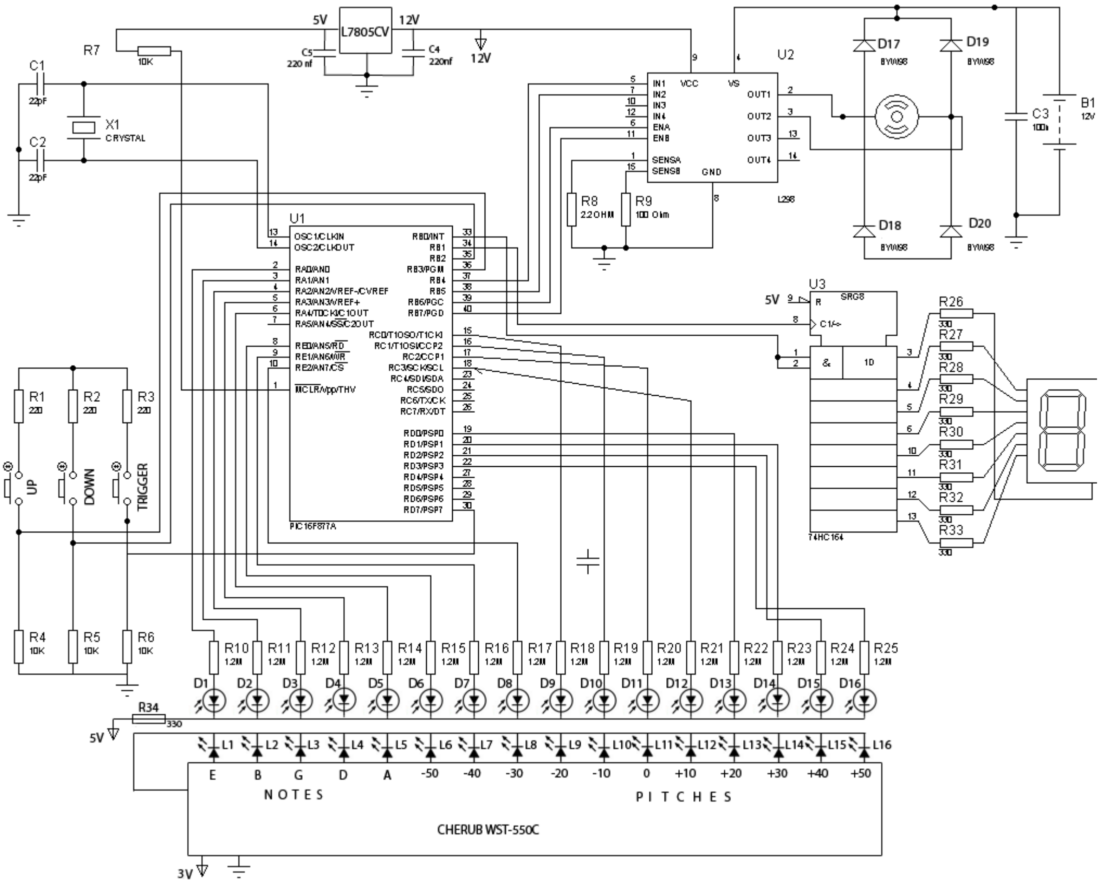


Figure 3.2: Main design of the circuit

3.1.1 Microcontroller

“A microcontroller is an entire computer manufactured on a single chip. Microcontrollers are usually dedicated devices embedded within an application. For example, microcontrollers are used as engine controllers in automobiles and as exposure and focus controllers in cameras. In order to serve these applications, they have a high concentration of on-chip facilities such as serial ports, parallel inputoutput ports, timers, counters, interrupt control, analog-to-digital converters, random access memory, read only memory, etc. The I/O, memory, and on-chip peripherals of a microcontroller are selected depending on the specifics of the target application. Since microcontrollers are powerful digital processors, the degree of control and programmability they provide significantly enhances the effectiveness of the application.”[10]

3.1.1.1 PIC Microcontrollers

“PIC is a family of modified Harvard architecture microcontrollers made by Microchip Technology, derived from the PIC1650 originally developed by General Instrument's Microelectronics Division. The name PIC initially referred to *Peripheral Interface Controller*.”[11] PIC’s are very popular among industrial and personal developers owing to their large user experiences and know-how abundance, resource low cost, wide availability on the market, free development tools and re-programmable flash memory.

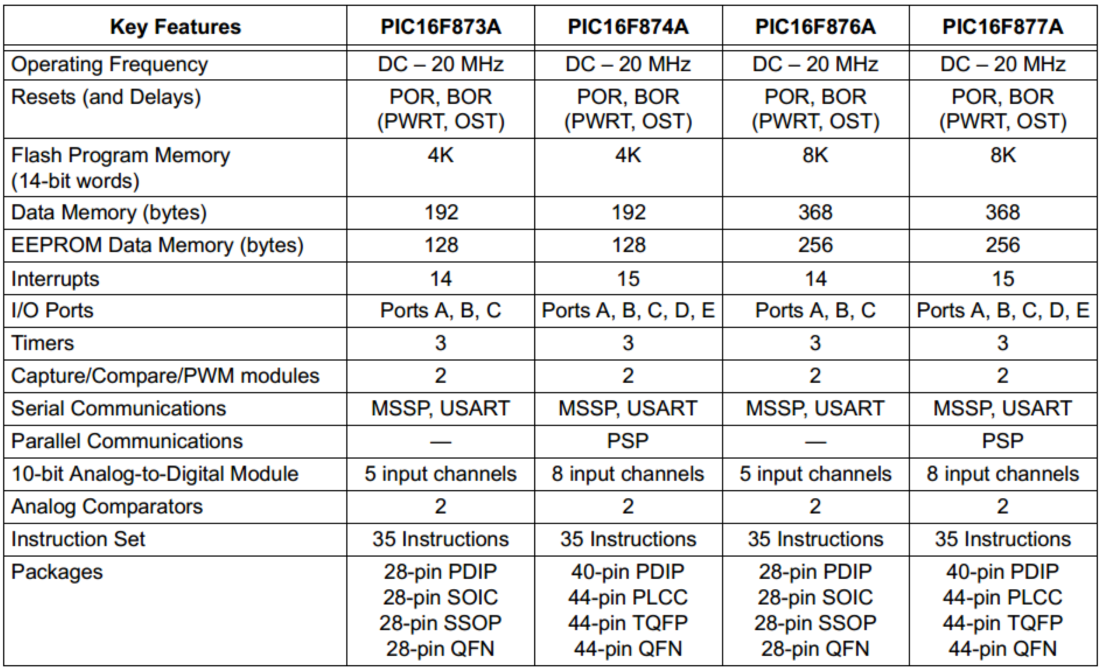
The most important part of the circuit is microprocessor. Because code is installed on it and it will is the main control and processing unit. The other parts will work depending on microprocessor’s capabilities.

There is 2 question to choose true chip for the Project: “What is the price of it?” and “Is it sufficient to the requirements?”. According to this questions, 16F877A is chosen as the optimal device. It is a RISC (Reduced Instruction Set Computer) machines that can be used with only 35 single word instructions. Each instruction executes approximately in one internal clock. It has reduced addressing modes (direct and indirect addressing).

3.1.1.2 PIC 16F877A

“16F877A has Flash type program memory, so it does not have to be taken off the board to load the program. Flash memory can be programmed onboard over 1.000.000 times. EEPROM memory makes it easier to apply microcontrollers to devices where permanent storage of various parameters is needed. On the other hand, CISC machines like Intel 80x86, 8051 or Motorola 68HC11 have many instructions usually more than 100, many addressing modes and it usually takes more than one internal clock cycle to execute instructions.”[12] But they would not be suitable for this project because of the expensive prices and resource scarcity. In conclusion, low consumption, low cost, easy handling, ,high speed, more development tools, durability and flexibility make PIC 16F877A applicable in this project.

The PIC16F877A features 8K x 14 words of Flash Program Memory, 368 x 8 bytes of Data Memory (RAM), 256 x 8 bytes of high speed EEPROM Data Memory, In-Circuit Serial Programming™ (ICSP™) via two pins, 10-bit, up to 8-channel Analog-to-Digital Converter (A/D), Brown-out Reset (BOR), Analog Comparator module with two analog comparators and voltage reference (VREF) module, Watchdog Timer (WDT), In-Circuit Debug (ICD) via two pins, Programmable code protection, Power saving Sleep mode, Selectable oscillator options and low power comsumption. Flash program memory can be written and deleted least 100000 times and EEProm Retention time is more than 40 years. It supports up to 20 Mhz clock input.



Key Features	PIC16F873A	PIC16F874A	PIC16F876A	PIC16F877A
Operating Frequency	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz
Resets (and Delays)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
Flash Program Memory (14-bit words)	4K	4K	8K	8K
Data Memory (bytes)	192	192	368	368
EEPROM Data Memory (bytes)	128	128	256	256
Interrupts	14	15	14	15
I/O Ports	Ports A, B, C	Ports A, B, C, D, E	Ports A, B, C	Ports A, B, C, D, E
Timers	3	3	3	3
Capture/Compare/PWM modules	2	2	2	2
Serial Communications	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
Parallel Communications	—	PSP	—	PSP
10-bit Analog-to-Digital Module	5 input channels	8 input channels	5 input channels	8 input channels
Analog Comparators	2	2	2	2
Instruction Set	35 Instructions	35 Instructions	35 Instructions	35 Instructions
Packages	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN

Table 3.1: Features of 16F877A [13]

3.1.1.3 Softwares compatible with 16F877A

“

- PCM - CCS C Compiler for Midrange Family of PIC MCUs
- PCWH CCS Windows Compiler for Baseline, Midrange, and PIC18 Families of PIC MCUs
- HI-TECH C Compiler for PIC10/12/16 MCUs (Standard)
- HI-TECH C Compiler Enterprise Edition
- HI-TECH C compiler for PIC10/12/16 MCUs (PRO)
- PCWHD Command-Line + IDE for PIC10/PIC12/PIC16/PIC18/PIC24/dsPIC

  PicBasic Gold Edition”[14]

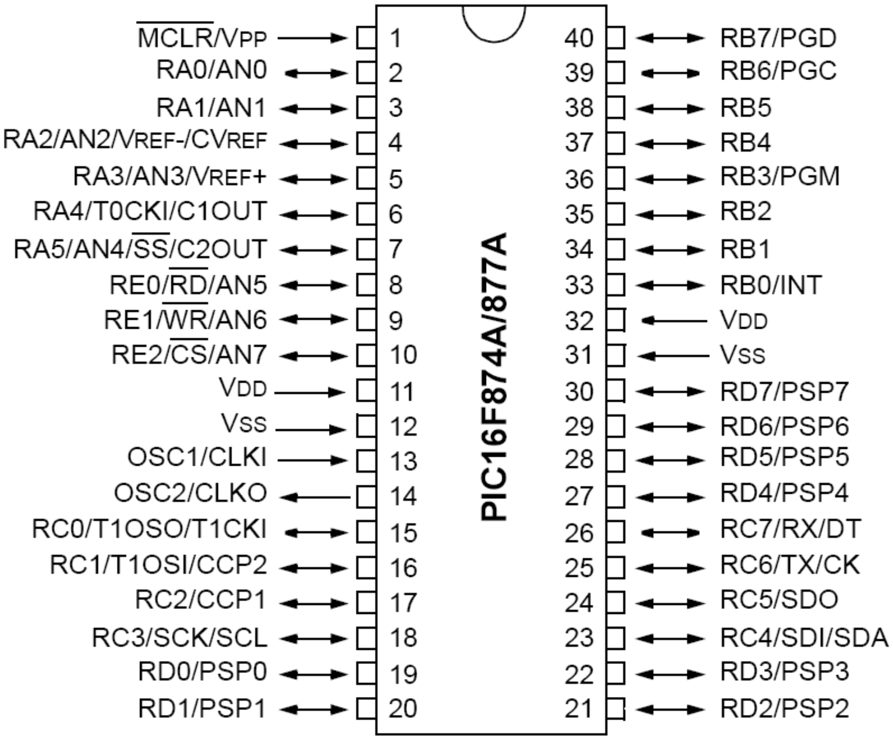


Figure 3.3: Pins of 16F877A

The code of the controller is written in CCS-C language and is compiled with PCWHD.

A 4Mhz Crystal oscillator is connected external timers to pins 13 and 14. The other parts are connected as shown in the figure 3.2.

3.1.2 Tuner Device

Cherub WST – 550 is chosen for this project. It is very small and useful device. It has a microphone and led interface. Interface is made up 23 leds. 12 of them Show the note and 11 of them show the pitch. The device Works with 3v battery and leds consume 1.8v.

Although connecting leds by transistors directly to the circuit seems as if it is a good idea, Connecting two different closed circuit to eachother is more complicated work than it seems. Because they works with different voltages and my electronic knowledge could be insufficient in case of any problem. So connecting the circuits by photodiodes seems more easier than connecting directly. To make this, a special box is prepared. The box has 16 slots to mount desired leds.

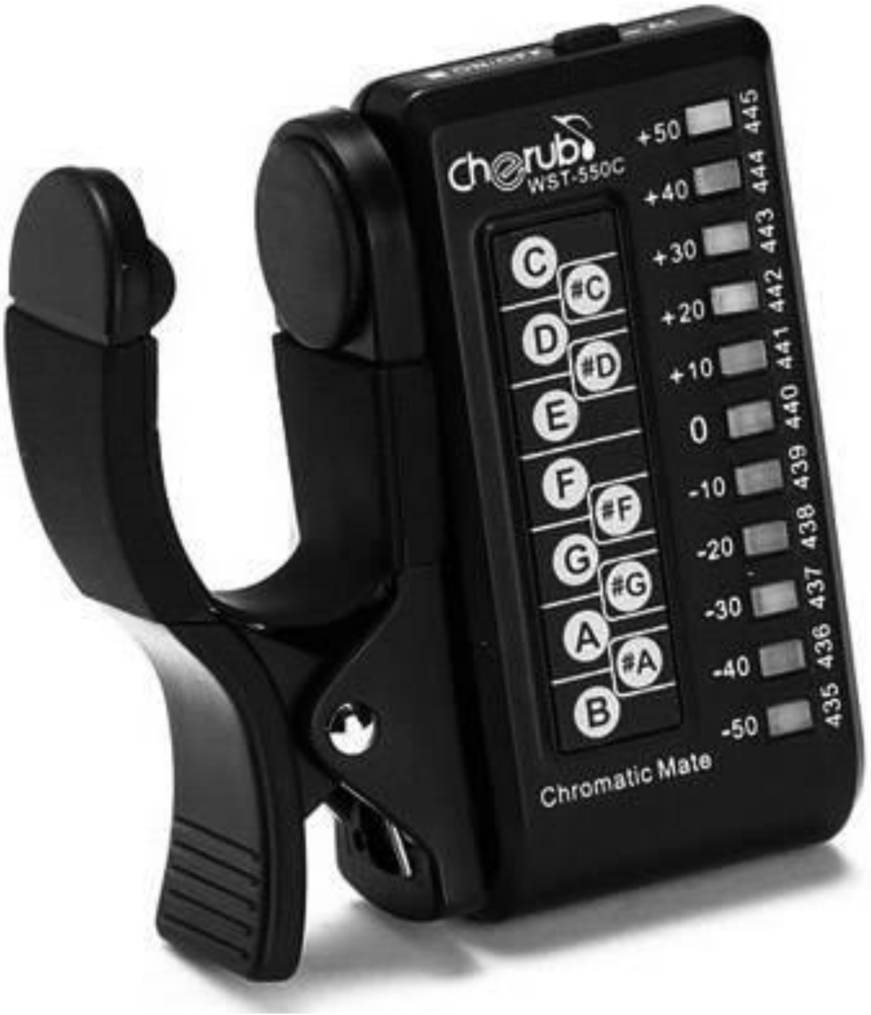


Figure 3.4: Cherub WST-550C

D,E,G,B,A and 11 pitch leds connected to the tuner are unmounted for cabling to the box.

After removing, 16 leds are placed in the box. 5 of them are used for analysing notes (e-a-d-g-b) and the others are used for analysing the pitch of the note (+5 to -5).

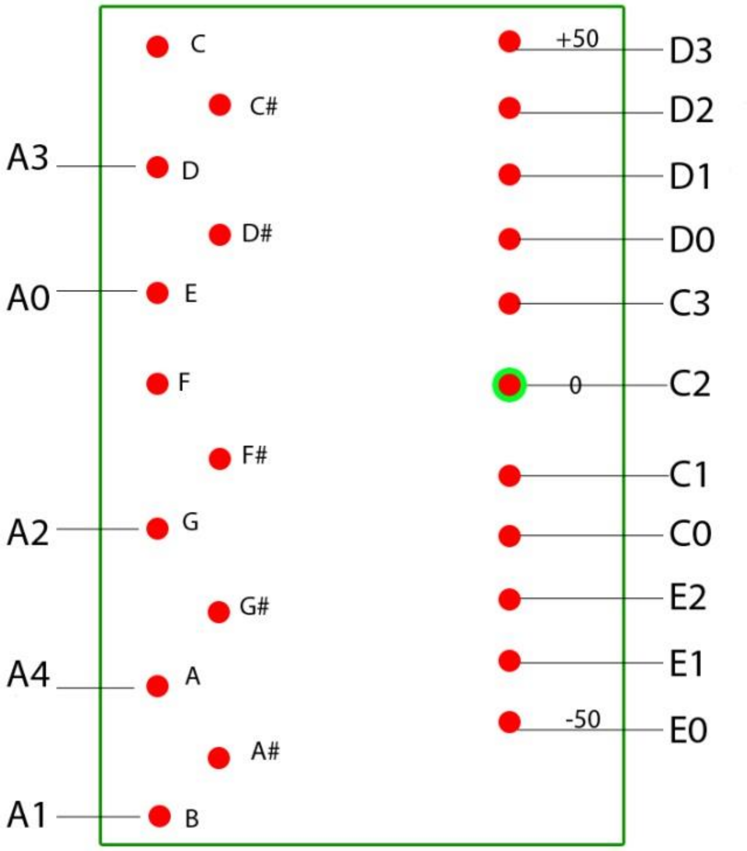


Figure 3.5 : Connection between outputs of tuner and input pins of PIC

After connecting the leds to port A legs of PIC by photodiodes, notes take 6 bit binary values but actually it has 5 bit data because 5th pin of port A is free.

PORTS	A5	A4	A3	A2	A1	A0
NOTES						
A	0	1	0	0	0	0
D	0	0	1	0	0	0
G	0	0	0	1	0	0
B	0	0	0	0	1	0
E	0	0	0	0	0	1

Table 3.2: Note values of Port A

Most of the tuner devices work according to chromatic scale. There are 12 pitches including all the tones and semitones. It can be displayed as “ F- F#- C - C# - G - G# - D - D# - A – A# - E - B ” Thus, we can tune an instrument also by transposing. On this Project, A chromatic tuner is used but all of the indicators weren’t used. To tune a guitar, we need only 5 notes.

3.1.3 Tuning Box

Tuning Box is a box containing a tuner device, 16 holes with 16 leds and photodiodes connected to respectively tuner and PIC in it. The Box comprises three parts. The main part is the board of a tuner device(Cherub WST – 550). Frequency of the incoming audio signal is processed in this part. After analysing frequency, device indicates the note of incoming signal by the leds in the Tuning Box. Cherub WST – 550 tuner device actually has 12 leds to indicate the note of the string and 11 leds to indicate the pitch of any note.

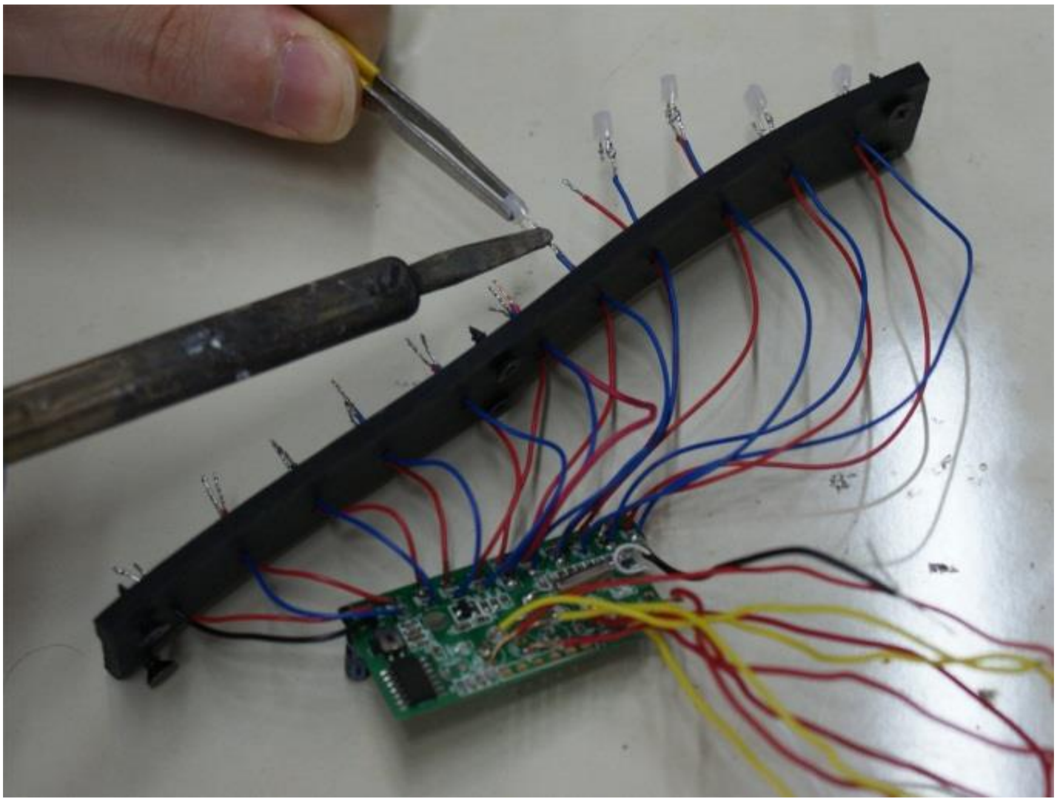


Figure 3.6: Construction phase of tuning box

From the photodiodes in the tuning box, the cables are connected to A port of the processor in the guitar string order. When any led light up, the photodiode in front of the led will send 2-5v to the PIC.

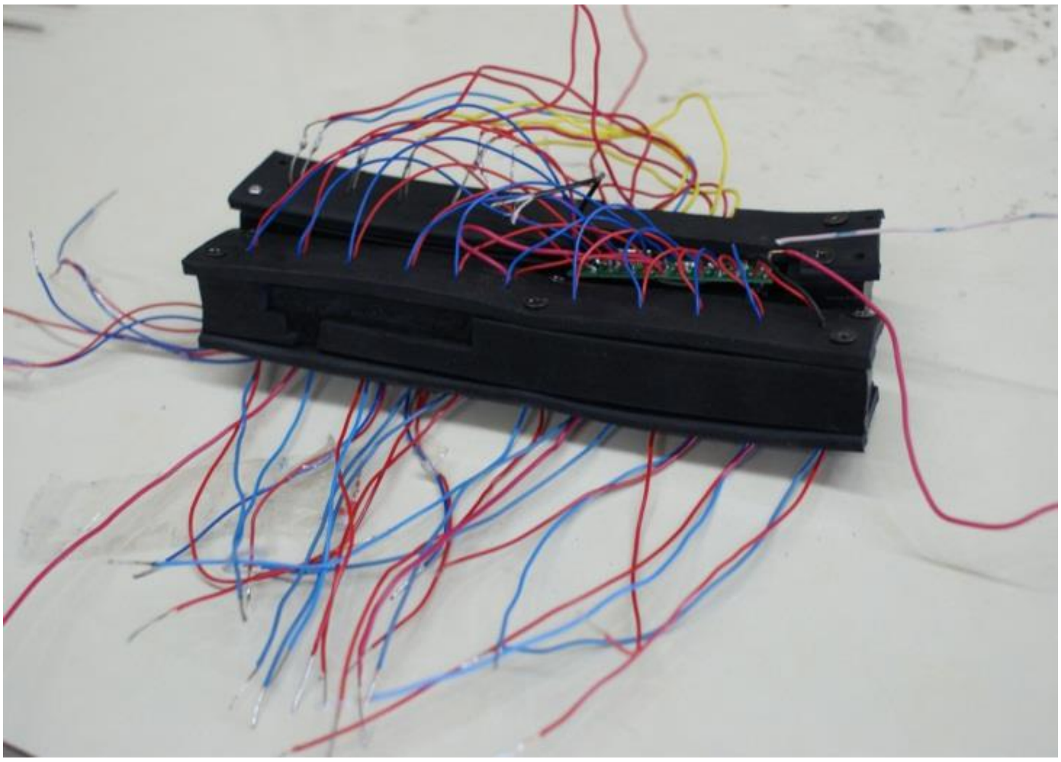


Figure 3.7: Tuning Box

This box designed to run and debug this device. It will be removed from the circuit and the device will be shaped again before mass production. It will be smaller and more ergonomic.

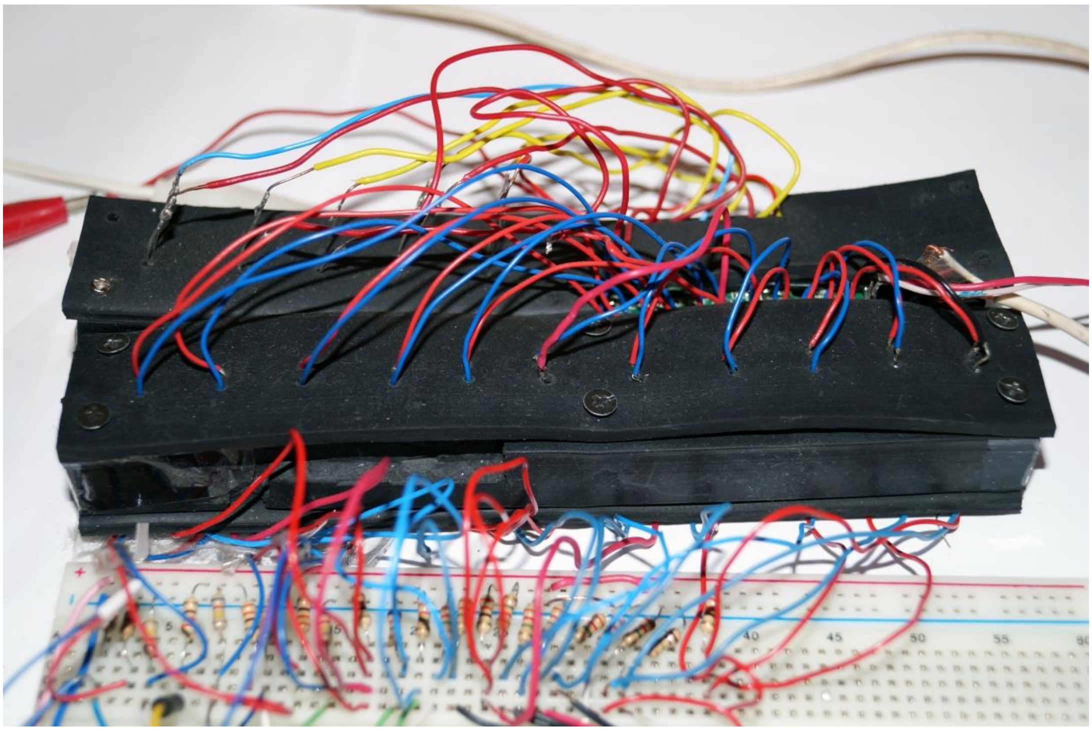


Figure 3.8 :Integration of tuning box into the system.

3.1.4 Photodiodes

Photodiodes were used to capture signal coming from the leds in tuning box. When any of the leds lights up, the photodiode beside led is triggered and current passes to connected pin of the PIC through the photodiode.

BPW34 is a PIN photodiode which provides high light sensitivity and high speed. It has two pin and 2.7mmx 4.3mm x 4.7mm top view case. In 7.5mm^2 area, it is very sensitive to visible and infrared frequencies.

Figure 3.9: BPW 34 Photodiode

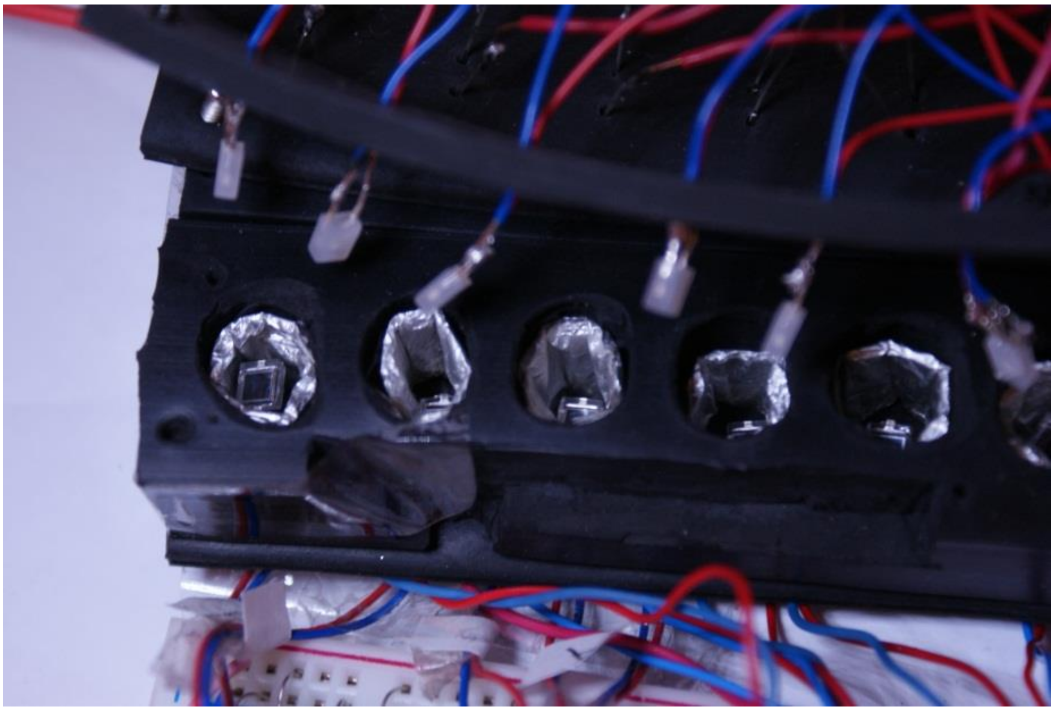


Figure 3.10: Photodiodes in the holes of Tuning Box.

COMPONENT	Ira (µA)	φ (deg)	λ 0.1 (nm)
BPW34	50	± 65	430 to 1100

Features of BPW34

“

- Dimensions (L x W x H in mm): 5.4 x 4.3 x 3.2
- Radiant sensitive area (in mm2): 7.5
- High photo sensitivity
- High radiant sensitivity
- Fast response times
- Angle of half sensitivity: 65°
- Compliant to RoHS Directive 2002/95/EC and in accordance to WEEE 2002/96/EC”[15]

3.1.5 Leds

“A **light-emitting diode** (**LED**) is a semiconductor light source.”[16] Because of their low costs and durability, LEDs are commonly used in consumer products as indicator and lighting part. To provide enough luminous efficacy, red is chosen as led color. “The TLH.46.. series was developed for applications which need a very wide radiation angle like backlighting, general indicating and lighting purposes. It is housed in a 3 mm tinted total diffused plastic package. The wide viewing angle of these devices provides a high on-off contrast. Several selection types with different luminous intensities are offered. All LEDs are categorized in luminous intensity groups. The green and yellow LEDs are categorized additionally in wavelength groups. That allows users to assemble LEDs with uniform appearance.”[17]

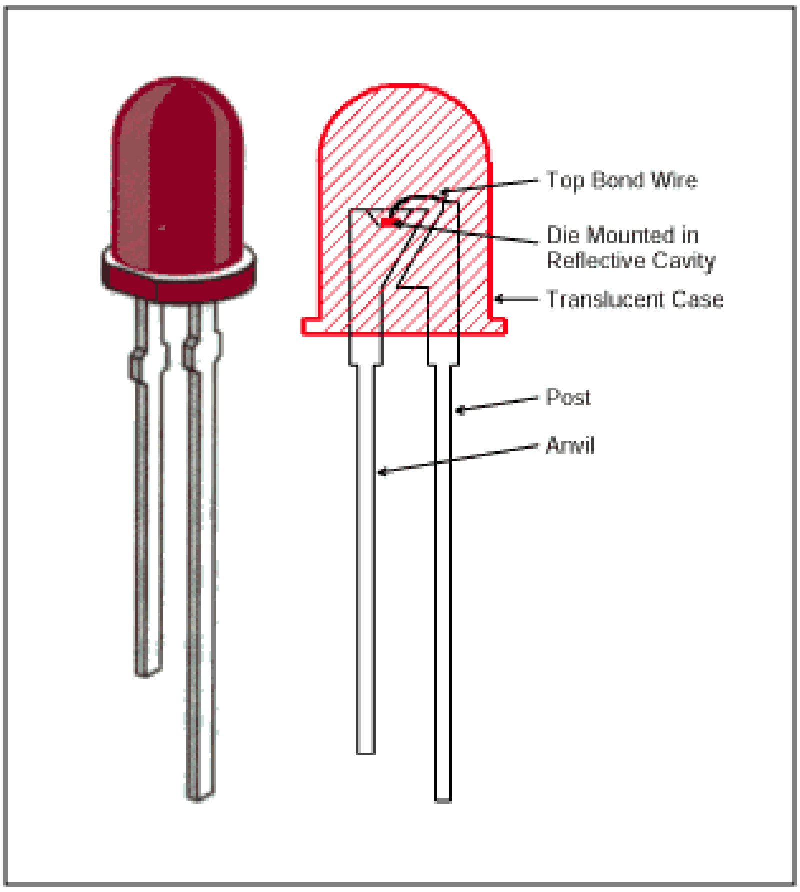
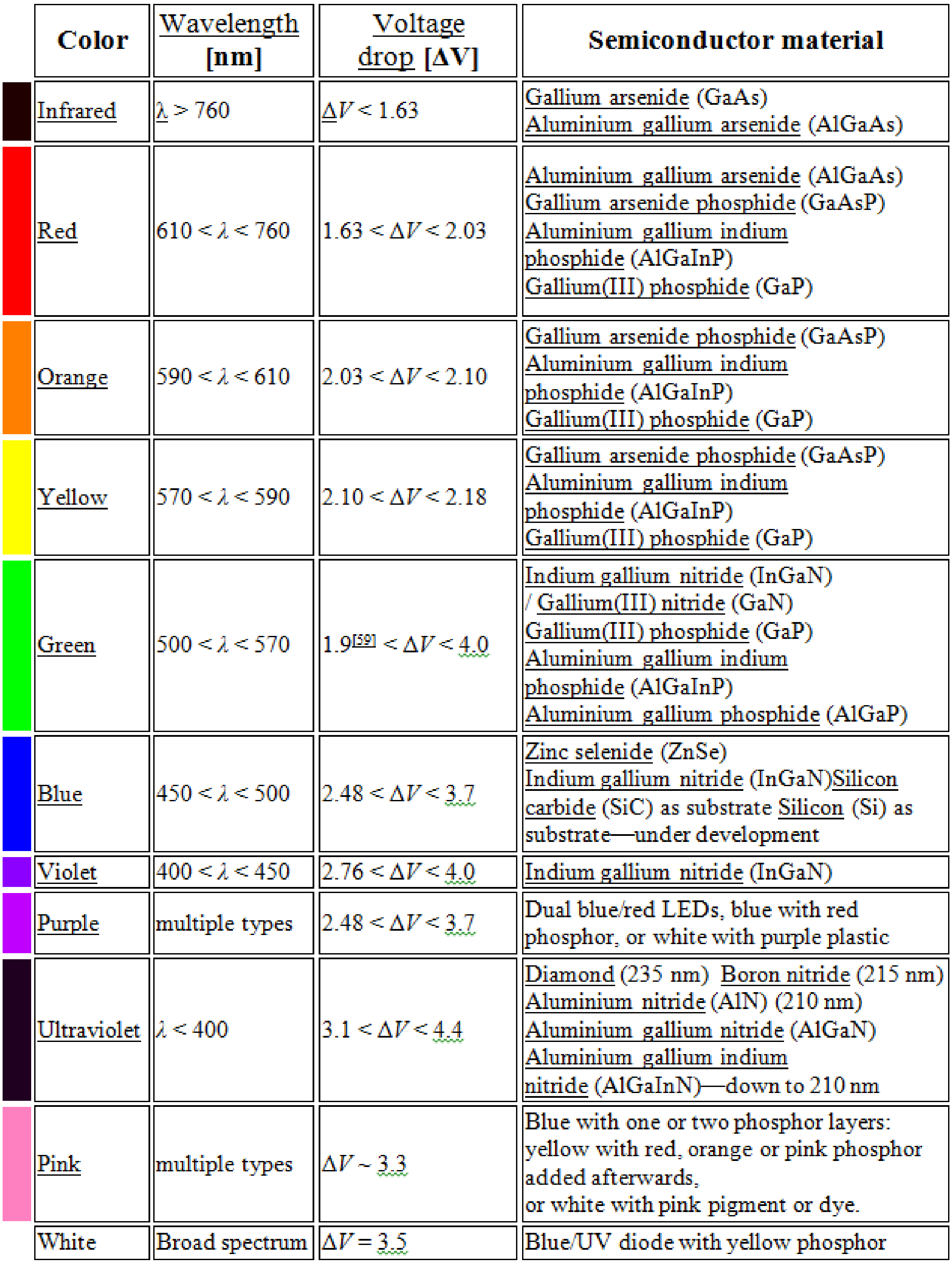


Figure 3.11: Structure of Led[18]

On this Project, 3mm red colored leds were used to operate photodiodes, because the maximum sensivity is reached at 850-950 nm wavelength . Accordance with this information, red led was most suitable and efficient led fort his project as shown in the table 3.3.



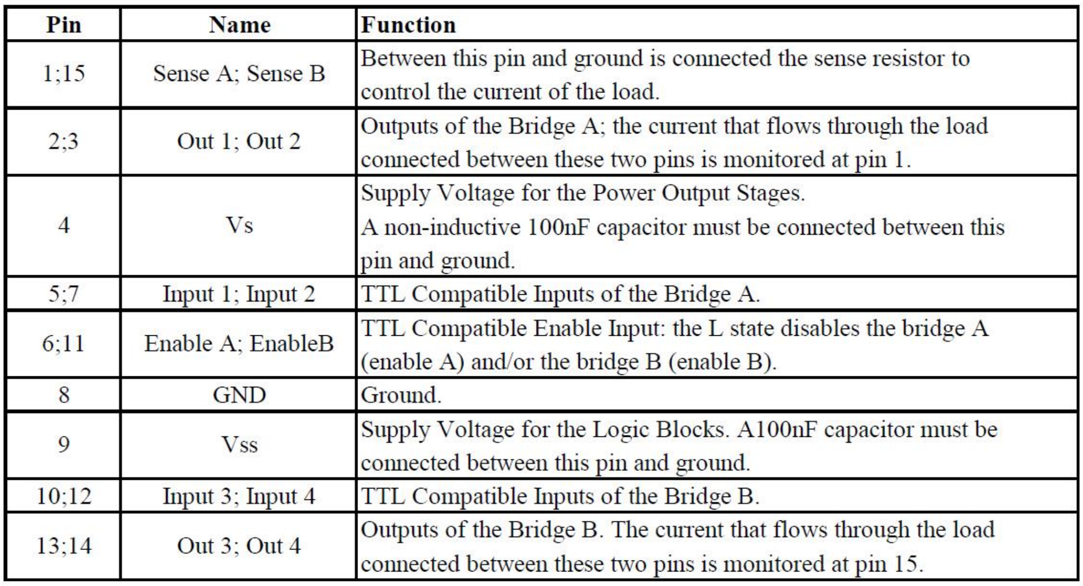
	Color	Wavelength [nm]	Voltage drop [ΔV]	Semiconductor material
⬛	Infrared	$\lambda > 760$	$\Delta V < 1.63$	Gallium arsenide (GaAs) Aluminium gallium arsenide (AlGaAs)
🟥	Red	$610 < \lambda < 760$	$1.63 < \Delta V < 2.03$	Aluminium gallium arsenide (AlGaAs) Gallium arsenide phosphide (GaAsP) Aluminium gallium indium phosphide (AlGaInP) Gallium(III) phosphide (GaP)
🟧	Orange	$590 < \lambda < 610$	$2.03 < \Delta V < 2.10$	Gallium arsenide phosphide (GaAsP) Aluminium gallium indium phosphide (AlGaInP) Gallium(III) phosphide (GaP)
🟨	Yellow	$570 < \lambda < 590$	$2.10 < \Delta V < 2.18$	Gallium arsenide phosphide (GaAsP) Aluminium gallium indium phosphide (AlGaInP) Gallium(III) phosphide (GaP)
🟩	Green	$500 < \lambda < 570$	1.9[59] $< \Delta V < 4.0$	Indium gallium nitride (InGaN) / Gallium(III) nitride (GaN) Gallium(III) phosphide (GaP) Aluminium gallium indium phosphide (AlGaInP) Aluminium gallium phosphide (AlGaP)
🟦	Blue	$450 < \lambda < 500$	$2.48 < \Delta V < 3.7$	Zinc selenide (ZnSe) Indium gallium nitride (InGaN)Silicon carbide (SiC) as substrate Silicon (Si) as substrate—under development
🟪	Violet	$400 < \lambda < 450$	$2.76 < \Delta V < 4.0$	Indium gallium nitride (InGaN)
🟪	Purple	multiple types	$2.48 < \Delta V < 3.7$	Dual blue/red LEDs, blue with red phosphor, or white with purple plastic
⬛	Ultraviolet	$\lambda < 400$	$3.1 < \Delta V < 4.4$	Diamond (235 nm) Boron nitride (215 nm) Aluminium nitride (AlN) (210 nm) Aluminium gallium nitride (AlGaN) Aluminium gallium indium nitride (AlGaInN)—down to 210 nm
■	Pink	multiple types	$\Delta V \sim 3.3$	Blue with one or two phosphor layers: yellow with red, orange or pink phosphor added afterwards, or white with pink pigment or dye.
	White	Broad spectrum	$\Delta V = 3.5$	Blue/UV diode with yellow phosphor

Table 3.3 : Wavelength range, voltage drop and material of the led [19]

3.1.6 DC Motor Driver

When a power supply is connected directly to any DC Motor, it will supply the motor with constant power. So, speed will change according to the load of motor. If load increases, rotational speed of the motor will decrease. In case of reduced load, rotational speed will increase. For this reason, the supply voltage of the DC motor must be controlled by any DC Motor Driver. Full bridge motor drivers enable to control high current motors . To wind and unwind the guitar pegs, sometimes high power can be needed. Thus motor demands higher current.

Most of the drivers on the market use the common "H Bridge" to provide directional control to the motors. H Bridge is a very effective method for driving motors. The DC motor driver that is used in this project is L298 dual H-bridge motor driver.
It comes in a staggered, 15-lead Multiwatt package. It supports 0-46V supply voltage and 0-2A current for each bridge. Overtemperature protection and high noise immunity are other important features. Table 3.4 shows function of each pin of chip L298.



Pin	Name	Function
1;15	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	Vs	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	Enable A; EnableB	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	GND	Ground.
9	Vss	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
10;12	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13;14	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.

Table 3.4 Pin function of chip L298

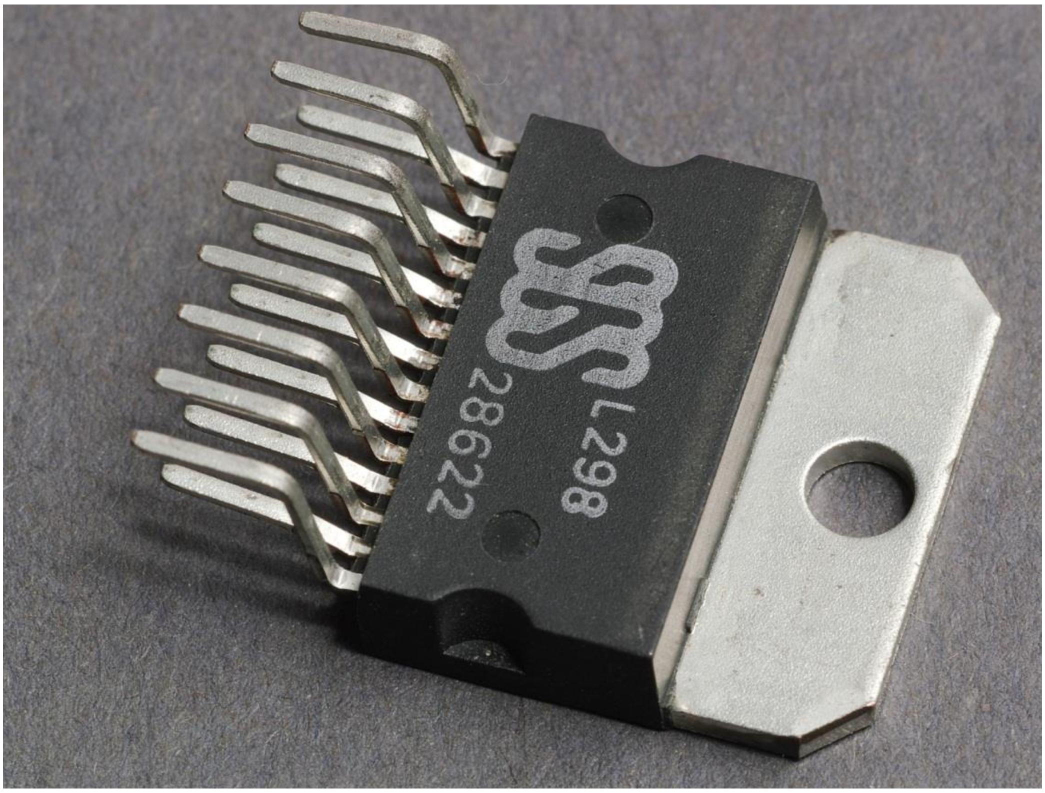
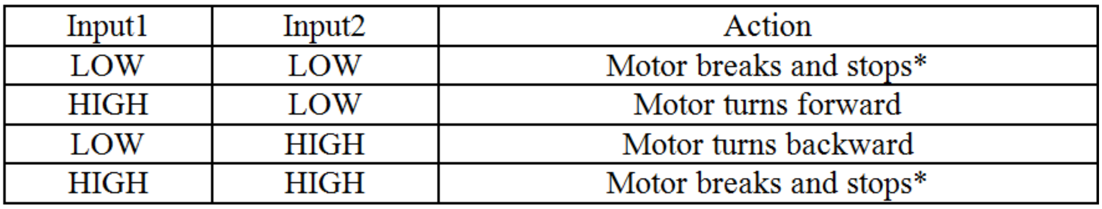


Figure 3.12: L298 Motor Driver

In this device, There are 15 pins. IN1, IN2, OUT1, OUT2, Enable A, Sense A is used for A Bridge and IN3, IN4, OUT3, OUT4, Enable B, Sense B is used for B Bridge.

IN1 and IN2 are inputs for A Bridge and IN3 and IN24are inputs for B Bridge. 6th (Enable A) and 11th (Enable B) pins are used for enabling the bridge. On this Project, only A Bridge is used.

If we assume that EnableA= HIGH and EnableB = LOW, according to signals coming into input1 and input2 of the motor driver, status and direction of the motor will be as shown in the table 3.5.



Input1	Input2	Action
LOW	LOW	Motor breaks and stops*
HIGH	LOW	Motor turns forward
LOW	HIGH	Motor turns backward
HIGH	HIGH	Motor breaks and stops*

Table 3.5: Actions of input values

220 Ohm resistors added between PIC and L298 to pretend PIC from reversed current which caused from L298.

2nd and 3rd pins are out pins used for A Bridge. They are connected to the motor through diodes to protect L298 from the reverse current.

two different supply voltages are used for L298. pin 9 is used to power the chip by 5 volts. Besides, 4th pin is able to supply 0-46V power for motors. But in this project, it supplies the motor by 12V because of the requirements. Sense A (pin 1) is used to set current which is sent to the motor. 2.2 Ohm resistor used to provide enough power to the motor in order to achieve 7 kg/ cm torque.

V=IxR => I=V/R => I= 12/ 2.2 => I=5.45A. But L298 can supply the motor with only 2A. From the formula W=VxI, power is 24W.

In theory,
Torque(N−m)= Power(W) /2π × Rotational speed(rps)
Torque= 12V x 2A / 6.28 x 4(rps)
T= 24 / 25,12 = 0,955N-m = 9,5 kg/cm.

9,5 kg/cm torque is enough to turn little oxidized pegs.It can control two motors simultaneously providing 2Amp current for each one, but just one motor is used in this project. L293 is enough to control one motor but it's supported current is maximum 1 A. Unlike the L293, the L298 does not have internal protection diodes to protect back EMF, so UF5408 diodes were inserted to the circuit. The L298 can handle 2 amps per motor and has a large cooling flange with a hole in it to suppress the overheating effect of high current. It is easy to attach a metal part to it.

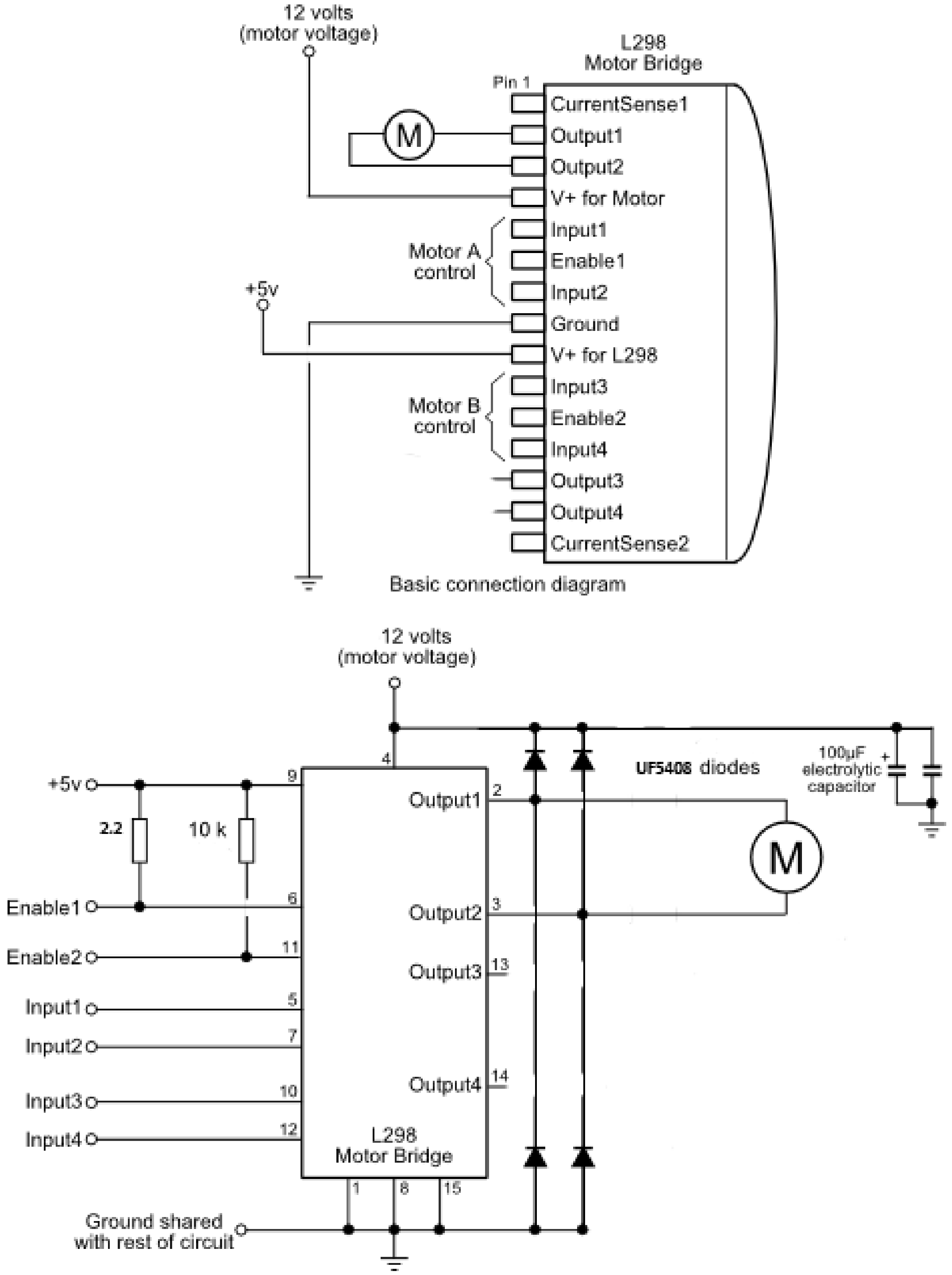


Figure 3.13: The circuit design of L298 motor bridge and connection of DC motor.

3.1.7 DC Motor

“A direct current (DC) motor is a device that converts electricity into the mechanical energy. A DC motor consists of two magnets of opposite polarity as rotor and stator to generate electromagnetic force. When current passes through the coil, the forces of polarized magnets create a torque which turns DC Motor. A DC motor requires at least one electromagnet. This electromagnet switches the current flow as the motor turns, changing its polarity to keep the motor running. The other magnet or magnets can either be permanent magnets or other electromagnets.”[20]

3.1.7.1 Why DC Motor?

Usually, Servo motor is choosen for similar projects, but in my opinion, we can get more precise results with using a strong DC motor on account of the fact that servo motors can turn step by step. But DC motors turn dependent on the given voltage, current and time. So controlling DC motor is much harder in comparison with servo motor for this project.

Figure 3.14: A DC Motor

When using a DC motor, the main problem is that speed and power are directly proportional. Namely, to turn the DC motor slowly, it is necessary to give low voltages. But geared DC motors can be used to overcome this problem. In Robotics, the most preferred motor type is DC motor. DC motors are cheap, small and efficient. The other reason is that they are very diverse in terms of shape, size and power.

DC motors can be used in any system directly or with gearboxes. Geared motors have a gear reduction system (gear box) to inrease torque reducing the output speed. There are some important basic factors in using DC motors.

3.1.7.2 Direction

When DC motors are connected to a power supply, the direction of Spin changes according to the direction of current. By reversing current direction, turn direction of the motor will reverse.

3.1.7.3 Speed

The speed of a motor is measured in rpm (revolutions per minute). Motor speed depends on the voltage and load.

There are two cases depending on the voltage and load to appraise the changes in the speed of a DC motor. The first of these; a system with a fixed load. In such a system, the speed of the DC motor depends on the applied voltage and voltage increases as speed increases. The second case; a system that the load of the motor changes according to the time or performed task. In this case, The speed of the motor will depend on the load. Applied power increases as the load increases and the speed decreases as power increases.

3.1.7.4 Voltage

Small DC motors can be found as having voltage values between 1,5v and 48v. Each DC motor has its own stable working voltage value on specified current, speed and power values. Failure to comply with this predefined values may result in severe mechanical problems due to the malfunction of the motor.

3.1.7.5 Current

“The stator is stationary in space by definition and therefore so is its current. The current in the rotor is switched by the commutator to also be stationary in space. This is how the relative angle between the stator and rotor magnetic flux is maintained near 90 degrees, which generates the maximum torque.” [21]

If a DC motor Works with fixed voltage, current is based on the load. If the load increases, it will result a proportional increase in current. Increasing current shouldn’t exceed the threshold value. In this case, short circuit can be occured and power turn into heat. According to the longevity of this situation, motor can burn. The supported current range of DC motors generally between 50mA and 2A.

3.1.7.6 Power

Product of current and voltage gives Power. But in mechanical systems, it is considered as Torque (**Moment of Force)**. Powerful motors have high torques. Torque is based on the electrical and mechanical characteristic and radius of the shaft of a motor. Torque of a motor can bechanged through gearboxes inserted to the motor. They provide to increase power causing a decrease on speed. Thus, to decide required motor, its torque and reduction ratio must be known.

High torque motors can be understand from their heavy weight, thick shaft and thick connection cables. When we try to rotate it, the resistance can be felt easily.

In this Project, An 60W high torque reduced DC Motor is used, but it has aproximately 10W power and 2A current on the circuit because of the capabilities of L298 Motor Driver. The motor provides aproximately 7-8 kg/cm torque and this is enough for tuning guitars.

3.1.8 String Selection Part

User can manually select the string with up/down buttons which he/she want to tune and can see the sign of the note which is desired to select on 7-Segment display. This part is made up of a seven-segment display, 74HC164 chip, two up-down buttons, one trigger button and our main processor (16F877A). By using up and down buttons, the note (e,a,d,g,b,e.) which is desired to select can be seen on the 7-Segment Display. By pressing trigger button, the data is sent to pic.

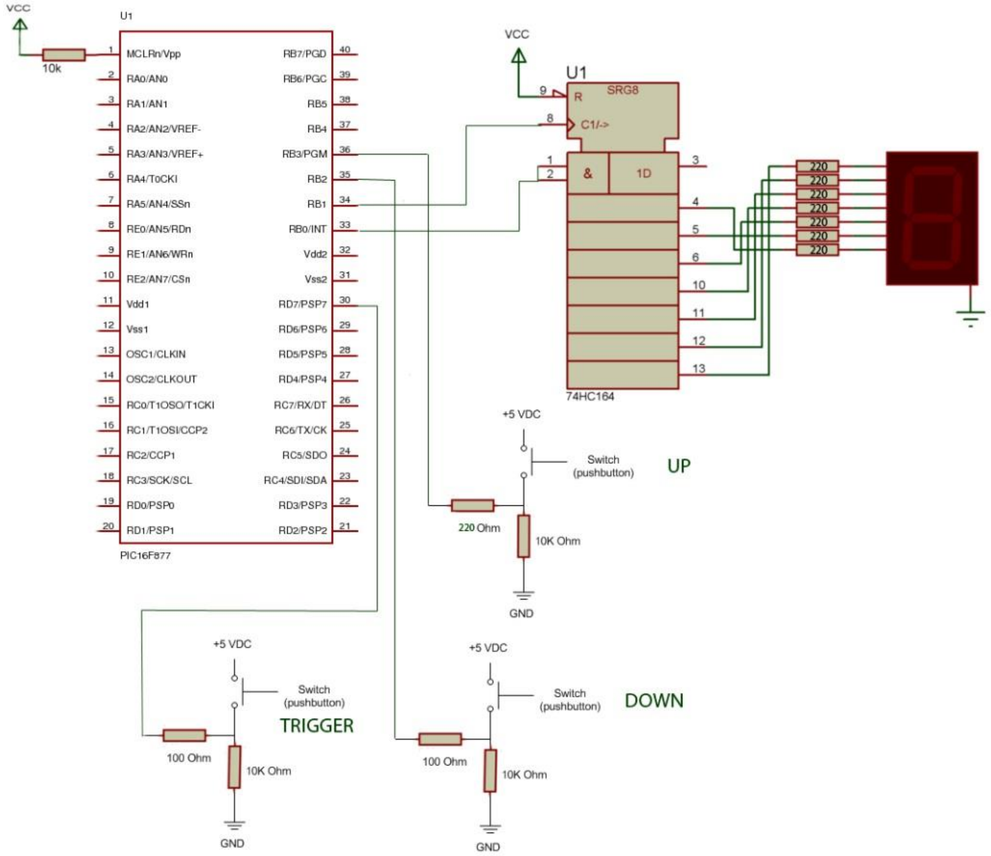


Figure 3.15: Circuit Design of String Selection Part

int digit[5]={0xde,0xfa,0x7a,0xf6,0x3e};

0xde= e (1. string)

0xfa = a (2. string)

0x7a= d (3. string)

0xf6= g (4. string)

0x3e= b (5. string)

6. string is same with first string.

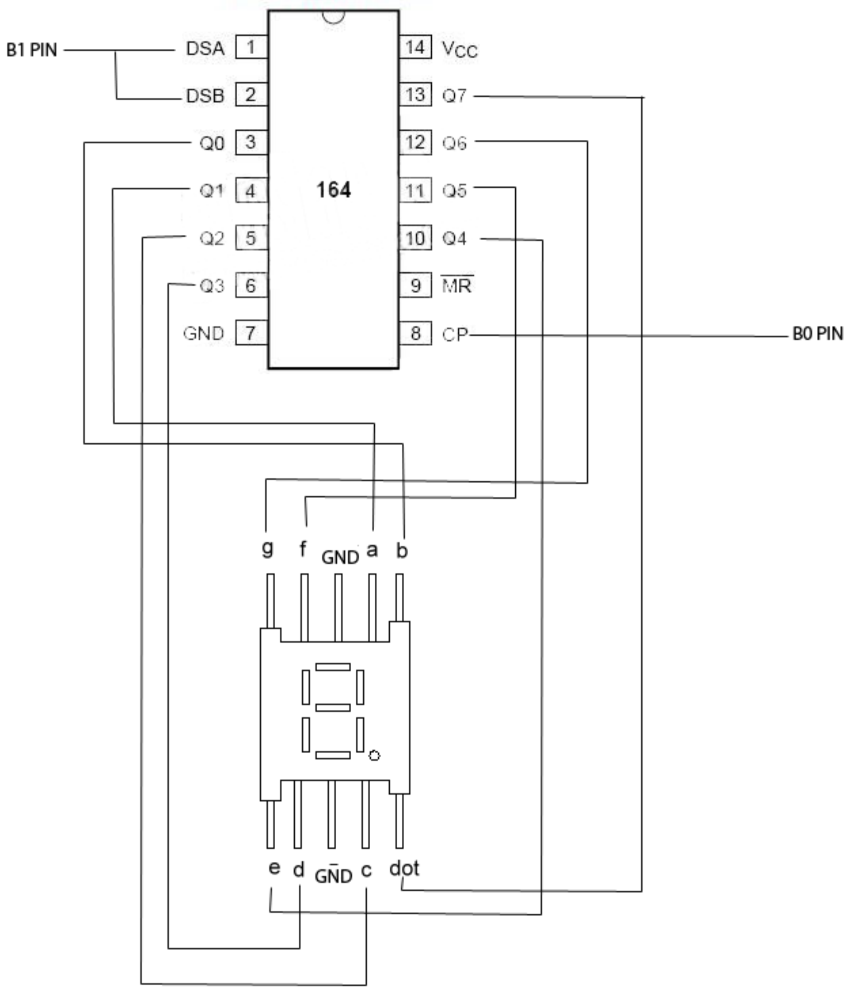


Figure 3.16: Circuit Design of 7-Segment Display and Shift Register

74HC164 is used as a shift register. It allows us to reduce the number of used pin for 7 – Segment Display. 2 pins are used instead of 8 pins.

It can be used by writing only 3 lines of code:

output_bit(data,bit_test(hangitel,j));

output_high(clock1);

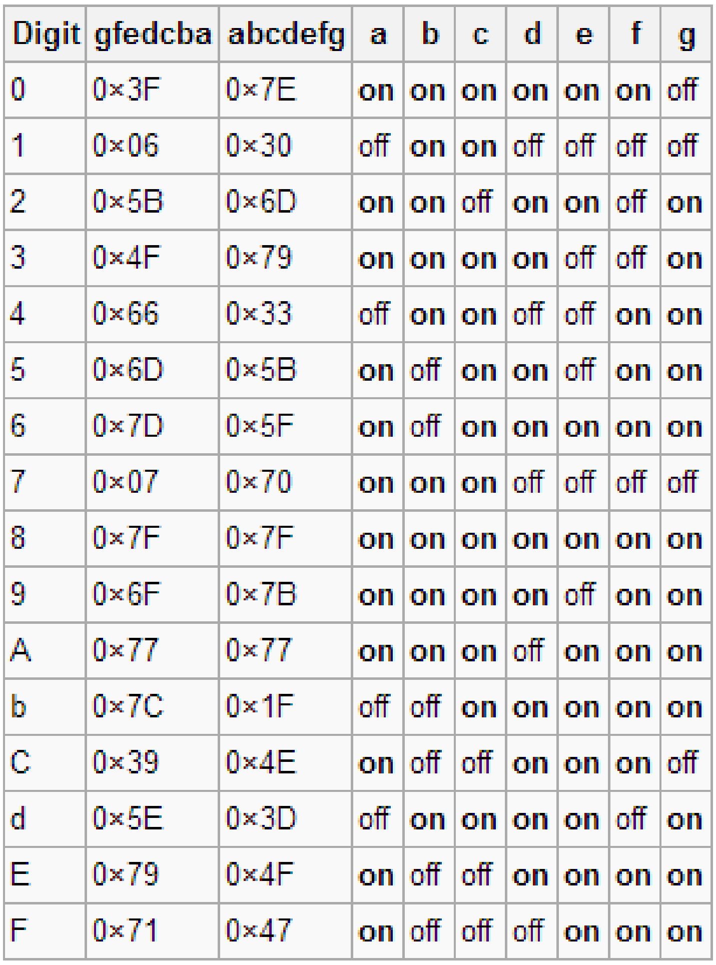
output_low(clock1);

3.1.9 7-Segment Display

“A seven-segment display (SSD), or seven-segment indicator, is a form of electronic display device for displaying decimalnumerals that is an alternative to the more complex dot-matrix displays. Seven-segment displays are widely used in digital clocks, electronic meters, and other electronic devices for displaying numerical information.”[22]

Decimal numbers, hexadecimal digits and some other letters easily can be displayed on 7-segment displays. D and B must be displayed as lowercase. Otherwise, they would look identical to 0 and 8.

On the Project, leds are not connected directly to the PIC. Because of the requirements of the Project, it is not useful to use 8 pins to connect 7-Segment Display. Owing to 74HC164, same work can be done with 2 pins.



Digit	gfedcba	abcdefg	a	b	c	d	e	f	g
0	0×3F	0×7E	on	on	on	on	on	on	off
1	0×06	0×30	off	on	on	off	off	off	off
2	0×5B	0×6D	on	on	off	on	on	off	on
3	0×4F	0×79	on	on	on	on	off	off	on
4	0×66	0×33	off	on	on	off	off	on	on
5	0×6D	0×5B	on	off	on	on	off	on	on
6	0×7D	0×5F	on	off	on	on	on	on	on
7	0×07	0×70	on	on	on	off	off	off	off
8	0×7F	0×7F	on	on	on	on	on	on	on
9	0×6F	0×7B	on	on	on	on	off	on	on
A	0×77	0×77	on	on	on	off	on	on	on
b	0×7C	0×1F	off	off	on	on	on	on	on
C	0×39	0×4E	on	off	off	on	on	on	off
d	0×5E	0×3D	off	on	on	on	on	off	on
E	0×79	0×4F	on	off	off	on	on	on	on
F	0×71	0×47	on	off	off	off	on	on	on

Table 3.6: Hexadecimal encodings for displaying the digits 0 to F.[22]

3.1.10 74HC164 Shift Register

“The 74HC164 is an advanced high-speed CMOS device fabricated with silicon gate CMOS technology. It achieves the high-speed operation similar to equivalent Bipolar Schottky TTL while maintaining the CMOS low power dissipation. The 74HC164 is a high-speed 8-Bit Serial-In/Parallel-Out Shift Register. Serial data is entered through a 2-input AND gate synchronous with the LOW-to-HIGH transition of the clock. The device features an asynchronous Master Reset which clears the register, setting all outputs LOW independent of the clock. An input protection circuit insures that 0V to 7V can be applied to the input pins without regard to the supply voltage. This device can be used to interface 5V to 3V systems and two supply systems such as battery backup. This circuit prevents device destruction due to mismatched supply and input voltages.”[23]

It features high Speed (fMAX = 175MHz at VCC = 5V), low power dissipation (ICC = 4µA (max.) at TA = 25°C), high noise immunity (VNIH = VNIL = 28% VCC (min.)), power down protection provided on all inputs, low noise (VOLP = 0.8V (max.))

It is used as a multiplexer on the project to reduce used pins of microcontroller.

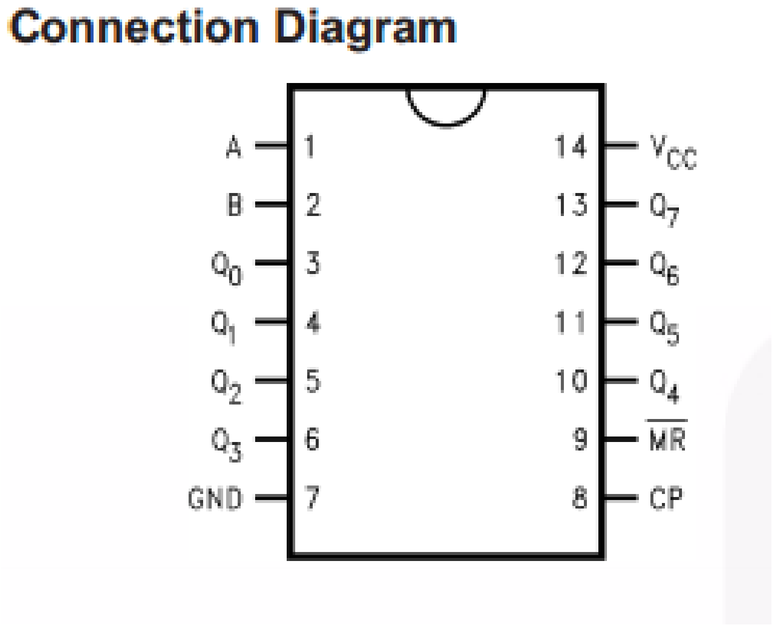


Figure 3.17: Connection Diagram of 74HC164

The 74HC164 is an edge-triggered 8-bit shift SIPO register. Data is entered serially through A and B inputs. These input pins are connected together with logical NOT AND. 9th pin is master reset. All output pins (Q0-Q7) is set to low if MR is low. Each Clock input on the 8th pin shifts data one place to the right.

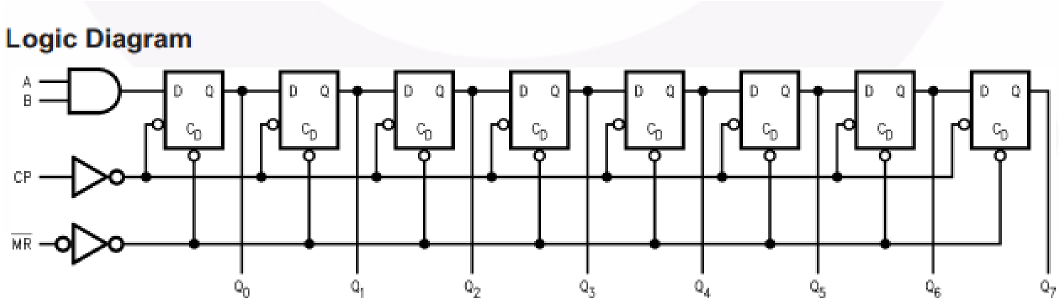


Figure 3.18: Logic Diagram of 74HC164

3.1.11 Diodes

Diodes provide safety to the circuit if the battery terminals turn into reversed order by any reason. In this Project, they are used to protect the circuit from back EMF (Back Electromotive Force) which is generated by the motor. [24]

Soft Recovery Ultrafast Plastic Rectifier UF5408 is used to build H- Bridge. It is needed to turn DC motor both clockwise and counterclockwise.

It features a glass passivated chip junction, ultrafast reverse recovery time, low forward voltage drop, low switching losses, high efficiency,high forward surge capability, solder dip 275 °C max. 10 s, per JESD 22-B106, compliant to RoHS directive 2002/95/EC and in accordance to WEEE 2002/96/EC.The circuit is designed as shown in the figure 3.13.

3.1.12 PIC Programmer Unit

The hex code of the program was imported to the PIC with using US-BURN Brenner-8 USB PIC Programmer and UsBurn USB Programmer software.

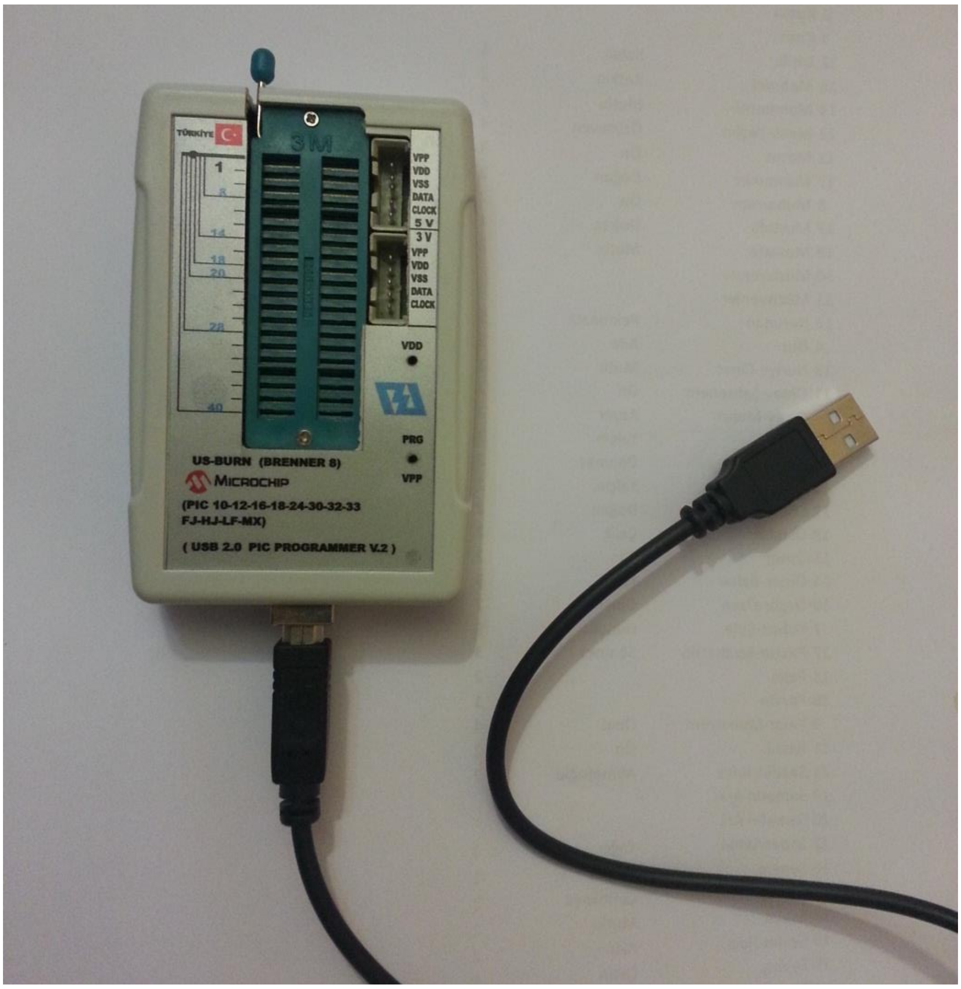
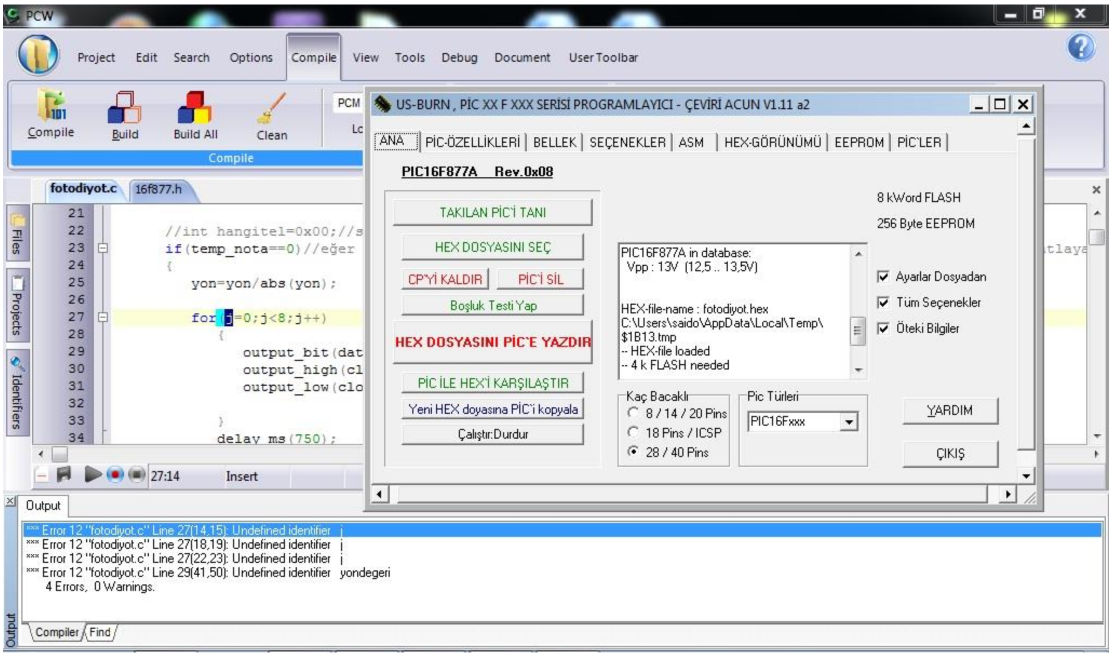


Figure 3.19: USB PIC Programmer

Brenner-8 supports Windows OS. USB v1.11 a2 is the software program of this device. It was executed on Windows 7 OS in this Project as Administrator. Interface of the program is shown in the figure 3.20 and 3.21. The program allows us to make detailed configuration in 8 different tab, but only the basic tab is used on this Project. To write a hex code into the PIC, we must press the buttons in order.

Firstly the PIC must be identified. Then hex code can be selected. To write the hex code into PIC, we must remove CP and erase PIC. After writing task, we can crosscheck the hex codes by pressing “Compare PIC with Hex file” button. If the status is OK, PIC is ready to run.



US-BURN , PİC XX F XXX SERİSİ PROGRAMLAYICI - ÇEVİRİ ACUN V1.11 a2

PIC16F877A Rev.0x08

8 kWord FLASH

256 Byte EEPROM

```
PIC16F877A in database:
 Vpp : 13V  (12,5 .. 13,5V)

HEX-file-name : fotodiyot.hex
C:\Users\saido\AppData\Local\Temp\
$1B13.tmp
-- HEX-file loaded
-- 4 k FLASH needed
```

☑ Ayarlar Dosyadan
☑ Tüm Seçenekler
☑ Öteki Bilgiler

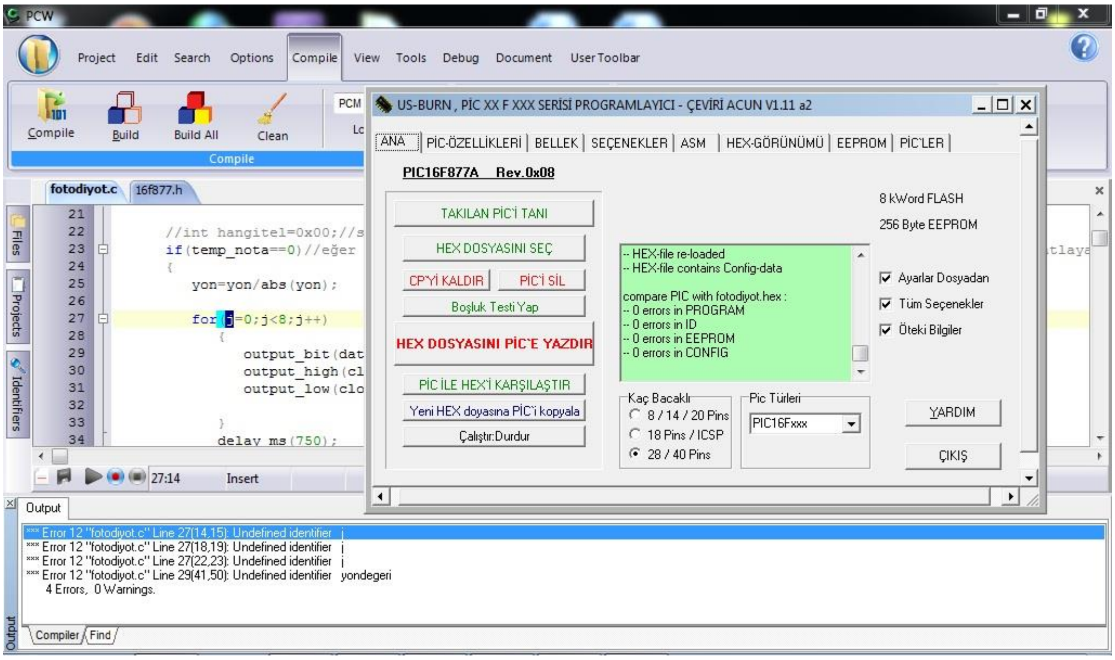
Kaç Bacaklı
○ 8 / 14 / 20 Pins
○ 18 Pins / ICSP
● 28 / 40 Pins

Pic Türleri: PIC16Fxxx

```
//int hangitel=0x00;//s
if(temp_nota==0)//eğer
{
   yon=yon/abs(yon);

   for(j=0;j<8;j++)
      {
         output_bit(dat
         output_high(cl
         output_low(clo

      }
      delay_ms(750);
```

```
*** Error 12 "fotodiyot.c" Line 27(14,15): Undefined identifier   j
*** Error 12 "fotodiyot.c" Line 27(18,19): Undefined identifier   j
*** Error 12 "fotodiyot.c" Line 27(22,23): Undefined identifier   j
*** Error 12 "fotodiyot.c" Line 29(41,50): Undefined identifier   yondegeri
    4 Errors,  0 Warnings.
```

Figure 3.20: Software interface of USB Pic Programmer



US-BURN , PİC XX F XXX SERİSİ PROGRAMLAYICI - ÇEVİRİ ACUN V1.11 a2

PIC16F877A Rev.0x08

8 kWord FLASH

256 Byte EEPROM

```
-- HEX-file re-loaded
-- HEX-file contains Config-data

compare PIC with fotodiyot.hex :
-- 0 errors in PROGRAM
-- 0 errors in ID
-- 0 errors in EEPROM
-- 0 errors in CONFIG
```

☑ Ayarlar Dosyadan
☑ Tüm Seçenekler
☑ Öteki Bilgiler

Kaç Bacaklı
○ 8 / 14 / 20 Pins
○ 18 Pins / ICSP
● 28 / 40 Pins

Pic Türleri: PIC16Fxxx

```
//int hangitel=0x00;//s
if(temp_nota==0)//eğer
{
   yon=yon/abs(yon);

   for(j=0;j<8;j++)
      {
         output_bit(dat
         output_high(cl
         output_low(clo

      }
      delay_ms(750);
```

```
*** Error 12 "fotodiyot.c" Line 27(14,15): Undefined identifier   j
*** Error 12 "fotodiyot.c" Line 27(18,19): Undefined identifier   j
*** Error 12 "fotodiyot.c" Line 27(22,23): Undefined identifier   j
*** Error 12 "fotodiyot.c" Line 29(41,50): Undefined identifier   yondegeri
    4 Errors,  0 Warnings.
```

Figure 3.21: Software interface of USB Pic Programmer

3.1.13 Other equipments

Three-terminal positive voltage regulator L7805 is used to convert 12V input into 5V and protect main circuit from the voltage surges. 5V is recommended operational voltage for the PIC and other electronic equipments except motor driver and tuner circuit of the tuner box.

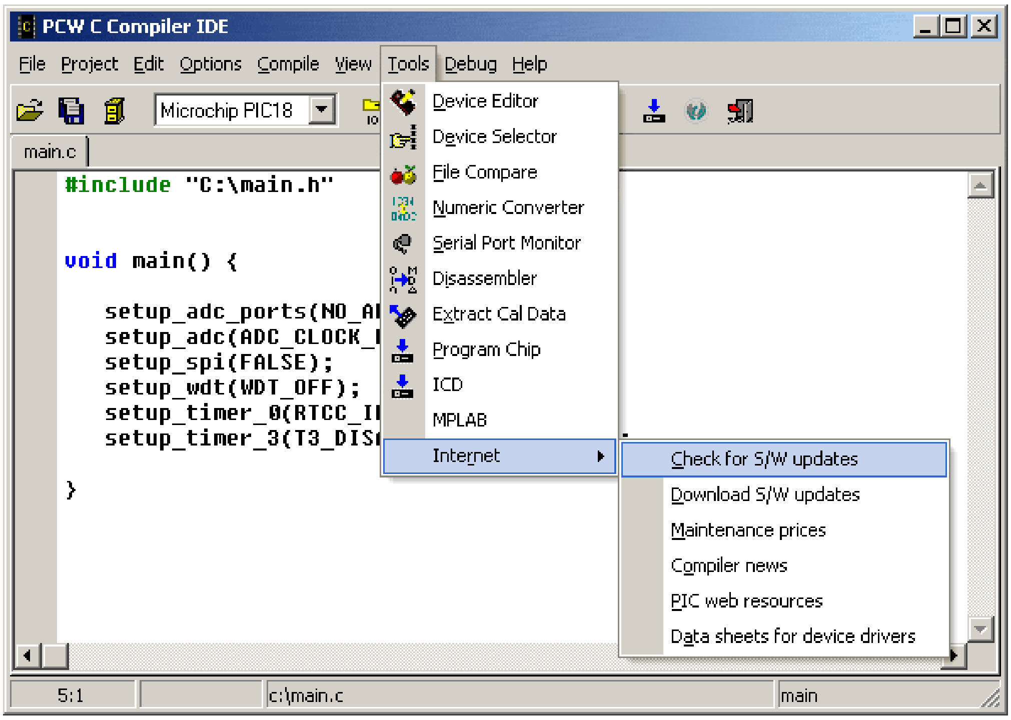
2 pieces of 22pf, 2 pieces of 220 nf and a 100nf Capacitors are used on the circuit. To protect 7- Segment Display and on the button circuit, 8 pieces of 330 Ohm resistors are used and also 1.2 MOhm and 330 Ohm resistors are used for each of 16 photodiodes. A 4 Mhz Crystal oscillator is connected 13th and 14th pins of PIC using 22pf capacitors.

3.2 SOFTWARE IMPLEMENTATION

3.2.1 CCS C Compiler

"CCS provides a complete, integrated tool suite for developing and debugging embedded applications running on Microchip PIC®MCUs and dsPIC® DSCs. This suite includes an IDE for project management, a context sensitive C aware editor, build tools and real time debugger...helping developers create, analyze, debug and document project code."[25] It supports the Microchip PIC12x, PIC16x, PIC18x, and the family of dsPIC devices. The compiler is nearly 100% ANSI compatible.

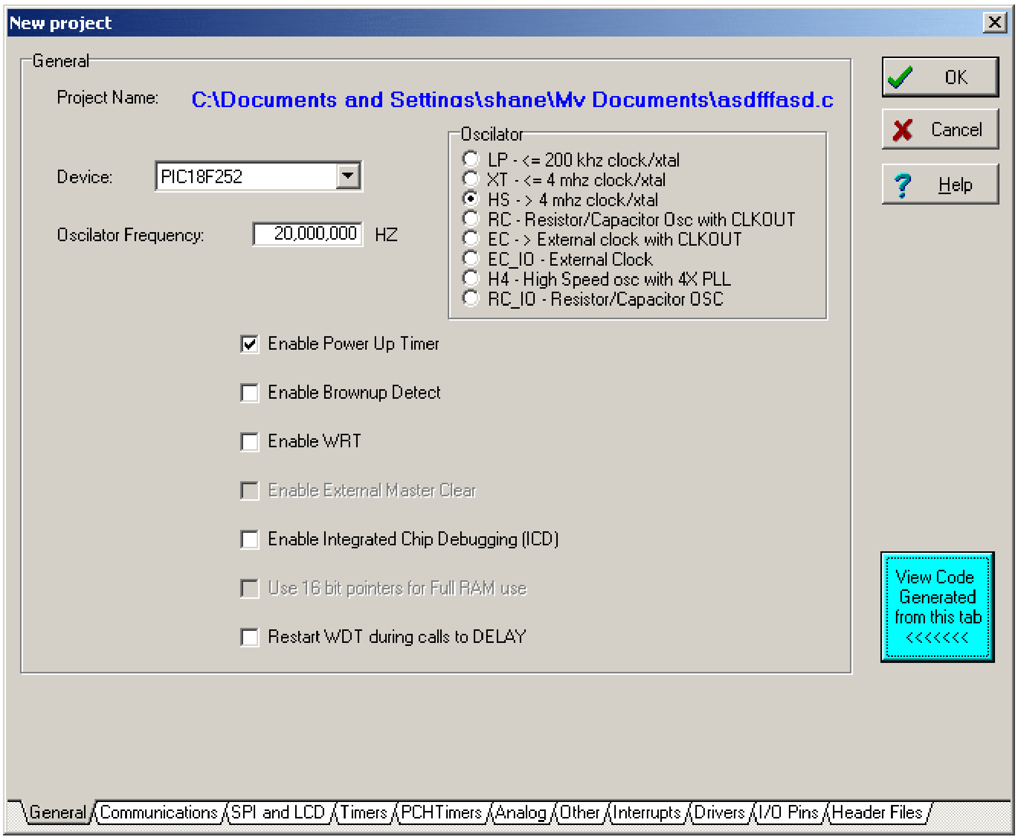
The IDE of the program is perfectly integrated into the compiler. It is configurable and all the menus can be customised again. The Interface is very user-friendly and very useful.



```
#include "C:\main.h"


void main() {

   setup_adc_ports(NO_Al
   setup_adc(ADC_CLOCK_I
   setup_spi(FALSE);
   setup_wdt(WDT_OFF);
   setup_timer_0(RTCC_II
   setup_timer_3(T3_DISı

}
```

Figure 3.22: IDE[26]

The Code Wizard is used to generate perfect template C code



New project
General
OK
Cancel
Help
View Code Generated from this tab <<<<<<<
General Communications SPI and LCD Timers PCHTimers Analog Other Interrupts Drivers I/O Pins Header Files

Project Name: C:\Documents and Settings\shane\My Documents\asdfffasd.c

Device: PIC18F252

Oscilator Frequency: 20,000,000 HZ

Oscilator
- ○ LP - <= 200 khz clock/xtal
- ○ XT - <= 4 mhz clock/xtal
- ● HS - > 4 mhz clock/xtal
- ○ RC - Resistor/Capacitor Osc with CLKOUT
- ○ EC - > External clock with CLKOUT
- ○ EC_IO - External Clock
- ○ H4 - High Speed osc with 4X PLL
- ○ RC_IO - Resistor/Capacitor OSC

☑ Enable Power Up Timer

□ Enable Brownup Detect

□ Enable WRT

□ Enable External Master Clear

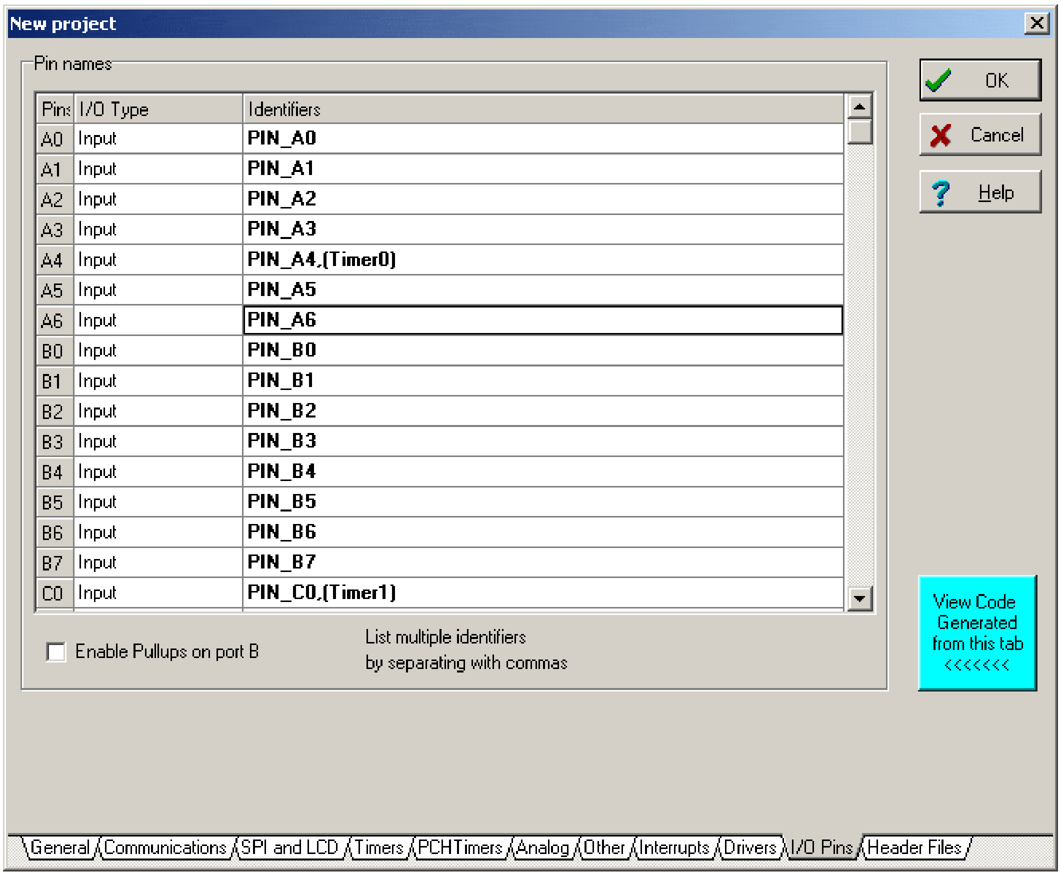
□ Enable Integrated Chip Debugging (ICD)

□ Use 16 bit pointers for Full RAM use

□ Restart WDT during calls to DELAY

Figure 3.23: Wizard setup for any particular PIC micro, and all its peripherals.[26]

The wizard provides user to set port directions, and values.



Pin names

Pins	I/O Type	Identifiers
A0	Input	PIN_A0
A1	Input	PIN_A1
A2	Input	PIN_A2
A3	Input	PIN_A3
A4	Input	PIN_A4,(Timer0)
A5	Input	PIN_A5
A6	Input	PIN_A6
B0	Input	PIN_B0
B1	Input	PIN_B1
B2	Input	PIN_B2
B3	Input	PIN_B3
B4	Input	PIN_B4
B5	Input	PIN_B5
B6	Input	PIN_B6
B7	Input	PIN_B7
C0	Input	PIN_C0,(Timer1)

□ Enable Pullups on port B

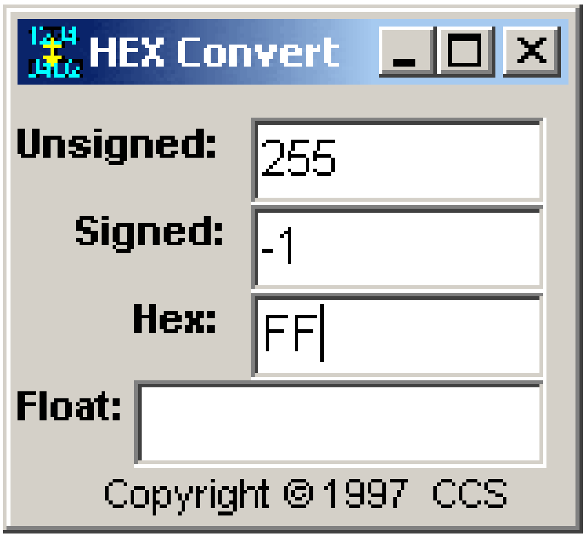
List multiple identifiers
by separating with commas

Figure 3.24: Wizard ports setup[26]

It has a wizard for RS232 and I2C support This specification was not used for the project because usb programming unit has its own interface for transmitting the code.

CCS C has a lot of official sample code and 70 complete projects. We can find millions of sample CCS C code also on the internet and market.

It has a hexadecimal convertor tool to make embedded programming easier. This little tool is used to convert between decimal, hexadecimal, and float data types under the menu "Tools >> Numeric Convertor"



Unsigned: 255

Signed: -1

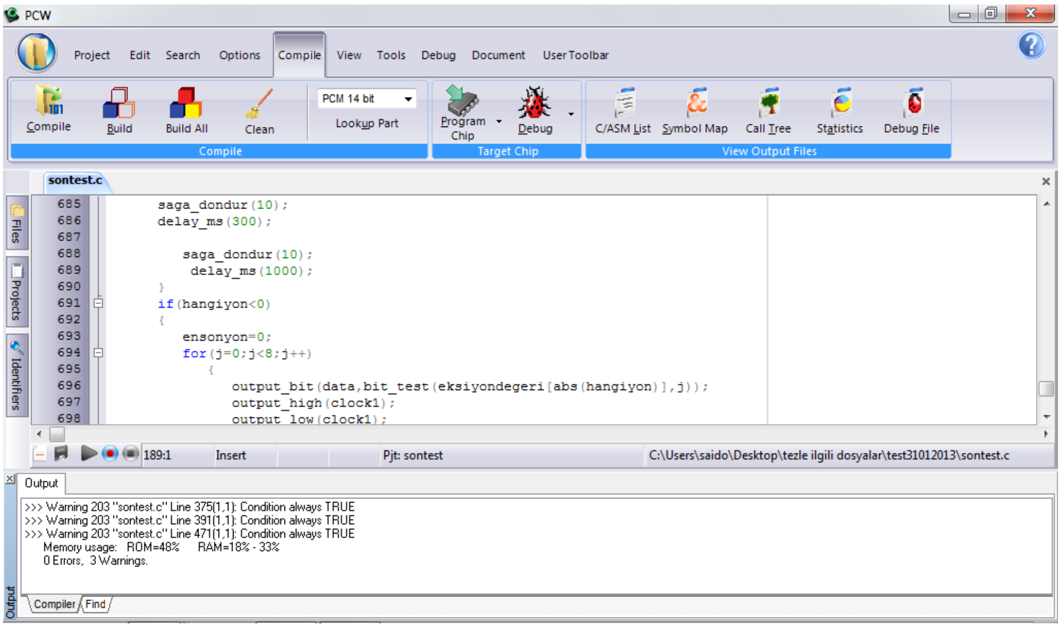
Hex: FF

Float:

Copyright ©1997 CCS

Figure 3.25: CCS Hexadecimal Conversion Tool.[26]

On this Project, Ccs C was used with PCWHD Compiler on Windows 7 OS. IDE, PCB, PCM,PCH and PCD versions are 4.084. It has very useful interface and writing code in this compiler is easy.



PCM 14 bit

sontest.c

```
        saga_dondur(10);
        delay_ms(300);

           saga_dondur(10);
            delay_ms(1000);
        }
        if(hangiyon<0)
        {
           ensonyon=0;
           for(j=0;j<8;j++)
              {
                 output_bit(data,bit_test(eksiyondegeri[abs(hangiyon)],j));
                 output_high(clock1);
                 output_low(clock1);
```

189:1 Insert Pjt: sontest C:\Users\saido\Desktop\tezle ilgili dosyalar\test31012013\sontest.c

```
>>> Warning 203 "sontest.c" Line 375(1,1): Condition always TRUE
>>> Warning 203 "sontest.c" Line 391(1,1): Condition always TRUE
>>> Warning 203 "sontest.c" Line 471(1,1): Condition always TRUE
    Memory usage:   ROM=48%      RAM=18% - 33%
    0 Errors,  3 Warnings.
```

Figure 3.26: CCS C Compiler

CHAPTER III

ALGORITHMS

The tuning box and A port of the processor are connected by the cables in the guitar string order. In other words, the led of the first note (E) is linked to a0 pin using a photodiode. Applying the same procedure, 2. note (B) is linked to a1 pin, 3. note (G) is linked to a2, 4. note (D) is linked to a3, 5. note (A) is linked to a4. A5 pin is left blank because the note of the 5. String (E) is same with the first one. Only the octaves are different.

Firstly tuning process seems so easy because the notes are connected in order. But because there are another notes between two strings, the process is much more complicated. When we first look, for example to tune third string, if there is a signal from the tuner to a port 3. Pin, then tuning process seems correct. If there is a signal to first and second pind, then the note value is high, and motor has to be turned to loosen the string. If there is a signal to 4. And 5. pin , then the note value is low and motor has to be turned to tighten the string.

But there are things beyond these processes. Between the two strings, there is at least one note of another string. For example there is E note between the 2. (B) note and 3. (C) note. Sequence of the notes are shown in figure 4.1. So when we tune the 3. String, if there is the B note from the guitar, it must be loosened a little. After loosen the string it will probably sound as A note. If we assume the guitar string order as the acord order, a note is actually a note of 5. String it must be tightened. But it actually must be loosend a little more.

Another complicated algorithm is needed to fix this problem. Figure 4.1 shows the notes and how they must be considered when we are tuning any string.

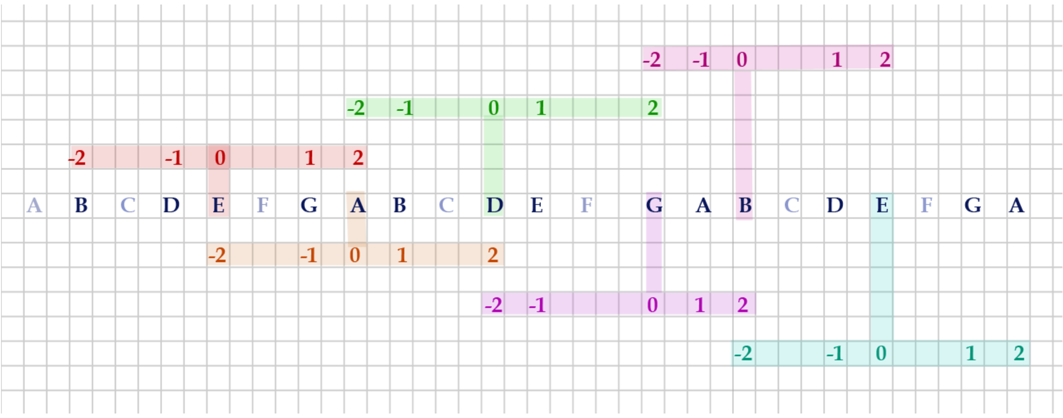
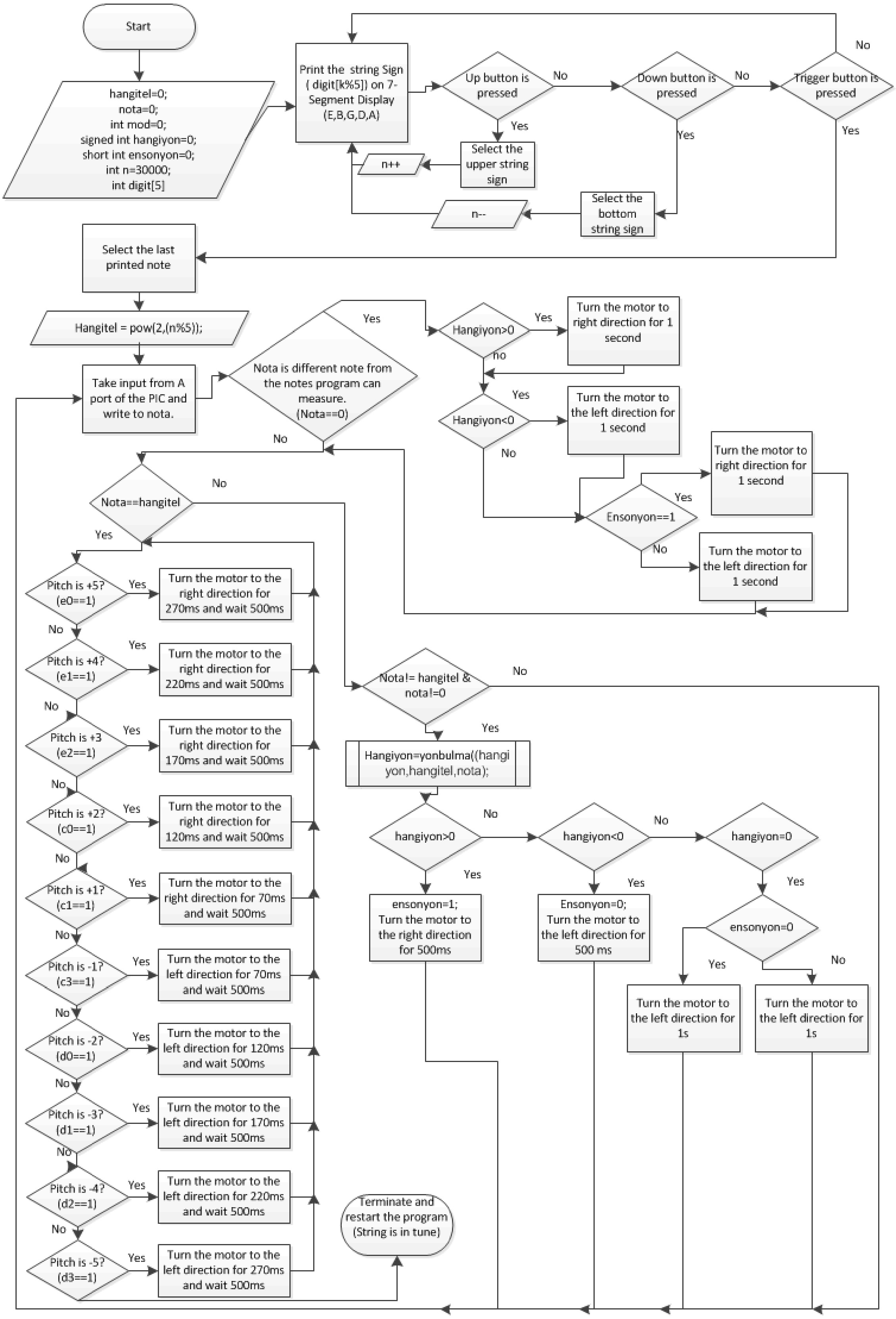


Figure 4.1: Direction finding method

Tuning range is determined as 1 octave. So if the note which we want to tune is in wrong octave, tuning job will be done successfully but in same octave with the wrong note at the first condition. It only works if the detuned note is in the true octave.

There are 7 basic notes in an octave, but in this project, 5 of them are used. Because there is no need to use all of them. A guitar has 6 strings and these strings have 5 seperate note because of the fact that first and last string has the same note (E).

Other strings have respectively B,G,D,A notes. It is not essentially needed to use C and F notes, because we can handle tuning work with using these 5 notes and an advanced algorithm.



Start
hangitel=0;
nota=0;
int mod=0;
signed int hangiyon=0;
short int ensonyon=0;
int n=30000;
int digit[5]
Print the string Sign (digit[k%5]) on 7-Segment Display (E,B,G,D,A)
Up button is pressed
No
Yes
Down button is pressed
No
Yes
Trigger button is pressed
No
Yes
Select the upper string sign
n++
Select the bottom string sign
n--
Select the last printed note
Hangitel = pow(2,(n%5));
Take input from A port of the PIC and write to nota.
Nota is different note from the notes program can measure. (Nota==0)
Yes
No
Hangiyon>0
Yes
no
Turn the motor to right direction for 1 second
Hangiyon<0
Yes
No
Turn the motor to the left direction for 1 second
Ensonyon==1
Yes
No
Turn the motor to right direction for 1 second
Turn the motor to the left direction for 1 second
Nota==hangitel
No
Yes
Pitch is +5? (e0==1)
Yes
No
Turn the motor to the right direction for 270ms and wait 500ms
Pitch is +4? (e1==1)
Yes
No
Turn the motor to the right direction for 220ms and wait 500ms
Pitch is +3 (e2==1)
Yes
No
Turn the motor to the right direction for 170ms and wait 500ms
Pitch is +2? (c0==1)
Yes
No
Turn the motor to the right direction for 120ms and wait 500ms
Pitch is +1? (c1==1)
Yes
No
Turn the motor to the right direction for 70ms and wait 500ms
Pitch is -1? (c3==1)
Yes
No
Turn the motor to the left direction for 70ms and wait 500ms
Pitch is -2? (d0==1)
Yes
No
Turn the motor to the left direction for 120ms and wait 500ms
Pitch is -3? (d1==1)
Yes
No
Turn the motor to the left direction for 170ms and wait 500ms
Pitch is -4? (d2==1)
Yes
No
Turn the motor to the left direction for 220ms and wait 500ms
Pitch is -5? (d3==1)
Yes
Turn the motor to the left direction for 270ms and wait 500ms
Terminate and restart the program (String is in tune)
Nota!= hangitel & nota!=0
No
Yes
Hangiyon=yonbulma((hangiyon,hangitel,nota);
hangiyon>0
No
Yes
ensonyon=1; Turn the motor to the right direction for 500ms
hangiyon<0
No
Yes
Ensonyon=0; Turn the motor to the left direction for 500 ms
hangiyon=0
Yes
ensonyon=0
Yes
No
Turn the motor to the left direction for 1s
Turn the motor to the left direction for 1s

Figure 4.2: Main algorithm of the system

The algorithm consists of four functions:

1-Yonbulma()

2-saga_dondur()

3-sola_dondur()

4- main()

4.1 DIRECTION FINDING FUNCTION

Yonbulma() function takes 3 parameters as *yon*, *hangitel* and *temp_nota*. *yon* indicates the last output value of the **yonbulma()** function. *hangitel* has the value of selected string and *temp_nota* has the value of the note coming from guitar by plucking the selected string. Firstly, function must look *temp_nota* and *yon* variables. If *temp_nota* is 0 and *yon* is not 0, motor will be turned to same direction with the last turn. It means that the note coming from the guitar is between two of measurable notes. So, output will be **yon/(abs(yon));** to be repeated last turning operation. If *yon* is equal to 0, it means this is the first call of **yonbulma**() function. Then comparision is made to find direction to turn the motor.

According to main algorithm, while we are tuning a string of a guitar, the tuning part of this device firstly will take the analog sound and convert to digital data. If this data is E,B,G,D or A, one of the leds in the tuning box will light up and this signal will be transmitted to a port of the pic16F877A through phtodiodes.

As it is mentioned in 3rd Chapter, this data is transmitted as 6 bit. Each bit is assigned to a pin of A port.

A0= E (BINARY :000001)

A1=B (BINARY :000010)

A2=G (BINARY :000100)

A3=D (BINARY :001000)

A4=A (BINARY :010000)

A5=Free (BINARY :000000)

Thanks to capabilities of CCS-C program, The signal reaches to A port is set to *nota* variable and the connection between pic and box is used as a data bus. After this job, the value of *nota* must be known, because direction finding method is different for each note.

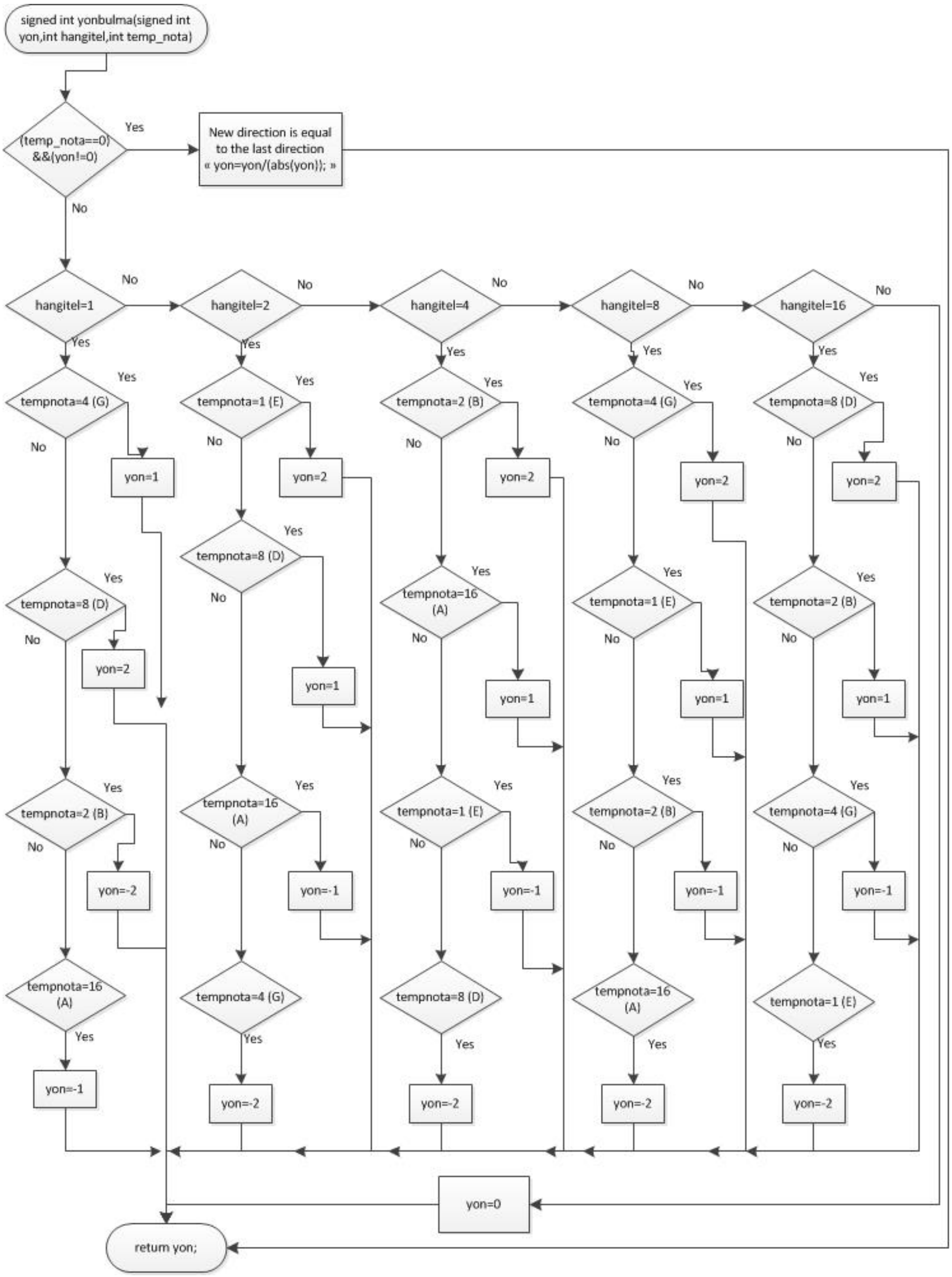


Figure 4.3: Algorithm of ***yonbulma***() function

If we suppose that our note sequence contains 5 note, *hangitel* is middle point of the sequence , right and left two notes are respectively lower and higher ones. As shown in the figure 4.1 , if our selected string is E, middle point will be E and value of it is zero. B is equal to 2, D is equal to 1, G is equal to -1 and A is equal to -2. “+” and “– ” values indicates respectively the right and left direction. For example, if the value of *note* which is taken by the yonbulma function is A, output will be -2 and motor will turn to the right direction for 1s with “**sola_dondur(10);**” command. If the value of *note* is G, output will be -1 and motor will turn to the left direction for 0.5 s with “**sola_dondur(15);**” command. If the note is B or D, output of the function will be 2 and 1 respectively.

If the selected string is A(*hangitel* = 16);

Note	E	G	A	B	D
Output	2	1	0	-1	-2

Table 4.1: *yon* values for A String

If the selected string is D (*hangitel* = 8);

Note	A	B	D	E	G
Output	2	1	0	-1	-2

Table 4.2: *yon* values for D String

If the selected string is G (*hangitel*=4);

Note	D	E	G	A	B
Output	2	1	0	-1	-2

Table 4.3: *yon* values for G String

If the selected string is B (*hangitel* =2);

Note	G	A	B	D	E
Output	2	1	0	-1	-2

Table 4.4: *yon* values for B String

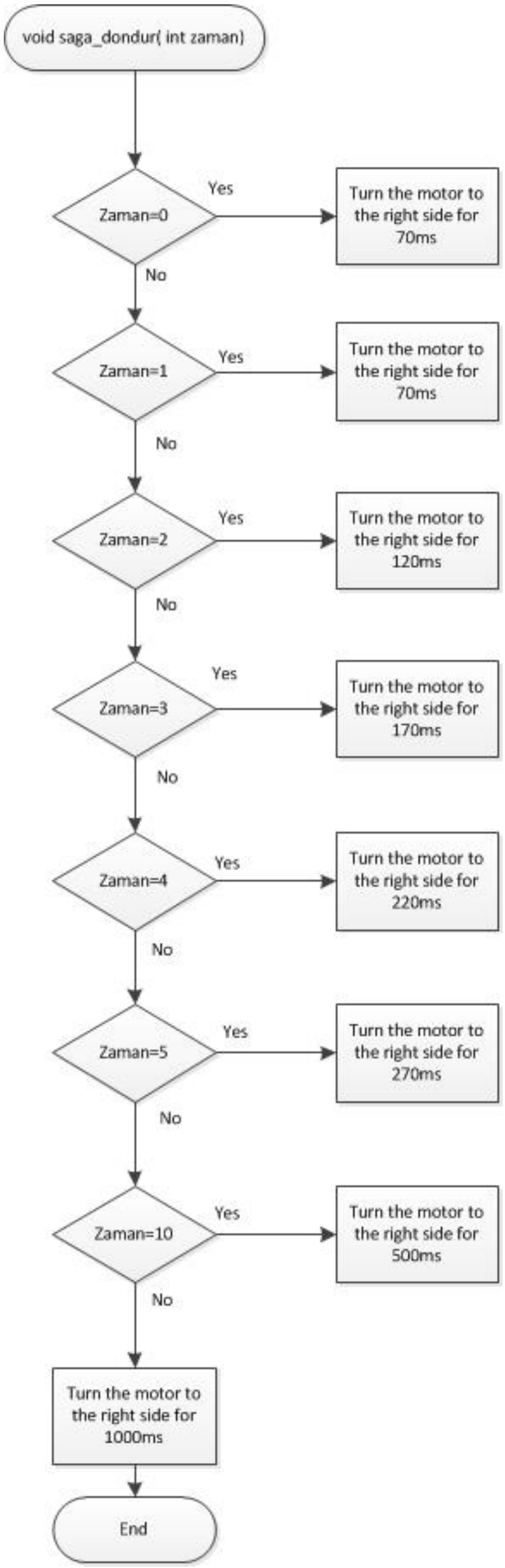
Then function returns *yon*.

4.2 saga_dondur FUNCTION

This function is required to turn the motor to the right direction. It only takes 1 parameter as *zaman*. It is not needed to adjust velocity by this function because L298 Motor driver provides turning the motor with fixed velocity.

Because of the capabilities of CCS C compiler, we can not use variables before allocate. So *zaman* parameter must be defined in the function. Then 7 possible different time is defined. They are as shown in the figure 4.4.

Figure 4.4: Algorithm of **saga_dondur()** function


void saga_dondur(int zaman)
Zaman=0
Yes
Turn the motor to the right side for 70ms
No
Zaman=1
Yes
Turn the motor to the right side for 70ms
No
Zaman=2
Yes
Turn the motor to the right side for 120ms
No
Zaman=3
Yes
Turn the motor to the right side for 170ms
No
Zaman=4
Yes
Turn the motor to the right side for 220ms
No
Zaman=5
Yes
Turn the motor to the right side for 270ms
No
Zaman=10
Yes
Turn the motor to the right side for 500ms
No
Turn the motor to the right side for 1000ms
End

4.3 sola_dondur FUNCTION

This function is required to turn the motor to the left direction. It takes *zaman* parameter as **saga_dondur**() function and Works as **saga_dondur**() function to turn on the motor opposite direction. 7 possible different time is defined as shown in the figure 4.5.

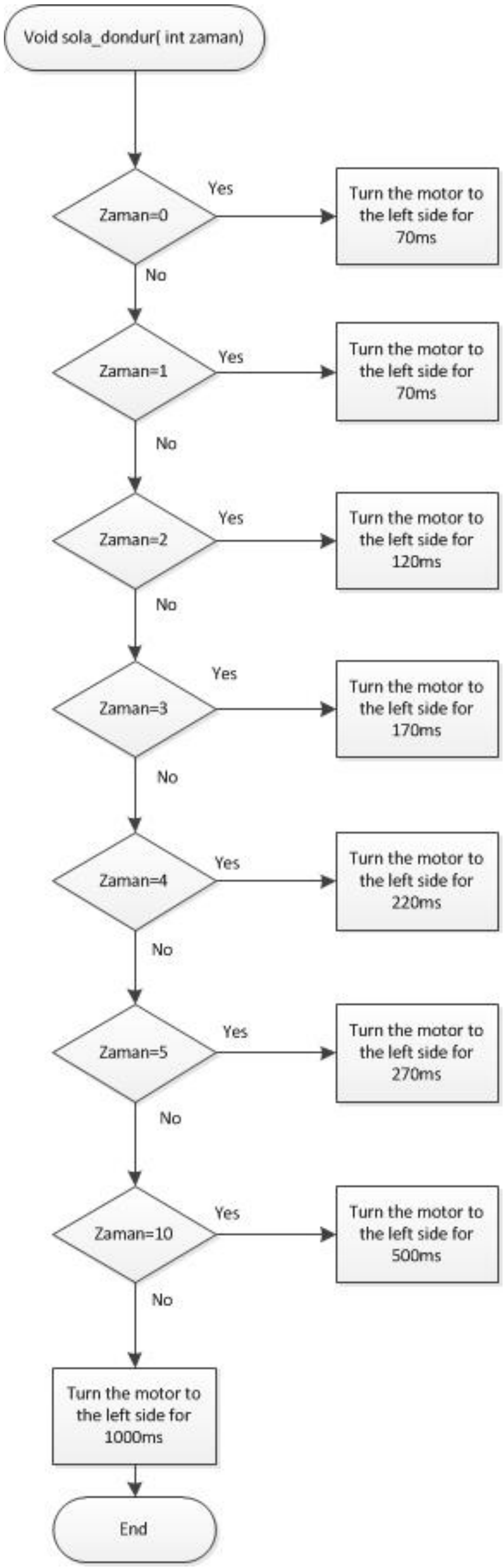

Void sola_dondur(int zaman)
Zaman=0
Yes
Turn the motor to the left side for 70ms
No
Zaman=1
Yes
Turn the motor to the left side for 70ms
No
Zaman=2
Yes
Turn the motor to the left side for 120ms
No
Zaman=3
Yes
Turn the motor to the left side for 170ms
No
Zaman=4
Yes
Turn the motor to the left side for 220ms
No
Zaman=5
Yes
Turn the motor to the left side for 270ms
No
Zaman=10
Yes
Turn the motor to the left side for 500ms
No
Turn the motor to the left side for 1000ms
End

Figure 4.5: Algorithm of **sola_dondur**() function

CHAPTER IV

TEST METHODS AND RESULTS

5.1 ROUTINE TESTS

The device is routinely tested after any code changes and new hardware upgrades. Then new software or hardware changes were made according to the test results. Usburn Brenner 8 USB PIC Programmer is used to uplad hex code into the PIC. Although removing and plugging the PIC was exhausting work, any virtual debugging tools was not used. It was plugged to the breadboard and unplugged more than 100 times during construction phase.

5.2 FINAL TESTS

For the final tests, an accoustic guitar was used. The microphone of the Automatic tuner device was sticked to the guitar. All the strings randomly detuned before first test. The device is used for each string by plucking them one by one. Cateater Frequency Counter program is used to measure the frequency of the string before and after tests. First test gave us the results below:

Note E (Lowest)	Test 1	Test 2	Test 3
Before Tuning	75.00 Hz	100.00Hz	Low Freq.
After Tuning	82.28Hz	82.12 Hz	Fails
Frequency of Note	82.41Hz	82.41 Hz	82.41Hz
Error	0.13 Hz	0.29 Hz	-
Test Time	23 s	47 s	-

Table 5.1: Test results of E Note

Note A	Test 1	Test 2	Test 3
Before Tuning	82.23Hz	96.50Hz	100.00Hz
After Tuning	109.70Hz	109.43Hz	109.98Hz
Frequency of Note	110.00Hz	110.00Hz	110.00Hz
Error	0.30 Hz	0.57 Hz	0.02Hz
Test Time	25 s	44 s	17 s

Table 5.2: Test results of A Note

Note D	Test 1	Test 2	Test 3
Before Tuning	109.70Hz	123.53Hz	110.25Hz
After Tuning	146.83Hz	146.83Hz	146.83Hz
Frequency of Note	146.51Hz	146.03Hz	146.03Hz
Error	0.32Hz	0.80Hz	0.80Hz
Test Time	63 s	38 s	47 s

Table 5.3: Test results of D Note

Note G	Test 1	Test 2	Test 3
Before Tuning	146.53Hz	166.42Hz	221.61Hz
After Tuning	196.00Hz	196.00Hz	196.88Hz
Frequency of Note	196.00Hz	196.00Hz	196.00Hz
Error	0 Hz	0 Hz	0.88Hz
Test Time	68 s	50 s	48 s

Table 5.4: Test results of G Note

Note B	Test 1	Test 2	Test 3
Before Tuning	197.26Hz	198.65Hz	247.75Hz
After Tuning	249.15Hz	Fails	Fails
Frequency of Note	246.94Hz	246.94Hz	246.94Hz
Error	2.21Hz	-	-
Test Time	64 s	-	-

Table 5.5: Test results of B Note

Note E (Highest)	Test 1	Test 2	Test 3
Before Tuning	249.15Hz	294.00Hz	249.75Hz
After Tuning	329.10Hz	331.58Hz	329.10Hz
Frequency of Note	329.63Hz	329.63Hz	329.63Hz
Error	0.53Hz	1.95Hz	0.53Hz
Test Time	40 s	39 s	92 s

Table 5.6: Test results of high E Note

Average operation time is 47 s and error ratio is too low for each successful tuning task. According to the test results of the B String, led of B note or the hole of it possibly causes to problem.

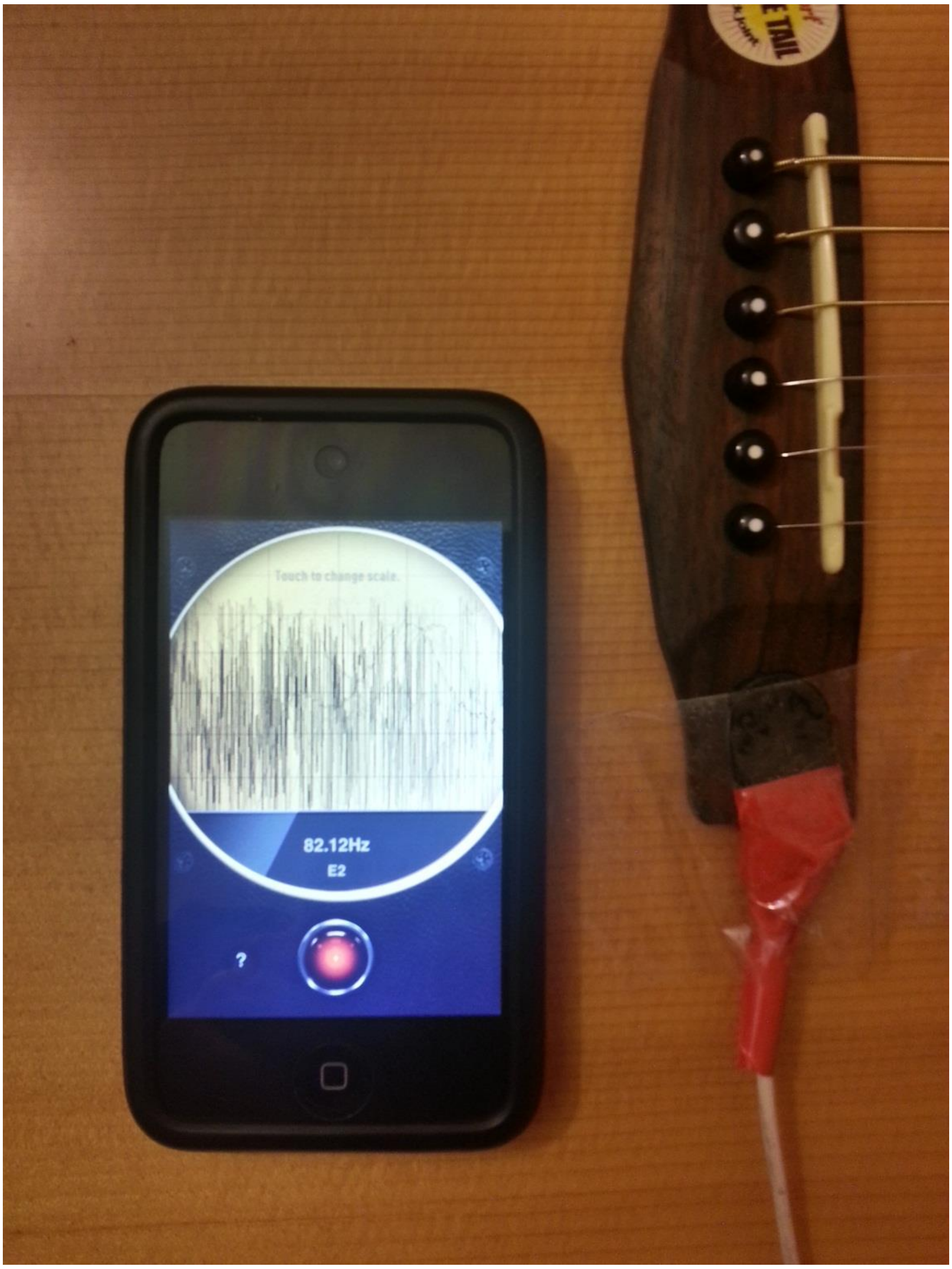


Figure 5.1: Frequency test with a program on IOS platform.

Test Results show that the system guarantees the precise tuning and errors are not considerable.

CONCLUSION

When I started this project, I was worried about whether the device will have enough stability and accuracy. Because there were not much project about automatic tuners and I couldn't find any resource about the capabilities of DC Motor on this subject. In some studies, It is stated that DC Motor in not useful mechanic part for automatic tuning device. But I wanted to try to maket his with DC Motor and wondered if it would be possible by an advanced programming. After some tests and improvements on the code, I believed that the device will work precisely.

The main disadvantage of the device is its third part frequency analyser. Because of my setup, microcontroller's analog inputs are not used. It is impossible to calculate the incoming frequency of the sound wave. So using the digital signals of analyser device is the one of the most important challenges. Therefore I passed much beyond the basic algorithm and had to make tests after every code alteration.

According to the results of my last tests, the device works beyond my predictions. The margin of errors are too narrow to be detected by human ear. In conclusion, though all difficulties in software and hardware, Automatic Guitar Tuner device works precisely and able to satisfy the consumer.

FUTURE WORKS

Even this device works properly, frequency analysing part shouldn't be used as a third part device connected to the microcontroller. Frequency of incoming sound wave can be received by the analog input of microcontroller and handled with programming and tuning box will be dismounted. Thus, different voltage problem will be fixed and also total cost will decrease. The first and most important upgrade will be about this issue. After this improvement, it can be formed in different versions to tune other instruments.

Another feature will be manual winding. While changing guitar strings, it is needed to wind the peg for 1-2 minute to same direction by hand. But using this new feature, it will take only 5-10 seconds. After these upgrades, the device will be designed again and formed to more aesthetic and ergonomic shape.

REFERENCES

[1] The American Heritage Dictionary of the English Language, Fourth Edition, Houghton Mifflin Company, 2000, archived from the original on June 25, 2008, [retrieved 20-05-2010]

[2] http://www.wisegeek.com/what-is-a-sound-wave.htm [10-12-2012]

[3] Professor Colin H Hansen, *Fundamentals of Acoustics*,24, Department of Mechanical Engineering University of Adelaide South Australia, Retrieved from http://www.who.int/occupational_health/publications/noise1.pdf [17-12-2012]

[4] http://en.wikipedia.org/wiki/C_(musical_note) [Last modified date: 30-01-2013 at 15:54]

[5] Harry F. Olson, *Music Physics and, Engineering*(New York Dover: 1952 reissued in 1967), retrieved from http://en.wikipedia.org/wiki/Sound [22-11-2012]

[6] (Strang,1994) Retrieved from http://en.wikipedia.org/wiki/Fast_Fourier_transform#cite_note-1 [17-11-2012]

[7] http://www.realmusicmaker.com/melody-what-you-hum/the-musical-staff/ [12-12-2012]

[8] http://www.actiontuners.com/ [03-07-2012]

[9] http://www.jigsaw.com/id703841/ccsinfo.com_companywiki.xhtml [10-11-2012]

[10] Sencer Yeralan, P.E., Ph.D. Helen Emery, *Programming and Interfacing the 8051 Microcontroller in C and Assembly,1,*Rigel Press, a Division of Rigel Corporation, Retrieved from http://www.rigelcorp.com/8051/P&ICH1.pdf [10-11-2012]

[11] 1977 Data Catalog", Micro Electronics from General Instrument Corporation. Retrieved from http://en.wikipedia.org/wiki/PIC_microcontroller#cite_ref-1977catalog_5-0[27-11-2011]

[12] Mustafa Kemal Güngör (12-2003), *A Programmable Control Unit for Industrıal Applıcatıons*,2, The Department of Electrıcal and Electronıcs Engıneerıng. Retrieved from http://etd.lib.metu.edu.tr/upload/772764/index.pdf [10-09-2011]

[13] PIC16F87XA Data Sheet 28/40/44-Pin Enhanced Flash Microcontrollers, 5, Retrieved from http://ww1.microchip.com/downloads/en/devicedoc/39582b.pdf

[14] http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010242 [10-11-2012]

[15] BPW34 Datasheet,1. Retrieved from http://www.vishay.com/docs/81521/bpw34.pdf [10-11-2012]

[16] "LED". *The American heritage science dictionary*, Houghton Mifflin Company. 2005. Retrieved from http://dictionary.reference.com/browse/led and http://www.thefreedictionary.com/LED, [22-06-2011]

[17] TLHx640 Datasheet,1. Retrieved from http://www.vishay.com/docs/83219/tlhg640.pdf [17-11-2012]

[18] http://notesonafilm.com/2012/05/24/led-vs-tungsten-not-such-a-hard-choice-afterall/ [09-01-2013]

[19] http://en.wikipedia.org/wiki/Light-emitting_diode [22-01-2013]

[20] J. S. Petersen, Niki Foster, *What is a DC Motor?* http://www.wisegeek.com/what-is-a-dc-motor.htm [Last Modified Date: 18-11-2012]

[21] http://en.wikipedia.org/wiki/DC_motor [21-11-2012]

[22] http://en.wikipedia.org/wiki/Seven-segment_display [21-11-2012]

[23] *74VHC164 8-Bit SIPO Shift Register Data Sheet*, 1,2. Retrieved from http://www.fairchildsemi.com/ds/74/74VHC164.pdf [17-11-2012]

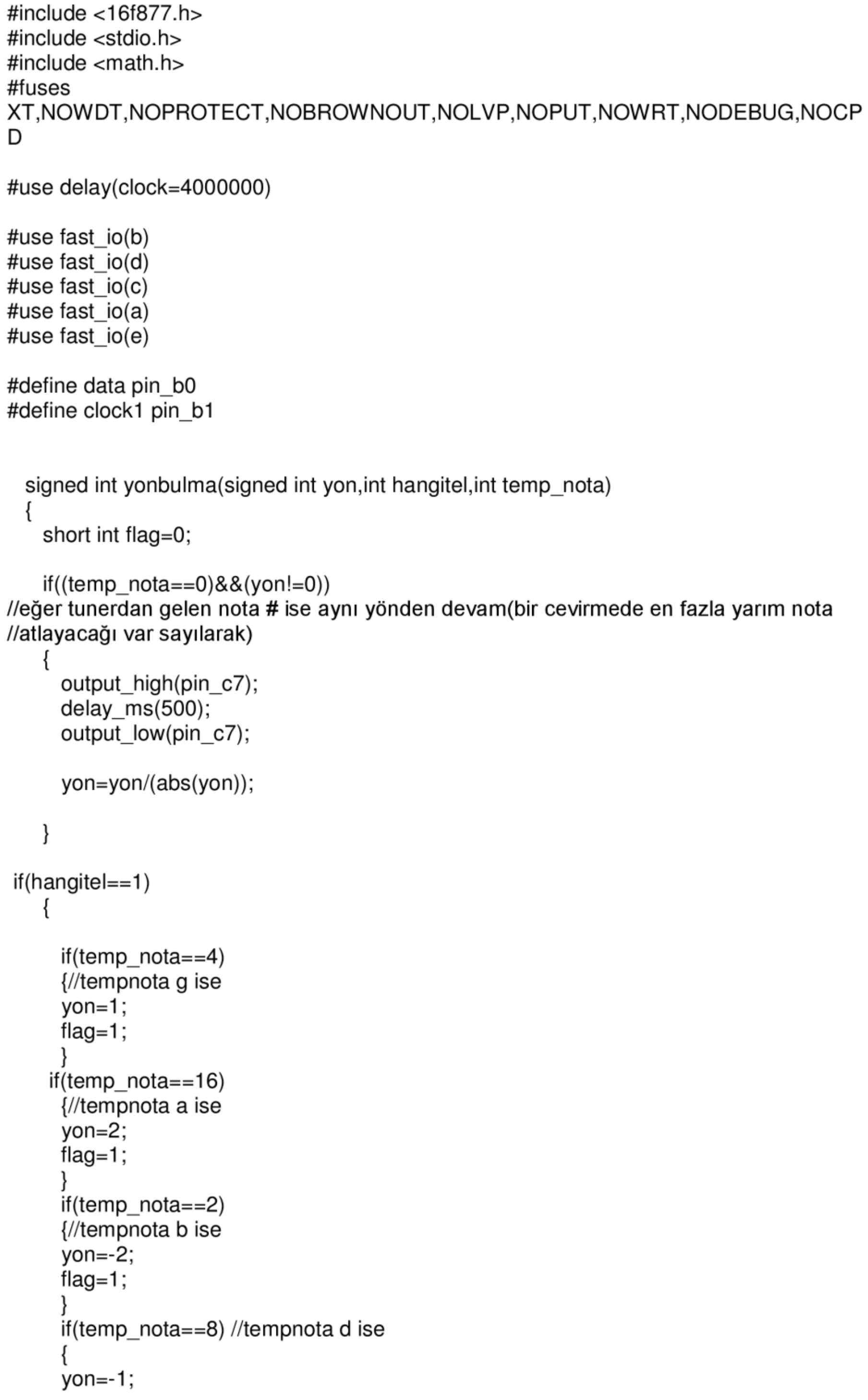
[24] Autonomous Ground Vehicle. Retrieved fromhttp://essaybank.degree-essays.com/engineering/autonomous-ground-vehicle.php#ixzz2IhEUJaXK [13-02-2013]

[25] http://www.ccsinfo.com/product_info.php?products_id=PCWHD_full [17-11-2012]

[26] http://www.microchipc.com/reviews/CCS_C/#compiler [27-01-2013]

APPENDICES

APPENDIX A: PROGRAM CODES



```
#include <16f877.h>
#include <stdio.h>
#include <math.h>
#fuses
XT,NOWDT,NOPROTECT,NOBROWNOUT,NOLVP,NOPUT,NOWRT,NODEBUG,NOCP
D

#use delay(clock=4000000)

#use fast_io(b)
#use fast_io(d)
#use fast_io(c)
#use fast_io(a)
#use fast_io(e)

#define data pin_b0
#define clock1 pin_b1


  signed int yonbulma(signed int yon,int hangitel,int temp_nota)
  {
    short int flag=0;

    if((temp_nota==0)&&(yon!=0))
//eğer tunerdan gelen nota # ise aynı yönden devam(bir cevirmede en fazla yarım nota
//atlayacağı var sayılarak)
    {
      output_high(pin_c7);
      delay_ms(500);
      output_low(pin_c7);

      yon=yon/(abs(yon));

    }

 if(hangitel==1)
    {

      if(temp_nota==4)
      {//tempnota g ise
      yon=1;
      flag=1;
      }
     if(temp_nota==16)
      {//tempnota a ise
      yon=2;
      flag=1;
      }
      if(temp_nota==2)
      {//tempnota b ise
      yon=-2;
      flag=1;
      }
      if(temp_nota==8) //tempnota d ise
      {
      yon=-1;
      flag=1;
      }

  }
   if(hangitel==2)
    {
       if(temp_nota==1)
       {//tempnota e ise
      yon=2;
      flag=1;
       }
  if(temp_nota==8)
  {//tempnota d ise
  yon=1;
  flag=1;
  }
   if(temp_nota==16) //tempnota a ise
   {
      yon=-1;
      flag=1;
   }
  if(temp_nota==4)
    {//tempnota g ise
    yon=-2;
    flag=1;
    }
  }
if(hangitel==4)
    {
     if(temp_nota==2)
     {//tempnota b ise
      yon=2;
      flag=1;
     }
  if(temp_nota==16)
  {//tempnota a ise
  yon=1;
  flag=1;
  }
      if(temp_nota==1)
      {//tempnota e ise
      yon=-1;
      flag=1;
      }
  if(temp_nota==8) //tempnota d ise
  {
  yon=-2;
  flag=1;
  }
  }
    if(hangitel==8)
    {
     if(temp_nota==4) //tempnota g ise
     {
      yon=2;
      flag=1;
     }
  if(temp_nota==1)
  flag=1;
  }
      if(temp_nota==2)
      {//tempnota b ise
      yon=-1;
      flag=1;
      }
  if(temp_nota==16)
  {//tempnota a ise
  yon=-2;
  flag=1;
  }
  }
    if(hangitel==16)
    {
      if(temp_nota==8) //tempnota d ise
    {
      yon=2;
      flag=1;
      }
  if(temp_nota==2)
  {//tempnota b ise
  yon=1;
  flag=1;
  }
      if(temp_nota==4) //tempnota g ise
      {
        yon=-1;
        flag=1;
      }
  if(temp_nota==1)
  {//tempnota e ise
  yon=-2;
  flag=1;
  }
  }

    return yon;
  }

void saga_dondur(int zaman)
{

  if(zaman==0)
  {
    output_high(pin_b4);
    output_low(pin_b5);
    delay_ms(70);
    output_low(pin_b4);
  }

  if(zaman==1)
  {
    output_high(pin_b4);
    output_low(pin_b5);
    delay_ms(70);
    output_low(pin_b4);
  }
```

```
    output_high(pin_b4);
    output_low(pin_b5);
    delay_ms(120);
    output_low(pin_b4);
  }
  if(zaman==3)
  {
    output_high(pin_b4);
    output_low(pin_b5);
    delay_ms(170);
    output_low(pin_b4);
  }
  if(zaman==4)
  {
    output_high(pin_b4);
    output_low(pin_b5);
    delay_ms(210);
    output_low(pin_b4);
  }
  if(zaman==5)
  {
    output_high(pin_b4);
    output_low(pin_b5);
    delay_ms(250);
    output_low(pin_b4);
  }

  if((zaman!=0)&&(zaman!=1)&&(zaman!=2)&&(zaman!=3)&&(zaman!=4)&&(zaman!=5))
  {
    output_high(pin_b4);
    output_low(pin_b5);
    delay_ms(450);
    output_low(pin_b4);
  }
  delay_ms(700);
}

void sola_dondur(int zaman)
{
    delay_ms(100);
   output_high(pin_b6);
   output_low(pin_b7);

  if(zaman==0)
  {
    output_high(pin_b5);
    output_low(pin_b4);
    delay_ms(70);
    output_low(pin_b5);
  }
  if(zaman==1)
  {
    output_high(pin_b5);
    output_low(pin_b4);
    delay_ms(70);
    output_low(pin_b5);
  }
  if(zaman==2)
```

```
    output_low(pin_b4);
    delay_ms(120);
    output_low(pin_b5);
  }
  if(zaman==3)
  {
    output_high(pin_b5);
    output_low(pin_b4);
    delay_ms(170);
    output_low(pin_b5);
  }
  if(zaman==4)
  {
    output_high(pin_b5);
    output_low(pin_b4);
    delay_ms(210);
    output_low(pin_b5);
  }
  if(zaman==5)
  {
    output_high(pin_b5);
    output_low(pin_b4);
    delay_ms(250);
    output_low(pin_b5);
  }

  if((zaman!=0)&&(zaman!=1)&&(zaman!=2)&&(zaman!=3)&&(zaman!=4)&&(zaman!=5))
  {
    output_high(pin_b5);
    output_low(pin_b4);
    delay_ms(450);
    output_low(pin_b5);
  }
  delay_ms(700);
}

int k=30000, j=0;

int digit[5]={0xde,0x3e,0xf6,0x7a,0xfa};

int
yondegeri[19]={0xfc,0x60,0xda,0xf2,0x66,0xb6,0xbe,0xe0,0xfe,0xf6,0xfa,0x3e,0x1a,0x7a
,0xde,0x8e,0x93,0xf1,0x9d};
int
eksiyondegeri[16]={0xfd,0x61,0xdb,0xf3,0x67,0xb7,0xbf,0xe1,0xff,0xf7,0xfb,0x3f,0x1b,0x
7b,0xdf,0x8f};

int sifir=0x07;
char dongukes=0;

int sayacn=0;


/********************** ANA PROGRAM **********************/

void main ()
{
  int nota=0x00;
```

```
 //int temp_nota=0x00;
 int tel=0;
 int mod=0;
 int sayac=0;
 int notabas=0;
 short int ensonyon=1;
 int pitchsayaci=0;
 short int ifkontrol=0;

    setup_psp(PSP_DISABLED);
    setup_spi(SPI_SS_DISABLED);
    setup_timer_1(T1_DISABLED);
    setup_timer_2(T2_DISABLED,0,1);
    setup_adc_ports(NO_ANALOGS);
    setup_adc(ADC_OFF);
    setup_CCP1(CCP_OFF);
    setup_CCP2(CCP_OFF);

 set_tris_a(0xff);              //A0,A1 giriş
 set_tris_b(0b00001100);  //B2,B3 hariç çıkış
 set_tris_e(0b111);
 set_tris_d(0b10001111);
 output_b(0b00000000);
 output_a(0b00000000);
 set_tris_c(0b00001111);
 output_c(0b00000000);
 output_d(0b00000000);
 output_e(0b000);

while(1)
{
dongukes=0;

 mod=k%5;

      for(j=0;j<8;j++)
      {
         output_bit(data,bit_test(digit[mod],j));
         output_high(clock1);
         output_low(clock1);

      }
      delay_ms(50);
      output_high(pin_c6);

 while(1)
 {

    if(input(pin_b2))
    {
      k=k+1;
      mod=k%5;
         for(j=0;j<8;j++)
         {
            output_bit(data,bit_test(digit[mod],j));
            output_high(clock1);
            output_low(clock1);
      output_high(pin_d2);

  }

  if(input(pin_b3))
  {
    k=k-1;
    mod=k%5;

      for(j=0;j<8;j++)
      {
        output_bit(data,bit_test(digit[mod],j));
        output_high(clock1);
        output_low(clock1);

      }
      delay_ms(250);
      output_low(pin_d2);
  }

 if(input(pin_d7))
 break;
}

//nota hex olarak seciliyor
hangitel=pow(2,mod);
output_low(pin_c6);
sayacn=hangitel;
notabas=0;

//HANGİTEL TEST
//saglama
while(sayacn!=1)
{
  sayacn=sayacn/2;
  notabas=notabas+1;
}
for(j=0;j<8;j++)
    {
      output_bit(data,bit_test(digit[notabas],j));
      output_high(clock1);
      output_low(clock1);

    }
delay_ms(500);



 //motor



output_a(0x00);
output_e(0x00);
output_b(0x00);
output_c(0x00);
output_d(0x00);
output_e(0x00);
```

```
while(1)
{

//a portundan gelen nota datasını okuma kısmı

nota=input_a();
delay_ms(20);
sayac=0;


//NOTA TEST
sayac=nota;
notabas=0;


//saglamasi
while(sayac>1)
{
  sayac=sayac/2;
  notabas=notabas+1;
}

if(nota==0)
{
  for(j=0;j<8;j++)
      {
        output_bit(data,bit_test(yondegeri[0],j));
        output_high(clock1);
        output_low(clock1);

      }
      delay_ms(250);

}
else
{
for(j=0;j<8;j++)
      {
        output_bit(data,bit_test(digit[notabas],j));
        output_high(clock1);
        output_low(clock1);

      }
      delay_ms(250);
}


if(nota==0x00)//eğer tunerdan gelen nota diyez ise
{

  if(hangiyon==0) //bu kısım düşünülcek
  {
    if(ensonyon==0)
      sola_dondur(10);
    else
      saga_dondur(10);
```

```
        output_bit(data,bit_test(yondegeri[17],j));
        output_high(clock1);
        output_low(clock1);

      }
      delay_ms(200);
  }
  if(hangiyon>0)
  {

    saga_dondur(10);
    for(j=0;j<8;j++)
      {
        output_bit(data,bit_test(yondegeri[18],j));
        output_high(clock1);
        output_low(clock1);

      }
      delay_ms(200);
  }
  if(hangiyon<0)
  {

   sola_dondur(10);
   for(j=0;j<8;j++)
      {
        output_bit(data,bit_test(yondegeri[19],j));
        output_high(clock1);
        output_low(clock1);

      }
      delay_ms(200);
  }
  delay_ms(500);
}

//gitardan gelen frekansın notası, seven segmentte seçilen nota ile aynıysa

if(nota==hangitel)
{

 while(1)
 {
 ifkontrol=0;
  if(input(pin_e0))
  {
    sola_dondur(5);
    delay_ms(500);
    ifkontrol=1;
  }
  if(input(pin_e1))
  {
    sola_dondur(4);
    delay_ms(500);
    ifkontrol=1;
  }
  if(input(pin_e2))
  {
    ifkontrol=1;
  }
  if(input(pin_c0))
  {
    sola_dondur(2);
    delay_ms(500);
    ifkontrol=1;
  }
  if(input(pin_c1))
  {
    sola_dondur(1);
    delay_ms(500);
    ifkontrol=1;
  }

  if(input(pin_c3))
  {
    saga_dondur(1);
    delay_ms(500);
    ifkontrol=1;
  }
  if(input(pin_d0))
  {
    saga_dondur(2);
    delay_ms(500);
    ifkontrol=1;
  }
  if(input(pin_d1))
  {
    saga_dondur(3);
    delay_ms(500);
    ifkontrol=1;
  }
  if(input(pin_d2))
  {
    saga_dondur(4);
    delay_ms(500);
    ifkontrol=1;
  }
  if(input(pin_d3))
  {
    saga_dondur(5);
    delay_ms(500);
    ifkontrol=1;
  }

  if(ifkontrol==0)
  pitchsayaci=pitchsayaci+1;
  delay_ms(100);

  if(pitchsayaci==7)
  reset_cpu();
  }
}
```

```
    hangiyon=yonbulma(hangiyon,hangitel,nota);
    delay_ms(100);
    if(hangiyon>0)
    {
      ensonyon=1;
      for(j=0;j<8;j++)
        {
          output_bit(data,bit_test(yondegeri[abs(hangiyon)],j));
          output_high(clock1);
          output_low(clock1);

        }
        delay_ms(300);

      saga_dondur(10);

    }
    if(hangiyon<0)
    {
      ensonyon=0;
      for(j=0;j<8;j++)
        {
          output_bit(data,bit_test(eksiyondegeri[abs(hangiyon)],j));
          output_high(clock1);
          output_low(clock1);

        }
        delay_ms(300);

      sola_dondur(10);

    }

    if(hangiyon==0)
    {
     for(j=0;j<8;j++)
        {
          output_bit(data,bit_test(sifir,j));
          output_high(clock1);
          output_low(clock1);

        }
        delay_ms(300);

      if(ensonyon==0)
        sola_dondur(10);
      else
        saga_dondur(10);
    }
  }


  }


 }
}
```

BIOGRAPHY

PERSONAL INFORMATION

Surname, Name	**: MUTLU, Sait**
Nationality	**: T.C**
Birthdate and birthplace	**: 31.12.1984 - MUĞLA**
Mobile phone	**: 0539 5560749**
E-Mail	**: samuray84@gmail.com**

EDUCATION

DEGREE	SCHOOL	YEAR OF GRADUATION
University	Çankaya University	2008
High School	Batıkent Nermin Mehmet Çekiç Anadolu Lisesi	2003

JOB EXPERIENCES

YEAR	PLACE	ENROLLMENT
2009-2011	T.C. Başbakanlık Sosyal Yardımlaşma ve Dayanışma Gen. Müd.	Computer Engineer
2011-…	DHMİ Esenboğa Havalimanı	Programmer

LANGUAGES

English – Advanced
Russian - Beginner