Group 3 project report – Doorbell

EEE2036 Lab Design and Professional Studies

J. Tyler

L. H. Hoang

U. Hassan

V. Wilke

University of Surrey

Faculty of Engineering and Physical Sciences

Project Specification

A doorbell is to be made by a team of 3-4 students as part of the laboratories module for the 2013-2014 session. The doorbell should be able to produce distinct and discernible short musical pieces using 3 AA batteries, i.e. 4.5V, using only the components available in the electronics laboratories. It should at least be able to produce 1 audible tone to a minimum distance of 1 meter from the push-button. The entire hardware should be fabricated on a PCB and should be enclosed in a proper box provided in the labs. Provided that the basics are met correctly, enhancements can be added on to the amplifier like LCD and activation/de-activation remotely.

Task Planning

The planning for the project was not individually done but everyone had a part in whole process. My plan for the 1st session was to finish the soldering on the PCB that was given to us and check that it's working properly. In 2nd session my aim was to have understanding of the programme on which our doorbell was designed to work and to help in drilling the box. By the 3rd session we had our own PCB for the project on which we could do our enhancements. So, my plan was to involve in soldering process and finish it by the end of session. In 4th session we had our remote design with LCD and a keyboard concept. We had to drill 2 boxes so I planned to work on boxes if I finish with the soldering process. In 5 and 6 the plan was to compile all the things together and sort out the errors if there are any.

Task Planning – Circuit Layouts

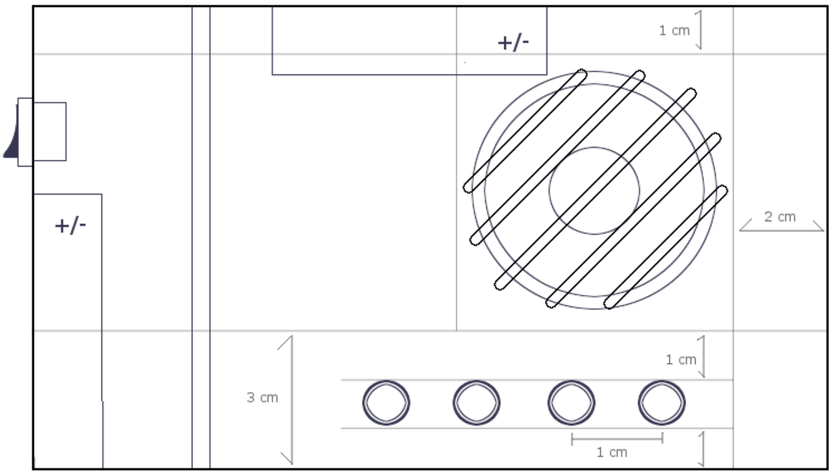
From the beginning of the project, all the components and the doorbell's circuit were given. So the first thing needed to be done during the first section of the project labs was to determine the position of all components and then solder them into the given circuit PCB, allowing some initial tests for the LEDs on the PIC controller programming. After the circuit was tested and worked perfectly, we realized that some addition features and innovations that we intended to do will not be able to add into the given PCB. Therefore, the best way is to build a whole new PCB based on the original one with addition components in it. By doing this, we can add an ON/OFF switch, move all the tune select buttons to the front side of the doorbell's box, making it easier to utilize for users. Moreover, we can also add programming buttons to help users to design their own favourite melodies besides three original tones. The last thing needed to be noted is the use of the remote control with which people can easily control the doorbell from the distance of three to four meters. After deciding all the addition features, the whole new doorbell and a remote control circuit were built and test initially on the breadboard carefully to make sure they work with specifications before I make circuit schematics and PCB layouts on Cadsoft EAGLE PCB. That process required a lot of times as I had to do them step by step carefully. After all, when the PCB layouts were done, I also did the soldering for the doorbell and remote control's PCBs before attaching the final designs to the given boxes.

Task planning – Case and user interface

The PCB circuit and PIC chip required a proper and professional way to allow use. Leaving the product as-is on the prototyping circuit would have looked poor and unprofessional, as well as make the product much too susceptible to damage from use. Furthermore, there would be several security concerns for a final product.

The final case and user interface system needed to provide the user with a simple way of using all the features, whilst keeping the inner workings of both the remote and the main circuit safe from most expected external sources of damage. A solid plastic case was provided as part of the project for the main product, but a smaller box for the remote was needed for our group's further developments on the project. The downside of the smaller remote box imposed limitations on the final size of the remote control's PCB, as well as other components used in the remote control device (such as the LCD screen and the 6V power source needed to power the final device),as well as the wires between components and the PCB. The large case for the main component had enough space for this to not be an issue, but thought had to be put into the placement of components (Such as the speaker, power and volume control, and the large amount of buttons) to allow for ease of use as well as making basic maintenance (such as replacing batteries) simple.

Initial design and mechanical sketch



The initial design uses the box provided, dimensions being 20 x 11 x 6 cm. These dimensions easily fit the testing circuit, speaker, 9v battery holder, and wiring for the buttons. Two locations are proposed for the battery, one being at the top near the speaker and the other one being at the side. The first location has the benefit of not being in the way of the speaker, but the downside to that option is that it takes up space that might be needed for further developments.

A keyboard for programming custom tunes had been proposed by this point, and the right side of the box was the first option for placing the additional buttons. For that reason, a margin was left between the right side and any other components. At this stage however, the setup of the keyboard was undecided and thus couldn't be planned out in full.

The speaker is 6 cm wide and can be attached to the front, reminiscent of a radio. The diagonal holes were proposed to provide a more unique look to the final product and make it more appealing and distinct.

Task Planning – Programming

To plan the programming, I used a three step process:

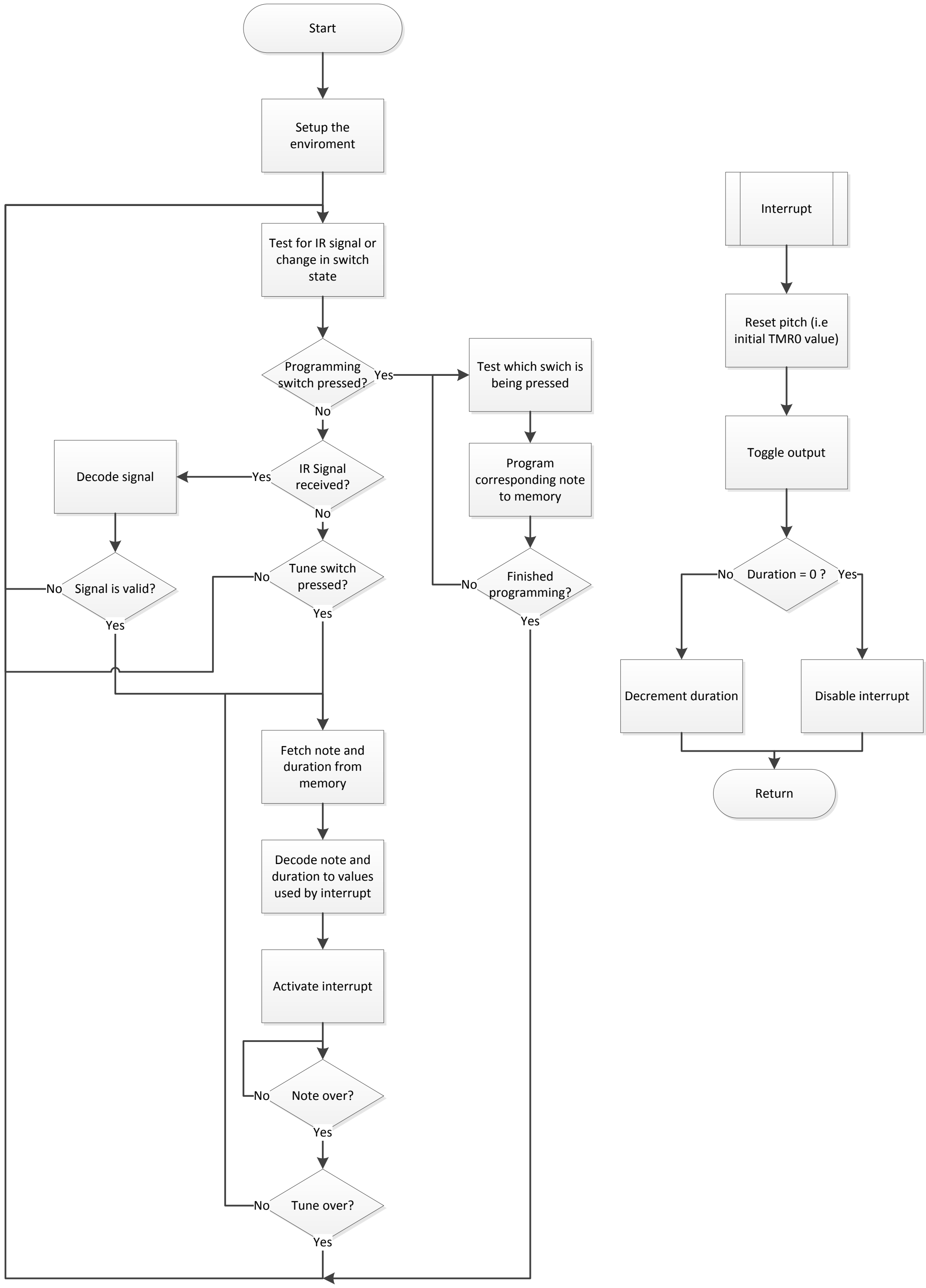
1. Set out what the code has to do
2. Break this down into individual modules
3. Translate each of these modules to blocks of assembly code.

This process allowed me to create modular source code which can be adapted me meet changing requirements.

To exemplify this approach, here is the list of requirements for the doorbell receiver code:

1. Play 4 different melodies when 4 different buttons are pressed.
2. Receive and decodes an IR signal from the remote and plays the appropriate tune.
3. Allow a custom tune to be programmed into EEPROM memory.

Requirements two and three were added part way through the project development, so the code had to be adapted, this was easy due to the modular nature. The block diagram for the receiver is shown below. Naturally, this is a simplified block diagram and many of the processes correspond to several blocks of code. However each block does represent a module which can be tested independently and reused as required.

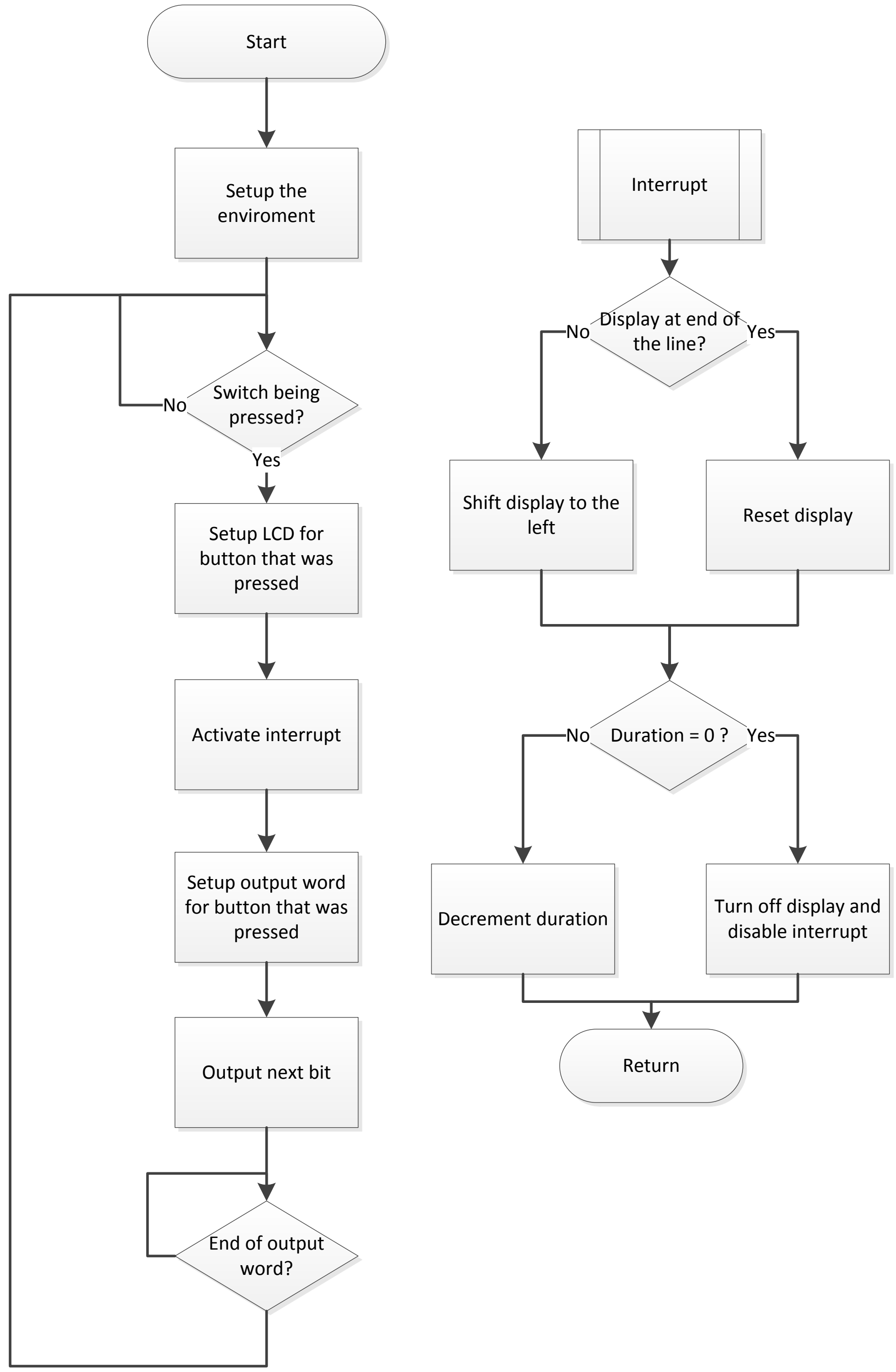


In a similar manner, these were the requirements for the remote code:

1. Send an appropriate IR signal to activate the doorbell receiver on four different button presses
2. Interface with an LCD to display tune data

Requirement two was added part way through the code development. However, as with the receiver, the modular nature of the code made it easy to adapt.

Below is the simplified flowchart for the receiver.



The other aspect of the code which needed to be planned is the mathematics behind how to play a note of a set pitch and duration from the doorbell receiver. To do this I created an Excel sheet which use information from the microcontroller datasheet, and the details of the code, to generate initial TMR0 values for the interrupt, and special values to be put into a duration register.

I won't go into detail here about how exactly this works, as it is far better explained in the code itself and the datasheet. The spreadsheets below, however, give the values needed to be put into set registers in the microcontroller in order to allow the output note to be of the desired pitch and of a duration.

		Desired note (Octave of middle C)	C	C♯/D♭	D	E♭/D♯	E	F	F♯/G♭	G	A♭/G♯	A	B♭/A♯	B
		Desired pitch (Hz)	261.6	277.2	293.7	311.1	329.6	349.2	370.0	392.0	415.3	440.0	466.2	493.9
		Desired Time period (s)	3.82E-03	3.61E-03	3.41E-03	3.21E-03	3.03E-03	2.86E-03	2.70E-03	2.55E-03	2.41E-03	2.27E-03	2.15E-03	2.02E-03
		Desired Half Time period (s)	1.91E-03	1.80E-03	1.70E-03	1.61E-03	1.52E-03	1.43E-03	1.35E-03	1.28E-03	1.20E-03	1.14E-03	1.07E-03	1.01E-03
	TMR0 Scaler	Mean of [the modulus of each percentage error]												
TMR0 Start value	1		-1655	-1548	-1447	-1351	-1261	-1176	-1095	-1020	-948	-880	-817	-756
Percentage error from real value	1	0.021	-0.01	0.01	0.02	0.00	0.01	0.02	-0.03	0.04	0.01	-0.03	0.04	-0.04
TMR0 Start value	2		-700	-646	-595	-548	-502	-460	-420	-382	-346	-312	-280	-250
Percentage error from real value	2	0.036	0.05	0.01	-0.04	0.06	-0.06	0.02	0.05	0.04	0.01	-0.03	-0.05	-0.04
TMR0 Start value	4		-222	-195	-170	-146	-123	-102	-82	-63	-45	-28	-12	3
Percentage error from real value	4	0.040	0.05	0.01	0.08	0.06	-0.06	0.02	0.05	0.04	0.01	-0.03	-0.05	-0.04
TMR0 Start value	8		17	31	43	55	66	77	87	97	106	114	122	129
Percentage error from real value	8	0.143	0.05	-0.21	0.08	0.06	0.21	0.02	0.05	-0.28	-0.33	-0.03	-0.05	0.36
TMR0 Start value	16		137	143	150	156	161	167	172	176	181	185	189	193
Percentage error from real value	16	0.327	-0.37	0.23	-0.39	-0.44	0.21	-0.54	-0.55	0.35	-0.33	-0.03	-0.05	-0.43
TMR0 Start value	32		196	200	203	206	209	211	214	216	218	220	222	224
Percentage error from real value	32	0.770	0.46	-0.66	-0.39	-0.44	-0.85	0.58	-0.55	0.35	1.00	1.38	1.44	1.15
TMR0 Start value	64		226	228	229	231	232	234	235	236	237	238	239	240
Percentage error from real value	64	0.986	0.46	-0.66	1.49	-0.44	1.26	-1.66	-0.55	0.35	1.00	1.38	1.44	1.15
TMR0 Start value	128		241	242	243	243	244	245	245	246	247	247	248	248
Percentage error from real value	128	2.147	0.46	-0.66	-2.27	3.54	1.26	-1.66	4.19	0.35	-4.31	1.38	-4.53	1.15
TMR0 Start value	256		249	249	249	250	250	250	251	251	251	252	252	252
Percentage error from real value	256	4.385	-6.23	-0.66	5.25	-4.42	1.26	7.28	-5.28	0.35	6.32	-9.89	-4.53	1.15

Constants:		This workbook shows the initial TMR0 start values for all the notes in the octave of middle C, for different values of the TMR0 scaler.
1 Instruction execution time (s)	1.00E-06	This works by calculating the number of increments of TMR0, at different scaler values, necessary to create a delay equal to a half period of a given note, then subtracting this from 256 (the overflow value of the TMR0 register)
		Negative values are impossible to implement, so are highlighted in red.
		The TMR0 scaler with smallest percentage errors (determined by the mean of (the modulus of each percentage error)) was chosen for the octave of middle C
		Therefore we use a TMR0 scaler of 8 and the corresponding initial TMR0 values for the octave of middle C
		Using these values initial TMR0 values for middle C with a TMR0 scaler of 8, sequential octaves can then be found by doubling (or halving) the TMR0 scaler
		E.g a scaler of 1:16 will give an octave below middle C, a scaler of 1:4 will give an octave above middle C

Duration value for a whole note (semibreve) at 120BPM, for the octave of middle C

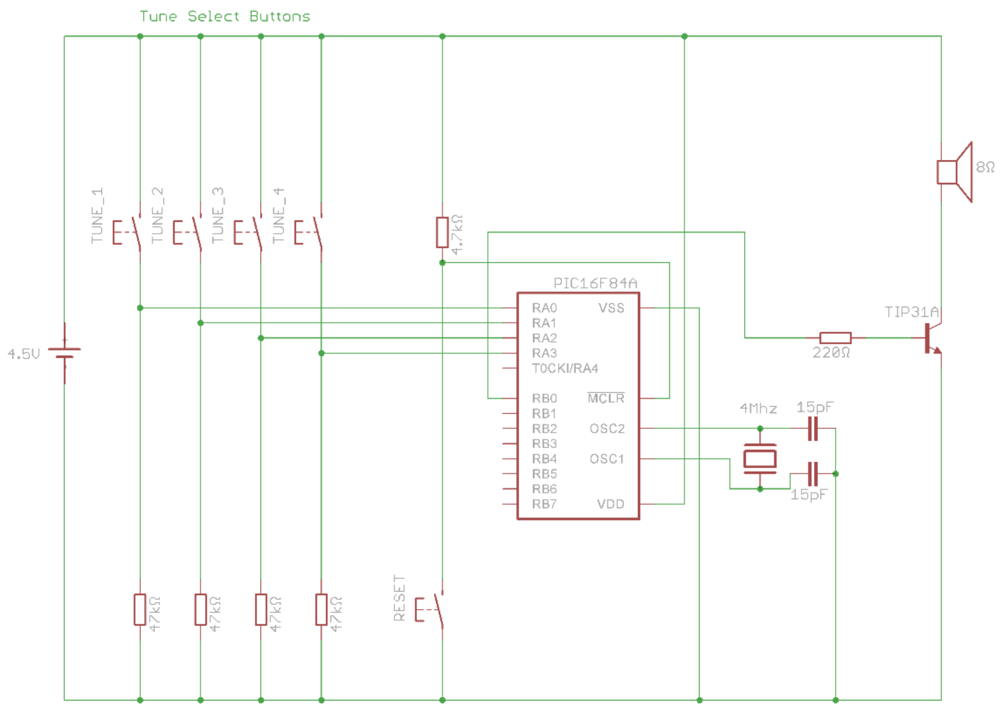
Note	Time period	No. of half cycles in one second	No. of half cycles in a whole note	Duration Value = [No. of half cycles in a whole note]/ 8
C	3.82E-03	5.23E+02	1.05E+03	131
C♯/D♭	3.61E-03	5.54E+02	1.11E+03	139
D	3.41E-03	5.87E+02	1.17E+03	147
E♭/D♯	3.21E-03	6.22E+02	1.24E+03	156
E	3.03E-03	6.59E+02	1.32E+03	165
F	2.86E-03	6.98E+02	1.40E+03	175
F♯/G♭	2.70E-03	7.40E+02	1.48E+03	185
G	2.55E-03	7.84E+02	1.57E+03	196
A♭/G♯	2.41E-03	8.31E+02	1.66E+03	208
A	2.27E-03	8.80E+02	1.76E+03	220
B♭/A♯	2.15E-03	9.32E+02	1.86E+03	233
B	2.02E-03	9.88E+02	1.98E+03	247

Constants:	
Duation of a whole note (semibreve) at 120BPM (s)	2
Dividing factor (referred to as "duration multiplier"	8

The duration value is equal to [the number of times the interrupt is called (i.e. the number of half cycles) in the duration of a semibreve] divided by [a constant]
This constant is referred to as the "duration multiplier"
Therefore if the interrupt stops after [duration value]*[duration multiplier] toggles of the output, the note will be a semibreve long
The duration multiplier has therefore been set as 8 (see constants table to the left), as this makes duration values of less than (2^8 -1 = 255), so will fit in one 8 bit register
The duration multiplier can therefore be halved and doubled in the same manner as the TMR0 scaler (see "Pitches" sheet) to allow different notes in different octaves to be the same length

Initial Schematic and circuit description

This is the initial draft of the circuit we used for our doorbell.



It is quite a simple circuit and is loosely split into input, process and output stages as you look from the left hand side of the circuit across to the right.

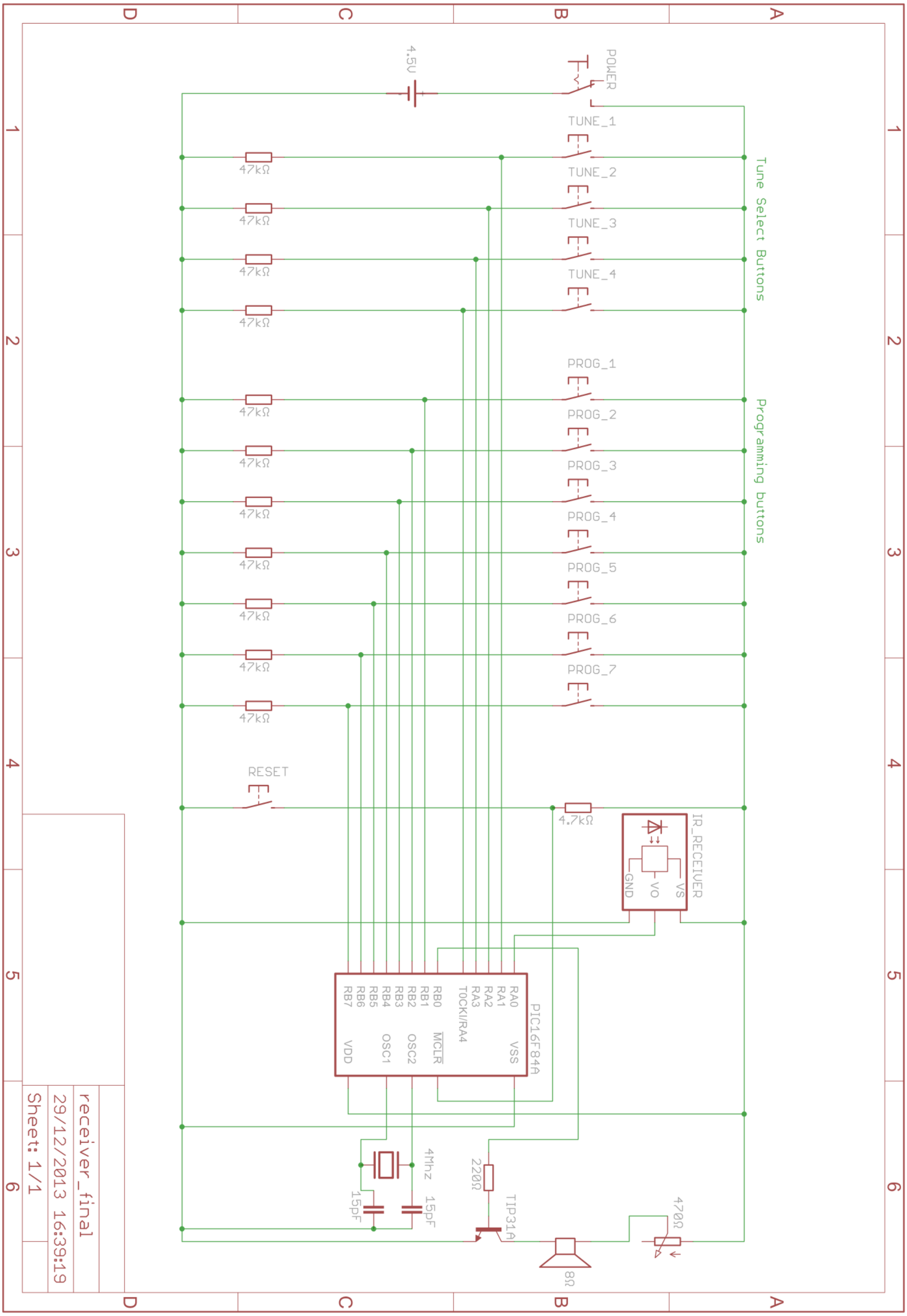
On the far left, we have the batteries which supply the power to the circuit. We chose to use three 1.5V AA batteries (3x1.5=4.5V) from an early stage, because the batteries and holders are cheap and readily available.

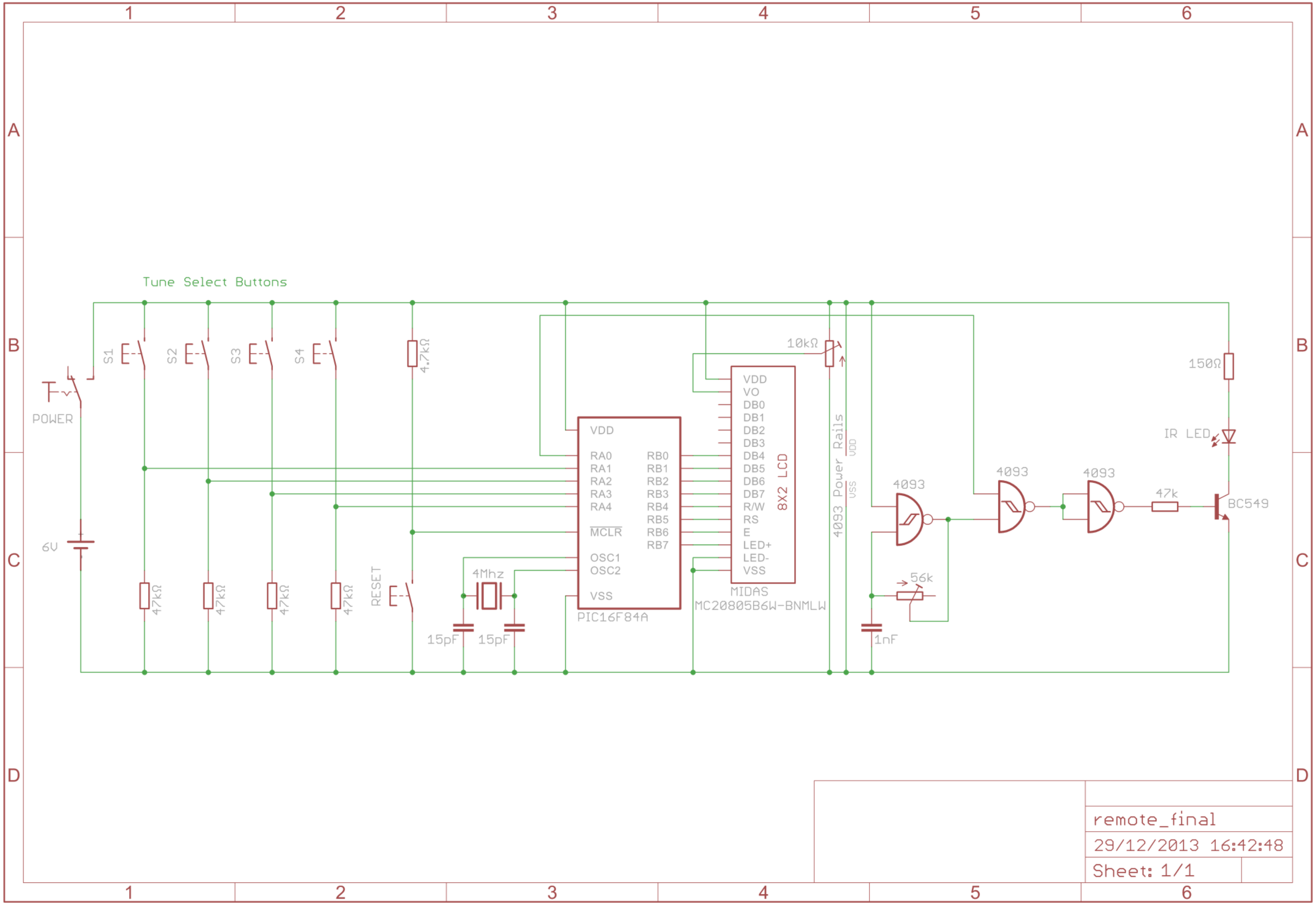
The next item on the left hand side of the diagram is the array of switches; we planned for four momentary switches with pull down resistors to act as the tune select switches. These switches connect to port A on the PIC. Additionally there is a reset switch, with a pull up resistor to account for the fact that the MCLR pin on the PIC is an active low pin.

In the centre of the diagram is the PIC itself, this is where all the processing happens according to the assembly code which we wrote in MPLAB. If you look just to the lower left hand side of the PIC on the diagram you can see the 4MHz crystal oscillator (with coupling capacitors to ground). This oscillator sets the timing of the PIC and corresponds to an instruction cycle time of 1μs.

On the right hand side of the diagram we have the output stage. This consists of the speaker which is turned on and off by the high-current NPN transistor. The transistor acts as a switch controlled by the microcontroller. When the PIC sets Port B, pin 0 to 5V, current flows through the base to the emitter of the transistor (via the 220Ω current limiting resistor). This in turn allows current to flow from the collector to the emitter, turning the speaker on. However, when the microcontroller output is 0V, no current can flow from the base to the emitter (because there is no potential difference) and so no current is allowed to flow from the collector to the emitter, thus turning the speaker off.

Final schematic diagrams




Tune Select Buttons
POWER
6V
S1
S2
S3
S4
47kΩ
47kΩ
47kΩ
47kΩ
4.7kΩ
RESET
4Mhz
15pF
15pF
VDD
RA0
RA1
RA2
RA3
RA4
MCLR
OSC1
OSC2
VSS
RB0
RB1
RB2
RB3
RB4
RB5
RB6
RB7
PIC16F84A
10kΩ
VDD
VO
DB0
DB1
DB2
DB3
DB4
DB5
DB6
DB7
R/W
RS
E
LED+
LED-
VSS
8X2 LCD
MIDAS
MC20805B6W-BNMLW
4093 Power Rails
VSS
VDD
4093
4093
4093
56k
1nF
47k
150Ω
IR LED
BC549
remote_final
29/12/2013 16:42:48
Sheet: 1/1

The receiver circuit hasn't changed much from the draft. However the changes that have been made, and their references on the diagram's grid system, are:

- A power switch has been added (B1)
- Seven more momentary switches with pull down resistors have been added, to act as the programming keyboard (B2:C4)
- An infra-red receiver module has been added. This module looks for a 36 kHz modulated infrared pulse train, demodulates the carrier, and outputs the data signal (B4)
- A volume control potentiometer has been added to allow the volume of the output to be adjusted (B6)
- Some of the inputs and outputs of the PIC have been moved to different pins for convenience (B5:C5)

These modifications, allow the circuit to meet the full specification.

The transmitter circuit has also been designed following the decision to add the remote control. This circuit is similar to the receiver circuit in terms of how the power rails, tune select buttons, and PIC are set up, with the exception of the fact that the receiver is powered by two CR2032 batteries to save space. However, as might be expected, the output stage is different.

The receiver features an 8x2 character LCD screen which uses a 4 bit parallel interface and connects to the PIC using port B (B4:C4). This interface features 4 data bits (RB0:RB3), one read/write select pin (RB4), one bit to select which register of the LCD we are accessing (RB5), one enable bit (RB6), and one bit to activate/deactivate the backlight (RB7). The display also has a 10kΩ potentiometer for contrast adjustment (B4).

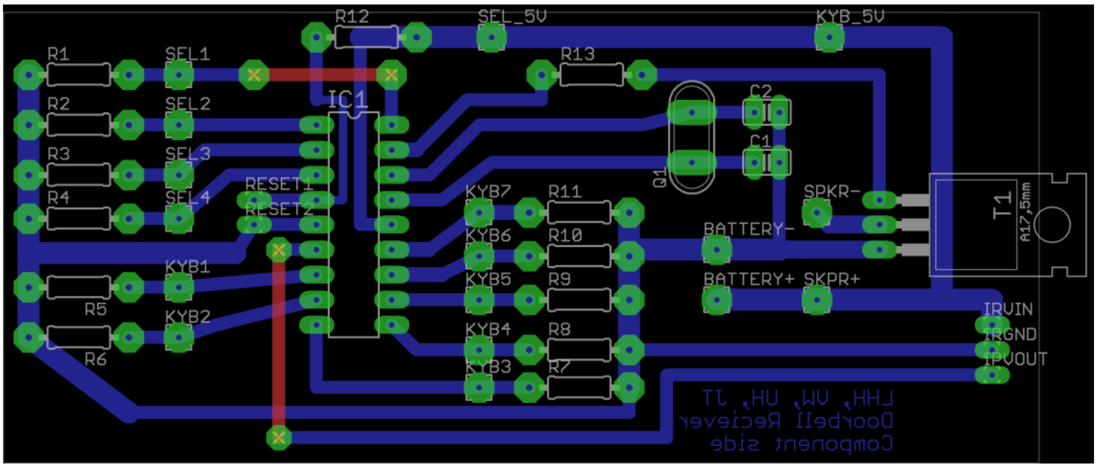
In addition to the LCD, we need to have the infra-red LED output. This consists of a pulse train from the PIC which is modulated onto a 36 kHz carrier. This is achieved by having a free running relaxation oscillator at 36 kHz (the Schmitt NAND gate in C5). The oscillator output is fed into a second NAND gate whose other input is the output from the PIC. The output of this NAND gate is therefore 0V when both inputs are HIGH, and 5V when either is LOW. It therefore acts as a modulator. This output is the opposite of what we want (i.e. we want a LOW output when the PIC output is LOW), we therefore feed this through the final NAND gate (in C6) to invert the output.

This logic level output is used (via a current limiting resistor) to provide a small current to the base of an NPN signal transistor. This acts as a switch in exactly the same way as the larger TIP31A transistor does with the receiver circuit.

PCB Layouts

This is the PCB layout for the doorbell receiver. It is a single sided design, blue lines indicate copper traces on the bottom of the board, and red lines indicate jumper wires.

The board was designed to be the width of the box meaning that it could slot directly into the moulded grooves of the box, therefore negating the need for mounting posts. Additionally the IR receiver position has been chosen so that drilling a single hole is all that needs to be done in order to make the IR light visible to the receiver.

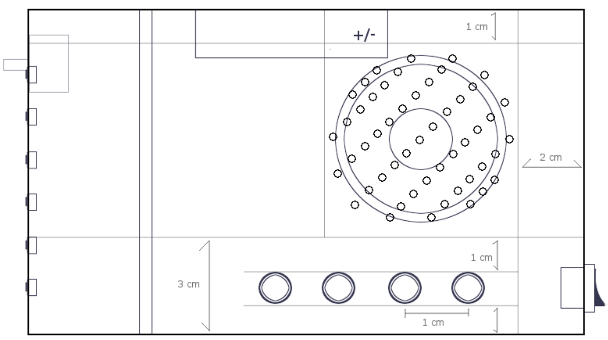


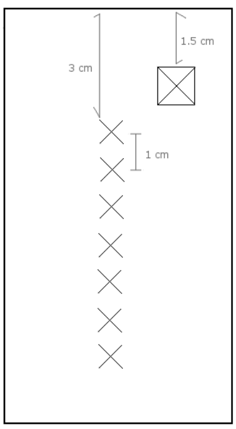
This is the PCB layout for the doorbell remote. It is a single sided design, blue lines indicate copper traces on the bottom of the board, and red lines indicate jumper wires.

This board uses all of the available space in order to fit the components into the small space afforded by the box. Care has been taken to ensure that tracks which carry a non-negligible current are as thick as possible. The buttons have been positioned so that they are in the same position on the PCB as the front of the box. Therefore trailing wires inside the box going to the buttons aren’t strictly necessary.

Mechanical sketches

Final design – main case



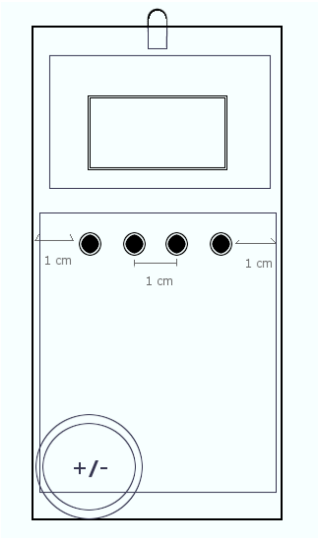


The final design largely retains the general layout if the initial design, with most changes made to accommodate further functionality. The development process mean that a seven-button keyboard input and a method of receiving an infrared signal sent from the remote control device.

The final design called for a keyboard made up of seven buttons to be accessible, which were placed on the left side of the box to remain somewhat out of the way of the four buttons at the front. The initial proposed location on the right-hand side would have made the internal wiring unnecessarily cluttered, so the left side of the box was used instead. Ultimately, this addition meant that the power switch and battery case had to be placed elsewhere.

Further changes include a small hole that had to be drilled at the top (not shown on diagrams) to receive signals for the remote. A potentiometer was added to the left side of the case, connecting to the speaker to provide volume control. The speaker system was reduced to a series of drilled holes, since it was easier and less time-consuming than the initial design of diagonal slits. The final result still results in a clear sound from the speaker, where it sits secured on the inside of the casing.

Final design – remote box



The initial design was intended to accommodate the minimum specifications of the project, and thus did not include a remote control device. A design had to be made at a later point, and thus did not need redesigns from the original design.

The remote was designed around the functionality it had to serve, with the final size being limited to one that would be comfortable to operate single-handedly. An excessively bulky design could have housed the internal components easily, but would also have deviated from the original purpose of the remote control device.

The requirements of the device dictated that the remote had to have the same basic inputs, save the keyboard, as well as prompt an LCD screen to display the user's selection of tune. The LCD screen was placed on the remote due to it being considered the device the user would most likely be looking at, compared to the main device. However, this meant more complexity in regards to the PCB, as well as more wiring. The final assembly has few pieces, mainly due to space constraints imposed by the PCB and wiring. A small 6V battery source is enough to power the LCD and send an appropriate signal to the main device.

Final assembly drawing

I) 3D models of the doorbell receiver.

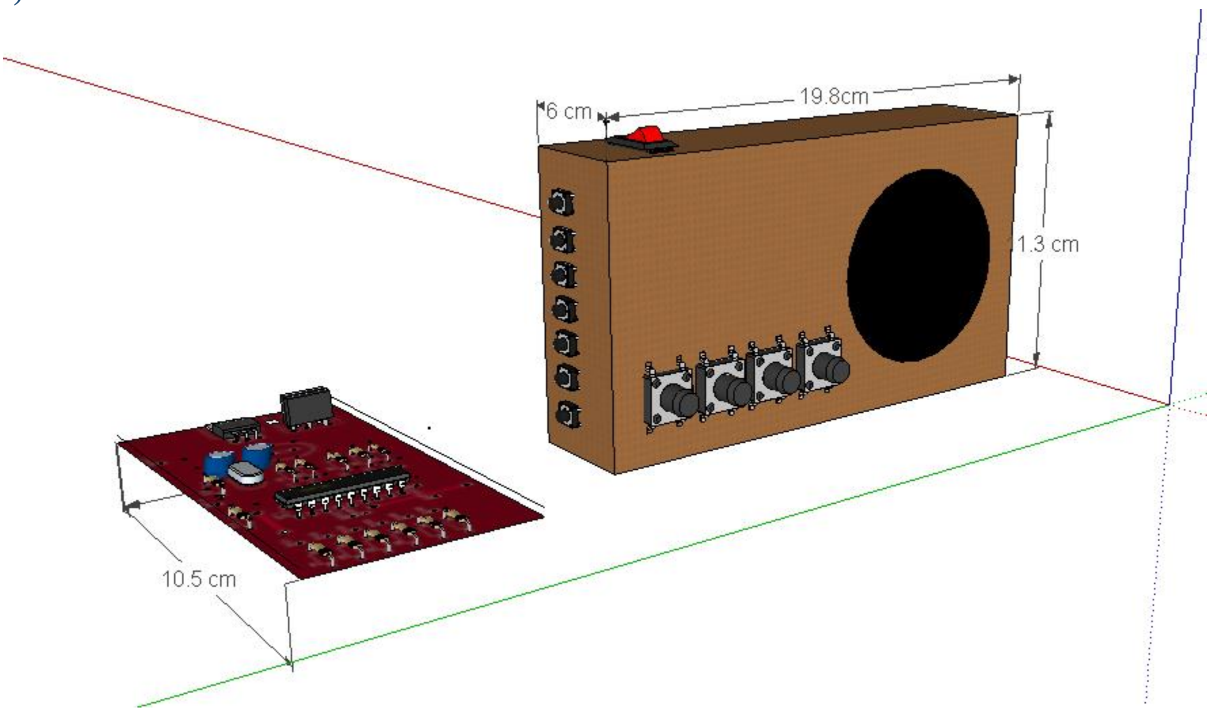


Fig 1.1: The front side 3D model of the doorbell.

Fig 1.1 is the front side model of the doorbell in which we have the speaker attached on the right hand side. On the bottom of the box are four control button placed in a straight line. Every time each of these buttons is pressed, a corresponding melody will be played. The dimensions of the box are also noted on the above model: 19.8 × 11.3 × 6 (cm).

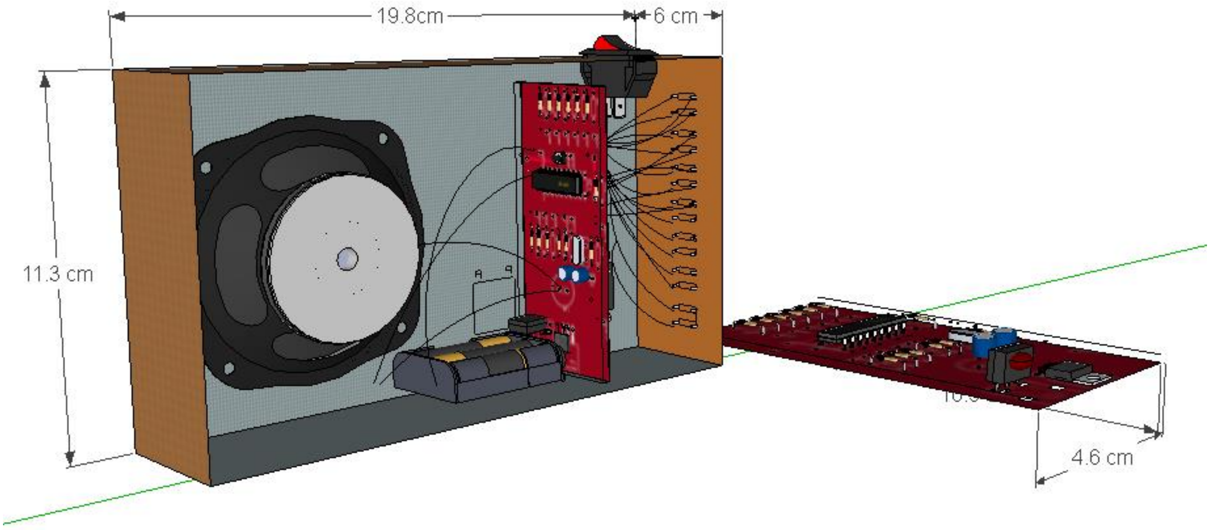


Fig 1.2: The 3D model inside the doorbell.

The figure 1.2 shows the inside model in which we can easily see how components are put in the box of the doorbell. The doorbell PBC is placed vertically inside the box, creating more space for another components such as the three AA battery holder and the speaker which is upside down attached to the box.

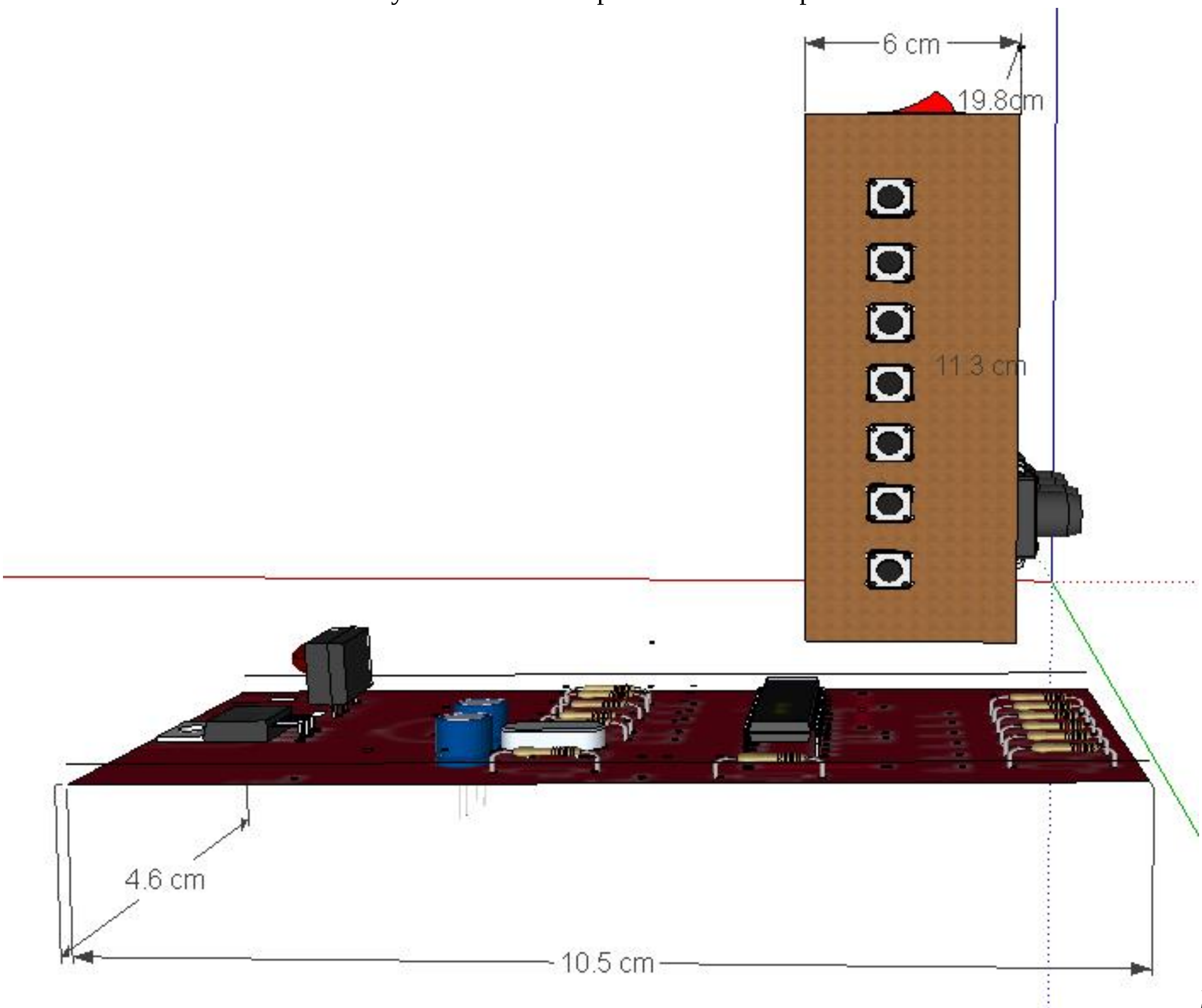


Fig 1.3: The 3D model of the side the doorbell.

Figure 1.3 shows the 3D view from the side of the box with the PCB taken out. On the side of the box, there are seven buttons placed in a horizontal line. These buttons will help user to design their own melodies besides three original tones. The PCB with components on is taken out of the box to measure the dimensions which is 10.5 × 4.6 (cm) so that the PCB can fit in the box.

II) 3D models of the remote control.

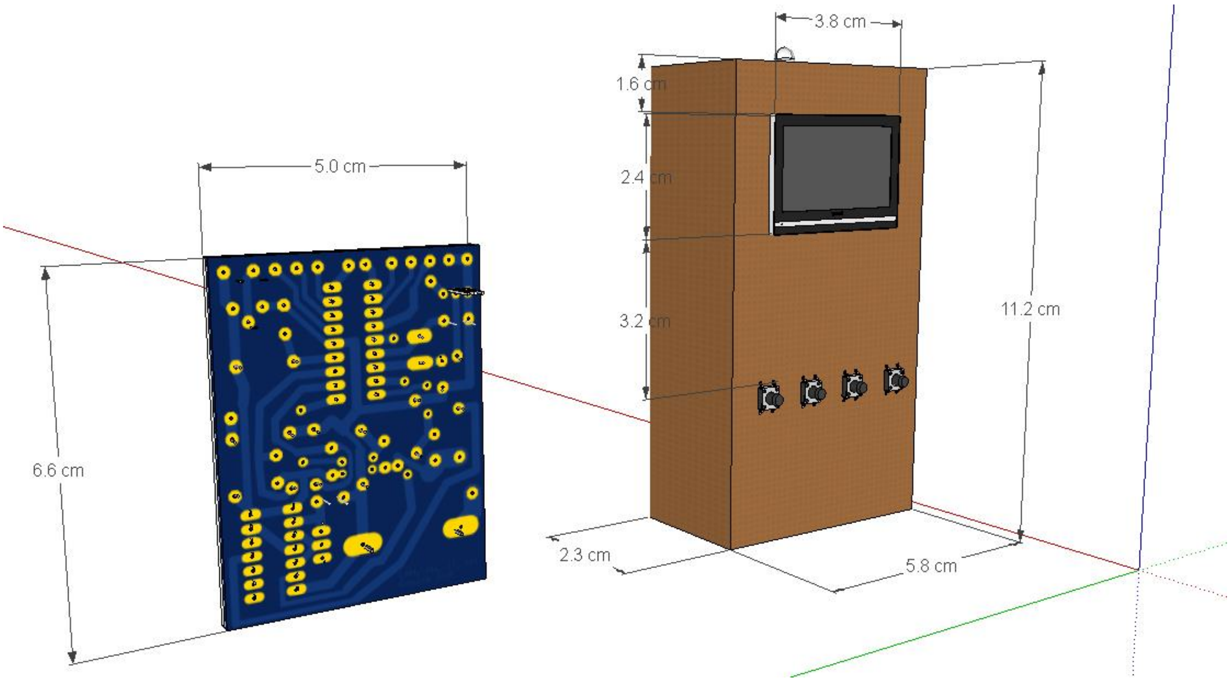


Fig 1.4: The 3D front side model of the remote control.

Figure 1.4 shows the 3D front side model of the remote control with all dimensions labelled. The remote control box has the dimensions of 11.2 × 5.8 × 2.3 (cm) with a LCD attached on the front of the box. This LCD is placed on the top half of the box and its dimensions are 3.8 × 2.4 (cm). There are four buttons placed horizontally 3.2 cm beneath the LCD screen.

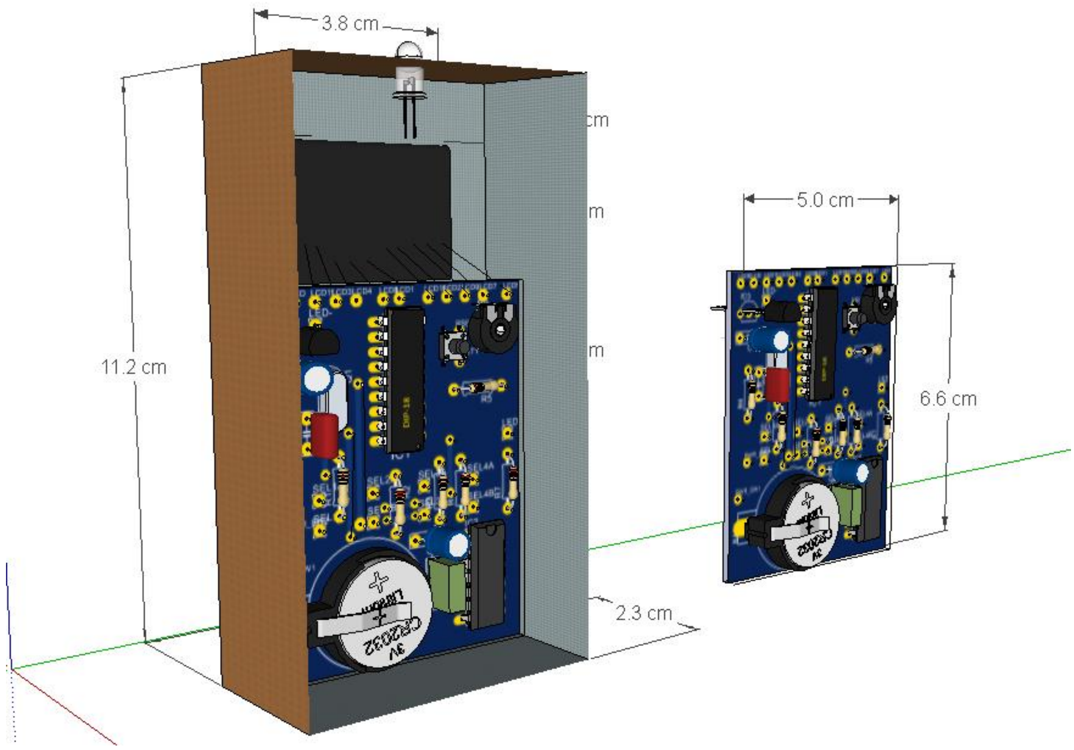


Fig 1.5: The 3D view to inside of the remote control.

Figure 1.5 shows the 3D view of the inside of the remote control box. It has a remote control PCB with the dimensions of 6.6 × 5 (cm) attached inside beneath the back of the LCD screen. On the top of the box, there is an IR LED which will transfer the IR signal to the doorbell receiver in order to play four different tunes.

Test at specification

For the minimum specifications:

The doorbell plays four different melodies when four different buttons are pressed. Those melodies are: "Happy birthday to you", "Imperial March", "The entire range of notes" and the last tune is the tune that user can programmed themselves. The sound quality of these tunes is clear and can be turned louder and down to mute with the presence of the volume controller.

The whole doorbell components including: doorbell PCB, three AA batteries, speaker and connecting wires are all fitly placed inside the given box of dimensions: 19.8 × 11.3 × 6 (cm).

For the innovations:

A whole new doorbell PCB was made in order to add some components such as IR receiver, tones design buttons as well as to move the four control buttons which was located on the given PCB to the front side of the box, making it easier to use for users.

The seven buttons, which are used for designing melodies is attached to the side of the box. This help people can design new doorbell melodies that they like besides three original tones. This make people to have more choices for the doorbell tunes.

The volume control is placed on the box so that users can set the appropriate sound volume depending to the distance between them and where the box is placed, This avoids the situation when the sound is too loud when user stay near the doorbell or too low when they are far apart from the box.

A remote control was made to send IR signal to activate the operation of the doorbell, helping user to control the operation of the doorbell at the distance of 2-3 meters. There are four buttons on the front side of the remote control which correspond to the four control buttons on the box.

A LCD screen is attached on the remote control to show the name of the melody which is being played.

Technical Challenges

The main technical challenges in this project were:

1. Programming a system which can accurately reproduce any musical note.
2. Creating a system which can store a tune that a user inputs in non-volatile memory.
3. Creating an IR link to communicate data between the remote and receiver.
4. Interfacing with the LCD display.
5. Creating a PCB layout for the remote which could fit all the components onto a single sided board in the incredibly tight space we had available.

I will briefly talk about how we approached, and developed solutions to, these challenges.

Challenge 1: Programming a system which can accurately reproduce any musical note.

This challenge arose due to the fact that musical notes are very precisely defined, and a deviation of even a few percent from the true value of a note means that it wouldn’t sound pleasant.

The first approach which we thought about taking was to use a delay loop style timer to waste time. This type of system works by decrementing a register continuously until the result is zero. These loops are often nested to allow for longer delays. The problem with this approach is that it is difficult to get a wide range of frequencies while still having a precisely defined timer, since each level of nesting effectively reduces the precision by an order of magnitude (unless you choose to employ some very tricky code to individually set the value of each delay loop counter).

We solved this problem using the TMR0 interrupt. This is a special register inside the PIC which is automatically incremented each instruction cycle. Once the register overflowed, the interrupt is called. The advantage of this approach is that by setting the initial value of the TMR0 register, the duration until the interrupt is called can be very precisely defined. Additionally, a scaler can be applied to the TMR0 register so that it is only incremented on every $(2^n)^{th}$ instruction. This is perfect for us because the musical scale is an exponentially growing function, so the same note in sequential octaves is exactly double the frequency of the note in the previous octave. The TRM0 scaler can therefore be used to set the octave.

Challenge 2: Creating a system which can store a tune that a user inputs in non-volatile memory.

The solution to this problem was inspired by carefully reading the datasheet. Upon careful inspection, one will notice that the PIC16F84A has 64 bytes of EEPROM storage. This is perfect for our use because it is non-volatile and at 2 bytes per note (with four bytes as a ‘finish’ marker) we can store a 30 note user defined tune.

With this information, we were able to create subroutines to read and write bytes to EEPROM memory. Once these were tested and working, it became almost trivial to integrate this with the existing system.

Challenge 3: Creating an IR link to communicate data between the remote and receiver.

This challenge was a hardware challenge, it involved finding a solution which could reliably communicate the data we required.

The initial approach which we thought about using was to use an IR led to transmit the data we required and then using an IR diode with an amplifier circuit to recover the signal at the other end. The problem we found with this system is that the ambient light levels were too variable to accurately calibrate an amplification circuit over a range larger than a few centimetres.

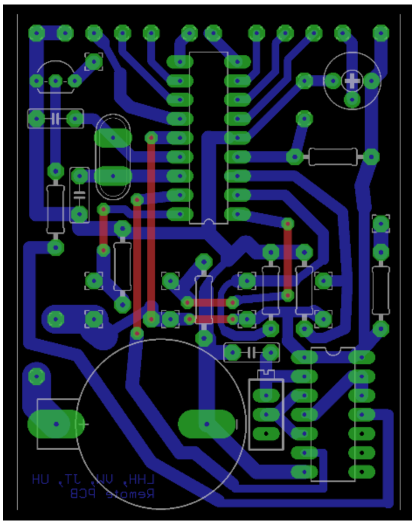
Having hit this roadblock, we did some research into how commercial IR communication channels (such as TV remotes) worked. We found that these systems usually send an IR signal that is modulated onto a carrier signal (usually 36 or 38kHz) using simple on-off modulation. The signal is then decoded at the other end using a specially bought module which looks for the carrier.

We managed to find a module in the lab which was designed to receive such IR signals and, having figured out the pinout using trial and error, we discovered that it was designed for communication using a 36 kHz carrier. Using a signal generator as a test input, we were able to get communication over a range of a few metres. This research allowed us to build our communication link around this established method.

Challenge 4: Interfacing with the LCD display.

Interfacing with the LCD display was mostly a matter of reading the datasheet and making sure to pay attention to the commands that the LCD expected. The procedure to initialise the LCD in 4-bit mode and get a flashing cursor was fairly complex; however once we had that working it meant that we had a fairly good understanding of how the display worked and which commands it expected in what format. Therefore we then programmed a subroutine to send arbitrary bytes to the LCD and so from there it wasn’t much more of a challenge to get the display to show whatever we wanted.

Challenge 5: Creating a PCB layout for the remote which could fit all the components onto a single sided board in the incredibly tight space we had available.



The remote PCB layout was quite a large challenge in that the space was incredibly limited, yet the circuit was still reasonably complex. The final design is shown to the right. Copper traces on the bottom side are shown as blue tracks, red tracks show jumpers which are added manually using wires on the top side. This result was achieved by placing connected components near to each other where possible, in order to reduce signal paths. In addition most traces were hand-routed (as opposed to using the auto router) in order to fit everything in and ensure that traces which carried a non-negligible current were as thick as possible.

Overall the result meets all the requirements it needs to. It works perfectly, fits inside the remote box, and has only six jumper wires on the top side.

Instructions

1. Setting up the system

To make the doorbell ready for use, simply press the power switches on the side of the remote and receiver so that both are in the 'on' position.

Additionally, one can adjust the knob on the side of the receiver to set the desired volume.

2. Playing a tune

To play a tune, press one of the four buttons on the top of either the remote or receiver. From left to right, the tunes that this will play are:

1. Happy Birthday to You
2. Imperial March
3. The full range of notes that the receiver is capable of
4. A tune which has been programmed by the user

If you activate a tune from the remote, you will get visual feedback of which tune you have chosen from the LCD display.

N.B. If using the remote control, there must be clear line of sight from the LED on top of the remote to the IR receiver on the side of the receiver.

3. Programming a custom tune

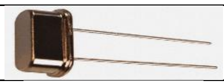
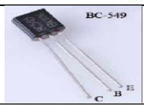
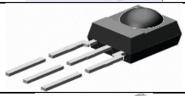
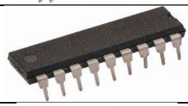
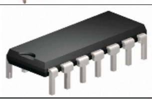
To program a custom tune into the doorbell's memory, look at the side of the doorbell receiver. These seven buttons form a miniature keyboard comprising the notes of the octave of middle C.

To start programming a tune, press one of the buttons, you will hear audible feedback of the note you have selected. Pressing a button will store that note in a tune. The doorbell is capable of storing a tune of up to 30 notes long.

To finish programming, press one of the tune buttons on the top of the receiver. The tune you have programmed will remain in the microcontroller's memory even if the power is turned off.

N.B. Pressing a programming button will overwrite any tune stored in the microcontroller's memory at that time.

List of components used in the doorbell circuit and remote control

No	Component	Value	Brand	RS Stock Code	Model	Price per unit	Quantity	Total Price
1	Resistor	150 Ohms	Ohmite	2508403624		£ 0.02	1	£ 0.02
2	Resistor	220 Ohms	RS	739-7061		£ 0.02	1	£ 0.02
3	Resistor	470 Ohms	Ohmite	2508405709		£ 0.03	1	£ 0.03
4	Resistor	4.7k Ohms	Arcol	386-919		£ 0.116	2	£ 0.232
5	Resistor	10k Ohms	TE Connectivity	753-6809		£ 0.67	1	£ 0.67
6	Resistor	47k Ohms	RS	740-0909		£ 0.022	16	£ 0.352
7	Resistor	56k Ohm	Bourns	522-2924		£ 0.58	1	£ 0.58
8	Capacitor	15 pF	Murata	652-9771		£ 0.23	4	£ 0.92
9	Capacitor	1 nF	AVX	537-3757		£ 0.116	1	£ 0.116
10	Crystal	4MHz	RALTRON	693-6936		£ 0.143	2	£ 0.286
11	NPN Bipolar Transistor	TIP31A	STMicroelectronics	485-9749		£ 0.374	1	£ 0.374
12	Transistor	BC 549	Fairchild Semiconductor	739-0442		£ 0.10	1	£ 0.10
13	IR Receiver	38KHz	Vishay	773-0367		£ 0.486	1	£ 0.486
14	IR LED		O SRAM OPTO Semiconductors	665-5476		£ 0.179	1	£ 0.179
15	Microcontroller	Pic 16F84A	Microchip	379-2897		£ 3.22	2	£ 6.44
16	NAND Gate	4093	Texas Instruments	663-0483		£ 0.46	1	£ 0.46
17	LCD		Farnell	2063156		£ 6.96	1	£ 6.96
18	Speaker	8 Ohms	Kingstate	771-6985		£ 1.098	1	£ 1.098
19	Tactile Switch		TE Connectivity	718-2415		£ 0.074	13	£ 0.962

20	Latching Push Switch		SCI	78-0235 (Rapidonline)		£ 0.8184	4	£ 3.2736
21	Rocker Switch		RS	419-750		£ 0.616	1	£ 0.616
TOTAL PRICE								**£ 24.1346**

As can be seen from the table above, twenty one components in total were used in the door bell and remote control circuits. And the total price for the whole product is about twenty four pounds which is acceptable because besides the minimum specifications, we also have some innovations which are a remote control integrated to a LCD as well as a keyboard buttons located on the box to help user can easily use this product. And the price for making these addition features is quite high with £ 6.96 and £ 3.22 spent on the LCD and an extra PIC controller. Moreover, we just ordered one or two of each of the components so the price we have to pay is a little bit higher in compare to buying them with more quantity. In overall, twenty four pounds for a doorbell with quite a lot of innovations integrated in is totally worth in our opinion.

CONCLUSION

We started with a PCB and some basic electronics components with PIC, speaker and a box. The basic plan was to change this all into a Doorbell with some enhancements. In the process we had to do programing, make new PCB designs, drilling boxes and to ordering of some components etc. We had some concept plans and some plans that we knew that we will achieve by the end of the session. We worked hard like a team and by the end of the day we manage to achieve everything what we had planned. The doorbell is working perfectly with 4 tunes additionally it works with IR remote that has LED on it displaying the tune name, it also has a keyboard that can make and save tunes. Overall it was a good project and we managed to make a good device with affordable price of twenty four pounds.