ELECTRONICS DESIGN LABORATORY

UNIVERSAL REMOTE CONTROLLER

Guide: *Prof. Rajbabu Velmurugan*

Group D-08

Shobhraj => 06D07039

Ashok => 06D07040

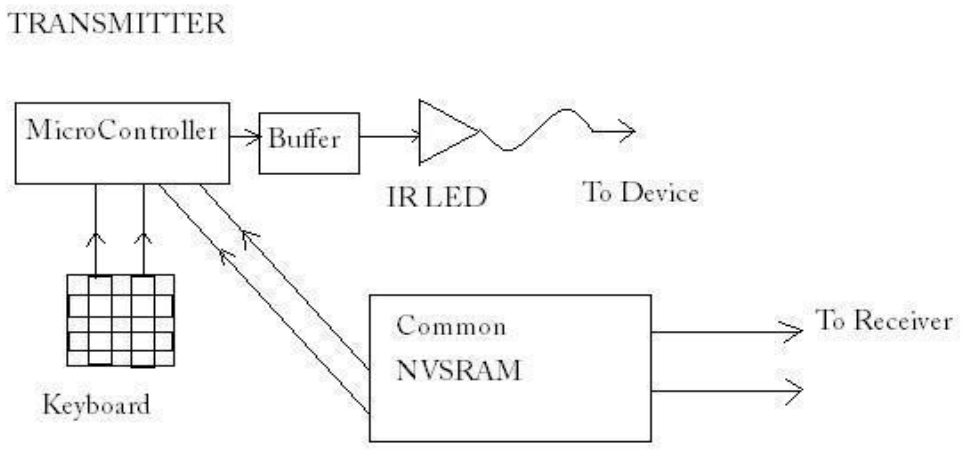
Santhosh => 06D07041

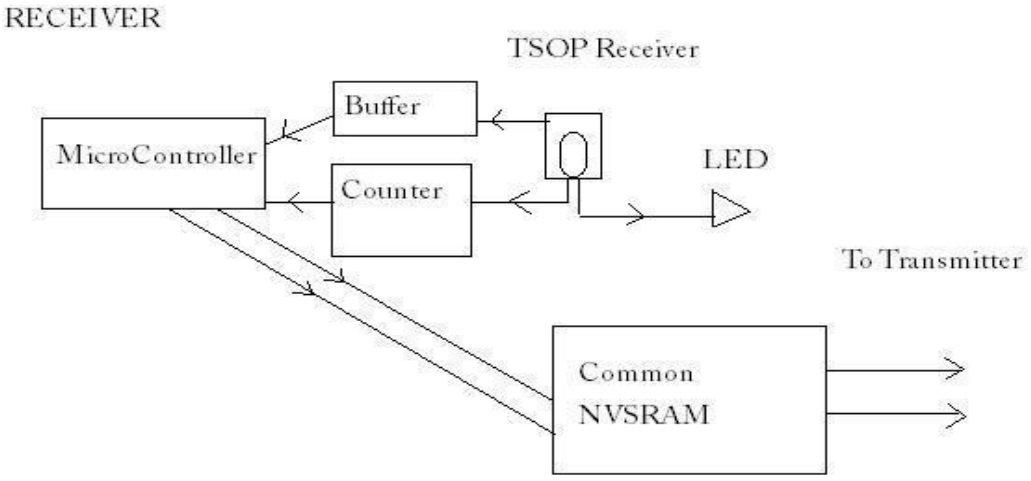
Introduction

With advances in technology, there are more electronic devices in our daily lives than we can actually efficiently handle! So, with the advent of the present technological revolution comes the difficult task of controlling our electronic equipments. Naturally comes into our minds the 'All for one, one for all' strategy and we ponder over the possibilities of a universal remote controller. One controller for all our devices does sound lucrative, doesn't it! With the objective of making lives easier, we set sail on our mission, to design a low cost 'learning' universal remote controller, a remote that could 'learn' from other remotes.

Initial Idea

The plan initially was to use two separate circuits as the transmitter and the receiver which would mean that two microcontrollers are utilised. It was an overkill to use two microcontrollers. An IR LED is the major transmitter component and a TSOP package is the major receiver component. A ROM/NVSRAM was to be connected common to both the microcontrollers. It would store the learned values through the receiver to be accessed later by the transmitter. The crude initial block diagrams for the two circuits are shown below.





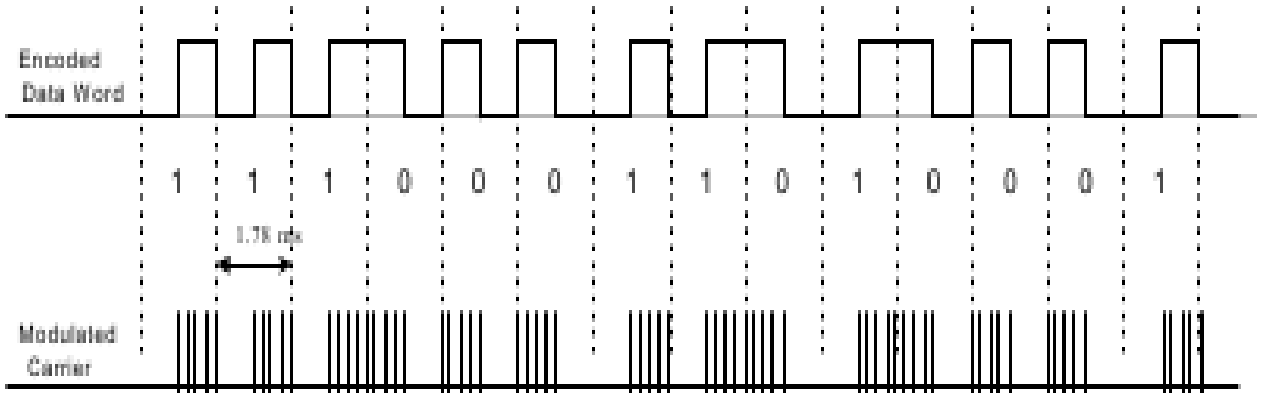
Also planned was an LCD display which didn't materialize due to the lack of time. There were several other changes to the initial plan. The circuit diagram and the block diagram are shown in figures later, but first; let us briefly discuss the RC5 protocol for remote controls.

The RC5 protocol

The RC5 protocol, devised by Philips is one the most commonly used remote control protocols and most of the big names in the industry use it. This protocol uses a bi-phase modulation technique to somewhat decrease the effect of stray IR radiation, which means that each bit transmitted consists of two levels based on a transition. So, technically, each bit consists of a transition. This is a type of Manchester coding. Logic '1' would be a low to high transition and logic '0' would be the opposite. One whole RC5 code, i.e. the code for one operation consists of 14 bits, each bit is 1.778 ms long. This signal is modulated using a 36-40 kHz carrier frequency. The figure shows the modulation of the signal. We can see that when the bit is high, there is an oscillation of the signal at the carrier frequency.

St1	St2	Ctrl	A4	A3	A2	A1	A0	S5	S4	S3	S2	S1	S0

The 14 bits of the RC5 code are designated functions as shown in the figure above. The first two bits are the start bits. These are almost always at logic ‘1’ and serve the function of calibrating the receiver loop. Next comes the control bit which toggles every time. This is used to distinguish the first press from the second as if the same button is pressed twice, the system can detect two different signals and hence perform the required function twice. The next 5 bits correspond to the device selection. Each device is associated with a different code. This value is usually ‘0’ for a television and ‘18’ for tape recorders and can be anything in between ‘0’ and ‘31’ for different devices. The next 6 bits are the command bits and specify the function to be done. As a convention, values are assigned. For example ‘16’ is volume up and ‘17’ is volume down. So, there can be 64 functions such assigned.



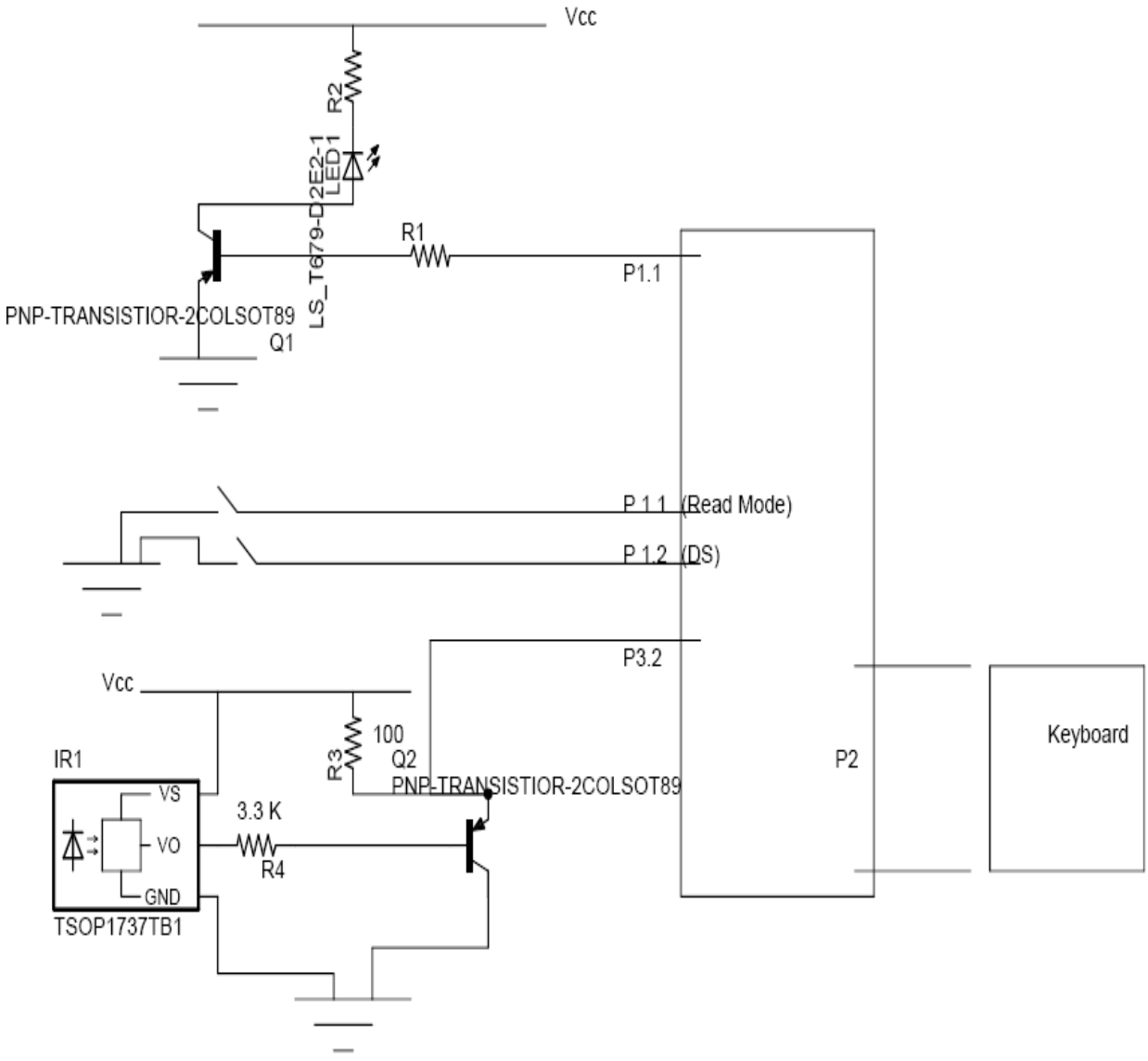
Implementation

Major Components

- **ATMEL 89C51**: A low-power, high-performance 8-bit microcontroller with 4Kb of flash programmable memory. A simple, but efficient CPU suitable for our needs. Also, a low-power idle mode is available which is utilised by the circuit.
- **TSOP 1738**: Vishay IR Receiver module for signals around 38 kHz. This effectively reduces noise by blocking out effects of DC light, fluorescent lamps and continuous signals even if they are centred around the rated frequency. Its output is reversed, i.e. if the value of the input is high, the output is low and vice-versa.

Hardware: The Circuit

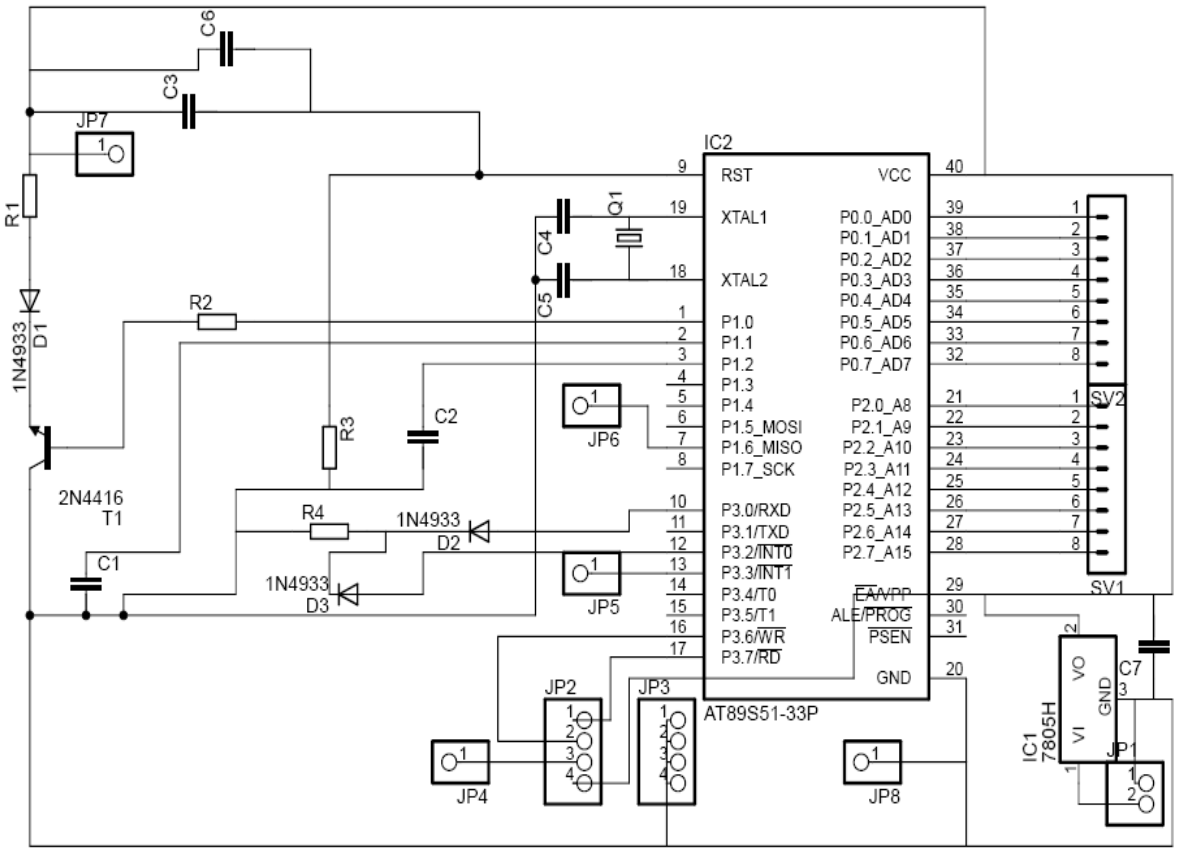
The final circuit is a smaller version of the initial plan. There is just one microcontroller, instead of the earlier planned two. This 89C51 controls both the reception and the transmission parts of the circuit. There is a switch to select between the two modes as shown in the circuit diagram below. Values for two devices can be stored at one time and another switch controls this. Each function is explained in some detail below.



- **Transmitter**: The circuit around the IR LED is the transmission part of the project. As shown, a BJT(pnp) is connected in a common emitter configuration to drive the LED as the current from the microcontroller

would not suffice. The used resistance values are shown. P1.0 is used to control the transmission. Software delays are used to generate the required carrier frequency. The microcontroller accesses the stored values from the memory and outputs it.

- **Receiver**: The TSOP is the main receiver element in the receiver circuit. The output is amplified using a common-emitter configuration with a BJT(pnp). The output of the amplifier is used to poll through P3.2 and the analysed values are stored in the memory which can be accessed by the transmitter later. When there is a modification in a value, all the other stored values are correspondingly changed to suit the particular device, i.e. the device address bits of all the stored values are modified. This is possible as the command codes for all media devices are the same as described in the RC5 discussion.



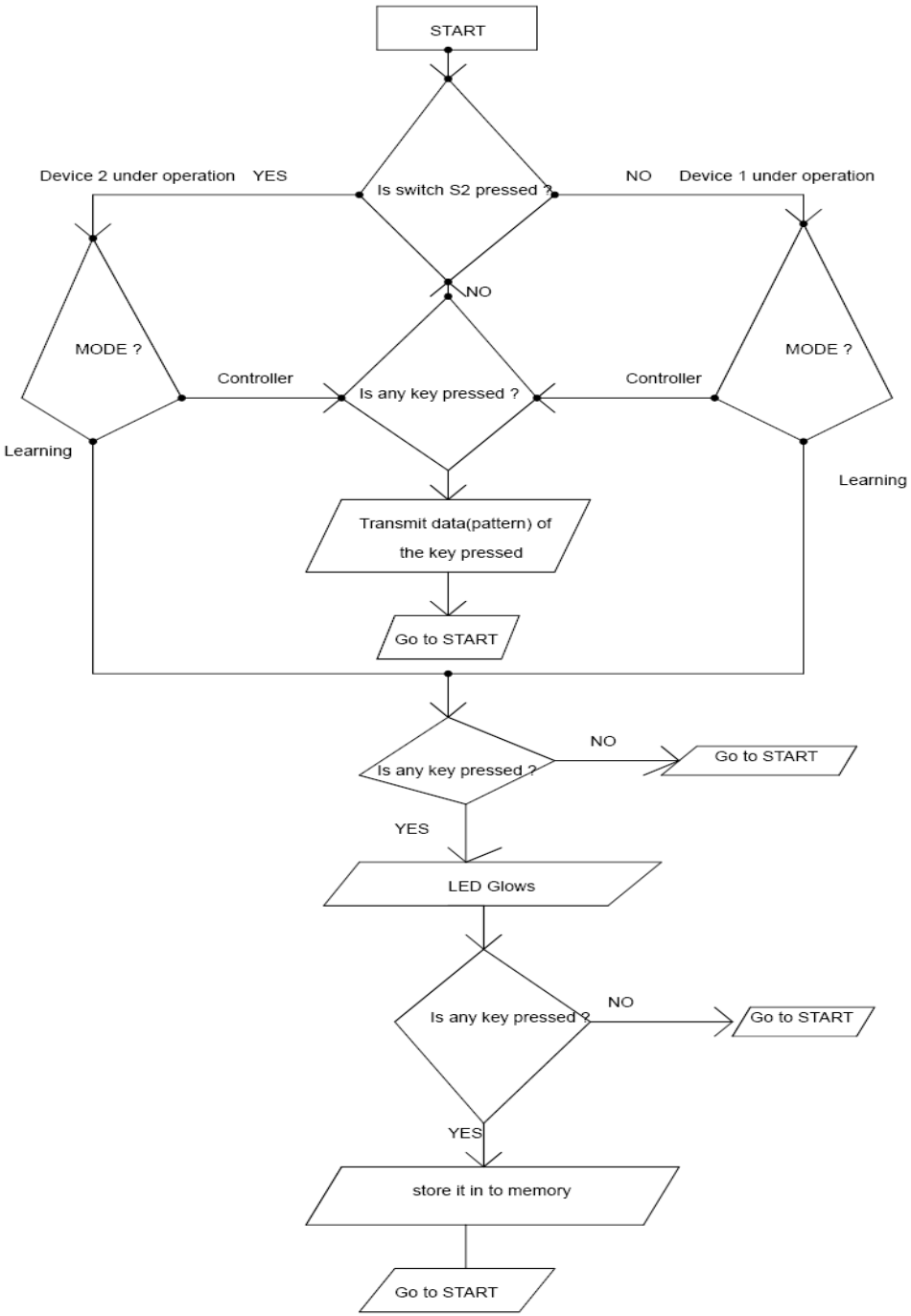
- **Switches**: As shown, a 4x4 keyboard is connected to Port 2 of the microcontroller. These keys are used to access the memory locations which store the codes for transmission. In the transmission mode, the press of a key outputs the corresponding memory location value through

the IR LED. There are two other single switches. The switch connected to P1.1 is used to select the transmission mode or the receiving mode. The other switch which is connected to P1.2 is used to select the device. Values for two devices can be stored at one time. Polling is done at all times on these pins to select the mode and the device.

Software: The Coding

The coding procedure is depicted in the flowchart shown below.

- At the start of program execution, the microcontroller checks P1.1 to select the device.
- After the device is selected, the microcontroller checks P1.2 to select the mode.



- If the write mode, i.e. the receiver is activated, with the press of a key in the 4x4 keyboard and the pressing of a key in the remote control of the required device, the code is captured by the receiver and stored in the memory location corresponding to the key press. An LED glows to show this. Then, all the other stored memory locations are correspondingly changed to suit this particular device and then the program starts again.
- If the read mode, i.e. the transmitter is activated, with a press of a key, a stored code corresponding to the key gets transmitted through the IR LED and the program starts again.

Problems Faced

- The TSOP output carried a lot of noise. Though not supposed to do so, stray signals of frequency 40 Hz were acting as a hindrance to the actual signals which had to be measured. We tried all sorts of tricks to get rid of it and spent a lot of our time trying to fix it. Finally, a change in the TSOP did the needful. Then, its output was amplified using a common emitter pnp transistor.
- According to the initial plan, a ROM or NVSRAM was to be connected to store the codes. The NVSRAM that was available had a low rated voltage and hence its use was ruled out as it would require more circuitry and would compromise on the efficiency. Then, according to availability in the lab, we settled on 24C02 serial ROM, which is compatible to I2C interfacing. Now, the I2C protocol requires a high degree of synchronicity, which was difficult to obtain using the microcontroller. We spent a lot of time in making this work and in the end decided on using the microcontroller's volatile memory instead as we thought the pain wouldn't be worth the gain. This would not be able to keep the values when switched off, but still, the read and write rate would be much faster and owing to the availability of a low-power idle mode in the 89C51 microcontroller, power consumption would be comparatively less.
- The IR LEDs require a comparatively higher current to be driven properly. The microcontroller on its own couldn't do so and hence, another common emitter pnp transistor was used to amplify the current.

Testing

A digital signal oscilloscope and any media remote controller can be used to test the device as the device isn't fully ready to be checked on an actual appliance.

- In the write mode, point the media remote controller to the receiver module and store the values under a particular device location in the microcontroller RAM.
- In the read mode, transmit the saved code and compare it with the code produced by the actual remote controller.
- We can see that the waveforms match and hence the test is successful.

Improvements

- To increase the distance of operation, power amplifiers can be connected to amplify the signals further.
- A ROM can be connected to store the codes even when the device is switched off. With increase in memory, we can also have codes for more number of devices stored.
- To add to the aesthetic appeal, an LCD which displays the current mode and the selected device could be connected.