EE 318 Electronic Design Lab -I, Project Report, EE Dept., IIT Bombay, April 2006

# RFID Reader at 13.56 MHz

Group No. B01
Lokesh Kumar Singh (03007008) <lokesh@ee.iitb.ac.in>
Vinay Verma (03007021) <vverma@ee.iitb.ac.in>
Pradeep Nannamu (03007038) <pradeepn@ee.iitb.ac.in>

Supervisor: Girish Saraph

Course Instructor: Prof. P.C.Pandey

**Abstract**

The objective of this project is to build a RFID reader at 13.56 MHz. The various components used for the reader design is the transceiver circuitry, the microcontroller circuitry, the antenna and the tags. Ideally, the transceiver sends out an RF signal at a preset interval to locate the positions of the tags. If the tag is in some particular range it will reply back for the request send. This response is received by the reader and sent to PC terminal using RS232 serial interface. The protocol implemented is ISO 15693, using S6700 transceiver IC manufactured by Texas Instruments. We are able to successfully communicate and transmit the data though the transceiver, since we don't have the antenna ready we are unable to check the response from the tags.

## 1. Introduction

The main aim of this project is to design a RFID reader at 13.56 MHz which is able to communicate with one tag tuned to 13.56MHz frequency. For this purpose, we used the transceiver IC S6700, manufactured by Texas Instruments. S6700 is capable of implementing different protocols and has minimal interface circuitry which is readily available in the data sheet of the transceiver. The transceiver is interfaced with PC through a microcontroller. The microcontroller we used is Atmel's AT89C2051.

## 2. Design approach

The basic block diagram for a reader module is shown in Fig. 1

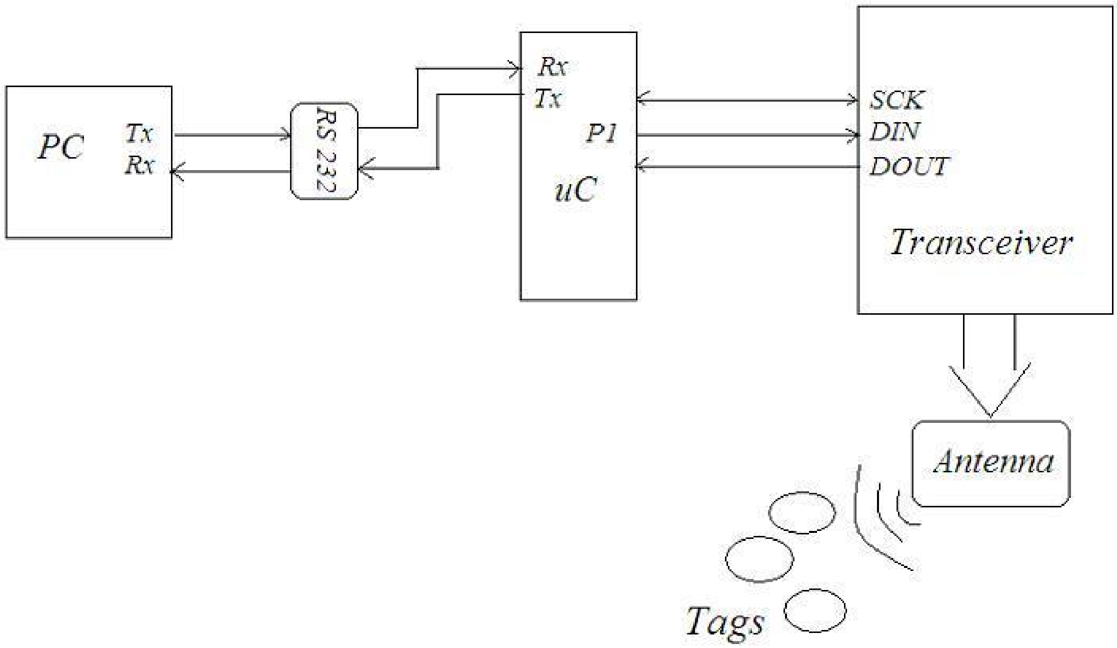


**Figure 1 Block diagram for reader**

### Transponders/Tags/Labels

The tags in our system are passive inductive tags and draw power from the RF signals generated by the transceiver. The tags work with ISO 15693 protocol and have a limited operating range.

### Transceiver IC

The transceiver circuit consists of a TI S6700 transceiver chip, a 50-ohm antenna, and other passive circuit components. In ISO 15693 mode, once activated by the microcontroller, the transceiver sends out a data stream at 13.56 MHz. The data stream contains a command code, address of a specific tag, and checking bits. Tags within the operation range of the system reply the transceiver by sending a data stream indicating whether they are the tags the transceiver requests. The transceiver then outputs to the microcontroller the status of the requested tag – present or absent.

### Microcontroller

The microcontroller sends commands to the transceiver for implementing the desired protocol and decodes input data from the transceiver and outputs the received data to the PC terminal.

## Serial Interface

The response from the tag is received by the transceiver and sent to the microcontroller. This response is sent form the microcontroller to the PC terminal using the MAX 232 voltage converter and DB9 converter.

# 3. Design Details

a. The Transceiver Unit

The TI-S6700 IC is a HF RFID Reader operating at 13.56 MHz. It is the heart of our project. This transceiver works according to the “reader talks first” principle, meaning that the transponder will not respond unless the transceiver sends a request to the transponder. Our transceiver serves as both an RF reader and receiver, and it can operate in four different modes: Tag-it protocol, ISO/ IEC 15693-2, ISO/ IEC 14443-2, and Direct-Mode. In our design, we initially chose Direct-Mode. The main advantage with Direct-Mode is that it modulates and transmits the input stream directly without processing anything. The output can be easily verified using this mode. However, Direct-Mode is not allowed in many transponders. For our project, we had to use the ISO 15693 protocol because our tags support that mode.

For ideal RF output, we designed the transceiver/antenna component with the capacitor and resistor values with the values but we haven’t tuned the circuit yet, but it will be done soon. As shown in the transceiver circuit diagram, a 12Ω resistor is used to create a 10% modulation depth. However, in our design it was replaced with an 18Ω resistor to achieve the recommended 20% modulation depth. The inductors and capacitors were added to TX_OUT to create impendence matching for maximum transmission output. Fig. 2 shows the biasing circuit for the transceiver:

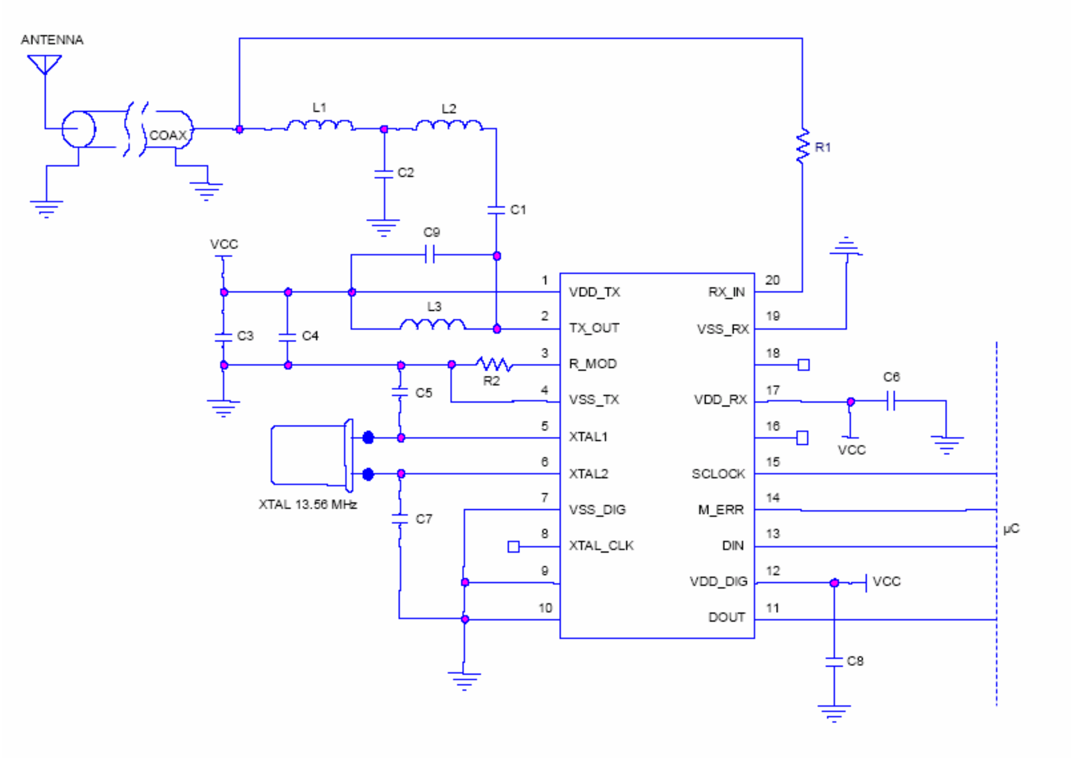


**Figure 2 Transceiver circuitry from TI S6700 manual**

Table 1: List of components (taken from s6700 datasheet)

| Component | Value | Component | Value |
|---|---|---|---|
| C1 | 10nF | L1 | 4.2µH |
| C2 | 56pF | L2 | 5.6µH |
| C3 | 10µF Tantalum | L3 | 1.2µH |
| C4 | 100nF | - | - |
| C5 | 22pF | R1 | 2.2kΩ |
| C6 | 100nF | R2 | 12Ω |
| C7 | 22pF | - | - |
| C8 | 100nF | - | - |
| C9 | 47pF | - | - |

b. Microcontroller

Atmel AT89C2051 is 8051 compatible (supporting two 8-bit I/O ports, two 16-bit timers) micro-controller. It has 2Kbytes on-chip flash program/data memory that can be programmed. The main reason to use this microcontroller is because we required only one port to interface with the transceiver IC and it can handle memory requirements of the code. The microcontroller sends the received response from the transceiver to the pc terminal using the maxim's MAX232 chip (voltage converter) and DB9 connector.

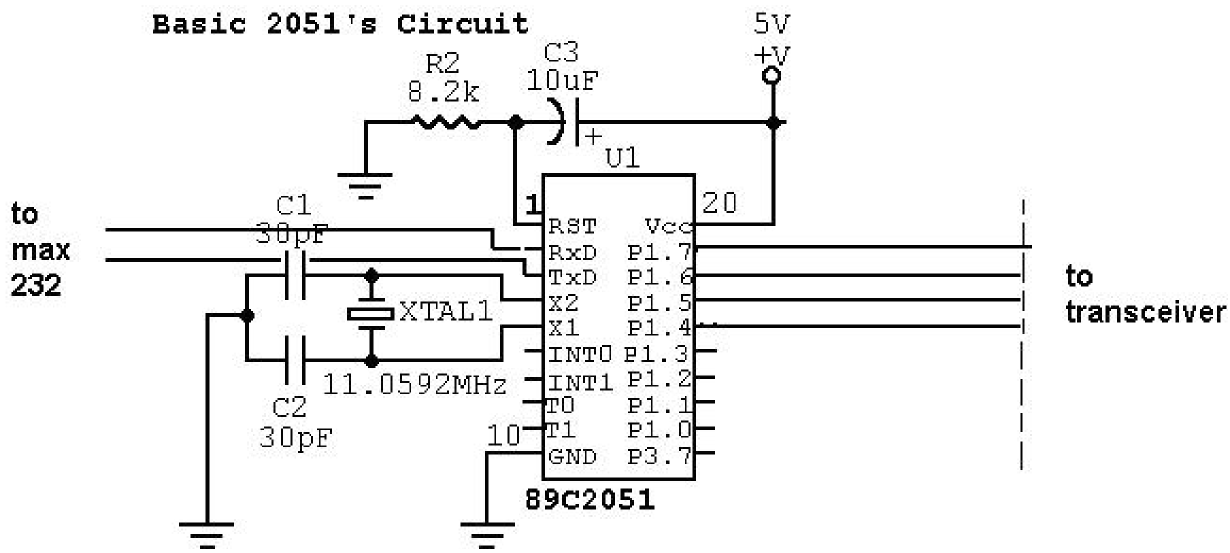


**Figure 3 89c2051 circuit diagram**

## 4. Design of Circuit

The whole design can be broken largely into three blocks. They are:

1) PC and Microcontroller interface
2) UC and Transceiver Chip interface
3) Antenna Matching and Tuning Circuit

1) PC to Microcontroller interface:

The 9-pin serial port of computer (COMM1/COMM2) converts the data from parallel to serial and sends it to MAX232 chip, which is essentially a voltage level converter. A standard serial interfacing for RS232 requires negative logic, i.e., logic '1' is -3V to -12V and logic '0' is +3V to +12V. This conversion of TTL level to RS232 standard is done by using a converter chip. The chip used here is the MAXIM 232 which operates at 5 volts DC. The baud rate used is 9600 as it is widely applicable for all systems. The circuit diagram for the following circuit is given in Fig. 4

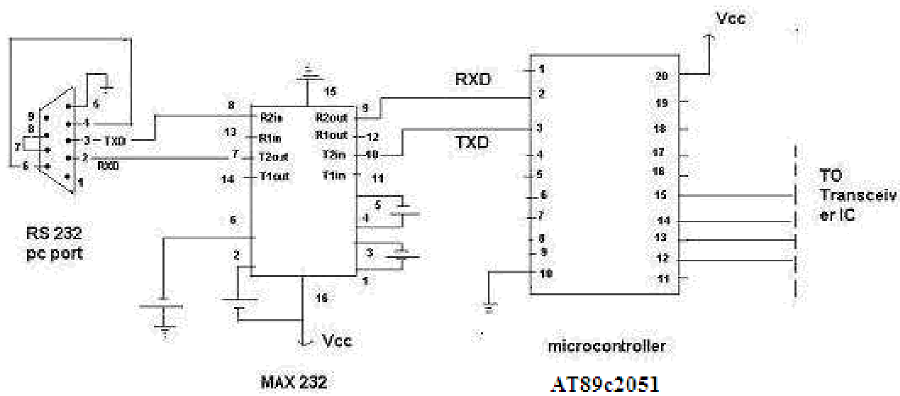


**Figure 4 Serial Interface circuitry**

We are able to send the data successfully to the pc using the serial link. This has been tested and the same code is used in the main program.

2) Microcontroller and transceiver chip interface:

The communication between transceiver and uC takes place serially through three lines: sclock, din and dout. The sclock line is a bidirectional handshake signal while sending and receiving data. The data is transmitted with sclock high. While microcontroller sends the commands to the transceiver the sclock is handled by the uC, while transceiver sends the tags response, the sclock is handled by the transceiver.

We are able to communicate with the transceiver and able to send the request commands. For checking the data that we are sending to the transceiver we used the direct mode of the transceiver, in which the same data that we input to the transceiver will be transmitted through the Tx_out pin. This has been checked using the DSO. The frequency of the oscillator circuit is also checked and has been verified that the carrier frequency is 13.56 MHz. For communication with the tags, we are using ISO 15693 protocol, in “1 out of 256 mode”. The protocol details will be explained later in the report. We are requesting the unique identifier from the tag.

The rate at which the data is received from the transceiver is not mentioned properly, also since the data is first buffered inside the transceiver, we thought of polling the sclock line so that we will be able to read the data at the maximum frequency specified by the transceiver which is 1.5 MHz. So we thought of using high speed microcontroller from Dallas Semiconductors DS89C420, using which we can easily poll the sclock frequency even if its operating at the highest specified. We ordered a sample from Dallas Semiconductors but we received a damaged uC so, we used Atmel’s microcontroller. Using this microcontroller we will be able read the data from the transceiver, if we are getting it at the same rate of communication between the transceiver and the tag which is 6.62 kbits/s. However for checking this we need an antenna, to communicate with the tags. We got the antenna made on pcb but we havn’t tuned it.

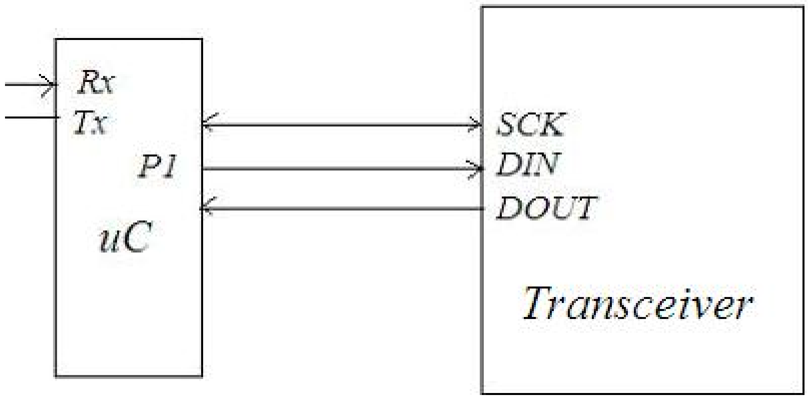

**Figure 5 Microcontroller and transceiver interface**

3) Antenna tuning circuit:

The antenna matching circuit and antenna tuning circuit is yet to be designed and calculated.

## 5. Testing of Blocks

As discussed above the circuit can be broken into three parts:

- Serial link interface:

This part of the circuit is successfully implemented. In our project we need to send data that we receive from the transceiver to the pc terminal. We used 9600 baud rate to transmit the data. The code that we used to check this is given below:

```
main :
mov sp,#30h
anl pcon,#7fh
anl tmod,#0Fh
orl tmod,#20h
mov th1,#0fdh
setb tr1
mov scon,#50h
clr p3.4
orl ie,#90h

clr ea
clr ti
setb ren
setb ea
```

This code sets the uC for sending the data to the pc terminal. The data that needs to be send to the pc terminal is to be kept in sbuf register and we have to wait till the TI bit goes high indicating the transmission of byte.

- Transceiver and uC's interface:

First the transceiver is checked in directmode of the transceiver. The command sequence for Direct Mode is 00000000. Once direct mode is initiated, a 101010 test stream is sent to the transceiver unit. The transceiver then immediately encodes the input digital data stream in Manchester code and transmits it. By probing the transceiver output, it can be easily verified whether the transceiver was accepting and understanding the inputs from the microcontroller input stream.The while loop repeats the command sequence infinitely, allowing us to actually see and verify the transceiver output on the oscilloscope. The final working program, however, will not be run in this format.

Once we verified that Direct Mode was outputting properly, we moved on to add the ISO 15693 Mode. To initiate ISO15693 mode, the following sequence was used for the command sequence, 00110100. Once the transceiver was set to this mode, we used the request flags as mentioned in the protocol documentation; 00000000.This request is not addressed to a particular tag but can be answered by any tag in the vicinity. This was done because we don't know the unique id of tags. Next, we attached the request command 0x01 that requests the unique id of each of the tags. Technically, once the transceiver receives this information, it will process it and output a Manchester encoded output to the Tag, telling the tag to respond. However how this output stream will look is unknown, as it depends on how the transceiver encodes the data. Nevertheless, it is known that if the desired Tag sees this input, the transceiver will output a particular sequence on the D_OUT. An error message from the tag indicates that the input command contains an error, most likely an incorrect CRC. The tag will completely ignore the transceiver request if it is out of range or it cannot decode the incoming sequence. For testing purposes, we sent the get version request sequence without the SID present. As long as the tags understand it is a get version request command, it will send its ID back. We did this to see if we are using ISO 15693 protocol properly and to see if we can actually capture the DATA_OUT from the transceiver.

For the final code, we implemented the ISO 15693 protocol, and the request flags are sent accordingly. We didn't check this part as we still didn't make the antenna completely. For the actual receive code, only ISO 15693 mode is initialized. The controller sends out the get version request for the first SID to test if the tag is in range. It is important to note that the code will become inefficient when there are more tags. Also we added a delay after the function call to give the transmitter enough time to process and send out the information before it is turned off. If the response of the tags is as expected, the response from the tag shall be shown on the pc terminal.

Since we were unable to get the actual tags to work like in the datasheet, in order to test this receive code, we imitated the response of the transceiver using another uC. This part is still being tested and we like to demonstrate it if time permits.

- Antenna tuning circuitry:

We got the pcb layout of the antenna but we didn't calculate the tuning elements of the circuit. This will be done as early as possible, if we are given some time.

Since we are unable to communicate with the tags at present, we tried to demonstrate the communication between two uC's one acting as a transceiver and other uC containing the main code. We are working on this, and if time permits this will be demonstrated.

## 6. Conclusions and Future Work

In this project, we showed that the transceiver was able to transmit the data both in direct mode and ISO 15693 protocol mode. The serial communication with the pc terminal was successful. We were unable to check communication between the tag and transceiver due to unavailability of the antenna.

## Future work

The antenna tuning circuit should be soldered on the pcb and tuned to 13.56 MHz. If this is done we will be able to implement the ISO 15693 protocol for which the code is ready. The receiving code of the main program is still under testing and can be completed soon as there are some minor errors while reading the incoming data.

**References:**


[1] Data sheet of transceiver S6700
Source: http://www.ti.com/rfid/docs/manuals/appNotes/lf_reader_intro.pdf

[2] Data sheet of vicinity tags:
Source: http://www.ti.com/rfid/docs/manuals/pdfSpecs/RI-TH1-CB1A.pdf

[3] Data sheets for ISO 15693 protocol documentation.
Source: http://www.ti.com/rfid/docs/manuals/refmanuals

[4] Kenneth J. Ayala, The 8051 Microcontroller, Architecture, Programming and Applications, second edition, Thomson Delmar Learning, 1997.

[5] www.ti-rfid.com for material reference.