SUNROM TECHNOLOGIES
Your Source for Embedded Systems

Email: info@sunrom.com
Visit us at http://www.sunrom.com

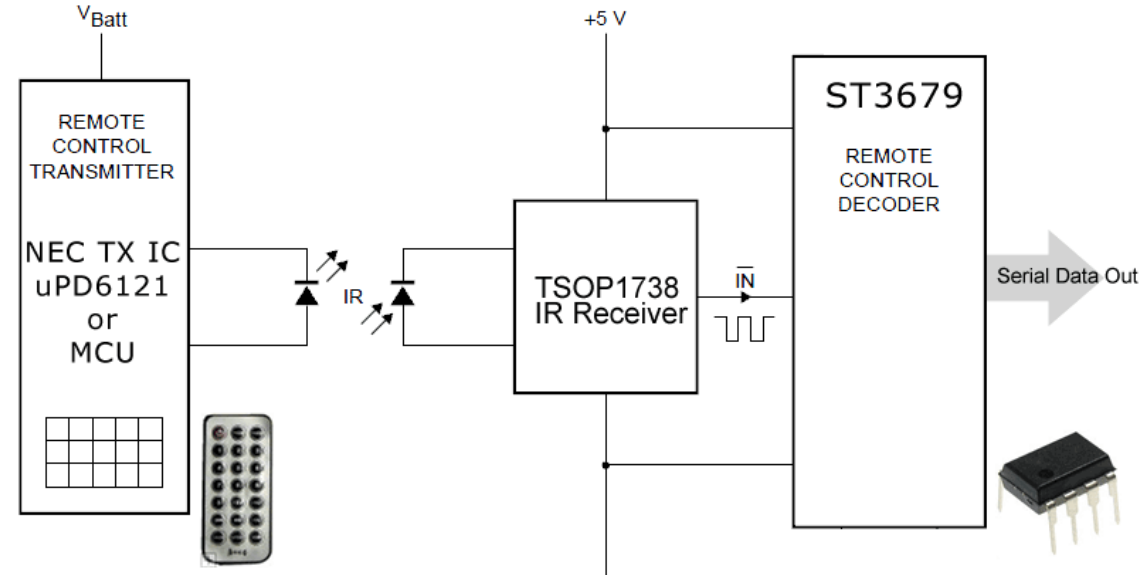
Document: Datasheet Date: 22-Mar-11 Model #: 3679 Product's Page: www.sunrom.com/p-1044.html

# ST3679 - Infrared remote control decoder NEC

The main function of the ST3679 is to decode the NEC data received from NEC format remote control into 16 bit serial data output. There are two ways of getting data from chip, Data of 9600 baud rate or Serial Shift of data through RDY, CLK and DAT pins. Many remote control applications can be developed using the decoder. Most TV, DVD, AC remotes are either RC5 type or NEC type. Remote of these labels NEC, APEX, HITACHI, PIONEER uses NEC protocol.

## Block Diagram

ST3679 needs only one component to work that is IR receiver like TSOP1738 or similar. The data is output as simple 2 bytes of serial data consisting of 16 bits of information for each key press on the remote.




NEC format Remote Control Transmitter
http://www.sunrom.com/p-1022.html

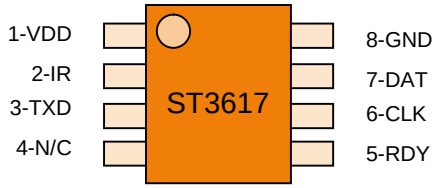
ST3679 - NEC Format Decoder
http://www.sunrom.com/p-1044.html

## Specification

| Parameter | Value |
|---|---|
| Working Voltage | 3-5V DC |
| Current | 10mA |
| Serial UART Interface | 9600 bps, 8 bit data, no parity, 1 stop bit at 5V or 3V level |
| Serial Shift Interface | Uses three pins RDY, CLK and DAT to output decoded RC5 data |

ST3679 is a Microchip PIC Controller that is programmed by us and labeled to work as per details in this datasheet.

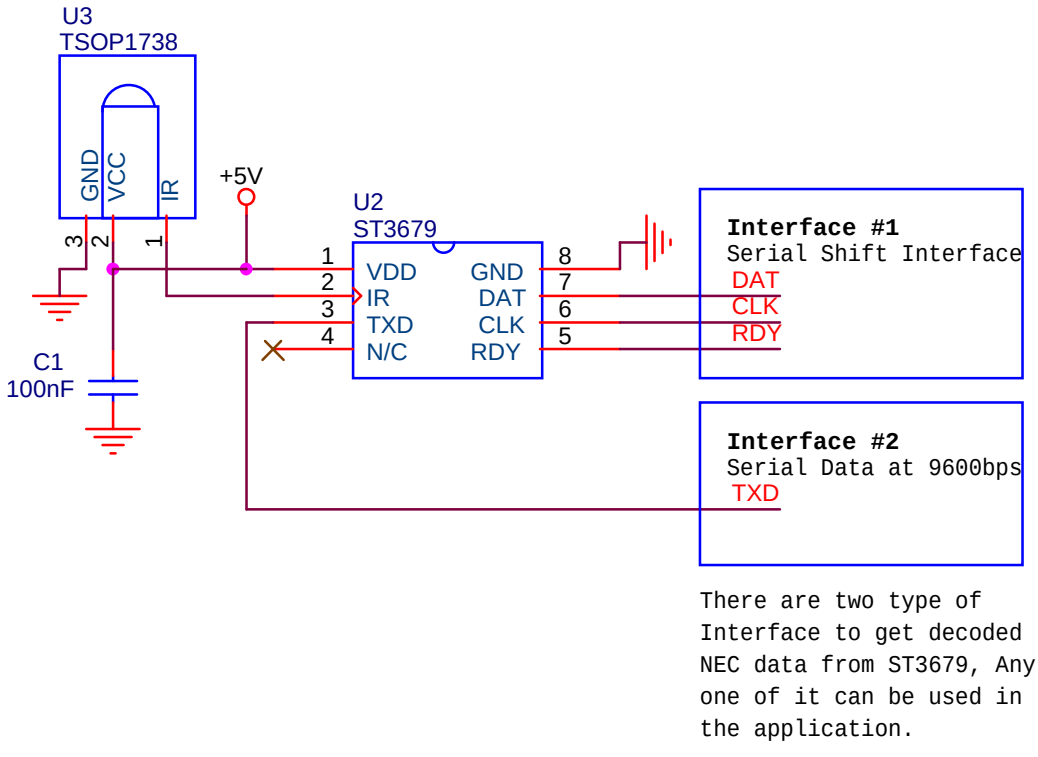
## Pin Details




| Pin# | Pin Name | Type | Notes |
|---|---|---|---|
| 1 | VDD | Supply | 3 to 5V DC regulated power supply input |
| 2 | IR | Input | Connect IR receiver output to this pin |
| 3 | TXD | Output | Outputs serial UART data at 9600 bps at TTL level |
| 4 | N/C | Not Used | Not Connected, This pin is not used and should be left unconnected |
| 5 | RDY | Output | Ready Signal Output of Serial Shift Interface |
| 6 | CLK | Input | Clock Input of Serial Shift Interface |
| 7 | DAT | Output | Data Output of Serial Shift Interface |
| 8 | GND | Ground | Power Supply Ground |

## Data Interface for ST3679 NEC Decoder

There are two ways you can get the output from ST3679. For each output the data is two bytes long containing total 16 bits NEC data.




Two types of Interface for reading NEC data from ST3679

Interface #1 is Serial Shift Interface, which consist of three pins. In this interface you have to monitor RDY pin to go low which means new RC5 data has arrived. Then you shift output data using DAT and CLK pins. You can use any I/O of your application microcontroller.

Interface #2 is Serial Data at 9600 bps is particular useful when you have a dedicated serial input pin available on your application microcontroller to get the 2 bytes of data. You can also use the serial data to interface to PC using MAX232 level convertor for serial port or use USB-TTL

chip to get a virtual serial port on PC to which many software like Hyperterminal can be connected. Custom software can also be made to monitor the incoming data.

## Output Data format

Output from ST3679 is in two bytes, thus making total 16 bits of data, let us see meaning of each bit

High Byte - First

| Bit Position-> | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Value | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |

Low Byte - Second

| Bit -> | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Value | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

A7-A0 = NEC address of remote control. This is also called custom code or device address.
D7-D0 = NEC command for each keypress at remote control.
If you get output as 0x0000 means it's a Repeat command since user has kept key pressed.

**Example**
For example pressing Key 1 on remote control can output 0x0301 where 0x03 is high byte and 0x01 is low byte. Here 0x03 is address or custom code for the transmitter which remains same for all key press. The data command will change on each key press. Special case of data is repeat command if a key is kept press. It will be output as 0x0000.

If we interpret, in terms of NEC data we get below

High: 0x03 in binary is

| Bit -> | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Data-> | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |

Low: 0x01 in binary is

| Bit -> | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Data-> | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Value | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

**Important: If you keep any key pressed, you will get 0x0000 which means repeat code. This is part of NEC format when user keeps key pressed, it send repeat command and not the whole data packet. Only on first key press you get actual data, then if key is kept press you get repeat code which is 0x0000.** If you leave Key1 and then press again, full value of data/address will be visible. Therefore this data of repeat packet tells you if user is keeping the key press or left the key once and pressed again. This is particular useful if you are implementing Toggle output like Relay ON and OFF logic.

## Serial Data Output Format

When you see data output from chip in serial at 9600 baud rate, you will get total six bytes output as each key press in ASCII format so you can view it on screen.

Let us see what data output you will get in serial mode. The last two bytes in serial mode are new line characters so that when you see this data in terminal you can see each new data in new line. If you press Key1 on remote, you will get following output

Example Output of Serial in terminal software for Key1:
0301<CR><LF>
Interpreting above result in below table

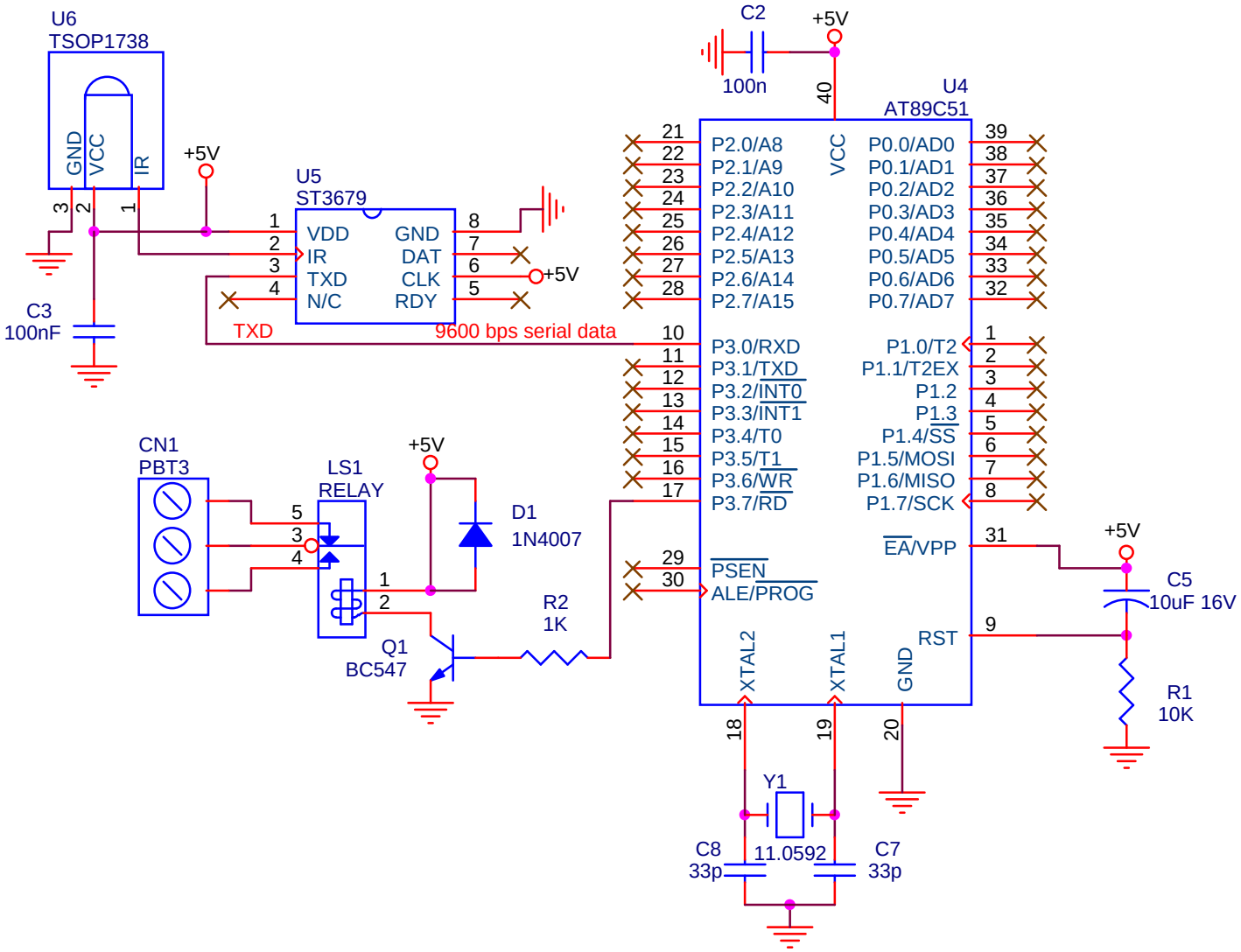
| BYTE COUNT | HEX | DECIMAL | CHARACTER DISPLAYED | Details |
|---|---|---|---|---|
| 1 | 0x30 | 4 | '0' | Data High Byte |
| 2 | 0x33 | 51 | '3' | Data High Byte |
| 3 | 0x30 | 48 | '0' | Data Low Byte |
| 4 | 0x31 | 49 | '1' | Data Low Byte |
| 5 | 0x0D | 13 | '\r' = CR | New Line Character |
| 6 | 0x0A | 10 | '\n' = LF | New Line Character |

The above values in serial data are ASCII characters. You can convert the value to binary to use in your program by deducting 0x30 from ASCII value. Our sample code given on next page uses this technique to convert this ASCII buffer of four digit to single integer of NEC data variable containing 16 bits.

## Application example using serial data

We have used AT89C51's RXD pin to receive serial data from ST3679 and we switch on Relay ON or OFF for Key1 pressed on Remote. Key1 when first pressed will turn relay ON and Key1 pressed pressed again will make Relay OFF. Relay used is 5V type. But can be any voltage if you have higher voltage available on your application. You can use any microcontroller to interface using this interface. We have chosen AT89C51 to show since it is more widely used. The sample code we have given can be adapted to any C compiler or any microcontrollers like AVR or PIC since with minor changes.

Source code can be downloaded from
http://www.sunrom.com/files/3679-samplecode.zip
Code is compiled using keil compiler

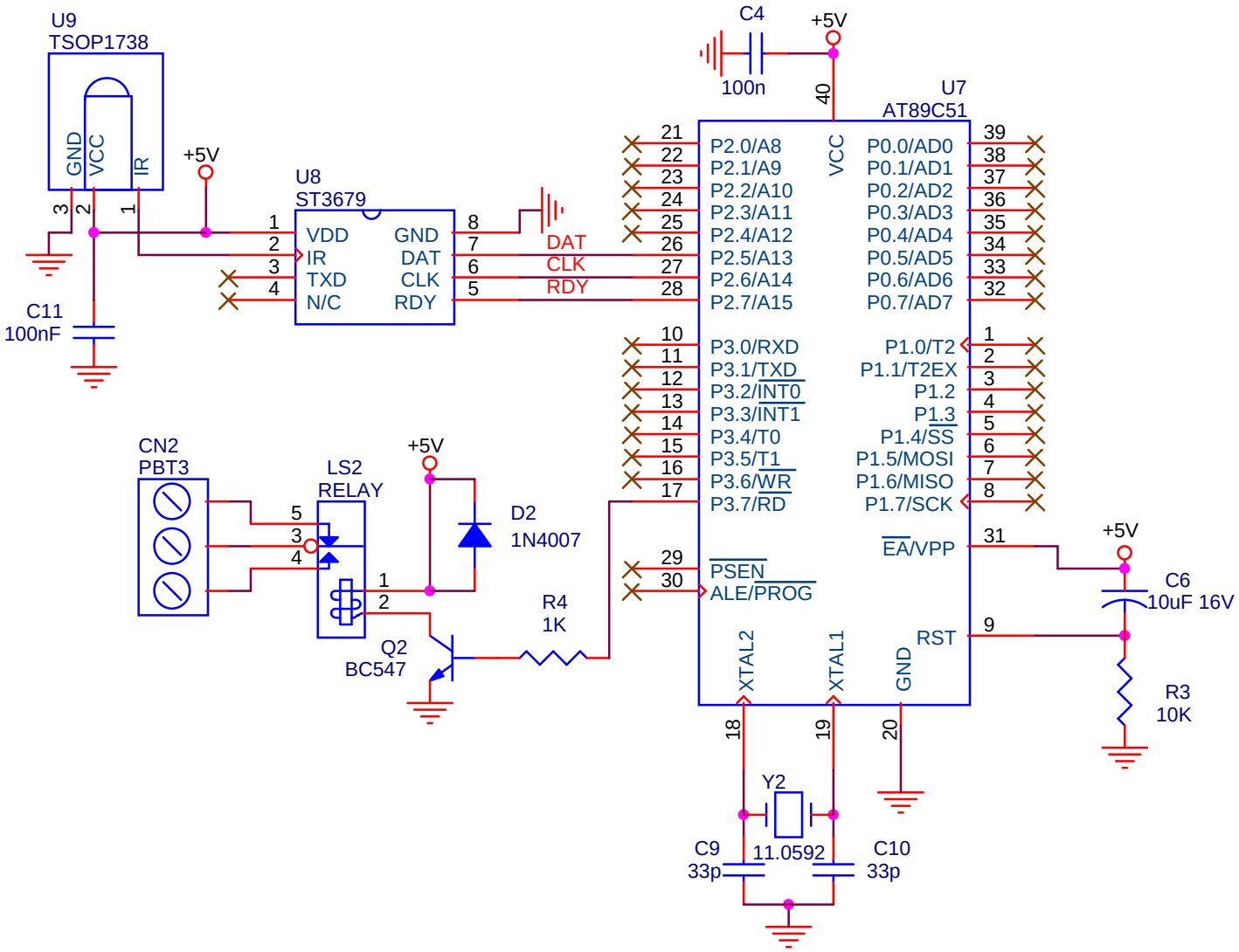



**IMPORTANT: In serial data mode, the CLK pin#6 has to be connected to +5V to get data from TXD pin as shown above.**

## Application example using serial shift data

Advantage of this technique is it uses three general purpose I/O pins of MCU instead of dedicated RXD pin. This will make RXD available for other applications .We have used AT89C51's I/O pin to receive data from ST3679 and we switch on Relay ON or OFF for Key1 pressed on Remote. Key1 when first pressed will turn relay ON and Key1 pressed pressed again will make Relay OFF. Relay used is 5V type. But can be any voltage if you have higher voltage available on your application. You can use any microcontroller to interface using this interface. We have chosen AT89C51 to show since it is more widely used. The sample code we have given can be adapted to any C compiler or any microcontrollers like AVR or PIC since with minor changes.

Source code can be downloaded from
http://www.sunrom.com/files/3679-samplecode.zip
Code is compiled using keil compiler



## NEC Infrared Transmission Protocol

The NEC IR transmission protocol uses pulse distance encoding of the message bits. Each pulse burst (mark – RC transmitter ON) is 562.5µs in length, at a carrier frequency of 38kHz (26.3µs).

Logical bits are transmitted as follows:

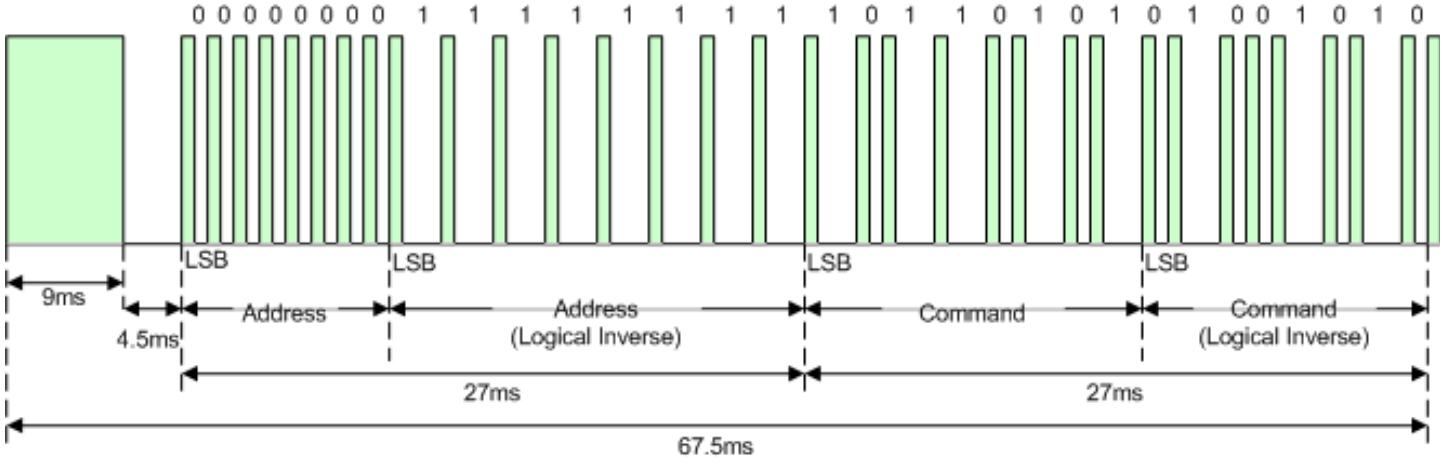
- Logical '0' – a 562.5µs pulse burst followed by a 562.5µs space, with a total transmit time of 1.125ms
- Logical '1' – a 562.5µs pulse burst followed by a 1.6875ms space, with a total transmit time of 2.25ms

When a key is pressed on the remote controller, the message transmitted consists of the following, in order:

- a 9ms leading pulse burst (16 times the pulse burst length used for a logical data bit)
- a 4.5ms space
- the 8-bit address for the receiving device
- the 8-bit logical inverse of the address
- the 8-bit command
- the 8-bit logical inverse of the command
- a final 562.5µs pulse burst to signify the end of message transmission.

The four bytes of data bits are each sent least significant bit first. Figure 1 illustrates the format of an NEC IR transmission frame, for an address of 00h(00000000b) and a command of ADh (10101101b).

Example message frame using the NEC IR transmission protocol.




Notice from Figure 1 that it takes:

- 27ms to transmit both the 16 bits for the address (address + inverse) and the 16 bits for the command (command + inverse). This comes from each of the 16 bit blocks ultimately containing eight '0's and eight '1's - giving (8 * 1.125ms) + (8 * 2.25ms).
- 67.5ms to fully transmit the message frame (discounting the final 562.5µs pulse burst that signifies the end of message).

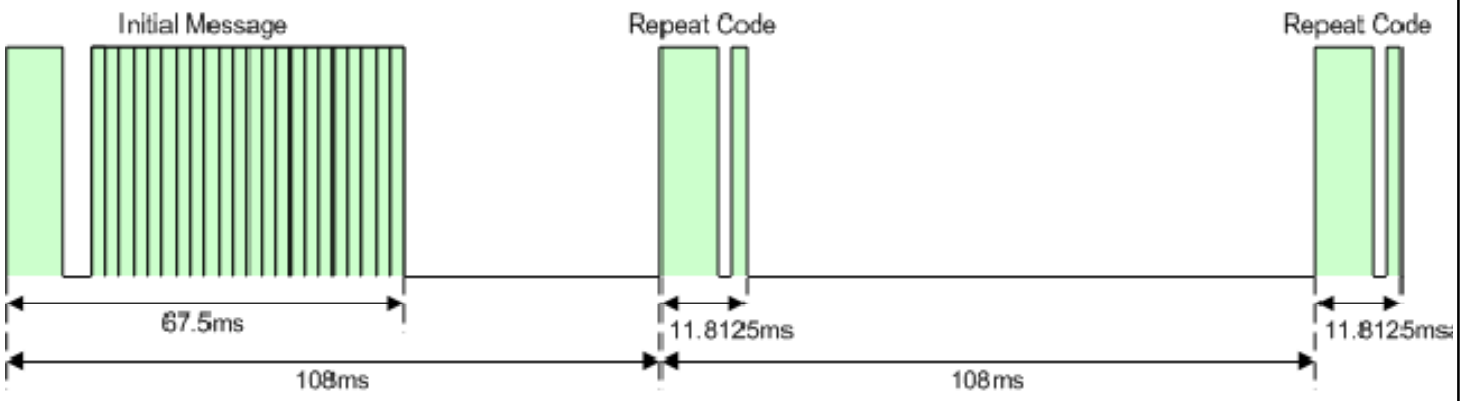
### Repeat Codes

If the key on the remote controller is kept depressed, a repeat code will be issued, typically around 40ms after the pulse burst that signified the end of the message.

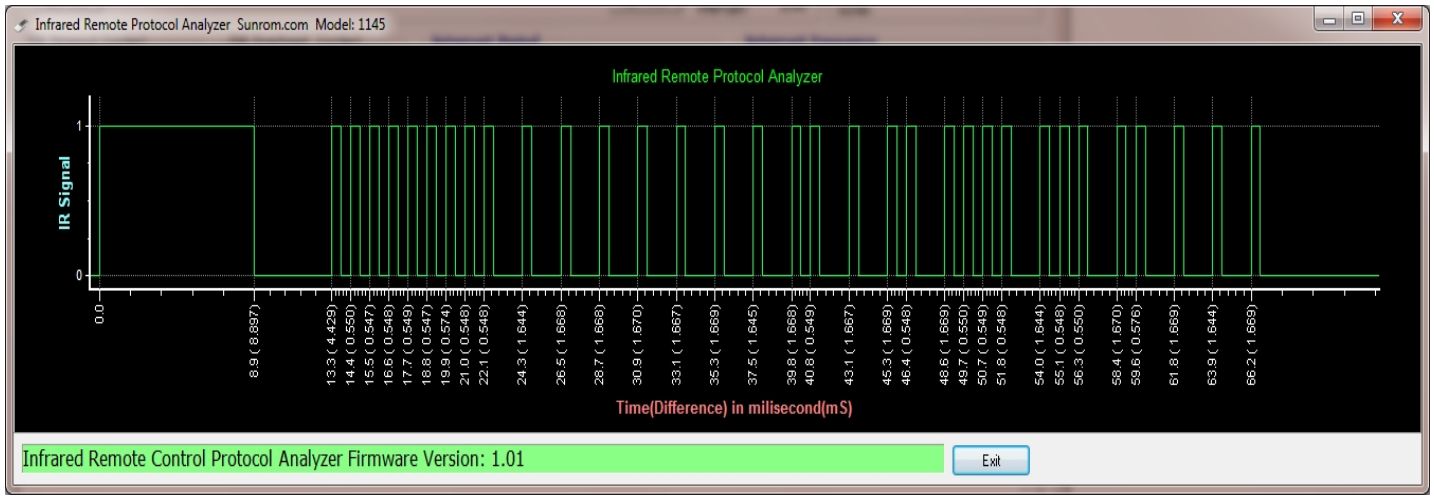
A repeat code will continue to be sent out at 108ms intervals, until the key is finally released. The repeat code consists of the following, in order:

- a 9ms leading pulse burst
- a 2.25ms space
- a 562.5µs pulse burst to mark the end of the space (and hence end of the transmitted repeat code).

Figure illustrates the transmission of two repeat codes after an initial message frame is sent.

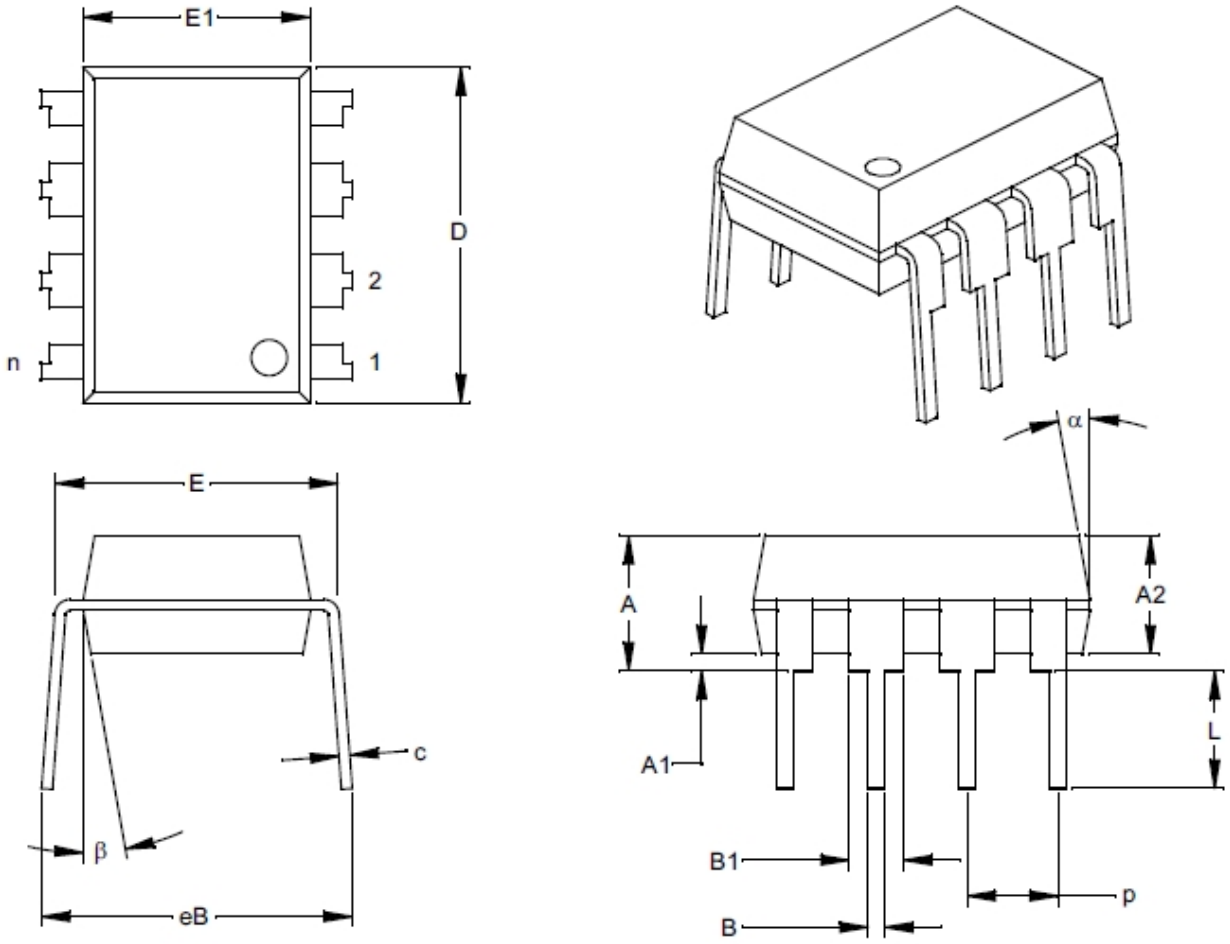



Screenshot of actual capture from NEC transmitted data, using Sunrom's IR Protocol Analyzer



## Dimensions DIP Package

**8-Lead Plastic Dual In-line (P) – 300 mil (PDIP)**




| Units | | INCHES* | | | MILLIMETERS | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 8 | | | 8 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .140 | .155 | .170 | 3.56 | 3.94 | 4.32 |
| Molded Package Thickness | A2 | .115 | .130 | .145 | 2.92 | 3.30 | 3.68 |
| Base to Seating Plane | A1 | .015 | | | 0.38 | | |
| Shoulder to Shoulder Width | E | .300 | .313 | .325 | 7.62 | 7.94 | 8.26 |
| Molded Package Width | E1 | .240 | .250 | .260 | 6.10 | 6.35 | 6.60 |
| Overall Length | D | .360 | .373 | .385 | 9.14 | 9.46 | 9.78 |
| Tip to Seating Plane | L | .125 | .130 | .135 | 3.18 | 3.30 | 3.43 |
| Lead Thickness | c | .008 | .012 | .015 | 0.20 | 0.29 | 0.38 |
| Upper Lead Width | B1 | .045 | .058 | .070 | 1.14 | 1.46 | 1.78 |
| Lower Lead Width | B | .014 | .018 | .022 | 0.36 | 0.46 | 0.56 |
| Overall Row Spacing § | eB | .310 | .370 | .430 | 7.87 | 9.40 | 10.92 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter
§ Significant Characteristic

## Related Products

**NEC Remote Control**

http://www.sunrom.com/remote-control-infrared/ir-remote-control-transmitter-nec-format

Remote control for transmitting NEC data

**NEC Remote Decoder Serial Out board**

http://www.sunrom.com/p-1045.html

**USB to Serial TTL**

http://www.sunrom.com/usb-to-serial-rs232/usb-to-serial-ttl-board

Access ST3679 data to PC using this board, It installs virtual COM port on PC to which any terminal software can connect.

**MAX232 Board**

http://www.sunrom.com/usb-to-serial-rs232/max232-board

Convert TTL(3-5V) level data from ST3679 to RS232 level(+/- 12V) suitable for connected to serial port of PC.