

# Radio Remote Control of Appliances

**Author: Li-Jun Lai**

**Graduate Student, Institute of Electrical Engineering,  
National Yunlin University of Science and Technology, Touliu, Yunlin 640, Taiwan**

## Introduction

Nowadays, many electrical appliances are equipped with controllers that allow the user to turn them on or off from a remote site. Among the existing control techniques, radio remote control plays an important role since this technique provides a reliable and flexible way to control appliances. Moreover, by combining this technique with the telephone line network or with the GSM network, one can control these appliances from a long distance, providing a means to send control commands from almost any place.

In this article, it will be shown how to use the Holtek HT95L100 MCU, to design an electrical appliance radio remote controller. In the designed circuit, the role of the HT95L100 MCU is to select a desired appliance and turn it on or off. In addition to the MCU, the main components of the designed circuit also include the RF transmitting and receiving modules, a Holtek encoder IC, the HT12E, and a Holtek decoder IC, the HT12D. For display convenience, seven LEDs are used, with each of them denoting which appliance is on or off. With a proper antenna, the transmission distance of the designed circuit is about 200~300 meters.

The HT95L100 has an operating voltage range of 2.4V~5.5V and an integrated 20×8 LCD driver internal circuitry giving it the ability to drive LCD panels directly. Since it has a 4K program ROM and 20 I/O pins, it is not difficult to interface the device with external components such as keypads. This device is capable of controlling external power components such as triacs or relays. To complement its microcontroller device range, Holtek has developed a development system known as the HT-IDE3000. The development system incorporates both software and hardware tools, such as real-time simulation, step, breakpoint and full trace features to ease the development and debug process.

## HT95L100 Specifications

The following are the major features of the HT95L100 MCU:

- Operating voltage: 2.4V~5.5V
- Operating frequency: External RC oscillator or Crystal
- Maximum of 20 I/O lines
- Dual system clock: 32.768kHz and 3.58MHz
- 4K×16 program memory ROM
- 1152×8 data memory RAM
- Up to 1.117μs instruction cycle with 3.58MHz system clock
- Internal LCD driver

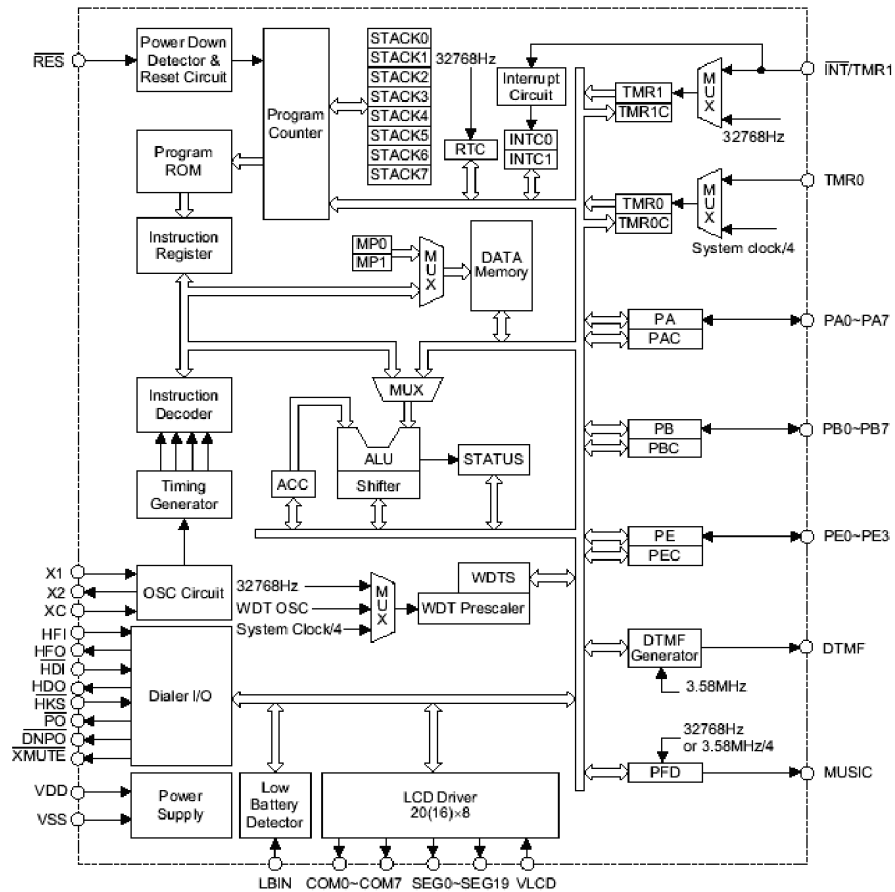


Figure 1. HT95L100 Block Diagram

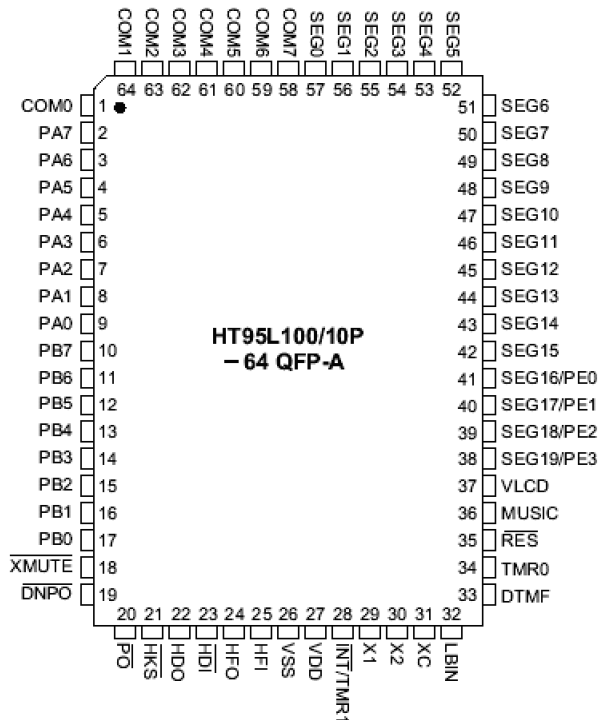


Figure 2. HT95L100 Pin-assignment

## Radio Remote Controller Implementation

### RF Modules

RF modules are used in this paper for data transmission and reception. The frequency used is 315MHz and the power source is a DC voltage of 5V. With a proper antenna, the transmission range can be up to 200~300 meters. The dimensions of the transmitting and receiving modules are shown in Figures 3 and 4, respectively.

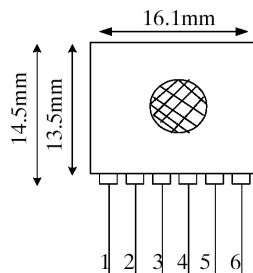
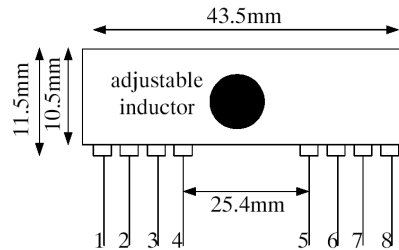


Figure 3. The RF Transmitting Module

- Pin 1: Positive power source
- Pin 2: Positive power source
- Pin 3: Ground
- Pin 4: Ground
- Pin 5: Output
- Pin 6: Input
- Input voltage: 1.5V~15V
- Power consumption: 8 mW

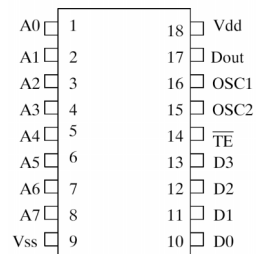


**Figure 4. The RF Receiving Module**

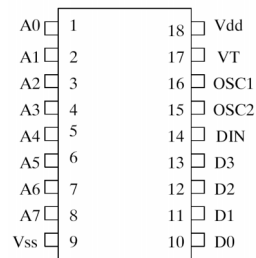
- Pin 1: Ground
- Pin 2: Digital output
- Pin 3: Analog output
- Pin 4: Positive power source
- Pin 5: Positive power source
- Pin 6: Ground
- Pin 7: Ground
- Pin 8: Antenna (30cm~35cm)
- Input voltage: 4.5V~5.5V
- Output: Digital and analog

### Encoding and Decoding ICs

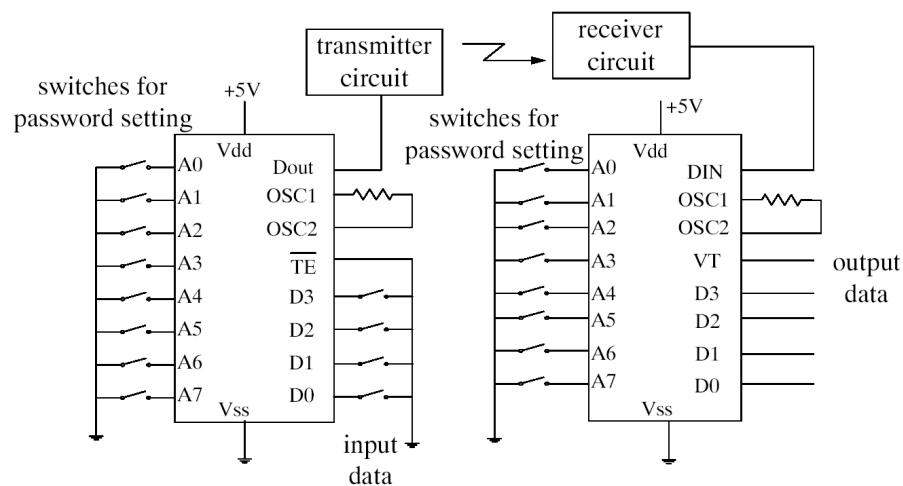
The encoding and decoding ICs used in this paper are the Holtek HT12E and the HT12D, respectively, the pin-assignments of which are shown in Figures 5 and 6. An encoding/decoding circuit using these two ICs is shown in Figure 7. The HT12E is a 12-bit encoder which has an 8-bits address line and a 4-bit data line. From Figures 5 and 6, one can find that both HT12D and HT12E have 8 address lines and 4 data lines.



**Figure 5. HT12E Pin-assignment**



**Figure 6. HT12D Pin-assignment**



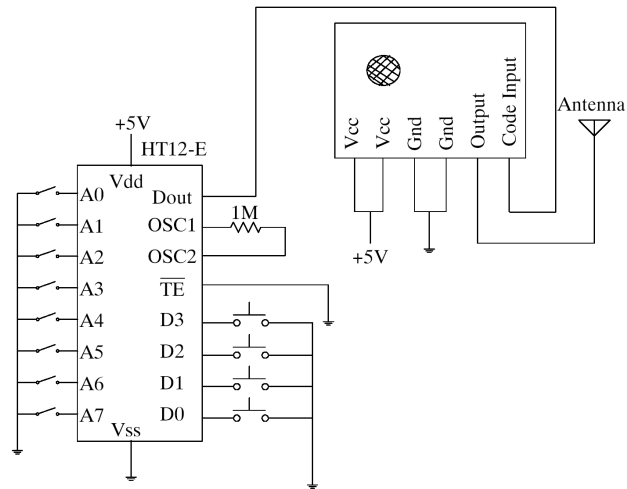
**Figure 7. An Encoding/Decoding Circuit Using HT12E and HT12D**

The following are the pin descriptions and major features of the HT12E and the HT12D.

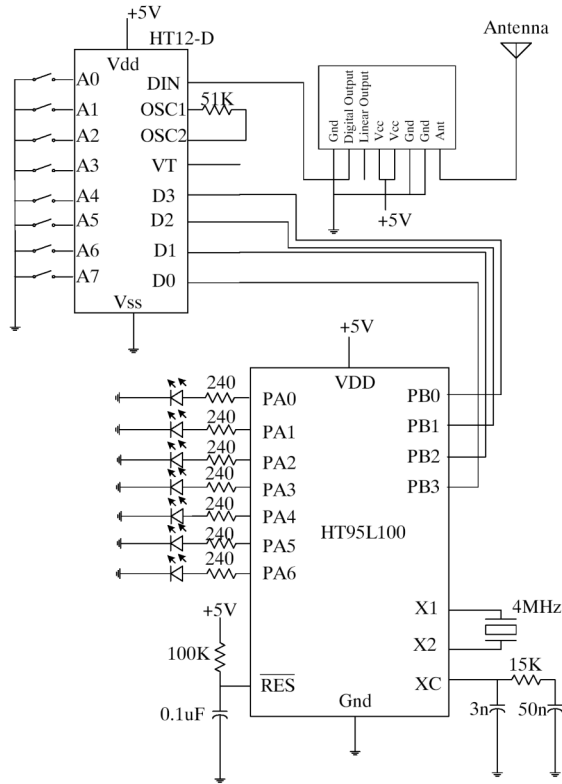
HT12E Pin Descriptions	
A0~A7	Address lines. A0 is the MSB and A7 is the LSB.
D0~D3	Data lines. D0 is the MSB and D3 is the .
OSC1	Oscillator input.
OSC2	Oscillator output.
TE	Transmission enable bit - transmission is active when low.
DOUT	Encoder serial data output.
VSS	Ground.
VDD	Positive power supply, operating voltage: 2.4V~12V.
HT12D Pin Descriptions	
A0~A7	Address lines. A0 is the MSB and A7 is the LSB
D0~D3	Data lines. D0 is the MSB and D3 is the LSB
OSC1	Oscillator input.
OSC2	Oscillator output.
VT	Transmission valid bit-high when transmission is active
VSS	Ground
VDD	Positive power supply, operating voltage: 2.4V~12V
Major Features of the HT12E and HT12D	
<ul style="list-style-type: none"> <li>• Operating voltage: 2.4V~12V</li> <li>• Low power and high noise immunity CMOS technology</li> <li>• Built-in oscillator, only one external resistor is needed</li> <li>• 8 address bits and 4 data bits</li> <li>• Received codes are checked 3 times for accurate transmission</li> <li>• Standby current 1<math>\mu</math>A</li> </ul>	

## The Hardware Circuit

Figures 8 and 9 show the transmitting circuit and the receiving circuit, respectively. In designing the circuit, the following tools, which include the Holtek TICE95L100 and HT-IDE3000, are used to test the feasibility of the designed circuit.



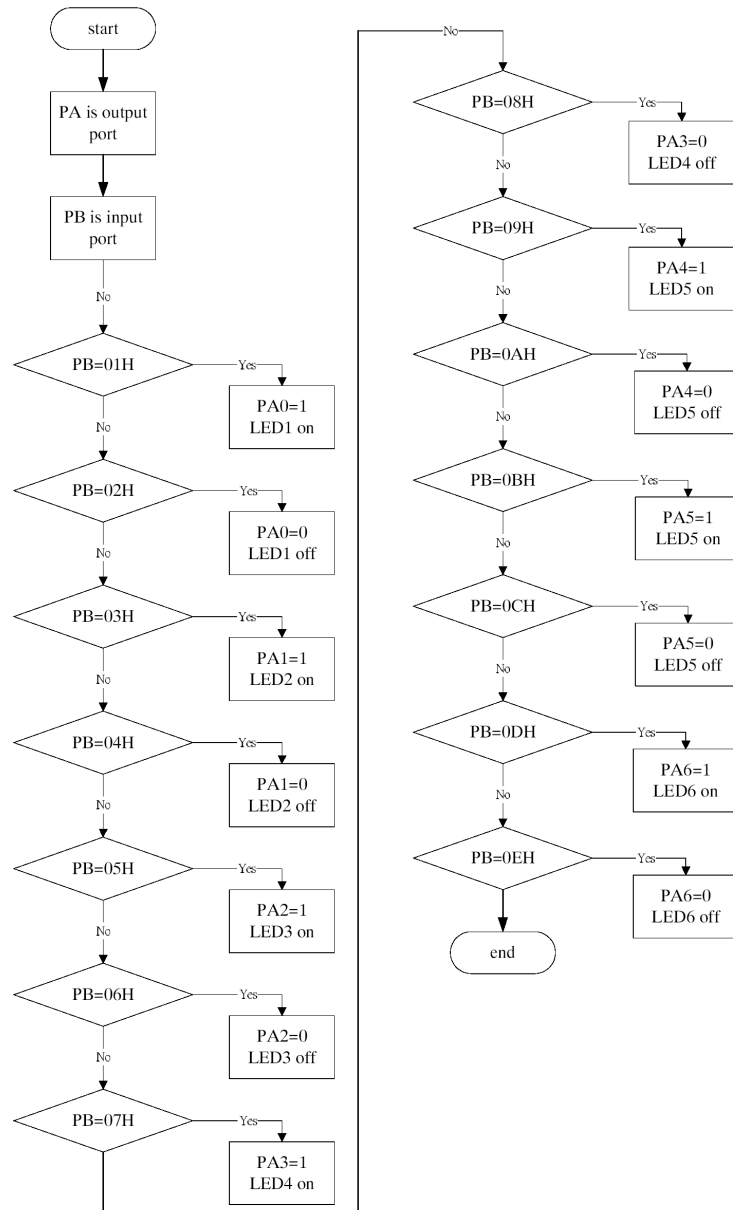
**Figure 8. Radio Remote Controller Transmitting Circuit**



**Figure 9. Radio Remote Controller Receiving Circuit**

## Software Development

The software flowchart is shown in Figure 10 for which the detailed description of the software can be found at the end of this paper.



**Figure 10. Software Flowchart**



## Conclusion

In this paper, the author showed how to use an HT95L100 MCU, together with a set of RF modules, an HT12E, and an HT12D to design a radio remote controller. From the hardware circuit, in addition to the MCU, the RF modules, and the encoding/decoding modules, one can find that very few external components are required. With a proper antenna, the transmission distance of the designed circuit is 200~300 meters. This distance is adequate for most of the application cases, however, if one is interested in increasing the transmission distance, it is suggested to combine the designed circuit with the GSM network.

## Appendix

### Source Codes

```
void main(void)
{
    unsigned char k      ;define variable k
    pokePAC(0x00)        ;set PA as an output port
    pokePA(0x00)          ;clear PA port
    pokePBC(0xff)         ;set PB as an input port
    k=peekPB()            ;store the value of PB into k
    if (k==0x01)
    {
        v1=1              ;flag to denote the first appliance is on
    }
    if (k==0x02)
    {
        v1=0              ;flag to denote the first appliance is off
    }
    if (k==0x03)
    {
        v2=1              ;flag to denote the second appliance is on
    }
    if (k==0x04)
    {
        v2=0              ;flag to denote the second appliance is off
    }
    if (k==0x05)
    {
        v3=1              ;flag to denote the third appliance is on
    }
    if (k==0x06)
    {
        v3=0              ;flag to denote the third appliance is off
    }
    if (k==0x07)
    {
        v4=1              ;flag to denote the fourth appliance is on
    }
    if (k==0x08)
    {
        v4=0              ;flag to denote the fourth appliance is off
    }
}
```

```
if (k==0x09)
{
    v5=1                ;flag to denote the fifth appliance is on
}
if (k==0x0a)
{
    v5=0                ;flag to denote the fifth appliance is off
}
if (k==0x0b)
{
    v6=1                ;flag to denote the sixth appliance is on
}
if (k==0x0c)
{
    v6=0                ;flag to denote the sixth appliance is off
}
if (k==0x0d)
{
    v7=1                ;flag to denote the seventh appliance is on
}
if (k==0x0e)
{
    v7=0                ;flag to denote the sixth appliance is off
}
if (v1==1)
{
    setPA0 ()           ;set PA0=1
}
else
{
    clrPA0 ()           ;clear PA0
}
if (v2==1)
{
    setPA1 ()           ;set PA1=1
}
else
{
    clrPA1 ()           ;clear PA1
}
if (v3==1)
{
    setPA2 ()           ;set PA2=1
}
else
{
    clrPA2 ()           ;clear PA2
}
if (v4==1)
{
    setPA3 ()           ;set PA3=1
}
else
{
    clrPA3 ()           ;clear PA3
}
}
```

```
        if (v5==1)
        {
            setPA4()          ;set PA4=1
        }
        else
        {
            clrPA4()          ;clear PA4
        }
        if (v6==1)
        {
            setPA5()          ;set PA5=1
        }
        else
        {
            clrPA5()          ;clear PA5
        }
        if (v7==1)
        {
            setPA6()          ;set PA6=1
        }
        else
        {
            clrPA6()          ;clear PA6
        }
    }
```