Design and Implementation of *DTMF Detector 2.0* Basic Extension Module for Embedded System Prototype

By:
Ivan Christian (e-mail: chris.ivan@gmail.com)
Daniel Widyanto (e-mail: kunilkuda@yahoo.com)

Introduction – ***Telephone units are ubiquitous. They can be found in almost every house. In some houses that has many rooms a line is paralleled so that anybody can receive and make call from different rooms. Their presence could be used for other purposes instead of making call per se. Application of this module can be found in smart home automatic system which incorporates telephone unit as input module. By using telephone unit somebody can make a control action to a remote system, as far as the telephone line can reach. To do so, such an interface between telephone line and the control system is needed.*** DTMF Detector 2.0 ***is designed for the purpose.***

1. Description

DTMF Detector 2.0 is a microcontroller system extension module which detects the presence of DTMF tones, and then decodes the tones to coded binary digits. By using *DTMF Decoder 2.0* design engineer could interface the analog signal of telephone line with digital logic of microcontroller system in straight ahead manner.

2. Module Specifications

The module specifications are as follows:

1. Input:
   - DTMF tones from telephone line (analog signal)
2. Output:
   - Decoded binary digits which correspond with DTMF tones.

3. Block Diagram

Figure 3.1 shows the block diagram of *DTMF Detector 2.0*. The module consists of decoupling block and DTMF Decoder block.

When someone presses a number, the phone unit encodes the number into two unique frequencies (DTMF) [3]. *DTMF Detector 2.0* taps the telephone line in order to catch the DTMF tone being sent.

Decoupling block interfaces the telephone line with the DTMF decoder. By using 1:1 ratio transformer, DTMF tone signals in telephone line would be decoupled electrically, but magnetically coupled, with input signals in the module.

DTMF decoder block does the DTMF detection and decodes it into unique binary digits. A specific IC CM8888 is used for decoding the DTMF tone. Actually CM8888 has other features instead, such as Call Progress (CP) detection and DTMF generator. In this module we only use the DTMF Receiver feature. CM8888 interface with microcontroller unit by 9 bit signals: 4 bits for I/O data and the rest 5 bits for control signals. To make it works, additional circuits is needed as mentioned in the datasheet. Refer to [1] for detail explanation of the circuit.

4. Tools

Primary tools used for design and implementation of *7-Segment-Board 1.0* are as follows:

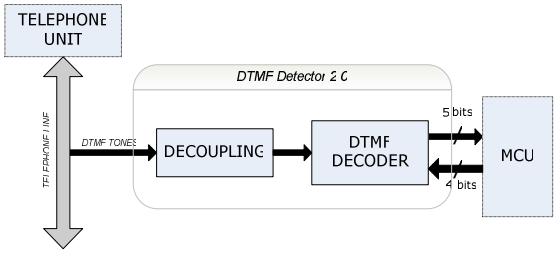

Figure3.1 Block diagram of *DTMF Detector 2.0*

1. Hardware
   - AVR ISP Development Board (designed by Erwin)
   - Atmel ATmega8535
2. Software
   - *Altium Design Explorer Version 7.2.85*
   - *WinAVR 20050214*

5. Schematic Capture

See schematic file in the attached zip file (`DTMF_detector.SchDoc`). We use single ended circuit configuration for the input stage of CM8888. One signal control, i.e. N_IRQ is placed separated from others to simplify cabling when we want to connect it to any pin of the microcontroller. The schematic is self explanatory.

6. PCB Layout Capture

Please refer to PCB file in the attached zip file for (`DTMF_detector_20060530.PcbDoc`). Note that physical board is implemented using 1-layer PCB. **Figure 6.1** is the picture of physical board. Pin assignments for interfacing with MCU are shown in ***Figure 6.2***. **Table 6.1** gives us list of module components.

7. Firmware Design

This project use Atmel AVR family microcontroller for driver implementation. No particular reason but tools availability that makes the decision. The module driver, written in C, consists of 4 separate files to simplify the maintenance: `main.c`, `port.h`, `cm8888_drv.c`, and `cm8888_drv.h`.

Figure 6.1 Physical Board of *DTMF Decoder2.0*

D1	D3	N_RD	RS0	GND
D0	D2	N_WR	N_CS	VCC

Legend:

D3 .. D0 = decoder I/O signals

N_RD = Read register signal contro

N_WR = Write register signal control

N_CS = chip select signal control

RS0 = Register select signal contro

Figure 6.2 Pin configuration of I/O Port

Table 6.1 List of Primary Components

Component Name	Units	Notes
RJ11 Socket	2	Input sockets
CM8888	1	DTMF Decoder
XTAL 3.579545 MHz	1	
1:1 Transformer	1	Decoupling
Header 5x2	2	Output
C 100nF Ceramic	4	Decoupling C
R100K 1/4W	2	Input Voltage Divider
R30K 1/4W	1	Pull up R for N_IRQ
R360K 1/4W	3	Steering Circuit

The `main.c` file is the main program that is used to test and verify the driver. The `port.h` and `cm8888_drv.h` files are self-explanatory. The `cm8888_drv.c` contents 3 functions, that is:

1. `void init_DTMF(void)`
   This function initiates data direction register (DDRx) and Data Register (PORTx) of AVR ports used. After that, the program does power up initialization by writing 0000b to all control registers, and then it set up the operating mode of CM8888. The procedure of power up initialization is described clearly at [2].
2. `unsigned char read_stat(void)`
   This function read the status register from CM8888. It is provided for debugging purpose.
3. `unsigned char scan_DTMF(void)`
   This function is responsible to scan DTMF tone received. When the microcontroller unit receives logic low from N_IRQ control signal, it's time to execute this function. The result of DTMF detection is converted into ASCII format, including the button '*' and '#' at the telephone unit keypad.

8. Testing and Verification

To do testing and verification, a simple program was made in `main.c` file. A LED board is used in PORTA as debugging tools. Program first initializes module and supporting peripheral. ***Figure 8.1*** shows the module under test.

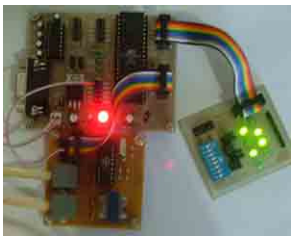

Figure 8.1 *DTMF Detector 2.0* Under Test

Test result shows that all the DTMF tones received can be decoded into their unique ASCII number (**Table 8.1**). In ***Figure 8.1*** the LED board displayed 0x35, which matched with the number '5' of the telephone unit keypad (MSB is on the bottom).

Table 8.1 Test Result

Number	Output (hex)	Number	Output (hex)
1	0x31	7	0x37
2	0x32	8	0x38
3	0x33	9	0x39
4	0x34	*	0x2A
5	0x35	0	0x30
6	0x36	#	0x23

9. Portability

The source can be translated to other MCU (8051, PIC) with minor modification.

10. Conclusion

We conclude that both hardware and firmware of *DTMF Detector 2.0* is designed successfully.

11. Advices

- To improve *DTMF Detector 2.0*, it is recommended to redo the layout, adding more free space between nearest component. The components placement seems too cozy.
- For more cost effective implementation, CM8870 can be used instead with minor modification in hardware and software.
- To reduce unnecessary output header, N_CS can be tied to GND and N_IRQ is included in one header.

References

[1] CM8888 CMOS Integrated DTMF Transceiver – datasheet. California Micro Device.2000

[2] MT8888 Integrated DTMF Transceiver with Intel Micro Interface– datasheet. Mitel Semiconductor. March 2001

[3] ATmega8535 8-bit AVR Microcontroller with 8 Kbytes In-System Programmable Flash – Preliminary datasheet. Atmel Corporation. June 2004