

Morse Code CW Identifier

Entry #A3604

Introduction

Federal Communications Commission (FCC) rules require the periodic identification of Amateur Radio Transmissions. It is easy to comply with this rule when chatting with a friend, but what about when you are experimenting with the development of transmitter circuits? Time flies while you are chasing bugs, and before you know it, it's past time to identify! My contest entry of an automated CW identifier allows you to concentrate on your electronic circuits while an ATmega8L CPU takes care of required transmission identifications.

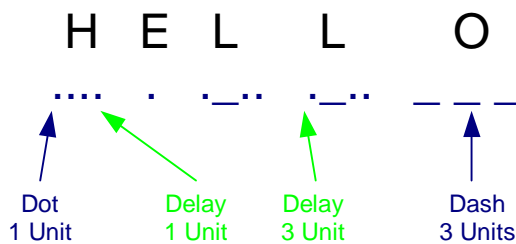
Support of four independent messages allows the CW Identifier to additionally be used as a radio contest memory keyer.

About Morse Code

International Morse Code has been in use since the first Spark Gap transmitter bridged the aether to introduce the world to wireless communications. Morse Code is the simplest form of radio communications, sending intelligence by a coded presence or absence of a signal. (Think binary 1 and 0.) Until recently, knowledge of Morse Code was required for a license to operate high frequency bands. Recent changes to international and federal law allow for licensing without proficiency in Morse Code. In spite of changes in requirements, Morse Code remains a popular and efficient method of communications. Morse Code is recognized as a universal method of identifying a transmission.

Morse Code is the On-Off keying of a transmitter's signal. Morse Code is composed of short (Dot) and long (Dash) transmissions separated by delays. Shorter or longer basic timing can increase or decrease the throughput of a message (commonly expressed as words-per-minute).

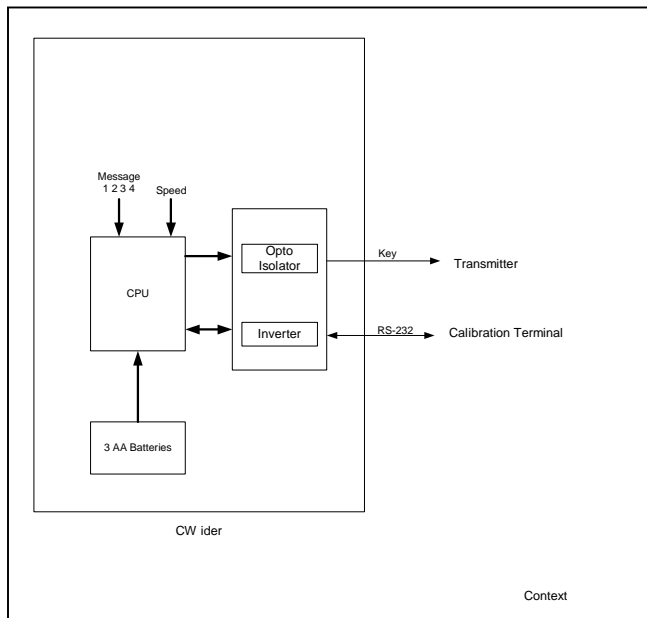
Typical timing for a message is shown below:



By combining these delays, any textual message can be sent and received with basic radio equipment.

Theory of Operation

The Morse Code CW Identifier is a stand alone controller used for sending Morse Code via a Ham Radio transceiver. A block diagram for the system is shown below.



CW Ider System Diagram

Sending CW

The function *SendCharacter()* (shown below) acts as the send loop. It encodes and sends the ASCII characters. The Morse Code timing presented earlier is used in the encoding.

```
void SendCharacter(unsigned char symbol)
{
    char mask[8] = {0x80, 0x40, 0x20, 0x10, 0x08, 0x04, 0x02, 0x01};

    for (i=0; i<MORSE_SYMBOL_LENGTH; i++) {
        if ( MorseCode[symbol].Dot & mask[i]) {
            SendDot();
            DelayDot();

        } else if ( MorseCode[symbol].Dash & mask[i]) {
            SendDash();
            DelayDot();

        } else {
        }

    }

    DelayCharacter();           //Wait one character time after sending symbol
}
```

Main Loop

The command loop (see *main()* below) initializes the system and then continuously polls the message select buttons every 100ms to determine if it is time to transmit a message.

```
main(void)
{
    init();

    HoldTimer = Calibrations.Interval;
    for (;;) {                                /* loop forever */

        Menu_Print();

        Message = MessageInput();
        if ( Message != LastMessage) { //New Button Press
            SendMessage(Message);
            LastMessage = Message;
            HoldTimer = Calibrations.Interval;

        } else {                               //Button is being held down
            Delay_1ms(100);
            if (HoldTimer) {                   //Delay for resend of message
                HoldTimer--;

            } else {
                SendMessage(Message);
                HoldTimer = Calibrations.Interval;
            }

        }

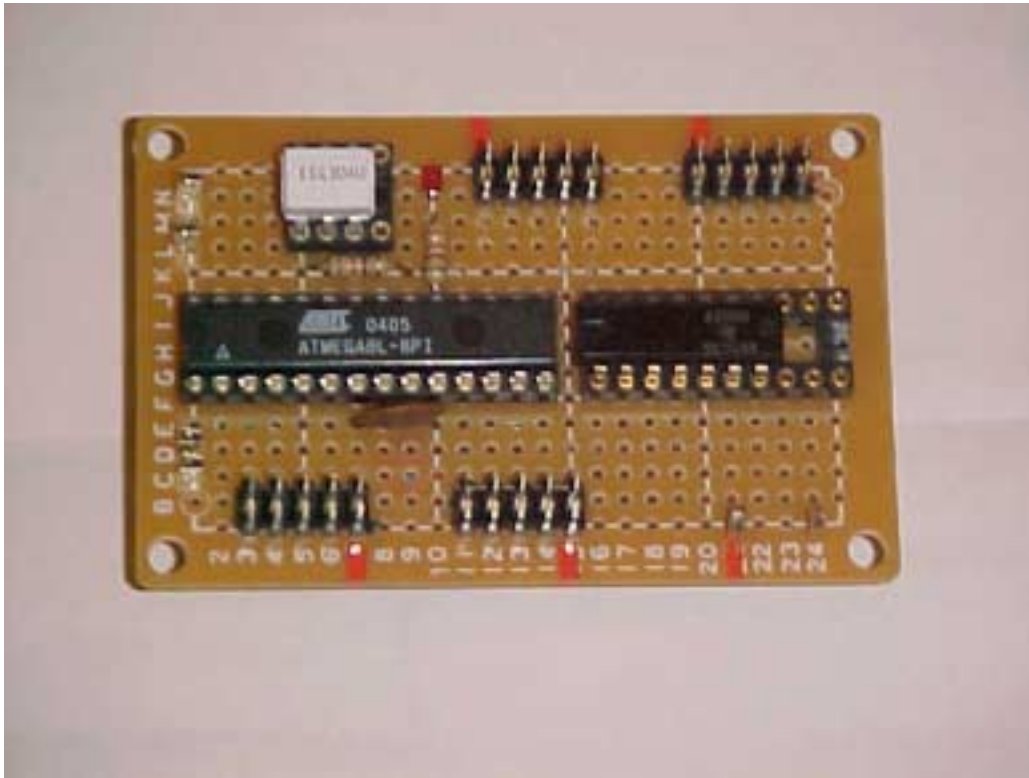
    }
}
```

At the start of every processing loop, *Menu_Print()* gives the user the opportunity to change calibration information via the serial port.

User Controls

The user can control the CW Identifier by a serial port, 4 buttons, and a variable resistor. Four Push Buttons initiate the transmission of the selected message. Each time a button is pressed and released, the selected message is transmitted. If a button is held down, the selected message will be sent and then repeated as programmed. A variable resistor adjusts the CW transmission speed (words per minute).

The Hardware



CW Ider Circuit Board

Overview

The CW Ider hardware consists of five major components: a set of user interface switches, a transmitter keying circuit, a serial calibration interface, an Atmel ATmega8L CPU, and a battery powered power supply. Refer to the schematic appendix for complete circuit information.

User Interface

Four push button switches select four individual transmission messages. A press of a button will transmit a message once.

Keying Circuits

The CW ider has a combination of three concurrent outputs: an opto-isolator, an LED, and a speaker. The optical isolator provides an isolated switch closure to key the radio transmitter. An LED blinks to provide a visual indication that the circuit is operating. A piezo speaker provides audible feedback of the system's operation.

Serial Interface

The serial interface consists of two inverter gates to match the RS-232 signal polarity. The signal levels DO NOT comply with RS-232 specifications. I have used this inverter approach on several projects without problems. True RS-232 levels could be achieved by using a MAX218 level shifter.

CPU

The Central Processing Unit is the Atmel ATmega8L using the internal oscillator running at 2 Mhz. The CW Ider circuitry is based on the low power family of the AVR microcontroller to allow for battery powered operation

