A Keypad Interface for your FT-991A rig

Contents

Description

I created this project as a result of having to spin my rig's VFO dial *a lot* to jump from spot to spot when chasing parks, countries, or other QSO's during contests. I had been using an app (ComCAT) to do this for some spots, but it did not pick up all spots, and there was sometimes a time lag between the spot and it appearing in the app. Also, my logging program, while capable of changing my rig's frequency from the PC, was not showing POTA (Parks on The Air) contacts in the spotting table. As a result, I thought that it might be faster to just key in the frequency directly from a keypad. As it turned out, it was!

The project I will be describing offers the capability of punching in a 4 to 6-digit frequency and passing that, in the correct format, to the rigs that I use. For example, keying in "7235#" on the keypad will send a request to change the frequency of my rig to 7.235 MHz and set the mode to LSB. Keying in "146520#" will send a request to change the frequency to 146.520 MHz and set the mode to FM.

Another capability is to repeat the previous frequency. That comes in handy if I need to retune the rig since I always spin the dial 2 to 3 KHz lower than the frequency I want to transmit on in order to tune. You would never tune up on an active frequency, would you? Keying in "#" repeats that previous frequency.

If a mistake is made on the entry, the "*" (asterisk) key will clear out that entry.

At the time of writing this manual, consideration is being given to providing 1 key access to tuning the rig's VFO 3KHz down with one key. This would allow tuning up off of the desired contact's frequency.

In the code, you will note that the 2 rigs I have need slightly different entries and a Define parameter sets this up at compile time. If you have a different type of rig, go ahead and add in the required changes. Successful changes can be sent to me for publication, should you desire.

Hardware Details

The core of this design is an Atmel ATMega328 processor and associated circuitry, Y1, R1, C1, C2 and C8. You will quickly recognize this as a stripped-down Arduino. This allows you to program the code on an Arduino UNO and then move the processor to this project. Alternatively, you can just use an Arduino UNO and wire up that additional circuitry to it.

The core processor communicates via its built-in serial port to U2, a MAX232 voltage convertor. The MAX232 translates the unipolar 5 Volt levels to and from the processor to the bipolar RS232 levels required for standard serial communication. Those voltages are typically +/- 10V to +/- 15V.

Note that a FT-450D rig has additional RS232 needs, specifically the use of RTS (Request To Send) and CTS (Clear To Send) which requires that the J1 port have a short between these 2 pins on the connector; pins 7 and 8.

You will note that there are 2 RS232 interfaces. This is available for rigs that have multiple connections to their RS232 port. In my case, I use the RS232 port on my FT991A to communicate frequency and mode to my logging software on my shack's PC. U4 "mixes" the 2 logic level signals to permit both my PC software and the Keypad to communicate with my rig. On the logic level side of the MAX232 (pins 9, 10, 11 and 12), a logic level "1" is the normal level seen when there is no communication in process, so U4 will accept that and then transfer and logic "0" seen from either the keypad or the other serial device to the rig.

If there is no need for a second port to be used this way, a "Bypass" connection is available. Place a jumper at position J9 and then U4 and J2 are not needed.

In addition, I have noticed that there are some ports that may not be set up to match J2's standard connection, so a jumper pair is available (J6 and J7) that provides the ability to reverse the TXD and RXD connections on J2. Figure 1 shows how to do this.

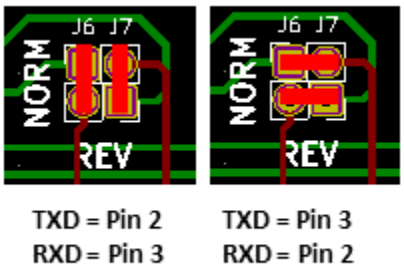

Figure 1 Jumper Settings

U3 is a 6-digit seven-segment LED display that interfaces to the core processor via 3 pins (DIN, CS and CLK). This is a standard Arduino accessory and available from many sources. Not that on the schematic, U3 and J5 are wired pretty much in parallel. The DIN connection, however, is only to J5. This was done so that the display could be mounted on the back side of the PCB and the J5 connector is thus in the proper position. If you are wiring this without the PCB, then wire the DIN pin of U3 to the processor U1 pin 18. The Arduino "LEDControl" library provides the software interface between the processor and the display.

LED D2 is an optional heartbeat indicator. It will flash once per second to indicate that the software is running.

J4 is a header connector that connects a 3x4 matrix keypad to the processor. U1 will scan the rows and columns to determine which key was pressed. Using the Arduino "Keypad" library, only one keypress at a time will be decoded. Debounce of the keys is provided by that library as well. I used a keypad from an old phone. A source is also provided in Appendix 3.

Lastly, the D1, U5 combination will convert the shack's 12V power to the 5V required by the rest of the circuitry. U5 will likely need a heatsink since the LED display draws a fair bit of current in some cases.

Photos of the prototype and finished units are shown in Figures 2, 3 and 4.

If you want build your own Keypad using the PCB, I have included a link to the Gerber Zip file in Appendix 3. Note: your browser may balk at downloading a Zip file. If this concerns you, contact me.

You can order PCB's using this file from just about any PCB house. I use JLPCB in China and OshPark in the USA for PCB's.

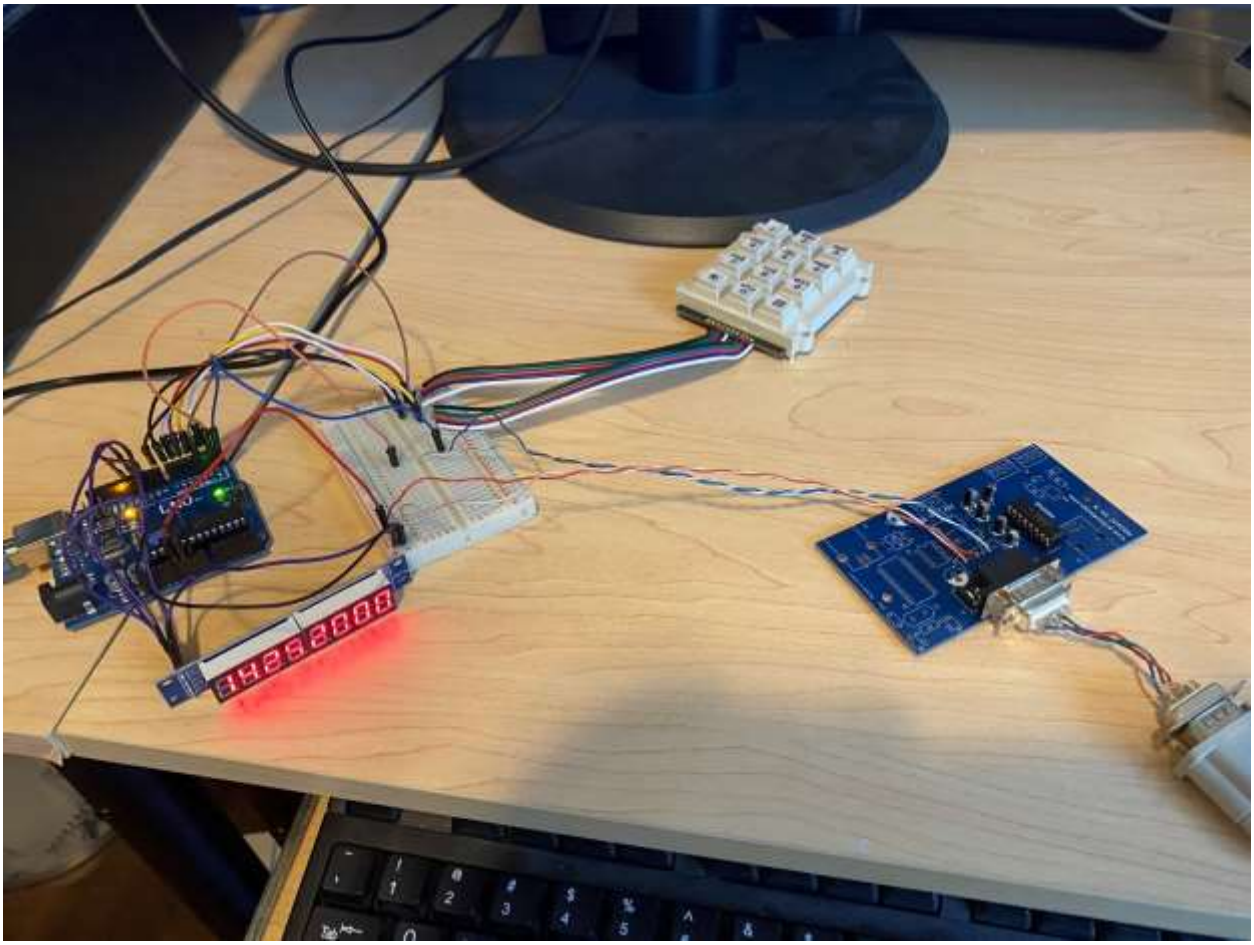

Figure 2 The Prototype

Figure 3 The Finished Version "A" Unit

Figure 4 Working Final Unit in a 3D Printed Case

Software Details

I would like to acknowledge Alexander Brevig (alexanderbrevig@gmail.com) for his work in developing the keypad code that I modified for this project. That library is included in the project's code.

Also included in this code is the LEDControl library by Eberhard Fahle.

Like all Arduino software, the code begins by setting up global parameters and initializes the LEDControl library with settings for the pins used to communicate with the LED driver (MAX7219).

After this, the "setup" function sets up the serial port, and LED Display by displaying the software version number on both. Change the baud rate and other serial port settings in the "Serial.begin (xxx)" statement. My rig uses 38,400 bps, 8 bits, no parity and 2 stop bits (8N2). The once per second timer interrupt is then set up.

The next function is the Interrupt service routine. All it does is turn on or off the optional heartbeat LED.

The next function is the main Loop. This section of code waits for key presses to occur and, if detected, to act on them. Note that there are two different reactions depending on which rig is being communicated with (and defined in the global parameter section of code). The main difference is that the length of the command being sent to the rig is 1 digit longer for the FT-991A than for the FT-450D. This is due to the lack of VHF/UHF frequencies on the FT-450D. As a result, the FT-450D only needs to worry about LSB or USB modes, whereas the FT-991A also needs to determine if the FM mode is needed.

Special attention is paid to the "#" key. If it is received twice in a row, then the previous command sequence is re-sent to the rig.

Conclusion

If you wish to duplicate the case shown, send an email to me (“I am good on QRZ”) and I will provide you with the STL file.

I hope that this project has given you some ideas to work on a similar project for your shack.

73

Mitch NØDIM

Appendix 1 – Schematic

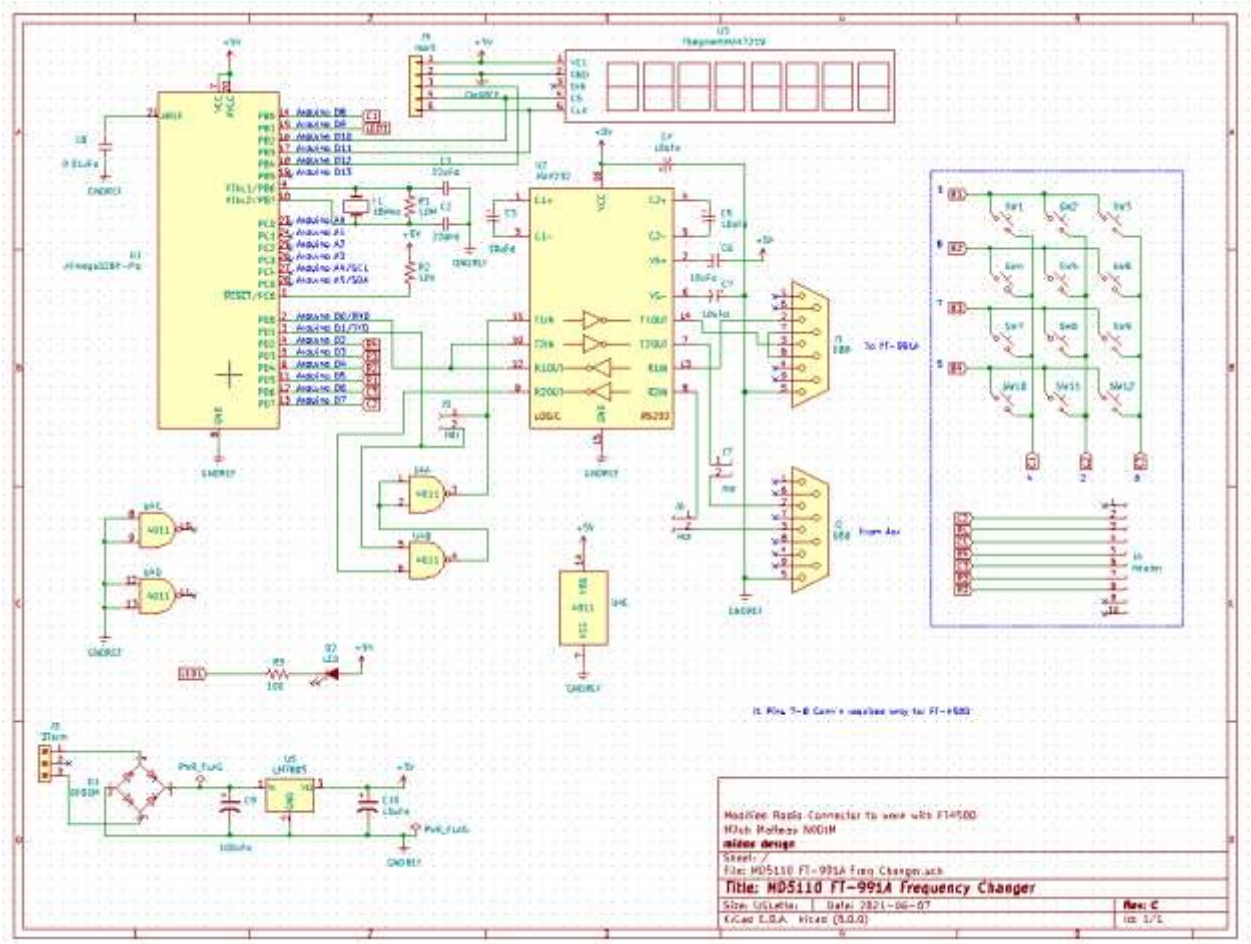

Figure 5 MD5110 Display Schematic

Appendix 2 – Bill of Material

Designator	Quantity	Designation
C1, C2	2	22pFd disk capacitor
C3, C4, C5, C6, C7, C10	6	10uFd 16V electrolytic capacitor
C8	1	0.01uFd disk capacitor
C9	1	100uFd 16V electrolytic capacitor
D1	1	DF01M Full Wave Bridge
D2	1	LED 1206 Package
J1, J2	2	DB9 Female PCB Mount
J3	1	3 Terminal Block
J4	1	10 Pin Header connector
J5	1	5 pin Header connector
J6, J7, J9	3	2 pin Header connector
R1	1	10M ¼ W resistor
R2	1	10K ¼ W resistor
R3	1	100 ¼ W resistor
U1	1	ATmega328P-PU
U2	1	MAX232
U3	1	7SegmentMAX7219
U4	1	4011
U5	1	LM7805
Y1	1	16MHz

Appendix 3 - Sources

1. 6-digit seven Segment Display: https://www.ebay.com/itm/LED-Display-Module-MAX7219-8-Digit-7-3-IO-Ports-Segment-for-Arduino-Raspberry-Pi/353071454298?_trkparms=aid%3D1110006%26algo%3DHOMESPLICE.SIM%26ao%3D1%26asc%3D227685%26meid%3Da55a949ed86948d6821968735c527496%26pid%3D100005%26rk%3D4%26rkt%3D12%26mehot%3Dco%26sd%3D353180339935%26itm%3D353071454298%26pmt%3D1%26noa%3D0%26pg%3D2047675%26algv%3DSimplAMLv5PairwiseWebWithDarwoV3BBEV2b%26brand%3DAdvance&_trksid=p2047675.c100005.m1851
2. ATMega328 Processor with Arduino Bootloader: https://www.jameco.com/z/A000048-Arduino-ATMega328P-32kB-Microcontroller-with-Arduino-Uno-Optiboot-Bootloader_2129334.html
3. PCB Gerbers for this project: http://n0dim.com/Documents/MD5110 FT-991A Freq Changer-VC.zip
4. PCB manufacturers: https://jlcpcb.com/ and https://oshpark.com/
5. Original Keypad library and code: https://www.arduinolibraries.info/libraries/keypad
6. LED Control library: https://www.arduinolibraries.info/libraries/led-control
7. Matrix Keypad: https://www.adafruit.com/product/3845 for example

Appendix 4 – Arduino Code

```
/* @file MD5110FT-991AFreqChanger
|| @version 1.2
|| @original author Alexander Brevig
|| @contact alexanderbrevig@gmail.com
|| @ Revision 20-07-31 by M.Matteau N0DIM
||  Get input from Matrix Keypad, and display on serial port AND on 7 Segment MAX7219 display
|| #Revision 20-09-16 by M.Matteau N0DIM
||  Added repeat last frequency by pressing # key again.
||  Added interrupt driven LED 1pps blink
*/
#include "LedControl.h"
//#define DEVMODE 1          // === Comment this out for production version
#define RIG_991 1
//#define RIG_450 1
/*
 Now we need a LedControl to work with.
 ***** These pin numbers will probably not work with your hardware *****
 pin 12 is connected to the DataIn
 pin 11 is connected to the CLK
 pin 10 is connected to LOAD
 We have only a single MAX72XX.
 */
LedControl lc=LedControl(12,11,10,1);  //(MOSI, CLK, CS, #Devices)

/* we always wait a bit between updates of the display */
unsigned long delaytime=250;


#include <Keypad.h>



const byte ROWS = 4; //four rows

const byte COLS = 3; //three columns

char keys[ROWS][COLS] = {

  {'1','2','3'},

  {'4','5','6'},

  {'7','8','9'},

  {'*','0','#'}

};

String KeysReceived ="";

String KeysReceivedCopy = "";

byte rowPins[ROWS] = {5, 4, 3, 2}; //connect to the row pinouts of the keypad

byte colPins[COLS] = {8, 7, 6}; //connect to the column pinouts of the keypad

boolean toggle1 = 0;


int Digit = 7;    // which digit will receive the next character (scroll it from left to right)

// use     lc.setDigit(0,Digit,Character,false);

// OR     lc.setChar(0,Digit,Character,false);

int ModeFreq = 0;
```

```
String SSB;
String ModeType;
Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

void setup(){
  // Setup Serial port first
  Serial.begin(38400,SERIAL_8N2);
#if defined(DEVMODE)
  Serial.println(F("Sketch: MD5110FT-991AFreqChanger v1.2"));
#endif
//=============================================================================================
//     The MAX72XX is in power-saving mode on startup, we have to do a wakeup call
//=============================================================================================

  lc.shutdown(0,false);
  /* Set the brightness to a medium values */
  lc.setIntensity(0,8);
  /* and clear the display */
  lc.clearDisplay(0);

  lc.setChar(0,7,'P',false);    //display 'P1.2 5110'
  lc.setChar(0,6,'1',true);     //firmware ID
  lc.setChar(0,5,'2',false);
  lc.setChar(0,3,'5',false);
  lc.setChar(0,2,'1',false);
  lc.setChar(0,1,'1',false);
  lc.setChar(0,0,'0',false);

  delay(5000);
  lc.clearDisplay(0);
//=============================================================================================
//  Setup Timer Interrupt
//=============================================================================================

//set pin as outputs
  pinMode(9, OUTPUT);   // Heartbeat LED
```

```
cli();//stop interrupts


//set timer1 interrupt at 1Hz
  TCCR1A = 0;// set entire TCCR1A register to 0
  TCCR1B = 0;// same for TCCR1B
  TCNT1  = 0;//initialize counter value to 0
  // set compare match register for 1hz increments
  OCR1A = 15624;// = (16*10^6) / (1*1024) - 1 (must be <65536)
  // turn on CTC mode
  TCCR1B |= (1 << WGM12);
  // Set CS12 and CS10 bits for 1024 prescaler
  TCCR1B |= (1 << CS12) | (1 << CS10);
  // enable timer compare interrupt
  TIMSK1 |= (1 << OCIE1A);



sei();//allow interrupts


}       //end setup


//================================================================================


ISR(TIMER1_COMPA_vect){//timer1 interrupt 1Hz toggles pin 13 (LED)
//generates pulse wave of frequency 1Hz/2 = 0.5kHz (takes two cycles for full wave- toggle high
then toggle low)
  if (toggle1){
    digitalWrite(9,HIGH);   //Turn off Heartbeat LED
    toggle1 = 0;
  }
  else{
    digitalWrite(9,LOW);   //Turn on Heartbeat LED
    toggle1 = 1;
  }
}
//================================================================================
```

```
void loop(){
  char key = keypad.getKey();

  if (key){
    if (Digit ==7) {
        lc.clearDisplay(0);
    }
    #if defined(DEVMODE)
        Serial.print(F("Key Received: "));
        Serial.println(key);
        Serial.print(F("Character Count: "));
        Serial.println(Digit);
    #endif

    if (key ==char(42)) {         // was it "*"?
        lc.clearDisplay(0);
        Digit = 7;
        KeysReceived = "";
    }
    else if (key ==char(35)) {         // was it "#"?

        #if defined(DEVMODE)
            Serial.print(F("String Received: "));          // just for debugging
            Serial.println(KeysReceived);                  // just for debugging
        #endif

        if (KeysReceived == "") {
            KeysReceived = KeysReceivedCopy;
        }
        else {
        KeysReceived = KeysReceived +"000";     // Adjust input to KHz
        KeysReceivedCopy = KeysReceived;
        }
#if defined(RIG_991)

//--------------------------------------------------------------------------------------------
```

```
// for FT-991A
//-------------------------------------------------------------------------------------------------
        switch(KeysReceived.length()) {
          case 5:
              KeysReceived = "0000" + KeysReceived;
              break;
          case 6:
              KeysReceived = "000" + KeysReceived;
              break;
          case 7:
              KeysReceived = "00" + KeysReceived;
              break;
          case 8:
              KeysReceived = "0" + KeysReceived;
              break;
        }

        if (KeysReceived.toInt() < 10000000) {
            SSB = "LSB";
            ModeType = "MD01;";
        }
        else if (KeysReceived.toInt() < 140000000){
            SSB = "USB";
            ModeType = "MD02;";
        }
        else {
            SSB = "FM";
            ModeType = "MD04;";
        }

#endif
//-------------------------------------------------------------------------------------------------
#if defined(RIG_450)

//-------------------------------------------------------------------------------------------------
// for FT-450D
```

```
//-------------------------------------------------------------------------------------------------

        switch(KeysReceived.length()) {

          case 4:

              KeysReceived = "0000" + KeysReceived;

              break;

          case 5:

              KeysReceived = "000" + KeysReceived;

              break;

          case 6:

              KeysReceived = "00" + KeysReceived;

              break;

          case 7:

              KeysReceived = "0" + KeysReceived;

              break;

        }



        if (KeysReceived.toInt() < 10000000) {

            SSB = "LSB";

            ModeType = "MD01;";

        }

        else {

            SSB = "USB";

            ModeType = "MD02;";

        }


 #endif

//-------------------------------------------------------------------------------------------------




//        KeysReceived = "FA" + KeysReceived + ";FB" + KeysReceived + ";" + ModeType;

        KeysReceived = "FA" + KeysReceived + ";" + ModeType;      // Set VFOA and Mode to USB,
LSB, or FM depending on what frequency was selected

//        KeysReceived = "FA" + KeysReceived + ";" ;    // Set VFOA


        #if defined(DEVMODE)
```

```
            Serial.print(F("For Transmission: "));
        #endif

        for (int i = 0; i <KeysReceived.length();i++) {
          Serial.write(KeysReceived.charAt(i));
        }

        #if defined(DEVMODE)
            Serial.println(" ");                    // just for debugging
            Serial.print(F("Raw Data: "));          // just for debugging
            Serial.println(KeysReceived);           // just for debugging
            Serial.print(F("Mode: "));              // just for debugging
            Serial.println(SSB);                    // just for debugging
        #endif

        lc.clearDisplay(0);

        if(KeysReceived.charAt(2)!='0') {
              lc.setChar(0,7,KeysReceived.charAt(2),false);
        }

        #if defined(DEVMODE)
              Serial.print(F("First Character = "));          // just for debugging
              Serial.println(KeysReceived.charAt(2));          // just for debugging
        #endif

        lc.setChar(0,6,KeysReceived.charAt(3),false);
#if defined(RIG_991)
        lc.setChar(0,5,KeysReceived.charAt(4),true);
#else
        lc.setChar(0,5,KeysReceived.charAt(4),false);
#endif
        lc.setChar(0,4,KeysReceived.charAt(5),false);
        lc.setChar(0,3,KeysReceived.charAt(6),false);
#if defined(RIG_991)
        lc.setChar(0,2,KeysReceived.charAt(7),true);
```

```
#else
        lc.setChar(0,2,KeysReceived.charAt(7),false);
#endif
        lc.setChar(0,1,KeysReceived.charAt(8),false);
        lc.setChar(0,0,KeysReceived.charAt(9),false);

        KeysReceived = "";
        Digit = 7;
    }
    else {
        KeysReceived = KeysReceived + key;
        lc.setChar(0,Digit,key,false);
        Digit--;
    }
    if (Digit ==-1) {
        Digit = 7;
    }

  }
}
```