## Electronic Design.

By RICARDO JIMENEZ, Imperial Valley College, Imperial, Calif., and GABRIEL LEE **ÁLVAREZ**, Instituto Tecnológico de Mexicali, B.C. México.

## Microcontroller Sends Voltage and Frequency via Low-Cost Modules

In this Idea for Design, voltage and frequency can be sent wirelessly using PIC microcontrollers.

This project pairs a set of 8-bit microcontrollers with a 433-MHz industrial, scientific and medical (ISM) band transmitter/receiver. The FS1000A RF transmitter has a range of up to 200 m; the XY-MK-5V RF

receiver operates at 5 V and uses only 4 mA (*Fig. 1*). These modules are readily available and used in projects with platforms like Arduinos.

The projects uses <u>Microchip</u> PIC microcontrollers including the <u>16F1619</u> and <u>16F1614</u>. These utilize the radio modules employing the on-chip EUSART (Enhanced Universal Synchronous/Asynchronous Receiver Transmitter) interfaces. On the transmitter side, we use the PIC's analog and digital interfaces to read a voltage and a frequency source within the range of 0 to 4.99 V dc and 0 to 65.5 kHz, respectively.

The logic diagram for the transmitter (*Fig. 2*) includes an LCD display to provide feedback. *Listing 1* has the code for the 16F1619.

The receiver side (*Fig. 3*) also includes a pair of LCD displays. The displays aren't needed in an application, but they're



1. The FS1000A RF transmitter (right) has a range of up to 200 m. The XY-MK-5V RF (left) receiver operates at 5 V and uses only 4 mA.

## Transmission in ASCII Code for Letter "A"

Start Bit	LSB Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	MSB Bit 7	Stop Bit
0	1	0	0	0	0	0	1	0	1

handy for debugging. *Listing 2* has the code for the 16F1614. Serial Communication

First, we will touch on the serial communication support that's tied to the wireless modules. To establish communication, it's necessary to have a starting bit for a period of time to alert the receiver that a data package is about to be transmitted. This forces the receiver clock to start synchronization with a 0 bit. Then each bit is sent individually, starting with the LSB bit through the MSB bit (*see table*). Each bit has the same period. Once all bits are transmitted, it must wait for the Stop bit to indicate end of transmission. This is achieved with a High logic, where the communication ends.

To transmit the alphanumeric character "A," whose ASCII code is 0b01000001 (*Fig. 4*), the bits are organized as shown in the table. Each bit has a period determined by the transmission speed (baud rate), which can vary from 115.2 kb/s to 200 bits/s. The time for each bit is given by:

$$T = \frac{1}{Baud Rate}$$

To transmit a voltage, we will read it using the 10-bit analogto-digital converter (ADC), which by default has its reference voltage connected to 5 V. This defines the ADC resolution:

$$V_{bit} = \frac{V_{ref+}*(bit \ read)}{2^{10}} = \frac{5v*1 \ bit}{1024} = 4.882 mV$$

To perform the binary to decimal conversion, we use this code:



 $\label{eq:VINBCD} \mbox{VINBCD} = \mbox{VIN}^{\star}4887 \quad ; \mbox{ multiplying by RE-} \\ \mbox{Slsb} = 4.8887 \quad . \label{eq:VINBCD}$ 

VR = div32 1000 ; perform 16-bit division

where VIN is the binary voltage read by the ADC, and VR is the voltage result.

TIMER1 takes care of reading frequencies and is configured to read the number of pulses in one second. After that, the PIC transmits and receives those readings. With the data ready, the instruction HSEROUT sends the serial data. The code is always waiting to be activated by the user. Once activated, it sends a control variable to the receiving unit with the instruction HSE-ROUT, which is sent at a speed of 2400 bits/s. This variable indicates that data transmission is starting:

## HSEROUT ["BZ0",10]; SEND ACTIVATION INSTRUCTION

For the transmitter code, it's necessary to read a voltage and frequency with the microcontroller's ADC and TIMER1, respectively. The maximum frequency is:

$$F_{max} = 2^{16} - 1 = 65535$$

Then the code proceeds to obtain each decimal digit with the command DIG, and each digit is sent at 2400 bits/s with the serial data with the instruction Hserout. Two digits in ASCII code are inserted at the beginning of each data package, so that the receive identifies which data is receiving.

HSEROUT ["BZ",DEC A,10];

HSEROUT ["AZ",VD[3],VD[2],VD[1], VD[0],10]; send serial data

HSEROUT ["CZ",H[4],H[3],H[2],H[1] ,H[0],10];

When the receiver gets the data "BZ," it represents the activity control in the serial port. The data package AZ represents the Voltage, while the data package CZ represents Frequency.

When the Micro receives the stop instruction

2. The transmit module is tied to the PIC16F1619 microcontroller. The voltage and frequency inputs are not shown. Our transmitter prototype uses a potentiometer to adjust the voltage and frequency inputs.





LCD\_2

AONEN0984

with the pushbutton, it transmits a "1" in the control data package, indicating end of transmission.

The receiver waits for the control variable BZ with the instructions HSERIN and WAIT, which waits specifically for the data "BZ". Because that RF data transmission has external noise, after receiving the data BZ, the next data is stored in a variable to manipulate as follows:

HSERIN 10,MAIN1,[WAIT("BZ"), STR A\1]; wait one second to receive instruction

When the second micro receives the control variable, it starts the data input where the instructions HSERIN wait for ASCII codes in each package that's transmitted. If it doesn't receive any data, the code jumps to the next instruction as follows:

HSERIN 10,JUMP1,[WAIT ("BZ"), STR A\1]

JUMP1: HSERIN 100,JUMP2,[WAIT ("AZ"), STR VD\4] JUMP2: HSERIN 10,HERE, [WAIT ("CZ"), STR H\5]

When the data is received, it's transferred to two LCD displays. By using a CMOS switch <u>HCF4066</u>, it's possible to control the Enable and R/W functions in each LCD. Two bits in the microcontroller select which LCD will be operating as shown in the following code:

LCD1 = 1 ; ENABLE LCD1

LCD2 = 1 ; ENABLE LCD2

The receiver PIC microcontroller drives two 16X2 LCD displays controlled by a quad CMOS switch—HCF4066— configured as multiplexer to select which LCD will receive data. In this case, the first LCD shows Voltage, and the second one shows Frequency.

For the receiver, we use the instruction HSERIN, which receives two ASCII characters, and then saves the respective data of both characters.

Ricardo Jimenez, holds a Master's degree in Electronics Engineering. He is the author of several Lab Practices Notebooks on PIC Microcontrollers applied to Instrumentation.

*Gabriel Lee Alvarez is pursuing his Electronics Engineering degree at Instituto Tecnologico de Mexicali (ITM).* 

3. The receiver is based on a PIC16F1614 that handles incoming messages to display on the LCD displays. For the receiver side, we have the voltage and frequency displayed on different LCD displays.



4. This timing diagram shows an ASCII character A (0b10000010) using the usual asynchronous process with a start bit and the trailing stop bit.

Listing 1: Code for the PIC16F1619 Used as a transmitter ADCON0 = %00001111;AN3 IS ENABLED ADCON1 = %10000000; FOSC/2, VDD WIRELESS VOLTMETER/FREO RS232 ;-----RF ; TRANSMITER DEFINE HSER\_RCSTA 90h; RX MODULE IS ENABLED ; PBP3 COMPILER from melabs.com DEFINE HSER\_TXSTA 20h; TX MODULE IS ENABLED ; Authors: Ricardo Jimenez and Gabriel Lee Alvarez DEFINE HSER\_BAUD 2400; BAUD RATE IS 2400 ; © July 13, 2020 rc1sta.7 = 1; SERIAL COMUNICATION IS ENABLED ; PIC16F1619 RB7PPS = %10010;; TRANSMITTER PIC16F1619 #CONFIG ;--LCD CONFIGURATION -----\_\_\_\_config \_CONFIG1, \_FOSC\_INTOSC & \_PWRTE\_ DEFINE LCD\_DREG PORTC ' PORTC is LCD data port ON & \_MCLRE\_OFF & \_CP\_OFF & \_BOREN\_ON & \_ DEFINE LCD\_DBIT 0 ' PORTC.0 is the data LSB DEFINE LCD\_RSREG PORTA ' RS is connected to CLKOUTEN\_OFF \_\_\_\_\_CONFIG2, \_\_\_\_\_\_OFF & \_\_PPS1WAY\_OFF PORTA.0 &\_ZCD\_OFF &\_PLLEN\_OFF &\_STVREN\_ON &\_BORV\_ DEFINE LCD\_RSBIT 0 LO & LVP\_OFF DEFINE LCD\_EREG PORTA ' E is connected to PORTA.1 \_\_config \_CONFIG3, \_WDTCPS\_WDTCPS4 & \_ DEFINE LCD\_EBIT 1 WDTE\_ON & \_WDTCWS\_WDTCWS100 & \_WDTCCS\_ DEFINE LCD\_BITS 4 ' 4 data line **LFINTOSC** DEFINE LCD\_LINES 2 ' 2-line display #ENDCONFIG DEFINE LCD\_COMMANDUS 1500 ' Use 1500uS DEFINE OSC 16; command delay OSCCON = %01111010; intenal Osc set to 16 MHZ DEFINE LCD\_DATAUS 44 ' Use 44uS data delay OSCTUNE = 0OSCSTAT = %00011111; TRISA = %00111110; RA0 AS A OUTPUTS, LCDOUT \$FE,\$28; \$28 FUNCTION SET, 4 BITS RA1:RA2:RA3:RA4:RA5 AS A INPUTS LCDOUT \$FE,\$10; \$10 SHIFT DISPLAY RA0:RA1:RA4:RA5 LCDOUT \$FE,\$0C; \$0C DISPLAY ON ANSELA %000100; AS = DIGITAL.RA2:RA3 AS ANALOG, LCDOUT \$FE,\$06; \$06 ENTRY MODE SET TRISC = %100000;RC0:RC1:RC2:RC3:RC4 AS OUTPUTS, ;-----RC5 AS INPUTS ;CREATING ALIAS TRISC=0; Clearing PORTC ;TX VAR PORTC.0;; TRISB = 0; Clearing PORTB **PB VAR PORTA.3** ANSELB = 0; PORTB set as digital :----;PPSLOCK=0; ;---VARIABLES ;ANSEL PULL-UP resistors disabled X VAR byte[4]; WPUA = 0; AC = 0; Y VAR WORD; WPUC = 0; PULL-UPS DISABLED VIN VAR WORD; OPTION\_REG.7 = 0; PULL-UPS ENABLED VINBCD var word H VAR BYTE[5]; WPUA.3 = 1; PULL-UP IN RA3 ENABLED TMR VAR WORD; T1CON = %10000101; TMR1 ENABLED VD var byte[4];

IN we byte: OUT VAR BYTE: (	A VAR BIT	ENDIF;				
OUT VAR BYTE; ; A = 1IF A = 0 THEN STAY; If CONTROL IS "0" GO TO STAY; A = 1; CONTROL ARIABLE IS "1" A = 1; CONTROL ARIABLE IS "1" 	IN var byte;					
	OUT VAR BYTE;	IF A = 0 THEN STAY; If CONTROL IS "0" GO TO STAY				
$ \begin{array}{llllllllllllllllllllllllllllllllllll$	;	A = 1;; CONTROL VARIABLE IS "1"				
for $x = 0$ to 5GOSUB SENDvd(g) = "0";NEXT Xnext xLCDOUT SFE,S80," END ";send a message to the LCDLCDOUT SFE,S80," FEQUENCY METER.";LCDOUT SFE,S80,"FEQUENCY METER.";PAUSE 1000LCDOUT SFE,S00,"-VOLTMETER RF";PAUSE 1000TMR = 0;goto main;TMR = 0;ADC;TICON.0 = 0; TICON input Frequency DISABLEDADC:MAINVIN.8YTE = ADRESL;SAVE LOWUT SFE,S00,"-VOLTMETER RF";VIN.8YTE = ADRESL;yVIN.8YTE = ADRESL;LCDOUT SFE,S00,"-VOLTMETER RF";VIN.8YTE1 = ADRESL;yVIN.8YTE1 = ADRESL;LCDOUT SFE,S00,"-VOLTMETER RF";VIN.8YTE1 = ADRESL;yVIN.8YTE1 = ADRESL;IF PB = 0 THEN; wait for Push button to go LowPAUSE 100PAUSE 100THE ADC IN VARIABLE VINNEXT X;VIN.8YTE1 = ADRESL;HSTRUTIONHSEROUT ["BZ0";10]; SEND ACTIVATIONNEXT X;FOR X = 0 TO 3;IF A = 0 THEN; START QUESTION FOR CONTROLVIN = div32 1000;IF A = 0 THEN; START QUESTION FOR CONTROLVD[X] = OUT;VDIX[ = OUT;SAVE DECODED DIGITSNEXT X;LCDOUT SFE,S80,"TAZ ";ICDOUT SFE,S80,"TAZ ";LCDOUT SFE,S80,"TAZ ",TICON.0=1;PAUSE 1000;TMRIH = 0;LCDOUT SFE,S80,"TAZ ",TICON.0=1;ENDIEPAUSE 1000;SEND; LABEL STIMERGOSUB ADC;GOT D, DEC AND RETURN;MUSE 1000;HSEROUT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; SENDGOSUB ADC;GOT D, DEC A	A = 1	FOR $X = 0$ TO 2;				
	for $x = 0$ to 5	GOSUB SEND				
h(x) = "0"; next x send a message to the LCD LCDOUT \$FE,\$S0," FREQUENCY METER."; LCDOUT \$FE,\$S0," FREQUENCY METER."; PAUSE 2000; TMR = 0; TICON.0 = 0; TICON input Frequency DISABLED MAINLCDOUT \$FE,\$S0," TX "; PAUSE 1000LCDOUT \$FE,\$S0," FREQUENCY METER."; LCDOUT \$FE,\$S0," FREQUENCY METER."; LCDOUT \$FE,\$S0," FREQUENCY METER."; LCDOUT \$FE,\$S0," FREQUENCY METER."; LCDOUT \$FE,\$S0," -VOLTMETER RF"; 'PAUSE 150 HSEROUT ["BZ0",10]; SEND ACTIVATION INSTRUCTION NEXT X; ENDIF;LCDOUT \$FE,\$CO,"-VOLTMETER RF"; 'PAUSE 150 HSEROUT ["BZ0",10]; SEND ACTIVATION INSTRUCTION NEXT X; ENDIF;LCDOUT \$FE,\$CO,"-VOLTMETER RF"; 'VIN PUSE 150 HSEROUT ["BZ0",10]; SEND ACTIVATION INSTRUCTION NEXT X; ENDIF;LCDOUT \$FE,\$CO," AND HIGHER REGISTER OF THE ADC IN VARIABLE VIN 'VIN PUSE 100; THE ADC SHAWLAWARMS7; MULTIPLYING BY RESISE 4.4887 VIN Edv32 1000; PERFORM 16-BIT DIVISION enable; ENABLE INTERRUPTS FOR X = 0 TO 3; START LOOP IN = VIN DIG X; GET DIGIT X LCDOUT \$FE,\$SO," TX "; PAUSE 1000; TMRH1 = 0; TICON.0 = 1; ENABLES TIMER STAY;; TICON.0 = 1; ENABLES TIMER STAY; TICON.0 = 1; ENABLES TIMER STAY; TICON.0 = 1; ENABLES TIMER STAY; TICON.0 = 1; ENABLES TIMER STAY; TICON.0 = 1; ENABLES TIMER GOSUB ADC; GOSUB ADC; GOSUB ADC; GOSUB ADC; GOSUB ADC; GOSUB ADC; GOSUB ADC; GOSUB ADC; GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURN GOSUB SEND; if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END FRANKISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TRANSMISSION TARIA SUB SUB TRANSMISSION TRANSMIS	vd[x] = "0";	NEXT X				
next xLCDOUT \$FE,\$C0," TXTXgend a message to the LCDLCDOUT \$FE,\$S0,"FRRQUENCY METER.";PAUSE 1000LCDOUT \$FE,\$S0,"FRRQUENCY METER.";goto main;LCDOUT \$FE,\$S0,"FRRQUENCY METER.";ADC(N0.1 = 1;LCDOUT \$FE,\$S0,"FRRQUENCY METER.";HERE1: IF ADCON0.1 = 1 THEN HERE1; CONVER.LCDOUT \$FE,\$S0,"FRRQUENCY METER.";HERE1: IF ADCON0.1 = 1 THEN HERE1; CONVER.LCDOUT \$FE,\$S0,"FRRQUENCY METER.";HERE1: IF ADCON0.1 = 1 THEN HERE1; CONVER.LCDOUT \$FE,\$S0,"FRRQUENCY METER.";HERE1: IF ADCON0.1 = 1 THEN HERE1; CONVER.LCDOUT \$FE,\$S0,"CONTROL VARIABLE IS 0HERE1: IF ADCON0.1 = 1 THEN HERE1; CONVER.A = 0;CONTROL VARIABLE IS 0FOR X = 0 TO 3;HSEROUT ("BZ0,"10]; SEND ACTIVATIONINSTRUCTIONHSEROUT ("BZ0,"10]; SEND ACTIVATIONINSTRUCTIONFITHENITICON0 = 1; ENABLES TIMERHOW_LCD; GOTU \$FE,\$S0,"HZ= ",H[4],H[3],H[2],H[1],H[0],"STAY;TICON0 = 1; ENABLES TIMERSTAY;TICON0 = 1; ENABLES TIMERSTAY;TICON0 = 1; ENABLES TIMERGOSUB SLD0;HSEROUT ("BZ7,DC A,10];GOSUB SEND;HSEROUT ("BZ7,DC A,10];ITANN H = 0;HSEROUT ("BZ7,DC A,10];TICON0 = 1; ENABLES TIMERSEND; LABEL SEN	h[x] = "0";	LCDOUT \$FE,\$80," END ";				
send a message to the LCD LCDOUT SFE,850,"TREQUENCY METER."; LCDOUT SFE,850,"TREQUENCY METER."; PAUSE 2000; TICON.0 =0; TICON input Frequency DISABLED MAIN TICON.0 =0; TICON input Frequency DISABLED MAIN TICON.0 =0; TICON input Frequency DISABLED MAIN TRAINSMISSION PAUSE 1000 TICON.0 =0; TICON input Frequency DISABLED MAIN TRAINSMISSION FOR X= 0 TO 3; TICON.0 =0; TICON VARIABLE IS 0 FOR X= 0 TO 3; TICON.0 =0; TIMER TART QUESTION FOR CONTROL VARIABLE INTERRUPTS TICON.0 =1; ENABLES TIMER STAY; TICON.0 =1; ENABLES TIMER STAY; TICON.0 =1; ENABLES TIMER TRAINSMISSION FOR X= 0; CONTROL variable is 1, END FOR X= 0; CONTROL variable is 1, E	next x	LCDOUT \$FE,\$C0," TX ";				
LCDOUT \$FE,\$80,"TREQUENCY METER."; LCDOUT \$FE,\$80,"TREQUENCY METER."; PAUSE 2006; TICON.0-0; TICON input Frequency DISABLED MAIN LCDOUT \$FE,\$80,"FREQUENCY METER."; LCDOUT \$FE,\$80,"FREQUENCY METER."; LCDOUT \$FE,\$80,"FREQUENCY METER."; LCDOUT \$FE,\$80,"FREQUENCY METER."; LCDOUT \$FE,\$60,"VOLTMETER RF"; ''''''''''''''''''''''''''''''''''''	send a message to the LCD	PAUSE 1000				
LCDOUT \$FE,\$C0,"VOLTMETER RF"; PAUSE 2006; TICON.0 =0; TICON input Frequency DISABLED MAIN LCDOUT \$FE,\$80," FREQUENCY METER."; LCDOUT \$FE,\$80," -VOLTMETER RF"; : IF PB = 0 THEN; wait for Push button to go Low PAUSE 150 A = 0; CONTROL VARIABLE IS 0 FOR X = 0 TO 3; HSEROUT ("BZ0",10]; SEND ACTIVATION INSTRUCTION NESTRX X; ENDIF; ICDOUT \$FE,\$80," START "; LCDOUT \$FE,\$80," START S, RETURN; IF DEC; LABEL \$LDEC; TMR BYTED = MRL1; OBTAIN LOWER REGISTER OF TMRL P, DEC; LABEL \$LDEC; TMR BYTED = TMRL1; OBTAIN LOWER REGISTER OF TMRL P, DEC; LABEL \$LDEC; TMR BYTED = TMRL1; OBTAIN LOWER REGISTER OF TMRL P, DEC; LABEL \$LDEC; TMRL P, DEC; LABEL \$LDEC; TMRL P, DEC; LABEL \$LDEC; TMRL P, DEC; LABEL \$LDEC; TMRL P, DEC;	LCDOUT \$FF.\$80."FREOUENCY METER-":	ENDIF				
PAUSE 2000; TMR = 0; TICON.0 = 0; TICON input Frequency DISABLED MAINgoto main; ADC; <td>LCDOUT \$FF.\$C0."VOLTMETER RF":</td> <td></td>	LCDOUT \$FF.\$C0."VOLTMETER RF":					
TMR = 0; TICON.0 = 0; TICON input Frequency DISABLED MAIN ADC; ADC:	PAUSE 2000:	goto main:				
TICON.0 = 0; TICON input Frequency DISABLED MAIN LCDOUT \$FE,\$80,"FREQUENCY METER."; LCDOUT \$FE,\$60,"VOLTMETER RF"; ; IF PB = 0 THEN; wait for Push button to go Low PAUSE 150 A = 0; CONTROL VARIABLE IS 0 FOR X = 0 TO 3; HSEROUT ["BZ0,"10]; SEND ACTIVATION INSTRUCTION IF A = 0 THEN; START QUESTION FOR CONTROL variable LCDOUT \$FE,\$80," START "; LCDOUT \$FE,\$80," TX "; PAUSE 1000 TICON,0=1; PAUSE 1006 TICON,0=0; TIMER DISABLED GOSUB ADC; GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURN; If pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END TRANSMISSION PAUSE 200; TMRENTE = TMRLI; OBTAIN LOWER REGISTER OF TMRENTE = TMRLI; OBTAIN LOWER REGISTER OF TMRENTER = TMRLI OF TMRLI OBTAIN LOWER REGISTER OF TMRENTER = TMRLI OBTAIN LOWER REGISTER OF TMR	TMR = 0	ADC::				
MAIN MAIN	T1CON = 0, T1CON = 0. T1CON input Frequency DISABLED	ADCON0.1 – 1: FNABLE ADC MODULE				
$ \begin{array}{llllllllllllllllllllllllllllllllllll$	MAIN	HEREI: IF ADCON0.1 $-$ 1 THEN HEREI: CONVER-				
ICDOUT \$FE,\$80,"FREQUENCY METER."; ICDOUT \$FE,\$80,"FREQUENCY METER."; ICDOUT \$FE,\$C,"VOLTMETER RF"; iCDOUT \$FE,\$C,"VOLTMETER RF"; iCDOUT \$FE,\$CO,"VOLTMETER RF"; iCDOUT \$FE,\$CO,"YOLTMETER RF"; iCDOUT \$FE,\$CO,"YOLTMETER RF"; iCDOUT \$FE,\$CO,"; iCDOUT \$FE,	WITHIN .	SION in progress				
ICDOOT \$FE,\$GO,"VOLTMETER RF";ILCDOUT \$FE,\$CO,"VOLTMETER RF";;IF $PB = 0$ THEN; wait for Push button to go LowPAUSE 150 $A = 0;$ CONTROL VARIABLE IS 0FOR X= 0 TO 3;HSEROUT ["BZ0",10]; SEND ACTIVATIONINSTRUCTIONNEXT X;ENDIF;IF A = 0 THEN; START QUESTION FOR CONTROLLCDOUT \$FE,\$SO," START ";LCDOUT \$FE,\$SO," START ";LCDOUT \$FE,\$SO," START ";LCDOUT \$FE,\$SO," TX ";PAUSE 1000;TMRLI = 0;CLEAR TIMERTMRIH = 0;TICON.0 = 1; ENABLES TIMERGOSUB ADC;GOSUB ADC;GOSUB ADC;GOSUB ADC;GOSUB SEND;if $pb = 0$ then; wait for Push button to go highA = 1;CONTROL variable is 1, ENDTRANSMISSIONPAUSE 200;TRANSMISSIONPAUSE 200;TRANSMISSIONPAUSE 200;TRANSMISSIONPAUSE 200;TRANSMISSIONPAUSE 200;TANNIME A = 1;CONTROL variable is 1, ENDTRANSMISSIONPAUSE 200;TANNIME A = 1;CONTROL variable is 1, ENDTRANSMISSIONPAUSE 200;TIMENTADALE INTERCENT FORTADALE INTERCENT FORTRANSMISSIONPAUSE 200;TADALE INTERCENT FORTADALE INTERCENT FORTADALE INTERCENT FORTADALE INTERCENT FORTADALE INTERCENT FORTADALE INTERCENT	I COALIT ÉEE ÉQUIEDEALIENCY METED ".	VIN DVTED - ADDESL. SAVE LOWED DECISTED OF				
ICDOUT SPE,SEQ, "VOLTMETER KF-";ITRE ADC IN VARIABLE VINis	LODOUT \$FE,\$60, FREQUENCI METER DE	V IN.DI IEU = ADRESL, SAVE LOWER REGISTER OF				
IF PB = 0 THEN; wait for Push button to go Low PAUSE 150VIN.BIT IE = ADRESH: SAVE HIGHER REGISTER OF THE ADC IN VARIABLE VIN disable; DISABLE INTERRUPTSA = 0; CONTROL VARIABLE IS 0 FOR X= 0 TO 3; HSEROUT ["BZ0",10]; SEND ACTIVATION INSTRUCTION IT CON.0= 1; PAUSE 1000 TICON.0=1; PAUSE 200;VIN EI = ADRESH; SAVE HIGHER REGISTER OF TIRANSMISSION PAUSE 200;ITON ADDATION TO ADDATION FOR CONTROL VIN EI E ADREST SAVE THER TICON ADDATION FOR CONTROL VIN EI E ADREST SAVE THER PAUSE 200;VIN EI E ADREST SAVE THER TICON ADDATION FOR CONTROL VIN EI E ADREST SAVE THER SHOW_LCD; GO TO SHOW_LCD ADD RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; RETURN; <b< td=""><td>LCDOUT \$FE,\$C0, VOLIMETER RF ;</td><td>I THE ADU IN VARIABLE VIN</td></b<>	LCDOUT \$FE,\$C0, VOLIMETER RF ;	I THE ADU IN VARIABLE VIN				
In PB = 0 THEN; wait for Push button to go LowTHE ADC IN VARIABLE VINPAUSE 150disable;DISABLE INTERRUPTSA = 0;CONTROL VARIABLE IS 0disable;DISABLE INTERRUPTSFOR X = 0 TO 3;HSEROUT ["BZ0",10]; SEND ACTIVATIONenable;ENABLE INTERRUPTSINSTRUCTIONNEXT X;ENDIF;UIN = div32 1000;PERFORM 16-BIT DIVISIONIF A = 0 THEN; START QUESTION FOR CONTROLenable;ENABLE INTERRUPTSVariableLCDOUT SFE,\$80," START ";LCDOUT SFE,\$80," START ";LCDOUT SFE,\$80," START ";DECODINGVDIX1 = 0; CLEAR TIMERVD[X] = OUT;SAVE DECODED DIGITSTHCON.0 = 1;ENABLES TIMERSTAY:;TICON.0 = 0; TIMER DISABLEDSHOW_LCD; display LABEL on LCDGOSUB ADC;GOT b_DEC AND RETURN;GOSUB SHOW_LCD; GO TO SHOW_LCD ANDSEND; LABEL SEND;RETURNGOSUB SHOW_LCD; GO TO SHOW_LCD ANDRETURNGOSUB SEND;if pb = 0 then; wait for Push button to go high A = 1;CONTROL variable is 1, ENDTRANSMISSIONPAUSE 200;TAMISH SIONPAUSE 200;		VIN.BY IEI = ADRESH; SAVE HIGHER REGISTER OF				
PAUSE 150Disable: Nexpected by the second	IF $PB = 0$ THEN; wait for Push button to go Low	I HE ADU IN VARIABLE VIN				
A = 0; CONTROL VARIABLE IS 0 FOR X = 0 TO 3; CONTROL VARIABLE IS 0 HSEROUT ["BZ0",10]; SEND ACTIVATION INSTRUCTION NEXT X; ENDIF; IF A = 0 THEN; START QUESTION FOR CONTROL variable LCDOUT \$FE,\$80," START "; LCDOUT \$FE,\$80," TX "; PAUSE 1000; TMR1L = 0; TICON.0 = 1; ENABLES TIMER STAY;; TICON.0 = 1; ENABLES TIMER GOSUB ADC; GOSUB ADC; GOSUB ADC; GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURN GOSUB SHOM_LCD; GO TO SHOW_LCD AND RETURN; If pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END TRANSMISSION PAUSE 200; HSEROLT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; STAP:; ITMENTE O = TIMEL ; OBTAIN LOWER REGISTER OF TMR.BYTEO = TMR1L; OBTAIN LOWER REGISTER OF TIMENTE	PAUSE 150	disable; DISABLE INTERRUPTS				
$\begin{array}{llllllllllllllllllllllllllllllllllll$	A = 0; CONTROL VARIABLE IS 0	VINBCD = VIN <sup>488</sup> /; MULTIPLYING BY RESLSB				
HSEROUT ["BZ0,10]; SEND ACTIVATIONVIN = dw221000; PERFORM 16-BIT DIVISIONINSTRUCTIONenable; ENABLE INTERRUPTSNEXT X;enable; START LOOPENDIF;IN = VIN DIG X; GET DIGIT XLCDOUT \$FE,\$80," START QUESTION FOR CONTROLIN = VIN DIG X; GET DIGIT XvariableLCDOUT \$FE,\$80," START ";LCDOUT \$FE,\$c0," TX ";DECODINGPAUSE 1000;TMRIH = 0;TMRIH = 0;SHOW_LCD; display LABEL on LCDTICON.0 = 1; ENABLES TIMERLCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0],"STAY;;TICON.0=1;PAUSE 1000TICON.0=0; TIMER DISABLEDGOSUB ADC;GOSUB ADC;GOSUB SHOW_LCD; GO TO h_DEC AND RETURN;GOSUB SHOW_LCD; GO TO SHOW_LCD ANDRETURNGOSUB SHOW_LCD; GO TO SHOW_LCD ANDRETURNGOSUB SEND;if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, ENDTRANSMISSIONPAUSE 200;TRANSMISSIONPAUSE 200;TIMERI	FOR $X = 0$ TO 3;					
INSTRUCTIONenable;ENABLE INTERRUPTSNEXT X;FOR X = 0 TO 3;START LOOPENDIF;FOR X = 0 TO 3;START LOOPIN = VIN DIG X;GET DIGIT XLCDOUT \$FE,\$80," START QUESTION FOR CONTROLDECODINGvariableUD(X) = 0UT;SAVE DECODED DIGITSLCDOUT \$FE,\$c0," TX ";NEXT X;PAUSE 1000;SHOW_LCD;display LABEL on LCDTMRIH = 0;SHOW_LCD;display LABEL on LCDTICON.0 = 1;ENABLES TIMERSHOW_LCD;PAUSE 1000TICON.0=0;TIMER DISABLEDGOSUB ADC;GOSUB ADC;SEND;GOSUB SHOW_LCD;GO TO h_DEC AND RETURN;SERIAL dataRETURNGOSUB SHOW_LCD;GO TO SHOW_LCD ANDRETURNGOSUB SEND;RETURN;if pb = 0 then; wait for Push button to go high A = 1;CONTROL variable is 1, ENDTRANSMISSIONFMUSE 200;TMR.BYTE0 = TMRIL; OBTAIN LOWER REGISTER OF TIMERI	HSEROUT ["BZ0",10]; SEND ACTIVATION	VIN = div32 1000; PERFORM 16-BIT DIVISION				
NEXT X; ENDIF;FOR X = 0 TO 3;START LOOP IN = VIN DIG X;ICDOUT \$FE,\$C0,"IN = VIN DIG X;GET DIGIT X LOOKUP IN,["0123456789ABCDEFG"],OUT;DIGITSVariableICDOUT \$FE,\$80,"START "; ICDOUT \$FE,\$80,"DECODING VD[X] = OUT;SAVE DECODED DIGITSICDOUT \$FE,\$80,"START "; ICDOUT \$FE,\$C0," TX "; PAUSE 1000; TICON.0=1; PAUSE 1000 TICON.0=1; PAUSE 1000 TICON.0=0; TICON.0=0; TICON.0=0; GOSUB ADC; GOSUB SHOW_LCD; GOSUB SHOW_LCD; GOSUB SHOW_LCD; GOSUB SHOW_LCD; GOSUB SHOW_LCD; GOSUB SHOW_LCD; GOSUB SEND; If pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END PAUSE 200;STAR: LOOP IN = VIN DIG X; DECODING VD[X] = 0UT; STAY: return; SHOW_LCD; display LABEL on LCD LCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0]," "; ICDOUT \$FE,\$C0,"V= ",VD[3],",VD[2],VD[1],VD[0]," "; RETURN; HSEROUT ["BZ",DEC A,10]; HSEROUT ["BZ",DEC A,10]; HSEROUT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; SEND SERIAL data HSEROUT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; SEND SERIAL data HSEROUT ["AZ",H[4],H[3],H[2],H[1],H[0],10]; RETURN;	INSTRUCTION	enable; ENABLE INTERRUPTS				
ENDIF;IN = VIN DIG X;GET DIGIT XIF A = 0 THEN; START QUESTION FOR CONTROL variableIN = VIN DIG X;GET DIGIT XLCDOUT \$FE,\$80," START "; LCDOUT \$FE,\$80," TX "; PAUSE 1000;DECODINGTMR1L = 0;CLEAR TIMER TMR1H = 0; TICON.0 = 1; ENABLES TIMERSHOW_LCD; display LABEL on LCD LCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0]," "; LCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0],"STAY; TICON.0=1; PAUSE 1000 TICON.0=0; TIMER DISABLED GOSUB ADC; GOSUB ADC; GOSUB SHOW_LCD; GO TO h_DEC AND RETURN; GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURN GOSUB SEND; if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END PAUSE 200;IN = VIN DIG X;GET DIGIT X LOOUT \$FE,\$67,9ABCDEFG"],OUT; DIGITS DECODING VD[X] = OUT;SAVE DECODED DIGTS NEXT X; return;NEXT X; return;SHOW_LCD; display LABEL on LCD LCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0],"STAY; TICON.0=1; GOSUB ADC; GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURN GOSUB SEND;SEND; LABEL SEND; HSEROUT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; SEND SERIAL data HSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10]; RETURN;RETURN;H_DEC; TMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF TIMERI	NEXT X;	FOR $X = 0$ TO 3; START LOOP				
$      ICOCKUP IN, ["0123456789ABCDEFG"], OUT; DIGITS \\       LOOKUP IN, ["0123456789ABCDEFG"], OUT; DIGITS \\       LCDOUT $FE, $80," START "; \\       LCDOUT $FE, $80," START "; \\       LCDOUT $FE, $20," TX "; \\       PAUSE 1000; \\       TICON.0 = 1; ENABLES TIMER \\                                   $	ENDIF;	IN = VIN DIG X; GET DIGIT X				
IF A = 0 THEN; START QUESTION FOR CONTROL variableDECODINGvariable $DECODING$ LCDOUT \$FE,\$80," START "; LCDOUT \$FE,\$00," TX "; PAUSE 1000; TMR1L = 0;CLEAR TIMERNEXT X; return;TMR1H = 0; T1CON.0 = 1; ENABLES TIMERSHOW_LCD; display LABEL on LCD LCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0]," "; LCDOUT \$FE,\$00,"V= ",VD[3],",VD[2],VD[1],VD[0],"STAY; T1CON.0 = 1; ENABLES TIMER"; LCDOUT \$FE,\$00,"V= ",VD[3],",VD[2],VD[1],VD[0]," "; RETURN GOSUB ADC; GOSUB SHOW_LCD; GO TO h_DEC AND RETURN; GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURN GOSUB SEND; if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END PAUSE 200;SERIAL data H_DEC; LABEL h_DEC; TMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF TIMER1		LOOKUP IN,["0123456789ABCDEFG"],OUT; DIGITS				
variableVD[X] = OUT;SAVE DECODED DIGITSLCDOUT \$FE,\$80,° START ";NEXT X;LCDOUT \$FE,\$C0,° TX ";NEXT X;PAUSE 1000;SHOW_LCD;;TMR1H = 0;SHOW_LCD;;T1CON.0 = 1;ENABLES TIMERSTAY;;";T1CON.0 = 1;ENABLES TIMERSTAY;;";T1CON.0 = 1;ENABLEDGOSUB ADC;RETURN;GOSUB ADC;SEND;; LABEL SEND;GOSUB SHOW_LCD;SO TO h_DEC AND RETURN;GOSUB SHOW_LCD;GO TO SHOW_LCD ANDRETURNSERIAL dataHSEROUT ["AZ", VD[3], VD[2], VD[1], VD[0], 10]; SENDSERIAL dataHSEROUT ["CZ", H[4], H[3], H[2], H[1], H[0], 10];GOSUB SEND;RETURN;if pb = 0 then; wait for Push button to go high A = 1;CONTROL variable is 1, ENDH_DEC; LABEL h_DEC;TMR.BYTE0 = TMRIL; OBTAIN LOWER REGISTER OF TMRENT LOWER REGISTER OFPAUSE 200;TIMER1	IF $A = 0$ THEN; START QUESTION FOR CONTROL	DECODING				
LCDOUT \$FE,\$80," START"; ICDOUT \$FE,\$C0," TXNEXT X; return;PAUSE 1000; TMR1H = 0; TICON.0 = 1; ENABLES TIMERSHOW_LCD; display LABEL on LCD LCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0]," "; ICDOUT \$FE,\$80,"HZ= ",U[4],H[3],H[2],H[1],H[0]," "; ICDOUT \$FE,\$C0,"V= ",VD[3],"",VD[2],VD[1],VD[0]," "; RETURN;STAY:; TICON.0 = 1; ENABLES TIMER"; ICDOUT \$FE,\$C0,"V= ",VD[3],"",VD[2],VD[1],VD[0]," "; RETURN;GOSUB ADC; GOSUB ADC; GOSUB SHOW_LCD; GO TO h_DEC AND RETURN; GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURN GOSUB SEND;SEND:; LABEL SEND; HSEROUT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; SEND SERIAL data HSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10]; RETURN;if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END PAUSE 200;H_DEC;; LABEL h_DEC; TMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF TIMER1	variable	VD[X] = OUT; SAVE DECODED DIGITS				
LCDOUT \$FE,\$C0," TX "; PAUSE 1000;return;PAUSE 1000;return;TMR1L = 0;SHOW_LCD;; display LABEL on LCD LCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0],"T1CON.0 = 1; ENABLES TIMER"; LCDOUT \$FE,\$C0,"V= ",VD[3],",VD[2],VD[1],VD[0],"STAY;; T1CON.0 = 1; PAUSE 1000 T1CON.0 = 0; TIMER DISABLED GOSUB ADC; GOSUB SHOW_LCD; GO TO h_DEC AND RETURN; GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURN GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURN GOSUB SEND; if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END TRANSMISSIONSEND; LABEL h_DEC; TMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF TIMER1	LCDOUT \$FE,\$80," START ";	NEXT X;				
PAUSE 1000;SHOW_LCD;; display LABEL on LCDTMR1H = 0;LCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0],"T1CON.0 = 1; ENABLES TIMER";T1CON.0 = 1; ENABLES TIMER";T1CON.0 = 1;LCDOUT \$FE,\$C0,"V= ",VD[3],","VD[2],VD[1],VD[0],"STAY:;";T1CON.0 = 1;RETURN;PAUSE 1000SEND; LABEL SEND;GOSUB ADC;SEND;; LABEL SEND;GOSUB H_DEC; GO TO h_DEC AND RETURN;HSEROUT ["BZ,"DEC A,10];GOSUB SHOW_LCD; GO TO SHOW_LCD ANDSERIAL dataRETURNSERIAL dataRETURNSERIAL dataRETURNMSEROUT ["CZ,"H[4],H[3],H[2],H[1],H[0],10];GOSUB SEND;RETURN;if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, ENDH_DEC;; LABEL h_DEC; TMR.BYTE0 = TMR1L ; OBTAIN LOWER REGISTER OF TIMER1	LCDOUT \$FE,\$C0," TX ";	return;				
TMR1L = 0;CLEAR TIMERSHOW_LCD:; display LABEL on LCDTMR1H = 0;LCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0],"T1CON.0 = 1; ENABLES TIMER";STAY:;LCDOUT \$FE,\$C0,"V= ",VD[3],"",VD[2],VD[1],VD[0],"T1CON.0=1;RETURN;PAUSE 1000SEND:; LABEL SEND;GOSUB ADC;SEND:; LABEL SEND;GOSUB H_DEC; GO TO h_DEC AND RETURN;HSEROUT ["BZ",DEC A,10];GOSUB SHOW_LCD; GO TO SHOW_LCD ANDSERIAL dataRETURNHSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];GOSUB SEND;RETURN;if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, ENDH_DEC;; LABEL h_DEC; TMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF TIMER1	PAUSE 1000;					
TMR1H = 0; T1CON.0 = 1; ENABLES TIMERLCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0],"STAY:; T1CON.0=1; PAUSE 1000 T1CON.0=0; TIMER DISABLED GOSUB ADC; GOSUB H_DEC; GO TO h_DEC AND RETURN; GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURN GOSUB SEND;SEND.; LABEL SEND; HSEROUT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; SEND SERIAL data HSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];RETURN GOSUB SEND; if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END PAUSE 200;RETURN;TRANSMISSION PAUSE 200;H_DEC; TIMER1	TMR1L = 0;CLEAR TIMER	SHOW_LCD:; display LABEL on LCD				
TICON.0 = 1; ENABLES TIMER"; LCDOUT \$FE,\$C0,"V= ",VD[3],"",VD[2],VD[1],VD[0],"STAY;; TICON.0=1; PAUSE 1000"; RETURN;TICON.0=0; TIMER DISABLED GOSUB ADC; GOSUB H_DEC; GO TO h_DEC AND RETURN; GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURNSEND;; LABEL SEND; HSEROUT ["BZ",DEC A,10]; HSEROUT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; SEND SERIAL data HSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];RETURN GOSUB SEND;RETURN;if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END PAUSE 200;H_DEC; TMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF TIMER1	TMR1H = 0;	LCDOUT \$FE,\$80,"HZ= ",H[4],H[3],H[2],H[1],H[0],"				
STAY:;LCDOUT \$FE,\$C0,"V= ",VD[3],"",VD[2],VD[1],VD[0],"STAY:;";T1CON.0=1;RETURN;PAUSE 1000RETURN;GOSUB ADC;SEND:; LABEL SEND;GOSUB ADC;HSEROUT ["BZ",DEC A,10];GOSUB SHOW_LCD; GO TO h_DEC AND RETURN;SERIAL dataGOSUB SHOW_LCD; GO TO SHOW_LCD ANDSERIAL dataRETURNHSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];GOSUB SEND;RETURN;if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, ENDRETURN;TRANSMISSIONH_DEC;; LABEL h_DEC; TMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF TIMER1	T1CON.0 = 1; ENABLES TIMER	" ,				
STAY:;";T1CON.0=1;RETURN;PAUSE 1000SEND;; LABEL SEND;T1CON.0=0; TIMER DISABLEDSEND.; LABEL SEND;GOSUB ADC;HSEROUT ["BZ",DEC A,10];GOSUB H_DEC;GO TO h_DEC AND RETURN;GOSUB SHOW_LCD; GO TO SHOW_LCD ANDSERIAL dataRETURNHSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];GOSUB SEND;RETURN;if pb = 0 then; wait for Push button to go highH_DEC.; LABEL h_DEC;TRANSMISSIONH_DEC.; LABEL h_DEC;PAUSE 200;TIMER1		LCDOUT \$FE,\$C0,"V= ",VD[3],"",VD[2],VD[1],VD[0],"				
T1CON.0=1; PAUSE 1000RETURN;T1CON.0=0; TIMER DISABLEDSEND;; LABEL SEND; HSEROUT ["BZ",DEC A,10];GOSUB ADC; GOSUB H_DEC; GO TO h_DEC AND RETURN; GOSUB SHOW_LCD; GO TO SHOW_LCD ANDHSEROUT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; SENDRETURN GOSUB SEND;SERIAL data HSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];return A = 1; PAUSE 200;RETURN;TRANSMISSION PAUSE 200;H_DEC; TIMER1	STAY:;	" ,				
PAUSE 1000SEND:; LABEL SEND;T1CON.0=0; TIMER DISABLEDSEND:; LABEL SEND;GOSUB ADC;HSEROUT ["BZ,"DEC A,10];GOSUB H_DEC;GO TO h_DEC AND RETURN;GOSUB SHOW_LCD; GO TO SHOW_LCD ANDHSEROUT ["AZ,"VD[3],VD[2],VD[1],VD[0],10]; SENDRETURNSERIAL dataGOSUB SEND;HSEROUT ["CZ,"H[4],H[3],H[2],H[1],H[0],10];if pb = 0 then; wait for Push button to go highRETURN;A = 1;CONTROL variable is 1, ENDTRANSMISSIONH_DEC;; LABEL h_DEC;PAUSE 200;TIMER1	T1CON.0=1;	RETURN;				
TICON.0=0; TIMER DISABLEDSEND:; LABEL SEND;GOSUB ADC;HSEROUT ["BZ",DEC A,10];GOSUB H_DEC;GO TO h_DEC AND RETURN;GOSUB SHOW_LCD;GO TO SHOW_LCD ANDRETURNSERIAL dataGOSUB SEND;HSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];if pb = 0 then; wait for Push button to go highRETURN;A = 1;CONTROL variable is 1, ENDTRANSMISSIONH_DEC;; LABEL h_DEC;PAUSE 200;TIMER1	PAUSE 1000					
GOSUB ADC;HSEROUT ["BZ",DEC A,10];GOSUB H_DEC;GO TO h_DEC AND RETURN;HSEROUT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; SENDGOSUB SHOW_LCD;GO TO SHOW_LCD ANDSERIAL dataRETURNHSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];RETURN;if pb = 0 then; wait for Push button to go high A = 1;RETURN;H_DEC;; LABEL h_DEC;TRANSMISSIONH_DEC:; LABEL h_DEC;TMR.BYTE0 = TMR1L ; OBTAIN LOWER REGISTER OFPAUSE 200;TIMER1	T1CON.0=0; TIMER DISABLED	SEND;; LABEL SEND;				
GOSUB H_DEC;GO TO h_DEC AND RETURN;HSEROUT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; SENDGOSUB SHOW_LCD;GO TO SHOW_LCD ANDSERIAL dataRETURNHSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];GOSUB SEND;RETURN;if pb = 0 then; wait for Push button to go high A = 1;H_DEC;; LABEL h_DEC;TRANSMISSIONTMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF TIMER1	GOSUB ADC;	HSEROUT ["BZ",DEC A,10];				
GOSUB SHOW_LCD; GO TO SHOW_LCD AND RETURNSERIAL data HSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];GOSUB SEND;RETURN;if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, ENDH_DEC.; LABEL h_DEC; TMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF TIMER1	GOSUB H_DEC; GO TO h_DEC AND RETURN;	HSEROUT ["AZ",VD[3],VD[2],VD[1],VD[0],10]; SEND				
RETURNHSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];GOSUB SEND;RETURN;if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, ENDH_DEC:; LABEL h_DEC;TRANSMISSIONTMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF TIMER1	GOSUB SHOW_LCD; GO TO SHOW_LCD AND	SERIAL data				
GOSUB SEND; if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END TRANSMISSION PAUSE 200; TIMER1 RETURN; H_DEC:; LABEL h_DEC; TMR.BYTE0 = TMR1L ; OBTAIN LOWER REGISTER OF	RETURN	HSEROUT ["CZ",H[4],H[3],H[2],H[1],H[0],10];				
if pb = 0 then; wait for Push button to go high       RETURN;         A = 1;       CONTROL variable is 1, END         TRANSMISSION       H_DEC:; LABEL h_DEC;         PAUSE 200;       TIMER1	GOSUB SEND;					
if pb = 0 then; wait for Push button to go high A = 1; CONTROL variable is 1, END TRANSMISSION PAUSE 200; H_DEC;; LABEL h_DEC; TMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF TIMER1		RETURN;				
A = 1;       CONTROL variable is 1, END       H_DEC:; LABEL h_DEC;         TRANSMISSION       TMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF         PAUSE 200;       TIMER1	if $pb = 0$ then; wait for Push button to go high					
TRANSMISSION TMR.BYTE0 = TMR1L; OBTAIN LOWER REGISTER OF PAUSE 200; TIMER1	A = 1; CONTROL variable is 1, END	H_DEC:; LABEL h_DEC;				
PAUSE 200; TIMER1	TRANSMISSION	TMR.BYTE0 = TMR1L ; OBTAIN LOWER REGISTER OF				
	PAUSE 200;	TIMER1				

	I			
TMR.BYTE1 = TMR1H; OBTAIN HIGHER REGISTER	mand delay			
OF TIMER1	DEFINE LCD_DATAUS 44 <sup>c</sup> Use 44uS data delay			
FOR $X = 0$ TO 4; START LOOPS	;;			
IN = TMR DIG X; OBTAIN DIGITS	;SET ALIAS			
LOOKUP IN,["0123456789"],OUT; DECODING DIG-	PB VAR PORTA.3			
ITS	LCD1 VAR PORTA.1;			
H[X] = OUT; SAVE DIGITS	LCD2 VAR PORTA.2;			
NEXT X; NEXT LOOPS	;CREATE VARIABLES			
TMR1L = 0; $CLEAR TIMER$	X VAR byte;			
TMR1H = 0;	VIN VAR WORD;			
RETURN;	VIN2 var word			
END;	VD var byte[4];			
	Y VAR BYTE;			
Listing 2: Code for the PIC16F1614 Working as a RF Re-	H VAR BYTE[5];			
ceiver	A VAR BYTE;			
; PBP3 COMPILER from melabs.com	R VAR BYTE[7]			
; Authors: Ricardo Jimenez and Gabriel Lee Alvarez	IN var byte;			
; © July 13, 2020	OUT VAR BYTE;			
; PIC16F1614 for the Receiver module	;			
;Include "modedefs.bas"; include library	;-CLEAR VARIABLES			
OSCCON = %01111010; 16 MHZ	FOR $Y = 0$ TO 5;			
OSCTUNE = 0;	VD[Y] = "0"			
OSCSTAT = %00011111; PLL IS OFF,HFINTOSC AND	H[Y] = "0"			
MFINTOSC IS READY	NEXT Y;			
	X =0;			
DEFINE OSC 16; CLOCK SET TO 16MHZ				
TRISA = %0000000; ALL PINS ARE OUTPUTS	;INITIALIZE LCD			
ANSELA = %000000; ALL PINS ARE DIGITAL	LCD1 = 1; ENABLE $LCD1$			
WPUA = 0; INTERN PULL-UPS DISABLED	LCD2 = 0; DISABLE LCD2			
TRISC = %100000; RC5 INPUT, RC0:RC4 OUTPUTS	PAUSE 10;			
ANSELC = 0; ALL PINS ARE DIGITAL	LCDOUT \$FE,\$28; \$28 FUNCTION SET, 4 BITS			
WPUC = 0; INTERN PULL-UPS IS DISABLED	LCDOUT \$FE,\$10; \$10 SHIFT DISPLAY			
OPTION_REG.7 = 0; WEAK PULL-UPS ENABLED BY	LCDOUT \$FE,\$0C; \$0C DISPLAY ON			
INDIVIDUAL WPUX	LCDOUT \$FE,\$06; \$06 ENTRY MODE SET			
;UART-HSERIN CONFIGURATION	;			
RXPPS = %10101; EUSART CR/RX PORTC.5	LCD1 = 0; DISABLE LCD1			
DEFINE HSER_RCSTA 90h; RECEIVER ENABLED	LCD2 = 1; ENABLE LCD2			
DEFINE HSER_TXSTA 20h; TRANSMITTER ENABLED	PAUSE 10;			
DEFINE HSER_BAUD 2400; 2400 BAUD RATE	LCDOUT \$FE,\$28; \$28 FUNCTION SET, 4 BITS			
DEFINE HSER_CLROERR 1; CLEAR OVERRUN ERROR	LCDOUT \$FE,\$10; \$10 SHIFT DISPLAY			
rc1sta.7 = 1; SERIAL PORT IS ENABLED	LCDOUT \$FE,\$0C; \$0C DISPLAY ON			
;LCD CONFIGURATION	LCDOUT \$FE,\$06; \$06 ENTRY MODE SET			
DEFINE LCD_DREG PORTC; PORTC IS A DATA PORT	;			
DEFINE LCD DBIT 0 ; RC0 IS THE LSB,	LCD1 = 1; ENABLE LCD1			
RC1,RC2,RC3 ARE MSB	LCD2 =1; ENABLE LCD2			
DEFINE LCD RSREG PORTC ' RS IT IS IN PORTC	PAUSE 1;			
DEFINE LCD_RSBIT 4; RS IN RC4	LCDOUT \$FE,\$80,"VOLTMETER":			
DEFINE LCD_EREG PORTA; EN IS IN PORTA	LCDOUT \$FE,\$C0,"RF RECEIVER":			
DEFINE LCD_EBIT 0; RA0 IS EN				
DEFINE LCD_BITS 4 '4; IMES	PAUSE 1000			
DEFINE LCD_LINES 2 ' It is a 2-line display	MAIN:			
DEFINE LCD_COMMANDUS 1500 ; Use 1500uS com-				

```
LCD1 = 1; ENABLE LCD1
 LCD2 = 0; DISABLE LCD2
 PAUSE 1;
 LCDOUT $FE,$80,"----WAITING-----";
 LCDOUT $FE,$C0,"----VOLTAGE-----";
 LCD1 = 0; ENABLE LCD1
 LCD2 = 1; DISABLE LCD2
 PAUSE 1;
 LCDOUT $FE,$80,"----WAITING-----";
 LCDOUT $FE,$C0,"---FREQUENCY----";
 LCD1 = 1; ENABLE LCD1
 LCD2 = 1; ENABLE LCD2
 ;ON INTERRUPT GOTO READ_UART
 MAIN1:
 HSERIN 10, MAIN1, [WAIT("BZ"), STR A\1]; WAIT 1S
TO RECEIVE INSTRUCTION
 if a = "0" then
   LCD1 = 1; ENABLE LCD1
   LCD2 = 1; ENABLE LCD2
   PAUSE 1;
   LCDOUT $FE,$80," STARTING ";
   LCDOUT $FE,$C0," RX ";
   PAUSE 1000;
   HERE START:
   HSERIN 10, JUMP1, [WAIT ("BZ"), STR A\1]
   JUMP1:HSERIN 100,JUMP2,[WAIT ("AZ"), STR VD\4]
   JUMP2:HSERIN 10,HERE, [WAIT ("CZ"), STR H\5]
   HERE:;
   LCD1 = 1; ENABLE LCD1
   LCD2 = 0; DISABLE LCD2
   PAUSE 2;
   LCDOUT $FE,$80,"RECEIVED VOLTAGE";
   LCDOUT $FE,$C0,VD[0],"",VD[1],VD[2],VD[3],"
                                               ";
   HERE3:; LABEL HERE3
   LCD1 = 0; DISABLE LCD1;
   LCD2 = 1; ENABLE LCD2
   PAUSE 2;
   LCDOUT $FE,$80," FREQUENCY ";
   LCDOUT $FE,$C0,H[0],H[1],H[2],H[3],H[4]," HZ
                                              ";
   ;ENABLE
   LCD1 = 1; ENABLE LCD1;
   LCD2 = 1; ENABLE LCD2;
   ;ENABLE INTERRUPT
   if a = "0" then HERE_START
   LCD1 = 1; ENABLE LCD1
   LCD2 = 1; ENABLE LCD2
   PAUSE 1
   LCDOUT $FE,$80," END ";
   LCDOUT $FE,$C0," RX ";
   PAUSE 1000;
 endif
 goto main;
 END
```