Indian Territory Amateur Radio Society

The Beacon

Volume 1, Issue 1 ***May, 2002***

Inside this issue: Of the ITARS Beacon

Presidential QRM

Hello everyone.

We are putting together a first rate club in the Okmulgee County area and need your help.

We have a lot of plans for the future and with the challenges ahead we can use good people for the hard work ahead.

We now have a call of KD5SDJ. We are setting up for everything from storm spotting for our communities to trying to figure out what activities we want to make a part of our tradition.

Everyone is welcome to come to the club meetings and all club members will have a voice that will be heard on where you would like to see this club go.

Keep in mind we want to have fun, make friends and help folks in the process.

If this sounds like something you would like to learn more about.

Come see us.

From the Editor

I am very happy to announce that Saturday, April the 20th, the Indian territory Amateur Radio Society was issued a call sign by the Federal Communications Commission.

As we put in for the call not all the necessary documentation was sent with the application. I spoke with the fine folks at the **W5YI VEC** group who was responsible for processing the application and they were kind enough to download a copy of the necessary documents from the FCC database. We thank them for their patience and help. DE W5TAZ

And So it Begins……...

Well, rumor has it that the first Monday night of February a group of folks got together and started forming what would become the **Indian Territory Amateur Radio Society Of Oklahoma.** We have a long way to go but the "crew" is willing to go the distance to get our club up and running. Centered mostly in Okmulgee County we all have big plans on making this a club where all can be a part of the whole club and no one will be excluded. Everyone, state wide is welcome to join and we plan on having a field day as well as some picnics that are sure to be a great time to be had by all who attend. There is a need for more storm spotting in the area and there is no shortage of community services that need to be fulfilled. Leading this group will be James Strauss, KA5WHM as President, David Killion, KD5OND as Vice President, David, KD5OND's wife Jerri will fill in as the club's secretary while Joel Meeks, KC5UGY will serve as the Club Treasurer. Tom Moore, KD5BOW will serve as the Public Service Liaison Officer with Brian Jones, KD5OII as the Public Safety officer. We still have a few spots that need to be filled and we are looking for just the right person to do the job.

The club By Laws have been drawn up and approved by the executive board, the club license has been applied for and we are officially a club! We will have floating meetings for a while until we land the best site we can get. Membership dues will be $15 per year per person, $10 extra for a family membership. We have a lot of good things in mind and welcome any and all input from anyone who joins in. right now we plan on field day, but I doubt we will have it this year, and a cookout or two per year. The club has a web page up at **http://www.qsl.net/w5taz/itars.html**

ITARS Web Page

ITARS now has a web page up at **http://www.qsl.net/w5taz/itars.html.** The web page is very informative and has a lot to offer. On it you will find a monster link page to a lot of helpful web sites to the radio amateur. Weather, radios, new and used, amplifiers, antenna's and more. A projects page has several antenna plans and there are more to come in the future. The newsletter page has loads of stuff on it. Humor, information, newsletters, membership flyer with the membership application in it and even some scanner frequencies for our area are there. All the club officers with links to the e-mail for those who have it. We even have a "for sale" page for those who would like to post something to sell, or list a want or just window shop at your convenience. And please sign the logbook.

The guy said he was sorry! He said he was running CW mobile! Darned Ham Radio Operators!

The Magic Of Serial Port Communications

I've been tossing around the idea of writing an article on serial ports for quite some time now. No, this is not going to be about the orifice that you put your breakfast food into every morning. This is going to be about the little 9 pin port (sometimes 25 pin) on the back of your PC and, if it's new enough, your rig and any other piece of HAM equipment that supports serial communication. I don't have a clue how this is going to turn out at this point. One thing that I can be sure of, however, is that it will, in some places, be incredibly boring to most people. I don't believe that a good article on serial communications could exclude the boring parts and be effective, however. I am by no means an expert on the subject of serial communication though I do make use of it in my work. I have not tried to cover serial communication in the PIC (or other) micro controllers. That is outside of my knowledgebase (though I am studying). I've borrowed heavily from "Serial Port Complete" by Jan Axelson and numerous tech articles that I use at work to be able to write this article. I hope that you like it. For the purposes of this article, I am going to stick with just two of the serial protocols that are available. RS-232 and RS-485. RS-232 is for shorter, slower links and RS-485 is for longer, faster links. RS-485 also has multidrop capabilities. Ie. You can connect more than two devices (nodes) to an RS-485 network. None of this information is required to be able to use a serial link but it can be useful for design work or troubleshooting. Toward the end, I will list a connection diagrams. That should be somewhat useful.

Formats and Protocols

The devices in a serial link may be different from one another, but all of the devices must agree on conventions and rules for the data that they exchange. This section will introduce the data formats and protocols used in serial links.

Sending Serial Data

In a serial link, the transmitter, or driver, sends bits one at a time, in sequence. Each transmitter may have it's own separate communication path or each transmitter may be required to take turns. When there are three or more devices on the network all three usually share a path and a network protocol determines when each can transmit. All serial links require a clock, or timing reference, to control data flow. The transmitter and the receiver use a clock to decide when to send and when to receive data. The two types of serial-data formats that we will concern ourselves with in this article are synchronous and asynchronous. Each uses the clocks differently. **Synchronous Format**

In a synchronous transmission, all devices use a common clock generated by one of the devices or an external source. The clock may toggle at irregular intervals or have a fixed frequency. All transmitted bits are synchronized to the clock. Each transmitted bit is valid at a defined time after a clock transition, rising or falling edge, cycle. The receiver uses the clock cycle transitions to decide when to read each incoming bit. The exact details of the protocol can vary. Synchronous formats use a variety of ways to signal the start and end of a transmission, including Start and Stop bits and dedicated chip-select signals. Because of the need to transmit a clock signal, synchronous formats are useful for short links, cables 15 feet or less or even circuit board component to circuit board component.

Asynchronous Format

Synchronous transmisssions don't use a clock line. Each end of the link has it's own clock. Each end must agree on the clock's frequency and all of the clocks must match within a few percent of one another. Each transmitted byte includes a Start bit to synch the clocks, and one or more Stop bits to signal the end of a transmitted word. Typical devices that use asynchronous formats to communicate are modems and PC's. One of the more common formats for asynchronous transmission is 8-N-1, where the transmitter sends each data byte as 1 Start bit, then 8 Data bits, beginning with bit 0 (the LSB, or least significant bit), and ending with 1 Stop bit. The N in 8-N-1 indicates that the transmission does not use a parity bit. Parity bits are used, in other formats, as a simple method of error checking. Parity can be Odd, Even, Mark, or Space. RTTY operators may recognize the terms MArk and Space. With Even parity, the parity bit is set or cleared so that the data bits plus the parity bit contain an even number of 1's. With odd Parity so that the opposite is true (data plus parity equal an odd number of 1's). It's important to know that data is transmitted on serial line by alternating voltage from high or 1 to Low or 0. The voltages are low and noise can be a problem. Lets say that you send a data packet that looks like this 00011001. With Even parity, the Parity bit is 1 since there are 3 1's in the data packet, which is an odd number. When you add the 1 from the Parity bit you get Even

The Magic Of Serial Port Communications

parity. The receiver, if set up correctly, will have the same Parity setting and it will expect data + Parity to equal Even. Any added signal, noise, should change the bit pattern therefore triggering error correction algorithms in the hardware or software where applicable. Mark and Space are forms of *Stick* parity: with Mark parity, the parity bit is always 1, and with Space parity it is always 0. These are less useful as error indicators, but one use for them is in 9-bit networks. These networks use a parity bit to indicate whether a byte contains an address or data. A link's bit rate is the number of bits per second transmitted or received per unit of time, usually expressed as bits per second (bps). Baud rate is the nuber of possible events or data transitions, per second. The two values are often identical because in many links each transmission period represent a new bit. High-speed modems, over phone lines, use phase shifts and other tricks ot encode multiple bits in each data period, so the baud rate is actually much lower than the bit rate. All of the bits required to transmit a value, from Stop to Start bit, form a word. The data bits in a word form a character. In some links the characters are actually text characters (letters and numbers). In other links the character values are binary and have nothing to do with text. For instance, the CAT protocol used by my Yaesu FT-847 uses 1 Start bit, 8 Data bits, No Parity bit, and two Stop bits. All of the commands sent from the computer to the receiver consist of five-byte blocks, with 200ms (milliseconds) between each byte. The last byte in each block is the instruction opcode, while the first four bytes of each block are arguements (either parameters for that instruction, or dummy values required to pad the block out to five bytes). As an example, let's say that we want to set the VFO frequency to 439.70 Mhz. Set Frequency is Opcode 0A (set by the P1 command byte). Placing the opcode in the 5th data bit position, we would then enter the frequency into the first four data bit positions. It looks like this (hexadecimal format): Data 1Data 2Data 3Data 4Data 5000097430A Parameters Op Code, In this example the Start bit is sent, then Data 1, then Data 2, etc, etc. until the OpCode is sent. The receiver sees the OpCode and follows the instruction. The CAT DATA BYTE FORMAT looks like this: Start Bit Bit 0 Bit 1 Bit 2 Bit 3 Bit 4Bit 5Bit 6Bit 7Stop BitStop Bit The CAT 5-BYTE COMMAND STRUCTURE looks like this: L.S.D. Parameter 4Parameter 3Parameter 2Parameter 1M.S.D. Command Let me save you some time, 97h and 43h when converted to decimal, do not resemble 439.7 Mhz. The number of characters transmitted per second equals the bit rate times the number of bits in a word. Adding one Startbit and one Stop bit to a byte increases the transmission time of each byte by 25 percent (because there are 10 bits per byte instead of 8). With 8-N-1 format, a byte transmits at 1/10th the bit rate: a 9600 bits-per-second link transmits 960 bytes per second. If the receiver requires a little extra time to accept received data, the transmitter may stretch the Stop bit width to 1.5 or 2 bits. The original purpose of using a longer Stop bit was to allow time for mechanical teletype machines to settle to an idle state.

Transmitting a Byte The Bit Format

A transmitters output is at logic state 1 when it is idle. The output sends a logic 0 for the length of one bit to signal the beginning of a transmission. This is the Start bit. At 300 bps (bits-per-second), a bit is 3.3 milliseconds, while at 9600 bps, it is 0.1 millisecond. The Start bit synchronizes the transmit and receive clocks at the beginning of each new word that is transmitted. At the receiving end, the transition from logic 1 to the Start bits logic 0 signals that a byte is arriving and determines the timing for detecting the following bits. The receiver measures the logic state of each bit near the middle of the the bit. This is done so that the receiver reads the bits correctly even if the respective clocks do not exactly match. More on clocks in a minute. After the Start bit, the transmitter sends the 8 data bits in sequence with bit 0, the LSB (least-significant bit). Then the transmitter sends a logic 1, which functions as the Stop bit. The output reains at logic 1 for at least the width of one bit. Immediatly following this, or at any time after, the transmitter may send a new Start bit 0 to announce the beginning of a new byte. Some interfaces, RS232 for instance, use inverted voltages from those listed above. In that case, the Stop bit is logic state 0 (negative voltage) and the Start bit is positive. The serial port in a PC is an asynchronous port controlled by a UART. The UART (universal asynchronous receiver / transmitter) is responsible for performing the main task in serial communications with computers. The device changes incomming parallel information to serial data which can be sent on a communication line. A second UART can be used to receive the information. The UART performs all the tasks, timing, parity checking, etc. needed for the communication. The only extra devices attached are line driver chips capable of transform-

The Magic Of Serial Port Communications

ing the TTL level signals to line voltages and vice versa. An entire article on the UART could be written. The UART typically uses a receive clock frequency of 16 times the bit frequency. If the data rate is 300 bps then the clock must be 4800 bps. After detecting the transition that signals a Start bit, the UART waits 16 clock cycles (1 bit) for the Start bit to end, then waits 8 more cycles to read bit 0 in the middle of the bit. It then reads each of the following bits 16 clock cycles after the previous one. If the transmit and receive clocks do not exactly match, the receiver will read each successive bit closer and closer to an edge of the bit. To read all of the bits in a 10-bit word correctly, the transmit and receive clocks should vary no more than 3 percent. Any more difference than this, and the timing may be off by so much that it will read bits either before they have begun or after they have finished. The clocks only need to stay in synch for the length of one word since each word begins a new Start bit the resynchronizes the clocks. I've said this twice in this section. It's an important concept. Because accurate timing is very important, asynchronous interfaces require a stable timing reference. Most are controlled by a crystal or ceramic resonator. The frequency of the reference should allow even division by the frequencies the receive clocks use for standard bit rates. In PC's, the standard UART clock frequency is 1843.2 kHz. Division by 16 gives 115,200, which is the top bit rate that the UART supports.

Data Formats

Data bits in a serial transmission may represent anything at all. Sensor readings, error codes, status information, text messages or device commands and responses. The information may be encoded as binary or text data.

Binary Data

With binary data, the receiver interprets a received byte as a binary number with a value from 0 to 255. The bits are conventionally numbered 0 through 7, with each bit representing the bit's value (o or 1) multiplied by a power of 2. For example:

- Bit 0 = BitValue * (2^0)
- Bit 1 = BitValue * (2^1)
- Bit 2 = BitValue * (2^2)

A byte of 1111 1111 translates to 255, or FFh (the small h indicates that the number is hexadecimal format), and 0001 0001 translates to 17 or 11h. In asynchronous links, bit 0, the least-significant-bit (LSB) arrives first. If you are viewing the data on an oscilloscope or logic analyzer, remember to reverse the order when translating to conventional notation of most-significant-bit (MSB) first.

Text Data

Binary data works fine most of the time. Sometimes you
need to send messages or files containing text. For that reason it is also possible to send binary data encoded as text. To send text, the program uses a code that assigns a numeric value to each text. There are several coding conventions. The most common coding convention is ASCII (American Standard Code for Information Exchange), which consists of 128 codes and requires only 7 data bits. An 8th bit, if used, may be 0 or a parity bit. ANSI (American National Standards Institute) uses 256 codes, with the higher codes representing special chaacters. Many of the higher codes of the ASCII text used in the original IBM PC were used by DOS programs for line and box drawing characters. These were used to add simple graphics to text screens and printouts. The Unicode standard uses 16 bits per character and allows 65,536 different characters. This standard supports hundreds of additional alphabets. DBCS (Double-Byte Character Set is an earlier standard tha supports many Asian languages.

ASCII Hex

You can use text to transfer binary data by expressing the data in ASCII Hex format. Each byte is represented by a pair of ASCII codes that represents the bytes two hexadecimal characters.This format can represent any value using only

Continued page 7

ASCII codes 30h to 39h (for 0 thru 9) and 41h to 46h (for A thru
F). Instead of sending a byte to represent a value from 0 - 255, the sending device sends two, one for each character in the hex number that represents the byte. The receiving computer treats the values like ordinary text. After the computer receives the values, it can process the data any way that is necessary, including converting it back to binary data. For example, consider the decimal number *163* Expressed as a binary number, it is *1010 0011* In hexadecimal it is *A3h* The ASCII codes for A and 3 are *41h and 33h* So the binary representation of this value in ASCII hex consists of these two bytes *01000001 00110011* A serial link that uses ASCII hex format would send the value A3h by transmitting these two bytes. An obvious downside to using ASCII hex is that each data byte requires two characters, so data takes twice as long to transfer. ASCII hex has it's uses though. For one, it frees all of the other codes for other uses, like handshaling or an end-of-file indicator. It also allows protocols that only support 7 data bits to transmit a numeric value. I see this quite a bit in my

The Magic Of Serial Port Communications

work. I'm going to end this article with two RS232 Pin function charts. The first is a 25-pin configuration and the second is the 9-pin configuration. I will continue with preventing missed data (Handshaking, buffers, polling an interrupts, acknowledgements, and error checking) and more info specific to RS2323 and RS485 at a later time if this article generates sufficient interest. Pins marked with a * represent signals that are included in a PC interface. DTE stands for Data Terminal Equipment and DCE stands for Data Circuit-terminating equipment. It doesn't matter which device in a link is the DTE or DCE, but each link must have one of each. In a PC / Modem application the DTE is the PC and the modem is the DCE. All of the signal names are from the perspective of the DTE. For example, TD is an output on a DTE and an input on an DCE. Pin NumberCircuit NamePopular NameSourceTypeDescription1Shield----2*BATDDTEdataTransmitted data3*BBRDDCEdataReceived data4*CA/CJRTSDTEcontrolRequest to Send / Ready to Send5*CBCTSDCEcontrolClear to Send6*CCDSRDCEcontrolDCE (Data Set) Ready7*ABSG-commonSignal Common8*CFCDDCEcontrolReceived Line Signal Indicator (Carrier Detect)9----Reserved for Testing10----Reserved for Testing11----Unassigned12SCF/CI-DCEcontrolSecondary Received Line, Signal Detector/data signal rate selector13SCB-DTEcontrolSecondary Clear to Send14SBA-DCEdataSecondary Transmitted Data15DB-DCEtimingTransmitter Signal Element Timing16SBB-DCEdataSecondary Received Data17DD-DCEtimingReceiver Signal Element Timing18LL-DTEcontrolLocal Loopback19SCA-DTEcontrolSecondary Request to Send20*CDDTRDTEcontrolDTE Ready21RL/CG-DTE/DCEcontrolRemote Loopback / Signal Quality Detector22*CE/CKRIDCEcontrolRing Indicator / Received Energy Present23CH/CI-DTE/DCEcontrolData Signal Rate Selector24DA-DTEtimingTransmit Signal Element Timing25TM-DCEcontrolTest Mode9 pin functions- 9 Pin Number 25 PinNumberSignalSourceTypeDescription18CDDCEControlCarrier Detect2*3RDDCEDataReceived data3*2TDDTEDataTransmitted data4*20DTRDTEControlData Terminal Ready5*7GND--Signal sound6*6DSRDCEControlData set ready7*4RTSDTEControlRequest to send8*5CTSDCEControlClear to send922RIDCEControlRing Indicator

73 Mike, **WM5LL**
Email: WM5LL@ARRL.net

Robotron

Reddy Kilowatt

The DX Cat

There are often things that defy explanation. Yesterday, one of the local QRPers made his way up the hill, this one puffing a bit the last few yards. He was carrying a cardboard box with the cover folded over and he set it gently on the table in our shack.

"Want a cat?" he said, getting right to the point and looking us right in the eye. We are not often lost for words, and many will attest to this fact. This time, however, we came close. "A cat!" was all we could get out . . . but we quickly recovered and put forth our best DX face.

"Yes" the QRPer continued, "a cat. And not just any cat either. A real DX cat!" We thought about this for a moment, for it wasn't immediately clear what the difference was between a DX cat and a regular cat. So we carefully lifted the cover of the box and had a look. "Looks like any other cat to us", we said to the QRPer. "What makes this one any different?" At this point we were ready for about anything, including some variant of the tale in QST years ago about the cat that copied CW . . . and we were aware that story had appeared in the April issue.

The QRPer was serious. "I've had this cat for almost five years now. And I'm convinced that at least half of the DX I've worked is a direct result of this cat. He's always in the shack with me. He likes the heat from the amp. Why, he's spent hour after hour laying on top of it, sometimes sleeping, but most of the time he watches me break pileups. He knows when I make a QSO . . . he perks right up when he hears me send RR TU 5NN. A real DX cat! And I'd never have done it if he wasn't there with me."

We had to know more, for instead of answering questions, the QRPer's explanation was generating more. "How does this cat, or any cat for that matter, help you break pileups?" The QRPer was prepared, "You know that in this world of DX, not everything is obvious, and that to be a real DXer, you have to be a believer. And that if something works, like your method of tail-ending or the way you time your transmissions, or any of a dozen other things, then you don't change it. And while you may never be quite sure why, if it works, you keep doing it, right?" We had to agree that this was indeed true. For we too had some techniques that worked, some of which defied logical explanation.

"Well" the QRPer continued, "it's like that with my cat. If he's in the shack, lying on the amp, I usually break the pileup in a call or two. If he's somewhere else in the house, I call for hours. This cat is one of the Mysteries of the Ages, one of the Eternal Enigmas of DXing. I don't know how he does it, but he helps me work the DX." At this point we were still a bit skeptical, but we had learned long ago that the road to DX understanding often took strange turns. Maybe the QRPer was on to something. So we shrugged and nodded in agreement.

We still were confused, so we asked the obvious question, "If this cat is so good, and for whatever reason, helps you blast your way through all these pileups, why are you giving him away?" The QRPer looked us right in the eye and replied, "It's like this", he said, "I've worked a lot of DX with this cat . . . probably got over 150 new ones while he was in the shack. Now, I've been thinking that, in keeping with the amateur's code, this is giving me an unfair advantage. As DXers, we all should help each other, right? And this cat has helped me get my DXCC and then some. So I figured I should let someone else have him for awhile. You've always helped me with DX advice, steered me in the right direction and taught me most of what I know about DXing. And, in appreciation of all this, I'm going to give you my DX cat!"

We didn't know quite what to say. And we really didn't have a chance, for the QRPer was out the door and making his way down the hill, with his hands in his pockets, whistling away. We looked over at the cat. By now he had crawled out of the box and was starting to explore the shack. What could we do? We'd never had a cat before. What does one do with a cat? Especially a valuable one like this, a DX cat. So we got out some milk and gave it to him. The cat was agreeable, drank the milk and then hopped up on our amp, sniffed it a few times and lay down with a sleepy look. Son of a Gun! Maybe the QRPer was right. The amp was on, warmed up and ready for the next DX spot that might show up on the DX cluster. We had this nagging feeling that something wasn't right. We couldn't put our finger on it, but something in the QRPers stride when he'd left seemed a bit too carefree.

So we did what we always did when we were in need of enlightenment. We made our way up the hill and found the Old Timer. He was touching up the base of his tower with some anti-rust paint. We told him about our new cat and the QRPer's newly found desire to share the secret of his DX success. The Old Timer put down his can of paint and his brush, wiped off his hands and looked at us with an amused grin. "This QRPer", he said, "do you recall him working the 3B7 that was on last week?" We had to admit that we hadn't. In fact, now that we thought about it, we remembered the QRPer complaining about not being able to crack the pileup with his 100 watts. "Why didn't he work the 3B7?" we asked the Old Timer, "and why was he only running 100 watts?"

The Old Timer's grin grew broader. "You didn't hear? His amp is in for repairs. His cat has a bladder control problem." A wave of enlightenment swept past us! And it wasn't DX Enlightenment, either! We recalled the cat making himself comfortable on our own amp less than an hour ago . . . and he probably was still there. "Gotta run!" we told the Old Timer as we made our way out the door and back to the shack. Maybe, if we hurried, we could get the cat boxed up and over to that new DXer who had just
moved in from Palos Verdes. For once we were in complete agreement with the QRPer. DXers should help each other. It sure would be in keeping with the amateur's code if we were to share this DX cat with the new kid on the block. Absolutely! Dxers share. Always! That's one of the reasons why DX IS!

73 / DX Paul VE1DX

INDIAN TERRITORY AMATEUR RADIO SOCIETY

P.O.Box 548
Beggs Ok. 74421

Email: davew5taz@juno.com

Visit the ARRL at **www.arrl.org**

Indian Territory Amateur Radio Society Club Officers

President:
James Strauss, KA5WHM
Vice President:
David Killion, KD5OND
Secretary:
Jerri Killion
Treasurer:
Joel Meeks, KC5UGY
Trustee/Chief Operator:
Dave Lugo, W5TAZ
Public Service Liaison
Tom Moore, KD5BOW
Public Safety:
Brian Jones, KD5OII
Newsletter Editor:
Dave Lugo, W5TAZ

KD5SDJ

Indian Territory Amateur Radio Society

Visit us on the web Www.qsl.net/w5taz/itars.html

Membership is $15 ($10.00 additional family members) per year. Article deadline is the 25th of the month. Articles printed in the Bulletin may or may not express the view of the club membership or it's officers. The club is not responsible for any articles or ads that are placed in the Bulletin. Articles may be edited for utilization of space and content. All articles become the property of Indian Territory Amateur Radio Society unless previously copyrighted. Copyrighted articles must have approval of the author before being printed in the Bulletin. Corrections should be sent to ITARS, at Post Office Box 548, Beggs, OK 74421.

Membership Application for the Indian Territory Amateur Radio Society

KD5SDJ

Name _____ Call_____

Address_____

City _____ State _____ Zip _____

Phone _____ Work Phone_____

E-Mail _____

Year First Licensed _____ License Class and Expiration Date _____ARRL Member? _____

I, the undersigned, do hereby agree to abide by the Indian Territory Amateur Radio Society's Constitution and Bylaws, the Federal Communications Commission rules and regulations and support the Indian Territory Amateur Radio Society in, the advancement of amateur radio and public services.

Signature of Applicant _____Date _____Type of Membership _____

Dues received, Treasurer _____ Date _____ Amount Paid Dues:_____

$15.00 Regular Yearly - ($25.00 family membership) ***Also use membership application for change of address.***