An Introduction to Telephone Line Interfacing Using the PIC Microcontroller.

by Ken Boak, Development Engineer, Stirling Services.

December 2001.

Contents.

Chapter 1 Introduction.

This series of notes explains how low cost microcontrollers can be interfaced successfully to the public switched telephone network (PSTN) and used for transferring small packets of data between the microcontroller device and a PC or between two microcontroller devices. This has immediate applications for remote data-logging, home security, home automation, short messaging and numerous other interesting devices.

This tutorial will first describe the basics of telephone line interfacing, giving an explanation of the various voltages and currents found on the line during normal operation. Next the ringing signal, line reversal and the sending and receiving of audio signals to and from the line are covered. An explanation of the gyrator, the key component used to separate the dc power from the audio tones is described in detail, plus how it may be used for sending tones to the line and form the principal part of the line power supply.

The second chapter covers the hardware in more detail giving circuit examples that have been evaluated on a current design. This builds up a picture of how the system sub-circuits interconnect to produce a complete PIC based design for a small Caller ID decoder, messaging terminal or remote datalogger.

Chapter 3 looks at the firmware techniques used to produce high purity sine waves and then combine a pair of frequencies to produce the DTMF dial signals. Chapter 4 builds on the understanding of the tone generation from chapter 3 and presents a description of how the tone generator is used to produce V23 modem tones.

The PIC is capable of receiving and demodulating V23 modem signals using a minimum of external hardware and some interrupt driven firmware techniques. These methods are explained in Chapter 5. No product would be complete without a series of general routines for interfacing and driving LCDs and interfacing to other equipment via a serial port. The routines to do these tasks are described in Chapters 6 & 7.

Chapter 8 puts together all that has been learned in the previous chapters, and presents a design for a simple PIC based device. The RAT_1 is a Remote Applications Terminal based on the PIC 16F84A. It can be further enhanced by fitting a pin compatible PIC16F628, that effectively triples the code space available for the User's application. A complete circuit description and schematic design is presented.

1.1 Telephone Line Basics.

The telephone line is an interesting but hostile environment for the PIC microcontroller. It is essential that the correct interfacing techniques are used, not only to protect the PIC from damage, but equally to prevent situations which would interfere with the functioning of the telephone network.

The telephone line carries both dc and ac (audio) signals at the same time. The dc is used to power the circuitry within the telephone during a conversation, and is derived from a large battery located at the **Central Office (CO)** or in the UK, the **Telephone Exchange**. This battery is a nominal 48V, traditionally made up from lead acid cells.

The telephone line between the CO and the home may be several kilometres long, and has a resistance proportional to the length of the line. The further you are away from the CO, the more **line resistance**. The telephone line is made from a pair of copper wires and usually about 0.5mm in diameter. The resistance of the line varies from 90 ohms to 180 ohms per kilometre. Remember to take into account the fact that the line length is twice that of the actual distance from the central office.

The telephone signals the CO when the handset is lifted, by completing the line circuit by closing a switch known as the **hook-switch.** The expressions **off-hook** and **on-hook** were derived from the days when you literally hung up the receiver on a hook and this operated a mechanical switch. As we will see later, the term hook-switch is now used for the simple transistor circuit is used to connect the device and draw line current. This is known as **looping the line**, or putting the line in its **looped state**.

The circuit between the CO and the domestic phone is known as the **local loop**. When current flows in this loop (during the off-hook condition) it is known as the **loop current** or **line current**. Off-hook is sometimes referred to as the **looped condition**.

When on-hook, only a very small **leakage current** is permitted to flow in the loop. In Europe this is the current which would flow if a 1 megohm resistor were connected across the line. For a nominal 50V **line voltage,** the maximum leakage current can therefore be assumed to be **50uA.** In the USA, the leakage current is stated as the dc current that would be drawn if a 5 megohm resistor were connected across the line. **10uA** may be assumed as typical, although no-one is going to bother too much if you draw 50uA.

When the hook-switch closes, the loop current will flow and this will be in the order of several tens of milliamps. About **20mA** may be assumed to be a typical value. The exact value will depend on the line resistance and the dc resistance of the telephone equipment. This will be illustrated in a following section.

When a few mA of line currrent are drawn from the line, a detection circuit at the CO, recognises that a telephone receiver has gone off-hook, and is effectively signalling the exchange, that it requires attention. Traditionally a relay coil was connected in series with the battery and the telephone. When the telephone handset was lifted it closed the hook-switch, allowing current to flow through the relay and energise the coil thus closing a set of contacts. This then switched in a uniselector to decode the dialled digits. The days of pulsed dialling are long gone in the Western World, but it is worthwhile appreciating how up until about 10 years ago many exchanges were electromechanical.

The CO presents a dial tone to the line, and allocates a dial detection circuit, in readiness for the user to start dialling the number. When this number has been received and decoded by the exchange, it can then start to switch in the correct voice circuits to enable the call to be set up. Although this is a very simplified explanation of the workings of a modern CO, it is sufficient to understand the basics for the purpose of this tutorial.

Summary: The telephone line has two conditions **on-hook** and **off-hook.** The current which flow in these conditions are referred to as **leakage current** and **loop current.** Typical values are **50uA** and **20mA** respectively**.**

1.2 Line Voltages and Line Currents

The line resistance can be determined by measuring the voltage across the line and then putting a variable resistance across the line (starting at about 600 ohms) until the voltage across the line measures about half its original on-hook value. The line resistance and the applied resistor form a simple potential divider, and when half line voltage is reached the line resistance is equal to the applied resistance. By this method I calculated that my line resistance was approximately 660 ohms.

The resistance of the line defines the short circuit or maximum current available to the telephone. This is equal to the battery voltage (48V) divided by the line resistance (660R). In my case, about 72mA. **It is not recommended that you ever apply a short circuit across the telephone lines.**

For a telephone device containing electronic circuitry to function, it needs to derive a dc power supply from the line, and this is known as **line powering**. The equipment has a current-voltage behaviour known as the **dc characteristic** or **dc mask**. This is effectively a plot of line current against line voltage when in the off-hook state.

In Europe, it is mandatory that when a device draws 20mA from the line, that the voltage across the device is less than 9 V.

The telecom device needs approximately 4 or 5 volts to power its PIC microcontroller, and this has to be taken into account from the 9V available in the dc characteristic. There will be other voltage drops in the power supply circuit and the bridge rectifier, and these also must be considered. Fortunately, a PIC micro controller does not need 20mA to operate and will work with only 3 or 4mA. This means that the power supply circuit can afford to waste current in order to meet the dc mask conditions. More on this subject later.

Suffice to say at this time that if the line power supply can give a maximum of 5 V at 5mA , then the PIC will be happy.

1.3 Audio Signals and Modulation.

Telephony works by superimposing an audio signal (voice or tones) of a few hundred millivolts amplitude onto the dc line voltage. This can be achieved by producing a device that draws a current which varies at an audio frequency. The varying current sets up a varying voltage on the line.

To illustrate this, think of a device which looks like a 450 ohm resistor connected to our 48V 660 ohm line. The line current will be 48/(450 +660) = 43.2mA and there will be nearly 19.5V across the device.

Now if the device resistance falls to 425 ohm the current will increase to 48/(660+425) = 44.2mA and the voltage across the device falls to 18.8V.

Now if the device resistance rises to 475 ohm the current will increase to 48/(660+475) = 42.2mA and the voltage across the device rises to 20.0V.

Thus by varying the resistance of the device we can cause the line current to vary from 42.2 to 44.2mA and the voltage swing between 18.8 and 20V.

If we use a transistor circuit to make this "variable resistance" and make it vary in sympathy with an audio frequency we get the desired effect, an audio frequency modulated onto a dc voltage.

The simplest transistor circuit to achieve this modulation is an amplifier made from a single NPN transistor connected across the line in such a way that it can pass a varying current through a resistor connected in series with its emitter. If this resistor is small compared to the line resistance, say 10to 100 ohms compared to 660 ohms then it will amplify with reasonable gain. The resistance of the line acts as the series load for the amplifier to work against.

So with just an NPN transistor and a few resistors we can provide a means of sending audio signals down the telephone to the CO, where they can be detected and switched to complete a voice circuit to a distant telephone or PC modem.

1.4 The Ringing Signal and Line Reversals

The ringing signal is a high voltage ac signal of perhaps 100V peak to peak and 25Hz to 50Hz frequency which is superimposed onto the dc line voltage in order to signal to a telephone that an incoming call is approaching.

Because of its high voltage, it is essential that the ringing signal is kept away from the PIC, or at least divided down to a safe level before connected to a port pin. Large value resistors of between 1M and 10M are used to attenuate the ringing signal and protect the transistor circuitry from is high potential. The capacitors used in the ring detection circuit and line event circuit must be rated at least 200V (preferably 400V) working voltage.

Another signal, in common use, which also presents a large voltage swing, is the line reversal signal. This is used to indicate that a Caller ID message is due and is literally the CO swapping over the polarity of the telephone lines using the equivalent of a DPCO relay. Line reversals are relatively easy to detect using a minimum of hardware.

1.5 DTMF Signalling.

Modern telephones use a pair of audio frequencies to encode the digits used in dialling. The frequencies can be thought of as representing the rows and columns of digits on the standard telephone keypad. Thus if you press 3, you need to select the frequencies associated with row 1 and column 3 i.e. 697 Hz and 1477Hz.

The frequencies need to be accurate to within +/- 1.5% of the following standard frequencies measured in Hz.

		Col 1	Col 2	Col 3	(Col 4)
		1209	1336	1477	1633
			High Group		
Low Group					
Row 1	697	1	2	3	A
Row 2	770	4	5	6	B
Row 3	852	7	8	9	C
Row 4	941	*	0	#	D

Col 4 is not usually present on most telecom equipment - but it is decoded by the CO, and this is useful for sending data as pairs of DTMF digits at a rate of 3 bytes per second.

The DTMF digit needs to be sent for a minimum of 65mS and there should be at least 65mS between the digits. The upper tone is generally set at a level of between 1 and 3dB higher that the lower tone. This is to compensate for the natural low pass filtering of the line and the subsequent attenuation of the higher frequencies. In the Uk and Europe the low tone is typically sent at -11dBVrms and the upper tone sent at -9dBVrms.

IC Devices known as DTMF decoder chips are used to decode the audio tones back into digits so that a micro can interpret them. The devices are quite cheap, but it should be noted that the microcontroller can be made to decode the tones directly using a clever algorithm running in firmware - provided that the micro has a fast enough clock (at least 16MHz). These techniques will be covered in a later chapter.

1.6 Separating the audio tones from the dc line current.

By connecting a simple 600 ohm 2W resistor across the line, you will draw enough line current to trigger the circuits in the CO to present a dial tone to the line and to get ready to receive a string of DTMF digits.

However, any tone you put across the resistor will just be dissipated and very little will be detected by the CO.

It is therefore necessary to separate the dc current from the ac audio signal in such a way that the dc can be to power the microcontroller device and associated circuitry and the ac is unaffected and available for detection.

Traditionally this would have been done using an inductor made from a coil of wire wound on an iron or ferrite core. The dc current would flow through the inductor, thus drawing line current from the CO, but the ac signal would not pass through the inductor and could be tapped off using a capacitor to couple it into an amplifier.

There is now a cheaper and simpler method to do this using a single transistor circuit which simulates the effect of an inductor, in that is passes direct current but blocks the ac tones. This device is known as a **gyrator**. A gyrator can be made for only a few cents compared to the much greater cost and weight penalty of a wound inductor.

One attractive advantage of the gyrator is that it is basically an amplifier and offers the means to modulate audio tones back onto the line, and consequently may save the need for a separate amplifier. This will be fully explained in Chapter 2.

1.7 Line Powering.

As stated in the introduction, the telephone device is permitted to draw a small leakage current in the order of a few tens of microamps when in the on-hook state. As this is permitted, it might as well be exploited, and so most telephones use a high value resistor of between 1 megohm and 10 megohms to charge up an electrolytic capacitor from the line voltage. They then use a gyrator device and regulator circuit contained within the line interface IC to provide a stable dc power supply for the telephone. The capacitor is sometimes called the **reservoir** or **leakage capacitor**. It will have a value of between 100uF and 2200uF. It will also have a zener diode of perhaps 5V1 connected in parallel across it, to prevent the voltage from rising above about 5V. Without the zener, its voltage would rise to about 50V and it could do a lot of damage if discharged suddenly - possibly directly into a port pin on the PIC!

Provided that the PIC uses an average of less that 50uA for its operation, it is possible to completely power the device from the telephone line, with no requirement for batteries or dc adaptors. If the PIC require more than this, then it is possible to build a small voltage converter circuit which will provide up to 300uA and is suitable for trickle charging a NiMH or nicad battery. This circuit will be covered in a later chapter.

Chapter 2 A description of the hardware sub-circuits

2.1 The bridge rectifier, leakage supply and hook-switch circuits.

The bridge rectifier is made from 1A, 400Vdiodes and is capable of withstanding the highest voltage and current conditions ever likely to be present on the line. Either discrete rectifier diodes such as the common 1N4004 type or a low cost 1A bridge rectifier like the General Instruments W004 may be used.

The bridge serves two purposes, firstly to ensure that regardless of the line polarity, only the correct polarity of voltage is applied across the subsequent circuits. The negative terminal of the bridge becomes the signal ground reference terminal and also the ground supply rail.

Secondly the bridge offers a convenient way of sending and receiving audio signals to and from the telephone line.

The incoming lines are often referred to as Line_A and Line_B and these connect to the ac (~) side of the bridge. The ground rail connects to the (-) terminal and the (+) terminal is used to connect to all the power supply and tone detection circuits.

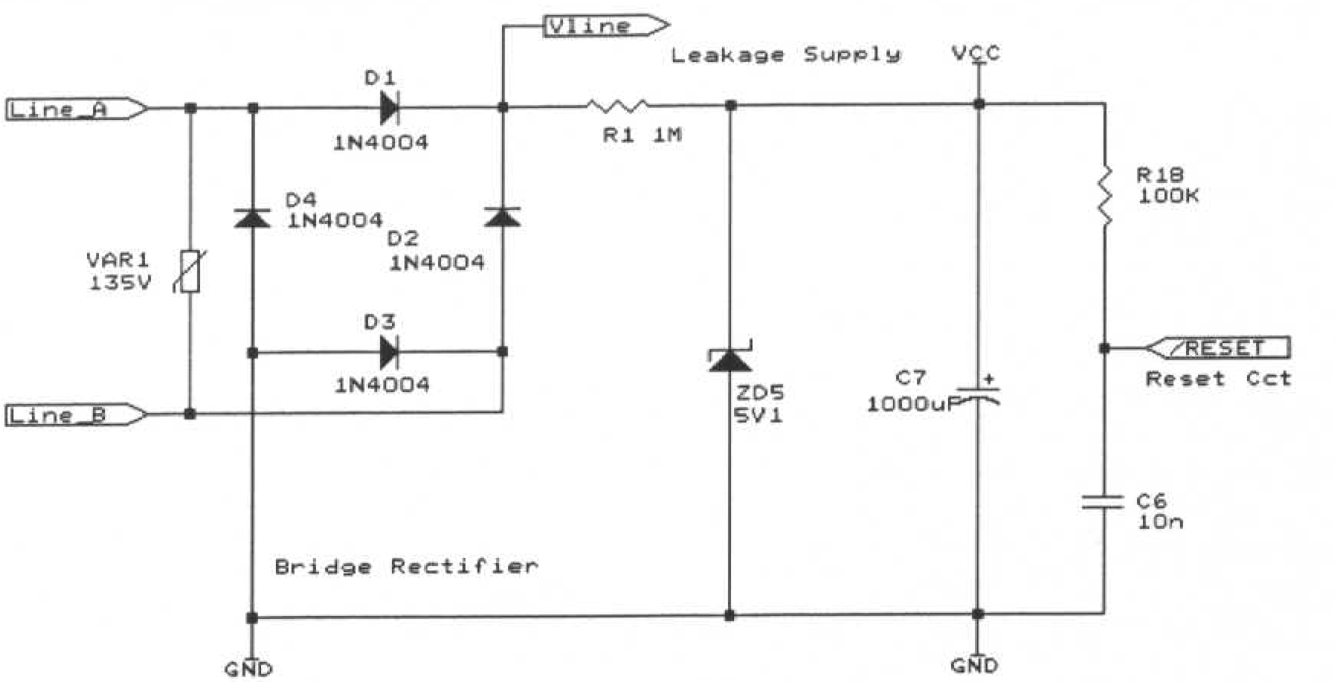


Figure 2.1 Bridge Rectifier and Leakage Power Supply

Now referring to the circuit schematic Figure 2.1.

The incoming telephone wires, Line_A and Line_B are rectified to provide the Vline rail and ground. The varistor VAR1 is a protection device used to absorb voltage transients on the line greater than 135 Vrms. In the United States, an in-line fuse rated at 1A antisurge type and a sidactor transient suppression component are required in place of the varistor. D1 to D4 ensures that the polarity of the voltage is always correct for the later signal detection and processing circuits. The voltage on Vline is always positive with respect to ground and consists of a dc voltage with the ac audio signal or ac ring signal superimposed. The line sensing circuits monitor Vline, for sudden changes in level, or the presence of audio or ring signals and convert these into a form which can be recognised by the PIC micro. Additionally if the PIC wishes to send a signal to the line such as a DTMF or modem tone, it needs first to modulate this signal onto Vline by varying the current drawn by the device. This audio tone passes back through the bridge rectifier and onto the telephone line so that it can be received at the far end.

2.2 The Leakage Supply.

The leakage supply allows a few microamps of current to be drawn from the telephone line in the on-hook condition. This is useful for 2 reasons. It allows the PIC processor to remain powered, but in sleep mode most of the time. It provides a few microamps of power to power the line event detection circuit, which is used to wake up the PIC from sleep mode. This happens when a phone is lifted or when there is an incoming line reversal or ring signal.

Referring to the circuit diagram Figure 2.1 above.

A 1Meg resistor R1, allows a very small current <50uA, to flow into the 1000uF reservoir capacitor C7 and charge it to provide the Vcc rail. The Vcc rail is used to supply the few uA required for the line sensing circuits. The zener diode ZD5 clamps this voltage to about 5V, and prevents an excessive voltage appearing across C7.

A 1Meg leakage supply resistor is used to charge a 1000uF capacitor, a 5V1 zener across the 1000uF prevents the voltage from getting too high. The PIC draws just 5uA when asleep and about 350uA when it wakes.

2.3 The Hook-Switch

The hook-switch isolates the rest of the circuit from the line and prevents any dc from flowing into the power supply section. When the hook-switch is open, no line current will flow and the device presents a very large dc resistance to the line - in the order of 1 megohm, or equivalent to the value of the resistor used in the leakage supply. Traditionally a small relay was used as the hook switch, and these can sometimes be heard clicking in older modems. The advantage with using a relay is that it isolates the telephone line potential from the rest of the circuit - but with a line powered circuit this is not necessary or desirable. It is easier and cheaper to implement the hook-switch using a high voltage PNP transistor driven by a high voltage NPN transistor. A relay may cost 75 cents but the transistor circuit will cost about a tenth of that. Hook switches may also be implemented using FETs or optocouplers. The key point to note is that whatever device(s) is used, they must be able to withstand the high voltage ringing signal, and as such transistors with a 300V collector emitter voltage rating are used.

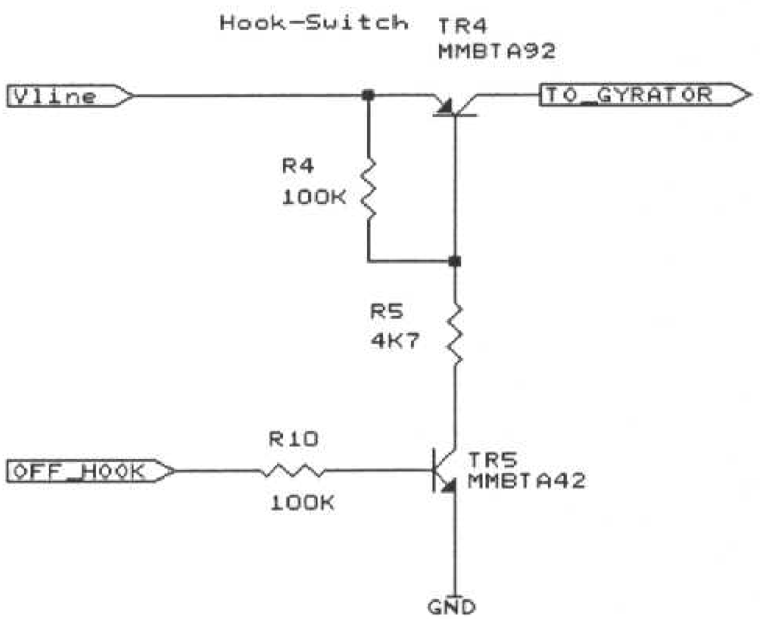


Figure 2.2 The Hook-Switch

Referring to Figure 2.2

The hook-switch is made from a high voltage PNP transistor TR4 which is turned on by an NPN high voltage transistor TR5. TR5 is activated when the OFF_HOOK signal is raised to logic 1. Both these transistors need to have a Vce rating of 300V to survive the high voltages present during the ringing signal. The hook-switch is the means by which the circuit can be effectively disconnected from the line and is key to any modern electronic line powered circuit. When the hook-switch is activated, the RAT_1 will start to draw its own line current - in the order of 20mA to 60mA depending on the line voltage. In this condition it is said to be in the self-off hook mode.

When the hook-switch is closed, the device begins to draw line current in the order of a few tens of milliamps. The value of this current depends on the resistance of the line and the value of the resistance following the hook-switch.

2.4 The gyrator as an inductance simulator and muting circuit

If there is one circuit that can be identified as the key to minimalist PIC telecoms, it has to be the **Gyrator.**

As stated earlier, the gyrator allows the flow of direct current, but offers high impedance to ac signals on the line. As the ac signals cannot flow through the gyrator circuit, they cannot be dissipated in the power supply section and so they can be capacitively coupled into the tone decoding circuit using a small electrolytic capacitor.

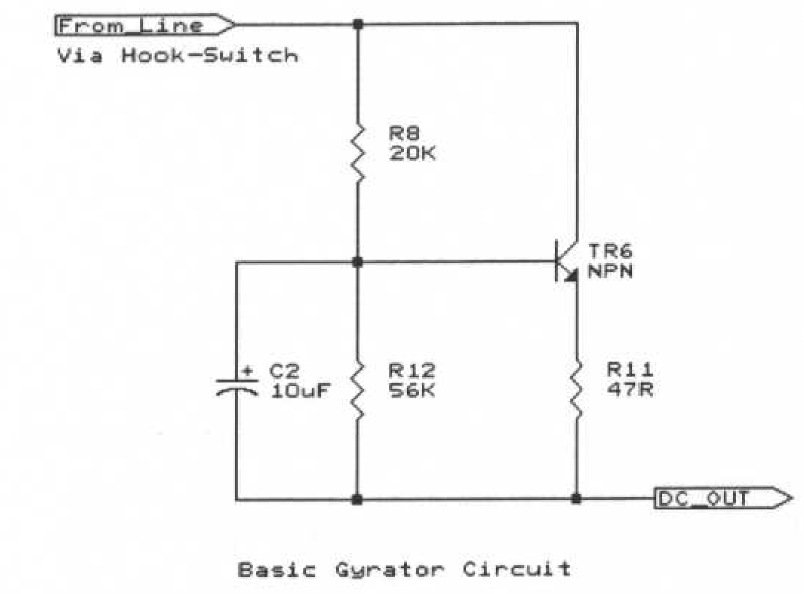


Figure 2.3 Basic Gyrator Circuit.

Referring to Figure 2.3.

A gyrator in its simplest form consists of a current amplifier based on an NPN transistor, shown as TR6. The base of the transistor is held at a particular bias voltage using a pair of biasing resistors R8 and R12 arranged in the usual potential divider layout. An electrolytic capacitor C2 connected in parallel with the lower base resistor is instrumental in controlling the behaviour of the gyrator. To complete the amplifier, a low value resistor of between 10 and 100 ohms is placed in series with the emitter of the transistor R11.

The collector of the transistor is considered to be the input to the gyrator, and current flow is through the transistor and out through the emitter resistor. This dc current is then used to feed a voltage regulator an provide dc power for the microcontroller and its associated circuitry.

As stated earlier, a gyrator simulates a moderate sized inductor, the impedance of which rises with the frequency of the applied signal. The value of the inductance is entirely controlled by the value of the electrolytic capacitor C2. This circuit with C2 as 10uF simulates an inductance of approximately 5H. In simplest terms the gyrator uses the action of the transistor to convert capacitance in the base circuit into simulated inductance in the collector circuit.

If we connect the input to the gyrator to a telephone line via a bridge rectifier and hook switch, (thereby ensuring that the collector of the transistor is kept at a higher potential that the emitter), then the base biasing resistors will hold the base at a given fraction of the collector voltage, and bias the transistor into steady state conduction.

The voltage present at the base is stabilised by the action of the capacitor and can be assumed to be more or less at a fixed voltage. If there is an ac signal present at the collector, then a fraction of this will also try to appear at the base, but the action of the capacitor is to offer a low impedance path away from the base for audio frequency signals. It can be thought that the upper base resistor R8 and the capacitor C2 act as an RC low pass filter and only permit low frequencies and dc levels to appear at the base. The values of R8 and C2 (typically 10K and 10uF) define the cut of point of the low pass filter. The lower base resistor R12 is usually about 56K, and so the impedance of a 10uF capacitor at 2000Hz, being only 8 ohms, offers an almost short circuit to audio frequencies.

As there is no ac current flow into the base, then the transistor does not act as an amplifier to ac signals, and as such there can be no ac signals appearing in the collector emitter circuit. It is this lack of ac signal amplification, which makes the gyrator appear like a high impedance to ac. The impedance increases with frequency, just like an inductor.

The emitter resistor R11 is chosen to allow a modest amount of line current to flow through the gyrator. If the device needs to go self-off-hook (off-hook with no other device, like a parallel connected phone) then the emitter resistor should be 10 or 12 ohms and of about 1W power handling capacity. The NPN transistor should be capable of handling currents of about 2A and the ZTX651 transistor is ideal for this. Such a circuit will happily sink 50 or 60mA of line current, dependent on the line resistance.

Under these off-hook conditions, there will be a voltage drop of approximately 2V across the gyrator.

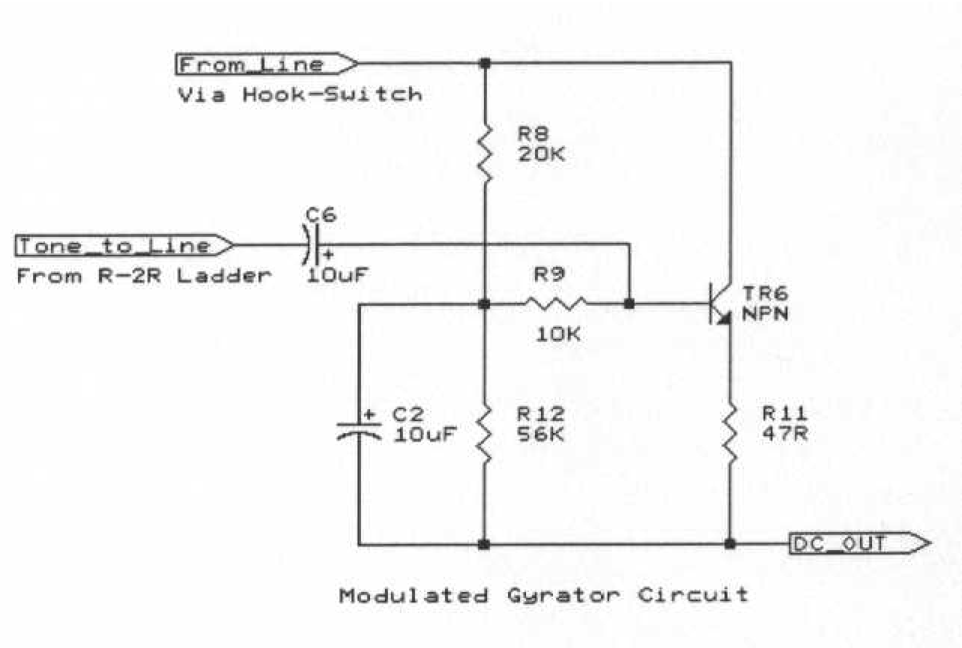


Figure 2.4. The Modulated Gyrator.

2.5 The Modulated Gyrator as a means of amplifying the signals and sending to the line.

As mentioned earlier, the gyrator is fundamentally an amplifier which will amplify the signal at its base and cause a larger signal to flow in its collector emitter circuit which in this case is the telephone line and emitter resistance.

Referring to Figure 2.4

We said earlier that the capacitor on the base of the gyrator transistor, bleeds the ac component away preventing it from being amplified. However, by placing a additional 10K resistor R9 between the capacitor and the base allows an independent audio signal to be applied to the base using a 10uF electrolytic capacitor C6 to couple the signal to the input.

This clever trick allows the gyrator not only to block the ac signal from the power supply circuit but also to amplify audio signals from the DTMF tone generator and present them to the line. This technique is known as a modulated gyrator, and in minimalist designs can save the expense of further circuitry. Referring to the circuit diagram:

TR6, the gyrator transistor also acts as the tone amplifier, the DTMF signal is fed in from the PIC via 10uF electrolytic capacitor C6 and into the base of TR6. This signal is not filtered from the base and causes the transistor to modulate the dc current drawn from the line and thus amplify the tone. To summarise the gyrator blocks the passage of ac but allows dc to flow. It is also a very convenient way of coupling an audio signal into a telephone line using just a single transistor.

2.6 Tone Generation using an R-2R ladder Network

The R-2R resistor network is one of the simplest ways of converting a sampled digital signal back into an analogue voltage.

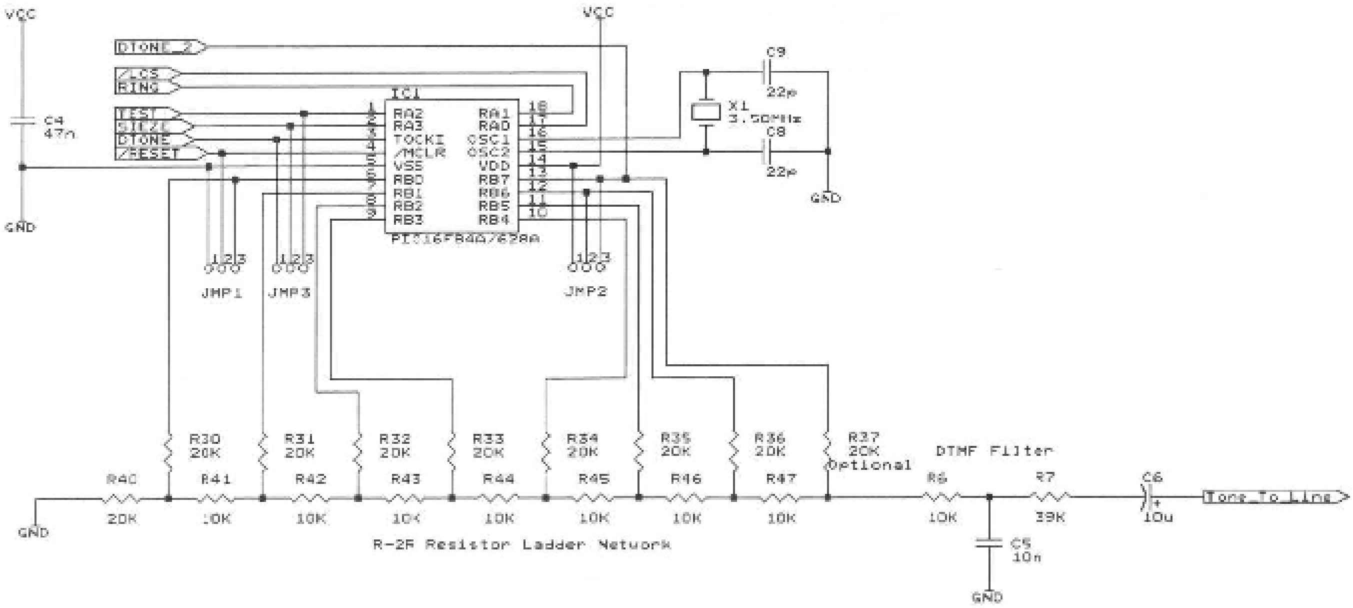


Figure 2.6 The PIC Connected to the R-2R Resistor ladder Network.

Refer to the Figure 2.6.

The R-2R ladder consists of 8 resistors which are 20K in value R30-R37, and 7 resistors which are 10K in value, R41 to R47. The 20K resistors make up the "rungs" of the ladder and connect directly to the 8 pins of the parallel port B of the PIC microcontroller. The 10K resistors which make up the "spacers", connect in a chain between the rungs. The final resistor R40 at the 0V or ground end of the ladder is also 20K.

The resistors form a series of potential dividers which provide an almost linear means of converting the output of Port B into an audio signal. The R-2R ladder network is cheap and fairly compact. Unlike PWM methods of producing audio tones, it needs very little low pass reconstruction filtering , just a single resistor R6 and capacitor C5, and the signal is in a form which can be fed directly into the base of the gyrator transistor via attenuation resistor R7, which controls the level of the signal and capacitor C6.

2.7 Line current sensing

Often it is useful to detect whether a series connected phone has been lifted and this can be done by sensing the line current.

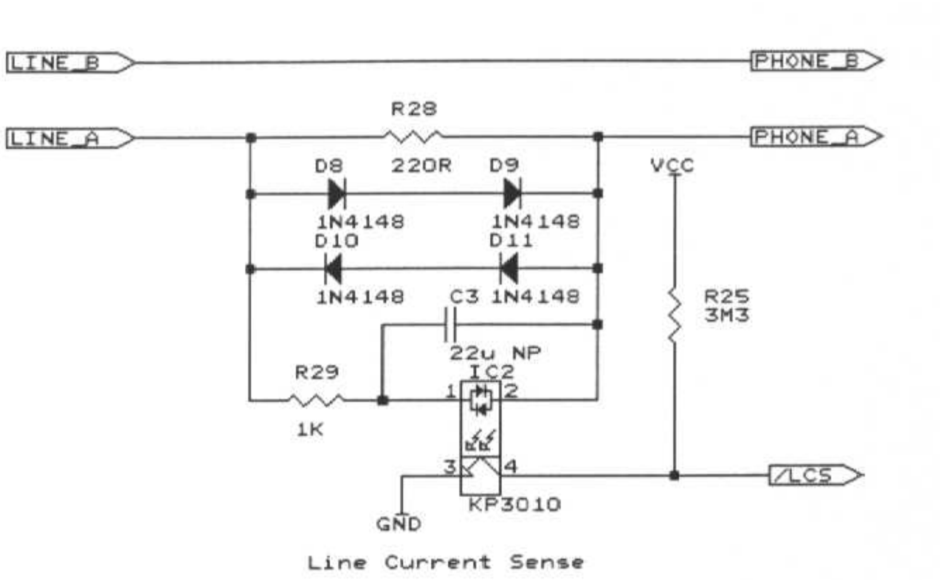


Figure 2.7 Line Current Sensing Circuit.

Referring to Figure 2.7

The Line Current Sense (LCS) circuit is based around a 220R series resistor R28 and an opto-coupler IC2. If a phone plugged into the phone socket SK2 is lifted off-hook then the line current it draws will cause a voltage drop across the 220R resistor R28. This voltage causes the LED in the opto-isolator IC2 to illuminate and turn on the photo-transistor. As a result the /LCS signal goes low. Capacitor C3 and resistor R29 form a low impedance path for ac and prevent the LCS from being falsely tripped by the ringing signal. The opposing pairs of double diodes DD1 and DD2 prevent the voltage across the sensing resistor from exceeding 1.2V - regardless of line current direction.

2.8 The zener clamp and regulator circuits.

Figure 2.8 shows the hook-switch and gyrator connected as a chain network feeding dc current into the switched voltage supply rail Vsw. This rail is only energised when the hook-switch is closed. The clamp and regulator circuits provide a stable, regulated dc supply to power the PIC microcontroller, whilst drawing an acceptable amount of line current. TR7 is a PNP transistor with its base kept at a reference voltage using the zener diode ZD1. The base of the transistor will try to remain at 0.6V or 1 diode voltage drop from the emitter, and so the emitter and the Vsw rail will lie at approximately 4.4V. The excess line current not used by the microcontroller is bled by TR7 and is effectively wasted. Using a transistor here is optional, and by lowering the value of R13 to 47 ohms, the transistor may be omitted.

As stated earlier, for a 20mA line current, the voltage across the device must be less that 9 volts to meet European regulations on dc mask conditions.

Every device in the "power supply chain" from the bridge rectifier, hook-switch, gyrator and regulator will have an associated voltage drop across it. These voltage drops should add up to less than 9V when the feed resistor (line resistance) permits 20mA to flow.

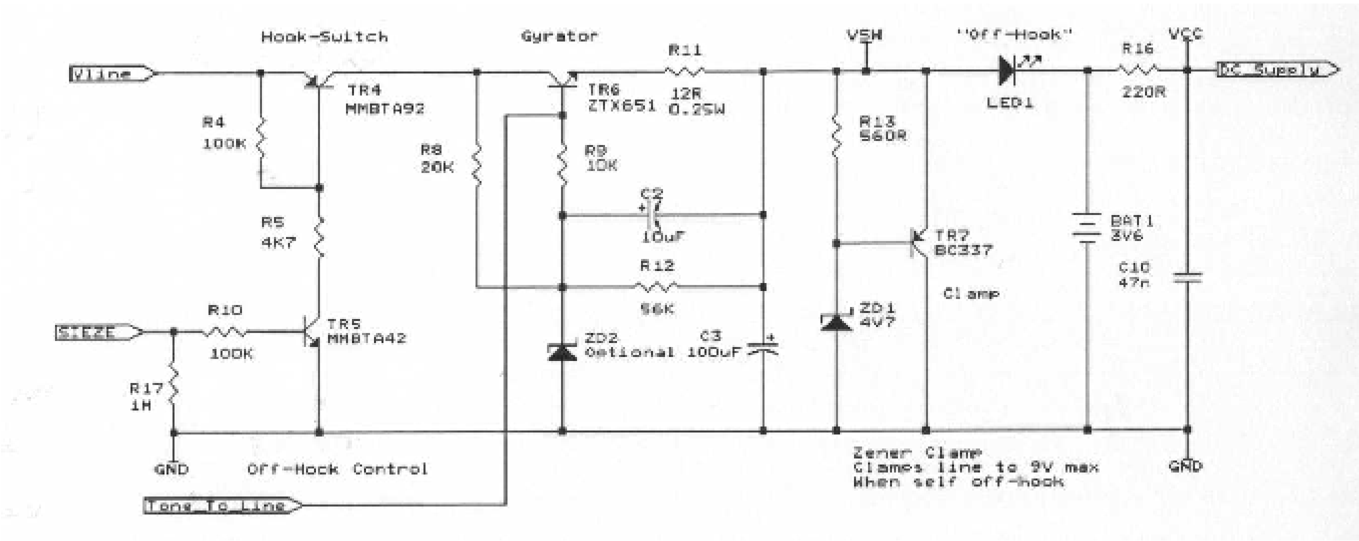


Figure 2.8 The Power Supply Circuit

The zener clamp circuit uses a zener diode and transistor to clamp the voltage behind the gyrator to about 4.4V, which is suitable for supplying the 3.6V for the PIC supply. By keeping the Vdd supply on the PIC to well below 5 volts, helps reduce supply current in the sleep mode. The regulator is not essential and provision is made to supply the PIC from a zener regulated rail.

With 4.4V behind the gyrator, 2V across the gyrator, and about 0.6V across the hook-switch and 1.2V across the bridge rectifier. The zener clamp effectively controls the voltage across the device and keeps it below 9V. The current which is not used by the PIC when it is running (it only needs 3mA) is wasted by the zener clamp circuit.

Refer to Figure 2.8 for the circuit of the power supply section.

The gyrator allows dc current to flow through the 47 ohm resistor R11and charge up the reservoir capacitor C3 to supply the rail Vsw. Vsw is the switched supply as it can be turned on and off by the hook-switch. The transistor TR7, zener ZD1 and R13 form a zener clamp circuit. This prevents the voltage on Vsw from rising above the value set by the zener - in this case about 4.5V (3.9V plus 0.6V for the Vbe of the transistor). By clamping the maximum voltage on the right side of the gyrator, and knowing that the voltage drop across the gyrator will be somewhere between 1.8V and 2.4V, plus the diode drop of the input bridge rectifier allows us to say that the voltage across the lines when the product is off-hook will be under 9.0V. An optional zener diode ZD2 can be added to change the gyrator so that it exhibits a constant current behaviour when reaching a certain line current. This can be useful when the unit has to work in conjunction with other telephones which may be off-hook at the same time.

2.9 The line event detector circuit and frequency detection.

This circuit is the design engineer's dream! It solves all the problems of detecting the various line events and conditions in a very simple circuit based on a general purpose NPN transistor. Moreover, it uses a trickle of leakage current to operate, and so will work well when the device is in the on-hook condition.

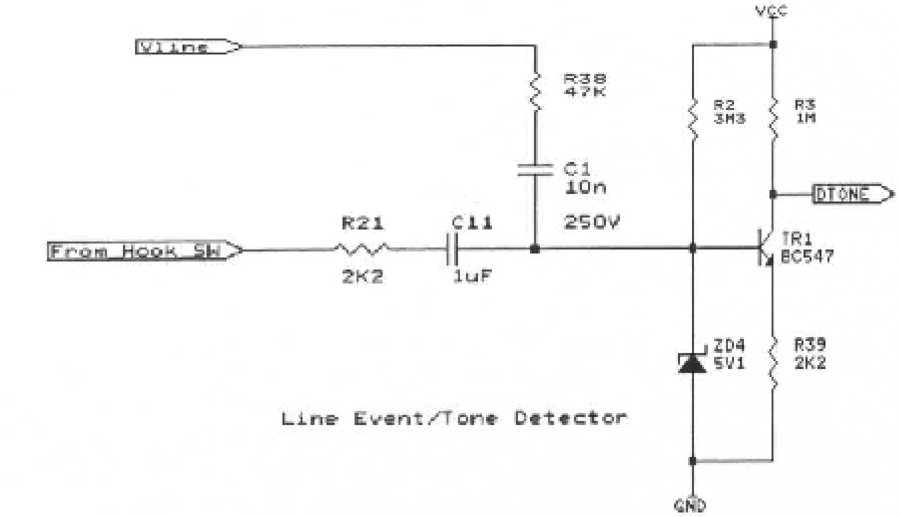


Figure 2.9 Line Event and Tone Detection Circuit

Refer to figure 2.9 for the line event detector circuit

It uses a BC547B transistor TR1, which has an hfe of about 280. The collector is held high to the leakage supply rail using a 1Meg collector resistor R3, and the base is biased with 3M3 base pull-up resistor R2. There is a 2K2 resistor in the emitter R39, but any value around 2K will do. This emitter resistor helps to control the gain of the circuit.

A 10nF (250V) capacitor C1 is used to couple it to the positive side of the bridge rectifier with a 47K series resistor. It has also been tried with a 100nF capacitor which works well but lengthens the pulse times by a factor of 10. The 47K resistor in series with the capacitor was chosen to give good squaring of the input signal when V23 modulation is present on the line during the Caller ID signal.

Any disturbance on the line, when the unit is in the On-Hook state, passes through the bridge rectifier and appears on Vline. The ac component passes through C1 and modulates the base of TR1. This causes the transistor to conduct and produce voltage pulses on the collector as a result of its high gain. These pulses are fed into the PIC port B7 input, which is an interrupt on change input.

When the device is off-hook, the audio tones on the line pass from the hook-switch via R21 and C11 into the base of the tone detector transistor TR1. The value of R21 and C11 are chosen such that a greater signal magnitude is applied to the tone detector than in the on-hook state – i.e. the high value resistor R38 is effectively bypassed for ac signal flow.

The circuit is sensitive to disturbances on the line and produces a high going pulse which is used to wake up the processor through input pin B7 (interrupt on change). It is also connected to the T0CKI input, so that the CPU can use Timer 0 to do frequency measurements on the input signal if a tone signal is applied.

This circuit will respond to the following line events: - producing in each case a high level pulse. When quiescent the collector output from this transistor is low.

i. Unplugging the device from the line - gives 50mS with 10nF or 500mS with 100nF
ii. Lifting a parallel connected phone - gives 25mS with 10nF or about 350mS with 100nF
iii. Plugging a parallel phone across the line - gives 4mS with 10nF or about 8mS with 100nF

In fact almost any sudden change in line conditions will make this circuit respond. The action of the capacitor value is to time-stretch the line transient into a useful length of pulse. For example lifting a parallel phone gives a 350mS square pulse, before the dialtone starts.

This circuit will wake the PIC if the wire is cut - Ideal for burglar alarms, and auto-sensing of parallel phones being lifted which might be useful for bugging devices or conversation recording.

Miscellaneous Points

A small NiMH battery is used to provide current when the PIC is woken up. When the PIC is asleep the battery is trickle charged at about 20uA from the line. It works down to a battery voltage of 2V.

When the PIC asserts the SIEZE line, current flows through the gyrator and raises the Vsw rail to about 4.4V. This forward biases the LED and it illuminates to indicate that the device is self off-hook. When the LED is lit, current will flow into the NiMH batttery and cause it to charge up.

A reset circuit consisting of the usual RC arrangement is used to reset the PIC. R18 and C6 provide a reset time constant of about 1mS.

Chapter 3 Generating High Purity Sine Tones and DTMF on an 8 bit PIC microcontroller.

The DTMF routine described in the PIC application note AN 655, although code efficient has a drawback with the higher frequency tones. The 1477Hz wave table has only 18 samples forming the sine wave, and as such is very crudely quantised in the time domain. As there are only 18 discrete quantisation levels, this represents little better than 4 bit coding. As the quantisation noise is approximately -6N dB where N is the number of bits, then the noise figure can be not much better than -24dB. Ideally all tones should have the harmonic content well below -30dB lower in order to meet the harmonic distortion requirements in European telecoms approvals.

An alternative DTMF algorithm was sought which would improve the high frequency tones and give good frequency accuracy. The solution was to find a series of integers that related the DTMF frequencies.

1)	697	770	852	941	1209	1336	1477

If we take the 1336Hz tone as a reference and allocate a value of 10 then the others are in the following ratios:

2)	5.217	5.76	6.377	7.04	9.049	10	11.055

Although the upper 4 are near integer, the lower 3 are not, so we multiply up by 8 to get a better result

3)	41.73	46.1	51.01	56.347	72.395	80	88.44
4)	42	46	51	56	72	80	88

This will yield the following frequencies fixed relative to the 1336Hz tone.

Actual	701.4	768.2	851.7	935.2	1202.4	1336	1469.6
% Error	+0.631	-0.233	-0.035	-0.616	-0.545	0.000	-0.501

All of these are well within the +/-1.5% allowable frequency error band.

The actual method of generation will involve stepping through an 8-bit sinusoidal look-up table using two pointers that are incremented by the integer step sizes shown in 4). These steps are fairly large compared to a 256 byte long wave table and so for convenience are divided into a series of 4 smaller steps of varying step size

Step 1	10	11	12	14	18	20	22
Step 2	11	12	13	14	18	20	22
Step 3	10	11	13	14	18	20	22
Step 4	11	12	13	14	18	20	22

It will be seen that the lower frequencies have different step sizes e.g. 10,11,10,11. This jigging may be achieved by setting a basic step size of 10 and every other step, adding 1 to produce a step of 11. An elegant method of doing this is to use a single 8 bit register to hold a bit pattern which represents the step modifications - and to rotate the register out through the carry bit C. By testing the carry bit, it is easily determined whether the step size should be incremented or left the same. After 8 cycles the register needs to be reloaded with the original bit pattern.

Eg. 1 0 1 0 1 0 1 0 -> C will alternately increment every other step size.

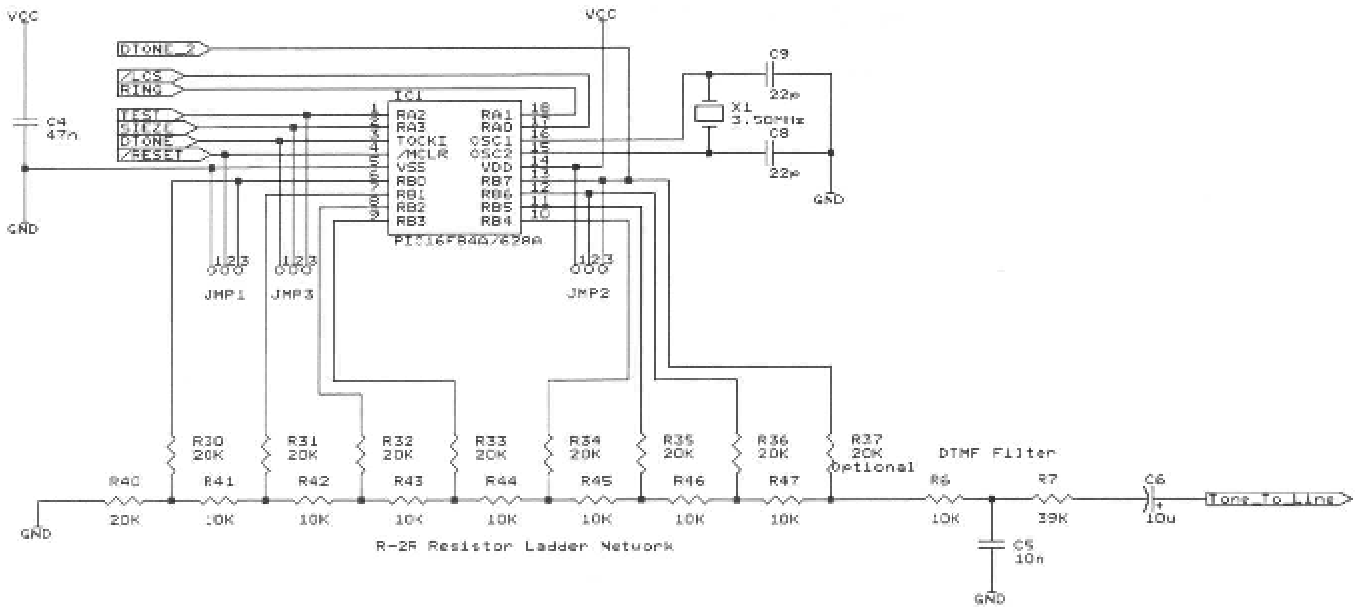
In order to generate the correct 1336Hz tone, the table lookup routine has to run every 20/(256*1336) seconds or 58.476uS. For a PIC running with a 3.579545MHz resonator this allows 52.33 cycles to complete the timing loop. Inaccuracies in clock timing meant that 53 cycles were used in practice.

The routine used the natural overflow of the 8-bit pointer to move from the tail end of the sinusoid back to the beginning. As the step sizes are not true factors of the 256 point table, the restart point will naturally vary from cycle to cycle so a series of different points will be selected for each cycle of sinewave output to the DAC. A Dial tone of 330Hz/400Hz may be generated by using steps of 5,5,5,5 and 6,6,6,6 respectively.

This routine is not limited to generating DTMF, with suitable parameters, it can generate a realistic 350/450 Hz dialtone or other single frequency sinusoidal tones. In the next chapter we will see how the routine is modified to produce the tones required for V23 modem transmission, and how the tone generator is controlled in firmware.

Listing 3.1 shows the complete DTMF generation routine. For it to generate the correct tones the 4 parameters, STEPA, STEPB, ADDERA and ADDERB first need to be loaded from a service routine, shown in listing 3.2. This in turn is called by the dialnumber routine shown in listing 3.3. The tone generation loop takes 53uS to output a sample and is cycled 8 times using a counter SINCOUNT. Another counter, SINCOUNTH is used to ensure that the correct number of 8 sample tone bursts are output – so as to make a complete DTMF digit burst of 65mS.

The PIC Microcontroller Connected to an R-2R ladder network for tone generation



APPENDIX TO CHAPTER 3

Listing 3.1 DTMF Generation Routine senddtmf

```
;*********************************************************************************************
;* This routine generates the DTMF tones at the following frequencies and outputs to Port B  *
;* Crystal Frequency:      3.579545 Mhz                                                     *
;* Sampling Frequency:     16884 Hz                                                         *
;* Instructions/Loop:      53                                                               *
;* Row/Column freq.        <-----Column----->      <-------------Row----------->*
;* Required Frequency:     1209    1336    1477    697     770     852     941 Hz           *
;* Actual Frequency:       1200    1332    1464    700     768     848     932 Hz                *
;* % Error                 0.74%   0.3%    0.88%   0.43%   0.26%   0.47%   0.95%            *
;* Equivalent 4 step:      72      80      88      42      46      51      56                    *
;*                         <----STEP B------>      <------------STEPA---------->*
;* Actual 1 step           18      20      22     10.5     11.5    12.75   14    *
;*                                                                         *
;* It addresses a 128 entry sine lookup table using 2 pointers, POINTERA and POINTERB  *
;* to generate the correct DTMF frequency using a modified phase-accumulation routine  *
;* The pointers are incremented by a specific INTEGER step size STEPA and STEPB        *
;* STEPB is a fixed integer step size, either 18, 20 or 22                              *
;* STEPA is modified every cycle by ADDERA to approximate the fractional part           *
;* Where 0.5 fractional part is required ADDERA =01010101, for 0.75 ADDERA =01110111         *
;* As the modifier is corrupted it is copied into ADDERB so that ADDERA remains intact       *
;* Code Space required 55 locations                                                *
;* Calls: sinelookup - a 128 location half sine wave look-up table                      *
;*********************************************************************************************

senddtmf
             movwf  STEPB               ; get STEPB from W (needs to be loaded 1st time only)

dtmfstart2
             movlw   090h               ; Place 144 decimal into W for loop counter
             movwf   SINECOUNTH                 ; Initialize loop counter to 144 (65mS tone
burst)

             nop
             nop
             goto   firstload           ; load up the octal loop counter

sineA
             nop                        ; add in 6 nops to cancel out extra reload time
             nop
             nop
             nop
             nop
             nop

; This starts the loop properly


loopsine
             movfw  STEPA               ; get STEPA in W

             rrf    ADDERB,1            ; get modifier into carry
             btfsc  STATUS,0            ; check carry
             incf   POINTERA            ; add 1 to POINTERA if carry=1
             btfss  STATUS,0            ; otherwise add nothing (nop)
             nop                        ; Now we have modified step size in w
             addwf  POINTERA,1          ; Add into POINTERA
             movf    POINTERA,W                 ; Place sine wave address into W
             andlw  07fh                ; zero top bit of W so as to get 7 bit index
                                        ; because the look up table is only 128 locations
             call    sinelookup                 ; Look-up first sine wave value

             btfsc  POINTERA,7          ; If the pointer was the second half of the sinewave
             subwf  ALLONES,0           ; calculate (255-W) and put back into W
             btfss  POINTERA,7
             nop                        ; else take the original sample

             movwf   NEXTVALUE          ; Place first sine wave into NEXTVALUE
             bcf    STATUS,0            ; clear carry bit - this is ESSENTIAL!!
             rrf    NEXTVALUE,1         ; get 50% - to get correct level of twist
```

```
            movfw  NEXTVALUE            ; put into W
            bcf    STATUS,0             ; clear carry bit
            rrf    NEXTVALUE            ; get 25%
            nop
;           bcf    STATUS,0             ; clear carry bit
;           rrf    NEXTVALUE            ; get 12.5%
            addwf  NEXTVALUE,1          ; get 62.5% in NEXTVALUE

sineB
            movfw  STEPB                ; Get step size B
            addwf  POINTERB,1           ; Add into POINTERB
            movf    POINTERB,W          ; Place sine wave address into W
            andlw  07fh                 ; zero top bit of W so as to get 7 bit index
            call    sinelookup          ; Look-up 2nd sine wave

            btfsc  POINTERB,7           ; Test which part of sinewave
            subwf  ALLONES,0            ; calculate (255-W) and put back into W
            btfss  POINTERB,7
            nop

            addwf   NEXTVALUE,1         ; Add second sine wave into NEXTVALUE
            rrf     NEXTVALUE,0         ; Divide by 2, Place output into PORTB
dacout
            movwf   PORTB                     ; Update PORTB with new DTMF value

reload
            decfsz  SINECOUNT,F         ; Do we need to reload ADDERB? (every 8 loops)
            goto    sineA               ; Not yet - so go around again


firstload
            movfw  ADDERA               ; get uncorrupted ADDERA
            movwf  ADDERB               ; reload ADDERB
            movlw  08h
            movwf  SINECOUNT            ; reload sinecount

            decfsz  SINECOUNTH,F        ; Dec high loop counter for correct tone burst time
            goto    loopsine            ; Do it again
            retlw   0                   ; Return from sine output
```

Listing 3.2 DTMF Service Routine

```
; ******************************************************************************************
; These routines preload the row frequency parameters for generating the various DTMF tones
; First load up the row by calling the subroutine then jump into the column loading code
; then return (back to the dialnumber routine). The STEPB (column frequency parameter) is
; returned as a literal in w, so must be loaded into STEPB at the start of the DTMF routine
; Code Space required 42 locations
; ******************************************************************************************

key1
      call   row1                       ; load up the row frequency parameters
      goto   col1                       ; go and get the column frequency and return
key2
      call   row1
      goto   col2
key3
      call   row1
      goto   col3
key4
      call   row2
      goto   col1
key5
      call   row2
      goto   col2
key6
      call   row2
      goto   col3
key7
      call   row3
      goto   col1
key8
      call   row3
```

```
        goto   col2
key9
        call   row3
        goto   col3
keystar
        call   row4
        goto   col1
key0
        call   row4
        goto   col2
keypound
        call   row4
        goto   col3

;* These next subs load up the row frequency parameters by putting the stepsize into STEPA
;* and returning with the literal step-modifier which is loaded into ADDERA by the col
routines

row1
        movlw  0ah                          ; 0Ah is the stepsize to load into STEPA
        movwf  STEPA
        retlw  B'01010101'                  ; 01010101 is the step modifier to put into ADDERA

row2
        movlw  0bh
        movwf  STEPA
        retlw  B'01010101'
row3
        movlw  0ch
        movwf  STEPA
        retlw  B'01110111'

row4
        movlw  0dh
        movwf  STEPA
        retlw  B'11111111'

;       These load up the column frequency parameters and return to the dialnumber routine

col1
        movwf  ADDERA               ; load the step modifier (previously loaded into w)
        retlw  012h                 ; return with the step size in w ready to load into STEPB
col2
        movwf  ADDERA
        retlw  014h
col3
        movwf  ADDERA
        retlw  016h
```

Listing 3.3   The Dial number routine

```
dialnumber

call    key0
        call    senddtmf
        call    delay65             ; 65mS delay
        call    key1
        call    senddtmf
        call    delay65             ; 65mS delay
        call    key7
        call    senddtmf
        call    delay65             ; 65mS delay
        call    key3
        call    senddtmf
        call    delay65             ; 65mS delay
        call    key7
        call    senddtmf
        call    delay65             ; 65mS delay
```

```
call     key7
call     senddtmf
call     delay65                    ; 65mS delay
call     key6
call     senddtmf
call     delay65                    ; 65mS delay
call     key4
call     senddtmf
call     delay65                     ; 65mS delay
call     key6
call     senddtmf
call     delay65                     ; 65mS delay
call     key7
call     senddtmf
call     delay65                    ; 65mS delay
call     key1
call     senddtmf
call     delay65                    ; 65mS delay
return
```

Chapter 4 Generating FSK ModemTones using a PIC.

As seen in Chapter 3, the PIC can be made to generate high purity sinusoidal tones and output them to the line via the R-2R resistor ladder.

The next logical extension to this process, is to use the basic tone generation routine to generate modem tones, so that the PIC can communicate via the phone lines with other equipment. The simplest form of modem modulation to generate is FSK or frequency shift keying, where one tone represents a zero and another tone represents logic 1.

The tone generator is controlled in such a way that it seamlessly shifts from one tone to the other without a sudden discontinuity in the phase of the signal. The PIC tone generator described in Chapter 3, has the advantage that it can make frequency changes on the fly, without losing its position in the sinusoidal look-up table, and it is this ability, which allows FSK to be generated simply.

V23 is a commonly used FSK modulation scheme used by low data rate modems. It has the advantage that it is robust and easily produced with inexpensive devices such as 8 bit microcontrollers.

V23 was originally used for Prestel and other Viewdata applications, (Minitel in France) but these have recently been superseded by internet based information services, which being data intensive (verbose), have meant an ever increasing move to faster modulation schemes such as 56kBaud V90 and ADSL techniques.

V23 is used almost universally in Europe for CLI presentation, and as such there has been a proliferation of very low cost V23 decoder chips. Some devices offer V23 and DTMF decoding on the same chip, which makes an attractive proposition for any product requiring to decode both human originated and machine generated information.

A device capable of generating V23 could communicate with other systems over the PSTN at 1200 baud half duplex. V23 has the advantage of simple handshaking on answer, ruugged and robust data transfer and is ideal for devices which only have a small amount of data to transfer. Such products include diallers, short messaging systems, automatic utility meter reading devices, and low rate information retrieval systems.

Described below is a means of generating the necessary V23 modulation using a PIC microcontroller such as the popular, low cost, Flash reprogrammable 16F84A or any of the 18 pin devices.

V23 uses 1300Hz and 2100Hz as the modulation frequencies. The baud rate is 1200Bd.

1) At 1300Hz , the number of cycles produced is (1300/1200) = 1.083333

2) At 2100Hz, the number of cycles produced is (2100/1200) = 1.75

A PIC is used to address a 128 point sine wave lookup table and write the data out to a DAC based on a R/2R resistor ladder. The lookup table may be shortened to 128 bytes and a simple inversion used for the negative half cycle.

A pointer is used to increment through the table to address the next sample to be written to the DAC port. This pointer is incremented by a fixed integer known as the ADDER. The value of the ADDER is previously calculated to obtain the correct frequency output.

The larger the ADDER, the more quickly will the table be stepped through, and so the frequency will be higher. For V23, the ratio of the high and low tone ADDERS needs to be 21:13.

The sinusoidal lookup routine needs to be executed as many times as is required to produce a 1/1200 second burst of sinusoid, as calculated in 1) and 2).

If we use a standard 3.579545MHz resonator for the clock, The PIC instruction time will be 4/3579545 s or 1.117uS.

If we simply choose 13 and 21 as the adders, then we calculate the time allowed in instructions for the sinusoidal lookup routine. This then equates to the sampling frequency Fs.

256 points = 1 cycle which at 1300 Hz is 769.2uS

As we are stepping every 13 samples the number of steps per cycle = 256/13 = 19.692

So the time per step = 1/1300 x 13/256 = 39.0625uS

This gives a sampling frequency of 1/39.0625uS = 25.600 kHz

Dividing this by the instruction cycle time 1.117uS = 34.956 instructions.

So the sinusoidal lookup routine must execute in 35 instructions.

To calculate the number of steps to be output we divide the bit time 1/1200 by the step time or more easily the sampling frequency by the baud rate

=25600/1200 = 21.33.

This means that the lookup routine needs to be executed 21.33 times to get the correct baud rate.

If we round this down to 21, then the baud rate increases to 1219 Baud, which is acceptable.

By reducing the sampling frequency, other values for the adders could be used, which would possibly make compromises to the accuracy of the 1300 and 2100 Hz tones. The two tones are approximately in the ratio of 8:5 so either these or 16:10 or 23:14 could be tried, with a corresponding change in the number of instructions used in the sine lookup routine.

The V23 tone generation firmware consists of the 3 routines shown in listings 4.1, 4.2 and 4.3.

Listing 4.1 Main V23 Modem Code

```
;*************************************************************************************
; This routine is the main modem code
; It takes a character from W, stores it into XmtReg, ready for transmission as V23 tones
; First it sends a low start bit (2100Hz), then it rotates the bits of XmtReg through the
; carry bit, testing them each in turn. Low bits are sent as a high tone of 2100Hz
; high bits are sent as as a low tone 1300Hz.  It finally sends a single high stop bit.
; It calls tonehi and tonelo which in turn call the sendtone routine. 54 locations
; ************************************************************************************

modem

       MOVWF  XmtReg
       movlw   8
       movwf  Count

       call    tonehi            ;Send low Start Bit

nextbit
       bcf     STATUS,C          ;clear the carry
       rrf     XmtReg,Same       ;to send LSB first


       movfw  STATUS
       movwf  STATUS1            ;make a copy of STATUS

       btfsc   STATUS,C
       call   tonelo             ;bit is 1 so send low tone
       btfss   STATUS1,C
       call   tonehi             ;bit is 0 so send high tone

       decfsz  Count,Same
       goto    nextbit
       call   tonelo             ;Send high Stop Bit

       RETURN
```

Listing 4.2 The tone generator control parameter routines

```
tonehi
       movlw  015h               ;21 should give 2100Hz
       movwf  STEPA
       call   sendtone
       retlw  0

tonelo
       movlw  0Dh                ;13 should give 1300Hz
       movwf  STEPA
       call   sendtone
       retlw  0

;************************************************************************
```

Listing 4.3 The V23 Tone Generation routine

```
;*****************************************************************************************
; This routine generates the single sine tone needed for FSK modem communications
; Each sample takes 34 instruction cycles (38uS)
; For V23 communications at 1200 Bd,  21 samples are output for each bit.
; The routine looks up the sample from a halfwave sine lookup table and calculates the
; negative half wave cycle. The output tone is divided by 4 to get a reasonable signal level
; A short delay of 10 cycles sets the correct timing - but could be used for serial input
; 24 locations required
; *****************************************************************************************
sendtone
             movlw   015h               ; Place 21 decimal into W for loop counter
             movwf   SINECOUNTH                ; Initialize loop counter to 21

v23loopsine2cyc
             goto    v23loopsine               ; Waste two cycles to maintain 34 cycle loop
count
v23loopsine

             movfw  STEPA               ; Get step size A
             addwf  POINTERA,1          ; Add into POINTERA

             movf    POINTERA,W                ; Place sine wave address into W
             andlw  07fh                ; zero top bit of W so as to get 7 bit index
                                        ; because the look up table is only 128 locations
             call    sinelookup                ; Look-up first sine wave value

             btfsc  POINTERA,7          ; If the pointer was the second half of the sinewave
             subwf  ALLONES,0           ; calculate (255-W) and put back into W
             btfss  POINTERA,7
             nop                        ; else take the original sample

             movwf   NEXTVALUE                 ; Place first sine wave into NEXTVALUE
             bcf    STATUS,0                   ; clear carry bit - this is ESSENTIAL!!
             rrf    NEXTVALUE,1         ; get 50% - to get correct level of twist
             bcf    STATUS,0                   ; clear carry bit - this is ESSENTIAL!!
             rrf    NEXTVALUE,1         ; get 50% - to get correct level of twist


             movfw  NEXTVALUE           ; put into W

v23dacout
             movwf   PORTB                     ; Update PORTB with new R2R value
;            nop                        ; some nops to get the right delay
             nop
             nop
             nop
             nop

             movlw  120
             subwf  T0_count,0          ; 2100Hz gives 106, 1300 Hz gives 172
             btfsc  STATUS,0                   ; carry is SET IF T0_count > threshhold
             bsf    PORTA,1                    ; set PORTA,1
             btfss  STATUS,0
             bcf    PORTA,1                    ; clear port A,1



             decfsz  SINECOUNTH,F              ; Skip if we are done
             goto    v23loopsine               ; Do it again!
             retlw   0                         ; Return from sine output
```

Chapter 5 V23 Demodulation and Decoding.

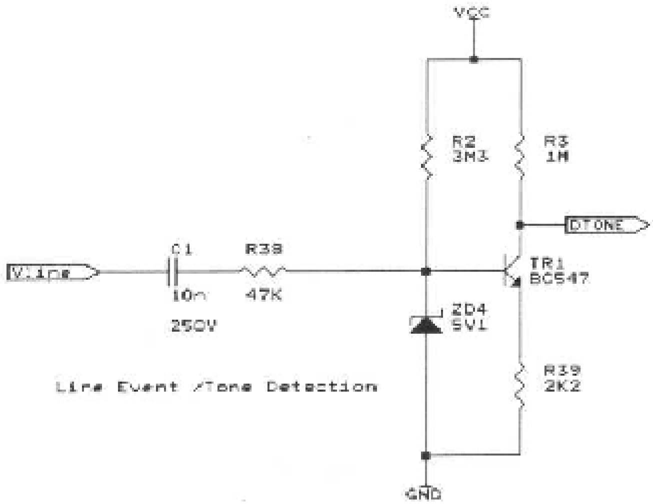
5.1 Line Event Detection and Tone Discrimination.

Any disturbance on the line, generally signifies that something is about to happen.Typical line events include incoming rings, line reversals, local parallel connected phones being lifted or replaced on-hook. All of these events have an associated sudden change in the line voltage and it is this voltage change that the line event detector circuit looks for.

The positive output of the bridge rectifier is at a potential with respect to ground which will be referred to as Vline. If the device is on hook then Vline will be close to 50V but if either the device or a parallel connected phone is taken off hook, then Vline will fall to nearer 10V. It is this sudden change in Vline which can be used to trigger a sensitive transistor amplifier and produce a positive pulse on the interrupt pin of the PIC.

The "interrupt on change" is one of the most useful features available on the PIC, and it is essential that the new user becomes confident in using this powerful technique. It forms the primary mechanism, by which the PIC is first awoken from sleep mode, when a line event occurs, and then it is used to considerable advantage in the subsequent decoding of the V23 Caller ID signal.

5.2 The Universal Detector Circuit



This is a single transistor circuit, which is capacitively coupled to the positive side of the bridge and responds to changes in Vline. These may be large amplitude changes, caused by line reversals or incoming ring, or they may be changes caused by audio tones present on the line.

The input circuit required is a bridge rectifier made from 4 diodes, the signal being capacitively coupled with a 10nF 250V capacitor C1 and a 47K resistor R38 into the base of an NPN transistor TR1 connected as a common emitter amplifier. The value of the capacitor controls the pulse stretching behaviour of this circuit when triggered by sudden fluctuations in line voltage. The zener diode ZD4 protects the base from large voltage swings caused by the ringing signal effectively clamping the base to 5V.

A general purpose NPN transistor type BC547B has been found to provide satisfactory performance. It has a gain (hfe) of about 280. There is a 3M3 pull-up resistor on the base and a 1M pull-up on the collector. A small amount of resistance (2K2) in series with the emitter can be used to lower the gain if needed.

The signal measured at the base should be about 200mVpp. The collector output goes directly into Port B7 (pin 13) with no other filtering. The input signal is also connected to the T0CKI pin, so that this feature may be used for simple tone detection.

The incoming signal into the base of the transistor is effectively inverted and squared by the high gain of the transistor. In quiescent state, the output of this device is logic 0, but as soon as a line event occurs, the output rises to Vcc, and the change in input level to port B7, causes an interrupt.

This interrupt is used firstly to awaken the PIC from its low power sleep mode, and once awake, the interrupt is used in conjunction with Timer 0, to time the half period (between rising and falling edges) of the input signal, thereby allowing the frequency to be discriminated.

The next section explains how this technique may be used to detect an incoming call and decode the accompanying Caller ID message, by demodulating the V23 tones.

5.3 V23 and Caller ID Decoding on a low cost PIC Microcontroller.

This section outlines the way in which a PIC microcontroller such as the 16F84 can be used with minimum additional components to decode V23 and Caller ID signals from the telephone line.

V23 is a CCITT standard used for low data rate modem data communications over the public switched telephone network (PSTN).
It has been adapted in many countries as the data standard for conveying small packets of data over the PSTN and applications include Caller ID presentation and the sending and receiving of short messages (SMS).

V23 at its simplest level uses two tones; 2100Hz and 1300Hz to represent the low and high logic levels in an asynchronous serial bit stream. Logic 0 is known as space and logic 1 is known as mark. When no data is being transmitted, continuous 1300Hz mark tone is sent.

V23 allows data to be sent in 1 direction at 1200 baud and each ascii character is made up from a low start bit, 8 data bits LSB sent first, and a high stop bit. Thus with 10 bits in the octet, V23 can transfer data at 120 characters per second, which is ideal for short messages.

Decoding of a V23 modulated signal back into a 1200 baud serial data stream is a task which can be readily achieved using a low cost PIC microcontroller, using a minimum of resources.

The way in which this is achieved is an extension to the usual method of frequency measurement using the onboard 8-bit timer T0, to time the period of the signal.

First the incoming signal has to be buffered and amplified in a way which preserves a mark-space ratio of close to unity. This means that the signal fed into the input port needs to be close to a square wave. This is done using a simple high gain transistor buffer amplifier connected in common emitter mode. The output of this amplifying stage approximates a square wave and is of an amplitude suitable to be applied to the input pin.

The PIC is configured to generate inputs on a changing logic level applied to the upper 4 pins of port B. These interrupts are used to control the counter/timer T0, so that it measures the half-period of the applied signal waveform in suitable units of time. Using the counter pre-scaler set to a divide ratio of 4 was found to be suitable for this application.

The timer T0 is set to count “quad instructions” until an interrupt is received. The count present is read into a ram location T0_count and the counter is then cleared to restart its counting process. This continues until the next interrupt occurs. The value stored in T0_count is then equal to the half-period of the waveform.

A frequency of 2100Hz (space) gives a half period of 53 decimal, 1300Hz gives a half period of 86 decimal. By testing bit 6 of T0_count (to compare the T0_count with 64 decimal) it is possible to distinguish between the two frequencies. If you then set the Port A1 pin to the same logic level as T0_count bit 6, the demodulated serial data will appear on Port A1.

The demodulated data can then be passed out to a Psion 3c or similar device, running a 1200 baud terminal.

For this to work well, the mark-space ratio of the incoming signal needs to be close to 1:1.

The PIC is wired in the usual way using a 3.579545 MHz ceramic resonator, simple RC reset circuit and running from a batter of 3.6 to 4.8V. An LCD display could be added to Port B to display the incoming number or message.

5.4 The Caller ID Message Packet Format

Typically the Caller ID packet will have a set of timings starting with the incoming line reversal and ending with the first ring. This is as they were measured in reality - not as per BT technical notes SIN 242, SIN227

Event	**Occuring Between**	**Duration**	**Comment**
Line reversal	0 -29mS	29 mS	Triggers B7 interrupt to wake up the PIC from sleeping
Silence 1	29 - 250 mS	221 mS	
Alert tone	250-350 mS	100 mS	Mix of 2300Hz/2750 Hz tones - can be ignored
Silence 2	350- 600mS	250 mS	
Uppercase U's	600 - 850mS	150 mS	Upper case U's as a training signal looks like 1,0,1,0 etc
Message	850- 1200mS	350 mS	
Silence 3	1200-1700mS	500 mS	500mS for the PIC to match the number & answer the call
First ring	1700 mS ->		

Listing 5.1 V23 Demodulator Routines

```
;******************************************************************************************
; V23 demodulator code
; This code uses timer 0 to time the pulse widths of the signal applied to the B7 pin
; It stores the time (in half clock cycles) in T0-count
; The count is compared with certain limits to see if the input signal frequency matches
; that of V23 high tone 2100Hz or V23 low tone 1300Hz
; Port pin A1 is used to signify 2100Hz or 1300Hz and can thus be taken as the demodulated
; serial output.  Port A2 is used to examine the output of the demod routine
;******************************************************************************************

demod_init

       movlw  0FFh
       movwf  ALLONES

       clrf    PORTB                    ; Init output latches for port B to 0
       movlw  B'10000001'         ; set port B as output except bit 7, 0
       tris   PORTB               ; Set all of PORT B to outputs


       MOVLW  B'00010001'         ; Set A0, A4 as inputs, A1, A2, A3 as output
       tris   PORTA

; set up timer 0 with a divide 4 on the prescaler

init_rtcc

       bsf    STATUS,RP0
       movlw  B'11000001'         ; weak pull ups off, rising edge interrupts, internal clock/4
       movwf  OPTION_REG
       bcf    STATUS, RP0
       clrf   TMR0                ; start timer 0

;      movlw  B'10010000'         ; enable INT interrupts
       movlw  B'10001000'         ; enable port B change interrupts
       movwf  INTCON

; Demodulate the incoming tones into 1200 baud asyn serial bits and put them out on port A2
; The demod routine (plus an allowance for the ISR) should take 833uS to run, to set the right
; 1200 baud bit time.
; this next routine re-times the bits into 833uS chunks

demod

       movlw  120                 ; Is sum >165 then mark
       subwf  T0_sum,0            ; 2100Hz gives 106, 1300 Hz gives 172
       btfsc  STATUS,C            ; carry is SET IF T0_count > threshhold
       bsf    PORTA,2             ; set PORTA,2  send the bit to the "Test Pin"
       btfss  STATUS,C            ;
       bcf    PORTA,2             ;

       goto   continue

       movfw  T0_last
       subwf  T0_count,0          ; if T0_count > T0_last then mark
       btfsc  STATUS,C
       bsf    PORTA,2

       movfw  T0_last
       subwf  T0_count,0          ; if T0_count > T0_last then mark
       btfss  STATUS,C
       bcf    PORTA,2

continue
       nop
       nop
       nop
       nop
       nop

       movlw  215                 ; this gives 833uS delay
```

```
      movwf        bit_count           ; hold the space for 833uS

redo_A
      decfsz bit_count,Same
      goto   redo_A
      goto   demod               ; this whole loop goes every 833uS - as near as possible
```

Listing 5.2 Interrupt Service routine (ISR)

```
;*************************************************************************************
;This is the Interrupt service routine for determining the frequency on pin 13  Port B7

ISR
;*********************************************************************
;
;      CONTEXT SAVE
;
;*********************************************************************

;Save STATUS and W registers into RAM before servicing timer interrupt

C_SAVE MOVWF  W_TEMP
       SWAPF  STATUS,W
       BCF    STATUS,5      ; ENSURE BANK0 SAVE
       MOVWF  STATUS_TEMP

;      BTFSS  INTCON,INTF   ; Test for external interrupt
;      GOTO   C_RESTORE

       BTFSS  INTCON,RBIF   ; Exit ISR if it wasn't a port B change
       GOTO   C_RESTORE

;      btfsc  PORTB,7       ; Test for rising edge interrupt
;      goto   clr_rbintf    ; exit if falling edge


; The signal applied to pin B7 causes an interrupt on change on both rising and falling edges.
; On receipt of an interrupt we stop the T0 counter and store its value into T0_count reg.
; We then clear the counter to set it counting again. We use the divide by 4 prescaler
; 1300Hz gives a half period of 86 in T0_count, 2100Hz gives a T0_count of 53
; So if we test bit 6 of T0_count we can determine whether the signal is mark or space.
; and set Port A1 pin or flag accordingly.  Constant mark will keep the port pin high.
; T0_sum contains the sum of the last 2 half periods - so can be used to qualify the frequency
; For this to work well, the signal needs to have close to equal high and low periods. If the
; signal  M/S ratio varies too much from 1:1,  the 1300Hz low period may be falsely
; interpreted as 2100Hz etc.

Changepin_isr

       movfw  T0_count      ; get the last count
       movwf  T0_last       ; save it

       movfw  TMR0          ; read Timer 0

       movwf  T0_count      ; save it in T0_count for next time
       clrf   TMR0          ; start the timer again

       btfsc  T0_count,6    ; If T0_count <64 then lower Port A1-  a space
       bsf    PORTA,1
       btfss  T0_count,6    ; If T0_count >=64 then raise Port A1 - a mark
       bcf    PORTA,1

       addwf  T0_last,0     ; get the sum of last and current in W
       movwf  T0_sum        ; put it in T0_sum for testing


clr_rbintf

       movf   PORTB,1       ; read port b to clear the mismatch
       BCF    INTCON,RBIF   ;clear port B changed interrupt bit
;      BCF    INTCON,INTF   ;clear external interrupt bit


;********************************
;
; Context Restore
;
;********************************

C_RESTORE
```

```
BCF    STATUS,5      ;ENSURE BANK0 RESTORE
SWAPF  STATUS_TEMP,W
MOVWF  STATUS
SWAPF  W_TEMP,F
SWAPF  W_TEMP,W

RETFIE
```

Chapter 6 Asynchronous Serial Communications

Listing 6.1 Asynchronous Serial Receiver Routine

```
;*********************************************************************
;RX    equ     2          ; PC-Link Receive Pin ( Bit 2 of Port A )
;TX    equu   3           ; PC-Link Transmit Pin ( Bit 3 of Port A )

;      Following code sets the delay times for the selected baudrate
;      X is the unknown delay
;      Values below are for 1200bd on a 3.579545MHz resonator
;      BAUD_4 is half the reqd 1.25B delay so delay4 is repeated

BAUD_1  equ     .248     ; 3+3X = CLKOUT/Baud
BAUD_2  equ     .247     ; 6+3X = CLKOUT/Baud
BAUD_3  equ     .123     ; 3+3X = 0.5*CLKOUT/Baud
BAUD_4  equ     .150     ; 3+3X = 1.25*CLKOUT/Baud
BAUD_X  equ     .245     ; 11+3X = CLKOUT/Baud
BAUD_Y  equ     .246     ; 9 +3X = CLKOUT/Baud
;
;*************************************************************

;* Serial Communications
;* Code space requirement 56 locations


;*************************************************************************************
;* Talk routine waits for low start bit on PORTA,0 before continuing with receiver routine
; Note Port A2 RX needs 100K pullup resistor.
;*************************************************************************************

talk

      clrf   RcvReg               ; Clear all bits of RcvReg
      btfsc  PORTA,RX             ; check for a Start Bit on local RX pin (norm btfsc)
      goto   User                 ; delay for B  (833uS for 1200Bd)
      call   Delay4               ; delay for 1.25 B
      call   Delay4               ; delay for 1.25 B

; Receiver Routine

Rcvr

      movlw   8                   ; 8 Data bits
      movwf   Count

R_next
      bcf     STATUS,C
      rrf     RcvReg,Same         ; to set if MSB first or LSB first
      btfsc   PORTA,RX
      bsf     RcvReg,MSB

      call    DelayY
      decfsz  Count,Same
      goto    R_next

      movf    RcvReg,0            ; Get Received character in w

      return

User:

      movlw   BAUD_3
      movwf   DlyCnt

redo_3
      decfsz  DlyCnt,Same
      goto    redo_3

      goto    talk                ; Loop Until Start Bit Found
```

Listing 6.2 Serial transmission Routine.

```
;*****************************************************************************************

; Transmitter Routine has has stop/start and data bits inverted for MAX232 level converter

TRANSMIT

      MOVWF  XmtReg              ; Copy W into the transmit reg.
Xmtr

      movlw   8
      movwf   Count
      bcf     PORTA,TX                 ; Send Start Bit
      call    DelayB

X_next
      bcf     STATUS,C
      rrf     XmtReg,Same              ; Send LSB first
      btfsc   STATUS,C
      bsf     PORTA,TX
      btfss   STATUS,C
      bcf     PORTA,TX
      call    DelayX
      decfsz  Count,Same
      goto    X_next
      bsf     PORTA,TX                 ; Send Stop Bit
      call    DelayB

      RETURN

;***************************************************************************************
;* Various Delays for serial timing                                                     *
;* For variable delay, w can be loaded with delay constant and then call routine "save"*
;* Subroutine returns after given delay of 5 + 3N cycles                                *
;***************************************************************************************

DelayY
      movlw   BAUD_Y
      goto    save
DelayX
      movlw   BAUD_X
      goto    save
Delay4
      movlw   BAUD_4
      goto    save
DelayB
      movlw   BAUD_1             ;833uS for 1200Bd
      goto    save
Delay2
      movlw   BAUD_2
save
      movwf   DlyCnt
redo
      decfsz  DlyCnt,Same
      goto    redo
      retlw   0
;*********************************************************************************************

Send_Char    retlw  00
```

Chapter 7 The RatCore Firmware and Utility Routines

Listing 7.1 The HEXOUT Hexadecimal Dump Routine

```
;*********************************************************************************************
; This routine converts a byte to 2 ascii characters to sends to terminal
; It also sends a space


Send_Char     retlw  00

HEXOUT
       MOVWF  ASCIIBYTE     ;copy w to save it for later



       ANDLW  B'00001111'   ;Mask off top 4 bits
       ADDLW  0X30          ;Add 30H

       MOVWF  KEYBASE       ;temp store
       SUBLW  0X39          ;test each accumulator against 39
       BTFSS  STATUS,C      ;
       CALL   ADJUST        ;Modify the ASCII for ALPHA

       MOVF   KEYBASE,W
       CALL   TRANSMIT      ;send first nibble to serial port
       MOVF   KEYBASE,W
       CALL   Send_Char     ;send first nibble to LCD

       MOVF   ASCIIBYTE,W   ;get upper byte back
       SWAPF  ASCIIBYTE,W   ;get upper nibble in low half

       ANDLW  B'00001111'   ;Mask off top 4 bits
       ADDLW  0X30          ;Add 30H
       MOVWF  KEYBASE       ;temp store

       SUBLW  0X39          ;test each accumulator against 39
       BTFSS  STATUS,C      ;
       CALL   ADJUST        ;Modify the ASCII for ALPHA

       MOVF   KEYBASE,W
       CALL   TRANSMIT      ;send second nibble to serial port
       MOVF   KEYBASE,W
       CALL   Send_Char     ;send second nibble to LCD

       MOVLW  0X20          ;ASCII space

       CALL   TRANSMIT

       RETURN

ADJUST

       MOVF   KEYBASE,W           ;If character >3A add 7 to make alpha
       ADDLW  07H
       MOVWF  KEYBASE

       RETURN


delay                             ; This gives a delay of 99.75mS
       movlw 057h

delay2                            ; a variable delay of  1.15mS x W
       movwf Count
       movlw 00h
       movwf COUNT
delay3
       nop
       decfsz COUNT1
       goto delay3
       decfsz Count
       goto delay3
```

```
        retlw  0ffh

delay1s                          ; This is a 1 second delay

        call   delay250
        call   delay250
        call   delay250
        call   delay250
        return
```

Chapter 8 The RAT Remote Access Terminal

RAT_1S is an experiment in low data-rate telecommunications using low cost PIC Microcontrollers.

Introduction.

Using a microcontroller with an 8 bit parallel port, such as the PIC16F84A, it is possible to make a high purity tone generator, capable of synthesising two tones simultaneously. This forms the basis of an automatic DTMF telephone dialler unit constructed at minimum cost with a low parts count.

The tone generator hardware is also capable of generating low data-rate FSK modem tones, such as used in the V23, Bellcore GR-30 or Kansas City (Tape) standards.

With these simple techniques the PIC dialler can dial up a remote computer system and send blocks of data to it . One application might be a remote weather station or thermometer, which gathers readings of temperature every 15 minutes, then once a day uploads them to a main PC system using nothing more than the public phone lines.

RAT is an acronym for Remote Access Terminal, an open ended amateur construction project for those wishing to experiment.

There are 3 variants of the RAT.

RAT_1S is the simplest possible design intended for a very low cost implementation using the PIC16F84A. It is an introduction to telephone line interfacing techniques and contains 4 of the principal line sensing circuits plus a DTMF and V23 tone generator.

It is designed for applications such as a simple slave remote datalogger, with very small application code space. RAT_1 may be enhanced by fitting a 16F628 processor allowing the possibility of a real time clock for time of day datalogging, and the benefits of additional I/O (Port B is released from DTMF generation by using the PWM option to generate the audio tones).

It is also possible to decode V23 modem signal using an interrupt driven signal demodulation routine.

All the key signal points and I/O are made available on RAT_1 so that it can be used as just a line interface for more sophisticated designs. It is also possible to slave the RAT_1 on to an I2C bus or TTL serial interface of another processor so that RAT_1 can be used as a telephone dialler stage for a larger system. RAT_1 can easily be built on a piece of 0.1" stripboard sized 6" x 4" and leave plenty area for the users application hardware.

The RAT_1 circuit consists of several building blocks, some of which may not be needed in the final design. This modular construction allows the user to just populate the parts required.

RAT2 is the midrange unit, and would also benefit form the additional codespace and features offered by the 16F628 microcontroller. It has a relay based muting block, which allows signalling to and from other units over telephone extension wiring. RAT2 will accept a NPC 8223 Caller ID and DTMF decoder IC, greatly increasing its functionality for little additional cost.

RAT3 is the top of the range unit utilising all the features of the PIC16F877 microcontroller and the benefits of an isolated RS232 interface, a keypad/display interface and an external 32k x 8 EEPROM for data storage. RAT3 could be used as a standalone master unit for co-ordinating a local cluster of RAT1/RAT2 units.

8.1 Circuit Description.

Referring to the circuit diagram Figure 8.1.

The circuit of RAT_1 is laid out in 3 distinct zones running left to right across the page. At the top are the telephone line jacks SK1 and SK2 allowing the RAT_1 to be plugged into a normal telephone line via jack SK1 and have a telephone plugged in locally into SK2 the phone connector. The jacks should either be RJ11 type (6 pin Western Electric) or chosen to suit the country specific variants. SK1 and SK2 are wired together and incorporate a line current sensing circuit. This LCS circuit is used to alert the PIC micro if the phone or modem plugged into SK2 has gone off-hook and is drawing line current.

The second part of the circuit is the analogue electronics required to process the various signals. This is shown laid out in a long strip, starting with the bridge rectifier on the left and working through the various stages of ring detector, tone detector, parallel phone detector, hook-switch, gyrator, zener clamp and reset circuit.

Following the positive output of the bridge, Vline, there is a leakage supply used to charge a capacitor up through a 1M resistor. This produces the main on-hook power Vcc rail. Most of the line sensing circuits derive their power from Vcc, and so are fully operative when the RAT_1 is on-hook or sleeping.

When the RAT_1 is activated, its hook-switch transistor is turned on and a switched power rail Vsw is turned on. Vsw is used to power the PIC microcontroller.

The PIC is activated by some external event which causes the hook-switch to be closed and the Vsw rail be energised. The PIC then resets and starts executing instructions. One of the first instructions will be to raise Port A, 2 the Seize line, which keeps the hook switch transistor turned-on. In this way the PIC will remain active until Port A2 is lowered and the PIC turns off its own power rail. In this simple design the PIC is turned off until required, and is only active when the RAT_1 is self-off hook.

The bottom section of the circuit diagram shows the PIC microcontroller with is 3.58MHz crystal oscillator circuit and the R-2R resistor ladder network. This ladder network is a very low cost means of producing a digital to analogue converter using just 16 resistors connected to the Port B of the microcontroller. It is used to produce the audio tones needed for DTMF dialling and V23 modem modulation. It can however be used to produce any waveform either generated by calculation or synthesised from a look-up table in ROM.

The basic telephone line interface consists of the following stages:

Bridge Rectifier made from diodes D1 to D4 ensures that the polarity of the voltage is always correct for the later signal detection and processing circuits. The incoming telephone wires, Line_A and Line_B are rectified to provide the Vline rail and ground. The voltage on Vline is always positive with respect to ground and consists of a dc voltage with the ac audio signal or ac ring signal superimposed. The line sensing circuits monitor Vline, for sudden changes in level, or the presence of audio or ring signals and convert these into a form which can be recognised by the PIC micro. Additionally if the PIC wishes to send a signal to the line such as a DTMF or modem tone, it needs first to modulate this signal onto Vline by varying the current drawn by the device. This audio tone passes back through the bridge rectifier and onto the telephone line so that it can be received at the far end.

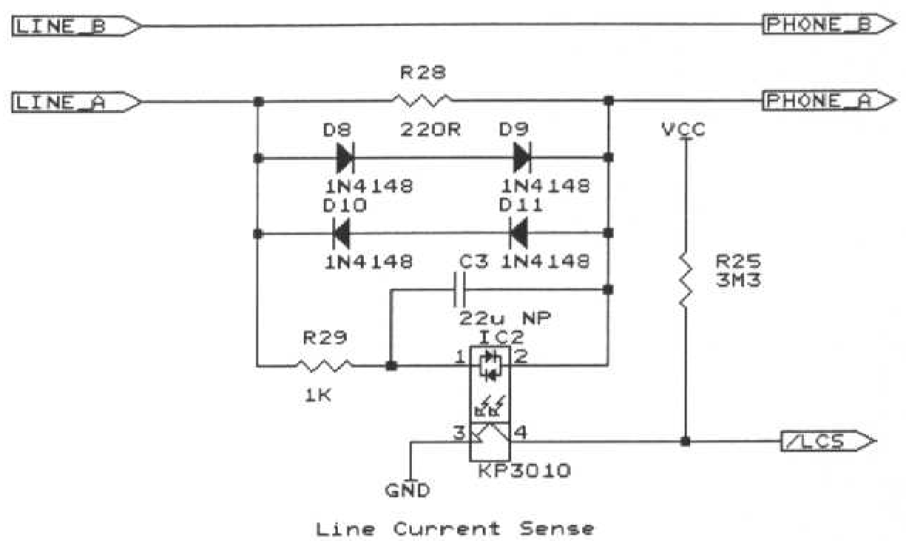
The hook-switch is made from a high voltage PNP transistor TR4 which is turned on by an NPN high voltage transistor TR5. Both these transistors need to have a Vce rating of 300V to survive the high voltages present during the ringing signal. The hook-switch is the means by which the circuit can be effectively disconnected from the line and is key to any modern electronic line connected circuit. When the hook-switch is activated, the RAT_1 will start to draw its own line current - in the order of 20mA to 60mA depending on the line voltage. In this condition it is said to be in the self-off hook mode.

The purpose of the **gyrator** is to obtain a source of dc power from the telephone line and use this current to power the RAT_1. It also has the key function of preventing the ac audio signals from being dissipated in the dc power supply section. The gyrator is made from an NPN transistor TR6, a ZTX651 or any equivalent capable of handling about 2A peak collector current. It serves two distinct purposes. Firstly a gyrator circuit is an electronic means of simulating inductance, and in this respect it acts as a high impedance to ac signals in the audio range. This prevents audio tones from being sunk in the dc power supply stage. The second function of the gyrator is to allow audio tones generated by the PIC to be amplified to a suitable level to ensure correct DTMF dialling. As a gyrator, the transistor is biased by resistors R8 and R12, and C10 connected to the base effectively keeps the base at near constant voltage. Any rising ac signal appearing across R8 and R12 is low pass filtered by the combination of R8 and C10 and causes the transistor exhibit a high impedance to the rising signal. As an amplifier, the DTMF signal is fed in from the PIC via C6 and into the base of TR6. This signal is not filtered from the base and causes the transistor to modulate the dc current drawn from the line and thus amplify the tone. To summarise the gyrator blocks the passage of ac but allows dc to flow. It is a very convenient way of coupling an audio signal into a telephone line using just a single transistor.

Power Supply Section. The gyrator allows dc current to flow through the 47 ohm resistor R11and charge up the reservoir capacitor C11 to supply the rail Vsw. Vsw is the switched supply as it can be turned on and off by the hook-switch. The transistor TR7, zener ZD1 and R13 form a zener clamp circuit. This prevents the voltage on Vsw from rising above the value set by the zener - in this case about 4.5 (3.9V plus 0.6V for the Vbe of the transistor). By clamping the maximum voltage on the right side of the gyrator, and knowing that the voltage drop across the gyrator will be somewhere between 1.8V and 2.4V, plus the diode drop of the input bridge rectifier allows us to say that the voltage across the lines when the product is off-hook will be under 9.0V. An optional zener diode ZD2 can be added to change the gyrator so that it exhibits a constant current behaviour when reaching a certain line current. This can be useful when the unit has to work in conjunction with other telephones which may be off-hook at the same time.

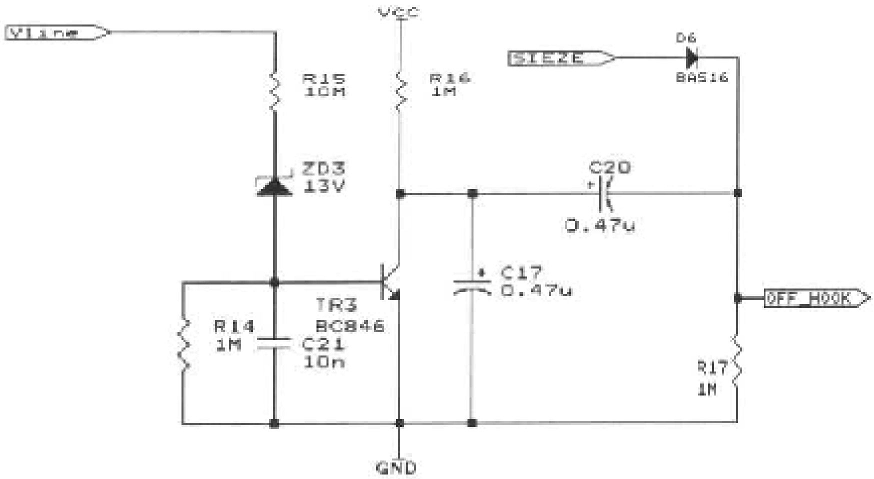
The Leakage supply. This allows certain parts of the circuit to be powered even when the dialler is on-hook. A 1Meg resistor R1, allows a very small current <50uA, to flow into the reservoir capacitor C14 and provide the Vcc rail. The Vcc rail is used to supply the few uA required for the line sensing circuits. Zener diode ZD5 clamps this voltage to about 5V.

Line Sensing Circuits. This design offers 4 useful line sensing circuits:



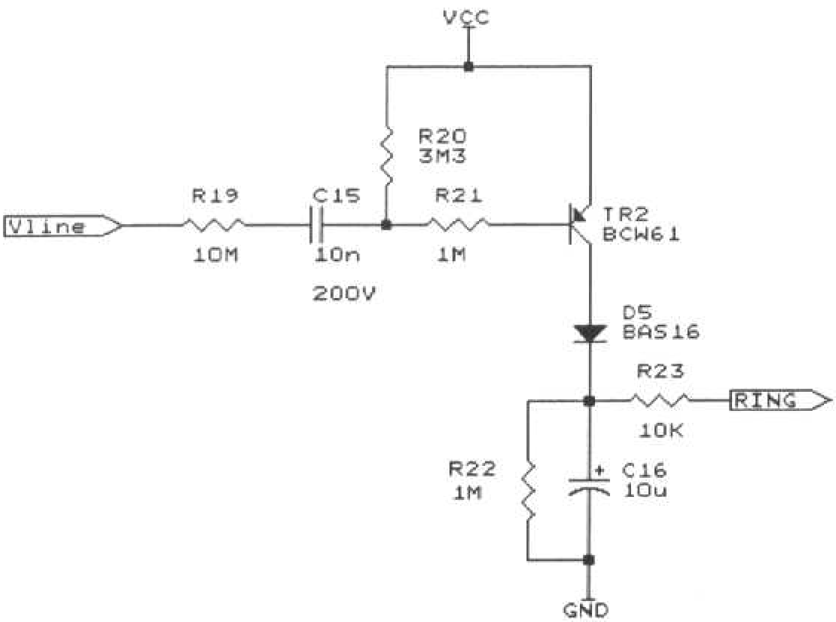
Line Current Sense (LCS)

If a phone plugged into the socket SK2 is lifted off-hook then the line current it draws will cause a voltage drop across the 220R resistor R48. This voltage causes the LED in the opto-isolator IC2 to illuminate and turn on the photo transistor. As a result the /LCS signal goes low. Capacitor C3 prevents the LCS from being falsely tripped by the ringing signal. The opposing pairs of double diodes DD1 and DD2 prevent the voltage across the sensing resistor from exceeding 1.2V - regardless of line current direction.



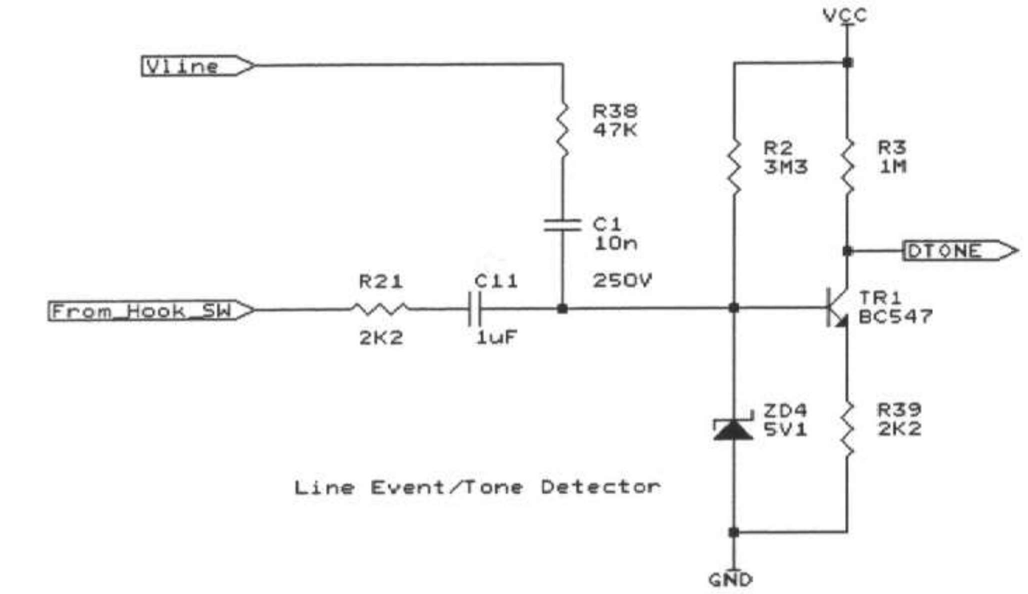
Parallel Phone Detect   Off-Hook Control

The Parallel Phone Detector circuit allows the unit to detect a phone connected in parallel when it goes off-hook. When there is 50V or so on the line, this is sufficient voltage to breakdown the zener voltage of diode ZD3 and turn on transistor TR3. When a phone is lifted, the line voltage drops to below 15V and ZD3 no longer conducts. TR3 is turned off causing a pulse of current to flow through C20. This current pulse can be used to signal the PIC or as in this case, it is used to turn on the hook-switch which effectively powers up the PIC from the Vsw supply rail. The IC starts executing code, and the first thing it does is to raise its SIEZE line. This keeps the hook-switch closed an the PIC in the powered state.



Ring Detect (latching)

The incoming ring detector looks at the voltage on the line, and if a ring signal is present, current flows out of the base of TR2 on the negative half of the ring cycle, causing the PNP transistor TR2 to conduct and charge up the capacitor C16 through the diode. The voltage on this capacitor is sensed by the PIC, so when someone answers the phone (waking up the PIC), it can check the level of Port pin RA1, and see whether the capacitor is charged - signifying an incoming ring. The elegance of this circuit is that the PIC does not need to be awake when the ring occurs, it just senses the after effect, so the ring detector is in that sense latching. After having sensed the ring, the port pin can be set to logic low to discharge the capacitor. This circuit can then be used as a means to cancel operation. If in the case of the auto-dialler the user does not want auto-dial, the PIC can be programmed to charge up the capacitor on first pick-up. To cancel the receiver is replaced, and when next lifting the receiver, as a result of the charge on the capacitor, the PIC will self cancel.



The Line Event & Tone Detector Circuit.

This is truly a multipurpose circuit, able to wake up the processor from sleep mode on a wide variety of line events or disturbances. The circuit can be tuned using the value of the capacitor and series resistor to respond to various events. These include:

Incoming ring or line reversal
Parallel connected phone being lifted
Phone being plugged in parallel
Disruption in line - line cut or device being unplugged, ideal for alarm circuits

It can sense incoming rings, it can sense a line polarity reversal - used in the UK to indicate the arrival of a Caller ID packet. It can sense the call progress tones such as dial tone, engaged tone, busy tone, modem answer tone, unobtainable tone etc. It can also be used to detect Caller ID and V23 modem tones. It evolved out of the desire to be able to do simple signal processing using a very low power amplifier circuit using a minimum of parts. TR1 is an NPN transistor which is base biased by the 3M3 resistor R2. The zener diode ZD4 clamps the bas to a safe voltage - particularly if there is an incoming ring. The collector output of TR1 will produce an approximate squarewave output for most audio tones and modulations down to an input signal level of -24dBm.

The PIC Microcontroller.

This can be a 16F84A or better still a 16F628, which has twice the application memory space. Port A is generally used for line condition sensing and line control. Port B is almost entirely used for generating DTMF and modem tones using the R-2R resistor ladder network. The R-2R ladder network is low pass filtered via R45 and C5 and fed to the line via gain resistor R46 and ac coupling capacitor C6.
Port B0 can be reserved as the external interrupt pin and this is used for sensing the frequency of the tones present on the line. Port B0 can be used to drive the lowest bit of the resistor network, but may be connected to the DTONE signal via a jumper between JMP1 and JMP3.

Port A is used for sensing the line conditions via LCS, Ring and DTONE. A spare TEST input on port RA2 can be used for serial diagnostics or serial input. It is brought out to an unused pin on the BT phone connector SK2. The jumper pins JMP1,2,3 provide a convenient method of connecting a cable to allow in circuit reprogramming of the flash series PICs. A standard low cost ceramic resonator is used to provide the 3.579545 MHz oscillator.

8.2 Applications

8.3 Enhancements

The basic design can be enhanced in a number of ways depending on the application. One of the most obvious enhancements is to use one of the larger PIC processors, such as the PIC16F877. This device has 8K of ROM space, 384 bytes of RAM, 32 I/O lines, 6 channel on-chip ADC and a USART, plus numerous other features. It immediately has the resources to scan a keypad, drive an alpha-numeric LCD and even record and playback audio sample from the telephone line, which could be used in answerphone and voice mail applications.

Some of these enhancements have been incorporated into the RAT_3 design which is the subject of another paper.

8.4 Conclusions

9.0 Glossary of Terms

10.0 Appendices