AN1687

A FULL-FEATURED WIRELESS INTERFACE FOR RS-232 COMMUNICATIONS

Prepared by; Paul Sofianos
       Motorola, Inc., WSSG RF/IF Applications Engineering

INTRODUCTION

This application note describes a full–duplex, wireless data communication link targeted for RS–232 applications. An encoding technique has been designed which addresses many of the problems incurred when attempting to implement the RS–232 (EIA–232) standard, including but not limited to: hardware flow control, the DC component of the transmitted signal, automatic synchronization from host to slave and error detection. The design emulates a RS–232 null modem cable for computer–to–computer communications.

The actual design was realized with standard SSI logic from the high speed CMOS family (MC74HCxxx), an HC05 based MCU, and Motorola's ISM Band RF chipset. The targeted data rate was 57,600 Baud, although both higher and lower data rates are easily attainable. It is expected that most applications would embed the logic functions (and possibly the MCU functions) into a FPGA, CPLD, ASIC, or other LSI logic building block.

Throughout this application note, it is assumed the user is familiar with standard TTL–compatible CMOS devices and the ISM Band RF chipset. Please refer to DL110/D and DL129/D for additional details on individual device specifications.
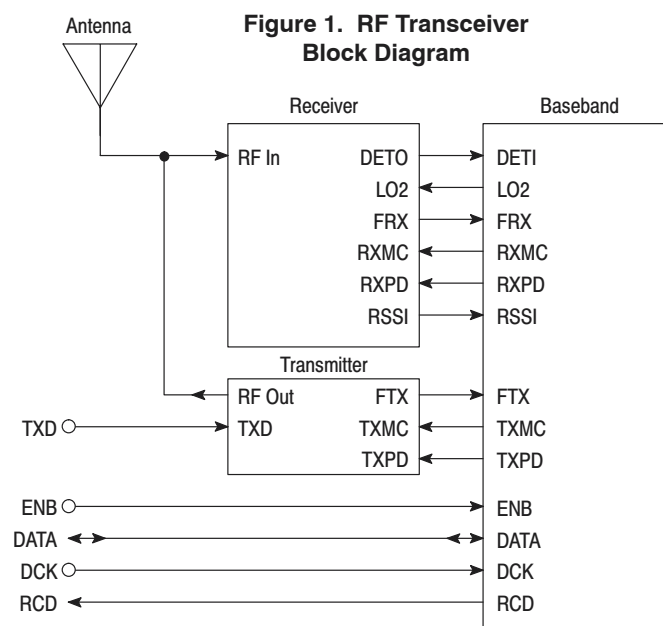
THE WIRELESS LINK

The actual implementation of the wireless link transceiver was accomplished with the Motorola's ISM Band RF chipset. This consists of a MC13145 RF Receiver, MC13146 RF Transmitter and MC33411 Baseband. Figure 1 depicts the block diagram of the RF transceiver. Figures 2, 3, and 4 are the actual schematics for the Receiver, Transmitter, and Baseband, respectively.

The transceiver was designed to operate in the unlicensed (i.e. FCC Part 15) 902–928 MHz Industrial, Scientific and Medical (ISM) band with low–power transmission. Since direct–conversion FSK modulation is used in conjunction with a PLL synthesized carrier, the digital modulation source must meet certain requirements:

1. The DC component should be as close to zero as possible. This maintains the best noise immunity at the receiver.
2. A minimum frequency component must be maintained at all times. If this condition is not met, the transmitter's PLL and receiver's coilless demodulator will tend to "track–out" the modulating signal.
3. The maximum frequency component should be known. This will help define the modulation index and total bandwidth required for the transceiver.
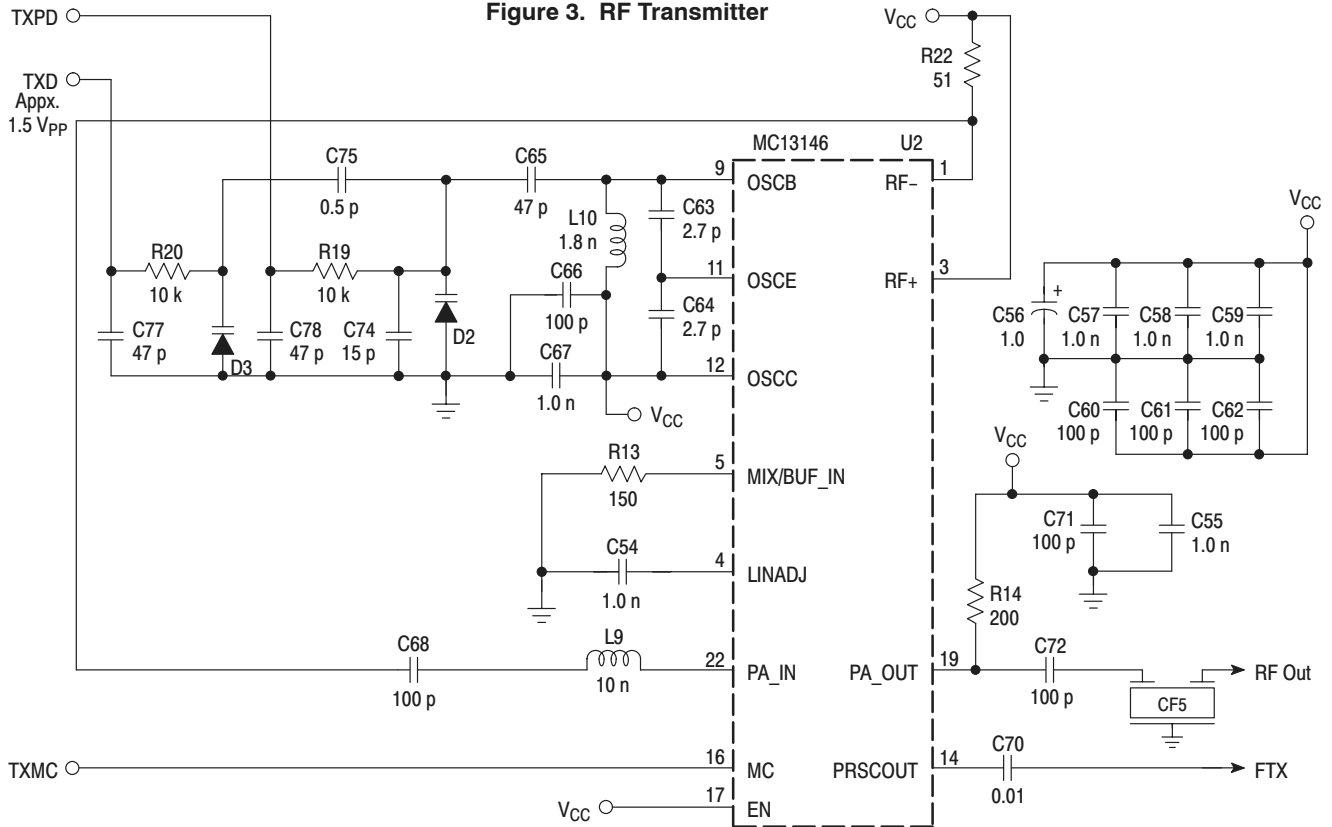4. The system should be able to tolerate reasonable bit–errors.

The MC33411 baseband controls all of the synthesizer functions via a MCU SPI compatible interface. None of the audio processing capabilities of the device are used. Table 1 lists various MC33411 register values for both baseset and handset for the 5 channels used for the prototype.

**Figure 1. RF Transceiver
Block Diagram**

MOTOROLA

Figure 2. RF Receiver

Default Units: Ohms, Microfarads and Microhenries

CF1,CF2	Toko Type CFSK Series
	SK107MX–AE–XXX, 330 kHz BW
CF3,CF4	Handset: TDK CF6118702
	Baseset: TDK CF6118902
D1	MMBV809LT1
T1	Toko A638AN–A099YWN

Figure 3. RF Transmitter

Default Units: Ohms, Microfarads and Microhenries
CF5   Handset: TDK CF 6118902
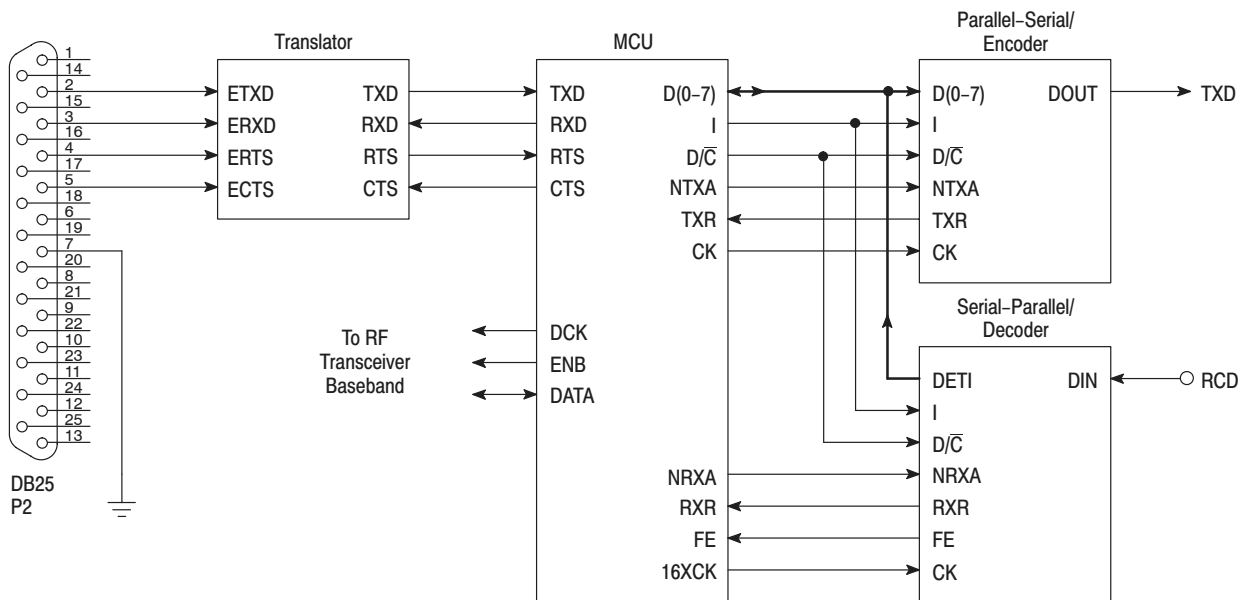   Baseset: TDK CF 6118702
D2,D3   MMBV809LT1

Figure 4. Baseband

Default Units: Ohms, Microfarads and Microhenries

DIGITAL ENCODING DESCRIPTION

As mentioned above, it is necessary to encode the raw RS–232 data prior to RF transmission since the incoming data stream can, and usually will, contain a DC component and has no pre–defined minimum frequency component. Figure 5 is a block diagram of the digital encoder/decoder section, and Figure 6 shows a possible implementation of the encoder.

Figure 5. Encoder/Decoder Block Diagram
(Baseset Shown)

Figure 6. Encoder
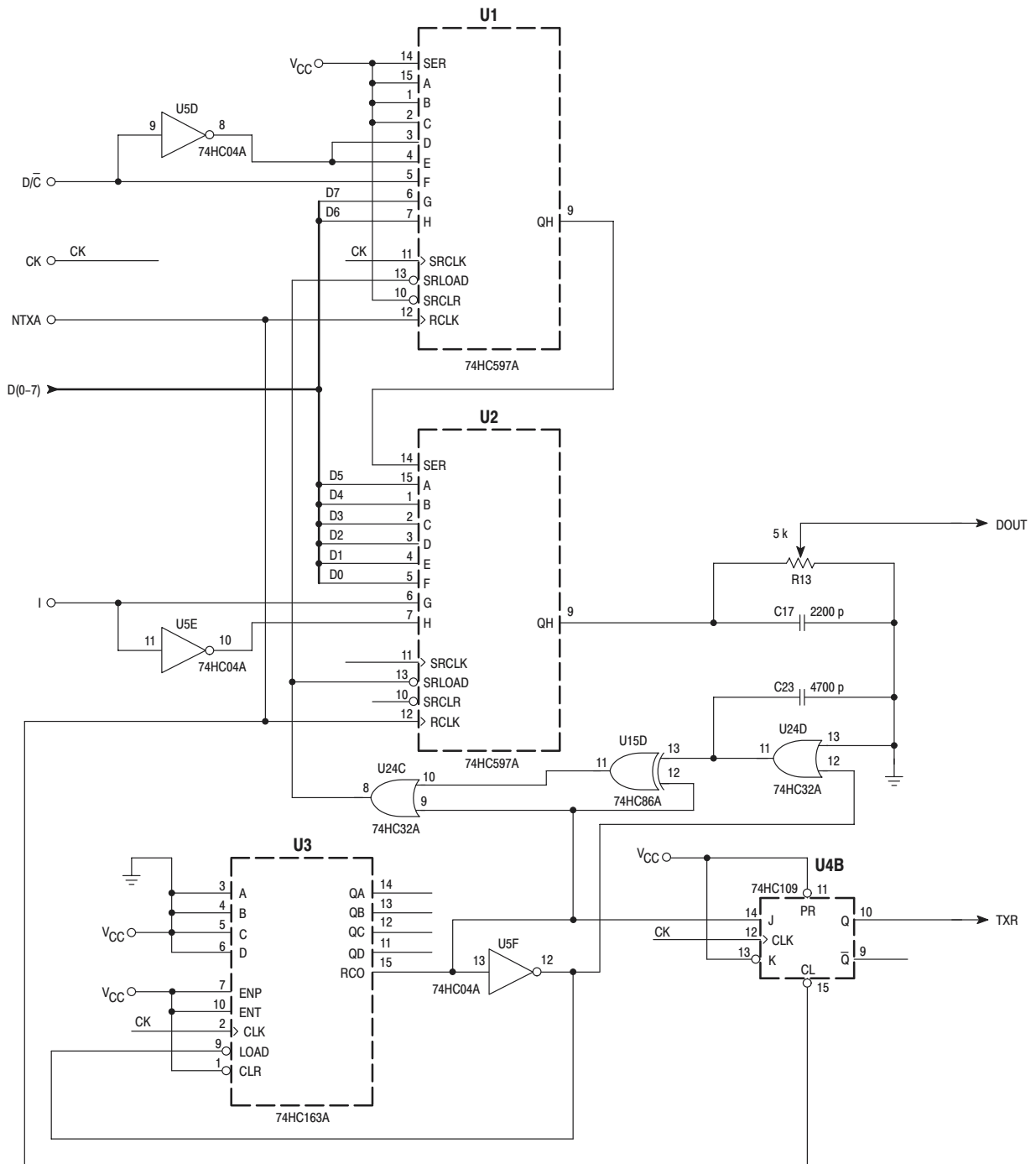
Figure 7 illustrates the encoding scheme which was developed for this purpose. Four additional bits surround a data byte: the $\overline{I}$ bit, I bit, D/$\overline{C}$ bit and $\overline{D}$/C bit. The function of these bits are:

I (Invert) Bit: A logic low on this bit indicates that the data byte and D/$\overline{C}$ bit are in true form. A logic high on this bit indicates that the data byte and D/$\overline{C}$ bit are complemented from their original form.

$\overline{I}$ (Invert Bar) Bit: Just the complement of the I bit.

D/$\overline{C}$ (Data/$\overline{Control}$) Bit: A logic low on this bit indicates that the data byte should be interpreted as a control word. A logic high on this bit indicates that the data byte contains real data.

$\overline{D}$/C ($\overline{Data}$/Control): Just the complement of the D/$\overline{C}$ bit.

Figure 7.
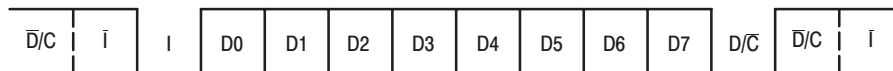
This encoding scheme allows for the representation of 256 unique data words and 256 unique control words. The control byte $h00 is reserved and referred to as the idle byte.

When the Parallel–Input/Serial–Output (PISO) register is ready to transmit a data byte (TXR asserted), a check is made to see if real data has been transferred into the Serial Communications Interface (SCI) data register of the MCU. If data has been received, the data byte will be read, and the initial state of D/$\overline{\text{C}}$ will be set to 1. If data has not been received, the data byte will be set to $h00 (the idle byte) and the initial state of D/$\overline{\text{C}}$ will be set to 0.

Next, the data byte is examined for a DC component. Each 0 bit of the data byte represents −1 and each 1 bit of the data byte represents +1. All of these values are summed together: a negative result indicates a low DC component, zero indicates no DC component, and a positive (non–zero) result indicates a high DC component. This component is compared to a cumulative sum (which may be negative, zero, or positive) and the following actions are taken:

If the current DC component sum is negative, and the cumulative sum is positive or zero

<div align="center">OR</div>

if the current DC component sum is positive or zero and the cumulative sum is negative

<div align="center">THEN</div>

clear the I bit (I=0). The new cumulative sum is equal to the old cumulative sum plus the current sum.

<div align="center">OTHERWISE</div>

set the I bit (I=1). The new cumulative sum is equal to the old cumulative sum minus the current sum.

If the I bit is set, the contents of the data byte and D/$\overline{\text{C}}$ bit are complemented.

The updated value of the I bit, data byte, and D/$\overline{\text{C}}$ bit are placed on the PISO, and a transmission acknowledge signal (NTXA) is asserted. Please note, the net effect of the DC component contributed by the I and $\overline{\text{I}}$ bits and D/$\overline{\text{C}}$ and $\overline{\text{D}}$/C bits will always equal zero.

An analysis of this encoding scheme brings to light a few interesting observations:

1. The average DC component over time will approach zero.

2. The minimum frequency component which will be observed in the data stream will equal 1/(2 x transmitted bit period x 10).

3. The maximum (fundamental) frequency component which will be observed in the data stream will equal 1/(2 x transmitted bit period).

4. A sequence of ten consecutive zeros or ones indicates the presence of an idle byte.

Item 4 is perhaps the most interesting observation, since it will allow the receiver to synchronize the incoming data and align the serial stream on a byte–wide basis.

TRANSMITTING FREQUENCY for ENCODED DATA

For RS–232 communications which take the form of one start bit, eight data bits, no parity, and one stop bit, the SCI will receive 10 bits of data to represent one actual data byte. For our encoding scheme, 12 bits must be transmitted for each data or control byte received. If the transmit pipeline is set to a frequency of at least 1.2 times the SCI receive pipeline, the receive bandwidth will not have to be reduced (i.e. no stop or hold conditions would be required).

In actual practice, the transmit pipeline was set to a frequency 25% greater than the receive pipeline. As a result, at a minimum, there will be at least one idle byte transmitted for every 24 real data bytes. This useful feature allows the receiver to re–synchronize from time to time.

ADDITIONAL FEATURES

As mentioned above, the opportunity presents itself to transmit a control word (the idle byte just being a special case of a control word) from time to time. With 255 control words remaining, various special features can be built into the link, all transparent to the actual RS–232 data communications.

One of the more obvious features which can be implemented is hardware (RTS/CTS) flow control. The RTS signal (for the baseset) and CTS signal (for the handset) can be monitored and transmitted/received and interpreted by the link. The latency will mostly be a function of the overhead bandwidth.

Other features which can be implemented include, but are not limited to:

<div align="center">
Remote channel changing
Adaptive channel selection
Acknowledgments
DCD/DSR, etc. commands
CRC or other error checking
Half duplex handshaking
Power conservation modes
</div>

Figure 8. Decoder
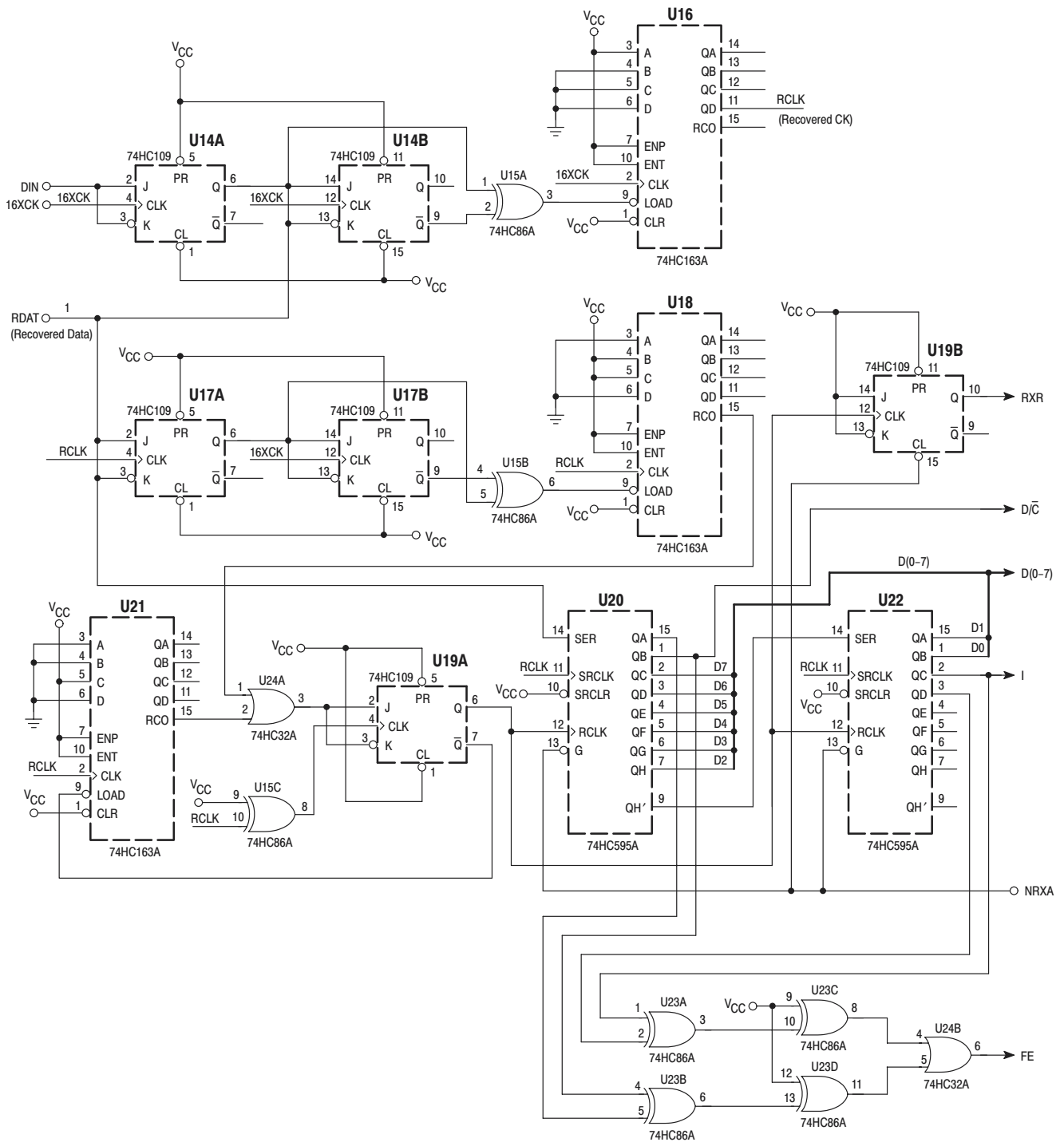
Figure 8 illustrates a possible implementation of the digital decoder section of the design.

Received data is "squared up" by the data slicer of the MC33411 baseband IC. The transmitted data is generally frequency limited in order to preserve bandwidth (i.e. low–pass filtered). Because of this limiting, as well as noise components and hysteresis in the data slicer, the duty cycle of the received data stream can vary substantially from that of the transmitted data. For this reason, a data and clock recovery block is utilized which oversamples the incoming data (digital noise filtering) and captures the embedded clock.

Once the clock and data have been recovered, they are presented to the Serial–Input/Parallel–Output (SIPO) register. Data is transferred into the register every 12 bits, this representing the $\bar{I}$, I, data byte, D/$\overline{C}$ and $\overline{D}$/C bits. Another circuit analyzes the serial data stream looking for ten consecutive bits without a transition. If this condition is observed, it indicates an idle byte has been received, and the SIPO register clock can be synchronized.

When the SIPO register indicates that a byte has been received (RXR asserted), the MCU asserts an acknowledgement (NRXA), loads the data byte, the I bit and D/$\overline{C}$ bit from the bus. At this time, a comparison is made which verifies that the I bit is the complement of the $\bar{I}$ bit and the D/$\overline{C}$ bit is the complement of the $\overline{D}$/C bit. If either of these conditions is not met, a framing error has occurred and the received data is simply ignored.

If a valid byte has been received, the MCU checks the status of the I bit. If the I bit is set, the byte, as well as the D/$\overline{C}$ bit, is complemented. Next, the MCU checks the value of the updated D/$\overline{C}$ bit; a logic zero indicates a control word, and the MCU can take appropriate action.

If the D/$\overline{C}$ bit indicates real data has been received, the data is placed on an internal First–In/First–Out (FIFO) memory stack. The SCI transmitter is checked: if empty, the next data byte is placed into the SCI transmitter and if full, the data will be transferred at a later time.

As can be seen, the decoding of the data is a relatively simple task. If desired, the MCU can consider the lack of an idle byte, within a given period of time or reception of some number of bytes, an indication that the RF link has failed. Again, this condition can be used to re–initialize the RF link, or other courses of action can be taken.

SUMMARY

This application note has described a robust, full featured RS–232 wireless interface which can be implemented with an inexpensive MCU. For slower data rates, it is possible to eliminate all of the external "glue logic" shown in this note. A plethora of additional features can be added by the use of embedded control words which are transparent to the actual data transceiver.

Motorola's inexpensive and easy to use ISM Band RF chipset is easily capable of accomplishing the wireless portion of the task as long as the digital information presented to the transmitter and receiver have been properly preconditioned prior to modulation and demodulation.

Table 1.

Baseband Register Address	Handset Value	Transmit Frequency (MHz)	Receive Frequency (MHz)	Baseset Value	Transmit Frequency (MHz)	Receive Frequency (MHz)	Channel Number
$h01	$h004822	925.0	–	$h004686	903.0	–	0
$h02	$h004C27	–	903.0	$h004E03	–	925.0	0
$h01	$h004827	925.5	–	$h00468B	903.5	–	1
$h02	$h004C2C	–	903.5	$h004E08	–	925.5	1
$h01	$h00482C	926.0	–	$h004690	904.0	–	2
$h02	$h004C31	–	904.0	$h004E0D	–	926.0	2
$h01	$h004831	926.5	–	$h004695	904.5	–	3
$h02	$h004C36	–	904.5	$h004E12	–	926.5	3
$h01	$h004836	927.0	–	$h00469A	905.0	–	4
$h02	$h004C3B	–	905.0	$h004E17	–	927.0	4
$h03	$h0E0276	–	–	$h0E0276	–	–	X
$h04	$h160070	–	–	$h160070	–	–	X
$h05	$h000010	–	–	$h000010	–	–	X
$h06	$h0000FF	–	–	$h0000FF	–	–	X
$h07	$h01C000	–	–	$h01C000	–	–	X

NOTES

ARCHIVE INFORMATION

ARCHIVE INFORMATION