



**MCRF355/360 Reader Reference Design**

**1.0 INTRODUCTION**

This chapter provides a reference guide for the 13.56 MHz reader designer. The schematic included in this chapter is for the 13.56 MHz Reference Reader included in the DV103003 microID™ Developer's Kit. The circuit is designed for short read-range applications. The basic design can be modified for long-range or other applications with MCRF355/360 devices. An electronic copy of the PICmicro® microcontroller source code is available upon request.

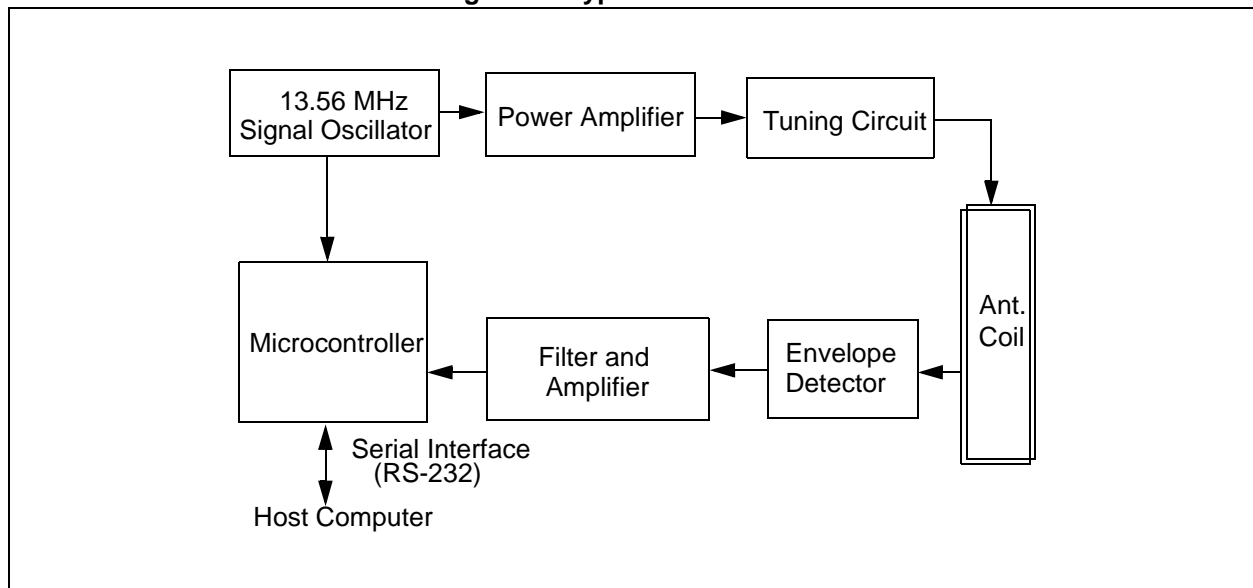
**2.0 READER CIRCUITS**

The RFID reader consists of transmitting and receiving sections. It transmits a carrier signal (13.56 MHz), receives the backscattered signal from the tag, and performs data processing. The reader also communicates with an external host computer. A basic block diagram of a typical RFID reader is shown in Figure 2-1.

The transmitting section contains a 13.56 MHz signal oscillator (74HC04), power amplifier (Q2), and RF tuning circuits. The tuning circuit matches impedance between the antenna coil circuit and the power driver at 13.56 MHz. The radiating signal strength from the antenna must comply with government regulations. For best performance, the antenna coil circuit must be tuned to the same frequency of the tag. The design for antenna circuits is given in Application Note AN710 (DS00710).

The receiving section contains an envelope detector (D6), hi-pass filters, and amplifiers (U2 and U3). When the tag is energized, it transmits 154 bits of data that is encoded in Biphase-L (Manchester). In the Manchester encoding, data '1' is represented by a logic high-to-low level change at midclock, and data '0' is represented by a low-to-high level change at midclock. There is always a level change at middle of every bit clock.

**FIGURE 2-1: Functional Block Diagram of Typical RFID Reader**



# microID™ 13.56 MHz DESIGN GUIDE

FIGURE 2-2: Signal Waveforms

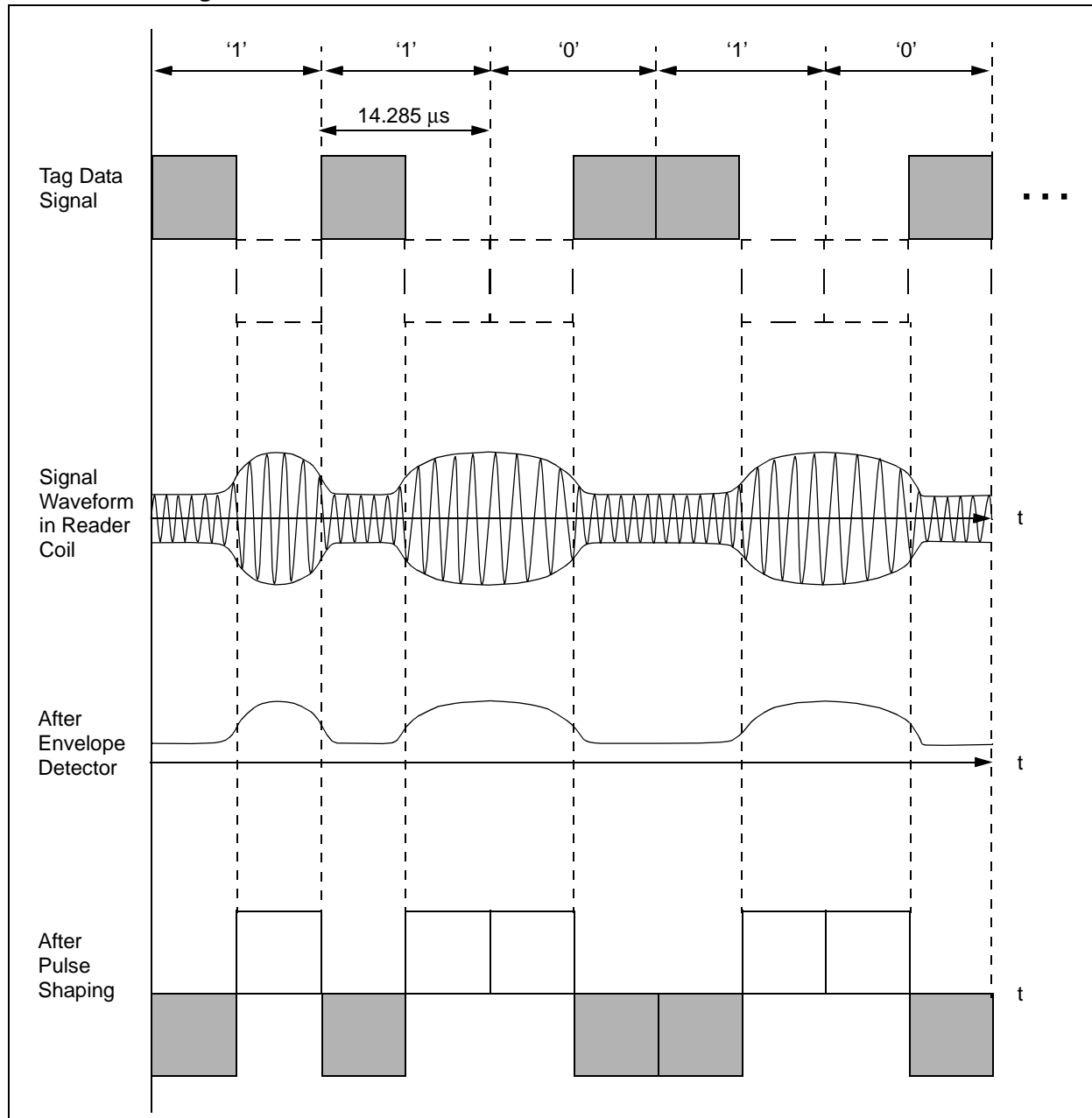
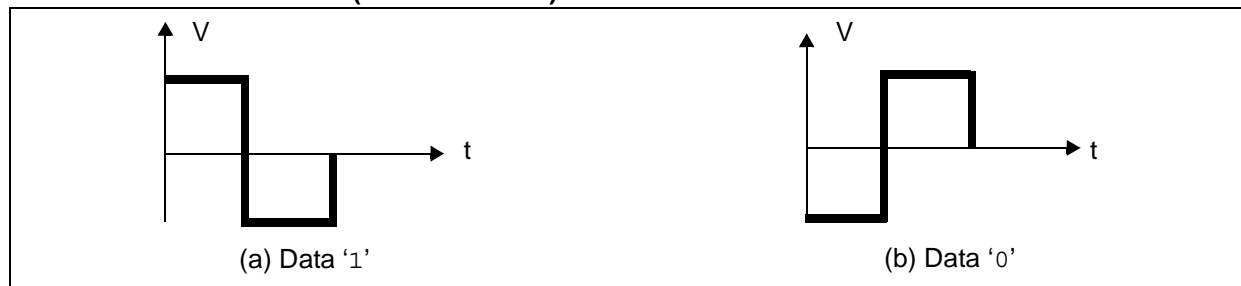


FIGURE 2-3: BIPHASE-L (MANCHESTER) SIGNAL



# microID™ 13.56 MHz Design Guide

---

When the tag is energized by the reader's carrier signal, it transmits back with an amplitude modulated signal. This results in a perturbation in the voltage amplitude across the reader antenna coil. The envelope detector detects the changes in the voltage amplitude and passes it into an RC filter (R7, C11). The charged signal in the capacitor passes through active filters and amplifiers. The signal that is passing through this receiving section is the data signal. This filtered-shaped data signal is fed into Pin 10 of the microcontroller for data processing.

## 2.1 FCC Specifications on Transmitting Signal

Each country limits the signal strength of the radio frequency signal that is intentionally radiated from the device. In the USA, the maximum signal strength that is radiated from the device is regulated by Federal Communication Commission (FCC). Any device operating at 13.56 MHz frequency band must comply with the FCC Part 15.225 of the federal regulation. FCC limits for 13.56 MHz frequency band are as follows:

1. Tolerance of the carrier frequency: 13.56 MHz  
+/- 0.01% = +/- 7 kHz.
2. Frequency bandwidth: +/- 7 kHz.
3. Power level of fundamental frequency: 10 mv/m at 30 meters from the transmitter.
4. Power level for harmonics: -50.45 dB down from the fundamental signal.

The transmission circuit including the antenna coil must be designed to meet the FCC limits.

## 3.0 OPTIMIZATION FOR LONG-RANGE APPLICATIONS

The reader circuit provided is designed for about a 5-inch read-range, using a 2-inch by 2-inch tag coil that is printed on PCB with the MCRF355. The read-range can be increased by increasing the reader power, sensitivity, and antenna size. A read-range of more than 30-inches can be achieved with the MCRF355 and an optimized reader. In order to optimize the reader circuit for long-range applications, the following aspects may be considered:

1. **Optimize the output power level within FCC limits.** The reader should provide a sufficient signal level to the tag. The tag needs about 4 V<sub>PP</sub> across the coil circuit for operation. The power level radiating from the reader antenna must comply to the government regulations such as FCC specifications in the USA. The FCC limits for 13.56 MHz band are described in Section 2.1. For long-range applications, the designer may start with about 50 V<sub>PP</sub> of antenna voltage and optimize the signal strength for a read-range within the government regulations.
2. **Increase the size of the antenna.** The read-range, in general, is proportional to the size of the reader coil (see Equation 12 in Application Note 710). An optimum radius of antenna is 1.414 times of the read-range.
3. **Increase the  $Q$  of the antenna circuit.** The read-range increases with  $Q$  of the antenna circuit. This is because the induced voltage is directly proportional to  $Q$  of the circuit. The recommended  $Q$  for long-range applications is as follows:

$$\begin{array}{ll} 40 < Q < 96 & \text{for reader} \\ 40 < Q & \text{for tag} \end{array}$$

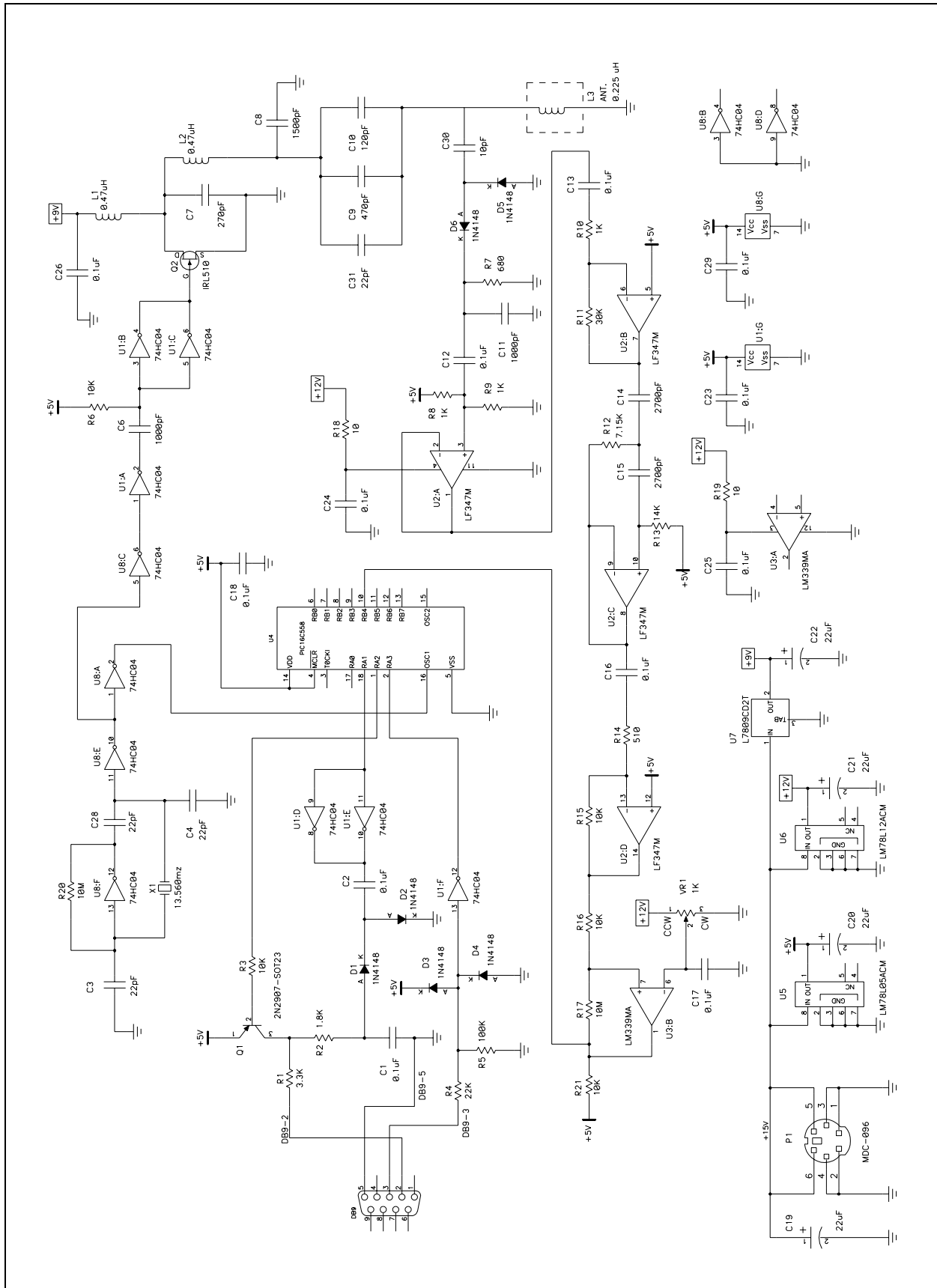
# microID™ 13.56 MHz DESIGN GUIDE

---

4. **Optimize the input sensitivity of the reader.** The sensitivity is a measure of how weak a signal can be and still be satisfactorily received. The sensitivity is proportional to the carrier power and square of the modulation index (1 for 100% modulation such as MCRF355). It is inversely proportional to the noise signal. The limit to the sensitivity of the receiving section of the reader is noise, both external and internal. The external noises may come from various sources such as computers, televisions, appliances, motors, power lines, transformers, etc. The internal noise is mostly due to a thermal noise of components. To reduce noise, the reader should be operated a distance away from the noise sources. The receiving section may have a 70 kHz bandpass filter to reduce the noises. The 70 kHz bandpass filter will pass only the 70 kHz data signal for processing. The receiving section should have sensitivity of about -120 dBm for long-range applications.
5. **Optimize the amplitude gain circuit.** The receiving circuit amplifies the modulated signals before data processing. The input signal contains both real data and noise. Typically, op amplifiers are used for both as a gain amplifier and filter. The gain must be optimized within the circuit to obtain gains only at the real data signal.

# microID™ 13.56 MHz Design Guide

## 4.0 READER SCHEMATIC



# microID™ 13.56 MHz DESIGN GUIDE

## 5.0 READER BILL OF MATERIALS

| Assembly # | Line # | Qty | Part #          | Part Description  | Reference Designator                                  |
|------------|--------|-----|-----------------|---|---|
| 02-01523   | 1      | 1   | 02-01523-D      | PCB ASSY DWG,<br>MCRF355 microID READER                             | —   |
| 02-01523   | 2      | 1   | 03-01523        | SCHEMATIC, MCRF355 microID<br>READER                                | —   |
| 02-01523   | 3      | 1   | 04-01523        | PCB FABRICATION,<br>MCRF355 microID READER                          | —   |
| 02-01523   | 4      | 2   | MM74HC04M       | IC, SMT, CMOS HEX INVERTER, 14P<br>SOIC                             | U1, U8  |
| 02-01523   | 5      | 1   | LF347M          | IC, SMT, QUAD BI-FET OP AMP, 14P<br>SOIC                            | U2  |
| 02-01523   | 6      | 1   | LM339M          | IC, SMT, LOW POWER LOW OFFSET<br>VOLT QUAD COMPARATORS, 14P<br>SOIC | U3  |
| 02-01523   | 7      | 1   | PIC16C558-20/SO | IC, PIC16C558-20/SO EPROM-BASED<br>8-BIT CMOS MICROCONTROLLER       | U4  |
| 02-01523   | 8      | 1   | LM78L05ACM      | IC, REG, +5V 100 mA REGULATOR                                       | U5  |
| 02-01523   | 9      | 1   | LM78L12ACM      | IC, REG, +12V 100 mA REGULATOR                                      | U6  |
| 02-01523   | 10     | 1   | L7809CD2T       | IC, +9V, REG 1.5A TO-263  | U7  |
| 02-01523   | 11     | 1   | MMBT2907ALT1    | TRANSISTOR, PNP, 2N2907A, SOT-23                                    | Q1 Flip upside and<br>bend legs toward the<br>PCB     |
| 02-01523   | 12     | 1   | IRL510          | TRANSISTOR, N-CHANNEL HEX FET,<br>TO220AB                           | Q2  |
| 02-01523   | 13     | 6   | RLS4148TE11C    | DIODE SMT, ROHM DIODE LL-34 SIG<br>DIODE                            | D1-D6   |
| 02-01523   | 14     | 1   | ERJ-3GSYJ332V   | RES SMT, 3.3K OHM, 1/16W, 5%, 0603                                  | R1  |
| 02-01523   | 15     | 1   | ERJ-3GSYJ182V   | RES SMT, 1.8K OHM, 1/16W, 5%, 0603                                  | R2  |
| 02-01523   | 16     | 5   | ERJ-3GSYJ103V   | RES SMT, 10K OHM, 1/16 W, 5%, 0603                                  | R3, R6, R15, R16, R21                                 |
| 02-01523   | 17     | 1   | ERJ-3GSYJ223V   | RES SMT, 22K OHM, 5% 0603   | R4  |
| 02-01523   | 18     | 1   | ERJ-3GSYJ104V   | RES SMT,<br>100K OHM 1/16W 5% TYPE 0603                             | R5  |
| 02-01523   | 19     | 1   | ERJ-3GSYJ681V   | RES SMT, 680 OHM 1/16W 5% 0603                                      | R7  |
| 02-01523   | 20     | 3   | ERJ-3GSYJ102V   | RES SMT, 1K OHM 1/16W 5% 0603                                       | R8-R10  |
| 02-01523   | 21     | 1   | ERJ-3GSYJ303V   | RES SMT, 30K OHM 1/16W 5% 0603                                      | R11   |
| 02-01523   | 22     | 1   | ERJ-3EKF7151V   | RES SMT, 7.15K OHM 1/16W 1% 0603                                    | R12   |
| 02-01523   | 23     | 1   | MFR-25FRF 14K0  | RES, 14K OHM 1/4W 1% MF   | R13, connected from<br>U2 pin 12 to top pad of<br>R13 |
| 02-01523   | 24     | 2   | RM73B1JT106J    | RES SMT, 10M OHM 1/16W 5% 0603                                      | R17, R20  |
| 02-01523   | 25     | 2   | ERJ-3GSYJ100V   | RES SMT, 10 OHM 1/16W 5% 0603                                       | R18, R19  |
| 02-01523   | 26     | 1   | EVM-7JSX30B13   | RES SMT, POT,<br>1K OHM 3MM SEALED, 3 TT                            | VR1   |
| 02-01523   | 27     | 12  | ECU-V1H104KBW   | CAP SMT, 0.1uF 50V 10%,<br>X7R CER 1206                             | C1, C2, C12, C13,<br>C16-18, C23-C26, C29             |
| 02-01523   | 28     | 3   | ECU-V1H220JCV   | CAP SMT,<br>22 pF CERAMIC 5% 50V 0603 NPO                           | C3, C4, C28   |
| 02-01523   | 29     | 2   | ECU-V1H102KBV   | CAP SMT,<br>1000 pF 50V CERAMIC 10% 0603 X7R                        | C6, C11   |

# microID™ 13.56 MHz Design Guide

| Assembly # | Line # | Qty | Part #                  | Part Description  | Reference Designator |
|------------|--------|-----|-------------------------|---|----------------------|
| 02-01523   | 30     | 1   | ECU-V1H271JCV           | CAP SMT,<br>270 pF 50V CERAMIC 5% 0603 NPO  | C7                   |
| 02-01523   | 31     | 1   | ECU-V1H152KBV           | CAP SMT, 1500 pF 50V CERAMIC 10%<br>0603 X7R                                      | C8                   |
| 02-01523   | 32     | 1   | GRM42-<br>6C0G471G500AL | CAP SMT, 470 pF 500V 2% 1206 C0G  | C9                   |
| 02-01523   | 33     | 1   | GRM42-<br>6C0G121J500AL | CAP SMT, 120 pF 500V 5% 1206 C0G  | C10                  |
| 02-01523   | 34     | 2   | ECU-V1H272KBV           | CAP SMT,<br>2700PF 50V CERAMIC 10% 0603 XR7                                       | C14, C15             |
| 02-01523   | 35     | 4   | ECE-A1EU220             | CAP, 22UF 25V RADIAL<br>ELECTROLYTIC 20%  | C19-C22              |
| 02-01523   | 36     | 1   | GRM42-<br>6C0G100J500AL | CAP SMT, 10 pF 500V 5% 1206 C0G   | C30                  |
| 02-01523   | 37     | 1   | GRM42-<br>6C0G220J500AL | CAP SMT, 22 pF 500V 5% 1206 C0G   | C31 (AS NEEDED)      |
| 02-01523   | 38     | 2   | 43LS477                 | INDUCTOR, 0.47 $\mu$ H  | L1, L2               |
| 02-01523   | 39     | 1   | MCX0001                 | OSCILLATOR, CUSTOM 13.560 MHz,<br>PARALLEL MODE, 22 pF LOAD, HC49<br>CASE, 30 PPM | X1                   |
| 02-01523   | 40     | 1   | MDC-096                 | CONN, MINI-DIN, 6-PIN   | P1                   |
| 02-01523   | 41     | 1   | KF22-E9S-NJ             | CONN, D-SUB 9P RECPT RT ANGLE<br>WITH JACK SCREWS                                 | DB9                  |
| 02-01523   | 42     | 1   | 08-00170                | LABEL, MCRF355 READER FIRM-<br>WARE, 355READ.HEX, 1/25/99, U4                     | @ U4                 |
| 02-01523   | 43     | 1   | ERJ-3GSYJ511V           | RES SMT, 510 OHM 1/16W 5% 0603  | R14                  |

# microID™ 13.56 MHz DESIGN GUIDE

---

## Software License Agreement

The software supplied herewith by Microchip Technology Incorporated (the "Company") is intended and supplied to you, the Company's customer, for use solely and exclusively on Microchip products.

The software is owned by the Company and/or its supplier, and is protected under applicable copyright laws. All rights are reserved. Any use in violation of the foregoing restrictions may subject the user to criminal sanctions under applicable laws, as well as to civil liability for the breach of the terms and conditions of this license.

THIS SOFTWARE IS PROVIDED IN AN "AS IS" CONDITION. NO WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE. THE COMPANY SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.

## 6.0 READER SOURCE CODE FOR THE PICmicro® MCU

```
;receiver.asm
```

```
;Processor: PIC16C558 operating at 13.56 MHz
;      Ti= 295 nsec
```

```
processor 16c558
#include "P16c558.inc"
__config h'3ff2' ;protection off,PWRT enabled,watchdog disabled,HS oscillator
```

```
#define _CARRY      STATUS,0
#define _ZERO      STATUS,2
```

```
#define _125KHZ    PORTA,1
#define _RS232TX   PORTA,2
#define _RS232RX   PORTA,3
#define _RS232     PORTA
#define SIGNAL     PORTB,4
invmask          = h'2'
```

```
;.....
```

```
;Define variables and constants here--
```

```
delay          =h'20'
wait           =h'21'
acctime        =h'22' ;accumulated sync interval sum--also used as halfbit interval threshold
#define halfthr acctime ;halfbit interval threshold
halfthr        =acctime ;halfbit interval threshold
rcv_csumhi     =h'23' ;2 bytes for storing received checksum
rcv_csumlo     =h'24'
bitcnt         =h'25' ;RS232 bit counter
cycle_cnt      =h'26'
halfthr        =h'27' ;threshold value between halfbit and fullbit intervals
ptr1           =h'28' ;temporary FSR storage
ptr2           =h'29' ;temporary FSR storage
TXchar         =h'2a' ;character to transmit over RS232
temp           =h'2b' ;temporary storage
shiftcnt       =h'2c' ;used to strip the framing '0' bits from the rec'd data array
letters        =h'2d' ;storage area for next character to send
charcnt        =h'2e'
lastbit        =h'2f' ;the Lsb stores the last rec'd bit--flip it by complementing f
```

```
;;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!bit storage area--16 bytes of storage, indirectly addressed
;;;Note that s/w tests for MSb to detect end of area--be careful if move to different
;;;processor or relocate this storage area
rcvbits        =h'40' ;32 bytes set aside for storing the received bits--actual number of bytes
                ;in transmission is 18
;;;Note that main loop uses bit tests to determine bit receive or runaway condition (to limit
```



# microID™ 13.56 MHz Design Guide

```
;;processing time). Keep this in mind if recvbits storage area changed in the future.
;;40h-60h is reserved for received bits--actual bit receiving area 40h-51h, rest is overrun area

;;52h-73h set aside for ASCII conversion of received bytes before RS232 transmission. Note that
;;52h-60h contains no useful information from the use during receive of demodulated bits. Also,
;; bits are not being received while the ASCII conversion and serial transmission are
;; taking place.
;; 'G' 1st character: "go"
;; Character 2-37: ASCII representation of received 18 bytes (until checksum used)
;; Character 38: '\n' newline

sendascii =h'52' ;begin of storage area for ASCII conversion of received bytes
xfercnt =d'14' ;defines number of received bytes to convert to ASCII & transmit
```

```
.....
;.....
;Overall function- To recover Manchester encoded RFID message after AM demodulation and
; comparator decision. The comparator input trips the interrupt on PORTB change.
;The steps are:
;
; 1- Initialize registers to seek synch field.
; 2- Determine bit width from synch field by averaging the periods between transitions
; over the synch field. TMR0 is cleared at each edge. If the timer overflows before
; the next edge, synch seek starts over. The synch field is composed of 9 bits.
; 3- Use the measured bit width to establish a threshold period between repeat bits and
; complement of previous bit. This is due to the Manchester encoding method. Since there
; is always a transition in the middle of each bit interval transmitted, a repeated bit
; will appear as a pair of edges that occur with a halfbit interval period. A bit that
; is the complement of the last received bit will appear as an interval between edges
; of a full bit interval period.
; 4- Shift in bits as they are received into the storage array. When the timer overflows,
; consider the data field over. The received data format is MSb to LSb, where the MSb
; is the first bit received.
; 5- There are 16 bytes in the message, followed by a 16 bit checksum of the message
; contents. The remaining bit is unused.
; 6- Compute the checksum of the received 16 byte message and compare to the received
; checksum.
; 7- If checksums match, convert the message and the checksum into ASCII form and transmit
; over the RS232 serial link. The message format is:
; "GG" :the go characters (start of message)
; 36 bytes which are the ASCII representation of the 18 bytes received
; "\n" : closing newline character
; The serial data rate is 9600 bps, 8 data bits, 1 stop, no parity
;.....
```

```
org h'000' ;RESET vector location
goto init
org h'004' ;interrupt vector location
```

```
=====
;;isr(): interrupt service routine
; interrupts enabled for transition on PORTB
;
; 1- BEWARE! To minimize interrupt response time, the w & status register are NOT
; archived.
; 2- The isr execution path is determined by w register and uses calculated goto's.
; The w for next isr is set at end of current isr execution and is dependent on
; signal context (i.e. sync start, w/in sync, w/in data, etc.)
; Be very cautious here--must stay w/in 255 instructions for this to work!
; 3- Sync field processed as follows:
; -Ignore the first 4 transitions, they may be in response to tag power on reset
; -Accumulate the sum of next 8 intervals
; -Establish half bit width from full bit width threshold value based on
```

# microID™ 13.56 MHz DESIGN GUIDE

---

```
;          average interval measured above. Due to Manchester encoding, repeat of previous
;          bit will be a series of 2 halfbit width intervals, complement of previous bit
;          will be a fullbit width interval. halfbit defined as 1.5x(average sync).
;          -wait for interval over the fullbit threshold. This is end of sync. In accordance
;          w/ Manchester encoding, the sync field will be: 1 1 1 1 1 1 1 0
;=====
ISR
    addwf PCL,f      ;4 calculated goto
;first sync edge is calculated goto here
    clrf TMR0       ;5
    movf PORTB,f    ;6 must read PORTB before clearing RBIF
    bcf INTCON,RBIF ;7 just in case timer interrupt happened just at 1st edge
    bcf INTCON,T0IF ;8
    movlw (first_cycle - isr-d'1') ;9 next isr calculated goto offset
    clrf lastbit    ;10 lastbit @ end of sync = 0
    retfie         ;12
;end of first cycle here. Note that first 4 transitions are ignored, because sync start is
;corrupted by tag power on reset.
first_cycle
    clrf TMR0       ;5
    movf PORTB,f    ;6 must read PORTB before clearing RBIF
    bcf INTCON,RBIF ;7
    movlw (second_cycle - isr-d'1') ;8 next isr calculated goto offset
    retfie         ;10
;end of 2nd cycle here. Note that first 4 transitions are ignored, because sync start is
;corrupted by tag power on reset.
second_cycle
    clrf TMR0       ;5
    movf PORTB,f    ;6 must read PORTB before clearing RBIF
    bcf INTCON,RBIF ;7
    movlw recvbits  ;8
    movwf FSR       ;9 set up to store data bits
    movlw (third_cycle - isr-d'1') ;10 next isr calculated goto offset
    retfie         ;12
;end of 3rd cycle here. Note that first 4 transitions are ignored, because sync start is
;corrupted by tag power on reset. The 3rd cycle is the 4th transition, so from here we measure
;the longest interval in sync field.
third_cycle
    clrf TMR0       ;5
    movf PORTB,f    ;6 must read PORTB before clearing RBIF
    bcf INTCON,RBIF ;7
    clrf acctime    ;8 reset accumulated sync interval for average
    movlw (fourth_cycle - isr-d'1') ;9 next isr calculated goto offset
    retfie         ;11
;end of 4th cycle here. Start looking for longest sync interval here.
fourth_cycle
    movf TMR0,w     ;5
    clrf TMR0       ;6
    movf PORTB,f    ;7
    bcf INTCON,RBIF ;8
    addwf acctime,f ;9 first measured sync cycle, must be the largest
    movlw (fifth_cycle - isr-d'1') ;10
    retfie         ;12
;end of 5th cycle here.
fifth_cycle
    movf TMR0,w     ;5
    clrf TMR0       ;6
    movf PORTB,f    ;7
    bcf INTCON,RBIF ;8
    addwf acctime,f ;9 acctime = acctime + TMR0
    movlw (sixth_cycle - isr-d'1') ;10
    retfie         ;12
;end of 6th cycle here.
sixth_cycle
    movf TMR0,w     ;5
```

# microID™ 13.56 MHz Design Guide

```
    clrf    TMR0        ;6
    movf   PORTB,f      ;7
    bcf    INTCON,RBIF ;8
    addwf  acctime,f    ;9    acctime = acctime + TMR0
    movlw  (seventh_cycle - isr-d'1') ;10
    retfie ;12
;end of 7th cycle here.
seventh_cycle
    movf   TMR0,w      ;5
    clrf   TMR0        ;6
    movf   PORTB,f     ;7
    bcf    INTCON,RBIF ;8
    addwf  acctime,f   ;9    acctime = acctime + TMR0
    movlw  (eighth_cycle - isr-d'1') ;10
    retfie ;12
;end of 8th cycle here.
eighth_cycle
    movf   TMR0,w      ;5
    clrf   TMR0        ;6
    movf   PORTB,f     ;7
    bcf    INTCON,RBIF ;8
    addwf  acctime,f   ;9    acctime = acctime + TMR0
    movlw  (ninth_cycle - isr-d'1') ;10
    retfie ;12
;end of 9th cycle here.
ninth_cycle
    movf   TMR0,w      ;5
    clrf   TMR0        ;6
    movf   PORTB,f     ;7
    bcf    INTCON,RBIF ;8
    addwf  acctime,f   ;9    acctime = acctime + TMR0
    movlw  (tenth_cycle - isr-d'1') ;10
    retfie ;12
;end of 10th cycle here.
tenth_cycle
    movf   TMR0,w      ;5
    clrf   TMR0        ;6
    movf   PORTB,f     ;7
    bcf    INTCON,RBIF ;8
    addwf  acctime,f   ;9    acctime = acctime + TMR0
    movlw  (eleventh_cycle - isr-d'1') ;10
    retfie ;12
;end of 11th cycle here. --this is last of sync cycles to be accumulated. Average the result
;and determine halfbit threshold in remaining sync cycles.
eleventh_cycle
    movf   TMR0,w      ;5
    clrf   TMR0        ;6
    movf   PORTB,f     ;7
    bcf    INTCON,RBIF ;8
    addwf  acctime,f   ;9    acctime = acctime + TMR0
    movlw  (twelfth_cycle - isr-d'1') ;10
    retfie ;12
;end of 12th cycle here. Start averaging the sync interval accumulated time
twelfth_cycle
    movf   PORTB,f     ;5
    bcf    INTCON,RBIF ;6
    rrf    acctime,f   ;7    acctime/2
    rrf    acctime,f   ;8    acctime/4
    rrf    acctime,f   ;9    avg interval = acctime/8
    movlw  h'1f'      ;10    clear 3 MSBs that may have been set by carry
    andwf  acctime,f   ;11
    movlw  (cycle13 - isr-d'1') ;12
    retfie ;14
;end of 13th cycle here. Calculate the halfbit threshold = 1.5(sync interval avg) Note that
;that the threshold value will be kept in acctime (=halfthr)
```

# microID™ 13.56 MHz DESIGN GUIDE

---

```
cycle13
    clrf    TMR0           ;5
    movf   PORTB,f        ;6
    bcf    INTCON,RBIF    ;7
    rrf    acctime,w      ;8 half the sync interval avg
    addwf  acctime,f      ;9 halfthr = 1+1.5x(sync interval avg)
    incf   acctime,f      ;10
    movlw  (sync_end - h'100'-h'1'-isr) ;11
    bsf    PCLATH,0       ;12 adjust for origin @ 100h
    retfie                   ;14

    org    h'100'
;sync end wait. End of sync is distinguished by a fullbit interval. ( T > halfthr )
sync_end
    movf   TMR0,w         ;5
    clrf   TMR0           ;6
    movf   PORTB,f        ;7
    bcf    INTCON,RBIF    ;8
    subwf  halfthr,w      ;9    Test interval to detect end of sync field (halfthr - w)
    movlw  (sync_end - h'100'-isr-d'1') ;10
    btfs   STATUS,C       ;12    Carry set for halfthr >= w
    movlw  (bit1 - h'100'-isr-h'1');12    If T > halfbit, end of sync detected. Proceed to data
processing
    retfie                   ;14
;rec'd bit processing here --bit1 is 1st bit of 8 bit block
bit1
    movf   TMR0,w         ;5
    clrf   TMR0           ;6
    movf   PORTB,f        ;7
    bcf    INTCON,RBIF    ;8
    subwf  halfthr,w      ;9    Test interval to determine bit. C = 1 for repeated bit
    btfs   STATUS,C       ;11
    goto   halfabit1      ;12
;fullbit processing here
    comf   lastbit,f      ;12    Complement lastbit for fullbit measurement
    rrf    lastbit,w      ;13
    rlf    INDF,f         ;14    shift in the new bit
    movlw  (bit2 - h'100'-isr-h'1') ;15
    retfie                   ;17
halfabit1
;repeated bit (1 of 8)
    rrf    lastbit,w      ;13
    rlf    INDF,f         ;14
    movlw  (half21-h'100'-isr-h'1') ;15
    retfie                   ;17
;2nd half of bit interval processing
half21    ;2nd half, bit1
    clrf   TMR0           ;5
    movf   PORTB,f        ;6
    bcf    INTCON,RBIF    ;7
    movlw  (bit2-h'100'-isr-h'1');8
    retfie                   ;10

;rec'd bit processing here --bit2 is 2nd bit of 8 bit block
bit2
    movf   TMR0,w         ;5
    clrf   TMR0           ;6
    movf   PORTB,f        ;7
    bcf    INTCON,RBIF    ;8
    subwf  halfthr,w      ;9    Test interval to determine bit. C = 1 for repeated bit
    btfs   STATUS,C       ;11
    goto   halfabit2      ;12
;fullbit processing here
    comf   lastbit,f      ;12    Complement lastbit for fullbit measurement
    rrf    lastbit,w      ;13
```

# microID™ 13.56 MHz Design Guide

---

```
    rlf    INDF,f        ;14  shift in the new bit
    movlw (bit3 - h'100'-isr-h'1') ;15
    retfie                ;17
halfabit2
;repeated bit (2 of 8)
    rrf    lastbit,w    ;13
    rlf    INDF,f        ;14
    movlw (half22-h'100'-isr-h'1') ;15
    retfie                ;17
;2nd half of bit interval processing
half22        ;2nd half, bit2
    clrf  TMR0          ;5
    movf  PORTB,f       ;6
    bcf   INTCON,RBIF  ;7
    movlw (bit3-h'100'-isr-h'1') ;8
    retfie                ;10
;rec'd bit processing here --bit3 is 3rd bit of 8 bit block
bit3
    movf  TMR0,w        ;5
    clrf  TMR0          ;6
    movf  PORTB,f       ;7
    bcf   INTCON,RBIF  ;8
    subwf halfthr,w    ;9   Test interval to determine bit. C = 1 for repeated bit
    btfsc STATUS,C     ;11
    goto  halfabit3    ;12
;fullbit processing here
    comf  lastbit,f     ;12  Complement lastbit for fullbit measurement
    rrf   lastbit,w     ;13
    rlf   INDF,f        ;14  shift in the new bit
    movlw (bit4 - h'100'-isr-h'1') ;15
    retfie                ;17
halfabit3
;repeated bit (3 of 8)
    rrf    lastbit,w    ;13
    rlf    INDF,f        ;14
    movlw (half23-h'100'-isr-h'1') ;15
    retfie                ;17
;2nd half of bit interval processing
half23        ;2nd half, bit3
    clrf  TMR0          ;5
    movf  PORTB,f       ;6
    bcf   INTCON,RBIF  ;7
    movlw (bit4-h'100'-isr-h'1') ;8
    retfie                ;10
;rec'd bit processing here --bit4 is 4th bit of 8 bit block
bit4
    movf  TMR0,w        ;5
    clrf  TMR0          ;6
    movf  PORTB,f       ;7
    bcf   INTCON,RBIF  ;8
    subwf halfthr,w    ;9   Test interval to determine bit. C = 1 for repeated bit
    btfsc STATUS,C     ;11
    goto  halfabit4    ;12
;fullbit processing here
    comf  lastbit,f     ;12  Complement lastbit for fullbit measurement
    rrf   lastbit,w     ;13
    rlf   INDF,f        ;14  shift in the new bit
    movlw (bit5 - h'100'-isr-h'1') ;15
    retfie                ;17
halfabit4
;repeated bit (4 of 8)
    rrf    lastbit,w    ;13
    rlf    INDF,f        ;14
    movlw (half24-h'100'-isr-h'1') ;15
    retfie                ;17
```

# microID™ 13.56 MHz DESIGN GUIDE

---

```
;2nd half of bit interval processing
half24      ;2nd half, bit4
    clrf   TMR0      ;5
    movf   PORTB,f   ;6
    bcf    INTCON,RBIF ;7
    movlw (bit5-h'100'-isr-h'1');8
    retfie          ;10
;rec'd bit processing here --bit5 is 5th bit of 8 bit block
bit5
    movf   TMR0,w    ;5
    clrf   TMR0      ;6
    movf   PORTB,f   ;7
    bcf    INTCON,RBIF ;8
    subwf  halfthr,w ;9   Test interval to determine bit. C = 1 for repeated bit
    btfs  STATUS,C  ;11
    goto   halfabit5 ;12
;fullbit processing here
    comf   lastbit,f ;12  Complement lastbit for fullbit measurement
    rrf    lastbit,w ;13
    rlf    INDF,f    ;14  shift in the new bit
    movlw (bit6 - h'100'-isr-h'1') ;15
    retfie          ;17
halfabit5
;repeated bit (5 of 8)
    rrf    lastbit,w ;13
    rlf    INDF,f    ;14
    movlw (half25-h'100'-isr-h'1') ;15
    retfie          ;17
;2nd half of bit interval processing
half25      ;2nd half, bit5
    clrf   TMR0      ;5
    movf   PORTB,f   ;6
    bcf    INTCON,RBIF ;7
    movlw (bit6-h'100'-isr-h'1') ;8
    retfie          ;10

;rec'd bit processing here --bit6 is 6th bit of 8 bit block
bit6
    movf   TMR0,w    ;5
    clrf   TMR0      ;6
    movf   PORTB,f   ;7
    bcf    INTCON,RBIF ;8
    subwf  halfthr,w ;9   Test interval to determine bit. C = 1 for repeated bit
    btfs  STATUS,C  ;11
    goto   halfabit6 ;12
;fullbit processing here
    comf   lastbit,f ;12  Complement lastbit for fullbit measurement
    rrf    lastbit,w ;13
    rlf    INDF,f    ;14  shift in the new bit
    movlw (bit7 - h'100'-isr-h'1') ;15
    retfie          ;17
halfabit6
;repeated bit (6 of 8)
    rrf    lastbit,w ;13
    rlf    INDF,f    ;14
    movlw (half26-h'100'-isr-h'1') ;15
    retfie          ;17
;2nd half of bit interval processing
half26      ;2nd half, bit6
    clrf   TMR0      ;5
    movf   PORTB,f   ;6
    bcf    INTCON,RBIF ;7
    movlw (bit7-h'100'-isr-h'1') ;8
    retfie          ;10
;rec'd bit processing here --bit7 is 7th bit of 8 bit block
```

# microID™ 13.56 MHz Design Guide

```
bit7
    movf  TMR0,w      ;5
    clrf  TMR0       ;6
    movf  PORTB,f    ;7
    bcf   INTCON,RBIF ;8
    subwf halfthr,w  ;9   Test interval to determine bit. C = 1 for repeated bit
    btfsc STATUS,C   ;11
    goto  halfabit7  ;12
;fullbit processing here
    comf  lastbit,f  ;12   Complement lastbit for fullbit measurement
    rrf   lastbit,w  ;13
    rlf   INDF,f     ;14   shift in the new bit
    movlw (bit8 - h'100'-isr-h'1') ;15
    retfie           ;17
halfabit7
;repeated bit (7 of 8)
    rrf   lastbit,w  ;13
    rlf   INDF,f     ;14
    movlw (half27-h'100'-isr-h'1') ;15
    retfie           ;17
;2nd half of bit interval processing
half27      ;2nd half, bit7
    clrf  TMR0       ;5
    movf  PORTB,f    ;6
    bcf   INTCON,RBIF ;7
    movlw (bit8-h'100'-isr-h'1') ;8
    retfie           ;10
;rec'd bit processing here --bit8 is 8th bit of 8 bit block
bit8
    movf  TMR0,w      ;5
    clrf  TMR0       ;6
    movf  PORTB,f    ;7
    bcf   INTCON,RBIF ;8
    subwf halfthr,w  ;9   Test interval to determine bit. C = 1 for repeated bit
    btfsc STATUS,C   ;11
    goto  halfabit8  ;12
;fullbit processing here
    comf  lastbit,f  ;12   Complement lastbit for fullbit measurement
    rrf   lastbit,w  ;13
    rlf   INDF,f     ;14   shift in the new bit
    movlw (bit1 - h'100'-isr-h'1') ;15
    incf  FSR,f      ;16
    retfie           ;18
halfabit8
;repeated bit (8 of 8)
    rrf   lastbit,w  ;13
    rlf   INDF,f     ;14
    movlw (half28-h'100'-isr-h'1') ;15
    retfie           ;17
;2nd half of bit interval processing
half28      ;2nd half, bit8
    clrf  TMR0       ;5
    movf  PORTB,f    ;6
    bcf   INTCON,RBIF ;7
    movlw (bit1-h'100'-isr-h'1') ;8
    incf  FSR,f      ;9   advance to next byte in recvbits storage array
    retfie           ;11

;The negative RS232 supply is generated by an inverter clocked at ~125 KHz by port pin RA1.
    ;first pump up the -5V, i.e. generate 125 KHz clock (T=8 usec, ~27 Ti)
    ;run for a total of 128 cycles before sending data
    ;put line at stop bit level
```

# microID™ 13.56 MHz DESIGN GUIDE

---

```
alphabet
    clrwdt
    bcf  INTCON,GIE ;make sure interrupts are off
    movlw sendascii
    movwf FSR

    movlw xfercnt ;# of ASCII represented received bytes to xfer
    addlw xfercnt ;x2
    addlw h'3' ;plus 2 start character "G" and newline character at end
    movwf charcnt
;;set up registers in bank 1
    bsf  STATUS,RP0 ;point to bank 1
    movlw h'8'
    movwf TRISA ;RA3 input, RA2-0 output
    movlw h'10'
    movwf TRISB ;RB7-5,3-0 output, RB4 input
    movlw b'00001100' ;set up timer option for internal clock, prescale-->watchdog/16
    movwf OPTION_REG ;port B pullups enabled
    bcf  STATUS,RP0 ;point back to bank 0
;;done setting up registers in bank 1, back to bank 0
    bsf  _RS232TX ;default is mark mode
    call gen125khz

;start the test transmission
sendA
    movf INDF,w
    movwf TXchar
    movlw d'8'
    movwf bitcnt
;stop bit last
    bsf  _RS232TX
    call TX_RS232 ;stop bit = 3Ti

    call ti17 ;burn 17Ti (includes the 2Ti for the call)
;start bit first
    bcf  _RS232TX
    call TX_RS232
    call ti17 ;burn 17Ti (includes the 2Ti for the call, adjusts the bit timing)
sendchar
    btfsc TXchar,0 ;1Ti
    goto setbit ;3Ti
    bcf  _RS232TX
    goto nextbit
setbit
    bsf  _RS232TX ;4Ti
nextbit
    call TX_RS232 ;6Ti
    rrf  TXchar,f ;7Ti
    call ti10 ;17Ti
    decfsz bitcnt,f ;18Ti
    goto sendchar ;20Ti
;stop bit last
    bsf  _RS232TX
    call TX_RS232 ;stop bit = 3Ti

    incf FSR,f ;1
    decfsz charcnt,f ;2
    goto inalpha ;4
    movlw d'255'
    movwf charcnt
    movlw d'10'
    movwf bitcnt
waiting
```



# microID™ 13.56 MHz Design Guide

```
    call    ti17
    decfsz charcnt,f
    goto    waiting
    decfsz bitcnt,f
    goto    waiting
    goto    seekinit
inalpha
    call    ti10
    goto    sendA

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;;subroutine--RS232 bit timing & 125 KHz voltage inverter maintenance
;;      baud rate set to 9600 bps--this is a bit time of 104 usec
;;      Timing for this subroutine: to104 loop is 5.605 usec, additional setup
;;                                  overhead is 1.77 usec. If do 17 to104 loops,
;;                                  that leaves 5.844 usec to make up in the calling
;;                                  routine to meet 104 usec target. 5.844= 19.8 Ti
;;                                  (20 Ti)
;;                                  Note that 5.844 is not evenly divisible by the
;;                                  instruction cycle time. Need to save one
;;                                  instruction every 5th bit sent--w/ the stop & start
;;                                  bit overhead, easier to save 2 extra instructions
;;                                  every character sent (10 bits)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
TX_RS232
    movlw   d'17'      ;time out 104 usec, Ti=295 nsec
    movwf   wait

to104
    movlw   invmask    ;flip voltage inverter bit
    xorwf   _RS232,f
    movlw   d'4'
    movwf   delay

wait4usec
    decfsz  delay,f      ;4 usec is half inverter clock period
    goto    wait4usec
    decfsz  wait,f
    goto    to104
    movlw   invmask
    xorwf   _RS232,f
    nop
    nop
    nop
    return

;;=====

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;;subroutine--generates 128 cycles at ~125 KHz for the RS232 voltage inverter
gen125khz
    movlw   d'128'
    movwf   cycle_cnt

next125
    bsf     _125KHZ
    movlw   d'4'
    movwf   delay

highside
    decfsz  delay,f
    goto    highside
    bcf     _125KHZ
```

# microID™ 13.56 MHz DESIGN GUIDE

---



---

```

        movlw d'4'
        movwf delay
lowside
        decfsz delay,f
        goto lowside
        decfsz cycle_cnt,f
        goto next125
        return
;;end gen125khz
subroutine;;;;;;;;;;;;;
;;;;;;;;;;;;;
;

;;;;;;;;;;;;;
;;subroutine-ti17: burn 17 Ti--includes the 2Ti to call this subroutine
;;          ti15: burn 15 Ti, including call
;;          ti10: burn 10 Ti, including call
;;;;;;;;;;;;;
ti17
        movlw d'3'          ;1
        movwf delay        ;2
burn9
        decfsz delay,f
        goto burn9          ;11
        clrwdt              ;12
        nop                 ;13
        return              ;15+2 for call ti17=17Ti

ti15
        movlw d'3'          ;1
        movwf delay        ;2
burn9Ti
        decfsz delay,f
        goto burn9Ti       ;11
        return              ;13+2 for call ti15=15Ti

ti12
        nop
        clrwdt
ti10
        goto dly1           ;2Ti
dly1
        goto dly2           ;4Ti
dly2
        goto leaveti10      ;6Ti
leaveti10
        return              ;8Ti+2Ti=10Ti

;=====
;;initialization
;=====
init
        ;1st set up the I/O configuration--note that setting PORTB 7,6,5,4,0 as outputs disables
        ;them as external interrupt sources. In this application PORTB-4 is utilized as an
        ;external interrupt source upon change of state. All other external interrupt sources are
        ;set as outputs to disable them as interrupts.

;;set up registers in bank 1
        bsf STATUS,RP0      ;point to bank 1
        movlw h'8'
        movwf TRISA         ;RA3 input, RA2-0 output
        movlw h'10'
        movwf TRISB        ;RB7-5,3-0 output, RB4 input

```

# microID™ 13.56 MHz Design Guide

```
        movlw  b'00001000' ;set up timer option for internal clock, no prescaler
        movwf  OPTION_REG  ;port B pullups enabled
        bcf   STATUS,RPO   ;point back to bank 0
;;done setting up registers in bank 1, back to bank 0
        movlw  HIGH isr
        movwf  PCLATH      ;setup for calculated goto's dependent on context when entering
                          ;isr
;=====
;;initialization for sync field search- done @ turn on & after data recovery complete (or failed)
;=====
seekinit
        clrwdt
        movlw  d'19'
        movwf  bitcnt     ;clear the bit storage field
        movlw  recvbits
        movwf  FSR
clrbits
        clrf   INDF
        incf   FSR,f
        decfsz bitcnt,f
        goto  clrbits

        movlw  recvbits
        movwf  FSR        ;start of the received bits field
        movf   PORTB,w    ;read PORTB before clearing INTCON to be sure RBIF=0
        clrf   INTCON
        clrf   TMR0
;=====
; From here on, the w register represents the PCL offset when answering the isr.
; It is to be used for no other purpose until interrupts are disabled.
;=====
        movlw  d'0'
        clrf   PCLATH
        bsf   INTCON,RBIE ;enable portB change interrupt enable
        bsf   INTCON,GIE  ;global interrupts are now enabled.

;=====
;;tag word search
;=====
;The main loop monitors the TOIF flag to detect successfully received word (subject to
;checksum test). Tag word processing is isr driven. A calculated goto method is used for
;position context in tag word for speed. FOR THIS REASON, THE W REGISTER CANNOT BE USED
;BY THE MAIN LOOP! If the main loop detects a timer overflow, the w register is cleared to
;return processing to first sync edge search.
;Also, expect recvbits area to be @ 40h-52h while receiving data. The ptr will be tested to
;determine this bitwise (because w can't be used in the main loop).
;=====
seeksync
        bcf   INTCON,RBIE
        movlw d'0'    ;calculated goto offset for 1st sync edge processing
        clrf  PCLATH
        clrf  FSR     ;FSR = 0 to indicate not gathering bits
        bsf  INTCON,RBIE
        bcf  INTCON,TOIF
main
        clrwdt
        btfsc FSR,6
        goto  datamain ;receiving data, monitor progress
        btfsc INTCON,TOIF
        goto  seeksync ;if TMR0 overflows w/o receiving bits, seeksync
        goto  main
;check for done receiving bits using TMR0 overflow as indicator. Also test for overflow from
;proper bit storage area for runaway condition (non tag noise tripping comparator)
```

# microID™ 13.56 MHz DESIGN GUIDE

---

```
datamain
    clrwdt
    btfsc   INTCON,T0IF
    goto   calc_checksum ;if timer overflows, calculate checksum of received data
    btfsc   FSR,5         ;if bit 5 set, FSR > 5fh and has overrun its proper area.
    goto   seeksync      ;search for sync.
    goto   datamain

;Data received at this point. Two processing tasks remain:
;1- the framing '0' bits must be removed from the received 14 data bytes and 16 bit checksum
;2- the checksum of the 14 data bytes must be calculated and compared to the received
; 16 bit checksum
;If checksums match, transmit data over RS232 link.

calc_checksum
    clrf   INTCON
clrgie
    bcf    INTCON,GIE
    btfsc  INTCON,GIE ;make sure it's clear before proceeding
    goto   clrgie
    movf   PORTB,f
    clrf   INTCON      ;disable all interrupts while processing received data
;remove the framing '0' bits by bit shifting the data array left until all framing 0s are
;shifted out

    movlw  d'17'
    movwf  bitcnt
    movwf  shiftcnt
shiftout
    movlw  recvbits+d'17'
    movwf  FSR
roll_left
    rlf    INDF,f
    decf   FSR,f
    decfsz shiftcnt,f
    goto   roll_left    ;rotate left shiftcnt # of bytes
    decfsz bitcnt,f
    goto   next_RL
    goto   framestripped
;bit shift left through the array (successively 1 byte less each time)
next_RL
    movf   bitcnt,w
    movwf  shiftcnt
    goto   shiftout
framestripped

;1st check for all 0s in data--This is an illegal combination
    movlw  recvbits
    movwf  FSR
    movlw  d'14'
    movwf  bitcnt
zerotest
    movf   INDF,w
    btfss  STATUS,Z
    goto   nonzero
    decfsz bitcnt,f
    goto   zerotest
    goto   seekinit     ;all zeros received. Ignore the message
nonzero
;do 16 bit checksum of first 14 bytes received. It should match the last 2 bytes received.
    movlw  recvbits
    movwf  FSR
    movlw  d'14'
    movwf  bitcnt
    clrf   recv_csumlo
    clrf   recv_csumhi
```

# microID™ 13.56 MHz Design Guide

---

```
sumbytes
    movf    INDF,w
    addwf   recv_csumlo,f
    btfsc   STATUS,C
    incf    recv_csumhi,f ;carry into high byte as necessary
    incf    FSR,f ;point to next data byte
    decfsz  bitcnt,f
    goto    sumbytes
;now compare the received checksum w/ the calculated checksum. Transmit data if they match.
    movf    recv_csumhi,w
    subwf   INDF,f
    btfss   STATUS,Z
    goto    seekinit
    incf    FSR,f ;point to received checksum LSB
    movf    recv_csumlo,w
    subwf   INDF,f
    btfss   STATUS,Z
    goto    seekinit
;message passes checksum. Convert to ASCII and transmit.
;now convert to ASCII form
    movlw   recvbits
    movwf   ptr1 ;keep track of where in conversion
    movlw   sendascii
    movwf   ptr2
    movwf   FSR
    movlw   "G"
    movwf   INDF
    incf    ptr2,f
    incf    FSR,f
    movwf   INDF ;double "G" to indicate start
    incf    ptr2,f ;next ascii character
    movlw   xfercnt ;how many bytes to convert to ASCII
    movwf   bitcnt
    movlw   h'4'
    movwf   PCLATH ;set up PCLATH for lookup table
asciiconv
    movf    ptr1,w
    movwf   FSR
    swapf   INDF,w
    andlw   h'f' ;isolate the MSN
    call    hex2ascii
    movwf   temp ;hold the ASCII character
    movf    ptr2,w
    movwf   FSR
    movf    temp,w ;store ASCII representation of received byte MSN
    movwf   INDF
    incf    ptr2,f ;advance ASCII ptr
    movf    ptr1,w ;back to received bytes
    movwf   FSR
    movf    INDF,w
    andlw   h'f' ;isolate the LSN
    call    hex2ascii
    movwf   temp
    movf    ptr2,w
    movwf   FSR
    movf    temp,w ;store ASCII representation of received byte LSN
    movwf   INDF
    incf    ptr2,f ;advance ASCII ptr
    incf    ptr1,f ;advance received byte ptr
    decfsz  bitcnt,f
    goto    asciiconv
;done data conversion, now indicate newline before sending
    movlw   "\n" ;newline character
    incf    FSR,f
    movwf   INDF
```

# microID™ 13.56 MHz DESIGN GUIDE

---

```
;cleared for RS232 transmission
    goto    alphabet

;hexadecimal to ASCII conversion table
    org     h'3ff'
hex2ascii
    addwf   PCL,f
    retlw   "0" ;ascii 0
    retlw   "1" ;ascii 1
    retlw   "2" ;ascii 2
    retlw   "3" ;ascii 3
    retlw   "4" ;ascii 4
    retlw   "5" ;ascii 5
    retlw   "6" ;ascii 6
    retlw   "7" ;ascii 7
    retlw   "8" ;ascii 8
    retlw   "9" ;ascii 9
    retlw   "A" ;ascii A
    retlw   "B" ;ascii B
    retlw   "C" ;ascii C
    retlw   "D" ;ascii D
    retlw   "E" ;ascii E
    retlw   "F" ;ascii F

    end
```

# microID™ 13.56 MHz Design Guide

## Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable”.
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

## Trademarks


The Microchip name and logo, the Microchip logo, PIC, PICmicro, PICMASTER, PICSTART, PRO MATE, KEELoQ, SEEVAL, MPLAB and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

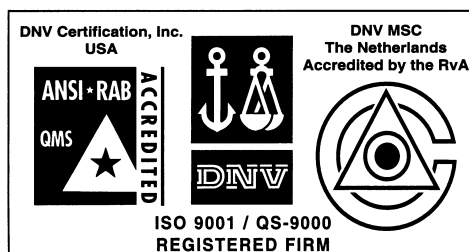
Total Endurance, ICSP, In-Circuit Serial Programming, FilterLab, MXDEV, microID, FlexROM, fuzzyLAB, MPASM, MPLINK, MPLIB, PICC, PICDEM, PICDEM.net, ICEPIC, Migratable Memory, FanSense, ECONOMONITOR, Select Mode and microPort are trademarks of Microchip Technology Incorporated in the U.S.A.

Serialized Quick Term Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2001, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper



Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELoQ® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.



---

## WORLDWIDE SALES AND SERVICE

---

### AMERICAS

#### Corporate Office

2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7200 Fax: 480-792-7277  
Technical Support: 480-792-7627  
Web Address: <http://www.microchip.com>

#### Rocky Mountain

2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7966 Fax: 480-792-7456

#### Atlanta

500 Sugar Mill Road, Suite 200B  
Atlanta, GA 30350  
Tel: 770-640-0034 Fax: 770-640-0307

#### Austin

Analog Product Sales  
8303 MoPac Expressway North  
Suite A-201  
Austin, TX 78759  
Tel: 512-345-2030 Fax: 512-345-6085

#### Boston

2 Lan Drive, Suite 120  
Westford, MA 01886  
Tel: 978-692-3848 Fax: 978-692-3821

#### Boston

Analog Product Sales  
Unit A-8-1 Millbrook Tarry Condominium  
97 Lowell Road  
Concord, MA 01742  
Tel: 978-371-6400 Fax: 978-371-0050

#### Chicago

333 Pierce Road, Suite 180  
Itasca, IL 60143  
Tel: 630-285-0071 Fax: 630-285-0075

#### Dallas

4570 Westgrove Drive, Suite 160  
Addison, TX 75001  
Tel: 972-818-7423 Fax: 972-818-2924

#### Dayton

Two Prestige Place, Suite 130  
Miamisburg, OH 45342  
Tel: 937-291-1654 Fax: 937-291-9175

#### Detroit

Tri-Atria Office Building  
32255 Northwestern Highway, Suite 190  
Farmington Hills, MI 48334  
Tel: 248-538-2250 Fax: 248-538-2260

#### Los Angeles

18201 Von Karman, Suite 1090  
Irvine, CA 92612  
Tel: 949-263-1888 Fax: 949-263-1338

#### Mountain View

Analog Product Sales  
1300 Terra Bella Avenue  
Mountain View, CA 94043-1836  
Tel: 650-968-9241 Fax: 650-967-1590

#### New York

150 Motor Parkway, Suite 202  
Hauppauge, NY 11788  
Tel: 631-273-5305 Fax: 631-273-5335

#### San Jose

Microchip Technology Inc.  
2107 North First Street, Suite 590  
San Jose, CA 95131  
Tel: 408-436-7950 Fax: 408-436-7955

#### Toronto

6285 Northam Drive, Suite 108  
Mississauga, Ontario L4V 1X5, Canada  
Tel: 905-673-0699 Fax: 905-673-6509

### ASIA/PACIFIC

#### Australia

Microchip Technology Australia Pty Ltd  
Suite 22, 41 Rawson Street  
Epping 2121, NSW  
Australia  
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

#### China - Beijing

Microchip Technology Beijing Office  
Unit 915  
New China Hong Kong Manhattan Bldg.  
No. 6 Chaoyangmen Beidajie  
Beijing, 100027, No. China  
Tel: 86-10-85282100 Fax: 86-10-85282104

#### China - Shanghai

Microchip Technology Shanghai Office  
Room 701, Bldg. B  
Far East International Plaza  
No. 317 Xian Xia Road  
Shanghai, 200051  
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060

#### Hong Kong

Microchip Asia Pacific  
RM 2101, Tower 2, Metroplaza  
223 Hing Fong Road  
Kwai Fong, N.T., Hong Kong  
Tel: 852-2401-1200 Fax: 852-2401-3431

#### India

Microchip Technology Inc.  
India Liaison Office  
Divyasree Chambers  
1 Floor, Wing A (A3/A4)  
No. 11, O'Shaughnessey Road  
Bangalore, 560 025, India  
Tel: 91-80-2290061 Fax: 91-80-2290062

#### Japan

Microchip Technology Intl. Inc.  
Benex S-1 6F  
3-18-20, Shinyokohama  
Kohoku-Ku, Yokohama-shi  
Kanagawa, 222-0033, Japan  
Tel: 81-45-471-6166 Fax: 81-45-471-6122

### ASIA/PACIFIC (continued)

#### Korea

Microchip Technology Korea  
168-1, Youngbo Bldg. 3 Floor  
Samsung-Dong, Kangnam-Ku  
Seoul, Korea  
Tel: 82-2-554-7200 Fax: 82-2-558-5934

#### Singapore

Microchip Technology Singapore Pte Ltd.  
200 Middle Road  
#07-02 Prime Centre  
Singapore, 188980  
Tel: 65-334-8870 Fax: 65-334-8850

#### Taiwan

Microchip Technology Taiwan  
11F-3, No. 207  
Tung Hua North Road  
Taipei, 105, Taiwan  
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

### EUROPE

#### Denmark

Microchip Technology Denmark ApS  
Regus Business Centre  
Lautrup høj 1-3  
Ballerup DK-2750 Denmark  
Tel: 45 4420 9895 Fax: 45 4420 9910

#### France

Arizona Microchip Technology SARL  
Parc d'Activite du Moulin de Massy  
43 Rue du Saule Trapu  
Batiment A - 1er Etage  
91300 Massy, France  
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

#### Germany

Arizona Microchip Technology GmbH  
Gustav-Heinemann Ring 125  
D-81739 Munich, Germany  
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

#### Germany

Analog Product Sales  
Lochhamer Strasse 13  
D-82152 Martinsried, Germany  
Tel: 49-89-895650-0 Fax: 49-89-895650-22

#### Italy

Arizona Microchip Technology SRL  
Centro Direzionale Colleoni  
Palazzo Taurus 1 V. Le Colleoni 1  
20041 Agrate Brianza  
Milan, Italy  
Tel: 39-039-65791-1 Fax: 39-039-6899883

#### United Kingdom

Arizona Microchip Technology Ltd.  
505 Eskdale Road  
Winnersh Triangle  
Wokingham  
Berkshire, England RG41 5TU  
Tel: 44 118 921 5869 Fax: 44-118 921-5820

01/30/01

All rights reserved. © 2001 Microchip Technology Incorporated. Printed in the USA. 6/01  Printed on recycled paper.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, except as maybe explicitly expressed herein, under any intellectual property rights. The Microchip logo and name are registered trademarks of Microchip Technology Inc. in the U.S.A. and other countries. All rights reserved. All other trademarks mentioned herein are the property of their respective companies.