User Manual for Development System

MICRF505
MICRF506
MICRF600
MICRF610
MICRF620

FW v. 4

Micrel Norway AS
Strandveien 13 * 1366 LYSAKER * Norway
Tel: +47 67 83 08 00 * Fax: +47 67 83 08 10

Table of contents:

1. Introduction

This document, "MICRFXXX User Manual for Development System" describes the features of the development system for MICRFXXX, where XXX = 505, 506, 600, 610 or 620. The development system is made of 2 "development boards". For details on the MICRF505 and MICRF506 transceivers (products in the RadioWire™ product line): Please refer to the appropriate data sheet. For the latest updates on products related to RadioWire™, please visit www.micrel.com. Development system for:

MICRF505: The MICRF505 chip is used
MICRF506: The MICRF506 chip is used
MICRF600: RF module with MICRF505 on-board, for 915MHz band
MICRF610: RF module with MICRF505 on-board, for 868MHz band
MICRF620: RF module with MICRF506 on-board, for 433.92MHz band

The operation and features of the development system are equal for all parts. The main differences are the frequency band, the RF modulation, the on-the-air bit rate and number of frequency channels:

Dev System	Frequency band	Modulation	On-the-air bitrate (bps)	Manchester coded	Number of frequencies
MICRF505	915MHz	Closed loop VCO mod.	38462	Yes	25
MICRF506	433.92MHz	Closed loop VCO mod	38462	Yes	4
MICRF600	915MHz	Modulation using 2 sets of dividers	19231	No	25
MICRF610	868MHz	Modulation using 2 sets of dividers	15152	No	2
MICRF620	433.92MHz	Modulation using 2 sets of dividers	19231	No	4

Note that all systems have the same firmware version number, and all source files are available in a common set of files (a "Project").

Outline of this document:
Following this introduction, an overview of the board's inputs/outputs is given. Then a Get-started chapter is included, and the different modes of operation are described. An overview of how to program/update the firmware is also included. Finally, a list of changes for this firmware version is given.

Purpose of the Development system:
The development system provides hands-on experience with the MICRFXXX transceiver. The user can use the included firmware (the program used in the micro controller) and hardware, or make a new program and flash it into the micro controller. That is, the user can use the boards both to evaluate the MICRFXXX, and as an aid in the development of a radio communication system.

A separate PC program is available. Through this program, it is possible to set the programming word for the MICRFXXX (both the frequency dividers and the control bits can be set). This program is referred to as "RF TestBench". Please observe: It is possible to read out the firmware version of the development boards via RF TestBench.

Main modes of operation of the development system:

- **RF Test Modes.** The user can test the RF properties of the transceiver. It is possible to transmit a 1010… pattern or a random pattern, transmit a carrier or to stay in receive mode, searching for a 1010… pattern. In addition, the user can select to bypass the MCU completely, and control the RF part via header pins.

- **2-way Link-Test Mode.** In this mode of operation, one board is "Master" and another board is "Slave". Messages (5 printable ascii characters) are transmitted back and forth between the two boards. This mode is useful when testing radio communication in different environments, testing antennas, testing encapsulation etc. 16-bit CRC is implemented. A LED ("LED1") indicates link OK.

- **Simple Byte Transfer Mode.** In this mode, the user can enter a number of bytes into a development board. The board will then transmit the bytes. If no user-bytes are entered, the board will search for RF messages, and, if found, output the message to the user. 16-bit CRC is implemented (messages received through RF are only given to user if the CRC is OK). A 64-byte deep buffer is implemented for both RX and TX. Only 1 frequency is used.

- **Advanced Byte Transfer Mode.** In this mode of operation, one board is "Master" and another board is "Slave". Operates like the "Simple Byte Transfer Mode", except: When transmitting user data, the board will construct a frame by adding frame type, frame ID and sync info. Automatic frequency hopping is implemented. ARQ (Automatic Repeat reQuest), CRC and duplicate control are implemented as well. "Master" is responsible for maintaining the link (by transmitting "timestamps"). If "Slave" is reset or gets out of sync, it requests a "timestamp" from Master.

Development System Features:

- Pre-programmed frequencies:
  - MICRF505: 25 frequencies, f0=903MHz, f1=904MHz, … f24=927 MHz
  - MICRF506: 4 frequencies, f0=433.4MHz, f1=433.7MHz, f2=434.1MHz, f3=434.4MHz
  - MICRF600: 25 frequencies, f0=903.25MHz, f1=924.25MHz, … f24=927.25MHz
  - MICRF610: 2 frequencies, f0 = 868.307692MHz, f1=868.952381MHz
  - MICRF620: 4 frequencies, f0=433.4MHz, f1=433.7MHz, f2=434.0MHz, f3=434.3 MHz
- Default frequency (used in RF test mode and Simple byte transfer mode):
  - MICRF505: 915.00MHz
  - MICRF506: 433.4MHz
  - MICRF600: 915.25MHz
  - MICRF610: 868.307692MHz
  - MICRF620: 433.4MHz
- Automatic hopping
  - In ”2-way Link Test Mode” and “Advanced Byte Transfer Mode”. Development boards jump to a new frequency channel approx 10 times/sec. Channels are used randomly
- Flash-based micro controller (PIC18LF4320) for easy firmware upgrade/testing. Plug directly into ICD2 from Microchip.
- Antenna: External antenna
- User interface: Asynchronous serial interface: RS232-level or logic level I/O pins (9600-8-N-1)
- 4 DIP switches to select mode of operation
- 1 LED to indicate power on
- 1 LED to indicate RF powered
- 4 LEDs to indicate status or control information
- Possible to use 5 external I/O pins (for user FW development)
- Possible to monitor the interface between RF chip and micro controller via header pins
- Possible to measure RF-part power consumption

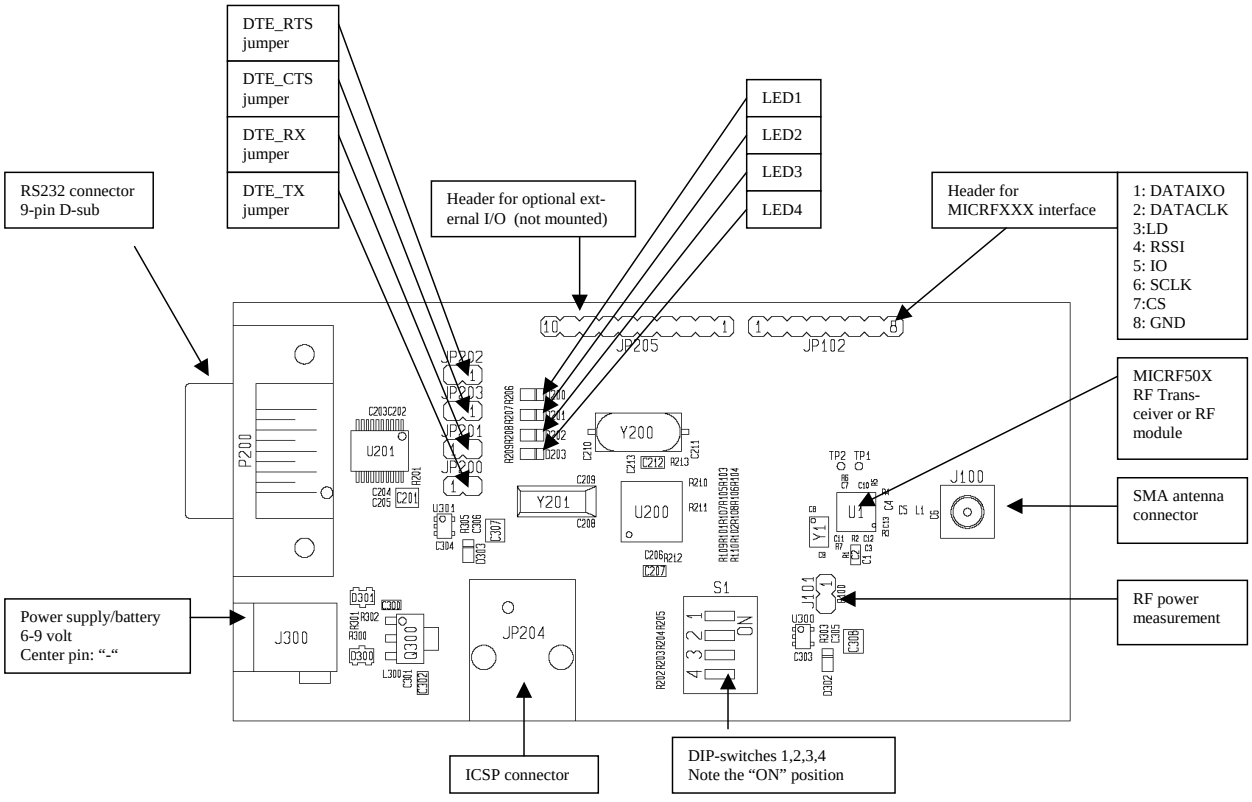
Power supply:

- 6 - 9 volt DC
- Apply “ - “ to the centre pin (note!)

External antenna connector:
Type: SMA

2. Development Board Inputs/Outputs

Please observe: The MICRF505 or -506, -600, -610 or -620 is mounted. The user I/Os are equal for all parts.



2.1. Summary of DIP-Switch Setting

To select mode of operation, bring DIP-switches ON or OFF according to the table below.

The modes of operation (described in detail in the following sections) are:

- "RF Test Modes": Enter RX mode, TX a carrier, TX1010… or a random pattern, enter PC-configurable mode or bypass (tristate) micro controller for user-operation of the RF part.
- "Byte Transfer Mode": Transfer bytes between e.g. 2 PCs. Select "Simple" or "Advanced"
- "2-way Link Test Mode": Use 2 development boards to communicate. If "LED1" goes on, it indicates "Link OK".

The 4 DIP-switches (labelled 1 2 3 4) are placed in a single component. In addition "ON" is written on the component.

Note: The 2-way link test is changed in fw version 4. If one of your development boards is v.3 or older, we recommend to flash v.4 into the micro (available from www.micrel.com). If v.4 is used together with a previous version, select the "Link Test, Pre_v4" setting.

DIP1	DIP2	DIP3	DIP4	Mode of operation
OFF	OFF			=> Byte transfer modes
		OFF	OFF	Simple Byte Transfer Mode
		OFF	ON	reserved
		ON	OFF	Advanced Byte Transfer Mode, Master
		ON	ON	Advanced Byte Transfer Mode,Slave
OFF	ON			=> RF Test Modes
		OFF	OFF	Bypass micro controller
		OFF	ON	TX a random, Manchester-coded pattern
		ON	OFF	reserved
		ON	ON	reserved
ON	OFF			=> 2-Way Link Test Modes
		OFF	OFF	LINK_Test_Master
		OFF	ON	LINK_Test_Slave
		ON	OFF	Pre_V4: LINK_Test_Master
		ON	ON	Pre_V4: LINK_Test_Slave
ON	ON			=> RF Test Modes
		OFF	OFF	RF_Test_Rx
		OFF	ON	RF_Test_Tx1010
		ON	OFF	RF_Test_TxCarrier
		ON	ON	Enable Configuration via PC

3. Getting Started

Before power-up of the development board:

- Get familiar with the inputs/outputs (refer to “Development Board Inputs/Outputs”)
- Select mode of operation via DIP-switches
- If Byte Transfer Mode (“Simple” or “Advanced”) is selected: Select interface: Insert the 4 jumpers “DTE_RTS”, “DTE_CTS”, “DTE_RX”, “DTE_TX” to select RS232 interface. Remove the jumpers to select “logic level” interface (connect to e.g. another micro controller).

To get familiar with the boards, the following sequence is suggested:

First, try the 2-way Link Test Mode. On one board, set only DIP1 ON. This will be the “Master”. On the other board: Set DIP1 and DIP4 ON. This will be the “Slave”. Try to increase/decrease the distance between the boards, and observe LED1 at the boards. If LED1 at the Master is on: A 2-way link is established (from Master to Slave and back to Master). LED1 at the Slave will be on as long as the link from Master to Slave is OK.

Then try the RF Test Mode. Set one board to transmit 1010 … (DIP1, DIP2 and DIP4 ON), and another board to receive (DIP1 and DIP2 ON). Observe LED1 on the receiving board (indicates received 1010… OK). Optionally, use an oscilloscope to observe the DATACLK and DATAIXO pins (refer “Development Board Inputs/Outputs”).

Set the boards in Simple Byte Transfer Mode (all DIPs OFF). Connect each board in Simple Byte Transfer Mode to a PC running HyperTerminal. Send characters or text files between the PCs. Note: No ARQ is used – number of “data packets” getting through depends on link quality. Make sure the cable is 1:1, and that RTS, CTS, TX, RX and GND are used. Make sure the data format is 9600-8-N-1 and hardware handshake is used. Also make sure the jumpers for RS232-use are in place.

Finally, set two boards in Advanced Byte Transfer Mode. Set one as “Master”, the other as “Slave”: Only DIP3 ON for the “Master”, both DIP3 and DIP4 ON for the “Slave”. Select RS232 interface (that is, make sure RS232 jumpers are in place) and connect the boards to PCs. Start HyperTerminal (or other “terminal” program) and observe that text written on one PC are received and displayed on the other one and v.v. Try to transmit a textfile (note: textfile) from one PC to the other. ARQ is used in this mode; that is: all “data packets” should get through.

4. RF Test Modes

In RF Test Modes, only 1 frequency is used. If not changed via the PC program, the frequency is:

MICRF505: 915.00MHz
MICRF506: 433.40MHz
MICRF600: 915.25MHz
MICRF610: 868.307692MHz
MICRF620: 433.4MHz

DIP-switch setting:

Combin. #	DIP1	DIP2	DIP3	DIP4	Text
1	ON	ON	OFF	OFF	Receive mode only
2	ON	ON	OFF	ON	Transmit 1010 …
3	ON	ON	ON	OFF	Transmit carrier
4	ON	ON	ON	ON	Enable configuration via PC
5	OFF	ON	OFF	OFF	Bypass micro controller
6	OFF	ON	OFF	ON	Transmit a random pattern

Combination #1, Receive mode only:
The board will continuously search for a 1010… pattern.

The board utilizes the on-board bit synchronizer feature of the MICRF50X.

If a 1010… pattern is received: LED1 will be on. If an error in the 1010… pattern is detected, LED1 will be off for at least ½ second. The LED1 "off-time" is set to ½ second to let the user visually observe an error.

The received data is also clocked out directly to a header. DATACLK and DATAIXO are available at the header close to the antenna-side:

Test points in header; Pin #8 is the one near the "antenna-side":

8 Gnd
7 CS (Chip Select)
6 SCLK (Programming clock-line)
5 IO (Programming data-line)
4 RSSI (Received signal strength indicator; analogue level)
3 LD (Lock detect)
2 DATACLK (Data clock)
1 DATAIXO (Data input/output)

Combination #2, Transmit 1010… :
The board will enter transmit mode and transmit a 1010… pattern. This can be combined with Combination #1 (Receive mode) to test range and error rate. Combination #2 can be used to test parameters like deviation and transmit spectrum.

Combination #3, Transmit carrier:
The board will enter transmit mode and transmit a carrier, that is: no modulation is applied. This can be used to check the frequency spectrum, the output power, and the current consumption.

Combination #4, Enable configuration via PC:
This setting is also referred to as "PC mode".

Refer to the chapter called "PC-Program" for details on the PC program.

Make sure the 4 jumpers "DTE_RTS", "DTE_CTS", "DTE_RX" and "DTE_TX" are in place.

If the power-LEDs are off: Make sure the board is powered-on without the RS232 cable connected, and then connect the cable.

Entering "PC mode": Before any command is given from the PC, the RF chip will stay in receive or transmit mode, depending on the selected mode before entering "PC mode".

While in "PC mode": When a control word is sent from the PC to the board, the board will store the entered word in EEPROM and program the RF chip with it.

It is possible to transfer a control word from 2 dialog boxes (press the "Transfer" button inside the dialog box):
Main menu →RF TestBench →Quick Setup or
Main menu →RF TestBench →Complete Setup

When leaving "PC mode": The board will use the default control word already programmed into the flash program memory, or it will use the control word entered from the PC program and stored in the EEPROM. Which one to select (the source of the control word) is selected in the PC program (select→RF TestBench →Development Board Commands)[1].

From the PC program, try e.g. to change the output power level (PA setting) and observe the effect on spectrum, power consumption and link range.

Combination #5, Bypass Micro Controller:
Using this combination, it is possible to use an external micro controller to control the RF part. Connect to the interface pins as described in "Development Board Inputs/Outputs".

[1] The "EEPROM control word" can be used in "RF Test Modes", "2-way Link Test Mode" and "Simple Byte Transfer Mode". It will not be used in "Advanced Byte Transfer Mode". If the "EEPROM control word" is used in "2-Way Link Test Mode": No frequency jumping is done – only the control word stored in EEPROM will be used.

All I/O pins of the on-board micro are tristated. New in v.4: None of the MCU's IOs are set as output at power-on, if this combination is selected. That is: The User has full control, without the MCU interacting at all (not programming the RF chip etc). The DIPs are tested before the IOs are initiated.

Combination #6, Transmit a Random Pattern:

The board will enter transmit mode and transmit a random, Manchester-coded pattern (i.e. without a DC component). Combination #6 can be used to test parameters like deviation and transmit spectrum.

Suggestions for RF Testing:

1) Set one board to transmit a carrier. Use a spectrum analyser to observe the frequency spectrum. Repeat for transmit 1010…
2) Set one board to receive mode and another to transmit 1010… Use an oscilloscope to observe the tx'ed and rx'ed data (Data I/O). Set tx'ed data as the trigger source.
3) Set one board in receive-mode. Turn the other board off and use a signal generator to input a 1010… pattern. Increase/Decrease the output power level from the signal generator and observe LED1 and the rx'ed data. This can be used to test the sensitivity: As long as the LED1 is on, the board is able to read a 1010… pattern. Although a random pattern should be used in sensitivity measurements, this test gives an indication of the possible detectable signal level. Note: Due to the long off-level when an error is detected, this method will only give an indication of the "zero-error" input level. For example, a BER (bit-error rate) of 10^{-3} can't be read using this "LED on" method.

5. 2-way Link Test Mode

DIP-switch setting:

Combin. #	DIP1	DIP2	DIP3	DIP4	Text
1	ON	OFF	OFF	OFF	Master
2	ON	OFF	OFF	ON	Slave

Note: This mode is changed in version 4. Please refer to "Summary of DIP-Switch Setting".

In this mode of operation, one card is "Master", and the other is "Slave".

If not overruled from the PC-program, all pre-programmed frequencies are used (refer to "*RF Test Mode: Enable configuration via PC"*. Automatic hopping between the frequencies is implemented. For the MICRF505/MICRF600: Made to meet the FCC part 15.247 regulations.

If < 25 frequencies are pre-programmed: The frequency table still holds 25 entries. That is, several "frequency indexes" points to the same frequency (Refer to "Introduction".)

In this mode of operation, a "message" is ping-pong'ed between the boards. The message is build of 5 printable ascii characters:

- A letter 'a'… 'y', indicating the frequency index for the frequency used (not the frequency number, but the index in a look-up table)
- 2 letters = "BC"
- 'm' or 's, if the message is sent form a master or a slave, resp.
- carriage return (0x0D)

Combination #1, Master:

The following procedure is implemented for a "2-Way Link Test Master":

LOOP
    Enter transmit mode on frequency *n*
    Transmit "Message" and CRC checksum
    Enter receive mode on frequency *n*
    Search for ack. Ack = "Message"
    If ack found: LED1 on
    If no ack for 5 attempts: LED1 off
    Use "next frequency" (randomly selected) next time
END_LOOP

CRC (16 bits) is used.

LED1 will go on if an ack is received, and will go off if no ack is received during 5 attempts.

Note: If LED1 at Master is ON, it indicates 2-way communication OK, because:

- Master has to transmit message correctly,
- Slave has to receive message correctly,

- Slave has to transmit ack correctly, and finally,
- Master has to receive ack correctly – only then is LED1 set ON.

A new message is transmitted every 100 msec (approximately).

Combination #2, Slave:
The following procedure is implemented for a "2-Way Link Test Slave":

```
LOOP
        Enter receive mode on frequency n
        Search for "Message"
        If "Message" found:
                LED1 on
                Enter transmit mode on frequency n
                Transmit ack and CRC checksum. Ack = "Message"
        If no "Message" for 5 attempts: LED1 off
        Use "next frequency" (randomly selected) next time
END_LOOP
```

CRC (16 bits) is used.

LED1 will go on if a message is received, and will go off if no message is received during 5 attempts.

Note: If LED1 at Slave is ON, it indicates 1-way communication OK, because:

- Master has to transmit message correctly and
- Slave has to receive message correctly – only then is LED1 set ON

After reset or if Slave gets out of Sync, it starts to jump "slowly": It stays on every frequency a "long time": At least long enough for master to transmit a "Message" on all the frequencies.

Master/Slave sync mechanism in 2-way Link Test:
Both Master and Slave have an internal timer with a period of approx. 100 msec.

The internal timer of the master is never changed. Whenever a period is reached (timer wrap-around), the Master will enter transmit-mode, transmit a message, turn to receive mode and search for "ack". If ack is received or not received: At the next wrap-around, a "new" frequency is entered and the procedure is repeated.

The internal timer of the Slave is reset whenever a message from the Master is received. Then the Slave enters transmit-mode and transmits "ack", then it enters receive mode at the next frequency and start to search for a new message.

6. Interface in Byte Transfer Modes

In “Simple” or “Advanced” Byte Transfer Mode: Interface to the board in the following way.

Select interface:

- Insert the 4 jumpers (A, B, C, D) to select RS232 interface.
- Remove the 4 jumpers (A, B, C, D) to select logic levels. Connect jumper pins directly to e.g. a micro controller).

The board is treated as a DCE (data communications equipment).

The connected user is treated as a DTE (data terminal equipment).

That is, a DCE-DTE cable should be used. This cable should be 1:1, that is: Connect pin ***i*** on the DCE side to pin ***i*** on the DTE side. The following pins are used (referred to a 9-pin connector, names related to DTE):

Pin # (both DTE and DCE)	Name (DTE side)	Direction DTE DCE
2	RX	←
3	TX	→
7	RTS	→
8	CTS	←
5	Ground	

Interface setting:

Baudrate	9600
Number of databits	8
Parity	None
Number of stopbits	1
Handshake:	RTS/CTS

The interface works as follows:

Note that “active state” of RTS and CTS is “low” (a logical “0” seen by the micro controller).

RTS is an output from DTE (the user). It is used to control data flow from DCE (the board) to DTE. DTE brings RTS active to signal “DTE ready for data bytes from DCE”. DTE brings RTS inactive to signal “DTE not ready for data bytes from DCE”. Before DCE outputs bytes to DTE, the RTS line is tested. If RTS is active, DCE knows that DTE is ready to receive bytes, and DCE transfer the bytes on the RXD line.

CTS is an input to the DTE. It is used to control data flow from DTE to DCE. DCE brings CTS active to signal “DCE ready for bytes from DTE”. DCE brings CTS inactive to signal “DCE not ready for bytes from DTE”. Before DTE outputs bytes to DCE, the CTS line is tested. If CTS is active, DTE knows that DCE is ready to receive bytes, and DTE transfer the bytes on the TXD line.

Maximum number of bytes to transmit in one “data frame” is 64 bytes. The board will tell user to “stop enter bytes” by setting CTS inactive when the “bytes-from-user buffer” holds 64 bytes. Note: the board can buffer 2 additional bytes. Other bytes will be lost.

The bytes can be entered with random delay between them. As soon as a byte is entered, the board enters transmit mode and send the presently entered bytes, while still buffering new bytes from user (note that the board does not wait until buffer is full or until user stops entering bytes).

7. Simple Byte Transfer Mode

DIP-switch setting:

Combin. #	DIP1	DIP2	DIP3	DIP4	Text
1	OFF	OFF	OFF	OFF	

Note: There is no “Master” or “Slave” in this mode of operation.

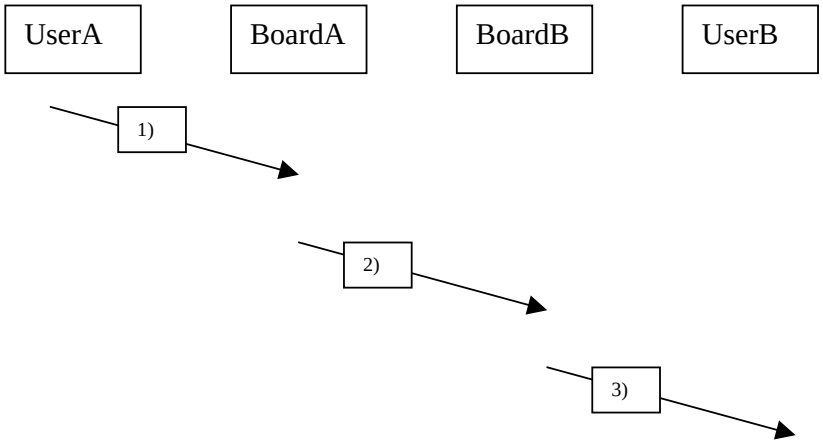
Combination #1:

Refer to interface description in “Interface in Byte Transfer Modes”.

- There is no difference between “master” and “slave” in this mode.
- Error detection (CRC 16 bit) is implemented.
- Only 1 frequency is used (no frequency hopping)
- Duplicate control or ARQ (packets being acknowledged or re-transmitted) are not implemented in this mode.

Data flow from UserA to BoardA , then to BoardB and finally to UserB:

1) If Board A is ready to get bytes (CTS is active): UserA enters bytes into BoardA
2) Bytes are buffered in BoardA. When 1 or more bytes are entered, BoardA constructs a frame (it adds preamble, sync, length and CRC to the presently entered bytes). BoardA then transmits the frame and BoardB receives it. If >=64 bytes in buffer, then user is told to stop entering bytes (CTS goes inactive).
3) When the frame is completely received by BoardB: The CRC is tested. If CRC is OK: BoardB gives the data to UserB if UserB is ready (RTS is active)



Example, how to use Simple Byte Transfer Mode:

User A (connected to board A) wants to transfer 6 bytes (= “Hello!”). User B (connected to board B) receives the bytes.

User A:

- Test if CTS at board A is active. If CTS is active, the board is ready to accept bytes from the user.
- Transfer “Hello!” into the board, using 9600-8-N-1. Transfer lsb of every byte first (after the start-bit).

Board A:

- Read bytes from User A
- Format a packet to transmit 1-6 bytes of “Hello!”)
- Enter transmit mode
- Transmit the packet using Manchester encoding
- Enter Receive mode
- If more bytes entered by user: Format a new packet and transmit it

Board B:

- Receive a packet
- Un-pack the bytes from the packet
- Test that the number of bytes is a legal number of bytes (1-64)
- Test that CRC is OK
- Test if RTS is active. If RTS is active, the user is ready to accept bytes from the board.
- Transfer the bytes to User B. Transfer lsb of every byte first.

User B:

Read bytes from board

Note:
Since no addressing is used, other users (let’s say User C, User D and User E) will receive the 6 bytes as well. If the boards are used in a network, the message has to include some address information (and the user’s protocol has to handle it).

8. Advanced Byte Transfer Mode

DIP-switch setting:

Combin. #	DIP1	DIP2	DIP3	DIP4	Text
1	OFF	OFF	ON	OFF	Master
2	OFF	OFF	ON	ON	Slave

Make sure one board is master and one is slave.

This mode of operation has a similar functionality as “Simple Byte Transfer Mode”; in addition, some “advanced” features are included (described below).

Error detection (CRC 16 bit) is implemented.

Stop-and-wait ARQ (packets being acknowledged or re-transmitted) is implemented. This means: A “source” (Master or slave) sends a frame to a “destination” (master or slave). The frame must be ack’ed by the destination, or else the source will retransmit the frame. If the source is Slave, it will give up transmitting after approximately 10 seconds (the time-out for no sync-info from master). If the source is Master, it will not quit until DIP-switches are changed or the board is powered off.

Duplicate control is implemented. If the frame-ID of a “data frame” is equal to the last received frame-ID, the frame is identified as a duplicate. The frame is ack’ed, but no data is given to user (since the user already has received it).

Note: After power-on the boards establish contact (get synchronised). The master board keeps the boards in synch by transmitting a timestamp every 4 seconds (approximately). If the slave does not get a timestamp for 10 seconds (approximately), it starts to transmit timestamp requests.

In Advanced Byte Transfer Mode, LED3 is set on when a message is transmitted, and LED4 is set on when a message is received. This is done to visualize the RF activity. Please read the examples below.

Example 1: Every time Master transmits a “timestamp”, a short flash can be seen on LED3 at the Master-board. And, if the Slave receives the timestamp, a short flash can be seen on LED4 at the Slave-board.

Example 2: When a Slave is reset, it starts to transmit requests for timing info. This can be seen as short flashes on LED3. Then, when the slave receives timing info from Master, a short flash can be seen on LED4, and the flashing of LED3 stops. This can be used to visually identify “Master and Slave are sync’ed”. Sync time for a slave being powered off and on, should be < 1 second (depending on link quality, it may take several seconds).

Example 3: When a Master is reset, it will transmit timestamps for approx 3 seconds to be sure Slave is resync’ed. The LED3 will flash during this period (it appears to be on, due to short time between transmissions). The Slave will flash LED4 for every timestamp it receives (it will not receive all, because the jump-pattern is updated when a timestamp is received).

Automatic hopping between frequencies is implemented. For MICRF505/MICRF600: To meet the FCC part 15.247 regulations. The units will jump approx. 10 times/second:

For the MICRF506, -610 and -620: less than 25 frequencies are used, but the frequency table holds 25 entries. That is, several “frequency indexes” points to the same frequency.

There is an internal timer in both Master and Slave. The timer is a 3-byte (24-bit) timer; the 3 bytes are called Timer_Upper, Timer_High and Timer_Low. A Timer_Low overflow will generate an interrupt every 0.37msec approx (with a 11.0592MHz xtal connected to the micro controller). In the interrupt service routine (ISR) the Timer_High and Timer_Upper registers are updated.

Timer_High will overflow every 256*0.37msec = 94,8msec. This overflow is used to indicate “time-to-change-frequency” (a flag is set in ISR, the flag is tested in the main program).

Whenever Timer_High overflows, Timer_Upper is incremented.

When the main program detects time-to-change-frequency, it uses Timer_Upper as an index in a frequency look-up table, where the frequencies are randomly stored. That is: The next index will not give the next frequency, but a random frequency.

The internal timer of the master is never changed. The internal timer of the slave is updated whenever a frame from master is received.

9. PC program " RF TestBench"

The PC program is referred to as "RF TestBench".

The RF TestBench can be used without the development board. It can be used to calculate fields in the control word to enter into MICRF50X. Examples: Find settings for a specific bit rate, or get the frequency dividers for a number of frequencies.

"RF TestBench" should be self-explaining, refer to the user guide for the PC program as well.

Installation and running are straightforward:

- Copy the files into a separate directory
- Change filenames from ".eee" to ".exe" (2 files) and from ".bbb" to ".bat" (1 file)
- Start the program by double-clicking "PCxx.bat" (xx = sw version number)

If using RF TestBench together with a development board:

Make sure the DIP-switches are set like this:

Combin. #	**DIP1**	**DIP2**	**DIP3**	**DIP4**	**Text**
1	ON	ON	ON	ON	

Using RF TestBench:

- Power on the board

- Connect the development board to a PC via a cable:
  - 1:1 RS232 cable
  - Make sure RTS, CTS, TX, RX and GND are included in the cable (pins 2, 3, 5, 7, 8)
  - Note: Power-on the board before connecting the cable to the board if the power-LEDs do not get on.

- Start RF TestBench by double-clicking "PCxx.bat" (xx = sw version number)
- Read the welcome-message
- From the main menu, select serial port (COM1 or COM2) via "Setup" → "Communication port"
- From the main menu, select device from "Setup". Select MICRF505, -506, -600, -610 or -620.

- From the main menu, select "RF TestBench" → "Quick Setup" or →"Complete Setup":
  - Select "Quick Setup" if you want the program to calculate parameters based on basic parameters like xtal frequency, RF frequency and bit rate. Enter values and press the "Calc!" button. To see the resulting control word fields, press the "See Details" button (this opens the "Complete Setup" window).
  - Select "Complete Setup" to enter the control word fields manually. Press the "Calc" button to see resulting frequencies etc. Press "More Info" button to get more information.
  - Note that the complete resulting control word can be found in "RF TestBench" → "Overview"

  - If both “Complete” and “Quick” windows are open: Changes in one window are reflected in the other.

- To transfer a control word to the development board from “Quick Setup” or “Complete Setup”: Enter/change the fields and press “Transfer!”. The entered values are stored in the micro’s EEPROM.

- Observe the “Status” field. If “Success!” is not displayed, please check connections/power and try again.

In the following, some suggestions are made for the user to get familiar with “RF TestBench”.

- Experiment with e.g. power levels. Suggestion: Change power level, press “Transfer!” and then give a “TX1010” command. Observe the result on a spectrum analyser, or use the other board in the RX RF-test mode, and observe change in range (distance between transmitter and receiver).

- Try to give a “Transmit 1010” command (→RF TestBench →Development Board Commands).

- It is possible to use the pre-programmed settings (stored in the flash program memory) or the settings stored in EEPROM for the other modes of operation as well (that is: not only in “PC-mode”). Select “Flash” or “EEPROM” via a command in RF TestBench (→RF TestBench →Development Board Commands). Exception: In Advanced Byte Transfer Mode, only the pre-programmed settings can be used. Also note: If EEPROM settings are used in 2-way Link Test Mode, no frequency hopping will be done – only the frequency and control bits stored in EEPROM will be used.

- If “EEPROM settings” is selected, the board can be disconnected from the PC, and the different modes of operation can be run- and now the settings stored in EEPROM are used. To use the pre-programmed settings again: Enter “PC-Mode” (DIP-setting described above) and give command from RF TestBench: Instruct the board to use the flash memory as the source (where to get the control word from).

- Other possibilities are (→RF TestBench →Development Board Commands):
- Read out the firmware version number (the firmware stored in flash program memory of the micro controller)
- Restart micro controller (uC) program (like a power-on reset)
- Reset EEPROM control word - settings to “default” (= pre-programmed settings)
- Read out the present EEPROM control word

10. Firmware change/upgrade

The development boards are equipped with a socket for ICSP (in-circuit serial programming).

Through ICSP, it is possible to download new firmware into the micro controller.

The download feature is made for use with MPLAB ICD2 from Microchip. Connect the development board directly to the ICD2 using the cable from the ICD2 kit.

Via ICD2, it's possible to step through and debug firmware code as well. That is: Make your own program, download it, debug it and run it!

MPLAB IDE (integrated development environment) is available for free from Microchip's homepage. (C-compiler must be bought).

How to upgrade firmware ("xxx.hex", xxx=firmware name) (skip reading this if you are familiar with ICD2):

- Connect ICD2 to your PC and to power (please refer to ICD2 user manual)
- Connect ICD2 and development board via the "standard ICD2 cable"
- Start MPLAB IDE
- Select Configure → Select Device… → "PIC18F4320" → OK
- File →Import… → "xxx.hex" → Open
- Programmer → Select Programmer → MPLAB ICD2
- Programmer → Settings → Power Tab : Make sure "Power target circuit from MPLAB ICD2" is checked
- Programmer → Settings → Program Tab : Make sure all fields are checked in the "Select Memories" box. → OK
- Identify and press the "Reset and connect to ICD" button in the main menu field.
- If the "Target not found" warning pops up, repeat steps above.
- Identify and press the "Program target device" button in the main menu field
- Wait for programming to finish (refer to the "Output" window) and remove the development board from ICD2.
- Optional: Confirm firmware version via RF TestBench

11. Changes in this Firmware Version

V4 2005 06 21 PKB

- Included 610 and 620 Modules
- Link test payload 5 bytes
- Previous Link Test still available
- Read DIPs before setting IOs: This will make sure all pins are kept in tristate if "Bypass MCU" is selected at power-on
- Included bit-error-rate measurement as a command from PC program (requires PC program v.7)
- Not programming the complete control word into the RF chip every time
- Several code upgrades, refer to main.c

- A single "Project" is made, including all RF parts. After opening the project (in MPLAB IDE): Please make sure to #define the correct part in "p18def.h".