Freescale Semiconductor
Application Note

Document Number: AN4116
Rev. 0, 07/2010

Using MM/JE Flexis Families for Infrared Communication

by: Carlos Casillas
RTAC Americas
Guadalajara, Mexico

Contents

1 Introduction

This application note explains the hardware that facilitates the infrared communication implementation provided by Freescale's MM and JE Flexis microcontroller families. It is implemented porting the communication protocol already implemented on the MCF51EM256, explained in the Freescale application note AN3938, "Using the MCF51EM Family for Infrared Communication." Protocol structure and customization process are included in that document. It is highly recommended to first read section 4.2 of that document to help in understanding this application note.

This application is implemented in these devices which use Tower modules:

- MC9S08MM128 → TWR-S08MM128-KIT
- MCF51MM256 → TWR-MCF51MM-KIT
- MC9S08JE128 → TWR-S08JE128-KIT
- MCF51JE256 → TWR-MCF51JE-KIT

2 Hardware for infrared communication implementation

As with the MCF51EM256, microcontrollers in the MM and JE Flexis families have features that help with infrared communication implementation, such as:

- Output driving
  Transmission pins can drive up to 50 mA, allowing an IR diode to be connected simply by adding a series resistor.
- Signal conditioning
  Reception pins can be internally connected to an analog comparator (ACMP). This is useful for conditioning the infrared analog signal to a digital signal.
- Modulation
  Serial Communication Interface (SCI) transmission pins are capable of being internally modulated through a timer output.
- Dual flash array
  Both devices have two flash memory arrays, allowing erasing and writing one of them while code is executing from the other.

Similar characteristics for each device are summarized in Table 1.

Table 1. MCF51EM256 and MM/JE device comparison

MCF51EM256	MC9S08MM128, MCF51MM256, MC9S08JE128, MCF51JE256
Two arrays of 128 KB each	Two arrays of 128 KB each (Coldfire V1) Two arrays of 64 KB each (S08)
Three Serial Communication Interfaces (SCI): • Two transmission pins support modulation • Two reception pins support input through analog comparator	Two Serial Communication Interfaces (SCI): • One transmission pin supports modulation • One reception pin supports input through analog comparator
Modulation sources for transmission are: • TPM channel 0 • TPM channel 1 • MTIM2 • MTIM3	Modulation sources for transmission are: • TPM1 channel 0 • TPM1 channel 1 • TPM2 channel 0 • TPM2 channel 1
Pin functions are configured in PTxPF registers	Pin functions are automatically configured when a module is enabled

Figure 1 shows the internal connections of the Serial Communication Interface to other modules to help to implement infrared communication with MM/JE Flexis families.

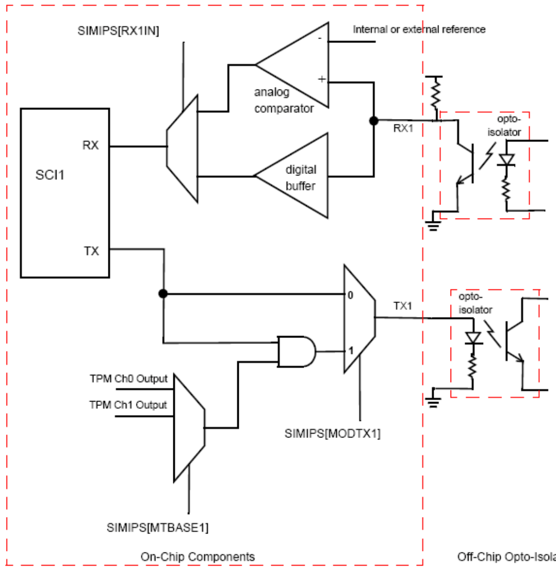

Figure 1. On-chip signal conditioning associated with SCI RX and TX pins

3 Protocol overview

The protocol is based on data interchange from and to information tables. These tables have properties that allow partial or complete read and write operations.

In this implementation, the example includes four tables of data, three located in RAM (which lose their value if the system is restarted or powered off) and one located in flash (to conserve its value even if the system is powered off or restarted). More tables could be added, as is shown in the example included in section 4.2.3.1 of application note AN3938, previously mentioned.

Also, the protocol is based on the ANSI C12.18 command structure. In this application note here are the services/commands that are implemented:

1. Full read: Reads a complete data table.
2. Offset read: Reads a portion of a table, specifying the offset to be added to the table origin and the amount of data to be read.
3. Full write: Provides a fast command to write a complete data table.
4. Offset write: Useful to write a portion of a table.

For more details about packet, command and response structures, please refer to section 4.2.1, “Packet structure,” of the application note AN3938.

4 Application execution

For this implementation, data is sent through an ASCII interface, allowing the use of a simple terminal to send and receive data in hexadecimal format. This ASCII interface can be removed easily to support binary communication. The general block diagram of the application is shown below in Figure 2.

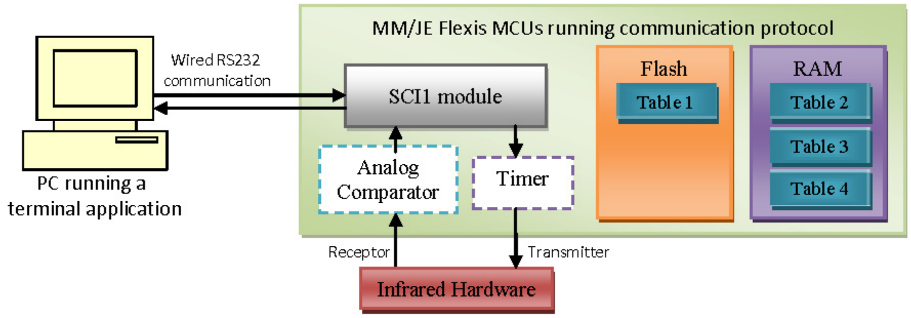

Figure 2. General block diagram

These steps must be used to execute the protocol.

1. Assemble the Tower System with one of the MM/JE boards, the two elevators, and the Serial module.
2. Connect a mini-USB cable to the USB connector of the MM/JE board (it is the open-source BDM), and an RS-232 cable in the DB9 connector of the serial module, as shown in Figure 3 and Figure 4. Make sure that jumper J16 (USB MODE) of the serial module is connecting pin 1 and pin 2.

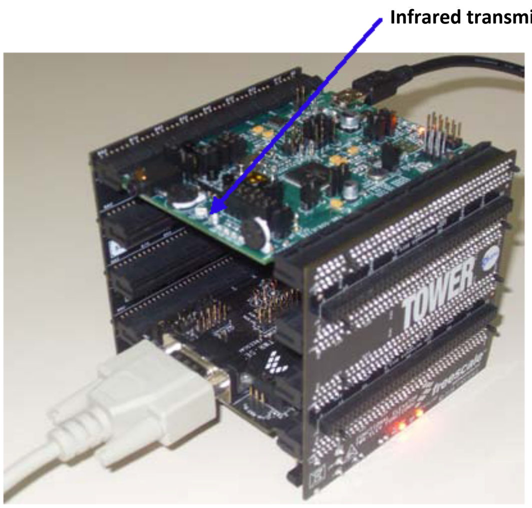

Figure 3. MC9S08JE128 Tower board

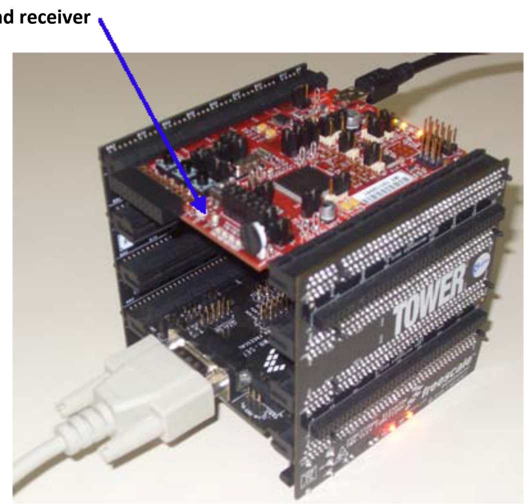

Figure 4. MCF51MM256 Tower board

1. Connect the cables to a PC and open Freescale CodeWarrior.
2. Depending on the device being used, open one of the four projects included in the AN4116SW.zip file available with this application note at freescale.com.
3. Click on the Debug button to download the application to the MCU.
4. Open the HyperTerminal application or another application to connect to the serial port.
5. Adjust the baud rate, defined in the file com_protocol.h, to match the MCU.
6. The number of stop bits and parity are also configurable. Flow control is not used.
7. Send the correspondent command through the terminal window and the MCU will respond. Figure 5 shows a list of commands and responses.

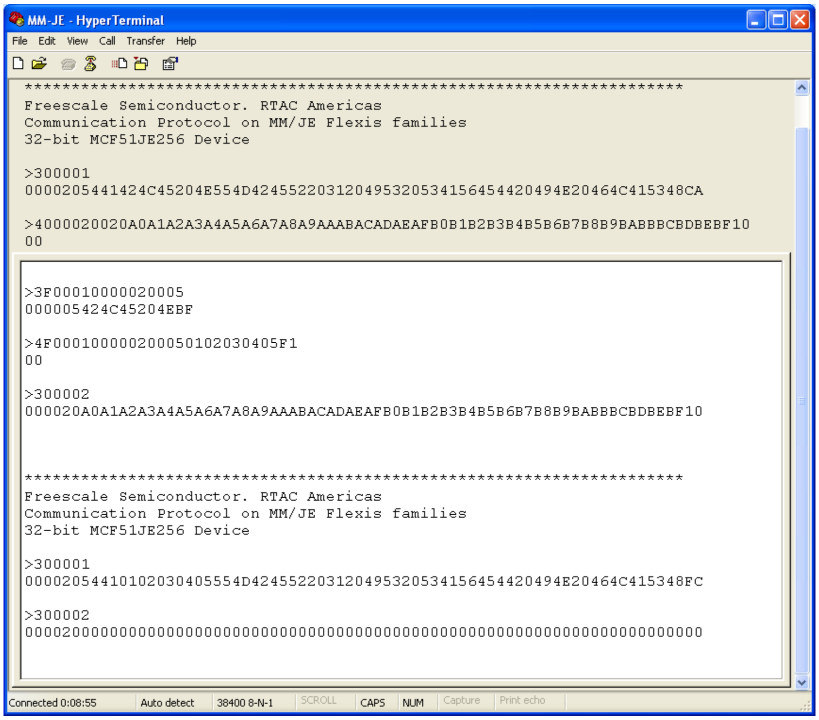

Figure 5. Terminal application showing the protocol executing commands

- The first command is Full Read on Table 1.
- Next command is Full Write on Table 2.
- Next command is an Offset Read command on Table 1, followed by an Offset Write command on Table 1.
- Then a reset was applied to the board, and the protocol was restarted.The next command, a Full Write of Table 1, shows the nonvolatile data stored in Table 1. But the data in Table 2 was lost, as shown in the last command, a Full Read of Table 2.

5 Migrating the communication protocol

In this section the differences between software for MM/JE devices and software for MCF51EM devices are shown.

5.1 Modifications in linker files

This section explains the differences between linker files for S08 and for Coldfire V1 projects.

5.1.1 Linker file in Coldfire V1 projects

For Coldfire V1 projects, the linker file is project.lcf. In this file the modification consists of renaming flash array two for use as non-volatile storage. The default memory ranges in the file of a new project are shown here:

```
MEMORY {
   code           (RX)  : ORIGIN = 0x00000414, LENGTH = 0x0001FFE8
   code_00020410  (RX)  : ORIGIN = 0x00020410, LENGTH = 0x0001FBF0
   userram        (RWX) : ORIGIN = 0x00800000, LENGTH = 0x00008000
}
```

The same lines after the required modifications are:

```
MEMORY {
   code         (RX)  : ORIGIN = 0x00000414, LENGTH = 0x0001FFE8
   my_tables    (RWX) : ORIGIN = 0x00020410, LENGTH = 0x00000400
   userram      (RWX) : ORIGIN = 0x00800000, LENGTH = 0x00008000
}
```

The length is 0x400 because that is the size of a memory page. In addition, a new memory section is created to allocate non-volatile tables:

```
#-----------------------------------------------------------------------------------------
#-----------------------------------------------------------------------------------------
  .rom_symbols :
        {
                __ROM_SYMBOLS =   . ;   #start address of the 4x1024 bytes symbols
                *(.romsymbols)
        } > my_tables

   #FSL:copy during runtime
  ___ROM_SYMBOLS   =       __ROM_SYMBOLS; #ADDR(.my_tables);

#-----------------------------------------------------------------------------------------
#-----------------------------------------------------------------------------------------
```

Complete details about these modifications are explained in Freescale document AN3938, and the modified file is included in the entire project supplied in the software for this application note.

5.1.2 Linker file in S08 projects

For S08 projects, the linker file is named project.prm. In this file the modification consists of commenting out PPAGE 4 to avoid storing code in this memory page. This is necessary because it is erased and reprogrammed in the application.

```
PAGED_ROM   /* routines which can be banked */
            INTO  PPAGE_0_1,PPAGE_1,PPAGE_2,/*PPAGE_4,*/PPAGE_5,PPAGE_6,PPAGE_7,ROM1,PPAGE_0;
```

5.2 Board header files

Files XTWR-S08MM.h, XTWR-MCF51MM.h, XTWR-S08JE.h, and XTWR-MCF51JE.h provided in this application note include some definitions for Tower modules for each part number, supporting LEDs, push buttons, and declarations for easy integration of code in the application. The appropriate board header file is included in several of the .c files in projects for this application note, similar to the file m51em256demo.h included in the projects included in AN3938.

5.3 Modifications in sci.h and sci.c files

The file SCI.h has no modifications, but the file SCI.c has the following changes:

- Remove support for Port 3 (initialization, data send, data receive, and baud rate adjust functions)
- MM/JE devices configure the pin functions automatically, so they do not have pin function registers (PTxPF), and the code to write these registers is not necessary.

5.4 Modifications in infrared.h and infrared.c files

The changes in the file infrared.h are summarized in Table 2.

Table 2. File infrared.h modifications

EM characteristics	MM/JE characteristics	Change
Independent TX and RX pin location registers	Only one register to configure pin location	Definitions deleted: • TX_PIN_OPTION_DEFAULT • TX_PIN_OPTION_ALTERNATIVE • RX_PIN_OPTION_DEFAULT • RX_PIN_OPTION_ALTERNATIVE Definitions inserted: • PIN_OPTION_DEFAULT • PIN_OPTION_ALTERNATIVE
Pin options are selected in TX and RX configuration	Pin options are selected in SCI initialization	Function prototypes have different input parameters
TX can be modulated by: • TPM channel 0 • TPM channel 1 • MTIM2 • MTIM3	TX can be modulated by: • TPM1 channel 0 • TPM1 channel 1 • TPM2 channel 0 • TPM2 channel 1	Definitions deleted: MODULATION_BY_TPMCH0 MODULATION_BY_TPMCH1 MODULATION_BY_MTIM2 MODULATION_BY_MTIM3 Definitions inserted: • MODULATION_BY_TPM1CH0 • MODULATION_BY_TPM1CH1 • MODULATION_BY_TPM2CH0 • MODULATION_BY_TPM2CH1

The changes in file infrared.c are summarized in Table 3.

Table 3. File infrared.c modifications

EM characteristics	MM/JE characteristics	Change
Pin options are selected in TX and RX configuration	Pin options are selected in SCI initialization	Correspondent functions are modified. Comments for function explanation are also modified, and provide information about input parameters for each function.
SCI1, SCI2, and SCI3	SCI1 and SCI2	Functions remove any configuration, initialization, baud rate adjust, drive strength, and modulation enabling for port 3.
Two analog comparators (ACMP1 and ACMP2)	One analog comparator (ACMP)	Analog comparator register names are modified.
SCI1 and SCI2 support modulation	SCI1 supports modulation	Modulation enabling is modified.
SCI1 and SCI2 support reception through analog comparator	SCI1 supports reception through analog comparator	Reception through analog comparator enabling is modified.

5.5 Modifications in ascii.h and ascii.c files

The file ascii.h has no modifications.

File ascii.c has the following changes:

- Remove enabling and disabling of port 3.
- Code for port 3 interrupt is deleted.

5.6 Modifications in com_protocol.h and com_protocol.c files

The file com_protocol.h has the following modifications:

- The RAM address definitions are changed because MCF51EM256 has 16 KB, while MM/JE Coldfire V1 devices have 32 KB, and S08 devices have 12 KB.
- The flash address definitions are changed because MCF51EM256, MCF51MM256 and MCF51JE256 devices have two arrays of 128 KB with independent configuration registers, while MM/JE S08 devices have two arrays of 64 KB and only one set of registers.
- The definitions for command responses are changed from OK and ERR to COMMAND_OK and COMMAND_ERR.

For the Coldfire V1 projects, the file com_protocol.c is not modified, but for S08 projects the following modifications are included:

- The declaration of Table1 (stored in flash) is different:

```
const UINT8 Table1[TABLE1_SIZE] @0x48000
```

- In the function write_verify, the variable UINT8 u8ppage_backup is created to back up the value of the PPAGE register. This is because flash array 1 contains pages 0, 1, 2, and 3, and flash array 2 contains pages 4, 5, 6, and 7. This is necessary because the address of Table1 as shown above is 0x48000 (the four indicates that it will be stored at the beginning of page 4).

5.7 Modifications in main.c file

In this file the following code line must be inserted at the beginning, to bypass the bootloader:

For S08 devices these lines are inserted:

```
const unsigned char boot1@0x040A = 0; //zero out checksum to bypass boot loader
const unsigned char boot2@0xFFBA = 0; //bypass checksum
```

For Coldfire V1 devices this line is inserted:

```
const unsigned char boot@0x040A = 0x0000; //zero out checksum to bypass boot loader
```

Also, the welcome message is replaced. When the board is turned on, the message below is sent by the SCI port and can be read using a serial port terminal on a PC.

```
**********************************************************************
Freescale Semiconductor. RTAC Americas
Communication Protocol on MM/JE Flexis families
```

Finally, the last line indicates which device is being used.

6 Conclusions

This application note discussed the similar hardware characteristics between Freescale device MCF51EM256 and devices in the Freescale MM/JE Flexis families, allowing easy portability of the metering protocol by simply applying a few modifications.

The principal difference between the devices is the number of modules of peripherals, such as SCIs and ACMPs, that require the majority of modifications in the code. The EM family is ideal for electricity meter applications, while the MM/JE families are better for instrumentation metering.

7 References

Software for this application note is developed on Code Warrior v.6.3, and can be found on the Freescale website, as zip file AN4116SW, and, for application note AN3938, zip file AN3938SW.

To better understand the paging memory of S08 devices, it is highly recommended to read the application note AN3730, “Understanding Memory Paging in 9S08 Devices,” which can be found on the Freescale website.

If you want to learn more about Freescale's MC9S08MM, MC9S08JE, MCF51MM, or MCF51JE microcontrollers, please visit the Freescale website, and check the product summary pages, fact sheets, data sheets, etc.

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN4116
Rev. 0
07/2010