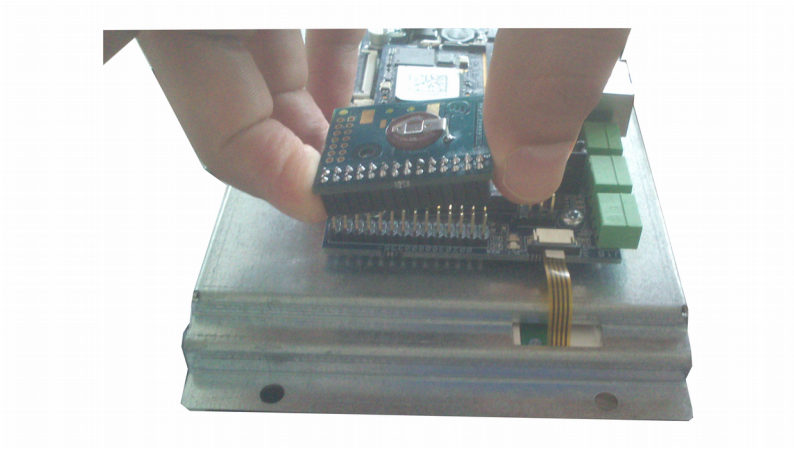
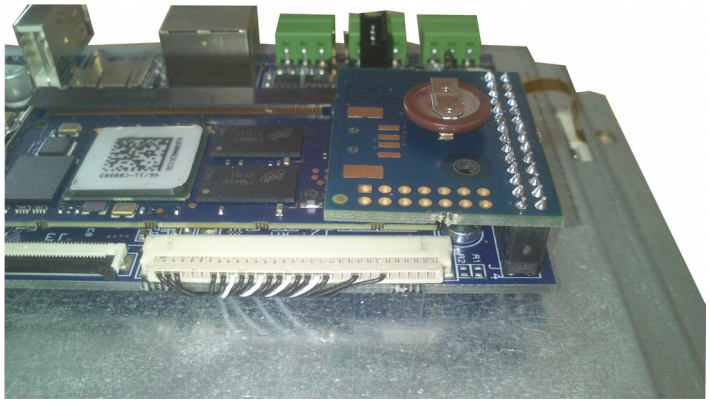
EVB RTC Manual 1.0.4





REV 1.0.4

DATE	REVISION	CHANGE DESCRIPTION
03/07/2013	1.0.0	Release
23/12/2013	1.0.1	Added sw instructions
25/02/2014	1.0.2	Added patch info for using i.Core53
24/04/2015	1.0.3	Added RTC settings for YOCTO
03/05/17	1.0.4	Added “Product Compliance” chapter

Summary

1. Hw Interface

The RTC board is interfaced by 2x14 pin connector. In the following table are reported the map of connector with the signals function description and the reference voltage.

Pin Number	Pin Name	Function Description	Voltage Reference
1	+3V3	Power Signal	-
2	+3V3	Power Signal	-
3	NC	-	-
4	+5V	Power Signal	-
5	I2C_SDA	I2C Bus	+3,3V
6	I2C_SCL	I2C Bus	+3,3V
7	+5V	Power Signal	-
8	USB_OTG_DN	USB on the go interface	USB OTG Comp
9	USB_OTG_DP	USB on the go interface	USB OTG Comp
10	GND	Power Signal	-
11	GND	Power Signal	-
12	SD2_D1	eSDHC 2 Data signal	+3,3V
13	SD2_D2	eSDHC 2 Data signal	+3,3V
14	SD2_CLK	eSDHC 2 CLK signal	+3,3V
15	SD2_D0	eSDHC 2 Data signal	+3,3V
16	SD2_CMD	eSDHC 2 CMD signal	+3,3V
17	SD2_D3	eSDHC 2 Data signal	+3,3V
18	NC	-	-
19	NC	-	-
20	NC	-	-
21	NC	-	-
22	I2S_MCLK	I2S MCLK signal	+3,3V
23	NC	-	-
24	NC	-	-
25	I2S_DOUT	I2S Data signal	+3,3V
26	I2S_LRCLK	I2S RCLK signal	+3,3V
27	I2S_SCLK	I2S SCLK signal	+3,3V
28	GND	Power Signal	-

Tab 1

WARNING: *It's mandatory to mount RTC board by mating the 14x2 pin connector, aligning the pin 1 with the pin 1 of the Open-frame*

Following the connections through I2C interface with RTC device PCF8563T

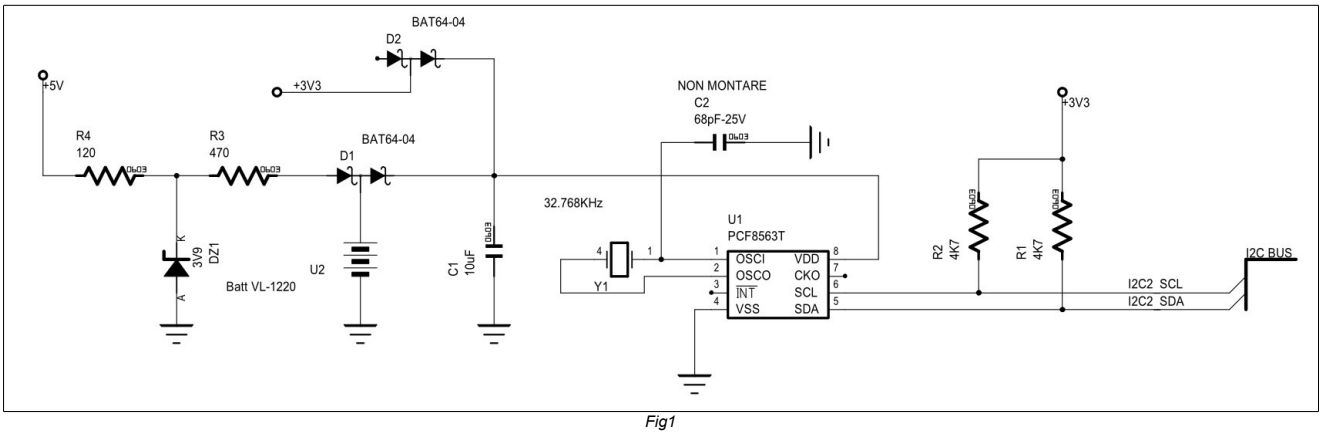


Fig1

In the standard configuration only the I2C bus is used on the board to use all the possible functionality please refer to the “Optional Features” chapter

2. SW Instructions

To read the clock through the RCT board you must setup and compile the kernel with the appropriate driver for the PCF8563T device

1. You have to modify the file description of the target
2. You have to add the driver for PCF8563
3. Use the following command:

date -s "YYYY-MM-DD hh:mm:ss"

hwclock --systohc

The instruction's strings to make this operation are described following.

2.1 Using GEAM6425

- Insert the driver inside the kernel. Then enter in the directory:

  /home/user/ev-sdk/workspace/evelin-bsp

  and edit the following command:

  sb2 -t imx25 make kernelconfig

- Enable in the kernel the drivers of the RTC as follow

```
Device Drivers  --->
  <*> Real Time Clock  --->
    <*>   Philips PCF8563/Epson RTC8564
```

- Setup the i2c references in the platform file of the kernel:

  gedit linux/linux-2.6.31-geaM6425/arch/arm/mach-mx25/mx25_gea_m6425.c

  adding the following code:

```
addr = 0x51, 0x51               /* I2C address */

static struct i2c_board_info mxc_i2c_board_info[] __initdata = {
            {
            .type = "pcf8563",
            .addr = 0x51,
            },
};
```

- Compile the modified kernel and make the kernel image running the script inside the directory:

  cd /home/user/ev-sdk/workspace/evelin-bsp

  ./make_kernel

- Copy the generated image into the server's tftp directory (/tftp_boot) as follows:

  cd /home/user/ev-sdk/workspace/evelin-bsp/linux/linux-2.6.31-geaM6425/arch/arm/boot

  cp uImage /tftp_boot

2.2 Using GEAM6428

- Insert the driver inside the kernel. Then enter in the directory:

  /home/user/Desktop/GEAM6428/ltib

  run the following command:

  ./ltib -c

  to include the RTC driver into the kernel:

```
Device Drivers  --->
 <*> Real Time Clock  --->
<*>   Philips PCF8563/Epson RTC8564
```

- Setup the i2c references in the platform file of the kernel:

/home/user/Desktop/Linux kernel/arch/arm/mach-mx28/mx28evk.c

adding the following code in the file:

```
static struct i2c_board_info __initdata mxs_i2c_device[] = {
        { I2C_BOARD_INFO("sgtl5000-i2c", 0xa), .flags = I2C_M_TEN },
        { I2C_BOARD_INFO("pcf8563", 0x51), .flags = I2C_M_TEN },
};
```

- Compile the modified kernel and make the kernel image running the script inside the directory:

/data/GEAM6428/ltib

./make_linux

- Copy the generated image into the server's tftp directory (/tftp_boot) as follows:

cd /home/user/Desktop/Linux kernel/arch/arm/boot

cp uImage /tftp_boot

2.3 Using i.Core53 and Open-Frame carrier revision -

- Insert the driver inside the kernel. Then enter in the directory:

/home/user/ev-sdk/workspace/evelin-bsp

then run the following command:

sb2 -t imx53 make menuconfig

to include the RTC driver into the kernel:

```
Device Drivers  --->
 <*> Real Time Clock  --->
  <*>   Philips PCF8563/Epson RTC8564
```

- Setup the i2c references in the platform file of the kernel:

gedit /home/user/ev-sdk/workspace/evelin-bsp/linux/linux-2.6.35-icoreM53/arch/arm/mach-mx5/mx53_loco.c

adding the following code in the file:

addr = 0x51, 0x51 indica l'address I2C.

```
static struct i2c_board_info mxc_i2c0_board_info[] __initdata = {
        {
        .type = "pcf8563",
        .addr = 0x51,
        },
};
```

- Compile the modified kernel and make the kernel image running the script inside the directory:

```
cd /home/user/ev-sdk/workspace/evelin-bsp
rm images/.evelin_done_linux
sb2 -t imx53 make linux
```

- Copy the generated image into the server's tftp directory (/tftp_boot) as follows:

```
cd /home/user/ev-sdk/workspace/evelin-bsp/linux/linux-2.6.35-icoreM53/arch/arm/boot
cp uImage /tftp_boot
```

2.4 Using i.Core53 and Open-Frame carrier revision A

- Insert the driver inside the kernel. Then enter in the directory:

  /home/user/ev-sdk/workspace/evelin-bsp

  then run the following command:

  sb2 -t imx53 make menuconfig

  to include the RTC driver into the kernel:

  ```
  Device Drivers  --->
   <*> Real Time Clock  --->
    <*>   Philips PCF8563/Epson RTC8564
  ```

- Setup the i2c references in the platform file of the kernel:

  gedit /home/user/ev-sdk/workspace/evelin-bsp/linux/linux-2.6.35-icoreM53/arch/arm/mach-mx5/mx53_loco.c

  adding the following code in the file:

  addr = 0x51, 0x51 indica l'address I2C.

  ```
  static struct i2c_board_info mxc_i2c0_board_info[] __initdata = {
                {
                .type = "pcf8563",
                .addr = 0x51,
                },
  };
  ```

- Compile the modified kernel and make the kernel image running the script inside the directory:

  ```
  cd /home/user/ev-sdk/workspace/evelin-bsp
  rm images/.evelin_done_linux
  sb2 -t imx53 make linux
  ```

- Copy the generated image into the server's tftp directory (/tftp_boot) as follows:

  ```
  cd /home/user/ev-sdk/workspace/evelin-bsp/linux/linux-2.6.35-icoreM53/arch/arm/boot
  cp uImage /tftp_boot
  ```

2.5 Using LTIB with i.Core6 and Open-Frame carrier revision

- Insert the driver inside the kernel. Then enter in the directory:

  /data/iCoreM6/linux-3.0.35.eng

  then run the following command:

  ./config_icore.sh

  to include the RTC driver into the kernel:

  ```
  Device Driver-->
   [*]Real Time Clock-->
    <*>Philips PCF8563/Epson RTC8564
  ```

- Setup the i2c references in the platform file of the kernel:

```
cd /data/iCoreM6/linux-3.0.35.eng/arch/arm/mach-mx6

gedit board-mx6q_icore.c
```

adding the following code into the file:

```
static struct i2c_board_info mxc_i2c2_board_info[] __initdata = {
        #if (defined CONFIG_MACH_MX6Q_ICORE_OF_CAP_EDT_7 || defined
        CONFIG_MACH_MX6Q_ICORE_STARTERKIT_CAP_EDT)
        {
                I2C_BOARD_INFO("edt-ft5x06", 0x38),
                #ifdef CONFIG_MACH_MX6Q_ICORE_OF_CAP_EDT_7
                .irq  = gpio_to_irq(OFC_FT5X06_TS_IRQ),
                #endif

                #ifdef CONFIG_MACH_MX6Q_ICORE_STARTERKIT_CAP_EDT
                .irq  = gpio_to_irq(STARTERKIT_CAPEDT_IRQ),
                #endif

                .platform_data = (void *) & mx6_icore_ft5x06_data,
        },
        #endif
        #ifdef CONFIG_MACH_MX6Q_ICORE_OF_CAP_AMPIRE
        {
                I2C_BOARD_INFO("ili210x", 0x41),
                .platform_data = (void *) & mx6_icore_ili210x_data,
                .irq  = gpio_to_irq(OFC_FT5X06_TS_IRQ),
        },
        #endif
        {
                I2C_BOARD_INFO("sgtl5000", 0x0a),
        },
        #ifdef     CONFIG_MACH_MX6Q_ICORE_OF_CAP
        {
                I2C_BOARD_INFO("pcf8563", 0x51),
        },
        #else
        {
                I2C_BOARD_INFO("adv7180", 0x21),
                .platform_data = (void *)&adv7180_data,
        },
        #endif
        {                                                       //
                I2C_BOARD_INFO("pcf8563", 0x51),                // CODE TO ADD
        },                                                      //
};
```

- Compile the modified kernel and make the image, then running the script inside the directory:

```
/data/iCoreM6/linux-3.0.35.eng

./build_linux.sh
```

- Copy the generated image into the server's tftp directory (/tftp_boot) as follows:

```
cd /data/iCoreM6/linux-3.0.35.eng/arch/arm/boot
cp uImage /tftp_boot
```

2.6 Using YOCTO with i.Core6 and Open-Frame carrier revision

Add the probe of the device I2C inside the dts. Open the dev shell of the kernel (see chapter 13 and 14 of YOCTO sw manual) edit the dts of the built-in R.Touch (see chapter 23 of Engicam's Open-Frame board setup) and add, into the I2C3 branch, the probe:

```
&i2c3 {
    pcf8563: rtc@51 {
      compatible = "pcf8563";
      reg = <0x51>;
   };
}
```

Note:
Make sure to be aligned to the last release of meta-engicam (for instruction see Chapter 3.2 of the YOCTO sw manual)

3. Optional Features

The board is also equipped with other 2 connectors the first used to the USB OTG interface

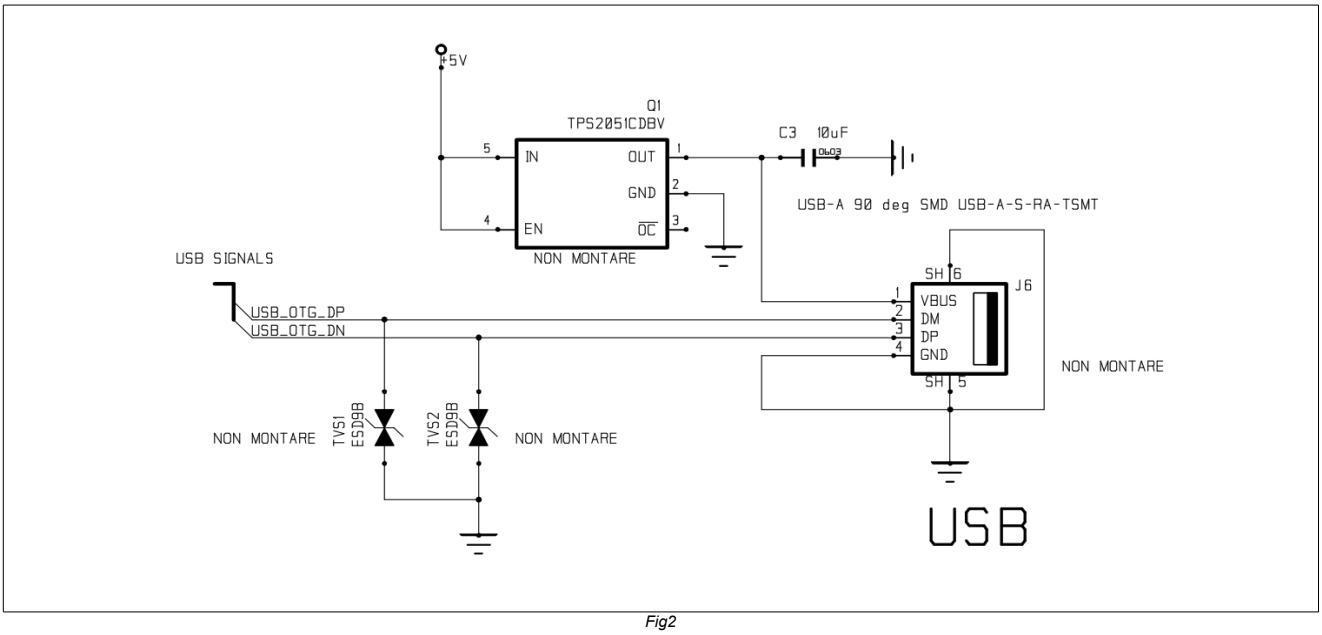


Fig2

The second used to implement other peripheral interfaces through the ESDH and I2S buses

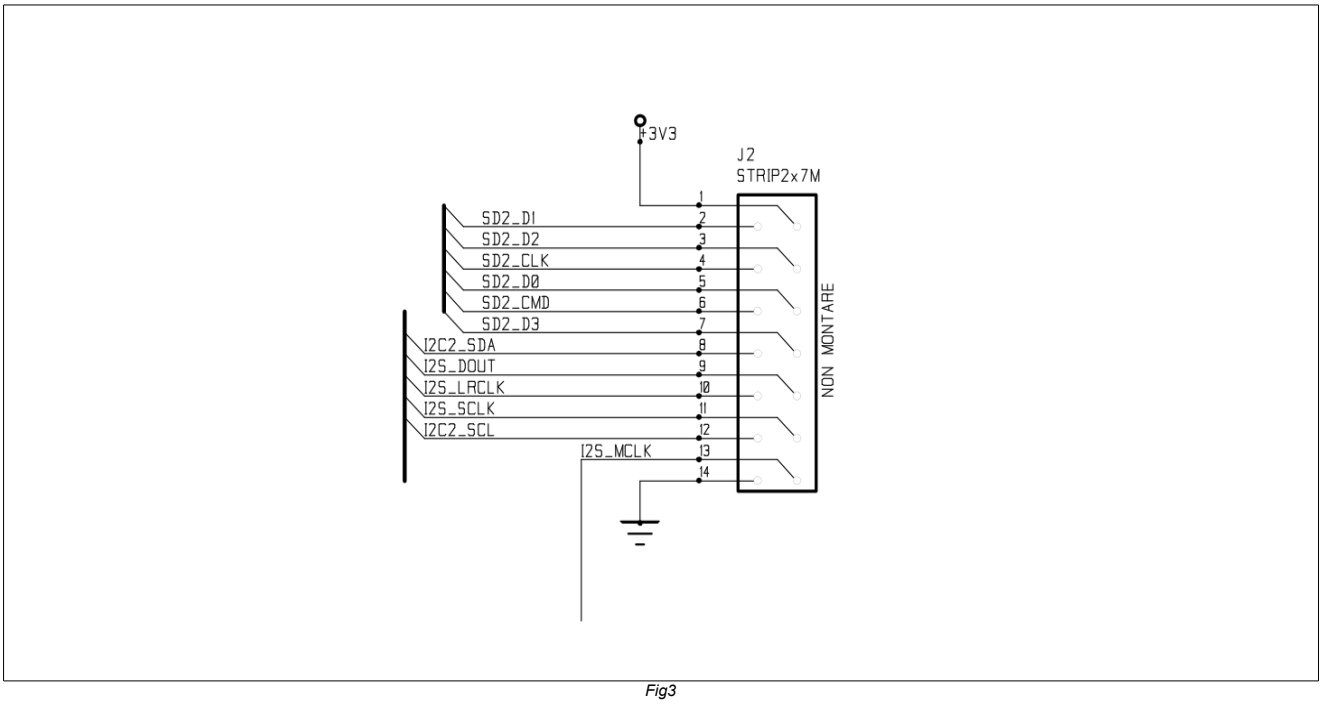


Fig3

In the following table is reported the map of connector J2 and the corresponding signals with the Module's connector

PIN on J2 Strip Connector	Primary signal Function	Pin on Module's connector
1	+3,3V	134-135
2	SD2_D1	171
3	SD2_D2	170
4	SD2_CLK	169
5	SD2_D0	168
6	SD2_CMD	167
7	SD2_D3	166
8	I2C_SDA	24
9	I2S_DOUT	122
10	I2S_LRCLK	115
11	I2S_SCLK	124
12	I2C_SCL	23
13	I2S_MCLK	34
14	GND	-

Tab 2

Comparing the above map with the PIN OUT of the used module and using the Hardware manuals it's possible to find the implementable mux functions.

Note: In the standard assembly the connectors are not present but they can be mounted when required

4. How to apply 001.03_iCore53_openframe_rtc.patch (only for the Open-Frame carrier revision -)

Using the following command to apply the patch:

1. Download the patch from FTP server.
2. Move the patch into the kernel directory inside the virtual machine:

   /home/user/ev-sdk/workspace/evelin-bsp/linux/linux-2.6.35-icoreM53
3. To apply the patch, enter in the kernel directory using the ubuntu shell and run the command:

   patch -p1 < 001.03_iCore53_openframe_rtc.patch

```
user@ubuntu1004desktop:~/ev-sdk/workspace/evelin-bsp/linux/linux-2.6.35-icoreM53$ patch -p1 < 001.03_iCore53_openframe_rtc.patch
(Stripping trailing CRs from patch.)
patching file arch/arm/mach-mx5/mx53_loco.c
Hunk #1 succeeded at 650 (offset -5 lines).
Hunk #2 succeeded at 667 (offset -5 lines).
Hunk #3 succeeded at 691 (offset -5 lines).
(Stripping trailing CRs from patch.)
patching file drivers/rtc/rtc-pcf8563.c
```

4. Now adding the RTC driver from the kernel menu running the following command:

   sb2 -t imx53 make menuconfig

   Device Drivers --->
   <*> Real Time Clock --->
   <*> Philips PCF8563/Epson RTC8564

5. After that, erase the data of the previous compiling:

   rm images/.evelin_done_linux
6. Remake the kernel image with the new changes:

   sb2 -t imx53 make linux
7. Export the new image on the target (using the tftp or a SD card) and then reboot
8. Set the date with command:

   date -s “YYYY-MM-DD HH:mm:ss”
9. Then, syncronise the RTC using the command:

   hwclock –systohc
10. Reboot the system.

Note:

For further information please refer to the Open-Frame User manual.

4.1 Product Compliance

In order to respect own internal policy regarding the environmental regulations and safety laws, Engicam in this chapter confirms the compliant, when applicable, of its own products to the normatives ROHS and REACH and to the recognized hazards.

Warning!
The current product board mounts a VL-1220/HFN Rechargeable Battery, that has the following elements included into the SVHC list:

- ***1,2-dimethoxyethane, ethylene glycol dimethyl ether (EGDME)***

5. On-line Support

We offer an on-line support to allow the customer to stay updated on the development of software release and on the enhancement of the documentation.

Following is shown the references for ENGICAM on-line support.

5.1 Support

ENGICAM Product Experts are available to answer questions via email:

support@engicam.com

5.2 Disclaimer