Pawel Gburzynski

Interfacing CC1000PP Chipcon RF Board to PicOS

Application Note

June 29, 2004

1. Introduction

This application note describes the interface between the CC1000PP RF module and the eCOG evaluation board and explains how to access the RF module from PicOS. To take full advantage of the most useful features of CC1000, including interrupt-driven operation at the maximum bit rate, the way CC1000PP has been configured into PicOS is slightly different from the other RF boards handled by the system (described in the PicOS manual). The biggest difference is the lack of an explicit `RADIO` device: CC1000PP has no device driver, but is visible via a physical interface module (a "phys") that makes the device accessible through TCV (see the TCV manual). Thus, the application programmer has no option of raw access to CC1000 (available for other boards) but must use TCV to talk to the transceiver. Notably, the way the module is visible through TCV makes the chip not much different from other radio devices configurable with PicOS. Thus, from the viewpoint of an application accessing CC1000PP via TCV, the difference is solely in a few configuration options specific to the chip and described in this note. The TCV interface simply bypasses the `RADIO` device, which never shows up explicitly in the implementation.

2. Hardware connections

CC1000PP is connected to the eCOG evaluation board via 8 wires listed in Table 1. The five GPIO ports (L3-L7) are available on connector J12, together with Vcc and ground (GND), in a range of adjoining pins, 6-12 (see Figure 1), which makes it easy to connect seven of the eight wires via a compact female connector. The remaining single wire connects RSSI to J23, which is the first ADC input (Vin1). The jumper on J23 should be removed, and the jumper on J24 (the second ADC input – Vin2) should be set to 1-2, i.e., to feed Vin2 with Vcc through the potentiometer divider. The digitized received power level is computed as Vin1-Vin2; thus, the potentiometer of the second ADC input (VR2) is used to calibrate the indications of the received power level presented to the application (as described below). The connection of RSSI to the eCOG board is shown in Figure 2.

Table 1. The list of connections between CC1000PP and eCOG board.

Pin on eCOG board	eCOG Function	CC1000 Pin
J12-06	GPIO L3	PALE
J12-07	GPIO L4	DIO
J12-08	GPIO L5	DCLK
J12-09	GPIO L6	PCLK
J12-10	GPIO L7	PDATA
J12-11	Vcc (3V)	Vcc
J12-12	GND	GND
J23	ADC Input Vin1	RSSI

Figure 3 illustrates how the wires should be connected to the CC1000PP. The GND connection has been skipped as trivial.

J12

PortD 2	1	2	PortD 3
PortL 0	3	4	PortL 1
PortL 2	5	6	PortL 3
PortL 4	7	8	PortL 5
PortL 6	9	10	PortL 7
Vcc3.3	11	12	GND
IntAct U Fin	13	14	IntAct U Din
IntAct U Cout	15	16	GND
VCC3.3	17	18	IntAct U Cin
IntAct U Dout	19	20	IntAct U Fout
PortJ 0	21	22	PortJ 1
PortJ 2	23	24	PortJ 3
PortJ 4	25	26	PortJ 5
PortJ 6	27	28	PortJ 7
PortI 7	29	30	PortI 6
PortI 5	31	32	PortI 4

Figure 1. Location of the relevant pins on J12.

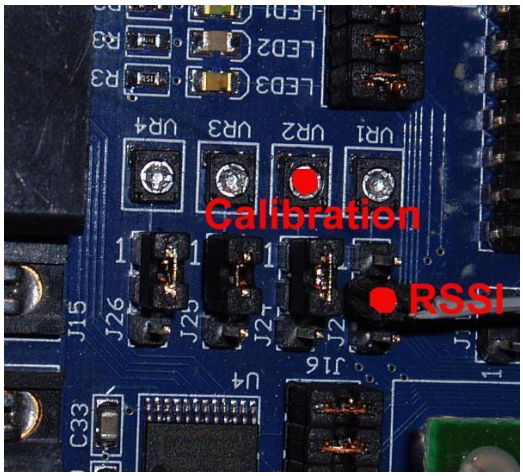

Figure 2. Connecting RSSI to the ADC input.

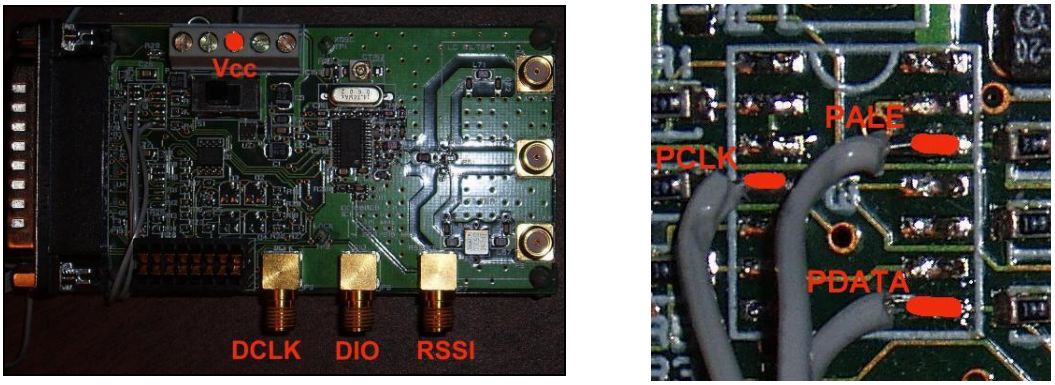

Figure 3. Connecting the wires to CC1000PP.

3. Configuring PicOS with CC1000PP

In PicOS, the CHIPCON module is selected by setting `CHIPCON` in `options.sys` in the application's directory to 1. The module can coexist with all optional devices configurable under PicOS, except for `RADIO`, which **must not** be configured, i.e., `RADIO_DRIVER` **must not** be set in `options.sys`. Also, as the only way to see the CHIPCON board from the application is via TCV, TCV must be included in any sensible setup involving the CHIPCON module. The RFPing application that comes with PicOS illustrates how to configure CC1000PP into the system. Files `Makefile_chipcon` and `option.sys_chipcon` must be substituted for `Makefile` and `options.sys` before executing `make`.

The `LEDS` device, although formally available as an independent device, is taken over by the CHIPCON module: all four LEDs are used to display the transceiver status – in the following way:

RED (0) when lit, indicates that the logical receiver is on, meaning that the phys module is willing to receive data

GREEN (1) when lit, indicates that the logical transmitter is on, meaning that the phys module is willing to extract packets queued for transmission and transmit them

BLUE (2) is lit after the reception of a valid preamble, and remains lit for as long as a packet is being received;

YELLOW (3) remains lit while a packet is being transmitted

One static configuration parameter of the CHIPCON module is the radio frequency range for which the board has been pre-configured. The frequency range is determined

by the constant `RADIO_FREQ` in `chipcon.h` (directory `PicOS`), whose value can be either `868` or `433` and directly corresponds to the frequency band in MHz.

4. Module initialization

An application using the CHIPCON board connects to the required phys module by including `phys_chipcon.h`. The transceiver board is initialized with the following function:

```
---> void phys_chipcon (int phy, int mode, int mpl, int baud)
```

where `phy` is the numerical module identifier (see the TCV manual). The remaining arguments have the following meaning:

`mode` — if nonzero, it declares the station Id and selects the framed reception mode, whereby the first word (two bytes) of a received packet is expected to include a station Id. This Id must match the station Id of the receiving station or be zero; otherwise, the packet will be ignored. Note that for a transmitted packet, the station Id **will not** be automatically inserted, so it is up to the application/plugin to take care of this end.

`mpl` — indicates the maximum full length of a legitimate packet in bytes, excluding the preamble and including the possible 4-byte checksum to be automatically inserted by the transmitter (see below). The specified value can be zero, in which case the default length of 32 bytes is assumed. If nonzero, the specified length must be at least 8 and it must be divisible by 4.

`baud` — is the requested baud rate divided by 100, or zero, in which case the default rate of 38,400 bps will be assumed. The specified number is multiplied by 100 and then rounded up to the nearest settable rate, which can be 600, 1,200, 2,400, 4,800, 9,600, 19,200, 38,400, or 76,800. A value larger than 768 is OK and selects 76,800 bps.

As part of the chip initialization procedure, the module performs the so-called calibration. If the initialization formally succeeds, PicOS will produce a message (on `UART_A`) looking like this:

`CHIPCON 1000 calibrated at` *`bbbbb`* `bps in` *`nn`* `attempts`

where *`bbbbb`* is the effective baud rate and *`nn`* is the number of calibration attempts (a single attempt may occasionally fail). The limit on the number of unsuccessful calibration attempts is 31.

5. Control parameters

As explained in the TCV manual, a phys module configured with TCV provides a control function whose role is to dynamically set or modify the operational parameters of the

device/module. The following operations available via `tcv_control` for the CHIPCON phys module: `PHYSOPT_STATUS`, `PHYSOPT_TXON`, `PHYSOPT_RXON`, `PHYSOPT_TXOFF`, `PHYSOPT_TXHOLD`, `PHYSOPT_RXOFF` are standard and act as described at **`phys_ether`** in Section 5 of the TCV manual. The CHIPCON-specific parameters settable this way are listed below.

`PHYSOPT_CAV`

The `value` argument points to a word-sized unsigned number specifying the CAV (collision avoidance vector), i.e., the minimum delay until the next transmission attempt (in milliseconds). This delay will not be obeyed if the first outgoing packet is urgent, unless there are other reasons for deferring the transmission (e.g., a reception in progress).

`PHYSOPT_SENSE`

The operation returns the perceived status of the ether. The returned value is 1, if the chip is currently receiving or transmitting a packet, and 0 otherwise.

`PHYSOPT_SETPOWER`

The operation sets the transmitter power. The argument should point to a word-sized unsigned value between 0 and 255 inclusively. A number larger than 255 is treated as 255 (i.e., it indicates the maximum power). Value zero means default power, which corresponds to value 1 (i.e., the lowest selectable power). This is also the transmitter power level at which the chip starts after initialization.

`PHYSOPT_GETPOWER`

The operation returns the reception power of the last received packet normalized to a value between 0 and 255 inclusively. It is only effective if RSSI monitoring has been turned on (see below). Note that, as packets are queued by TCV before being presented to the application, it may be difficult to match the value returned by `PHYSOPT_GETPOWER` to the packet for which it was obtained. Received power indications are also available on the per-packet basis (see below).

`PHYSOPT_SETPARAM`

This operation sets one of six dynamic parameters of the transceiver. The argument points to a two-word structure whose first word specifies the parameter index (0-5) and the second word contains the value to be assigned to the respective parameter. The parameters are as follows:

`delmnbkf(0)` This is the minimum amount of backoff (in milliseconds) used by the transmitter after an activity perceived by the receiver. The minimum is 1, the maximum is 32768, and the default is 32.

`delbsbkf(1)` This is the number of ones counting from the right in a bit mask determining the range of the extra delay added to `delmnbkf` to

produce the actual (randomized) backoff value. The actual backoff is generated as `delmnbkf + (random & mask)`, where `random` is a pseudo random word-sized integer number. The minimum is zero, the maximum is 15, the default is 8 (which corresponds to the mask of 0xff).

`delxmspc(2)` This is the minimum amount of packet space (in milliseconds) separating two consecutively transmitted packets. The minimum is 0, the maximum is 256, and the default is 8.

`preamble(3)` The length of the transmitted packet preamble in bits. The minimum is 32, the maximum is 256, and the default is 48. A transmitted preamble consists of an alternating sequence of zeros and ones ending with two consecutive ones.

`chsmmode(4)` The checksum mode: 0 – no checksum, 1 – checksum inserted at transmission and calculated at reception (the default). Packets with illegal checksum are not formally received. The last four bytes of every transmitted packet are overwritten with the checksum calculated on all the preceding bytes (whose number must be divisible by 4). 2 – as 1 with the last byte of checksum used to store the received power level (for a received packet). With 1, the received checksum (in addition to being automatically verified) is passed verbatim to the application along with the received packet. With 2, the last byte of the checksum is changed to the received power for the packet. This will only work if RSSI monitoring is turned on (see below).

`rssimntr(5)` If nonzero, switches on RSSI monitoring, i.e., received power level will be collected and made available through `PHYSOPT_GETPOWER` or/and via the last byte of received checksum. If zero, no RSSI will be collected, `PHYSOPT_GETPOWER` will always return zero, and the last byte of received checksum will not be modified.

If `chsmmode` is not zero, a packet submitted for transmission must include room for the 4-byte checksum at its length. Regardless of the submitted contents of those bytes, they will be overwritten by the checksum calculated by the transmitter before the packet is sent. Similarly, a received packet will be passed to the application (returned by TCV) with the checksum bytes, and it will be up to the application to ignore them. If `chsmode` is 2 and `rssimntr` is nonzero, then the last of the checksum bytes, i.e., the last byte of the complete received packet, will contain the received power level normalized to a value between 0 and 255.

6. Calibrating the received power level

The received power level returned to the application is determined as the difference between the voltage on J23 (fed by the RSSI signal from CC1000PP) and J24, fed from Vcc on the eCOG board via a potentiometer divider (VR2 – see Figure 2). The digitized

difference is linearly transformed into a single-byte value from 0 to 255. To calibrate the power level indications, VR2 should be adjusted such that the voltage on J24 is equal to the maximum voltage on J23, which, according to the documentation of CC1000PP, is 1.2V.