Version 1.0

eRIC_ADC:

eRIC has ADC10 and ADC12. But only ADC12 is used in eRIC module. ADC12 supports fast 12-bit analog to digital conversions. There are 6 ADC pins available on eRIC module. They are Pin 1,2,3,4,5 and Pin22. ADC reference voltage can be set only on Pin22 which is explained further below.

Refer latest eRIC_eROS_Developers_Manual_x.x for complete list of ADC definitions. Refer SLAU259E document by Texas Instruments to use core ADC(including ADC10) registers.

Choosing which Pin to be ADC, setting reference voltage on Pin22 if necessary and read ADC using lprs function is all it needs to perform ADC.

ADC:

Any one of the six pins(1,2,3,4,5,22) should be set first, to perform ADC which is explained below:

eRIC_SetAdcPin(eRICPinNumber); One of the six eric Pins is set as ADC using this.

eRICPinNumber can only be 1,2,3,4,5 and 22.
For example to set Pin2 as ADC, use eRIC_SetAdcPin(2);

eRIC_SetAdcRefVoltage(eRIC_ReferenceVoltage,RefOut_OnorOff_Pin22); On chip reference voltage can be set using this function. This reference voltage can only be set to output on Pin22.
eRIC_ReferenceVoltage can be:

a) eRICADCRef_1_5v: The reference voltage of 1.5v is set
b) eRICADCRef_2_0v: The reference voltage of 2.0 is set
c) eRICADCRef_2_5v: The reference voltage of 2.5v is set

RefOut_OnorOff_pin22 : If it is 1, the internal reference voltage is output on Pin22. If it is 0, internal reference voltage is not output on pin22.
For example to set reference of 2v on Pin22, use eRIC_SetAdcRefVoltage(eRICADCRef_2_0v,1);
Pin22 can also be used a normal ADC if reference voltage is not needed.

In version V1.5.5 header files, only internal voltage reference of maximum 2.5v is defined. However, Pin22 has the capability to be used as external voltage. So to set external voltage reference on Pin22, the following line should be added after setting any ADC pin:
Pin22_FunctionA2D();
ADC12MCTL0 |= ADC12SREF_2; //This will set Pin22 as external reference voltage.
Maximum external voltage to be applied .is 3.6v

ReadAdc(); This will read ADC value on whichever pin is set as ADC. Since this is 12bit ADC, the maximum value will be 4095 in decimal or 0x0FFF. If Pin3 is set as ADC , then ReadAdc() will read digital value on Pin3 for applied voltage on it. Say if internal voltage reference is selected as 2v then any voltage greater than 2v ,applied on Pin3 will read 4095 value.

To convert ADC digital value into analog value, the following formula can be used:

Analog value = ((Digital value read by ReadAdc()) x Reference voltage)/4095

So for example, if external voltage reference of 3 v is used on Pin22. Then if value read by ReadAdc() is 2000 on Pin3. To get real analog value, we need to substitute in above formula:

Analog value = (2000 X 3)/4095 = 1.465v . So 1.465 voltage is applied on Pin3.

Code Example:

```
1) #include<cc430f5137.h>
2) #include "eRIC.h"
3) int main(void)
4) {
5)        eRIC_WDT_Stop();                    //Stop watch dog timer, just
   incase
6)        eRIC_GlobalInterruptDisable();      //Global interrupts disabled
7)        eRIC_SetCpuFrequency(5000000);     //Sets CPU clock to 5mHz
8)        eRIC_SetAdcPin(1);                 //Set Pin1 as ADC
9)       Pin22_FunctionA2D();
10)       ADC12MCTL0 |= ADC12SREF_2; // //Set Pin22 for external voltage
   source.
11)      eRIC_UARTAInitialise(19200);   //Initialise Uart with 19200 baud to
   send captured value
12)       Pin3_FunctionUartATxOUT();
13)       eROS_Initialise(434000000);
14)       eRIC_GlobalInterruptEnable();
15)       volatile int ADCvalue;
16)       volatile int TempADCvalue = 0;
17)       volatile unsigned int diff = 0;
18)       while(1)
19)       {
20)           ADCvalue =  ReadAdc(); //Read adc value on PIn1, it will be 0-
   4095 as it is 12 bit adc. 0v will give 0 and 2.5 v will give 4095 value
21)          diff = TempADCvalue-ADCvalue;
22)          if(diff>100)
23)          {
24)              TempADCvalue = ADCvalue;
25)              eRIC_UartASendByte(ADCvalue>>8);        //send high byte to
   Uart
26)              while(eRIC_UartATxBufferIsBusy());
27)              eRIC_UartASendByte(ADCvalue);        //send low byte to Uart
28)              while(eRIC_UartATxBufferIsBusy());
29)             eRIC_RadioTx_BuffCount = 0;
30)              eRIC_RadioTx_Buffer[eRIC_RadioTx_BuffCount++] = ADCvalue>>8;
   //send high byte over air
31)             eRIC_RadioTx_Buffer[eRIC_RadioTx_BuffCount++] = ADCvalue;
   //send low byte over air
32)              eRIC_RfSenddata();//send data
33)           }
34)       }//end of while(1)
35) }//end of main()
36) void eRIC_RfDataReceivedInterrupt()   //V1.5.4 Add code here to deal with
   available received data..This is triggered when interrupt is enabled and a
   packet is received
    {
```

```
}
```

The above example code will read ADC value onPin1 and send the value to Uart at 19200 baud rate and also send the data over air at 434Mhz, 38400 baud rate.

Line1 and line2 includes cc430F5137 and eRIC.h which is must for any program code. Main starts at line3.

Watchdogtimer is stoped in line5. Global interrupts are disabled in line6, any interrupts even radio interrupts in eros will be disabled.

Cpuclock speed is set at 5Mhz in line 7.

Pin1 is set as ADC on line 8. Pin22 is also set as ADC in line 9 and 10 for external voltage source.

Uart is set on Pin3 at 19200 baud in line 11 and 12.

Radio is initialised at 434Mhz and global interrupts are enabled at line 13,14.

Local variable initialised at 15,16,17.

ADC value on Pin1 is read at line 20.

If difference between current adc and previous adc value is greater than 100 is checked at line 22.

ADC high byte is sent though Uart at line 25

ADC low byte is sent through uart at line 27

Similary ADC high byte and low byte is filled in radio tx buffer in line 30 and 31.

The data is sent over air in line 32.

eRIC_RfDataReceivedInterrupt() is copied from eRIC.c which is removed and pasted in main at line 36-39. This has no effect in this example as there is no receiver enabled. But this code is needed for compiler to compile without error or un-remove this same code in eRIC.c.