PDK82C series
User guide v0.02

Note: This user guide is to remind user who use PDK82C series IC in order to avoid frequent errors upon operation. For more technical details, please refer to the latest datasheet.

1. IO pin usage and setting

(1) IO pin as analog input

- Configure IO pin as input
- Set ADCDI register to configure corresponding IO as analog input
- Set PBPH register to configure corresponding IO pull-up resistor as disabled

(2) Set PA5 as PRST# input pin

- No internal pull-up resistor for PA5
- Configure PA5 as input
- Set CLKMD.0=1 to enable PA5 as PRST# input pin

(3) PA5 as input pin to connect with a push button or a switch by a long wire

- Needs to put a >100Ω resistor in between PA5 and the long wire
- Avoid using PA5 as input

(4) PA7 and PA6 as external crystal oscillator

- Configure PA7 and PA6 as input
- Disable PA7 and PA6 internal pull-up resistor
- EOSCR register bit [6:5] selects corresponding crystal oscillator frequency :
  - ✧ 01 : for lower frequency, ex : 32KHz
  - ✧ 10 : for middle frequency, ex : 455KHz，1MHz
  - ✧ 11 : for higher frequency, ex : 4MHz
- Program EOSCR.7=1 to enable crystal oscillator
- Ensure EOSC working well of EOSC before switching from IHRC or ILRC to EOSC, refer to 3.(2)

2. Interrupt

(1) Only FPPA0 can accept interrupt, which means only FPPA0 can accept ENGINT and DISGINT instructions.

To use the interrupt function, the steps should be:

Step1: Set INTEN register, enable the interrupt control bit

Step2: Clear INTRQ register

Step3: In the main program, use ENGINT to enable FPPA0 interrupt function

Step4: Wait for interrupt. When interrupt occurs, enter to Interrupt Service Routine

Step5: After the Interrupt Service Routine being executed, return to the main program

* Use DISGINT in the main program to disable all interrupts

* When interrupt service routine starts, use PUSHAF instruction to save ALU and FLAG register. POPAF instruction is to restore ALU and FLAG register before RETI as below:

```
void Interrupt (void)     // Once the interrupt occurs, jump to interrupt  service routine
{                         // enter DISGINT status automatically, no more interrupt is accepted
    PUSHAF;
    …
    POPAF;
}    // RETI will be added automatically. After RETI being executed, ENGINT status will be
restored
```

(2) FPPA1 will not be affected by interrupt at all

(3) INTEN and INTRQ have no initial values. Please set required value before enabling interrupt function

3. System clock switching

(1) System clock can be switched by CLKMD register. Please notice that, NEVER switch the system clock and turn off the original clock source at the same time. For example: When switching from clock A to clock B, please switch to clock B first; and after that turn off the clock A oscillator through CLKMD.

◆ Case 1 : Switch system clock from ILRC to IHRC/2

```
CLKMD   =   0x36;          // switch to IHRC, ILRC can not be disabled here
CLKMD.2 =   0;             // ILRC can be disabled at this time
```

◆ Case 2 : Switch system clock from ILRC to EOSC

```
CLKMD   =   0xA6;          // switch to EOSC, ILRC can not be disabled here
CLKMD.2 =   0;             // ILRC can be disabled at this time
```

◆ **ERROR.** Switch ILRC to IHRC and turn off ILRC simultaneously

```
CLKMD   =   0x50;          // MCU will hang
```

(2) Please ensure the EOSC oscillation has established before switching from ILRC or IHRC to EOSC. MCU will not check its status. Please wait for a while after enabling EOSC. System clock can be switched to EOSC afterwards. Otherwise, MCU will hang. The example for switching system clock from ILRC to 4MHz EOSC after boot up is as below:

```
.ADJUST_IC    DISABLE
CLKMD.1 = 0;                    // turn off WDT for executing delay instruction.
$   EOSCR     Enable, 4MHz;     // 4MHz EOSC start to oscillate.
delay 255                       // Delay for EOSC establishment
CLKMD   = 0xA4;                 // ILRC -> EOSC;
CLKMD.2 = 0;                    // turn off ILRC only if necessary
```

The delay duration should be adjusted in accordance with the characteristic of the crystal and PCB. To measure the oscillator signal by the oscilloscope, please select (x10) on the probe and measure through PA6(X2) pin to avoid the interference on the oscillator.

4. Power down mode、wake up and watchdog

(1) Please turn off watchdog before executing STOPSYS or STOPEXE instruction, otherwise IC will be reset due to watchdog timeout. It is the same as in ICE emulation.

(2) It is recommended to disable Watchdog and enable fast wakeup before entering STOPSYS mode. When the system is waken up from power down mode, please firstly disable fast wakeup function, and then enable Watchdog. It is to avoid system to be reset after being waken up.

5. TIMER time out

When select T16M counter BIT8 as 1 to generate interrupt, the first interrupt will occur when the counter reaches to 0x100 (BIT8 from 0 to 1) and the second interrupt will occur when the counter reaches 0x300 （BIT8 from 0 to 1）. Therefore, selecting BIT8 as 1 to generate interrupt means that the interrupt occurs every 512 counts. Please notice that if T16M counter is restarted, the next interrupt will occur once Bit8 turns from 0 to 1.

6. Using ADC

(1) Configure corresponding IO as input through PXDIER register

(2) The recommended highest ADC conversion frequency is 500kHz and maximum output impedance of analog signal source is 10KΩ.

(3) For PDK82C series IC, the ADC has around 15mV OFFSET voltage. It means the input signal voltage below 15mV will obtain “0” data.

(4) Please pay attention on sequence of operation if the program fits for below conditions,

1. Use the FPP (ex. FPPA0) for handling power-save mode to disable ADC.
2. Use the FPP (ex. FPPA1) for handling ADC conversion to enable ADC and wait for completion of ADC conversion with **WAIT1 ADC_Done** instruction.
3. Execute【1】【2】simultaneously

In case the above sequence is not properly arranged, there may be a chance that FPPA1 can not pass through **WAIT1 ADC_Done** instruction because the ADC may be disabled by FPPA0 before **WAIT1 ADC_Done** instruction being executed.

Recommendation:

Define a Flag which represents the operation of ADC. Every time FPPA1 set the flag when enable the ADC and reset it when the completion of ADC conversion. FPPA0 checks this flag and decides to enter power-save and disable ADC if it is reset.

(5) When switching ADC channel, the memory effect may cause a big error at the first ADC conversion value. Please discard and use the latter conversion values.

(6) If the input signal voltage is higher than VDD, the ADC conversion may be interfered.

(7) When using PB1 as ADC reference voltage input (Vref), PB1 should be set as input without pull-up. The ADCD1.1 should be set to 1 for reduce power consumption. At that time the PB1 input resistance is about 100KΩ～200KΩ. Therefore, when using resistive voltage divider circuit to generate the reference voltage, the lower is the resistance values, the smaller is the effect from the PB1 input resistance.

7. About supply voltage, system frequency and circuit power-on time

(1) The Power-on time is related to system frequency. When system clock frequency is 8MHz after boot-up, the power-on time of the customer application circuit must be completed within 1024 ILRC cycles. Customer is recommended to keep power-on time within 10ms. For the longer power-on time circuit, it is recommended to stay in the ILRC mode first and switch to the high speed mode after the supply voltage is stable.

(2) If VDD is below 3V, IHRC frequency will drop rapidly as long as VDD decrease. For more details please refer to datasheet “Typical measurement of IHRC frequency vs VDD” diagram.

