Freescale Semiconductor

GPIO example in Kinetis Design Studio (KDS) with FDRM-K64F

By:

Paul Garate
Augusto Panecatl

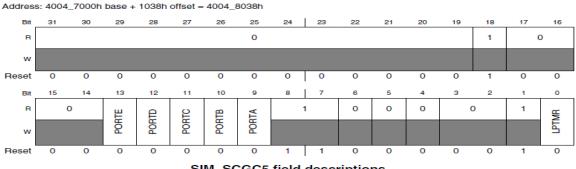
Description

In this document you will find a detailed step by step guide of how to configure the GPIO's on Kinetis K devices using Kinetis Design Studio, using a Switch and the RGB Led included in the FRDM-K64F120 evaluation board.

1. Clock Gating

First of all, we need to enable the clock gate corresponding to the ports we will use. The RGB Led and the switch 2 are in ports B, C and E respectively.

12.2.12 System Clock Gating Control Register 5 (SIM_SCGC5)



SIM_SCGC5 field descriptions

	13 PORTE	Port E Clock Gate Control
		This bit controls the clock gate to the Port E module.
		0 Clock disabled
		1 Clock enabled
	12 PORTD	Port D Clock Gate Control
		This bit controls the clock gate to the Port D module.
		0 Clock disabled
		1 Clock enabled
Ī	11	Port C Clock Gate Control
	PORTC	This bit controls the clock gate to the Port C module.

SIM_SCGC5_PORTn_MASK are defined as mask to enable the module's clock, where "n" corresponds to the specific GPIO PORT we want to activate, i.e:

SIM SCGC5 = SIM SCGC5 PORTB MASK;

By declaring the mask we are writing 0x400 to the SIM_SCGC5 register, setting up the 10th bit of the System Clock Gating Control Register 5 which enables Port B; since the RGB led and Switch 2 in the FRDM-K64 board are assigned to GPIO pins in the B, C and E ports we need to enable the clock gating to all those ports.

2. Pin Control Register configuration

Once the clock gating has been setup we need to configure the pin function using the multiplexor, according to the FRDM-K64's schematic the RGB led is assigned to pins:

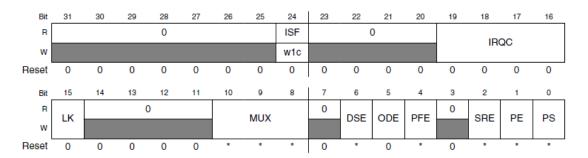
LED	K64
RED	PTB22/SPI2_SOUT/FB_AD29/CMP2_OUT
BLUE	PTB21/SPI2_SCK/FB_AD30/CMP1_OUT
GREEN	PTE26/ENET_1588_CLKIN/UART4_CTS_b/RTC_CLKOUT/USB0_CLKIN

Switch 2 is assigned to:

Switch	GPIO Function
SW2	PTC6/SPI0_SOUT/PD0_EXTRG/I2S0_RX_BCLK/FB_AD9/I2S0_MCLK/LLWU_P10
SW3	PTA4/FTM0_CH1/NMI_b/LLWU_P3

The next step is to configure the Pin Control Register to define the pin function:

Pin Control Register n (PORTx_PCRn)



The only bits we need to configure are those assigned to the MUX field, the pins need to be set as GPIO.

2 Freescale Semiconductor

```
Pin Mux Control

Not all pins support all pin muxing slots. Unimplemented pin muxing slots are reserved and may result in configuring the pin for a different pin muxing slot.

The corresponding pin is configured in the following pin muxing slot as follows:

000 Pin disabled (analog).
001 Alternative 1 (GPIO).
010 Alternative 2 (chip-specific).
011 Alternative 3 (chip-specific).
100 Alternative 4 (chip-specific).
110 Alternative 5 (chip-specific).
111 Alternative 7 (chip-specific).
```

According to the board's schematic we need to configure the following pins as GPIOs: **R= PortB pin** 22, **G= PortE pin** 26, **B= PortB pin** 21, **Switch** 2= **PortC pin** 6.

In configuration register **PORTx_PCRn** "x" corresponds to the port whilst "n" corresponds to the pin

```
PORTB_PCR21 = 0x100; /*Blue Led, configured as Alternative 1 (GPIO)*/
PORTB_PCR22 = 0x100; /*Red Led, configured as Alternative 1 (GPIO)*/
PORTE_PCR26 = 0x100; /*Green Led, configured as Alternative 1 (GPIO)*/
PORTC_PCR6 = 0X100; /*Switch 2, configured as Alternative 1 (GPIO)*/
PORTA_PCR4 = 0x100; /*Changing the NMI to GPIO*/
```

3. Setting up the port data direction

Now that the clock gating and the pin function have been configured we need to setup the pin direction either as Input or Output; this is configured in the **GPIOx PDDR** register:

55.2.6 Port Data Direction Register (GPIOx_PDDR)

The PDDR configures the individual port pins for input or output.

Field	Description
31–0 PDD	Port Data Direction Configures individual port pins for input or output.
	Pin is configured as general-purpose input, for the GPIO function. Pin is configured as general-purpose output, for the GPIO function.

^{*(0}x100 = 100000000)

^{*}The 4th pin of Port A is configured as GPIO to avoid Switch 3 triggering the NMI.

As in the previous registers the " \mathbf{x} " corresponds to the port, and each of the 32 bits in the register corresponds to a single port pin. The pins can either be setup one by one or the whole port can be configured in a single write.

The three pins connected to the leds must be set as outputs and the switch as an input:

In this case we configure the pins by shifting the value to the corresponding bit, 1 or 0 shifted "n" number of places.

4. Code

Declare a variable and value for the delay value

```
unsigned long Counter = 0x100000; /*Delay Value*/
```

Now the code starts reading the status of Switch2 (GPIOC_PDIR).

If the switch is press the microcontroller will start with the sequence, if not the microcontroller will remain idle.

The sequence turns on the Red led (GPIOB_PDOR = (1 << 21)), then the Counter starts decreasing until it reaches 0, the counter will recharge its value (Counter = 0x100000), and the Red led will turn off (GPIOB PDOR = (1 << 22) | (1 << 21)) repeteadly.

In the next part of the sequence the Green Led turns on ($GPIOE_PDOR = (0 << 26)$), wait for Counter = 0, after that the Green Led turns off and waits again for Counter = 0.

And finally the Blue Led turns on, wait for Counter = 0, then the Blue Led turns off and waits again for Counter = 0.

All this code is in a **For(;;)** infinite cycle, the code will be reading the status of Switch 2, waiting for it to be pressed.

5 Freescale Semiconductor