IO Ports – structure and operations

KF6010 – Distributed Real Time Systems

Dr Alun Moon
alun.moon@northumbria.ac.uk

Lecture 1a

Useful documentation

Cortex-M4 Devices Generic User Guide [M4-UG] gives an overview of the device and details of the instruction-set.

Cortex-M4 Technical Reference Manual [M4-TRM] has details about the memory model (including bit-aliasing [M4-TRM, sec 3.7]) and the interrupt controller [M4-TRM, chap 6]

FRDM-K64F Freedom Module User's Guide [K64F-User] gives an overview of the board and the connectivity (circuit diagrams) of the devices on the board.

K64 Sub-Family Reference Manual [K64F-Ref]

Extract – blinky-bare

```
/* Enable the clock to PORT B */
SIM_SCGC5 |= SIM_SCGC5_PORTB_MASK;

/* Select the GPIO function (Alternative 1) for pin 21 o
PORTB_PCR21 &= ~PORT_PCR_MUX_MASK;
PORTB_PCR21 |= (1u << PORT_PCR_MUX_SHIFT);

/* Set the data direction for pin 21 of PORT B to output
GPIOB_PDDR |= PIN21_MASK;

while (true) {
    /* Turn on the blue LED */
    GPIOB_PDOR &= ~PIN21_MASK;
    /* Wait for about 1 second */
        delay(1000);
    /* Turn off the blue LED */
    GPIOB_PDOR |= PIN21_MASK;
```

Setup Clock Signals

```
/* Enable the clock to PORT B */
SIM_SCGC5 |= SIM_SCGC5_PORTB_MASK;
```

- enables the GPIO port-B
- provides clock-signal and power
- set bit 10 in the "System Clock Gating Control Register 5 (SIM_SCGC5)" [M4-TRM, sec 12.2.12]

Setup Pin Function

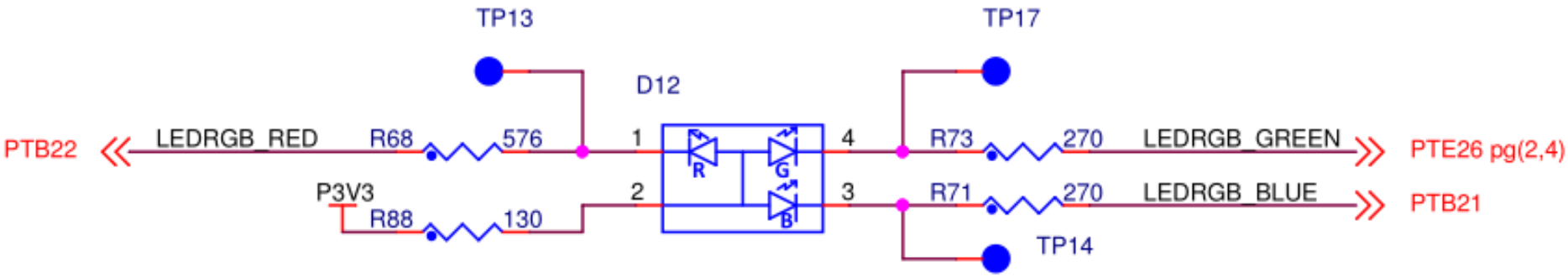
Pin Control Register

Every output pin has a 32bit register that controls the overall operating mode of the pin.

```
PORTB_PCR21 &= ~PORT_PCR_MUX_MASK;
PORTB_PCR21 |= (1u << PORT_PCR_MUX_SHIFT);
```

- selects the GPIO function for bit 21 in port-B
- set bits 10–8 to pattern 001
- clear three bits then set one [M4-TRM, sec 11.5.1, pg283]

LED configuration




```
/* Turn on the blue LED */
GPIOB_PDOR &= ~PIN21_MASK;
```

- Why do we write a 0 to the pin to turn on the LED?

Bit Twiddling

- The operations on the registers are examples of *bit-twiddling*.
- These make use of Boolean identities

And

$$A \wedge \top = A$$

$$A \wedge \bot = \bot$$

Or

$$A \vee \top = \top$$

$$A \vee \bot = A$$

Masks

- By creating a pattern of 0s and 1s in a Mask we can operate on 32bit numbers
- A Mask has 1s in the bits of interest and zeros elsewhere.
- Set a bit `val |= mask`
- Clear a bit `val &= ~mask`

note

These can be 'expensive' operations

```
****            GPIOB_PDOR &= ~PIN21_MASK;
    ldr r3, .L3+12
    ldr r3, [r3]
    ldr r2, .L3+12
    bic r3, r3, #2097152
    str r3, [r2]
```

Port Set, Clear, and Toggle Registers

- To make for more efficient operation the designers have included Set, Clear, and Toggle registers
- Writing a 1 to these registers effects the corresponding pin

```
****  /* alternate clear register */
****  GPIOB_PCOR = PIN21_MASK;
    ldr r3, .L3+16
    mov r2, #2097152
    str r2, [r3]
```

Bibliography

M4-TRM. *Cortex-M4 Technical Reference Manual*. ARM Limited, 2010. http://hesabu.net/kf6010/DDI0439B_cortex_m4_r0p0_trm.pdf.

M4-UG. *Cortex-M4 Devices Generic User Guide*. ARM Limited, 2010. http://hesabu.net/kf6010/DUI0553A_cortex_m4_dgug.pdf.

K64F-Ref. *K64 Sub-Family Reference Manual*. Freescale Semiconductor, Inc., rev. 2 edition, January 2014. http://hesabu.net/kf6010/K64P144M120SF5RM.pdf.

K64F-User. *FRDM-K64F Freedom Module User's Guide*. NXP Semiconductors, rev. 1 edition, August 2016. http://hesabu.net/kf6010/FRDMK64FUG.pdf.