NXP LPC1768 fast usage guide

Reference manual:

LPC176x_USER_MANUAL.pdf

Schematic:

HY-LandTiger_BOARD_SCHEMATIC.pdf

Example:

Sample project

Paolo Bernardi – Politecnico di Torino

Composition of the chip

- ARM 32-bit Cortex-M3 Microcontroller with MPU, CPU clock up to 100MHz,
- 512kB on-chip Flash ROM with enhanced Flash Memory Accelerator,
- Four 32-bit Timers with capture/compare, Standard PWM Timer block,
- 64kB RAM, Nested Vectored Interrupt Controller,
- Eight channel General purpose DMA controller, AHB Matrix, APB,
- System Tick Timer, Repetitive Interrupt Timer, Brown-out detect circuit,

Composition of the chip (II)

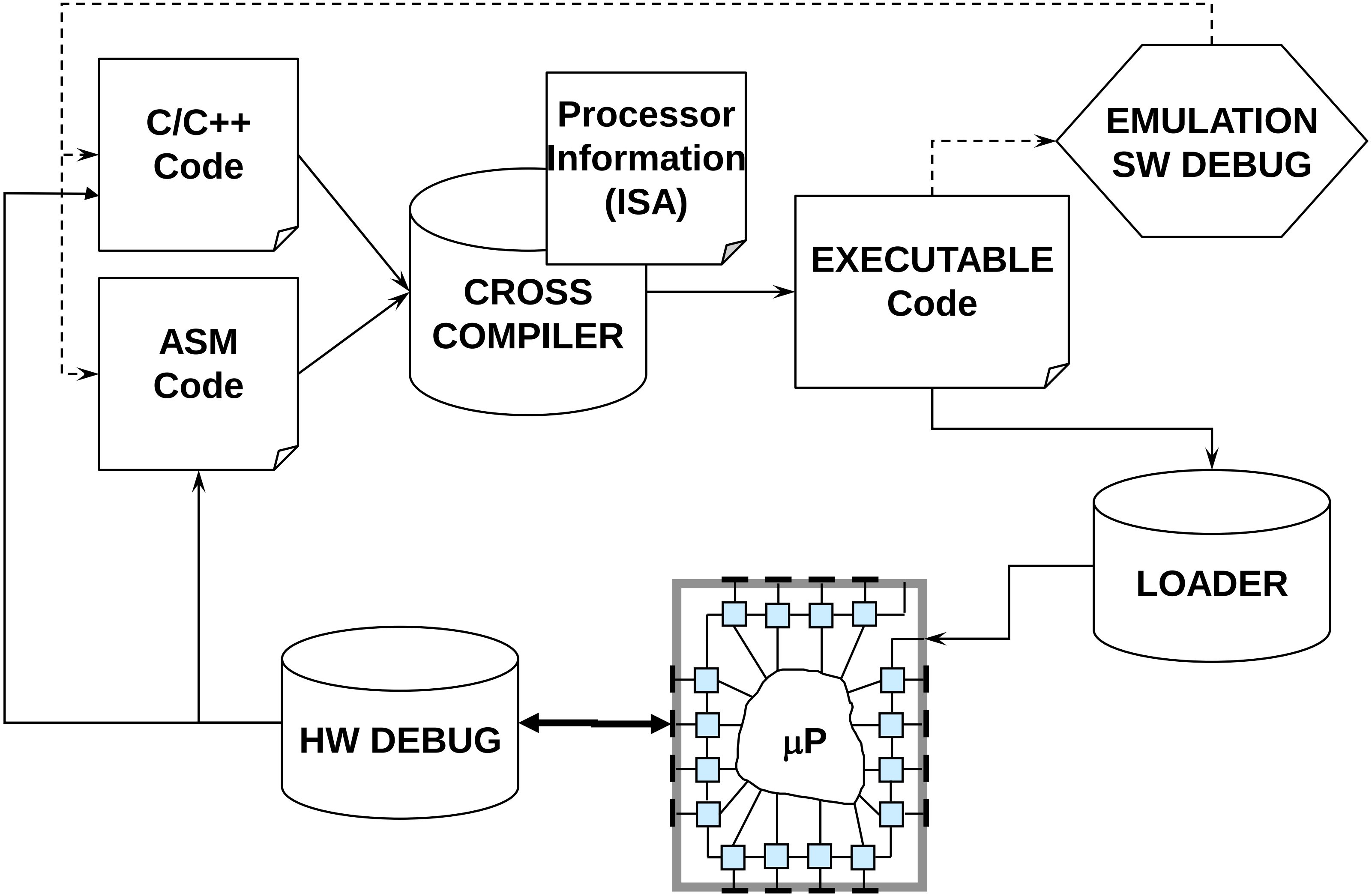
- Ethernet 10/100 MAC with RMII interface and dedicated DMA,
- USB 2.0 full-speed Device controller and Host/OTG controller with DMA,
- CAN 2.0B with two channels, Four UARTs, one with full Modem interface,
- Three I2C serial interfaces, Three SPI/SSP serial interfaces, I2S interface,
- General purpose I/O pins, 12-bit ADC with 8 channels, 10-bit DAC,

Composition of the chip(III)

- Motor control PWM for three-phase Motor control, Quadrature Encoder,
- Watchdog Timer, Real Time Clock with optional Battery backup,

- Power-On Reset, Power Management Unit, Wakeup Interrupt Controller,
- Crystal oscillator, 4MHz internal RC oscillator, PLL,

Tool chain


C/C++ Code
Processor Information (ISA)
EMULATION SW DEBUG
CROSS COMPILER
EXECUTABLE Code
ASM Code
LOADER
HW DEBUG
µP

Block diagram of the Chip (p.9)

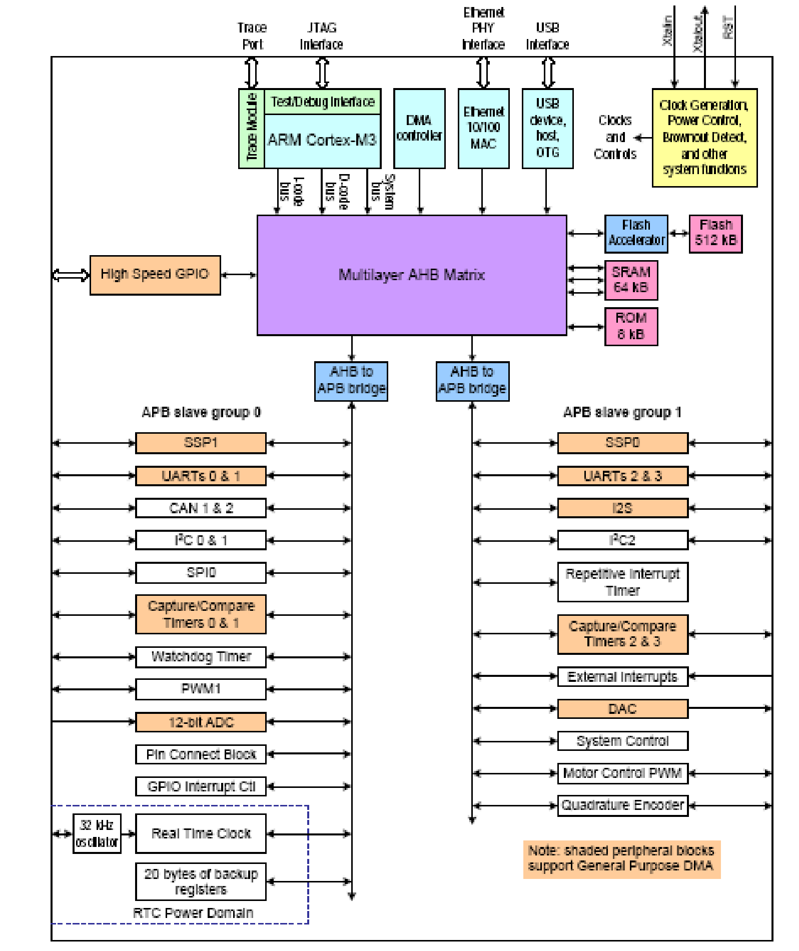
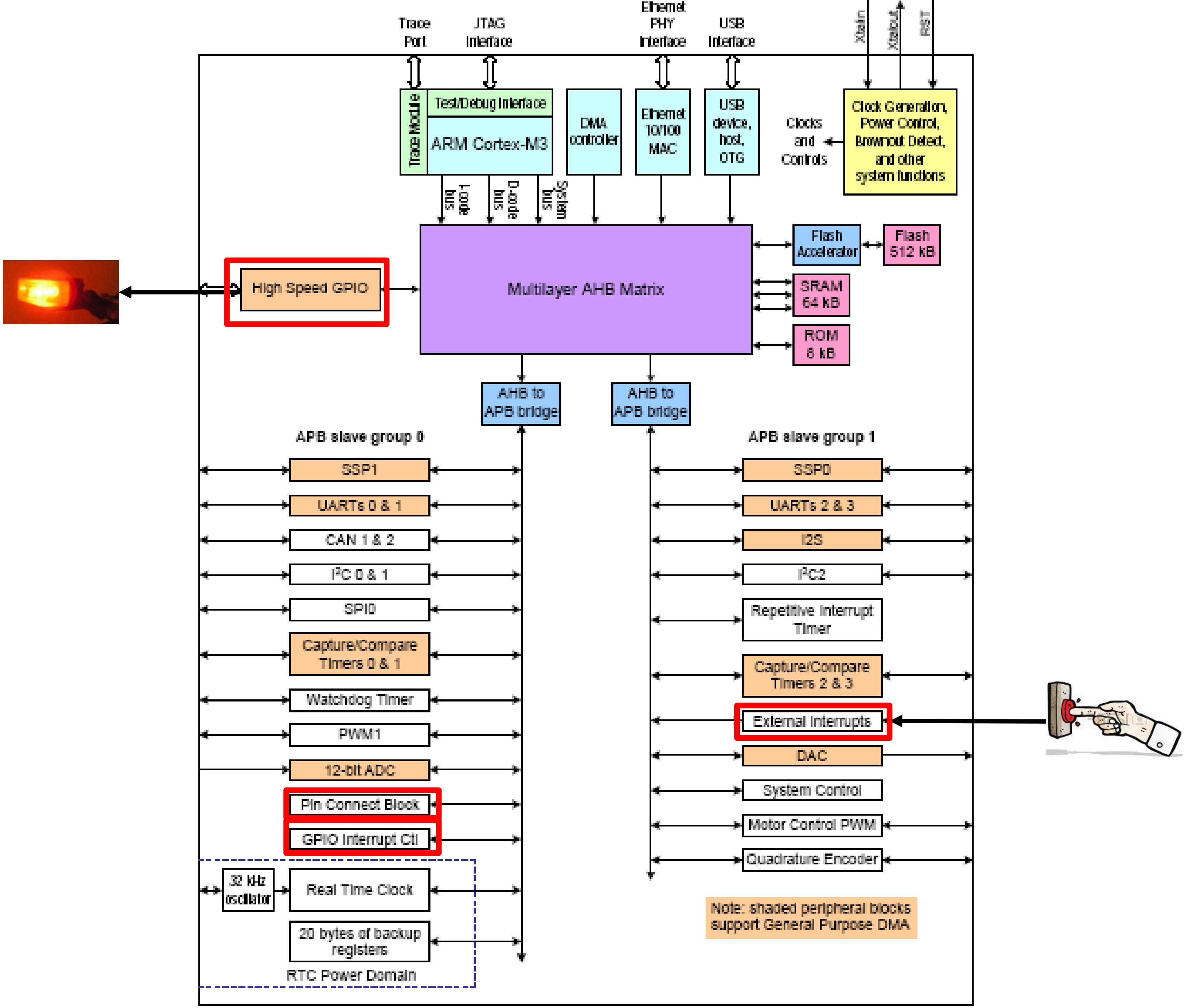


Fig 1. LPC1768 simplified block diagram


Trace Port
JTAG Interface
Ethernet PHY Interface
USB Interface
Xtalin
Xtalout
RST
Trace Module
Test/Debug Interface
ARM Cortex-M3
DMA controller
Ethernet 10/100 MAC
USB device, host, OTG
Clocks and Controls
Clock Generation, Power Control, Brownout Detect, and other system functions
I-code bus
D-code bus
System bus
High Speed GPIO
Multilayer AHB Matrix
Flash Accelerator
Flash 512 kB
SRAM 64 kB
ROM 8 kB
AHB to APB bridge
AHB to APB bridge
APB slave group 0
SSP1
UARTs 0 & 1
CAN 1 & 2
I²C 0 & 1
SPI0
Capture/Compare Timers 0 & 1
Watchdog Timer
PWM1
12-bit ADC
Pin Connect Block
GPIO Interrupt Ctl
32 kHz oscillator
Real Time Clock
20 bytes of backup registers
RTC Power Domain
APB slave group 1
SSP0
UARTs 2 & 3
I2S
I²C2
Repetitive Interrupt Timer
Capture/Compare Timers 2 & 3
External Interrupts
DAC
System Control
Motor Control PWM
Quadrature Encoder
Note: shaded peripheral blocks support General Purpose DMA

Sample project

- It is a base project that includes
  - boot of the chip
    - startup_LPC17xx.s

    includes libraries
    - core_cm3.c
    - system_LPC17xx.c & system_LPC17xx.c
  - The libraries of some peripheral core according to the following convention
    - *peripheral*.h /* prototypes */
    - lib_*peripheral*.c /* base functions */
    - IRQ_*peripheral*.c /* interrupt service routines */
    - funct_*peripheral*.c /* advanced user functions */

startup.s

- Has two main functionalities
  - To define stack e heap
  - To define the interrupt vector table.

Clock

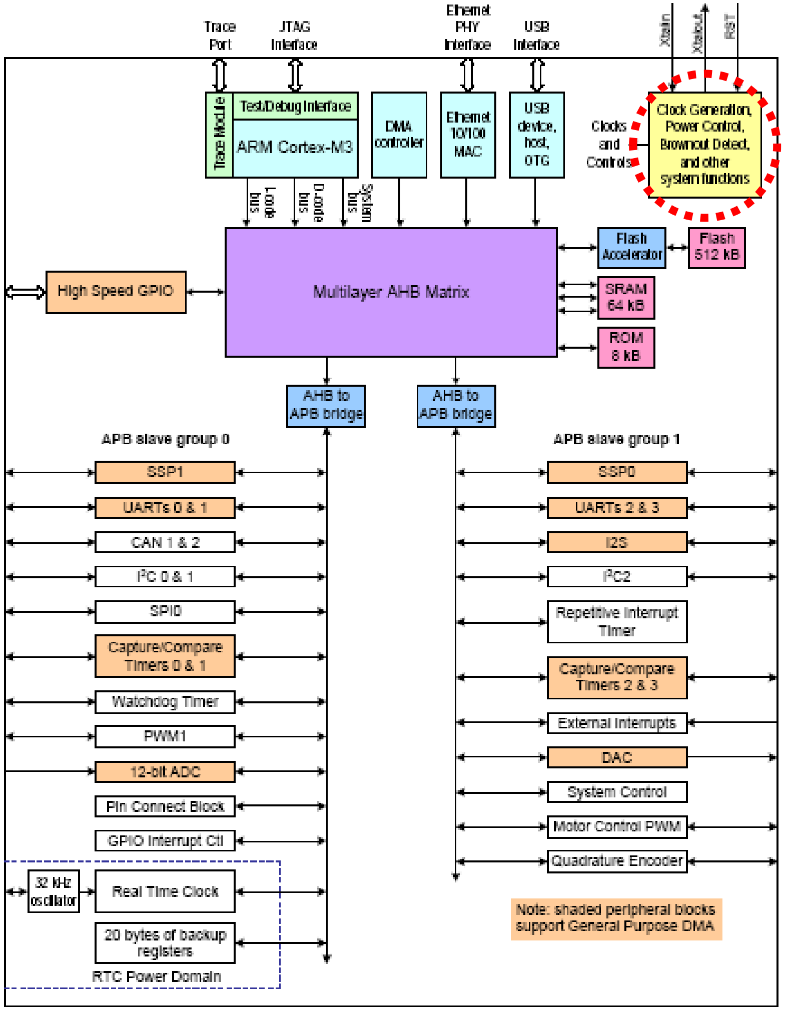
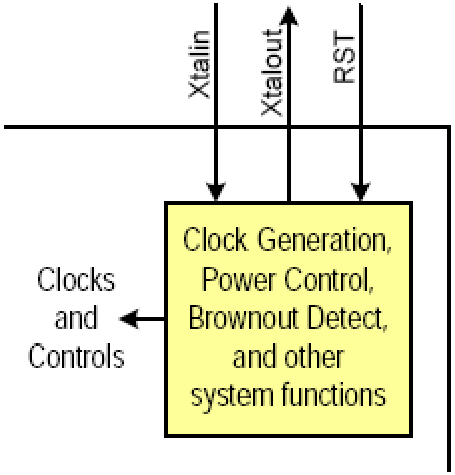
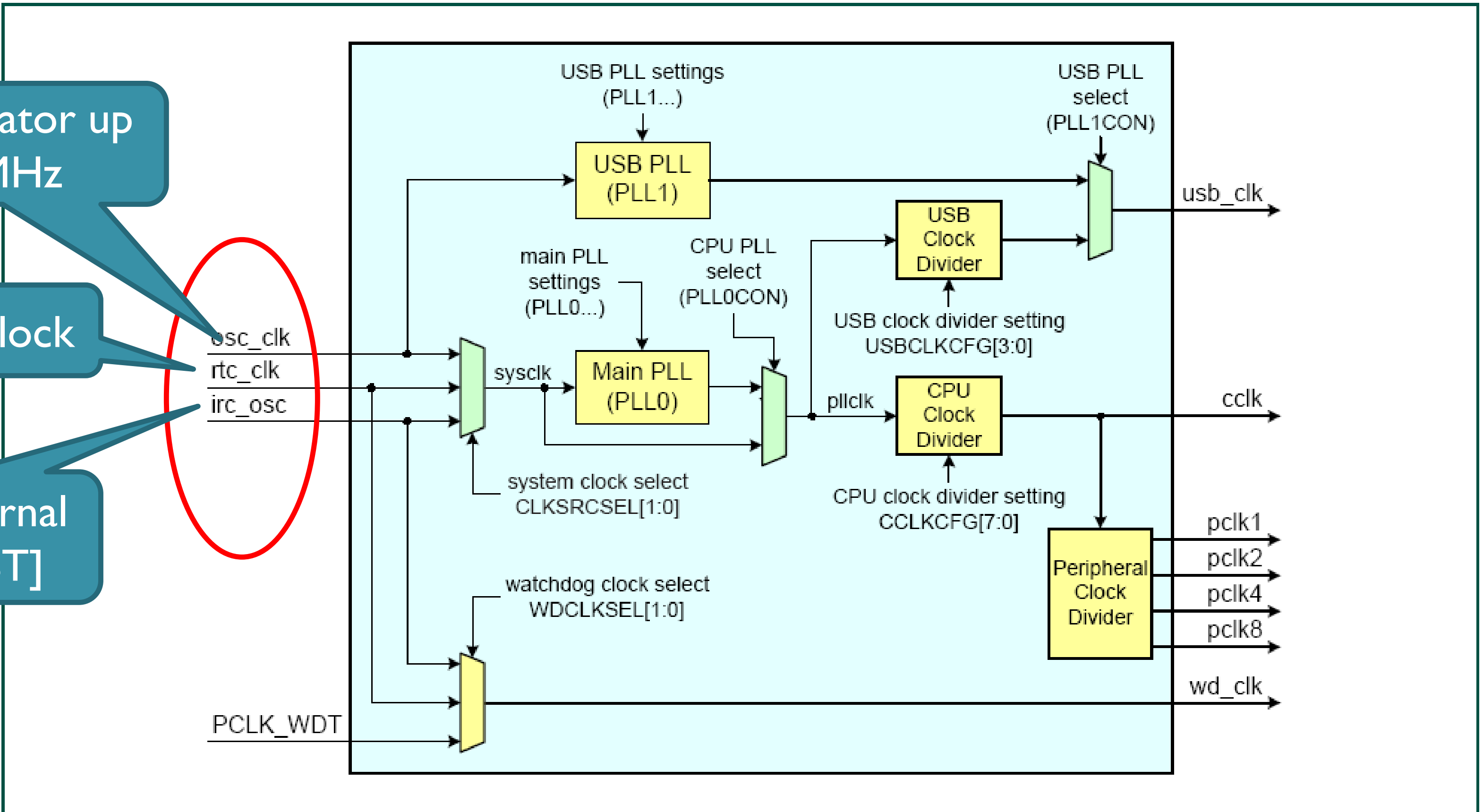


Fig 1. LPC1768 simplified block diagram


Xtalin
Xtalout
RST
Clocks and Controls
Clock Generation, Power Control, Brownout Detect, and other system functions

Clock selection



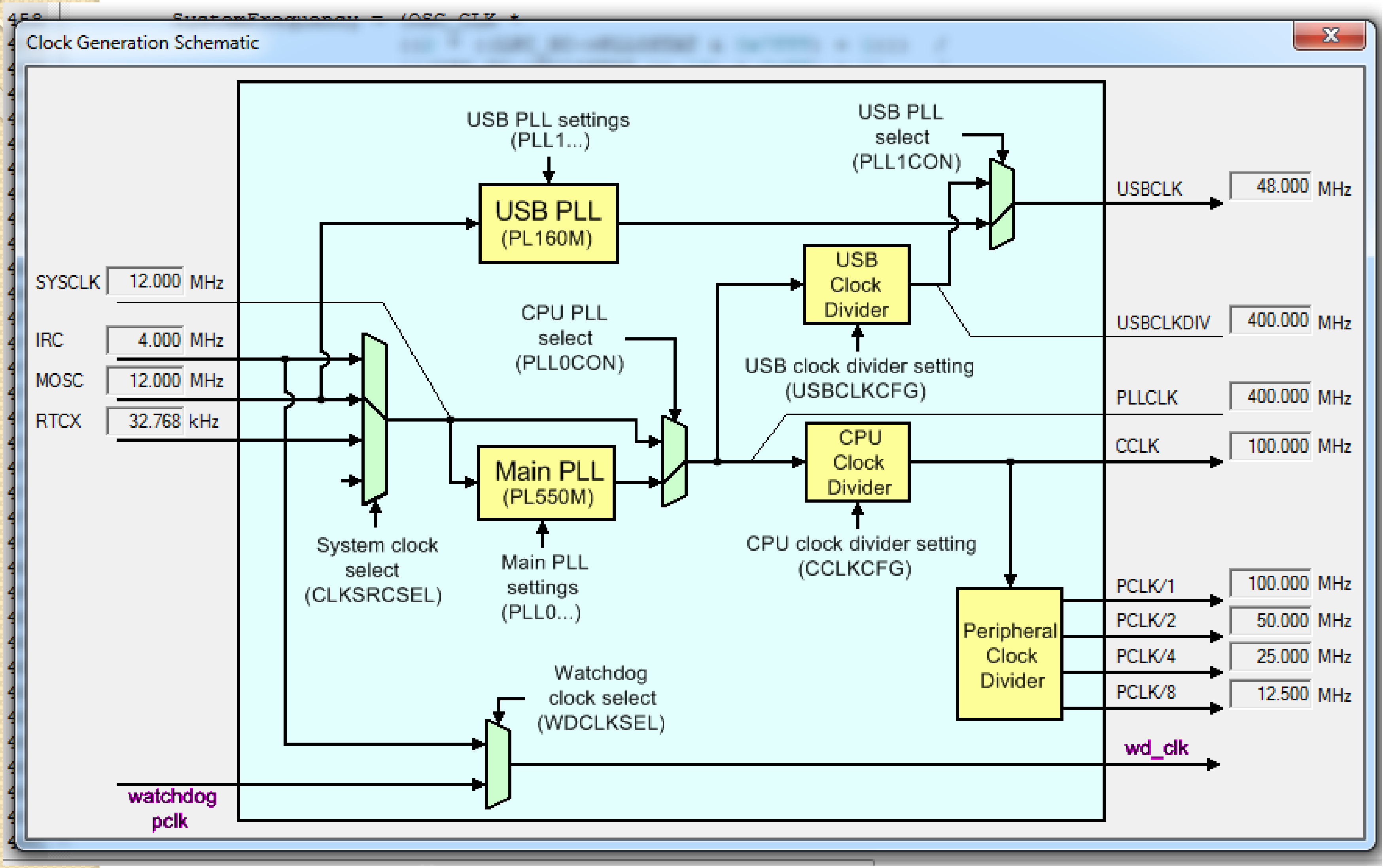
Main oscillator up to 25MHz

32KHz wd clock

4MHz internal clock [RST]

Fig 7. Clock generation for the LPC176x/5x

Current values in *sample*



Clock Generation Schematic

SYSCLK 12.000 MHz

IRC 4.000 MHz

MOSC 12.000 MHz

RTCX 32.768 kHz

USBCLK 48.000 MHz

USBCLKDIV 400.000 MHz

PLLCLK 400.000 MHz

CCLK 100.000 MHz

PCLK/1 100.000 MHz

PCLK/2 50.000 MHz

PCLK/4 25.000 MHz

PCLK/8 12.500 MHz

USB PLL settings (PLL1...)
USB PLL select (PLL1CON)
USB PLL (PL160M)
USB Clock Divider
CPU PLL select (PLL0CON)
USB clock divider setting (USBCLKCFG)
CPU Clock Divider
Main PLL (PL550M)
System clock select (CLKSRCSEL)
Main PLL settings (PLL0...)
CPU clock divider setting (CCLKCFG)
Peripheral Clock Divider
Watchdog clock select (WDCLKSEL)
wd_clk
watchdog pclk

Current configuration

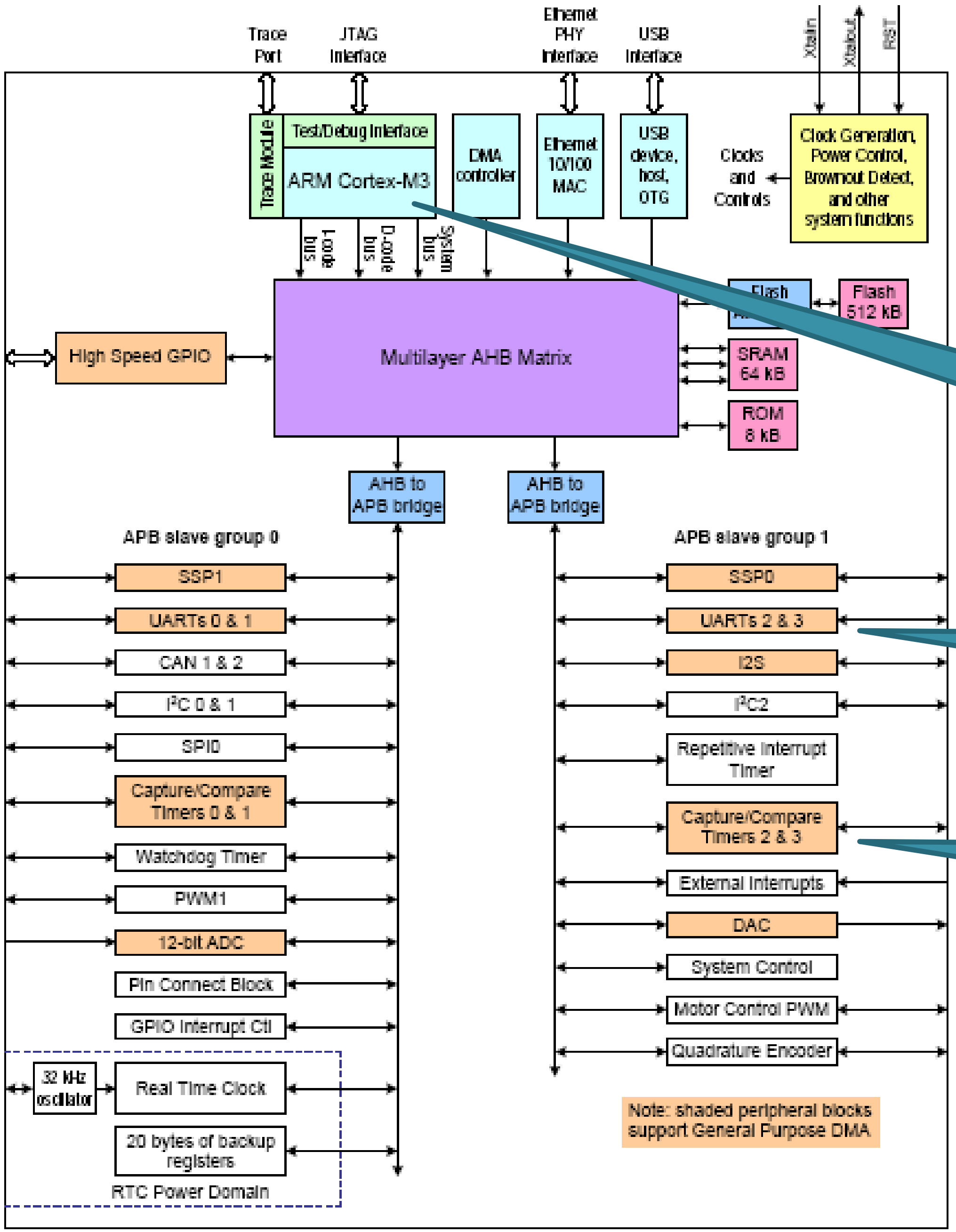


Fig 1. LPC1768 simplified block diagram

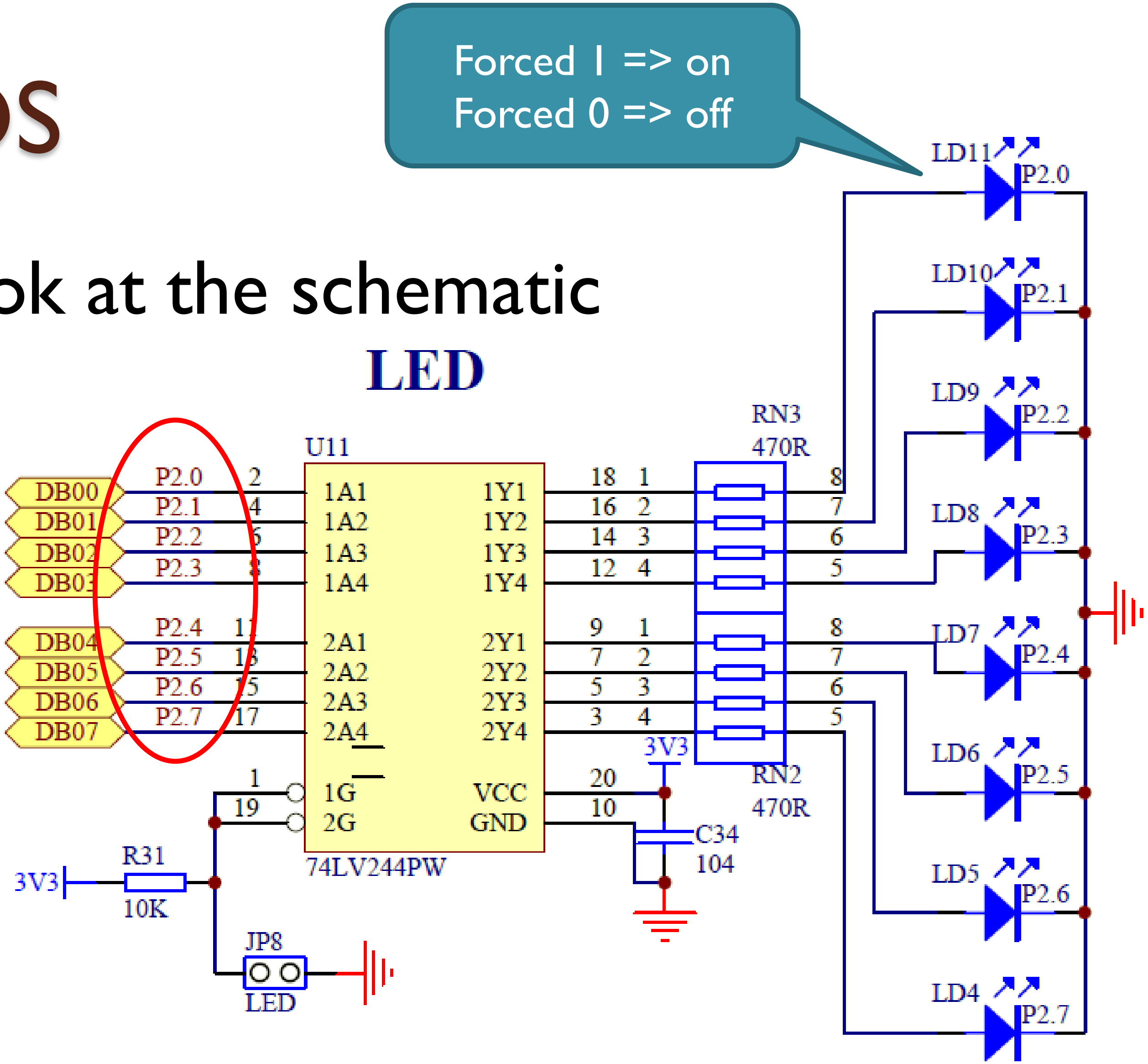
CCLK
100MHz

CCLK/4
25MHz

CCLK/4
25MHz

LEDS

- Look at the schematic



Forced 1 => on
Forced 0 => off

- LD4 → p2.7 LD5 → p2.8 LD11 → p2.0 etc....

Pin connect block (pag. 113)

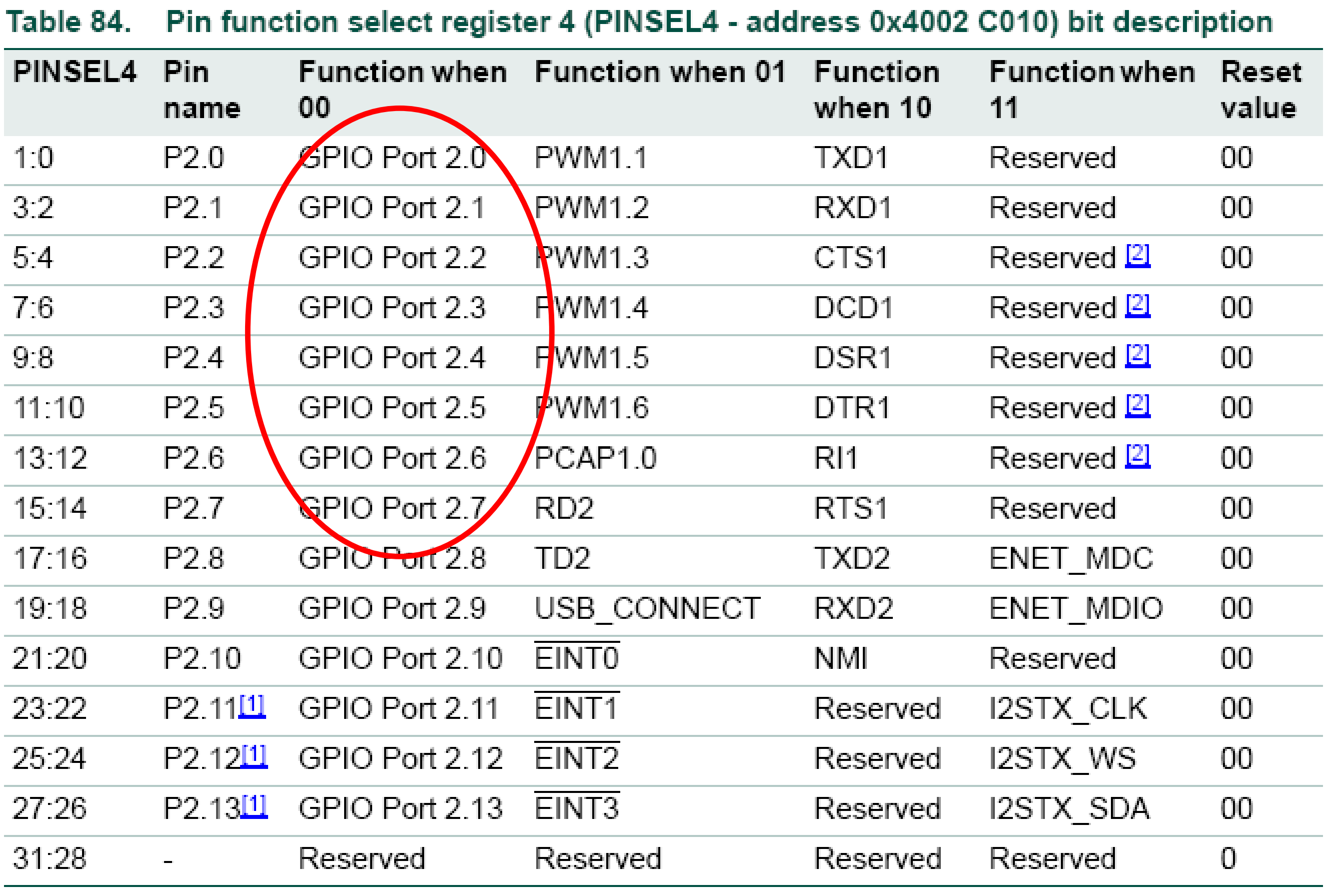


Table 84. Pin function select register 4 (PINSEL4 - address 0x4002 C010) bit description

PINSEL4	Pin name	Function when 00	Function when 01	Function when 10	Function when 11	Reset value
1:0	P2.0	GPIO Port 2.0	PWM1.1	TXD1	Reserved	00
3:2	P2.1	GPIO Port 2.1	PWM1.2	RXD1	Reserved	00
5:4	P2.2	GPIO Port 2.2	PWM1.3	CTS1	Reserved [2]	00
7:6	P2.3	GPIO Port 2.3	PWM1.4	DCD1	Reserved [2]	00
9:8	P2.4	GPIO Port 2.4	PWM1.5	DSR1	Reserved [2]	00
11:10	P2.5	GPIO Port 2.5	PWM1.6	DTR1	Reserved [2]	00
13:12	P2.6	GPIO Port 2.6	PCAP1.0	RI1	Reserved [2]	00
15:14	P2.7	GPIO Port 2.7	RD2	RTS1	Reserved	00
17:16	P2.8	GPIO Port 2.8	TD2	TXD2	ENET_MDC	00
19:18	P2.9	GPIO Port 2.9	USB_CONNECT	RXD2	ENET_MDIO	00
21:20	P2.10	GPIO Port 2.10	EINT0	NMI	Reserved	00
23:22	P2.11[1]	GPIO Port 2.11	EINT1	Reserved	I2STX_CLK	00
25:24	P2.12[1]	GPIO Port 2.12	EINT2	Reserved	I2STX_WS	00
27:26	P2.13[1]	GPIO Port 2.13	EINT3	Reserved	I2STX_SDA	00
31:28	-	Reserved	Reserved	Reserved	Reserved	0

General purpose I/O

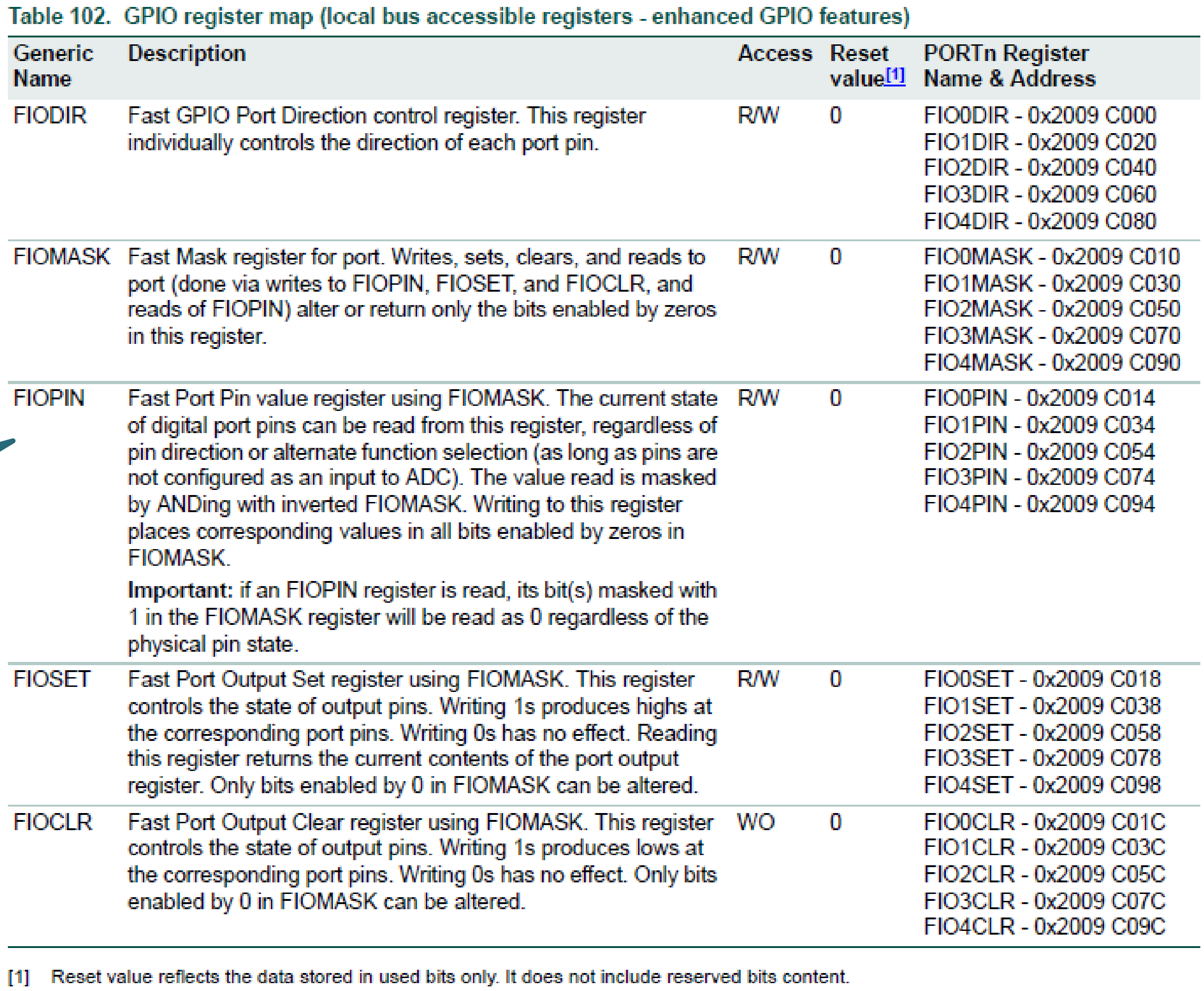


Table 102. GPIO register map (local bus accessible registers - enhanced GPIO features)

Generic Name	Description	Access	Reset value[1]	PORTn Register Name & Address
FIODIR	Fast GPIO Port Direction control register. This register individually controls the direction of each port pin.	R/W	0	FIO0DIR - 0x2009 C000 FIO1DIR - 0x2009 C020 FIO2DIR - 0x2009 C040 FIO3DIR - 0x2009 C060 FIO4DIR - 0x2009 C080
FIOMASK	Fast Mask register for port. Writes, sets, clears, and reads to port (done via writes to FIOPIN, FIOSET, and FIOCLR, and reads of FIOPIN) alter or return only the bits enabled by zeros in this register.	R/W	0	FIO0MASK - 0x2009 C010 FIO1MASK - 0x2009 C030 FIO2MASK - 0x2009 C050 FIO3MASK - 0x2009 C070 FIO4MASK - 0x2009 C090
FIOPIN	Fast Port Pin value register using FIOMASK. The current state of digital port pins can be read from this register, regardless of pin direction or alternate function selection (as long as pins are not configured as an input to ADC). The value read is masked by ANDing with inverted FIOMASK. Writing to this register places corresponding values in all bits enabled by zeros in FIOMASK. **Important:** if an FIOPIN register is read, its bit(s) masked with 1 in the FIOMASK register will be read as 0 regardless of the physical pin state.	R/W	0	FIO0PIN - 0x2009 C014 FIO1PIN - 0x2009 C034 FIO2PIN - 0x2009 C054 FIO3PIN - 0x2009 C074 FIO4PIN - 0x2009 C094
FIOSET	Fast Port Output Set register using FIOMASK. This register controls the state of output pins. Writing 1s produces highs at the corresponding port pins. Writing 0s has no effect. Reading this register returns the current contents of the port output register. Only bits enabled by 0 in FIOMASK can be altered.	R/W	0	FIO0SET - 0x2009 C018 FIO1SET - 0x2009 C038 FIO2SET - 0x2009 C058 FIO3SET - 0x2009 C078 FIO4SET - 0x2009 C098
FIOCLR	Fast Port Output Clear register using FIOMASK. This register controls the state of output pins. Writing 1s produces lows at the corresponding port pins. Writing 0s has no effect. Only bits enabled by 0 in FIOMASK can be altered.	WO	0	FIO0CLR - 0x2009 C01C FIO1CLR - 0x2009 C03C FIO2CLR - 0x2009 C05C FIO3CLR - 0x2009 C07C FIO4CLR - 0x2009 C09C

[1] Reset value reflects the data stored in used bits only. It does not include reserved bits content.

Direction
In/out

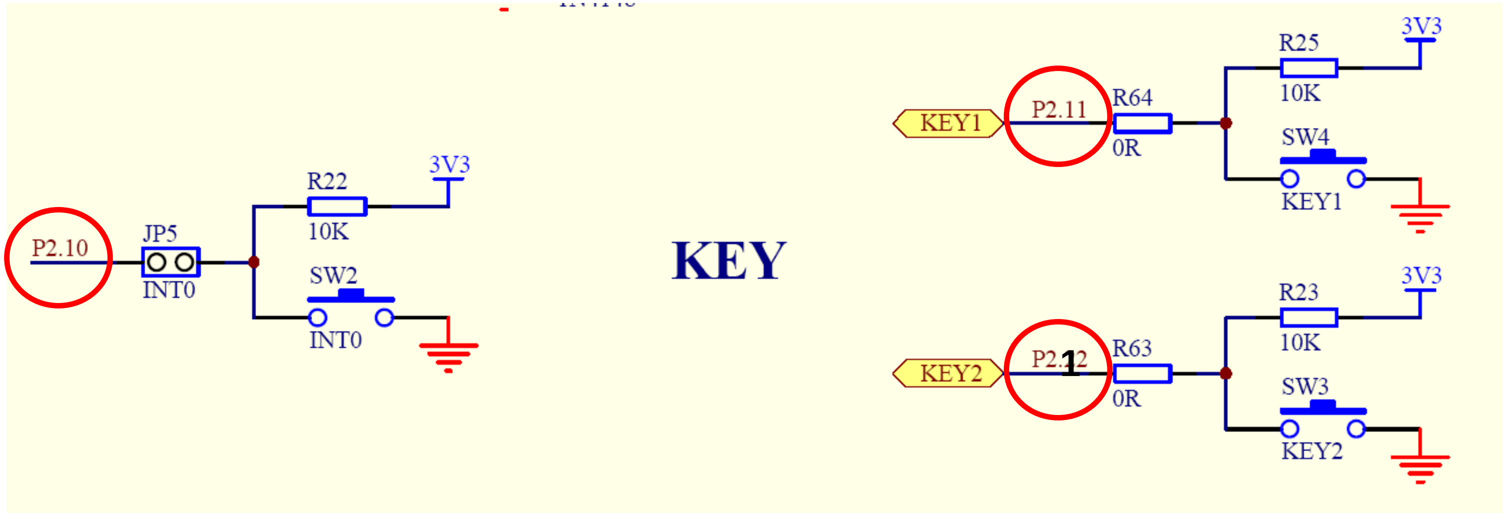
If input
Read pin
value

If output
set/clr

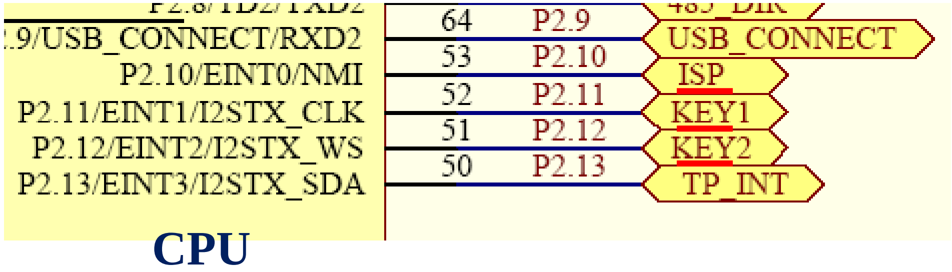
Buttons

released = 1
pressed = 0

- Look at the schematic



- INT0 → p2.10
- KEY1 → p2.11
- KEY2 → p2.12



Pin connect block (pag. 113)

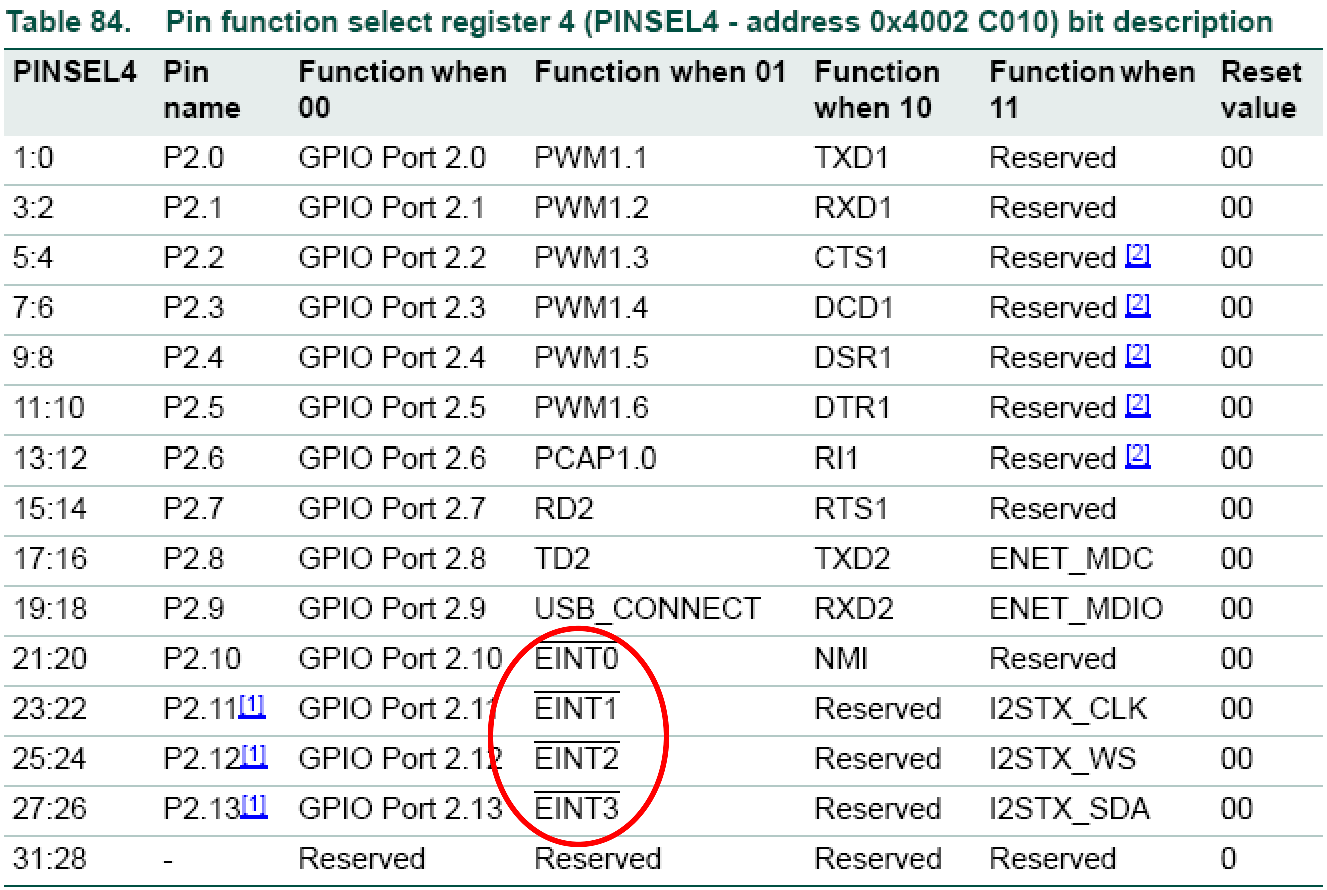
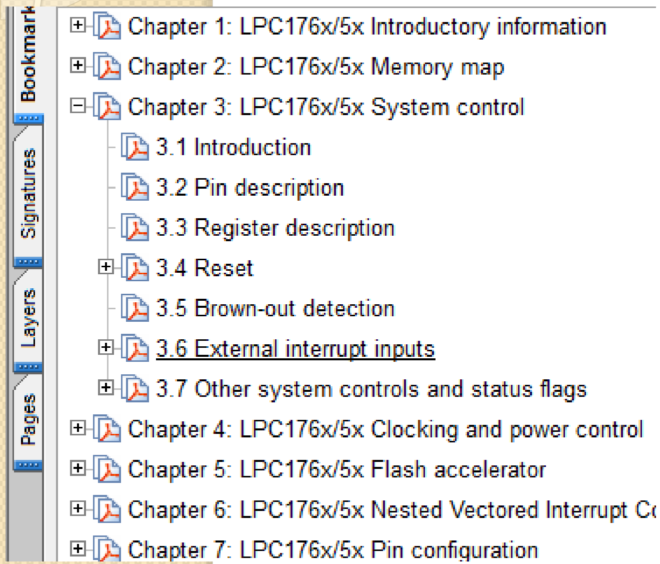


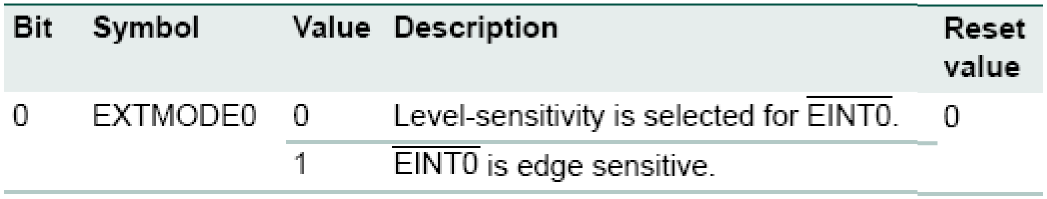
Table 84. Pin function select register 4 (PINSEL4 - address 0x4002 C010) bit description

PINSEL4	Pin name	Function when 00	Function when 01	Function when 10	Function when 11	Reset value
1:0	P2.0	GPIO Port 2.0	PWM1.1	TXD1	Reserved	00
3:2	P2.1	GPIO Port 2.1	PWM1.2	RXD1	Reserved	00
5:4	P2.2	GPIO Port 2.2	PWM1.3	CTS1	Reserved [2]	00
7:6	P2.3	GPIO Port 2.3	PWM1.4	DCD1	Reserved [2]	00
9:8	P2.4	GPIO Port 2.4	PWM1.5	DSR1	Reserved [2]	00
11:10	P2.5	GPIO Port 2.5	PWM1.6	DTR1	Reserved [2]	00
13:12	P2.6	GPIO Port 2.6	PCAP1.0	RI1	Reserved [2]	00
15:14	P2.7	GPIO Port 2.7	RD2	RTS1	Reserved	00
17:16	P2.8	GPIO Port 2.8	TD2	TXD2	ENET_MDC	00
19:18	P2.9	GPIO Port 2.9	USB_CONNECT	RXD2	ENET_MDIO	00
21:20	P2.10	GPIO Port 2.10	$\overline{\text{EINT0}}$	NMI	Reserved	00
23:22	P2.11[1]	GPIO Port 2.11	$\overline{\text{EINT1}}$	Reserved	I2STX_CLK	00
25:24	P2.12[1]	GPIO Port 2.12	$\overline{\text{EINT2}}$	Reserved	I2STX_WS	00
27:26	P2.13[1]	GPIO Port 2.13	$\overline{\text{EINT3}}$	Reserved	I2STX_SDA	00
31:28	-	Reserved	Reserved	Reserved	Reserved	0

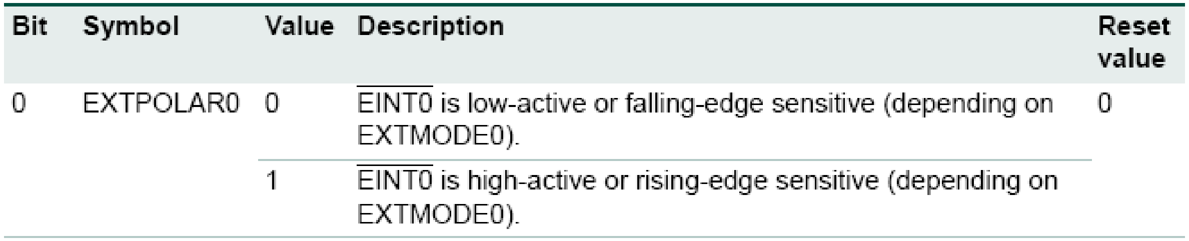
External Interrupt mode



- Chapter 1: LPC176x/5x Introductory information
- Chapter 2: LPC176x/5x Memory map
- Chapter 3: LPC176x/5x System control
  - 3.1 Introduction
  - 3.2 Pin description
  - 3.3 Register description
  - 3.4 Reset
  - 3.5 Brown-out detection
  - 3.6 External interrupt inputs
  - 3.7 Other system controls and status flags
- Chapter 4: LPC176x/5x Clocking and power control
- Chapter 5: LPC176x/5x Flash accelerator
- Chapter 6: LPC176x/5x Nested Vectored Interrupt Co
- Chapter 7: LPC176x/5x Pin configuration



Bit	Symbol	Value	Description	Reset value
0	EXTMODE0	0	Level-sensitivity is selected for $\overline{\text{EINT0}}$.	0
		1	$\overline{\text{EINT0}}$ is edge sensitive.	



Bit	Symbol	Value	Description	Reset value
0	EXTPOLAR0	0	$\overline{\text{EINT0}}$ is low-active or falling-edge sensitive (depending on EXTMODE0).	0
		1	$\overline{\text{EINT0}}$ is high-active or rising-edge sensitive (depending on EXTMODE0).	

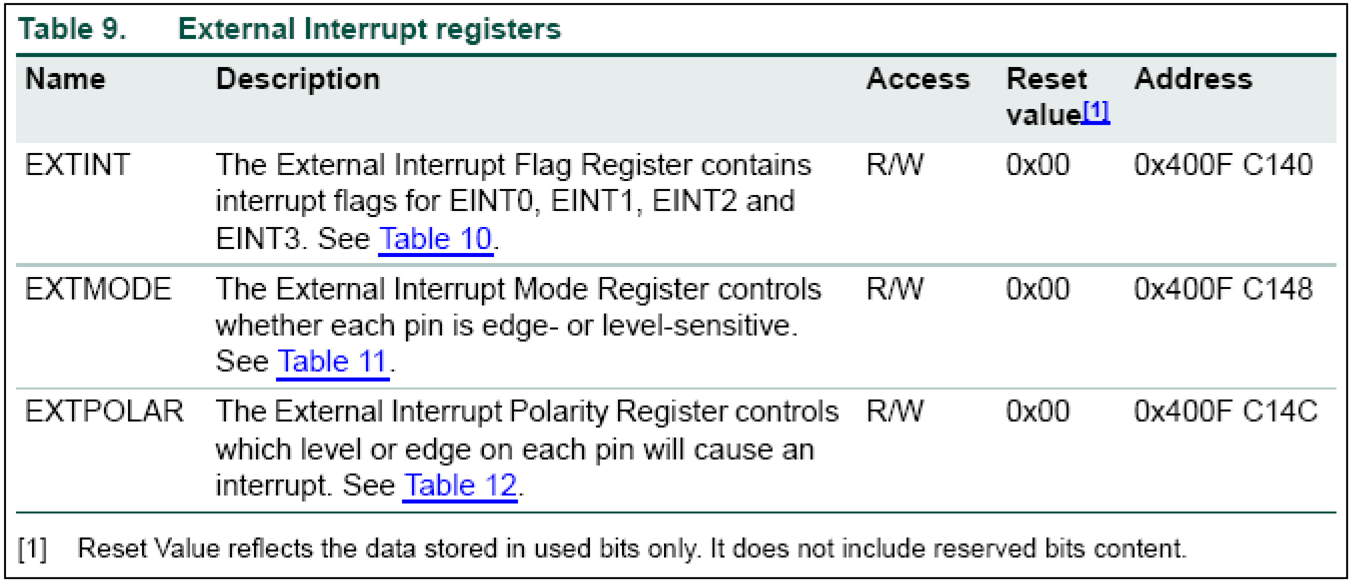


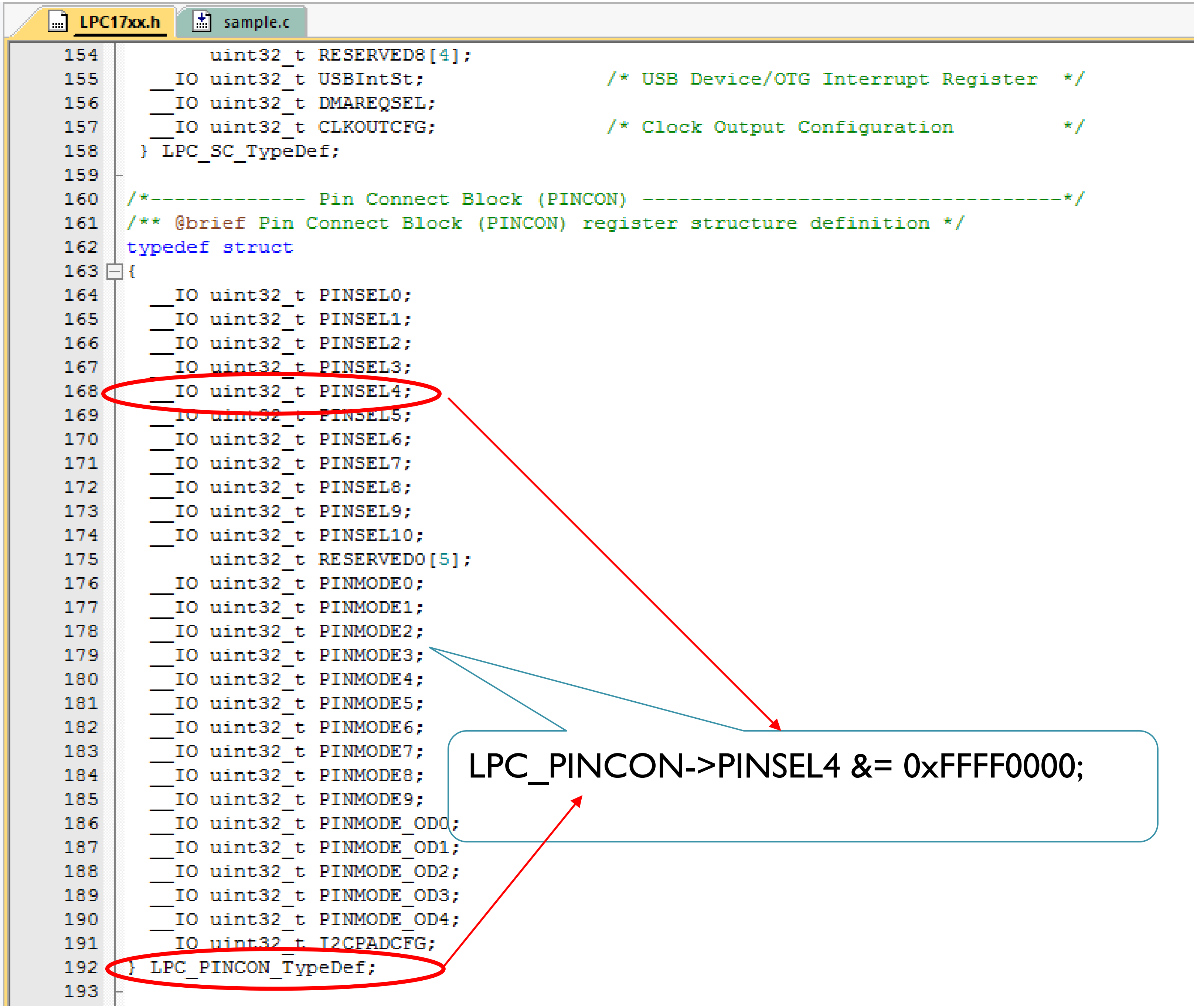
Table 9. External Interrupt registers

Name	Description	Access	Reset value[1]	Address
EXTINT	The External Interrupt Flag Register contains interrupt flags for EINT0, EINT1, EINT2 and EINT3. See Table 10.	R/W	0x00	0x400F C140
EXTMODE	The External Interrupt Mode Register controls whether each pin is edge- or level-sensitive. See Table 11.	R/W	0x00	0x400F C148
EXTPOLAR	The External Interrupt Polarity Register controls which level or edge on each pin will cause an interrupt. See Table 12.	R/W	0x00	0x400F C14C

[1] Reset Value reflects the data stored in used bits only. It does not include reserved bits content.

C programming

- The main program is written in C language, while boot is in ASM
- Please refer to libraries
  - Constants definition in *LPC17xx.h* makes easier the board programming (see next slide)
- Useful directives to link ASM to C
  - EXPORT : makes visible a function outside the file defining it
  - IMPORT : makes visible a function from other files



```c
        uint32_t RESERVED8[4];
  __IO uint32_t USBIntSt;              /* USB Device/OTG Interrupt Register  */
  __IO uint32_t DMAREQSEL;
  __IO uint32_t CLKOUTCFG;             /* Clock Output Configuration         */
 } LPC_SC_TypeDef;

/*------------- Pin Connect Block (PINCON) -----------------------------*/
/** @brief Pin Connect Block (PINCON) register structure definition */
typedef struct
{
  __IO uint32_t PINSEL0;
  __IO uint32_t PINSEL1;
  __IO uint32_t PINSEL2;
  __IO uint32_t PINSEL3;
  __IO uint32_t PINSEL4;
  __IO uint32_t PINSEL5;
  __IO uint32_t PINSEL6;
  __IO uint32_t PINSEL7;
  __IO uint32_t PINSEL8;
  __IO uint32_t PINSEL9;
  __IO uint32_t PINSEL10;
        uint32_t RESERVED0[5];
  __IO uint32_t PINMODE0;
  __IO uint32_t PINMODE1;
  __IO uint32_t PINMODE2;
  __IO uint32_t PINMODE3;
  __IO uint32_t PINMODE4;
  __IO uint32_t PINMODE5;
  __IO uint32_t PINMODE6;
  __IO uint32_t PINMODE7;
  __IO uint32_t PINMODE8;
  __IO uint32_t PINMODE9;
  __IO uint32_t PINMODE_OD0;
  __IO uint32_t PINMODE_OD1;
  __IO uint32_t PINMODE_OD2;
  __IO uint32_t PINMODE_OD3;
  __IO uint32_t PINMODE_OD4;
  __IO uint32_t I2CPADCFG;
} LPC_PINCON_TypeDef;
```

LPC_PINCON->PINSEL4 &= 0xFFFF0000;

C programming (II)

- the extern directive
  - Permits to import a variable from other file (where it is defined)
  - Please see:
    - *unsigned char led_value;*
    - It is allocated file *lib_led.h*
    - It is used in *funct_led.h*
      - *extern unsigned char led_value;*
    - Can be imported (and manipulated with care) by any file in the project
      - like in *sample.c.*

TIMER

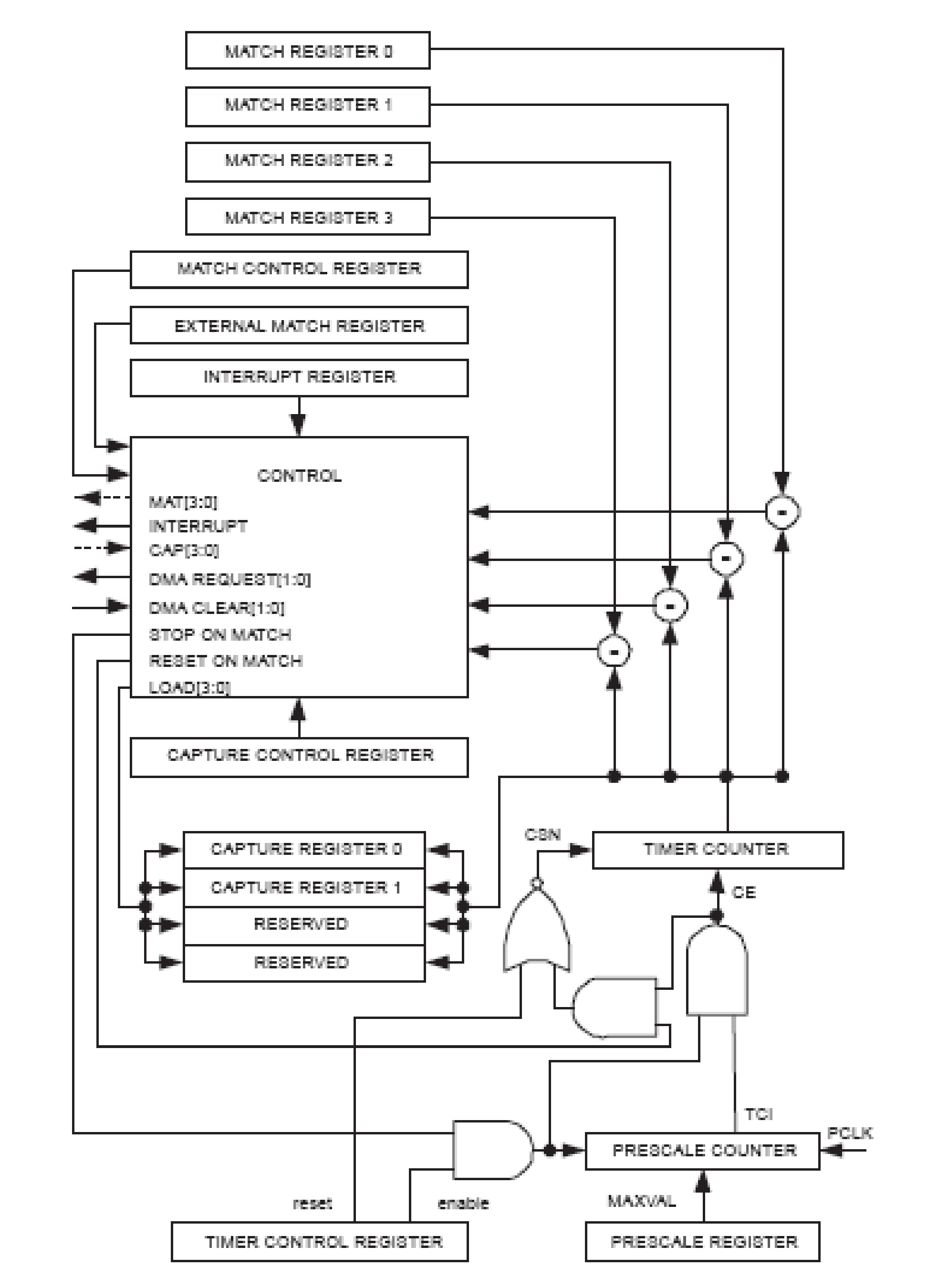


Fig 117. Timer block diagram

Timer 0/1/2/3

- Match register value calculation:

m = t [s] * f [1/s]

- To obtain

t = 10s

→ m = 10 * 25*10^6

→ M = 25*10^7

→ M = 0xEE6B280

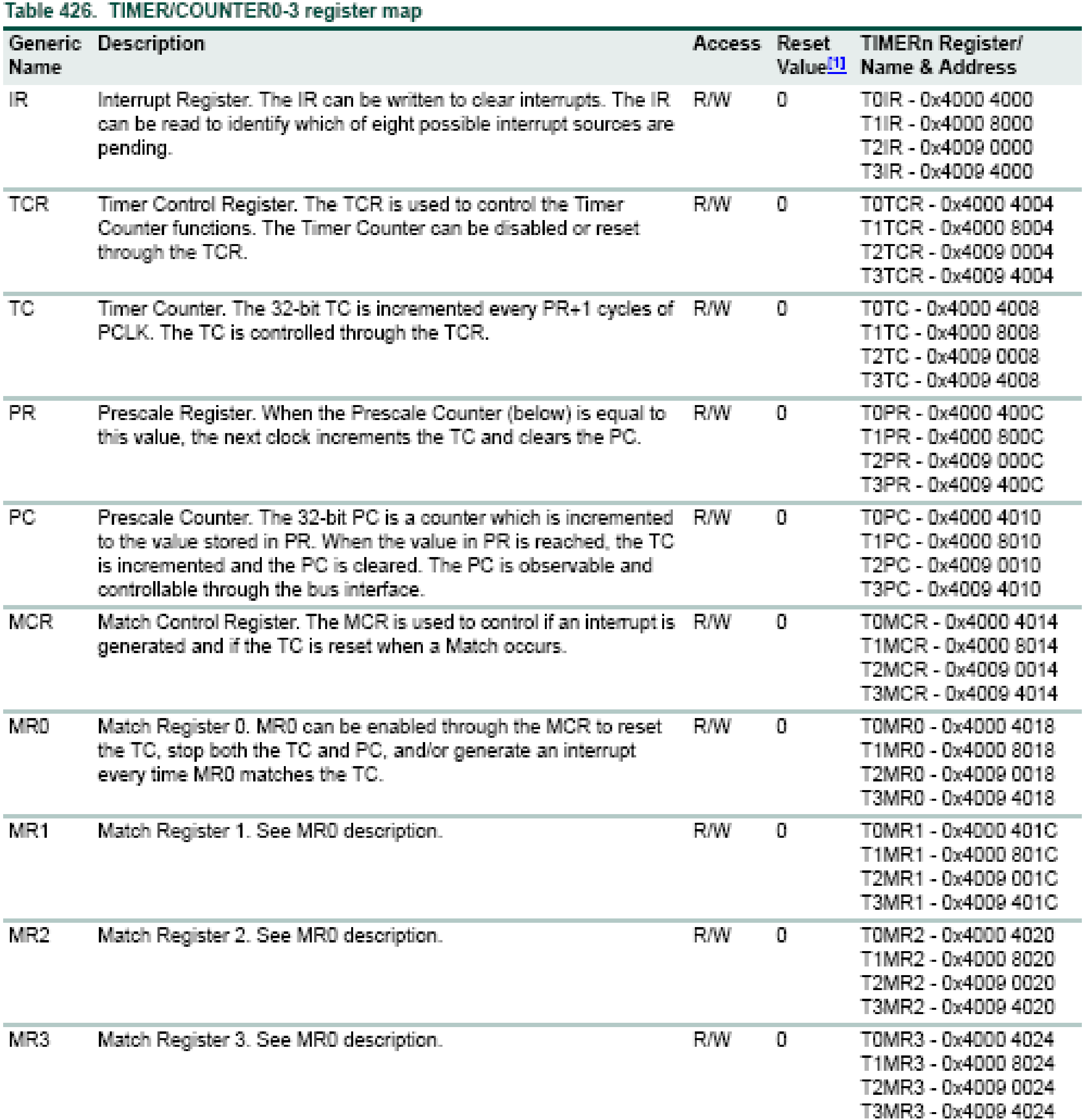
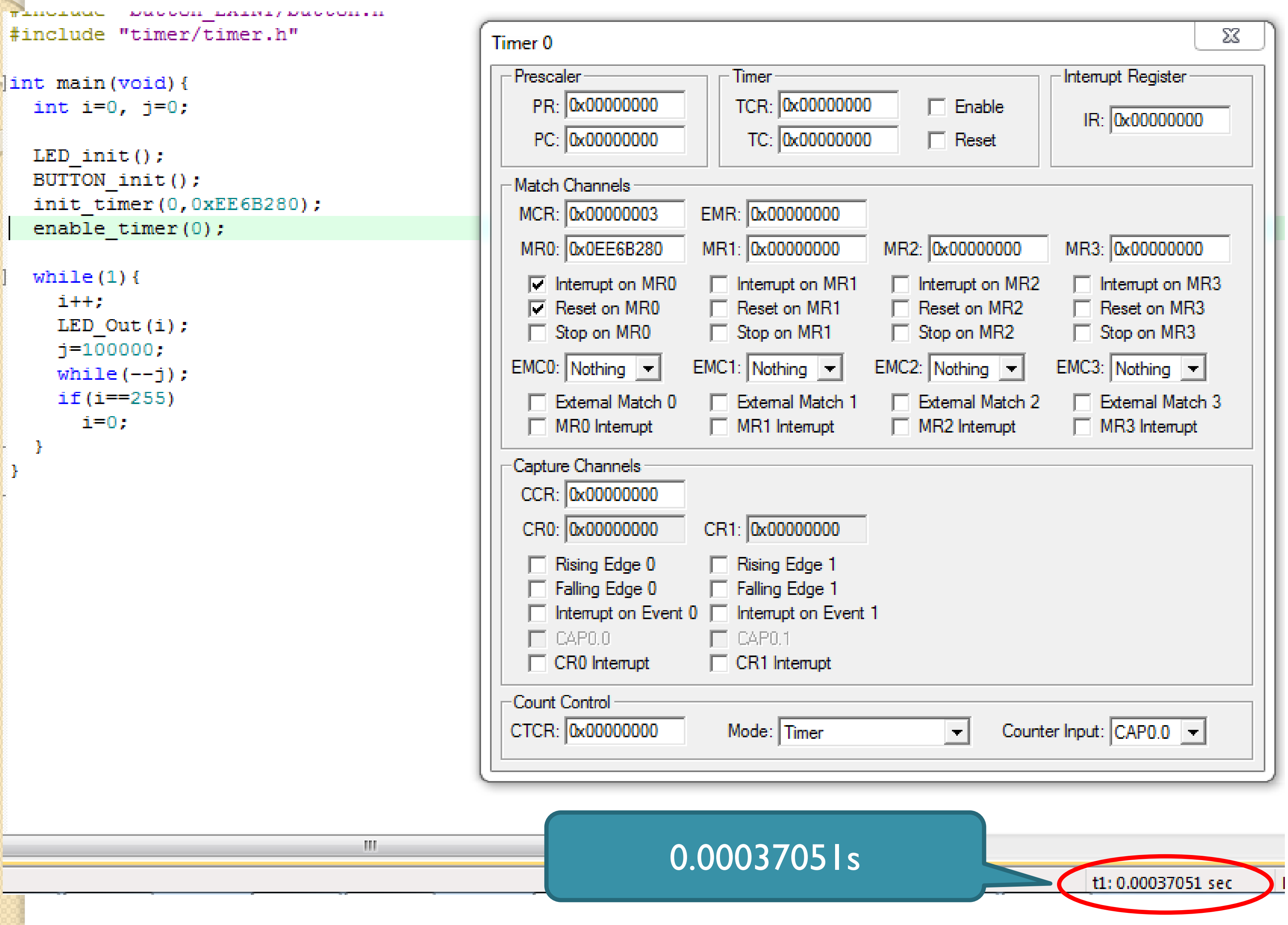


Table 426. TIMER/COUNTER0-3 register map

Generic Name	Description	Access	Reset Value[1]	TIMERn Register/ Name & Address
IR	Interrupt Register. The IR can be written to clear interrupts. The IR can be read to identify which of eight possible interrupt sources are pending.	R/W	0	T0IR - 0x4000 4000 T1IR - 0x4000 8000 T2IR - 0x4009 0000 T3IR - 0x4009 4000
TCR	Timer Control Register. The TCR is used to control the Timer Counter functions. The Timer Counter can be disabled or reset through the TCR.	R/W	0	T0TCR - 0x4000 4004 T1TCR - 0x4000 8004 T2TCR - 0x4009 0004 T3TCR - 0x4009 4004
TC	Timer Counter. The 32-bit TC is incremented every PR+1 cycles of PCLK. The TC is controlled through the TCR.	R/W	0	T0TC - 0x4000 4008 T1TC - 0x4000 8008 T2TC - 0x4009 0008 T3TC - 0x4009 4008
PR	Prescale Register. When the Prescale Counter (below) is equal to this value, the next clock increments the TC and clears the PC.	R/W	0	T0PR - 0x4000 400C T1PR - 0x4000 800C T2PR - 0x4009 000C T3PR - 0x4009 400C
PC	Prescale Counter. The 32-bit PC is a counter which is incremented to the value stored in PR. When the value in PR is reached, the TC is incremented and the PC is cleared. The PC is observable and controllable through the bus interface.	R/W	0	T0PC - 0x4000 4010 T1PC - 0x4000 8010 T2PC - 0x4009 0010 T3PC - 0x4009 4010
MCR	Match Control Register. The MCR is used to control if an interrupt is generated and if the TC is reset when a Match occurs.	R/W	0	T0MCR - 0x4000 4014 T1MCR - 0x4000 8014 T2MCR - 0x4009 0014 T3MCR - 0x4009 4014
MR0	Match Register 0. MR0 can be enabled through the MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt every time MR0 matches the TC.	R/W	0	T0MR0 - 0x4000 4018 T1MR0 - 0x4000 8018 T2MR0 - 0x4009 0018 T3MR0 - 0x4009 4018
MR1	Match Register 1. See MR0 description.	R/W	0	T0MR1 - 0x4000 401C T1MR1 - 0x4000 801C T2MR1 - 0x4009 001C T3MR1 - 0x4009 401C
MR2	Match Register 2. See MR0 description.	R/W	0	T0MR2 - 0x4000 4020 T1MR2 - 0x4000 8020 T2MR2 - 0x4009 0020 T3MR2 - 0x4009 4020
MR3	Match Register 3. See MR0 description.	R/W	0	T0MR3 - 0x4000 4024 T1MR3 - 0x4000 8024 T2MR3 - 0x4009 0024 T3MR3 - 0x4009 4024

Timing checks (I)



```
#include "button_EXINT/button.h"
#include "timer/timer.h"

int main(void){
  int i=0, j=0;

  LED_init();
  BUTTON_init();
  init_timer(0,0xEE6B280);
  enable_timer(0);

  while(1){
    i++;
    LED_Out(i);
    j=100000;
    while(--j);
    if(i==255)
      i=0;
  }
}
```

Timer 0

Prescaler
PR: 0x00000000
PC: 0x00000000

Timer
TCR: 0x00000000 □ Enable
TC: 0x00000000 □ Reset

Interrupt Register
IR: 0x00000000

Match Channels
MCR: 0x00000003 EMR: 0x00000000
MR0: 0x0EE6B280 MR1: 0x00000000 MR2: 0x00000000 MR3: 0x00000000

☑ Interrupt on MR0 □ Interrupt on MR1 □ Interrupt on MR2 □ Interrupt on MR3
☑ Reset on MR0 □ Reset on MR1 □ Reset on MR2 □ Reset on MR3
□ Stop on MR0 □ Stop on MR1 □ Stop on MR2 □ Stop on MR3

EMC0: Nothing EMC1: Nothing EMC2: Nothing EMC3: Nothing

□ External Match 0 □ External Match 1 □ External Match 2 □ External Match 3
□ MR0 Interrupt □ MR1 Interrupt □ MR2 Interrupt □ MR3 Interrupt

Capture Channels
CCR: 0x00000000
CR0: 0x00000000 CR1: 0x00000000

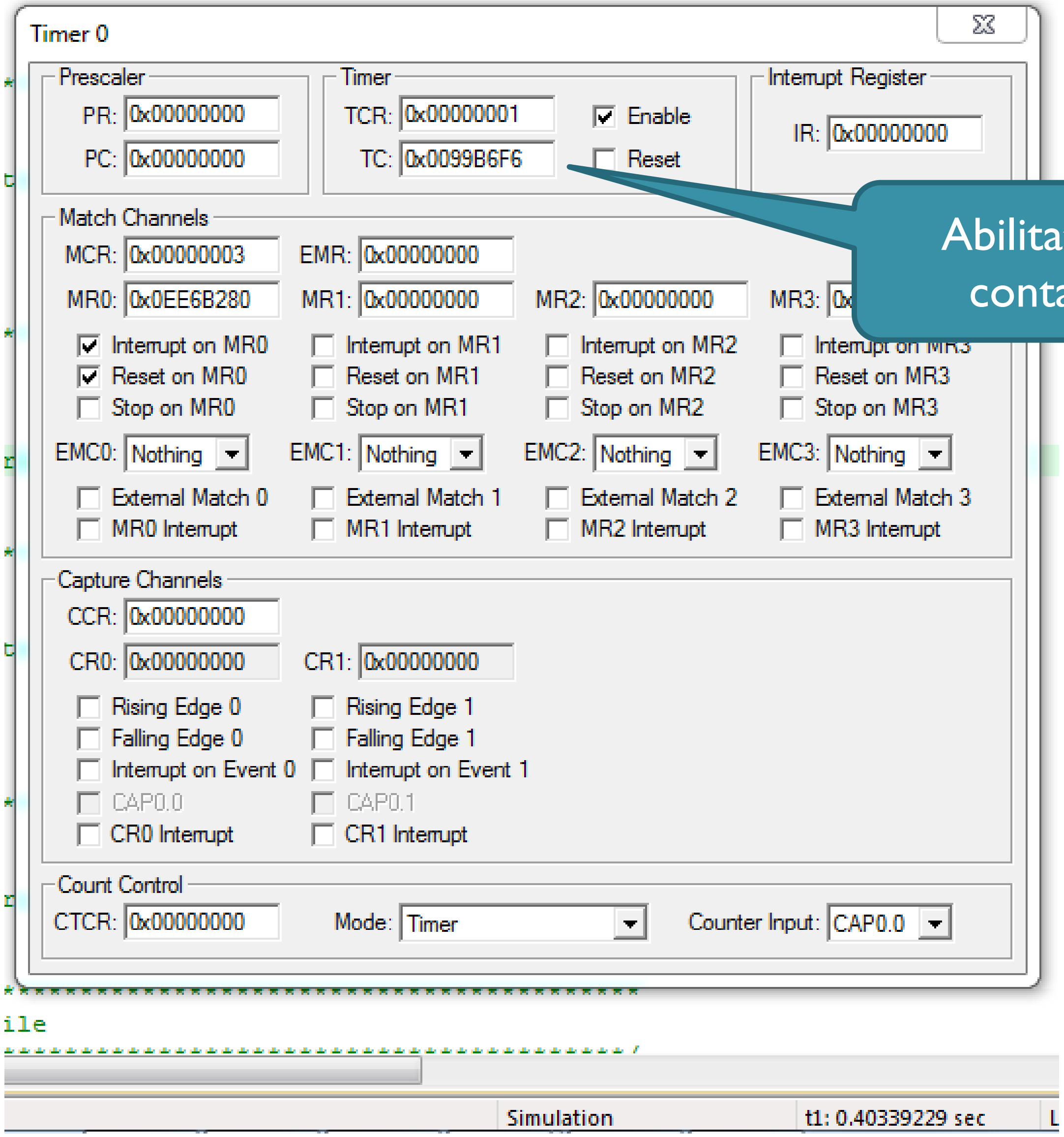
□ Rising Edge 0 □ Rising Edge 1
□ Falling Edge 0 □ Falling Edge 1
□ Interrupt on Event 0 □ Interrupt on Event 1
□ CAP0.0 □ CAP0.1
□ CR0 Interrupt □ CR1 Interrupt

Count Control
CTCR: 0x00000000 Mode: Timer Counter Input: CAP0.0

t1: 0.00037051 sec

0.00037051 s

Timing check (II)



Timer 0

Prescaler
PR: 0x00000000
PC: 0x00000000

Timer
TCR: 0x00000001 ☑ Enable
TC: 0x0099B6F6 □ Reset

Interrupt Register
IR: 0x00000000

Match Channels
MCR: 0x00000003 EMR: 0x00000000
MR0: 0x0EE6B280 MR1: 0x00000000 MR2: 0x00000000 MR3: 0x

☑ Interrupt on MR0 □ Interrupt on MR1 □ Interrupt on MR2 □ Interrupt on MR3
☑ Reset on MR0 □ Reset on MR1 □ Reset on MR2 □ Reset on MR3
□ Stop on MR0 □ Stop on MR1 □ Stop on MR2 □ Stop on MR3

EMC0: Nothing EMC1: Nothing EMC2: Nothing EMC3: Nothing

□ External Match 0 □ External Match 1 □ External Match 2 □ External Match 3
□ MR0 Interrupt □ MR1 Interrupt □ MR2 Interrupt □ MR3 Interrupt

Capture Channels
CCR: 0x00000000
CR0: 0x00000000 CR1: 0x00000000

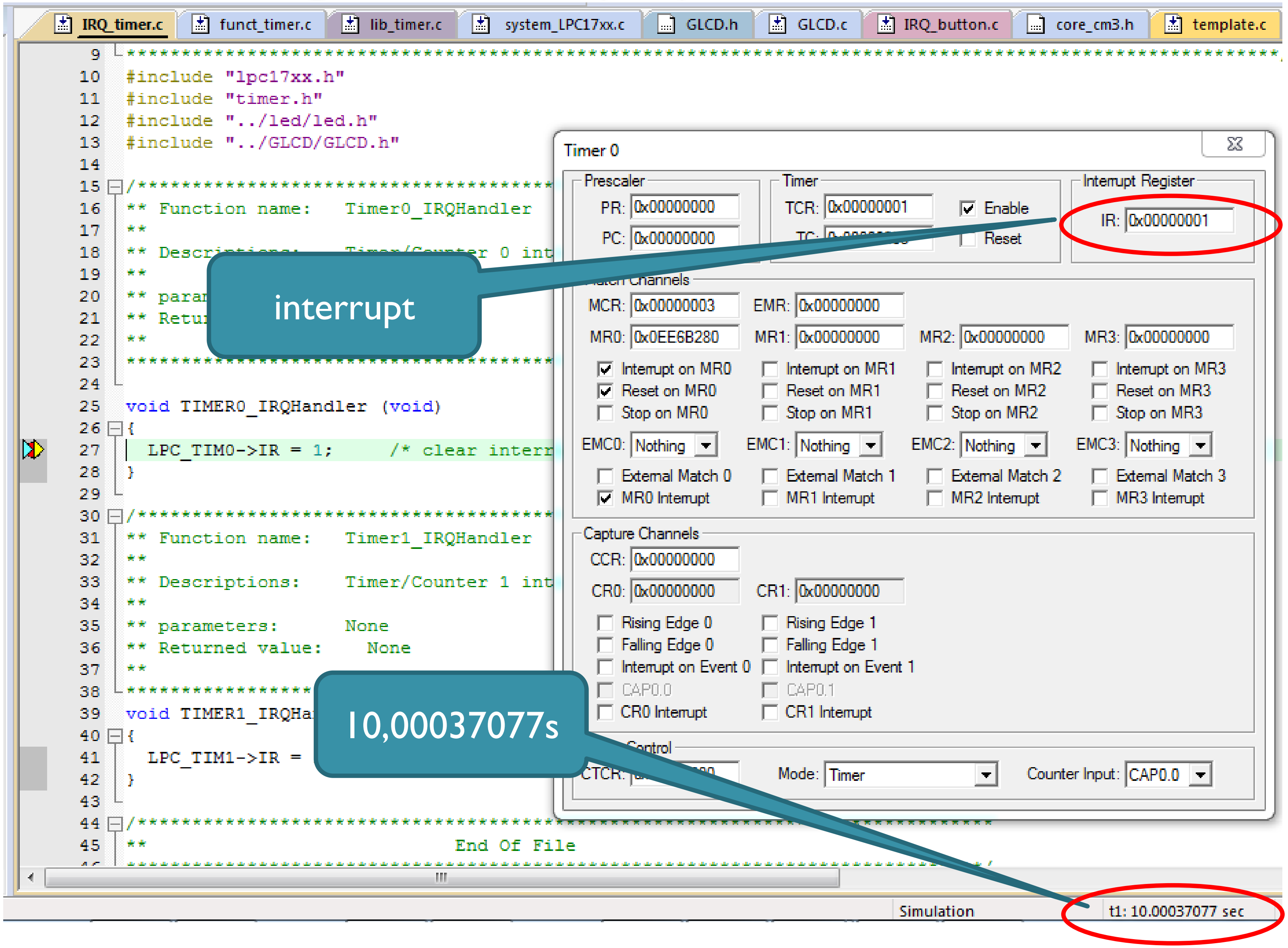
□ Rising Edge 0 □ Rising Edge 1
□ Falling Edge 0 □ Falling Edge 1
□ Interrupt on Event 0 □ Interrupt on Event 1
□ CAP0.0 □ CAP0.1
□ CR0 Interrupt □ CR1 Interrupt

Count Control
CTCR: 0x00000000 Mode: Timer Counter Input: CAP0.0

Simulation t1: 0.40339229 sec

Abilitato, sta contando

Timing check(III)



```
*********************************************************************************************
#include "lpc17xx.h"
#include "timer.h"
#include "../led/led.h"
#include "../GLCD/GLCD.h"

/***********************************
** Function name:    Timer0_IRQHandler
**
** Descriptions:     Timer/Counter 0 int
**
** param
** Retu
**
***********************************

void TIMER0_IRQHandler (void)
{
  LPC_TIM0->IR = 1;       /* clear interr
}

/***********************************
** Function name:    Timer1_IRQHandler
**
** Descriptions:     Timer/Counter 1 int
**
** parameters:       None
** Returned value:    None
**
****************
void TIMER1_IRQHa
{
  LPC_TIM1->IR =
}

/*****************************************************************
**                              End Of File
```

Timer 0

Prescaler
PR: 0x00000000
PC: 0x00000000

Timer
TCR: 0x00000001 ☑ Enable
TC: 0x000000 ☐ Reset

Interrupt Register
IR: 0x00000001

Match Channels
MCR: 0x00000003 EMR: 0x00000000
MR0: 0x0EE6B280 MR1: 0x00000000 MR2: 0x00000000 MR3: 0x00000000

☑ Interrupt on MR0 ☐ Interrupt on MR1 ☐ Interrupt on MR2 ☐ Interrupt on MR3
☑ Reset on MR0 ☐ Reset on MR1 ☐ Reset on MR2 ☐ Reset on MR3
☐ Stop on MR0 ☐ Stop on MR1 ☐ Stop on MR2 ☐ Stop on MR3
EMC0: Nothing EMC1: Nothing EMC2: Nothing EMC3: Nothing
☐ External Match 0 ☐ External Match 1 ☐ External Match 2 ☐ External Match 3
☑ MR0 Interrupt ☐ MR1 Interrupt ☐ MR2 Interrupt ☐ MR3 Interrupt

Capture Channels
CCR: 0x00000000
CR0: 0x00000000 CR1: 0x00000000
☐ Rising Edge 0 ☐ Rising Edge 1
☐ Falling Edge 0 ☐ Falling Edge 1
☐ Interrupt on Event 0 ☐ Interrupt on Event 1
☐ CAP0.0 ☐ CAP0.1
☐ CR0 Interrupt ☐ CR1 Interrupt

Control
CTCR: 0x000000 Mode: Timer Counter Input: CAP0.0

Simulation t1: 10.00037077 sec

interrupt

10,00037077s