



**LPC-P2129**

Get Started Guide

Revision 1.0 28/03/2005



## LPC-P2129 Board

### Introduction

The **LPC2129** are based on a 16/32 bit ARM7TDMI-S CPU with real-time emulation and embedded trace support, together with 128/256 kilobytes (kB) of embedded high speed flash memory. A 128-bit wide memory interface and a unique accelerator architecture enable 32-bit code execution at maximum clock rate. For critical code size applications, the alternative 16-bit Thumb Mode reduces code by more than 30% with minimal performance penalty.

With their compact 64 pin package, low power consumption, various 32-bit timers, 4-channel 10-bit ADC, 2 advanced CAN channels, PWM channels and 46 GPIO lines with up to 9 external interrupt pins these microcontrollers are particularly suitable for automotive and industrial control applications as well as medical systems and fault-tolerant maintenance buses. With a wide range of additional serial communications interfaces, they are also suited for communication gateways and protocol converters as well as many other general-purpose applications.

The **LPC-P2129** Development board is designed to evaluate LPC2129 processor. It has the following features:

- standard JTAG connector with ARM 2x10 pin layout for programming/debugging with ARM-JTAG
- three on board voltage regulators 1.8V, 3.3V and 5V with up to 800mA current
- single power supply: +7-9VDC required
- power supply status LED
- power supply filtering capacitor
- two channel RS232 interface with two RS232 SUB-D 9 pin connectors
- two channel CAN interface with drivers and two SUB-D 9 pin connectors
- iButton interface circuit
- Frequency input connector
- two buttons
- two status LEDs
- Potentiometer connected to AIN
- RESET circuit with external control of Philips ISP utility via RS232
- RESET button
- DBG jumper for JTAG enable
- BSL jumper for bootloader enable
- JRST jumper for enable/disable external RESET control by RS232
- HF crystal
- extension headers for all uC ports
- PCB: FR-4, 1.5 mm (0,062"), green soldermask, white silkscreen component print
- Dimensions: 115x80 mm (4.5x3.2")

The purpose of this guide is to describe LPC-P2138 Development board.

**Board** LPC-P2129

**Hardware details** Describes the hardware peripherals in detail

**Programming** describes how to write programs for the P2129 Board.

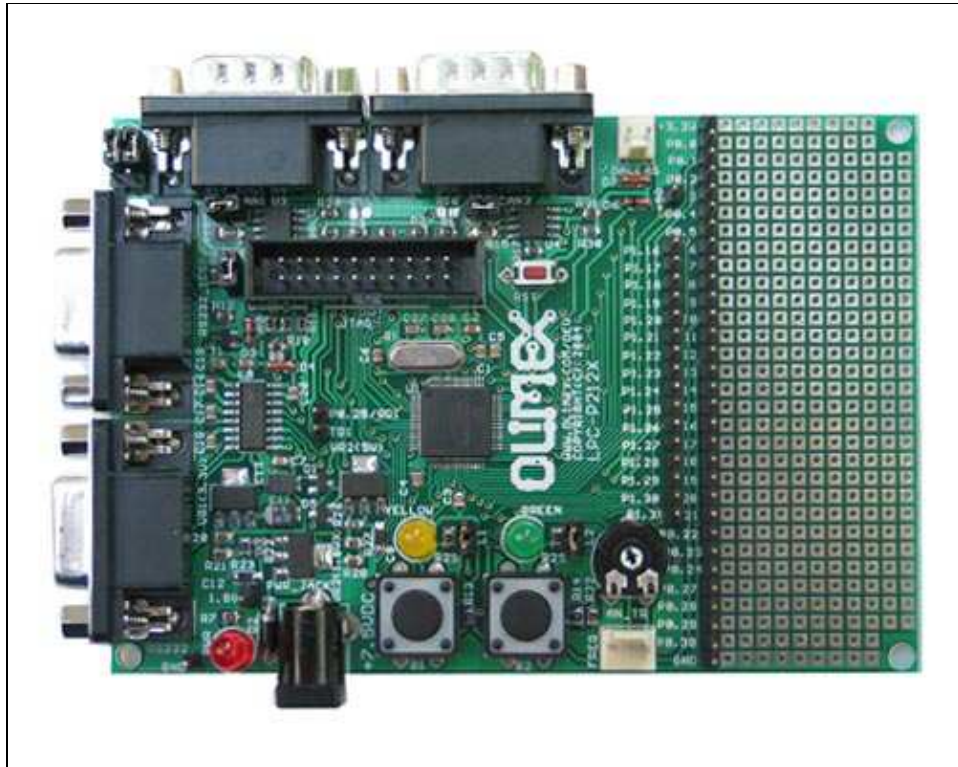
**Revision** 01.01.2005 Creating



## LPC-P2129 Board

### Picture

This is picture of LPC-P2129 Development board.

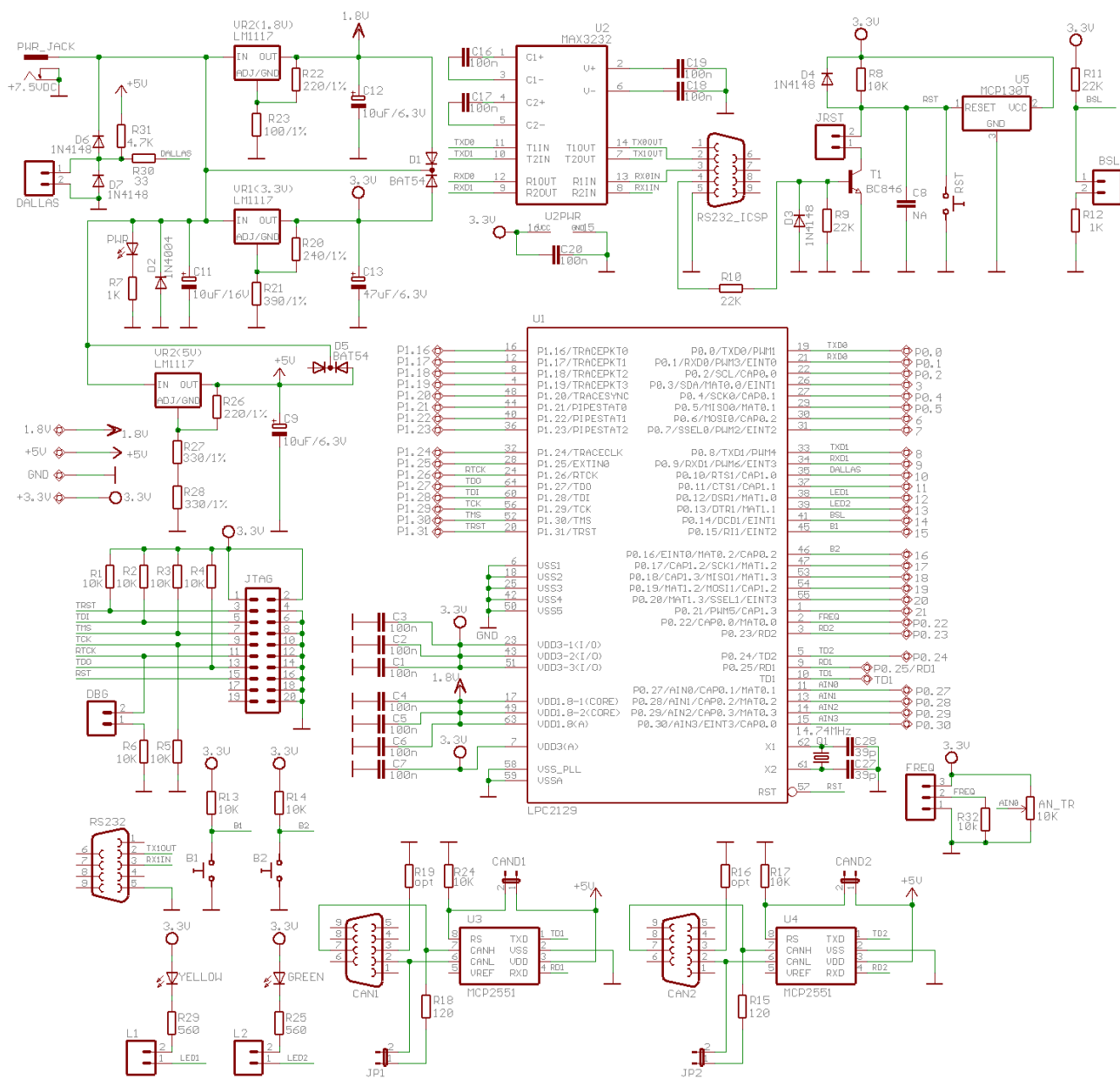


**Board** LPC-P2129



LPC-P2129 Board

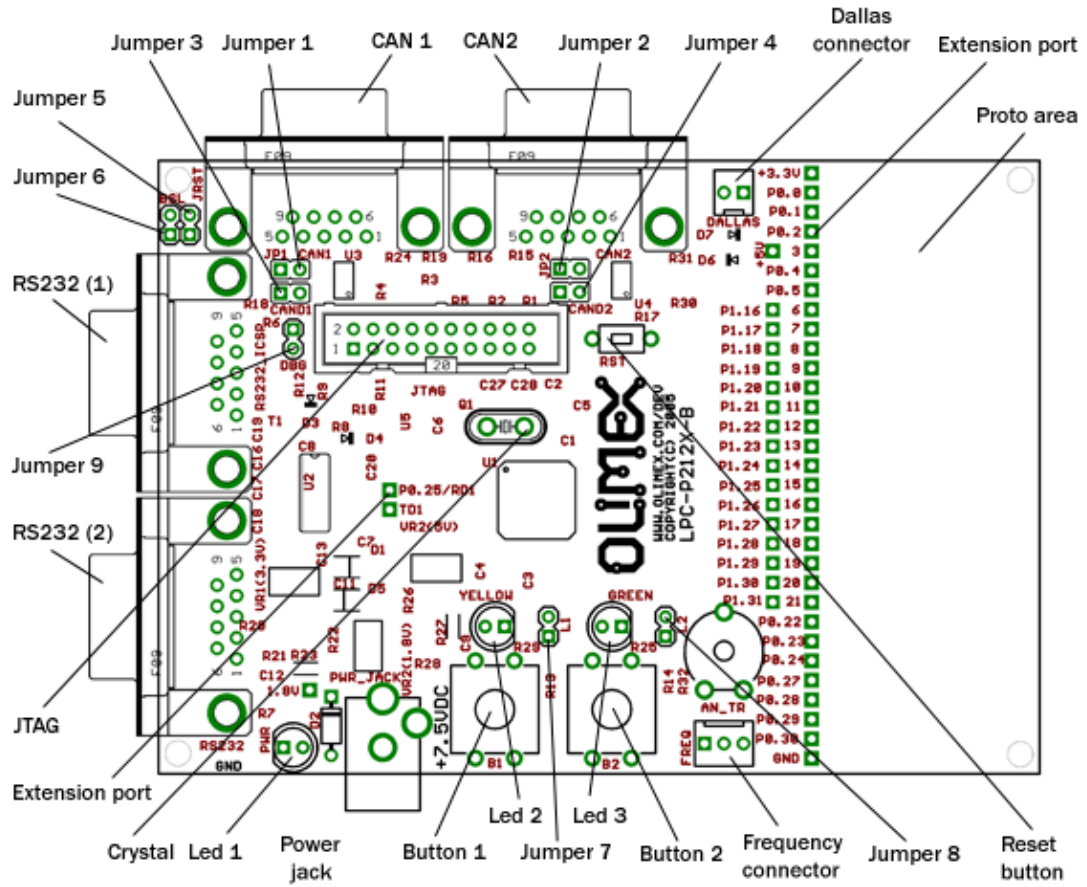
P2129





# LPC-P2129 Board

## P2129 Board





## LPC-P2129 Board

### LPC P2129 Hardware description

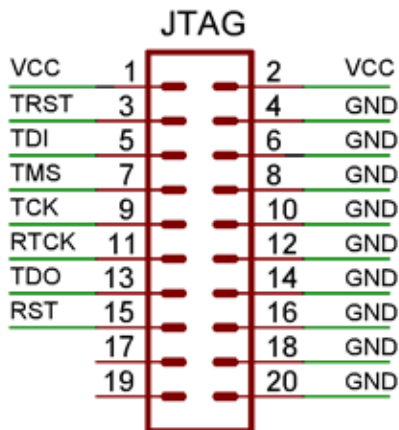
#### Peripherals

Unit	Description
COM Port1	RS232 DB9 Female connector for LPC2129 UART0.
COM Port2	RS232 DB9 Female connector for LPC2129 UART1.
CAN Connector 1	RS232 DB9 Male connector realizing CAN interface.
CAN Connector 2	RS232 DB9 Male connector realizing CAN interface.
JTAG Connector	2x10 0,1" step connector for programming with ARM-JTAG.
Frequency connector	Three pins frequency input connector
Dallas Connector	Interface to Dallas device connected to P0.10/RTS1/CAP1.0 (PIN 35).
Buttons	Two buttons connected to interrupt ports Button 1 - P0.15 / RI1 / EINT2 (PIN 45) Button 2 - P0.16 / EINT0 / MAT0.2 / CAP0.2 (PIN 46)
Leds	Yellow status led connected to P0.12 (PIN 38), green status led connected to P0.13 (PIN 39) and power supply indicator.

#### Technical characteristics

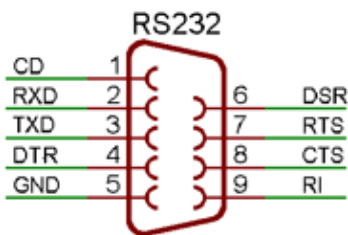
Parameter	Description
Voltage Supply	min 7.5V DC, max 9.0V DC
CPU	LPC2129
Crystals	Crystal 1 - Q1 - 14,745 MHz HF crystal
Board dimensions	115x80 mm (4.5x3.2")
PCB	FR-4, 1.5 mm (0,062"), green soldermask, white silkscreen component print
Operating Temperature	form 0°C to 70°C

#### JTAG Connector



Pin / Name	Connected to:	Functionality
1 - VCC	VCC	-
2 - VCC	VCC	-
3 - TRST	PIN 20	P1.31 / TRST
4 - GND	GROUND	-
5 - TDI	PIN 60	P1.28 / TDI
6 - GND	GROUND	-
7 - TMS	PIN 52	P1.30 / TMS
8 - GND	GROUND	-
9 - TCK	PIN 56	P1.29 / TCK
10 - GND	GROUND	-
11 - RTCK	PIN 24	P1.26 / RTCK
12 - GND	GROUND	-
13 - TDO	PIN 64	P1.27 / TDO
14 - GND	GROUND	-
15 - RST	PIN 57	RST
16 - GND	GROUND	-
17 - -	no connected	-
18 - GND	GROUND	-
19 - -	no connected	-
20 - GND	GROUND	-

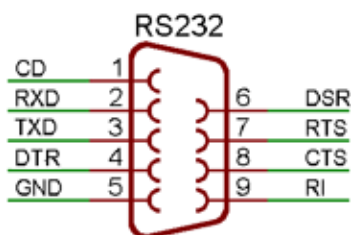
#### RS232 Connector 1



Pin / Name	Connected to:	Functionality
1 - CD	not connected	-
2 - RXD	PIN 19	P0.0 / TXD0 / PWM1
3 - TXD	PIN 21	P0.1 / RXD0 / PWM3 / EINT0
4 - DTR	SLIDE-SWITGH	-
5 - GND	GROUND	-
6 - DSR	not connected	-
7 - RTS	not connected	-
8 - CTS	not connected	-
9 - RI	not connected	-

Max communication baud rate supported by LPC2138F is 448 kbps, maximum baud rate with MAX3232 driver on board is 250 kbps.

#### RS232 Connector 2

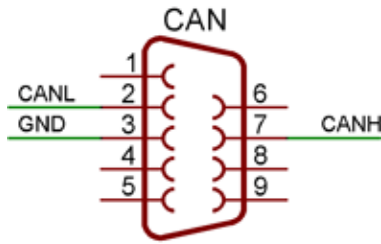


Pin / Name	Connected to:	Functionality
1 - CD	not connected	-
2 - RXD	PIN 33	P0.8 / TXD1 / PWM4 / AD1.1
3 - TXD	PIN 34	P0.9 / RXD1 / PWM6 / EINT3
4 - RTS	not connected	-
5 - GND	GROUND	-
6 - DSR	not connected	-
7 - RTS	not connected	-
8 - CTS	not connected	-
9 - RI	not connected	-

Max communication baud rate supported by LPC2138F is 448 kbps, maximum baud rate with MAX3232 driver on board is 250 kbps.

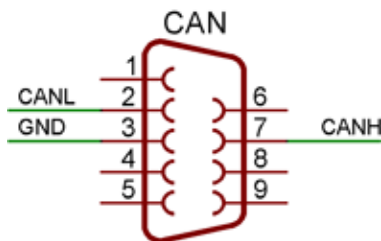


## CAN Connector 1



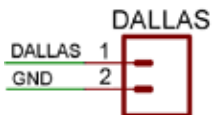
Pin / Name	Connected to:	Functionality
1 -	not connected	-
2 - CANL	MCP2551 PIN 6	CANL
3 - GND	GROUND	-
4 -	not connected	-
5 -	GROUND	-
6 -	not connected	-
7 - CANH	MCP2551 PIN 7	CANH
8 -	not connected	-
9 -	not connected	-

## CAN Connector 2



Pin / Name	Connected to:	Functionality
1 -	not connected	-
2 - CANL	MCP2551 PIN 6	CANL
3 - GND	GROUND	-
4 -	not connected	-
5 -	GROUND	-
6 -	not connected	-
7 - CANH	MCP2551 PIN 7	CANH
8 -	not connected	-
9 -	not connected	-

## Dallas Connector



















Pin / Name	Connected to:	Functionality
1 - DALLAS	PIN 35	P0.10 / RTS1 / CAP1.0 / AD1.2
2 - GND	GROUND	-

## Frequency Connector



Pin / Name	Connected to:	Functionality
1 - GND	+3.3V	-
2 - FREQ	PIN 2	P0.22 / CAP0.0 / MAT0.0
3 - VCC	GROUND	-

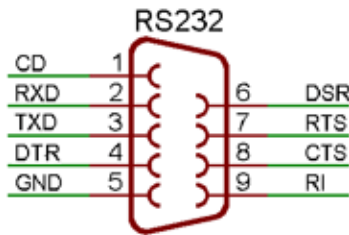
## Jumpers

Jumpers	Position	Description
Jumper 1 (J1)		CAN1 Terminator disable
		CAN1 Terminator enable (120 ohm)
Jumper 2 (J2)		CAN2 Terminator disable
		CAN2 Terminator enable (120 ohm)
Jumper 3 (CAND1)		Enable CAN1 driver
		CAN1 driver is in standby mode (don't use)
Jumper 4 (CAND2)		Enable CAN2 driver
		CAN2 driver is in standby mode (don't use)
Jumper 5 (JRST) Jumper 6 (BSL)		Disable ICSP programming.
		
Jumper 7 (L1)		Yellow led is not connected.
		Yellow led connected to P0.12 / DSR1 / MAT1.0 (PIN 38)
Jumper 8 (L2)		Green led is not connected.
		Green led connected to P0.13 / DTR1 / MAT1.1 (PIN 39)
Jumper 9 (DBG)		Disable JTAG programming.
		Enable JTAG programming.



**Programming: RS232**

**RS232 Connector**



Pin / Name	Description
1 - CD	Carrier Detected.
2 - RXD	Received Data.
3 - TXD	Transmitted Data.
4 - DTR	Data Terminal Ready.
5 - GND	Signal Ground.
6 - DSR	Data Set Ready.
7 - RTS	Request to Send.
8 - CTS	Clear to Send..
9 - RI	Ring Indicator.

**Register description**

Register	Address	Function
U0RBR	0xE000C000 DLAB = 0	Receiver Buffer Register. Input data buffer.
U0THR	0xE000C000 DLAB = 0	Transmit Holding Register. Output data buffer.
U0DLL	0xE000C000 DLAB = 1	Divisor Latch LSB.
U0DLM	0xE000C000 DLAB = 1	Divisor Latch MSB.
U0IER	0xE000C004 DLAB = 0	Interrupt Enable Register.
U0IIR	0xE000C008	Interrupt ID Register.
U0FCR	0xE000C008	FIFO Control Register.
U0LCR	0xE000C00C	Line Control Register.
U0LSR	0xE000C014	Line Status Register.
U0SCR	0xE000C01C	Scratch Pad Register.
U0TER	0xE000C030	Transmit Enable.

**1.Initialization**

**1.1. Set Line Control Register**

U0LCR	Function	Description	Reset Value
1:0	Word Length Select	00: 5 bit character length 01: 6 bit character length 10: 7 bit character length 11: 8 bit character length	0
2	Stop Bit Select	0: 1 stop bit 1: 2 stop bits (1.5 if U0LCR[1:0]=00)	0
3	Parity Enable	0: Disable parity generation and checking 1: Enable parity generation and checking	0
5:4	Parity Select	00: Odd parity 01: Even parity 10: Forced 1 stick parity	0

		11: Forced 0 stick parity	
6	Break Control	0: Disable break transmission 1: Enable break transmission. Output pin UART0 TxD is forced to logic 0 when U0LCR6 is active high.	0
7	Divisor Latch Access Bit	0: Disable access to Divisor Latches 1: Enable access to Divisor Latches	0

### 1.2. UART0 Baudrate Calculation

The U0DLL and U0DLM registers together form a 16 bit divisor where U0DLL contains the lower 8 bits of the divisor and U0DLM contains the higher 8 bits of the divisor.

```
divisor = pclk / (16 * baud);
```

### 1.3. Set Functionality to pins

Set functionality to P0.0 -> TX0 and P0.1 -> RXD0

## 2. RS232 Communication

### 2.1. Write to RS232

Use follow algorithm to send data:

- fill U0THR register with data to write
- wait shift all data
- clear interrupt flag

### 2.2. Read from RS232

Use follow algorithm to receive data:

- wait read all data
- clear interrupt flag
- get data from U0RBR

## 3. Example

*Initialize:*

```
//set Line Control Register (8 bit, 1 stop bit, no parity, enable DLAB)
U0LCR_bit.WLS = 0x3;      //8 bit
U0LCR_bit.SBS = 0x0;     //1 stop bit
U0LCR_bit.PE = 0x0;     //no parity
U0LCR_bit.DLAB = 0x1;   //enable DLAB

//divisor
U0DLL = Pclk / (16 * baud);    //low bite
U0DLM = Pclk / (16 * baud)>>8; //high bite
U0LCR &= ~0x80;

//set functionality to pins: port0.0 -> TX0, port0.1 -> RXD0
PINSEL0_bit.P0_0 = 0x1;
PINSEL0_bit.P0_1 = 0x1;
```

*Read Data:*

```
//when U0LSR_bit.DR is 1 - U0RBR contains valid data
while (U0LSR_bit.DR == 0);
return U0RBR;
```

**Write Data:**

```
//when U0LSR_bit.THRE is 1 - U0THR contains valid data.  
while (U0LSR_bit.THRE == 0);  
U0THR = ch0;
```



## Programming: Real Time Clock

### Register description

Register	Address	Function
ILR	0xE0024000	<b>Interrupt Location.</b> Reading this location indicates the source of an interrupt. Writing a one to the appropriate bit at this location clears the associated interrupt.
CTC	0xE0024004	<b>Clock Tick Counter.</b> Value from the clock divider.
CCR	0xE0024008	<b>Clock Control Register.</b> Controls the function of the clock divider.
CIIR	0xE002400C	<b>Counter Increment Interrupt.</b> Selects which counters will generate an interrupt when they are incremented.
AMR	0xE0024010	<b>Alarm Mask Register.</b> Controls which of the alarm registers are masked. RW
CTIME0	0xE0024014	<b>Consolidated Time Register 0</b>
CTIME1	0xE0024018	<b>Consolidated Time Register 1</b>
CTIME2	0xE002401C	<b>Consolidated Time Register 2</b>

### 1. Initialization

#### 1.1. Turn on the 32KHz external clock

CLKSRC

(bit 4 from CCR Register)

- 0 - Disable 32kHz external clock
- 1 - Enable 32kHz external clock

#### 1.2. Enable Interrupt

CIIR	Function	Description
0	IMSEC	When one, an increment of the Second value generates an interrupt.
1	IMMIN	When one, an increment of the Minute value generates an interrupt.
2	IMHOUR	When one, an increment of the Hour value generates an interrupt.
3	IMDOM	When one, an increment of the Day of Month value generates an interrupt.
4	IMDOW	When one, an increment of the Day of Week value generates an interrupt.
5	IMDOY	When one, an increment of the Day of Year value generates an interrupt.
6	IMMON	When one, an increment of the Month value generates an interrupt.
7	IMYEAR	When one, an increment of the Year value generates an interrupt.

#### 1.3. Start the Real Time Clock

**CLKEN (bit 0 from CCR Register)** Enable/Disable Real Time Clock

- 0 - Disable Real Time Clock
- 1 - Enable Real Time Clock

## **2. Example**

*Initialize:*

```
CCR_bit.CLKEN = 0;      //rtc disable
CCR_bit.CLKSRC = 1;    //set external 32kHz oscillator
CCR_bit.CTCRST = 0;   //disable reset
CCR_bit.CTTEST = 0;   //disable test
AMR           = 0;    //initialize interrupt mask register of RTC
CIIR_bit.IMSEC = 1;  //enable interrupt every seconds
ILR           = 3;   //clear all interrupt of RTC
CCR_bit.CLKEN = 1;    //rtc enable
```



## Programming: Blinking LED

### GPIO Register map

Generic name	Description
IOPIN	<b>GPIO Port Pin value register.</b> The current state of the GPIO configured port pins can always be read from this register, regardless of pin direction and mode. Activity on non-GPIO configured pins will not be reflected in this register.
IOSET	<b>GPIO Port Output set register.</b> This register controls the state of output pins in conjunction with the IOCLR register. Writing ones produces highs at the corresponding port pins. Writing zeroes has no effect.
IODIR	<b>GPIO Port Direction control register.</b> This register individually controls the direction of each port pin.
IOCLR	<b>GPIO Port Output clear register.</b> This register controls the state of output pins. Writing ones produces lows at the corresponding port pins and clears the corresponding bits in the IOSET register. Writing zeroes has no effect.

### Pin Connect Block Register Map

Register name	Description
PINSEL0	PINSEL0 Pin function select register 0 (from P0.0 to P0.15)
PINSEL1	PINSEL1 Pin function select register 1 (from P0.16 to P0.31)
PINSEL2	PINSEL2 Pin function select register 2

### 1. Initialization (general case)

#### 1.1. Set first functionality to port

```
PINSEL0 = 0x00; //set first functionality to port (from P0.0 to P0.15)
```

#### 1.2. Set port which is connected to LED as output

```
IODIR = 0xFF; //set P0.0 to P0.15 port as output
```

### 2. Led blink (general case)

```
IOCLR = 0xFF; // set P0.0 - P0.15 to low  
IOSET = 0xFF; // set P0.0 - P0.15 to high
```

### 3. Example - blink led, which is connected to P0.12

```
//Initialization
```

```
PINSEL0_bit.P0_12 = 0x0; // set first functionality to port  
IODIR_bit.P0_12 = 0x1; // set P0.12 port to output  
IOSET_bit.P0_12 = 0x1; // set P0.12 port to high
```

```
//loop forever
```

```
while(1)  
{  
    Delay(1000); // Simple delay  
    IOSET_bit.P0_12 = 0x1; // set P0.12 port to high  
    Delay(1000); // Simple delay  
    IOCLR_bit.P0_12 = 0x1; // set P0.12 port to high
```



}



## LPC-P2129 Board

### Links

#### 1. Philips web site

LPC2129 product datasheets, application notes, etc info:

<http://www.semiconductors.philips.com/>

#### 2. LPC microcontrollers discussion forum

<http://groups.yahoo.com/group/lpc2000/> - forum for discussions on LPC2000 ARM microcontrollers

<http://groups.yahoo.com/group/arm-olimex/> - forum for discussions on Olimex ARM boards

#### 3. IAR Systems EW-ARM C compiler and debugger

<http://www.iar.com/Products/?name=EWARM>

#### 4. Rowley associates CrossWorks for ARM C compiler and debugger

<http://www.rowley.co.uk>