LPC 2148
Single Chip 16-bit/32-bit Microcontroller

- Reference: NXP Semiconductors Data sheets

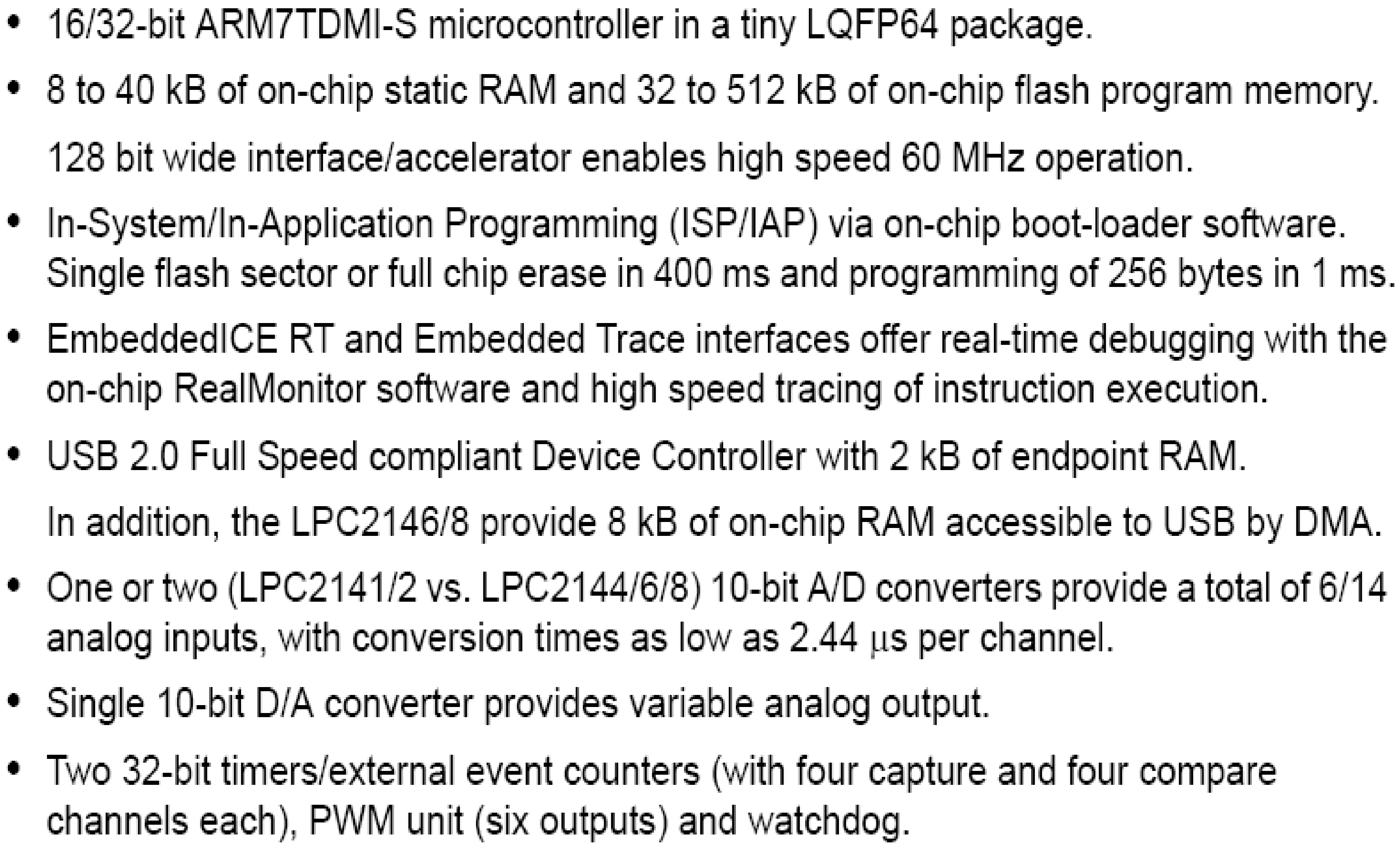
ARM 7 TDMI-S

- T : 16-bit **T**HUMB mode
- D : JTAG **D**ebug (**J**oint **T**est **A**ction **G**roup)
- M : fast **M**ultiplier
- I : enhanced **I**n- **C**ircuit **E**mulation
- S : synthesizable code

Some basics:

- LQPF package: Low-profile Quad Flat Package
- SRAM: Static RAM
- Flash Memory
- Accelerator
- In-System Programming (ISP)
- In-Application Programming (IAP)
- EmbeddedICE RT and Embedded Trace Interface
- USB 2.0 Full speed compliant Device Controller
- Timers/external event counters
- Watchdog Timer
- JTAG: Joint Test Action Group

Features



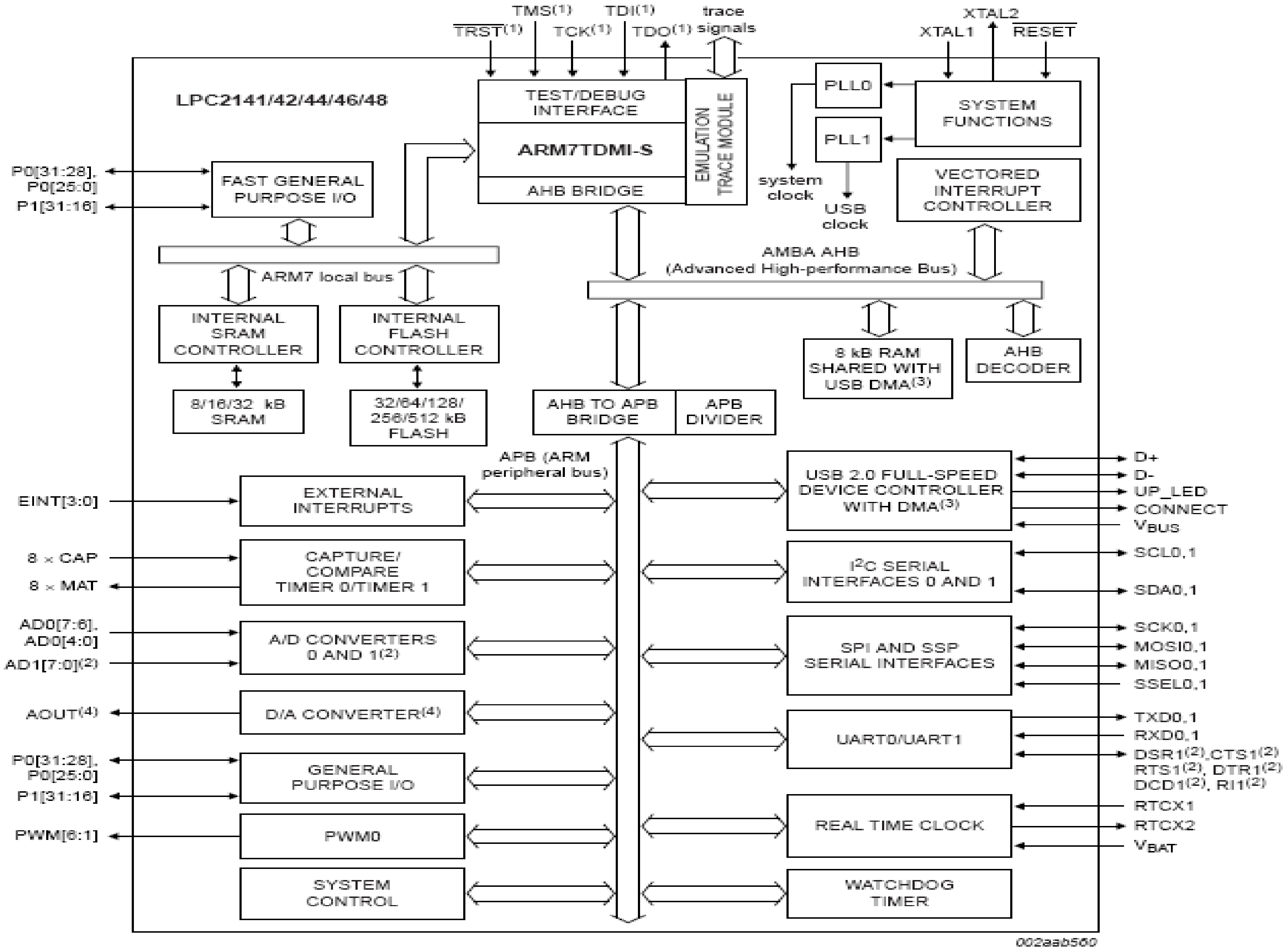
- 16/32-bit ARM7TDMI-S microcontroller in a tiny LQFP64 package.
- 8 to 40 kB of on-chip static RAM and 32 to 512 kB of on-chip flash program memory.

  128 bit wide interface/accelerator enables high speed 60 MHz operation.
- In-System/In-Application Programming (ISP/IAP) via on-chip boot-loader software. Single flash sector or full chip erase in 400 ms and programming of 256 bytes in 1 ms.
- EmbeddedICE RT and Embedded Trace interfaces offer real-time debugging with the on-chip RealMonitor software and high speed tracing of instruction execution.
- USB 2.0 Full Speed compliant Device Controller with 2 kB of endpoint RAM.

  In addition, the LPC2146/8 provide 8 kB of on-chip RAM accessible to USB by DMA.
- One or two (LPC2141/2 vs. LPC2144/6/8) 10-bit A/D converters provide a total of 6/14 analog inputs, with conversion times as low as 2.44 μs per channel.
- Single 10-bit D/A converter provides variable analog output.
- Two 32-bit timers/external event counters (with four capture and four compare channels each), PWM unit (six outputs) and watchdog.

- Low power real-time clock with independent power and dedicated 32 kHz clock input.
- Multiple serial interfaces including two UARTs (16C550), two Fast I^{2}C-bus (400 kbit/s), SPI and SSP with buffering and variable data length capabilities.
- Vectored interrupt controller with configurable priorities and vector addresses.
- Up to 45 of 5 V tolerant fast general purpose I/O pins in a tiny LQFP64 package.
- Up to nine edge or level sensitive external interrupt pins available.

- 60 MHz maximum CPU clock available from programmable on-chip PLL with settling time of 100 μs.
- On-chip integrated oscillator operates with an external crystal in range from 1 MHz to 30 MHz and with an external oscillator up to 50 MHz.
- Power saving modes include Idle and Power-down.
- Individual enable/disable of peripheral functions as well as peripheral clock scaling for additional power optimization.
- Processor wake-up from Power-down mode via external interrupt, USB, Brown-Out Detect (BOD) or Real-Time Clock (RTC).
- Single power supply chip with Power-On Reset (POR) and BOD circuits:
  - CPU operating voltage range of 3.0 V to 3.6 V (3.3 V ± 10 %) with 5 V tolerant I/O pads.



(1) Pins shared with GPIO.

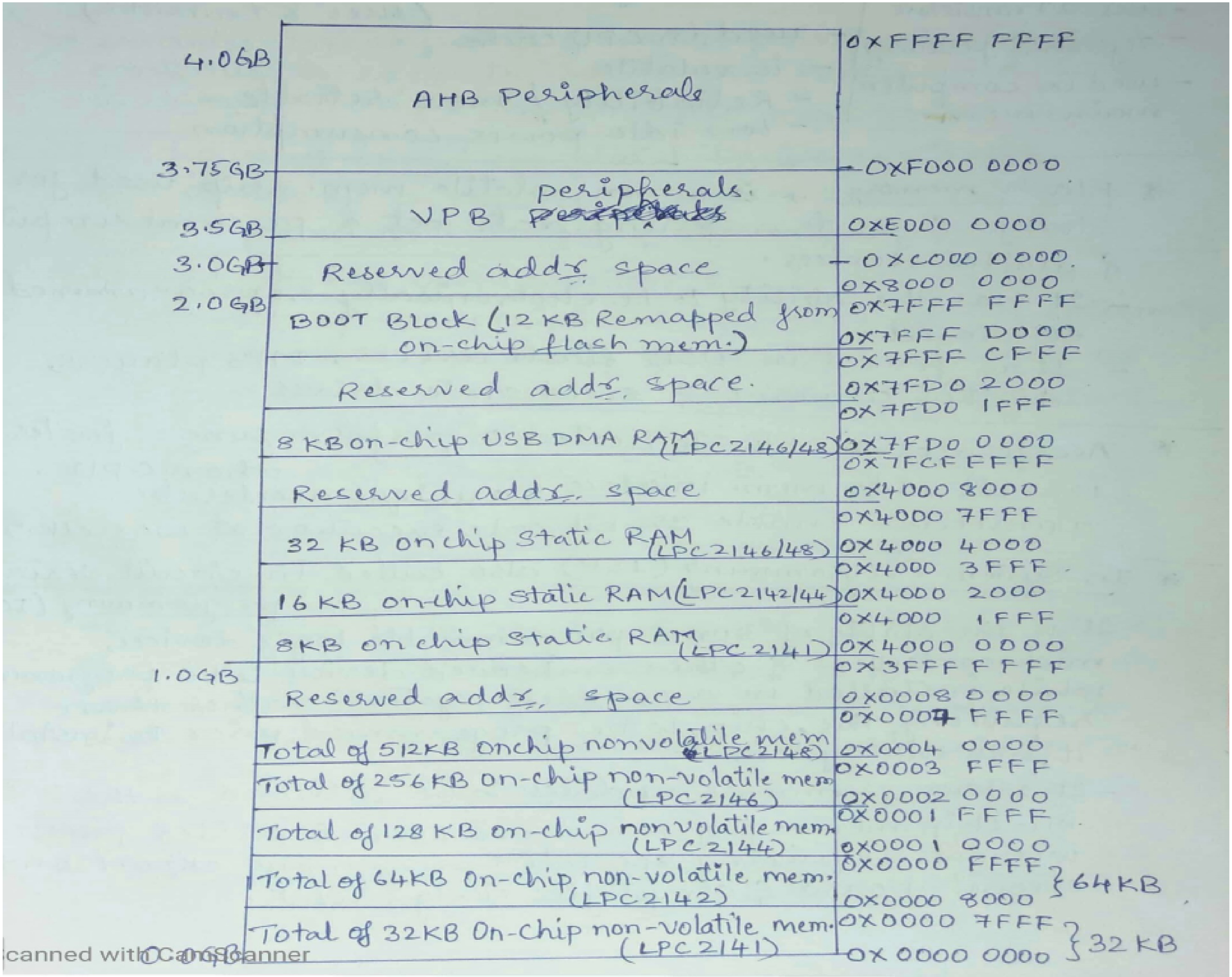
(2) LPCC2144/6/8 only.

(3) USB DMA controller with 8 kB of RAM accessible as general purpose RAM and/or DMA is available in LPC2146/8 only.

(4) LPC2142/4/6/8 only.

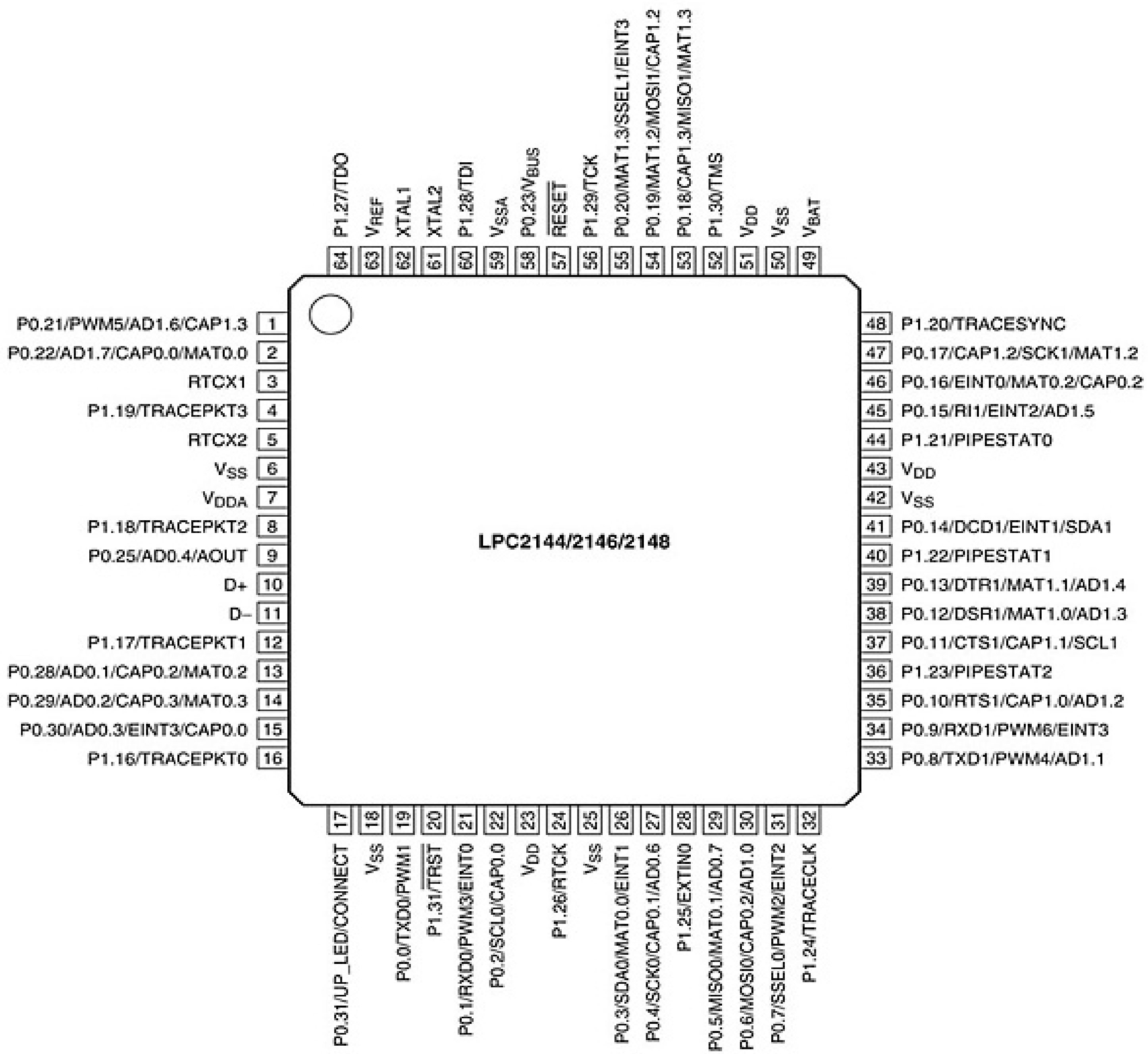
Fig 1. LPC2141/2/4/6/8 block diagram

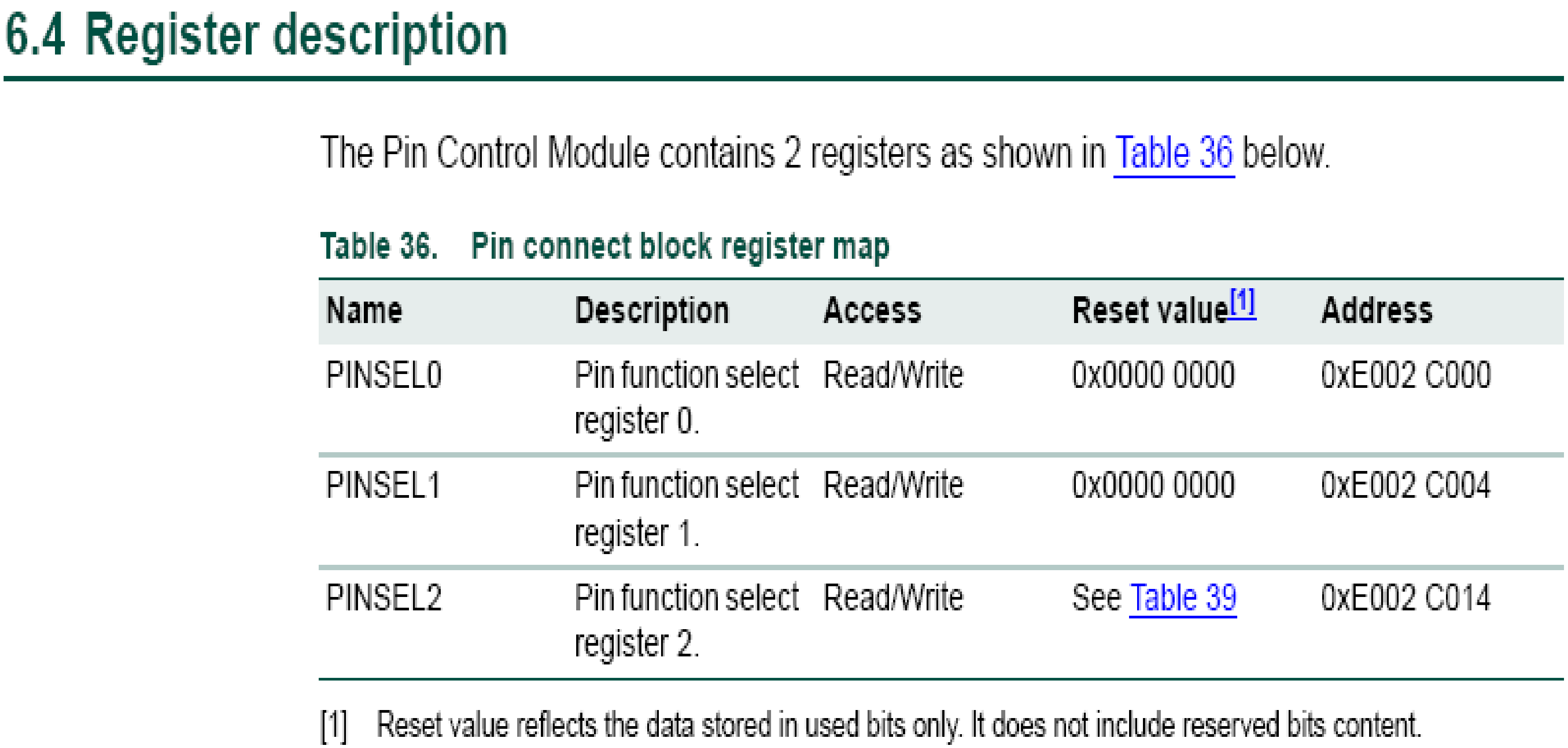
Memory Map



4.0GB	AHB Peripherals	0XFFFF FFFF 0XF000 0000	
3.75GB	VPB ~~peripherals~~ peripherals.	0XE000 0000	
3.5GB 3.0GB	Reserved addr. space	0XC000 0000. 0X8000 0000	
2.0GB	BOOT Block (12 KB Remapped from on-chip flash mem.)	0X7FFF FFFF 0X7FFF D000	
	Reserved addr. space.	0X7FFF CFFF 0X7FD0 2000	
	8 KB on-chip USB DMA RAM (LPC2146/48)	0X7FD0 1FFF 0X7FD0 0000	
	Reserved addr. space	0X7FCF FFFF 0X4000 8000	
	32 KB on chip Static RAM (LPC2146/48)	0X4000 7FFF 0X4000 4000	
	16 KB on-chip static RAM (LPC2142/44)	0X4000 3FFF 0X4000 2000	
	8KB on chip Static RAM (LPC2141)	0X4000 1FFF 0X4000 0000	
1.0GB	Reserved addr. space	0X3FFF FFFF 0X0008 0000	
	Total of 512KB onchip nonvolatile mem (LPC2148)	0X0007 FFFF 0X0004 0000	
	Total of 256KB on-chip non-volatile mem (LPC2146)	0X0003 FFFF 0X0002 0000	
	Total of 128 KB on-chip nonvolatile mem (LPC2144)	0X0001 FFFF 0X0001 0000	
	Total of 64KB On-chip non-volatile mem. (LPC2142)	0X0000 FFFF 0X0000 8000	} 64KB
0.0GB	Total of 32KB On-Chip non-volatile mem. (LPC2141)	0X0000 7FFF 0X0000 0000	} 32KB

Pin diagram


LPC2144/2146/2148
P0.21/PWM5/AD1.6/CAP1.3 1
P0.22/AD1.7/CAP0.0/MAT0.0 2
RTCX1 3
P1.19/TRACEPKT3 4
RTCX2 5
V_{SS} 6
V_{DDA} 7
P1.18/TRACEPKT2 8
P0.25/AD0.4/AOUT 9
D+ 10
D− 11
P1.17/TRACEPKT1 12
P0.28/AD0.1/CAP0.2/MAT0.2 13
P0.29/AD0.2/CAP0.3/MAT0.3 14
P0.30/AD0.3/EINT3/CAP0.0 15
P1.16/TRACEPKT0 16
17 P0.31/UP_LED/CONNECT
18 V_{SS}
19 P0.0/TXD0/PWM1
20 P1.31/$\overline{TRST}$
21 P0.1/RXD0/PWM3/EINT0
22 P0.2/SCL0/CAP0.0
23 V_{DD}
24 P1.26/RTCK
25 V_{SS}
26 P0.3/SDA0/MAT0.0/EINT1
27 P0.4/SCK0/CAP0.1/AD0.6
28 P1.25/EXTIN0
29 P0.5/MISO0/MAT0.1/AD0.7
30 P0.6/MOSI0/CAP0.2/AD1.0
31 P0.7/SSEL0/PWM2/EINT2
32 P1.24/TRACECLK
33 P0.8/TXD1/PWM4/AD1.1
34 P0.9/RXD1/PWM6/EINT3
35 P0.10/RTS1/CAP1.0/AD1.2
36 P1.23/PIPESTAT2
37 P0.11/CTS1/CAP1.1/SCL1
38 P0.12/DSR1/MAT1.0/AD1.3
39 P0.13/DTR1/MAT1.1/AD1.4
40 P1.22/PIPESTAT1
41 P0.14/DCD1/EINT1/SDA1
42 V_{SS}
43 V_{DD}
44 P1.21/PIPESTAT0
45 P0.15/RI1/EINT2/AD1.5
46 P0.16/EINT0/MAT0.2/CAP0.2
47 P0.17/CAP1.2/SCK1/MAT1.2
48 P1.20/TRACESYNC
49 V_{BAT}
50 V_{SS}
51 V_{DD}
52 P1.30/TMS
53 P0.18/CAP1.3/MISO1/MAT1.3
54 P0.19/MAT1.2/MOSI1/CAP1.2
55 P0.20/MAT1.3/SSEL1/EINT3
56 P1.29/TCK
57 $\overline{RESET}$
58 P0.23/V_{BUS}
59 V_{SSA}
60 P1.28/TDI
61 XTAL2
62 XTAL1
63 V_{REF}
64 P1.27/TDO



6.4 Register description

The Pin Control Module contains 2 registers as shown in Table 36 below.

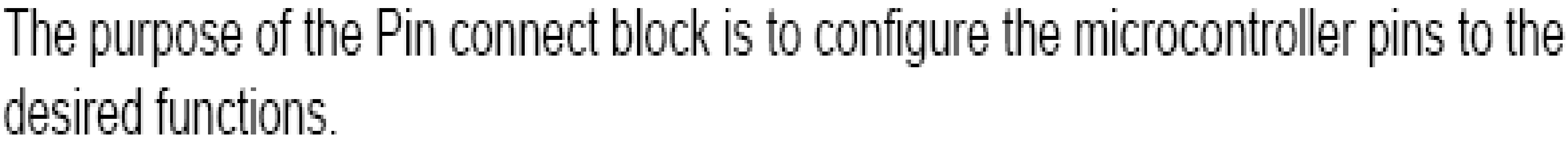
Table 36. Pin connect block register map

Name	Description	Access	Reset value[1]	Address
PINSEL0	Pin function select register 0.	Read/Write	0x0000 0000	0xE002 C000
PINSEL1	Pin function select register 1.	Read/Write	0x0000 0000	0xE002 C004
PINSEL2	Pin function select register 2.	Read/Write	See Table 39	0xE002 C014

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

Ports

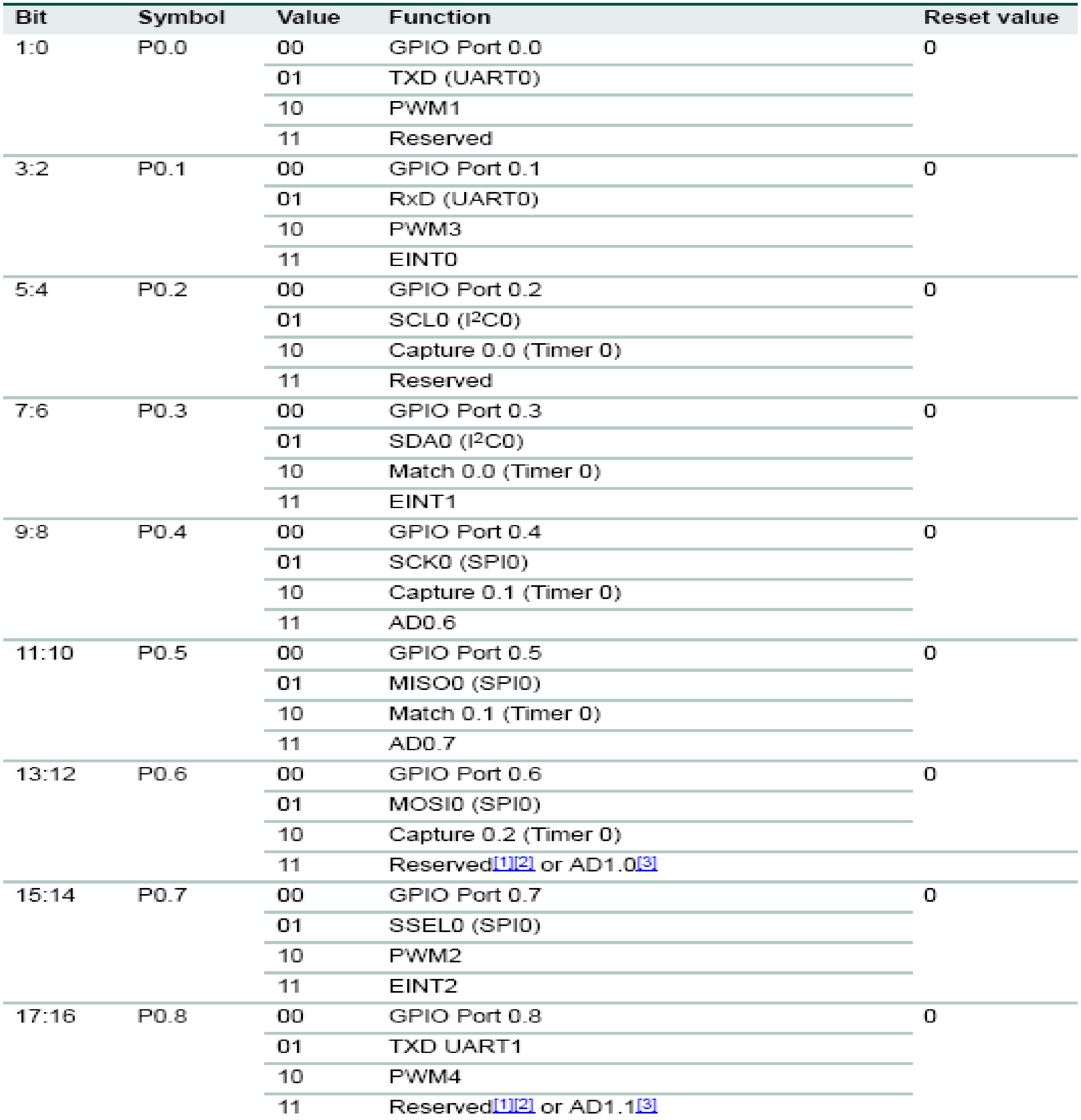
- Port 0: 32-bit size
- Port 1



The purpose of the Pin connect block is to configure the microcontroller pins to the desired functions.

- Pin connect block: all reg.s are 32bit in size
  - PINSEL0
  - PINSEL1
  - PINSEL2

PINSEL0



Bit	Symbol	Value	Function	Reset value
1:0	P0.0	00	GPIO Port 0.0	0
		01	TXD (UART0)	
		10	PWM1	
		11	Reserved	
3:2	P0.1	00	GPIO Port 0.1	0
		01	RxD (UART0)	
		10	PWM3	
		11	EINT0	
5:4	P0.2	00	GPIO Port 0.2	0
		01	SCL0 (I^2C0)	
		10	Capture 0.0 (Timer 0)	
		11	Reserved	
7:6	P0.3	00	GPIO Port 0.3	0
		01	SDA0 (I^2C0)	
		10	Match 0.0 (Timer 0)	
		11	EINT1	
9:8	P0.4	00	GPIO Port 0.4	0
		01	SCK0 (SPI0)	
		10	Capture 0.1 (Timer 0)	
		11	AD0.6	
11:10	P0.5	00	GPIO Port 0.5	0
		01	MISO0 (SPI0)	
		10	Match 0.1 (Timer 0)	
		11	AD0.7	
13:12	P0.6	00	GPIO Port 0.6	0
		01	MOSI0 (SPI0)	
		10	Capture 0.2 (Timer 0)	
		11	Reserved[1][2] or AD1.0[3]	
15:14	P0.7	00	GPIO Port 0.7	0
		01	SSEL0 (SPI0)	
		10	PWM2	
		11	EINT2	
17:16	P0.8	00	GPIO Port 0.8	0
		01	TXD UART1	
		10	PWM4	
		11	Reserved[1][2] or AD1.1[3]	

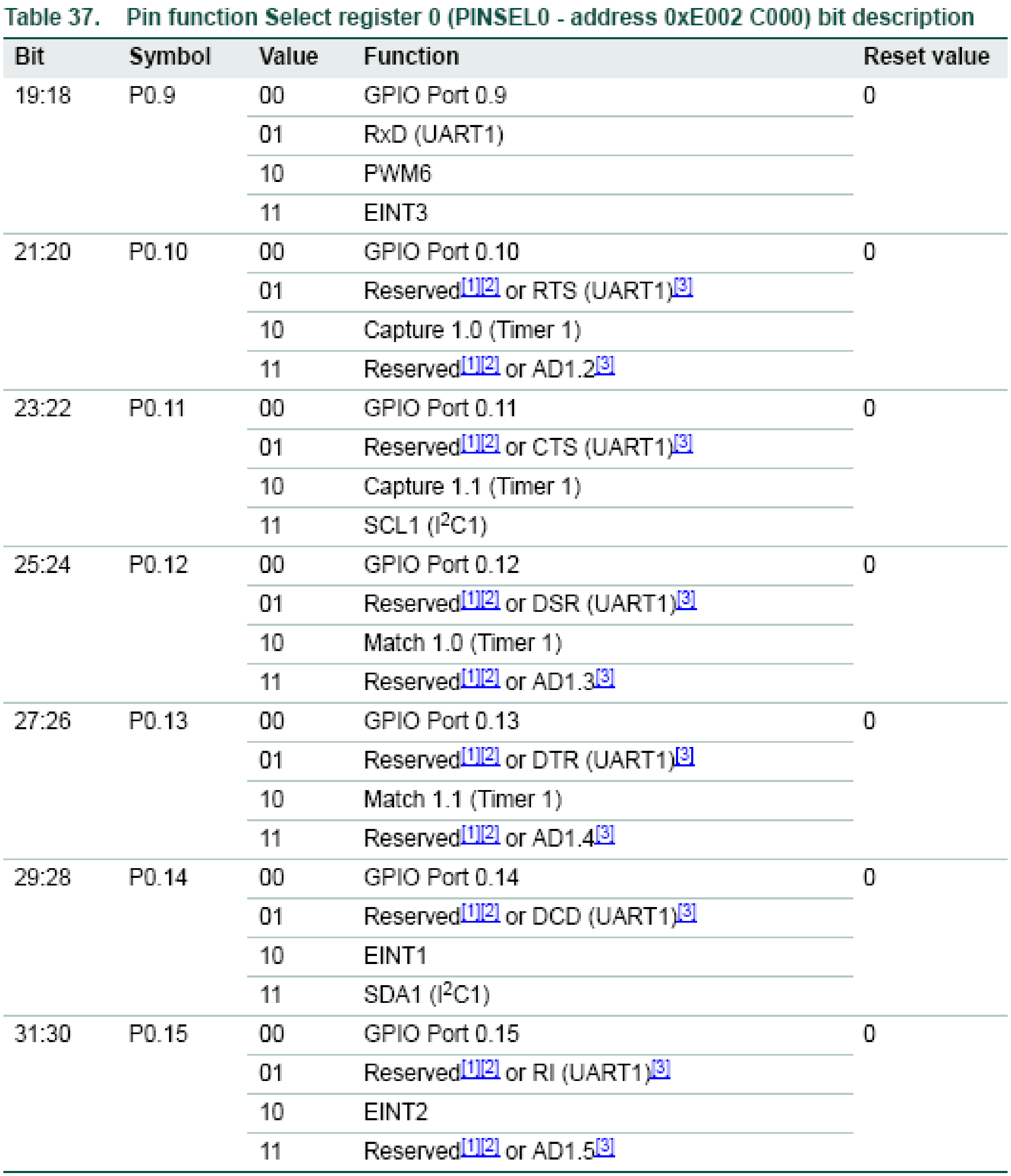


Table 37. Pin function Select register 0 (PINSEL0 - address 0xE002 C000) bit description

Bit	Symbol	Value	Function	Reset value
19:18	P0.9	00	GPIO Port 0.9	0
		01	RxD (UART1)	
		10	PWM6	
		11	EINT3	
21:20	P0.10	00	GPIO Port 0.10	0
		01	Reserved[1][2] or RTS (UART1)[3]	
		10	Capture 1.0 (Timer 1)	
		11	Reserved[1][2] or AD1.2[3]	
23:22	P0.11	00	GPIO Port 0.11	0
		01	Reserved[1][2] or CTS (UART1)[3]	
		10	Capture 1.1 (Timer 1)	
		11	SCL1 (I^{2}C1)	
25:24	P0.12	00	GPIO Port 0.12	0
		01	Reserved[1][2] or DSR (UART1)[3]	
		10	Match 1.0 (Timer 1)	
		11	Reserved[1][2] or AD1.3[3]	
27:26	P0.13	00	GPIO Port 0.13	0
		01	Reserved[1][2] or DTR (UART1)[3]	
		10	Match 1.1 (Timer 1)	
		11	Reserved[1][2] or AD1.4[3]	
29:28	P0.14	00	GPIO Port 0.14	0
		01	Reserved[1][2] or DCD (UART1)[3]	
		10	EINT1	
		11	SDA1 (I^{2}C1)	
31:30	P0.15	00	GPIO Port 0.15	0
		01	Reserved[1][2] or RI (UART1)[3]	
		10	EINT2	
		11	Reserved[1][2] or AD1.5[3]	

PINSEL1

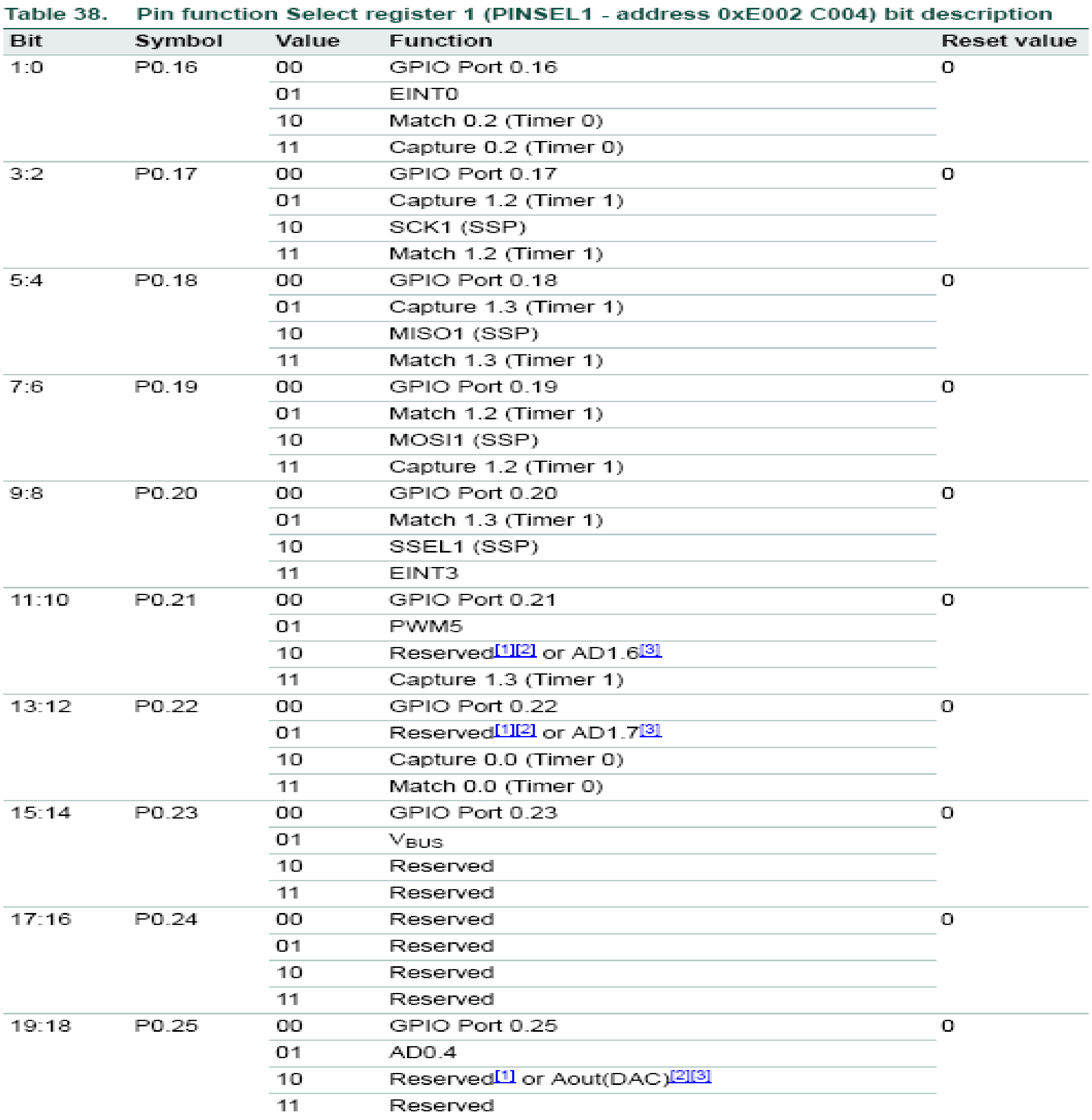


Table 38. Pin function Select register 1 (PINSEL1 - address 0xE002 C004) bit description

Bit	Symbol	Value	Function	Reset value
1:0	P0.16	00	GPIO Port 0.16	0
		01	EINT0	
		10	Match 0.2 (Timer 0)	
		11	Capture 0.2 (Timer 0)	
3:2	P0.17	00	GPIO Port 0.17	0
		01	Capture 1.2 (Timer 1)	
		10	SCK1 (SSP)	
		11	Match 1.2 (Timer 1)	
5:4	P0.18	00	GPIO Port 0.18	0
		01	Capture 1.3 (Timer 1)	
		10	MISO1 (SSP)	
		11	Match 1.3 (Timer 1)	
7:6	P0.19	00	GPIO Port 0.19	0
		01	Match 1.2 (Timer 1)	
		10	MOSI1 (SSP)	
		11	Capture 1.2 (Timer 1)	
9:8	P0.20	00	GPIO Port 0.20	0
		01	Match 1.3 (Timer 1)	
		10	SSEL1 (SSP)	
		11	EINT3	
11:10	P0.21	00	GPIO Port 0.21	0
		01	PWM5	
		10	Reserved[1][2] or AD1.6[3]	
		11	Capture 1.3 (Timer 1)	
13:12	P0.22	00	GPIO Port 0.22	0
		01	Reserved[1][2] or AD1.7[3]	
		10	Capture 0.0 (Timer 0)	
		11	Match 0.0 (Timer 0)	
15:14	P0.23	00	GPIO Port 0.23	0
		01	V_{BUS}	
		10	Reserved	
		11	Reserved	
17:16	P0.24	00	Reserved	0
		01	Reserved	
		10	Reserved	
		11	Reserved	
19:18	P0.25	00	GPIO Port 0.25	0
		01	AD0.4	
		10	Reserved[1] or Aout(DAC)[2][3]	
		11	Reserved	

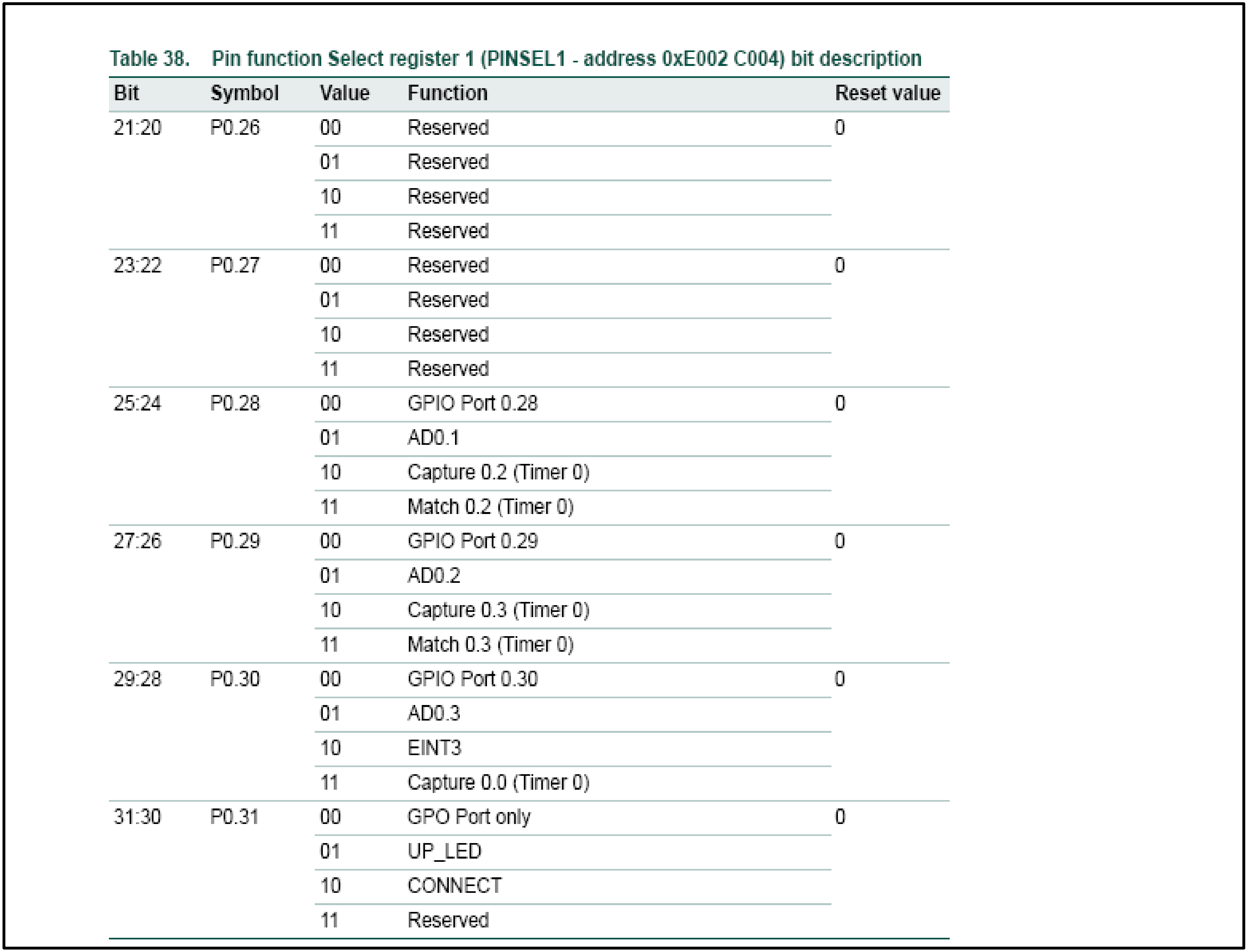


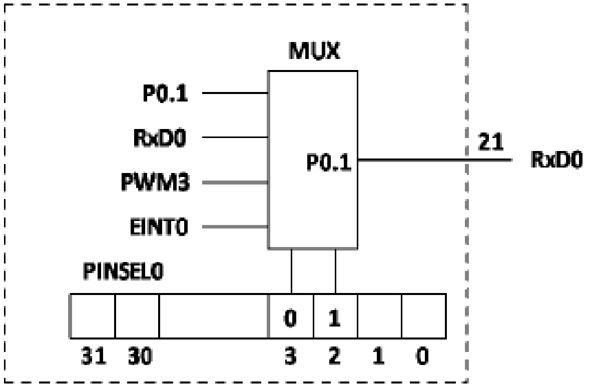
Table 38. Pin function Select register 1 (PINSEL1 - address 0xE002 C004) bit description

Bit	Symbol	Value	Function	Reset value
21:20	P0.26	00	Reserved	0
		01	Reserved	
		10	Reserved	
		11	Reserved	
23:22	P0.27	00	Reserved	0
		01	Reserved	
		10	Reserved	
		11	Reserved	
25:24	P0.28	00	GPIO Port 0.28	0
		01	AD0.1	
		10	Capture 0.2 (Timer 0)	
		11	Match 0.2 (Timer 0)	
27:26	P0.29	00	GPIO Port 0.29	0
		01	AD0.2	
		10	Capture 0.3 (Timer 0)	
		11	Match 0.3 (Timer 0)	
29:28	P0.30	00	GPIO Port 0.30	0
		01	AD0.3	
		10	EINT3	
		11	Capture 0.0 (Timer 0)	
31:30	P0.31	00	GPO Port only	0
		01	UP_LED	
		10	CONNECT	
		11	Reserved	

Pin Connect Block

- 64 pins are attached to two 32-bit I/O ports, Port-0 & Port-1.
- Port-0, Port-1 pins are designated as P0.0 – P0.31 & P1.0 - P1.31.
- Pins P0.24, P0.26, P0.27, P1.0-P1.15 are unavailable.
- Pin functions are multiplexed, up to 4 functions assigned to each pin.
  - Port-0 pins multiplex peripheral pin, & comm. interface pin functions
  - Port-1 pins multiplex JTAG interface, Trace function
  - Advantages: keeps size small, adds more functionalities to devices
  - Disadvantages: if functions not carefully selected, some can't be availed
- Pin function select Registers: PINSEL0, PINSEL1, PINSEL2
  - PINSEL0 selects functions of pins P0.0 to P0.15,
  - PINSEL1 selects functions of pins P0.16 to P0.31
  - PINSEL2 selects functions of pins P1.16 to P1.31

- Refer to technical manual for physical pin no., I/O port no., and functions assigned for each pin



- ## Pin function selection - Examples

1. Configuring P0.0 and P0.1 of Port-0 I/O pins for TxD0 and RxD0 functions of UART0 in ‘C’:

```
PINSEL0 &= ~(0xF);          //clear bits[3:2], [1:0] of PINSEL0 register, hence
                            //assign P0.0, P0.1 general purpose I/O function
PINSEL0 |=(1<<2)|(01);      //place 01 in bits[3:2], [1:0] of PINSEL0 register
                            //which selects TxD0 for P0.0 & RxD0 for P0.1 pins
                            //configuration of other pins remains same
```

2. For using DAC, select A_{out} function for P0.25 pin.

```
PINSEL1 &= ~(0x3<<18);      //clear bits[19:18] of PINSEL1 register
PINSEL1 |= (0x2<<18);       //place 10 into bits[19:18] of PINSEL1 reg.
                            //select Aout function for P0.25
```

Ports

- Port0 : P0.0 to P0.25 and P0.28 to P0.31

  P0.24, P0.26 and P0.27 are not available.

  P0.31 is output only.
- Port1 : only P1.16 to P1.31 are available

Registers associated with ports

- IO(0/1)DIR: control direction of pins
- IO(0/1)SET: setting at high level
- IO(0/1)CLR: clearing the bits
- IO(0/1)PIN: provides the value of port pins that are configured.

GPIO

- Pins not selected for peripheral functions are GP I/O port pin
- I/O port pins dynamically configured as input/output using GPIO reg.
- Two sets of GPIO registers -
  - Both control same I/O pins
  - One set on APB, provides legacy (normal) GPIO functionality
  - Another set on ARM local bus, provides enhanced (fast) GPIO function
    - In enhanced mode, registers are byte addressable
    - Includes mask registers to treat bits in groups
- GPIO registers
  - `IOxPIN` – To get logic value on a I/O pin
  - `IOxSET` – To set an output configured pin (by writing 1 in corresponding bit)
  - `IOxCLR` – To reset an output configured pin (by writing 1 in corresponding bit)
  - `IOxDIR` – To select input /output function (by placing 0/1) for an I/O pin

    (x = 0/1, i.e. Port-0 or Port-1)

- Examples: Configuring & initializing GPIO pins

```
PINSEL1  &= ~(0xFF);           // configure pins from P0.16 to P0.19 as GPIO

IO0DIR  |=  (0xF<<16);         // set pins P0.16 – P0.19 to output function

IO0SET  |=  (1<<16)|(1<<18);   // set pins P0.16 & P0.18 of Port-0 HIGH

IO0CLR  |=  (1<<16);           // reset logic level of pin P0.16 of Port-0 LOW
```

Blinking LEDS

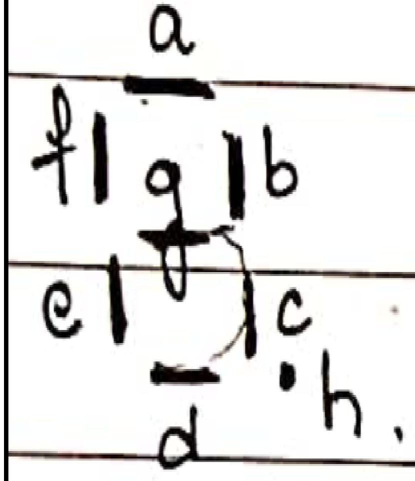
```
#include <LPC21xx.h>
unsigned int delay;
int main ()
{
    PINSEL1 = 0x00000000 ;    // Configure P0.16 to P0.31 as GPIO

   IO0DIR  = 0x00FF0000 ;        // Configure P0.16 to P0.23 as Output (o/p=1)

    while(1)
    {
        IO0CLR = 0x00FF0000;
          for(delay=0; delay<500000; delay++); // delay
        IO0SET = 0x00FF0000;
          for(delay=0;   delay<500000; delay++);      // delay
    }

}
```

7-Segment Display



h	g	f	e	d	c	b	a	Hex	decimal
0	0	1	1	1	1	1	1	= 3F	0
0	0	0	0	0	1	1	0	= 06	1
0	1	0	1	1	0	1	1	= 5B	2
0	1	0	0	1	1	1	1	= 4F	3

7-segment display

```
#include <LPC21XX.h>
```

Port0 Connected to data lines of all 7 segment displays (details given below)

a = P0.16
b = P0.17
c = P0.18
d = P0.19
e = P0.20
f = P0.21
g = P0.22
dot = P0.23

Select lines for four 7 Segments (each 7-seg display has 1 select line which enables it)

DIS1	P0.28
DIS2	P0.29
DIS3	P0.30
DIS4	P0.31

```
#include<LPC214x.h>
unsigned int delay;
unsigned int Switchcount=0;
unsigned int Disp[16]={0x003F0000, 0x00060000, 0x005B0000, 0x004F0000,

                        0x00660000,0x006D0000,  0x007D0000, 0x00070000, 0x007F0000,
                        0x006F0000, 0x00770000,0x007C0000,   0x00390000, 0x005E0000,
                        0x00790000, 0x00710000 };

#define SELDISP1 0x10000000                  //P0.28
#define SELDISP2 0x20000000              //P0.29
#define SELDISP3 0x40000000              //P0.30
#define SELDISP4 0x80000000              //P0.31
#define ALLDISP  0xF0000000              //Select all display
#define DATAPORT 0x00FF0000          //P0.16 to P0.23 Data lines connected to drive Seven Segments

int main (void)
{
        PINSEL0 = 0x00000000;
        PINSEL1 = 0x00000000;
        IO0DIR  = 0xF0FF0000;
        IO1DIR  = 0x00000000;
while(1)
    {
        IO0SET  |= ALLDISP;                         // select all digits
        IO0CLR = 0x00FF0000;                // clear the data lines to 7-segment displays
        IO0SET = Disp[Switchcount];          // get the 7-segment display value from the array

        if(!(IO1PIN & 0x00800000))                                 // if the key is pressed
        {
            for(delay=0;delay<100000;delay++)                                  // delay
             {}

          if((IO1PIN & 0x00800000))                                  // check to see if key has been
   released
            {
                Switchcount++;
                if(Switchcount == 0x10)         // 0 to F has been displayed ? go back to 0
              {
                 Switchcount = 0;
                 IO0CLR  = 0xF0FF0000;
              }
            }
        }
    }
}
```

Hex Counter

```
#include <LPC21xx.h>



unsigned int delay, count=0, Switchcount=0;
unsigned int Disp[16]={0x003F0000, 0x00060000, 0x005B0000, 0x004F0000,
      0x00660000,0x006D0000,
                              0x007D0000, 0x00070000, 0x007F0000, 0x006F0000,
      0x00770000,0x007C0000,
                              0x00390000, 0x005E0000, 0x00790000, 0x00710000 };

#define SELDISP1 0x10000000                        //P0.28
#define SELDISP2 0x20000000                        //P0.29
#define SELDISP3 0x40000000                        //P0.30
#define SELDISP4 0x80000000                        //P0.31
#define ALLDISP  0xF0000000                        //Select all display
#define DATAPORT 0x00FF0000            //P0.16 to P0.23 : Data lines connected to drive Seven
      Segments

int main (void)
{
        PINSEL0 = 0x00000000;
        PINSEL1 = 0x00000000;
        IO0DIR  = 0xF0FF0000;
        IO1DIR  = 0x01000000;
while(1)
    {
        //Display values on Seven Segment
        IO0SET |= ALLDISP;
        IO0CLR  = 0x00FF0000;

        for(delay=0;delay<100;delay++)
        IO0SET = Disp[Switchcount];          // display the values 0 to F
        for(delay=0;delay<1000000;delay++)
            {}

        Switchcount++;
        if(Switchcount == 16)                // after F go back to 0
        {
            Switchcount = 0;
        }
    }
}
```

Stepper motor

* A stepper motor direction is controlled by shifting the voltage across the coils. Port lines : P0.12 to P0.15

```
#include <LPC21xx.H>
void clock_wise(void);
void anti_clock_wise(void);

unsigned long int var1,var2;
unsigned int i=0,j=0,k=0;

int main(void)
{
        PINSEL0 = 0x00FFFFFF;              //P0.12 to P0.15 GPIo
        IO0DIR |= 0x0000F000;              //P0.12 to P0.15 output

        while(1)
        {
                for(j=0;j<50;j++)                      // 20 times in Clock wise Rotation
                 clock_wise();

                for(k=0;k<65000;k++);                              // Delay to show  anti_clock Rotation

                for(j=0;j<50;j++)                                          // 20 times in  Anti Clock wise Rotation
                        anti_clock_wise();

                for(k=0;k<65000;k++);   // Delay to show clock Rotation
        }                                                   // End of while(1)
}                                                           // End of main
```

```
void clock_wise(void)
{
       var1 = 0x00000800;             //For Clockwise
  for(i=0;i<=3;i++)            // for A B C D Stepping
       {
               var1 = var1<<1;        //For Clockwise
    var2 = ~var1;
    var2 = var2 & 0x0000F000;

               IO0PIN = ~var1;

    for(k=0;k<3000;k++);       //for step speed variation
   }

}
void anti_clock_wise(void)
{
       var1 = 0x00010000;             //For Anticlockwise
   for(i=0;i<=3;i++)           // for A B C D Stepping
   {
    var1 = var1>>1;            //For Anticlockwise
    var2 = ~var1;
    var2 = var2 & 0x0000F000;

               IO0PIN = ~var1;
    for(k=0;k<3000;k++);       //for step speed variation

   }
}
```

DC motor

```
#include<lpc214x.h>

void clock_wise(void);
void anti_clock_wise(void);
unsigned int j=0;

int main()
{
IO0DIR= 0X00000900;
IO0SET= 0X00000100;             //P0.8 should always high.

    while(1)
    {
    clock_wise();
    for(j=0;j<400000;j++);              //delay
    anti_clock_wise();
    for(j=0;j<400000;j++);              //delay
    }                               //End of while(1)
}                                         //End of Main
```

```
void clock_wise(void)
{
    IO0CLR = 0x00000900;              //stop motor and also turn off relay
    for(j=0;j<10000;j++);              //small delay to allow motor to turn off
    IO0SET = 0X00000900;   //Selecting the P0.11 line for clockwise and turn on
  motor
}

void anti_clock_wise(void)
{
    IO0CLR = 0X00000900;              //stop motor and also turn off relay
    for(j=0;j<10000;j++);        //small delay to allow motor to turn off
    IO0SET = 0X00000100;         //not selecting the P0.11 line for Anti clockwise
}
```

Binary counter

```
#include <LPC21xx.h>
void delay(void);
unsigned int HexValue;

int main ()
{
        unsigned int not_hexvalue=0;
        PINSEL0 = 0x00000000;
        IO0DIR = 0x00FF0000;
        while(1)
        {
                for(HexValue=0; HexValue <= 0xff; HexValue++)
                {
                        not_hexvalue= (~HexValue);              // for incrementing display from 00 to ff
                        not_hexvalue &= 0x000000ff;
                        IO0PIN = (not_hexvalue << 16) ;         // | 0xff00ffff;
                        delay();
                }
        }
}

void delay(void)
{       unsigned int count;
         for(count=0; count< 650000; count++)
        {}
}
```

Relay program

```
#include <LPC21xx.h>

unsigned int i;

int main ()
{
        IO0DIR = 0x00000400;                    //Set P0.10 as output
        IO0SET = 0x00000400;                    //P0.10 is set to a HI

        while(1)
        {
                for(i=0;i<1000000;i++)
                {}
                IO0SET = 0x00000400;            //RLY ON
                for(i=0;i<1000000;i++)
                {}
                IO0CLR = 0x00000400;            //RLY OFF
        }
}
```