Laboratorio 9

R. Ferrero, P. Bernardi

Politecnico di Torino

Dipartimento di Automatica e Informatica (DAUIN)

Torino - Italy

Diagramma a blocchi del SoC (p.9)

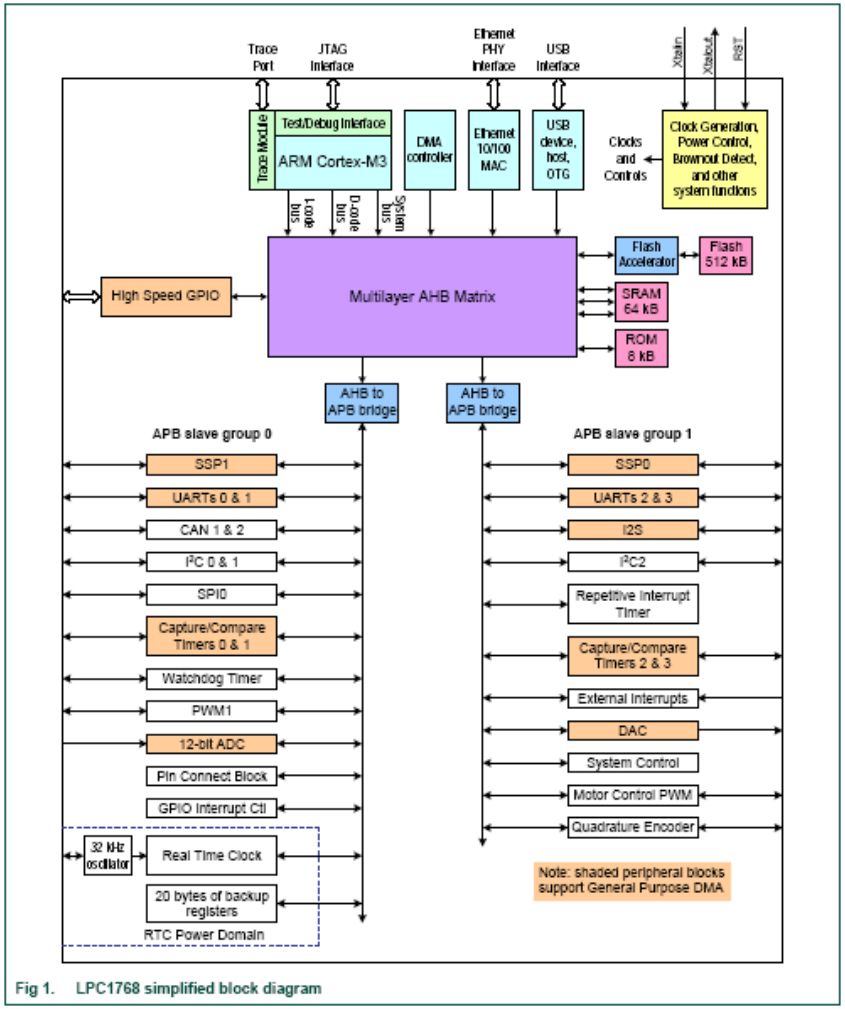
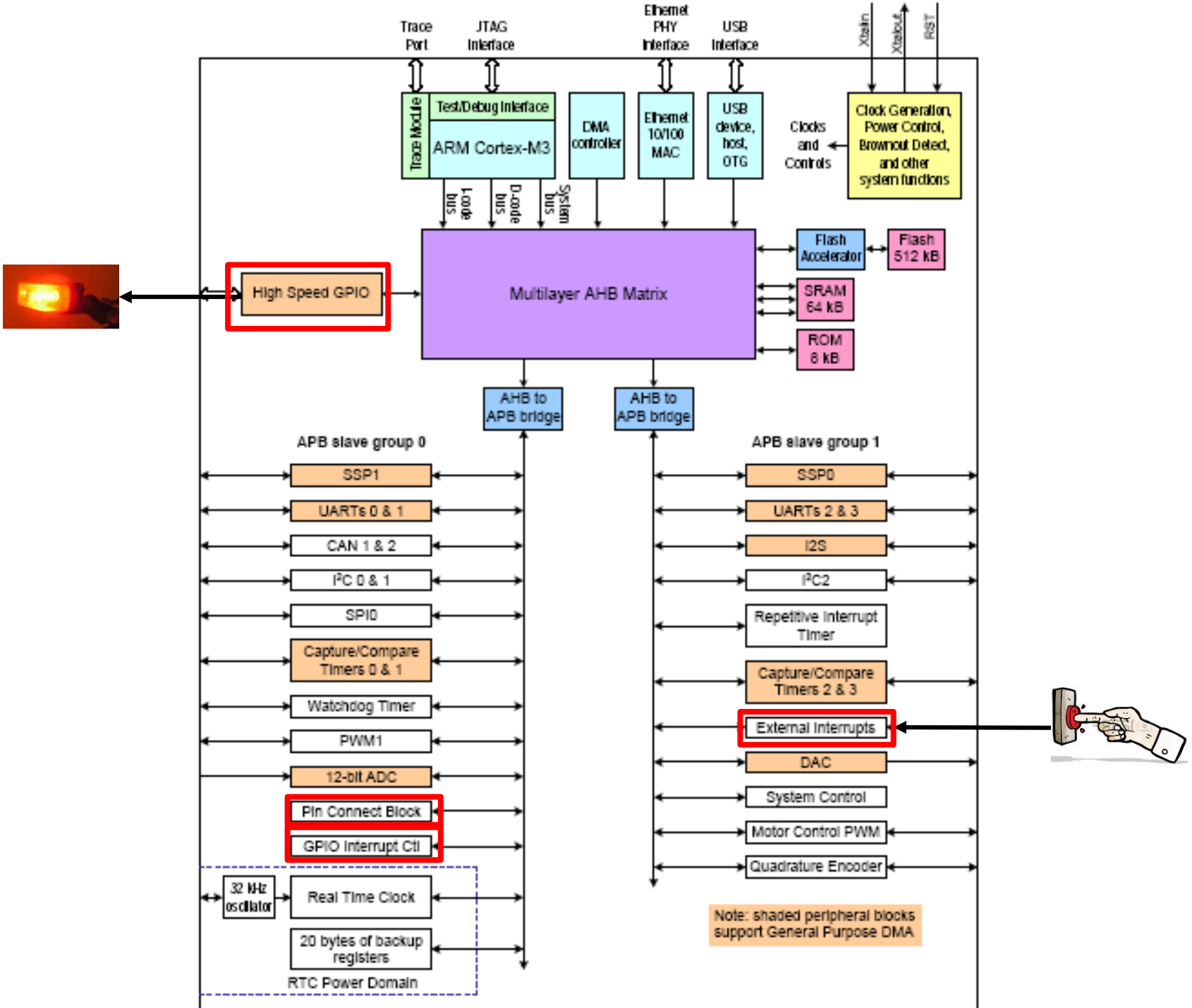


Fig 1. LPC1768 simplified block diagram

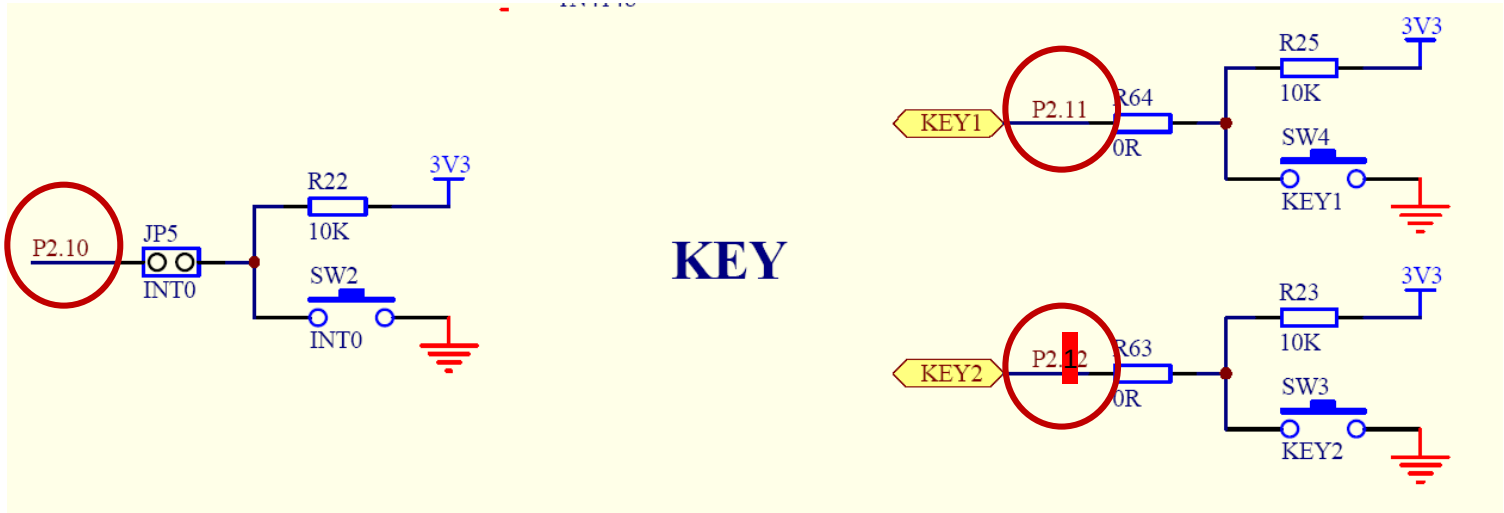

Trace Port
JTAG Interface
Ethernet PHY Interface
USB Interface
Xtalin
Xtalout
RST
Trace Module
Test/Debug Interface
ARM Cortex-M3
DMA controller
Ethernet 10/100 MAC
USB device, host, OTG
Clocks and Controls
Clock Generation, Power Control, Brownout Detect, and other system functions
I-code bus
D-code 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

Progetto Sample: contenuto

- file di avvio del chip:
  - startup_LPC17xx.s
- librerie di sistema
  - core_cm3.c
  - system_LPC17xx.c
- Librerie di alcuni periferici:
  - *peripheral*.h /* prototipi */
  - lib_*peripheral*.c /* funzioni di base */
  - IRQ_*peripheral*.c /* interrupt service routine */
  - funct_*peripheral*.c /* funzioni utente */

Pulsanti

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



Pin connect block (pag. 119)

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

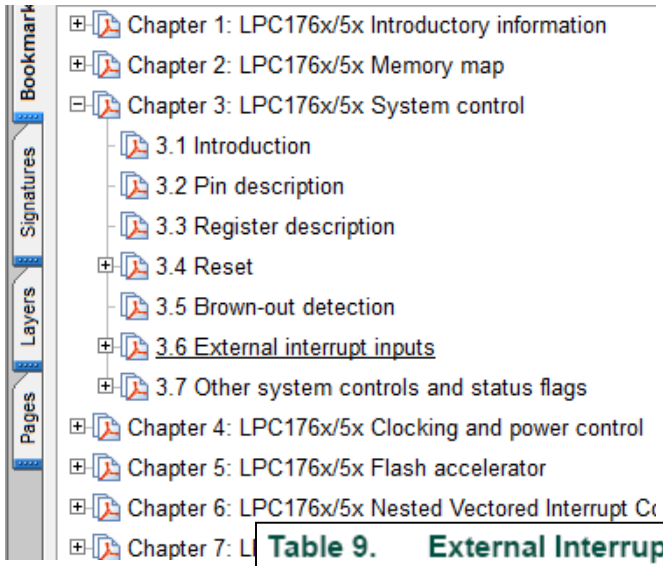
PINSEL4	Pin name	Function when 00	Function when 01	Function when 10	Function when 11	Reset value
1:0	P2.0	GPIO Port 2.0	PWM1.1	TXD1	Reserved	00
3:2	P2.1	GPIO Port 2.1	PWM1.2	RXD1	Reserved	00
5:4	P2.2	GPIO Port 2.2	PWM1.3	CTS1	Reserved [2]	00
7:6	P2.3	GPIO Port 2.3	PWM1.4	DCD1	Reserved [2]	00
9:8	P2.4	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

FIODIR (pag. 132)

Table 104. Fast GPIO port Direction register FIO0DIR to FIO4DIR - addresses 0x2009 C000 to 0x2009 C080) bit description

Bit	Symbol	Value	Description	Reset value
31:0	FIO0DIR FIO1DIR FIO2DIR FIO3DIR FIO4DIR		Fast GPIO Direction PORTx control bits. Bit 0 in FIOxDIR controls pin Px.0, bit 31 in FIOxDIR controls pin Px.31.	0x0
		0	Controlled pin is input.	
		1	Controlled pin is output.	

External Interrupt Mode



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

Nested Vectored Interrupt Controller (pag. 77)

Table 51. NVIC register map

Name	Description	Access	Reset value	Address
ISER0 to ISER1	Interrupt Set-Enable Registers. These 2 registers allow enabling interrupts and reading back the interrupt enables for specific peripheral functions.	RW	0	ISER0 - 0xE000 E100 ISER1 - 0xE000 E104
ICER0 to ICER1	Interrupt Clear-Enable Registers. These 2 registers allow disabling interrupts and reading back the interrupt enables for specific peripheral functions.	RW	0	ICER0 - 0xE000 E180 ICER1 - 0xE000 E184
ISPR0 to ISPR1	Interrupt Set-Pending Registers. These 2 registers allow changing the interrupt state to pending and reading back the interrupt pending state for specific peripheral functions.	RW	0	ISPR0 - 0xE000 E200 ISPR1 - 0xE000 E204
ICPR0 to ICPR1	Interrupt Clear-Pending Registers. These 2 registers allow changing the interrupt state to not pending and reading back the interrupt pending state for specific peripheral functions.	RW	0	ICPR0 - 0xE000 E280 ICPR1 - 0xE000 E284
IABR0 to IABR1	Interrupt Active Bit Registers. These 2 registers allow reading the current interrupt active state for specific peripheral functions.	RO	0	IABR0 - 0xE000 E300 IABR1 - 0xE000 E304
IPR0 to IPR8	Interrupt Priority Registers. These 9 registers allow assigning a priority to each interrupt. Each register contains the 5-bit priority fields for 4 interrupts.	RW	0	IPR0 - 0xE000 E400 IPR1 - 0xE000 E404 IPR2 - 0xE000 E408 IPR3 - 0xE000 E40C IPR4 - 0xE000 E410 IPR5 - 0xE000 E414 IPR6 - 0xE000 E418 IPR7 - 0xE000 E41C IPR8 - 0xE000 E420
STIR	Software Trigger Interrupt Register. This register allows software to generate an interrupt.	WO	0	STIR - 0xE000 EF00

Esercizio 1

- Nel main, prima di entrare nel ciclo infinito, accendere il led 8 tramite `LED_On`.
- Premendo il pulsante KEY1, spegnere il led corrente e accendere il led a sinistra (arrivati al led 4, si passa al led 11).
- Premendo il pulsante KEY2, spegnere il led corrente e accendere il led a destra (arrivati al led 11 si passa al led 4).
- Premendo il pulsante INT0, tornare alla configurazione iniziale, con il led 8 acceso.

Memorizzare il led acceso

- Per conoscere il led acceso si può:
  - leggere il contenuto di `LPC_GPIO2->FIOPIN`
  - leggere il contenuto di `LPC_GPIO2->FIOSET`
  - *definire* una variabile globale in funct_led.c:

```
unsigned int led_value;
```

`led_value` indica il led acceso.

Negli altri file si può accedere alla variabile *dichiarando*:

```
extern unsigned int led_value;
```

Rimbalzo del pulsante

- Una singola pressione del pulsante può scatenare più richieste di servizio di interrupt.
- Nell'esercizio 1, il led acceso potrebbe essere due posizioni a sinistra (o a destra) rispetto a quello spento.
- Occorre implementare via software un meccanismo anti-rimbalzo per servire la prima richiesta di interrupt e ignorare l'eventuale richiesta immediatamente successiva.

Esercizio 2

- Implementare una slot machine con tre rulli rotanti.

- Ogni rullo mostra uno fra due simboli:

Rullo	Simbolo 1	Simbolo 2
rullo 1	led 4 acceso	led 5 acceso
rullo 2	led 6 acceso	led 7 acceso
rullo 3	led 8 acceso	led 9 acceso

Esercizio 2: implementazione

- Il pulsante KEY1 fa iniziare una nuova partita e comanda il primo rullo:
  - spegne tutti i led
  - accende casualmente uno fra i led 4 e 5.
- Il pulsante KEY2 comanda il secondo rullo:
  - accende casualmente uno fra i led 6 e 7.
- Il pulsante INT0 comanda il terzo rullo e determina l'eventuale vincita:
  - accende casualmente uno fra i led 8 e 9
  - in base all'esito della partita, accende il led 10 o 11.

Esercizio 2: esito della partita

- Il giocatore vince se i 3 simboli sono uguali:
  - tutti i rulli mostrano il simbolo 1, oppure
  - tutti i rulli mostrano il simbolo 2.
- Al termine della partita (dopo la pressione di INT0), la vincita è indicata accendendo il led 11.
- Se invece i simboli non sono uguali, il giocatore perde e si accende il led 10.

Generazione di numeri casuali

- I led 4-9 devono essere accesi casualmente.
- Un semplice modo per ottenere numeri casuali è misurare il tempo intercorso fra la pressione di due pulsanti (incrementando una variabile).