Sistemi di bus su chip, sviluppo di SoC con FPGA

Esercitazione 10 di Sistemi dedicati

Docente: Giuseppe Scollo

Università di Catania
Dipartimento di Matematica e Informatica
Corso di Laurea Magistrale in Informatica, AA 2016-17

Indice

1. Sistemi di bus su chip, sviluppo di SoC con FPGA
2. argomenti dell'esercitazione
3. standard di bus su chip
4. componenti di bus
5. realizzazione fisica di bus su chip
6. convenzioni sui nomi nei bus
7. diagrammi di temporizzazione nei bus
8. definizione di un bus generico
9. corrispondenza di bus standard al bus generico
10. esperienza di laboratorio
11. riferimenti

argomenti dell'esercitazione

in questa esercitazione si trattano:

- alcuni standard di bus su chip
- componenti e realizzazione fisica di un bus su chip
- convenzioni sui nomi nei bus
- diagrammi di temporizzazione nei bus
- astrazione di alcuni bus standard in una definizione di bus generico
- esperienza di laboratorio:
  - costruzione di un semplice sistema Nios II su FPGA
  - costruzione di un proprio componente con interfaccia di bus Avalon-MM
  - integrazione del proprio componente con un sistema Nios II in un progetto Quartus

standard di bus su chip

quattro famiglie di standard di bus su chip, fra le più in uso:

- *AMBA* (Advanced Microcontroller Bus Architecture): famiglia di sistemi di bus usati da processori ARM
- *CoreConnect*: sistema di bus per processori PowerC di IBM
- *Wishbone*: sistema di bus open-source proposto dalla SiliCore Corporation, usato da molti componenti hardware open-source, e.g. quelli del progetto OpenCores
- *Avalon*: sistema di bus per applicazioni SoC dei processori Nios di Altera

due classi principali di configurazioni: *condivisi* e *punto-punto*

varianti ulteriori per velocità, interfaccia, topologia ecc., v. tabella 10.1

si considerano appresso un bus generico condiviso e uno punto-punto, astraendo caratteristiche comuni a tutti questi

Bus	High-performance shared bus	Periferal shared bus	Point-to-point bus
AMBA v3	AHB	APB	
AMBA v4	AXI4	AXI4-lite	AXI4-stream
CoreConnect	PLB	OPB	
Wishbone	Crossbar topology	Shared topology	Point to point topology
Avalon	Avalon-MM	Avalon-MM	Avalon-ST

AHB AMBA highspeed bus, *APB* AMBA peripheral bus, *AXI* advanced extensible interface, *PLB* processor local bus, *OPB* onchip peripheral bus, *MM* memory-mapped, *ST* streaming

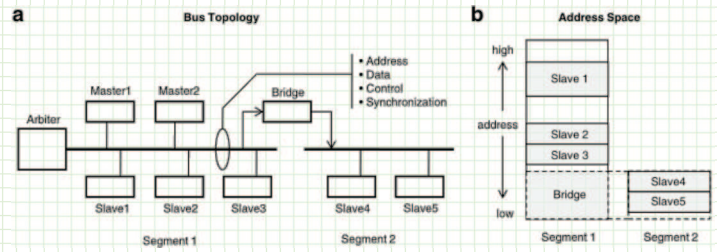
Schaumont, Table 10.1 - Bus configurations for existing bus standards

componenti di bus

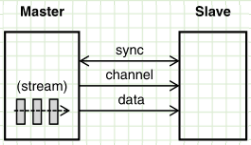
un bus condiviso su chip consiste tipicamente di più *segmenti*, connessi da *ponti*; ogni transazione è iniziata da un *master*, a cui risponde uno *slave*; se questi stanno su segmenti diversi, il ponte agisce da slave su uno dei segmenti e da master sull'altro, effettuando la *traduzione degli indirizzi*

quattro classi di segnali di bus:

- *dati*: linee dati separate per lettura e scrittura
- *indirizzi*: la decodifica può essere centralizzata o locale negli slave
- *comandi*: per distinguere lettura da scrittura, spesso qualificate da più segnali
- *sincronizzazione*: clock, distinti per segmenti diversi, e altri, quali: segnali di handshake, time-out ecc.



Schaumont, Figure 10.1 - (a) Example of a multi-master segmented bus system. (b) Address space for the same bus



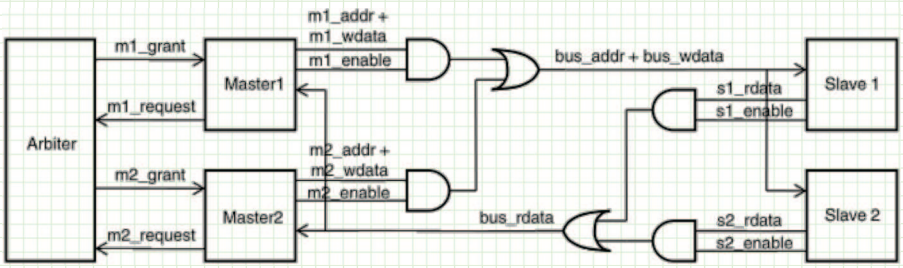
Schaumont, Figure 10.2 - Point-to-point bus

un bus punto-punto è una connessione fisica dedicata fra un master e uno slave, per trasferimenti di dati in flussi (*stream*) illimitati

- non ha linee di indirizzo, ma può averne per canale logico nel caso di multiplazione del bus fisico per più flussi di dati
- sincronizzazione molto simile al protocollo di handshake descritto prima

realizzazione fisica di bus su chip

la figura 10.3 mostra la disposizione fisica di un tipico segmento di bus su chip, con due master e due slave, dove le porte AND e OR al centro del diagramma fungono da multiplatori, sia delle linee indirizzo che delle linee dati



Schaumont, Figure 10.3 - Physical interconnection of a bus. The *__addr, *__wdata, *__sdata signals are signal vectors. The *__enable, *__grant, *__request signals are single-bit signals

convenzione sui nomi dei segnali di lettura/scrittura di dati:

scrittura di un dato significa il suo invio da master a slave

lettura di un dato significa il suo invio da slave a master

l'arbitraggio del bus assicura che solo un componente per volta piloti qualsiasi linea del bus

convenzioni sui nomi nei bus

una convenzione sui nomi aiuta a dedurre funzionalità e connettività di linee di bus dai loro nomi

> per esempio, può essere molto utile per leggere un diagramma di temporizzazione, o per visualizzare la connettività in una netlist (testuale) di un circuito

il nome di un pin di un componente rifletterà la funzionalità del pin; segnali di bus, creati da interconnessioni di pin di componenti, seguiranno anch'essi una convenzione, per evitare confusione fra segnali simili

> per esempio, in figura 10.3, vi sono due componenti master, ciascuno con un segnale `wdata`; per distinguere questi segnali, il nome dell'istanza del componente è incluso come prefisso nel nome di segnale del bus, quale `m2_wdata`

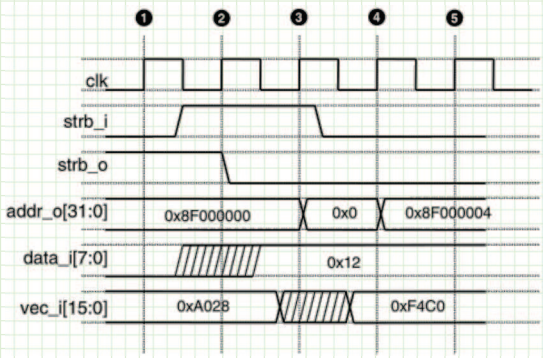
in alcuni sistemi di bus si includono nel nome del segnale di bus entrambi i nomi delle istanze di master e slave

> ne risultano identificatori molto lunghi, quale e.g. `cpu1_read_data_valid_sdram0`, forse noiosi da digitare ma che riflettono con precisione la funzione della linea

l'adozione delle convenzioni del particolare bus in uso evita errori di codifica ed errori di connessione, rende il codice riusabile e semplifica il debugging e la verifica

diagrammi di temporizzazione nei bus

per la natura inerentemente parallela di un sistema di bus, i diagrammi di temporizzazione sono in ampio uso per descrivere le relazioni temporali fra segnali di bus



Schaumont, Figure 10.4 - Bus timing diagram notation

il diagramma in figura 10.4 mostra la notazione per descrivere le attività in un bus generico per cinque cicli di clock

- i segnali fanno riferimento al fronte di salita del clock, mostrato in alto
- i segnali di input in un ciclo di clock hanno il valore acquisito *prima* del suo fronte di inizio
- i segnali di output hanno il valore prodotto in un ciclo di clock *dopo* il suo fronte di fine

questi diagrammi sono molto utili per descrivere le attività su un bus in funzione del tempo

> sono anche una parte di documentazione centrale al progetto di un'interfaccia HW/SW

la tabella 10.2 elenca i segnali di un sistema di bus generico, astraendo da specifici sistemi di bus

Signal name	Meaning
clk	Clock signal. All other bus signals are references to the upgoing clock edge
m_addr	Master address bus
m_data	Data bus from master to slave (write operation)
s_data	Data bus from slave to master (read operation)
m_rnw	Read-not-Write. Control line to distinguish read from write operations
m_sel	Master select signal, indicates that this master takes control of the bus
s_ack	Slave acknowledge signal, indicates transfer completion
m_addr_valid	Used in place of m_sel in split-transfers
s_addr_ack	Used for the address in place of s_ack in split-transfers
s_wr_ack	Used for the write-data in place of s_ack in split-transfers
s_rd_ack	Used for the read-data in place of s_ack in split-transfers
m_burst	Indicates the burst type of the current transfer
m_lock	Indicates that the bus is locked for the current transfer
m_req	Requests bus access to the bus arbiter
m_grant	Indicates bus access is granted

Schaumont, Table 10.2 - Signals on the generic bus

la tabella 10.3 mostra la corrispondenza di alcuni segnali del bus generico a segnali equivalenti dei bus CoreConnect/OPB, AMBA/APB, Avalon-MM e Wishbone

generic	CoreConnect/OPB	AMBA/APB	Avalon-MM	Wishbone
clk	OPB_CLK	PCLK	clk	CLK_I (master/slave)
m_addr	Mn_ABUS	PADDR	Mn_address	ADDR_O (master) ADDR_I (slave)
m_rnw	Mn_RNW	PWRITE	Mn_write_n	WE_O (master)
m_sel	Mn_Select	PSEL		STB_O (master)
m_data	OPB_DBUS	PWDATA	Mn_writedata	DAT_O (master) DAT_I (slave)
s_data	OPB_DBUS	PRDATA	Mb_readdata	DAT_I (master) DAT_O (slave)
s_ack	Sl_XferAck	PREADY	Sl_waitrequest	ACK_O (slave)

Schaumont, Table 10.3 - Bus signals for simple read/write on Coreconnect/OPB, ARM/APB, Avalon-MM and Wishbone busses

esperienza di laboratorio

i due tutorial sullo strumento Qsys, accessibile da Quartus, indicati nei riferimenti, rispettivamente mostrano come:

- costruire su FPGA un sistema Nios II, con memoria su chip e interfacce di I/O necessarie per l'esecuzione di un semplice programma che usa gli interruttori e i LED sulla board DE1-SoC, quindi integrare il sistema in un progetto Quartus e infine compilare ed eseguire il programma sotto il controllo dell'Altera Monitor Program;
- costruire un proprio componente con interfaccia di bus Avalon-MM, integrarlo in un sistema Nios II, caricare tale sistema su FPGA e osservarne il funzionamento mediante l'Altera Monitor Program

si propone di replicare l'esecuzione di questi due esperimenti e di esporre, in una relazione sull'esperienza, le difficoltà incontrate e le soluzioni adottate per superarle

i file sorgenti per questo sono disponibili nell'area riservata per il laboratorio

riferimenti

letture raccomandate:

Schaumont (2012) Cap. 10, Sez. 10.1

letture per ulteriori approfondimenti:

Schaumont (2012) Cap. 10, Sez. 10.2-10.4

materiali utili per l'esperienza di laboratorio proposta:

Introduction to the Altera Qsys System Integration Tool - For Quartus Prime 16.0, Altera University Program, May 2016

Making Qsys Components - For Quartus Prime 16.0, Altera University Program, May 2016

file sorgenti per l'esperienza di laboratorio (nell'area riservata per il laboratorio)