INTRODUZIONE AL BUS INDUSTRIALE VME

Paolo Musico

INFN Genova

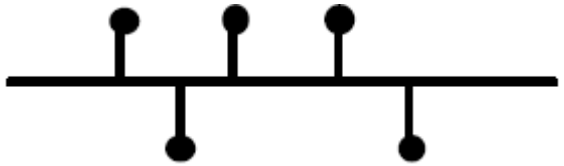
Novembre 2009

Sommario:

- ✓ Introduzione
- ✓ Meccanica
- ✓ Cicli di trasferimento dati standard
- ✓ Cicli di trasferimento dati innovativi
- ✓ Prove sul campo: analisi dei segnali, velocita', …

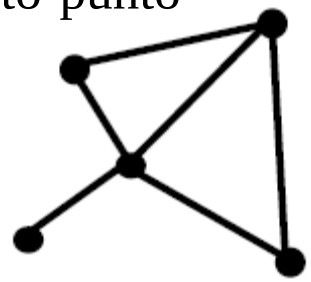
CONCETTO DI BUS

Un BUS e' una struttura di interconnessione tramite linee di comunicazione che permette lo scambio di informazioni tra varie unita' dello stesso sistema:
per esempio tra la CPU, la memoria centrale e le periferiche.
Le linee del BUS sono condivise tra tutti i dispositivi.
Esistono altre strutture di interconnessione utilizzate: punto-punto, anello, stella.

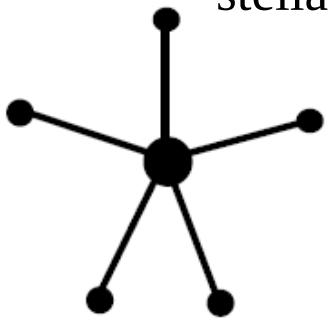


BUS

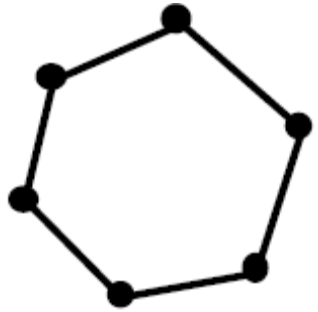
punto-punto



stella



anello



UN PO' DI STORIA E RIFERIMENTI

VME - Versa Module Europa
Introdotto da Motorola, Mostek e Signetics nel 1981
Definito dall' IEEE 1014-1987 standard
Motorola propose l'uso del VERSA bus, ma gli altri partners dissero che le schede erano troppo grosse, quindi si scelse il formato EUROCARD.
Quindi si adotto' il formato meccanico EUROCARD con lo standard elettrico VERSA.

Estensioni: VME64 (1995), VME64x (1998), VME320 (1999).

VITA - VMEbus International Trade Association (www.vita.com): organizzazione no-profit di costruttori e utenti interessati al real-time embedded computing, che definisce gli standards VME.

See also: http://esone.web.cern.ch/ESONE/Syscomms98/slides/23.HTML
http://ess.web.cern.ch/ESS/standards.htm

VELOCITA' DI TRASFERIMENTO

Topologia	Bus Cycle	Massima Velocita'
VMEbus IEEE-1014	BLT	40 Mbyte/sec
VME64	MBLT	80 Mbyte/sec
VME64x	2eVME	160 Mbyte/sec
VME320	2eSST	320 - 500+ Mbyte/sec

GENERALITA'

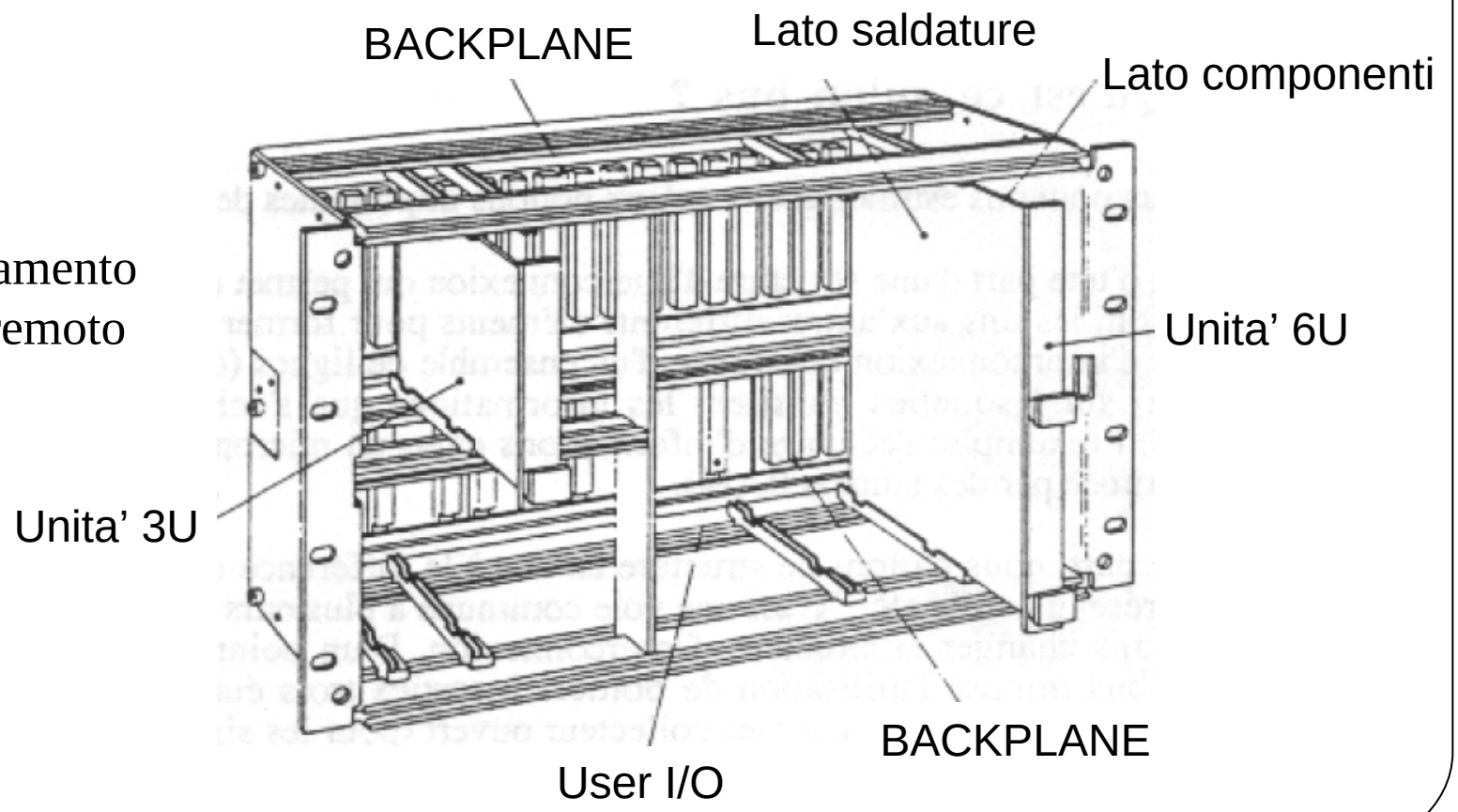
Il VME e' un bus multi-processore che permette l'uso di CPU a 8, 16, 32 bit.
La struttura del BUS e' master-slave: il master gestisce il ciclo e lo slave
che ha decodificato il proprio indirizzo risponde alla transazione.
Il bus permette trasferimenti dati asincroni: lo slave termina il ciclo con
un acknowledge quando ha finito di processare la transazione.
Esiste un ciclo indivisibile Read Modify Write, che permette l' implementazione
di semafori (per gestire risorse comuni in un sistema multi-master o multi-processore).
Sono disponibili 7 linee di interrupts per richiedere l' attenzione del/dei master.

Le specifiche definiscono il protocollo per il trasferimento dati e la gestione del bus
stesso, i livelli elettrici utilizzati e tutta la meccanica: formato schede, connettori, …

MECCANICA

Cestello (crate) 19” standard
21 posizioni scheda
Passo: 0.8”

Mancano:
- Alimentatore
- Cassetto raffreddamento
- Controllo locale/remoto



Formato schede standard

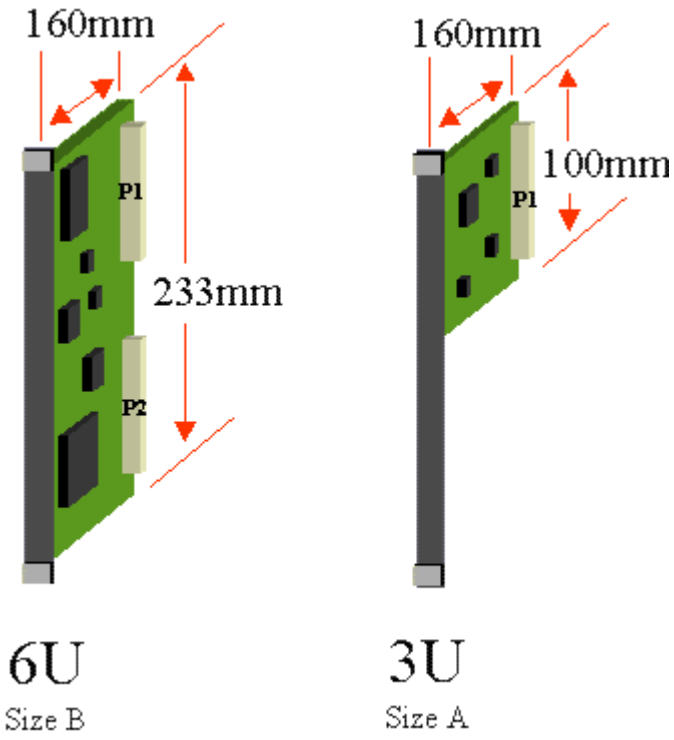
Il formato 3U e' poco usato.

Esistono estensioni del formato 6U:
6U × 220 mm, 6U × 280 mm

Esiste il formato 9U × 400 mm

Il VXI definisce altri formati:
6U × 340 mm (size C)
9U × 340 mm (size D)

(9U = 366 mm)



Segnali del bus

Le linee del bus possono essere classificate in 4 categorie.

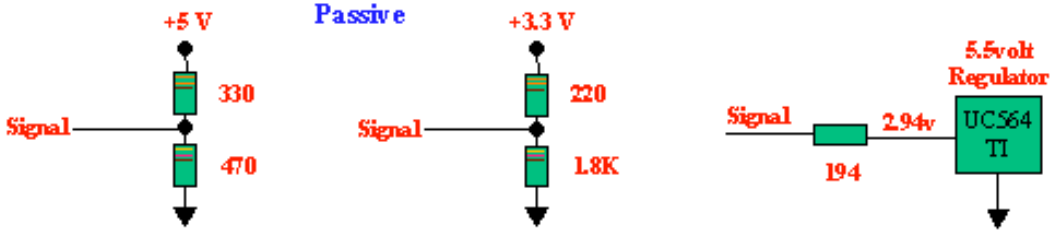
Trasferimento dati. Data Transfer Bus (DTB): indirizzi [A1:A31], dati [D0:D31], Segnali di controllo (AM[0:5], AS, LWORD, WRITE, DS0, DS1, DTACK, BERR)

Arbitraggio. Request/grant

Interrupts. 7 linee di richiesta interrupt, acknowledge

Utilities. Clock, reset, failure, alimentazione stand-by.

Tutte i segnali del bus sono single ended a livelli TTL, ad impedenza controllata, terminate ad entrambi gli estremi.



SIGNIFICATO DEI SEGNALI

A31:A01	31 linee indirizzi
D31:D00	32 linee dati bidirezionali
AM5:AM0	6 modificatori di indirizzo
AS*	Address strobe attivo basso
DS0*	Data strobe per D7:D0 , attivo basso
DS1*	Data strobe per D15:D8, attivo basso
LWORD*	trasferimento dati a 32 bit, attivo basso
WRITE*	scrittura da MASTER a SLAVE, attivo basso
DTACK*	ciclo terminato con successo, attivo basso, open collector
BERR*	ciclo terminato con errore, attivo basso, open collector
IRQ7*:IRQ1*	7 linee di interrupt, attive basse, open collector
IACK*	Interrupt acknowledge, attivo basso
IACKin*	Interrupt acknowledge daisy chain, attivo basso
IACKout*	Interrupt acknowledge daisy chain, attivo basso
BR3*:BR0*	4 linee di richiesta di accesso del bus, attive basse, open collector
BGin3*:BGin0*	4 linee di abilitazione del bus, attive basse
BGout3*:BGout0*	4 linee di abilitazione del bus, attive basse
BBSY*	Bus is busy, attivo basso, open collector
BCLR*	Bus is clear, attivo basso

TRASFERIMENTO DATI SUL DTB (1)

Il DTB e' un bus di trasmissione dati parallelo, asincrono.
Le opzioni per le transazioni sono le seguenti:

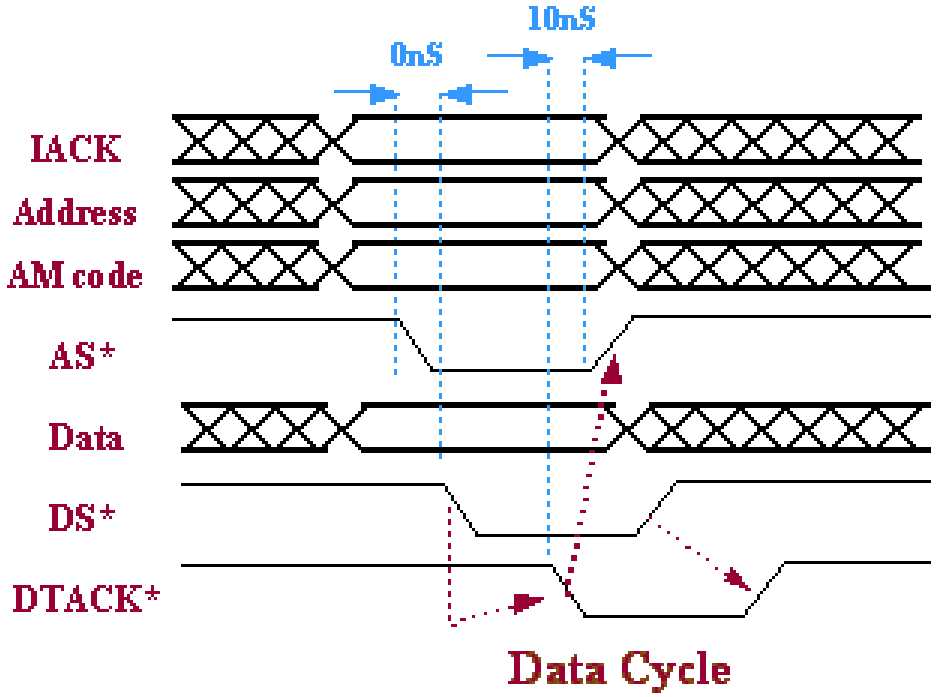
- D8, D16, D32: definisce la dimensione della parola dati da trasferire (esiste anche D64)
- A16, A24, A32: definisce il tipo di indirizzamento (esiste anche A64)
- BTO(x): Bus time-out di durata x microsecondi
- BLT: trasferimento sequenziale di dati (a blocchi)
- RMW: read modify write, usato per accesso a risorse condivise in caso multiprocessor
- ADO: trasferimento del solo indirizzo
- UAT: trasferimento dati non allineati

La transazione sul DTB coinvolge sempre 2 dispositivi: MASTER e SLAVE.
MASTER: controlla lo scambio dati con lo SLAVE. Forza sul bus le linee indirizzi e le linee di controllo.
SLAVE: Decodifica le linee di indirizzo e risponde alla transazione terminandola con un acknowledge: DTACK in caso positivo o BERR in caso di errore

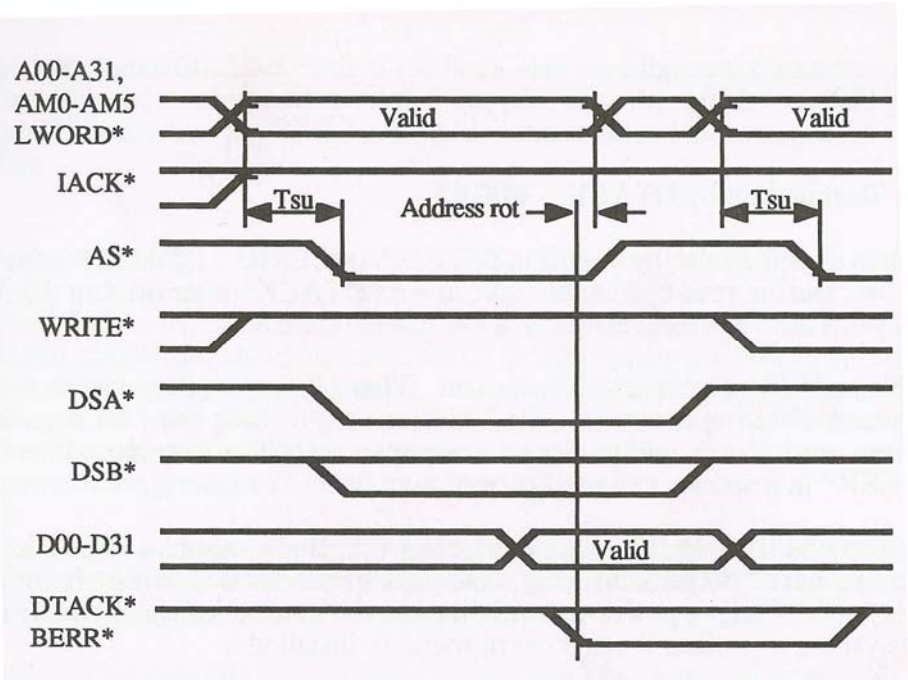
TRASFERIMENTO DATI SUL DTB (2)

Il **MASTER** controlla:
- 15, 23 o 31 linee indirizzi [A31:A01]
- 6 linee di selezione ciclo [AM5:AM0]
- Address strobe AS*
- Data strobes DS0* e DS1*
- 6, 16 o 32 linee dati [D31:D0] ciclo di scrittur
- La linea LWORD*
- La linea di direzione dati WRITE*

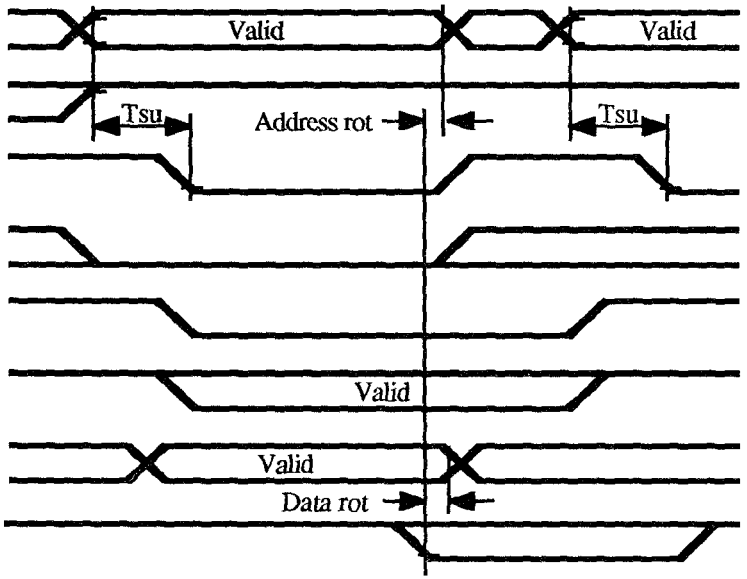
Lo **SLAVE** controlla:
- 8, 16 o 32 linee dati [D31:D0] ciclo di lettura
- La linea DTACK* (transazione valida)
- La linea BERR* (transazione errata)



TRASFERIMENTO DATI SUL DTB (3)

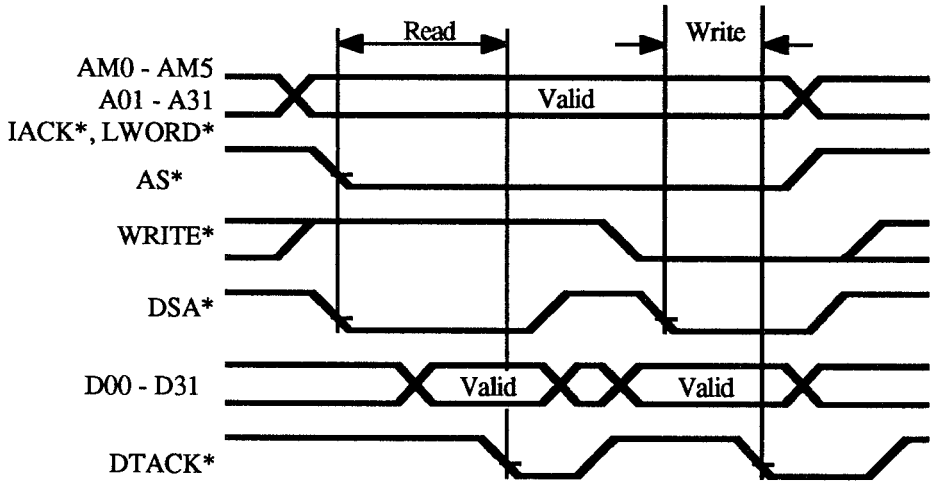


Singola lettura



Singola scrittura

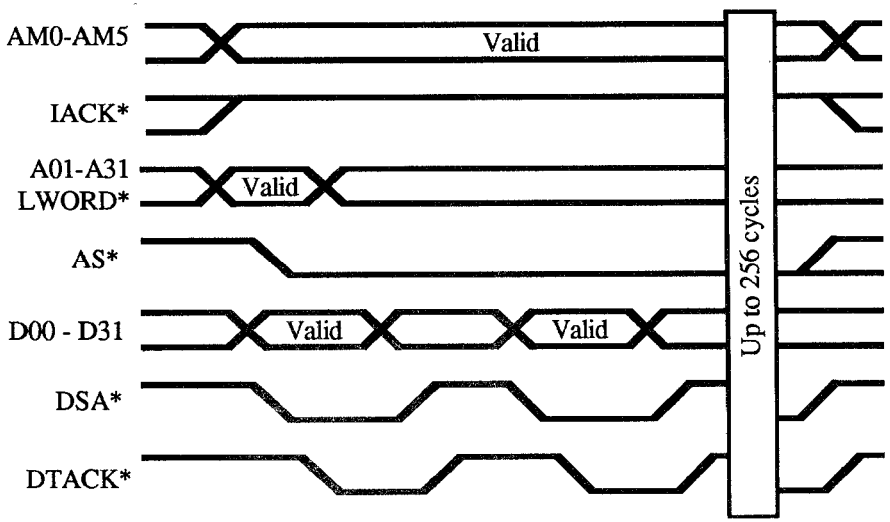
CICLO READ-MODIFY-WRITE



Ciclo RMW: AS* rimane attivo per tutto il ciclo che comprende una lettura seguita da una scrittura

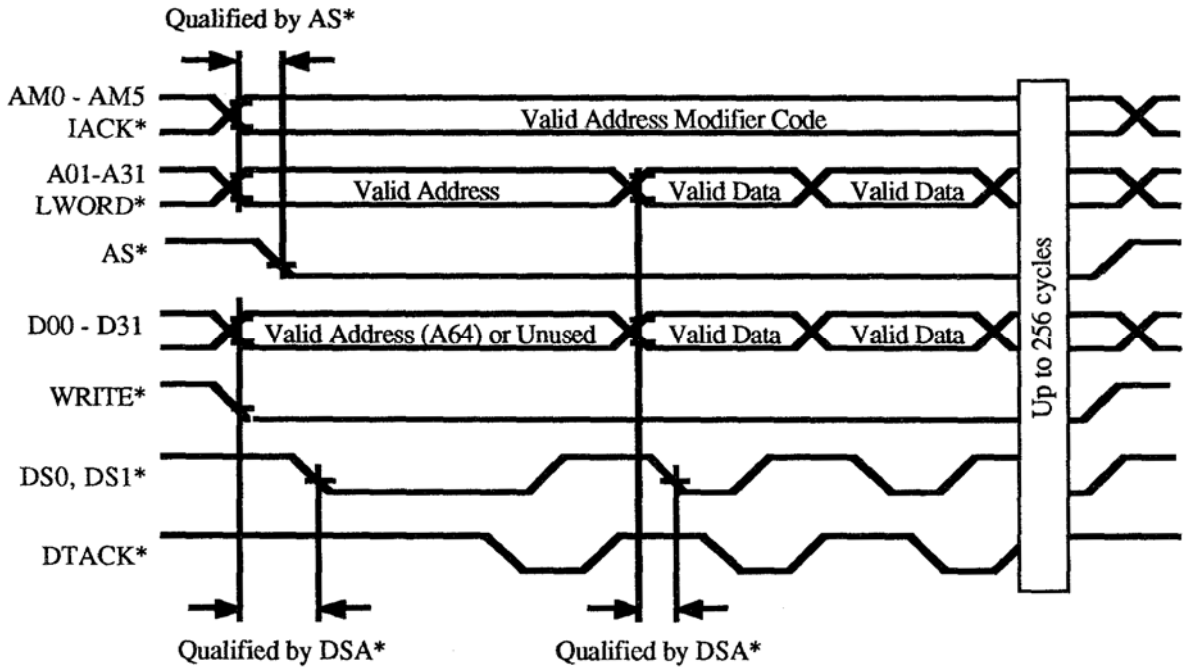
TRASFERIMENTO DATI A BLOCCHI (BLT)

Viene presentato l' indirizzo del primo accesso.
Lo SLAVE incrementa questo valore per accedere a locazioni sequenziali.
Il limite superiore dell' indirizzo e' a modulo 256, quindi il trasferimento e' limitato ad un massimo di 256 bytes.
Lo SLAVE risponde DTACK ad ogni trasferimento.
Se lo SLAVE genera BERR il trasferimento si interrompe.



Scrittura a blocchi

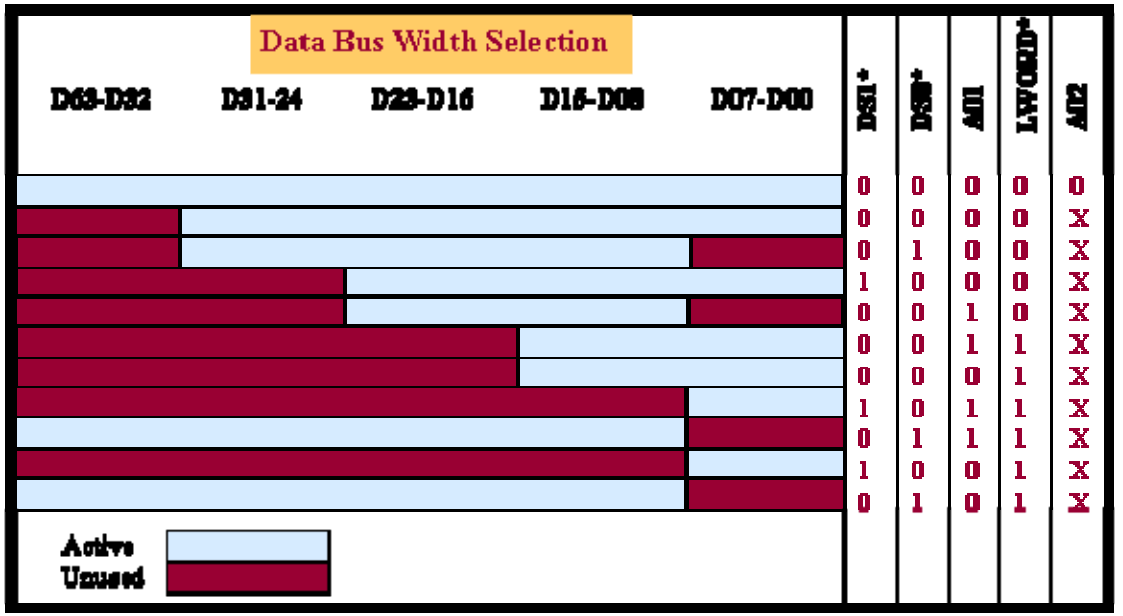
TRASFERIMENTI A 64 BIT



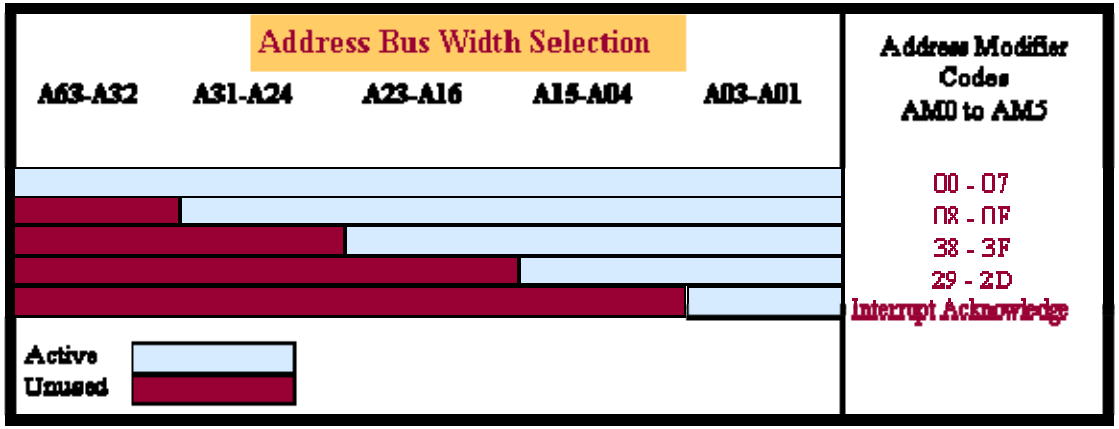
Scrittura a blocchi a 64 bit (MBLT)
Viene definita dalle specifiche VME64.
La lettura e' analoga: lo slave assicura i dati validi sul fronte di discesa di DTACK*

DIFFERENTI CICLI DI BUS

La dimensione del bus dati viene individuata da: DS0*, DS1*, A01, LWORD*



La dimensione del bus indirizzi dipende dal codice AM



CODICI AM

0x3F	A24 supervisor block transfer (BLT)
0x3E	A24 supervisor program access
0x3D	A24 supervisor data access
0x3C	A24 supervisor 64 bit block transfer (MBLT)
0x3B	A24 non privileged block transfer (BLT)
0x3A	A24 non privileged program access
0x39	A24 non privileged data access
0x38	A24 non privileged 64 bit block transfer (MBLT)
0x37	A40 block transfer (BLT)
0x36	Reserved
0x35	A40 lock command (LCK)
0x34	A40 access
0x33	Reserved
0x32	A24 lock command (LCK)
0x31 - 0x30	Reserved
0x2F	CR/CSR space
0x2E	Reserved
0x2D	A16 supervisory access
0x2C	A16 lock command (LCK)

0x2B - 0x2A	Reserved
0x29	A16 non privileged access
0x28 - 0x20	Reserved (some used for 2eVME & 2eSST)
0x1F - 0x10	User defined
0x0F	A32 supervisor block transfer (BLT)
0x0E	A32 supervisor program access
0x0D	A32 supervisor data access
0x0C	A32 supervisor 64 bit block transfer (MBLT)
0x0B	A32 non privileged block transfer (BLT)
0x0A	A32 non privileged program access
0x09	A32 non privileged data access
0x08	A32 non privileged 64 bit block transfer (MBLT)
0x07 - 0x6	Reserved
0x5	A32 lock command (LCK)
0x4	A64 lock command (LCK)
0x3	A64 block transfer (BLT)
0x2	Reserved
0x1	A64 single transfer access
0x0	A64 64 bit block transfer (MBLT)

INTERRUPTS (1)

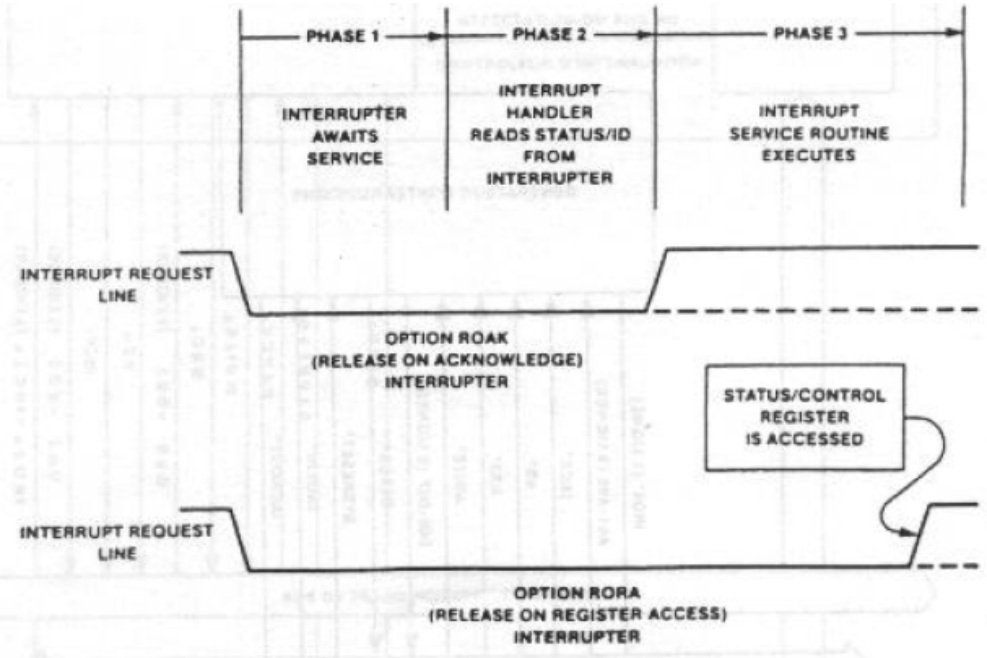
Un generico SLAVE richiede l' attenzione del MASTER attivando una linea IRQx*. IRQ7* ha la massima priorita'. IRQ1* la minima. Le linee sono condivise tra tutti gli SLAVE.
Il MASTER accetta l' interruzione e attiva la linea IACK*.
La linea IACK* e' collegata allo slot 1 al segnale IACKin*. IACKout* dello slot 1 e' collegato a IACKin* dello slot 2 e cosi' via.
Lo slave che ha generato l' interruzione deve attendere sia IACK* che IACKin*, per rispondere al ciclo di riconoscimento.
A questo punto lo slave deve fornire un ID (vettore) al master in modo da fargli sapere come gestire l' interrupt.
Gli slave che non hanno generato interruzioni dello stesso livello propagano IACKin* su IACKout*.

INTERRUPTS (2)

Gli SLAVE possono essere di 2 tipi:

ROAK (Release On Interrupt AKnowledge)

RORA (Release On Register Access)

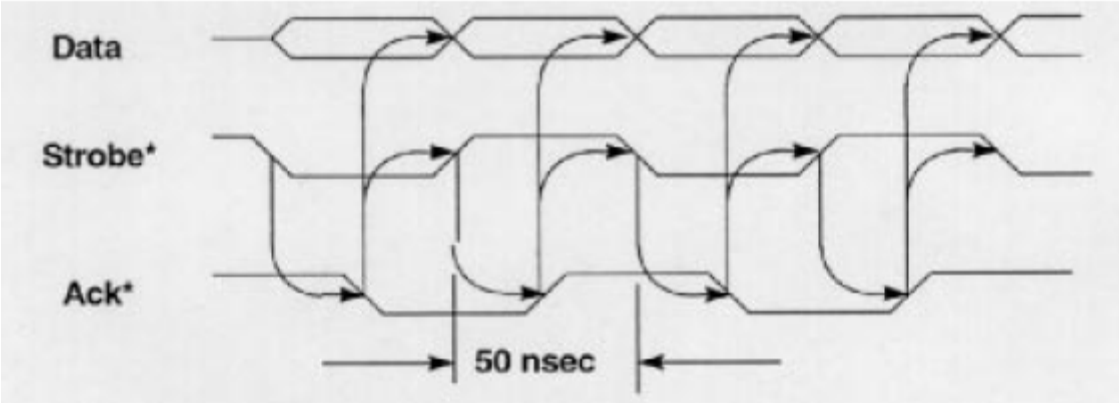


ESTENSIONE: CR/CSR

Il VME64 definisce lo spazio d' indirizzamento CR/CSR (Configuration Rom/ Control Status Register) con AM = 0x2F.
Lo spazio CR/CSR viene gestito con cicli A24 per compatibilita' tra schede 3U e 6U.
Ogni scheda nel crate ha a disposizione fino a 512KB di spazio CR/CSR.
I 5 bit alti dell' indirizzo A24 identificano la scheda con metodi non definiti.
Nella CR vengono rese disponibili informazioni tipo: ID costruttore, ID scheda, revisione software, …
Il CSR puo' venire usato per resettare la scheda e per leggere linee di stato.

ESTENSIONE: 2eVME

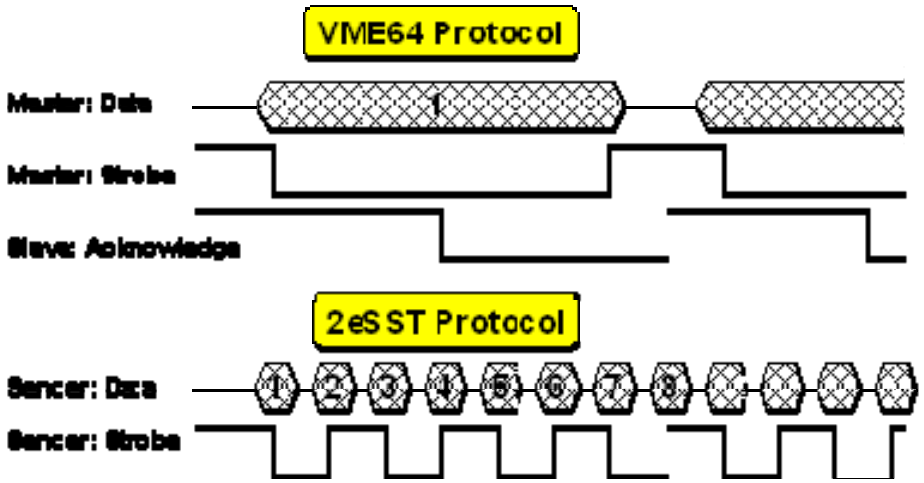
Definita nelle specifiche VME64x (2 edge VME), permette un raddoppio della velocita' di trasferimento dati rispetto al MBLT.
Il trasferimento dati viene effettuato a 64 bit (unita' 6U) o a 32 bit (unita' 3U).
Il trasferimento inizia con tre cicli di indirizzamento e fino a 256 cicli dati.
Si possono trasferire fino a 2 Kbytes per volta.
L' intero ciclo puo' essere interrotto sia dal MASTER che dallo SLAVE in varie condizioni.



ESTENSIONE: 2eSST

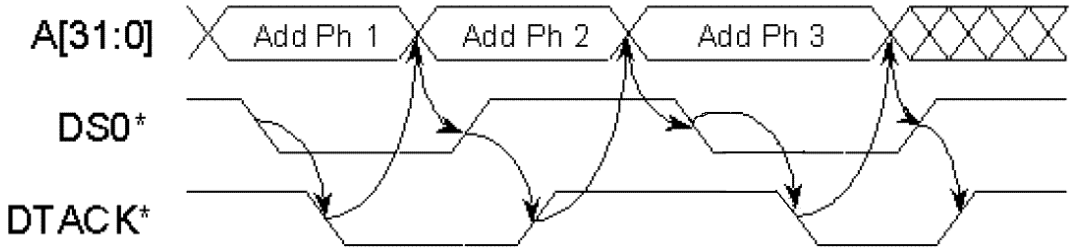
Definita in ANSI VITA 1.5-1999 (2 edge Source Synchronous Transfer).
E' simile al 2eVME, ma usa trasferimenti sincroni, senza handshake.
Le fasi di indirizzamento sono identiche a quelle 2eVME, con differenti AM (XAM).
Vengono definiti 6 possibili cicli a diverse velocita'.
Si puo' avere una velocita' massima di 320 Mbytes/s.

Per le scritture DS1* e' il clock di riferimento.
Per le letture DTACK* e' il clock di riferimento.



2eVME & 2eSST: 6U Address Phase

	Address phase 1	Address phase 2	Address phase 3	Data phase
AM[5:0]	0x20	0x20	0x20	0x20
A[7:0]	XAM Code	Internal Addr [7:0]	Reserved	D[39:32]
A[15:8]	Device Addr [15:8]	Beat Count	Reserved	D[47:40]
A[23:16]	Device Addr [23:16]	A[23:21]=0 A[20:16] = Master GA	Reserved	D[55:48]
A[31:24]	Device Addr [31:24]	Master SubUnit	Reserved	D[63:56]
D[31:0]	Device Addr [63:32] or 0	2eSST Xfer Rate Code	Reserved	D[31:0]



2eVME & 2eSST: XAM codes

Table 11-2 6U 2eVME Extended Address Modifier Codes

XAM Code	Function
00	Reserved
01	A32/D64 2eVME Transfer
02	A64/D64 2eVME Transfer
03-FF	Reserved

Table 11-3 3U 2eVME Extended Address Modifier Codes

XAM Code	Function
00	Reserved
01	A32/D32 2eVME Transfer
02	A40/D32 2eVME Transfer
03-FF	Reserved

From VME64x specifications

Table 2.4: Extended Address Modifier Code

Address Modifier Code Implementation	Extended Address Modifier Code	Address/Data Mode
0x20 6U	0x11	A32/D64 2eSST
	0x12	A64/D64 2eSST
	0x21	A32/D64, Broadcast 2eSST
	0x22	A64/D64, Broadcast 2eSST
0x21 3U	0x11	A32/D32 2eSST
	0x12	A40/D32 2eSST
	0x21	A32/D32, Broadcast 2eSST
	0x22	A40/D32, Broadcast 2eSST

Table 2.5: 6U 2eSST Transfer Rates

Transfer Rate Code	6U Transfer Rate (MB/s)	Nominal Strobe Width (ns)	Maximum Strobe Frequency (MHz)	Minimum Strobe Period (ns)	Nominal Strobe Duty Cycle (%)	Mnemonic
0x0	160	50	10	100	50	SST160
0x1	267	30	16.67	60	50	SST267
0x2	320	25	20	50	50	SST320
0x3-0xF	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved

Table 2.6: 3U 2eSST Transfer Rates

Transfer Rate Code	3U Transfer Rate (MB/s)	Nominal Strobe Width (ns)	Maximum Strobe Frequency (MHz)	Minimum Strobe Period (ns)	Nominal Strobe Duty Cycle (%)	Mnemonic
0x0	80	50	10	100	50	3USST80
0x1	133	30	16.67	60	50	3USST133
0x2	160	25	20	50	50	3USST160
0x3-0xF	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved

ESTENSIONI: CBLT e MCST

Definite in ANSI/VITA 23-1998 (VME64xP).
CBLT (Chained Block Transfers): legge dati da tutti i moduli coinvolti, in serie.
MCST (Multicast): trasferisce comandi a tutti i moduli in parallelo.
Gli 8 bit alti dell' indirizzo A32 individuano i moduli coinvolti.

CBLT:
Puo' essere D32 oppure D64.
Viene definito un meccanismo di "token passing" (tramite IACKIN*-IACKOUT*) per passare il controllo del trasferimento dati da un modulo al successivo.
L' ultimo modulo termina il trasferimento con un BERR*.

VME BUS PINOUT J1/P1 CONNECTOR

Pin	Signal	Signal	Signal
	Row A	Row B	Row C
1	D00	BBSY*	D08
2	D01	BCLR*	D09
3	D02	ACFAIL*	D10
4	D03	BG0IN*	D11
5	D04	BG0OUT*	D12
6	D05	BG1IN*	D13
7	D06	BG1OUT*	D14
8	D07	BG2IN*	D15
9	GND	BG2OUT*	GND
10	SYSCLK	BG3IN*	SYSFAIL*
11	GND	BG3OUT*	BERR*
12	DS1*	BRO*	SYSRESET*
13	DS0*	BR1*	LWORD*
14	WRITE*	BR2*	AM5
15	GND	BR3*	A23
16	DTACK*	AM0	A22

Pin	Signal	Signal	Signal
	Row A	Row B	Row C
17	GND	AM1	A21
18	AS*	AM2	A20
19	GND	AM3	A19
20	IACK*	GND	A18
21	IACKIN*	SERCLK	A17
22	IACKOUT*	SERDAT*	A16
23	AM4	GND	A15
24	A07	IRQ7*	A14
25	A06	IRQ6*	A13
26	A05	IRQ5*	A12
27	A04	IRQ4*	A11
28	A03	IRQ3*	A10
29	A02	IRQ2*	A09
30	A01	IRQ1*	A08
31	-12v	+5v Standby	+12v
32	+5v	+5v	+5v

VME BUS PINOUT J2/P2 CONNECTOR

Pin	Signal	Signal	Signal	Pin	Signal	Signal	Signal
	Row A	Row B	Row C		Row A	Row B	Row C
1	User Defined	+5v	User Defined	17	User Defined	D19	User Defined
2	User Defined	GND	User Defined	18	User Defined	D20	User Defined
3	User Defined	RETRY	User Defined	19	User Defined	D21	User Defined
4	User Defined	A24	User Defined	20	User Defined	D22	User Defined
5	User Defined	A25	User Defined	21	User Defined	D23	User Defined
6	User Defined	A26	User Defined	22	User Defined	GND	User Defined
7	User Defined	A27	User Defined	23	User Defined	D24	User Defined
8	User Defined	A28	User Defined	24	User Defined	D25	User Defined
9	User Defined	A29	User Defined	25	User Defined	D26	User Defined
10	User Defined	A30	User Defined	26	User Defined	D27	User Defined
11	User Defined	A31	User Defined	27	User Defined	D28	User Defined
12	User Defined	GND	User Defined	28	User Defined	D29	User Defined
13	User Defined	+5v	User Defined	29	User Defined	D30	User Defined
14	User Defined	D16	User Defined	30	User Defined	D31	User Defined
15	User Defined	D17	User Defined	31	User Defined	GND	User Defined
16	User Defined	D18	User Defined	32	User Defined	+5V	User Defined

VME64x BUS PINOUT EXTENSIONS

Uso di connettori a 5 file (160 pin).
Le 3 file interne mantengono il pinout originale.
Le file esterne aggiungono connessioni di GND, alimentazione a 3.3V (e ausiliarie), indirizzamento geografico, controlli e alimentazioni per live insertion.
E' possibile inserire connettori a 96 pin in un backplane VME64x.

Viene definito un connettore ausiliario (J0/P0) 19 x 7 pin, con passo metrico (2 mm).
Le file esterne sono connesse a GND e tutti i pin interni sono "User Definable".

COMPONENTI PER IL BUS VME

Slave Controllers:

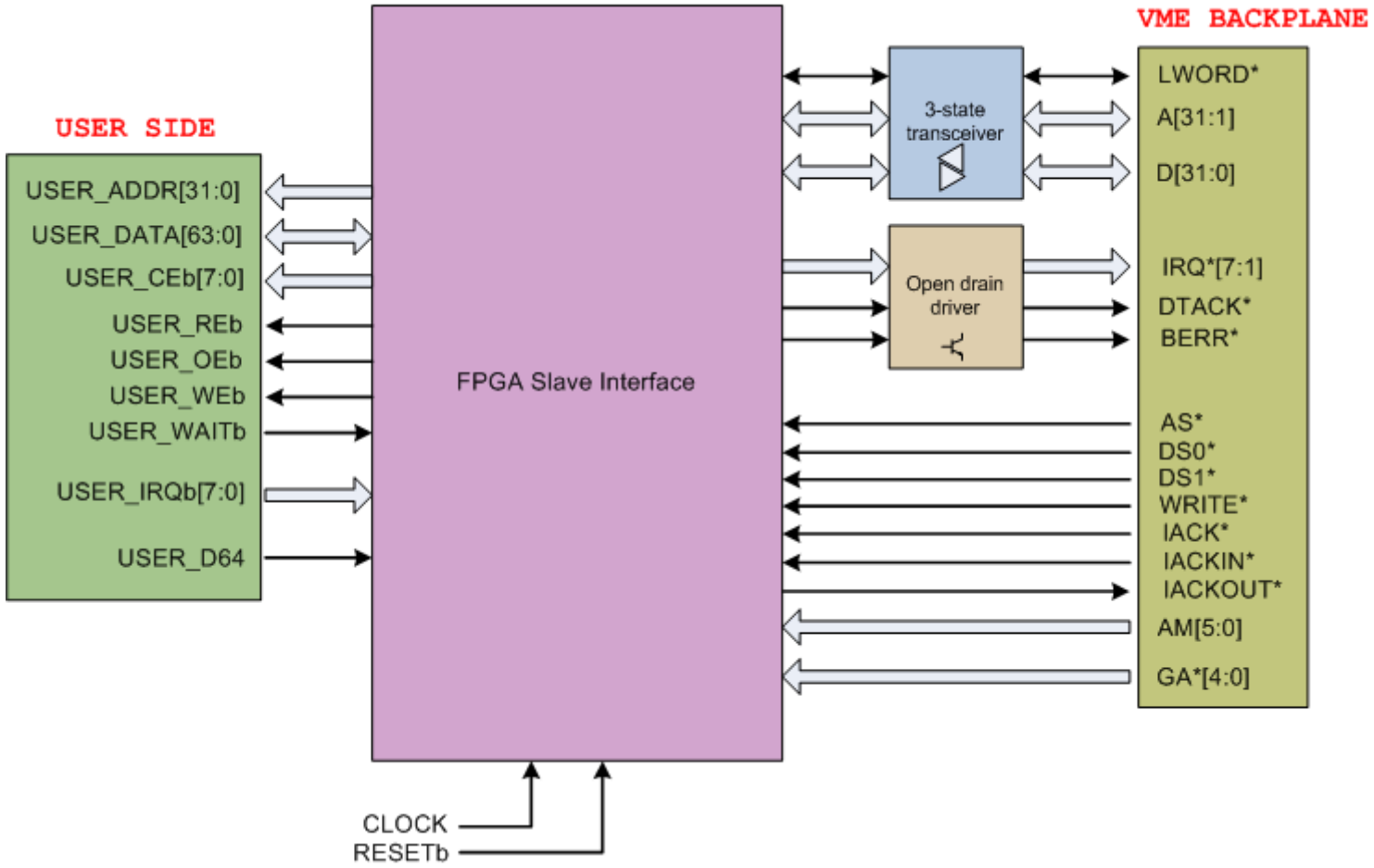
CY7C960/961, VIC068A, VIC64, SCV64

FPGA IP cores, Custom cores

Interfacce elettriche verso il backplane:

SN74VMEH22501A	8-bit + 2 × 1-bit Universal Bus Transceiver (up to VME320)
74FCT162244	16-bit Bus driver, balanced output and reduced ground bounce
74FCT162245	16-bit Bus transceiver, idem
74FCT162543	16-bit Bus latch, idem

ESEMPIO DI INTERFACCIA UTENTE …



…FUNZIONAMENTO

Definizione di 8 spazi di indirizzamento programmabili da 8 Mwords ciascuno (USER_CEb[7:0]).
Dimensione del bus dati selezionabile: 32 bit o 64 bit (USER_D64).
Segnali di lettura e scrittura (USER_REb, USER_Web) sincroni con il CLOCK, attivi 1 periodo di clock ⟹ possibilita' di collegamento di dispositivi sincroni e asincroni.
L' utente puo' ritardare le operazioni (USER_WAITb).
L' utente puo' generare interrupts (USER_IRQb[7:0]).

L'interfaccia supporta trasferimenti A32D32 singoli e a blocchi (BLT), 2eVME, 2eSST.
Implementa zona CR/CSR standard Vme64.

Modello in Verilog HDL: oltre 1000 righe di codice…:

Notizie utili

Questa presentazione puo' essere trovata sul web:
www.ge.infn.it/~musico/Vme/VME_Torino.pdf
Nella pagina: www.ge.infn.it/~musico/VmeStuff.html si puo' trovare
una raccolta di specifiche pubbliche, tutorials e application notes.

Domani mattina, ci sara' la seconda puntata orientata alle applicazioni pratiche,
con dimostrazione d'uso di un sistema VME da laboratorio,
visualizzazione dei cicli di bus e misure di velocita' di trasferimento.

INTERFACCIA VME64x SLAVE

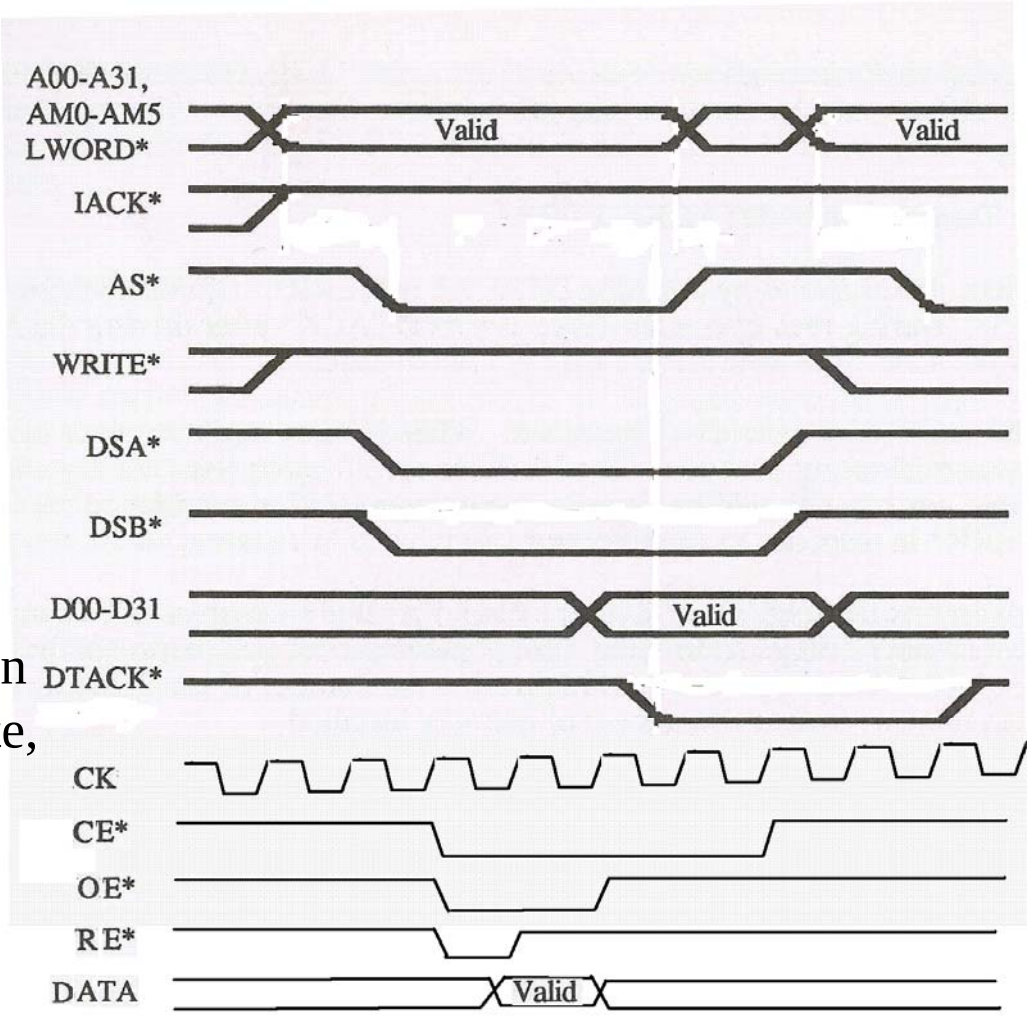
Decodifica un subset di tutti i possibili cicli VME.
Le specifiche impongono l' implementazione dello spazio CR/CSR, nel quale sono contenute le caratteristiche dell' unita'.
Si accede allo spazio CR/CSR con AM = 0x2F (A24D32).
In aggiunta decido cosa decodificare:

- A32D32 Single
- A32D32 Block transfer
- A32D64 Block transfer (MBLT)
- A32D32 2eVme
- A32D32 2eSST

Trasformazione dei cicli VME in cicli piu' semplici, sincroni.
Rimappatura degli indirizzi VME e decodifica di 8 dispositivi indipendenti.

TRASLAZIONE SEGNALI

Usando CK decodifico il ciclo
e genero CE*.
RE* dura sempre 1 ciclo di CK.
OE* dura sempre 2 cicli di CK.
In caso di lettura sequenziale
(a blocchi) CE* e OE* rimangono
attivi per tutto il trasferimento.
DATA viene “latchato” e copiato
sul bus.
L’ indirizzo VME viene caricato in un
contatore e reso disponibile all’ utente,
incrementato ogni ciclo.
Il ciclo di scrittura e’ analogo:
viene generaro WE* invece che la
coppia OE* RE*.



SEQUENZA OPERATIVA

- Sincronizzare tutti gli ingressi usando CK abbastanza veloce (regola generale).
- Caricare A_VME in un contatore ogni fronte di discesa di AS*.
- In parallelo decodificare il cicli usando AM, IACK*.
- Decodificare le uscite del contatore di indirizzi e generare CE* (usando anche le info ADER contenute in CR/CSR).
- Generare OE* e RE* oppure WE* in funzione del ciclo.
- Generare DTACK* o BERR*,
- La sequenza viene resettata ad ogni fronte di salita di AS*.

Vengono generate 8 linee CE* indipendenti.
Vengono generati gli interrupt (se abilitati) e viene gestito il ciclo di acknowledge ROAK.