UNIVERSITÀ DEGLI STUDI DI FERRARA

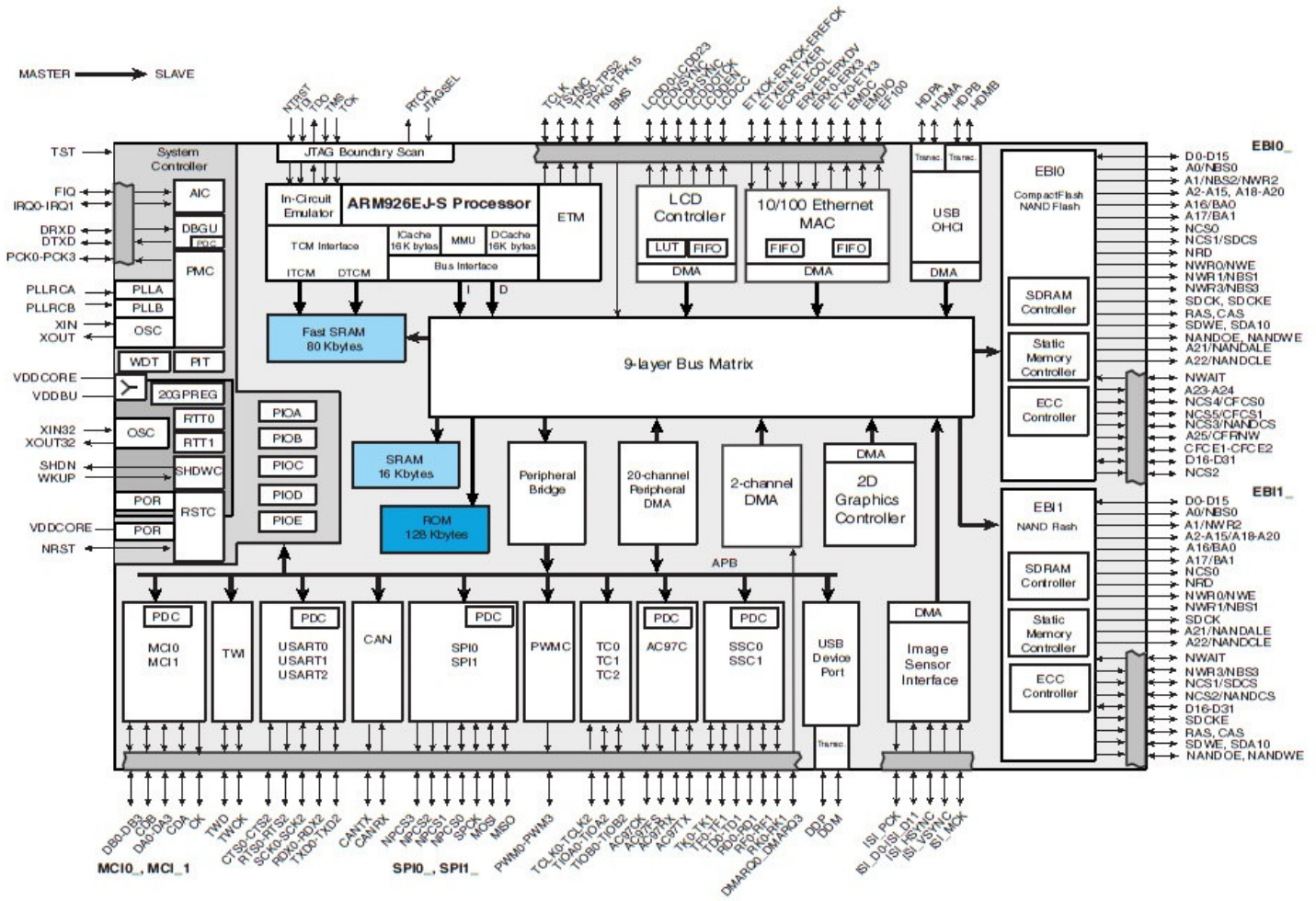
Corso di
SISTEMI DI ELABORAZIONE

ARM 9 – LAN91C111
Panoramica su architettura ARM e interfacciamento con controllore ethernet

A cura di Maurizio Bottaro

ARM 9 – AT91SAM9263 – ATMEL

SCHEMA A BLOCCHI



ARM 9 – AT91SAM9263 – ATMEL

CONCETTO DI SoC – System on Chip

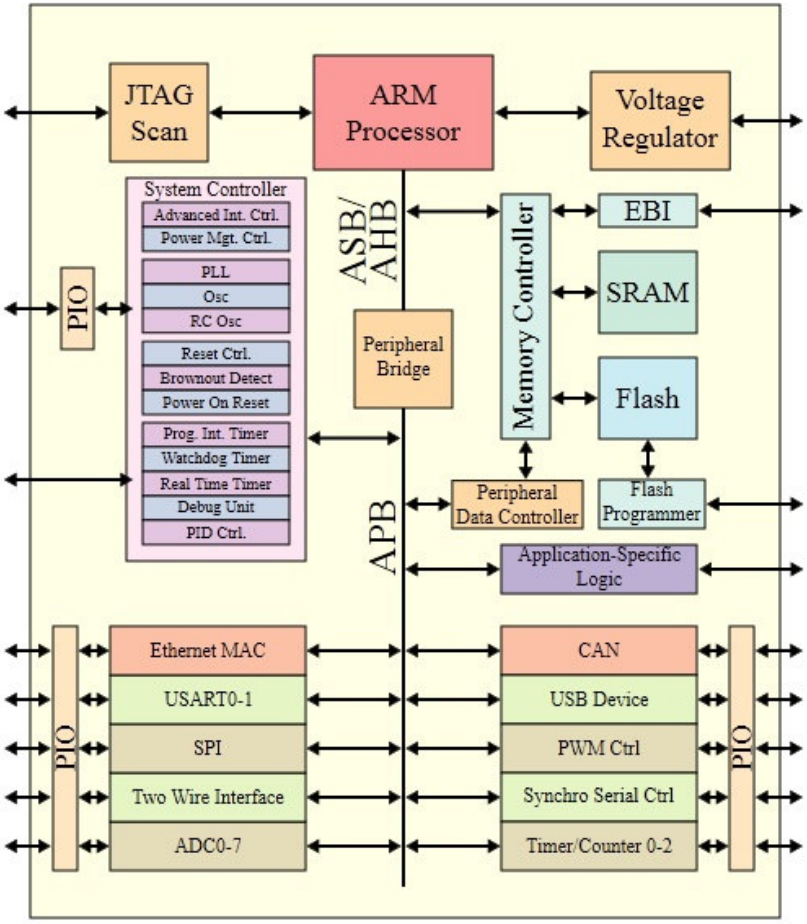
Un SoC è costituito da:

- Uno o più µP o µC
- Flash, RAM o ROM interne
- PLL o generatori di clock
- Convertitori AD o DA
- Connettori standard del tipo USB, Ethernet, CAN, SPI, Bluetooth o Firewire

Questi blocchi, sono collegati da bus. L'ARM utilizza il suo bus che è l'AMBA.
L'obbiettivo è quello di realizzare flussi paralleli di dati che vanno da un blocco all'altro.
Questo è il motivo per qui, anche per le strutture embedded, si sta abbandonando la struttura a single core per passare ad approcci di tipo dual core.

ARM 9 – AT91SAM9263 – ATMEL

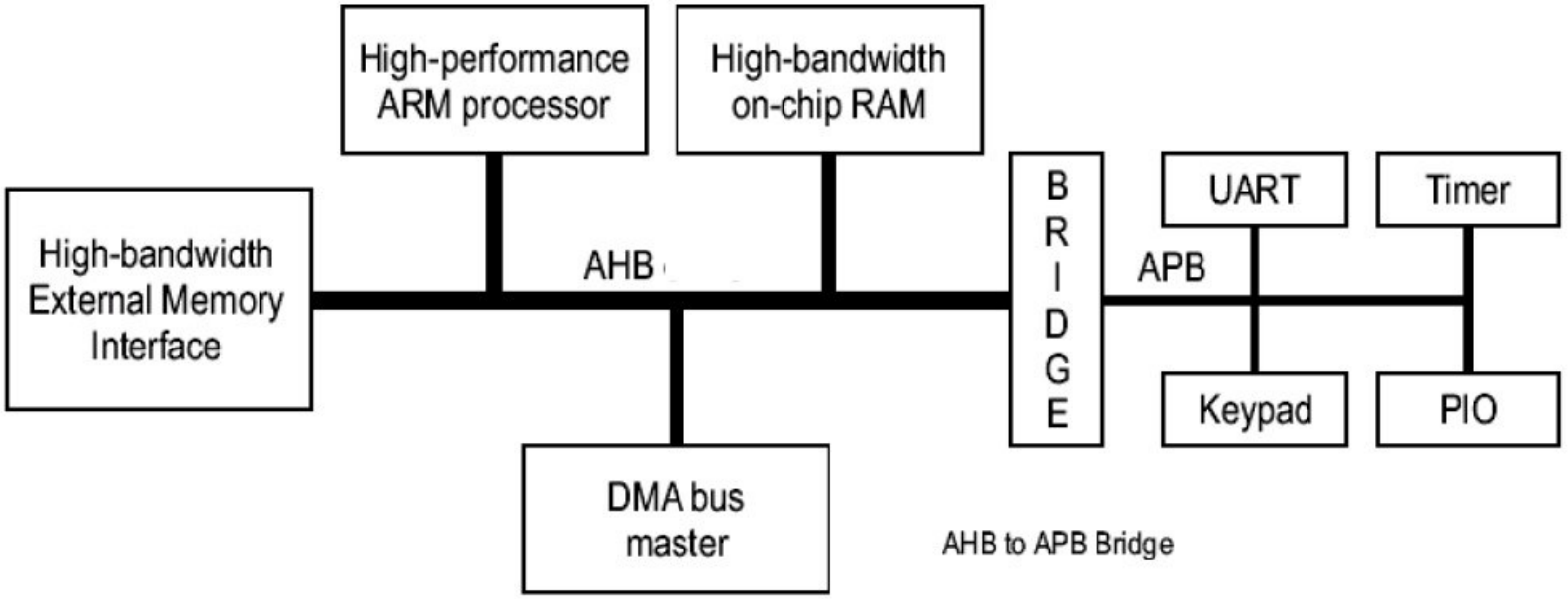
CONCETTO DI SoC – System on Chip



Di fianco di possono vedere i vari componenti di un SoC.
Ogni blocco è connesso sfruttando il bus condiviso AMBA.
Da notare i bus AHB e APB.
L'AHB connette direttamente il µP con un bridge che si occupa di interfacciare il µP con il resto dei moduli integrati.
Il bus periferico APB è dedicato alle periferiche.

ARM 9 – AT91SAM9263 – ATMEL

BUS CONDIVISO AMBA



AMBA: Advanced Microcontroller Bus Architecture
AHB: Advanced High-performance Bus
APB: Advanced Peripheral Bus

ARM 9 – AT91SAM9263 – ATMEL

CHI CI INTERFACCIA TRA MICRO E PERIFERICA? I BUS CONDIVISI!

AMBA AHB

- Alte prestazioni
- Operazioni in pipeline
- Trasferimenti burst
- Trasferimenti split
- Master a bus multipli

AMBA APB

- Bassa potenza
- Controlli e indirizzi 'latchati'
- Interfaccia semplice
- Adattabilità a molte periferiche

Il bus AHB si utilizza quando sono richieste alte prestazioni, quindi trasferimento di informazioni tra dispositivi ad alte frequenze di clock.
Alcuni esempi di applicabilità sono memorie on-chip, memorie esterne o microprocessori.

Il bus APB è un bus a basse prestazioni, quindi basse velocità di trasferimento delle informazioni. Applicabile a periferiche in generale lente.

Esistono bridge che permettono di passare da AHB ad APB. Ma ciò comunque impedirà di sfruttare la struttura in pipeline del bus AHB.

ARM 9 – AT91SAM9263 – ATMEL

DI COSA SI OCCUPA UN BUS CONDIVISO?

Un bus condiviso gestisce lo scambio di informazioni tra vari componenti di un SOC (System on Chip).
I componenti sono suddivisi tra Master e Slave, e il componente che arbitra lo scambio dei dati è definito come ARBITER.

N.B. SE PER QUALCHE MOTIVO, UNA PERIFERICA CONNESSA AD UN BUS INTRODUCE DEGLI STATI DI WAIT, TUTTI GLI ALTRI ASPETTANO!!

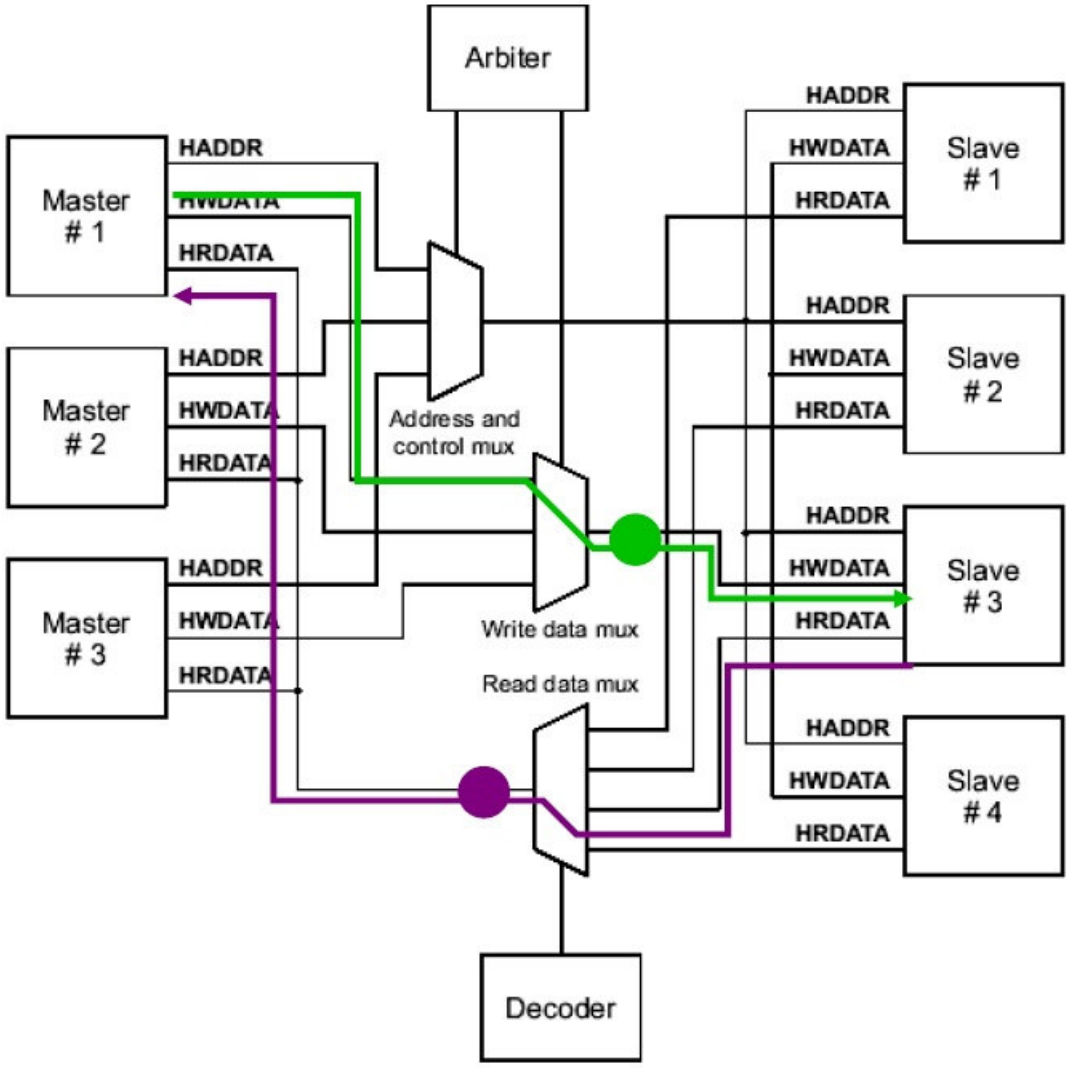
COMPONENTI DEL BUS

INITIATOR: unità che inizia la transazione.
TARGET: chi risponde alla transazione inviata dall'INITIATOR.
MASTER/SLAVE: MASTER è chi invia i comandi, SLAVE è chi obbedisce a tali.
ARBITER: controlla gli accessi al bus.
BRIDGE: connette due bus. Agisce come un TARGET da un lato e come INITIATOR dall'altro.

ARM 9 – AT91SAM9263 – ATMEL

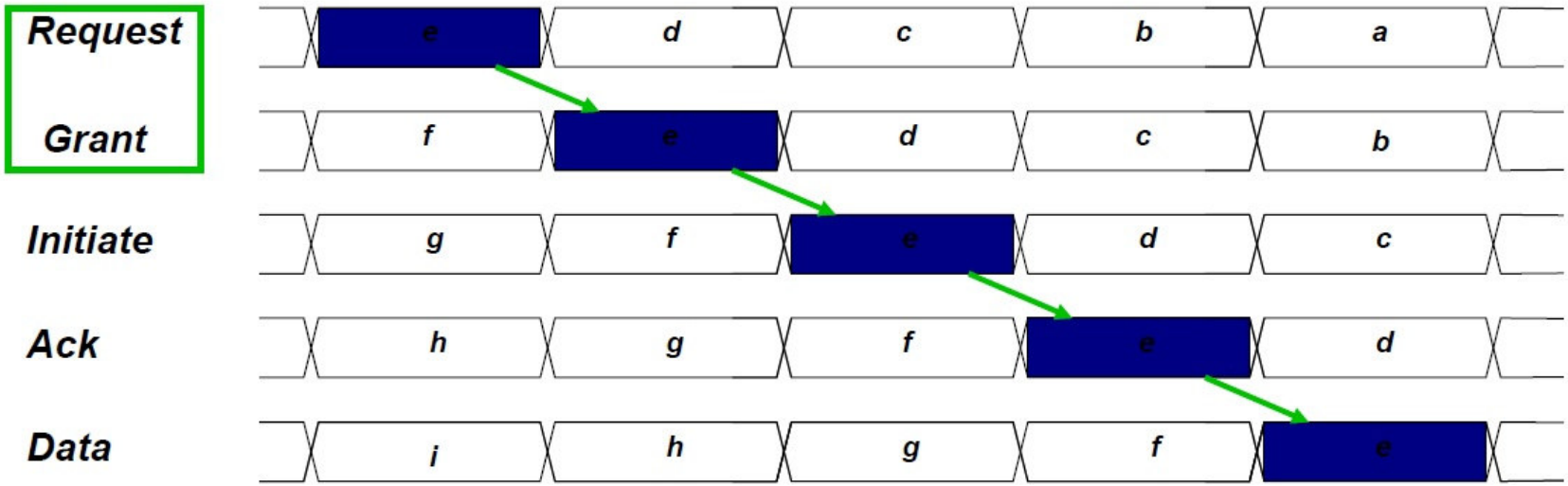
AMBA AHB

- 27 segnali di controllo
- Possibilità di connettere fino a 15 master
- Gestione pipeline di dati e indirizzi



ARM 9 – AT91SAM9263 – ATMEL

TRANSAZIONI IN UN BUS PIPELINE



PREGI

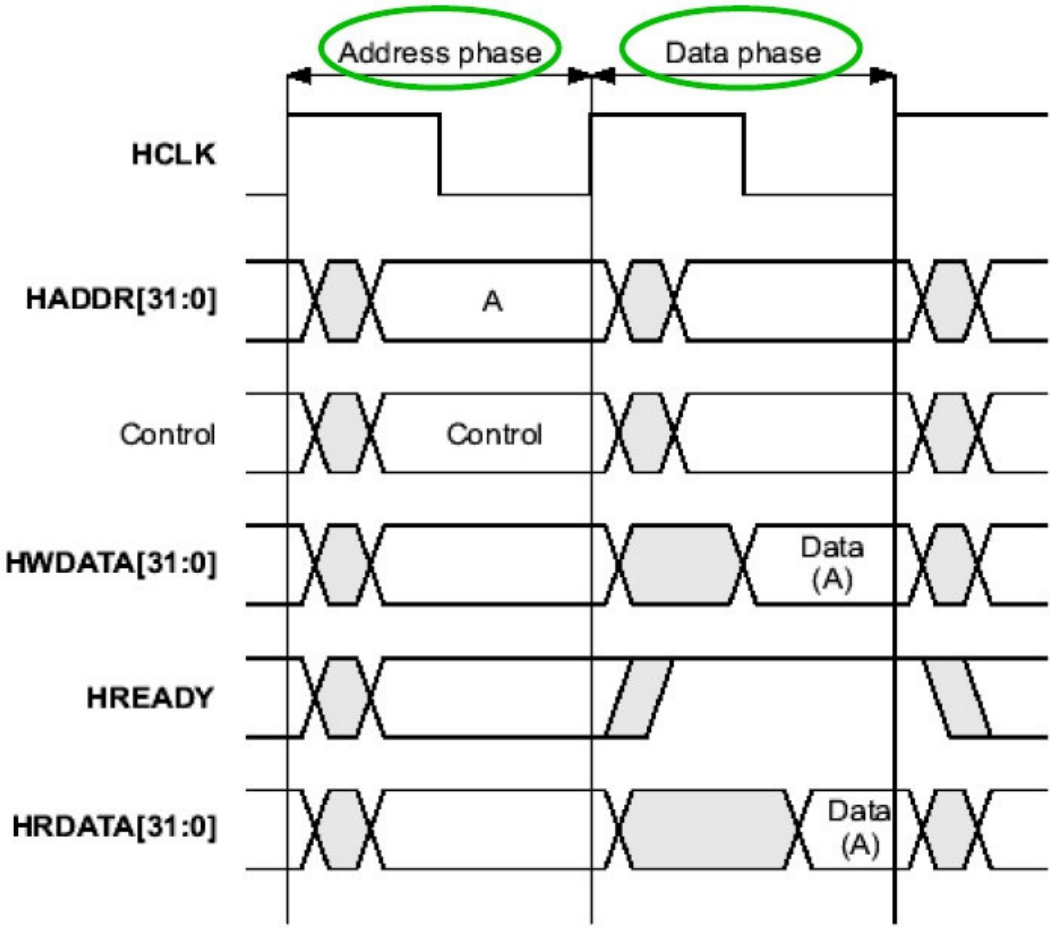
- Posso fare più cose contemporaneamente, nel senso che fin che sto finendo di elaborare un dato posso iniziare ad elaborarne un altro. Nessuno sta fermo. AUMENTA IL THROUGHPUT!!!

DIFETTI

- Struttura estremamente complessa.

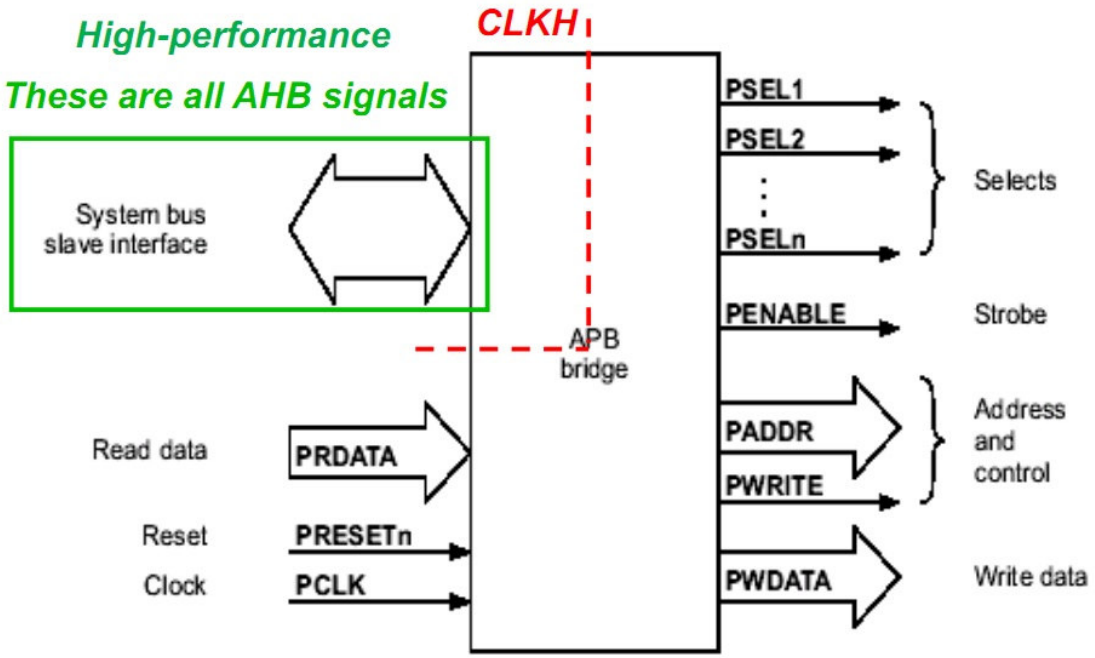
ARM 9 – AT91SAM9263 – ATMEL

TRASFERIMENTO BASE



ARM 9 – AT91SAM9263 – ATMEL

AHB-APB BRIDGE



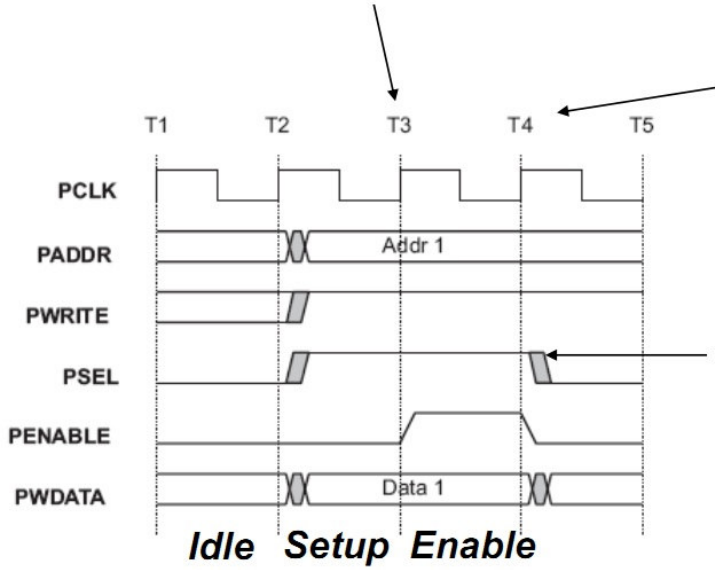
Questo elemento si interessa di interfacciare il bus pipeline AHB con il bus APB a cui sono connesse tutte le periferiche che non necessitano di ampia banda. Esso presenta solo 4 segnali di controllo. Può agire solo un master alla volta e funziona come una macchina a stati.

ARM 9 – AT91SAM9263 – ATMEL

CICLO DI SCRITTURA SU BUS APB

I dati e i segnali di controllo devono essere mantenuti stabili

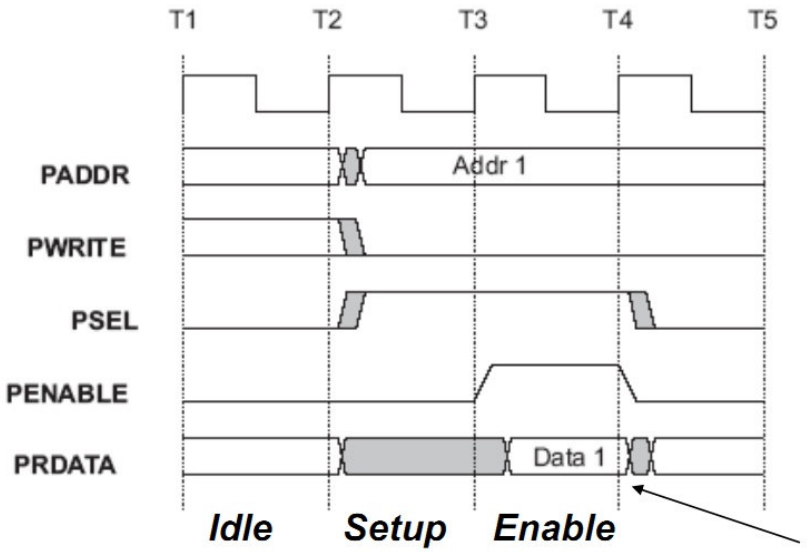
I segnali di controllo possono essere soggetti a glitch



Non cambia il suo stato finché non inizia un nuovo trasferimento sulla stessa periferica

ARM 9 – AT91SAM9263 – ATMEL

CICLO DI LETTURA SU BUS APB



È campionato dal bridge alla fine di un ciclo di enable

ARM 9 – AT91SAM9263 – ATMEL

AMPIA VERSATILITÀ

Questi µC sono utilizzati in moltissimi sistemi embedded di larga distribuzione data la loro estrema versatilità.
Il nostro ARM9 ci permetterà di gestire ad esempio:

- Controller LCD
- Operazioni 2D per applicazioni video
- Touch panel
- USB 2.0
- Ethernet 10/100 Base-T
- Batteria esterna per backup registri (tipo batteria tampone per mantenere le impostazioni del BIOS nei PC)
- Periferiche CAN
- Multimedia Card
- Audio AC97
- USART - IrDA

Esso è programmabile via JTAG.

ARM 9 – AT91SAM9263 – ATMEL

ALIMENTAZIONI

- 1.2V (da 1.08V a 1.32V) per la CPU, le memorie e le periferiche
- 1.8V (da 1.65V a 1.95V) o 3.3V (da 3.0V a 3.6V) per EBI0/EBI1 che sono i due bus esterni che consentono la gestione di hard disk IDE
- 3.3V (da 2.7V a 3.6V) per le linee I/O e le linee USB
- 1.8 – 2.5 – 3 – 3.3V per le linee I/O corrispondenti all'interfaccia dei sensori per l'immagine
- 1.2V (da 1.08V a 1.32V) per l'oscillatore interno del clock e per il controller del sistema
- 3.3V (da 3.0V a 3.6V) nominali per le celle del PLL
- 3.3V (da 3.0 a 3.6V) nominali per l'oscillatore principale

CONSUMI

CONSUMI STATICI

Nel caso peggiore i consumi di statici di corrente si attestano sui 700 uA @ 25°C.
Se si raggiungono temperature di 85°C si possono raggiungere i 7 mA.

CONSUMI DINAMICI

Si attestano a 70 mA con VDDCORE=1.2 V @ 25°C e con prestazioni massime di calcolo.

ARM 9 – AT91SAM9263 – ATMEL

Perché è importante parlare di consumi e alimentazioni?

Analizziamo la sezione alimentazioni. I voltaggi richiesti non sono mai 5V o 12V.

Questo cosa ci fa capire?
Ci fa capire che questo tipo di µC non è idoneo per la gestione di sistemi industriali. Infatti i sistemi industriali hanno tensioni di lavoro che di norma sono 5-12-24 Vdc.

E se analizziamo i consumi?
Dato il basso consumo statico del µC, è facilmente intuibile che esso può essere utilizzato per applicazioni low-power, quindi dispositivi portatili.
Ecco perché tale integrato si presta bene all'integrazione in sistemi cellulari o palmari che sono alimentati a batteria.

ARM 9 – AT91SAM9263 – ATMEL

È un microcontrollore RISC a 32 bit, basato sul μP ARM926EJ-S.

- Come tutti i μC, esso ha un'architettura di tipo Load/Store
- Ha registri a 16 e 32 bit
- L'operation code ha lunghezza fissa
- La maggior parte delle istruzioni è eseguita in un solo ciclo di clock

Perché è un μC innovativo?

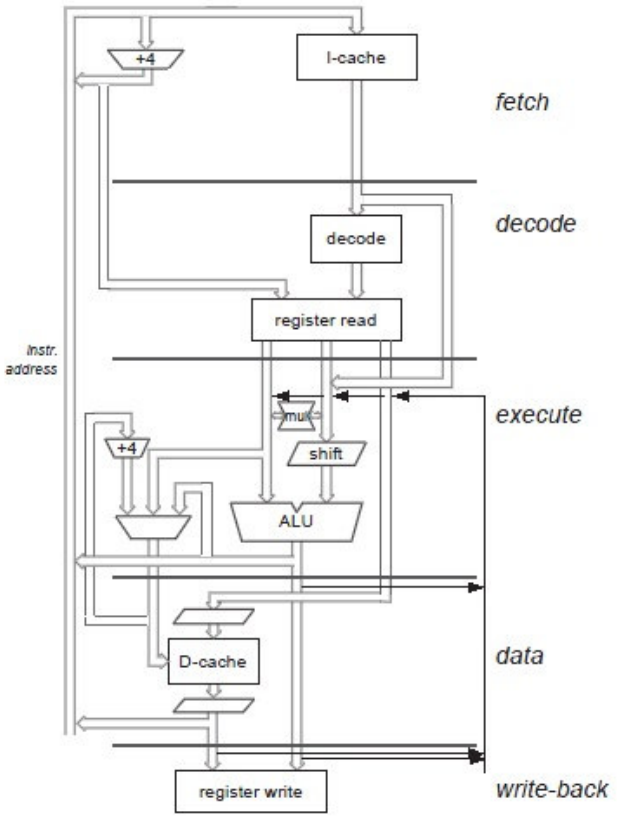
- La sua peculiarietà sta nella struttura delle istruzioni. Con bit opzionali aggiunti nella sintassi delle istruzioni, si possono realizzare codici condizionali per ogni istruzione
- Nel caso di 'if', tali tipi di codici evitano di compiere salti

ARM 9 – AT91SAM9263 – ATMEL

STADI PIPELINE – ELABORAZIONE

Sono presenti 5 stadi di pipeline:

- Fetch
- Decode
- Execute
- Data Memory
- Register Write



ARM 9 – AT91SAM9263 – ATMEL

SET DI ISTRUZIONI SUPPORTATI

THUMB

Set di istruzioni a 16 bit che utilizza 4 bit per ogni istruzione. È un codice leggero e solo i salti possono essere condizionati. Va bene per sistemi a ridotta larghezza di banda, quindi ove sono richieste prestazioni non irresistibili.
Quindi con set ridotti di istruzioni troveremo vantaggi nella gestione di periferiche semplici che si traducono in maggior velocità di esecuzione.

JAZELLE

È possibile eseguire nativamente il Java Bytecode. Lo vediamo spessissimo nei telefonini cellulari della Nokia, quindi atto ad eseguire efficacemente le applicazioni Java.

THUMB 2

È un'estensione del THUMB. Eredita tutte le istruzioni del suo predecessore più istruzioni a 32 bit per aumentare la potenza di calcolo.
Rispetto a prima possono essere eseguite esecuzioni condizionate.

ARM 9 – AT91SAM9263 – ATMEL

GESTIONE DI MEMORIE ESTERNE E REGISTRI

È inoltre possibile gestire vari tipi di memoria:

- SDRAM
- Controller ECC
- NAND Flash

I registri complessivi gestibili sono 37:

- 31 registri general purpose @ 32 bit
- 6 registri di stato @ 32 bit

ARM 9 – AT91SAM9263 – ATMEL

INSTRUCTION SET

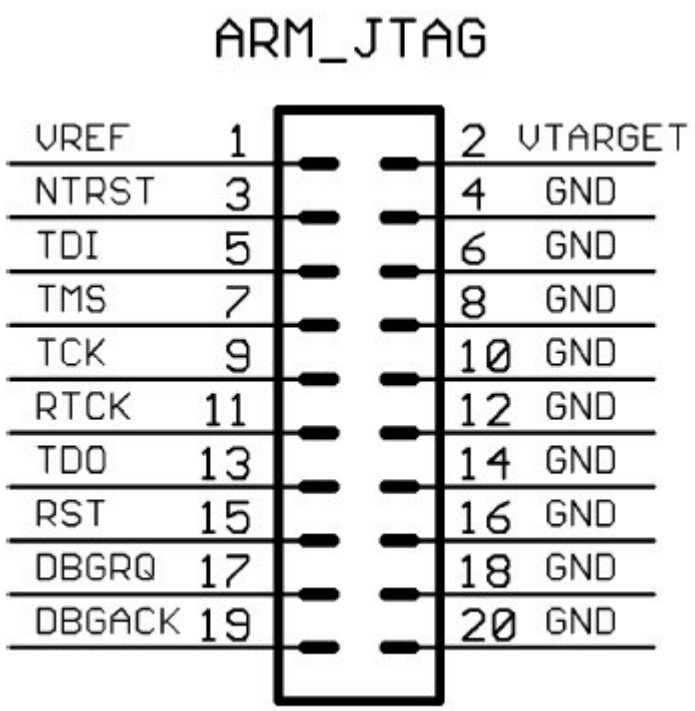
Le categorie in cui è suddiviso l'instruction set sono le seguenti:

- Istruzioni di salto (branch)
- Istruzioni di condizionamento
- Le classiche LOAD/STORE
- Istruzioni di trasferimento per i registri di stato
- Istruzioni per il coprocessore
- Istruzioni per la generazione di eccezioni

ARM 9 – AT91SAM9263 – ATMEL

PROGRAMMAZIONE – JTAG

Il µC implementa lo standard JTAG (IEEE 1149.1).



Di fianco c'è lo schema dei pin di un connettore JTAG. Esso deve essere montato sul PCB e in prossimità del µC (2-3 cm al massimo).

A tale connettore va collegato il sistema di programmazione per ARM.

L'alimentazione è presa direttamente dalla scheda, quindi è necessario che la scheda abbia a bordo un alimentatore stabile e preciso, al fine di evitare errori di scrittura sulla flash del µC.

ARM 9 – AT91SAM9263 – ATMEL

PROGRAMMAZIONE – DESCRIZIONE DEI PIN JTAG

- **VTREF** – Indica al tool di debug la tensione di lavoro del µC
- **VTARGET** – Tensione di lavoro del µC. Il tool di debug sfrutterà questa tensione fornita dalla scheda di controllo
- **nTRST** – Reset JTAG TAP. Necessita di pull-up verso Vcc
- **TDI** – Data in seriale JTAG. Necessita di pull-up verso Vcc
- **TMS** – JTAG TAP Mode Select. Necessita di pull-up verso Vcc
- **TCK** – Clock JTAG
- **RTCK** – Retimed Clock JTAG. Serve eventualmente per sincronizzare input esterni
- **TDO** – Data out seriale JTAG
- **nSRST** – Reset fornibile dall'utente per resettare il µC
- **DBGRQ** – Richiesta di debug asincrona. Un segnale esterno può forzare il µC ad entrare in modalità di debug. Necessita di pull-down verso GND
- **DBGACK** – Debug Acknowledge. Segnale di risposta a DBGRQ
- I pin 4, 6, 8, 10, 12, 14, 16, 18, 20 vanno a GND al fine di aumentare l'immunità ai disturbi in fase di programmazione o debug

ARM 9 – AT91SAM9263 – ATMEL

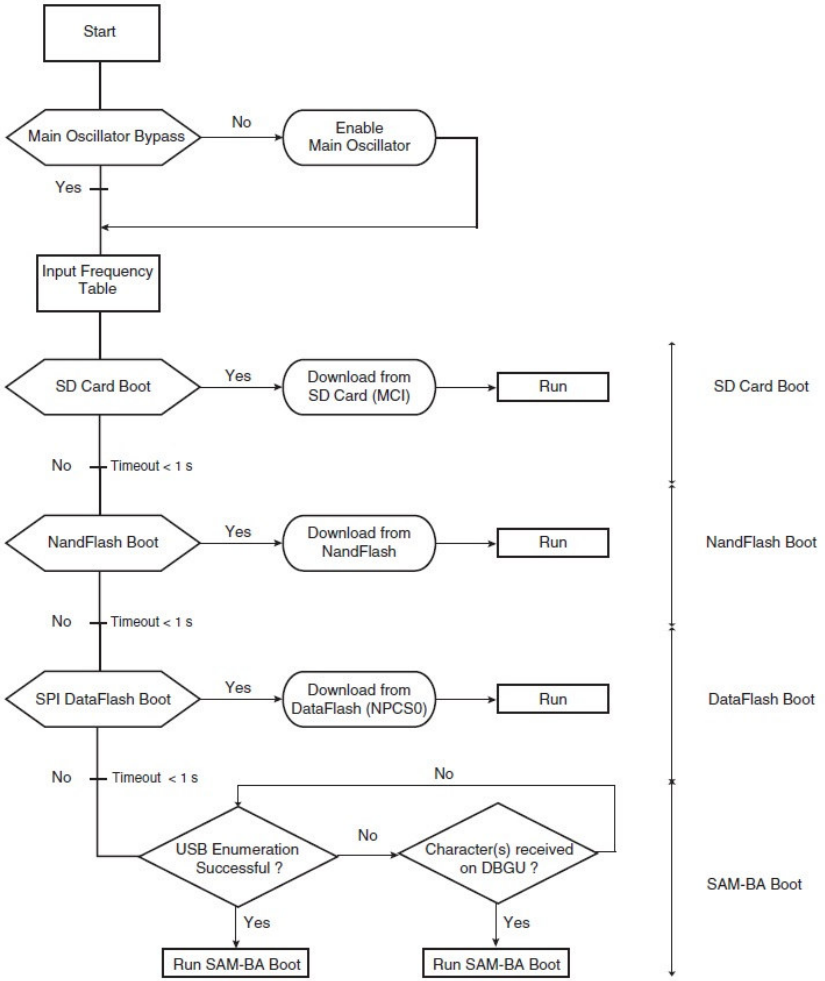
PROGRAMMAZIONE – BOOTLOADER

L'ARM9 è dotato della possibilità di bootloader. Il bootloader è un programma che carica in kernel di un sistema operativo e ne consente l'avvio.
Di fatto, tramite una memoria di massa esterna, è possibile caricare all'interno del µC, i files per la gestione degli I/O del sistema.

Come funziona?
Prima di accendere la scheda, previa opportuna programmazione del µC, si connette una memoria di massa (anch'essa opportunamente configurata) con il kernel del SO.
All'accensione della scheda, il µC, testa se all'ingresso dei pin dedicati al bootloader è presente una memoria di massa dalla quale cercare il kernel.
Al momento del riconoscimento della stessa, il µC caricherà nella sua flash interna i files contenuti nel supporto di massa ed avvierà il sistema operativo in base alle istruzioni appena scaricate.

ARM 9 – AT91SAM9263 – ATMEL

PROGRAMMAZIONE – BOOTLOADER



1. Stabilizzazione oscillatore. Può essere interno o esterno
2. Il µC cerca la memoria di massa che contiene il bootloader
3. L'ultimo step ricerca il bootloader sulla seriale o sulla USB => ISP (In System Programming)

ARM 9 – AT91SAM9263 – ATMEL

PROGRAMMAZIONE – DEBUGGER/PROGRAMMER

Per programmare il µC, è necessario disporre di una opportuna interfaccia PC/scheda di controllo.

Con questo strumento è possibile programmare e debuggare il µC. Per debug si intende poter visualizzare, mentre l'esecuzione del codice nel µC è messa in pausa, il contenuto dei registri interni.
È anche possibile modificare il loro contenuto.
Per poter fare queste operazioni è necessario utilizzare un software che permetta di gestire il debug, possiamo usare un ARM Debugger.

ETHERNET SU ARM9 - ATMEL

L'interfaccia MDIO permette di settare l'attività del DMA (Direct Memory Access), nonché i modi di operazione half-duplex e full-duplex.

Ma come vengono gestiti i dati dalla EMAC alla memoria?
Sono tutti trasferimenti di parole da 32 bit. Possono essere trasferite singolarmente oppure in modalità burst, quindi con un trasferimento è possibile trasferire 2, 3 o 4 parole da 32 bit.

Per la gestione del traffico dei dati è presente una FIFO da 28 byte in ricezione e 28 byte in trasmissione. Nel flusso di I/O passano 4 byte alla volta.

Per la ricezione, il bus manda la richiesta alla FIFO quando ha 12 byte liberi, quindi 3 parole da 4 byte.

Per la trasmissione, il bus manda la richiesta alle FIFO quando ha a disposizione 16 byte, quindi 4 parole da 4 byte.

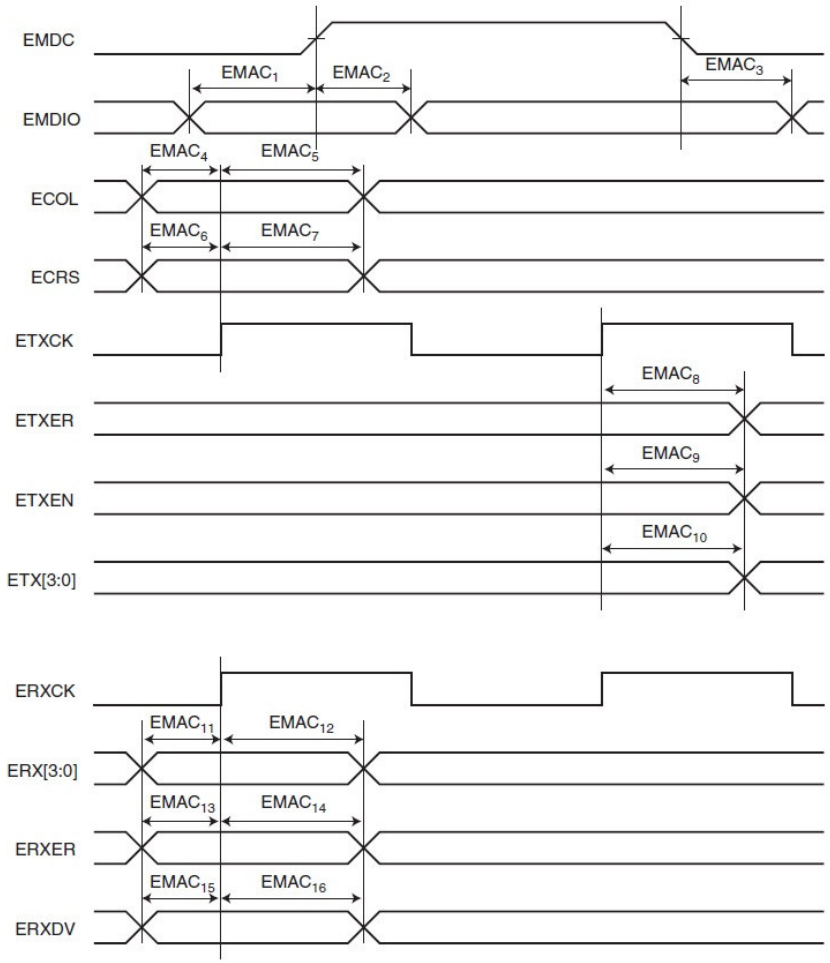
ETHERNET SU ARM9 - ATMEL

Di seguito i segnali dedicati alla gestione.

Signal Name	Function	Type	Active Level	Comments
		Ethernet 10/100		
ETXCK	Transmit Clock or Reference Clock	Input		MII only, REFCK in RMII
ERXCK	Receive Clock	Input		MII only
ETXEN	Transmit Enable	Output		
ETX0-ETX3	Transmit Data	Output		ETX0-ETX1 only in RMII
ETXER	Transmit Coding Error	Output		MII only
ERXDV	Receive Data Valid	Input		RXDV in MII, CRSDV in RMII
ERX0-ERX3	Receive Data	Input		ERX0-ERX1 only in RMII
ERXER	Receive Error	Input		
ECRS	Carrier Sense and Data Valid	Input		MII only
ECOL	Collision Detect	Input		MII only
EMDC	Management Data Clock	Output		
EMDIO	Management Data Input/Output	I/O		
EF100	Force 100Mbit/sec.	Output	High	RMII only

ETHERNET SU ARM9 - ATMEL

TEMPISTICHE



LAN91C111 - SMSC

È controllore integrato che interfaccia il microcontrollore con il livello fisico di una comunicazione ethernet.

Gli standard di comunicazione implementati dall'interfaccia sono:

- IEEE 802.3
- IEEE 802.3u-100Base-TX
- IEEE 802.3u-10Base-T

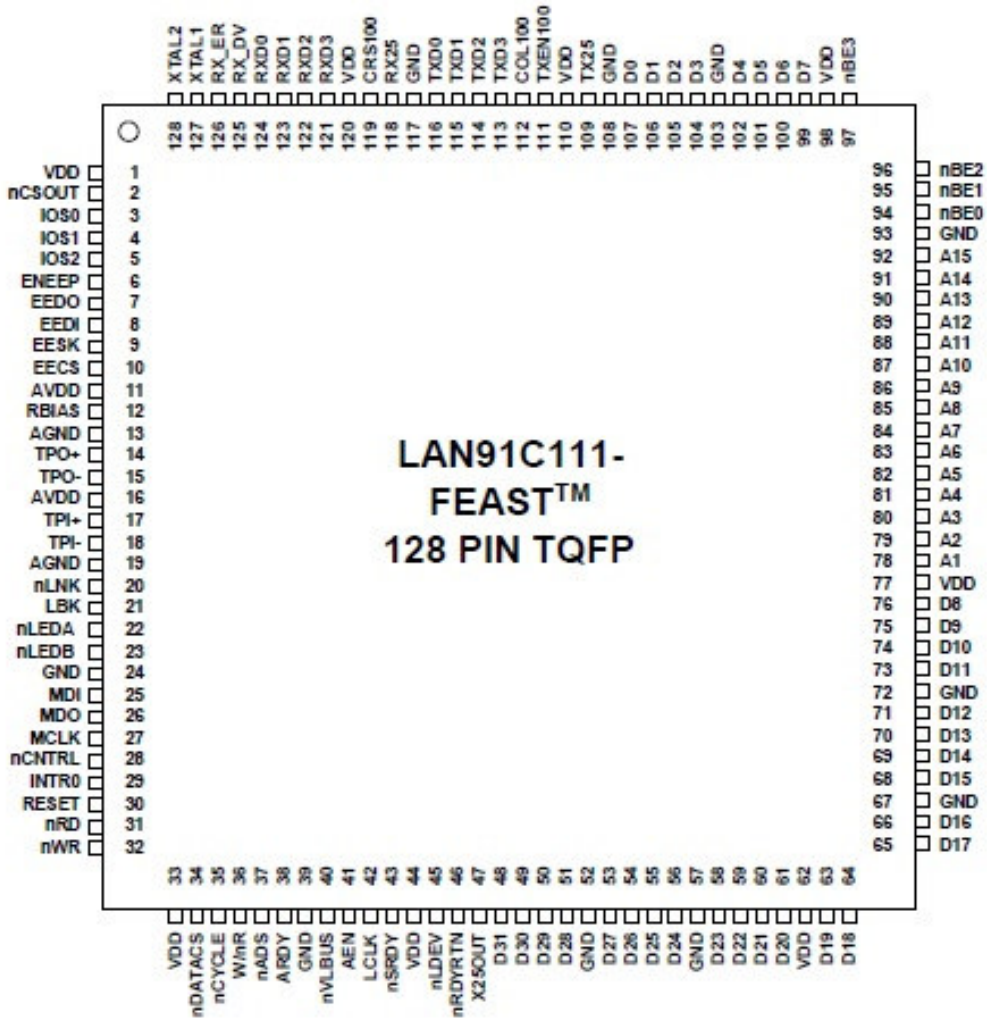
Supporta vari tipi di microcontrollori:

- ARM
- SH
- Power PC
- Coldfire
- 680X0, 683XX
- MIPS R3000

Supporta la MII.

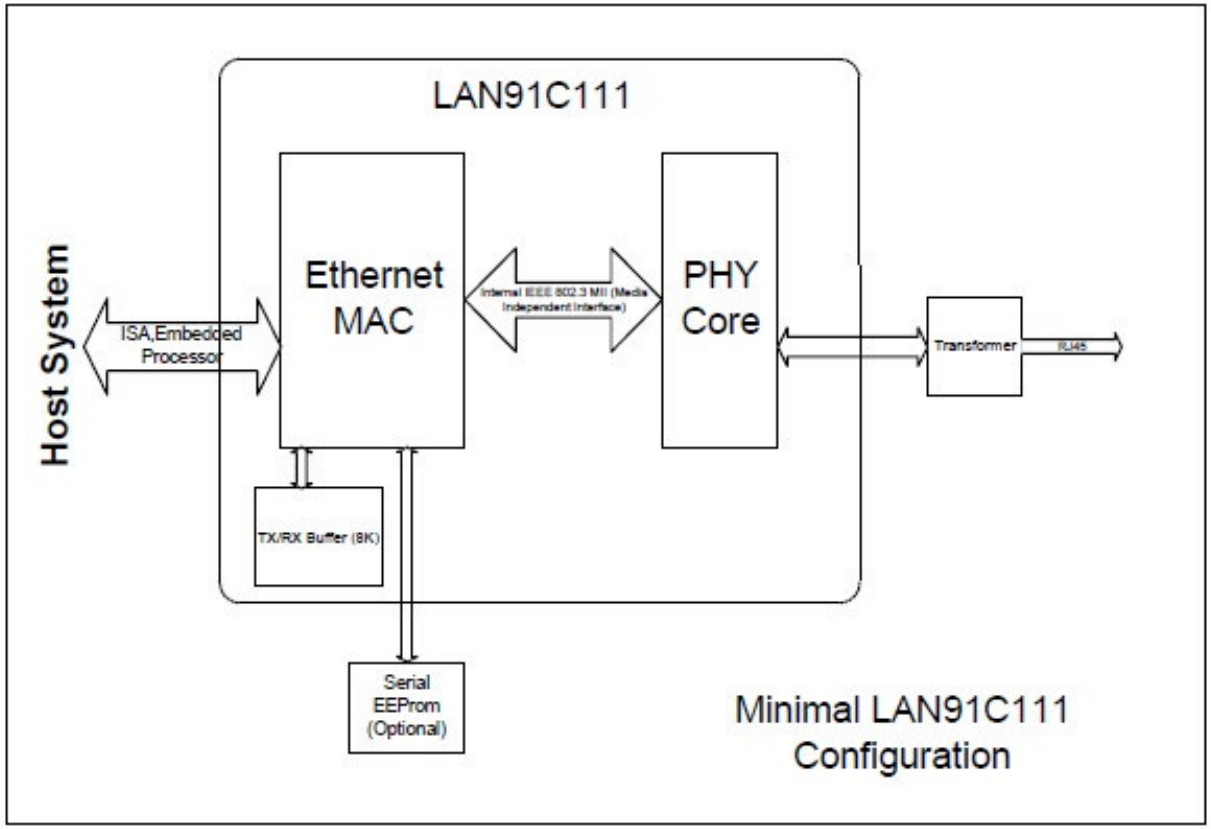
LAN91C111 - SMSC

CONFIGURAZIONE PIN



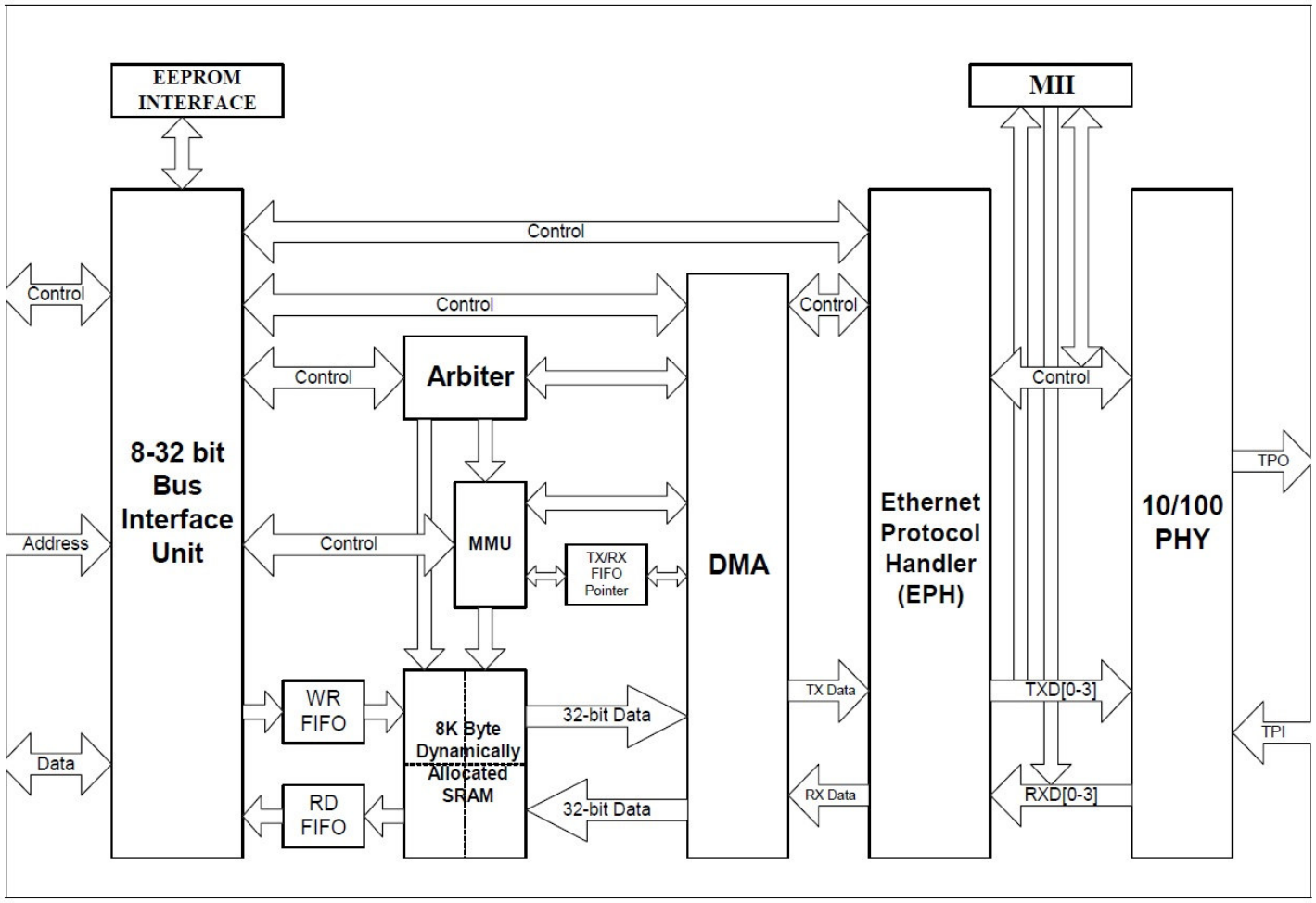
LAN91C111 - SMSC

SCHEMA A BLOCCHI



LAN91C111 - SMSC

INTERFACCE HOST SUPPORTATE



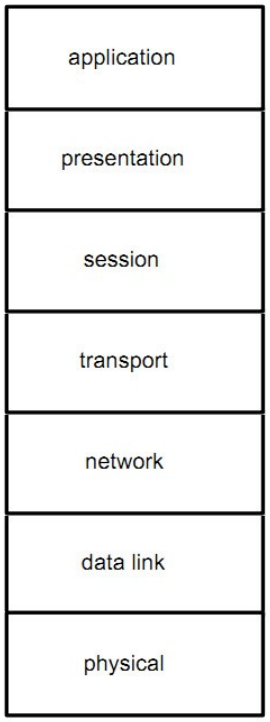
LAN91C111 - SMSC

GESTIONE PIN

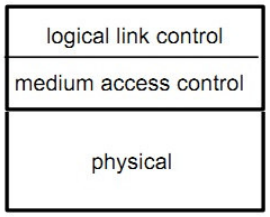
FUNCTION	PIN SYMBOLS	NUMBER OF PINS
System Address Bus	A1-A15, AEN, nBE0-nBE3	20
System Data Bus	D0-D31	32
System Control Bus	RESET, nADS, LCLK, ARDY, nRDYRTN, nSRDY, INTR0, nLDEV, nRD, nWR, nDATACS, nCYCLE, W/nR, nVLBUS	14
Serial EEPROM	EEDI, EEDO, EECS, EESK, ENEEP, IOS0-IOS2	8
LEDs	nLEDA, nLEDB	2
PHY	TPO+, TPO-, TPI+, TPI-, nLNK, LBK, nCNTRL, RBIAS	8
Crystal Oscillator	XTAL1, XTAL2	2
Power	VDD, AVDD	10
Ground	GND, AGND	12
Physical Interface (MII)	TXEN100, CRS100, COL100, RX_DV, RX_ER, TXD0-TXD3, RXD0-RXD3, MDI, MDO, MCLK, RX25, TX25	18
MISC	nCSOUT, X25OUT	2
TOTAL		128

LAN91C111 - SMSC

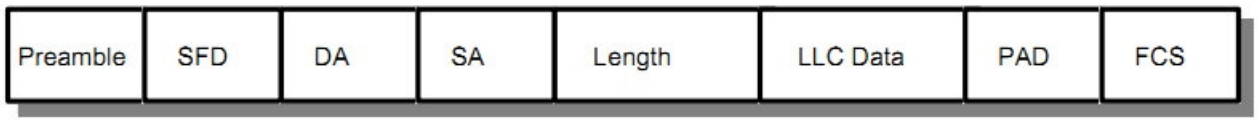
COMUNICAZIONE CON IL LIVELLO FISICO STANDARD 802.3



ISO/OSI



IEEE802



PDU (Protocol Data Unit)

Livelli implementati dello standard

LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
BIG/LITTLE ENDIAN

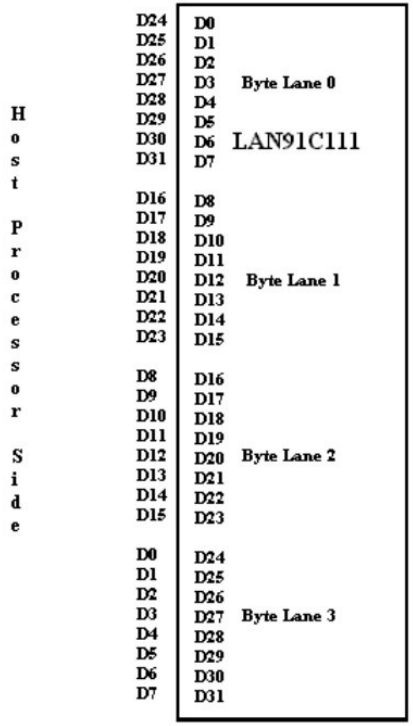
Il LAN91C111 è un dispositivo con architettura Little Endian. È comunque adattabile a strutture Big Endian.

BIG ENDIAN => il byte più significativo sta a sx
LITTLE ENDIAN => il byte più significativo sta a dx

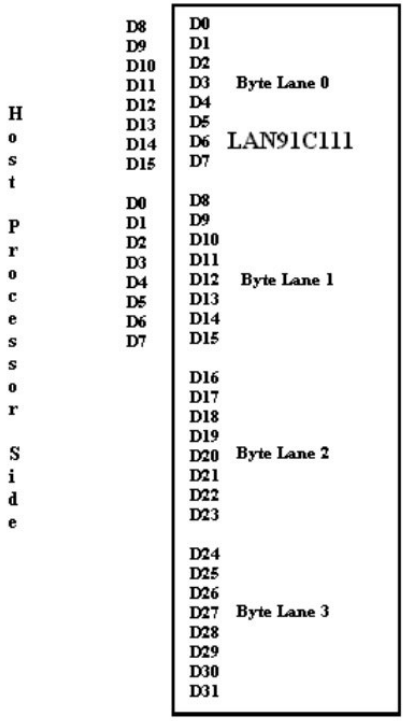
PROBLEMA:
Se il µC ha un'architettura di tipo Big Endian devo riconfigurare la lettura/scrittura di word e double-word.

LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
BIG/LITTLE ENDIAN – BYTE SWAPPING



Big Endian Byte Swapping @ 32 bit



Big Endian Byte Swapping @ 16 bit

LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
BUS PARALLELI

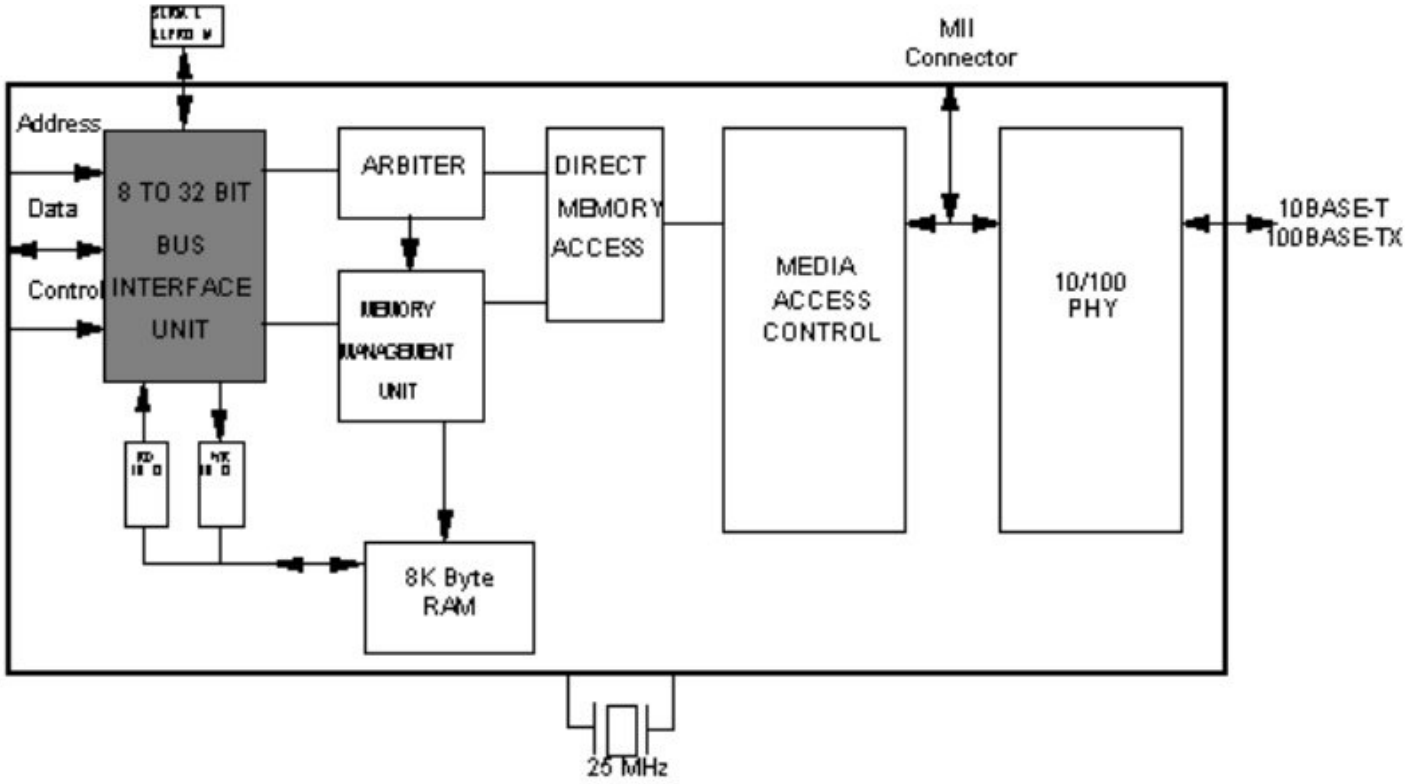
Questo controllore permette inoltre l'interfacciamento con vari tipi di bus paralleli.
I bus gestibili sono:

- VESA Local bus @ 32 bit (VLBUS)
- High-end ISA o non-burst EISA (bus asincrono ad uso industriale)
- EISA 32 bit slave

Questi bus risalgono a parecchi anni fa. I bus ISA ed EISA erano già visibili nei sistemi 80286 o 80386.
Il bus VESA era utilizzato nei sistemi 80486 e serviva per schede grafiche.
Ora tali bus sono sostituiti da PCI e PCI-EXPRESS.
Da bande di lavoro di qualche MHz per i bus ISA, si è arrivati a bande di qualche GHz per i nuovi connettori PCI.

LAN91C111 - SMSC

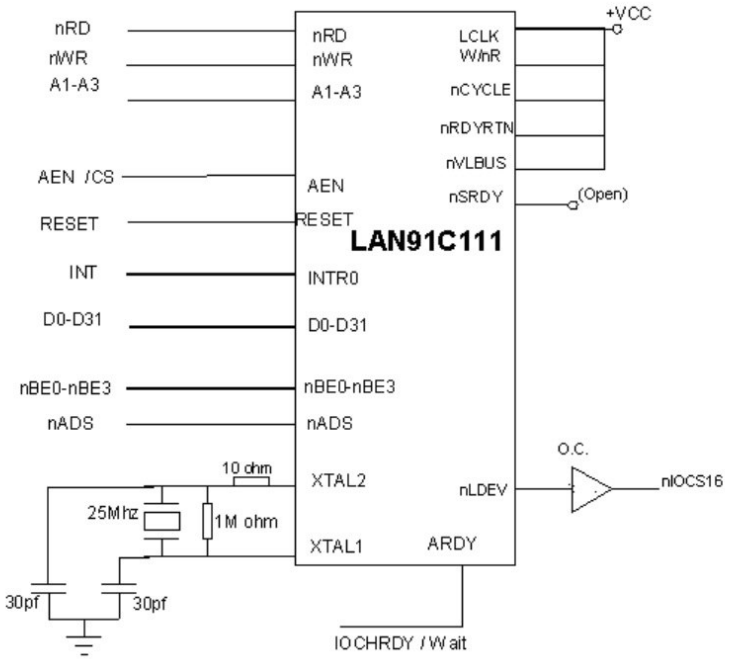
COMUNICAZIONE CON IL LIVELLO FISICO
BUS PARALLELI – ARCHITETTURA



LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
BUS PARALLELI – INTERFACCIA ASINCRONA

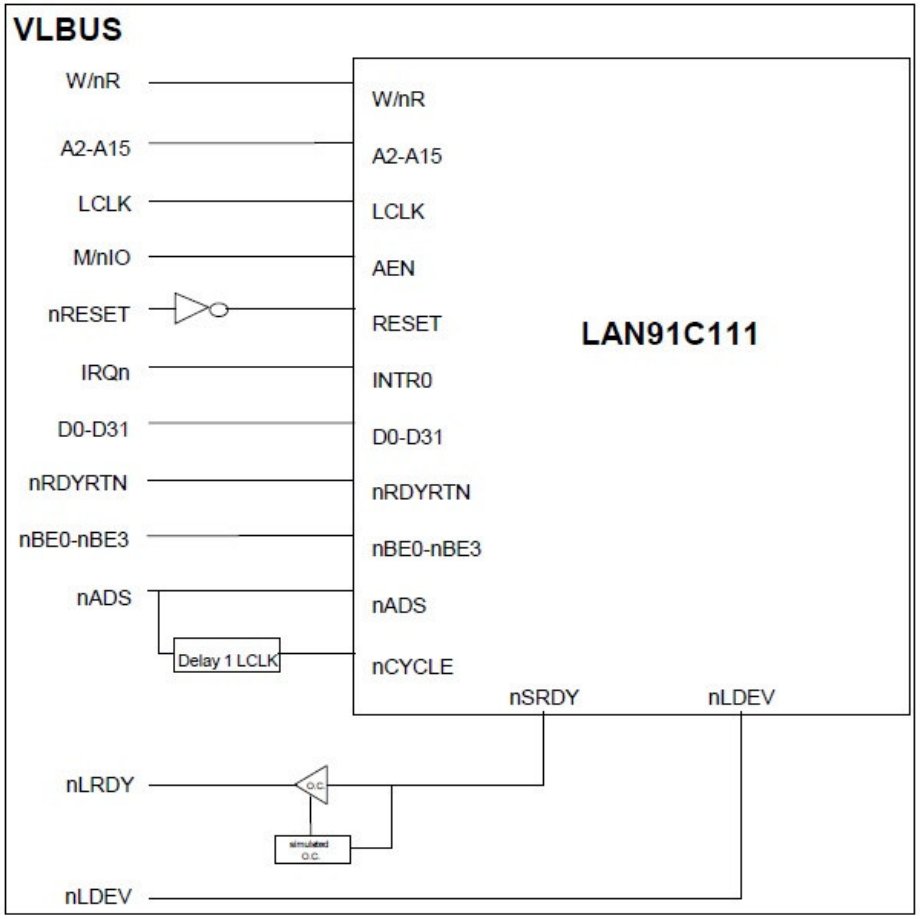
Le operazioni di lettura e scrittura sono controllate dai fronti di nRD e nWR.
ARDY è un segnale di controllo asincrono che notifica al sistema se può estendere il ciclo di accesso.



Schema generico di un'interfaccia asincrona

LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
VLBUS – VESA LOCAL BUS



Per poter utilizzare il bus VESA è necessario che l'ingresso nVLBUS sia '0'.

LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
VLBUS – SEGNALI DI CONTROLLO

- **A2-A15** => gestiscono gli indirizzi. Latchati dal fronte di salita di nADS
- **AEN** => abilita la decodifica degli indirizzi. Latchati dal fronte di salita di nADS
- **W/nR** => è campionato dal controllore sul primo fronte di salita del clock che ha nCYCLE attivo. Si mette a '1' per scrivere, a '0' per leggere
- **nRDYRTN** => ready return. Connessione diretta al VLBUS
- **nSRDY** => dedicato alla gestione del segnale nLRDY del bus VESA
- **LCLK** => local bus clock. I fronti di salita sono usati per le transazioni sincrone sul bus
- **RESET** => reset del micro
- **nBE0-nBE3** => byte enables. Latchati trasparentemente dal fronte di salita di nADS
- **nADS, nCYCLE** => per creare nCYCLE si usa nADS ritardato di un clock
- **INTR0** => usa la linea di interrupt proveniente dal segnale IRQn del connettore ISA

LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
VLBUS – SEGNALI DI CONTROLLO

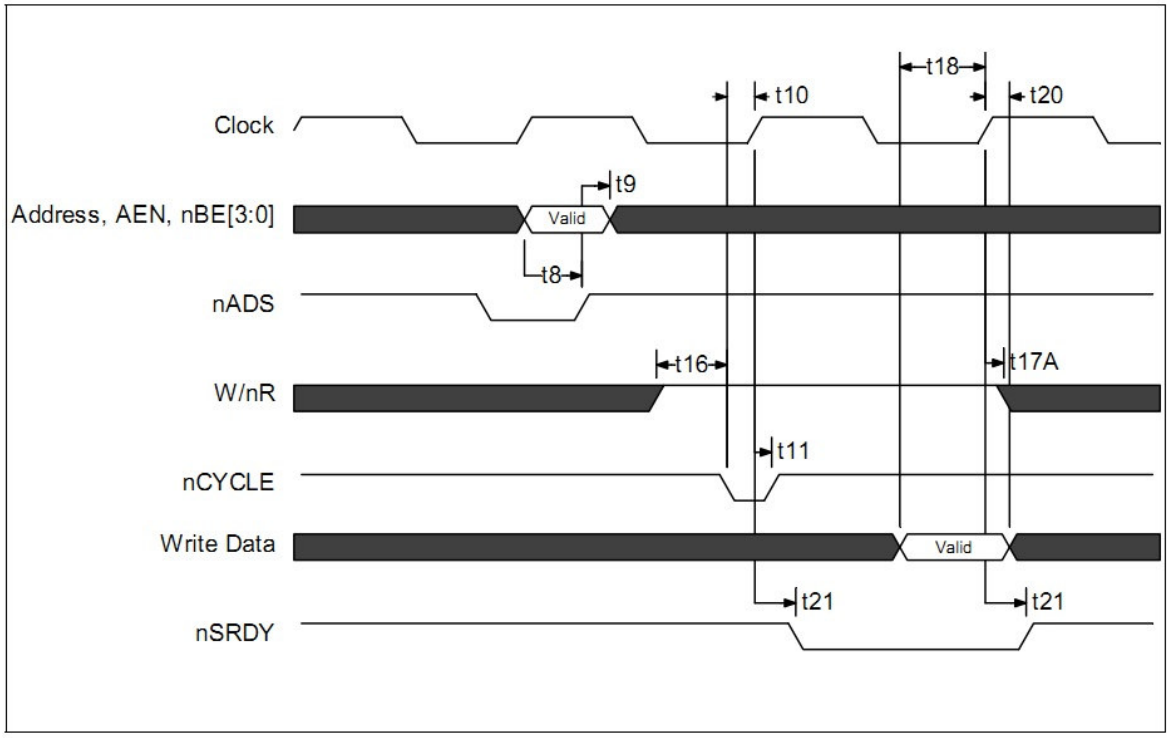
- **D0-D31** => data bus @ 32 bit. I metodi di accesso ai byte sono:

nBE0	nBE1	nBE2	nBE3	
0 0	0	0	Double word access	
0	0	1	1	Low word access
1	1	0	0	High word access
0	1	1	1	Byte 0 access
1	0	1	1	Byte 1 access
1	1	0	1	Byte 2 access
1	1	1	0	Byte 3 access

- **nLDEV** => uscita totem pole attiva su valide decodifiche di A15-A4 e AEN=0

LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
VLBUS – CICLO DI SCRITTURA SINCRONO



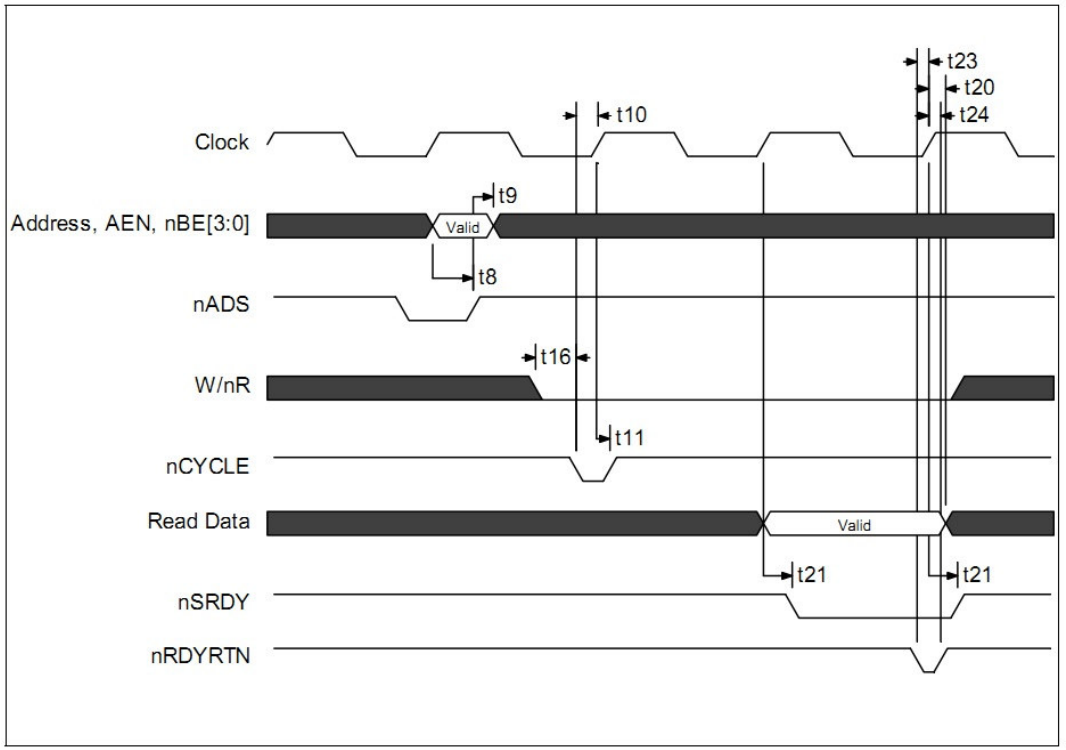
LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
VLBUS – CICLO DI SCRITTURA SINCRONO

- t8 => setup di A1-15, AEN, nBE[3:0] per il fronte di salita di nADS **(8 ns)**
- t9 => hold di A1-15, AEN, nBE[3:0] dopo il fronte di salita di nADS **(5 ns)**
- t10 => setup di nCYCLE per il fronte di salita di LCLK **(5 ns)**
- t11 => hold di nCYCLE per il fronte di salita di LCLK **(3 ns)**
- t16 => setup di W/nR per attivare nCYCLE **(0 ns)**
- t17A => hold di W/nR dopo il fronte di salita di LCLK con nSRDY attivo **(3 ns)**
- t18 => setup dei dati per il fronte di salita di LCLK **(15 ns)**
- t20 => hold dei dati dopo il fronte di salita di LCLK **(4 ns)**
- t21 => ritardo di nSRDY dal fronte di salita di LCLK **(7 ns)**

LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
VLBUS – CICLO DI LETTURA SINCRONO



LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
VLBUS – CICLO DI LETTURA SINCRONO

- t8 => setup di A1-15, AEN, nBE[3:0] per il fronte di salita di nADS **(8 ns)**
- t9 => hold di A1-15, AEN, nBE[3:0] dopo il fronte di salita di nADS **(5 ns)**
- t10 => setup di nCYCLE per il fronte di salita di LCLK **(5 ns)**
- t11 => hold di nCYCLE per il fronte di salita di LCLK **(3 ns)**
- t16 => setup di W/nR per attivare nCYCLE **(0 ns)**
- t20 => hold dei dati dopo il fronte di salita di LCLK **(4 ns)**
- t21 => ritardo di nSRDY dal fronte di salita di LCLK **(7 ns)**
- t23 => setup di nRDYRTN per il fronte di salita di LCLK **(3 ns)**
- t24 => hold di nRDYRTN dopo il fronte di salita di LCLK **(3 ns)**

LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
MII – MEDIA INDEPENDENT INTERFACE

I segnali interessati sono i seguenti:

- **TXD[3:0]** => 4 bit per la trasmissione dei dati
- **TX25** => clock per la trasmissione
- **TXEN100** => per abilitare la trasmissione
- **RXD[3:0]** => 4 bit per la ricezione
- **RX25** => clock per la ricezione
- **RX_DV** => receive data valid
- **CRS100** => carrier sense
- **RX_ER** => receive data error
- **COL100** => controllo collisione (CSMA-CD)

In base al tipo di trasmissione che si vuole adoperare il clock per RX25 e TX25 sarà:

- 25 MHz per 100 Mbps
- 2.5 MHz per 10 Mbps

LAN91C111 - SMSC

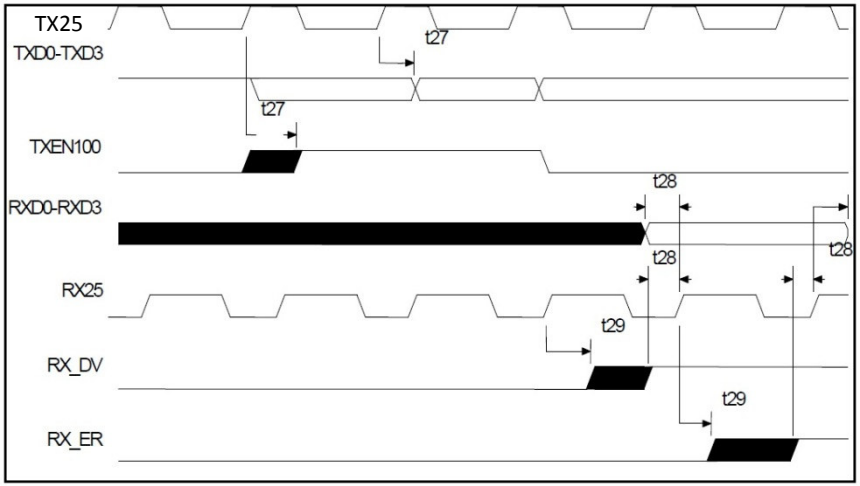
COMUNICAZIONE CON IL LIVELLO FISICO
MII – CONFIGURAZIONE PER 100 Mbps

I pin che mi servono sono:

- Per la TX => TXEN100, TXD0…TXD3, TX25
- Per la RX => RX_DV, RX_ER, RXD0…TXD3, RX25
- Per la comunicazione CSMA-CD => CRS100, COL100

LAN91C111 - SMSC

COMUNICAZIONE CON IL LIVELLO FISICO
MII – DESCRIZIONE TEMPISTICHE



- t27 => ritardo di TXD0-3 e TXEN100 dal fronte di salita di TX25 **(0 ns MIN, 15 ns MAX)**
- t28 => ritardo di RXD0-3, RX_DV e RX_ER dal fronte di salita di RX25 **(10 ns)**
- t29 => tempo di hold di RXD0-3, RX_DV e RX_ER dopo il fronte di salita di RX25 **(10 ns)**