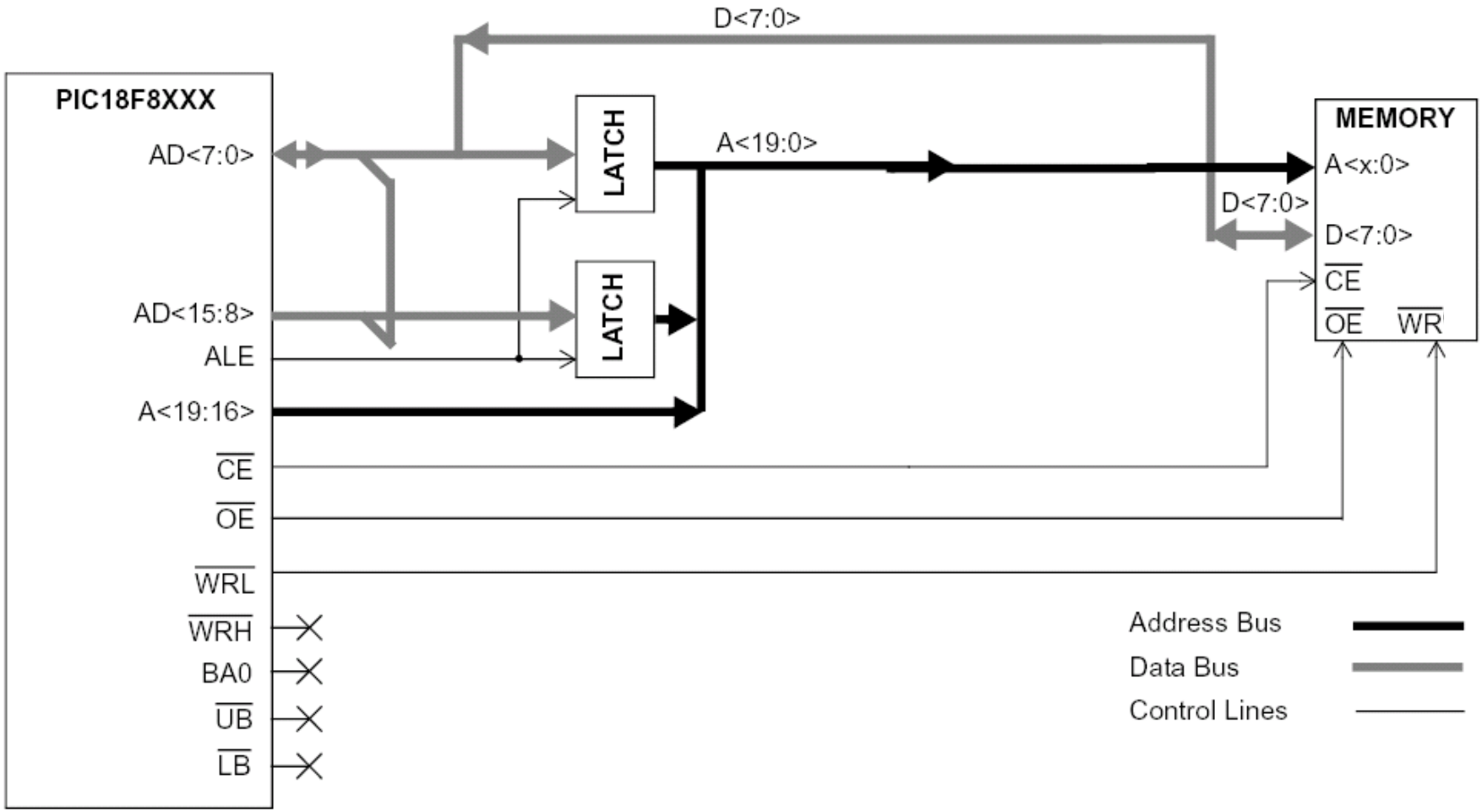
Soluzioni a 8 bit per il bus del PIC18F8x20



Questo schema necessita di adattamenti software (e quindi non è ottimizzato per la velocità di esecuzione del ciclo di bus esterno) e non sfrutta tutto lo spazio di indirizzamento del Microprocessore, ha il vantaggio di essere molto economico e no-glue-logic. Micro in modalità Byte Write Mode.

Driver per lettura/controllo dati in memoria (TBLRD/TBLWRT)

```
void main(void)
{
        rom unsigned int *DataPtr;

        unsigned char i;
        unsigned char v;

        // Setup your data memory address.
        DataPtr = (rom unsigned int*)0x20000;

        // Write 0 - 255 at address
        for ( i = 0; i < 255; i++ )
                DataPtr[i] = i;

        // Read back values
        for ( i = 0; i < 255; i++ )
        {
                v = DataPtr[i];
                if ( v != i ) // And compare it with expected value.
                while(1); // On failure, it will loop here forever.
        }

        // When done stop here.
        while(1);
}
```

Attenzione, se questo è un codice che non avrebbe nessun problema in un sistema a word, in un sistema a byte vale se e solo se nella word che si tenta di scrivere il byte alto non contiene informazioni (ovvero se si ha la necessità di memorizzare solo byte), in caso contrario, questo codice NON FUNZIONA, il sistema mostrato nella slide precedente necessita di driver software di interfacciamento non standard, bensì *hardware dependent*.

Funzioni di lettura/scrittura per i tipi di dato utilizzati

```
rom int *intVal = 0x20000; // Assume that external RAM is at 128K

int tempInt; // Temporary internal RAM

// For every access to external RAM variable, use a special
// function. For instance, there could be

// ReadInt(), WriteInt(), ReadLong(), WriteLong(), ReadMem(), WriteMem()

// Read intVal that is located in external RAM
TempInt = ReadInt(intVal) ;

// Write to tempInt
WriteInt(intVal, tempInt);
```

Queste funzioni sono *hardware-dependent* e dipendono dalla configurazione circuitale, sono tutte da scrivere

Esempio: Funzione ReadInt()

```
int ReadInt(rom int* ptr)
{
        union
        {
                int i; // approccio a word
                char v[2]; // approccio a byte
        } t;

        unsigned short long sl; //definisco il tipo del puntatore TBLPTR (21 bit)

        sl = (unsigned short long)ptr;
        sl <<= 1; //riporto gli address alla compatibilità circuitale (solo address pari)

        ptr = (rom int*)sl; // 16-bit to 8-bit address mapping

        t.v[0] = *ptr++; // Read int in little-endian format
        t.v[1] = *ptr; // Depending on where memory is wired,

        return t.i; // actual value is available at odd or even addresses only
}
```

Esempio: Funzione WriteInt()

```
void WriteInt(rom int* ptr, int val)
{
        union
        {
                int I; // approccio a word
                char v[2]; // approccio a byte
        } t;

        unsigned short long sl; //definisco il tipo del puntatore TBLPTR (21 bit)

        sl = (unsigned short long)ptr;
        sl <<= 1; //riporto gli address alla compatibilità circuitale (solo address pari)

        ptr = (rom int*)sl; // 16-bit to 8-bit address mapping.

        t.i = val;

        *ptr++ = t.v[0];      //scrivo il byte basso (pari)
        *ptr = t.v[1];         //scrivo il byte alto (dispari)
}
```

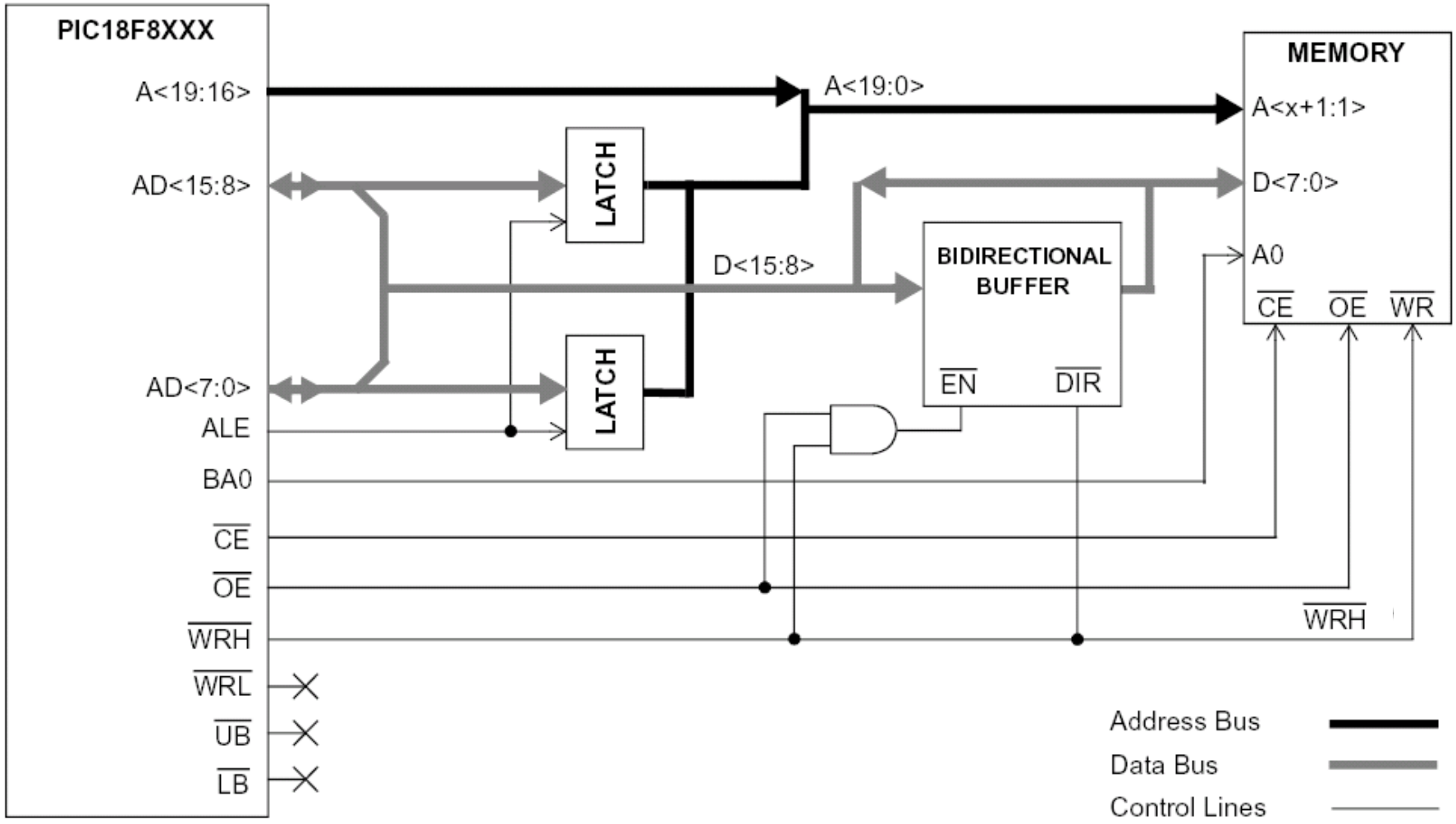
Esempio: Funzione ReadLong()

```
long ReadLong(rom long* ptr)
{
        rom int* ptr2;
        union
        {
                long l; // approccio a word
                char v[4]; // approccio a byte
        } t;

        unsigned short long sl; //definisco il tipo del puntatore TBLPTR (21 bit)
        sl = (unsigned short long)ptr;
        sl <<= 1; //riporto gli address alla compatibilità circuitale (solo address pari)

        ptr2 = (rom int*)sl; // 16-bit to 8-bit address mapping
        t.v[0] = *ptr2++;      // Read long in little-endian format
        t.v[1] = *ptr2++;
        t.v[2] = *ptr2++;
        t.v[3] = *ptr2;        // Depending on where memory is wired,
        return t.l;            // actual value is available at odd or even addresses only
}
```

Esempio: Funzione WriteLong()

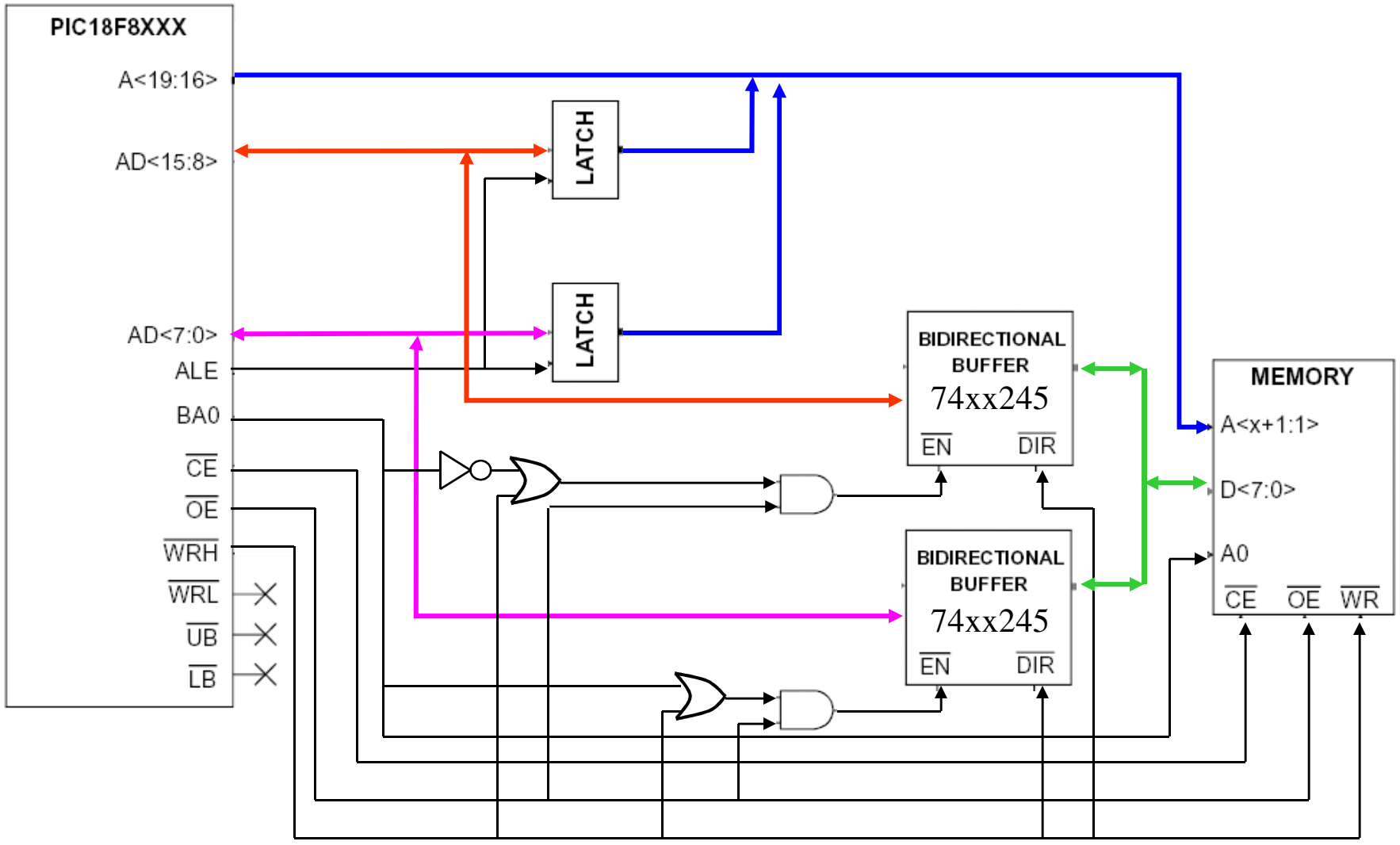
```
void WriteLong(rom long* ptr, long val)
{
        rom int* ptr2;
        union
        {
                long l; // approccio a word
                char v[4]; // approccio a byte
        } t;

        unsigned short long sl; //definisco il tipo del puntatore TBLPTR (21 bit)
        sl = (unsigned short long)ptr;
        sl <<= 1; //riporto gli address alla compatibilità circuitale (solo address pari)

        ptr2 = (rom int*)sl; // 16-bit to 8-bit address mapping.
        t.l = val;
        *ptr2++ = t.v[0];    //scrivo il byte basso (pari)
        *ptr2++ = t.v[1];    //scrivo il secondo byte (dispari)
        *ptr2++ = t.v[2];    //scrivo il terzo byte (pari)
        *ptr2 = t.v[3];      //scrivo il byte alto (dispari)
}
```

Collegamento con memoria a 8bit con utilizzo dell'intero spazio di indirizzamento

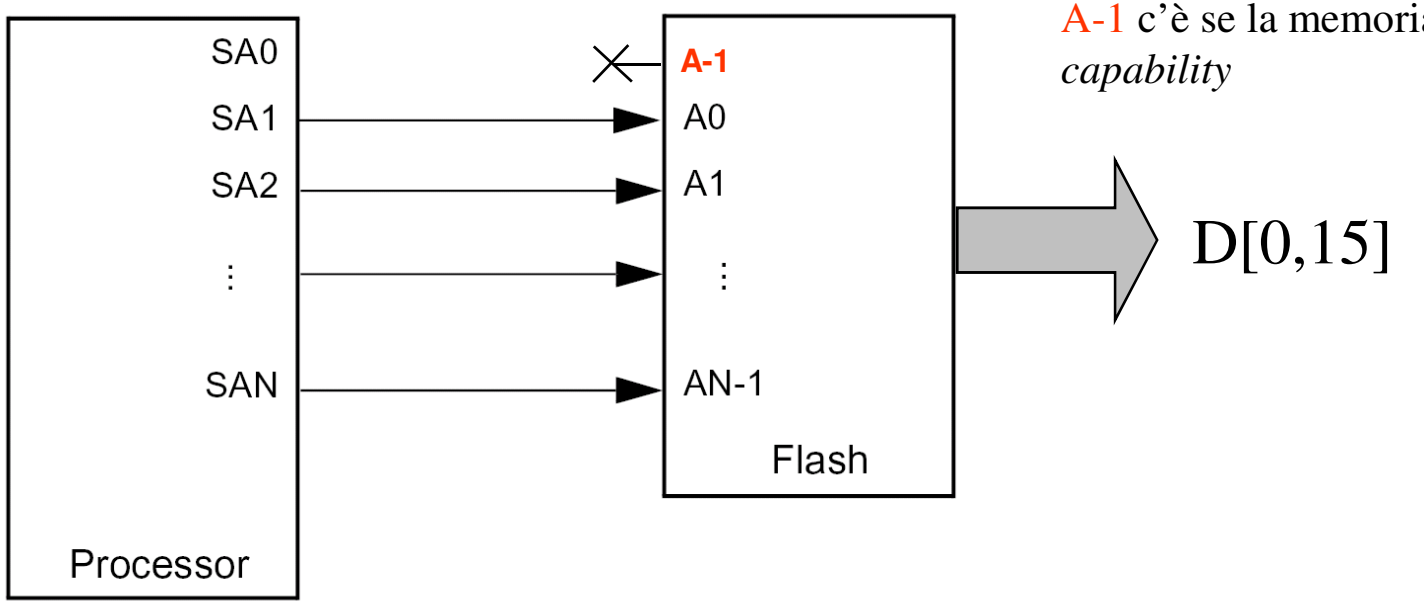


Questo schema non necessita di adattamenti software (e quindi è ottimizzato per la velocità di esecuzione del ciclo di bus esterno) e sfrutta tutto lo spazio di indirizzamento del Microprocessore, ha lo svantaggio di avere necessità di glue-logic. Micro in modalità Byte Select Mode.

Una soluzione analoga ma priva di conflitti sul bus



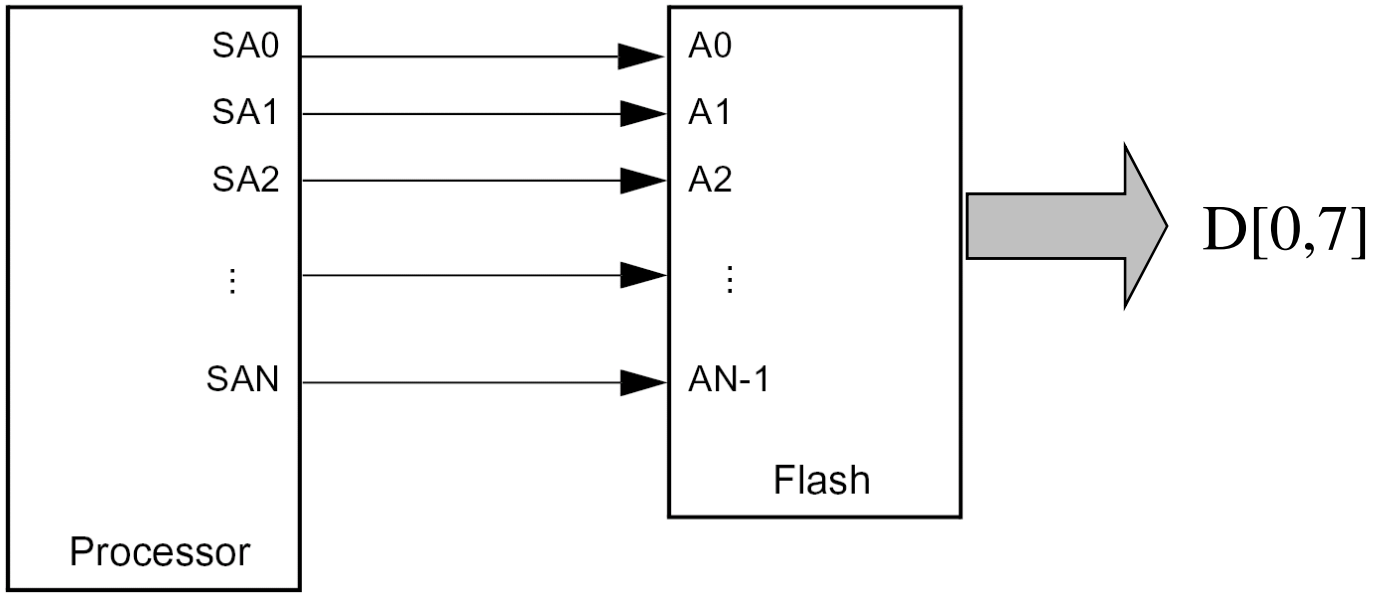
Possibili collegamenti micro/memorie



A-1 c'è se la memoria ha *byte address capability*

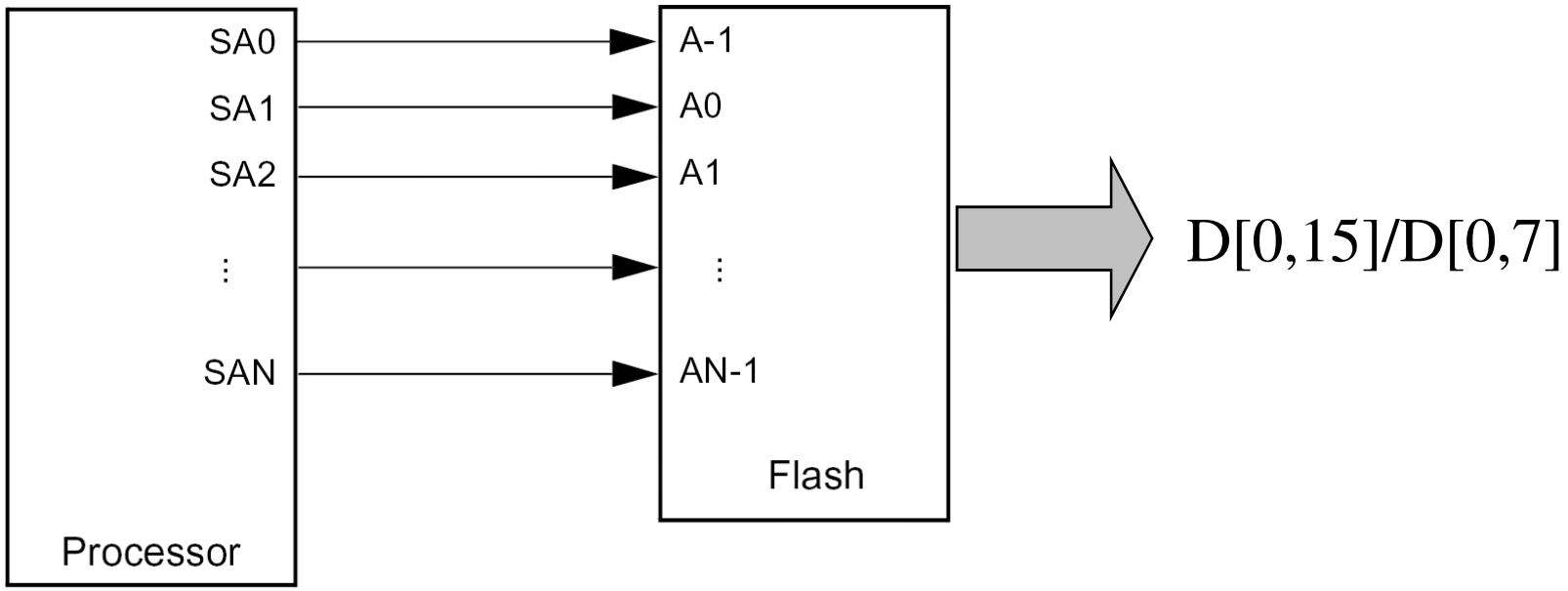
Byte address processor connesso a Flash *word-wide*: il pin SA0 è lasciato disconnesso, Può essere indirizzata solo la word (2 byte) e le word sono allineate ad indirizzi pari (limitazione nel software). In caso di bus multiplexato è probabile che il bus dati sia multiplexato a partire dal pin SA1 (ovvero SA1=D0).

Possibili collegamenti micro/memorie (2)



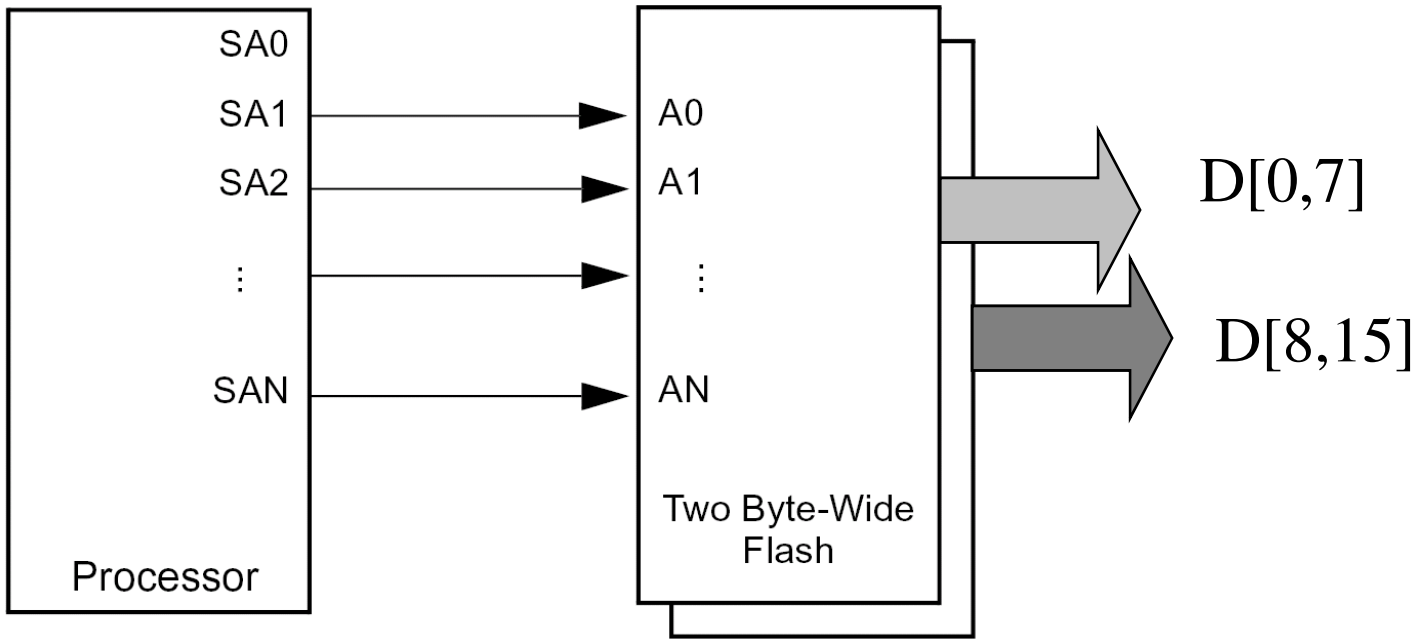
Byte address processor connesso a Flash *byte-wide*: il pin SA0 è collegato,
Può essere indirizzato il byte, possono essere indirizzate anche le word ad indirizzi sia pari che dispari, per la lettura/scrittura di una word sono necessari due cicli di bus esterno.
In caso di bus multiplexato è probabile che il bus dati sia multiplexato a partire dal pin SA0 (ovvero SA0=D0).

Possibili collegamenti micro/memorie (3)



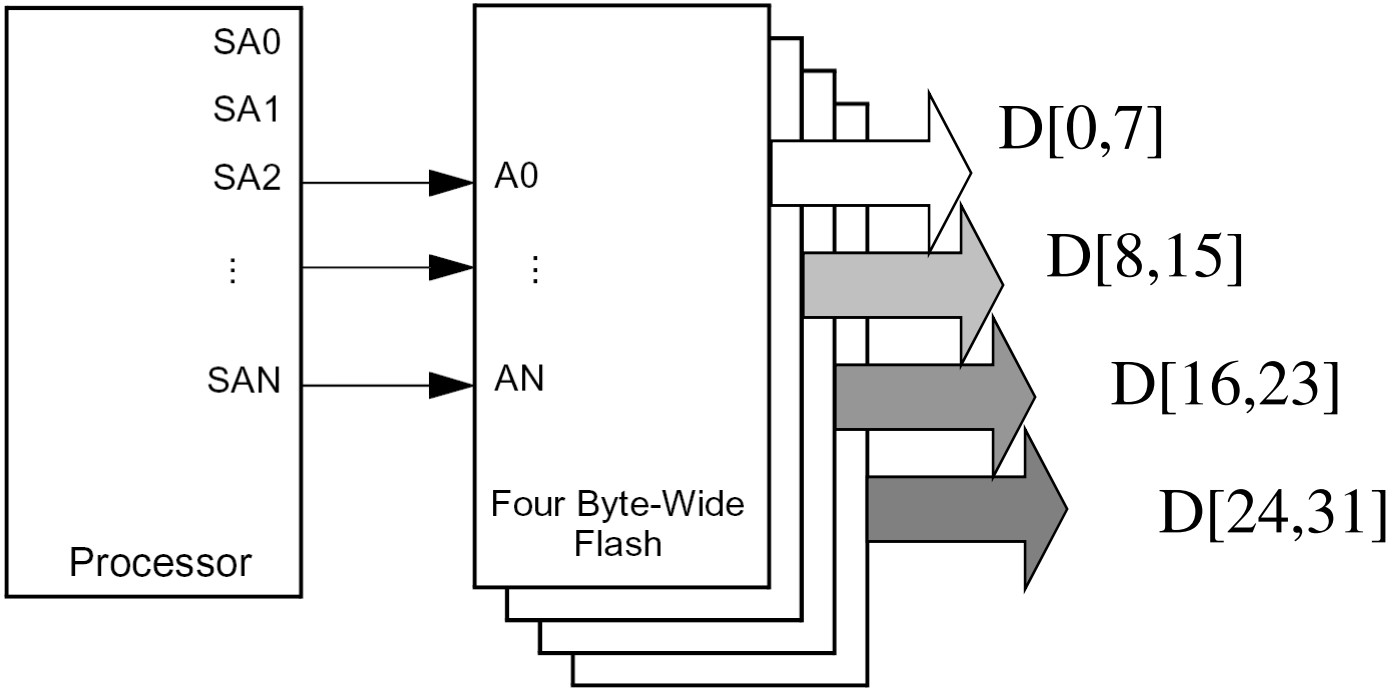
Byte address processor connesso a Flash *word-wide* con *byte address capability*: il pin SA0 è collegato al pin A-1.
Può essere indirizzato il byte, possono essere indirizzate anche le word ad indirizzi sia pari che dispari, per la lettura/scrittura di una word è necessario un ciclo di bus esterno, se indirizzata a indirizzi pari. In caso di bus multiplexato è probabile che il bus dati sia multiplexato a partire dal pin SA1 (ovvero SA1=D0).
In caso di lettura di un byte, la parte alta del data bus della flash viene resa *three-stated* e viene letto il byte nella parte bassa del *data bus*

Possibili collegamenti micro/memorie (4)



Byte address processor connesso a 2 Flash *byte-wide*: il pin SA0 non è collegato.
Le due Flash vengono abilitate contemporaneamente e si ha la lettura di una word per ogni ciclo di bus. I due semi-data bus sono connessi ciascuno ad una delle due Flash.
Può essere indirizzata la word ad indirizzi sia pari, per la lettura/scrittura di una word è necessario un ciclo di bus esterno, se indirizzata a indirizzi pari. In caso di bus multiplexato è probabile che il bus dati sia multiplexato a partire dal pin SA1 (ovvero SA1=D0).

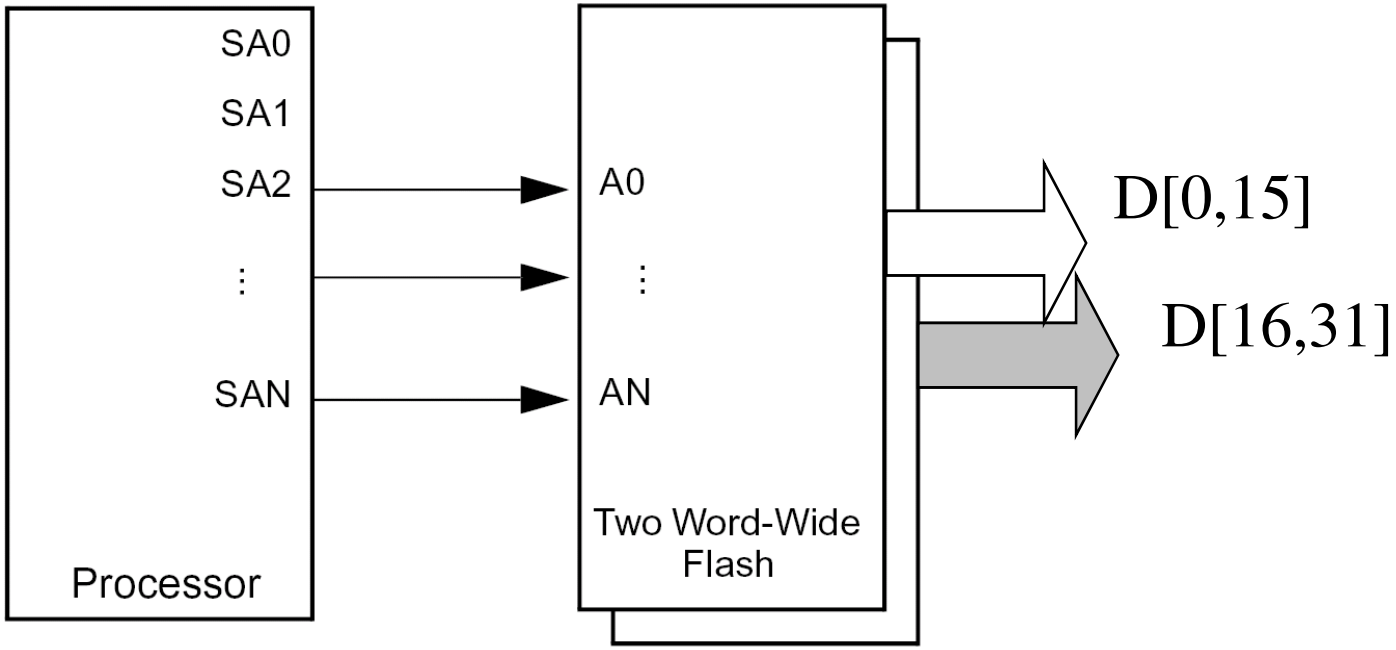
Possibili collegamenti micro/memorie (5)



Consideriamo un microprocessore con data bus a 32 bit (NON il PIC18F8x20!!!!!!).

Byte address processor connesso a 4 Flash *byte-wide*: i pin SA0 e SA1 non sono collegati. Le 4 Flash vengono abilitate contemporaneamente e si ha la lettura di una longword per ogni ciclo di bus. I 4 byte del data bus sono connessi ciascuno ad una delle 4 Flash.
Può essere indirizzata la longword ad indirizzi pari e multipli di 4, per la lettura/scrittura di una longword è necessario un ciclo di bus esterno. In caso di bus multiplexato è probabile che il bus dati sia multiplexato a partire dal pin SA1 (ovvero SA1=D0).

Possibili collegamenti micro/memorie (6)



Consideriamo un microprocessore con data bus a 32 bit (NON il PIC18F8x20!!!!!!).

Byte address processor connesso a 2 Flash *word-wide*: i pin SA0 e SA1 non sono collegati. Le 2 Flash vengono abilitate contemporaneamente e si ha la lettura di una longword per ogni ciclo di bus. Le 2 word del data bus sono connesse ciascuna ad una delle 2 Flash. Può essere indirizzata la longword ad indirizzi pari e multipli di 4, per la lettura/scrittura di una longword è necessario un ciclo di bus esterno. In caso di bus multiplexato è probabile che il bus dati sia multiplexato a partire dal pin SA1 (ovvero SA1=D0).

Relazione tra i bus indirizzi di Flash e micro

AD10	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	BA0

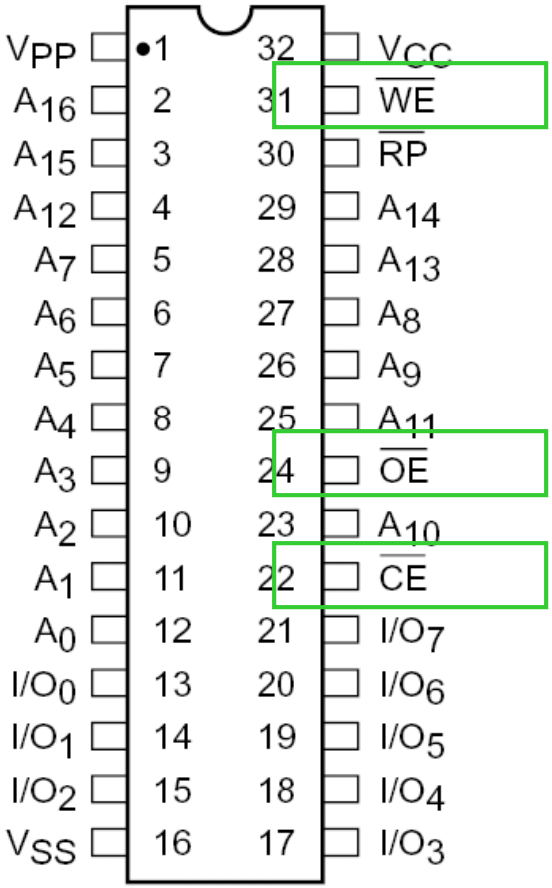
Description	Address Lines & Values (Note 1)											
System Address	SA11	SA10	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0
x8-only Flash (Note 2)	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
x8/x16 Flash Byte Mode (Note 2)	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	A-1
x8/x16 Flash Word Mode (Note 3)	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
System Binary Address	1	0	1	0	1	0	1	0	1	0	1	0
Flash Hex Address (x8/x16 device in word mode)	5			5				5				
Flash Hex Address (x8-only device or x8/x16 device in byte mode)	A				A				A			

Notes:

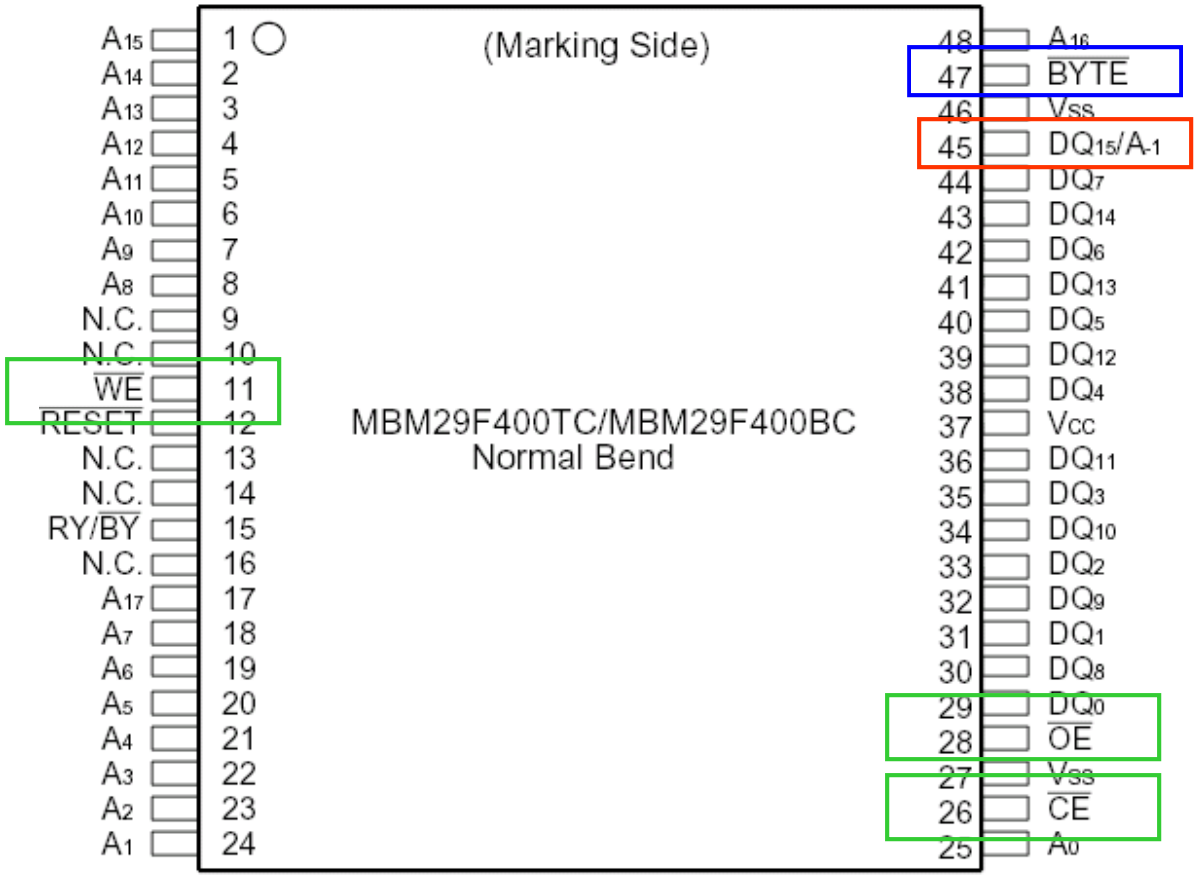
1. *Table assumes processor provides a byte-selecting system address.*
2. *Assumes single Flash memories are used to provide a byte-wide data bus.*
3. *Assumes single Flash memories are used to provide a word-wide data bus.*

La corrispondenza degli indirizzi è la stessa vista durante lo studio del micro, nel caso di flash *word-wide* si è orientati a word anche se si ha la possibilità di indirizzare anche il singolo byte (vedere le x8/x16 flash).

Tipi di flash



Memoria flash a byte 28F001 Catalyst 64Kbyte



Memoria flash a word con byte address capability, sono presenti il “mode pin” /BYTE e DQ15 multiplexato con A-1 per l’*even/odd byte addressing Fujitsu/AMD 4Mbit*

Pinout

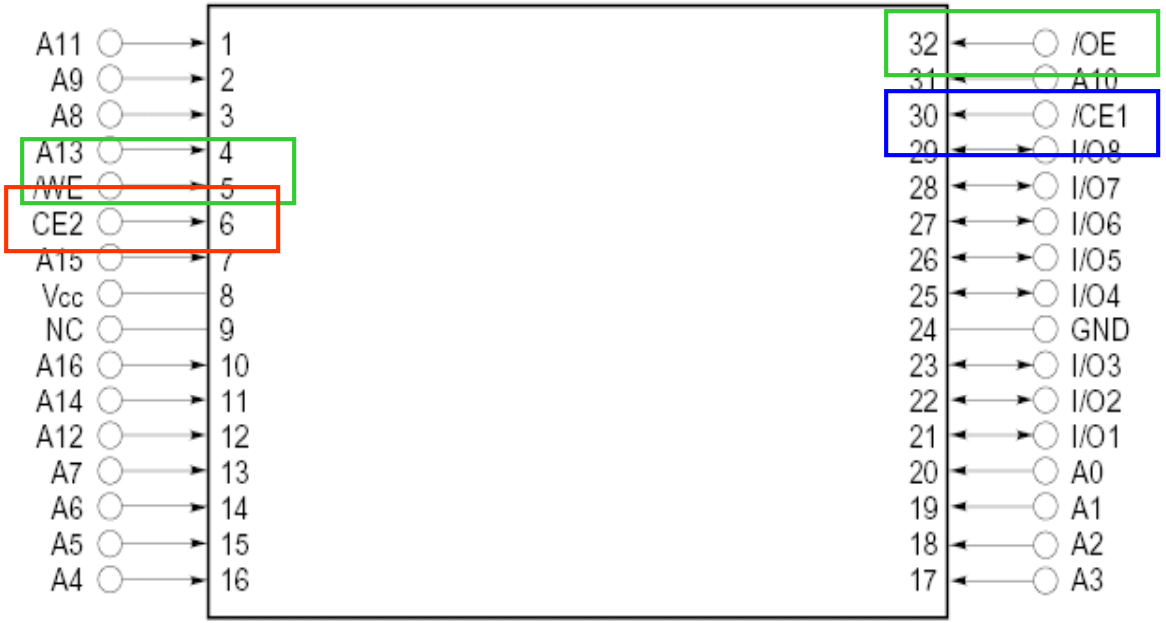
Memoria flash a byte

Pin Name	Type	Function
A_0–A_{16}	Input	Address Inputs for memory addressing
I/O_0–I/O_7	I/O	Data Input/Output
$\overline{CE}$	Input	Chip Enable
$\overline{OE}$	Input	Output Enable
$\overline{WE}$	Input	Write Enable
V_{CC}		Voltage Supply
V_{SS}		Ground
V_{PP}		Program/Erase Voltage Supply
$\overline{RP}$	Input	Power Down

Pin name	Function
A_{17} to A_0, A_{-1}	Address Inputs
DQ_{15} to DQ_0	Data Inputs/Outputs
$\overline{CE}$	Chip Enable
$\overline{OE}$	Output Enable
$\overline{WE}$	Write Enable
RY/$\overline{BY}$	Ready-Busy Output
$\overline{RESET}$	Hardware Reset Pin/Temporary Sector Unprotection
$\overline{BYTE}$	Selects 8-bit or 16-bit mode
N.C.	No Internal Connection
V_{SS}	Device Ground
V_{CC}	Device Power Supply

Memoria flash a word con byte address capability, sono presenti il “mode pin” /BYTE e DQ15 multiplexato con A-1 per l’*even/odd byte addressing*

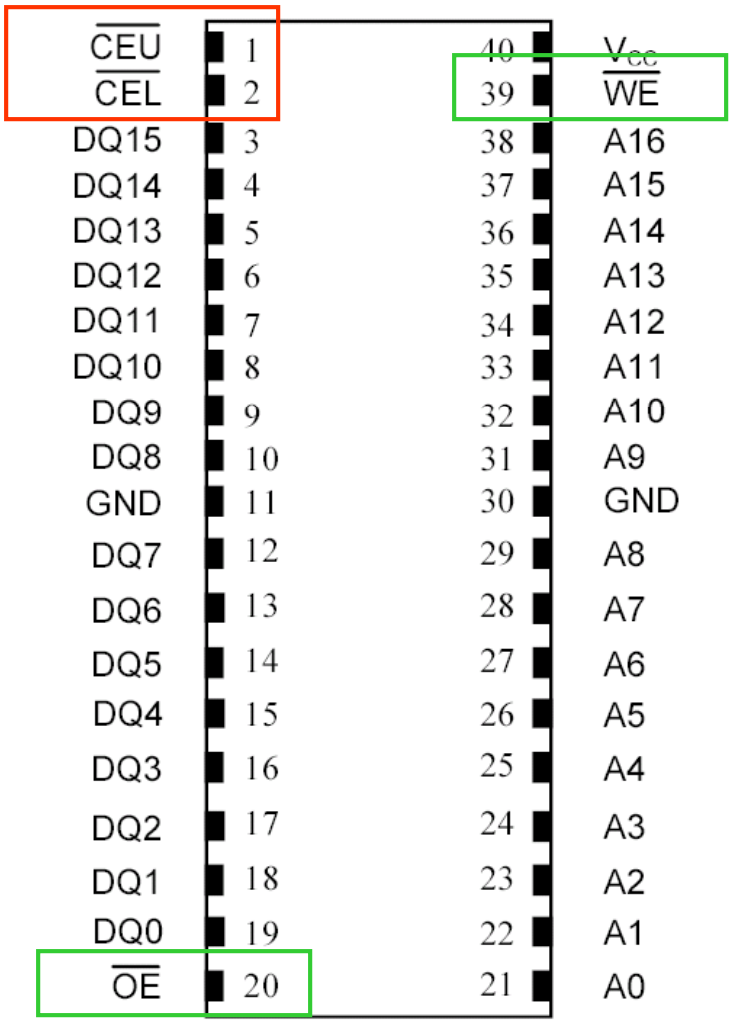
Tipi di RAM (byte wide)



Memoria SRAM a byte NEC PD431000A da 1 Mbit, sono presenti due CE in logica inversa per l'interfacciamento *no-glue-logic* con micro che attivano i *chip select* sia in logica positiva che negativa.

/CE1	CE2	/OE	/WE	Mode	I/O	Supply current
H	×	×	×	Not selected	High impedance	I_{SB}
×	L	×	×			
L	H	H	H	Output disable		I_{CCA}
L	H	L	H	Read	D_{OUT}	
L	H	×	L	Write	D_{IN}	

Tipi di RAM (word wide)



Memoria SRAM DALLAS DS1258 da 128k x 16 bit, memoria a word con Upper e Lower *byte address capability*, sono presenti due chip enable: **/CEU** e **/CEL** in logica negativa per l'interfacciamento *no-glue-logic* con micro che attivano i *chip select* separatamente per l'emi-bus basso e l'emi-bus alto dei dati.

A0 to A16	- Address Inputs
DQ0 to DQ15	- Data In/Data Out
$\overline{CEU}$	- Chip Enable Upper Byte
$\overline{CEL}$	- Chip Enable Lower Byte
$\overline{WE}$	- Write Enable
$\overline{OE}$	- Output Enable
V_{CC}	- Power (+5V)
GND	- Ground

Modo di funzionamento delle SRAM dotate di CEU e CEL

Cicli della Memoria SRAM DALLAS DS1258 da 128k x 16 bit, determinati dallo stato dei segnali di controllo del bus.
Da notare il fatto che la parte di bus non abilitata viene portata in alta impedenza dalla memoria

$\overline{OE}$	$\overline{WE}$	$\overline{CEL}$	$\overline{CEU}$	V_{CC} CURRENT	DQ0-DQ7	DQ8-DQ15	CYCLE PERFORMED
H	H	X	X	I_{CCO}	High-Z	High-Z	Output Disabled
L	H	L	L	I_{CCO}	Output	Output	Read Cycle
L	H	L	H		Output	High-Z	
L	H	H	L		High-Z	Output	
X	L	L	L	I_{CCO}	Input	Input	Write Cycle
X	L	L	H		Input	High-Z	
X	L	H	L		High-Z	Input	
X	X	H	H	I_{CCS}	High-Z	High-Z	Output Disabled