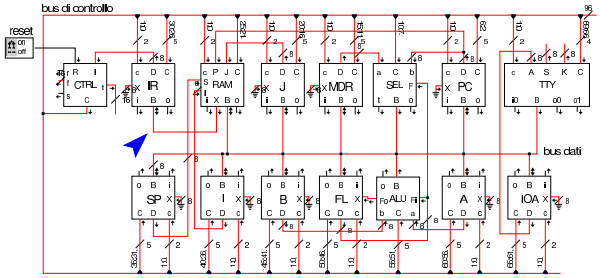
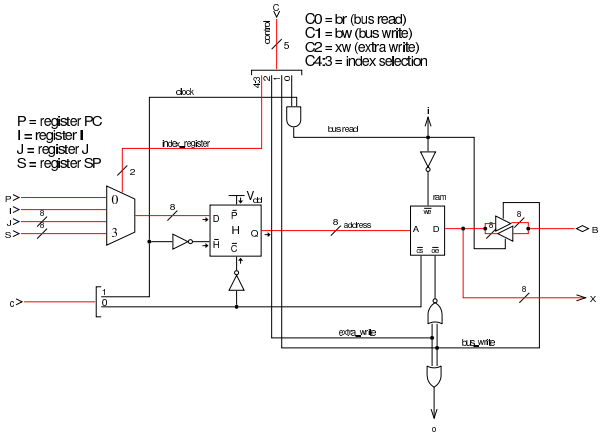
Viene proposta una modifica ulteriore della CPU dimostrativa, con lo scopo di migliorare leggermente la sua efficienza, attraverso il collegamento diretto tra il modulo **RAM** e il registro ***IR***, per non interferire con il bus dati quando si può trasferire un codice operativo dalla memoria al registro relativo.

Figura u115.1. CPU dimostrativa, versione «J».



Per realizzare il collegamento evidenziato nella figura, il modulo **RAM** viene modificato, per poter pilotare un'uscita ausiliaria (***X***); di conseguenza di ingrandisce il suo collegamento al bus di controllo, costringendo a riadattare i collegamenti degli altri moduli.

Figura u115.2. Modulo **RAM** modificato con l'aggiunta dell'uscita ausiliaria.



La dichiarazione dei campi del bus di controllo richiede quindi il cambiamento del significato delle linee di controllo relative al modulo **RAM**, assieme allo slittamento in avanti del collegamento degli altri moduli che seguono:

```
field ctrl[1:0]  = {nop=0, stop=1, load=2};
field pc[6:2]    = {br=1, bw=2, xr=4, inc=8, dec=16};
field sel[10:7]  = {if_carry=1, if_not_carry=3,
                    if_zero=5, if_not_zero=7,
                    if_negative=9, if_not_negative=11,
                    if_overflow=13, if_not_overflow=15};
field mdr[15:11] = {br=1, bw=2, xr=4, inc=8, dec=16};
field j[20:16]   = {br=1, bw=2, xr=4, inc=8, dec=16};
field ram[25:21] = {br=1, bw=2, xw=4, p=0, i=8, j=16, s=24};
field ir[30:26]  = {br=1, bw=2, xr=4, inc=8, dec=16};
field sp[35:31]  = {br=1, bw=2, xr=4, inc=8, dec=16};
field i[40:36]   = {br=1, bw=2, xr=4, inc=8, dec=16};
field b[45:41]   = {br=1, bw=2, xr=4, inc=8, dec=16};
field fl[50:46]  = {br=1, bw=2, xr=4, inc=8, dec=16};
field alu[55:51] = {not=1, and=3, or=5, xor=7, lshl=9, lshr=11,
                    ashl=13, ashr=15, rotl=17, rotr=19, rotcl=21,
                    rotcr=23, add_c=25, sub_b=27, add=29, sub=31};
field a[60:56]   = {br=1, bw=2, xr=4, inc=8, dec=16};
field ioa[65:61] = {br=1, bw=2, xr=4, inc=8, dec=16};
field ioc[69:66] = {br=1, bw=2, req=4, isack=8};
```

Nella dichiarazione del microcodice, cambia il fatto che la RAM,

per poter comunicare con il registro ***IR***, deve avere abilitata la linea ***xr*** (*extra write*), pertanto, tutte le microistruzioni di caricamento del codice operativo successivo (*fetch*), vanno cambiate nel modo successivo:

```
    ir=br ram=xw ram=p pc=inc ctrl=load;
```

Infine, dove possibile, la microistruzione di caricamento (*fetch*) che appare alla fine di ogni macroistruzione, si fonde con la penultima macroistruzione. Si tratta precisamente delle istruzioni relative alla copia di un valore da un registro a un altro e quelle che utilizzano la ALU: si riducono tutte a un solo ciclo di clock:

```
mv_mdr_a:
    a=br mdr=bw ir=br ram=xw ram=p pc=inc ctrl=load;
mv_mdr_b:
    b=br mdr=bw ir=br ram=xw ram=p pc=inc ctrl=load;
…
mv_j_i:
    i=br j=bw ir=br ram=xw ram=p pc=inc ctrl=load;
mv_j_mdr:
    mdr=br j=bw ir=br ram=xw ram=p pc=inc ctrl=load;
```

```
not:
    a=br alu=not fl=xr ir=br ram=xw ram=p pc=inc ctrl=load;
and:
    a=br alu=and fl=xr ir=br ram=xw ram=p pc=inc ctrl=load;
…
add:
    a=br alu=add fl=xr ir=br ram=xw ram=p pc=inc ctrl=load;
sub:
    a=br alu=sub fl=xr ir=br ram=xw ram=p pc=inc ctrl=load;
```