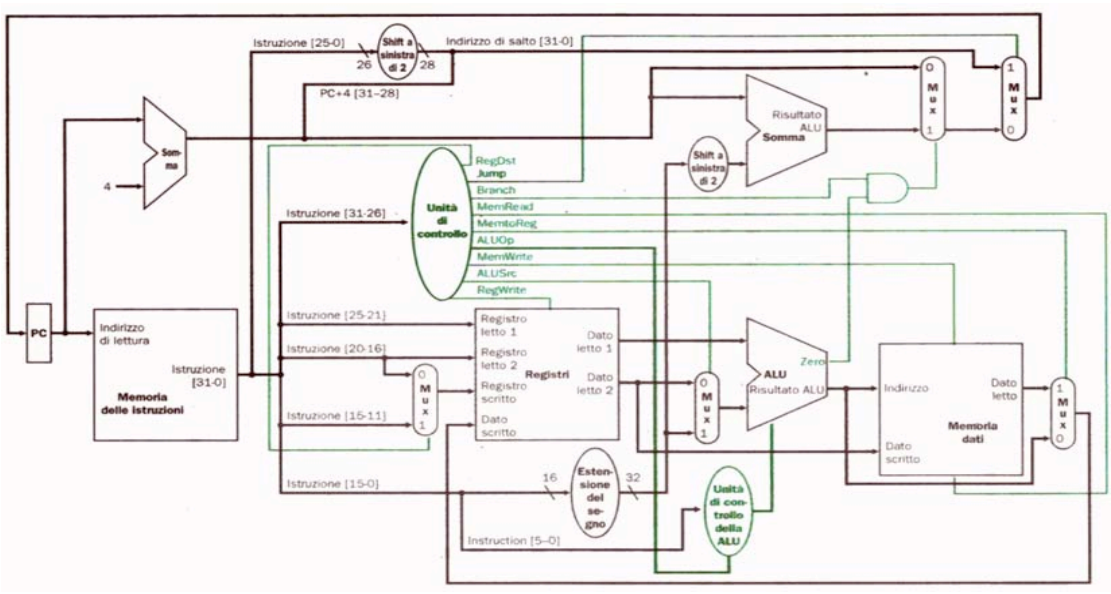
1. [5] Si progetti un circuito caratterizzato da un ingresso a 4 bit che rappresenta un numero binario A, e da un'uscita che vale '1' se A è multiplo di 3, altrimenti vale '0'.
   - Determinare la tabella di verità della funzione logica di uscita;
   - scrivere la funzione nella forma canonica più adatta;
   - semplificarla mediante mappa di Karnaugh e semplificarla ulteriormente, se possibile, mediante passaggi algebrici;

2. [5] Data l'architettura riportata nella figura sottostante, la quale esegue il codice riportato a lato, determinare:

   a) in che stato si trova la CPU dopo **20 cicli** di clock, a partire dal prelievo della prima istruzione (si supponga che **bne** effettui sempre il salto).

   b) Ci sono hazard? Se sì, quali?

```
add $s4, $s1, $s2
lw  $s0, 0($s5)
addi $s6, $s6, 1
bne $s0, $s6, -12
```



3. [8] Si sintetizzi una macchina a stati finiti (di Moore) che realizza un contatore modulo 4 che conta i fronti di discesa di un segnale A fornito in ingresso.

   Il valore dell'ingresso A viene controllato dalla macchina ogni millisecondo.

   L'uscita è costituita da 2 linee, che rappresentano il valore binario assunto dal contatore.

   Si determinino STG, STT, STT codificata e struttura circuitale del sistema completo, avendo cura di semplificare il più possibile le funzioni prima di tradurle in circuito.

4. [6] Si progetti (esplicitando le dimensioni di tutti i campi) una memoria cache a due vie, associata ad un processore con bus dati di 128 bit e bus indirizzi di 40 bit. La capacità totale della cache sia di 1 Mbyte e la dimensione del blocco di 16 parole. Se ne rappresenti lo schema circuitale dettagliato e si calcoli il valore (decimale) di tutti i campi di indirizzamento relativi all'indirizzo: **0x9876543210**.

5. [5] Una CPU viene dotata di una ALU vettoriale, mediante la quale è possibile eseguire 16 operazioni aritmetico/logiche in parallelo. Calcolare:
   - di quanto si velocizza il sistema (rispetto a quello con ALU scalare), quando esegue programmi composti per l' 80% da istruzioni aritmetico/logiche vettoriali;
   - quanto deve essere la percentuale di tempo dedicata a istruzioni vettoriali, tale per cui l'incremento di velocità globale del sistema risulti la metà di quello che si otterrebbe con il 100% di istruzioni vettoriali.

6. [3] Descrivere il funzionamento dell'algoritmo di "backoff" utilizzato nel protocollo "Ethernet" per la gestione delle collisioni, e spiegare quali sono i vantaggi derivanti dal suo impiego.

7. [4] Si traducano le seguenti pseudoistruzioni: **a)** in Assembly MIPS nativo e **b)** in linguaggio macchina MIPS (specificando dimensione in bit e valore dei campi di ogni istruzione).

```
bgei $14, 15, 16    # branch on greater or equal than immediate
divi $10, $20, 30   # divide by immediate
```

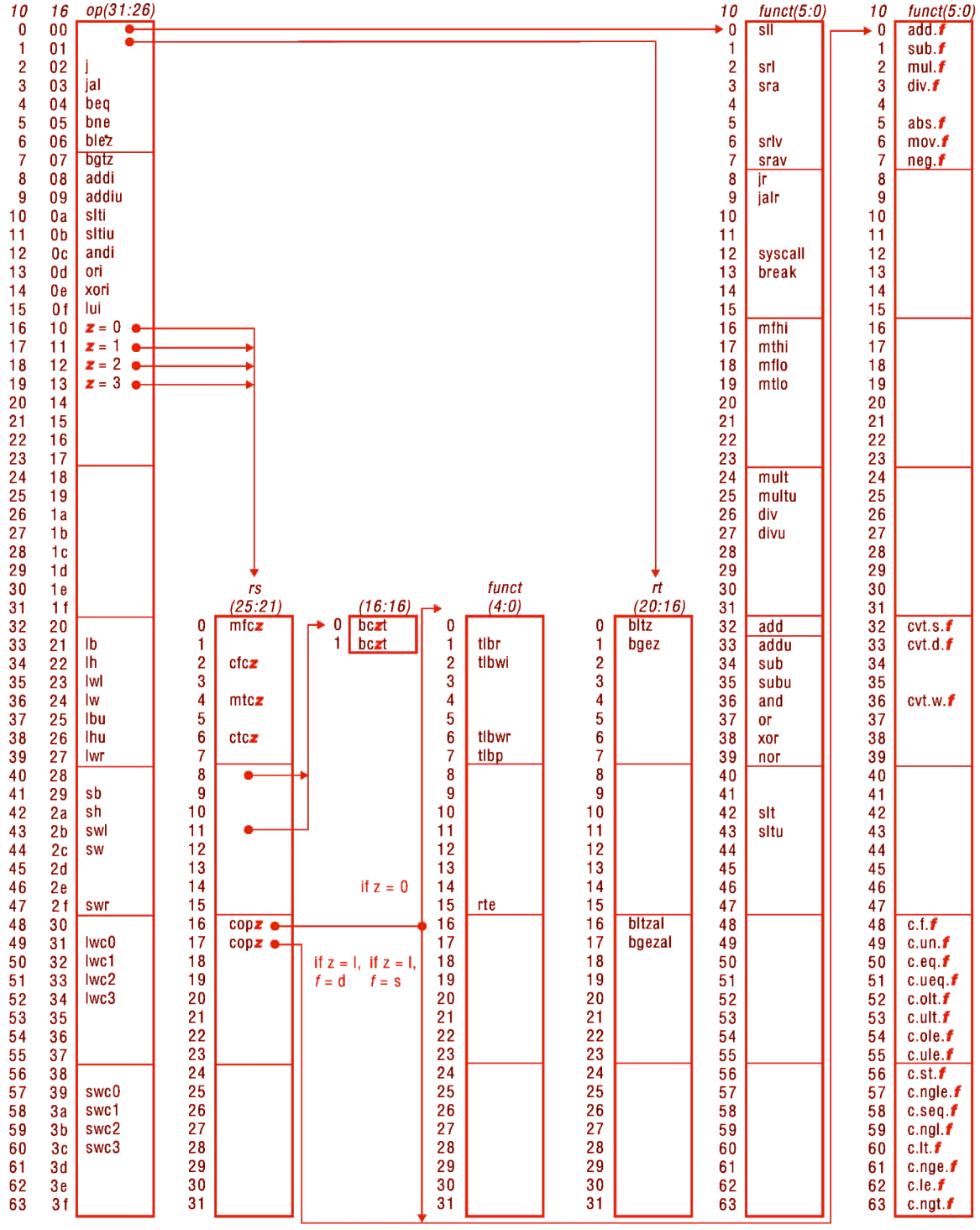


FIGURE A.19 MIPS opcode map. The values of each field are shown to its left. The first column shows the values in base 10 and the second shows base 16 for the op field (bits 31 to 26) in the third column. This op field completely specifies the MIPS operation except for 6 op values: 0, 1, 16, 17, 18, and 19. These operations are determined by other fields, identified by pointers. The last field (funct) uses "*f*" to mean "s" if rs = 16 and op = 17 or "d" if rs = 17 and op = 17. The second field (rs) uses "*z*" to mean "0", "1", "2", or "3" if op = 16, 17, 18, or 19, respectively. If rs = 16, the operation is specified elsewhere: if *z* = 0, the operations are specified in the fourth field (bits 4 to 0); if *z* = 1, then the operations are in the last field with *f* = s. If rs = 17 and *z* = 1, then the operations are in the last field with *f* = d. *(page A-54)*