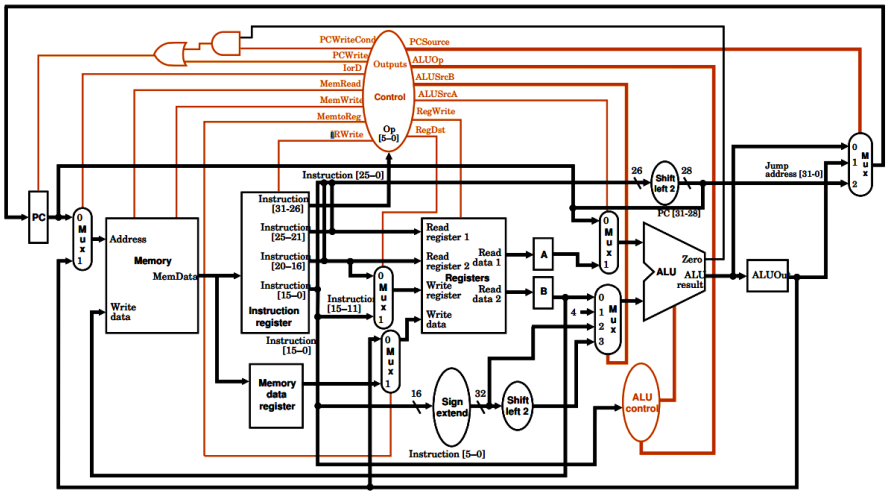
1. [6] Si consideri la CPU seguente mentre esegue il codice a lato.
a) Identificare gli eventuali casi di criticità nel codice e descrivere come vengono gestiti dalla CPU;
b) determinare tutti i valori presenti alle uscite dei registri **PC, IR, MDR, A, B, ALUOut** a **12 cicli di clock** eseguiti dall'inizio della prima istruzione.

```
0x4CC:lw $5, 0($12)
      sw $4, 5($5)
      lw $3, 0($4)
      slti $1, $3, 1
```



2. [4] Disegnare lo schema circuitale di una cella di memoria statica e descrivere il meccanismo di lettura e quello di scrittura di un bit. Disegnare la struttura circuitale globale di una RAM statica di 4M x 8bit e descrivere come viene gestita la parola di indirizzo.

3. [6] Un processore caratterizzato da una capacità di indirizzamento di 16 GB e da un bus dati di 64 bit viene dotato di una memoria cache associativa a 4 vie, di capacità totale C = 64 MByte e con linee di 64 parole. a) Dimensionare la cache determinando le dimensioni di tutti i campi; b) disegnare lo schema dettagliato di tale memoria; c) calcolare la posizione nella cache in cui viene memorizzato il byte di memoria situato all' indirizzo: 0x0...01248AC, determinando i valori in esadecimale di: byte offset, word offset, index e tag.

4. [4] Descrivere le strutture e le tecniche comunemente utilizzate per ridurre, in un sistema di memoria con cache, il tempo di scrittura di un blocco dalla memoria principale alla cache, in caso di miss.

5. [4] Si incrementano le prestazioni di calcolo di un elaboratore mediante l'introduzione di una FPU, che esegue calcoli in virgola mobile in 2 nsec; senza di essa, la CPU eseguiva gli stessi calcoli in 30 nsec. a) Quanto durerà ora l'esecuzione di un programma che esegue calcoli in virgola mobile per il 75% del tempo, se originariamente impiegava 25 secondi. b) Con che percentuale di calcoli in virgola mobile si otterrebbe una durata del programma di 5 secondi?

6. [6] Si traducano in linguaggio Assembly MIPS nativo, evitando cioè pseudo-istruzioni, le seguenti procedure in linguaggio C. Entrambe le procedure si aspettano l'argomento in **$a0** e restituiscono il risultato in **$v0**.

```
int FX(int n)
{
  if( n<5 )
     return( n );
  else
     return(FX(n/2) + FY(n-2));
}
```

```
int FY(int n)
{
  if( n<2 )
     return(2*n);
   else
     return(n+3);
}
```

System calls

	codice ($v0)	argomenti	risultato
print_int	1	$a0	
print_float	2	$f12	
print_double	3	$f12	
print_string	4	$a0	
read_int	5		$v0
read_float	6		$f0
read_double	7		$f0
read_string	8	$a0,$a1	
sbrk	9	$a0	$v0
exit	10		

Registri MIPS

0	zero		24-25	t8-t9
1	at		26-27	k0-k1
2-3	v0-v1		28	Gp
4-7	a0-a3		29	Sp
8-15	t0-t7		30	s8
16-23	s0-s7		31	Ra