

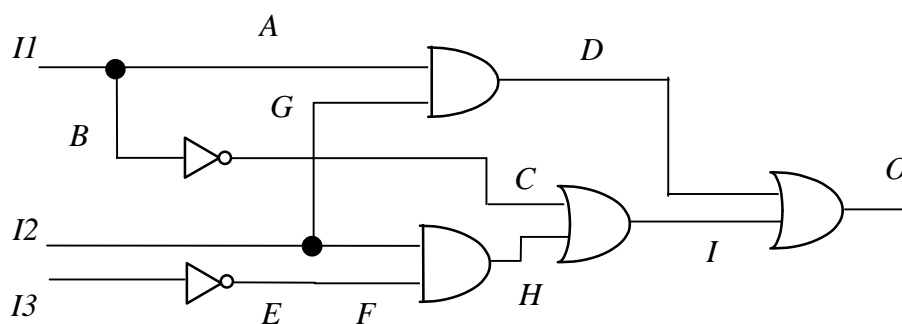
**Università di Verona**  
**Dipartimento di Informatica**

Sistemi per la Progettazione Automatica: prova finale del 12/03/02

**Cognome:** ..... **Nome:** ..... **Matricola:** .....

**Note:**     *le soluzioni devono essere opportunamente commentate,*  
              *è vietato utilizzare appunti o libri.*

- 1) Identificare tutti i vettori di test per il guasto D stuck-at 0. Quali altri guasti sono testati da questi vettori? Verificare l'esistenza di guasti ridondanti.



- Quali sono le tipologie delle tecniche di self test.

---

2) Quali sono le motivazioni alla base della progettazione a basso consumo di potenza?

- Avendo la possibilità di dividere la memoria in due blocchi separati, identificare una distribuzione delle variabili di questo programma sui due blocchi in modo da minimizzare il consumo di potenza. Motivare la scelta.

```
int a[MAX], b[MAX], c[MAX];
int j, i;
for (i = 0; i < MAX; ++i){
    a[i] = b[i] = i;
}
for (i = 0; i < MAX; ++i) {
    c[i] = a[i]+b[i];
    for (j = 0; j < 10*MAX; ++j)
        a[i] = a[i]+b[i];
}
```

---

3) Descrivere a quali livelli del flusso di progettazione è possibile applicare tecniche di verifica formali.

- Volendo verificare la correttezza della seguente realizzazione di un registro seriale/seriale, quali proprietà potrebbero essere scritte?

```
entity REGISTRO_SERIALE_SERIALE is
  generic (N: integer := 8);
  port ( I0: in bit;
         CLOCK: in BIT;
         O: out bit );
end REGISTRO_SERIALE_SERIALE;

architecture RTL of REGISTRO_SERIALE_SERIALE is
begin
  process(CLOCK)
    variable REG: bit_vector(N-1 downto 0);
  begin
    if (CLOCK = '1') then
      REG := I0 & REG(N-1 downto 1);
    end if;
    O <= REG(0);
  end process;
end RTL;
```



## Università di Verona

### Dipartimento di Informatica

Sistemi per la Progettazione Automatica: recupero prova del 12/03/02

- 1) Sia dato il seguente algoritmo ( $x$  e  $y$  sono variabili i cui valori iniziali sono forniti dall'esterno;  $a$ ,  $b$  e  $c$  sono costanti fornite dall'esterno; variabili e costanti sono tutte di 8 bit)

```
x1 = (x + z) * y + (x - b) / y;  
y1 = (x + b) / c - y - b;  
if (x1 > y1) then  
    z = (x1 - y1) * a;  
else  
    z = (y1 - x1) * b;  
x = x1 + z;  
y = y1 * z;
```

- si progetti la FSMMD che realizza tale algoritmo *a latenza minima*, cercando di ridurre il più possibile il numero di risorse. La FSMMD deve essere descritta in VHDL con un unico processo.

---

---