**SIMULAZIONE PRATICO SUL β2018 – TELECOMUNICAZIONI**

**Nome e Cognome:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Classe: \_\_\_\_\_\_\_\_\_\_\_\_ Data: \_\_\_\_\_\_\_**

videoLezione in streaming su You Tube con svolgimento completo della simulazione: <https://www.youtube.com/watch?v=o-nxAQgrvhE>

Vogliamo realizzare con il β2018 un contatore in avanti da 0 fino a un numero massimo (modulo) variabile. Il valore del modulo di conteggio viene salvato all’inizio in un registro e i numeri del conteggio vengono visualizzati in sequenza con OUT (0), A

Per esempio, inserendo come valore del modulo 9, il contatore produrrà in uscita 0,1,2,3,4,5,6,7,8 e poi ricomincerà daccapo all'infinito.

**AL TERMINE DELLA VERIFICA**

Inviare all'insegnante una cartella zippata di nome CognomeN.zip (es. PerloG.zip) contenente:

A) questo file Word con la mappa di memoria (punto 4)

B) i file *CognomeN.c, CognomeN.asm, CognomeN.b18, CognomeN.circ*

C) i file *CognomeN2.c, CognomeN2.asm, CognomeN2.b18, CognomeN2.circ* (solo se svolgi l'ultimo punto MIGLIORAMENTI)

**ATTENZIONE**: Questa è una simulazione. Non devi inviare nulla al termine. Le istruzioni precedenti servono solo in preparazione al giorno della verifica.

**1) LINGUAGGIO AD ALTO LIVELLO**

Scrivi il programma precedente in C (usa printf e scanf al posto delle istruzioni IN e OUT) e usando i nomi dei registri (A,B,C,D) come variabili. Testa la correttezza e il funzionamento del programma qui <https://www.onlinegdb.com/online_c_compiler>

Salva il file con nome *CognomeN.c*

**2) PROGRAMMA IN ASSEMBLY**

Traduci il programma precedente in linguaggio Assembly del β2018. Commenta le istruzioni con la corrispondente istruzione in linguaggio C. Vedi l'esempio qui sotto:

OUT (0), A) ; printf("%d",A)

Salva il programma con nome *CognomeN.asm*.

**3) PROGRAMMA IN LINGUAGGIO MACCHINA**

Assembla il programma con **Kappembler** (<http://www.logisim.altervista.org/>), verifica che non ci siano errori e salva il programma assemblato (Esporta ROM per Logisim) con nome *CognomeN.b18*

**4) MAPPA DI MEMORIA**

Completa la mappa di memoria del programma qui sotto, scrivendo accanto ad ogni istruzione il corrispondente codice operativo e il commento (in C o altro linguaggio ad alto livello) e inserendo il valore degli operandi (nelle istruzioni con operando). Aggiungi righe alla tabella a piacere (ho compilato la prima riga per darti un esempio, ma puoi modificarla)

|  |  |  |  |
| --- | --- | --- | --- |
| **Indirizzo in memoria** | **Istruzione Assembly** | **Codice operativo/ operando** | **Commento** |
| 0000h | IN A,(0) | DB | scanf("%d",&A) |
| 0001h |  | 00 |  |
| 0002h |  |  |  |

**5) CIRCUITO CON IL β2018**

In un file Logisim di nome *CognomeN.circ* realizza il contatore precedente, visualizzando la sequenza di conteggio su una coppia di display esadecimali e acquisendo il modulo di conteggio con un input pin a 8 bit.

**6) MIGLIORAMENTI**

Inserendo un valore negativo per il modulo, il contatore deve contare all'indietro. Es. se inserisco il valore -9, il contatore conta 8,7,6,5,4,3,2,1,0 e poi riprende da 8.

Se si fa questo punto, occorre produrre i file *CognomeN2.c, CognomeN2.asm, CognomeN2.b18, CognomeN2.circ* (fai una copia dei file che hai già scritto e rinominali per poterli poi modificare).