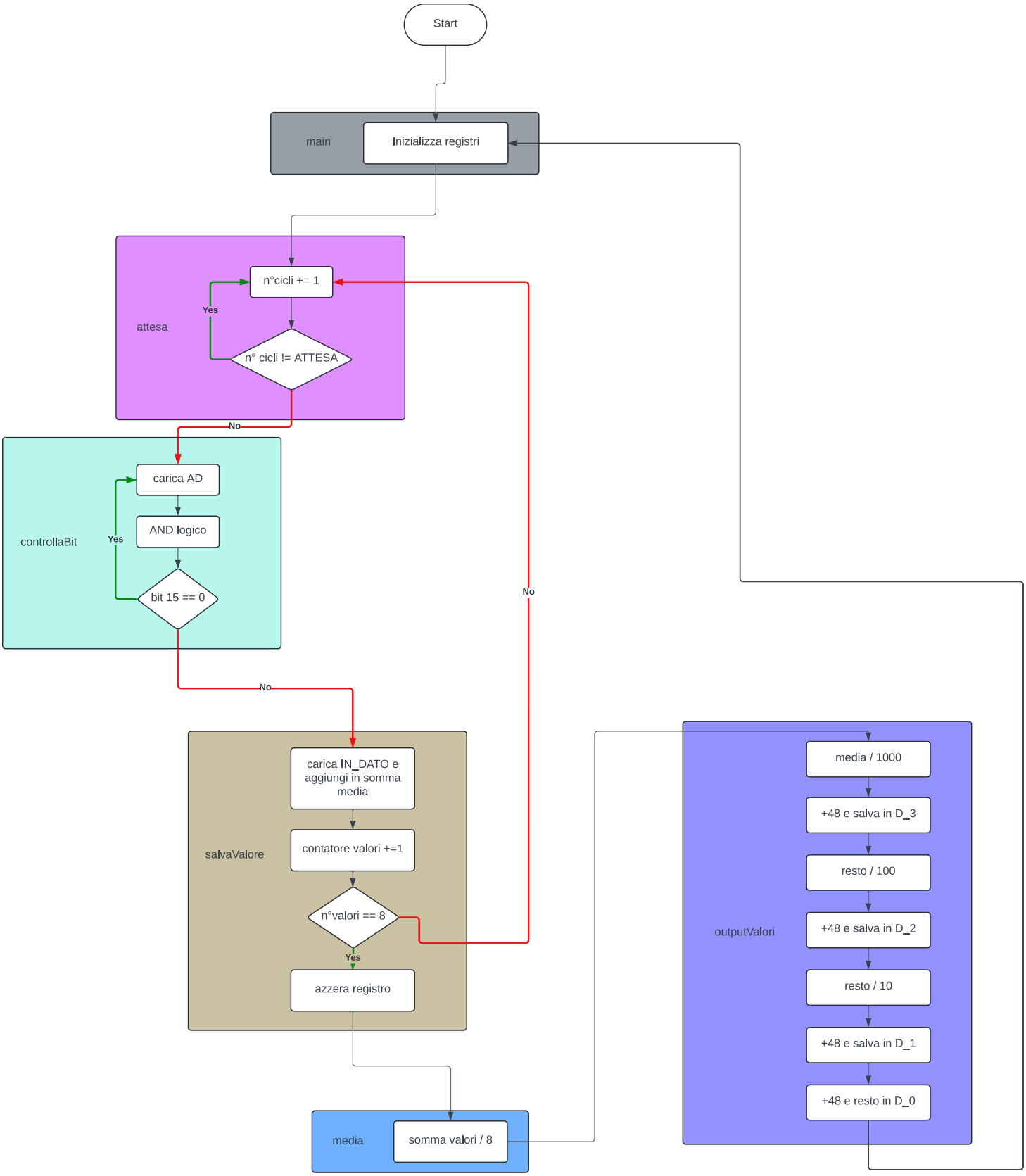
Progetto calcolatori elettronici - AA 2023/24

Esercizio 005-23-24

Sara Porco 1093346 - Samuele Stasi 1093316 - Tommaso Maistrello 1093176


Start
main
Inizializza registri
attesa
n°cicli += 1
Yes
n° cicli != ATTESA
No
controllaBit
carica AD
AND logico
Yes
bit 15 == 0
No
No
salvaValore
carica IN_DATO e aggiungi in somma media
contatore valori +=1
n°valori == 8
Yes
azzera registro
media
somma valori / 8
outputValori
media / 1000
+48 e salva in D_3
resto / 100
+48 e salva in D_2
resto / 10
+48 e salva in D_1
+48 e resto in D_0

```
.data 0x10000000

AD:      .half 0 # mettiamo half per ad perchè ci serve una cella a 16 bit,
                 # e mettiamo 0 per riempire la cella di 0
IN_DATO: .half 0
D_0:     .word 0 # perchè sono degli interi a 32 bit(word 32 bit) cella unità
D_1:     .word 0 # cella decine
D_2:     .word 0 # cella centinaia
D_3:     .word 0 # cella migliaia
ATTESA:  .word 5000000  # bisogna spettare 100 milli sec per leggere una cella
                        # alla volta
                        # 0.1 / 2 * 100 000 000 hz

.text
.globl main
main:
  la $t0, ATTESA      # carica l'indirizzo della cella attesa
    lw $t1, 0($t0)      # setta il valore di $t1 a $t0
    li $t0, 0           # setta il valore di $t0 a 0
    li $t2, 0           # setta il valore di $t2 a 0
    li $t3, 0           # setta il valore di $t3 a 0
    li $t4, 8           # setta il valore di $t4 a 8
    li $s0, 0           # setta il valore di $s0 a 0
    li $s1, 0           # setta il valore di $s1 a 0
    li $s2, 0           # setta il valore di $s2 a 0
    li $s3, 0           # setta il valore di $s3 a 0
    j attesa            # salto all'attesa

# label in minuscolo sono le funzioni
attesa:
    addi $t0, $t0, 1           # aggiunge 1 al valore di $t0
    bne  $t0, $t1, attesa      # se il valore di $t0 è diverso da ATTESA riparte
                               # con il ciclo
    j controllaBit             # esce dal ciclo andando a controllaBit

controllaBit:
    lh $t0, AD                   # carica il valore di AD nel registro $t0
                                 # (lh = load half {16 bit}). il valore AD viene dalla
                                 # memoria
    li $t1, 0x8000                 # li carica un valore numerico nel registro $t1
    and $t2, $t0, $t1            # fa l'AND tra il valore di AD e 1000000000000000
                                 # per vedere se il 16esimo bit è 0 o 1, il
                                 # risultato si salva in $t2
    beq $t2, $zero, controllaBit # se il valore dell'and è 1 salta al label per
                                 # salvare il valore
    j salvaValore                # ricontrolla il valore se è 0 fino a che non
                                 # diventa 1
```

```
salvaValore:
    lh $t3, IN_DATO             # carica l'indirizzo della cella di IN_DATO
                                # nel registro $t3
    add $s0, $s0, $t3           # aggiunto a un nuovo registro il valore di
                                # IN_DATO , e lo fa 8 volte, tutti i valori
                                # vengono messi qui dentro
    addi $s1, $s1, 1            # aggiunge 1 a $s1 per contare i cicli fatti
    beq $s1, $t4, media         # se sono arrivato a 8 cicli passo al label per
                                # fare la media
    li $t0, 0                   # setta il valore di $t0 a 0
    j attesa                    # ricomincia saltando ad attesa

media:
  div $s0, $t4                  # Divido gli 8 valori sommati($s0) per 8($t4) per
                          # fare la media
  mflo $s2                # l'istruzione div salva la parte prima della virgola
                          # nel registro speciale HI, cioè i 32 bit.
                          # Li salvo in $s2 usando mfhi
  j outputValori          # Salto ad outpuitValori

outputValori:
    li $t0, 1000          # Costante per estrarre le migliaia
    div $s2, $t0          # $s2 / 1000                    1350
    mflo $t1              # $t1 = Quotiente (migliaia)    1
    mfhi $s2              # $s2 = Resto                   350

    addi $t1, $t1, 48     # Converti in ASCII
    la $t2, D_3           # Carica l'indirizzo di D_3
    sw $t1, 0($t2)        # Scrive la cifra delle migliaia in D_3

    li $t0, 100           # Costante per estrarre le centinaia
    div $s2, $t0          # $s2 / 100
    mflo $t1              # $t1 = Quotiente (centinaia)
    mfhi $s2              # $s2 = Resto

    addi $t1, $t1, 48     # Converti in ASCII
    la $t2, D_2           # Carica l'indirizzo di D_2
    sw $t1, 0($t2)        # Scrive la cifra delle centinaia in D_2

    li $t0, 10            # Costante per estrarre le decine
    div $s2, $t0          # $s2 / 10
    mflo $t1              # $t1 = Quotiente (decine)
    mfhi $s2              # $s2 = Resto

    addi $t1, $t1, 48     # Converti in ASCII
    la $t2, D_1           # Carica l'indirizzo di D_1
    sw $t1, 0($t2)        # Scrive la cifra delle decine in D_1
```

```
addi $s2, $s2, 48    # Converti in ASCII
la $t2, D_0          # Carica l'indirizzo di D_0
sw $s2, 0($t2)       # Scrive la cifra delle unità in D_0

j main               # salto al main per ricominciare tutto
```