



SISTEMI EMBEDDED – PROGETTO FINALE

TASK 6 - REGRESSIONE LINEARE

S. BARONE - MAT. M63/610

A. DI MARTINO - MAT. M63/654

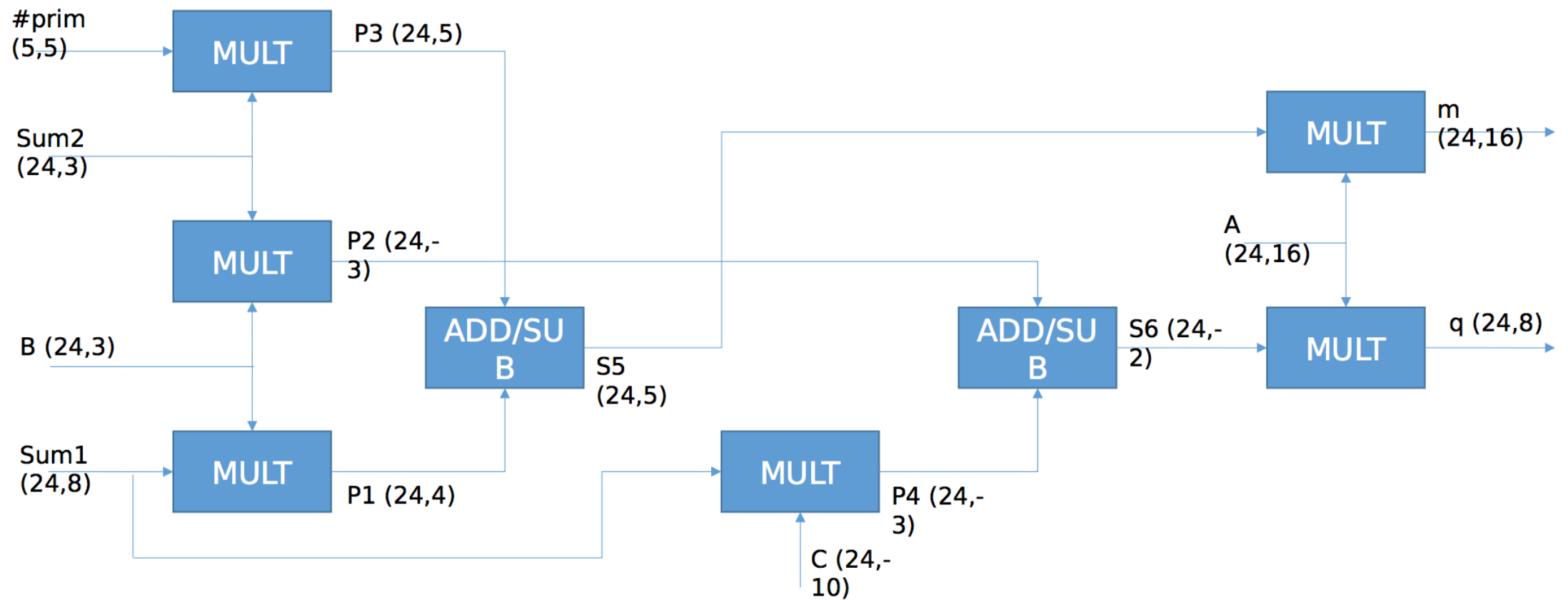
S. FIORILLO - MAT M63/624

P. LIGUORI - MAT. M63/556

TASK 6

- Si realizzi un IP Core che implementi le operazioni necessarie per ricavare i coefficienti m e q di una retta interpolatrice.
- Nell'ambito della statistica inferenziale questo metodo prende il nome di Regressione Lineare ed è usata per fare stime e previsioni dato un certo set di dati.
- Si tenga conto delle criticità introdotte dal calcolo con valori a **virgola fissa**.

SCHEMA DI PARTENZA

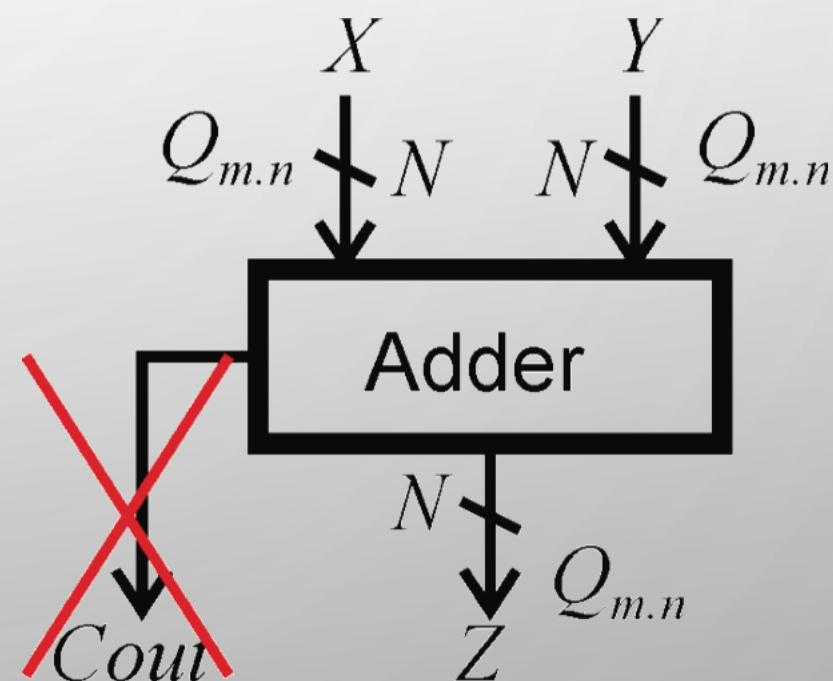


RAPPRESENTAZIONE SIGNED FIXED-POINT

- La convenzione utilizzata per rappresentare i numeri signed in fixed point è $Q_{m,n}$ dove:
 - m rappresenta il peso dell'MSB
 - n rappresenta il peso dell'LSB
- Caratteristiche:
 - Massimo valore rappresentabile: $2^m - 2^{-n} \approx 2^m$
 - Minimo valore rappresentabile: -2^m
 - Risoluzione: 2^{-n}

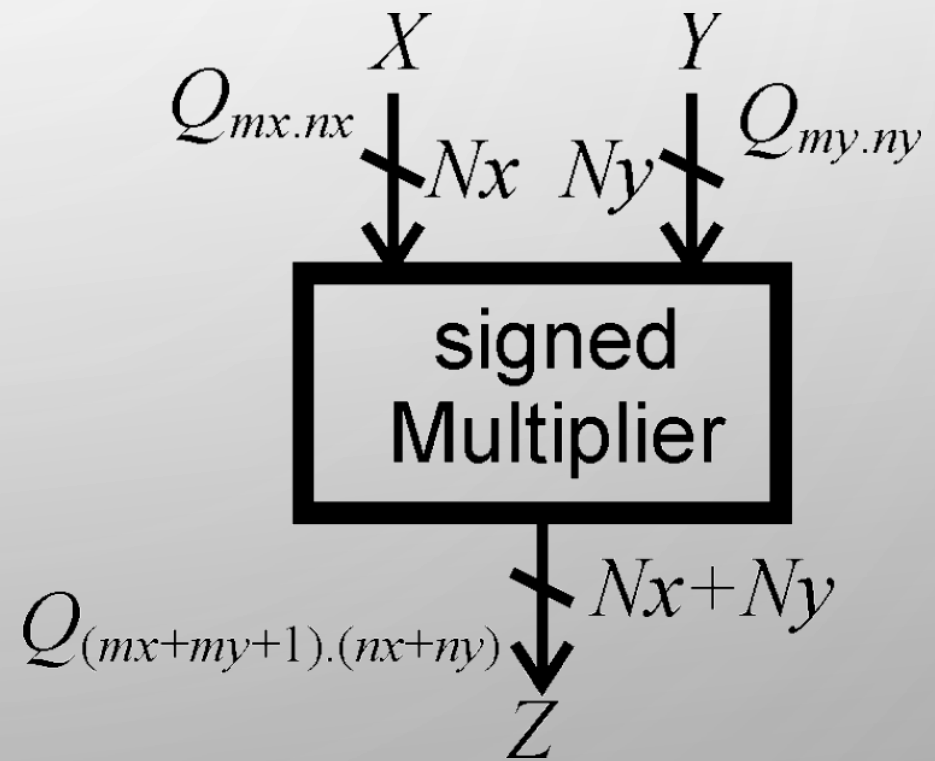
SOMMA SIGNED FIXED-POINT

- Nel caso di addizione / sottrazione, i due ingressi X ed Y devono essere nella medesima rappresentazione $Q_{m.n}$

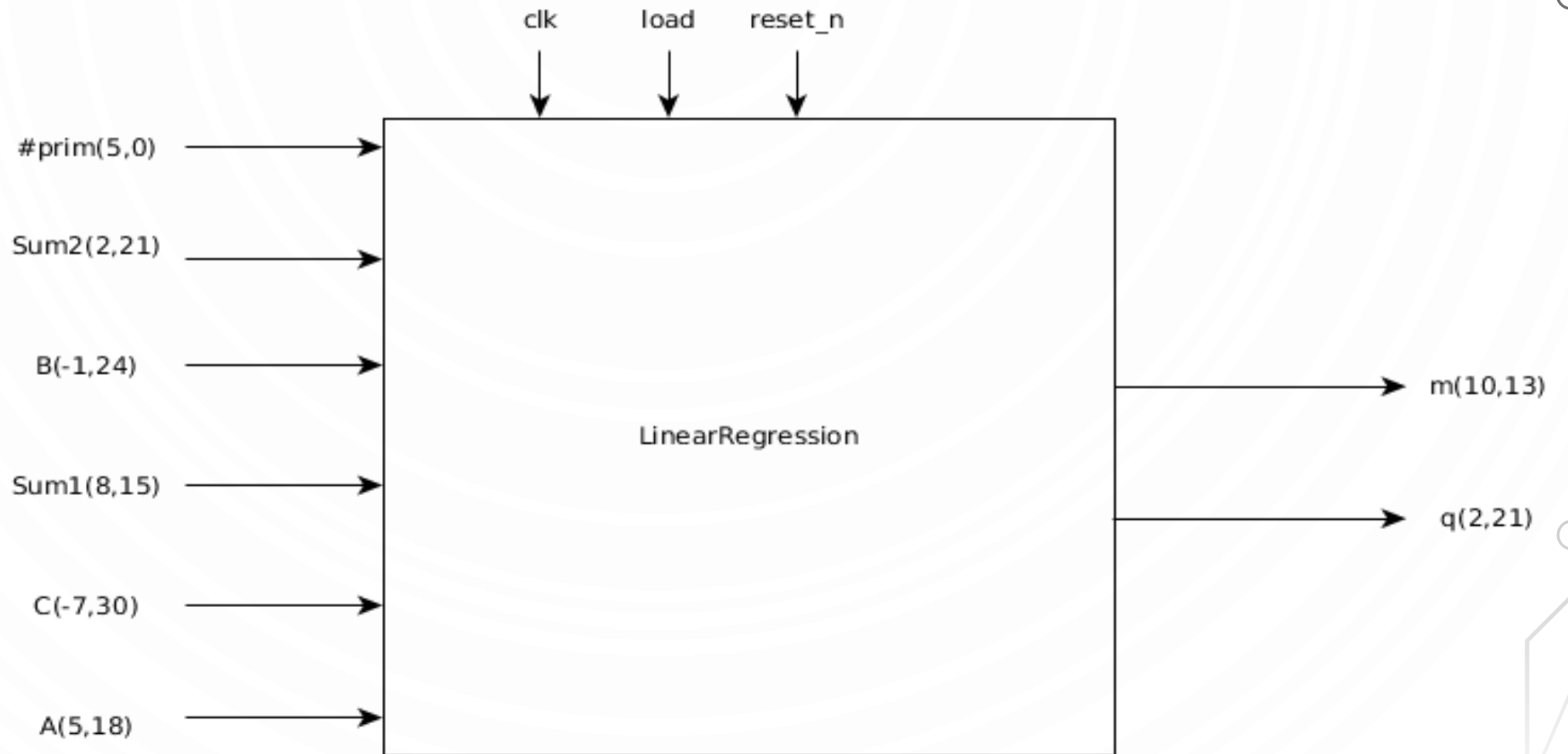


MOLTIPLICAZIONE SIGNED FIXED-POINT

- Consideriamo X espresso su N_x bit e Y su N_y bit.
- La rappresentazioni degli ingressi può essere diversa ($Q_{mx.nx}$ e $Q_{my.ny}$)
- La rappresentazione di uscita sarà $Q_{mx+my+1.nx+ny}$



INTERFACCIA COMPONENTE



RAPPRESENTAZIONE INPUT (1)

- La prima operazione da fare è capire la rappresentazione dei parametri. La dimensione dei parametri di ingresso e uscita è su 24 bit.
- Nel codice matlab ci sono dei parametri di ingresso che assumono valori costanti:
 - $A = \text{div1} = 30.769230769230795 \rightarrow Q_{5.18}$
 - $B = \text{sx} = 0.3 \rightarrow Q_{-1.24}$
 - $C = \text{sx2} = 0.0049 \rightarrow Q_{-7.30}$
 - $\text{prim} = n = 25 \rightarrow Q_{5.0}$

```
67 % Compute constants:
68 - n=25; % Number of elements for test
69 - sx=sum(x);
70 - s2x=(sum(x)).^2;
71 - sx2=sum(x.^2);
72 - div1=1/((n*sx2)-s2x);
73
```


RAPPRESENTAZIONE INPUT (2)

- I restanti due ingressi Sum1 e Sum2 cambiano perché dipendono da un valore random sig.
- Per capire la rappresentazione di Sum1 e Sum2 abbiamo effettuato 10M di test con Matlab per trovare il valore massimo assunto da tali parametri:
 - $\text{Sum1} \approx [-3; 189] \rightarrow Q_{8.15}$
 - $\text{Sum2} \approx [-0.09; 3] \rightarrow Q_{2.21}$

```
73 % Compute operations
74 - s1=sum(sig);
75 - s2=sum(x.*sig);
76
```

RAPPRESENTAZIONE OUTPUT

- Come per Sum1 e Sum2, abbiamo effettuato 10M di test in Matlab per trovare il massimo range di valori assunti dalle uscite m e q:
 - $m \approx [-27; 606]$ $\rightarrow Q_{10.13}$
 - $q \approx [-2.62; 2.59]$ $\rightarrow Q_{2.21}$

```
77 - m = (n*s2 - s1*sx) *div1;  
78 - q = (s1*sx2 - sx*s2) *div1;  
79
```

COMPONENTI UTILIZZATI

- Dalla formule per il calcolo dei parametri m e q , vediamo che abbiamo bisogno di:
 - Sei moltiplicatori
 - Due sottrattori
- Abbiamo utilizzato il moltiplicatore e il sottrattore istanziati dal sintetizzatore della Xilinx.
- Bisogna assegnare la giusta rappresentazioni alle uscite dei componenti per poter effettuare un corretto troncamento dei bit in eccesso.

```
77 - m = (n*s2 - s1*sx) *div1;  
78 - q = (s1*sx2 - sx*s2) *div1;  
79
```

RAPPRESENTAZIONI SEGNALE

- Il moltiplicatore MULT2 moltiplica B ($Q_{-1.24}$) e Sum2 ($Q_{2.21}$), di 24 bit ciascuno.
- In uscita abbiamo mult2_out con rappresentazione $Q_{2.45}$, quindi su 48 bit.
- Dai test Matlab, il range dei valori assunti dall'uscita di MULT2 è $[-0.02; 0.9090]$.
- Volendo rappresentare l'uscita su 24 bit, la rappresentazione ideale è $Q_{0.23}$
- Per passare da $Q_{2.45}$ su 48 bit a $Q_{0.23}$ su 24 bit tronciamo:
 - 2 bit in testa;
 - 22 in coda.
- Questo procedimento va applicato a tutti i componenti istanziati.

RAPPRESENTAZIONI SEGNALI (2)

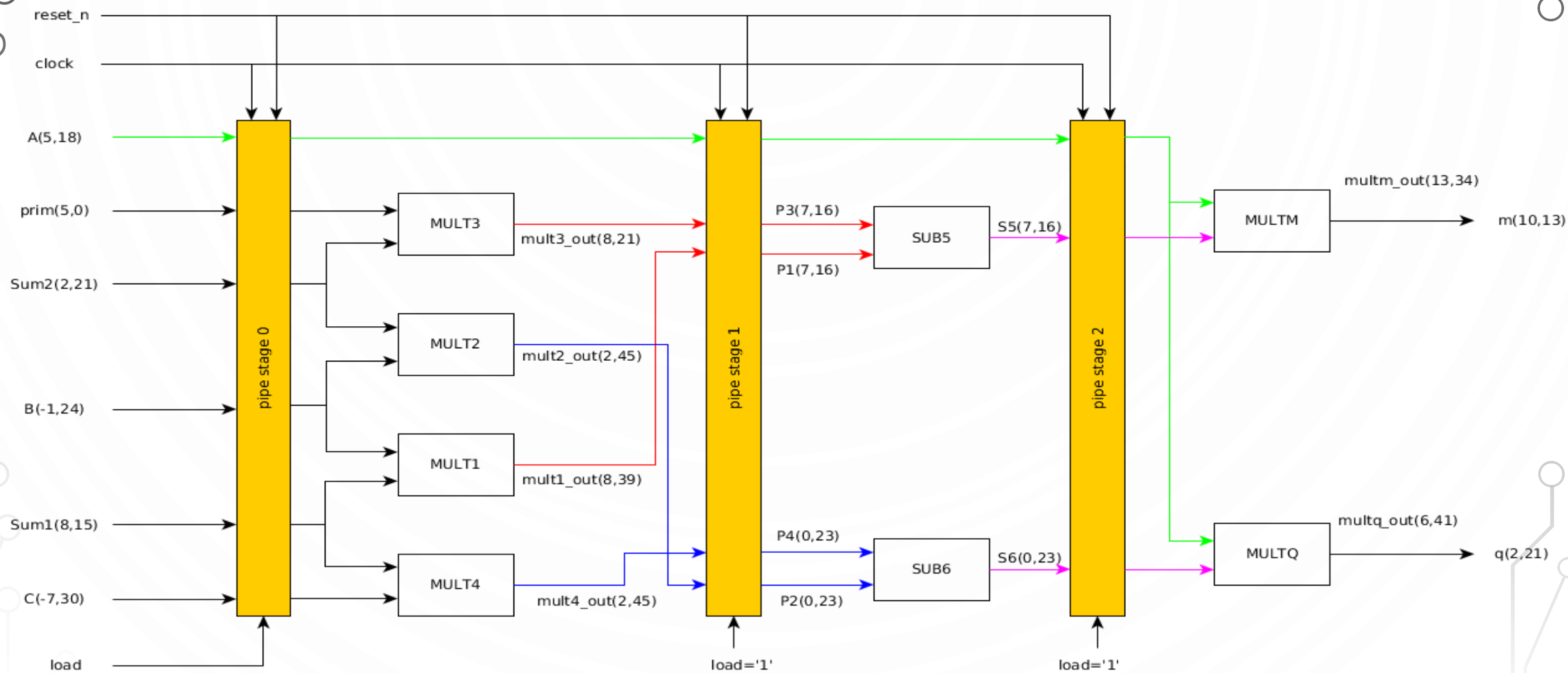
Ingressi	Componente	Uscita	RANGE [min; max]	Rappresentazione	Bit troncati
B ($Q_{-1.24}$) Sum1 ($Q_{8.15}$)	MULT1	mult1_out $\rightarrow Q_{8.39}$ (48 bit)	[-0.3; 56]	P1 $\rightarrow Q_{7.16}$ (24 bit)	1 in testa 23 in coda
Sum2 ($Q_{2.21}$) B ($Q_{-1.24}$)	MULT2	mult2_out $\rightarrow Q_{2.45}$ (48 bit)	[-0.02; 0.9090]	P2 $\rightarrow Q_{0.23}$ (24 bit)	2 in testa 22 in coda
Sum2 ($Q_{2.21}$) Prim ($Q_{5.0}$)	MULT3	mult3_out $\rightarrow Q_{8.21}$ (30 bit)	[-2.37; 80]	P3 $\rightarrow Q_{7.16}$ (24 bit)	1 in testa 5 in coda
Sum1 ($Q_{8.15}$) C ($Q_{-7.30}$)	MULT4	mult4_out $\rightarrow Q_{2.45}$ (48 bit)	[-0.0049; 0.95]	P4 $\rightarrow Q_{0.23}$ (24 bit)	2 in testa 22 in coda

- N.B. La rappresentazione ideale di P1 è $Q_{6.17}$, tuttavia, dato che va in ingresso a SUB5 insieme a P3 (con rappresentazione $Q_{7.16}$), assegniamo ad esso la stessa rappresentazione di quest'ultimo segnale (gli ingressi dell'adder/subtractor devono essere nella stessa rappresentazione).

RAPPRESENTAZIONI SEGNALI (3)


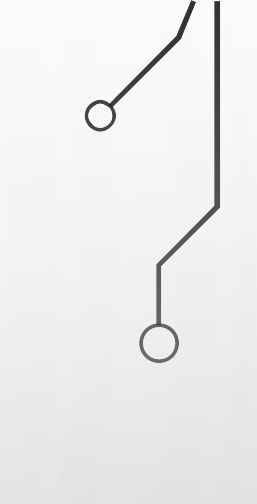

Ingressi	Componente	Uscita	RANGE [min; max]	Rappresentazione	Bit troncati
P3 (Q _{7.16}) P1 (Q _{7.16})	SUB5	S5 → (Q _{7.16}) (24 bit)	[-0.13; 19.21]	Resta la stessa degli ingressi	//
P4 (Q _{0.23}) P2 (Q _{0.23})	SUB6	S6 → (Q _{0.23}) (24 bit)	[-0.08; 0.08]	Resta la stessa degli ingressi	//
A (Q _{5.18}) S5 (Q _{7.16})	MULTM	multM_out → Q _{13.34} (48 bit)	[-27; 606]	m → Q _{10.13} (24 bit)	3 in testa 21 in coda
A (Q _{5.18}) S6 (Q _{0.23})	MULTQ	multQ_out → Q _{6.41} (48 bit)	[-2.62; 2.59]	q → Q _{2.21} (24 bit)	4 in testa 20 in coda

SCHEMA A BLOCCHI FINALE





TEST PER CALCOLO ERRORI

- Per verificare il corretto funzionamento del componente implementato abbiamo effettuato 10k test, confrontando i risultati ottenuti da Matlab e Vivado attraverso uno script Matlab.
 - Abbiamo generato attraverso Matlab i valori degli input non costanti (Sum1 e Sum2) in binario assegnando loro la corretta rappresentazione fixed point (funzione quantizer).
 - I valori generati sono stati utilizzati per calcolare le uscite m e q sia da Matlab e sia da Vivado.
- 
- 
- 

TEST PER CALCOLO ERRORI (2)



- Le uscite calcolate sono state riconvertite in numerico, ed abbiamo stimato gli errori relativi ed assoluti commessi dal componente implementato in Vivado rispetto ai risultati forniti da Matlab:
 - Errore Assoluto $m = 2.0195e-04$
 - Errore Relativo $m = 7.0579e-07$
 - Errore Assoluto $q = 1.9271e-06$
 - Errore Relativo $q = 1.4026e-06$

OCCUPAZIONE AREA

AREA	TOTALE	TOTALE %
SLICE LUT	50	5 %
SLICE REGISTER	223	1 %
DSP	11	14 %



REPOSITORY GITHUB

- Tutti i file del progetto realizzato, gli script Matlab utilizzati e i risultati dei test effettuati sono disponibili sul seguente repository github:
 - <https://github.com/piliguori/Linear-Regression>
- 
- 



RIFERIMENTI

- DSP-03-ProblematicheNumeriche.pdf – D. De Caro
- 