Il sensore di concentrazione deve rilevare 1900 livelli, di conseguenza basta un convertitore a 11 bit per ottenere la risoluzione voluta. Questo però può portare a una differenza di potenziale tra due livelli adiacenti molto bassa se il range delle tensione non è esteso.

Supponiamo il sensore venga alimentato a 12Volt e generiamo, otteniamo quindi una differenza fra due livelli di 12/2^11 = 4mV.

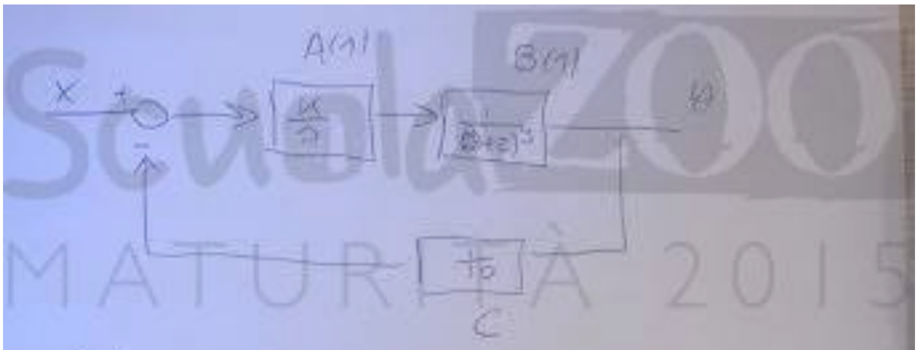
Questi dati vengono incapsulati attraverso un microcontrollore che gestisce il protocollo di comunicazione fra sensore e il controllore del processo, in modo da garantire affidabilità nella trasmissione dei dati. Questo può avvenire per esempio tramite protocollo Ethernet o Can. Inoltre in questo modo creare un canale di 11 bit connesso direttamente al controllore generale (ed evitiamo disturbi addizionali, tutto è fatto in loco nel sensore della concentrazione).

Il codice per calcolare i dati richiesti è il seguente:

```
typedef struct
{
      int letturaPH_1;
      int letturaPH_2;
      int concentrazione;
}letturaCampioni;

void calcola(letturaCampioni *dati, int numerototale)
{
      int x = 0;
      float media=0;
      int concInf500=0; //numero processi con concentrazione < 500

      for (x=0; x < numerototale; x++)
      {
            media = media+dati[x].concentrazione; //sommo tutti i valori
            if (dati[x].concentrazione < 500) concInf500=concInf500+1;
      }
      media = media/numeroTotale; //somma dei valori/numero totale

      printf("Media concentrazione %f\n", media);
      printf("Numero processi con concentrazione < 500: ", concInf500);
}
```

```
*/
void controllo(int *y2, letturaCampioni *dati, int *numeroProcesso)
{
      STATI stato = START;

      while(1)
      {
            switch(stato_attuale)
            {
            case START:
                  if (*y2 > 0 ) stato = AGENT_A;
                  break;
            case AGENT_A:
                  release_agent(0); /*rilascia l'agente A*/
                  wait(y2);
                  stato = READ_SPH;
                  break;
            case READ_SPH1:
                  dati[*numeroProcesso].letturaPH_1 = read_sensor(0);
/* legge dal sensore del Ph*/
                  if (dati[*numeroProcesso].letturaPH_1 == 0 ||
dati[*numeroProcesso].letturaPH_1 == 0x7)
                  {
                        led(1);
                        resetData(dati, numeroProcesso);
                        stato = RESET;
                  }
                  stato = AGENT_B;
                  break;
            case AGENT_B:
                  release_agent(1); /*rilascia l'agente B*/;
                  wait(y2);
            case  READ_SPH2:
                  dati[*numeroProcesso].letturaPH_2 = read_sensor(0);
/* legge dal sensore del Ph*/
                  wait(y2);
                  wait(y2);
                  stato= READ_SC;
                  break;
            case READ_SC:
                  dati[*numeroProcesso].concentrazione =
read_sensor(1); /*legge dal sensore della concentrazione*/
                  stato = END;
                  break;
            case RESET:
                  wait_for_reset(); //aspetto per il reset da un
agente esterno
                  led(0); //accensione led
                  stato = START;
            case END:
            default:
                  break;
            }
      }
}
```

```
*/
void controllo(int *y2, letturaCampioni *dati, int *numeroProcesso)
{
      STATI stato = START;

      while(1)
      {
            switch(stato_attuale)
            {
            case START:
                  if (*y2 > 0 ) stato = AGENT_A;
                  break;
            case AGENT_A:
                  release_agent(0); /*rilascia l'agente A*/
                  wait(y2);
                  stato = READ_SPH;
                  break;
            case READ_SPH1:
                  dati[*numeroProcesso].letturaPH_1 = read_sensor(0);
/* legge dal sensore del Ph*/
                  if (dati[*numeroProcesso].letturaPH_1 == 0 ||
dati[*numeroProcesso].letturaPH_1 == 0x7)
                  {
                        led(1);
                        resetData(dati, numeroProcesso);
                        stato = RESET;
                  }
                  stato = AGENT_B;
                  break;
            case AGENT_B:
                  release_agent(1); /*rilascia l'agente B*/;
                  wait(y2);
            case  READ_SPH2:
                  dati[*numeroProcesso].letturaPH_2 = read_sensor(0);
/* legge dal sensore del Ph*/
                  wait(y2);
                  wait(y2);
                  stato= READ_SC;
                  break;
            case READ_SC:
                  dati[*numeroProcesso].concentrazione =
read_sensor(1); /*legge dal sensore della concentrazione*/
                  stato = END;
                  break;
            case RESET:
                  wait_for_reset(); //aspetto per il reset da un
agente esterno
                  led(0); //accensione led
                  stato = START;
            case END:
            default:
                  break;
            }
      }
}
```



$$Y(s) = B(s)A(s)\left(X - \frac{Y}{10}\right)$$

$$Y(s)\left(1 + \frac{B(s)A(s)}{10}\right) = B(s)A(s)X$$

$$Y(s) = \frac{B(s)A(s)\,X(s)}{1 + \frac{B(s)A(s)}{10}}$$

FUNZIONE DI TRASFERIMENTO

$$= \frac{\frac{1}{(s+2)^3}\frac{K}{s}}{1 + \frac{K}{s(s+2)^3 10}}\,X(s) = \frac{10K}{s(s+2)^3 10 + K}\,X(s)$$

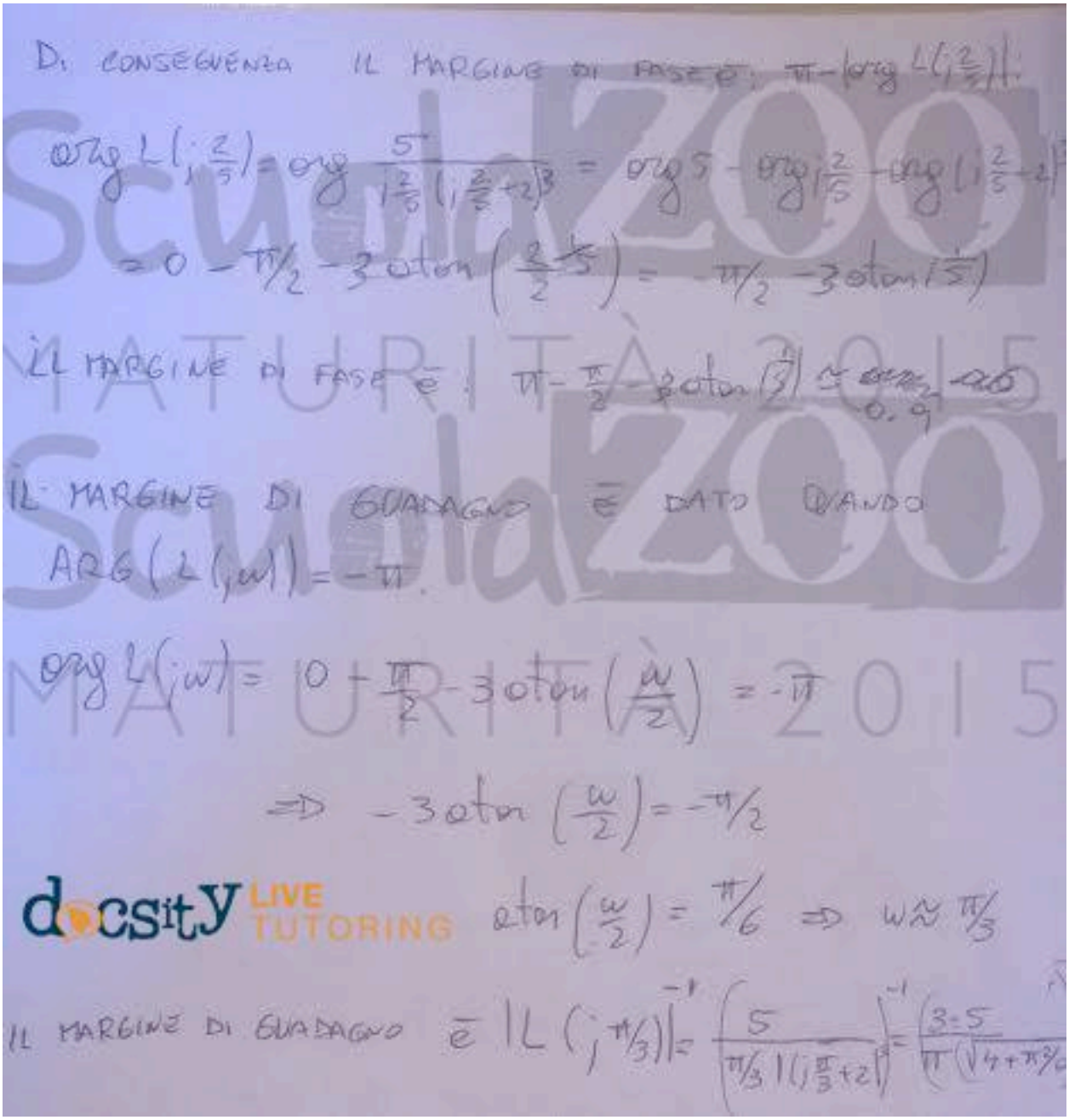
PER LA STABILITÀ ANALIZZO IL SISTEMA:

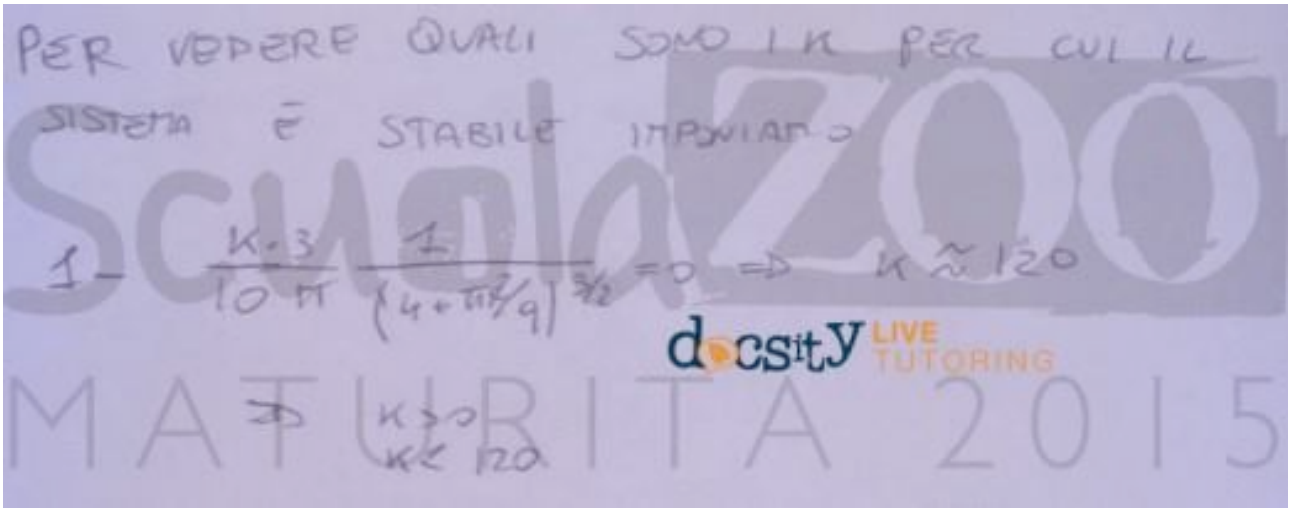
$$L(s) = \frac{K}{10}\frac{1}{s}\frac{1}{(s+2)^3}, \quad K = 50 \rightarrow L(s) = \frac{5}{s}\frac{1}{(s+2)^3}$$

LA FREQUENZA DI TAGLIO È DATA DAL DIAGRAMMA DI BODE QUANDO IL GRAFICO TAGLIA L'ASSE DELLE FREQUENZE.

OVVERO QUANDO $|L(j\omega)| = 1 \approx \omega = 0.4 = \frac{2}{5}\ \frac{rad}{sec}$

$$f = \frac{0.4}{2\pi} \approx 0.06\,Hz.$$




PER VEDERE QUALI SONO I K PER CUI IL
SISTEMA È STABILE IMPONIAMO
docsity LIVE TUTORING