Chapter 7

Switched Current Adaptive Filter for Estimating Flow rates in Micromechanical Flow Channels

In this chapter we present an adaptive filter with 96 filter taps, implemented using SI techniques. The adaptive filter is intended for use in conjunction with a micromechanical flow channel for flow estimation of very small flow rates i.e. flows in the range ml/min.

7.1 Flow Estimation

In order to estimate the flow of a fluid in a micromechanical flow channel, several methods have been presented. One widely used method is based on temperature measurement, where the fluid is heated to a constant temperature at a reference point, and then the temperature of the fluid is measured at two different locations in the flow channel. The temperature difference is then a function of the flow rate and of the characteristics of the fluid.

We have here tried to estimate the flow rate using a somewhat different approach. The idea is to view the flow channel as a lossy transmission line, whose characteristics depend on the fluid and the flow rate. Assuming that the flow channel can be modeled as a lossy transmission line, implies that the relationship between a temperature measured at two different locations down stream, can be described by the impulse response of the flow channel between these two measurement points.

Assuming that the transmission line is lossless, the impulse response must be a delta function located at the delay in the flow channel

$$h(t, v) = \delta(t - \frac{x_0}{v}) \tag{7.1}$$

Because the transmission line has some loss, that can be modeled as a lowpass filtering of the temperature in the fluid, the overall impulse response must be a convolution between this lowpass filter and the delta function. This implies that the peak of the pure lowpass filter must be located somewhere near the delta function, meaning that the flow rate can be estimated as the position of the peak of the impulse response for the flow channel.

$$h(t, v) = h(t - \frac{x_0}{v}) \tag{7.2}$$

The impulse response of the flow channel might look like the one shown in Fig. 7.1. If we sample the impulse response of the flow channel, then it can be estimated by the FIR filter

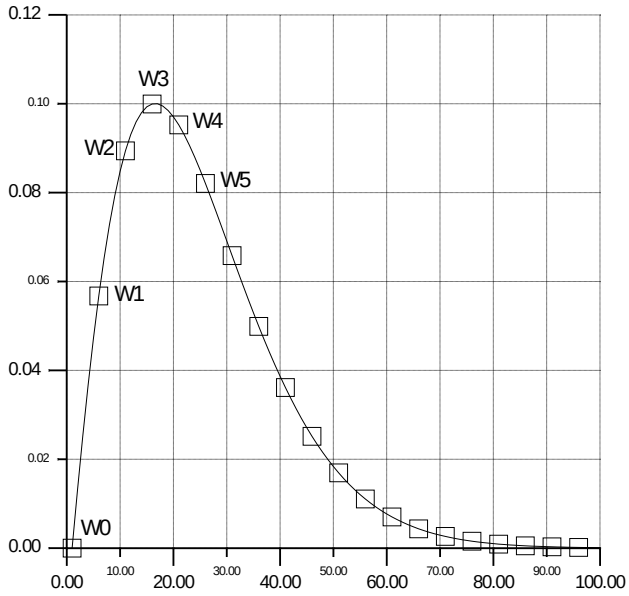

Figure 7.1: Impulse response of the flow channel

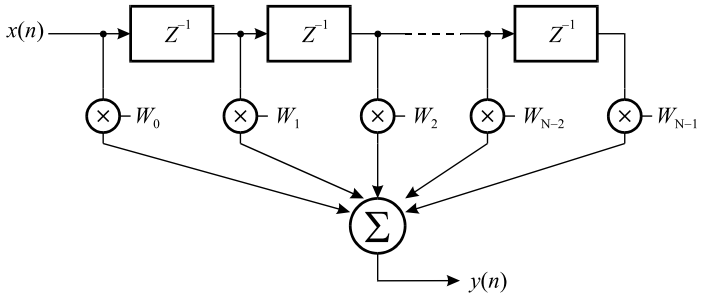

Figure 7.2: FIR filter used for estimating the impulse response

shown in Fig. 7.2. The operation of this FIR filter is very simple and can be described in the following way: if we enter a single pulse at the input of the FIR filter, the pulse will travel all the way down the delay line. For each timestep the pulse will generate an output pulse given by the tap weight W_i. By choosing the tap's so the correspond to the samples of the impulse response of the flow channel, the output sequence from the FIR filter will be the same as the sampled impulse response of the flow channel.

7.2 Adaptive Filter

In order to adjust the tap's of the FIR filter so they correspond to the samples of the impulse response for the flow channel, we will use the circuit shown in Fig. 7.3.

To estimate the flow rate of a fluid in the micro flow channel the fluid entering the channel is heated with white noise, shown as the "Heater" in Fig. 7.3. This generates some random temperature variations in the fluid. The flow of the fluid causes the temperature variations to propagate down the channel and the temperature variations are measured at two different locations in the flow channel, shown as "Sensor x" and "Sensor y" in Fig. 7.3.

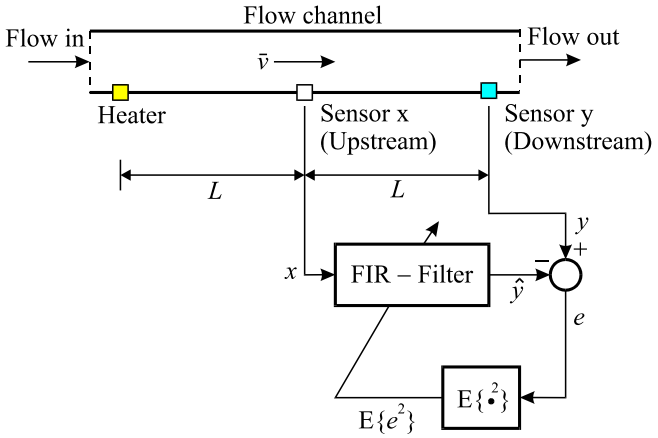

Figure 7.3: Flow channel including Adaptive filter for flow estimation

Between the two points where the temperature is measured the flow channel can be modeled as a (lossy) transmission line acting on the temperature variations in the fluid.

The flow rate is found by estimating the delay in the transmission line which is approximately the time where the impulse response sequence of the line has its maximum peak.

The heating of the fluid and the measuring of the temperature in the flow channel is achieved using diodes integrated in the flow channel [30].

The temperature at "Sensor x" is sent to the FIR filter and the output of the FIR filter is subtracted from the temperature at "Sensor y", which gives the error e between the impulse response of the flow channel and the impulse response of the FIR filter. The filter tap's of the FIR filter are now adjusted so that the mean square of the error e is minimized. This is done using the LMS algorithm, which states that the filter taps $\mathbf{W}(n)$ should be updated by the following scheme

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \mu \cdot \mathbf{x}(n) \cdot e(n) \tag{7.3}$$

$$e(n) = y(n) - \mathbf{x}^T(n) \cdot \mathbf{W}(n) \tag{7.4}$$

An adaptive filter incorporating the above equations and thereby the LMS algorithm [31] is shown in Fig. 7.4.

The adaptive filter is a discrete time system, operating on temperature samples taken from the two measurement points in the flow channel (Sensor x and Sensor y). To facilitate this the temperatures fluctuations are sampled and held (S/H) as shown in Fig. 7.4.

The heating of the fluid with white noise will cause the temperature measured at the two locations down the flow channel to fluctuate around a mean value determined by the ambient temperature and by the power of the white noise used to heat the fluid.

The mean temperature of the heated fluid is removed by taking the derivative of the sampled temperature. This is done by subtracting the previous temperature sample from the actual temperature sample, resulting in a transfer function of $(1-z^{-1})$. This is effectively a highpass filtering of the measured temperature. Because this operation is performed at both measurement points, i.e. upstream and down stream, it has no effect on the operation of the adaptive filter.

By digitizing the derivative of the sampled temperature into two levels (i.e. a binary signal) we introduce a very strong nonlinearity into the measured temperature. This will increase the variance of the filter taps in the adaptive filter which can be compensated for by

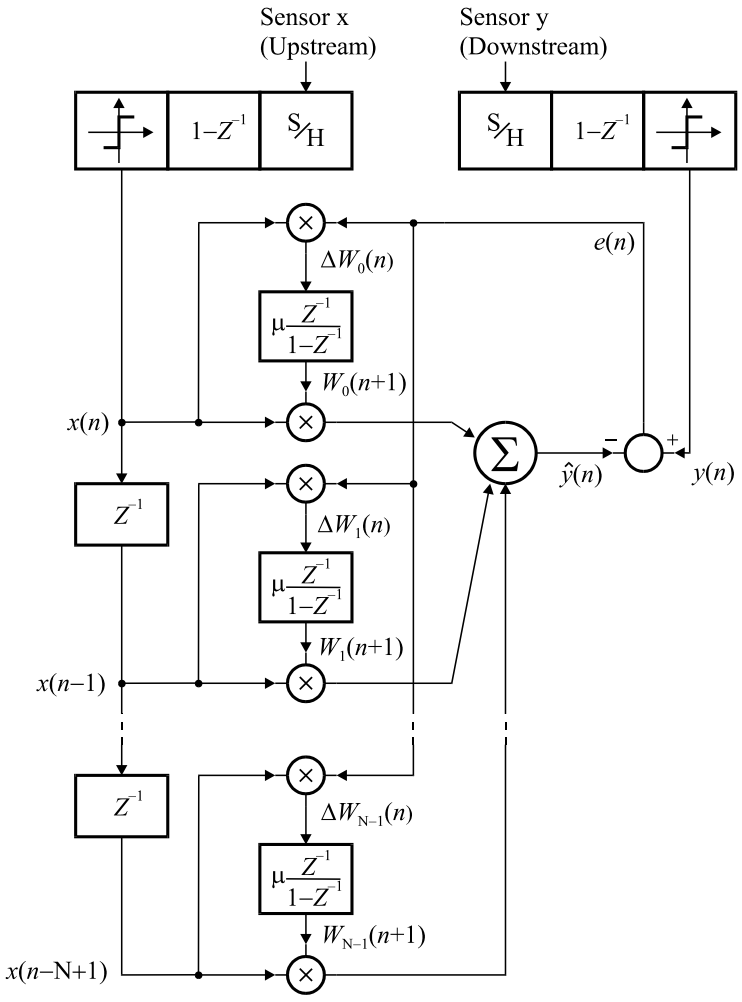

Figure 7.4: Adaptive filter incorporating the LMS algorithm

decreasing the update value μ, used by the LMS algorithm. The major advantage associated with the digitization of the measured temperature is that the hardware complexity of the adaptive filter is reduced considerably [32].

Because we have digitized the temperature samples, the tapped delay can be implemented as a digital shift register and all of the multiplication's in the adaptive filters are replaced by either a simple addition of a filter tap or not [32].

The delay of the thermal fluctuations between the two measurement points is approximately given by the peak of the impulse response sequence i.e. it can be found as the numerically largest filter tap in the adaptive filter. The resolution of the estimated delay is therefore determined by the time between two taps in the tapped delay line (FIR - Filter) which in turn is determined by the sampling frequency and the number of taps.

Because we only need to identify the numerically largest filter tap, the adaptation of the adaptive filter does not have to proceed until the final values of the weights have been reached. Hence, we obtain a very fast response time from the adaptive filter.

7.2.1 Simulation of the Adaptive Filter

A simulation illustrating the response time of the adaptive filter is shown in Fig. 7.5. this

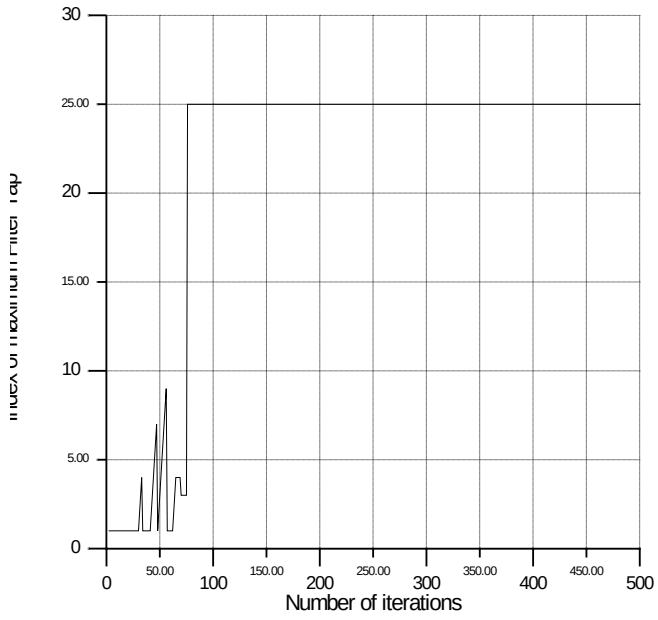

Figure 7.5: Time history of the numerically largest filter tap, estimating the delay in the flow channel (Flow rate)

figure shows the index of the numerically largest filter tap as a function of the number of iterations performed during adoption. The simulation is based on a simplified model of the flow channel, and is only intended to illustrate the operation of the system. We are not interested in having specific knowledge about the flow channel, it is the job of the adaptive filter to provide this information. The simulation is based on the parameters shown in Table 7.1. Before the simulation is started, all of the filter taps are reset to zero. Then the

Table 7.1: Parameters used for the simulation of the flow channel

Parameter	*Symbol*	*Value*
Sampling Period	T	$0.05s$
Distance between x and y	L	$1500\mu m$
Flow rate	ν	$1200\mu m/s$
Scaling	μ	0.001

adaptive filter starts the sampling of the temperature in the flow channel and updates the filter taps. In Fig. 7.3, the distance from the Heater to Sensor x is $1500mm$. With the flow rate shown in Table 7.1, this corresponds to a delay of $1.25s$ or 25 samples. This can be seen on Fig. 7.5 as the flat region from iteration 0 to 25. At this point, samples are filled into the tapped delay at Sensor x, which results in the fluctuations of the filter taps with the lowest index i.e. closest to Sensor x. This will continue for iteration 25 to 50. From iteration 50, Sensor y also gets samples from the flow channel. Hereafter the adaptive filter begins to

adjust its filter taps, in order to minimize the error e. From Fig. 7.5 we see that it only takes approximately 25 additional updates of the filter taps to get a stable maximum filter tap. We also see that the numerically largest filter tap has index 25, corresponding to a delay of $25 \cdot 0.05s = 1.25s$. The calculated delay is

$$\frac{1500\mu m}{1200\mu m/s} = 1.25s$$

which is the same as the estimated delay. The response time of the adaptive filter, defined as the time from iteration 0 to a stable estimate, is 75 samples, corresponding to 3.75s.

In Fig. 7.6 we have a simulation showing the filter taps after 4096 iterations. This figure shows the estimated impulse response of the flow channel between the measurement points at Sensor x and Sensor y. The fluctuations of the filter taps is a result of the coarse quantization

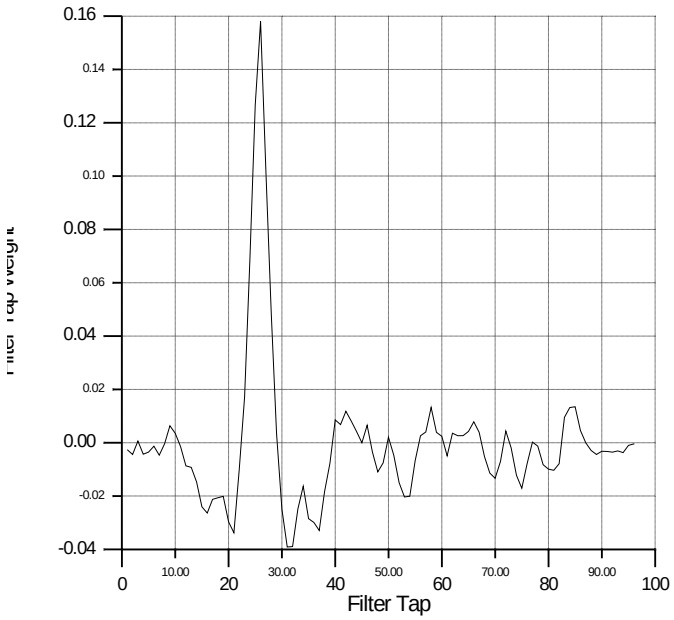

Figure 7.6: Filter taps after 4096 iterations of the adaptive filter

of the sampled temperature to a binary value.

Also, a simulation showing the evolution of the numerically largest filter tap (filter tap 25) as a function of the number of iterations is shown in Fig. 7.7. this figure shows that it takes more than 500 iterations for the filter tap to stabilize.

7.3 Implementation

As described in the previous section, the system consists of five basic building blocks: Sample and hold (S/H), differentiator, comparator, delay line and an adaptive filter which is built from SI integrators.

All of the circuits are based on switched current techniques using folded cascode Current Copiers, except for the delay line.

Based on noise analysis, we found that a storage capacitor of $1pF$ was sufficient. Also the following saturation voltages were found (assuming a supply voltage of 3.3 V): Memory transistors $\Delta v = 1V$, Bias transistors $\Delta v = 0.69V$ and Cascode transistor $\Delta v = 0.3V$.

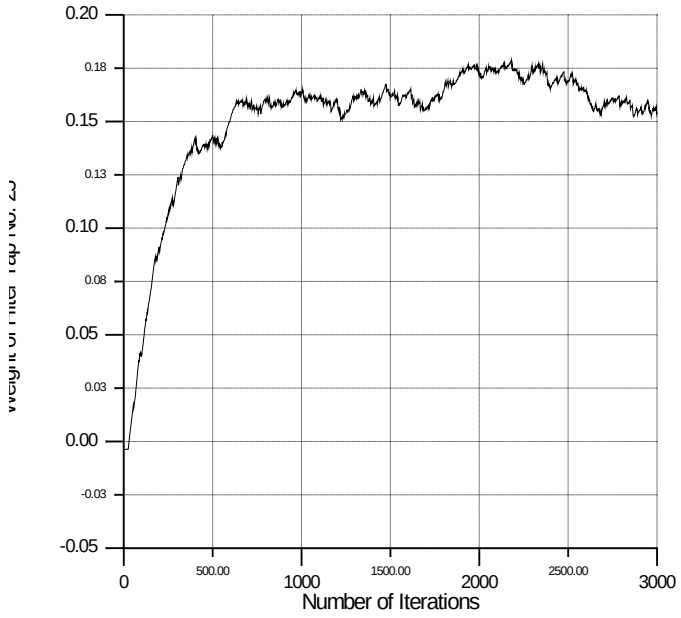

Figure 7.7: Filter tap No. 25 as a function of time

Because the input signal current to the filter taps (SI integrators) is scaled by μ, where $\mu \simeq 0.001$, the input current is very small. As a compromise between power consumption and the smallest signal current, we chose a bias current of $5\mu A$ in the memory transistors.

Based on the above parameters we get that the small signal bandwidth for the current copiers is given by:

$$\omega_0 = \frac{2 \cdot 5\mu A}{1V \cdot 1pF} = 10 \cdot 10^6 Rad/s \tag{7.5}$$

which corresponds to a small signal time constant of $\tau = 100nS$.

The memory transistors, shown as transistors with a switch connected at the gate, were implemented using PMOS transistors instead of NMOS transistors. This gives a more regular layout. All transistors that have their gate tied to V_{B1} and V_{B4} are bias transistors and transistors with their gate tied to V_{B2} and V_{B3} are cascode transistors.

7.3.1 Input Sections

The input signals at Sensor x (Upstream) and Sensor y (Downstream) shown in Fig. 7.4 are voltages. The internal signal processing on the chip relays on signals represented as currents, therefore an on-chip transconductor has been placed in front of each input section.

Sample/Hold and Differentiator

In Fig. 7.8 the circuit implementing the Sample/Hold and differentiator is shown. The first two current copiers CCOP1 and CCOP2 perform the Sample and Hold function. CCOP1 samples the signal at clock phase two and this sample is then fed into CCOP2 on clock phase one. The signal held in CCOP2 is fed to a differentiator, shown as DIFF. The differentiated output current is then fed to the comparator so that it can be quantized into a binary value.

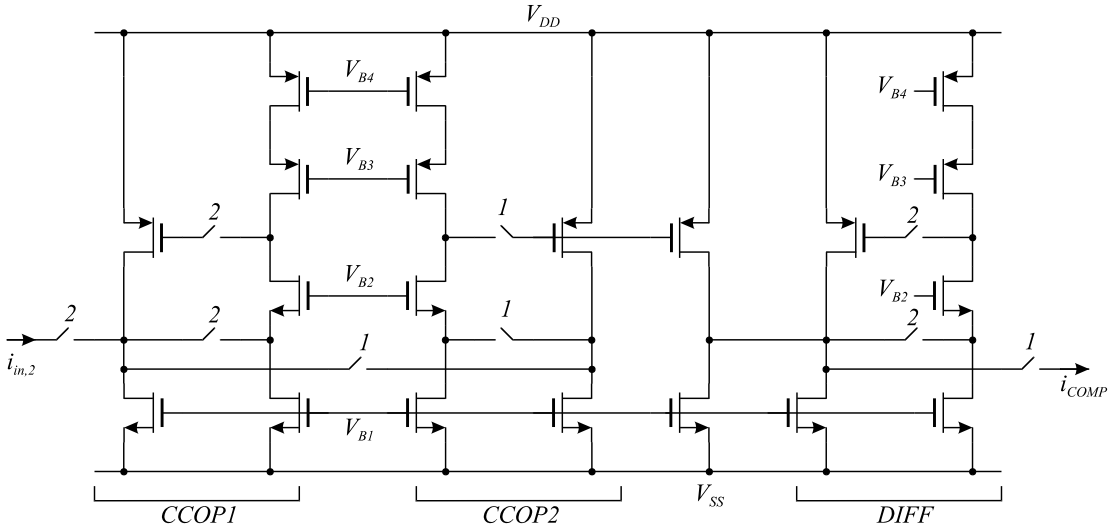

Figure 7.8: Sample/Hold and Differentiator

Comparator

The Comparator, shown in Fig. 7.9, is built around a differential pair. The input signal

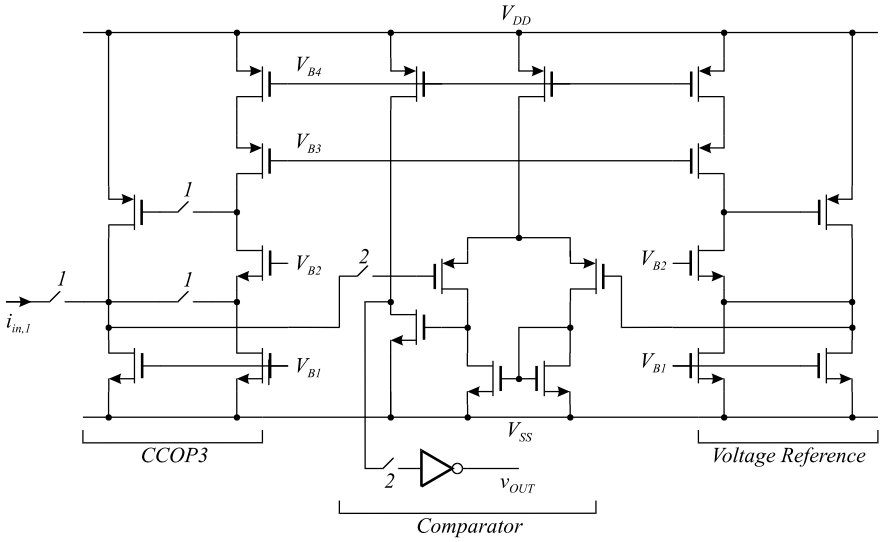

Figure 7.9: Comparator

is first clocked into the current copier CCOP3 at clock phase one. At clock phase two the current stored in the current copier CCOP3 is sent to the gate of one of the transistors in the differential pair. This is a very high impedance node, and this will cause the voltage to swing either to the positive or negative side, depending on the sign of the current stored in CCOP3. The differential pair compares this voltage with a reference voltage supplied by a voltage reference and delivers a binary output.

In the upstream section the comparator output is fed directly to the delay line and in the downstream section the comparator output is fed to a 1-bit DAC so that the quantized

signal can be converted to a current for further processing.

1-Bit DAC

The 1-bit DAC is used to convert the quantized downstream signal into a current. This current corresponds to $y(n)$ shown in Fig. 7.4 and $\widehat{y}(n)$ corresponds to the sum of the output currents from the integrators.

The error signal $e(n)$ is computed as the difference between the currents $y(n)$ and $\widehat{y}(n)$. The actual implementation of the DAC is shown in Fig. 7.10. This figure shows that the

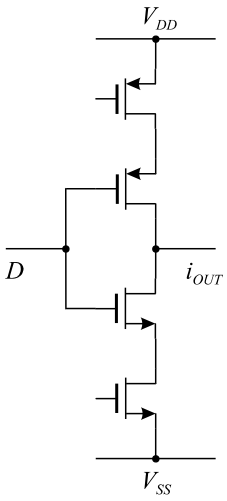

Figure 7.10: 1-Bit DAC

DAC is a compound inverter based on compound transistors (see Appendix A).

7.3.2 Filter Tap

Fig. 7.11 shows one of the filter taps. The filter tap consists of a programmable current divider implementing the scaling factor μ, and a switched current integrator. The output current of the integrator corresponds to the weight of the filter tap. The switches at the input and output of the integrator are the multipliers shown in Fig. 2. Because the temperature samples are digitized to a binary value, the multiplication's are replaced by switches. These switches are controlled by the digitized samples in the tapped delay.

For the LMS algorithm to converge, the scaling factor μ must be very small. Simulations have shown that μ should be smaller than 0.01, and preferably 0.001 to keep the variance of the filter taps reasonably low during adoption. Simulations also showed that the linearity of the scaling factor μ was of no importance for the adoption of the adaptive filter. In order to get such a small scaling factor it is not feasible to use current mirrors. Therefore we have used a simple resistive scaling based on MOS transistors operating in their linear region (Resistors). By controlling the gate voltages V_{C1} and V_{C2}, the scaling factor μ can be programmed. The voltage source V_B is used for canceling any DC-currents flowing to the integrator.

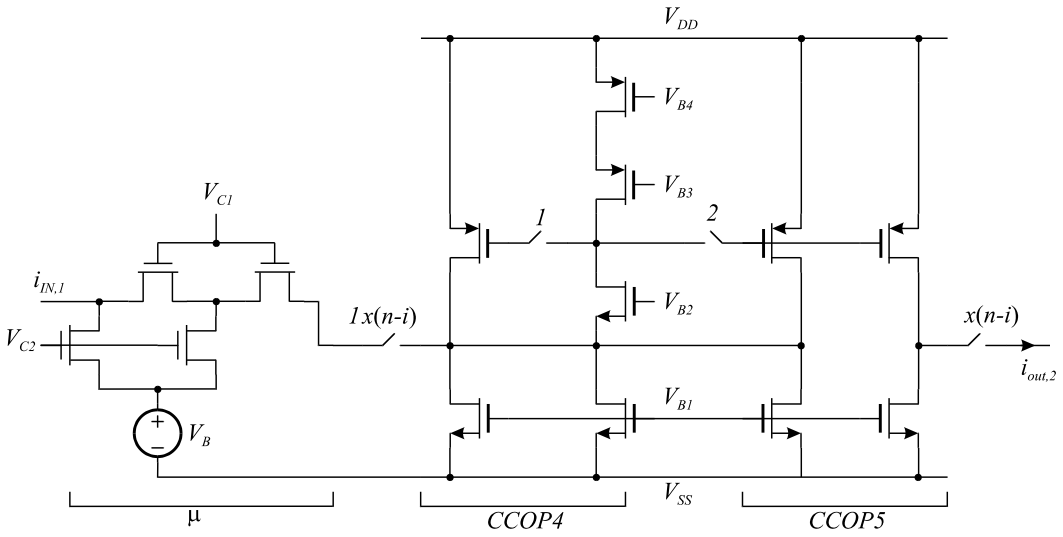

Figure 7.11: The ith filter tap

7.3.3 Delay line and FSM

The adaptive filter has two main modes of operation and is controlled by a Finite-State-Machine (FSM). In Fig. 7.12 we have shown how the FSM together with the delay line. The

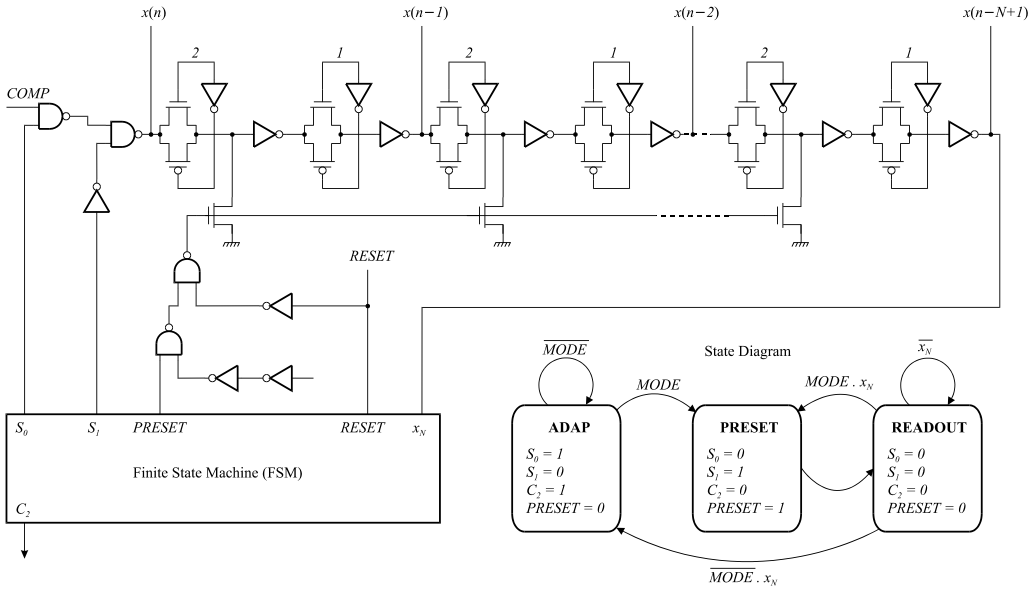

Figure 7.12: Delay line and FSM

two main operating modes of the adaptive filter are shown as **ADAP** mode and **READOUT** mode.

In **ADAP** mode the adaptive filter performs the actual adoption to the flow channel by estimating its impulse response.

In **READOUT** mode the adaptive filter periodically reads out all of its filter taps so the numerically largest filter tap can be found. This is done by clearing the delay line and then

rippling a logical high down the delay line. Each time a tap in the delay is logically high, it enables the output of the corresponding SI integrator which holds the actual filter tap.

The delay line is built from a cascade of CMOS inverters separated by CMOS switches. The CMOS switches operate on alternating clock phases, forming a two phase dynamic logic delay line.

7.4 Layout

The adaptive filter was implemented in an industry standard analog $2.4\mu m$ CMOS process (MIETEC). All of the Analog and the Digital building blocks were fully custom designed using the Layout editor L-Edit from Tanner Research.

A layout of the whole adaptive filter is shown in Fig. 7.13. This layout shows the main

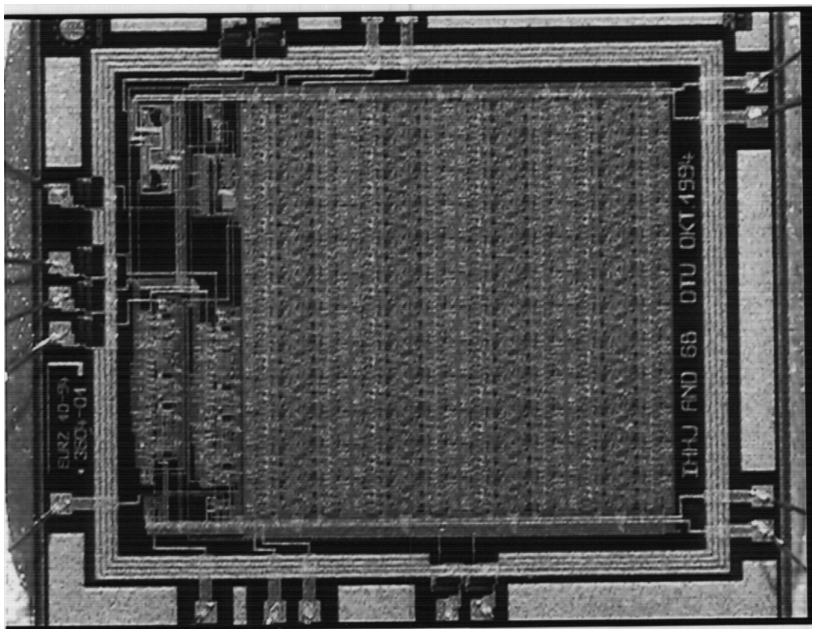

Figure 7.13: Layout of the Adaptive Filter

sections of the chip: Upper left corner the FSM and Bias circuitry, lower left corner the two input sections and to the right a large matrix of 96 SI integrators including the delay line.

The dimensions of the chip core are $3mm \times 4.5mm$ and the number of transistors is approximately 7500.

In Fig. 7.14 we have shown a detailed layout of one filter tap. At the top we have a single section of the delay line (z^{-1}) and immediately below, there is a clock and control bus. Below the clock and control bus there is a section of switches that control the input and output current to the integrator. At the bottom we have the integrator itself and right on top of it we have a signal bus where all of the currents entering and leaving the integrator are routed.

7.5 Experimental Results

Testing of the fabricated adaptive filter showed that it did not operate as expected.

Investigation of the design showed several design errors that had not been detected in the first design. Some of these errors were: The input transconductor in the input sections was

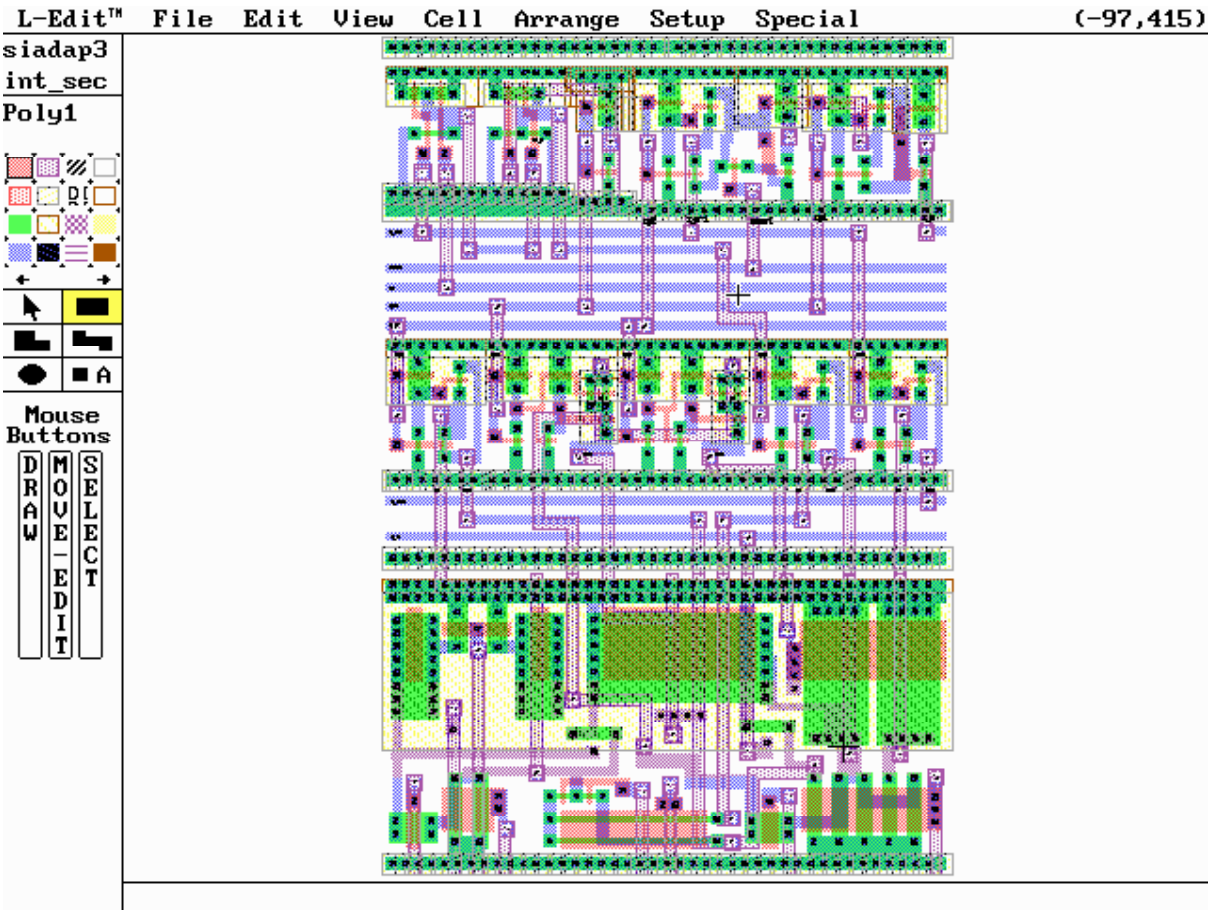

Figure 7.14: Layout of a single filter tap

not biased correctly which has the effect of limiting the input signal range. The programmable scaling factor μ collapsed for moderately large input currents. The current representing the error signal $e(n)$ (see Fig. 7.4) had larger variations than expected forcing it to overload some of the circuitry. The integrators did not have any reset capability making it difficult to predict the initial conditions when that chip was powered up.

The reason for these design errors, not beeing detected in the first place, is properly caused by the lack of a top-level simulation of the whole chip. We did not have sufficient computational resources at that time.

At the time being when this thesis is printed the adaptive filter chip is being redesigned for the above errors. A limited top level simulation using 24 filter taps has been performed. It took $1\frac{1}{2}$ week to simulate 150 clock periods on a 90MHz Pentium PC, and the simulation required approximately 32MB of RAM in order to run. This limited simulation indicated that the adaptive filter is capable of adapting.

The redesigned chip contains approximately 9500 transistors and the area of the chip core is approximately $3.8mm \times 3.8mm$. This chip will be shipped to EUROCHIP for fabrication very soon.

7.6 Conclusion and Future Work

In this chapter we have presented a adaptive filter intended for estimation of the flow rate in micromechanical flow channel. The estimate is based on the relationship between the shape of an impulse response and the delay caused by a filter with that impulse response, the filter being the flow channel seen between two measurement points. The accuracy of the flow estimator is not known at the moment.

An experimental chip with 96 filter taps has been designed in an industry standard $2.4\mu m$ CMOS process. The chip is designed to operate at a supply voltages down to $3.3V$ with a total current consumption of $2mA$ resulting in a power consumption of $6.6mW$.

We have shown that it is feasible to implement a multiplierless adaptive filter with 96 filter taps in switched current technique, with small power consumption and reasonable chip area.