Chapter 5

Interconnection Network of OTA-based FPAA

5.1 Introduction

Aside from CAB components, a number of different interconnect structures have been proposed for FPAAs. The choice of an interconnection architecture and its implementation will influence the routability of prototyped circuits and their performance. Analog circuits are far more sensitive than digital circuits to problems of fan-out, noise, and the presence of switches in the signal path [41].

5.2 Design Goals for Interconnect Network

a) Provide adequate flexible

The network must be capable of implementing the interconnection topology required by the programmed logic network with acceptable interconnect delays [43].

b) Use configuration memory efficiently

Space required for configuration memory can account for a reasonable fraction of the array real-estate. Configuration bits should be used efficiently.

c) Balance bisection bandwidth

Interconnection-wiring takes space and in some topologies, dominates the array size. The wiring topology should be chosen to balance interconnect bandwidth with array size so that neither strictly dominates the other.

d) Minimize delays

The delay through the routing network can easily be the dominate delay in a programmable technology. Care is required to minimize interconnect delays. Two significant factors of delay are,

i) Propagation and fan-out delay - Interconnect delay on a wire is proportional to distance and capacitive loading (fan-out). This makes interconnect runs roughly proportional to distance run, especially when there are regular taps into the signal run. Consequently, small/short signal runs are faster than long signal runs.

ii) Switched element delay - Each programmable switching element in a path (crossbar, multiplexor etc.) adds delay. This delay is generally much larger than the propagation or fan-out delay associated with covering the same physical distance. Consequently, one generally wants to minimize the number of switch elements in a path, even if this means using some longer signal runs. Switching can be used to reduce fan-out on a line by segmenting tracks, and large fan-out can be used to reduce switching by making a signal always available in several places. Minimizing the interconnect delay, therefore, always requires technology dependent tradeoffs between the amount of switching and the length of wire runs [43].

5.3 Different Types of Interconnection Network

5.3.1 Local Interconnection

At the base, of the interconnect hierarchy; there are local connections among array elements (Fig.5.1). These are generally worth keeping in any scheme because short, local interconnect can be fast and efficient (small fan-out and propagation delay) local interconnect resources are relatively cheap. Sub regions of logic networks are often heavily interconnected. How far local interconnect spans is, of course, highly dependent on the technology costs (*e.g.* how expensive is switching delay versus propagation delay? how much space do configuration bits require? how big are wires relative to fixed logic and memory?) and the detailed choice of a local interconnect scheme [15,45].

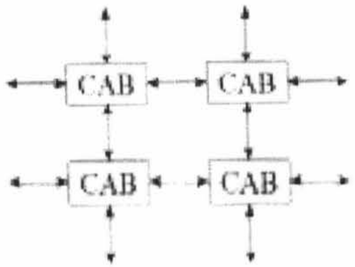

Figure 5.1 Local interconnection scheme

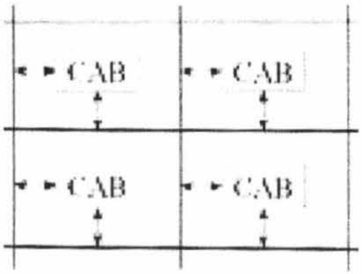

Figure 5.2 Global interconnection scheme

Quan et al. [51] proposed the use of local interconnects. In their architecture, each CAB can be connected to its eight neighbours and itself. This would seem to be a severe limitation on the flexibility of this FPAA; however, they focus on the large number of analog circuits with mostly local interconnections [51]. Pierzchala et al. tried an even more limiting architecture in which no electronic switches were included in the signal paths. While these designs may provide benefits in bandwidth and signal to noise ratio (SNR), they lack the flexibility and generality needed in a truly general purpose FPAA. Also one more FPAA having local interconnection network have been reported from Technical University of Gdansk, Poland, which utilized this structure very well by using a highly linear OTA as main computational element that eliminates the requirement of flexible interconnect structure [15].

5.3.2 Global Interconnection

In another design Fig. 5.2, an interconnect scheme with both local and global signal paths are introduced [15]. This configuration provided local routing paths for a cell's four neighbors (north, south, east and west), as well as connections to global busses that run horizontally, vertically, and diagonally. This two tiered hierarchy increases the routing flexibility within the FPAA [15].

5.3.3 Crossbar Interconnection

A crossbar interconnect network allows simultaneous connections from any input port to any output port. Fig. 5.3 shows a conceptual view and Fig. 5.4 shows the interconnection scheme of such a network. One possible implementation of the crossbar network is to assign a global bus to each input, and each of these busses can be connected to any outputs through a switch. Observe that such a network requires only one switching stage that is, every input and output pair is connected through a single switching element. This architecture provides full connection flexibility, but suffers from a large area overhead (wasting wire resources) and a high-energy consumption (due to the long global buses and the large number of switches)[52].

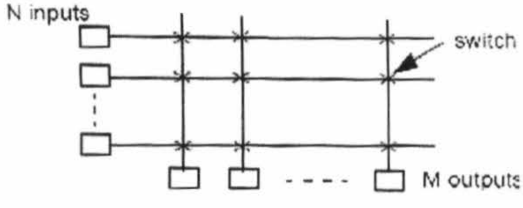

Figure 5.3 Conceptual crossbar network

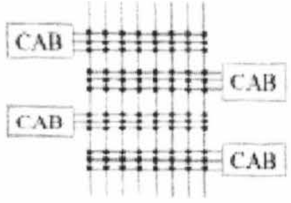

Figure 5.4 Cross-bar interconnection scheme

5.3.4 Fat-Tree Interconnect Network

The "fat tree" interconnect network, shown in Fig. 5.5 and Fig.5.6, is a network based on a complete (binary) tree. Logic blocks are located at the leaves of the network tree. The number of buses per channel increases as going up the tree [52].

Lee and Gulak [44] tried to solve this problem by using an area-universal fat tree network. They used a hierarchical fat tree network of small crossbar switches where the CABs were connected as the leaves of the tree. In an additional effort to minimize the size required by the switch networks, the number of connections was constrained (Lee and Gulak). Unfortunately, their prototype was too small to really test this interconnect concept.

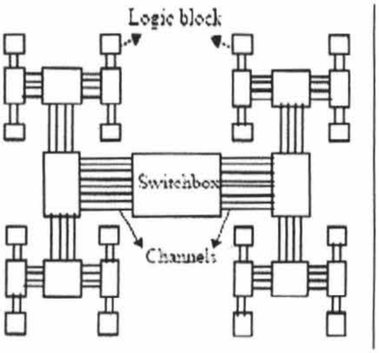

Figure 5.5 Conceptual fat-tree interconnect

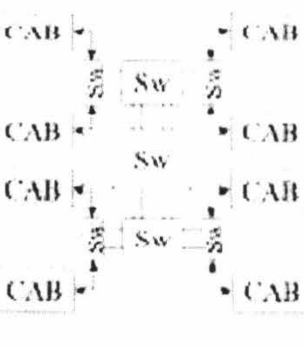

Figure 5.6 Fat-tree interconnection scheme

5.4 Interconnection Network of OTA-based FPAA

The structure of FPAA shown in Fig. 5.7 [14] consists of 40 CABs positioned in eight columns and five rows. Additionally three OTAs o1-o3 acts as signal buffers. Input signals are delivered through lines, i1 and i3. Because of this, up to three different filters can be realized simultaneously. The transconductance parameters of all the OTAs are controlled by external voltage and through digital switching of the output current mirrors. While voltage is common for all the amplifiers in the array, it is still possible to set the transconductance of every OTA separately by setting the gain of the OTAs current mirror. The structure of the FPAA is composed of universal configurable analog blocks (CABs). CAB shown in Fig. 5.8 consists of one programmable fully differential OTA, one programmable capacitor C_{EQ} and a set of switch S1–S12. The switches are placed in such a way that the OTA can be connected with or without the capacitor. It is also possible to pass the signals of other CABs through the switches situated at the top and the bottom of the CAB. Switches S1 and S2 enable to connect the input to the OTA directly or in an inverse way. The polarity of OTAs input signal could be straight or inverse, depending on the settings of switches S1 and S12, respectively [14].

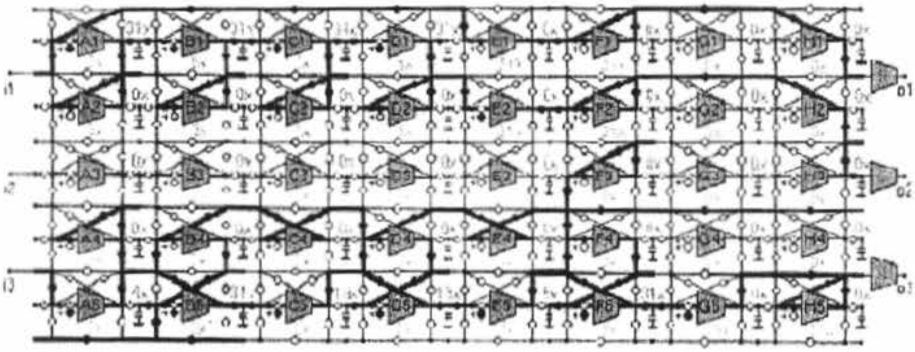

Figure 5.7 Structure of FPAA

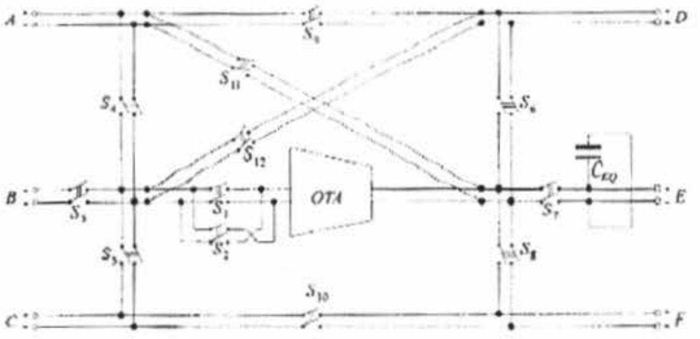

Figure 5.8 Structure of CAB

5.5 Proposed OTA-based FPAA Implementation

There are different methods for arranging FPAA architectures constituting CABs and their routing network are studied, mainly based on 'array' structure. Since area and efficiency is our concern a new routing architecture that reduces redundancy of the architecture was developed, i.e. reducing the number of unused CABs while implementing an analog processing function. So instead of 'array' architecture which is a common structure for FPAA implementation, a method of 'CAB clustering' in which FPAA is divided into a number of clusters and each cluster is having four fully differential programmable OTAs as CABs and differential switches was implemented, as shown in Fig. 5.9 and the versatile cluster consisting of four CABs are shown in Fig. 5.10. In comparison to the FPAA introduced by B. Pankiewicz et al [14], our FPAA uses less number of switches with the same functionality.

In each cluster shown in Fig. 5.9, CAB1 and CAB3 contain programmable fully differential OTA, while CAB2 and CAB4 are programmable fully differential OTAs with one programmable capacitor for each CAB. The switches are also differential switches and placed in such a way that the OTA can be connected with or without the capacitor. It is also possible to pass the signals of other CABs through the switches situated at the top and the bottom of the CAB. One input and one output terminal is available from each cluster.

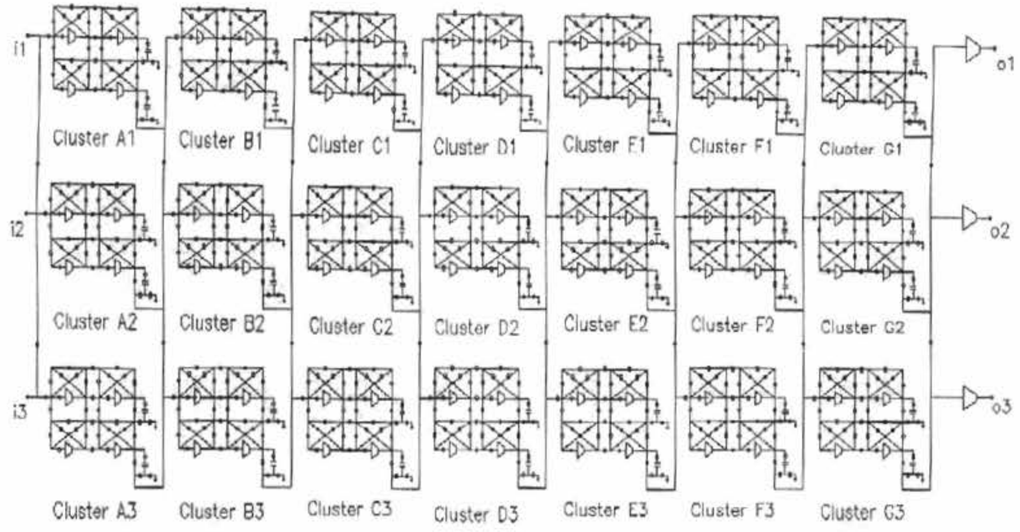

Figure 5.9 New architecture of CABs and routing network in the proposed FPAA

The Clusters and their arrangement are designed mainly for continuous time filter applications. Heterogeneous CABs used for realizing the different basic building blocks like grounded resistor, grounded inductors and floating inductors. The two PCA are used as grounded capacitor as well as floating capacitor.

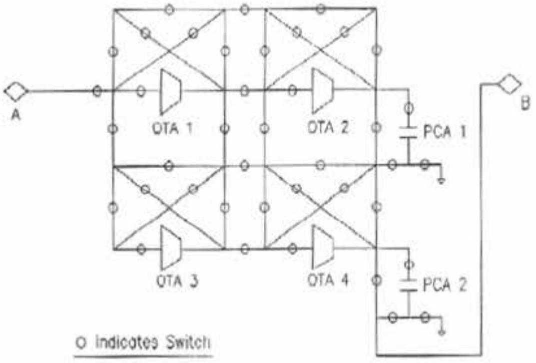

Figure 5.10 Structure of one cluster

5.6 Proposed FPAA using Clusters

The FPAA used for implementation of filters is presented in Fig. 5.9. It consists of 21 clusters and each cluster is having 4 CABs and 41 switches. Clusters are positioned in seven columns and three rows. Additionally three OTAs o1 and o3 – act as signal buffers. Input signals are delivered through lines i1,i2 and i3. Because of this, up to three different filters can be realized simultaneously. The transconductance parameters of all the OTAs are controlled by external voltage control voltage, called "V_{ctrl}", and through current mirrors. While voltage "V_{ctrl}", is common for all the amplifiers in the array, it is still possible to set the transconductance of every OTA separately by setting the gain of the OTAs current mirror.