Implementation aspects of ATM switches

Augustin, R.J.M.

Award date:
1993

EINDHOVEN UNIVERSITY OF TECHNOLOGY

FACULTY OF ELECTRICAL ENGINEERING

DIGITAL INFORMATION SYSTEMS GROUP

Implementation aspects of ATM switches

Graduation report by R.J.M. Augustin

Report of graduation work performed
from October 1992 to July 1993

Supervisor: Prof. ir. F. v.d. Dool
Coach: Ir. M. v. Weert

Abstract

Since in the future, the demand on broadband services (e.g. Lan and video interconnection services) is likely to increase, research is focusing on the introduction of the Broadband ISDN. In 1989 standards on the transfer mode for B-ISDN were introduced: ATM. From then a lot of ATM switch designs were introduced, but they all assume the VLSI technology to be more developed than it actually is. This report describes which modifications should be performed on the switch designs to make them better implementable. This includes a detailed examination of general ATM switch designs, an examination of Delta, Clos and Benes networks and an overview of actual introduced ATM switch designs.

Since for complex circuits maximum on-chip frequency is currently limited to about 50MHz (off-chip 25MHz) and ATM traffic has a speed of 155Mbit/s or more, parallellisation has to be performed to be able to reduce speed.
Bitparallellisation means a cell is divided in several bits wide packets. It has the advantages of having only a small speed adaptation buffer delay and of guaranteeing preservation of cell sequence integrity. Cellparallellisation means cells are divided over switch copies, which reduces the load per switch copy. This has the advantage of having more evenly spread traffic.

Since large switch designs can't be put on one chip and because they have to be modular, they have to be partitioned.
Slicing means the large switch design (with parallellisation) is cut in a way that one copy of a switch which results from the parallellisation, is kept in one piece, i.e. the switch is divided in planes. Slicing can only be performed after parallellisation is performed, and results in all equal parts.
Logical partitioning means the switch design is cut in a way that one part is a part of an ASE, one ASE or several ASE's. Logical partitioning is needed when slicing on its own is not sufficient and leads to several bits wide parts. This might be all equal parts (depending on the design).

How the partitioning should be performed depends on the switch design.
Small ASE's (size 2x2 or 4x4) don't have to be partitioned themselves. The MIN's these ASE's are used in, are based on Delta networks. Of these networks, the Batcher-Banyan network should be logical partitioned in a way that one part is one stage of ASE's. The other small ASE based networks should be partitioned in a way that one part is a stage of larger (than 2x2) networks.
Limited size ASE's are ASE's which can't grow beyond some size, because of limitations in technology. These ASE's should be sliced, to keep high (size dependable) speed links on-chip.
Unlimited size ASE's can grow without meeting limitations in technology. Because of their large complexity and their regular design, they should be logical partitioned.
Both limited size and unlimited size ASE's are used in folded and unfolded Clos and Benes networks. For Benes networks the unfolded network is preferred because it uses less chips. For the Clos networks, the folded network with all equal ASE's is preferred, because although it uses more chips than other networks, it only uses one kind of chip.

Table of contents

List of figures

List of tables

1 Introduction

In the past, the demands on the world-wide telecommunications network were ever increasing and this is not likely to change in the future: At this moment the narrowband Integrated Services Digital Network is introduced to the customers, but it's expected that in the future the demand on broadband services (e.g. video) will increase. That's why now research is focusing on the Broadband Integrated Services Digital Network. In 1989 the CCITT introduced standards on the transfer mode for B-ISDN: ATM. ATM is the abbreviation of Asynchronous Transfer Mode and it's main feature is that speed of the information in the network is independent of the speed of the information sources.

After ATM was introduced, a lot of switch designs have been presented. The problem of the large ATM switch designs is that they assume the current VLSI technology to be more developed than it actually is. This means, the designs first should be modified some way in order to avoid the limitations of current VLSI technology.
This report is the result of a graduation project in which these modifications on the basic switch designs were studied. It is divided in three parts:
First some general remarks are made on ATM and ATM switch designs (chapters 2 and 3). Next, the modifications that can be performed on a switch design, in order to get a better implementable switch, are described (chapters 4 and 5). The last part reviews how specific ATM switch designs (known from literature) can and should be modified in order to make them better implementable (chapter 6).

2 Broadband ISDN

The ideal telecommunications network can provide the user with all tele-services or combinations of them by one physical access. Because of the expected growing demand for broadband services in the future (e.g. video and LAN interconnection services), the ideal network has to provide a capacity as big as possible. Besides that, the network has to be flexible, so all services, including those who are developed in the future, can easily be implemented in the network. These demands require a narrowband ISDN follow-up: Broadband Integrated Services Digital Network (B-ISDN).

The B-ISDN has to provide the following kind of services:

- Conversational: Real time bidirectional information exchange between two users (e.g. telephone and data).
- Messaging: Not real time communication between individual users via storage units with store-and-forward, mailbox and/or message-handling functions.
- Retrieval: On demand retrieving public accessible information.
- Distribution: Broadcast services (e.g. tv and radio)

2.1 Asynchronous Transfer Mode

The whole of information transfer, including transmission, multiplexing and switching in a telecommunications network is called the transfer mode. A special transfer mode has been developed for the B-ISDN in which the speed of information is not related to the speed of the information sources. This transfer mode is called the Asynchronous Transfer Mode (ATM).

ATM is connection oriented, so a communication session consists of three phases:
- In the call set-up phase, the connection is established by allocating bandwidth and network resources.
- In the information transfer phase, the information is exchanged between the users.
- In the call termination phase the allocated network resources are released.

ATM is based on fixed length packets (cells), existing of a header and an information field. The information field consists of 48 octets of userdata. The header (5 octets) identifies the connection the cell belongs to. A connection (Figure 1) is a chain of virtual channels. These are logical links between two network nodes, which are defined when a connection is set up. The field in the header that identifies the virtual channel is called Virtual Channel Identifier (VCI, length 16 bits).
While setting up an ATM connection, virtual paths can also be used. These are semi-permanent defined groups of virtual channels. The Virtual Path Identifier (VPI, length 8/12

bits) indicates to which virtual path a cell belongs.

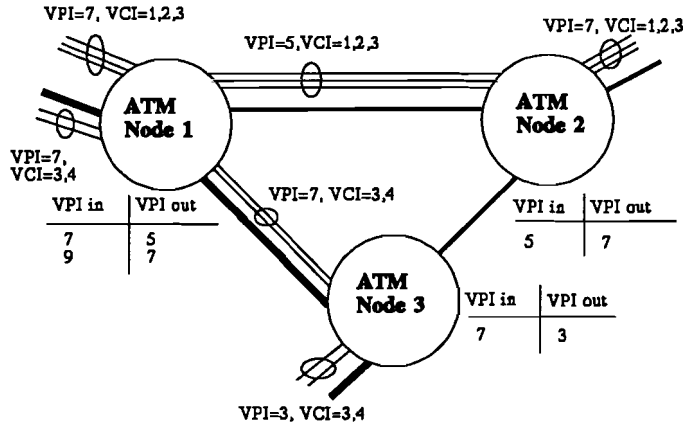

Figure 1: ATM connections

Because bit errors can cause a cell to enter the wrong connection, the header is protected by a Cyclic Redundancy Check (CRC, 8 bits) code. The header also contains fields which indicate how important a cell is with respect to cell loss and the kind of information the cell contains.

2.2 ATM Protocol Reference Model

It's hard to fit ATM in the OSI protocol reference model, that's why a new protocol reference model (Figure 2) is defined for ATM.

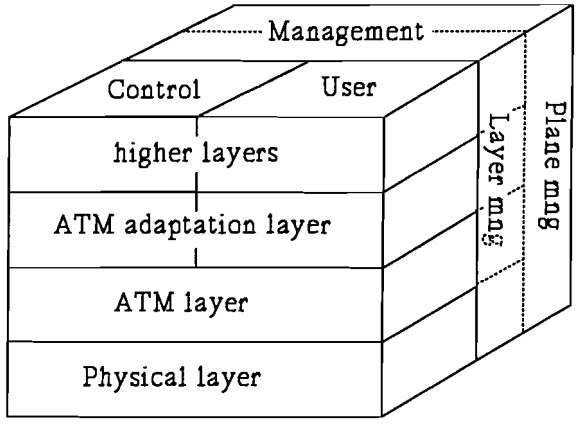

Figure 2: ATM Protocol Reference Model

There's a functional division in a management, a control and a user plane. Besides that there's a layered structure:
- The physical layer provides bit transmission, header error control and cell transmission.
- The ATM layer provides information transmission: The switching of cells, translation of VPI and VCI, cell (de)multiplexing and cell header generation/extraction.
- The ATM adaptation layer performs several interfacing functions for higher layers, but its main task is the dividing of a message into information fields of cells.

2.3 Bibliography

[BREE90] BREEDBAND ISDN
Leidschendam: PTT Research, 1990.

[DOOL92] Dool, prof. ir. F. van den
COMMUNICATIE CENTRALES EN NETWERKEN
Eindhoven: Technische Universiteit Eindhoven, 1992.

[PRYC91] Prycker, M. de
ASYNCHRONOUS TRANSFER MODE: SOLUTION FOR BROADBAND ISDN
New York: Ellis horwood, 1991.

[STAL92] Stallings, W.
ISDN AND BROADBAND ISDN, 2nd ed.
New york: Macmillan, 1992.

[VRIE92] Vries, R.J.F. de
SWITCH ARCHITECTURES FOR THE ASYNCHRONOUS TRANSFER MODE
Enschede: Technische Universiteit Twente, Doctoral dissertation.
Leidschendam: PTT Research, 1992.

3 Design considerations of an ATM switch

In the past, several switching systems have been developed for speech and data. They are all based on the Synchronous Transfer Mode and therefore unusable in an ATM environment. The two major issues on designing an ATM switching system are:
- The high speed of operation (155Mbit/s or more).
- The statistical behaviour of traffic in an ATM network.

A smaller but still significant influence have the small fixed cell-length and the limited header functionality. This chapter first discusses what an ATM exchange looks like and then an overview is given of the requirements of an ATM switching system. Finally the functions that are performed in an ASS and their impact on the design are discussed in detail.

3.1 General ATM exchange

The term exchange refers to all necessary hardware and software to implement an operational network node. Four functional parts are determined in an exchange (Figure 3):
- The input and output modules receive and transmit signals on incoming and outgoing links and perform synchronisation and light/electricity conversion.
- The exchange control and management part controls and manages the entire exchange. It takes care, for example, of administration and signalling.
- The ATM Switching Fabric (ASF) provides the actual switching of cells in the exchange and is often composed of smaller building blocks:

  ATM Switching Elements (ASE) are small blocks which can switch ATM cells themselves.

  ATM Switching Components (ASC) are all other building blocks within an ASF.

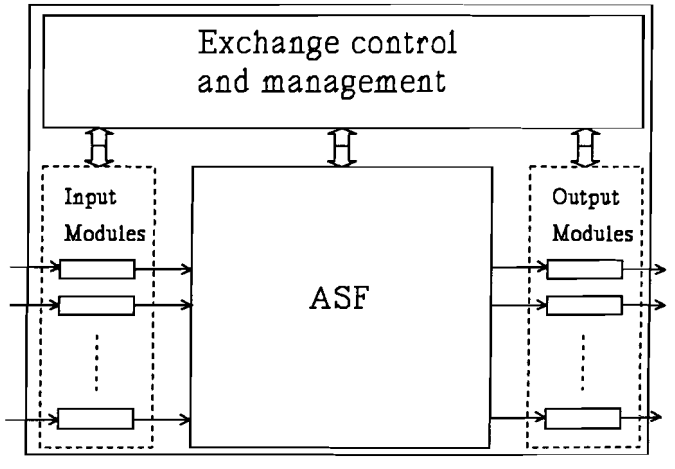

Figure 3: General ATM exchange

Besides the above described abbreviations, in this report, ATM Switching System (ASS) will be used for all blocks that can switch ATM cells (ASE and ASF).

3.2 ATM Switching System

In an ASS, cells are transported from an inlet to one or more outlets (Figure 4). This can be combined with multiplexing and demultiplexing of ATM traffic. Multiplexing refers to the concentration of traffic from a certain number of inlets to a smaller number of outlets. Demultiplexing is the opposite: Expansion of traffic from a certain number of inlets to a larger number of outlets.

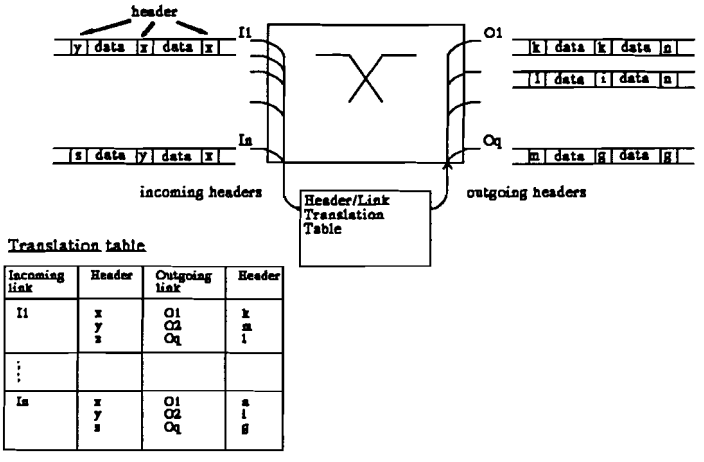

Figure 4: ATM Switching System principal

The transporting of cells from an inlet to an outlet is the actual switching: A cell arrives on a certain incoming logical channel en is transferred to one or more outgoing logical channels. A logical channel is determined by the combination of a physical port and a virtual channel or path (VCI/VPI).
This switching of cells in an ASS has certain aspects:
First of all, the VPI and VCI of the header of an arriving cell have to be translated, so they match the outgoing channel (Header translation). Next, the cell has to find the correct path to the outlet of the ASS (routing). And last, when two cells arrive simultaneously destined for the same output, one of them has to wait (buffering).

3.3 Requirements of an ASS

An ASS has to meet several requirements in order to operate properly in an ATM network and in order to make the ASS implementable. These requirements are divided in three groups: Functional, performance and general requirements.

3.3.1 Functional requirements

The functional requirements guarantee that the ASS can operate properly in an ATM environment.

- The ASS should switch 155Mbit/s traffic.
  The traffic an ASS has to switch originates from several sources. To switch speech originating traffic a speed of 64kbit/s is demanded, while video traffic of HDTV quality demands the highest speed (155Mbit/s).

- The ASS should perform space and time transposition on a cell.
  The routing and queueing necessary for this will be discussed in paragraphs 3.4, 3.6 and 3.7.

- The ASS should perform a header translation.
  This has been discussed in paragraph 3.2.

- The ASS, besides switching unicast traffic, should switch broadcast and multicast traffic.
  In an ASS, cells have to be switched from one inlet to any set of (also one and all) outlets. Unicast traffic means traffic from one inlet to one outlet (e.g. telephone). Broadcast traffic means traffic from one inlet to all outlets (e.g. television) and multicast traffic is traffic from one inlet to any set (except one or all) of outlets (e.g. tele-conferencing).

- The ASS should deal with priorities.
  In a header there's a Cell Loss Priority bit, which indicates how important a cell is with respect to cell loss. In case of congestion, cells with lower priority should be discarded first.

- The ASS should guarantee preservation of cell sequence integrity.

3.3.2 Performance requirements

The performance requirements guarantee maximum capacity for the ASS. Because B-ISDN should switch all services including those who are developed in the future, the performance requirements are very high.

- The throughput should be as large as possible.
  The throughput is the total switch capacity of the ASS in bits/s. This factor depends on the arriving traffic, so the ratio of arriving traffic and the amount of switched traffic can better be reviewed, i.e. the relative throughput.

- The cell delay and jitter should be minimal.
  In B-ISDN time transparency should be guaranteed for several services (speech for instance). That's why cells should have a small delay (max. several thousands of microseconds). The variation in cell delay (jitter) should be limited to several hundreds of microseconds.

- The cell loss probability and cell insertion probability should be minimal.
  There are several sources of cell loss. The first and most important is limited bufferspace. This means, because of increasing load, queues have grown up to a length at which the buffers don't have the capacity to store any additional arriving cells any more.
  It's also possible the design implicates extra cell loss, because of limited capacity of resources (e.g. providing only one path for every inlet-outlet combination). This is done in order to reduce the hardware complexity, which results in lower costs.
  The third factor are hardware errors. When a bit error occurs, the cell might be routed the wrong way, causing the cell to enter the wrong connection (cell insertion). The header is protected against these hardware errors by means of a Header Error Control code, but for instance hardware errors can also mutilate routing tags. So it might be necessary to introduce extra protection.
  Typical values for cell loss probability are between 10^{-8} and 10^{-11}. The cell insertion probability should be a thousand times smaller.

- The ASS should be fair.
  For every inlet, the same performance should be guaranteed.

- The ASS performance should not be too sensitive to the switched traffic.
  The ASS should not be too sensitive to the type of traffic: So it shouldn't matter whether the traffic has a high or low level of burstiness.
  The ASS should not be too sensitive to the load distribution: So it shouldn't matter whether one inlet has a high load while another inlet has a low load, or all inlets have equal load.
  The ASS should not be too sensitive to traffic mix: So it shouldn't matter what kinds of traffic are offered simultaneously.

3.3.3 General requirements

The general requirements guarantee the proper operation of any telecommunications system.

- Design and hardware complexity should be minimal.
  The design complexity should be small, so the system can meet the other requirements more easily. The hardware complexity, for instance the amount of bufferspace or the amount of silicon needed, should be small in order to have minimal costs.

- The ASS should be easily expandable and should be modular.
  A good telecommunications system is easily expandable, so when performance requirements are changed, the system can be adapted instead of being replaced. Modularity is needed so when adaption isn't possible, only a part instead of the whole system has to be replaced.

- The ASS should have a high cost efficiency.
  The cost efficiency is the ratio of overall performance and hardware complexity.

- The ASS should be fault tolerant, reliable, repairable, testable and maintainable.

- The ASS should be suitable for VLSI implementation.
  CMOS circuits use less space, have lower costs, but also have lower maximum possible speed than ECL (Bipolar) circuits. so far BiCMOS seems to be a good compromise for most designs.

3.4 Routing

In an ASS cells are transferred from a certain inlet to a certain outlet. This routing information can be contained in two media:

The first medium is a routing tag which is attached to the arriving cell. This tag is determined by a table at the entrance of the ASS and carries information for all routing decisions in the ASS. The use of a routing tag has the advantage that the switch is self-routing, i.e. inside the switch no extra routing control is needed. The implementation however can cause a synchronisation problem, because bits have to be inserted in a high speed traffic stream.
The other medium is a table. Every time a routing decision is made, the tables are searched. This kind of routing is more flexible and multicast and broadcast functions can be implemented more easily, but the tables need a lot of memory space and these memories need a very small access time.

A second aspect of routing is the moment the route of a cell is determined.

The first way is to determine the route for the duration of a connection: All cells of the same connection follow the same route through the ASS (connection oriented). This has the advantage that there's preservation of cell sequence. There's however the disadvantage, that traffic isn't distributed evenly over the switch.
The alternative is to determine the route every time a cell arrives (connectionless). This way the traffic is spread more evenly. Disadvantage of this method is that cells of the same connection follow different paths through the switch, so they get different delays. This means cell sequence must be recovered at the outlets, so extra bufferspace is needed.
The above results in four types of routing (Table 1).

Table 1: Types of routing

Routing info medium / Routing moment	Tag	Tables
Connection-oriented	I	III
Connectionless	II	IV

3.5 Transfer media

Every ATM switch should have a bufferspace and should transfer cells to the correct outlet. The transferring of cells from an inlet to the buffer and/or from the buffer to the outlet is done by the switching transfer medium. The switches in literature use three kinds of transfer media.

The first transfer medium is the matrix of slotted buses. This means for every inlet there's a bus to all output or crosspoint modules. These buses have the advantage of operating at the same speed as an inlet, but switches using this medium have a large hardware complexity.

The second transfer medium is the TDM-bus. This is a single bus from all inlets to all output modules or from all inlets to a common or shared bufferspace or from a common or shared bufferspace to all outlets. It has the disadvantage of operating at a speed of N times the speed of an inlet/outlet (with N being the number of inlets/outlets). The advantage of switches using the TDM-bus is they need less hardware than switches based on a matrix of slotted buses.

The two described transfer media are always part of an ASE which is used in a Multistage Interconnection Network (paragraph 3.8), because otherwise speed or hardware complexity would be too large. For switches which use no internal buffering or use distributed queueing, the ASE is very small and simple. In these kind of switches the MIN (based on Delta networks) itself is the transfer medium. So the third and final transfer medium is the 'on the Delta network based' MIN.

3.6 Buffering

A major aspect of an ASS is the buffering of cells when more than one cell arrive simultaneously, destined for the same outlet. Besides 'buffering', literature often uses 'queueing'. In this report 'buffering' is used when referring to the storage of cells, while queueing is used when referring to queueing theory: i.e. the way cells are served.

The storage of cells can be done in three different ways: Exclusive buffering, common buffering and shared buffering.
Exclusive buffering means every queue has its own equally sized physical memory space. The advantage of exclusive buffering is the simple control needed. The disadvantage of exclusive buffering is the large amount of total memory space needed.

Common buffering means every queue has its own equally sized memory space within one large memory space: i.e. the memory space is partitioned in a way that the maximum length of all queues are equal. The control needed for this kind of buffering is more complex than the exclusive buffering control and there's still a large amount of total memory space needed.

Shared buffering means all queues are within a large memory space. The maximum size of a queue depends on the space left by the other queues. This kind of buffering needs less memory space, but the control is very complex. Another disadvantage is the memory space might be dominated by a single queue handling burst traffic.

The above described buffering methods can also be combined in one switch design.
This paragraph now will end with some remarks on memory access time of the bufferspace, because this is also affected by the chosen buffering type.
When using a single ported memory (i.e. one write or one read operation per cycle) shared and common buffering need N write and N read operations per cell time, while exclusive buffering only needs one read and one write operation (crosspoint queueing) or one read and N write operations (output queueing) per cell time.
The memory access speed is reduced by using a dual ported memory (i.e. one read and one write operation per cycle). This is especially an advantage when used with shared or common buffering.

3.7 Queueing

The types of queueing discussed are related to the position in the ASS where the actual queueing takes place: Input, output, crosspoint and distributed queueing are distinguished:

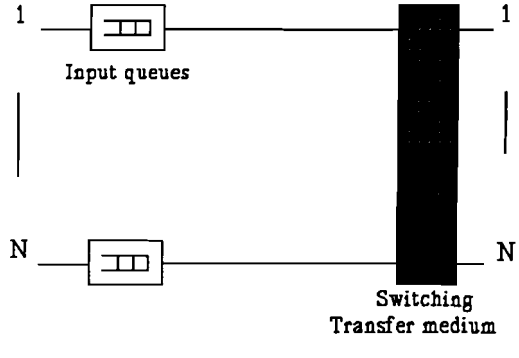

Figure 5: Input queueing

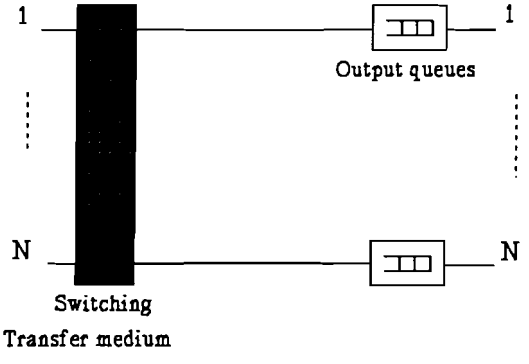

Figure 6: Output queueing

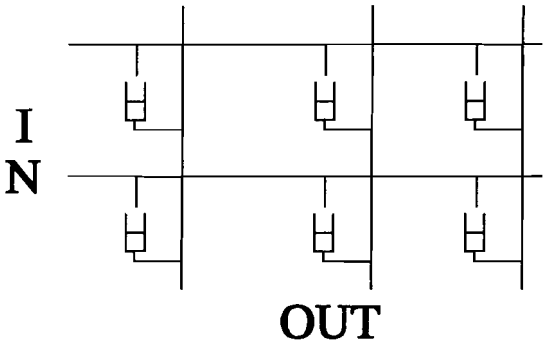

Figure 7: Crosspoint queueing

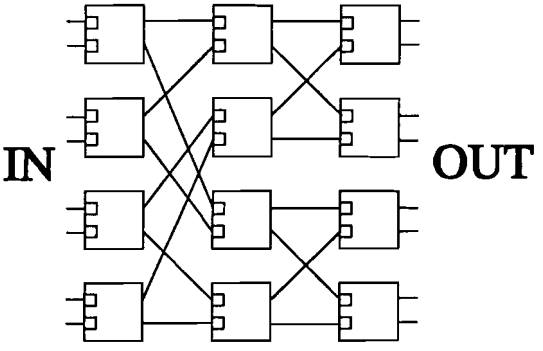

Figure 8: Distributed queueing

Input queueing (Figure 5) means there are FIFO queues on the inlet of the ASS. When two cells arrive simultaneously destined for the same outlet, one of them is actually switched to the outlet and the other is buffered at the inlet. The arbitration logic which decides which inlet to serve can range from simple round-robin to complex, e.g. taking into account the input buffer filling levels. In the transfer medium only one cell arrives per outlet, so the medium can operate at the same speed as the inlets.
A major disadvantage of input queueing is, that when a cell is queued, it blocks all other cells arriving at the same inlet, even when they're destined for unoccupied outlets. This effect, called Head Of Line blocking, degradates the relative throughput to 0.586.
This throughput degradation can be avoided in two ways:
The first way is to also examine the other cells in the queues, when HOL-blocking occurs. This means the FIFO service discipline is replaced by 'Bypass queueing'.
The second way is to have several queues instead of one, per inlet. These queues are filled equally. When HOL-blocking occurs in a queue, the other queues of the same inlet are examined. In this way the FIFO service discipline is preserved per queue, but now cell sequence might be lost, because cells of the same connection (same inlet) get in different queues and so they might get different delays.
The disadvantage of both solutions is the more complex queue control needed while the relative throughput is still limited to 0.63.

Output queueing (Figure 6) means the queues are located at the outlets of the ASS. Now, a maximum of N cells destined for one outlet can arrive (N is the number of inlets) in the transfer medium, so the transfer medium has to operate at a speed of N times the speed of the inlets or has to provide N parallel paths from every inlet to every outlet. Also the bufferspace has to be able to store N cells in a queue during one celltime.
To keep the ASS fair special arbitration logic is needed:
When several cells arrive simultaneously destined for the same outlet, they have to be placed in the queue of the correct outlet. The cell which is put first in the queue gets the smallest delay and the cell which is put last, the largest delay. To get a fair ASS, the arbitration logic which decides in which order to put the simultaneous arriving cells in the queue, should every time select a different inlet, whose cell is put first in the queue. This can be done for instance in a round-robin way.

In case of crosspoint queueing (Figure 7) there's a queue for every inlet-outlet combination. To place a cell in the queue, a broadcast medium is needed. Next, at every crosspoint is checked whether the cell is destined for the outlet to which this queue belongs.
Because of the broadcast medium, the broadcast and multicast facilities are easy to add. The disadvantage is the large hardware complexity of crosspoint queueing based ATM switches. Regarding performance, crosspoint queueing acts the same as output queueing, but the bufferspace only needs to store one cell in a queue per celltime instead of N.

Distributed queueing (Figure 8) means every ASE in the ASF owns a small queue of one or two cells. When a queue is full, this is indicated to the preceding ASE's by a 'stop sending' signal. This mechanism is called Back-Pressure. Cell loss can now only occur at the inlets of the ASF. This kind of queueing has the advantage that only a little, relative slow bufferspace is needed.

Combining the discussed types of queueing within one switch design is also possible.

3.8 Multistage Interconnection Networks

The most important element of an ATM exchange is the ASF. This is an ASS with a lot of inlets and outlets. Inside, the ASF is (often) constructed of multiple ASE's who are interconnected in a certain way. These networks of ASE's are called Multistage Interconnection Networks (MIN's). MIN's are divided in one-path and multi-path networks.

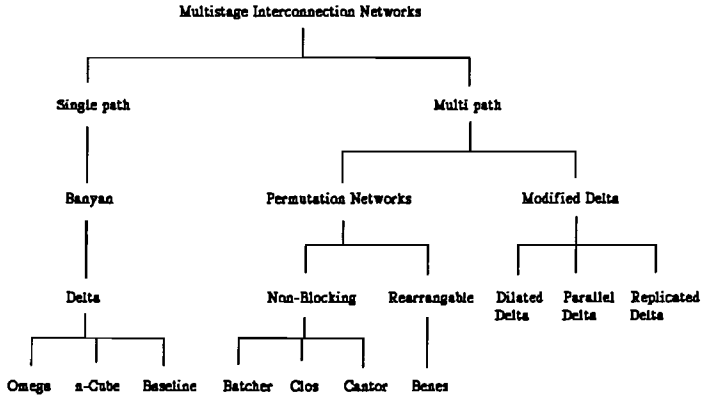

Figure 9: Multistage Interconnection Network classes

The one-path networks only have one path between every inlet and outlet. So every time a routing decision is taken there's only one possibility, which is why these networks are called 'self-routing' and no complex routing control is needed. On the other hand, cell contention will occur more often, because paths of different inlet-outlet combinations share internal links. The multi-path networks have more paths per inlet-outlet combination, causing cell contention probability to become very small, but there's a lot more hardware needed.

3.8.1 One-path networks

De general subclass of one-path networks (Figure 9) are the Banyan networks. They are defined as networks with one unique path for every inlet-outlet combination. The Delta networks are those Banyan networks in which small ASE's are placed in stages. The inlets of an ASE are only connected to the outlets of the ASE's in the previous stage and the outlets of an ASE are only connected to the inlets of the ASE's in the next stage.
There are three ways to connect the ASE's in a Delta network, resulting in the Omega network, the indirect binary n-Cube network and the Baseline network. All these networks have the same delay and throughput performance.

3.8.2 Multi-path networks

The low throughput performance of the one-path networks is a major disadvantage. The multi-path networks have a better throughput performance, but they need extra hardware and on top of that, except for Batcher-Banyan based MIN's, they're not self-routing. The multi-path networks are divided in permutation and modified Delta networks (Figure 9).

The permutation networks can handle all N! sets of inlet-outlet combinations. This means a permutation network can always connect a free inlet with a free outlet, if there are no other inlets who want a connection with the same outlet.
The rearrangable permutation networks can connect any free inlet with a free outlet, after the existing connections have been rearranged. A good example of these networks is the Benes network.
The nonblocking permutation networks can always connect a free inlet with a free outlet (without rearranging existing connections). The Clos network is a good example of these networks. Another good example is the Batcher network, which is also self-routing.

The modified Delta networks are multi-path networks, constructed of Delta networks with certain modifications. These modifications can be:
- Multiplying every single link between two ASE's.
- Placing multiple Delta networks as a layered structure.
- Adding extra stages to the Delta network.

The resulting networks are the dilated Delta network (multiple links), the replicated Delta network (multiple layers) and the parallel Delta network (extra stages and multiple layers).

3.9 Summary and conclusions

This chapter discusses the design of an ASS. It is shown that an ASS should transfer cells from an inlet to one or more outlets (routing). The VCI and/or VPI in the header of these transferred cells should be adapted (header translation) and one cell should wait, if two simultaneous arriving cells are destined for the same outlet (buffering).

Next the requirements an ASS has to meet, are given. These requirements are divided in functional requirements, which guarantee proper operation of the ASS in an ATM environment, performance requirements, which guarantee maximum capacity of the ASS, and general requirements, which guarantee proper operation of any telecommunications system. The last paragraphs discuss the aspects of an ASS design in more detail.
Routing is divided in four classes depending on the moment the routing takes place (connection oriented and connectionless) and the medium which carries the routing information (tag or table). The transfer media of an ASS are identified: The TDM-bus, the matrix of slotted buses and the small ASE based MIN. Storage of cells is divided in three types of physical storage (exclusive, common and shared buffering) and four types of logical storage (input, output, crosspoint and distributed queueing).
The chapter ends with an overview of Multistage Interconnection networks, which are divided in one-path networks (Delta networks) and multi-path networks (Batcher, Clos, Benes and modified Delta networks.

3.10 Bibliography

[BANN91] Banniza, T.R. en G.J. Eilenberger, B. Pauwels, Y. Therasse
DESIGN AND TECHNOLOGY ASPECTS OF VLSI'S FOR ATM SWITCHES
IEEE Journal on selected areas in communications
Vol 9, No 8 (oct 1991), p. 1255-1264.

[CHEN91] Chen, X.
A SURVEY OF MULTISTAGE INTERCONNECTION NETWORKS IN FAST PACKET SWITCHES
International journal of digital and analog communication systems
Vol 4, No 1 (jan-mar 1991), p. 33-59.

[DIRK90] Dirksen, M.J.G.
INTERCONNECTION ARCHITECTURES IN ATM: THE USE OF BATCHER BANYAN NETWORKS AS ATM SWITCHES
Eindhoven: Technische Universiteit Eindhoven, 1990
Graduationreport nr. 5697.

[DOOL92] Dool, prof. ir. F. van den
COMMUNICATIE CENTRALES EN NETWERKEN
Eindhoven: Technische Universiteit Eindhoven, 1992
Syllabus (from 5P460)

[JANS92] Jansen, J.W.H. en L.A.J. van den Heuvel
ONTWERP VAN EEN ATM PACKETSWITCH M.B.V. IDASS
Eindhoven: Technische Universiteit Eindhoven, 1992
Traineeship report nr. EB 388

[LIST89] Listanti, M en A. Roveri
SWITCHING STRUCTURES FOR ATM
Computer communications
Vol 12, No 6 (dec 1989), p. 349-358.

[NEWM92] Newman, P.
ATM TECHNOLOGY FOR CORPORATE NETWORKS
IEEE Communications magazine
Vol 30, No 4 (Apr 1992), p. 90-101.

[OIE91] Oie, Y. en T. Suda, M. Murata, H. Miyahara
SURVEY OF SWITCHING TECHNIQUES IN HIGH SPEED NETWORKS AND THEIR PERFORMANCE
International journal of satellite communications
Vol 9, No 5 (sep-oct 1991), p. 285-303.

[PRYC91] Prycker, M. de
ASYNCHRONOUS TRANSFER MODE: SOLUTION FOR BROADBAND ISDN
New York: Ellis horwood, 1991.

[RATH89] Rathgeb, E.P. en T.H. Theimer
ATM SWITCHES - BASIC ARCHITECTURES AND THEIR PERFORMANCE
International journal of digital and analog cabled systems
Vol 2, No 4 (oct-dec 1989), p. 227-336.

[TAKE91] Takeuchi, T. en H. Suzuki, T. Aramaki
SWITCH ARCHITECTURES AND TECHNOLOGIES FOR ASYNCHRONOUS TRANSFER MODE
IEICE Transactions
Vol E 74, No 4 (Apr 1991), p. 752-760.

[VRIE92] Vries, R.J.F. de
SWITCH ARCHITECTURES FOR THE ASYNCHRONOUS TRANSFER MODE
Enschede: Technische Universiteit Twente, Doctoral dissertation.
Leidschendam: PTT Research, 1992.

[WULL89] Wulleman, R. and T. van Landegem
COMPARISON OF ATM SWITCHING ARCHITECTURES
International Journal of digital and analog cabled systems
Vol 2, No 4 (oct-dec 1989), p. 211-225.

4 General aspects of implementation

After an ATM switch is designed, it is implemented. During this several problems, resulting from limitations in current semiconductor technology, will occur. The methods to overcome these problems and their impact on switch designs are now discussed. At the end of the chapter a switch classification based on these implementation aspects is presented.

4.1 Parallellisation

Because of limitations in current VLSI technology the maximum operation speed possible on-chip is 50MHz, while ATM traffic has a speed of 155Mbit/s or more, the cells should be processed parallel inside the switch. The input and output ports however have to operate at a speed of 155Mbit/s in order to receive and transmit ATM cells. These ports are not discussed any further, but the assumption is made it's possible to implement them (e.g. with BiCMOS technology).
A parallellisation of factor x (with $x\varepsilon\{2,3,...\}$) means, the switch is duplicated x times. So the hardware complexity grows x times, but speed can be reduced x times, because load per switch copy is reduced x times. An ATM switch with parallellisation x is visualised in Figure 10 by adding a third dimension.

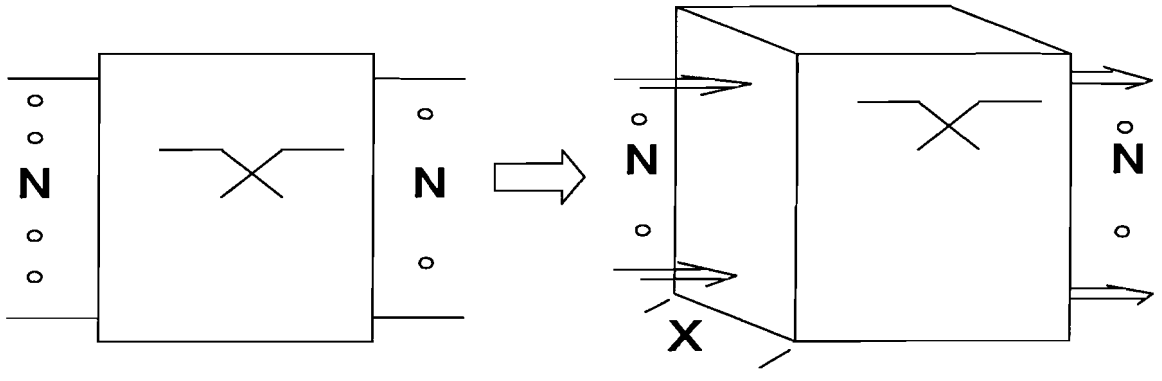

Figure 10: An ATM switch with parallellisation x

4.1.1 Description of types of parallellisation

Parallellisation of ATM traffic can be done in two ways. The first method is bit-parallellisation. This means a cell is converted from serial to parallel:
The maximum parallellisation possible this way is 424, i.e. a cell is converted to 424 parallel bits. This method of parallellisation has the advantage that for some values of parallellisation it's easy to add a routing tag to the cell, avoiding the problem of inserting some bits in a high speed traffic stream:
When a parallellisation of 16 is done, a cell is converted to 27 packets of 16 bits (A cell is 26.5 * 16 bits), leaving 8 bits unused, which can be used for routing tag. Because standard-

memory locations are several bits wide, it's an advantage to use bit-parallellisation. Another advantage is that a large bitparallellisation factor means that a cell only exists of a few packets. This means that to store a cell only a few memory accesses during one celltime have to be made, so the demanded memory access speed decreases. Finally, the cell sequence isn't affected by bitparallellisation.

The second method of parallellisation is cellparallellisation. This means, cells are still processed serial but they are distributed over the switch copies. This distribution is done in such way that all switch copies have an equal load. Actually, a parallellisation of x means x times more paths are present in the switch than in the original design. This means the load per switch copy is x times smaller, so the speed can be reduced. This method has the disadvantage that cell sequence integrity might be lost and it's still hard to insert routing tags, because this still requires adding some bits to a high speed traffic stream.

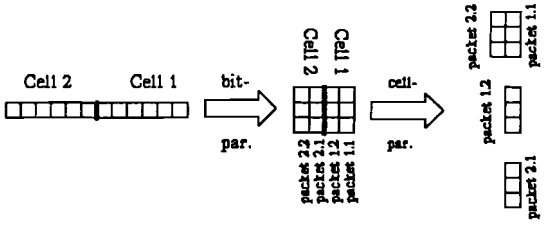

Figure 11: Cellparallellisation after bitparallellisation

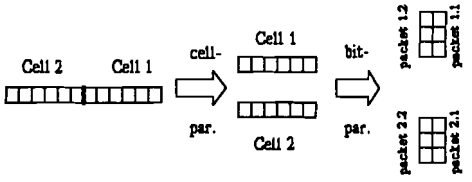

Figure 12: Bitparallellisation after cellparallellisation

A combination of these two methods is reached in two ways (Figure 11 and Figure 12):
The first method is to perform bitparallellisation first and then perform cellparallellisation. This means first a cell is divided in x bits wide packets, next these packets are distributed over the switch copies resulting from the cellparallellisation. This asks for very complex control because every single x bits wide packet needs its own routing information. Therefore this method of combination is not useful.
The second method of combination is to perform cellparallellisation first and then perform bitparallellisation. This means cells are distributed over the switch copies resulting from cellparallellisation and then each cell is divided in x bits wide packets. So all packets of one cell are switched by the same switch copy resulting from cellparallellisation.

4.1.2 Calculations of buffering delays for parallellisation types

In the following a cell will be looked upon, at discrete times 'at the middle of bits' (Figure 13). Distance between bits will refer to distance between 'the middle of bits' and bittime will refer to distance between bits at the original fast speed (e.g. 6.45 nanoseconds when the original speed is 155Mbit/s).

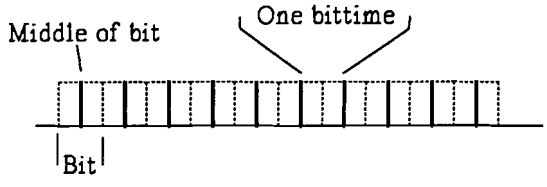

Figure 13: Used view on a bitstream

When using parallellisation somewhere speed adaptation buffering of cells should be performed. In this paragraph the delay will be calculated for all types of parallellisation. In Figure 14 the speed transition is shown. At speed translator p cells arrive at the fast rate (155Mbit/s or more). Next, they are sent through the slow part at a speed x times smaller (with x being the parallellisation factor). Finally they are transmitted at speed translator q at the original fast speed. In reality the slow part in Figure 14 is the actual ATM switch and the fast parts are the links between these switches.

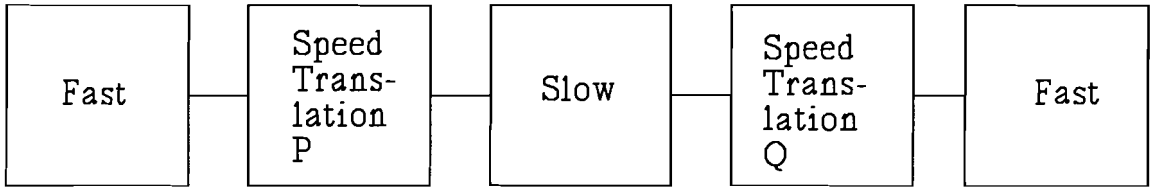

Figure 14: Buffering delay when using parallellisation

When using bitparallellisation first x bits are collected at speed translator p. These x bits are then sent parallel through the slow part. This slow part works x times slower, so after the first bits are sent parallel we have to wait x bittimes before we can send the next bits parallel. This is exactly the time we need to collect the next x bits at speed translator p.
When x bits arrive parallel at speed translator q we can start to send them (fast) immediately, because this takes x bittimes which is equal to the time it takes for the next parallel bits to arrive. This means there's no extra delay at speed translator q. At speed translator p the first bit of a cell is transmitted x bittimes after the last bit of the preceding cell. At fast speed this was one bittime, so the buffer delay for bitparallellisation is given by:

$$Delay_{bitparallellisation} = (x-1) \bullet bittime \qquad (1)$$

Next, the speed adaptation buffer delay for cellparallellisation is calculated:
When a cell arrives at speed translator p it should be buffered in order to get the x bittimes distance between bits in the slow part. The calculation of this speed adaptation buffer delay is included in the buffer delay calculation at speed translator q.

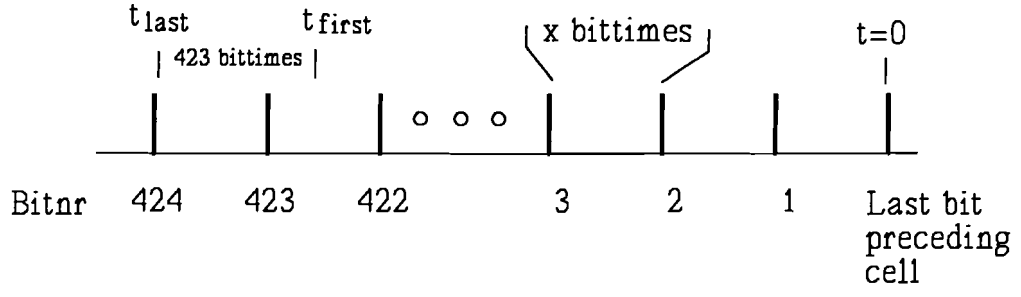

Figure 15: Cell arrival at speed translator q for cellparallellisation

When using cellparallellisation the easiest way to do speed translation is to collect a whole cell at speed translator q and then send it fast, but during the sending of bits at speed translator q some bits will arrive out of the slow part, so the sending at q can be started a little earlier. This makes the calculation more complex:
The buffer delay at speed translator q is minimal when the last bit of a cell arrives (slow) at the same time it should be transmitted fast (Figure 15). The time the last bit of a cell arrives at speed translator q is given by:

$$t_{last} = 424 \bullet x \bullet \text{bittime} \tag{2}$$

This is also the time this last bit should be transmitted (condition for minimal delay) at speed translator q, so the first bit should be transmitted at t_{first}:

$$\begin{aligned} t_{first} &= t_{last} - 423 \bullet \text{bittime} \\ &= ((x-1) \bullet 424 + 1) \bullet \text{bittime} \end{aligned} \tag{3}$$

At the original speed there was a distance of one bittime between the first bit of this cell and the last bit of the preceding cell. So now the total delay for cellparallellisation is given by:

$$\text{Delay}_{\text{cellparallellisation}} = (x-1) \bullet 424 \bullet \text{bittime} \tag{4}$$

The delay when using the combination of parallellisation in which first cellparallellisation is performed and then bitparallellisation, is calculated by simply adding the two delays and adding one bittime. This bittime has to be added because at both delay calculations we subtracted one bittime which was the original distance between the last bit of the preceding cell and the first bit of the current cell. This all results in (5).

$$\begin{aligned} \text{Delay}_{\text{combination}} &= ((x-1) \bullet 424 + x \bullet y) \bullet \text{bittime} \\ \text{With } x &= \text{cellparallellisation factor} \\ y &= \text{bitparallellisation factor} \end{aligned} \tag{5}$$

These calculations show that the speed adaptation delay is the least when using bitparallellisation (0.019 µs when x=4).
When using cellparallellisation the delay is much larger (8.2 µs when x=4) and now is of significant influence to the total delay of a cell. This might be compensated by the more evenly spread traffic, which reduces other delays and which overall increases the performance, but this should be further investigated.

4.2 Partitioning

When the ATM switch is implemented, the design should be divided in smaller parts in order to create a modular expandable switch and in order to make the switch better repairable, testable and maintainable. Actually, partitioning is 'cutting' the switch into smaller pieces, so in contrast to parallellisation, partitioning doesn't modify the original design. A three

dimensional object can be cut in three ways, resulting in two essentially different methods of partitioning (Figure 16 and Figure 17).

The first way to 'cut' the ATM switch is visualised in Figure 16 and is called slicing. This means the switch is divided in p planes which are exactly the same, built of y bits wide ASE's (with x=p*y and yε{1,2,...} and pε{2,3,...,x}). The major advantage of this kind of partitioning is it leads to all equal parts.

The second and third way of cutting the ATM switch (Figure 17) are both called logical partitioning. The division of logical partitioning in stage partitioning (cut a) and layer partitioning (cut b) can be made but makes no difference for the general ATM switch. Both these methods result in parts which exist of one or more ASE's or just a part of an ASE which are all x bits wide (with x = parallellisation) and which might all be equal (depending on the design).

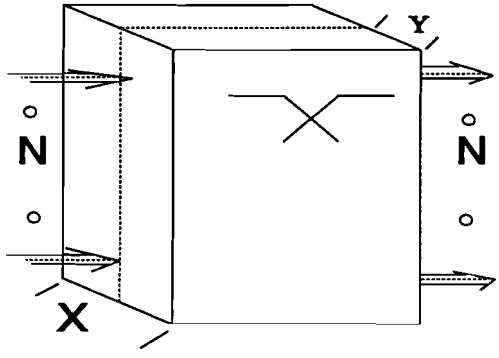

Figure 16: Slicing

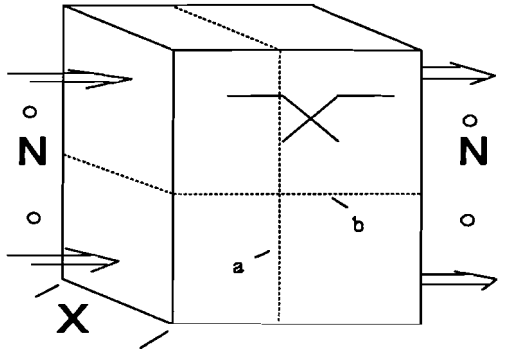

Figure 17: Logical partitioning

Because slicing often isn't sufficient, logical partitioning and slicing are often used in combination.

4.3 Classification based on implementation aspects

Based on the performed kind of partitioning and parallellisation, the switches can be classified. Table 2 shows the names I've given the switch classes. Figure 18 to Figure 21 are examples of each switch class.

This table doesn't include switches with the combination of bit- and cellparallellisation or the combination of slicing and logical partitioning, because I try to keep the classes as pure as possible.
Since partitioning has to be performed on all switch designs to get a modular design, the options for this part of the table are slicing and logical partitioning. The options for parallellisation are bitparallellisation, cellparallellisation and no parallellisation. This last option has been added because it might be possible that VLSI technology develops in a way that the ATM speed gets within limits. Besides that, there are already switch designs on

which parallellisation isn't performed. The combination of slicing and no parallellisation isn't possible because parallellisation has to be performed to be able to perform slicing.

Table 2: Five classes based on implementation aspects

Partitioning \ Parallellisation	Cell	Bit	None
Slicing	Replicated	Bit-sliced	XXXX
Logical	Dilated	Wide	Unmodified

The combination of bitparallellisation and slicing leads to the bit-sliced ATM switch. The switches of this kind exist of identical switch planes which are connected to one general controller. The use of a general controller is possible because all simultaneous arriving bits belong to the same cell, so each switch plane has to switch them to the same outlet. A good example of a bit-sliced switch is the Atom switch in Figure 18.

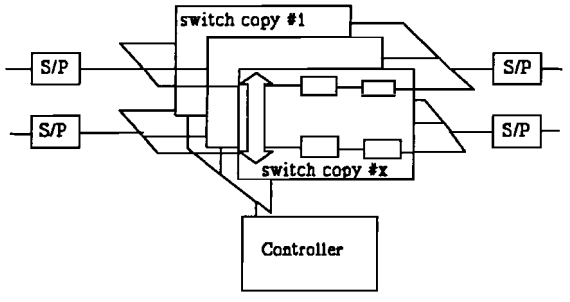

Figure 18: Bit-sliced Atom switch

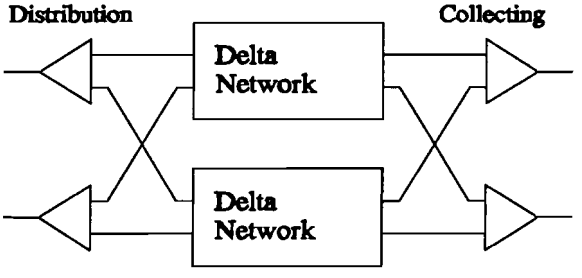

Figure 19: Replicated delta switch

The combination of slicing and cellparallellisation results in a replicated ATM switch. This means the switch is built of several identical switch copies which all have their own controller, because every switch copy handles a different cell. This increases fault tolerance, because when a switch copy is broken, the others can still switch cells. Figure 19 shows a 2-replicated Delta network. Typical for a switch with cellparallellisation are the expanders which distribute the cells over the switch copies, and the concentrators which collect the cells from the switch copies.

The combination of logical partitioning and bitparallellisation results in a wide switch. Figure 21 shows a wide Batcher-Banyan switch, which is logical partitioned in a Batcher network and a Banyan network.

The combination of no parallellisation and logical partitioning results in an unmodified switch. The switch design has not been modified by parallellisation in order to get a better

implementable switch. Figure 21 without the s/p converters and with one bit wide links between the ASE's is a good example of an unmodified switch.

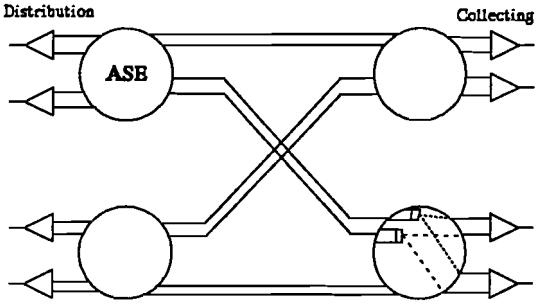

Figure 20: 2-Dilated 4x4 delta switch

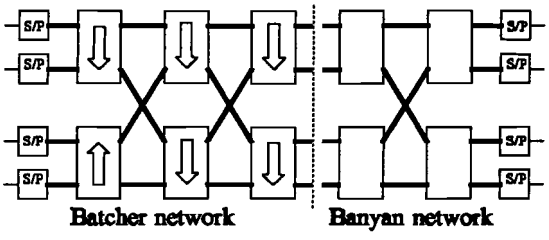

Figure 21: Wide Batcher-Banyan switch

The combination of logical partitioning and cellparallellisation results in a dilated ATM switch. This means there's only one real switch. But all links and buffers are multiplied x times. The cells are equally distributed over the x links and when a certain link is assigned to a cell, it is followed throughout the switch: When a cell uses the second link of an ASE, it should everywhere take the second link and buffer of the ASE's it passes. Figure 20 shows a 2-dilated Delta network. Typical are the duplicated links and queues.

4.4 Summary and conclusions

This chapter discusses the methods to overcome limitations in technology, when implementing ATM switches. To reduce speed, parallellisation should be performed. This can be bitparallellisation, which has a small buffer delay and guarantees preservation of cell sequence integrity, cellparallellisation, which spreads the traffic more evenly, or a combination of bit- and cellparallellisation.
To divide the design in parts which fit on one chip and to get a modular design, partitioning has to be performed. This can be slicing, which leads to exactly the same parts or logical partitioning, which is used when slicing on its own is not sufficient.
Based on this partitioning and parallellisation, five classes of switches are identified: The replicated switch, the bit-sliced switch, the dilated switch, the wide switch and the unmodified switch.

4.5 Bibliography

[JANS92] Jansen, J.W.H. en L.A.J. van den Heuvel
ONTWERP VAN EEN ATM PACKETSWITCH M.B.V. IDASS
Eindhoven: Technische Universiteit Eindhoven, 1992
Traineeship report nr. EB 388

[TAKE91] Takeuchi, T. en H. Suzuki, T. Aramaki
SWITCH ARCHITECTURES AND TECHNOLOGIES FOR ASYNCHRONOUS TRANSFER MODE
IEICE Transactions
Vol E 74, No 4 (Apr 1991), p. 752-760.

[TOBA90] Tobagi, F.A.
FAST PACKET SWITCH ARCHITECTURES FOR BROADBAND INTEGRATED SERVICES DIGITAL NETWORKS
Proceedings of the IEEE
Vol 78, No 1 (jan 1990), p. 133-167.

[VRIE92] Vries, R.J.F. de
SWITCH ARCHITECTURES FOR THE ASYNCHRONOUS TRANSFER MODE
Enschede: Technische Universiteit Twente, Doctoral dissertation.
Leidschendam: PTT Research, 1992.

[WULL89] Wulleman, R. and T. van Landegem
COMPARISON OF ATM SWITCHING ARCHITECTURES
International Journal of digital and analog cabled systems
Vol 2, No 4 (oct-dec 1989), p. 211-225.

5 Implementation related to design

In chapter 3 the design aspects of an ATM switch were described and in chapter 4 the implementation aspects of a general ATM switch were described. This chapter will take a closer look at combinations of switch designs and implementation aspects, because the best kinds of parallellisation and partitioning to be performed on a switch often depend on the switch design.
An ATM switch internally uses some kind of multistage structure and some kind of switching elements. Since the switches are presumed to be large (e.g. 1000 x 1000) they have to be partitioned. To determine the best way to do this partitioning the existing switches are divided in three categories based on complexity (which is why the switch has to be partitioned) and internal speed (which is why parallellisation should be performed).
The three categories are the small ASE (size 2x2 or 4x4) based switches, the limited size ASE (size 8x8 to 64x64) based switches and the unlimited size ASE (8x8 and larger) based switches.

5.1 Small ASE based switches

The small ASE based switches exist of simple ASE's. The size of these ASE's is small (2x2 or 4x4), which makes the MIN's these ASE's are used in, regular and modular. Because of the small size, the ASE's don't have to be partitioned themselves, but on the MIN logical partitioning should be performed.

The small ASE based MIN's are Delta networks, Batcher-Banyan networks or modified Delta networks (Figure 9). The modified Delta networks are in fact Delta networks with dilation and/or replication, i.e. these are Delta networks with cellparallellisation, but the goal of parallellisation in these networks is to create multiple paths in order to reduce the contention probability, instead of being able to reduce speed. This parallellisation doesn't modify the basic interconnection structure of the network, because only links are multiplied (dilation) or the whole network is multiplied (replication). So when logical partitioning is performed the basic structure is still the same as the one of the unmodified Delta network. This leaves only the structure of the unmodified Delta network and the Batcher-Banyan network to examine, to find the best way of logical partitioning small ASE based switches.

5.1.1 Structure of small ASE based networks

Batcher-Banyan networks belong to the class of non-blocking multi-path networks. A Banyan network guarantees that if the cells at the inlets are sorted by destined output, they will arrive at the outlets without contention. The ASE's of a Banyan network rout cells by examining the routing tag.
The function of the Batcher network is to sort cells in order of destined output (that's why the Batcher network precedes the Banyan network). Figure 22 shows the recursive structure of a Batcher network. A NxN Batcher network (S_N) is constructed of a NxN Delta network

(M_N) preceded by two N/2xN/2 Batcher networks ($S_{N/2}$). The ASE's in the Batcher network sort the arriving cells by comparing the routing tag (arrow up means descending order and arrow down ascending order).
In Figure 9 it is shown that Delta networks are Banyan networks, so a Batcher-Banyan network is only constructed of Delta networks, but the ASE's in the Batcher part sort cells and the ASE's in the Banyan part rout cells. This leaves only the structure of Delta networks to examine, to find the best way of logical partitioning small ASE based networks.

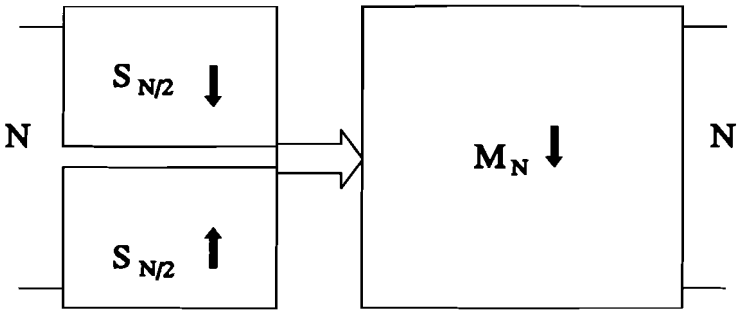

Figure 22: Recursive structure of the Batcher network

The Delta network is constructed of stages of ASE's in a way that the outlets of the ASE's of a stage are only connected to the inlets of the next stage ASE's. A Delta network of p stages has size $2^p x 2^p$ and uses $p*2^{p-1}$ ASE's.
Now, the best way to perform logical partitioning on the large Delta networks has to be found. Instead of finding the best way to divide a large Delta network into small parts, there's searched for a small network which is used as building block to construct a large Delta or Batcher-Banyan network.

Constructing a large Delta network, using the building parts, means at least two parts are put in a row. When the number of stages in one part is p, two parts in a row have 2p stages, which means the larger network has a size of $2^{2p} x 2^{2p}$. In literature it can be found it's possible to put a 32x32 Delta network and even a 64x64 Delta network on one chip, so the complexity of the small part should approximately be the same (100 to 200 ASE's).
The simplest would be to let a building part be one 32 x 32 Delta network, but then the larger networks can only have sizes (1024x1024, 32^3x32^3 etc.), while for a Batcher network also networks of sizes 2x2, 4x4, 8x8 etc. are needed.

A better solution is to let the basic part be a stage of smaller networks (Figure 23). If it is assumed the number of stages in such a smaller network is p, the size of these smaller networks is $2^p x 2^p$. So, to be able to create networks of size 2x2, 4x4 etc. p should be one, i.e. the smaller networks are in fact only one ASE. Next, it is assumed that t ASE's fit on one chip, i.e. a part is a stage of t ASE's. To create a larger network several (n) parts are put in a row, and next, several (k) of these rows are created. By placing n parts in a row, n*p=n stages are created. As described before, a Delta network of n stages has a size of $2^n x 2^n$. The number of rows (of n parts) that are created is k, so all parts are fully used if the number of

inlets and outlets (=2kt) matches with the number of stages (n), i.e. $2^n=k*2t$. This is only possible if both t and k are powers of 2. Another constraint on a part is that one part should exist of 100 to 200 ASE's, so $t=2^7=128$.
Which means the best solution is to let one part be a stage of 128 ASE's. Since $2^n=k*2^8$ and k is a power of two, all networks with size 2^n x 2^n with $n\geq 8$ can be created.
The ASE's in a part should be able to both rout cells and order cells, because the parts are used to construct both Batcher and Banyan networks with.

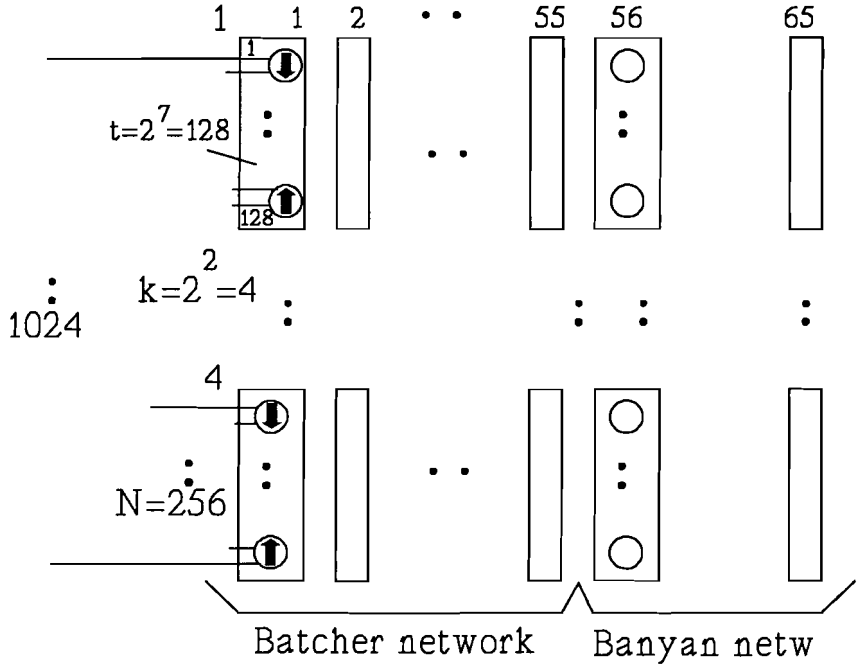

Figure 23: Batcher-Banyan network constructed of parts which are one stage of ASE's

In Figure 23 it is shown how these parts are used to construct a 1024x1024 Batcher-Banyan network, i.e. how a 1024 x 1024 Batcher-Banyan network should be logical partitioned. A Batcher network of size 1024x1024 exists of 1+2+...+10=55 stages and a Banyan network of size 1024x1024 exists of 10 stages, so the total network has 65 stages. Since each part exists of one stage of 128 ASE's ($t=2^7=128$), rows are created of 65 parts (65 stages). Since each part has 256 inlets and outlets (2t=256), four such rows are created ($k=2^2=4$).

A major disadvantage of partitioning in a way that one part is one stage of ASE's, is that the regular interconnection structure which is well suited to implement on-chip, now is off-chip. Since the small network sizes (which is why one part has to be one stage of ASE's) are only needed in the Batcher network, it is preferred to partition the other small ASE based networks (i.e. the unmodified and the modified Delta networks) in a way that one part is a stage of larger (than 2x2) networks, e.g. four 16x16 networks in which the ASE's only need to rout cells. In this way the regular interconnection structure is on-chip, but as described before, now only networks of sizes NxN, N^2xN^2, etc. can be created.

5.2 Limited size ASE's

A limited size ASE is an ASE which can't grow beyond some size, because of limitations in technology. These kinds of ASE's use a TDM bus as a switching transfer medium or use a shared or common buffer with a TDM-bus as access to the bufferspace (Figure 24). The bus has to operate at a speed of N times the inlet (outlet) speed (N is the number of inlets(outlets)). The ASE size is limited, because for large ASE's the bus speed gets too large. Typical sizes are 8x8 to 64x64 (bus speed 1.2Gbit/s to 9.9Gbit/s for an inlet speed of 155Mbit/s).
When partitioning these kinds of ASE's, slicing is preferred, which is now explained: The TDM-bus has the highest speed in the ASE, which means parallellisation should reduce the TDM-bus speed to get an implementable ASE. Since the maximum implementable speed between parts is smaller than the maximum implementable speed on one part, the TDM-bus should be on one part after partitioning is performed, because otherwise the speed should be reduced more, which means more parallellisation is needed. Since the only partitioning after which the TDM-bus is on one part, is slicing, this is preferred for partitioning limited size ASE's.
The MIN structure these ASE's are placed in can be (paragraph 3.8) a Benes network (rearrangable non-blocking regarding connection) or a Clos network (strictly non-blocking regarding connection). These MIN's are studied in more detail in paragraphs 5.4 and 5.5, but first is explained what growing of these ASE's means.

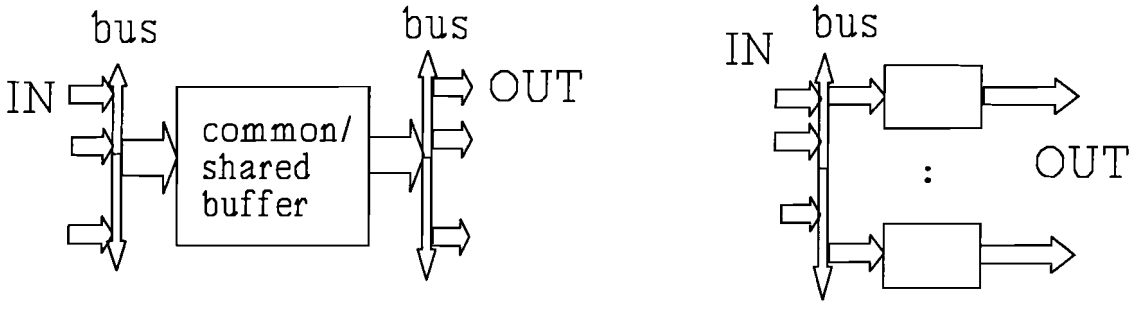

Figure 24: General design of limited size ASE's

5.2.1 Growing of limited size ASE's

As previously described, there are two kinds of limited size ASE's: The ASE's which use a TDM-bus as a switching transfer medium, and the ASE's which use a common or shared buffer.
When an extra inlet is added to a TDM-bus based ASE, the speed of the bus should increase. Since this bus-speed can't increase because of limitations in technology, extra parallellisation should be performed. Next to that the bufferspace should increase. In other words, adding one extra inlet to the ASE means every output module of the ASE should be adapted. So the costs of these ASE's increase quadratically regarding inlets.

When an extra outlet is added to a TDM-bus based ASE, an extra output module should be added to the ASE, in which the size of the buffer depends on the number of inlets of the ASE. This means the cost of these ASE's increase quadratically regarding outlets.

When an extra inlet is added to a shared or common buffer based ASE, the speed of the inlet TDM-bus should increase. Again, this speed can't increase because of limitations in technology, so extra parallellisation has to be performed and the bufferspace for every (logical) queue in the ASE should increase, so the costs of these ASE's increase quadratically, regarding inlets.
When an extra outlet is added to the ASE, the speed of the outlet bus should increase. So again extra parallellisation has to be performed. And again the bufferspace should increase depending on the number of inlets of the ASE. This means quadratically increasing costs of these ASE's, regarding outlets.

So to make a good cost comparison (paragraphs 5.4 and 5.5) the assumption that the limited size ASE costs increase quadratically is a good approximation. Actually, the costs are non-linear, because at some size extra partitioning has to be performed, which makes the costfunction discontinuous.

5.3 Unlimited size ASE based switches

An unlimited size ASE internally doesn't use ASE size dependable speeds (like the limited size ASE's TDM-bus speed). These ASE's usually use a matrix of slotted buses. These are buses from one inlet to all output or crosspoint modules which operate at the same speed as the external links, so new limitations in technology are not met. The disadvantage of these ASE's is their large hardware complexity, which is why these ASE's should be partitioned. These ASE's best can be logical partitioned, because they have a large hardware complexity and because logical partitioning can result in all equal parts (e.g. one part is x output/crosspoint modules) due to the regular structure of these ASE's. The MIN's these ASE's are placed in, are Clos and Benes networks, which are discussed in paragraphs 5.4 and 5.5, but first the growth of these ASE's is studied in more detail.

5.3.1 Growing of an unlimited size ASE

Figure 25 and Figure 26 clearly show the regular and modular designs of unlimited size ASE's. Growing of the crosspoint structure (Figure 26) means adding an input and an output bus and 2N+1 modules. Growing of the output module structure (Figure 25) means adding n buses and 2N+n modules. This grow factor 2N+x means the ASE hardware grows quadratically, so the costs of these ASE's also grow quadratically.

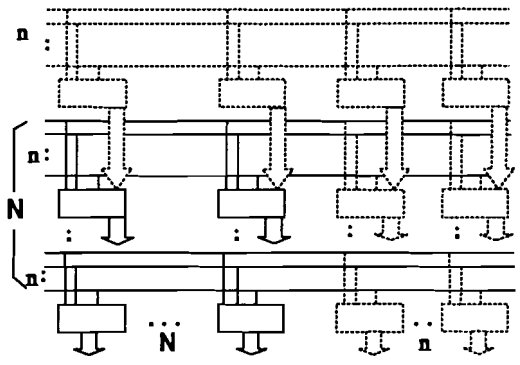

Figure 25: Growing of output module based unlimited size ASE

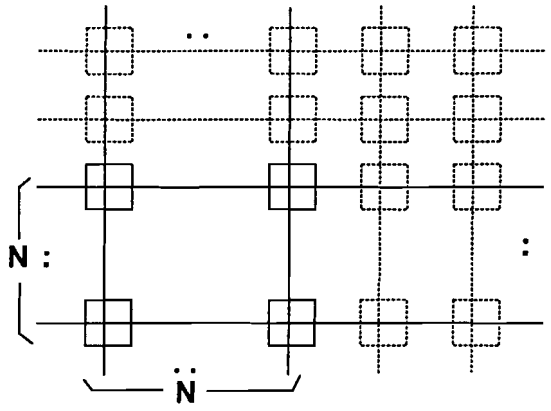

Figure 26: Growing of crosspoint module based unlimited size ASE

5.4 Benes networks

A general three-stage MIN with as many inputs as outputs is shown in Figure 27. It is the kind of network in which limited and unlimited size ASE's are used. The MIN should be logical partitioned in individual ASE's.

V.E. Benes proved the three-stage MIN of Figure 27 to be rearrangable non-blocking regarding connection, if m≥n. This means a connection can always be set up between a free inlet and a free outlet after the existing connections have been rearranged if m≥n, but cells still can get lost internally as a result from buffer overflow for instance. The rearrangable non-blocking property is only of interest when the ASF uses connection oriented routing (see paragraph 3.4).

The Benes network has the advantage that if m=n=r, all ASE's are of equal size. A disadvantage is that it may need some extra time for a connection set up, because existing connections might need to be rearranged.

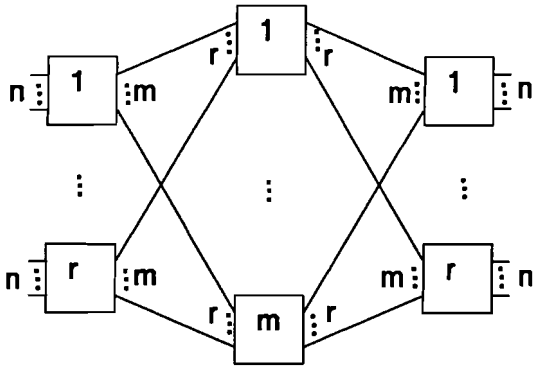

Figure 27: General three stage MIN

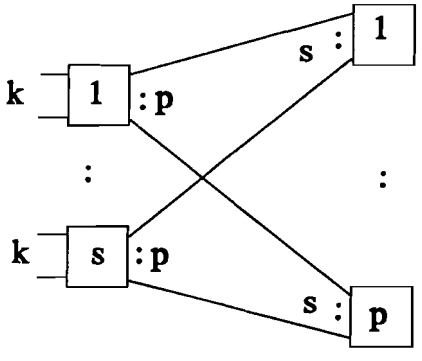

Figure 28: Folded MIN

To reduce the number of ASE's the three stage MIN might be implemented using

bidirectional ASE's (Figure 28), this means the ports of the switch and the interstage links are bidirectional. These networks are called folded networks. This folding is for instance easily achieved by pairing an inlet and an outlet if the ASE is symmetric (i.e. the number of inlets is equal to the number of outlets). When an arriving cell is destined for an outlet in the same first stage ASE, it is directly routed to that outlet by the first stage ASE and when a cell arrives destined for an outlet of a different first stage ASE it is transferred to a second stage ASE, which in turn will direct the cell to the correct first stage ASE.

Since for the folded Benes network still $p \geq k$ should be valid, the first stage ASE's each contain 2k bidirectional ports (p=k). The second stage ASE's contain s bidirectional ports, so if s=2k the folded Benes network is also constructed of all equal ASE's.

The folded MIN has the disadvantage that it can't grow any more, while the unfolded MIN can only grow using 3d arrangement of the MIN's (Figure 29). This arrangement means N switches are put in a rank and are connected to a second rank of N switches, which is placed orthogonal opposite the first rank.

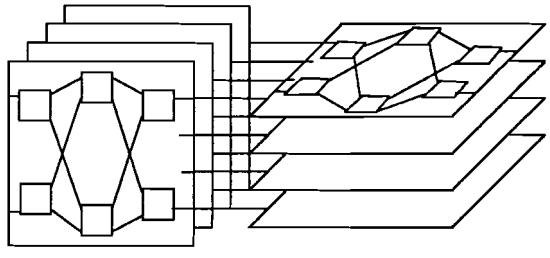

Figure 29: 3D arrangement of MIN's

The reduction of the number of ASE's by 'folding' the network doesn't guarantee the total ASF to be cheaper, because the ASE's of a folded ASF are more complex. So now equally sized folded and unfolded Benes networks are compared. This comparison is done by means of the chipcount, because this is the major factor determining the costs of an ASF.

The comparison can be done following two strategies. The first is to define some ASE to be the basic building block to construct both the folded and unfolded network with and to assume this basic building block exactly fits on one chip. This means every ASE in both networks is replaced by a basic building block. The search for the network which needs the least number of chips now is the same as the problem of finding the network which needs the least number of ASE's. The size of the basic building block should be the size of the largest ASE needed in both networks. If a smaller ASE is needed, simply some inlets and/or outlets of the basic building blocks are not used.
The second strategy is to assume that the chipspace needed for an ASE is fully determined by the number of inlets and outlets. Then by comparing the complexity of the ASE's, it is determined how many ASE's of a specific type (i.e. size) fit on one chip.
Both these strategies are now used to make the cost comparisons.

5.4.1 Comparison using basic building blocks

The use of basic building blocks to compare chipcount, has the advantage that the used chips are all exactly the same, so a network exists of all equal chips. Disadvantage of this strategy is that if the ASE's actually needed are smaller, several of the inlets and/or outlets are not used, which means the basic building block and thus chipspace is used inefficient.

As seen before, when using a basic building block the problem of minimizing the chipcount becomes the same as minimizing the ASE count. First the minimal ASE count for folded Benes networks is calculated, then it is calculated for unfolded Benes networks. Both the networks are assumed to have the same size (NxN).

A folded Benes network of size NxN means in Figure 28 p=k and s=N/k, so the number of ASE's in the folded Benes network is given by:

$$\text{ASEcount} = \frac{N}{k} + k \tag{6}$$

To minimize this, the derivative should be set to zero (7), which gives a result for k.

$$\begin{aligned} -\frac{N}{k^2} + 1 &= 0 \\ \Leftrightarrow k &= \sqrt{N} \end{aligned} \tag{7}$$

Now the total number of basic blocks needed, is calculated:

$$\text{basic blocks} = \frac{N}{k} + k = 2\sqrt{N} \tag{8}$$

The problem of minimizing the number of ASE's in the unfolded network is solved in a similar way. An unfolded Benes network means in Figure 27 r=N/n and m=n. Now the ASE count is determined for the general unfolded Benes network:

$$\text{ASEcount} = 2\frac{N}{n} + n \tag{9}$$

Again this is minimized by setting the derivative to zero, which gives a result for n:

$$\begin{aligned} -2\frac{N}{n^2} + 1 &= 0 \\ \Leftrightarrow n &= \sqrt{2N} \end{aligned} \tag{10}$$

Finally the number of basic blocks needed is determined:

$$\text{basic blocks} = 2\frac{N}{n} + n = 2\sqrt{2N} \qquad (11)$$

The largest ASE's in the folded network are the first stage ASE's, which each have $2\sqrt{N}$ bidirectional ports (this can be compared with unidirectional ASE's of size $2\sqrt{N} \times 2\sqrt{N}$). The largest ASE's in the unfolded network are the first and third stage ASE's which have a size of $\sqrt{(2N)} \times \sqrt{(2N)}$. This means the basic building blocks which are used to construct the MIN's with, should be ASE's of size $2\sqrt{N} \times 2\sqrt{N}$.

The conclusion of the comparison of the folded and unfolded Benes network using basic building blocks is that the folded Benes network with a minimum number of ASE's only needs 71% of the number of blocks needed for the unfolded Benes network with a minimum number of ASE's. The basic blocks of size $2\sqrt{N} \times 2\sqrt{N}$ are however inefficient when used to construct the unfolded network, because then only 71% of the inlets and outlets of them are actually used.

To finish this paragraph a calculation example for an ASF size of 1000 x 1000 is given:
Using formulas (7) and (10) it is found that k=32 and n=45. So the unfolded Benes network is constructed of: 46 ASE's of size 45 x 45
45 ASE's of size 23 x 23
The actual ASF size of the unfolded Benes network is 23x45=1035 and it needs 91 basic building blocks.
The folded Benes network is constructed of:
32 ASE's of size 64 x 64
32 ASE's of size 32 x 32
The actual ASF size of the folded Benes network is 32x32=1024 and it needs 64 basic building blocks.
The basic blocks should have a size of 64 x 64.

5.4.2 Comparison using ASE size for complexity

The use of ASE size to compare chipcount, has the advantage that chipspace is used more efficient, since it's possible to combine several ASE's on one chip. The disadvantage is that a network can exist of more than one kind of chip. Besides that, the assumption that ASE complexity fully depends on the number of inlets and outlets is not correct, because for instance buffer control logic, when using common or shared buffering, doesn't depend on ASE size.

Now the problem of minimizing complexity (i.e. the number of crosspoints) of the folded and unfolded Benes network is solved. Since the actual number of crosspoints is not of interest, but only the relation between the ASF size and the ASE size when having a minimum number of crosspoints, only the latter is calculated.

The unfolded Benes network has 2N/n ASE's of size n x n (first and third stage ASE's) and n ASE's of size N/n x N/n, so the total number of crosspoints is given by:

$$\text{crosspoints} = 2Nn + \frac{N^2}{n} \tag{12}$$

To minimize this, the derivative should be set to zero, which results in a relation between N and n:

$$\begin{aligned} 2N - \frac{N^2}{n^2} &= 0 \\ \Leftrightarrow N &= 2n^2 \end{aligned} \tag{13}$$

The folded Benes network has N/k ASE's of size 2k x 2k and k ASE's of size N/k x N/k. The total number of crosspoints is now calculated:

$$\text{crosspoints} = 4Nk + \frac{N^2}{k} \tag{14}$$

Again the derivative is set to zero, to get the relation between N and k, when the network has a minimum number of crosspoints:

$$\begin{aligned} 4N - \frac{N^2}{k^2} &= 0 \\ \Leftrightarrow N &= 4k^2 \end{aligned} \tag{15}$$

Now a cost comparison based on the second strategy as described before, is made:
The networks which are compared (Figure 30) are the folded and unfolded networks with a minimum number of crosspoints, because crosspoints determine ASE complexity in this comparison, the folded and unfolded networks with all equal ASE's, because the more kinds of chips there are, the higher the design costs will be, and last the unfolded and folded networks with a minimum number of ASE's.

$$\begin{aligned} N = 2q^2 = x^2 = \tfrac{1}{2}n^2 = 4z^2 = 2y^2 = k^2 \\ \Leftrightarrow q = y = z\sqrt{2} \\ \Leftrightarrow x = k = 2z \\ \Leftrightarrow n = 2z\sqrt{2} \end{aligned} \tag{16}$$

The networks that are compared should all have the same size. In Figure 30 the formulas for the ASF sizes (N) are given. In expression (16) these formulas are assumed to be equal. Now all networks can be described using only variable z.

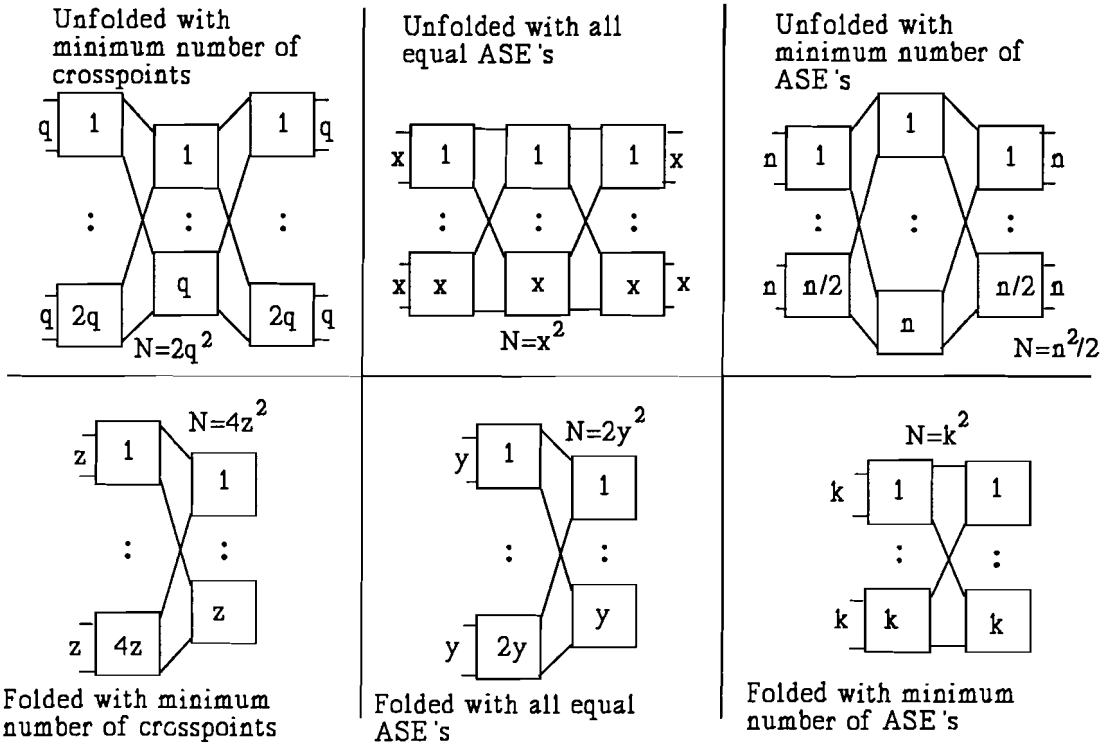

Figure 30: Six Benes networks to be compared

First the largest ASE of the compared networks is assumed to fit on exactly one chip. The largest ASE is the ASE with the largest complexity. For quadratically increasing costs of the ASE, the number of crosspoints determines the complexity. For linear increasing costs the complexity is determined by half the sum of the number of inlets and outlets.
Next, by comparing the complexity of this largest ASE with the complexity of the other ASE's, it is determined how many (whole) ASE's of one specific type (i.e. size) of ASE fit one chip (in a relative point of view to the largest ASE). Finally the chipcount of every network is calculated.

Table 3: Comparison of Benes networks

Network	#ASE's	Size	Quadr. costs		Linear costs	
			Compl.	/chip	Compl.	/chip
Unf. min. crossp.	$4z\sqrt{2}$ $z\sqrt{2}$	$z\sqrt{2} \times z\sqrt{2}$ $2z\sqrt{2} \times 2z\sqrt{2}$	$2z^2$ $8z^2$	8 2	$z\sqrt{2}$ $2z\sqrt{2}$	2 1
Unf. equal ASE's	6z	2z x 2z	$4z^2$	4	2z	2
Unf. min. ASE's	$2z\sqrt{2}$ $2z\sqrt{2}$	$z\sqrt{2} \times z\sqrt{2}$ $2z\sqrt{2} \times 2z\sqrt{2}$	$2z^2$ $8z^2$	8 2	$z\sqrt{2}$ $2z\sqrt{2}$	2 1
Fol. min. crossp.	4z z	2z x 2z 4z x 4z	$4z^2$ $16z^2$	4 1	2z 4z	2 1
Fol. equal ASE's	$3z\sqrt{2}$	$2z\sqrt{2} \times 2z\sqrt{2}$	$8z^2$	2	$2z\sqrt{2}$	1
Fol. min ASE's	2z 2z	4z x 4z 2z x 2z	$16z^2$ $4z^2$	1 4	4z 2z	1 2

The comparisons of the networks of Figure 30 for linear and quadratically increasing ASE costs are given in Table 3. The column *Network* describes the type of network. The column *#ASE's* gives the number of ASE's of each size (third column) in each network. Next, for both linear and quadratically increasing ASE costs, the complexity (*Compl.*) and the number of whole ASE's that fit on one chip (*/chip*) are given. The numbers of chips needed for every network are now calculated (Table 4) for both quadratically and linear increasing ASE costs.

Table 4: Chipcount of compared Benes networks

Network	Chipcnt (quadr)	Chipcnt (lin)
Unf. min. crossp.	$4z\sqrt{2}/8 + z\sqrt{2}/2 = z\sqrt{2}$	$4z\sqrt{2}/2 + z\sqrt{2}/1 = 3z\sqrt{2}$
Unf. equal ASE's	$6z/4 = 1.5z$	$6z/2 = 3z$
Unf. min. ASE's	$2z\sqrt{2}/8 + 2z\sqrt{2}/2 = 1.25z\sqrt{2}$	$2z\sqrt{2}/1 + 2z\sqrt{2}/2 = 3z\sqrt{2}$
Fol. min. crossp.	$4z/4 + z/1 = 2z$	$4z/2 + z/1 = 3z$
Fol. equal ASE's	$3z\sqrt{2}/2 = 3z/\sqrt{2}$	$3z\sqrt{2}/1 = 3z\sqrt{2}$
Fol. min. ASE's	$2z/1 + 2z/4 = 5z/2$	$2z/1 + 2z/2 = 3z$

The conclusion of this comparison is:
When quadratically increasing costs of ASE's are assumed the unfolded Benes networks need less chips than the folded Benes networks. Since the values for N and n and k have to be integers, all unfolded networks need about the same number of chips.
When linear increasing costs of the ASE's are assumed, there's no clear difference between the folded and the unfolded networks.
Since the cost increase of the ASE's is more close to quadratically than to linear, the conclusion of the comparison using ASE size for complexity is, that the unfolded Benes networks need less chips than the folded Benes networks. The unfolded Benes network with all equal ASE's is preferred, because of the lower chipdesign costs, since it only uses one kind of chip.

This paragraph will end with an example for an ASF of 1000 x 1000 in which the folded and the unfolded Benes networks with all equal ASE's are compared using crosspoints for complexity (quadratically increasing costs).
The unfolded Benes network with all equal ASE's (x=32) is constructed of:

96 ASE's of size 32 x 32 with an actual ASF size of 32x32=1024 [2 ASE's per chip]

The folded Benes network with all equal ASE's (y=23) is constructed of:

69 ASE's of size 46 x 46 with an actual ASF size of 23x46=1058 [1 ASE per chip]

The unfolded Benes network exists of 96/2=48 chips and the folded Benes network of 69/1 chips.

5.5 Clos networks

C. Clos proved the network of Figure 27 to be strictly non-blocking regarding connection, if $m \geq 2n-1$. This means a connection can always be set up between a free inlet and a free outlet if $m \geq 2n-1$, but cells can still get lost internally as a result from buffer overflow for instance. This strictly non-blocking property is only of interest if the ASF uses connection oriented routing (see paragraph 3.4). If the ASF uses connectionless routing, the Clos network will have the advantage over the Benes network of having a smaller cell loss probability, because there are more paths in the Clos network between every inlet and outlet. The unfolded Clos network has the disadvantage that it can't be constructed of all equal ASE's, while the folded network can be constructed of all equal ASE's: The folded Clos network (Figure 28) should have $p \geq 2k-1$. So the first stage ASE's contain 3k-1 bidirectional ports (p=2k-1) and the second (reflection or mirror) stage ASE's contain s bidirectional ports. The folded Clos network now exists of all equal ASE's, if s=3k-1.

The 3d arrangement as shown in Figure 29 is the only way to expand the Clos network, but the resulting MIN isn't strictly non-blocking regarding connection any more, but instead it is rearrangable non-blocking regarding connection, so the result is not a Clos network, but a Benes network.

5.5.1 Comparison using basic building blocks

As described in paragraph 5.4.1 the use of basic building blocks has the advantage that the folded and unfolded networks exist of all equal chips. This is especially an advantage for the unfolded Clos network, since it is impossible to construct it of all equal, 100% used ASE's. The disadvantage of using basic building blocks is still the inefficient use of chipspace, especially for the unfolded Clos network: The number of inlets of the first stage ASE's is half of the number of outlets (vice versa for third stage ASE's), while basic blocks have the same number of inlets as the number of outlets, because of symmetry of the networks (for every ASE with a certain number of inlets, there's also an ASE with the same number of outlets). So only half of the number of inlets or outlets are used of basic blocks in the first and third stage of the unfolded Clos network.

Again the problem of minimizing the number of ASE's is calculated, but now for the unfolded and folded Clos networks. This is done for networks of size NxN.
A folded Clos network means in Figure 28 p=2k-1 and s=N/k. The number of ASE's in a folded Clos network is now given by:

$$\text{ASEcount} = \frac{N}{k} + 2k - 1 \tag{17}$$

It's derivative is set to zero to find the value of k for which the network has a minimum number of ASE's:

$$\begin{aligned} -\frac{N}{k^2} + 2 &= 0 \\ \Leftrightarrow k &= \frac{1}{2}\sqrt{2N} \end{aligned} \tag{18}$$

And finally, the number of basic blocks needed is calculated:

$$\text{basic blocks} = \frac{N}{k} + 2k - 1 = 2\sqrt{2N} - 1 \tag{19}$$

Now the minimum number of ASE's in the unfolded Clos network is calculated. An unfolded Clos network means in Figure 27 m=2n-1 and r=N/n. The number of ASE's in the general unfolded Clos network is now given by:

$$\text{ASEcount} = 2\frac{N}{n} + 2n - 1 \tag{20}$$

Next, the derivative is set to zero to find the expression for n:

$$-2\frac{N}{n^2}+2=0 \quad (21)$$
$$\Leftrightarrow n=\sqrt{N}$$

And again the number of basic blocks needed is calculated:

$$\text{basic blocks} = 2\frac{N}{n}+2n-1 = 4\sqrt{N}-1 \quad (22)$$

The largest ASE's in the folded Clos network are the first stage ASE's which each have $1.5\sqrt{(2N)}$ bidirectional ports (Compare with a unidirectional ASE of size $1.5\sqrt{(2N)}$ x $1.5\sqrt{(2N)}$).
The largest ASE's in the unfolded Clos network are the first and third stage ASE's (size $\sqrt{N}$ x $2\sqrt{N}-1$). Since the first stage ASE's have $2\sqrt{N}-1$ outlets and the third stage ASE's have $2\sqrt{N}-1$ inlets, the basic building blocks should be ASE's of size $2\sqrt{N}-1$ x $2\sqrt{N}-1$.

The conclusion of this comparison is that the folded Clos network with a minimum number of ASE's only needs 70% of the number of basic blocks needed for the unfolded Clos network with a minimum number of ASE's. The basic blocks of size $2\sqrt{N}-1$ x $2\sqrt{N}-1$ are however inefficient, when used to construct the unfolded network, because then only about 50% of the inlets and/or outlets of them are actually used.

To finish this paragraph a calculation example for an ASF size of 1000 x 1000 is given:
Using formulas (21) and (18) it is calculated that n=32 and k=23. So now the unfolded Clos network is constructed of:

64 ASE's of size 32 x 63
63 ASE's of size 32 x 32

The actual ASF size of the unfolded Clos network is 32x32=1024 and it needs 127 basic building blocks.
The folded Clos network is constructed of:

44 ASE's of size 68 x 68
45 ASE's of size 44 x 44

The actual ASF size of the folded Clos network is 44x23=1012 and it needs 89 basic building blocks. The basic blocks should have a size of 68 x 68.

5.5.2 Comparison using ASE size for complexity

The advantages and disadvantages of the comparison based on using ASE size for complexity, have already been discussed in paragraph 5.4.2.
The relations between ASF size and ASE size when using a minimum number of crosspoints, for the folded and unfolded Clos networks are now calculated. The unfolded Clos network uses 2N/n ASE's of size n x 2n-1 (first and third stage) and 2n-1 ASE's of size N/n x N/n (mid stage). So the number of crosspoints in the unfolded Clos network is:

$$\text{crosspoints} = (2n-1)(2N+\frac{N^2}{n^2}) \tag{23}$$

To minimize this the derivative is set to zero, which gives the relation between N and n:

$$\begin{aligned} & 2n^3-Nn+N = 0 \\ \Leftrightarrow\ & N = \frac{2n^3}{n-1} \end{aligned} \tag{24}$$

Now the relation between N and k for the folded Clos network when having a minimum number of crosspoints is calculated. The folded Clos network has N/k ASE's of size 3k-1 x 3k-1 (first stage) and 2k-1 ASE's of size N/k x N/k (mirror stage). The number of crosspoints in this network is given by:

$$\text{crosspoints} = \frac{N}{k}(3k-1)^2+(\frac{N}{k})^2(2k-1) \tag{25}$$

Next, the derivative is set to zero to calculate the relation between N and k:

$$\begin{aligned} & 9k^3-(1+2N)k+2N = 0 \\ \Leftrightarrow\ & N = \frac{9k^3-k}{2k-2} \end{aligned} \tag{26}$$

The second strategy of comparison is now used to compare the unfolded and folded Clos networks with a minimum number of crosspoints, because crosspoints determine the ASE complexity in this comparison, the folded Clos network with all equal ASE's, because the less kinds of chips there are, the lower the design costs will be, and last the unfolded and folded Clos networks with a minimum number of ASE's. All these networks can be found in Figure 31.

First of all the compared networks should all have the same size. In expression (27) the formulas for network sizes N (see Figure 31) of each network are assumed to be equal (approximations are used because the formulas are too hard to solve). Now all networks can be described using only variable x:

$$\begin{aligned} N = \frac{2y^3}{y-1} = n^2 = 3z^2-z = \frac{9x^3-x}{2x-2} = 2k^2 \\ \Leftrightarrow\ z \approx 1.25x \\ \Leftrightarrow\ k = y \approx 1.5x \\ \Leftrightarrow\ n \approx 2.2x \end{aligned} \tag{27}$$

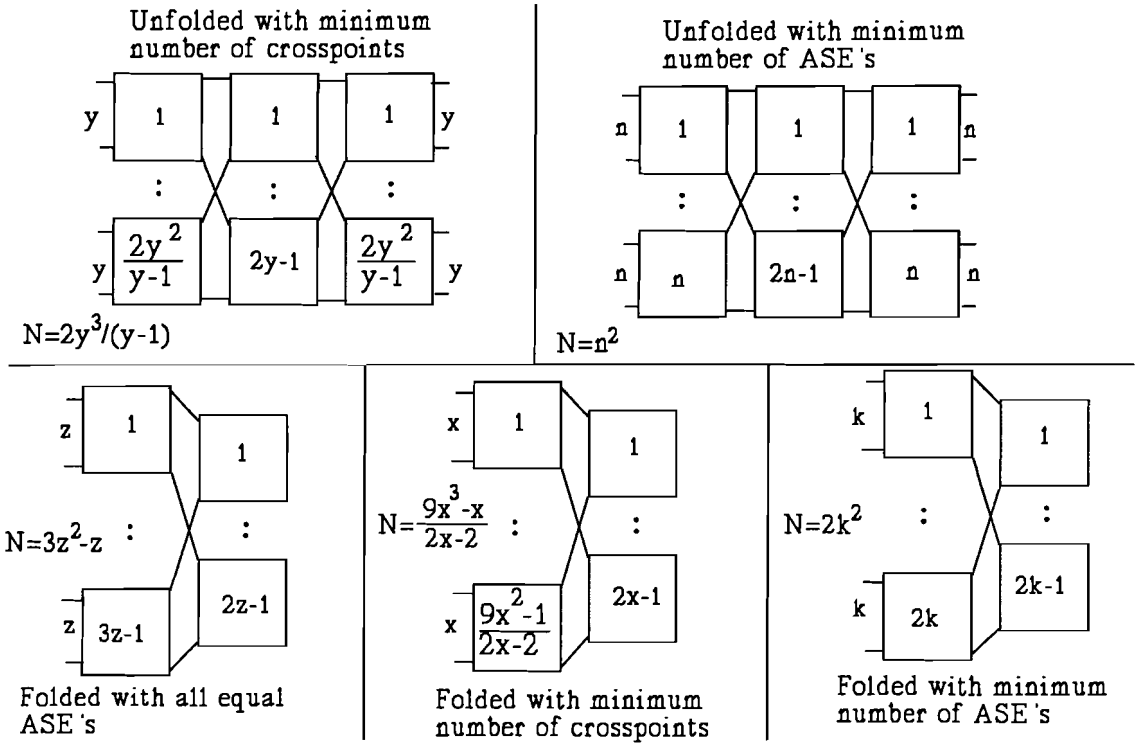

Figure 31: Five Clos networks to be compared

The comparisons of the networks of Figure 31 for linear and quadratically increasing ASE costs are given in Table 5. The column *Network* describes the type of network. The column *#ASE's* shows the number of ASE's of each size (third column) in each network. Next, for both linear and quadratically increasing ASE costs, the complexity (*Compl.*) and the number of whole ASE's that fit on one chip (*/chip*) are given. The number of ASE's per chip of every specific kind of ASE is hard to calculate using the formulas. Instead it's done by calculating actual numbers of crosspoints for values of x in the range 10 to 50 (which can be compared with ASF's of size 500 to 11500).

Table 5: Comparison of Clos networks

Network	ASE's	size	Quadr. costs		Linear costs	
			Compl.	/chip	Compl.	/chip
Unf. min. crossp.	3x-1	$(\frac{4.5x^2}{1.5x-1}) \times (\frac{4.5x^2}{1.5x-1})$	$(\frac{4.5x^2}{1.5x-1})^2$	2	$\frac{4.5x^2}{1.5x-1}$	1
	$\frac{9x^2}{1.5x-1}$	(1.5x) x (3x-1)	$4.5x^2-1.5x$	5	2.25x-0.5	2
Unf. min. ASE's	4.4x 4.4x-1	(2.2x) x (4.4x-1) (2.2x) x (2.2x)	$9.68x^2-2.2x$ $4.84x^2$	2 5	3.3x-0.5 2.2x	1 2
Fol. min. crossp.	2x-1	$(\frac{9x^2-1}{2x-2}) \times (\frac{9x^2-1}{2x-2})$	$(\frac{9x^2-1}{2x-2})^2$	1	$\frac{9x^2-1}{2x-2}$	1
	$\frac{9x^2-1}{2x-2}$	(3x-1) x (3x-1)	$(3x-1)^2$	2	3x-1	1
Fol. equal ASE's	6.25x-2	(3.75x-1) x (3.75x-1)	$(3.75x-1)^2$	1	3.75x-1	1
Fol. min. ASE's	3x 3x-1	(4.5x-1) x (4.5x-1) (3x) x (3x)	$(4.5x-1)^2$ $9x^2$	1 2	4.5x-1 3x	1 1

The numbers of chips needed for every network are now calculated (Table 6) for both quadratically and linear increasing ASE costs. These numbers are not very accurate, due to roundings and approximations which had to be used to avoid the difficult formulas. So the difference between the numbers of chips needed for the folded networks and the numbers of chips needed for the unfolded networks might be smaller than the actual numbers in Table 6.

Table 6: Chipcount of compared Clos networks

Network	Chipcnt (quadr)	Chipcnt (lin)
Unf. min. crossp.	$\frac{3x-1}{2} + \frac{9x^2}{5(1.5x-1)} \approx 2.7x$	$\frac{3x-1}{1} + \frac{9x^2}{2(1.5x-1)} \approx 6x$
Unf. min. ASE's	$4.4x/2 + (4.4x-1)/5 \approx 3x$	$4.4x/1 + (4.4x-1)/2 \approx 6.6x$
Fol. min. crossp.	$\frac{2x-1}{1} + \frac{9x^2-1}{2(2x-2)} \approx 4.25x$	$\frac{2x-1}{1} + \frac{9x^2-1}{1(2x-2)} \approx 6.5x$
Fol. equal ASE's	$(6.25x-2)/1 \approx 6.25x$	$(6.25x-2)/1 \approx 6.25x$
Fol. min. ASE's	$3x/1 + (3x-1)/2 \approx 4.5x$	$3x/1 + (3x-1)/1 \approx 6x$

The conclusion of this comparison is:
When quadratically increasing ASE costs are assumed, the unfolded Clos networks need less chips than the folded networks. But the unfolded Clos networks need more than one kind of chip, so the folded Clos network with all equal ASE's (and thus all equal chips) is a better choice, since it only uses one kind of chip and chipdesign costs are much larger than the costs of producing some extra chips of one kind.
When linear increasing costs are assumed, all networks need about the same number of chips. Since the cost increase of the ASE's is more close to quadratically than to linear, the conclusion of the comparison using ASE size for complexity is that although the unfolded networks use less chips, the folded Clos network with all equal ASE's is preferred, because it has smaller chipdesign costs.

The paragraph will end with a comparison of the unfolded Clos network having a minimum number of crosspoints and the folded Clos network having all equal ASE's for an ASF size of 1000 x 1000. The comparison uses the number of crosspoints for ASE complexity (quadratically increasing costs).
The unfolded Clos network (y=22) is constructed of:

92 ASE's of size 22 x 43 [3 per chip]
43 ASE's of size 46 x 46 [1 per chip]

The actual ASF size is 22x46=1012 and it exists of 92/3 + 43/1 = 74 chips.
The folded Clos network (z=19) is constructed of:

93 ASE's of size 56 x 56 [1 per chip]

The actual ASF size is 56x19=1064 and it exists of 93/1 = 93 chips.
This example clearly shows that the actual numbers of chips needed (74 and 93) are much closer than the numbers in Table 6 (2.7x and 6.25x).

5.6 Summary and conclusions

This chapter discusses the best way to perform partitioning on certain switch designs. The switch designs are divided in three categories, depending on internal speed and complexity: Small ASE based switches, limited size ASE based switches and unlimited size ASE based switches.
Small ASE based switches are all based on Delta networks. The Batcher-Banyan network should be partitioned in a way that one part is one stage of (128) ASE's. The other small ASE based networks should be partitioned in a way that one part is a stage of larger (than 2x2) networks, e.g. four 16x16 Delta networks.
Limited size ASE's internally have a high (ASE size dependable) speed bus as a transfer medium or as access to the bufferspace, which is why slicing is preferred for these ASE's. It is shown that these ASE's have approximately quadratically increasing costs regarding size.
Unlimited size ASE's often use a matrix of slotted buses (at the same speed as the inlets). Since these ASE's have a large complexity, logical partitioning should be performed on them. These ASE's also have quadratically increasing costs.
Next the chapter discusses the MIN's the limited and unlimited size ASE's are used in. The Benes network is rearrangable non-blocking regarding connection and the Clos network is strictly non-blocking regarding connection. A new implementation aspect for these MIN's is introduced: The folding of a MIN. A folded MIN uses less ASE's, but more complex ones. The folded and unfolded networks are compared in two ways. The first comparison is based on the use of a basic building block to construct the networks. This comparison shows that the folded networks need less basic building blocks, but the basic building blocks are used very inefficient by the unfolded networks, which is why this comparison is not quite good. The second comparison is based on the assumption that ASE complexity fully depends on the number of crosspoints, which is not quite correct, but still better than the above described comparison. This second comparison shows that the unfolded networks need less chips. So of the Benes networks the unfolded with all equal ASE's is preferred. Of the Clos networks the folded with all equal ASE's is preferred over the less chips needing unfolded networks, because of the lower chip design costs.

5.7 Bibliography

[BENE65] Benes, V.E.
MATHEMATICAL THEORY OF CONNECTING NETWORKS AND TELEPHONE TRAFFIC
New York: Academic press, 1965.

[CHEN91] Chen, X.
A SURVEY OF MULTISTAGE INTERCONNECTION NETWORKS IN FAST PACKET SWITCHES
International journal of digital and analog communication systems
Vol 4, No 1 (jan-mar 1991), p. 33-59.

[DIRK90] Dirksen, M.J.G.
INTERCONNECTION ARCHITECTURES IN ATM: THE USE OF BATCHER BANYAN NETWORKS AS ATM SWITCHES
Eindhoven: Technische Universiteit Eindhoven, 1990
Graduation report nr. 5697

[TAKE91] Takeuchi, T. en H. Suzuki, T. Aramaki
SWITCH ARCHITECTURES AND TECHNOLOGIES FOR ASYNCHRONOUS TRANSFER MODE
IEICE Transactions
Vol E 74, No 4 (Apr 1991), p. 752-760.

[TOBA90] Tobagi, F.A.
FAST PACKET SWITCH ARCHITECTURES FOR BROADBAND INTEGRATED SERVICES DIGITAL NETWORKS
Proceedings of the IEEE
Vol 78, No 1 (jan 1990), p. 133-167.

[VRIE92] Vries, R.J.F. de
SWITCH ARCHITECTURES FOR THE ASYNCHRONOUS TRANSFER MODE
Enschede: Technische Universiteit Twente, Doctoral dissertation.
Leidschendam: PTT Research, 1992.

[ZEGU93] Zegura, E.W.
ARCHITECTURES FOR ATM SWITCHING SYSTEMS
IEEE Communications magazine
Vol 31, No 2 (feb 1993), p. 28-37.

6 Overview of ATM switches

In this chapter a representative overview of ATM switches is given. The switches have been categorised in three groups: ASE's, complete switches and Batcher-Banyan based switches. For every switch the operation and performance are discussed and a relation is made with the implementation aspects discussed in chapter 4.

6.1 ATM switching elements

This paragraph discusses several ASE's. Some of them are limited size ASE's and others are unlimited size ASE's.

6.1.1 The Knockout ASE

The Knockout ASE (Figure 32) is a simple modular ASE based on output queueing with exclusive buffering and was first introduced in 1987 by the ATT Bell laboratories. It is an unlimited size ASE and uses a matrix of slotted buses to transfer cells to the output modules (bus interfaces). These buses operate at the same speed as the inlets.

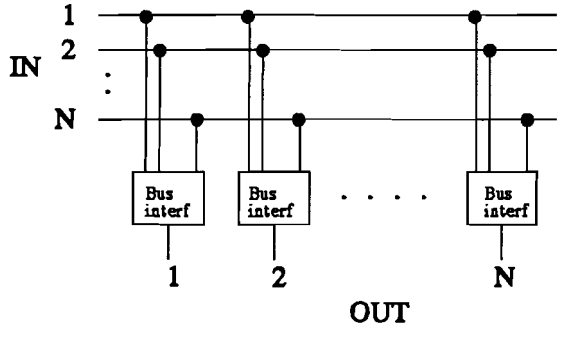

Figure 32: The Knockout ASE

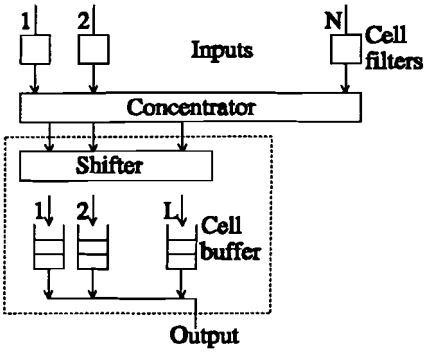

Figure 33: The businterface

All arriving ATM cells are transferred to all bus interfaces via the matrix of slotted buses. Inside the businterface (Figure 33), first the cell filters examine which cells are destined for the output belonging to this businterface. Next, the cells that are, enter the concentrator. The concentrator passes a maximum of L cells to the shifter, while the other cells are lost. The shifter equally distributes the cells over the L cell buffers. Finally, the cell buffers are served in a round-robin way by the outlets.
Broadcast capabilities are already present because of the use of the matrix of slotted buses. To handle multicast traffic, special multicast modules are needed. These multicast interfaces are almost the same as a bus interface. The only difference is, that at the outlet there's a cell duplicator, which makes the necessary number of copies of a cell and then places the cells back on the matrix of slotted buses.

Because the switch is based on output queueing, it would need a memory which can store N cells during one celltime, but to reduce this speed, a concentrator is used, which introduces a certain amount of cell loss. This means that besides queue dimensioning, also the concentration factor should be dimensioned. The memory now only needs to store L cells during one celltime (with L being the concentration factor). Calculations have shown, that for a concentration factor of 12 the cell loss probability for any switch size is smaller than 10^{-12}. The Knockout switch has the advantage of being modular. A disadvantage is that the Knockout switch uses routing tags, which gives speed adaptation problems. The ASE also requires cell synchronisation and uses a lot of hardware. Finally the switch is unfair, because the concentrator is unfair: The cells which enter the concentrator to the right are more likely to be discarded, while cells which enter the concentrator to the left are more likely to pass successfully.

6.1.2 The Gauss ASE

The Gauss ASE (Figure 34) is an ASS based on output and crosspoint queueing with exclusive buffering and was first introduced in 1990 by de Vries. Like the Knockout ASE, the Gauss ASE is an unlimited size ASE.

The input modules handle the bit synchronisation, the header control and translation and the adding of routing tags to the cells. The cells are transferred to all output modules via the matrix of slotted buses.

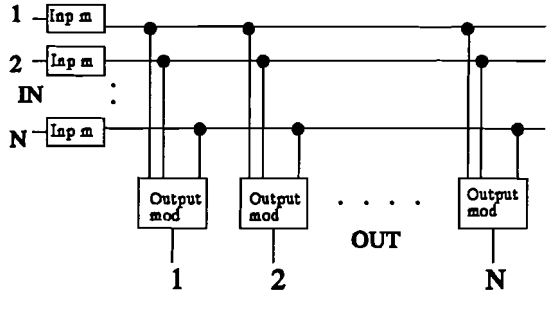

Figure 34: The Gauss ASE

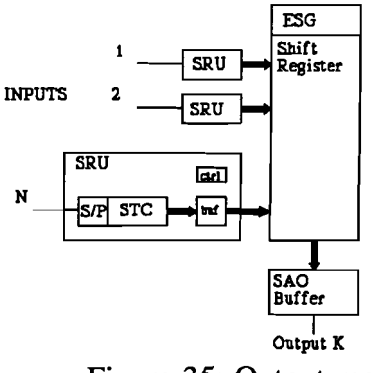

Figure 35: Output module

In the output modules (Figure 35), first the cells are translated from serial to parallel. Next, in the switching tag comparator is examined whether the cells are destined for the output belonging to the output module. Those cells are buffered. The Empty Slot Generator, places empty slots in the shift register, which can be filled with cells from the buffers. Finally the cells are placed on the outlet by the Speed Adaptation Buffer.
Like the Knockout switch, the Gauss ASE uses a matrix of slotted buses, which results in broadcast capabilities to be present. Multicast capabilities are easy to add to the Gauss switch. It can be done in the same way as used in the Knockout switch, but the cell copying can also be done in the input modules.

The ESG operates at a speed up of L. By keeping L smaller than the number of inlets, a certain amount of cell loss is introduced, but this way the speed is kept acceptable.
The Gauss ASE also has the disadvantage of being unfair, because the Shift Register Units close to the ESG are more likely to get an empty slot. This unfairness can be solved by placing a distribution network behind the input modules, but then the cell sequence integrity might be lost. Like the Knockout ASE, the Gauss ASE has the disadvantages of using routing tags and requiring a lot of hardware.

6.1.3 The Coprin ASE

The Coprin switch (Figure 36) is an ASE, first introduced by the French CNET in 1987 and also known as the Prelude switch. The ASE is based on output queueing with common buffering. Because of the limited memory access time which decreases when the ASE size increases, this is a limited size ASE.

The arriving cells are partitioned in parallel data streams of equal length by the supermultiplexer. This is done in a way that the first data stream covers the whole header. The headers are transferred to the control block, which handles routing and queue management in the common buffer. While the header is processed, the parallel data streams are buffered in the common memory. The output of the common buffer is still parallel data. Finally, the demultiplexer reassembles the cells out of the parallel data streams. Multicasting and broadcasting can easily be implemented in the control of the switch.

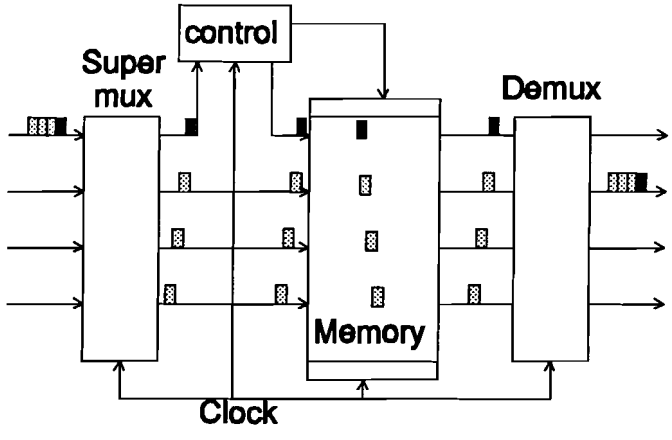

Figure 36: The Coprin ASE

The Coprin ASE has a lot of disadvantages:
The memory management is very complex because the cells are buffered in partitions.
In the Coprin ASE the headers of consecutive inlets should arrive in consecutive time slots, so the cells should be cell synchronized at the inputs. And finally, the number of inlets and outlets depends on the cell length.

6.1.4 Sigma ASE and Hitachi Shared Buffer Memory Switch ASE

The Sigma and Hitachi SBMS ASE's (Figure 37) are both first introduced in 1989 and are based on output queueing with shared buffering. Like the Coprin switch they are limited size ASE's, because of the limited memory access time.

The arriving cells are converted from serial to parallel and are latched in a register. The memory manager then takes care of the cells being buffered in the correct queue, and at the output the cells are latched in a register and then converted from parallel to serial.
The shared memory is logical partitioned in locations, which can contain a cell and a pointer. Actually, the queues are stored as linked lists in the shared memory, i.e. the pointer points at the next cell in the queue. The memory manager contains registers in which the head and the tail of every queue are stored. Multicasting can be implemented in the ASE's very easy because of the central control (memory manager).

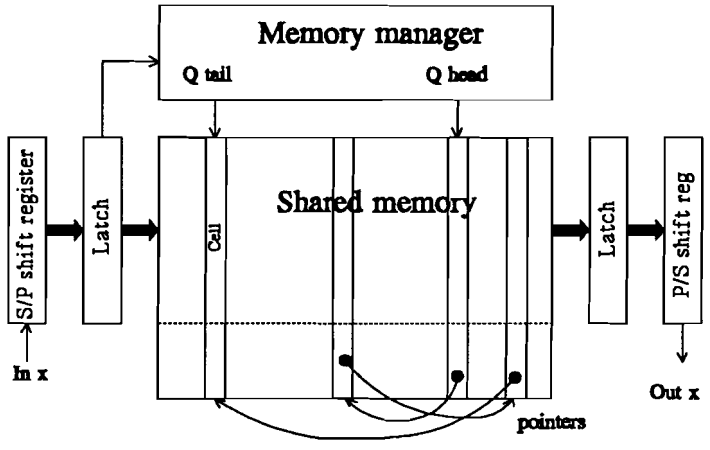

Figure 37: The Hitachi SMBS and Sigma ASE

The Sigma ASE and Hitachi SBMS ASE need a very reliable pointer management, to not lose a whole queue when a pointer is lost. This causes the memory manager and therefore the whole ASE to be very complex. An advantage is the use of routing tables, which makes routing very flexible.

6.1.5 The Atom ASE

The Atom ASE (Figure 38) is based on output queueing with exclusive buffering and was first introduced by NEC in 1989. This is a limited size ASE because of the limited bus speed, which increases with the ASE size.

The arriving cells are converted from serial to parallel and are then placed on a TDM-bus. Every output module contains a cell filter, which only passes cells destined for the output belonging to the output module. Next, the cells are buffered in a FIFO queue and when served, are translated from parallel to serial.

Broadcasting and multicasting are easy to implement because every cell is examined by all output modules.

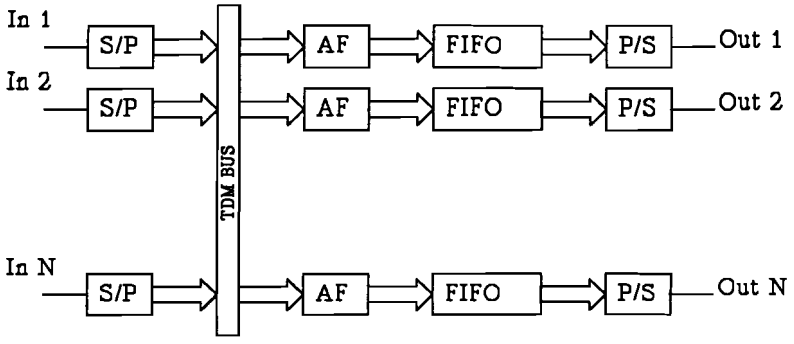

Figure 38: The Atom ASE

The use of a routing tag is a disadvantage but the switch has a very simple (FIFO) queue control.

6.2 Complete ATM switches

In this paragraph complete switches are discussed. This means both ASE and ASF designs are introduced.

6.2.1 The Athena switch

The Athena switch (Figure 39) was first introduced by de Prycker in 1987 and is based on output queueing with exclusive buffering. The ASE is a limited size ASE, because of the limited memory access time.

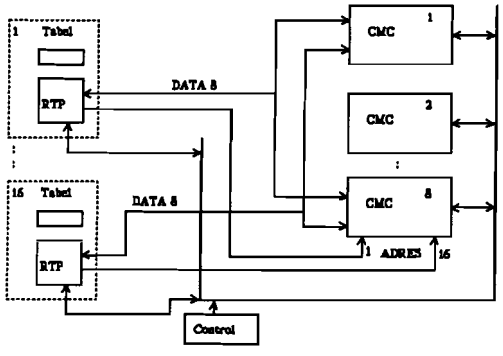

Figure 39: The Athena ASE

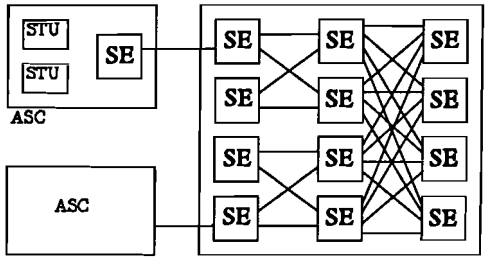

Figure 40: The Athena ASF

The cells arrive at transceiver ports, which perform the header check and the label translation. Then bitparallellisation (factor 8) is performed. Next every data stream is transferred to a Central Memory Chip. These CMC's contain the actual queues. Finally the cells are

transferred back to the transceiver ports and are then transmitted to the next stage of ASE's in the ASF.
The ASF (Figure 40) defined with the Athena ASE is a folded five stage MIN and uses type III routing (Table 1), which makes it easy to implement broadcast and multicast facilities.

The major advantage of the Athena switch is the simple control needed. There's however a lot of wiring needed. The use of routing type III causes the need for a large and fast memory to contain the tables and causes a certain probability on connection blocking but also causes the routing to be flexible.

6.2.2 The Roxanne switch

The Roxanne switch (Figure 41) was first introduced in 1990 by Alcatel and is based on output queueing with shared buffering. The ASE is basically the same as the Sigma ASE and therefore the Roxanne ASE is a limited size ASE because of limited memory access time.

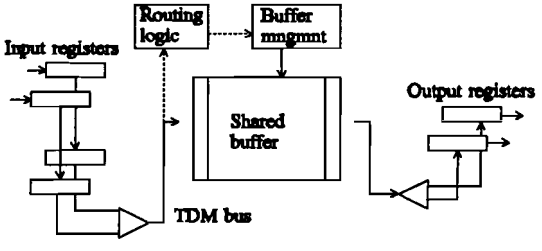

Figure 41: The Roxanne ASE

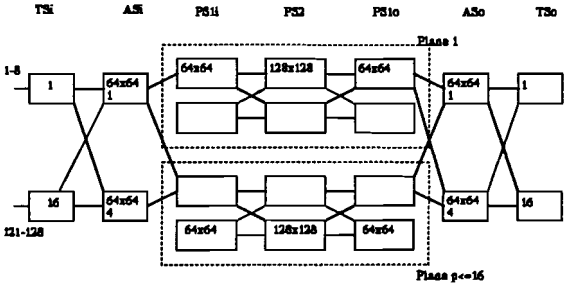

Figure 42: The Roxanne ASF

The arriving cells are converted from serial to parallel and are then latched in the input registers. Next, the cells are transferred to a shared buffer via a TDM-bus. At the same time the routing tag is processed by the routing logic. The buffer management controls and manages the logical queues in the shared buffer. The transmitted cells, leave the buffer via a TDM-bus and are then converted from parallel to serial in the output registers.
The ASF (Figure 42) defined with the Roxanne ASE belongs to the multi-path class, uses type II routing (Table 1) and is internally blocking. Multicasting in the Roxanne switch is done using routing type IV (Table 1).

Inside, the ASE uses Multi Slot Cells which makes the ASE very complex. Another disadvantage is the possible loss of cell sequence, so resequencing should be done at the outlets. Because of this resequencing 20% extra bufferspace is needed.
The Roxanne switch has the advantage that due to the traffic distribution, hardware failures only have a small effect and all traffic acts like geometric distributed traffic which results in a better performance. Disadvantages are the necessary cell synchronisation, the use of routing tags, the complex memory management and the possible loss of cell sequence.

6.2.3 The St. Louis switch

The St. Louis switch (Figure 43) was first introduced by Turner in 1986 and is based on distributed queueing. The switch was originally designed for switching variable length packets but is also suited for switching ATM cells.

All networks in the ASF are constructed of 2x2 ASE's (Figure 44) and are based on Delta networks. The input controller processes the routing tag and buffers the cells when the outgoing link is occupied. The output controller selects a cell from a buffer and transfers it to the next ASE. This is done using the earlier described backpressure mechanism.
The ASF belongs to the multi-path class and uses type II (Table 1) routing.
The Packet Processors (PP) attach a routing tag to the arriving cells. Then, the cells enter the Copy Network (CN) in which the necessary copies are made of the cells when handling multicast or broadcast traffic. Next, the Broadcast and Group Translators translate the tags of the copied cells to the correct ones before the cells enter the Distribution Network (DN). The distribution network distributes the cells as much as possible over the inlets of the Routing Network (RN). Finally the Routing Network switches the cells to the destined outlet.

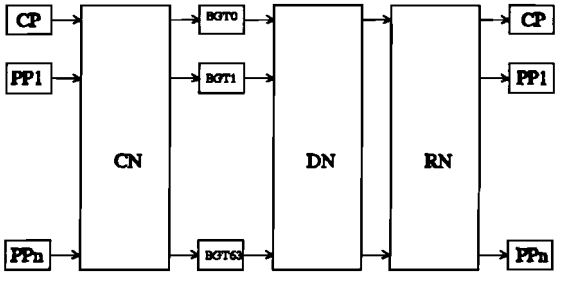

Figure 43: The St. Louis ASF

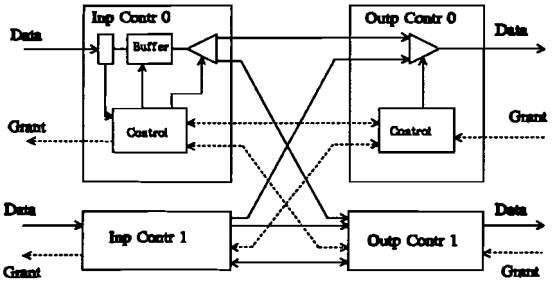

Figure 44: The St. Louis ASE

The St. Louis switch has one major disadvantage: Due to the backpressure mechanism cells can get lost at the inputs of the switch. The St. Louis switch also has the disadvantage of using routing tags. On the other hand, the switch is very modular and regular.

6.3 Batcher-Banyan based switches

The Batcher-Banyan network has been described in paragraph 5.1.1. It needs no internal buffering since there exists no contention internally. However there's still the possibility of two cells contending for the same outlet, so additional arbitration logic is needed. The advantage of switches based on the Batcher-Banyan network is it's easy to add multicasting or broadcasting facilities. This is done using a copy network like in the st. Louis switch. Another advantage is, these switches have a very regular and modular design. A disadvantage of all Batcher-Banyan based switches is the use of routing tags.

6.3.1 Moonshine switch

The Moonshine switch (Figure 45) is an ASF originally proposed by Hui in 1987 and is based on input queueing with exclusive buffering.

To avoid output contention, a three phase algorithm is introduced. The algorithm consists of the arbitration, the acknowledge and the data phase.
In the arbitration phase each input buffer containing a cell, sends a request packet, so the switch can determine whether there are contending cells waiting. When the winning requests are selected the waiting cells are informed during the acknowledge phase.
Finally, the cells who received a positive acknowledge, are sent during the data phase.

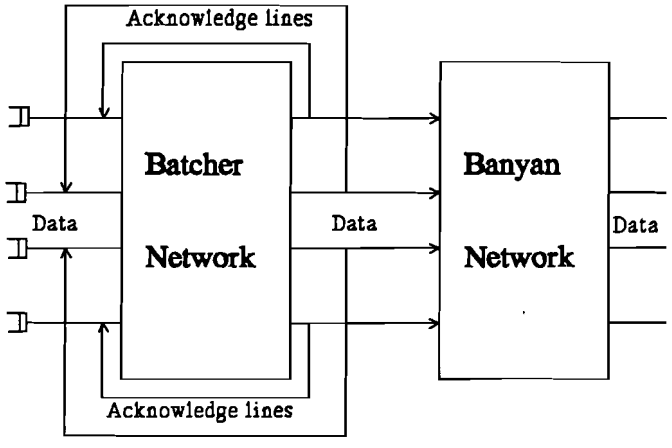

Figure 45: Moonshine switch

Because of the input queueing this switch suffers from HOL-blocking (paragraph 3.7).
Another disadvantage is the three-phase algorithm that has to be performed during one cell-time, so internal speed up is needed. This internal speed up is reduced by duplicating the Batcher network: Now, while cells are sent (data phase), the next cells in the input queues can start their arbitration and acknowledge phases. In this way, a partially pipelined switch is created.

6.3.2 Starlite switch

The Starlite switch (Figure 46) was introduced in 1984 by Huang and Knauer and is based on output queueing.
The output contention is overcome by a trap network between the Batcher and the Banyan network. The trap network detects cells destined for the same outlet and passes only one of them, the others are fed back to the entrance of the Batcher network. The recirculated cells then try again during the next cell time.

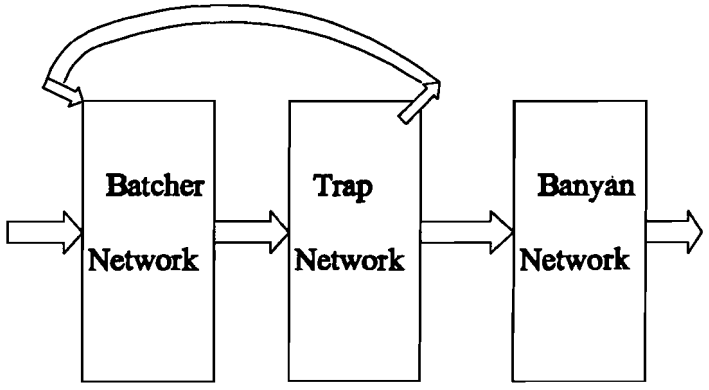

Figure 46: Starlite switch

The use of a recirculation network, besides introducing extra hardware, introduces a certain cell loss probability, because not all the cells are recirculated when more cells than the number of recirculation lines need to be recirculated. So besides dimensioning the bufferspace, also the number of recirculation lines should be dimensioned.

6.3.3 Sunshine switch

The Sunshine switch (Figure 47) was introduced by BellCoRe laboratories in 1990 and is based on output and crosspoint queueing.

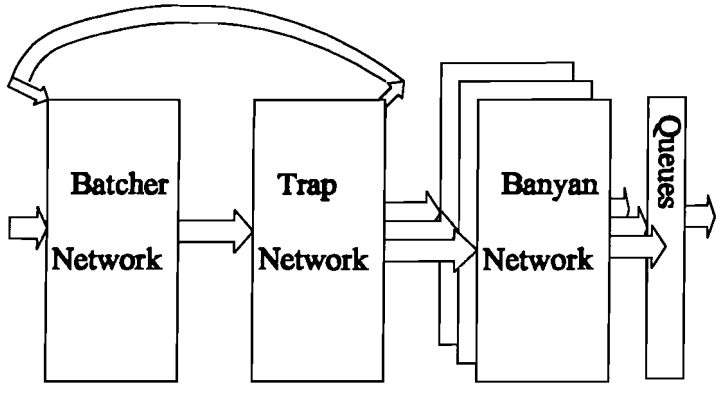

Figure 47: Sunshine switch

Actually, it's a Starlite switch with k parallel Banyan networks, instead of one. So up to k cells can be routed for the same outlet during one cell time. This means, output buffers are needed. If there are more than k cells contending for the same outlet, only k cells pass and the others are recirculated, just like in the Starlite switch.

Like the Starlite switch, the Sunshine switch needs accurate dimensioning of the number of recirculation lines. But, now also the number of parallel Banyan networks should be dimensioned. Overall this switch has a large hardware complexity.

6.4 Implementing the reviewed switches

Now, after a representative overview of known switches is given, the chapter will end with some remarks on the implementation of the reviewed switches. Bus and memory speeds in particular are studied and using this, some example calculations for an ASF size of 1000x1000 and a buffer access time of 10ns, are made at the end of the paragraph. Besides that, it is discussed which parallellisation and partitioning are preferred for each switch and using this, the switch classes to which each switch can belong are identified. The characters that are used in the formulas are p for parallellisation factor, N for ASE size and b for bitparallellisation factor (=p/cellparallellisation factor).

The Knockout switch as seen in paragraph 6.1.1 needs dimensioning of the concentration factor. As said before, if this factor (L) is 12 the cell loss probability is acceptable for all ASE sizes. In the output module are L buffers from which each maximally one cell has to be read and in which each one cell has to be written per celltime. Internally, the Knockout switch has no speed-up but the basic inlet speed of 155Mbit/s should be reduced by parallellisation. Parallellisation means a cell now becomes 424/b packets and speed is p times smaller, so the memory access time of 2 cells per celltime (without parallellisation) now becomes:

$$\frac{848}{b} \text{ packets per p celltimes} \Leftrightarrow t_{acc} = \frac{p \bullet b}{424 \bullet 155 \bullet 10^6} \text{ [sec]}$$

If the used memory is a dual ported memory (i.e. one write access and one read access per cycle), the necessary access time is doubled.
The results of the examples (Table 7) show, if only bitparallellisation is performed, the parallellisation factor needed is 40. The Knockout switch should be logical partitioned, because of the large hardware complexity, which makes the switch a wide switch. In addition to the logical partitioning also slicing could be performed (for instance 5 planes of 8 bits wide), which would make the switch partially a wide switch and partially a bit-sliced switch. When using both bit- and cellparallellisation and no slicing is performed, the switch is partially a wide and partially a dilated switch. If slicing is performed the switch is a combination of a replicated and a wide switch. The example calculations show that the basic inlet speed is reduced to less than 4Mbit/s which is implementable.

The Gauss switch as seen in paragraph 6.1.2, also needs dimensioning of the concentration factor. Like in the Knockout switch, the value of 12 is chosen to have an acceptable small cell loss probability. The buffer which needs the smallest access time is the SAO buffer, in which twelve cells might be written and from which one cell is read, per celltime (if no parallellisation is performed). Since parallellisation has to be performed, the SAO buffer

access time is:

$$t_{acc} = \frac{b \bullet p}{13 \bullet 424 \bullet 155 \bullet 10^6} \text{ [sec]}$$

If a dual ported memory is used, the t_{acc} is multiplied by 13/12. The Gauss switch belongs to the same combination of switch classes as the Knockout switch (depending on the kind of parallellisation and partitioning performed). Because of the speed-up of twelve more parallellisation is needed.
The examples show that the maximum speed of traffic is now under 20Mbit/s, which is implementable. When combining bitparallellisation and cellparallellisation, a bitparallellisation of eight would mean the total parallellisation should be 1072 (i.e. c=134), but to keep the total parallellisation acceptable (I used max. 424), the in Table 7 displayed values were used.

The Coprin, the Sigma, the Hitachi SBMS and the Roxanne ASE all use a common or a shared buffer and as access, TDM-buses (Figure 24 left). Every celltime N cells have to be written in the buffer and N cells are read from it (if no parallellisation is performed). Including the parallellisation, the memory access time is:

$$t_{acc} = \frac{p \bullet b}{2N \bullet 424 \bullet 155 \bullet 10^6} \text{ [sec]}$$

If a dual ported memory is used, t_{acc} is doubled. For an acceptable memory access time and N=50, the parallellisation is calculated in Table 7. The TDM-buses at the inlets and the outlets now have a speed smaller than 30Mbit/s. This is implementable but this TDM-bus speed should be on chip (i.e. within one part), so slicing is preferred. It is also preferred to have 8 or 16 bits wide parts, to be compatible with the standard memories, so these switches are a combination of bitsliced and wide switches (only bitparallellisation) or a combination of replicated and wide switches (bit- and cellparallellisation).

The Atom ASE and the Athena ASE use a TDM-bus as a switching transfer medium, in combination with exclusive buffering (Figure 24, right). Every celltime, one cell has to be read from the buffers and N might be written in it. The memory access time (including parallellisation) now is:

$$t_{acc} = \frac{p \bullet b}{(N+1) \bullet 424 \bullet 155 \bullet 10^6} \text{ [sec]}$$

If a dual ported memory is used, t_{acc} is multiplied by (N+1)/N. Table 7 shows the values for parallellisation for an acceptable memory access time. The resulting TDM-bus speed is under 43Mbit/s which is implementable on-chip, so slicing is preferred. Like the switches based on shared and common buffers, these switches are a combination of bit-sliced and wide switches or a combination of wide and replicated switches.

The remaining reviewed switches are all based on Banyan networks, although the st. Louis switch uses slightly different ASE's. As seen in chapter 5 Banyan based switches should be logical partitioned. Since the maximum speed off-chip is lower than the maximum speed on-chip, the speed should be reduced to approximately 25Mbit/s (instead of 50Mbit/s), which means a parallellisation of eight should be performed.
The moonshine switch uses FIFO input buffers, the Starlite switch uses for each recirculation line a FIFO buffer and the st. Louis switch a small (1 or 2 cells) FIFO buffer in each ASE. From each buffer one cell has to be read and in each buffer one cell has to be written per celltime. The memory access time of these buffers now becomes the same as for the buffers of the Knockout switch.
The Sunshine switch uses k Banyan networks parallel and output buffers. From every output buffer one cell has to be read and in them k cells might be written per celltime, so the memory access time becomes:

$$t_{acc} = \frac{p \bullet b}{(k+1) \bullet 424 \bullet 155 \bullet 10^6} \text{ [sec]}$$

Table 7 shows parallellisation values based on k=3.
The speeds which result from parallellisation, for all Banyan based switches, are implementable off-chip (between parts).

To get an idea on actual figures, some example calculations of parallellisation factors and traffic speed for a buffer access time of approximately 10ns (Table 7) are presented. The calculations are based on an ASF size of approximately 1000 x 1000. In paragraphs 5.4.2 and 5.5.2 it is shown that the matching ASE sizes for this ASF size are approximately 50 x 50. The calculations are further based on a basic inlet speed of 155Mbit/s.

Table 7: Example values for reviewed switches

Switch	Bitparallellisation		Bit- & cellparallellisation		
	b=p	s (Mbit/s)	b	p	s (Mbit/s)
Knockout	40	3.8	8	168	2.3
Gauss	96	19.4	21	420	4.4
Comm. buffer	264	29.4	182	364	21.3
TDM bus	184	42.1	82	410	18.9
Sunshine	52	3.0	8	336	0.5

Since standard memories are several bits wide it is preferred to perform some amount of bitparallellisation. This might be done in combination with cellparallellisation (which will

spread the traffic more evenly) or exclusively (which will increase the buffer access time t_{acc}, because of the factor p*b). This is why the calculations are done for performing only bitparallellisation and for performing a minimal amount of bitparallellisation (minimal 8, because of memory width) in combination with cellparallellisation. In latter case the overall parallellisation is kept under 424 to avoid too large parallellisation factors. In column *s* the reduced maximum speed in the switch after parallellisation is given. Columns *b* and *p* show the parallellisation factors. It is clearly shown that only for limited size ASE's the speed is still too large to implement between parts (29.4 and 42.1 Mbit/s).

6.5 Summary and Conclusions

This chapter, an overview of current ATM switch designs is given. Both limited size (Atom, Sigma and Coprin) and unlimited size ASE's (Knockout and Gauss) are reviewed, also complete (both ASE and ASF designs) switches (Roxanne, Athena and st. Louis) and Batcher-Banyan based switches (Moonshine, Starlite and Sunshine) are reviewed.
The chapter ends with some remarks on the actual parallellisation and partitioning, which have to be performed on these switches, to get implementable buffer access times and implementable maximum internal speeds.

6.6 Bibliography

[BANN91] Banniza, T.R. en G.J. Eilenberger, B. Pauwels, Y. Therasse
DESIGN AND TECHNOLOGY ASPECTS OF VLSI'S FOR ATM SWITCHES
IEEE Journal on selected areas in communications
Vol 9, No 8 (oct 1991), p. 1255-1264.

[CHEN91] Chen, X.
A SURVEY OF MULTISTAGE INTERCONNECTION NETWORKS IN FAST PACKET SWITCHES
International journal of digital and analog communication systems
Vol 4, No 1 (jan-mar 1991), p. 33-59.

[CORA91] Corazza, G. en C. Raffaelli
COMPLEXITY COMPARISON OF MULTISTAGE INTERCONNECTION NETWORKS FOR BROADBAND SWITCHING APPLICATION
Advanced computer technology, reliable systems and applications, proceedings 5th annual european computer conference CompEuro '91, p. 195-199
Bologna, Italy, 13-16 may 1991
Los Alamitos, CA, USA, IEEE Computer Society Press, 1991.

[DIRK90] Dirksen, M.J.G.
INTERCONNECTION ARCHITECTURES IN ATM: THE USE OF BATCHER BANYAN NETWORKS AS ATM SWITCHES
Eindhoven: Technische Universiteit Eindhoven, 1990
Graduation report nr. 5697

[DOOL92] Dool, prof. ir. F. van den
COMMUNICATIE CENTRALES EN NETWERKEN
Eindhoven: Technische Universiteit Eindhoven, 1992
Syllabus (from 5P460)

[LIST89] Listanti, M en A. Roveri
SWITCHING STRUCTURES FOR ATM
Computer communications
Vol 12, No 6 (dec 1989), p. 349-358.

[MELE92] Melen, R.
CURRENT ARCHITECTURES FOR ATM IMPLEMENTATION
European Transactions on Telecommunications & Related Technology
Vol 3, No 2 (mar-apr 1992), p. 145-155.

[OIE91] Oie, Y. en T. Suda, M. Murata, H. Miyahara
SURVEY OF SWITCHING TECHNIQUES IN HIGH SPEED NETWORKS AND THEIR PERFORMANCE
International journal of satellite communications
Vol 9, No 5 (sep-oct 1991), p. 285-303.

[PRYC91] Prycker, M. de
ASYNCHRONOUS TRANSFER MODE: SOLUTION FOR BROADBAND ISDN
New York: Ellis horwood, 1991.

[RATH89] Rathgeb, E.P. en T.H. Theimer
ATM SWITCHES - BASIC ARCHITECTURES AND THEIR PERFORMANCE
International journal of digital and analog cabled systems
Vol 2, No 4 (oct-dec 1989), p. 227-336.

[TAKE91] Takeuchi, T. en H. Suzuki, T. Aramaki
SWITCH ARCHITECTURES AND TECHNOLOGIES FOR ASYNCHRONOUS TRANSFER MODE
IEICE Transactions
Vol E 74, No 4 (Apr 1991), p. 752-760.

[TOBA90] Tobagi, F.A.
FAST PACKET SWITCH ARCHITECTURES FOR BROADBAND INTEGRATED SERVICES DIGITAL NETWORKS
Proceedings of the IEEE
Vol 78, No 1 (jan 1990), p. 133-167.

[VRIE92] Vries, R.J.F. de
SWITCH ARCHITECTURES FOR THE ASYNCHRONOUS TRANSFER MODE
Enschede: Technische Universiteit Twente, Doctoral dissertation.
Leidschendam: PTT Research, 1992.

[WULL89] Wulleman, R. and T. van Landegem
COMPARISON OF ATM SWITCHING ARCHITECTURES
International Journal of digital and analog cabled systems
Vol 2, No 4 (oct-dec 1989), p. 211-225.

7 Conclusions

The study presented in this report intended to find modifications on ATM switch designs in order to avoid limitations in VLSI technology. The most important limitations of VLSI technology are the maximum operation speed and the amount of integration possible.

To reduce the speed in an ATM switch design, parallellisation should be performed.
Bitparallellisation has the advantages of having a small speed adaptation buffer delay and guaranteeing preservation of cell sequence integrity. For all switches with queueing, more bitparallellisation means a larger buffer access time.
Cellparallellisation results in more evenly spread traffic, but cellparallellisation also means possible loss of cell sequence integrity, and for switches with queueing a smaller buffer access time (or more parallellisation).
The example calculations of chapter 6 show that for switches with queueing, the necessary amount of parallellisation needed is determined by the buffer access time instead of the highest internal speed of traffic.

Because switch designs are too large to put on one chip and to make the designs modular, the ATM switch designs should be partitioned. Slicing results in equal parts and also keeps high speed links within one part, while logical partitioning always has to be performed on the MIN of large switches, because slicing on its own is not sufficient.
Based on these implementation aspects, a classification is made. Actually switches will often belong to a combination of these classes, because bit- and cellparallellisation often are used in combination and slicing and logical partitioning are also often used in combination.

Which partitioning should be performed and how it should be performed, depends on the kind of switch design:
Batcher-Banyan networks should be logical partitioned in a way that one part is one stage of (128) ASE's. All other small ASE based networks should be partitioned in a way that one part is a stage of larger (than 2x2) networks.
Limited size ASE's should be partitioned themselves. Because of the high (size dependable) internal speed, they should be sliced. Unlimited size ASE's should also be partitioned themselves. They should be logical partitioned, because of their large hardware complexity.
The MIN's in which the unlimited size and limited size ASE's are used, are Benes and Clos networks. These can be 'folded' which results in less but more complex ASE's. The major disadvantage of the Clos and Benes networks is that only the unfolded networks can grow and only to sizes which are powers of the original MIN size.
For the Benes networks, the unfolded networks are preferred, because they use a little less chips than the folded networks. From the unfolded networks, the one with all equal ASE's is the best choice, because it only uses one kind of chip.
The unfolded Clos networks use less chips than the folded Clos network, but they use two kinds of chips, while the folded might only use one kind of chip. So the folded Clos network with all equal ASE's is preferred, because of lower chipdesign costs.

Further research should be done on the advantages of having more evenly spread traffic, when cellparallellisation is performed: How much are the queueing delays reduced? Does this compensate the larger speed adaptation buffer delay and the need for more parallellisation, because of the smaller queueing buffer access time?