(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
6 April 2006 (06.04.2006)

PCT

(10) International Publication Number
WO 2006/035202 A1

(51) International Patent Classification:
H04Q 3/68 (2006.01) *H04L 12/56* (2006.01)
H04Q 11/06 (2006.01)

(21) International Application Number:
PCT/GB2005/003671

(22) International Filing Date:
23 September 2005 (23.09.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
0421752.7 30 September 2004 (30.09.2004) GB

(71) Applicant *(for all designated States except US)*: **BRITISH TELECOMMUNICATIONS PUBLIC LIMITED COMPANY** [GB/GB]; 81 NEWGATE STREET, LONDON Greater London EC1A 7AJ (GB).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **HILL, Alan, Michael** [GB/GB]; The Old Police House, Park Road, Grundisburgh, Woodbridge Suffolk IP13 6TP (GB). **RAFEL, Albert** [ES/GB]; 19 Sadlers Place, Martlesham Heath, Ipswich Suffolk IP5 3SS (GB). **HODGKINSON, Terence Geoffrey** [GB/GB]; 46 Melton Grange Road, Melton, Woodbridge Suffolk IP12 1SD (GB).

(74) Agent: **LOFTING, Coreena, Fiona, Anne**; BT GROUP LEGAL INTELLECTUAL PROPERTY DEPARTMENT, PPC5A, BT CENTRE, 81 NEWGATE STREET, LONDON Greater London EC1A 7AJ (GB).

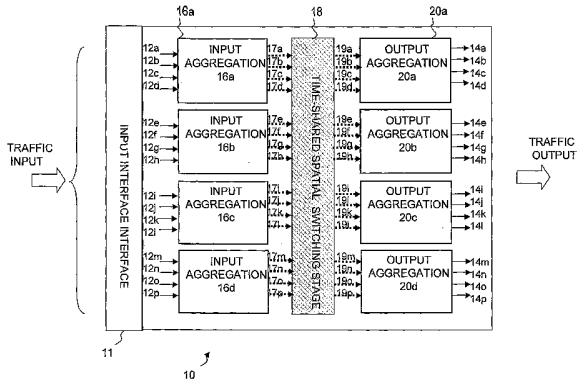
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) **Title:** CHANNEL ASSIGNMENT FOR A MULTI-STAGE SWITCH ARRANGEMENT



(57) **Abstract:** A constrained channel assignment process for a multi-stage switch arrangement for switching traffic segmented into a plurality of time-slots, the multi-stage switch arrangement comprising a plurality of input aggregations, each input aggregation comprising a separate subset of the plurality of inputs of the switch arrangement, said subset of inputs being logically associated with at least one aggregation switching stage of the switch arrangement arranged to operate only on traffic derived from said subset of inputs, a plurality of output aggregations, each output aggregation comprising a separate sub-set of the outputs of the switch arrangement; and a global spatial switching stage arranged to receive time-slots from any of the plurality of input aggregations of the switch arrangement and arranged to provide time-slots to any of the plurality of output aggregations, wherein the channel assignment process comprises the step of: identifying traffic segments which can be switched together as a logical switchable entity across the global switching stage of the switch arrangement; and imposing a constraint on the manner in which each logical switchable entity comprising a plurality of timeslots carrying identified traffic segments are assigned from switches in an aggregation switching stage in each input aggregation logically adjacent to a global spatial switching stage to the outputs of the switch arrangement.

WO 2006/035202 A1

CHANNEL ASSIGNMENT FOR A MULTI-STAGE SWITCH ARRANGEMENT

This invention relates to a channel assignment scheme for a multi-stage switch arrangement and related aspects.

Switch arrangements in which traffic segmented into discrete time-slots is transported from one side of the switch arrangement to the other side of the switch arrangement require processes to control contention resolution between traffic arriving at the inputs for the switch arrangement's outputs and also to determine a path or channel through the physical and logical architecture of the switch arrangement. Channel assignment schemes assigning paths through the physical and logical architecture of the switch arrangement are already known to those of ordinary skill in the art. However it is desirable for the channel assignment process to be computationally as efficient as possible, as it can affect the efficiency (in terms of capacity and actual throughput) of the switch arrangement.

The channel assignment process for a switch arrangement according to the invention seeks to obviate and/or mitigate the limitations of known channel assignment schemes by using a logical representation of the switch arrangement which reduces the computational complexity, and hence the computing time, of the channel assignment process.

SUMMARY STATEMENTS OF INVENTION

The aspects of the invention and preferred features are as set out in the appended independent and dependent claims respectively.

Where appropriate, one or more of the preferred features of the invention can be combined with one or more of any of the aspects of the invention using any suitable technique apparent to those skilled in the art.

Specific embodiments of the invention will now be described with reference to the accompanying figures which are by way of example only and in which:

Figure 1 shows schematically a representation of a switch arrangement according to the invention for which a channel assignment scheme may be implemented according to the invention;

Figure 2 shows schematically the internal structure of an input aggregation for the switch arrangement of Figure 1;

Figure 3 shows schematically the internal structure of an output aggregation for the switch arrangement of Figure 1;

Figure 4 shows an expanded view of the switch arrangement of Figure 1;

Figure 5 shows schematically a multi-slot;

Figure 6 shows how various switch arrangements may be schematically represented according to the number and type of switching stages logically associated with subgroups of the inputs and outputs of the switch arrangement;

Figure 7 shows steps in a constrained channel assignment process for the switch arrangement shown in Figure 1;

Figure 8 shows an example of how traffic segments might be assigned to inputs and outputs of the topmost input aggregation and topmost output aggregation shown in Figure 4;

Figure 9 shows an equivalent Benes network for assigning global spatial switches to multi-slots according to one embodiment of the invention.

Figure 10 shows the assignment of multi-slots to the time-domain switches between an input aggregation and an output aggregation;

Figure 11 shows the assignment of the individual time-slotted traffic segments to the outputs of an input aggregation and to the inputs of an output aggregation of the switch arrangement shown in Figures 1 and 4;

Figure 12 shows schematically how time-slots are reordered in one embodiment of the invention;

Figure 13 shows schematically how additional switching hardware may be provided in a switch arrangement according to another embodiment of the invention;

Figure 14 shows one embodiment of a spatial switch for a switch arrangement according to the invention;

Figure 15 shows a five stage (TST)S(S) switch arrangement according to one embodiment of the invention;

Figure 16 shows an alternative final (S) spatial aggregation switching stage embodiment to that shown in Figure 15 for a (TST)S(S) switch arrangement;

Figure 17 shows a (TST)S switch arrangement according to another embodiment of the invention; and

Figures 18 and 19 show two other embodiments of a five-stage switch arrangement according to the invention.

The best mode of the invention as currently contemplated by the inventors will now be described with reference to the accompanying drawings. Those skilled in the art will recognise that the detailed description provided of specific embodiments of the invention below is not intended to represent the only means of implementing the invention, and that any equivalent suitable means which is apparent to a person of ordinary skill in the art may instead be used.

Embodiments of the invention, each providing a channel assignment process, are described herein in the context of a computational model for a multi-stage switch arrangement. The multi-stage switch is capable of being decomposed into a plurality of time and spatial switching stages, at least one decomposition producing a time-space-time (T-S-T) series arrangement of switching stages. The computational model models the physical or logical structure of the switch arrangement for which the scheduling process is being performed. An appropriate control means to enable the switch arrangement to then implement channel assignment accordingly is an implicit feature of all switch arrangements, and can be implemented in any means well known to those of ordinary skill in the art.

The channel assignment process according to the invention may be used to assign channels to any suitably segmented traffic which is to be switched through a real (i.e., physical) switch arrangement. The traffic to be switched is time-slotted into

segments (for example, TDM channels, cells or packets) which are capable of being assigned a unique channel according to the channel assignment process. The term timeslot refers herein to a separately switchable segment of traffic. Any necessary conversion from variable length time-slotted traffic segments to fixed time-slotted traffic segment-length traffic (such as from a packet to a cell) is assumed to have been performed, for example, within appropriate switch interfaces.

Figure 1 of the accompanying drawings shows schematically a logical representation 10 of an N x N (here 16 x 16) multi-stage switch arrangement (comprising 16 inputs and 16 outputs). Henceforth the term switch arrangement will refer to the logical representation 10 of the switch arrangement unless the context indicates it refers to the actual switch arrangement.

In Figure 1, the top-level logical architecture of the multi-stage switch arrangement comprises just three logical switching structures. Firstly, an array of input aggregations 16a...d, each input aggregation 16a...d capable of receiving time-slotted traffic segments from just a subset of the inputs 12a...p of the switch arrangement. Secondly, a global spatial switching stage 18 comprising an array of time-shared spatial switches (not shown) arranged to receive inputs 17a...p from any of the input aggregations 16a...d. Finally, an array of output aggregations 20a...d, each output aggregation 20a...d capable of providing output to only a subset of the outputs 14a...p of the switch arrangement, and each arranged to receive inputs from just a sub-set of the outputs of the spatial switching stage 18, for example 19a...d in the case of output aggregation 20a.

Each input aggregation 16a...d comprises a logical association of four of the inputs 12a...p of the switch arrangement together with at least one switching stage which operates only on traffic derived from the four logically associated inputs. Each of the output aggregations 20a...d comprises a logical association of four of the outputs 14a...p of the switch arrangement together with at least one switching stage which operates only on traffic derived from a sub-set of just four of the inputs provided by the global spatial switching stage 18.

The global spatial switching stage 18 provided for the switch arrangement 10 is arranged to receive traffic along inputs 17a...p derived from any of the input aggregations 16a...d (and therefore from any of the inputs 12a...p of the switch arrangement). Spatial switching stage 18 is arranged to output traffic via outputs

19a...p to any of the outputs 14a...p of the switch arrangement. Also shown in Figure 1 is an input switch interface 11 both this and any other interface provided at the output of the switch arrangement (not shown in Figure 1) are optional in some embodiments of the invention, for example, where virtual output queue buffering is provided using random access memory (RAM) instead of first-in-first out memory (FIFO)

Traffic arriving at the switch arrangement is processed to determine which specific traffic segments should be inputted via the timeslots associated with the inputs 12a,...,p to the switch arrangement 10. Such processing may involve a preliminary matching process to ensure that any contention between incoming traffic segments destined for the same output as well as any contention between incoming traffic segments to depart from the same input are removed and to attempt to maximise the throughput of the switch arrangement. The channel assignment process according to the invention then determines a path for each traffic segment across the switch arrangement from its input to its scheduled output. This path involves the assignment of at least one time slot in tandem to each traffic segment as it passes through the switch arrangement.

In a preferred embodiment of the input and output aggregations 16, 20 comprise a plurality of switching stages. For example, a seven-stage switch arrangement can be provided if each of the input and output aggregations 16,20 internally implements three switching stages, referred to herein as "aggregation switching stages". The aggregation switching stages are arranged to be interconnected in series. Collectively, the series of interconnected aggregationswitching stages is referred to herein as an input/output (switching) aggregation (as appropriate) or equivalently as an aggregation element. The aggregation switching stages within each aggregation element operate only internally on traffic derived from or destined to the logically associated inputs/outputs forming the respective input/output aggregation element.

Figures 2 and 3 respectively show the internal structure of an input and output aggregation 16a, 20a respectively according to an embodiment of the invention in which the logical representation 10 of the switch arrangement represents a series of seven switching operations.

Figure 2 shows an input aggregation 16a comprising three internal aggregation switching stages 26,28, 30 which are arranged to operate only on timeslots inputted

via inputs 12a...d. In this instance, " aggregation switching stage" 26 operates only on timeslots received from a sub-set (here inputs 12a...d) of the total inputs 12a...p of the switch arrangement 10 shown in Figure 1. In Figure 3, described in more detail hereinbelow, aggregation switching stage 34 operates only on timeslots received from a sub-set (for example, 19a...d as shown in Figure 3) of the total inputs 19a..p from the global spatial switching stage 18 of switch arrangement 10.

Along each individual input 12a...p to an input aggregation 16a...d of the switch arrangement 10, a frame comprising here four time-slots at a time is provided as input to each switch 26a...d implementing the first aggregation switching stage 26. Thus for the input aggregation 16a time-slotted traffic segments (#A,#B,#C....#P) are provided. The inputs to each aggregation 16b ... d are similarly provided on a frame by frame basis of four time-slots at a time.

The other three input aggregations of the switch arrangement shown in Figure 1 are assumed to have an identical structure to input aggregation 16a. Internally, within each of the input aggregations 16a...d, a spatial aggregation switching stage 28 is provided to receive time-slots on a frame by frame basis from a time-domain aggregation switching stage 26, and to provide output to a time-domain aggregation switching stage 30 which is logically adjacent to the spatial (central) switching stage 18 of the (logical representation of the) switch arrangement.

Similarly in Figure 3, only output aggregation 20a is described but the other output aggregations in the switch arrangement are assumed to have an identical structure. In output aggregation 20a, time-domain aggregation switching stage 34 is logically adjacent to the spatial switching stage 18. Time-domain aggregation switching stage 34 in output aggregation 20a therefor receives input 19a..d from the global spatial switching stage 18 and provides output to spatial aggregation switching stage 36 which in turn provides output to time-domain aggregation switching stage 38 from which the outputs 14a..d of the output aggregation 20a of the switch arrangement are derived. The number of switches in each the aggregation switching stages of the output aggregations is equal to the number provided in each of the input aggregations in this embodiment of the invention.

The full logical seven-stage switch arrangement 10 comprising (TST)S(TST) switching stages is partially shown in Figure 4. Here "T" denotes switching in the time-domain and "S" denotes spatial switching and the brackets() denote aggregation

switching stages.

The inventors have determined that if a switch arrangement can be represented by certain logical arrangements of switching stages (which may or may not be physically present), the computational complexity of the channel assignment process for the switch arrangement can be made more efficient.

The best mode of the channel assignment process as currently contemplated by the inventors utilises a seven stage logical representation of a switch arrangement. However, as those skilled in the art would appreciate, in other embodiments of the invention, some examples of which are described later herein below, the switch arrangement may comprise a number of switching stages arranged in hierarchies of aggregations (for example, each logical (TST) input/output aggregation may be transformed logically into a (TST)S(TST) representation). The actual number of switching stages in the physical switch does not need to correspond to the number of switching stages in the logical representation of the switch arrangement.

The number of switching stages which a switch arrangement needs to implement in practice depends on many factors which are well known to those skilled in the art. For example, if Virtual Output Queuing (VOQing) is implemented on time-segmentable (e.g., cell-based or packet-based) traffic arriving at the interface 11, then the traffic segments input at ports 12a,...,p to the initial time-domain aggregation switching stage 26 will be firstly selected in a matching operation. Once matched, they can be assigned in the correct sequence to the four time-slots available in each frame for inputs 12a, b, c, d (e.g., timeslots #A....#D in ports 12a)The time-slots in each frames can be time-switched by time-domain aggregation switching stage 26, then spatially switched by spatial aggregation switching stage 28.

The inventors realise that traffic segments which have been assigned to the internal timeslots within the aggregation element for being spatially switched by spatial aggregation switching stage 28, have been assigned to timeslots which represent the timeslots in which the traffic segments must depart from their Virtual Output Queues (VOQs). This then renders the initial time-domain aggregation switching stage 26 shown in Figure 2 of the logical representation of the switch arrangement physically (though not logically), redundant, if the Virtual Output Queues are physically located in the interface 11 and they forward their traffic segments in the assigned internal timeslots.

Thus a choice exists, either the virtual output queues are placed in the interface 11 as in the above embodiment or they can be represented in the logical representation by the initial time-domain aggregation switching stage 26. The inventor prefers the latter as it clarifies that the VOQs are performing a time-switching function and makes it clear that the time-domain aggregation switching stage 26 is neither logically nor physically redundant.

If contiguity of timeslots at the outputs 14a...p of the switch arrangement is not required, it is also possible to omit the time-domain aggregation switching stage 38. In this case, the individual traffic segments must be reassigned to the internal timeslots at the outputs of the initial time-domain aggregation switching stage 26 to ensure there is no mis-sequencing of traffic segments (particularly cell-based or packet-based) coming from the same Virtual Output Queue.

Those skilled in the art will also appreciate that the number of inputs assigned to each aggregation is determined by and may vary according to each specific switch arrangement architecture for which the channel assignment process of the invention is to be implemented, similarly the frame-size may vary in different embodiments of the invention.

In the embodiments described herein the number of input aggregations matches the number of inputs in each input aggregation, and the switch is symmetrically arranged on its output side in terms of the number of outputs matching the number of output aggregations. However those skilled in the art will appreciate such a simple structure helps to clarify the concepts involved with the channel assignment process and is in no way limiting to the actual arrangements (logical or physical) which may be implemented in practice.

The channel assignment process does, however, require that at least one time-domain aggregation switching stage (30, 34) is retained logically adjacent to the spatial switching stage 18 of the switch arrangement 10 in order to implement the constraint which will be described later herein. This constrained channel assignment process can be implemented in practice therefore for any appropriate switch arrangement having an appropriate arrangement of aggregation stages of the form (...T)S(....) or (....)S(T...).

An individual switch in each aggregation switching stage may comprise a node, terminal or line-card as is appropriate to the particular configuration of switch arrangement. The number of switches in each aggregation switching stage is determined by the frame-size and desired optimum capacity of the switch. The number and nature of each aggregation switching stage is determined by many factors, including the nature of the channel assignment process and how it is constrained.

CONSTRAINED CHANNEL ASSISGNMENT

The invention imposes a constraint on the channel assignment process. To impose the constraint it is necessary to firstly identify if a plurality of timeslots carrying traffic through an input aggregation carry time-slotted traffic segments which can be switched as a single logical entity across the global spatial switching stage 18. The inventor uses the term "multi-slot" to denote the logical entity comprising a plurality of timeslots which are capable of being switched as a logical entity across the spatial switching stage of the switch arrangement. The actual end-to-end path of an individual traffic segment (e.g. a cell or packet) comprises different timeslots in different part of the switch arrangement.

The channel assignment process according to the invention first determines the number of potential multi-slots. The occupancy of each multi-slot is preferably maximised by selecting appropriate time-slots from the potential time-slots from each input aggregation. The multi-slots in each input aggregation are then matched from one or more available time-domain aggregation switches in time-domain switching stage 30 (which is logically adjacent to the input side of the global spatial switching stage 18) to one or more available time-domain aggregation switches in the time-domain switching stage 34 (which is logically adjacent to the output side of the global spatial switching stage 18).

The number of time-slots which are identified as containing time-slotted traffic segments which can be switched together as a single logical switching entity (i.e., as a "multi-slot" of size M, where M is the number of time-slots comprising the multi-slot) can be larger than F the frame-size of each single time-domain aggregation switch 30a...d, smaller than F, or equal to F depending on the specific embodiment of the invention.

Those skilled in the art will appreciate that in embodiments of the invention in which M = F, in each frame, each time-domain aggregation switch 30a...p of the time-domain aggregation switching stage 30 is constrained so that at all times the timeslots which it transmits are sent to the same switch 34a...p by the same spatial switch in the global spatial switching stage 18 to which the time domain switch 30a...p is initially connected. This results in pairs of switches in each of the time-domain aggregation stages 30, 34 logically adjacent to the spatial switching stage 18 being effectively "connected" back to back to each other. The result is that one of these switches becomes redundant effectively enabling it to be omitted in any physical implementation of the switch arrangement. Moreover, the spatial switching stage 18 can be implemented with switches that switch at the frame rate and not the time-slot rate.

Those skilled in the art will further realise that in embodiments of the invention in which M>F, two or more switches in aggregation switching stage 30 of an input aggregation are constrained to switch each multi-slot to the same number of switches in aggregation switching stage 34. In some embodiments of the invention, the plurality of switches receiving the multi-slot will comprise part of a single aggregation switching stage 34 (i.e., the multi-slot will be switched from a single input aggregation to a single output aggregation). However, in other embodiments of the invention, the plurality of switches receiving the multi-slot may be implemented in aggregation switching stages 34 in more than one output aggregation element.

Finally, if M < F, each switch in aggregation switching stage 30 of an input aggregation (16a, for example), is capable of switching more than one multi-slot, and thus each time-domain switch 30a..p in such embodiments are constrained so that they can only to switch time-slots to the same one or more switches in the aggregation switching stage 34, which may or may not comprise part of the time-domain aggregation switching stage 34 of the same output aggregation depending on the specific embodiment of the invention.

Referring now to Figure 4 in more detail, the embodiment shown comprises an N_1L_1 x N_2L_2 logical representation of an P x Q switch arrangement having P inputs and Q outputs. In the logical representation 10 of the switch arrangement the P inputs are arranged as N_1 logical associations of L_1 inputs, and the Q outputs are arranged as N_2 logical associations of L_2 outputs. In this embodiment $N_1 = N_2 = N = 4$ and $L_1 = L_2 =$

stage 18 is arranged to receive traffic from any of the N_1 logical association of inputs and to transmit traffic to any of the N_2 logical associations of outputs.

In the embodiment of the invention shown in Figure 4, each logical association of inputs further associated with a plurality of switches arranged to only receive traffic from the inputs forming that logical association form an input aggregation. Similarly, each logical association of outputs further associated with a plurality of switches arranged to provide traffic only to the outputs forming that logical association form an output aggregation as described above in the context of Figures 1 to 3. In Figure 4, only input aggregations 16a, 16b and output aggregations 20a, 20b are shown.

In this embodiment of the invention, each time-domain aggregation switching stage 26, 30, 34, 38 comprises L switches each arranged to switch F time-slots in each frame. Each spatial aggregation switching stage 28,36 comprises spatial switches which are shared over the F timeslots in each frame.

The number of spatial switches in the global spatial switching stage 18 logical representation is equal to the number N of input or output aggregations of the switch arrangement. Each spatial switch 18a...N is arranged to perform switching at the frame-rate of the switch arrangement.

Within each input/output aggregation 16, 20 each aggregation switch 26a...d, 28a...d, 30a...d, 34a...d, 36a...d, and 38a....d receives four time-slotted traffic segments (each of which occupies a single time-slot) so that switching is implemented over a frame length F of 4 time-slots (i.e., four time-slotted traffic segments are switched at a time), as was shown in more detail in Figures 2 and 3 described above.

Thus for the switch arrangement, in each frame duration the time-domain aggregation switching stages 26 of all of the input aggregation elements output LN sets of four time-slotted traffic segments to the N spatial switches forming spatial aggregation switching stage 28. The N spatial switches are thus time-shared over F time-slots, i.e., they switch four time-slotted traffic segments, each occupying a single time-slot, between each of their input and output ports in series.

A person skilled in the art would appreciate that, in specific embodiments, the number of time slots used within the input and output aggregations 16 and 20, and the number used within the global spatial switching stage 18, may differ from F and

from each other, depending on the precise nature of any potential blocking due to the matching and path-searching algorithms employed, e.g. strictly non-blocking or rearrangeably non-blocking.

By aggregating the switch arrangement inputs 12a,...,p into four input aggregations 16a,b,c,d, the time-slots associated with each of the input aggregations are able to share the frames output (along 17) from the input aggregation 16a..d to spatial switching stage 18. Thus whilst each frame received by an individual time-domain switch 26a...d forming aggregation switching stage 26 comprises four timeslots associated with a single input, e.g., 12a, each frame received by a spatial switch 28a...d forming the spatial aggregation switching stage in input aggregation 16a may comprise timeslots from any one of the inputs 12a...12d contributing to the input aggregation. Accordingly, each frame switched by the spatial switching stage 18 of the switch arrangement comprises effectively an aggregation of channels, each channel providing a path for a particular time-slot across the switch arrangement. In Figure 4, the global spatial switching stage 18 of the switch arrangement 10 is shown connecting individual switches in ingress aggregations 16a,b,c,d to individual switches in egress aggregations 20a,b,c,d.

The N= 4 spatial switches 18a ...d of the spatial switching stage 18 are time-shared over a frame so that each switch 18a, 18b, 18c, 18d is capable of switching F time-slots. Thus, for example, traffic can be switched from switches 30a,..,d in input aggregation 16a to switches 34a...d in output aggregation 20a by any and all of the particular switches 18a,b,c,d Moreover, each timeslot in a frame output by the logically adjacent switch 30a is shown connected to the same spatial switch 18a in the different timeslots $t_1...t_{F=4}$. Each frame of four time-slots from each logically adjacent switch in each input aggregation 16a...d will be connected to an input of a different space switch 18a...d of the time-shared spatial switching stage 18, and each space switch 18a...d is capable of receiving input from every input aggregation in each frame.

It is assumed that the switch arrangement is provided with appropriate means to implement the channel assignment process such as an appropriately configured switch scheduler. It is also assumed that no contention exists between outputs and inputs (i.e., that where appropriate each traffic segment is already matched from its input 12a..p to a specific output 14a,..,p). Where required, the matching process may be implemented in any suitable manner using known techniques and need not

subject to any equivalent constraint. Assuming no contention exists, each traffic segment thus requires a path to be assigned from its point of input 12a,...,p to its point of output 14a,..,p across the switch arrangement. The end to end path of an individual traffic segment will comprise different time-slots in different parts of the switch arrangement.

In embodiments of the invention in which each multi-slot comprises M time-slots, and M = F the number of time-slots in a frame switched by one of the time-domain switches, the channel assignment process determines a path for time-slots from each time-domain switch in an input aggregation to an output aggregation such that all timeslots originating from an individual time-domain switch 30a....p in any one of the input aggregations 16a ...16d are switched at all times by the spatial switching stage 18 to the same time-domain switch in an output aggregation 20a...d. This means that all timeslots transmitted by a time-domain switch 30a...p or received by a time-domain switch 34a...p must be going between the same pair of input and output aggregations. As a result, pairs of switches in each of the time-domain aggregation elements 30, 34 logically adjacent to each other are directly "connected" to each other, back to back. This means that either of one of the switches in each connected pair is redundant and therefore need not be implemented in practice. Moreover, it is possible for all of the switches in either of the time-domain aggregation switching stages 30, 34 to be omitted. Furthermore, if all of the switches 30a...d in an input aggregation 16a for example, are redundant, then the time-domain aggregation switching stage 30 is redundant. Similarly, if all of the switches 34a...d in an output aggregation 20a are redundant, then the output aggregation switching stage 34 is redundant.

Thus the channel assignment process is constrained in two ways. Firstly, each aggregation of LF channels is sub-divided into groups of time-slots termed "multi-slots". Each multi-slot comprises a group of timeslots which can be switched by one or more time-domain switches 30a,b,c,d in the time-domain aggregation switching stage 30 to an equivalent number of time-domain switches 34 in the time-domain aggregation switching stage 34. Secondly, all timeslots within a multi-slot are switched from one or more time-domain aggregation switches 30a...p to an equivalent number of time-domain switches 34a....p in the time-domain aggregation switching stage 34.

Each "multi-slot" thus comprises a number of channels (e.g. timeslots) which are

switched together by the spatial switching stage 18 of the switch arrangement from an input aggregation 16a,b,c,d to the same output aggregation 20a,b,c,d as a single switchable entity.

Figure 5 shows how multi-slots can be determined schematically. In Figure 5, time-slots from each input 12a...d are distinguished by their hatchings/fill patterns. Time-slots which carry traffic capable of forming the same multi-slot are shown having the same shape. Accordingly, in Figure 5 although four multi-slots are shown issuing from the time-domain aggregation switching stage 30 logically adjacent to the input side of global spatial switching stage 18, only one multi-slot (#4) is fully occupied with all four of its time-slots carrying traffic segments. The other three multi-slots have only three time-slots occupied. Thus whereas the timeslots in each frame outputted by each time-domain switch (for example, time-slot interchangers) 26a, b, c, d will comprise selective time-slots from a single input 12a,b,c,d, each frame output by a switch 30a,b,c,d which is logically adjacent to the global spatial switching stage 18 is not so restricted but instead is capable of comprising time-slots derived from any of the plurality of inputs 12a...d. Effectively, therefor, in this embodiment, where M = F, the multi-slot constraint limits the contents of each frame output by a time-domain switch 30a...d in each aggregation element 16a for example so that the spatial switching stage switches a smaller number of switchable entities (multi-slots) and so that each time-domain switch 30a...d can be associated with one or more specific switches in an output aggregation.

As mentioned above, it is possible for more than one switch in an input aggregation switching stage 30 logically adjacent to the spatial switching stage 18 to contribute timeslots to form a multi-slot (M>F). In other embodiments, such as recursive multi-slotting which is described in more detail herein below, the size of each frame is larger than the number of multi-slots (M<F). In such embodiments, while all time-slots within a multi-slot are switched to the same time-domain switch in time-domain switching stage 34, not all timeslots within a frame outputted by the time-domain switch need to be switched to the same switch in time-domain switching stage 34 . Thus, while all time-slots from a time-domain switch are switched to the same time-domain switch in the simplest type of multi-slotting (when M = F) this is not true for recursive multi-slotting or for other embodiments in which M < F. In recursive multi-slotting it is desirable to allow the number of time-slots in each multi-slot M to be less than F the number of time-slots in each frame. This means that while all timeslots within a multi-slot are switched to the same time-domain switch in time-domain

switching stage 34, not all timeslots within a time-domain switch in time-domain switching stage 30 need to be switched to the same time-domain switch in time-domain switching stage 34.

The result of imposing such constraints on the logical representation of the switch arrangement 10 effectively results in one of the two time-domain aggregation switching stages 30, 34 adjacent to the spatial switching stage 18 becoming redundant as the two aggregation switching stages are now effectively connected back to back across the switch arrangement 10. Moreover, as the spatial switches 18a,b,c,d forming spatial switching stage 18 can switch the multi-slots as a switchable entity (rather than individually switching each time-slot), they are able to switch multi-slotted aggregations of channels at the slower frame rate instead of the full time-slot rate.

Such an embodiment of the invention removes the requirement in large switch fabrics having multiple stages of individual, smaller space switches for an equivalent "speed-up" in switch fabric hardware; instead a single space switch fabric or connector suffices. The fabric and line-card requirements needed to implement the invention are the same or similar to the minimum requirements of the 2-stage load-balanced Birkhoff-von Neumann switch when using a switch fabric having multiple stages of 2x2 switches (i.e., when implemented as two sets of line-cards and 1 connector).

Whenever a switching stage is rendered redundant in the logical representation of the switch arrangement, it indicates that it is possible for the logical representation to be used to assign channels for a real switch arrangement which does not implement the redundant stage. Accordingly, the channel assignment scheme of the invention can be implemented for various real switch arrangements such as are shown schematically in Figure 6 of the accompanying drawings.

In Figure 6 the actual stages which a real switch arrangement may comprise are shown above and below the seven stage logical representation of the switch arrangement. Those skilled in the art will appreciate that certain stages may represent hops through stages in the actual switch arrangement. In practice, the channel assignment process according to the invention may be implemented by an appropriately configured scheduler comprising an appropriate distribution of processors amongst the components of the switch arrangement. Such a scheduler comprises means for scheduling time-slotted traffic to be switched across a multi-

stage switch arrangement capable of being represented by a seven-stage logical representation of the switch arrangement 10 according the invention such as is described herein above. A constrained channel assignment process will now be described in which time-slots carrying traffic-segments capable of forming multi-slots across the spatial switching stage are identified.

The constrained channel assignment process first determines the number of potentially available multi-slots, and ensures the maximum number of traffic segments are used to form the potentially available multi-slots (i.e., such as multi-slot #4 in Figure 5). Then the potential multi-slots are matched across the global spatial switching stage 18 of the switch arrangement. Where appropriate the number of multi-slots matched is determined to provide the largest number of occupied time-slots (i.e., to determine which requests from which input aggregations should be granted/accepted for connection to the appropriate output aggregations). The multi-slot matching sub-process is then followed by a channel (i.e. path) assignment sub-process. Both sub-processes enforce the constraint conditions.

The channel assignment process according to the invention thus comprises at least the features shown in Figure 7 of the accompanying drawings. Firstly, the constrained channel assignment process computes a request matrix between aggregations (step 70). Those skilled in the art will appreciate that prior to matching, it is necessary to determine the numbers of requests for time slots between input and output aggregations. For example, for a time-division multiplexed switch arrangement, each request represents a TDM channel (e.g., a time slot) from an input of the TDM switch arrangement. For cell (or packet) switch arrangement, each request represents a single cell (or packet) from an input of the switch arrangement.

The process then determines the number of potential multi-slots (step 72). For each input and each output aggregation, the number of potential multi-slots which are required to support the requests determined in the previous step is determined, subject to the constraints the invention imposes on what traffic-segments are able to occupy of the time-slots forming each multi-slot. For example, with the constraint that a multi-slot is switched from an input aggregation to only one output aggregation and that all time-slots within a multi-slot are switched from one or more of the switches in time-domain aggregation switching stage 30 in the input aggregation containing the multi-slot to the same number of switches in the one output aggregation to which the multi-slot is switched. The minimum number of multi-slots

which can switch all the requests is then determined. The number of requests for traffic segments, and hence the number of time-slots, in each multi-slot are identified. This means that each multi-slot is filled with as many requests as possible.

The matching phase of the constrained channel assignment process then matches the potential multi-slots between input aggregations and output aggregations (step 74) whilst maximising as much as possible the numbers of traffic segments in the chosen multi-slots. At this stage, a higher number of multi-slots will exist than will be able to be matched as contention will exist between the input aggregation switches for the switches in the output aggregations . The matching process is configured to optimise the selection of multi-slots which contain the largest numbers of requests satisfying the constraint conditions imposed. If selected, a multi-slot will be matched from its input aggregation to an output aggregation via the spatial switching stage 18 of the switch arrangement 10. Once matched, the requests contained within a multi-slot are accepted for switching between the relevant input and output aggregations.

However, not all multi-slots selected for matching will be able to be matched if the switch arrangement is configured in such a way that blocking can occur. Blocking can be mitigated and/or obviated in a variety of ways, which are described in more detail later.

Next the process assigns time-domain switches 30a..p and spatial switches 18a...d to the accepted multi-slots (step 76). This can be done by any appropriate path-searching process, preferably, a path-searching algorithm which mitigates and/or obviates multi-slotting. By enforcing the constraint, the assignment of specific switches 30a,..,p in aggregation switching stage 30 and specific switches 34a,...,p in aggregation switching stage 34 to each accepted multi-slot, spatial switches 18a,b,c,d are also assigned. This is done using a path-searching algorithm which is rearrangeably non-blocking to enable the number of spatial switches 18a,b,c,d to be minimised.

Finally, the process assigns individual time slots to time-slotted traffic segments within each input and output aggregation (step 78). This involves determining which time-slotted traffic segments to block and which should be assigned to which multi-slot (and to which time slots) through the global spatial switching stage 18 and to which internal time slots within the input and output aggregations. This enables an end-to-end path for each traffic segment to be created from input to output of the

switch arrangement.

Depending on the specific real switch arrangement for which the channel assignment process is to be implemented, other steps may also be implemented. Examples include: reassigning the order in which time-slotted traffic segments (e.g. in input aggregation 16a, the order in which timeslots TSLOT#A...#P) leave the first time-domain aggregation switching stage 26 if the final time-domain aggregation switching stage 38 is omitted from the physical switch arrangement; and/or reassigning the order in which time-slotted traffic segments leave the time-domain aggregation switching stage (30, 34) if time-domain switching stage (34, 30 respectively) is removed. Other steps may be implemented to mitigate blocking, for example by prioritising blocked time-slotted traffic segments to be switched with priority in the next frame or current frame.

Referring now to Figure 8 of the accompanying drawings, a switch arrangement for which channels are assigned according to one embodiment of the invention is shown. In Figure 8, each timeslot is assumed to contain a time-slotted traffic segment.

Each time-slot is associated with a traffic segment identifier which has at least two functions. Firstly, it identifies the input i along which the traffic carried by the time-slot originates (for example, its address, here presented by a number i where $1 \leq i \leq 16$ as the switch arrangement has N=4 and L = 4 i.e., NL = 16 inputs). Secondly it identifies the output j of the switch arrangement to which the traffic carried in that time-slot is to be switched (for example, the address of the output as represented by a number j, where $1 \leq j \leq 16$).

The identifier may also comprise an additional means to identify the logical (or physical) position of a time-slotted traffic segment in storage, for example, as represented here by a number which indicates its position from the head-of-line (HOL) in the relevant VOQ where it is queued for its destination output (here 1 means the HOL). For clarity, individual time-slotted traffic segments are assumed in this embodiment to be assigned in numerical order to the time-slots at the ingress to each input and to the time-slots at the egress from each output of the switch arrangement but those skilled in the art will appreciate this is just a simplification of many possible orders.

The number of traffic segments requesting connection between each input aggregation and each output aggregation can be represented by a traffic request matrix. Here, a row represents the number of time-slotted traffic segments queued at an input aggregation 16a,..,d which have been accepted for connection (i.e. for switching) in the next frame, for example by a previous matching algorithm, to each particular output aggregation 20a,b,c,d. Thus, given the example shown in Figure 8, the request matrix row entry for the input aggregation 16a is:

$$[r_{i,j}] = \begin{bmatrix} 2 & 2 & 6 & 6 \\ - & - & - & - \\ - & - & - & - \\ - & - & - & - \end{bmatrix} \quad \text{...eqtn. 1}$$

The remaining rows are similarly computed for the other input aggregations 16b,16c ,16d (taking N=4 in this embodiment). Each column of the request matrix represents the number of time-slotted traffic segments from each input aggregation which have been accepted for connection to an output aggregation in the next frame. Thus for the output aggregation shown in Figure 7, the partial request matrix is:

$$[r_{i,j}] = \begin{bmatrix} 2 & 2 & 6 & 6 \\ 3 & - & - & - \\ 5 & - & - & - \\ 6 & - & - & - \end{bmatrix} \quad \text{...eqtn. 2}$$

As an example, assume that the entire request matrix for the switch arrangement 10 shown in Figure 7 is

$$r_{i,j}] = \begin{bmatrix} 2 & 2 & 6 & 6 \\ 3 & 2 & 5 & 6 \\ 5 & 5 & 3 & 3 \\ 6 & 7 & 2 & 1 \end{bmatrix} \quad \text{...eqtn. 3}$$

In this embodiment, each multi-slot comprises at most the same number of timeslots as each time-domain switch 30a,..,p forming input aggregation time-switching stage 30 can switch per frame (assuming this is the same for the switches in the output aggregation switching stage 34), i.e., M = F. Each multi-slot therefore cannot contain more than 4 time-slots (which reflects the four time-slots each time-domain aggregation switch switches per frame). Potential multi-slots are filled as much as possible.

Accordingly, whenever the number of time-slotted traffic segments already matched in the previous matching stage of the scheduling process from input i to output j

exceeds four time-slotted traffic segments), a plurality of multi-slots is required to implement the channel assignment constraint. Ideally, each multi-slot is filled to capacity as much as possible. Thus for $r_{1,4} = 6$ requests, one multi-slot will contain requests for four traffic segments whereas the other multi-slot will contain only two requests.

To enforce the multi-slot constraint during the matching stage of the channel assignment process, known matching algorithms for example, maximum weight matching algorithms, maximum size matching algorithms, or other matrix decomposition techniques could be used or modified to impose the constraint.

In the best mode of the invention currently contemplated by the inventors, the following matching algorithm is used:

a) rows and columns in the request matrix are matched one at a time to ensure the next row or column to be matched has the lowest non-zero number of available matrix entries large enough to match at least one multi-slot of the current size in the current recursive step.

Optionally, the matching algorithm should update the number of available matrix entries large enough to match at least one multi-slot of the current size in the current recursive step in the relevant row and column after each multi-slot is matched.

b) the matching algorithm should share out the n granted (matched) multi-slots along each row or column as much as possible between different request matrix entries large enough to grant (accept) each matched multi-slot.

Optionally, the matching algorithm should search for multi-slot requests to grant starting from a pointer and proceed cyclically along the row or column as many times as necessary. Optionally, the matching algorithm updates the pointers after each recursive step or after each frame.

c) before each multi-slot request can be granted (matched), the matching algorithm must check each selected request matrix entry along a row (or column) to determine whether its corresponding column (or row) is already fully booked, i.e. whether its occupancy is n matched multi-slots. If it is,

another request matrix entry must be chosen.

d) the matching algorithm must up-date the occupancy of the corresponding row and column (number of multi-slots already matched in the row or column) after each multi-slot is matched.

Rules a) to d) listed above relate to the recursive (at the outset) multi-slot matching algorithm which currently represents the method mode of the invention contemplated by the inventor. However, as those skilled in the art will appreciate many alternative matching algorithms exist which could be implement using alternative rules.

In one embodiment of the invention, the matching algorithm enables each input aggregation or output aggregation (whose requests occupy each row or column respectively of the request matrix) to have all the necessary information made available when they make a matching decision (strictly when their associated processors or arbiters make a matching decision). To do this, matchings are performed sequentially one row or column at a time, and because all matching decisions have had all the necessary information available, only one booking phase is required, so that grants automatically become acceptances.

In addition, before each multi-slot request can be matched, the matrix entry chosen along the row (or column) must be checked as to whether its corresponding column (or row) is already fully booked. This can be achieved by a simple look-up process. Where a column (or row) is fully booked, another matrix entry must be chosen. To provide this information, the occupancy of the corresponding row and column (number of multi-slots already matched in the row or column) must be up-dated after each match. These modifications ensure that no row or column overbooks multi-slots. It is possible for 100% of all nN admissible multi-slot requests to be granted (and thereby accepted). The computing time for this matching process of the constrained channel assignment is O(nN) which is acceptable as a channel assignment process within a frame-based scheduling algorithm because the computing time allowable for the overall time-slotted traffic segment scheduling is O(F), where F is the frame duration (number of time slots per frame). In frame-based scheduling the frame duration F can be as many time slots as there are switch or network ports, for example, given the switch arrangement shown in Figure 4, F can be O(LN), which is itself O(nN).

Some detailed examples will now be given of matching algorithms which can be used to force the multi-slot constraint. Existing matching algorithms could be employed, such as a maximum weight matching algorithm, a maximum size matching algorithm, matrix decomposition techniques or even a modified version of the heuristic multi-slot matching algorithm proposed later below in which, for example, suitable modifications include a) to make the available matrix entries represent the number of matrix entries in a row or column that have a number of time-slot requests at least equal to the size of the multi-slot and b) to up-date the available matrix entries after each multi-slot is assigned (matched). Three examples are given of other suitable heuristic matching algorithms.

MULTIPLE-SELECTION NO-OVERBOOKING

This algorithm is similar to the iSLIP and no overbooking (NOB) algorithms (for more details see Bianco et al, "Frame-based matching algorithm for input-queued switches", HPSR 2002, Workshop on High Performance Switching and Routing, Kobe, Japan, 26-29 May, 2002, the contents of which are hereby incorporated by reference for details) in format, but the outputs and inputs select the potential multi-slots with the largest numbers of requests, all at once in each booking phase. It does not matter in which order input and output booking are performed. Here output booking is performed first, followed by input booking. Two or more iterations may be necessary, as the following example shows.

The example can be understood simply by manipulation of the request matrix. Consider again the request matrix between input aggregations (rows) and output aggregations (columns).

$$[r_{i,j}] = \begin{bmatrix} 2 & 2 & 6 & p6 \\ 3 & 2 & p5 & 6 \\ 5 & p5 & 3 & 3 \\ p6 & 7 & 2 & 1 \end{bmatrix} \quad \text{...eqtn. 4}$$

The symbol p represents the pointer location. A pointer up-date rule as used in NOB25 is assumed. For output booking, each output (column) selects the largest 4 potential multi-slots, starting from the pointer. This results in the grant matrix

$$[g_{i,j}] = \begin{bmatrix} 2 & 0' & 2' & p0'' \\ 0' & 2 & p1' & 2' \\ 1' & p1' & 0' & 0' \\ p0'' & 0'' & 0' & 1 \end{bmatrix} \quad \text{...eqtn. 5}$$

Each apostrophe represents the granting of a multi-slot. The numbers represent the remaining requests. Both granted multi-slots and remaining requests are now considered by the inputs (rows) during input booking. Rows 1 and 3 both have 4 multi-slots, all of which are accepted by the inputs. Row 2 is underbooked, with only 3 multi-slots granted. Input 2 will therefore make further requests in a 2nd iteration. Row 4 is overbooked, with 5 multi-slots granted. Input 4 therefore retains the 4 largest multi-slots closest to the pointer and therefore removes the smallest of the 5 multi-slots (3 requests) from column 3. The accepted multi-slots and numbers of requests at the end of the 1st iteration are therefore

$$[a_{i,j}] = \begin{bmatrix} 0 & 2' & 4' & p6'' \\ 3' & 0 & p4' & 4' \\ 4' & p4' & 3' & 3' \\ p6'' & 7'' & 0 & 0 \end{bmatrix} \quad \text{...eqtn. 6}$$

In the 2nd iteration, only input 2 makes its remaining requests to the outputs.

$$[r_{i,j}]_2 = \begin{bmatrix} 0 & 0' & 0' & p0'' \\ 0' & 2 & p1' & 2' \\ 0' & p0' & 0' & 0' \\ p0'' & 0'' & 0 & 0 \end{bmatrix} \quad \text{...eqtn. 7}$$

The only output (column) with an available multi-slot is output 3. Therefore request $[r_{2,3}]$ is granted with a fourth multi-slot. There is no need in this case to perform input booking again, but in general there may be. The final accepted requests and multi-slots after the 2nd iteration are

$$[a_{i,j}]_2 = \begin{bmatrix} 0 & 2' & 4' & p6'' \\ 3' & 0 & p5'' & 4' \\ 4' & p4' & 3' & 3' \\ p6'' & 7'' & 0 & 0 \end{bmatrix} \quad \text{...eqtn. 8}$$

The matching supports 51 of the 64 requests, i.e. 13 blocked requests. This is 20.3% blocking. The number of unsuccessful requests (blocked requests) in each input aggregation (row) is

$$\text{blocked input requests} = \begin{bmatrix} 4 \\ 4 \\ 2 \\ 3 \end{bmatrix} \quad \text{...eqtn. 9}$$

and the number of unsuccessful requests (blocked requests) in each output aggregation is

$$\text{blocked output requests} = [3 \;\; 3 \;\; 4 \;\; 3] \quad \text{...eqtn. 10}$$

Therefore no input or output aggregation has more than 4 requests blocked. This corresponds to the expected maximum number of requests that could ever be blocked. This example has been chosen to demonstrate the reason for this, when L=N. When an input aggregation has, for example, 3F/2 requests destined for each one of half of the N output aggregations, it requires N/2 multi-slots each containing F requests (NF/2 requests altogether) to these output aggregations, plus a further N/2 multi-slots each containing F/2 requests (NF/4 requests altogether). Hence N multi-slots are needed to transmit to just half of the output aggregations. When L=N, this is all of the available multi-slots. This leaves at most LF/4 requests blocked to the remaining N/2 output aggregations, which is 4 in our example, when the LF/4 requests are uniformly distributed between the remaining N/2 output aggregations.

There is some flexibility in assigning particular requests to multi-slots, but when the remaining requests are uniformly distributed between the output aggregations, it makes no difference which of the smaller requests are assigned to half of the multi-slots. The maximum blocking probability is therefore expected to be 25%; 4 requests in each aggregation in our example. Therefore we should expect only input aggregation (row) 1 to have 4 blocked requests, because it has 3F/2=6 requests to half of the N (i.e. to 2) output aggregations. But row 2 and column 3 also have 4 blocked requests. This algorithm for matching multi-slots to requests is therefore not performing ideally.

PRIORITISED SINGLE SELECTION NO-OVERBOOKING MULTI-SLOT MATCHING

This is a more complex algorithm in which inputs and outputs are continually re-prioritised, and the highest priority input or output selects only one potential multi-slot at a time; the one that has the largest number of requests starting from a pointer, consistent with the number of multi-slots in both rows and columns not exceeding the allowed number, which is 4 in our example. Priorities are defined by the remaining row-sums and column-sums for each input/output. Consider the same request matrix as in eqtn.1. The number of requests in each input and output are also shown as the row-sums and column-sums.

$$[r_{i,j}] = \begin{bmatrix} 2 & 2 & 6 & p6 \\ 3 & 2 & p5 & 6 \\ 5 & p5 & 3 & 3 \\ p6 & 7 & 2 & 1 \end{bmatrix} \begin{matrix} 16 \\ 16 \\ 16 \\ 16 \end{matrix}$$

$$16 \quad 16 \quad 16 \quad 16$$

...eqtn. 11

At this stage, all inputs (rows) and outputs (columns) have the same number of requests, and hence the same priority. Selecting the largest multi-slots starting from the pointers in the outputs (columns), the remaining requests after all four columns have selected their multi-slots are

$$[r_{i,j}] = \begin{bmatrix} 2 & 2 & 2' & p2' \\ 3 & 2. & p5 & 6 \\ 5 & p5 & 3 & 3 \\ p2' & 3' & 2 & 1 \end{bmatrix} \begin{matrix} 8 \\ 16 \\ 16 \\ 8 \end{matrix} \quad \text{...eqtn. 12}$$

$$12 \quad 12 \quad 12 \quad 12$$

The next two highest priority ports are inputs (rows) 2 and 3. When these have selected their multi-slots, the inputs 2 and 3 and outputs 1 and 3 all have the same priority (12). It doesn't matter which of these is done next, but after two more selections the result is

$$[r_{i,j}] = \begin{bmatrix} 2 & 2 & 2' & p2' \\ 3 & 2 & p1' & 2' \\ 1' & p1' & 3 & 3 \\ p2' & 3' & 2 & 1 \end{bmatrix} \begin{matrix} 8 \\ 8 \\ 8 \\ 8 \end{matrix} \quad \text{...eqtn. 13}$$

$$8 \quad 8 \quad 8 \quad 8$$

Starting again with the outputs, the result after the next four multi-slots are selected is

$$[r_{i,j}] = \begin{bmatrix} 2 & 2 & 2' & p2' \\ 0' & 2 & p1' & 2' \\ 1' & p1' & 0' & 0' \\ p2' & 0'' & 2 & 1 \end{bmatrix} \begin{matrix} 8 \\ 5 \\ 2 \\ 5 \end{matrix} \quad \text{...eqtn. 14}$$

$$5 \quad 5 \quad 5 \quad 5$$

The remaining requests after the next four multi-slots are selected are

$$[r_{i,j}] = \begin{bmatrix} 0' & 2 & 2' & p0'' \\ 0' & 0' & p1' & 2' \\ 1' & p1' & 0' & 0' \\ p2' & 0'' & 0' & 1 \end{bmatrix} \begin{matrix} 4 \\ 3 \\ 2 \\ 3 \end{matrix} \quad \text{...eqtn. 15}$$

$$3 \quad 3 \quad 3 \quad 3$$

The final accepted requests, accepted multi-slots and blocked requests are

$$[a_{i,j}] = \begin{bmatrix} 2' & 0 & 4' & p6'' \\ 3' & 2' & p4' & 4' \\ 4' & p4' & 3' & 3' \\ p4' & 7'' & 2' & 0 \end{bmatrix} \begin{matrix} 4 \\ 3 \\ 2 \\ 3 \end{matrix} \quad \text{...eqtn. 16}$$

$$3 \quad 3 \quad 3 \quad 3$$

There are now 52 requests accepted and 12 blocked (18.75%); 1 less blocked request than the multiple selection algorithm. None of the input or output aggregations has more than 4 requests blocked. Therefore none requires more than one additional multi-slot or more than one time-domain switch in the time-domain

aggregation switching stages 30, 34 adjacent to the spatial switching stage 18 of the switch arrangement per input or output aggregation(and consequently just one more spatial switch in the spatial switching stage 18) to remove the blocking. With this algorithm, only input aggregation (row) 1 has 4 blocked requests. All other rows and columns have fewer blocked requests, as one would expect ideally.

There are $O(N^2)$ main steps in this algorithm, one for selecting each multi-slot. Even if the requests in each row or column are sorted into size order (number of requests contained) before the algorithm begins, it may be necessary to inspect up to all requests in a row or column to select the multi-slot. Therefore the number of computing steps for the entire algorithm may be as high as $O(N^3)$. If remaining requests are re-sorted in size order after each multi-slot is assigned (matched), then the number of computing steps could be reduced to $O(N^2 \log_2 N)$. The above matchings are performed at the lowest possible level of matching, i.e. from each input aggregation to each output aggregation, for which only whole multi-slots are considered (see below, where multi-slots are shared between aggregations).

It is expected that at most 25% of a aggregation's requests could be blocked with the above technique and this suggests that blocking could be removed by employing 25% extra multi-slots per aggregation element; i.e. 25% extra 3rd stage and 5th stage switches in each of the time-domain aggregation switching stages 30, 34 logically adjacent to the spatial switching stage 18 of the switch arrangement, for example, 25% more time-slot interchangers (and line-cards). Both of the time-domain aggregation switching stages 30 and 34 logically adjacent to the spatial switching stage 18 of the switch arrangement are required for these extra multi-slots, because the requests within one multi-slot could be going to or coming from different aggregations. The extra switches (time-slot interchangers and/or line-cards) would need to be interconnected via 25% extra spatial switches provided by the spatial switching stage 18 of the switch arrangement.

HIERARCHICAL, MULTI-LEVEL MULTI-SLOT MATCHING

The multiple selection and prioritised, single selection multi-slot matching algorithms above both perform the matchings at a single overall level but it is possible to perform the matching process at more than one level, i.e., to implement the channel assignment process using a multi-level matching process.

The requests are first aggregated to the higher layers of the hierarchy, matched at the highest layer then de-aggregated back down the hierarchical layers. The benefit of multi-level matching is that requests from different input aggregations can share the same multi-slots, which improves the packing of requests into the multi-slots. Hence fewer multi-slots are required and fewer spatial switches in the spatial switching stage 18 are required. Multi-stage matching also enables parallel processing to be implemented in the lower levels of the hierarchy.

The following is an exemplary embodiment of the invention in which a two-level matching process is implemented for the switch arrangement shown in the drawings and described herein above. In this embodiment, where there are four input aggregations and four output aggregations, two matching "layers" will be implemented. Each matching "layer" comprises a different level of aggregation of the inputs of the switch arrangement. Accordingly, in this embodiment, the requests from two of the input aggregations 16a, b are aggregated into a larger aggregation of requests and the requests from the remaining two input aggregations 16c,d are aggregated into a second larger aggregation of requests. Then a second layer of matching is performed in which the number of requests aggregated into a set of requests is reduced (in this case back to the requests for each individual aggregation 16a...d). In this multi-layer matching embodiment, a prioritised, single-selection no-overbooking matching algorithm is utilised.

TWO-LEVEL, PRIORITISED, SINGLE SELECTION NO-OVERBOOKING

Again as in the previous examples a 4x4 request matrix between input and output aggregations is assumed to have been generated:

$$[r_{i,j}] = \begin{bmatrix} 2 & 2 & 6 & p6 \\ 3 & 2 & p5 & 6 \\ 5 & p5 & 3 & 3 \\ p6 & 7 & 2 & 1 \end{bmatrix} \begin{matrix} 16 \\ 16 \\ 16 \\ 16 \end{matrix} \qquad \text{...eqtn. 17}$$

$$16 \quad 16 \quad 16 \quad 16$$

Here each input aggregation has 16 requests for all output aggregations. The requests from the input aggregations are further aggregated into a 2x4 request matrix (for the highest matching layer) in which input aggregations 16a,16b contribute to the first row of the request matrix and input aggregations 16c,d contribute their requests to the second row of the request matrix:

$$[r_{i,j}] = \begin{bmatrix} 5 & 4 & p11 & p12 \\ p11 & p12 & 5 & 4 \end{bmatrix} \begin{bmatrix} 32 \\ 32 \end{bmatrix} \quad \text{...eqtn. 18}$$

16 16 16 16

It is possible to assign multi-slots in this first matching in a number of ways. For example, each input (row) can be assigned 8 multi-slots of up to 4 requests each and each output (column) can be assigned 4 multi-slots. Alternatively, the size of the multi-slot could be increased by the same ratio that the number of aggregations has been diminished, i.e. the size of each multi-slot could be doubled to reflect the fact that the preliminary matching process is being performed for only half the number of aggregations as will be present in the secondary matching process (in which the size of the multi-slots will be halved). So the rows could be assigned up to 4 multi-slots of up to 8 requests each and each output port assigned 2 multi-slots.

This embodiment will be described in terms of the first approach only in which the size of the multi-slot remains the same in each of the matchings, i.e., keeping 4 requests per multi-slot. The pointers for the rows are the two at the right-hand and left-hand extremities.

After four multi-slots have been assigned to the highest priority ports, the remaining requests and the multi-slot allocations are

$$[r_{i,j}] = \begin{bmatrix} 5 & 4 & p7' & p8' \\ p7' & p8' & 5 & 4 \end{bmatrix} \begin{bmatrix} 24 \\ 24 \end{bmatrix} \quad \text{...eqtn. 19}$$

12 12 12 12

After 8 multi-slots have been assigned:

$$[r_{i,j}] = \begin{bmatrix} 5 & 4 & p3'' & p4'' \\ p3'' & p4'' & 5 & 4 \end{bmatrix} \begin{bmatrix} 16 \\ 16 \end{bmatrix} \quad \text{...eqtn. 20}$$

8 8 8 8

After 12 multi-slots have been assigned:

$$[r_{i,j}] = \begin{bmatrix} 1' & 4 & p3'' & p0''' \\ p3'' & p0''' & 1' & 4 \end{bmatrix} \begin{bmatrix} 8 \\ 8 \end{bmatrix} \quad \text{...eqtn. 21}$$

4 4 4 4

After all 16 multi-slots have been assigned:

$$[r_{i,j}] = \begin{bmatrix} 1' & 0' & p0''' & p0''' \\ p0''' & p0''' & 1' & 0' \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \text{...eqtn. 22}$$

1 0 1 0

At this highest matching layer, only 2 requests are blocked and 62 accepted. The final accepted requests and multi-slots and blocked requests are

$$[a_{i,j}] = \begin{bmatrix} 4' & 4' & p11''' & p12''' \\ p11''' & p12''' & 4' & 4' \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \text{...eqtn. 23}$$

$$1 \quad 0 \quad 1 \quad 0$$

Those of ordinary skill in the art will find the process for de-aggregation back to the input aggregation straightforward.

In this example, as the time-slots are aggregated between two input aggregations in the preliminary matching process, it is possible for two input aggregations to share a single multi-slot. For example, while input aggregations 16c,d each has a whole multi-slot to output aggregation 20a, each also contributes to a third multi-slot. Any multi-slots that are shared between aggregations must be switched by both time-domain aggregation switching stages 30, 34 logically adjacent to the spatial switching stage 18 of the switch arrangement 10.

The foregoing suggests that if matching is performed at the highest possible level first, i.e. all of the input aggregations contribute requests to the request matrix so that the request matrix aggregates requests from all 4 input aggregations to each of the 4 output aggregations, there need be no blocking whatsoever, because multi-slots can be shared between all input aggregations.

This would imply that no "speed-up" is needed in terms of the number of spatial switches 18a...d in the spatial switching stage 18 of the switch arrangement middle-stage switches (or channels) required. In practice a proportion of the switches in one of the aggregation switching stages 30, 34 logically adjacent to the spatial switching stage 18 of the switch arrangement are likely to need to be retained to support a suitable proportion of the multi-slots being shared in this way, for example if around half of the multi-slots were shared between input aggregations around half of the switches 30a...p, 34a...p in either the time-domain aggregation switching stage providing input 17a...p to the spatial switching stage 18 or half of the switches 34a...p in the time-domain switching stage 34 receiving input 19a...p from the spatial switching stage 18 will need to be retained, and the rest may be removed.

ASSIGN SPECIFIC TIME-DOMAIN AGGREGATION SWITCHES TO THE ACCEPTED MULTI-SLOTS

The method of assigning specific time-domain aggregation switches (for example

time-slot interchangers (TSIs)) to the accepted multi-slots will now be described for an embodiment in which whole multi-slots are capable of being allocated between any pair of input and output aggregations (i.e. when there is no sharing of individual multi-slots between different aggregations) using prioritised, single selection no-overbooking at the lowest matching level (aggregation to aggregation). This embodiment could also be applied to the whole multi-slots and sub-multi-slots resulting from multi-level matching. The other multi-slot matching processes described above which used multi-slots for switching time-slots between input/output aggregations could be similarly adapted.

Once the number of multi-slots to be switched between pairs of input and output aggregations has been determined (as indicated by the apostrophes in eqtn.16), particular multi-slots must be assigned appropriate spatial switches 18a..d in the spatial switching stage 18 of the switch arrangement 10. This requires that specific switches in each of the time-domain aggregation switching stages 30, 34 must also be assigned to the multi-slots. At this point, blocking of the accepted multi-slots (i.e., contention for the specific switches available in each output aggregation can occur).

The multi-slot blocking can be considered as equivalent to a connection problem that can be solved by path-searching techniques from 3-stage circuit switches. Any known path-search algorithm could be used, preferably one that prevents blocking. In the embodiment described below, Andresen's rearrangeably non-blocking algorithm (for more details, see Steiner Andresen, "The looping algorithm extended to base 2^t rearrangeable switching networks" IEE Trans. On Comms., Vol. COM-25, No. 10, 1057-1063 (1977), the contents of which are hereby incorporated by reference) is used, so that blocking of accepted multi-slots is prevented without requiring additional switch hardware (i.e., no additional time-domain switches will be required in aggregation switching stages 30, 34 or spatial switches in spatial switching stage 18).

Andresen's algorithm is based on the looping algorithm for a multi-stage Benes network but can be mapped to a 3-stage Clos switch arrangement (e.g. a Clos network) when an integer power of 2 switch inputs and outputs are terminated on each switch in each of the first and final stages of the three stage switch arrangement.

The mappings between the first and third stage of the Clos network are not direct

equivalents to the input and output aggregations shown in Figures 1 to 4. They are merely logical representations to help us "connect" i.e. assign multi-slots to physical time-domain interchangers and physical spatial switches. The inputs to the first stage Clos switches and the outputs from the third stage Clos switches are multi-slot identities.

The multi-slot identities are treated as if they were the inputs and outputs of each of the first- and final-stage Clos switches, respectively. The spatial switches 18a..d forming the spatial switching stage 18 of the virtual representation 10 of the Clos switch arrangement represent the real switches 18a..d of the global spatial switching stage 18 represented by the seven-stage logical representation together with the logically adjacent time-domain switches of the time-domain aggregation switching stages 30, 34 to which they are connected. Such an arrangement is shown in Figure 10 of the accompanying drawings. The three stage Clos switch at the multi-stage Benes network are also only logical representations of a seven-stage logical representation of a real physical switch arrangement that can have between four and seven actual switching stages physically implemented.

Figure 10 shows an equivalent Benes network for assigning the spatial switches of the spatial switching stage 18 to accepted multi-slots using Andresen's adaptation of the looping algorithm. The identity of each accepted multi-slot is given by the row, column identity of its request $r_{i,j}$ and, when there is more than one accepted multi-slot per request $r_{i,j}$, another identifier in brackets. A meaningful identifier chosen here is the size ranking of the particular multi-slot for that request $r_{i,j}$. When more than one inlet and outlet represents multi-slots between the same input aggregation and output aggregation pair (i.e. for the same request $r_{i,j}$), the inlets and outlets of the actual 1st stage and 3rd stage Clos switches can be associated in pairs arbitrarily, to define a unique connection permutation through the 3-stage Clos switch arrangement.

Here the first and third stages of the logical close network do not represent the switching operations of the input and output aggregations. They serve a completely different purpose in that they assign multi-slots to the time-domain switches and spatial switches of the switch arrangement. In this example, it is assumed that LN=16 multi-slots have been accepted and accordingly Andresen's algorithm uses an equivalent 16x16 Benes network.

The four outermost 2x2 switching stages of the Benes network are shown in Figure

9 together with the four middle-stage switches of the Clos network representing the actual spatial switches 18a...d of the spatial switching stage 18 which are not decomposed further. Applying the looping algorithm results in the paths and 2x2 switch settings shown. The paths give the assignments of the spatial switches implementing the global spatial switching stage 18 and hence the logically adjacent time-domain switches 30a...p, 34a...p to the particular multi-slot identities. In Figure 9, boxes with the same style of hatching on the input side of the Benes network contribute to the same switch in the first Clos switching stage, and boxes with the same style of hatching represent the same receiving switch in the third stage of the Clos switching stage.

ASSIGN SPECIFIC TIME SLOTS WITHIN EACH OUTER SUB-NETWORK

Eqtn.16 gave the final matrix of accepted traffic segment requests and multi-slots between aggregations using the prioritised single-selection no overbooking approach. Figure 10 shows in more detail how for the input and output aggregations 16a, 20a, their accepted multi-slots are assigned to particular time-domain switches in aggregation switching stages 30, 34 and the spatial switches 18a...d in spatial switching stage 18 of the switch arrangement.

It is apparent from eqtn.16 that input aggregation 16a has four blocked requests for time-slots that cannot be allocated (i.e. matched) to a whole multi-slot. Two of these timeslots (designated as 1, 5 and 4, 7) are between aggregation elements 16a and 20b and comprise all of the requests for the time-slots requiring a path between this particular input aggregation 16a and the particular output aggregation 20b.

The other two requests for timeslots which cannot be switched are between input aggregation 16a and output aggregation 20c (comparing eqtn.3 with eqtn.16). Because the other four timeslots requesting a path between input aggregation 16a and output aggregation 20c can be switched, input aggregation 16a must determine which two of the six potential traffic segments requesting a path for the outputs 14j...l (see Figure 3) of output aggregation 20c should not be switched.

There are many potential criteria that could be employed to make the decision. For example, it may be preferable for as many traffic segments as possible from the same VOQ to be assigned timeslots which can be switched together, in order to minimise the chances of the traffic segments being mis-sequenced.

When mis-sequencing is to be avoided by adopting this criterion, the traffic segments designated 3,10(1) to 3,10(4) can be switched via a multi-slot in the same frame and the traffic segments designated as 2,12 and 4,11 need not use the same frame. Alternatively, the first two time-slotted traffic segments could be selected by searching through the inputs 12a...p of the switch arrangement (and their logical inputs in some order). In this embodiment the criterion adopted will be for as many traffic segments from the same VOQ as possible to be assigned timeslots which can be switched together, in order to minimise the chances of the traffic segments being mis-sequenced. Using this criterion the three traffic segments requesting a path which cannot be switched in a multi-slot to output aggregation element 20 are designated as 9,1, 14,1(1) and 14,1(2) in Figure 8.

The time slots within each input and output aggregation can now be assigned. By considering each aggregation as a three-stage switch arrangement in its own right, each aggregation can be path-searched using any known path-searching algorithm. In this embodiment, a rearrangeably non-blocking algorithm such as Andresen's adaptation of the looping algorithm is used. This has the advantage of not requiring extra time slots to prevent blocking. Alternatively, it is also possible to use the concept of multi-slots within each individual input/output aggregation, such that the timeslots switched from one aggregation switching stage to the next within an input/output aggregation may be subject to certain constraints.

Figure 10 of the accompanying drawings shows the assignment of particular multi-slots to particular time-domain switches 30a...d, 34a...din the aggregation switching stages 30, 34 adjacent to the spatial switching stage 18 of the switch arrangement and to the spatial switches 18a...d in spatial switching stage 18 of the switch arrangement. Figure 10 shows how multi-slots are used here to ensure that all logical connections from a time-domain switch 30a....d in the aggregation switching stage 30 are arranged to be switched by the spatial switching stage 18 to the same time-domain switch 34a....d in the aggregation switching stage 34 logically adjacent to the outputs of the spatial switching stage 18, by configuring each time-shared spatial switch 18a...d in spatial switching stage 18 of the switch arrangement to have the same spatial switching permutation between its input and output ports in all t_F time slots of a frame. The permutation can change from frame to frame. This ensures that pairs of logical switches in the aggregation switching stages 30,34 logically adjacent to the spatial switching stage 18 are connected back-to-back via the same

physical spatial switch in switching stage 18 during every frame, thus allowing either aggregation stage 30 or 34 logically adjacent to the spatial switching stage 18 to be removed.

In this embodiment, the constrained channel assignment process according to the invention searches for paths in each aggregation by firstly assigning individual time slotted traffic segments to the time slots at the outputs of the input aggregations and to the time slots at the inputs of the output aggregations. Figure 11 shows one technique which may be used (here mainly for clarity) which is to assign the time-slots in numerical order. In Figure 11 multi-slot 1,1 has only two time-slotted traffic segments going from input aggregation 16a to output aggregation 20a, denoted as time-slotted segments 1,3 and time-slotted segments 2,2.

The constrained channel assignment can search for a path by using a known path-searching process. This particular process also describes how the connections in aggregation switching stage 26 may be reordered in order to eliminate any need for the aggregation switching stage 38.

Once a path has been determined across the input and output aggregations, the redundant aggregation switching stage logically adjacent to the spatial switching stage 18 can be removed. For example, selecting to remove the aggregation switching stage 34, in this example the output aggregation now comprises a spatial aggregation switching stage 36. This yields a 5-stage TSTSS switch overall.

Alternatively, if the aggregation switching stage logically adjacent to the input of the spatial switching stage 18 is removed, then again a 5-stage TSSTS switch is provided.

Returning to the embodiment where the time-domain aggregation switching stage 34 which is logically adjacent to the outputs of the spatial switching stage 18 is removed, the logical connections (i.e. time slots) carrying the time-slotted traffic segments leaving the time-domain aggregation switching stage 30 logically adjacent to the inputs of the spatial switching stage 18 must be re-ordered.

As an example, for multi-slot 1,1, assuming that the result of the internal path-searches within the input/output aggregations has assigned traffic segments 1,3 and 2,2 to internal time slots within the input/output aggregations as shown in Figure 12,

then the re-ordered connections (time slots) to the spatial switch 18a in the spatial switching stage 18 of the switch arrangement would be as shown in Figure 12. Figure 12 clearly shows how traffic segment 1,3 now leaves time-domain switch 30a in time slot t_3 instead of time slot t_1.

BLOCKING

In certain embodiments, the channel assignment process proposed by the invention may result in blocking. To mitigate and/or obviate the problem of unacceptable levels of blocking occurring, some possible solutions will now be described.

The channel assignment process implemented using the multi-slotting constraint advantageously depends on the ability to represent a physical switch arrangement (whether in practice a three-stage or a more complex multi-stage switching arrangement) with a seven-stage logical switch arrangement. The multi-slot constraint enables certain stages of the logical switch arrangement to be effectively redundant so that in practice, a switch arrangement comprising just two aggregation stages of time-domain switches (for example, time-slot interchangers which may be associated with aggregations of line-cards, nodes or terminals depending on the type of inputs which are aggregated for the aggregation switching stages) and three stages of spatial switches, two of which are spatial aggregation switching stages 28, 36 and one of which is a spatial switching stage 18 for the whole switch arrangement. This means that by implementing the multi-slot constraint on the seven-stage logical switch arrangement, the logical switch arrangement can be used to assign channels to time-slotted traffic through a five-stage switch arrangement having a (TST)S(S) or (TS)S(TS) configuration of switching stages, where the bracketed stages are represented by aggregation switching stages.

As the spatial switches forming the spatial switching stage 18 switch frames of traffic either wholly or partially filled with multi-slot traffic, the spatial switches 18a...d are able to switch at the frame rate rather than at the time-slot rate, which means that their speed of operation can be much slower than the full time-slot rate would require.

In embodiments of the invention in which traffic is to be assigned channels through a large switch fabric, the large switch fabric may comprise multiple stages of smaller spatial switches which removes the need for any equivalent "speed-up" in switch fabric hardware; instead a single spatial switch fabric or connector switch can be

used. Such requirements are known from the two-stage Birkhoff-von-Neumann switch when using switch fabrics having multiple stages of 2 x 2 switches, for example, just 2 sets of line-cards and the equivalent of 1 connector can be used.

In some embodiments of the invention, the multi-slotting matching process may result in blocking at an unacceptable level, for example, up to 25% of time-slot requests may be blocked in each aggregation. Consider one such embodiment of the invention, in which switch arrangement 10 is an input queued switch implementing virtual output queuing. In this embodiment, input to the switch arrangement 10 is stored in a virtual queue addressed using a pointer process to a particular destination. One solution to the blocking problem is to choose which time-slotted traffic segments are to be blocked in any VOQ in the current frame, for example, the last queued time-slotted traffic segment could be selected to be blocked, and the blocked time-slot request can be given a higher priority in the next frame. This embodiment of the invention does not directly increase the fabric hardware, however, it has a disadvantage in that it reduces the switch throughput by an equivalent amount.

Alternatively, another means of reducing blocking can be implemented. For example, the scheduling scheme of the invention may be modified to relax the constraint condition so that it does not apply to all channels in an input/output aggregation. To support such a version of channel assignment, each input aggregation or each output aggregation must provide a suitable time-domain aggregation switching stage 30, 34 adjacent to the inner spatial switching stage 18 of switch arrangement 10 such as Figure 13 shows.

In Figure 13, to prevent blocking, the input aggregations 16a,b,c,d and/or the output aggregations 20a,b,c,d are provided with additional switching stages respectively to effectively circumvent multi-slot blocking. The additional switching stages comprise time-domain switching stages 30', 34', each of which comprises one or more time-domain switches (e.g. TSIs) which are logically adjacent to one or more additional spatial switches arranged to implement spatial switching stage 18'. This is shown schematically in Figure 13. The additional spatial switches implementing switching stage 18' differ from the spatial switches in spatial switching stage 18 as they do not require any delay compensation which must be provided in the global spatial switching stage 18 to compensate for the additional delay encountered by non-blocked time-slotted traffic segments passing in conventional multi-slot form through

the global switching stage 18 and just one of the logically adjacent time-domain switching stages 30, 34.

This technique is appropriate whenever some time-slotted traffic segments pass through both time-domain aggregation switching stages 30,34 adjacent to the spatial-switching stage 18 of the switch arrangement and other time-slotted traffic segments do not need to pass through both of these adjacent time-domain aggregation switching stages 30, 34. Any suitable method can be used to equalise the differing delays incurred by those time-slotted traffic segments switched using the constrained channel assignment process from those time-slotted traffic segments switched using an unconstrained process, for example, fibre delay lines such as are shown in the switch arrangements of Figure 14

Figure 14 shows an embodiment of a spatial switch such as 18a which comprises an array of tuneable lasers 141 arranged to provide input to an optical coupler 142 which feeds into a single fibre 143 arranged to implement a delay arrange, and which provides input to a wavelength demultiplexer 144. The spatial switch design shown in Figure 14 comprises a wavelength switch and inserts shared fibre delay lines to equalise the frame delays between the (otherwise blocked) time-slotted traffic segments which do pass through a time-domain switching stage 34' and the (unblocked) traffic segments which do not.

In Figure 14, the wavelength switches 18a...d are implemented by passive couplers and demulitpliexers and tuneable lasers are provided at the outputs of the time-domain switching stage 30 logically adjacent to the spatial switching stage 18 of the switch arrangement. By inserting a single fibre delay line 143 between each coupler and the corresponding demultiplexer implementing a spatial switch 18a for example, each fibre delay line can be shared by many wavelength channels. This enables the number of fibre delay lines to be reduced by a factor of 32. For a 1024 x 1024 switch arrangement therefore, only 32 fibre delay lines would be required, and the total fibre requirement for the switch arrangement is reduced to 640 fibre.km. Fibre delay lines are not inserted into the spatial switches 18' which would be used by the blocked time-slotted traffic segments which then go on to be passed through the time-domain switching stage 34'.

Accordingly, if the constrained channel assignment process resulted in a blocking rate per frame of 25%, by increasing the number of adjacent time-domain switches

(e.g. TSIs) in each time-switching stage 30 in the input aggregation 16 and in each output aggregation 20 to 125% of the original number producing the 25% blocking rate, all blocking should be prevented. The number of adjacent time-domain switches 30a,...,p and 34a,...,p for all inputs and outputs are increased, as the extra multi-slot channels may need to be shared by channels originating from different input aggregations and having different destination aggregations. In addition, extra time shared spatial switches (for example, such a switches in switching stage 18' shown in Figure 13) can be provided by the spatial switching stage 18 of the switch arrangement 10 to switch the blocked traffic at the full time-slot rate.

Assuming that the multi-slotting process is not implemented also within the input/output aggregations, the blocked time slots can be path-searched using a suitable rearrangeably non-blocking algorithm such as was used to implement scheduling multi-slots within an input/output aggregation, for example, a suitable Clos algorithm. However, this requires the spatial switching stage 18 to have an even higher number of extra spatial switches to support a higher number of time slots to prevent additional blocking from the non-blocking algorithm. This could double the size of the inner time-shared spatial switching stage 18 of the switch arrangement 10 for strict non-blocking.

Consider, for example, where 25% of time slots are blocked per frame. If the blocked time slots are path-searched using Clos with strict non-blocking, a line card "speed up" of 3 is expected together with a space-switch speed up of 1.5. However, if instead, blocked time-slotted traffic segments are switched with priority in the next frame using multi-slotting, or if multi-slotting is applied recursively to both blocked time-slotted traffic segments and to all time-slotted traffic segments at the outset, this reduce the potential speed-up requirement further and "speed-up" of around 2.666 and space-switch "speed-up" of around 1.333 can be obtained.

As mentioned herein above, in another embodiment of the invention, the blocked time-slotted traffic segments are switched in the current frame. Certain time-slotted traffic segments are selected to be blocked and then an initial time-domain constrained multi-slot channel assignment process is applied to the blocked time-slotted traffic segments. In this embodiment, additional time switches are provided in the time-switching stages adjacent to the central time-shared switching stage and additional time-shared space switches are provided in the central time-shared switching stage, and the constrained multi-slot channel assignment process is

applied recursively to the additional time switches and additional middle-stage time-shared space-switches.

In another embodiment of the invention, time-slotted traffic segments are selected to be blocked using an initial constrained time-slot scheduling process. Then additional time-slots are provided for the blocked time-slotted traffic segments. The constrained multi-slot channel assignment process is then applied recursively using the additional time-slots. This embodiment also enables time-slotted traffic segments time-slotted traffic segments to be switched in the current frame.

MIXED OPERATION WITH AND WITHOUT THE TIME-DOMAIN AGGREGATION SWITCHING STAGE 34 LOGICALLY ADJACENT TO THE OUTPUTS OF THE SPATIAL SWITCHING STAGE 18

Consider an embodiment of the invention in which blocked time-slotted traffic segments are switched with priority in the next frame. This means that neither the time-slotted traffic segments in that frame which are matched and assigned channels nor the time-slotted traffic segments which are blocked from the previous frame will require switching in the time-domain after they have passed through the spatial switching stage 18 of the switch arrangement. This means that the time-domain aggregation switching stage 34 of the switch arrangement which is logically adjacent to the output of the spatial switching stage 18 is now redundant. In this embodiment therefore, the same number of switching stages will be used by both the time-slotted traffic segments scheduled to be switched through the spatial switching stage 18 during the current frame and those segments which were blocked in the previous frame. Accordingly the time-slots from both the current frame and those to be switched in the current frame which were blocked from the previous frame will experience the same delays through the time-domain aggregation switching stages.

In embodiments of the invention where time-slotted traffic segments which are blocked are to be switched in the current frame using some additional switch resources (such as, for example, those shown schematically in Figure 14), compensation for mis-sequencing of the time-slotted traffic segments is provided. For example, by providing some delay compensation for any differential delay between the traffic switched using additional switch resources to that which the non-blocked traffic uses. One example, the time-slotted traffic segments which are not blocked can be assigned paths through the switch arrangement in such a way that the time-domain switching stage 34 which is logically adjacent to the outputs of the

spatial switching stage 18 of the switch arrangement is effectively redundant.

Accordingly, if the blocked time-slotted traffic segments require time-domain switching by a switching stage 34' equivalent to (or provided by) by the time-domain aggregation switching stage 34, a differential frame delay is created between the time-slotted traffic segments originally blocked in that frame and those which were never blocked. Accordingly, whenever some time-domain switches implement time-domain switching stage 34' for otherwise blocked time-slotted traffic segments, mis-sequencing may occur due to the two additional frame delays the originally blocked time-slotted traffic segments will encounter.

This delay may be compensated for by any suitable means such as, for example, by inserting a physical time delay in the path of the non-blocked time-slotted traffic segments through the main switch arrangement 10. For example, if interconnections are optical, by inserting fibre delay lines between real spatial switches 18a...d and the spatial switches 36a..p which implement aggregation switching stage 34 in each of the output aggregations 20a...d.

Consider an embodiment of the invention in which the channel assignment process is implemented for a 1,024x1,024 switch arrangement. In such an embodiment, a typical frame duration would be 1,024x50 nsec = 51 μsecs. Accordingly, compensating for a delay comprising two frame durations = 102 μsecs requires approximately 20 km of fibre delay line in each of LN=1,024 fibres, i.e. ~20,000 fibre.km. This solution however, is neither particularly practical and is also relatively expensive.

A more practical alternative can be implemented by designing the spatial switches 18a...d as wavelength switches. This enables shared fibre delay lines to be inserted which equalise the differential frame delays between time-slotted traffic segments which are not operated upon by the time-domain aggregation switching stage 34 logically adjacent to the outputs of the spatial switching stage 18 and those time-slotted traffic segments which are switched either by time-domain aggregation switching stage 34 or by an equivalent time-domain switching stage 34'.

RECURSIVE MULTI-SLOTTING FOR BLOCKED TIME-SLOTTED TRAFFIC SEGMENTS

The line-card and switch fabric "speed-ups" required to construct a large, multi-stage switch arrangement for time-slotted traffic (for example a time-slotted traffic segment and/or time-slotted traffic segment switch arrangement) can be reduced by applying the principle of multi-slotting recursively to the blocked time slots.

Two embodiments of the channel assignment process according to the invention will be described, the first embodiment is suitable where the switch arrangement can incorporate additional hard-ware to mitigate blocking and the second embodiment is suitable where addition time-slots can be provided for blocked time-slotted traffic segments.

RECURSIVE MULTI-SLOTTING FOR BLOCKED TIME-SLOTTED TRAFFIC SEGMENTS USING ADDITIONAL HARD-WARE

In this embodiment, the channel assignment process described herein above is implemented for a switch arrangement containing some additional hardware (such as, for example, Figure 14 shows schematically) and the channel assignment process is modified so that multi-slotting is applied recursively to the additional hardware in the switch arrangement. The additional hardware may comprise line-cards, time-domain switches such as TSIs and additional spatial switches, etc. There are reserved so that the principle of multi-slotting can be applied recursively to the additional inputs and switches which are required to switch the time-slotted traffic segments blocked by the previous application of multi-slotting for the non-reserved switch arrangement.

On the assumption that blocking is unlikely to ever exceed 25% of the traffic segments requesting connection, the additional hardware is presumed to make available 25% more time-slots for assigning channels across the switch arrangement. Accordingly, each recursive application of multi-slotting applied to the 25% extra hardware will quarter the number of time slots that are logically associated with each multi-slot.

In the embodiment of the channel assignment process for the switch arrangement shown in Figure 4 of the accompanying drawings, in which 4 time slots and 4 multi-slots (corresponding to the four time-domain switches 34a...d (e.g. TSIs)) per input aggregation, the second recursive step of the channel assignment process is applied to just 1 extra time-domain switch in each aggregation switching stage per input aggregation, so that each of the 4 time slots of the additional time-domain switch can

be considered to be a multi-slot in its own right.

Obviously, in this simplified embodiment of the invention, performing a recursive multi-slot matching and channel assignment process is trivial as now in the second step each multi-slots comprises only 1 time slot each! However, it demonstrates the fact that due to the time slots sharing the extra time-domain switch 30e potentially having different source and destination aggregations, the extra time-domain switch 34e in time-domain switching stage 34 must remain, although the remaining time-domain switches 34a...d can be removed. In this simple example of an embodiment of the invention, the line-card "speed-up" = 1+1+2x1/4+ = 2.5 and the space-Switch "speed-up" = 1/3(1+1/4)+1/3(1+1/4)+1/3(1+1/4) = 1.25

Those skilled in the art will appreciate that in large switches and switching networks, recursion could be applied multiple times, each time reducing the number of time slots in each new logical multi-slot by a factor 4 (assuming 25% maximum blocked time slots in each recursion). In each recursion, the number of new logical multi-slots is always the same quantity, which can be equal to the number of output aggregations N (but not necessarily).

In each recursion, the 25% extra TSIs relative to the previous recursion are interconnected by 25% extra spatial switches in the spatial switching stage 18 relative to the previous recursion. The same multi-slot matching algorithms which have already been described herein above can be used in each recursion in the recursive embodiments of the invention. Similarly, the channel assignment path-searching algorithms already described can be used either in each recursion or just once of all recursions.

By making the assumption that there are always N logical multi-slots in each recursion, containing smaller and smaller numbers of time slots in successive recursions, from the second recursion onwards the number of time slots is smaller than the frame length F. This means that now it is only possible to remove the time-domain aggregation switches in aggregation switching stage 34 under two conditions:

1) all time slots within a multi-slot in time-domain switching stage 30 are destined for the same time-domain switching stage 34 multi-slot; and
2) the frame size operated on by each time-domain switch (for example, the TSI

size) equals the multi-slot size in the time-domain aggregation switching stages 30, 34.

The second condition means that each logical time-domain switch in the time-domain aggregation switching stage 30 adjacent to the spatial switching stage 18 of the switch arrangement must have the same number of inlets as the number of time slots in the multi-slot relevant to each recursive step . Each time-domain switch in the time-domain aggregation switching stage 34 logically adjacent to the output of the spatial switching stage 18 of the switch arrangement 10 must have the same number of outlets, equal to the number of time slots in the multi-slot relevant to each recursive step.

This means that it is not sufficient simply to add extra time-domain switches (for example by providing line-cards with additional time-domain aggregation switches) with F time slots and partition those time slots into multi-slots with fewer than F time slots. This would result in asymmetric switches in each of the aggregation switching stages 30, 34 with F potential shared inlets (outlets) on one side of the aggregation switching stage switches 30a...p, 34a...p and fewer outlets (inlets) corresponding to the multi-slot size on the other side of the aggregation switches 30a..p, 34a...p.

Moreover, this would imply that within each input or output aggregation respectively, the time-domain switches 30a..p, 34a..p would not have access to all F internal time slots, which would imply that the input and output aggregations themselves would not be treated logically as 3-stage switches for the blocked time slots. Accordingly this means that condition 2) cannot be imposed when the multi-slot sizes are less than the frame length F. This means that for the extra physical hardware used to switch the blocked time slots (i.e. the extra inputs (e.g. line-cards), time-domain switches (e.g. TSIs) and spatial switches 18a...d, the corresponding logical connection to the time-domain aggregation switching stage 34 (or any other equivalent switching stage 34' whether solely implemented within each aggregation or somehow shared) cannot be removed.

Accordingly, for such an embodiment of the invention where the switch arrangement comprises 1024 inputs and 1024 outputs, the channel assignment process with multiple recursive steps can be implemented providing:

Line-card "speed-up" = 1 + 1 + 2(F/4 + F/16 +.....+ 4 + 1)/F = 2.666....; and

Space-switch "speed-up" = (1/3+1/3+1/3).(F + F/4 + F/16 +.....+ 4 + 1)/F = 1.333....

These values are in effect the same as when there is no recursion and blocked time-slotted traffic segments were forwarded in the next frame. But using recursive multi-slotting has two further benefits. Firstly, it reduces computing complexity for path-searching blocked time slotted traffic (e.g. for blocked time-slotted traffic segments/time-slotted traffic segments). Secondly, it reduces delays by enabling blocked time-slotted traffic segments to be forwarded in the same frame.

RECURSIVE CONSTRAINED SCHEDULING USING EXTRA TIME-SLOTS

In this embodiment of the invention, the constrained scheduling process uses the principle of multi-slotting recursively to extra time slots required to switch the time-slotted traffic segments blocked by the previous application of multi-slotting. The precise details depend on how many of the time-domain switches in a time-domain aggregation stage are sought to be removed.

For example, if a switch arrangement does not support in its output aggregations 20 time-domain aggregation switching stage 34 adjacent to the inner spatial switching stage 18 of the switch arrangement 10 such as Figure 4 shows, then each recursion of the multi-slot scheduling scheme re-uses the entire 7-stage logical architecture (and thus the same physical line-cards, time-domain aggregation switches and space switches) but with a different number of time slots per time-domain switch (e.g. TSI) in each recursion.

The logical size of the time-domain aggregation switches in aggregation switching stages 30,34 is made the same as the multi-slot size in each recursion. In effect, the entire switch is re-used with extra time slots instead of using extra hardware; i.e. conventional bandwidth speed-up. The number of time slots needed between 3rd and 5th stages through each space switch of the inner spatial switching stage 18 of the switch arrangement for all recursions is just $F(1+1/4+1/16+\ldots) = 4F/3$. Therefore, for an embodiment in which the switch arrangement comprises a large network with multiple recursive steps bandwidth speed-up = $O(\log_2 F)$, which can be large, line-card "speed-up" = $(O(2\log_2 F),j$ and space-switch "speed-up" = $O((2/3)\log_2 F + (4/9))$.

In an alternative embodiment of the invention, none of the time-domain switches in time-domain aggregation switching stage 34 are removed. In this embodiment, each input and output aggregation is path-searched in the channel assignment sub-

process as a fully available 3-stage TST switch, so the number of extra time slots needed in each recursion is one quarter of those in the previous recursion. In this case, the time-domain aggregation switching stage 34 adjacent to the inner spatial switching stage 18 of the switch arrangement 10 retains all of its switches (e.g., all of the TSIs are retained), which require only the extra time slots. This results in a bandwidth speed-up of (F+F/4+...+4+1) = 4/3 and a line-card "speed-up" = (1+1) x Bandwidth speed-up + (Bandwidth speed-up-1) x Bandwidth speed-up = 8/3 + 4/9 = 3.111, whereas space-switch "speed-up" = (1/3+1/3+1/3) x Bandwidth speed-up = 1.333.

This indicates that this embodiment, by attempting to use extra time slots to transmit the initially blocked time slots, requires a line-card "speed-up" only slightly greater than that required by the embodiment that uses extra hardware to transmit the initially blocked time slots.

RECURSIVE MULTI-SLOTTING FOR ALL TIME-SLOTTED TRAFFIC SEGMENTS/TIME-SLOTTED TRAFFIC SEGMENTS AT THE OUTSET

In embodiments of the invention in which the channel assignment process applies recursive multi-slotting at outset the following can be achieved:

1) a reduction in computing complexity to allow rearrangeably non-blocking path searching to be used instead of Clos algorithms, thus enabling the spatial switching stage 18 to switch at the frame-rate instead of the time-slot rate;
2) the maximum use of as many whole time-domain switches 30a..p and 34a...p in the time-domain aggregation switching stages 30, 34 implemented for each input and each output aggregation as is possible to provide multi-slots, which enables entire time-domain switches 34a...p in the output aggregation switching stage logically adjacent to the spatial switching stage 18 to be removed
3) a reduction in "speed-up"

Whilst the next embodiment will be described in terms of a switch arrangement suitable for switching any time-slotted traffic, those skilled in the art will appreciate that it is particularly (although not exclusively) suitable for time-slotted traffic segment/time-slotted traffic segment switches and networks, and also for general application to path searching in 3-stage Clos networks.

In this embodiment of the invention, the recursive multi-slotting channel assignment process applies a recursive multi-slotting constraint to all time-slotted traffic at the outset, rather than only to blocked time-slotted traffic segments. This enables blocking due to multi-slotting to be completely removed in some embodiments of the invention.

In this embodiment, instead of assigning all multi-slots in each input/output aggregation as whole multi-slots in one go, and accepting that up to 25% of the time-slotted traffic segments may be expected to be blocked, we choose multi-slots of different sizes in a number of recursive steps.

In every recursive step, a number of multi-slots is considered per aggregation (per row and column), as appropriate. In the first step the largest multi-slots are assigned (their size could exceed that of a single time-domains switch 30a, 34a, i.e. multi-slots can consist in multiple time-domain switches , e.g., over 30a,b in input aggregation 16a and over 34a, b in output aggregation 20a for example. In each subsequent recursive step the maximum number of time slots involved is halved.

In the last recursive step the multi-slot size is 1. In all recursive steps, matching of time-slotted traffic segments to multi-slots can be followed by assignment of multi-slots to specific time-domain switches and spatial switches 18a...d (path searching). The path searches in each recursive step treat the switching resources independently of the switching resources used in other recursive steps. Assignment of multi-slots to specific time-domain switches 30a...p, 34a...p and spatial switches 18a...d in each recursive step can be performed using rearrangeably non-blocking path searching methods such as Andresen's adaptation of the looping algorithm . Alternatively, path-searching need not be performed in each recursion, and could even be performed just once for all recursions.

MULTI-SLOT BLOCKING

Even when time-slotted traffic segments are matched to whole time-domain switches (e.g. 30a...p, 34a...p) there are traffic request matrices between input/output aggregations for which recursive multi-slotting can still produce blocking. For example, consider a single row or column of the request matrix, in our L=N=F=4

example with LF=16 time-slotted traffic segments per aggregation, with the following requests

$$[3 \ 3 \ 7 \ 3] \qquad \textbf{eqtn. 24}$$

If the largest multi-slot in the first recursive step has 4 time-slotted traffic segments, so that 4 time-slotted traffic segments could be matched to whole time-domain switches (e.g. 30a...p, 34a...p), then no more than one time-domain switch (e.g., 30a,.. p, 34a...p) per input/output aggregation could be matched as a whole one, because a second one would possess only 3 time-slotted traffic segments leaving one time slot unused, thus producing blocking.

In fact, when considering an entire request matrix with several rows and columns having these numbers, it can be impossible to match only one whole time-domain switch (30a...p, 34a...p) per row or column; for example:

$$\begin{bmatrix} 3 & 3 & 4 & 6' \\ 3 & 3 & 4 & 6' \\ 7' & 7' & 1 & 1 \\ 3 & 3 & 7' & 3 \end{bmatrix} \qquad \textbf{eqtn. 25}$$

In the above request matrix, if we ensure that each row and column has a whole time-domain switch (e.g. 30a etc) having 4 time-slotted traffic segment requests matched in it (marked as apostrophes), we are forced to match two whole time-domain switches to both row 3 and column 4.

This results in a form of multi-slot blocking whereby some rows and columns are prevented from using the single whole time-domain switch that they possess. Thus with this quantity of time-slotted traffic segments not one whole time-domain switches (30a...p, 34a...p) can be used in order to remove the time-domain switches (34a...p) forming the time-domain aggregation switching stage 34 logically adjacent to the outputs of the spatial switching stage 18 of the switch arrangement.

The condition that causes this multi-slot blocking, in this particular matrix, is when all but one (i.e. N-1) entries in both columns 1 and 2 have a size just below the size of the time-domain switch 30a...p, 34a...p, i.e. (F-1), and the total number of requests per row and column is less than or equal to the quantity obtained when the sum of the remaining entries in each column (the two 7s) equals the total number of requests per row and column.

This quantity is given by

$$(C-(N-1)(F-1))+(C-(N-1)(F-1))=C \qquad \textbf{eqtn. 26}$$

$$\therefore C=(2N-3)(F-1)+F-1=18 \qquad \textbf{eqtn. 27}$$

In this example, there are LF=16 time-slotted traffic segment requests per aggregation, and so multi-slot blocking will occur. Even with 18 time-slotted traffic segment requests per row and column, multi-slot blocking can still occur, e.g.

$$\begin{bmatrix} 3 & 3 & 5 & 7' \\ 3 & 3 & 4 & 8' \\ 9' & 9' & 0 & 0 \\ 3 & 3 & 9' & 3 \end{bmatrix} \qquad \textbf{eqtn. 28}$$

NON-BLOCKING CONSTRAINT

In embodiments of the invention in which more time-slotted traffic segment requests are added in each input/output aggregation, then more opportunities are created for whole multi-slots (of 4 time-slotted traffic segments) to exist which enables this type of multi-slot blocking to be mitigated or even completely removed.

For example, let us add just one more time-slotted traffic segment request per row and column, making 19 in total:

$$\begin{bmatrix} 3 & 3 & 5 & 8' \\ 3 & 4' & 4 & 8 \\ 10' & 9 & 0 & 0 \\ 3 & 3 & 10' & 3 \end{bmatrix} \qquad \textbf{eqtn. 29}$$

Now $r_{2,2}$, for example, is forced to become a full multi-slot (4 time-slotted traffic segments) and every row and column possesses only one whole multi-slot, as desired. Hence the total number of time-slotted traffic segment requests C in all rows and columns required to guarantee filling one whole multi-slot of F time-slotted traffic segment requests subject to this type of multi-slot blocking is

$$C \geq (2N-3)(F-1)+F \qquad \textbf{eqtn. 30}$$

To support 19 time-slotted traffic segment, five time-domain switches (e.g. 30a...p, 34a...p plus one extra time-domains switch in each input aggregation 16a...d and each output aggregation 20a..p) , each switch having F=4 time slots are needed, although one switch in each aggregation will not be completely filled.

Accordingly, if the total number of time-slotted traffic segments per input/output aggregation is increased to 20 to fill 5 Time-domain switches (e.g. 30a...p, 34a...p).

This number is still not blocked for one whole time-domain switch:

$$\begin{bmatrix} 3 & 3 & 5 & 9' \\ 3 & 5' & 4 & 8 \\ 11' & 9 & 0 & 0 \\ 3 & 3 & 11' & 3 \end{bmatrix} \qquad \textbf{eqtn. 31}$$

However, 20 time-slotted traffic segment requests do not allow two whole time-domain switches 30a...p, 34a....p to be assigned to all rows and columns:

$$\begin{bmatrix} 3 & 3 & 5 & 9'' \\ 3 & 5' & 4 & 8 \\ 11'' & 9 & 0 & 0 \\ 3 & 3 & 11'' & 3 \end{bmatrix} \qquad \textbf{eqtn. 32}$$

Entry $r_{2,2}$ can provide only one multi-slot of 4 time-slotted traffic segment requests. To support two whole time-domain switches (taken from 30a...p, 34a...p), more requests are needed to provide yet more opportunities, and so on.

In general, the total number of time-slotted traffic segment requests C in each row and column required to guarantee filling n whole multi-slots of F time-slotted traffic segment requests (n whole time-domain switches, subject to this type of multi-slot blocking, is

$$C \geq (2N-3)(F-1)+nF \qquad \textbf{eqtn. 33}$$

Since the number of time-domain switches (e.g. 30a...p, 34a...p) per input/output aggregation is C/F, the proportion p of all time-domain switches 34a...p in the time-domain aggregation switching stage 34 logically adjacent to the output of the spatial switching stage 28 that can be removed is

$$p = \frac{n}{C/F} = \frac{nF}{C} = \frac{1}{1+\frac{(2N-3)(F-1)}{nF}} \quad \rightarrow \frac{n}{2N} \text{ as } N \rightarrow \infty$$

$$\rightarrow 1 \quad \text{as } n \rightarrow \infty \qquad \text{eqtn. 34}$$

Evidently larger and larger proportions of time-domain switches 34a...p could be removed as more and more whole time-domain switches 34a...p are filled, without blocking, by adding more and more time-slotted traffic segment requests within the aggregations.

The proportion can in principle approach 1. Assuming no other types of blocking exist that would reduce the proportion of time-domain switches 34a...p that could be removed, in the limit it is possible for the "speed-up" in the inputs (line-cards) to approach 2.0 and the "speed-up" in the spatial switches (relative to a 3-stage space-

switch fabric switch arrangement) to approach 1.0, which is the same requirement as the 2-stage Birkhoff-von Neumann switch using multiple stages of 2x2 switches.

MULTI-SLOT MATCHING ALGORITHM

In one embodiment of the invention, the matching process for multi-slot requests in the time-slot request matrix comprises matching rows and columns one at a time in the order that always chooses the next row or column with the lowest non-zero number of available matrix entries large enough to match at least one multi-slot of the current size in the current recursive step.

As an example, in one embodiment the corresponding row's and column's available matrix entries are updated after each multi-slot is matched. The n multi-slot assignments (granted matches) are shared out along the row or column as much as possible between different matrix entries large enough to grant (accept) each matched multi-slot. For example, in one embodiment the assignments start from a pointer and proceed cyclically along the row or column as many times as necessary. The pointer does not need to be up-dated after each recursive step or after each frame.

Before each multi-slot request can be matched, the matrix entry chosen along the row (or column) must be checked as to whether its corresponding column (or row) is already fully booked, i.e. whether its occupancy is n matched multi-slots. If it is, another matrix entry must be chosen. The occupancy of the corresponding row and column (number of multi-slots already matched in the row or column) must be up-dated after each multi-slot is matched.

These requirements are similar in some respects to the operation of the no-overbooking (NOB) algorithms described in the paper referred to hereinabove by Bianco et al which can be used for matching time-slotted traffic segment requests between switch ports in frame-based scheduling with frames of length F time slots.

Certain differences exist however. Firstly, in the NOB algorithm, there are two booking phases (e.g. outputs followed by inputs), in which firstly every output can grant requests simultaneously (in parallel) with all other outputs. Then secondly, every input can accept or reject these grants simultaneously with all other inputs.

This means that overbooking can occur in the first phase, because input/output processor (for example, port processors) are not aware of decisions taken by other input/output processors when making their own decisions. Overbooking is then remedied in the second input booking phase, when any overbooked grants are removed (i.e. not accepted).

This can result in under booking of some inputs, even after multiple iterations. Under booking can also result without removal of overbooked grants. Therefore the NOB algorithms can result in less than 100% of the number of admissible requests in inputs and outputs (rows and columns) being matched.

This situation is tolerable for matching of F time-slotted traffic segments or time-slotted traffic segments in a frame because; a) unsuccessful time-slotted traffic segments can be forwarded in future frames and b) parallel processing in each of the N inputs provides important reductions in computing times (i.e. to O(F)).

However, such a scenario is not tolerable when matching n multi-slots within each recursive step of a channel assignment scheme according to the present invention. The channel assignment process according to the invention seeks essentially to assign time-slots to time-slotted traffic segments and ideally 100% of the number of admissible time slots, and hence multi-slots, must be matched, otherwise some time-slotted traffic segments already accepted by a previous matching algorithm will be blocked. This may be acceptable in one embodiment, for example where blocked time-slotted traffic segments are given priority in the next frame, but we also want an embodiment in which no time slots and hence no time-slotted traffic segments are blocked. Accordingly, to prevent blocking of time-slot requests the NOB algorithm described in the paper by Bianco et al referred to herein above is modified so that the inputs or outputs (rows or columns, i.e. input or output aggregations here) have all the necessary information available to them when they make their matching decisions.

To do this, matchings are performed one row or column at a time, i.e. sequentially, as in 1) above. There is now only one booking phase, so that grants and acceptances are synonymous. In addition, before each multi-slot request can be matched, the matrix entry chosen along the row (or column) must be checked as to whether its corresponding column (or row) is already fully booked.

This can be achieved by a simple look-up process well known to those skilled in the art. If the output (column) is fully booked, another matrix entry must be chosen. To provide this information, the occupancy of the corresponding row and column (number of multi-slots already matched in the row or column) must be up-dated after each match. These modifications ensure that no row or column overbooks multi-slots.

This enables 100% of all nN admissible multi-slot requests to be granted (and thereby accepted) albeit with the computing time becomes O(nN). This is tolerable for matching of time-slot assignments because the computing time allowable for the overall time-slotted traffic segment scheduling is O(F), where F is the frame duration. In frame-based scheduling using multi-stage channel assignment, as here, F can be O(LN), which is itself O(nN). This reduces the computing steps per time slot for each recursion of multi-slot matching to O(1).

DETAILED EMBODIMENT OF RECURSIVE MULTI-SLOT CONTRAINED CHANNEL ASSIGNMENT.

An embodiment of the invention which demonstrates the complete constrained scheduling process with recursive multi-slotting will now be described in more detail for the switch arrangement 10 shown in Figure 4 which is assumed to be implementing VOQ input queuing.

In each recursive step, the available multi-slots are matched and then path searched across the spatial switching stage 18 of the switch arrangement in an appropriate manner such as one of those already described (for example, using Andresen's looping algorithm).

This embodiment requires the total number of requests C in each row and column of the request matrix to satisfy the condition $C \geq 31$ requests, where each request corresponds to the transmission of a single time-slotted traffic segment across the switch arrangement. In this embodiment, it is assumed that C = 32 which results in C/F=8 time domain switches in each of the time-domain aggregation switching stages 30, 34 adjacent to the inner spatial switching stage 18 of the switch arrangement per input/output aggregation, (i.e. 50% of the total of 16 time-domain switches 30a,...,p or 34a,...p) being effectively redundant. Thus the switch can be implemented with only 50% of the switches shown in Figure 4 in the time-domain

aggregation switching stage 34.

In order to match and path-search the C = 32 requests, four recursive steps are required.

In each input/output aggregation, firstly, n=4 multi-slots (where each multi-slot contains four time-slotted traffic segment requests (i.e., each multi-slot is equivalent to a time-domain switch (for example, a TSI)'s time-slot output per frame) are matched by the constrained matching sub-process of the constrained channel assignment process and then the four time-slotted traffic segment requests are path-searched by the constrained channel assignment sub-process. Secondly, four multi-slots are matched and path-searched as above, but this time where each multi-slot comprises only two time-slotted traffic segment requests. Thirdly, four multi-slots comprising just a single time-slotted traffic segment request are matched and path-searched as above. Finally, the four remaining multi-slots which also contain just a single time-slotted traffic segment request are matched as path-searched as above.

The example request matrix now has 32 requests per row and column, which is sufficient to ensure that 4 multi-slots of 4 requests each could be matched in the first recursive step, i.e.

$$r_{1i,j} = \begin{bmatrix} 3 & 3 & 7 & p19 \\ 3 & 17 & p2 & 10 \\ 23 & p9 & 0 & 0 \\ p3 & 3 & 23 & 3 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \\ 2 \\ 1 \end{bmatrix}$$

row & column order: *row4*, *column1*, *row1*, *row2*, ...

$$[1 \quad 2 \quad 2 \quad 2]$$

eqtn. 36

Matching could be performed by any appropriate matching algorithm. In this example, a heuristic multi-slot matching algorithm as described hereinabove is used. Eqtn. 36 shows the pointer positions in the request matrix.

In the first recursive step, the numbers of available matrix entries in each row and column capable of matching at least one multi-slot of size 4 time slots are shown to the right and below the time-slot request matrix. The numbers of available matrix entries represent priorities of rows and columns for matching. The order in which rows and columns are matched is therefore as shown. The following matrices give the multi-slot matches (marked as apostrophes) after each row and column has been matched in the first recursive step. (In another embodiment the priorities, i.e.

numbers of available matrix entries that are large enough, are re-computed for each row and column involved after each multi-slot has been matched, and the next row or column is chosen for matching accordingly).

row 4 column 1 row 1 row 2

$$\begin{bmatrix} 3 & 3 & 7 & p19 \\ 3 & 17 & p2 & 10 \\ 23 & p9 & 0 & 0 \\ p3 & 3 & 23'''' & 3 \end{bmatrix} \begin{bmatrix} 3 & 3 & 7 & p19 \\ 3 & 17 & p2 & 10 \\ 23'''' & p9 & 0 & 0 \\ p3 & 3 & 23'''' & 3 \end{bmatrix} \begin{bmatrix} 3 & 3 & 7 & p19'''' \\ 3 & 17 & p2 & 10 \\ 23'''' & p9 & 0 & 0 \\ p3 & 3 & 23'''' & 3 \end{bmatrix} \begin{bmatrix} 3 & 3 & 7 & p19'''' \\ 3 & 17'''' & p2 & 10 \\ 23'''' & p9 & 0 & 0 \\ p3 & 3 & 23'''' & 3 \end{bmatrix}$$

eqtn. 37

All 16 multi-slots of size 4 are matched (granted and accepted). The remaining time-slot requests, available matrix entries capable of matching at least one multi-slot of size 2 time slots and the order in which rows and columns should be matched in the second recursive step are:

$$r_{2i,j} = \begin{bmatrix} 3 & 3 & 7 & p3 \\ 3 & 1 & p2 & 10 \\ 7 & p9 & 0 & 0 \\ p3 & 3 & 7 & 3 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \\ 2 \\ 4 \end{bmatrix}$$

row & column order: *row3*, *row2*, *column2*, *column3*, *column4*, *row1*, ...

$$[4 \ 3 \ 3 \ 3]$$

eqtn. 38

The following matrices give the multi-slot matches after each row and column has been matched in the second recursive step:

row 3 row 2 column 2 column 3

$$\begin{bmatrix} 3 & 3 & 7 & p3 \\ 3 & 1 & p2 & 10 \\ 7'' & p9'' & 0 & 0 \\ p3 & 3 & 7 & 3 \end{bmatrix} \begin{bmatrix} 3 & 3 & 7 & p3 \\ 3' & 1 & p2' & 10'' \\ 7'' & p9'' & 0 & 0 \\ p3 & 3 & 7 & 3 \end{bmatrix} \begin{bmatrix} 3 & 3' & 7 & p3 \\ 3' & 1 & p2' & 10'' \\ 7'' & p9'' & 0 & 0 \\ p3 & 3' & 7 & 3 \end{bmatrix} \begin{bmatrix} 3 & 3' & 7' & p3 \\ 3' & 1 & p2' & 10'' \\ 7'' & p9'' & 0 & 0 \\ p3 & 3' & 7'' & 3 \end{bmatrix}$$

column 4 row 1

$$\begin{bmatrix} 3 & 3' & 7' & p3' \\ 3' & 1 & p2' & 10'' \\ 7'' & p9'' & 0 & 0 \\ p3 & 3' & 7'' & 3' \end{bmatrix} \begin{bmatrix} 3' & 3' & 7' & p3' \\ 3' & 1 & p2' & 10'' \\ 7'' & p9'' & 0 & 0 \\ p3 & 3' & 7'' & 3' \end{bmatrix}$$

eqtn. 39

All 16 multi-slots of size 2 are matched (granted and accepted). The remaining time-slot requests, available matrix entries capable of matching at least one multi-slot of size 1 time slot and the order in which rows and columns should be matched in the third recursive step are:

$$r_{3i,j} = \begin{bmatrix} 1 & 1 & 5 & p1 \\ 1 & 1 & p0 & 6 \\ 3 & p5 & 0 & 0 \\ p3 & 1 & 3 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \\ 2 \\ 4 \end{bmatrix}$$

row & column order: *row*3, *column*3, *row*2, *column*4, *row*1, *row*4, ...

$$[4 \quad 4 \quad 2 \quad 3]$$

eqtn. 40

The following matrices give the multi-slot matches after each row and column has been matched in the third recursive step:

row 3

$$\begin{bmatrix} 1 & 1 & 5 & p1 \\ 1 & 1 & p0 & 6 \\ 3'' & p5'' & 0 & 0 \\ p3 & 1 & 3 & 1 \end{bmatrix}$$

column 3

$$\begin{bmatrix} 1 & 1 & 5'' & p1 \\ 1 & 1 & p0 & 6 \\ 3'' & p5'' & 0 & 0 \\ p3 & 1 & 3'' & 1 \end{bmatrix}$$

row 2

$$\begin{bmatrix} 1 & 1 & 5'' & p1 \\ 1' & 1' & p0 & 6'' \\ 3'' & p5'' & 0 & 0 \\ p3 & 1 & 3'' & 1 \end{bmatrix}$$

column 4

$$\begin{bmatrix} 1 & 1 & 5'' & p1' \\ 1' & 1' & p0 & 6'' \\ 3'' & p5'' & 0 & 0 \\ p3 & 1 & 3'' & 1' \end{bmatrix}$$

row 1

$$\begin{bmatrix} 1' & 1 & 5'' & p1' \\ 1' & 1' & p0 & 6'' \\ 3'' & p5'' & 0 & 0 \\ p3 & 1 & 3'' & 1' \end{bmatrix}$$

row 4

$$\begin{bmatrix} 1' & 1 & 5'' & p1' \\ 1' & 1' & p0 & 6'' \\ 3'' & p5'' & 0 & 0 \\ p3 & 1' & 3'' & 1' \end{bmatrix}$$

eqtn. 41

All 16 multi-slots of size 1 are matched. The remaining time-slot requests, available matrix entries capable of matching at least one multi-slot of size 1 time slot and the order in which rows and columns should be matched in the fourth and final recursive step are:

$$r_{4i,j} = \begin{bmatrix} 0 & 1 & 3 & p0 \\ 0 & 0 & p0 & 4 \\ 1 & p3 & 0 & 0 \\ p3 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 2 \\ 2 \end{bmatrix}$$

row & column order: *row*2, *column*4, *row*1, *row*3, *row*4, ...

$$[2 \quad 2 \quad 2 \quad 1]$$

eqtn. 42

The following matrices give the multi-slot matches after each row and column has been matched in the fourth recursive step:

row 2 column 4 row 1 row 3

$$\begin{bmatrix} 0 & 1 & 3 & p0 \\ 0 & 0 & p0 & 4'''' \\ 1 & p3 & 0 & 0 \\ p3 & 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 3 & p0 \\ 0 & 0 & p0 & 4'''' \\ 1 & p3 & 0 & 0 \\ p3 & 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 1' & 3''' & p0 \\ 0 & 0 & p0 & 4'''' \\ 1 & p3 & 0 & 0 \\ p3 & 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 1' & 3''' & p0 \\ 0 & 0 & p0 & 4'''' \\ 1' & p3''' & 0 & 0 \\ p3 & 0 & 1 & 0 \end{bmatrix}$$

row 4

$$\begin{bmatrix} 0 & 1' & 3''' & p0 \\ 0 & 0 & p0 & 4'''' \\ 1' & p3''' & 0 & 0 \\ p3''' & 0 & 1' & 0 \end{bmatrix}$$

eqtn. 43

All 16 multi-slots of size 1 are matched in the final recursive step. This means that all multi-slots have been matched successfully and so none of the time slots (time-slotted traffic segments) are blocked. Four of the eight time-domain aggregation switches o f the fifth stage, for which the multi-slot size of 4 time slots fills a whole time-domain aggregation switch, can be removed.

RIGOROUS DERIVATION OF RECURSIVE STEPS

The above example indicates how recursive steps can be determined in one embodiment of the invention by defining the number of multi-slots in each step to be the same as the number of input/output aggregations N. The number of multi-slots in each step could also be defined to be a multiple of the number of input/output aggregations in other embodiments of the invention.

In some embodiments of the invention, the number of multi-slots in each recursive step is determined in a more rigorous manner from the non-blocking constraint. For example, it is possible to freely determine the number of multi-slots and to match different quantities of multi-slots in each of the recursive steps (although some of the recursive steps may use the same quantity). Alternatively, the same quantity m of multi-slots in all recursive steps could be selected and matched. This later embodiment will now be described in more detail.

In this embodiment, it is assumed that the total number of time-slot requests C in any row or column are all to be completely matched by the recursive steps, i.e.

$$C = (2N-3)(F-1)+nF = m\left(F+\frac{F}{2}+\frac{F}{4}+....+1\right) \qquad \textbf{eqtn. 35}$$

Here nF is the total number of time-slot requests that can be matched while searching for multi-slots of a larger size than F/2, i.e. to guarantee no blocking of this number of requests. The actual values of n and m must be determined.

The values of n and m in this embodiment will be chosen so that the total of all the requests in any number of the recursive steps are guaranteed not to be blocked, i.e. the total of all the requests equal nF.

For example, if we insist that the number of requests matched in just the first recursive step, mF, precisely equals the number nF of requests available, i.e.

$$mF = nF$$
$$\therefore m = n \qquad \textbf{eqtn. 36}$$

then

$$C = (2N-3)(F-1)+nF = n\left(F+\frac{F}{2}+\frac{F}{4}+....+1\right) = n(2F-1) \qquad \textbf{eqtn. 37}$$

$$\therefore m = n = (2N-3) \qquad \textbf{eqtn. 38}$$

and $$C = (2N-3)(2F-1) \qquad \textbf{eqtn. 39}$$

Hence for N=F=4, in our example,

$$\therefore m = n = (2N-3) = 5$$
$$C = (2N-3)(2F-1) = 5.7 = 35 \qquad \textbf{eqtn. 40}$$

So the recursive steps would be:

$$5\left(F+\frac{F}{2}+...+1\right) = 5(4+2+1) \qquad \textbf{eqtn. 41}$$

Thus, instead of matching N=4 multi-slots in each recursive step, using 31 (or 32) requests in total and 4 recursive steps, the number of recursive steps is reduced to 3 by matching 5 multi-slots in each recursive step and increasing the total number of requests in each aggregation to 35. The recursions are more efficient. In practice one might choose to use 36 requests, thereby filling 9 time-domain aggregation switches completely. The recursive steps would become

$$5\left(F+\frac{F}{2}+...+1\right)+1 = 5(4+2+1)+1 \qquad \textbf{eqtn. 42}$$

The number of whole time-domain switches that can be removed due to the first recursive step is 5, so the proportion of all time-domain switches 30a...p or 34a....p that are removable is 5/9=0.555... In general, the proportion is

$$\frac{n}{C/F} = \frac{nF}{C} = \frac{(2N-3)F}{(2N-3)(2F-1)} = \frac{F}{(2F-1)} \to 0.5 \text{ as } F \to \infty \qquad \textbf{eqtn. 43}$$

Larger proportions of time-domain switches can be removed from either time-domain aggregation switching stages 30, 34 logically adjacent to the spatial switching stage

18 of the switch arrangement if the number nF of requests available is increased to cover requests to be matched in more of the recursive steps, such that m≠n.

COMPUTING STEPS

In general terms, for large N and F, the number of steps required for recursive multi-slotting is

$$O\left(\log_2(F)\right) \qquad \text{eqtn. 44}$$

For each recursive step, at most each matching takes $O(N^2)$ sequential computing steps using the above heuristic algorithm and each path search (if implemented for each recursive step) would take $O(N^2 \log_2 N)$ computing steps (using Andresen's sequential adaptation of the looping algorithm as described in the paper referenced above by Andresen). Parallel processing techniques can reduce these computing steps.

Path searching dominates the computing time. For frame-based scheduling, the overall computing steps per frame for assigning specific time slots to the time-domain aggregation switches in aggregation switching stages 30, 34 and hence to the spatial switches in the global spatial switching stage 18 are therefore

$$O\left(N^2 \log_2(F) + N^2 \log_2 N \log_2(F)\right) \qquad \text{eqtn. 45}$$

With a frame duration of F time slots, the overall computing steps per time slot become

$$O\left(\frac{N^2}{F}\log_2(F) + \frac{N^2}{F}\log_2 N \log_2(F)\right) \qquad \text{eqtn. 46}$$

Therefore by using a small number N of aggregations in conjunction with a sufficiently large frame size F (which can be of the same order as the total number of switch or network ports LN), the computing steps per time slot can be made acceptably low. The computing steps for path searching using Andresen's algorithm could be reduced by employing parallel processing. This could be achieved either by using Andresen's own method of parallel processing or by using separate processor(s) for path-searching of each recursion (pipelining). In the limit the computing steps per time slot can be reduced to

$$O\left(\frac{N^2}{F}\log_2(F)\right) \qquad \text{eqtn. 47}$$

In embodiments where the number of input/output aggregations N is chosen to be relatively small, so that the number of computing steps can become acceptable, each of the ingress or output aggregations contains a relatively large number of line-cards and time-domain aggregation switches. The size of space switch required within each input/output aggregation to interconnect the line-cards (time-domain aggregation switches) must therefore become larger than the size of the middle-stage switches needed to interconnect the input aggregations to the output aggregations.

This asymmetry could become such that the space switches within the ingress and output aggregations are decomposed into 3-stage switches, if suitable large single-stage space switches do not exist. Such 3-stage switches would themselves require additional path-searching to be performed for every time slot. Although the number of path searches required across all input/output aggregations could be large (2NF), the number of computing steps required for each path search would be quite small (e.g. $O((C/F).\log_2 N)$ using Andresen's algorithm).

Such a computational task can be achieved using an acceptable number of processors performing the path searches simultaneously, in parallel with each other, with each processor performing a number of path searches sequentially.

RECURSIVE MULTI-SLOTTING APPLIED TO INPUT/OUTPUT AGGREGATIONS

In embodiments where computing requirements are acceptably low, rearrangeably non-blocking algorithms can be employed to perform the channel assignment in the input/output aggregations. This ensures that no further speed-up is required within the input/output aggregations. However, if the resulting computing steps are unacceptably high, then the principles of recursive multi-slotting can also be employed for path searching within the input/output aggregations, to reduce the computing steps needed for rearrangeably non-blocking algorithms. In this embodiment, there is no requirement to remove any switching stages as none are made redundant in the way that the switches in the redundant time-domain aggregation switching stage can be removed when assigning channels across the

spatial switching stage 18 of the switch arrangement 10 shown in Figure 4. When multi-slotting is applied in a recursive way internally within each input/output aggregation, it can be applied either just to blocked time-slotted traffic segments or to all time-slotted traffic segments at the outset regardless of whether they are blocked or not.

SPEED-UP

If recursive multi-slotting is only applied to blocked time-slotted traffic segments, so that blocking is allowed but then subsequently time-slotted traffic segments are unblocked, further speed-up will result in the input/output aggregations. In some embodiments where recursive multi-slotting is employed and no blocking is allowed, no further speed-up is incurred if the frame size is large enough to satisfy the non-blocking constraint.

The table below indicates the effective "speed-ups" possible in the limit with various combinations of types of embodiment of recursive multi-slotting as described herein, for large switches with multistage space-switch fabrics. It is assumed that when blocking is allowed, each recursion could suffer a maximum of 25% blocking, and when blocking is not allowed, the non-blocking constraint described herein can prevent all blocking.

type of embodiment		line-card "speed-up"	Spatial switch "speed-up"
Spatial switch time-slots	input/output aggregation time-slots		
blocked	blocked	(1+1).(4/3).(4/3)=3.555 ...	2.(1/3).(4/3).(4/3)+ (1/3).(4/3)=1.63
blocked	all	(1+1).(4/3)=2.666...	(1/3).(4/3)+ (1/3).(4/3)+ (1/3).(4/3)=1.333...
all	blocked	(1+1).(4/3)=2.666...	(1/3).(4/3)+ (1/3).1+ (1/3).(4/3)=1.222...
all	all	(1+1)=2.0	(1/3)+ (1/3)+ (1/3)=1.0

Table 1. Effective speed-ups believed to be possible in the limit with various combinations of recursive multi-slotting applied to blocked or all time slots.

COMPUTING STEPS

If recursive multi-slotting is applied to blocked time-slotted traffic segments, and blocking is allowed then subsequently unblocked, the non-blocking constraint does not need to be applied, but the resulting "speed-ups" are relatively large (as indicated in Table 1 above). But if recursive multi-slotting is applied to all time-slotted traffic segments at the outset and no blocking is allowed, the non-blocking constraint required to achieve lower "speed-ups" forces the number L (=C/F, see eqtn. 33) of time-domain switches (e.g., TSIs) in each time-domain aggregation switching stage per input or output aggregation to be relatively larger than the number N of input or output aggregations. But the L time-domain switches (e.g.TSIs) in each input/output aggregation now represent logical aggregations themselves when multi-slotting is applied within the input/output aggregations.

Thus by replacing N with L in eqtn.55, the number of computing steps required when applying multi-slotting within each input/output aggregation 16, 20 of the switch arrangement is greater than when applying multi-slotting across the spatial switching stage 18 of the switch arrangement 10 shown in Figure 4. This is the penalty for reducing the "speed-up". The processor speeds could be brought down to acceptable values by, for example:

a) further increasing the frame length F;

b) employing multiple processors to compute path searches for each recursion simultaneously (in parallel), after each multi-slot matching has been performed sequentially.

In addition, it is possible to take the same measures when applying multi-slotting across the spatial switches 18a,b,c,d, of the spatial switching stage 18 of the switch arrangement.

PHYSICAL PROCESSOR IMPLEMENTATIONS

There are 3 major aspects of the invention for which physical processor

implementations are required to force the time-domain multi-slot constraint:

1) Non-recursive multi-slotting allowing blocking
2) Recursive multi-slotting for blocked time-slotted traffic segments
3) Recursive multi-slotting for all time-slotted traffic segments at the outset

NON-RECURSIVE MULTI-SLOTTING ALLOWING BLOCKING

To force the time-domain multi-slot constraint and assigning time slots to time-slotted traffic segments throughout the switch arrangement the following steps may be implemented:

a) compute request matrix between input/output aggregations, and assign time-slotted traffic segments to time slots at the inputs and outputs of the switch arrangement;

b) determine numbers of potential multi-slots filled with as many requests as possible;

c) match potential multi-slots between input and output aggregations while maximising as much as possible the numbers of time-slotted traffic segments in the chosen multi-slots;

d) assign particular time-domain switches in the time-domain aggregation switching stages 30, 34 logically adjacent to spatial switching stage 18 of the switch arrangement 10 (as shown in Figure 4), and hence also assign the spatial switches 18a...d to the accepted multi-slots;

e) assign particular time-slots to traffic segments within each aggregation by deciding which particular time-slotted traffic segments to block and by assigning time-slotted traffic segments to particular multi-slots.

Some embodiments of the invention may further implement one or more steps to:

f) re-assign time slots in which time-slotted traffic segments leave the time-domain switches 26a...p in the time-domain aggregation switching stage 26 logically adjacent to the inputs of the switch arrangement (assuming that the

final time-domain switching stage 38 logically adjacent to the outputs of the switch arrangement is to be removed);

g) re-assign the time slot in which each time-slotted traffic segment leaves its time-domain switch (e.g. 30a or 34a) in the time-domain aggregation switching stage logically adjacent to the input or output of the spatial switching stage 18 of the switch arrangement;

h) switch blocked time-slotted traffic segments with priority in the next frame or switch blocked and non-blocked time-slotted traffic segments in the current frame.

The following examples will be based on an embodiment in which the switch arrangement 10 shown in Figure 4 provides a logical structure for which the channel assignment constraint is imposed. The following examples will be described in terms of a distributed switch arrangement arranged to switch time-slotted traffic segments comprising a cell or packet switching network etc. In this embodiment, each input aggregation and output aggregation represents as a sub-network of the switch arrangement, and the spatial switching stage 18 of the switch arrangement 10 represents an appropriately configured hub switch. Thus in the embodiment described below hub switch 18 is configured to switch traffic from one sub-network to another sub-network in accordance with the constrained channel assignment process.

The implementation of each step outlined above will now be described.

a) Request Matrix Between Input and Output Aggregations

In this embodiment, each time-domain switch in each input and output aggregation comprises a time-slot interchanger (TSI). TSIs 26a,b,c,d in input aggregation 16a are associated with a line-card providing ingress to the switch arrangement and having an associated processor. Each TSI 38a,b,c,d in an output aggregation 20a is associated with an egress line-card, and each egress line-card has a processor associated with it. (Those skilled in the art will appreciate that alternatively, the same processor component may be provided to implement processing of on both the ingress and egress line-cards.)

The ingress/egress processors could assign individual time-slotted traffic segments to the ingress and egress time slots (inputs and outputs) of the ingress and egress line-cards, respectively, in many ways. For example, the processors could simply assign individual time-slotted traffic segments in numerical order to the ingress and egress time slots (inputs and outputs) of the ingress and egress line-cards, respectively.

By associating a processor with each line-card in this way, the time-slotted traffic segments can be assigned in just O(F) steps, i.e. computing time. This is preferred, since it represents only O(1) computing step per time slot within a frame. Alternatively, if a processor were associated with each aggregation (i.e., each sub-network) instead of each input/output (e.g. each line-card), it would take O(LF) steps to assign LF time-slotted traffic segments across L line-cards. Alternatively, if a processor were associated with the entire network, then it would take O(LNF) steps to assign LNF time-slotted traffic segments across LN line-cards in N sub-networks. Computing times between these extremes could be obtained by associating each processor with any number of line-cards.

One way in which processors can compute the rows and columns of the traffic request matrix between ingress and egress sub-networks is the following. Each line-card processor converts the destination (egress) line-card addresses of each of its time-slotted traffic segments previously accepted for switching in the frame to an address for the destination (egress) sub-network.

If the number of line-cards per sub-network and the number of sub-networks are both integer powers of 2 and the addresses are represented as binary words, then a simple way to do this is to mask the binary word and take the $\log_2 N$ most significant bits of the line-card address. If the number of sub-networks is not an integer power of 2, then the number of most significant bits taken is the next integer greater than $\log_2 N$. This takes O(F) computing steps.

Each line-card processor counts the number of time-slotted traffic segments destined for each egress sub-network, also in O(F) steps. A processor associated with each sub-network, which could be one of the processors associated with the line-cards, then adds the counts to each egress sub-network from each of the line-card processors within its sub-network, which takes O(LN) sequential computing steps.

At this stage, each processor associated with an ingress sub-network possesses one row of the resulting sub-network-to-sub-network request matrix.

b) Determine Numbers of Potential Multi-Slots

The numbers of potential multi-slots needed to fulfil the requests and the number of requests within each potential multi-slot must be determined. A set of rules for processors to use when forcing the multi-slot constraint has already been described herein above.

c) Match Potential Multi-Slots Between Ingress and Egress Sub-Networks

One or more processors must match the potential multi-slots between ingress and egress sub-networks, while maximising as much as possible the numbers of time-slotted traffic segments in the chosen multi-slots. One possible physical processor implementation for this is described for one embodiment of a heuristic matching algorithm, called Prioritised Single Selection No-Overbooking as described in the paper by Bianco et al, referred to herein above.

A processor is associated with each ingress and egress sub-network. In effect this associates a processor with each row and each column of the request matrix.

Each ingress sub-network processor transmits the number of requests from its associated ingress sub-network to each egress sub-network processor associated with each egress sub-network. This requires at least $N^2 \log_2 F$ bits to be transmitted per frame.

One possible way of transmitting the N^2 numbers between ingress and egress sub-network processors would be across the switching network itself. This could be done, for example, in parallel for all N^2 numbers in just one time slot (assuming all necessary space switches within the switching network operate at the time-slot rate), in the case when L=N, as follows. Each ingress sub-network processor transmits sequentially its N numbers of requests destined for each egress sub-network to a different ingress line-card within its associated sub-network. Each line-card within an ingress sub-network transmits its number of requests to one line-card within each egress sub-network, in such a way that each line-card within an egress sub-network receives a number from one line-card within each ingress sub-network. Each egress line-card within each egress sub-network then transmits its received number to the

associated egress sub-network processor, such that all N numbers are received sequentially by the processor.

Each ingress sub-network processor determines the priority for its sub-network, by calculating its row-sum. Each egress sub-network processor determines the priority for its sub-network, by calculating its column-sum. A processor associated with all ingress sub-networks, which may be one of the processors associated with each ingress sub-network, finds the row having the highest priority. This could be performed by a linear search, or perhaps by sorting the priorities. A processor associated with all egress aggregations, which may be one of the processors associated with each egress aggregation, finds the column having the highest priority. One of these two processors decides whether the highest priority row or highest priority column has the higher priority. Whichever row (and corresponding ingress aggregation) or column (and corresponding egress aggregation) it is, this one is chosen to have one multi-slot matched within it.

One possible way to match a multi-slot within a row or column of the request matrix would associate a processor with each request matrix element of that row or column, i.e. to each request from an ingress aggregation to all egress-subnetworks and to each request to an egress aggregation from all ingress aggregations. In practice, if L=N, these could be the same set(s) of processors that are associated with the ingress and egress line-cards. More would be needed if N>L. The Prioritised Single Selection No-Overbooking algorithm must select the request matrix element having the largest number of requests starting from a pointer, consistent with the number of multi-slots in both rows and columns not exceeding the allowed number. To do this, the request matrix elements can firstly be sorted in terms of request size, then re-ordered after a multi-slot has been matched and the remaining number of requests in that matrix element has been calculated. Initial sorting, which is performed only once, could be performed in $O(N\log_2 N)$ computing steps by the sub-network processor, which then transmits results to each matrix element processor. Re-ordering could be computed by the sub-network processor and performed by the matrix element processors. Since re-ordering of matrix element request sizes is performed with only one matrix element per row and column reduced in value (either reduced by the size of the multi-slot or reduced to zero), re-ordering need take only $O(\log_2 N)$ computing steps to find the correct new matrix element processor position for the reduced request size.

This is not just a matter of finding the correct position in terms of size. If there are request values of the same size as the reduced request size, then the correct position must also take into account the original position of the matrix element with respect to the pointer for its row or column. All matrix elements having the same value in a row or column must be ordered in terms of their position relative to the pointer. Once the correct new position is found by the sub-network processor, this instructs the appropriate matrix element processors to shift their request value to their neighbouring matrix element processor in the direction of increasing size in O(1) step, in such a way as to remove the reduced request size (if necessary) from its initial matrix element processor, close up the gap, open up a new gap in the correct new processor position and insert the reduced request size into the new gap. Obviously, therefore, not all matrix element processors are instructed to shift their request value to their neighbour.

Once a multi-slot has been matched in a row (or column), the location of the selected request matrix element and the remaining number of requests (value) of the element are transmitted to the corresponding column (or row) processor. Row and column processors now up-date the number of multi-slots assigned to the request matrix element and its remaining number of requests. They also calculate remaining row-sums and column-sums, i.e. the up-dated priorities, as well as the total number of multi-slots assigned in their row or column. If this number is the maximum allowable number, the corresponding request matrix element is removed from the matrix element processors in that row and/or column and the request values are re-ordered (closed up) on the matrix element processors in that row and/or column. The processor associated with all ingress sub-networks and the processor associated with all egress sub-networks find the row and column respectively having the highest priority. Since there are LN multi-slots to be matched altogether, it is preferable for highest priorities to be found by means of binary searches in $O(\log_2 N)$ steps, requiring $O(LN\log_2 N)$ computing steps altogether to match all multi-slots. It is therefore preferable for the priorities to have been sorted initially, so they can be re-ordered after each multi-slot has been matched. This could be achieved in a similar way to the re-ordering of the request matrix elements above. Either the processor associated with all ingress sub-networks or the one associated with all egress sub-networks decides whether the highest priority row or highest priority column has the higher priority, ready for the next multi-slot to be matched.

d) Assign Particular TSIs, and Hence Middle-Stage Space Switches, to the

Accepted Multi-Slots

Preferably a rearrangeably non-blocking, path-searching algorithm such as Andresen's algorithm is used, because this does not require any additional resources (in terms of time-domain switches in aggregations switching stages 30, 34 or spatial switches in global spatial switching stage 18) as Clos' strictly non-blocking algorithm would require to prevent blocking. Andresen's algorithm applies the looping algorithm for a multi-stage Benes network to a 3-stage Clos network when an integer power of 2 switch inlets and outlets are terminated on each 1st- and 3rd-stage switch of the Clos network.

The realisation that a path-searching algorithm can be used to assign time-domain aggregation switches and global spatial switching-stage switches to the accepted multi-slots is inventive, as are the following appropriate mappings for using Andresen's algorithm:

- input aggregations(i.e., input sub-networks) map to 1st-stage switches in the 3-stage Clos network
- output aggregations(i.e., egress sub-networks) map to 3rd-stage switches in the 3-stage Clos network
- multi-slots map to inlets and outlets on these 1st- and 3rd-stage switches, respectively
- real spatial switches in the global spatial switching stage of the 7-stage logical architecture (and hence the time-domain aggregation switches 30a..p, 34a..p to which they are connected) map to the middle-stage switches in the 3-stage Clos network.

The processor requirements for Andresen's algorithm are known. It can be run on a single sequential processor in $O(LN\log_2(LN))$ computing steps, in our switching network case. Alternatively, parallel processors can be used to reduce to O(LN) computing steps.

e) Assign Particular Time-Slots to Time-slotted traffic segments Within Each Outer Sub-Network

There are three procedures to be implemented:

i. Decide which particular time-slotted traffic segments to nominate as blocked time-slotted traffic segments
ii. Assign particular time-slots to the non-blocked time-slotted traffic segments through the middle-stage switches
iii. Path-search (i.e. assign particular time-slots) through the space switches within the outer (ingress and egress) sub-networks

i) Decide which particular time-slotted traffic segments to nominate as blocked time-slotted traffic segments

The following processor implementation ensures that the appropriate VOQs having the most accepted time-slotted traffic segments are selected for filling the multi-slots, leaving the remaining time-slotted traffic segments to be blocked. Each ingress sub-network has a set of processors associated with it, each of which is associated with a different one of its ingress line-cards, and a set of processors associated with it, which may be the same set of processors, each of which is also associated with all the requests going from all of the line-cards within the ingress sub-network to a different one of the egress sub-networks; i.e. the second set can correspond to the request matrix element processors that have already been used.

Firstly, the ingress line-card processors transmit the numbers of time-slotted traffic segments in each of their VOQs, which have been accepted for switching by the previous matching phase of the overall scheduling procedure (i.e. not the multi-slot matching), to the request matrix element processors associated with their corresponding ingress sub-networks (i.e. associated with the corresponding row of the request matrix). They may also transmit the VOQ identity in association with the quantity of time-slotted traffic segments accepted, unless this is inferred by the request matrix element processors during transmission, for example by means of a pre-arranged order in which transmissions are undertaken. Each ingress line-card processor may have up to LN VOQ numbers to transmit to the request matrix element processors associated with its own ingress sub-network. Therefore each ingress sub-network may have as many as L^2N VOQ numbers to transmit between all of its ingress line-card processors and its request matrix element processors. One possible way to transmit all of these within an acceptably short time would be to transmit from all ingress line-card processors simultaneously, in parallel, each to a different request matrix element processor, then change the pairings of ingress line-card processor and request matrix element processor in a number of transmission

steps, such that each ingress line-card processor transmits to all request matrix element processors (if required). The connections between ingress line-card processors and request matrix element processors could be switched between the transmission steps by means of the 2nd-stage space switches of the 7-stage logical architecture (although the switch must have L inputs and N outputs, so would need to have more outputs if N>L). This means that for the duration of these transmission steps the 2nd-stage space switches would not be able to perform their primary function of switching time-slotted traffic segments between ingress line-cards (i.e. 1st-stage TSIs) and 3rd-stage TSIs. Each ingress line-card processor may need to transmit up to L VOQ numbers to each of N request matrix element processors in N transmission steps. If VOQ identities do not also need to be transmitted, the total sequential transmission time is $LN\log_2 F$ bits. For a time-slotted traffic segment switch having LN=4,096 line-cards at 10 Gbit/s, a frame duration F=LN=4,096 time slots of 51.2 nsec each, this would take 96 time slots, i.e. only 2.3% of the switching network capacity. Alternatively, if different, additional space switches were used, there would be no loss of switching time for time-slotted traffic segments, i.e. network efficiency.

Each request matrix element processor now possesses up to L^2 numbers, each representing the number of time-slotted traffic segments in a VOQ accepted for switching to the corresponding egress sub-network. Each request matrix element processor may have to fill up to N multi-slots with time-slotted traffic segments from up to L^2 VOQs. Each request matrix element processor must choose the longest available VOQs, i.e. the VOQs with the most accepted time-slotted traffic segments, to fill the multi-slots. The L^2 numbers in each request matrix element processor must therefore be sorted into size order, in conjunction with their VOQ identities, taking $O(L^2\log(L^2))$ sequential computing steps. It will then take up to $O(L^2)$ sequential steps in each request matrix element processor (performed in parallel with the others), to choose from up to L^2 VOQs, in size order, to fill up to N multi-slots of up to F time-slotted traffic segments each, identifying the VOQ identities and the number of time-slotted traffic segments from each VOQ chosen for each multi-slot.

Each request matrix element processor must then transmit back to the ingress line-card processors the identities of the multi-slots, the identities of the VOQs chosen for each multi-slot, and the number of time-slotted traffic segments within each VOQ chosen for each multi-slot. This can be achieved in a similar manner to the transmission steps from ingress line-card processors to request matrix element processors.

Finally, for each of its VOQs, each ingress line-card processor must record the number of time-slotted traffic segments from the head of line that have been chosen. Accepted time-slotted traffic segments beyond that number, which have not been chosen, will be blocked in the frame. This means that they will either not be switched in the frame, but be given priority in the next frame, or will be switched in the current frame via additional 3rd-stage TSIs, middle-stage space switches and 5th-stage TSIs, without the use of multi-slots.

ii) The particular time-slots are then assigned to the non-blocked time-slotted traffic segments through the middle-stage switches in the manner described above.

iii) Assign particular time slots through the space switches within the outer (ingress and egress) sub-networks

Either the parallel Clos path-searching algorithm or a parallel version of Andresen's rearrangeably non-blocking algorithm are preferred. The former algorithm would require the number of time slots within each ingress and egress sub-network to be almost doubled, because strict non-blocking is required for the algorithm to work, whereas the latter algorithm would not.

<u>f) Re-Assign the time slots in which time-slotted traffic segments leave TSIs 30a,...,p in aggregation switching stage 30 if TSIs 38a,...,p are removed in aggregation switching stage 38</u>

Time slots in which particular time-slotted traffic segments leave the 1st-stage TSIs may need to be re-assigned if the 7th-stage TSIs are to be removed and time-slotted traffic segment mis-sequencing is to be prevented. This is described in A30295 and its Cognate.

<u>g) Re-Assign the Time Slot In Which Each Time-slotted traffic segment Leaves Its TSIs (30a...p or 34a...p) in a time-domain aggregation switching stage adjacent to the spatial switching stage 18 order to remove the corresponding time-domain aggregation switching stage TSIs (34a...p or 30 a...p) respectively</u>

In order to remove a 5th-stage TSI, some time-slotted traffic segments switched through the corresponding 3rd-stage TSI in a frame may need to have their departing

time slots to their middle-stage space switch re-assigned, so that they correspond to the time slots assigned within the egress sub-network. Alternatively, in order to remove a 3rd-stage TSI, some time-slotted traffic segments switched through the corresponding 5th-stage TSI in a frame may need to have their arriving time slots from their middle-stage space switch re-assigned, so that they correspond to the time slots assigned within the egress sub-network.

A possible way for processors to implement this re-assignment in parallel, for the case where the parallel Clos path-search algorithm is used to assign time slots within the ingress and egress sub-networks, is as follows. When a 5th-stage TSI is to be removed, a processor associated with that logical TSI transmits the time slot identities assigned to its time-slotted traffic segments within the corresponding egress sub-network to a processor associated with the corresponding 3rd-stage TSI. The latter processor assigns these time slots to the time-slotted traffic segments departing from the 3rd-stage TSI, then computes the internal switch settings for that 3rd-stage TSI. Alternatively, When a 3rd-stage TSI is to be removed, a processor associated with that TSI transmits the time slot identities assigned to its time-slotted traffic segments within the corresponding ingress sub-network to a processor associated with the corresponding 5th-stage TSI. The latter processor assigns these time slots to the time-slotted traffic segments arriving at the 5th-stage TSI, then computes the internal switch settings for that 5th-stage TSI. Thus O(LN) processors can transmit the time slot identities in parallel, with $O(F\log_2 F)$ sequential bits. These could be transmitted through the switching network itself.

1.1 Switch Blocked Time-slotted traffic segments With Priority In The Next Frame

If blocked time-slotted traffic segments are to be switched with priority in the next (or a subsequent) frame, this implies that the blocked time-slotted traffic segments will make use of multi-slots in the next frame. Otherwise, if they are to be switched via additional TSIs and middle-stage switches, they might as well have been switched in the current frame. Before the multi-slot matching algorithm is run in each frame, any sub-network-to-sub-network request matrix element that had time-slotted traffic segments blocked in the previous frame must firstly be initialised by assigning to it the appropriate number of multi-slots to support its number of blocked time-slotted traffic segments. This could be performed, for example, by the ingress (egress) sub-network processors in parallel for each row (column) of the request matrix. Alternatively, the request matrix element processors could be used in parallel.

1.2 Switch Blocked And Non-Blocked Time-slotted traffic segments In The Current Frame

Additional TSIs are associated with each ingress and egress sub-network and interconnected by additional middle-stage space switches. This means, for example if the 5th stage of TSIs has been removed, that the 3rd stage of TSIs of each ingress sub-network will have more TSIs than the 1st stage of TSIs. Therefore the 2nd-stage space switch within each ingress sub-network requires more output ports than input ports, in order to connect time-slotted traffic segments to the additional TSIs. Similarly, in the egress sub-networks, although TSIs may have been removed, their space switches will still require more input ports than output ports to accommodate the additional TSIs in the 5th stage.

Non-blocked time-slotted traffic segments are identified and assigned to multi–slots as previously described. They are also assigned as previously described to time slots through the middle-stage space switches. However, they are not yet assigned to particular time slots within the ingress and egress sub-networks, because time-slot assignment within the ingress and egress sub-networks must be performed during the same path-searching procedure for all time-slotted traffic segments, both non-blocked and blocked. Before this can be done, the blocked time-slotted traffic segments remaining from the multi-slot matching procedure must be assigned to time slots going through the additional middle-stage space switches via the additional TSIs that are associated with each sub-network for switching the blocked time-slotted traffic segments. If the number of blocked time-slotted traffic segments is very large, the Clos strictly non-blocking path-search algorithm could be used for this, which requires the number of additional TSIs and middle-stage switches in essence to be doubled if we wish to prevent further blocking. (Smaller quantities may be acceptable, if some lower level of further blocking were achievable and allowable for switching with priority in the next frame). If the number of blocked time-slotted traffic segments is sufficiently small, a rearrangeably non-blocking path-search algorithm could be used, thus preventing further blocking.

Time-slot assignment within the ingress and egress sub-networks, for both non-blocked and blocked time-slotted traffic segments together, could also be performed by either the Clos algorithm or a rearrangeably non-blocking algorithm.

Implementation of Recursive Multi-Slotting For Blocked Time-slotted traffic segments

The principle of recursive multi-slotting applied to blocked time-slotted traffic segments is that either additional hardware (3rd- and 5th-stage TSIs and middle-stage space switches) or additional time slots within the initial hardware are used so that multi-slotting can be performed in a number of recursive steps, each time on the remaining blocked time-slotted traffic segments from the previous recursion. If, for example, the number of time slots within the multi-slot is reduced in each recursion, then the number of recursions required can be finite. A possible reduction factor is 4, for which the number of recursive steps is a logarithmic function of the initial number of time slots within the multi-slot. However, it may be necessary to employ a size of multi-slot more than once in the recursive steps. Furthermore, some recursions may require no reduction in the size of the multi-slot from the previous recursion. It is also possible for the multi-slot sizes to be applied in an arbitrary order in the recursive steps, although this is not preferable. Recursive multi-slotting eliminates or reduces the number of blocked time-slotted traffic segments.

There is nothing different in principle involved in recursive multi-slotting of blocked time-slotted traffic segments, apart from the size of the multi-slots and application to remaining time-slotted traffic segments blocked in the previous recursion, with respect to physical processor implementation.

Recursive Multi-Slotting For All Time-slotted traffic segments At The Outset

Important differences in physical processor implementation between non-recursive multi-slotting and recursive multi-slotting applied to time-slotted traffic segments at the outset are as follows:

a) calculate numbers of matrix elements available for matching
b) calculate order of rows and columns for matching
c) match up to n multi-slots in a row or column of the request matrix
d) assign multi-slots from each recursion to particular middle-stage switches

Each recursive step could employ multi-slots of any size (any number of time slots). However, in order to ensure a small number of recursive steps, it is beneficial to use a range of sizes. One embodiment reduces the number of time slots in each recursion by a factor 2, which beneficially provides just $O(\log_2 F)$ recursive steps.

1.3 Calculate Numbers of Matrix Elements Available for Matching

The priorities with which the rows and columns should be matched are represented

by the number of sub-network-to-sub-network request matrix elements in each row and column that are available for matching at least one multi-slot of the size (i.e. number of time slots) used in the current recursive step. The smaller the number available (except zero), the higher the priority. Zero availability means that matching of multi-slots should not be performed in the row or column.

A processor is associated with each ingress sub-network and a processor associated with each egress sub-network. In parallel (i.e. simultaneously), each ingress (egress) sub-network processor calculates the number of available matrix elements in its respective row (column) that are capable of matching at least one multi-slot of the size used in the current recursive step. This requires only O(N) sequential computing steps per recursion, i.e. $O(N\log_2 F)$ steps altogether.

1.4 Calculate Order of Rows and Columns For Matching

The order in which rows and columns should be matched in the current recursive step can be computed by a single processor associated with the entire switching network. This could be one of the ingress or egress sub-network processors.

Firstly, the ingress and egress sub-network processors must transmit their available numbers of matrix elements, as well as the corresponding row and column identities (unless the identities are pre-arranged to be implicit in the time sequence of the transmissions), sequentially to the single network processor. This processor then sorts the row and column identities in terms of their number of available request matrix elements, which represents the order in which rows and columns should be matched. Transmission requires $O(N\log_2 N)$ bits per recursion and therefore $O(N\log_2 N\log_2 F)$ bits altogether. Sorting requires $O(2N\log_2(2N))$ sequential computing steps.

1.5 Match Up To n Multi-Slots In A Row Or Column of The Request Matrix

When a row or column of the request matrix is to be matched, up to n multi-slots may need to be assigned to the matrix elements. Altogether $O(nN\log_2 F) = O(N^2\log_2 F)$ multi-slots must be matched in all rows and columns, by performing matching in rows and columns $O(N\log_2 F)$ times. It is important therefore that the number of computing steps required to match up to n multi-slots within a row or column should be as low as possible. Although sorting of request matrix elements in a row or column could be employed in the matching procedure, the need to re-order after every individual multi-slot has been matched, or to re-sort after up to n multi-slots have been matched,

would require a total of $O(N^2 \log_2 F \log_2 N)$ computing steps altogether, which is rather large. In order to reduce the computing steps, a system of variable pointers is preferred. This requires a total of $O(N^2 \log_2 F)$ computing steps to match all multi-slots.

The system of variable pointers is as follows. Each ingress (egress) sub-network processor maintains a set of 2N variable pointers for its associated row (column) of the request matrix. For each row and column 2 variable pointers are associated with each one of the request matrix elements; a "previous" pointer points to the previous nearest matrix element back along the row or column that is able to be assigned a multi-slot and the "next" pointer points to the next nearest matrix element forward along the row or column that is able to be assigned a multi-slot. Each ingress (egress) sub-network also maintains i) records of the total number of multi-slots already assigned within its own row (column), ii) the total number of multi-slots already assigned within the columns (rows) in which each of its matrix elements also exists and iii) the remaining number of accepted requests possessed by each request matrix element.

A sub-network processor assigns multi-slots in its row or column according to the following procedure. The variable "previous" and "next" pointers are initialised such that each one points appropriately to the previous (next) nearest request matrix element back (forward) along the row or column, starting from the fixed pointer marked p and folding back from the end of a row or column to the start of the row or column. The sub-network processor chosen to assign multi-slots within its row or column now begins at the request matrix element identified by the fixed pointer p, and attempts to assign one multi-slot of the size (number of time slots) being used in the current recursive step. This is possible if a) the number of multi-slots already matched in either the row or column in which the matrix element exists is less than n and if b) the number of accepted time-slotted traffic segments represented by the value of the request matrix element is equal to or greater than the current size of the multi-slot. If successful, the ingress (egress) sub-network processor calculates the remaining number of time-slotted traffic segment requests in the matrix element, transmits the identity of the column (row) of the chosen matrix element to all other ingress (egress) sub-network processors and to the corresponding egress (ingress) sub-network processor for up-dating. Whether successful or not, if the remaining number of accepted time-slotted traffic segment requests in the matched matrix element is now less than the size of the multi-slots in the current recursion, or if the number of multi-slots already matched in either the row or column in which the matrix

element exists is now less than n, the ingress (egress) sub-network processor updates the "next" pointer of the matrix element pointed to by the "previous" pointer of the matrix element that has just been dealt with in the row (column) to point to the matrix element pointed to by the "next" pointer of the matrix element just dealt with. The sub-network processor now moves to the matrix element pointed to by the "next" pointer associated with the matrix element just dealt with. This procedure is repeated, in general terms, in the forward direction along the row or column until all possible multi-slots have been assigned to matrix elements, using the variable "next" pointers to move to the next possible matrix element. By this procedure, unavailable matrix elements can be ignored (missed out), thus ensuring that up to n multi-slots can be assigned in a row or column in only O(N) computing steps.

1.6 Assign Multi-Slots From Each Recursion To Particular Middle-Stage Switches

Assignment of multi-slots from each of the recursive steps to particular middle-stage switches, and hence to particular 3rd-stage and 5th-stage TSIs, is similar to the procedure for non-recursive multi-slotting, but multi-slots of different sizes must be assigned for the different recursions (although some of the recursions may use the same size). Any known path-searching algorithm can be used to assign multi-slots to middle-stage switches, but a rearrangeably non-blocking one, such as Andresen's adaptation of the looping algorithm, is desirable in order not to need additional middle-stage switches and TSIs to prevent blocking. Path-searching could be performed separately for each recursion or possibly at the end for all recursions simultaneously.

The particular recursive step in which each size of multi-slots is matched could be chosen arbitrarily. Preferably, however, the largest multi-slots are matched in the first recursion and the size of multi-slots reduces monotonically in each subsequent recursion, unless the size is the same (e.g. 1 time slot). Generally n multi-slots are to be matched in each ingress and egress sub-network in each recursion, but some recursions may need to match less than n multi-slots per sub-network. When path-searching is performed separately for each recursion, the order in which the different multi-slot sizes are path-searched need not be the same as the order in which they are matched, but it is preferable for the order to be the same. This is particularly true if the time taken for path-searching of the multi-slots from each recursion is long, in which case it would take an even longer time if the first recursion were not path-searched immediately, first. If path-searching times for each recursion are long, then multiple sets of path-searching processor(s) can path-search multi-slots from

different recursions in parallel, to reduce the total path-searching time.

When path-searching is performed separately for each recursion and, for example, the first recursion uses a multi-slot size equal to the number of time slots in a whole TSI, then an arbitrary sub-set of the TSIs in the 3rd and 5th stages, as well as middle-stage switches to which they are connected, are assigned for time-slotted traffic segments using the multi-slots from the first recursion. Since the multi-slots fill whole TSIs, some TSIs in the 3rd and/or 5th stages need not be physically implemented. The multi-slots matched in the first recursion are assigned to particular middle-stage switches by the path-searching algorithm. In the case of Andresen's algorithm, it is known that this could be implemented sequentially on a single processor, or a mixture of sequential and parallel processing on multiple processors. The same is true for Clos' strictly non-blocking algorithm. A parallel processor implementation of Clos' algorithm is already known in the art, for example, in WO 01/67802 entitled "Packet Switching" the contents of which are hereby incorporated by reference. (N.B. the first recursion could have multi-slots with more time slots than a whole TSI has, for example equivalent to an integer multiple of whole TSIs).

In subsequent recursions, the multi-slots have less time slots. In recursions where the multi-slots have less time slots than a whole TSI (hence less than a middle-stage switch can support), each TSI can support multiple multi-slots by arbitrarily assigning sub-sets of its time slots to different multi-slots. In each of such subsequent recursions, therefore, i) a sub-set of the TSIs in each sub-network is arbitrarily assigned to support the required number of smaller multi-slots and ii) path-searching is employed to assign the smaller multi-slots to the sub-sets of time slots within the TSIs (and hence within the middle-stage switches).

Although in the above embodiment reference is made to aggregations comprising sub-networks, and time-domain switches comprising TSIs, and middle-stage switches (which can comprise an spatial switching stage for example), etc, those skilled in the art will appreciate that the specific nature of the components described does not limit to the scope of the invention.

EXAMPLES OF PHYSICAL SWITCH ARRANGEMENTS

The channel assignment process according to the invention can be used to assign paths through switch arrangements which are relatively large, for example, switching

networks. Typically, these may have over 1000 inputs/outputs and may need to have throughputs of the order of several tera-bits per second.

The constrained channel assignment process is suitable for switching arrangements comprising just five switching stages, for example, a (TST)S(S) switching stage such as Figure 15 shows. The used of brackets here denotes that the switching stage is logically associated with a subset of the total inputs or outputs of the switch arrangement. The lack of brackets around the spatial switching stage 18 indicates that this is a global spatial switching stage capable of receiving and passing traffic from all and to all points of ingress to and egress from the switch arrangement.

The 5-stage (TST)S(S) switch arrangement 1 shown in Figure 15 comprises three spatial switching stages , one global (switching stage 18) and two aggregate (28, 36). The global spatial switching stage 18 is implemented by a plurality of wavelength switches 18a...d. The switch arrangement shown in Figure 15 has two time-domain aggregation switching stages implemented by time-slot interchangers (TSIs). In the embodiment shown in Figure 15, appropriate buffering of virtual output queues (VOQs) is also required at the inputs (left-hand-side).

The input aggregations 16a, 16b are not shown explicitly in the (TST)S(S) switch arrangement 1 of Figure 15. However, logically, each input aggregation 16a comprises the time-domain aggregation switching stages 26, and 30 which are implemented by suitable buffering and timeslot interchangers (comprising optical and/or electronic technology) and a spatial aggregation switching stage 28 implemented using optical technology.

In Figure 15, the spatial switches forming the global spatial switching stage 18 are implemented as wavelength switches, to benefit from insertion of simple fibre delay lines where needed, to equalise delays between cells/packets that require switching by the time-domain switches forming the time-domain aggregation switching stage 34 logically adjacent to the output of the global spatial switching stage 18 and those that do not.

In some embodiments of the switch arrangement, there will be some time-domain switches 34' provided to mitigate the effects of blocking, however, these are not shown in the embodiment of the switch arrangement shown in Figure 15. Where such additional hardware is utilised, the time-domain switches will be connected to

spatial switches which do not require any means for delay compensation (which would be required in switches 18a, b shown in Figure 15).

Figure 16 shows an alternative electronic embodiment of a spatial switch which could, for example, be implemented in the aggregation spatial switching stages 28 and 36 of the switch arrangement shown in Figure 15. In Figure 16, however, the spatial switches use shows the same structure employing wavelength switches as space switches throughout, using tuneable lasers.

If outlet grouping is to be used, such that all time-slotted traffic segments transmitted to line from an output aggregation (i.e., from the spatial switching stage 36 in the embodiment of the invention shown in Figure 15) go out on the same link on the same fibre to the same next destination switch arrangement, then there is no need for the final spatial aggregation switching stage 36 to be implemented (i.e., there is not need to provide space switch or wavelength switches to implement switching stage 36). This means that all wavelengths can simply be multiplexed into a single output fibre. In effect this means that it does not matter which particular output within an output aggregation each time-slotted traffic segment departs from.

The overall switch then becomes a 4-stage (TST)S switch (or (TS)S(T)). Furthermore, wavelength routers (e.g. arrayed waveguide grating devices) can be used between the spatial switching stages 28, 18 in order to reduce the amount of fibre "wiring" between these stages, which can be particularly beneficial when integrated optical components become available. When outlet grouping is used, the 4-stage switch can be implemented as the simple structure shown in Figure 17.

In Figure 17, a four stage (TST)S switch arrangement is shown for use with outlet grouping and employing wavelength routers to reduce internal fibre "wiring".

APPLICATION TO LARGE OPTICAL CELL/PACKET NETWORKS

The seven-stage switch arrangement shown in Figure 4 has an architecture which is also appropriate for representing a switch arrangement comprising a plurality of interconnected individual, smaller time-slotted traffic segment switching networks (e.g. cell/packet switching networks).

In one embodiment of the invention, the switch arrangement comprises a plurality of optical cell/packet rings or optical cell/packet passive (and amplified) optical networks (PONs), which constitute some of the switching stages of a switch arrangement in the form of a switching network, which is arranged to interconnect the rings or PONs. In this embodiment of the invention therefore, the switch arrangement 1 can be considered equivalent to a distributed switching network consisting of the rings or PONs plus the remaining switching stages of the switch arrangement, the latter of which provide the interconnection between the rings or PONs and are contained within a central hub or switching node.

The central hub switching node also contains the upstream and downstream head-ends of the PONs or their equivalents in rings. In some embodiments each head-end or its equivalent constitutes only part of a space switch (implemented as a wavelength switch), which is distributed between the end terminals (PON) or end nodes (ring) of the PON or ring and the central hub switching node. The end terminals (PONs) or end nodes (rings) constitute the input and output ports of switch 1; input ports in the upstream direction and output ports in the downstream direction. In some embodiments they also constitute time-slot interchangers for intermediate 3rd and 5th stages of the 7-stage architecture, where appropriate.

If the time-domain, multi-slot constraint is not implemented in the channel assignment process, then it would be necessary to implement the channel assignment scheme for the switch arrangement using a seven stage (TST)S(TST) architecture to represent the actual ring interconnection switch arrangement. This requires four time-domain switching stages, e.g. four stages of time-slot interchange (TSI) unless time-slotted traffic segments from the same VOQ are not required to be contiguous (they will still be in the correct relative time-sequence) when received at the destination node or terminal, in which case the final time-domain switching stage 38 can be omitted. This leaves three TSI stages to be implemented.

In this embodiment, the spatial switching stage 36 in the output aggregation cannot be removed because outlet grouping of the type discussed hereinabove cannot apply to the receiving nodes or terminals of an output (or input) aggregation comprising a PON or ring. It therefore does matter from which output of an output aggregation each time-slotted traffic segment departs the switch arrangement (i.e. it does matter which node or terminal a time-slotted traffic segment arrives at in the PON or ring context).

Those skilled in the art will be aware that multiple PONs (or rings) can be interconnected in two fundamental ways; single pass (one hop) or multi-hopping. In a single pass, a central hub switch must provide most if not all of the switching functions between the PONs or rings, except the first time-domain aggregation switching stage 26.

By multi-hopping it is possible to implement most if not all of the time-domain aggregation switching stages in the end terminals (or nodes) (i.e., so that they can be represented by the input/output aggregations), leaving purely optical switching functions in the central hub switch (represented by the global spatial switching stage 18) . If all the time-domain aggregation switching stages are in the end terminals (nodes), then the 7 stage architecture (such as is shown in Figure 4 of the drawings) makes 3 hops necessary. In this embodiment, the terminals (or nodes) comprise tuneable wavelength lasers and/or tuneable receivers (for example, comprising tuneable filters or wavelength selectors). These components allow the individual PONs or rings to become part of the spatial aggregation switching stages 28, 36 in a distributed manner. The remaining parts of the outer spatial switching stages are located in the central hub switch.

The constrained channel assignment process implements a time-domain, multi-slot constraint on the assignment of time-slotted traffic segments. This can be used to reduce the number of time-domain aggregation switching stages to two resulting in a either a (TST)S(S) or a (TS)S(TS) switch arrangement being used to represent the actual switch arrangement structure.

Figure 18 indicates how the switch arrangement shown in Figure 15 can be modified to enable a single pass through a central hub switch for interconnecting multiple PONs, implementing the (TST)S(S) structure with distributed wavelength switching in the PONs. Because the intermediate time-slot interchange stage is located in the central hub switch, the single pass provides a single hop between end terminals or nodes.

In Figure 12, a switch arrangement is shown in which multiple WDM PONs are interconnected using just two hops through a central hub switch with only optical functionality which can be implemented by a (TST)S(S) switch arrangement in the channel assignment scheme according to the invention.

It is the multi-slotting which has here enabled the number of hops to be reduced from 3 hops to 2 hops as the multi-slotting constraint enables a time-domain aggregation switching stage (i.e., one of the time-slot interchange stages of the switch arrangement shown) to be eliminated. The hub design shown in Figure 12 is an embodiment of the invention suitable for interconnecting multiple WDM PONs in which all tuneable components are located in the hub. The tuneable components may be shared between multiple terminals, offering a potentially lower-cost solution.

Alternatively, instead of using a set of tuneable lasers or tuneable wavelength converters at the output of the hub switch to provide distributed wavelength switching into the downstream PONs, a set of space switches (represented by spatial switching stage 36t) is shown associated with each downstream PON, together with a set of fixed wavelength converters. In another alternative of the invention, the space switches could be replaced by receivers and fixed-wavelength lasers.

A wavelength router is used at the input side of the hub, in order to reduce the amount of fibre "wiring". In embodiments of the invention in which additional switching stages are inserted to mitigate or obviate the effects of multi-slot blocking, fibre delay lines (not shown) can be easily inserted between the router and the global spatial switching stage 18 to equalise any delay between time-slotted traffic segments which pass through the time-domain switching stage 34 and time-slotted traffic segments which do not.

In such embodiments of the invention, the small number of time-domain switches (e.g. TSIs) (also not shown) are implemented at the outputs of some of the spatial switches 18a...d of the global spatial switching stage 18 of the switch arrangement. The spatial switches forming the spatial switching stage do not need to include fibre delay lines. Alternatively, fibre delay lines and the time-domain aggregation switching stage 34 can be implemented in the hub. This simply makes use of the terminals' buffers to delay the time-slotted traffic segments that do not require a time-domain aggregation switching stage 34 adjacent to the logical outputs of the global switching stage 18 relative to the time-slotted traffic segments that do. Distributed wavelength switching is not implemented within the PONs.

In the embodiment shown in Figure 19, in the first hop, the fibre delay lines (if used) and any time-domain switches 34a...p (if used) are switched out (e.g. by 2x2

switches). The first set of space switches 124 shown in Figure 19 connect all wavelength channels from each upstream PON (i.e., from each input aggregation) straight through, so that they all return to their own downstream PONs (i.e. the same input aggregation). In this hop, the second set of space switches 124 shown in Figure 19 form spatial aggregation switching stage 28 in the 7 stage logical representation of the switch arrangement shown in Figure 4. The switches 126 thus provide the first spatial switching operation between wavelength channels within the same PON (input aggregation). The PON terminals buffer and time-slot interchange the cells/packets (represented by time-domain switching stage 30 in Figure 4), ready for the second hop.

In the second hop, the fibre delay lines and any time-domain switches (e.g. TSIs) representing a time-domain aggregation switching stage 34' are switched in (if used).

The first set of spatial switches 124 performs the role of the global spatial switching stage 18 in the seven stage logical representation of the switch. The second set of switches 126 shown in Figure 19 now performs the role of the spatial aggregation switching stage 36 (i.e. the final stage of space switches (the 6th stage) in the 7 stage switch architecture represented by Figure 4). The fibre "wiring" between the two sets of space switches can be reduced by inserting wavelength converters at the outputs of the first set of spaces switches, which allows a second wavelength router to be used to interconnect the two sets. The second set of space switches must switch at the time-slot rate, but the first set of switches can switch at the frame rate (except for those few switches connected to the few additional time-domain switches 30' (e.g. time-slot interchangers, if these are used) providing extra hardware). This embodiment makes use of extra time slots to support multi-slotting. Since there are no more time-domain switches in the time-domain aggregation switching stage 34 than the number of terminals, then in the time-domain aggregation switching stage 34 (i.e., the 3rd stage of the switch arrangement) some of the PON terminals are used only to provide time-slot interchangers for traffic segments that will pass through time-domain switches in the time-domain aggregation switching stage 34 (i.e.., the 5th stage time-slot interchangers). However, the majority of the PON terminals are used only to provide time-domain switches for traffic segments that will not pass through time-domain switches in the time-domain aggregation switching stage 34. This means that not all time slots available to PON terminals in each frame can be used to carry traffic segments (here typically, cells), providing a type of line-card "speed-up"

where time slots can remain empty. Those skilled in the art will appreciate that the particular time slots remaining empty can vary from frame to frame.

Those skilled in the art will appreciate that any modifications to the functionality or features described herein which are apparent or obvious to those skilled in the art are intended to be encompassed by the scope of the claims. Those skilled in the art will appreciate that the above representation of the invention is limited to specific embodiments and that the scope of the invention should not be limited to the specific embodiments described.

The text of the abstract repeated below is hereby incorporated into the description:

A constrained channel assignment process for a multi-stage switch arrangement for switching traffic segmented into a plurality of time-slots, the multi-stage switch arrangement comprising a plurality of input aggregations, each input aggregation comprising a separate subset of the plurality of inputs of the switch arrangement, said subset of inputs being logically associated with at least one aggregation switching stage of the switch arrangement arranged to operate only on traffic derived from said subset of inputs, a plurality of output aggregations, each output aggregation comprising a separate sub-set of the outputs of the switch arrangement; and a global spatial switching stage arranged to receive time-slots from any of the plurality of input aggregations of the switch arrangement and arranged to provide time-slots to any of the plurality of output aggregations, wherein the channel assignment process comprises the step of:

identifying traffic segments which can be switched together as a logical switchable entity across the global switching stage of the switch arrangement; and

imposing a constraint on the manner in which each logical switchable entity comprising a plurality of timeslots carrying identified traffic segments are assigned from switches in an aggregation switching stage in each input aggregation logically adjacent to a global spatial switching stage to the outputs of the switch arrangement.

CLAIMS

1. A channel assignment process for a multi-stage switch arrangement, the channel assignment process comprising the steps of:

identifying traffic segments arriving at the inputs of said switch arrangement which can form a logical switchable entity to which one or more channels can be assigned across a spatial switching stage of the switch arrangement,

determining a plurality of time-slots capable of carrying said traffic segments across at least one switching stage of said multi-stage switch arrangement; and

constraining how each logical switchable entity is assigned to said plurality of timeslots for carrying said identified traffic segments from one or more switches in a switching stage logically adjacent to the input of the spatial switching stage to switches in a switching stage logically adjacent to the output of the spatial switching stage,

wherein the number of computational steps required to implement channel assignment process for the switch arrangement is reduced by said constraining step resulting in at least part of at least one of the switching stages of the multi-stage switch arrangement being logically rendered redundant.

2. A channel assignment process as claimed in claim 1, wherein the number of timeslots in a frame of the channel assignment process is equal to the maximum number of time-slots which is required by the logical switchable entity.

3. A channel assignment process as claimed in claim 2, wherein each switch in a switching stage logically adjacent to the input of spatial switching stage is constrained to be connected by said spatial switching stage to a switch in an switching stage logically adjacent to the output of the spatial switching stage, such that one of said switching stages logically adjacent to the spatial switching stage is redundant being connected back to back with said other switching stage logically adjacent to said spatial switching stage.

4. A channel assignment process as claimed in claim 1, wherein the number of timeslots in a frame of the channel assignment process is less than the maximum number of time-slots required by a logical switchable entity.

5. A channel assignment process as claimed in claim 4, wherein a plurality of switches in each switching stage logically adjacent to the input of the spatial

switching stage are constrained collectively to switch each logical switchable entity to the same number of switches in a switching stage logically adjacent to the output of the spatial switching stage.

6. A channel assignment process as claimed in any previous claim, wherein each switching stage logically adjacent to the input of said spatial switching stage comprises a plurality of aggregation switching stages operating in parallel on the inputs to the switch arrangement, wherein each of said parallel aggregation switching stages operates on a unique subset of the inputs of the switch arrangement.

7. A channel assignment process as claimed in any previous claim, wherein each switching stage logically adjacent to the output of said spatial switching stage comprises a plurality of aggregation switching stages operating in parallel to provide output from said switch arrangement, wherein each of said parallel aggregation switching stages provides output to a unique subset of the outputs of the switch arrangement.

8. A channel assignment process as claimed in any previous claim, wherein at least one of the switching stages comprising a plurality of parallel aggregation switching stages is rendered redundant by constraining how said logical switchable entity is assigned timeslots.

9. A channel assignment process as claimed in any previous claim, wherein the computational complexity of implementing the channel assignment process is further reduced by:

decomposing the switch arrangement into a plurality of time-space-time aggregation switching stages, each aggregation switching stage being associated with a unique sub-set of inputs or outputs of the switch arrangement, and

constraining the channel assignment process such that at least one of said time-space-time switching stages is redundant.

10. A process as claimed in claim 9, wherein said time-space-time aggregation switching stages are hierarchical.

11. A process as claimed in clam 9 or 10, wherein said channel assignment for a switch arrangement is further performed on a logical element comprising said plurality of time-space-time aggregation switching stages.

12. A process as claimed in any previous claim, wherein said switching arrangement comprises a hierarchy of switching stages and said channel assignment process is performed within one or more levels of said hierarchy of switching stages.

13. A constrained channel assignment process for a multi-stage switch arrangement for switching traffic segmented into a plurality of time-slots, the multi-stage switch arrangement comprising:

a plurality of input aggregations, each input aggregation comprising a unique subset of the plurality of inputs of the switch arrangement, said subset of inputs being logically associated with at least one aggregation switching stage of the switch arrangement arranged to operate only on traffic derived from said subset of inputs;

a plurality of output aggregations, each output aggregation comprising a unique sub-set of the outputs of the switch arrangement; and

a global spatial switching stage arranged to receive time-slots from any of the plurality of input aggregations of the switch arrangement and arranged to provide time-slots to any of the plurality of output aggregations, wherein the channel assignment process comprises the steps of:

identifying traffic segments which can be switched together as a logical switchable entity across the global switching stage of the switch arrangement; and

imposing a constraint on the manner in which each logical switchable entity comprising a plurality of timeslots carrying identified traffic segments is assigned from one or more switches in an aggregation switching stage in each input aggregation logically adjacent to a global spatial switching stage to the outputs of the switch arrangement.

14. A channel assignment process as claimed in claim 13, in which the time-slots are constrained so that each switchable entity is matched from one or more available time-domain aggregation switches in a time-domain switching stage which is logically adjacent to the input side of the global spatial switching stage to one or more available time-domain aggregation switches in a time-domain switching stage which is logically adjacent to the output side of the global spatial switching stage.

15. A channel assignment process as claimed in claim 13, in which the number M of time-slots which are identified as containing time-slotted traffic segments which can be switched together as a single logical switching entity is larger than the number F of time-slots forming each frame of traffic switched by each time-domain switch in

an input aggregation.

16. A channel assignment process as claimed in claim 13, in which the number M of time-slots which are identified as containing time-slotted traffic segments which can be switched together as a single logical switching entity is less than the number F of time-slots forming each frame of traffic switched by each time-domain switch in an input aggregation.

17. A channel assignment process as claimed in claim 13, in which the number M of time-slots which are identified as containing time-slotted traffic segments which can be switched together as a single logical switching entity is equal to the number F of time-slots forming each frame of traffic switched by each time-domain switch in an input aggregation.

18. A channel assignment process as claimed in claim 17, in which in each frame, each time-domain aggregation switch of the time-domain aggregation switching stage in an input aggregation is constrained so that at all times the timeslots which it transmits are sent to the same switch by the same spatial switch in the global spatial switching stage.

19. A channel assignment process as claimed in any one of claims 13 to 18 in which pairs of switches in each of time-domain aggregation stages logically adjacent to the inputs and outputs of the global spatial switching stage are connected back to back to each other.

20. A channel assignment process as claimed in any one of preceding claims 13 to 19, in which the global spatial switching stage is implemented by a plurality of spatial switches and each of said plurality of spatial switches switches at the frame rate.

21. A channel assignment process as claimed in claim 17, in which two or more switches in aggregation switching stage of an input aggregation switch each switchable entity to the same number of switches in an output aggregation switching stage.

22. A channel assignment process as claimed in any one of preceding claims 13 to 21, in which each switchable entity is switched from a single input aggregation to a

single output aggregation.

23. A channel assignment process as claimed in any one of preceding claims 13 to 22, in which each switchable entity is switched from a single input aggregation to more than one output aggregation element.

24. A channel assignment process as claimed in claim 16, in which each switch in an aggregation switching stage of an input aggregation is capable of switching more than one switchable entity to one or more switches not comprising part of the same output aggregation.

25. A channel assignment process as claimed in any one of preceding claims 13 to 24, in which within each input aggregation of said switch arrangement, the arrangement of aggregation switching stages supports the implementation of the channel assignment process according to any preceding claim for the assignment of channels within the input aggregation.

26. A channel assignment process as claimed in any one of preceding claims 13 to 25, in which the process is performed recursively for in increasingly smaller number M of time-slots which are identified as containing time-slotted traffic segments which can be switched together as a single logical switching entity.

27. A multi-stage switch arrangement comprising means to implement a channel assignment process according to any one of the previous claims.

28. A multi-stage switch arrangement as claimed in claim 27, wherein said arrangement comprises a plurality of switching apparatus platforms interconnected by a communications network.

29. A method of generating control information for a switch arrangement using a representation of a multi-stage switch arrangement for performing a channel assignment process, the method comprising the steps of:

representing the switch arrangement as a plurality of switching stages which operate in series on traffic received by the switch arrangement, wherein said plurality of switching stages comprise a global spatial switching stage in series with a remaining plurality of switching stages, each remaining switching stage comprising a plurality of parallel aggregation switching stages, each aggregation switching stage

being arranged to receive input from and/or provide output to a subset of the inputs and outputs of the switch arrangement respectively,

assigning time-slots to traffic segments for routing across said global switching stage in said representation in such a manner that at least one of said plurality of aggregation switching stages of the switch arrangement is rendered redundant in said representation; and

generating control information comprising said time-slot assignments and communicating said control information to said switch arrangement.

30. A multi-stage switch arrangement comprising:

a plurality of input aggregations, each input aggregation comprising a unique subset of the plurality of inputs of the switch arrangement, said subset of inputs being logically associated with at least one aggregation switching stage of the switch arrangement arranged to operate only on traffic derived from said subset of inputs;

a plurality of output aggregations, each output aggregation comprising a unique sub-set of the outputs of the switch arrangement;

a global spatial switching stage arranged to receive time-slots from any of the plurality of input aggregations of the switch arrangement and arranged to provide time-slots to any of the plurality of output aggregations; and

a control interface whereby the switch arrangement receives scheduling information from a switch control means, the switch control means generating said scheduling information by performing a channel assignment process according to any one of claims 1 to 26.

31. Control apparatus for scheduling traffic through a multi-stage switch arrangement as claimed in claim 30, the control apparatus comprising:

means to identify traffic segments arriving at the inputs of said switch arrangement which can form a logical entity to which a channel can be assigned across at least one spatial switching stage of the switch arrangement,

means to determine a plurality of time-slots capable of carrying said traffic segments across at least one switching stage of said multi-stage switch arrangement; and

processing means to constrain how each logical switchable entity is assigned to said plurality of timeslots for carrying said identified traffic segments from one or more switches in a switching stage logically adjacent to the input of the spatial switching stage to switches in a switching stage logically adjacent to the output of the spatial switching stage;

means to convey control information to the switch arrangement to implement channel assignment for said traffic segments through the switch arrangement.

32. A suite of one or more computer programs arranged to implement a channel assignment process as claimed in any one of claims 1 to 26.

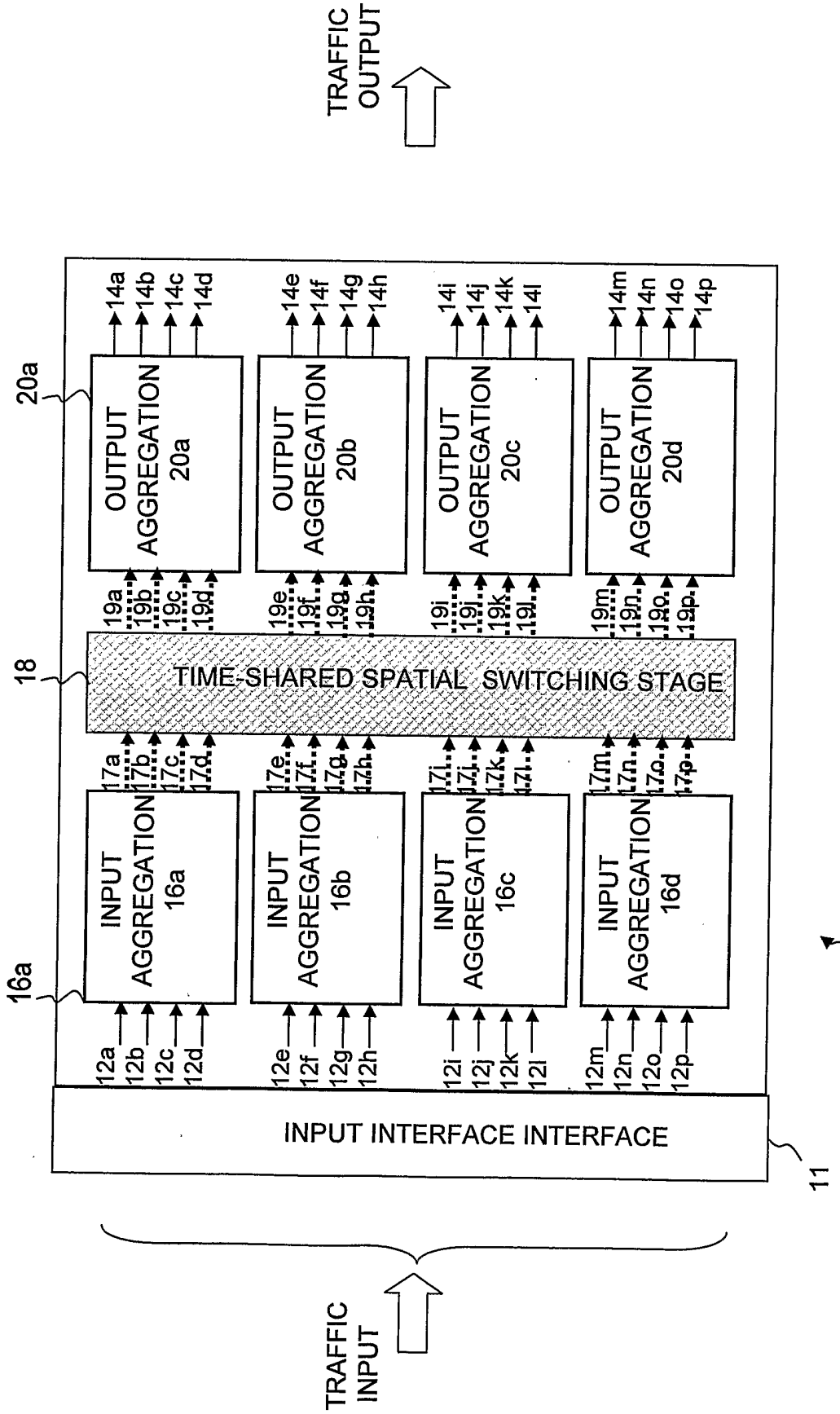

TRAFFIC OUTPUT
20a
OUTPUT AGGREGATION 20a
OUTPUT AGGREGATION 20b
OUTPUT AGGREGATION 20c
OUTPUT AGGREGATION 20d
14a 14b 14c 14d
14e 14f 14g 14h
14i 14j 14k 14l
14m 14n 14o 14p
19a 19b 19c 19d
19e 19f 19g 19h
19i 19j 19k 19l
19m 19n 19o 19p
18
TIME-SHARED SPATIAL SWITCHING STAGE
17a 17b 17c 17d
17e 17f 17g 17h
17i 17j 17k 17l
17m 17n 17o 17p
16a
INPUT AGGREGATION 16a
INPUT AGGREGATION 16b
INPUT AGGREGATION 16c
INPUT AGGREGATION 16d
12a 12b 12c 12d
12e 12f 12g 12h
12i 12j 12k 12l
12m 12n 12o 12p
INPUT INTERFACE INTERFACE
11
10
TRAFFIC INPUT

FIGURE 1

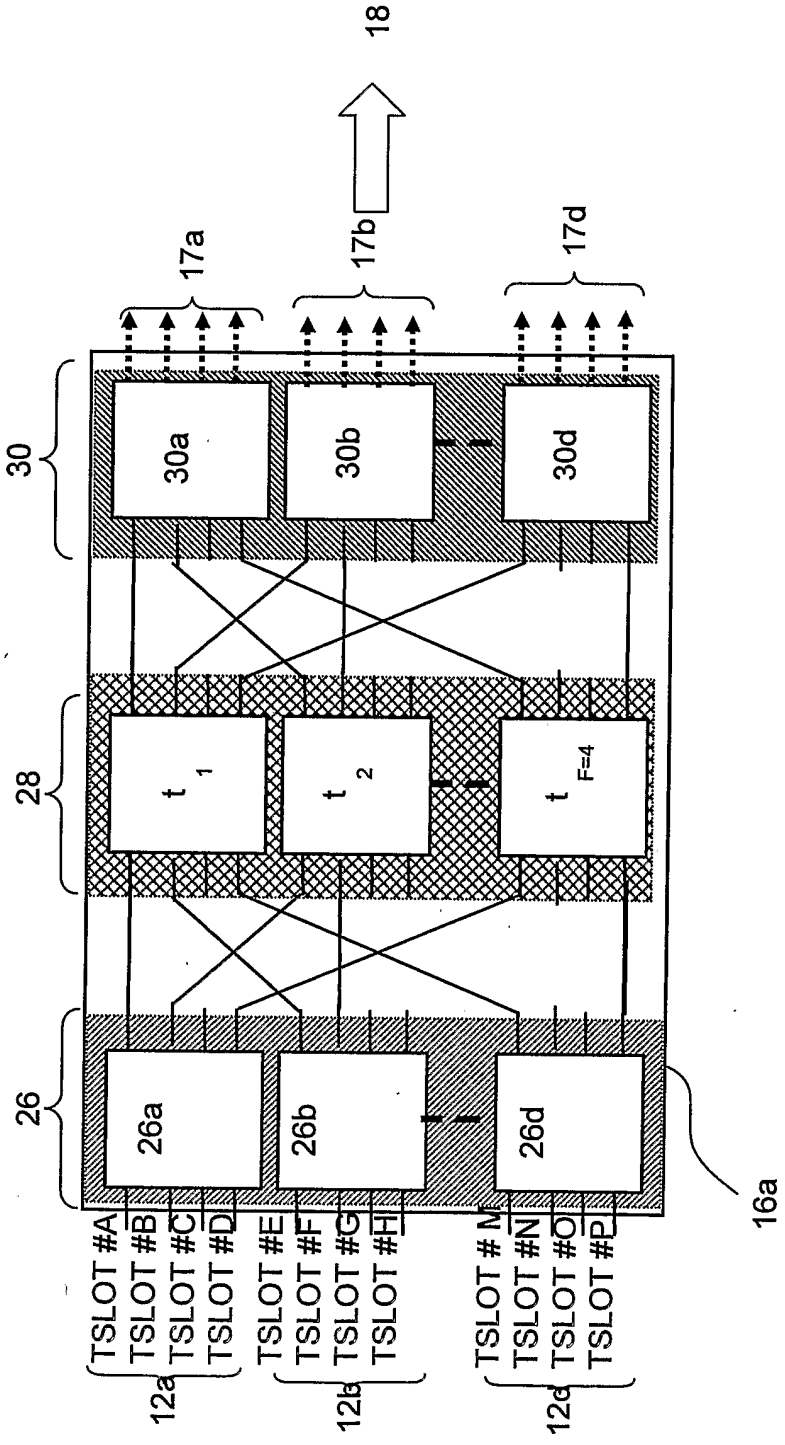

18
17a
17b
17d
30
30a
30b
30d
28
t 1
t 2
t F=4
26
26a
26b
26d
16a
TSLOT #A
TSLOT #B
TSLOT #C
TSLOT #D
TSLOT #E
TSLOT #F
TSLOT #G
TSLOT #H
TSLOT # M
TSLOT #N
TSLOT #O
TSLOT #P
12a
12b
12d

FIGURE 2

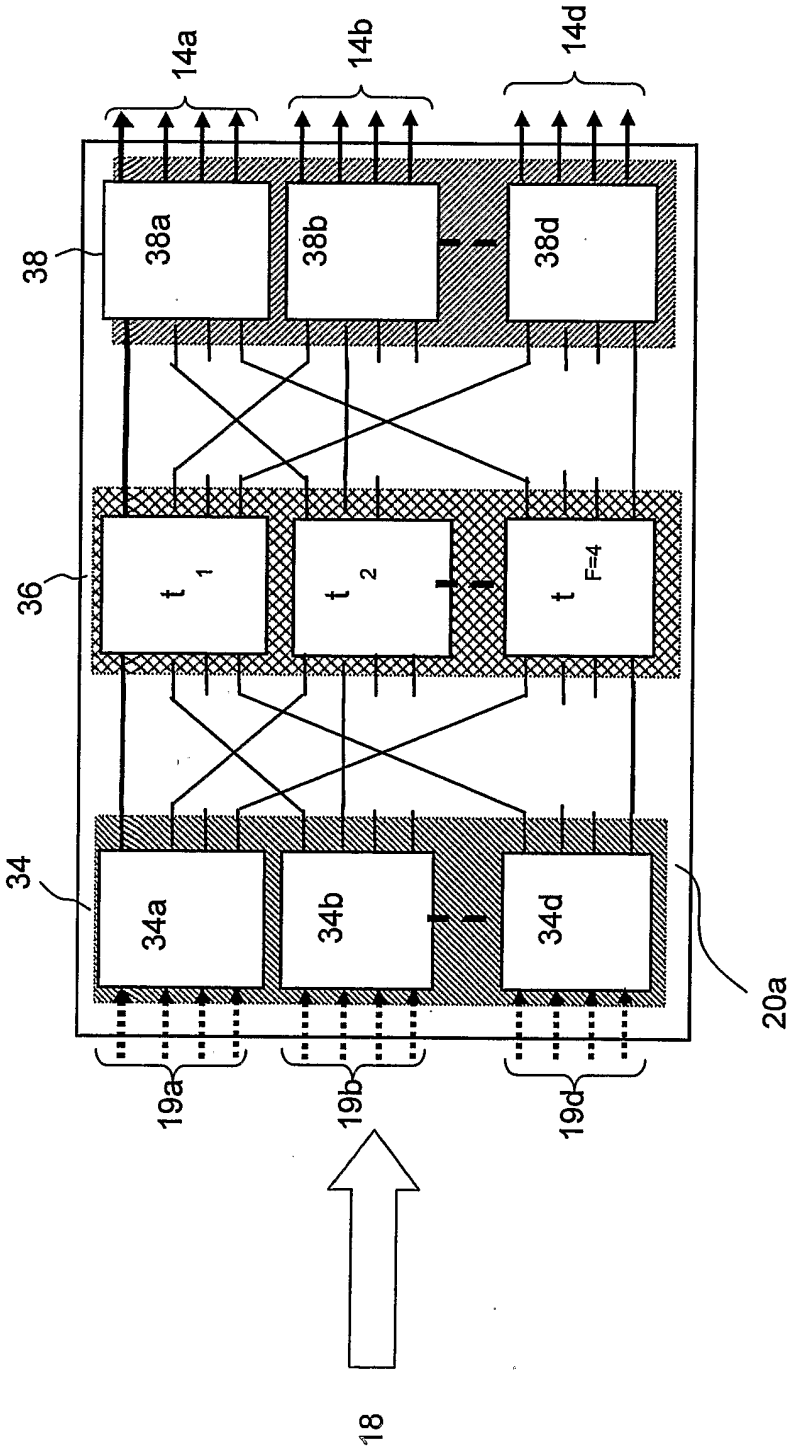

14a
14b
14d
38
38a
38b
38d
36
t 1
t 2
t F=4
34
34a
34b
34d
20a
19a
19b
19d
18

FIGURE 3

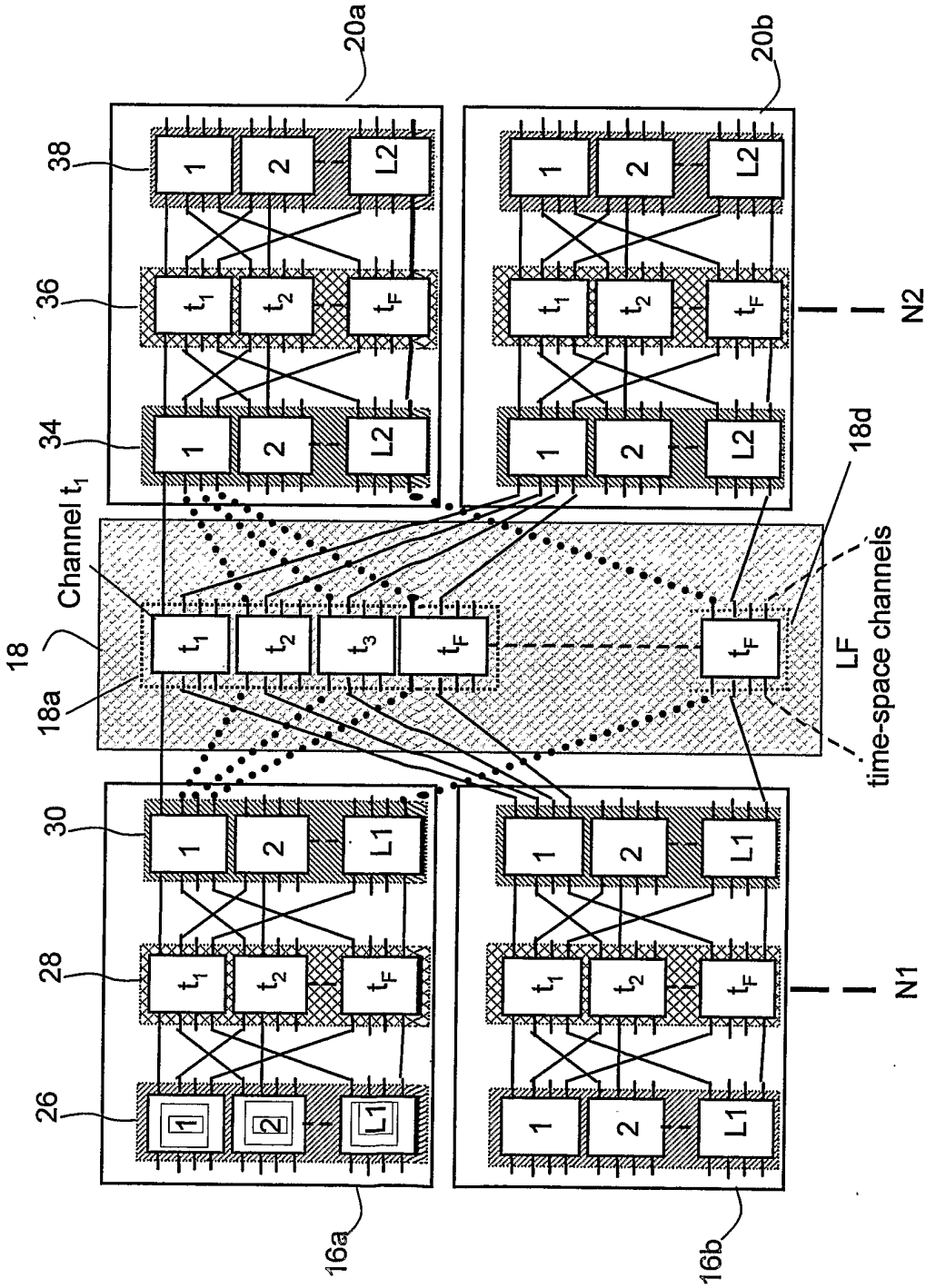

20a
20b
38
36
34
30
28
26
18
18a
18d
16a
16b
N2
N1
LF
time-space channels
Channel t_1
1
2
L2
L1
t_1
t_2
t_3
t_F

FIGURE 4

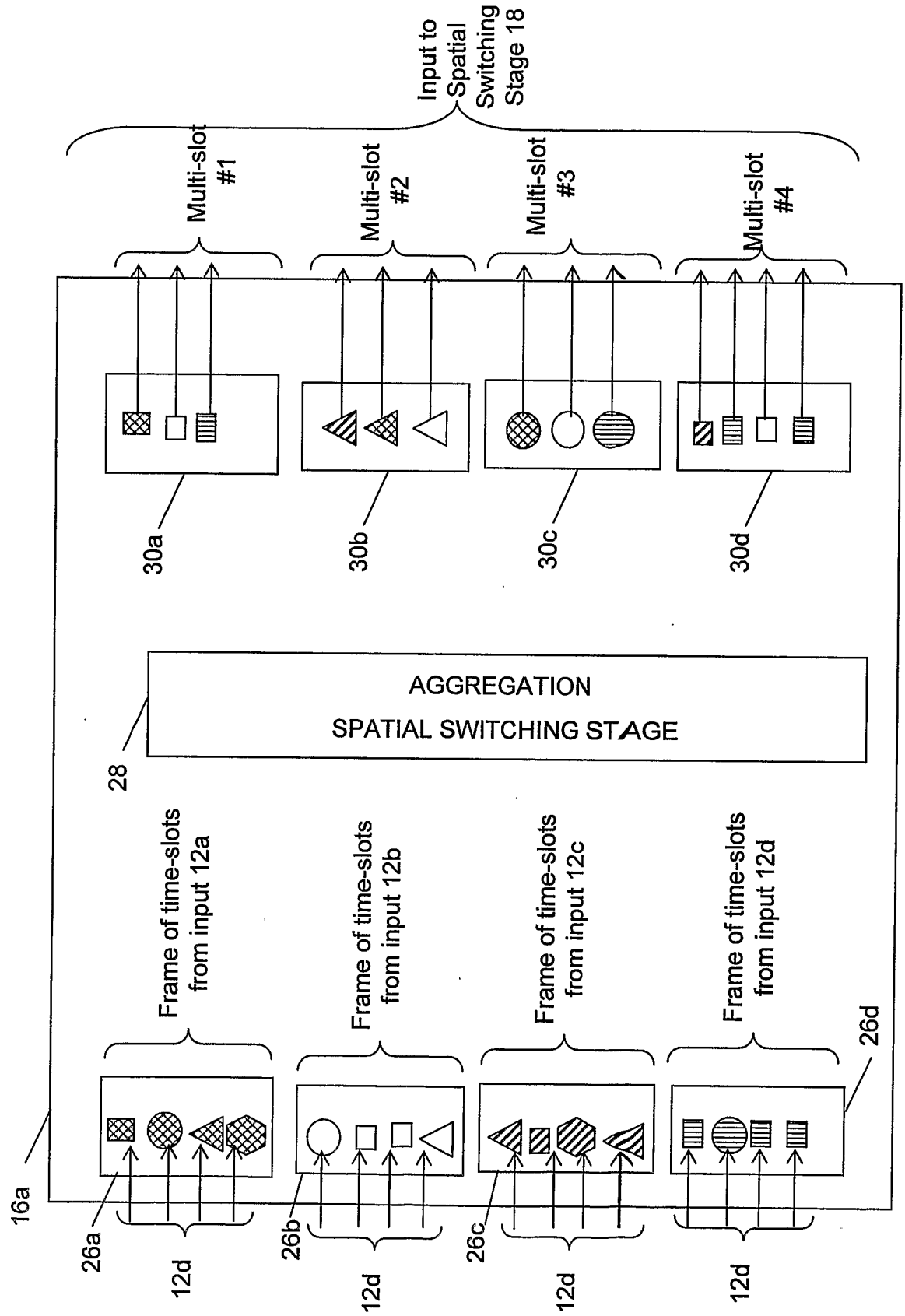

Input to Spatial Switching Stage 18
Multi-slot #1
Multi-slot #2
Multi-slot #3
Multi-slot #4
30a
30b
30c
30d
AGGREGATION
SPATIAL SWITCHING STAGE
28
Frame of time-slots from input 12a
Frame of time-slots from input 12b
Frame of time-slots from input 12c
Frame of time-slots from input 12d
16a
26a
26b
26c
26d
12d
12d
12d
12d

FIGURE 5

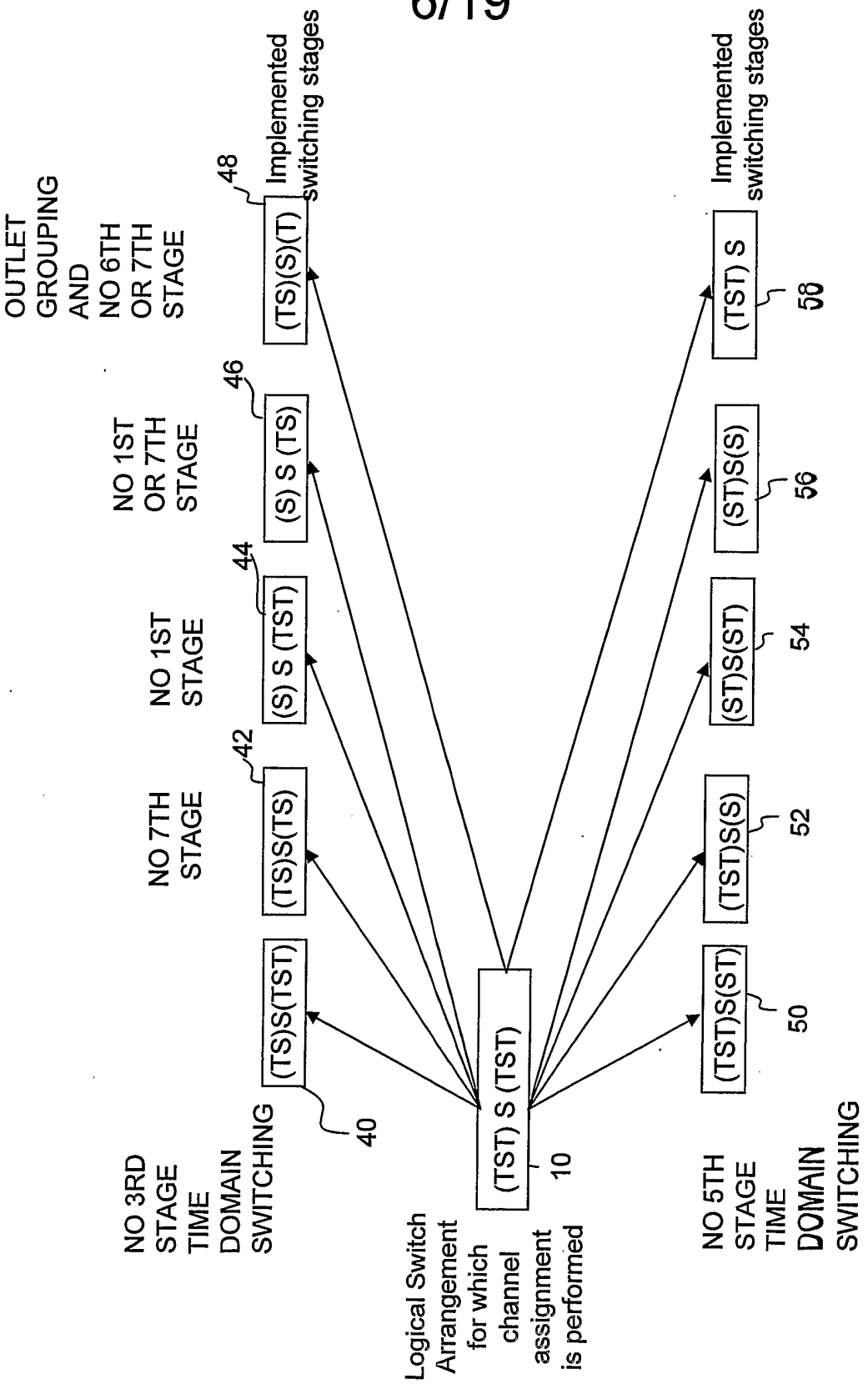

NO 3RD STAGE TIME DOMAIN SWITCHING
NO 7TH STAGE
NO 1ST STAGE
NO 1ST OR 7TH STAGE
OUTLET GROUPING AND NO 6TH OR 7TH STAGE
NO 5TH STAGE TIME DOMAIN SWITCHING
(TS)S(TST)
(TS)S(TS)
(S) S (TST)
(S) S (TS)
(TS)(S)(T)
(TST) S (TST)
(TST)S(ST)
(TST)S(S)
(ST)S(ST)
(ST)S(S)
(TST) S
40
42
44
46
48
50
52
54
56
58
10
Implemented switching stages
Implemented switching stages
Logical Switch Arrangement for which channel assignment is performed

FIGURE 6

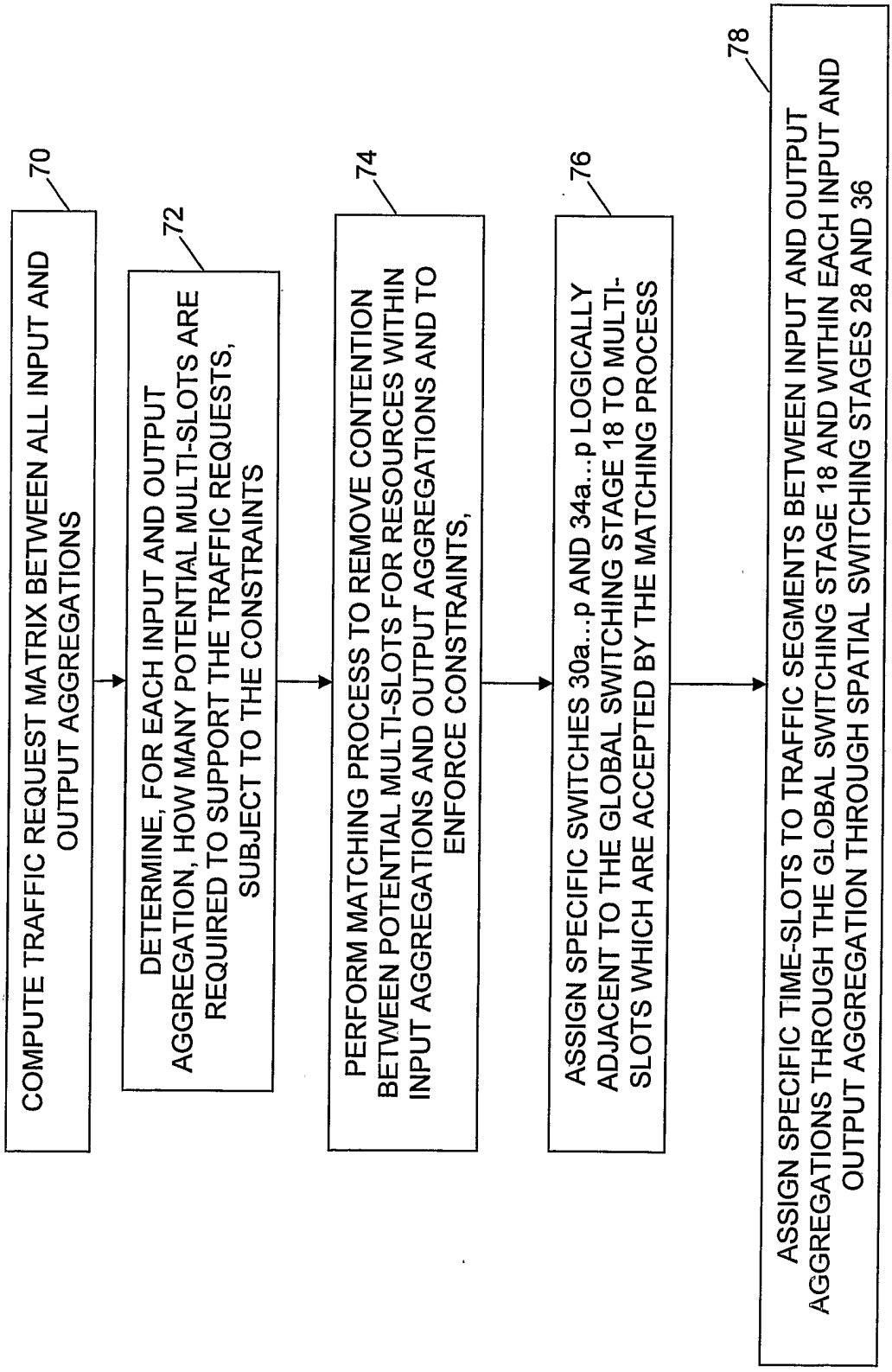

70
COMPUTE TRAFFIC REQUEST MATRIX BETWEEN ALL INPUT AND OUTPUT AGGREGATIONS
72
DETERMINE, FOR EACH INPUT AND OUTPUT AGGREGATION, HOW MANY POTENTIAL MULTI-SLOTS ARE REQUIRED TO SUPPORT THE TRAFFIC REQUESTS, SUBJECT TO THE CONSTRAINTS
74
PERFORM MATCHING PROCESS TO REMOVE CONTENTION BETWEEN POTENTIAL MULTI-SLOTS FOR RESOURCES WITHIN INPUT AGGREGATIONS AND OUTPUT AGGREGATIONS AND TO ENFORCE CONSTRAINTS,
76
ASSIGN SPECIFIC SWITCHES 30a...p AND 34a...p LOGICALLY ADJACENT TO THE GLOBAL SWITCHING STAGE 18 TO MULTI-SLOTS WHICH ARE ACCEPTED BY THE MATCHING PROCESS
78
ASSIGN SPECIFIC TIME-SLOTS TO TRAFFIC SEGMENTS BETWEEN INPUT AND OUTPUT AGGREGATIONS THROUGH THE GLOBAL SWITCHING STAGE 18 AND WITHIN EACH INPUT AND OUTPUT AGGREGATION THROUGH SPATIAL SWITCHING STAGES 28 AND 36

FIGURE 7

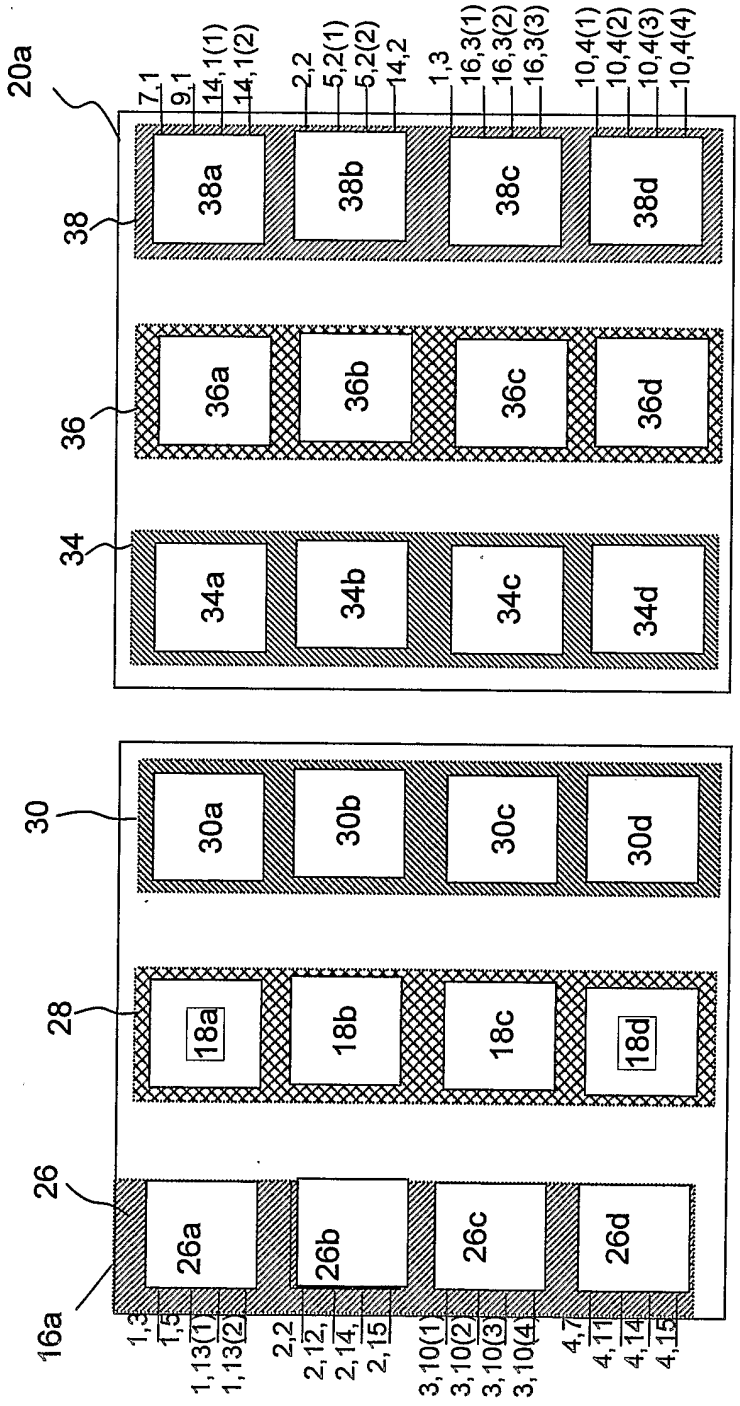

20a
38
36
34
30
28
26
16a
38a
38b
38c
38d
36a
36b
36c
36d
34a
34b
34c
34d
30a
30b
30c
30d
18a
18b
18c
18d
26a
26b
26c
26d
7,1
9,1
14,1(1)
14,1(2)
2,2
5,2(1)
5,2(2)
14,2
1,3
16,3(1)
16,3(2)
16,3(3)
10,4(1)
10,4(2)
10,4(3)
10,4(4)
1,3
1,5
1,13(1)
1,13(2)
2,2
2,12
2,14
2,15
3,10(1)
3,10(2)
3,10(3)
3,10(4)
4,7
4,11
4,14
4,15

FIGURE 8

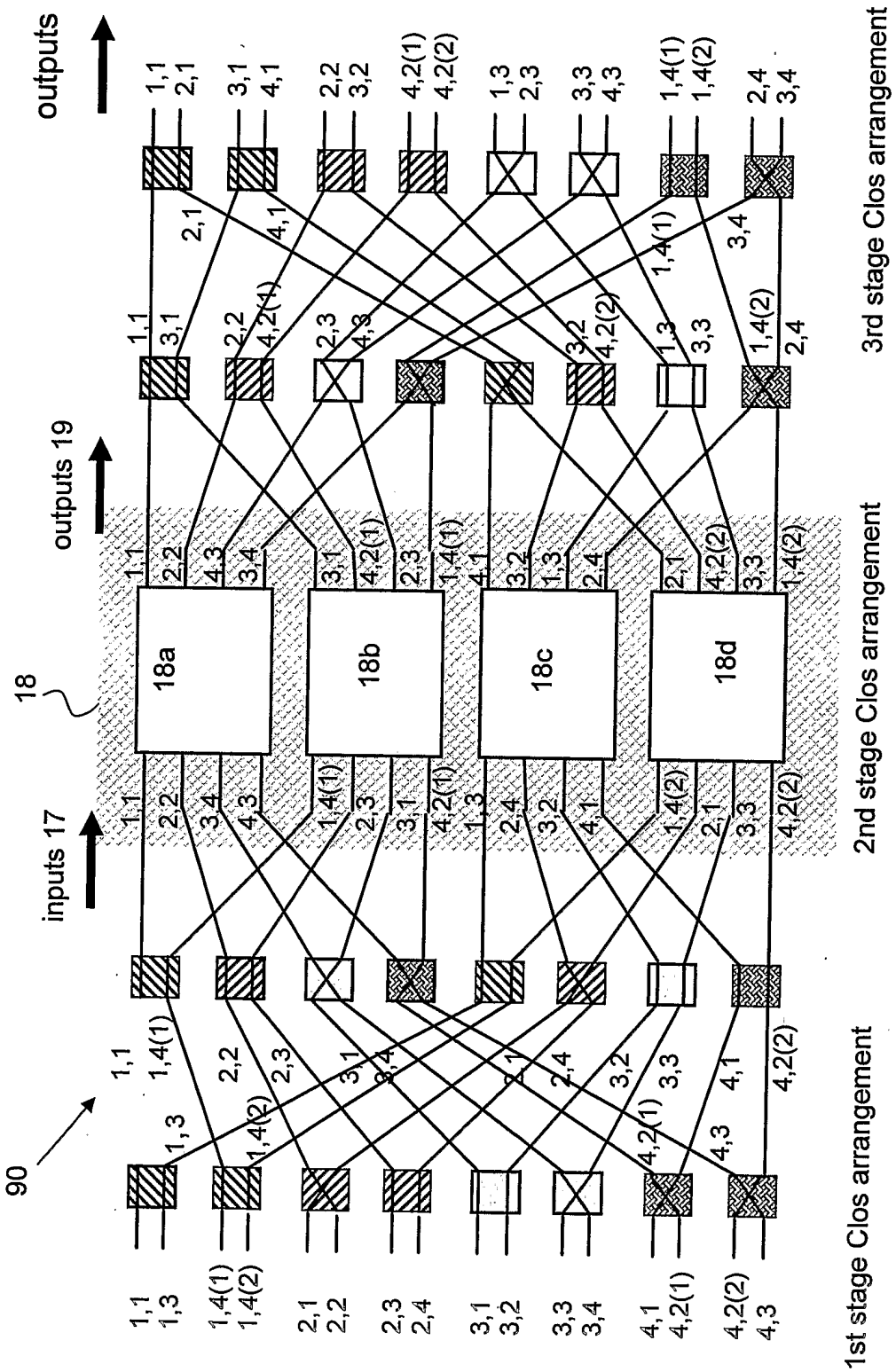

outputs
outputs 19
inputs 17
18
90
18a
18b
18c
18d
1st stage Clos arrangement
2nd stage Clos arrangement
3rd stage Clos arrangement

FIGURE 9

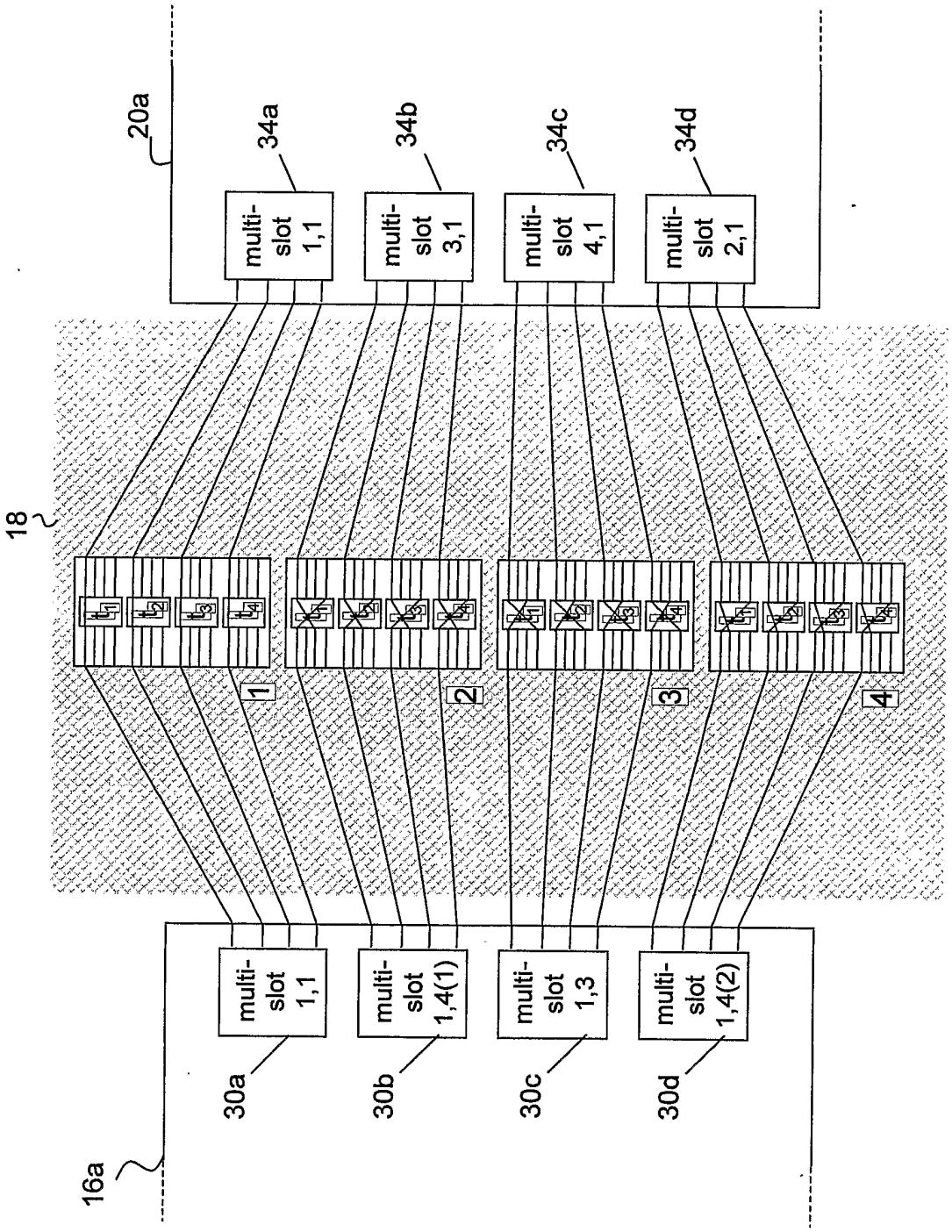

20a
34a
34b
34c
34d
multi-slot 1,1
multi-slot 3,1
multi-slot 4,1
multi-slot 2,1
18
1
2
3
4
multi-slot 1,1
multi-slot 1,4(1)
multi-slot 1,3
multi-slot 1,4(2)
30a
30b
30c
30d
16a

FIGURE 10

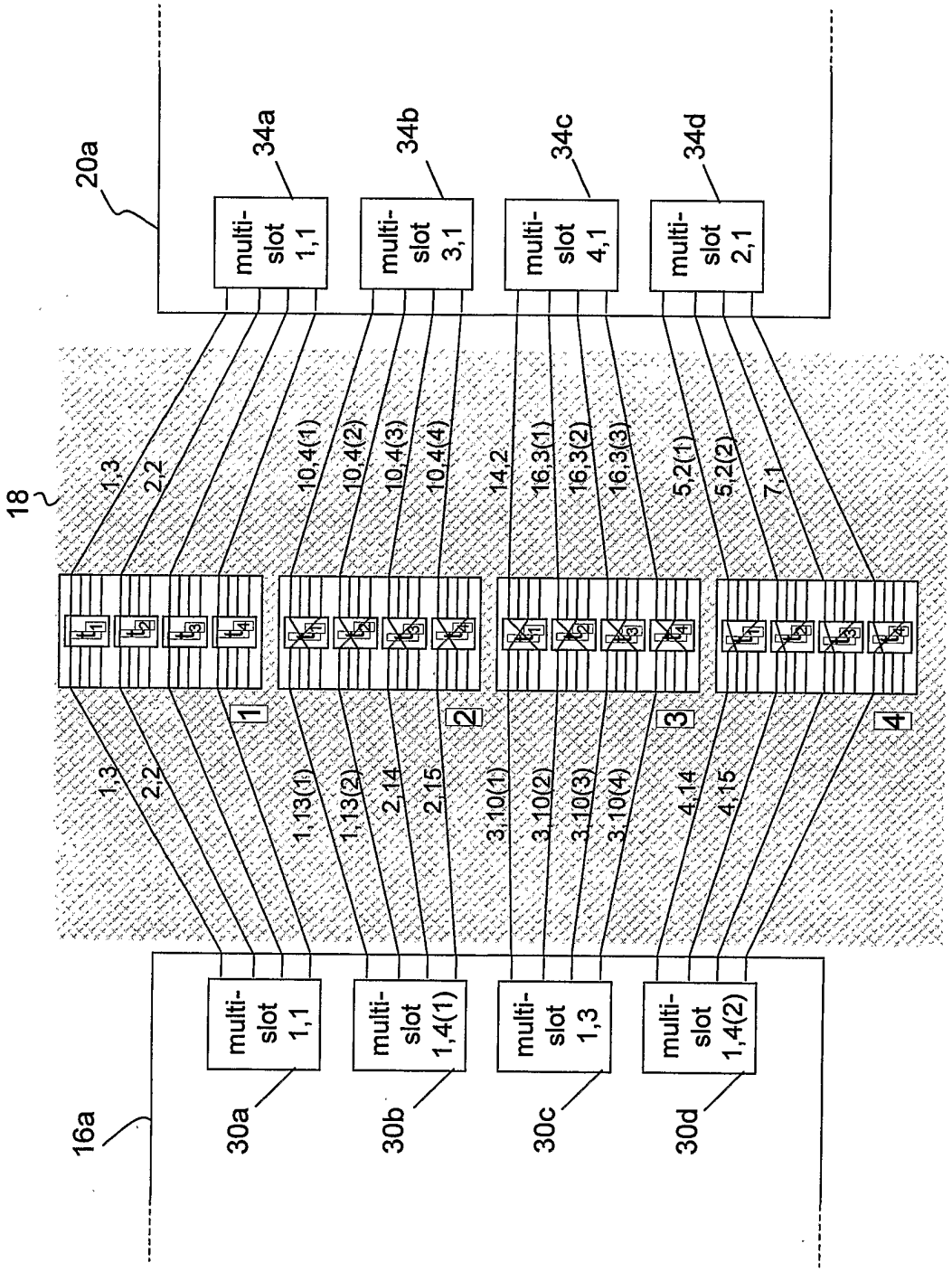

20a
34a
34b
34c
34d
multi-slot 1,1
multi-slot 3,1
multi-slot 4,1
multi-slot 2,1
18
1,3
2,2
10,4(1)
10,4(2)
10,4(3)
10,4(4)
14,2
16,3(1)
16,3(2)
16,3(3)
5,2(1)
5,2(2)
7,1
1
2
3
4
1,3
2,2
1,13(1)
1,13(2)
2,14
2,15
3,10(1)
3,10(2)
3,10(3)
3,10(4)
4,14
4,15
16a
30a
30b
30c
30d
multi-slot 1,1
multi-slot 1,4(1)
multi-slot 1,3
multi-slot 1,4(2)

FIGURE 11

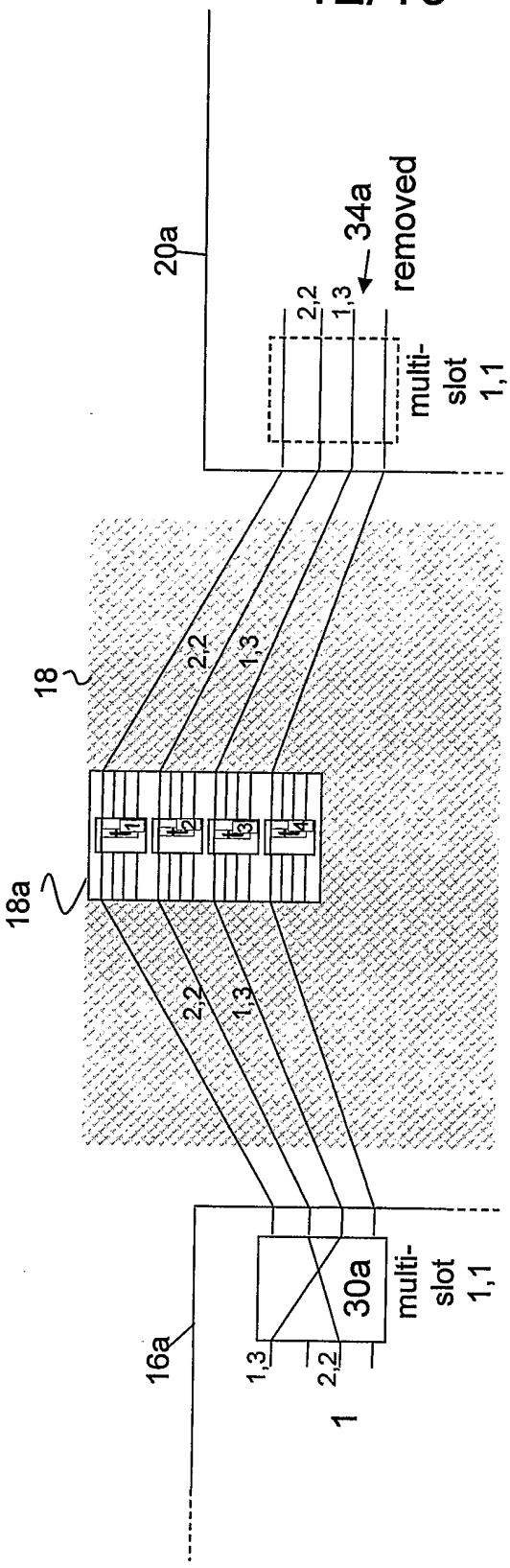

20a
34a
removed
2,2
1,3
multi-
slot
1,1
18
2,2
1,3
18a
2,2
1,3
16a
30a
multi-
slot
1,1
1,3
2,2
1

FIGURE 12

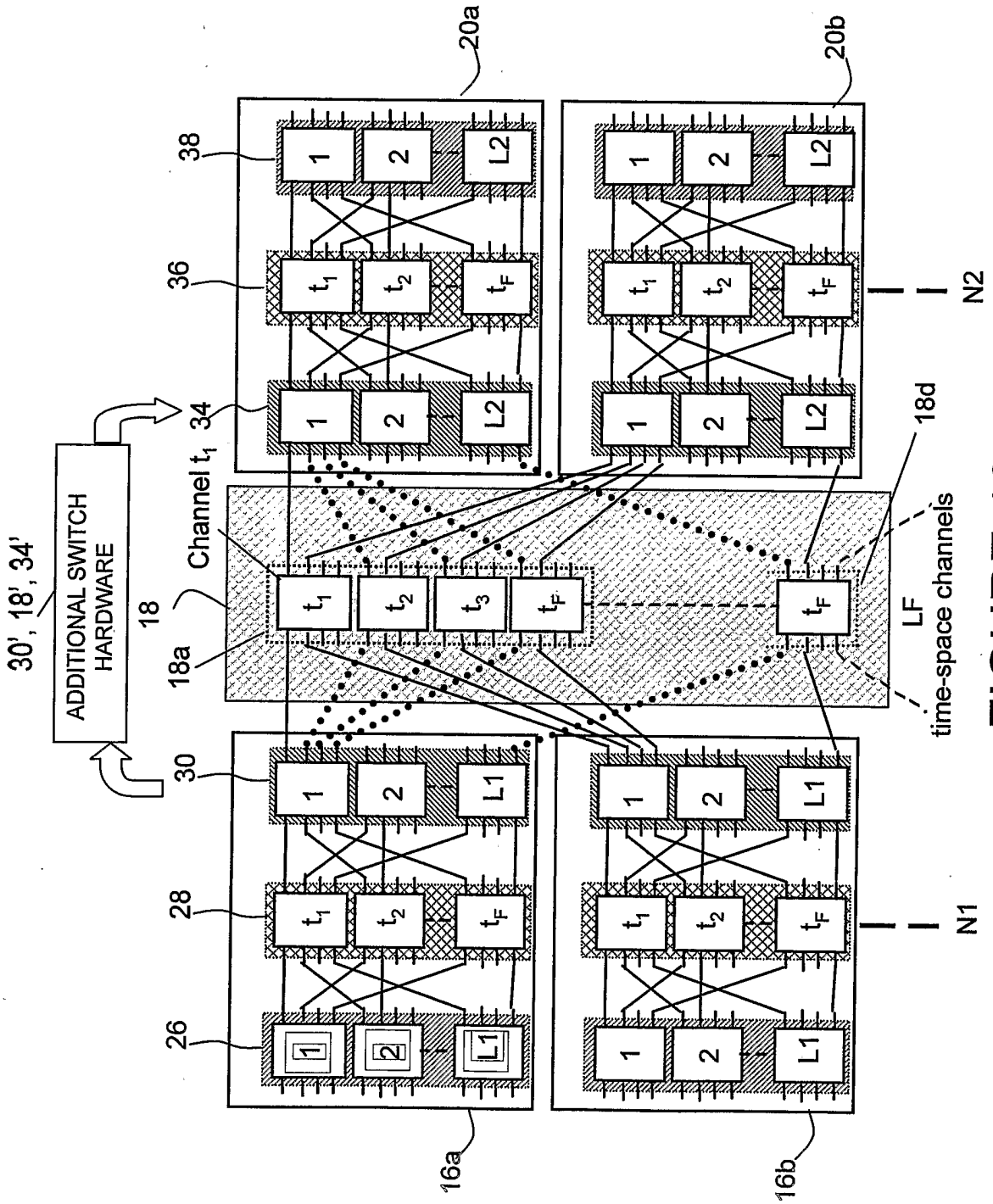

20a
20b
38
36
34
30
28
26
16a
16b
18
18a
18d
Channel t_1
ADDITIONAL SWITCH HARDWARE
30', 18', 34'
LF
time-space channels
N1
N2
1
2
L1
L2
t_1
t_2
t_3
t_F

FIGURE 13

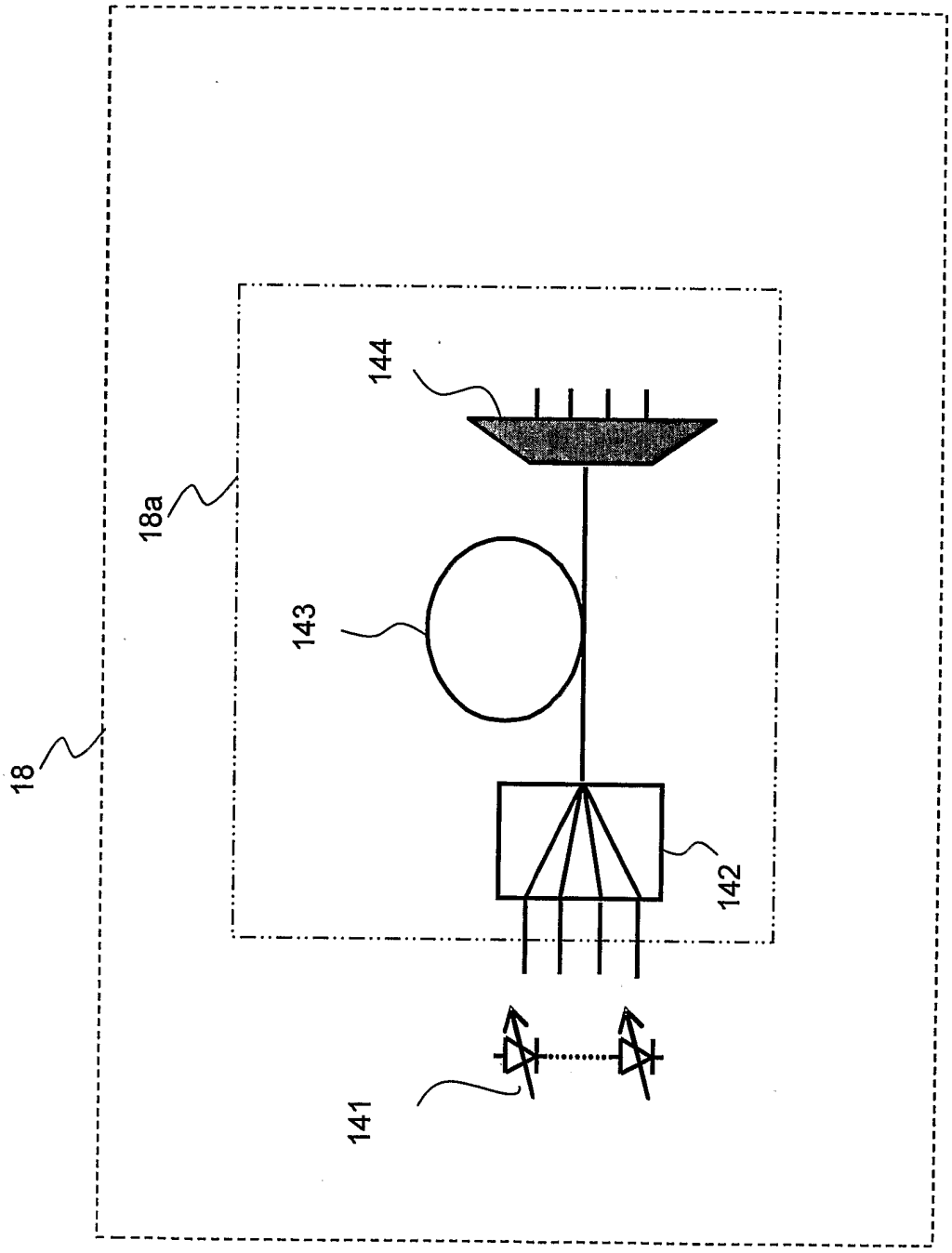


FIGURE 14

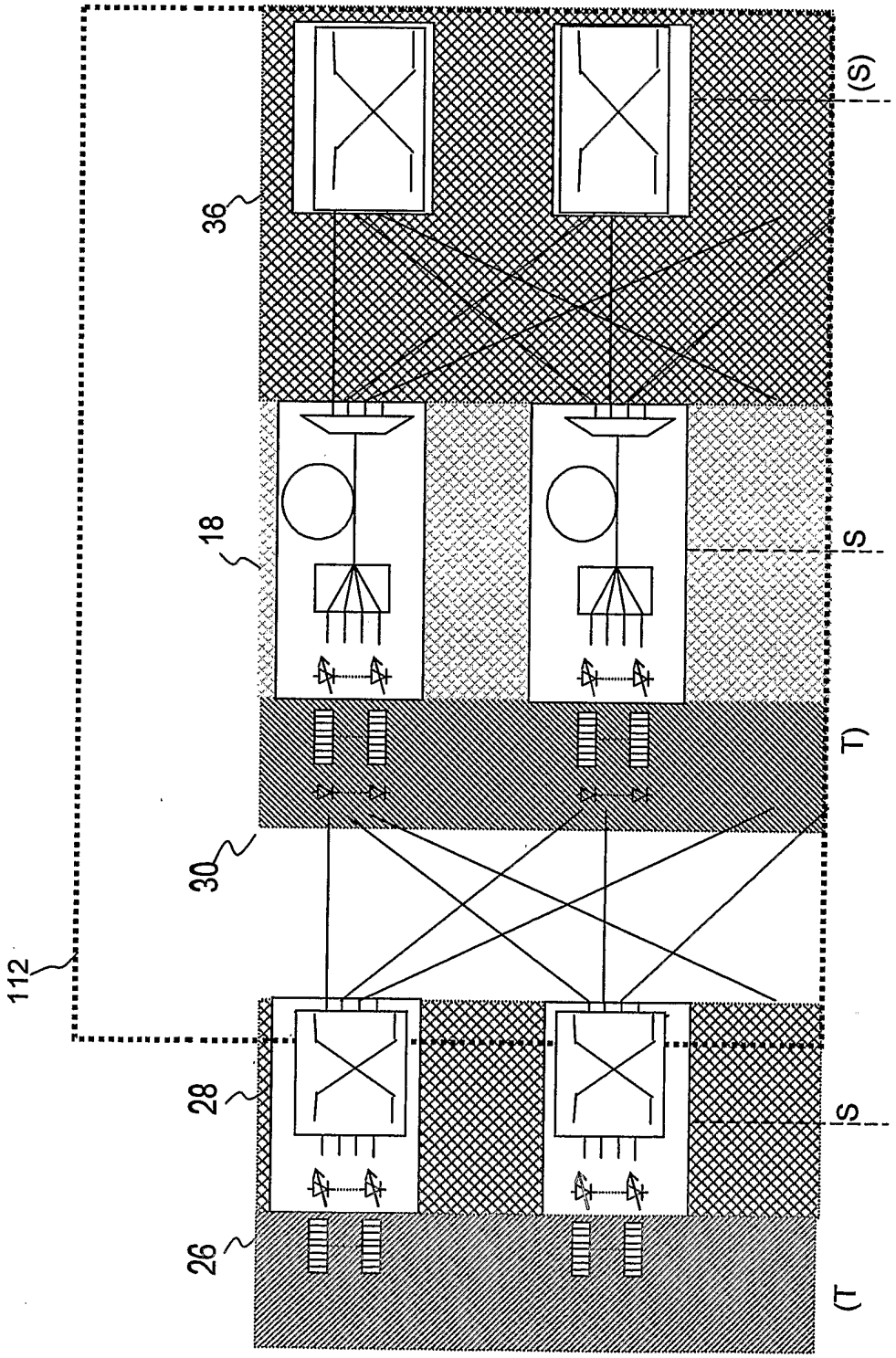


FIGURE 15

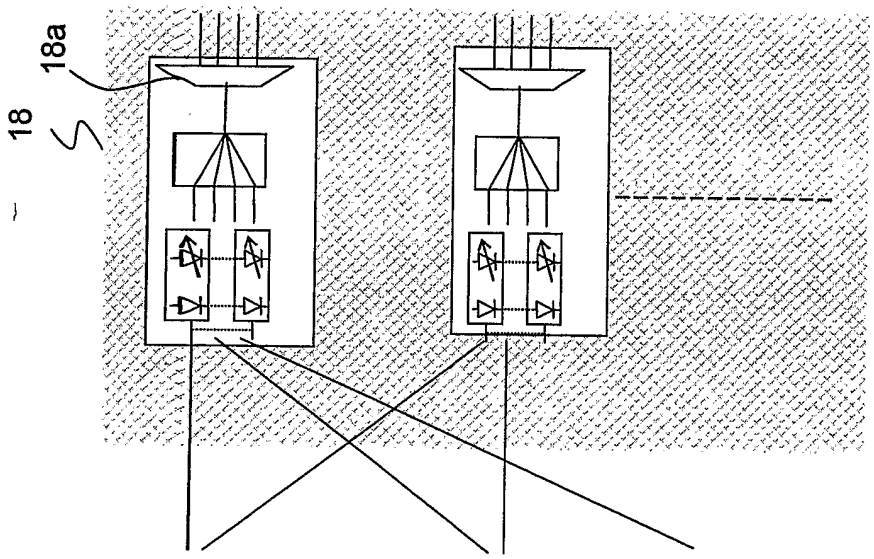


FIGURE 16

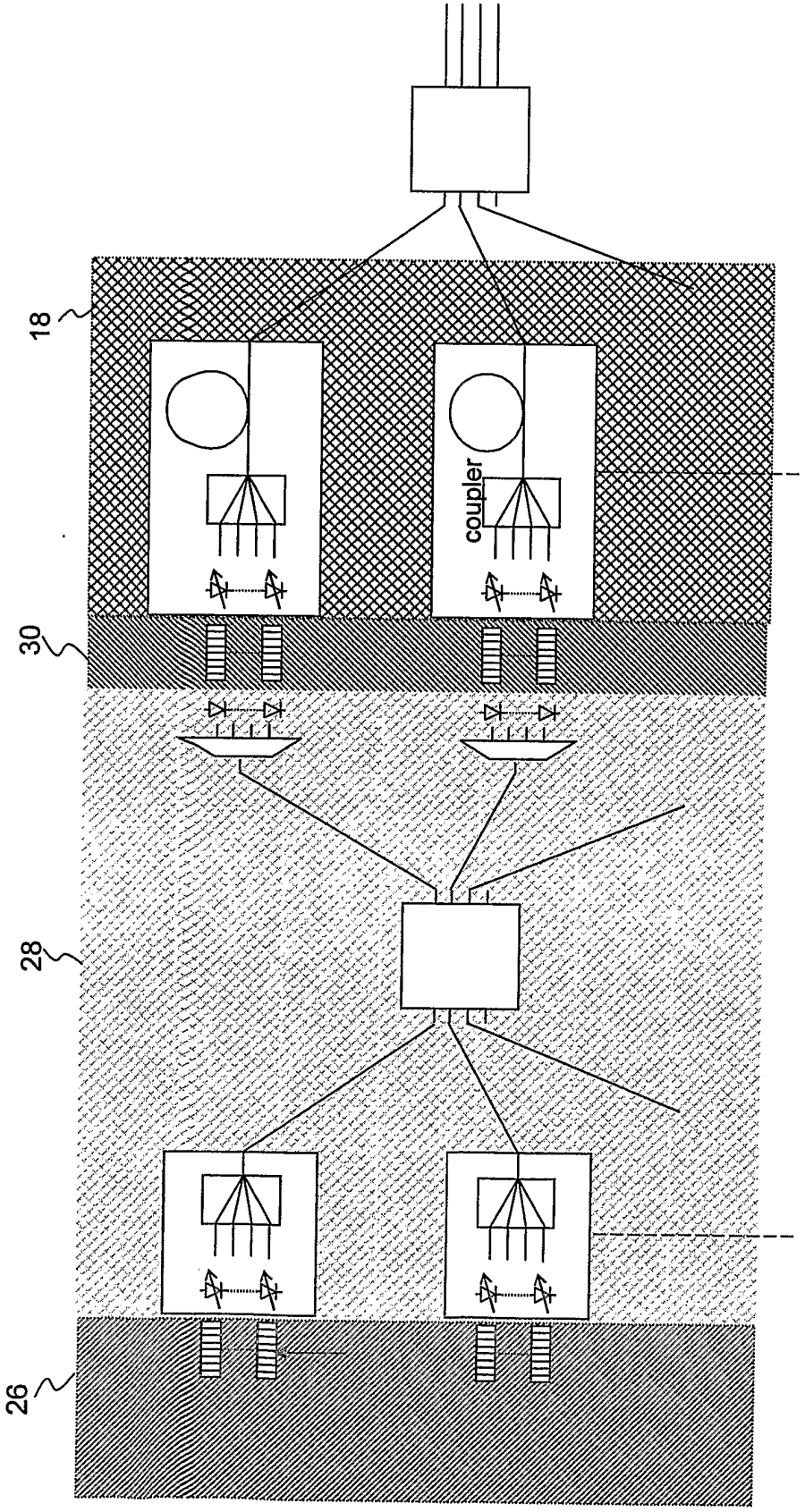


FIGURE 17

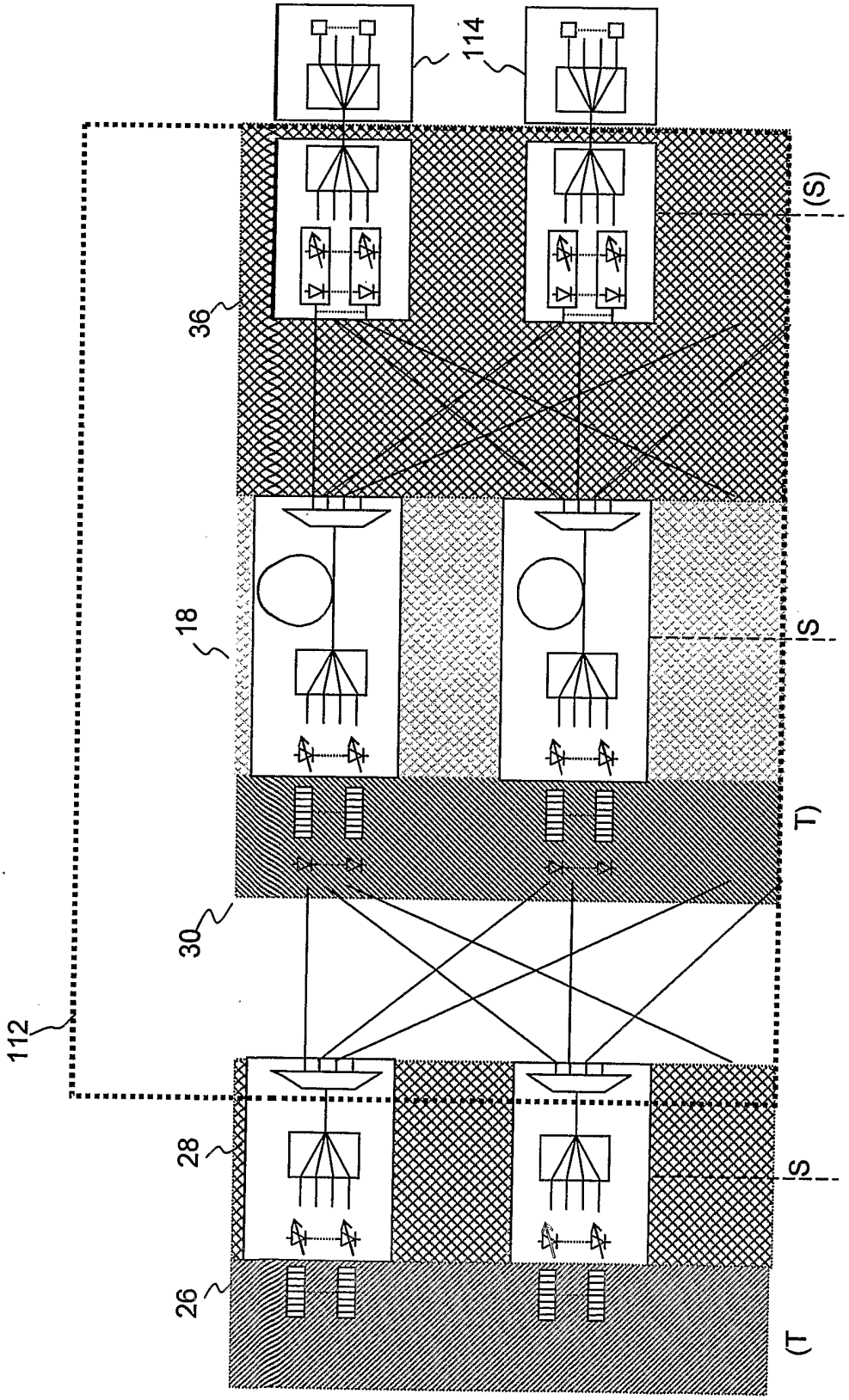


FIGURE 18

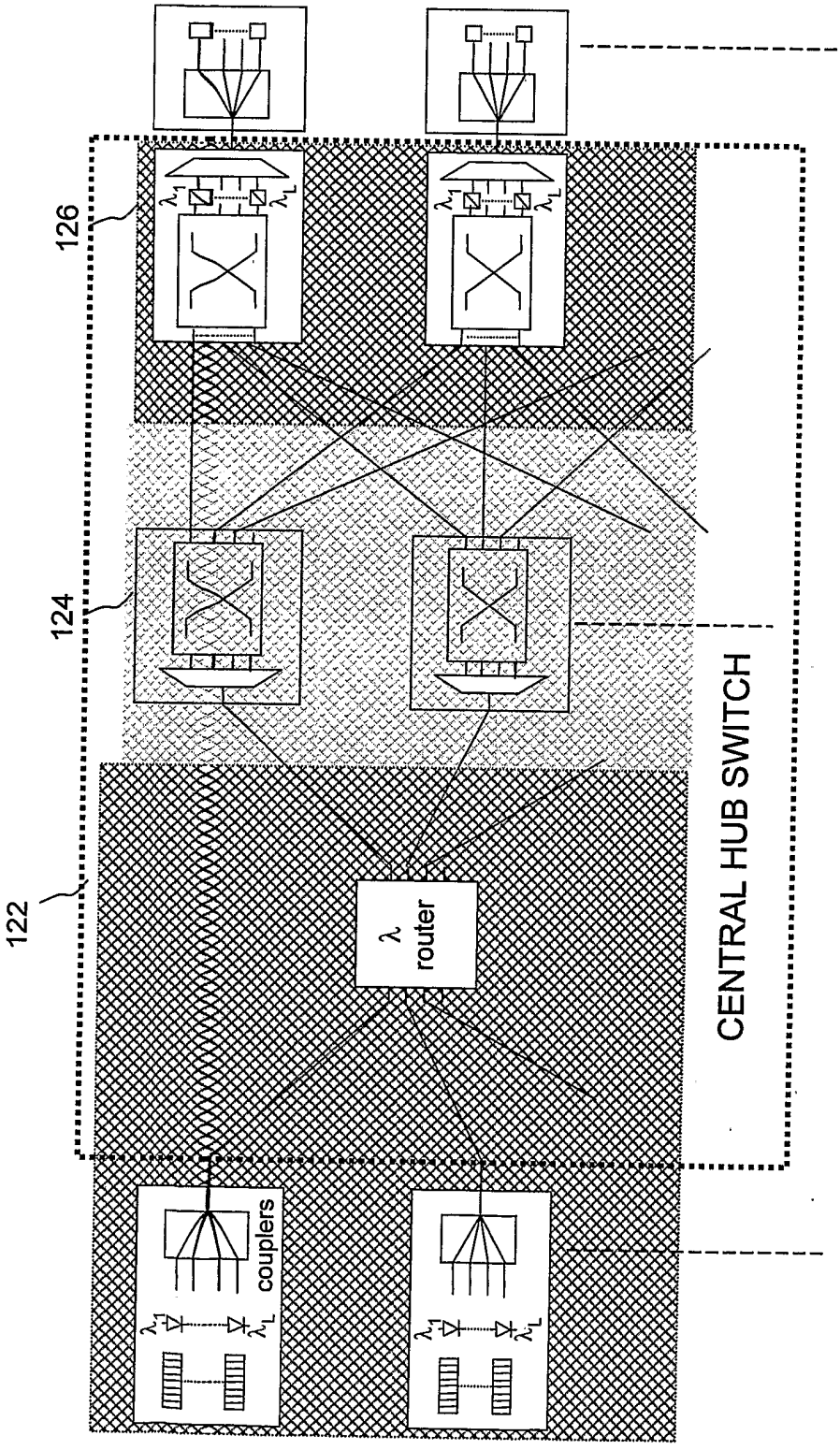


FIGURE 19

INTERNATIONAL SEARCH REPORT

International Application No
GB2005/003671

A. CLASSIFICATION OF SUBJECT MATTER
H04Q3/68 H04Q11/06 H04L12/56

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04Q H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, WPI Data, PAJ, INSPEC, IBM-TDB

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2002/093952 A1 (GONDA RUMI SHERYAR) 18 July 2002 (2002-07-18) paragraphs '0013! - '0015! paragraph '0041! paragraphs '0047!, '0048! paragraphs '0056! - '0058! claims 1,2	1,13,27, 29-32
A	US 2003/021267 A1 (WU EPHREM C ET AL) 30 January 2003 (2003-01-30) paragraphs '0019! - '0023! claims 1-3	1,13,27, 29-32
X,P	WO 2005/032166 A (BRITISH TELECOMMUNICATIONS PUBLIC LIMITED COMPANY; HILL, ALAN, MICHAEL) 7 April 2005 (2005-04-07) the whole document	1-32

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
30 November 2005	13/12/2005

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Barbelanne, A

2

Form PCT/ISA/210 (second sheet) (January 2004)

INTERNATIONAL SEARCH REPORT

nformation on patent family members

ational Application No
PCT/GB2005/003671

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2002093952 A1	18-07-2002	AU 7311801 A WO 0203594 A2	14-01-2002 10-01-2002
US 2003021267 A1	30-01-2003	WO 03007621 A2	23-01-2003
WO 2005032166 A	07-04-2005	NONE	