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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
17 April 2003 (17.04.2003)

PCT

(10) International Publication Number
WO 03/032540 A2

(51) International Patent Classification[7]: H04J 3/06, 3/07

(21) International Application Number: PCT/CA02/01515

(22) International Filing Date: 9 October 2002 (09.10.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/972,686 9 October 2001 (09.10.2001) US

(71) Applicant: NORTEL NETOWRKS LIMITED [CA/CA]; 2351 Boulevard Alfred-Nobel, St Laurent, Quebec H4S 2A9 (CA).

(72) Inventors: ROBERTS, Kim, B.; 10 Mission Inn Grove, Ottawa, Ontario K2R 1C6 (CA). GAGNON, Ronald, J.; 718 Seyton Drive, Ottawa, Ontario K2H 9R9 (CA). SHIELDS, James, A.; 14 Aspen Grove, Ottawa, Ontario K2H 9A1 (CA).

(74) Agents: HARRIS, John, D. et al.; Gowling Lafleur Henderson LLP, Suite 2600, 160 Elgin Street, Ottawa, Ontario K1P 1C3 (CA).

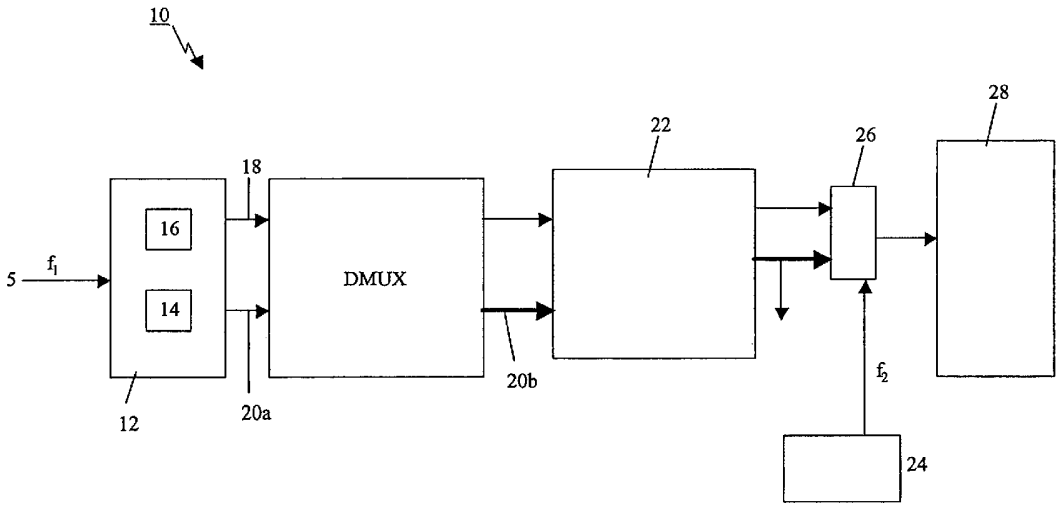
(81) Designated States *(national)*: CA, JP.

(84) Designated States *(regional)*: European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, SK, TR).

Published:
— *without international search report and to be republished upon receipt of that 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: METHOD AND APPARATUS FOR DIGITAL DATA SYNCHRONIZATION



(57) Abstract: A method and apparatus for digital data synchronization and desynchronization is provided. A multibit value is determined and communicated with a block of data at bit rate f_1 and mapped into a synchronous data stream of bit rate f_2 in a synchronizer. Bit rate f_1 may be derived from the multibit values in a desynchronizer. A method of rate-adapting asynchronous signals into synchronous signals is provided which minimizes waiting-time jitter.

WO 03/032540 A2

METHOD AND APPARATUS FOR DIGITAL DATA SYNCHRONIZATION

Background of the Invention

Optical communications networks predominantly follow the Synchronous Optical Network (SONET)/Synchronous Digital Hierarchy (SDH) synchronous standard for optical transport, with the SONET standard in North America and SDH elsewhere. For the purpose of this specification, while reference will be made to SONET, it will be readily appreciated that the disclosure herein is readily adaptable to the SDH standard.

The SONET standard defines rates, interfaces and formats, network operation, and network features, and the like, for high speed, high bandwidth transport of optical signals. SONET synchronously multiplexes signals which provides the advantage of single-step multiplexing and of cross-connect and add-drop capability. Further, sync characters and message framing for clock synchronization between network equipment is unnecessary as SONET relies on a synchronization network to transport timing reference between locations.

The base format of SONET is the synchronous transport signal level 1 (STS-1). The STS-1 frame has two main areas, transport overhead and synchronous payload envelope (SPE). The SPE is divided into two parts: the STS path overhead and the payload. The STS-1 frame is a fixed size of 810 bytes (6480 bits) and a frame length of 125 microseconds, or 8000 frames per second, with a line rate of 51.84 Mbps. STS-n signals are higher level signals formed by byte interleaving n STS-1 signals, where n is an integer. STS-1 signals may also be concatenated with a number of STS-1 signals forming an STS-nc signal, which will be multiplexed, switched and transported through the network as a single entity. An STS-3c has a single payload but is at a frame rate of 155.52 Mbps.

Incoming signals that comply with this standard are readily transported over SONET networks. Incoming data is mapped into the SPE. The data may start anywhere in the SPE due to asynchronous arrival of data from different

systems. A pointer mechanism (H1 and H2 pointer bytes in the line overhead) is used in SONET to indicate the starting address of the data in the SPE, thereby aligning and synchronizing frame to the SONET network. The pointer mechanism also allows straightforward connection from one network to another, until the incoming signal is delivered to the far end.

Incoming signals that do not comply with the SONET standard need specific synchronization in order to be carried in a SONET network. For example, incoming signals may be asynchronous, being of lower or higher in rate, than the local frequency.

Mapping of one rate or format into another rate or format is known. For example, Bellcore TR-0253 describes the standard mapping into SONET of some common asynchronous lower rate transmission formats such as DS1, DS2 and DS3, and the like. Similarly, standard mappings are also available for other electrical signals as E1. Generally, format specific hardware is required to map such specific sets of signals.

Most optical signal formats do not have such standard mappings into SONET. In fact, even a SONET signal itself can become a non-compliant incoming signal where, rather than the standard of passing of only the SPE, some or all of the overhead bits are passed as well.

Copending European Patent Application Number EP1067722, published January 10, 2001 and titled "Mapping Arbitrary Signals Into SONET", Roberts, et al., and copending Application Number 09/438,516, titled "Detection of Previous Section Fail for a Transparent Tributary", describe methods and systems of universally mapping incoming signals having a continuous format into SONET frames that can then be carried by a SONET network, thereby allowing any qualified format to be transparently transported. In these applications, incoming signals, even those varying significantly from the standard are uniformly mapped into SONET. However, it does not mitigate problems associated with waiting time jitter.

Jitter and wander are important system parameters and address transparency of data with respect to timing. Phase variations between 10 Hz

and 100 MHz are referred to as jitter. Jitter above the clock recovery bandwidth reduces system margin. Wander is phase variation below 10 Hz. SONET/SDH signals that carry timing reference have strict wander requirements. Jitter and wander are measured after data is desynchronized and recovered.

Waiting time jitter is a low-frequency phase jitter introduced whenever asynchronous digital signals are synchronized for multiplexing by stuffing. Stuffing techniques are commonly employed in the mapping process to rate adapt asynchronous signals to produce data signals of the appropriate SONET frequency prior to multiplexing, and thereby account for timing differences. A detailed discussion on the subject of waiting time jitter is contained in D. Duttweiler, *Waiting Time Jitter,* BELL SYS. TECH.. J., vol 51, pp. 165-207, January 1972. In this article, Duttweiler demonstrates that waiting time jitter is the result of imperfect sampling of the phase of incoming data to the synchronizer. At the desynchronizer, the reference clock of the incoming data is being recovered based on the imperfect phase samples.

Imperfect phase sampling is due to limited resolution in the expression of analog phase information with digital words, which is expressed as a stuff bit or stuff word. It is also due to limited resolution in time where there is one stuff opportunity at discrete times in the mapping domain.

Stuffing occurs at a synchronizer. Traditionally, data bits from incoming asynchronous signals are written into an elastic store (also referred to as an elastic buffer or FIFO) of the synchronizer for temporary storage, under the influence of a write clock derived from the incoming data of the respective asynchronous signals. A read clock is derived from a local clock, local to the synchronizer and synchronous with SONET, which determines the rate at which data bits are read out (outputted) from the elastic store. Where the write clock is not precisely synchronous with the read clock, overflow or underflow may occur in the elastic store resulting in a discontinuity, or "slip", in the output data signal destined for multiplexing. Usually, the read clock rate is

higher than the write clock rate with the consequence that the elastic store will be emptied.

The fill of the elastic store is constantly monitored and compared to a threshold by a phase comparator, the output being the phase difference between the write and read clocks. Where the fill is below the threshold level, the phase comparator and associated circuitry operate to inhibit the read clock and a stuff bit is inserted to the output signal. The resulting read clock is a gapped clock, with an irregular signal containing pulses separated by regularly occurring gaps for control bits and irregular occurring gaps for stuff bits. At the same time, the input data bits continue filling the elastic store. The output signal, following synchronization to the local rate by stuffing, is then mapped into SONET frames, which will then be subsequently multiplexed with other signals to form higher rate signals. Information regarding stuffing and stuff bit location is transmitted with each frame.

Waiting time jitter is a strong function of "stuff ratio", which is the ration of stuff opportunities used to the total number of stuff opportunities available. Stuff ratios close to fractions with small denominators, for example, 1/1, 1/2, 1/3, and the like, exhibit the worst jitter. This is because these stuff ratios result in changes in stuff opportunities that vary periodically over a long timescale, which results in a recovered clock signal with a low frequency content that is very difficult to filter out by a phase locked loop at the desynchronizer.

Generally, each stuff bit is identified in a frame by three stuff indicators for redundancy. This in turn affects the maximum amount of payload that can be carried in a frame, which in turn affects available bandwidth in the signal. Maximum convenient speed for running CMOS circuits is typically 150 Mb/s Further, bit stuffing is complicated in parallel implementations in complementary metal oxide semiconductors (CMOS), which is a presently employed technology used to implement digital circuits at reasonable speeds. This would limit bit stuffing to client rates up to 150 Mb/s.

Where the fill rate is higher than the output data rate, negative stuffing occurs, and extra data is transmitted through another channel.

Instead of bit stuffing, word stuffing may be employed. A word is a string of data units that is handled as a unit. The word size may be, for example, of 8 times n bits with n being the number of bytes in the word, with the result that, for example, 24 bits are stuffed at a time. Word stuffing is designed to be used in parallel track.

The delay in time between attaining the threshold level and the stuffing function produces low frequency jitter and wander in the output of the phase comparator. Waiting time jitter is a strong function of the stuff ratio. Waiting time jitter may have low frequency content close to 0 Hz, especially at low stuff ratios. Low frequency content is difficult to filter out at the desynchronizer.

More particularly, a phase locked loop with a low pass filter may be employed to filter out low frequency jitter. However, this requires a correspondingly larger elastic store to absorb jitter and wander. This is inefficient and results in greater delays.

Further, where word stuffing is employed, the waiting time jitter problem is substantially magnified, growing with about the cube of the word size, as the amplitude of the phase transient is proportional to the word size, the error due to rounding of the analog phase to digital information is proportional to the word size, and the spectral location of waiting time jitter is inversely proportional to the word size.

U.S. patent no. 4,811,340 issued on March 7, 1989 to McEachern et al. discloses a method of threshold modulation that changes the spectral location of waiting time jitter. More particularly, the low frequency jitter is changed to a higher frequency that is easily removed by a conventional PLL circuit at the desynchronizer. This technique reduces the amount of waiting time jitter for specific incoming signals but not to sufficiently acceptable levels, for example, for high speed applications.

Asynchronous Transfer Mode (ATM) is a standard that is asynchronous in nature. However, the transport of voice requires that the data be synchronized so as to maintain the temporal relationship between a speaker and listener. In simple point to point applications, two end points may be synchronized by two standard mechanisms: adaptive clocking or synchronous residual time stamping (SRTS). These mechanisms adjust the clock rate at one end of the circuit based on the clock rate of the other end. The adaptive clock adjusts clock rate by monitoring the depth of the SAR receive buffers. It these adjusts the clock rate of the "slave" end to maintain an appropriate buffer depth. On the other hand, SRTS monitors the rate of the line clock at the "master" end of the circuit, with reference to a standard clock. The difference between the two clocks is then encoded and transmitted as part of the data stream. At the slave end, this "difference" signal is retrieved and used to adjust the slave clock by reference to the difference signal and the standard clock.

In this manner, the ATM standard for transporting and time stamping bit-streams explicitly communicates the frequency of a data signal. However, it assumes that the far end reference frequency exactly matches the head end reference frequency. This is generally not the case. As signals frequently cross asynchronous boundaries, temporary failures of the exact network synchronizations occur. That method suffers from excessive wander or failures in these cases.

There remains a need for a universally applicable low jitter method of synchronizing and desynchronizing asynchronous signals for transport on a synchronous network.

Brief Summary of the Invention

The present invention seeks to provide a synchronizer and desynchronizer, system and method, which minimize the above problems.

According to one aspect of the invention, there is provided a synchronizing element and method of synchronization in a system for

transporting a data signal of bit rate f_1 in a data stream of bit rate f_2, where the ratio of f_1 and f_2 is variable. The synchronizing element determines the bit rate f_1 of an incoming data signal and the bit rate f_2 of the data stream to which the incoming data is to be rate-adapted. A relationship is then determined between two bit rates f_1 and f_2. The relationship is expressed as a value of significant resolution, which may be communicated with the data signal.

In an embodiment, the relationship is based on the phase of the data signal and the data stream, and expressed in the form of a digital multibit value. The multibit value may also represent an average fraction of a data clock that is ungapped and the average fraction of a read clock that is ungapped.

The relationship between f_1 and f_2 may be determined within a circuit analogous to a digital phase-locked loop. A detailed phase comparator may be provided for determining phase of the incoming data signal to a fine resolution over a specified time interval.

A multibit value so-determined for an internal of a data signal may be communicated with a corresponding block of data which will be mapped into a synchronous data stream. The multibit value may also be combined with a stuff indicator to be used in mapping function. In another embodiment, the determination of the relationship is decoupled from the level of fill of data store, including a data store where the data output is gapped so as to produce word stuffing.

In another aspect, there is also provided a desynchronizing element and method of desynchronization for reconstructing a data signal of bit rate f_1 from a data stream of bit-rate f_2, the data signal having been transported in the data stream, the ratio of f_1 and f_2 being variable. The desynchronizing element has a phase transfer function that has a substantially broad bandwidth. In an embodiment, the desynchronizing element has a voltage controlled oscillator and a phase-locked loop. The VCO suppress phase noise from the phase-locked loop and the phase-locked loop significantly mitigates jitter. The PLL may have a loop control bandwidth greater than 1kHz or 100 kHz in some embodiments.

Phase error determination and correction occurs in the desynchronizer. In an embodiment, the data store exists in parallel, separate from the phase error determination and correction function. The output from the data store is gapped so as to produce word stuffing Multibit values previously communicated with a corresponding block of data may be used to derive the bit rate f_l.

Advantageously, a transparent and universal method of rate-adapting asynchronous signals into synchronous signals is provided which minimizes waiting-time jitter.

Brief Description of the Drawings

Other aspects and features of the present invention will become apparent to those ordinarily skilled in the art upon review of the following description of specific embodiments of the invention in conjunction with the accompanying figures in which:

Figure 1 depicts a communications network.

Figure 2 depicts a transparent synchronizer in accordance with an embodiment of the invention.

Figure 3 depicts a synchronizer circuit of Figure 2.

Figure 4 depicts a digital synchronizer PLL of Figure 3.

Figure 5a depicts a graph representing the actual phase of a data signal over time, overlaid with phase estimates of the data signal, in accordance with an embodiment of the invention.

Figure 5b depicts the graph of Figure 5a where phase estimates are taken at fixed clock intervals synchronous to a local clock

Figure 5c depicts one phase estimate over one sampling interval in detail..

Figure 6 depicts a transparent desynchronizer in accordance with an embodiment of the invention.

Figure 7 depicts a desynchronizer circuit of Figure 6.

Figure 8 depicts a desynchronizer PLL of Figure 7.

Like numerals denote like features in the drawings to facilitate an understanding the invention.

5

Detailed Description of the Invention

This invention relates to transparent ports, which are data terminals, which map (in the case of synchronizers), or reverse maps (in the case of 10 desynchronizers), data of arbitrary rates into, or from, synchronous frames such as SONET or SDH frames.

By way of background, as summarized in Figure 1, a plurality of services are transparently transported over a synchronous network such as SONET. Continuous format digital data signals are generated by local users 2, 15 such as terminals or access networks, and driven by (timed to) a local reference clock 4. Each signal has a bit rate f_n, which is a measure of the rate of digital transitions in the signals, and has a particular format associated with it. As such, bit rates may vary from data signal to data signal.

The digital signals are transmitted to a plurality of terminals 6 in a 20 synchronous network, the terminals are each provided with one or more synchronizers and/or desynchronizers (not shown). The terminals 6 may be backplane digital cross-connects or digital switches, and the like, and are junctions where data signals are received and retransmitted.

At terminals 6 provided with synchronizers, data bits from a number of 25 signals are received by a received and recovered, mapped by a number of synchronizers into envelopes of a synchronous signal, such as STS-1, multiplexed in a multiplexer with other synchronous signals into a high rate signal and launched by a transmitter over a synchronous optical network 8 to an appropriate site.

30 The size of the frame is selected in software for satisfactory bandwidth utilization for desired purposes. Similarly, the higher level signals into which

lower level transport signals are multiplexed, and as the case may be, concatenated, for example, STS-3nc, is selected for desired purposes, including high speed, high bandwidth transport.

At terminals 6 provided with desynchronizers, optical receivers recover data signals from the high speed synchronous data stream, the demultiplexer separates out the lower rate data signals and presents the data signals to various desynchronizers, which rearranges bits into the respective formats associated with each data signal. The recovered data signals are then launched by transmitters to an associated network, or an associated end user, at the far end.

Within the synchronous system, the average frequency of all clocks in the system is the same or nearly the same, although there may be a degree of phase differences between transitions of signals within specific limits. Each clock can be traced back to a stable primary reference. In a SONET network, a STS-1 signal bears a rate of 51.84 Mb/s, an STS-3 bears a rate of 155.520 Mb/s, while an STS-192 bears a rate of 9953.280 Mb/s. This synchronicity enables the stacking of multiple synchronous signals without bit stuffing.

A terminal 6 has an internal clock that is derived from and is synchronized to a building integrated timing supply (BITS), the clock being traceable to a primary reference clock (PRC), and accordingly is referenced to a synchronous rate. The terminal provides timing on its outgoing OC-N signal. The internal clocks of other nodes are timed by the incoming signal.

Reference clocks driving data signals from users are generally asynchronous to the reference clocks of the synchronous network. Indeed, generally in multipoint services, signals may cross a number of asynchronous boundaries, from one network to another, before it reaches the far end. These signals are required to be appropriately synchronized in order to be transported in the synchronous network.

Referring to Figure 2, a single data signal and a single synchronizer 10 in the SONET standard is depicted to facilitate an understanding of the invention. The signal 5 is an optical carrier originating from a user, modulated with a continuous data stream and comprises a digital optical data signal 5, for

example, a 10 gigabit OC-192 signal 5. The signal 5 originates from a user and is driven by a local reference clock of arbitrary rate. The signal 5 is converted from an optical signal into an electrical signal in an optical to electrical converter (not shown) and is received into a transparent synchronizer through an input optical fibre into a data recovery unit 12.

The data recovery unit 12 recovers data bits from signal 5 and includes a receiver 14 and an agile clock recovery circuit 16. The receiver 14 includes a data recovery circuit (not shown). The agile clock recovery circuit 16 is a phase locked loop (PLL) and is capable of clock recovery over a broad continuous range of bit rates. The agile clock recovery circuit 16 determines the average bit arrival rate of data from signal 5 thereby deriving a data clock 18 from the signal 5. The data clock 18 is a simple clock, reflective of the bit rat of the signal 5, and is not gapped. An example of such a clock recovery circuit is disclosed in U.S. patent application 09/218053 filed on December 22, 1998, entitled "Apparatus and Method for Versatile Digital Communication" by Habel et al., assigned to Nortel Network Corporation.

The data clock 18 and the data bits 20a (or "data") are of the same bit interval and are outputted from the data recovery unit. The data 20a may be demultiplexed into multiple parallel tracks for signal processing at lower rates. In Figure 2, the data 20a is demultiplexed in multiple stages, first by a factor of 8 and then by a factor of 12, to provide 96 parallel tracks of data 20b. The bit interval in the data clock 18 is correspondingly increased in multiple stages.

Both the data clock 18 and the data 20b are passed into a synchronizer circuit. In Figure 2, the synchronizer circuit 22 is a field programmable gate array circuit in which the data bits 20b are mapped into a format suitable for a synchronous standard, for example, SONET SPE frame of a STS-1 signal. The mapping function is synchronized using an internal local SONET clock 24 of bit rate f_2, which is a simple clock synchronous to the SONET standard.

In an embodiment, the data bits 20b are mapped into SONET frames mapping with evenly interspersed fixed stuff bits and adaptive stuff bits by a mapping unit 26 in accordance with a mapping algorithm. The frame has two

areas: transport overhead (TOH) and the synchronous payload envelope (SPE). The TOH is composed of section overhead (SOH) and line overhead (LOH). The SPE is further divided into two parts: the STS path overhead (POH) and the payload. Data bits 20b are carried in the payload. The overhead information facilitates multiplexing and has OAM&P capabilities.

The mapping unit 26 divides the frame into evenly sized blocks. For example, an STS –192 SPE can accommodate 1138 such blocks. The mapping unit 26 also determines the number of fixed stuff bits for each block. Each block includes data bits, fixed stuff bits and adaptive stuff bits. As an example, each block may comprises a data field having 1023 bits for data, a control field having 16 bits, and a spare field having 17 bits for other use. The number of adaptive stuff bits to be inserted in a block are determined based on a stuff indicator β, as will be further discussed below.

Based on β, the mapping unit 26 defines a valid location, which is a location for a data bit, and an invalid location, which is a location for a stuff bit for the next block based on the phase information accumulated when the data bits of the current block are mapped. In addition, the mapping unit 26 also distributes evenly the overhead bits at the time of the actual mapping, but realigns them in appropriate timeslots provided according to the SONET standard after mapping so that the frame is recognized by SONET equipment. An example of the mapping function is set out in Copending European Patent Application number EP1067722, published January 10, 2001 and titled "Mapping Arbitrary Signals Into SONET", Roberts, et al.

Rather than stuff bits, stuff words may be used, for example, n*8 bits word size, for example, so as to accommodate high rate mapping in CMOS.

The data mapped into the SONET frames of a SONET signal is of the synchronous rate f_2 and is multiplexed in one or more stages with other SONET signals into a higher rate data streams which are then inputted into a suitable bandwidth manager 28, converted from electrical into optical signals, for routing into a SONET network

Referring to Figure 3, the synchronizer circuit 22 is depicted in greater detail. The synchronizer circuit 22 is a digital circuit, for example, a field programmable gate array (FPGA), a programmable ASIC, or the like, and includes a framer and gapper unit 30, a buffer unit 32, and a mapper unit 26.

The data bits 20b and the data clock 18 are passed into the synchronizer FPGA. In the example of Figure 3, the data clock 18 rate f_1 is 100 Mb/s and the data bits have a bandwidth of 96 tracks. For processing at lower rates, the data bits 20c are further demultiplexed to 192 parallel tracks and the bit interval of the data clock is correspondingly increased to 50 Mb/s. The data clock 18 of the appropriate bit interval and the data bits 20c are inputted into the framer and gapper unit 30.

The framer and gapper unit 30 provides framing, error checking and counting, code conversion, parity correction, and other performance monitoring functions. Additional functions are performed by the framer and gapper unit but are dependent upon the particular type of service.

The framer and gapper unit 30 may also be provided with a tributary control system (TCS) (not shown), which is a microprocessor that receives clock rate information on line from the receiving unit 12, examines the data rate, selects the logic configuration of the gapper unit adequate for that data rate, and configures gapper unit accordingly. The TCS contain circuits specific to the client data and clock rate.

The framer and gapper unit may also have minimal functionality and merely pass on the clock, or divide the clock.

Within the framer and gapper unit 30, a multibit frequency value, or "multibit value" is determined in association with the data clock 18. Using a multibit clock circuit (not shown), multibit values are generated for every cycle of the data clock 18, thereby producing a data multibit clock 34 of multibit values of rate f_1, in association with the data clock 18. The multibit value is based on the word size of the data clock 18, namely 2^n, where n is the bit size of the data clock 18. For example, a 16 bit data clock has a constant multibit clock word value of 2^{16}, or 65536. This value also represents the maximum multibit

value for a 16 bit clock. Similarly, the maximum multibit value for a 32 bit clock would be 2^{32} or 4294967295.

The framer and gapping unit 30 may also provide removal, or "gapping", of overhead bits that are desired not to be mapped into an SPE frame, for example, SONET Line and Section Overhead bits, thereby increasing mapping efficiency. Where overhead is removed, in this embodiment the multibit value is varied based on the number of bits removed. For example, SONET Line and Section Overhead are removed (gapped), representing the removal of 3 of 90 bits (3/90 bits). This in turn represents the retention of 87 of 90 bits (87/90 bits). This fraction is then applied to the 2^n value, in this example, 2^{16} or 65536, to produce a value of 63351.466667. This value is rounded to the closest integer to produce a multibit value of 63351 for the particular cycle of the data clock 18. An error value is then computed as the difference between the fractional multibit value and the integer value (0.466667 in this example). This error value is retained for the calculation of the multibit value for the next data clock cycle. It is added to the fractional multibit value before rounding to the closest integer to produce the next multibit value (in our example, 63351.466667+0.466667=63351.93332, resulting in a multibit value of 63352 for the next data clock cycle. This is required to ensure that there is no error accumulation due to rounding.

In this manner Multibit values also represent the fraction of gapping in terms of the maximum multibit clock value and the associated simple data clock. The multibit clock value is an instantaneous phase step. It conveys frequency only through its associated simple data clock. It communicates frequency by indicating gapping information in relation to its associated simple clock. For example, if the multibit value is the full scale multibit value, this indicates that there is no gapping. In this case, the frequency of the multibit clock is the frequency of its associated simple clock. If the multibit value is some fraction of the full scale value, the frequency represented by the multibit clock is the frequency of the simple clock scaled by that same fraction.

Multibit values are transferred in the form of integers and may be communicated every cycle or less frequently, for example, every fourth cycle or every eighth cycle, to allow for more processing time. The stream of communicated multibit values constitutes a data multibit clock 34.

The decimal component (ie. the fractional part) of the value, namely: the .466667 in this example, is retained and used for the correction of inaccuracies due to rounding. Periodically, a particular multibit value in the multibit clock 34 is increased or decreased by an integer, depending on whether there was rounding up or down to obtain the integer, to compensate for the decimal components. For example, a multibit clock 34 may communicate the following stream of multibit values at a particular internal of time:

63351 63352 63351 63351 63351 63352 63351 63351 63351 63352

The data bits 20c from the framer and gapper unit 30 are written into the buffer unit 32, for example, a FIFO elastic store, in association with a write clock 18w, which is used to clock the input of data into a buffer. The write clock is derived from the data clock 18 of the asynchronous signal.

If not all data is written into the FIFO 32, the write clock 18w will be appropriately gapped (ie. data at the appropriate byte location will not be written into the elastic store).

Data is emptied, or read 20d, from the FIFO 32 into the mapper unit 26 under the control of a mapping clock 24m, which is a read clock driven by the SONET clock 24. The SONET clock 24 is gapped at appropriate instants as determined by the mapping algorithm in the mapper unit. In this way, the data bits 20d are synchronized to the mapping frequency. The mapping clock 24m is also referenced so as to maintain the FIFO fill at about half so as to avoid buffer depletion or overflow, which would result in lost data.

In the mapper unit 26, the data bits 20d are read out of the FIFO 32 and mapped into the respective SONET SPE frame in blocks with evenly interspersed fixed stuff bits and adaptive stuff bits, as set out above, under the control of the mapper clock 24m derived from the SONET clock 24. Large

blocks may be used for bit rate efficiency. Further, stuff words may be selected to be of a size that scales linearly with n in an STS-3nC frame.

A block clock 24b also derived from the SONET clock 24 is appropriately gapped, to maintain room for TOH bits, POH bits, and fixed stuff bits, of the SONET overhead. The block clock 24b clocks out data bits, fixed bits, and adaptive stuff bits from the mapper unit 26. The mapper clock 24m has the same rate as the block clock 24b but is further gapped under the control of β.

The value β represents the number of data words (ie. valid data) used within a fixed block. These data words are uniformly distributed across the block. β is analogous to a traditional stuff bit indicator. β is a value between 0 and the maximum word size of the block. The remaining available words in a block are stuff words. β is added to a block and is used in mapping the next block. The manner in which β is generated is set out in greater detail below.

The mapping unit 26 also generates a SONET multibit clock 36 based on multibit values per cycle of the SONET clock 24. The multibit values represent the bit rate of the SONET clock 24. The bit rate of the SONET multibit clock is f_2, based on the SONET clock rate 24. As with the data multibit values, the SONET multibit values represent the fraction of SONET clock used to read out data bits (ie. ungapped) to a maximum multibit clock. Again, for a 16 bit clock, the maximum multibit value is 65536.

A control loop unit 38 generates the β value that is included per block and is used by the mapper unit 26 in justifying and mapping the asynchronous data signals into appropriate SONET frames. The control loop unit 38 further generates a frequency value F that represents the relationship between the bit rate f_1 of the asynchronous data signals and the bit rate of a SONET signal f_2, expressed as fraction of a multibit word for high resolution.

More particularly in this embodiment, the control loop unit is analogous to a Phase locked loop (PLL), as later described, and is referred to as a digit PLL 38. The Digital PLL 38 calculates F which represents the frequency of the asynchronous signals of rate f_1 in a specified sampling interval, in this case,

phase sampling interval, which is a specified multiple of the SONET clock rate f_2, and is a fraction representing the data clock rate $f_1$over the SONET clock f_2 rate, expressed as a multibit word. In other words, it calculates F as the block relative frequency, which is a non-integer value F, which represents the exact frequency (to an arbitrary accuracy) of the data signals in units of words per block relative to the ungapped SONET clock 24 at the synchronizer. F is a higher resolution indicator of fractional frequency information than a traditional stuff indicator inside a circuit.

The value β is an integer output truncated from the value (F + the last residual fraction). The fractional part of the above becomes the next residual fraction.

Values for F and β are transmitted with each block of words. Because the synchronizer and desynchronizer are synchronous, being ports of network elements in a synchronous network, the data clock 18 bit rate f_1 can then be recovered at the far end at the desynchronizer based on the F values transmitted with each block, as further detailed below. Traditionally, this frequency was reconstructed at the desynchronizer.

Referring to Figure 4, the digital PLL 38 is described in greater detail. The 16 bit data multibit clock 34 from the framer and gapper unit 30 is inputted frequency to phase converter such as into a digital summer subcircuit 40, which continuously adds together the multibit values received to a 32 bit accuracy. The sum is periodically sampled. For example, the digital summer 40 may sum multibit values and sampled every 10 or 100 cycles, as selected. The summed output yields phase information. The output is inputted into a phase comparator 42, such as a digital subtractor.

The SONET multibit clock 36 is generated in the mapper unit 26 and is formed using the F value from previously transmitted blocks to gap the SONET clock 24 (with its associated simple clock). As set out above, transport overhead, path overhead and block overhead have already been gapped. Preferably, all overhead not available for data is gapped.

The SONET multibit clock value is further reduced by the amount of the average number of gaps transmitted (ie. gaps in relation to available data words not being used for incoming data (ie. stuff bytes) are removed) in an *f*-gapping subcircuit 44. The *f*-gapping unit 44 is analogous to a voltage controlled oscillator in that the *f* value is a frequency that is imposed through the gapping function, which is analogous to voltage controlling the frequency of a VCO. Having regard to the proportion of data bits unavailable due to overhead in a given clock cycle, the remaining available data bits is reduced by the proportion of bytes used for data. The resulting proportion is to be applied to the corresponding SONET multibit values.

For example, in a given block, 10% of the bytes in a block are unavailable for data or stuffing being overhead. As such, 90% of the bytes remain available. Of this 90%, half of the bytes are used for data. Therefore 90% x 50% = 45% of the block is non-data and accordingly stuffed with stuff bytes. The 45% proportion is applied to the corresponding multibit value to generate a multibit value reflecting the *f*-gapping and overhead gapping, for example 45% x 65536 = 29491.2 The fractional part of *f* will not accumulate because of the synchronizer PLL action that compensates for any errors due to the fractional part of *f*. Therefore it is not necessary to pass the 0.2 through the feedback loop.

The multibit values from the SONET multibit clock 36 as reduced by the proportion computed following the *f*-gapping unit 44 are continuously summed in a frequency to phase converter 46 such as a digital summer and integrated to yield phase information

The digital subtractor receives the inputted phase information from the data multibit clock 34p and from the SONET multibit clock 36p and relates the phases. The SONET multibit phase values are subtracted from the data multibit values, the value of the difference being an error signal ϵ. In determining the difference in phase (ie. phase imbalance) between the data multibit clock 34p and the SONET multibit clock 34, the digital subtractor

operates as a phase comparator, where traditionally, the fill of the FIFO operated in that capacity.

A constant offset reference value R from the FIFO 32 is also inputted into the digital subtractor 42 to be subtracted from the data multibit phase values. In this manner, the fill of the buffer can be kept centred.

The output ϵ of the phase comparator 42 is inputted into a low pass filter (LPF) 48 that maintains and outputs a running average of the error signals. It smoothes out any rapid changes in error signals (changes that occur at a frequency above its cutoff frequency) to enable the control loop to converge to a correct and stable value. Preferably, the LPF 48 is selected to maintain a whole loop bandwidth, for example, of approximately 200 Hz. An output F from the LPF is then returned to the *f*-gapping unit 44 for use in *f*-gapping of subsequent SONET multibit values.

The combination of the *f*-gapping unit 44, the SONET multibit clock frequency to phase converter 46, the phase comparator 42 and the LPF 48 collectively operate as a PLL, respecting the data multibit clock 34.

The value *f* is added to the block overhead for a block, and is transmitted with the next block. *f* is transmitted with every block *f* is added to block overhead (ie part of the control field) in the mapper unit.

For example, assuming a 16 bit clock and a block size of a maximum of 1024 words, and an *f* value of 20000, β may be calculated as follows:

$$F_n = \frac{f}{2^{16}} * \text{max. no. words/block} = \frac{20000}{65536} * 1024 = 312.5$$

β_n = integer portion of $(F_n + \rho_{n-1})$

ρ_n = fractional portion of $(F_n + \rho_{n-1})$

where n represents the particular time of the particular block.

β is added to the block overhead of the next block in the form of an integer. The fractional portion ρ is retained for the next calculation of β to be included in the overhead of the next block.

Preferably, block overhead is Forward Error Correction (FEC) encoded, so as to provide a means to ensure that the block overhead associated with the F and β values are transmitted with high reliability when received.

While the values for F and β are included in block overhead for each block transported, alternatively, the values may be included in each block in the form of F-β and β. In this manner, a degree of data compression is achieved. Alternatively, Σ(F-β) and β can be transmitted. This provides a degree of data compression as before and by transmitted ΣF rather than F, any corrupted value of ΣF avoids any long term phase error and causes only a phase transient.

Additional high resolution phase correction may be provided so as to ensure the generation of accurate F values outputted from the phase comparator 42.

The data multibit clock 34 and the SONET multiple clock 36 are each timed from their corresponding simple clocks. The corresponding simple clocks have no harmonic relationship to each other and are generally asynchronous.

The instant of time at which a data multibit clock 34 is digitally integrated by the digital summer 40 will not generally correspond to the timing instant for the SONET clock 34. The time difference resulting in imperfect sampling creates waiting time jitter.

More particularly, referring to Figure 4, the phase of the data multibit clock 34p is determined by the integration of the sum of the multibit values received by the digital summer 40. It is a continuous function that is synchronous to the data's simple clock 18. Graphically represented as a function of time, yields a constant linear relationship between real phase and time, as exemplified in Figure 5a. However, as the data phase information is only updated periodically (ie. the integration of the summed data multibit values is performed periodically) in fixed intervals, for example, once every 100 cycles, the estimated phase as a function of time yields a stepped relationship, also as set out in Figure 5a.

Referring to Figure 4, phase detection is performed by the digital subtractor 42 using phase information derived from the data multibit values and the SONET multibit values. The SONET multibit clock 36 and the phase comparator 42 operate synchronously to the SONET simple clock 24, which is generally asynchronous with data simple clock 18. As a result, phase comparison of the clock phases are conducted asynchronously. As depicted in Figure 5b, there is a varying amount of lag time between the SONET clock sampling interval and the interval in which the data phase information is updated, resulting in an underestimation of the phase of the data multibit clock 34p. The lag time difference represents waiting time jitter. Because the lag time varies for different phase samplings at the phase comparator 42, it cannot be averaged or filtered.

The jitter may be corrected by obtaining a real time measurement of the time lapse between the last update of the SONET multibit clock and the time at which phase difference is estimated at the phase comparator/digital subtractor. Referring to Figure 4, a detailed phase detector 50 uses a real time measurement (ΔT) of the difference in sampling interval between the data multibit clock 34 and the SONET multibit clock 36 and computes from it a phase error correction as described below. The real time measuring function (ΔT) may be performed by an ASIC sub-circuit, capacitor chain, external components, or the like, and receives frequency input from the data multibit clock 34 and from the SONET multibit clock 36 and measures the real time lapsed. The time lapse is measured to an adequate resolution, for example, 0.1 nanosecond resolution.

Referring to Figure 5c, the phase step at the data multibit clock integration instant is the data multibit clock value F. The real time measurement is the time lapse x. The detected phase error measurement y can be determined by scaling the data multibit clock value F as follows:

x / integration interval = y / F

Or alternatively, y = F* [x / integration interval]

where F is the actual multibit clock value, with *f* being the local and imperfect estimate of F.

Once *y* (the phase error for a sampling interval) is determined, this value is inputted to the phase detector/digital summer and added to the integrated phase values for the sampling interval to derive an actual data phase value, as follows:

Actual data phase = integrated phase values + F* [*x*/integration interval].

In this manner, waiting time jitter may be eliminated within the arbitrary resolution of the phase detection and the multibit clocks.

Figure 6 is a block diagram depicting an overview of a transparent desynchronizer 60. A single desynchronizer is depicted to facilitate an understanding of the invention. An optical to electrical backplane interface 62 is an ASIC electrical interface and cross-connect. A high rate, SONET signal 64 comprising appropriately multiplexed signals is received from the SONET network and is converted from optical to electrical signals at the backplane. A plurality of parallel data tracks 64a (for example, 5) are outputted from the backplane and inputted into the transparent desynchronizer 60.

The SONET signal 64a may be demultiplexed, for example, into 80 parallel data tracks, and inputted into the desynchronizer circuit 68, in this example, also an FPGA. As will be further detailed below, the desynchronizer performs a reverse mapping function in this embodiment using the F and β values sent with each block to extract the original data bits, absorb the fixed stuff bits and the adaptive stuff bits and for clock recovery of the original data clock 18. SONET overhead and stuff bits in the SPE are removed. The desynchronizer 60 is synchronized to a clock 66, which is a clock synchronous to the SONET network.

The data signal 5, recovered in the original format and data rate f_l, is received by a transmitter, converted from electrical to optical format, and transmitted to the associated network or local user in optical format.

The data received from the payload of the SPE frame is presented to the mapper, and includes data bits, together with the fixed and adaptive stuff bits.

Referring to Figure 7, the desynchronizer FPGA 68 is depicted in greater detail. The SONET signal 64a multiplexed into 80 parallel paths and the local SONET clock 66 are inputted in a pointer processor/gapper unit 70. The pointer processor/gapper unit 70 includes a pointer generation function. The pointer is an offset value that points to the byte where the SPE begins thereby compensating for frequency and phase variations (including transmission signal wander and jitter) by phase-aligning incoming payloads. In this unit, gaps are included or deleted for pointer adjustments.

The pointer processor/gapper unit 70 also performs an overhead gapping function to avoid the writing of section, line and block overhead into a buffer unit 72, in this case, FIFO, elastic store, while enabling the writing of extracted data from the SPE payload into the buffer. The SONET clock 66 is gapped to form a block clock (not shown). The block clock has gaps evenly spread across the frame to account for TOH, POH, and remainder fields so as to reject TOH and fixed stuff bits.

Using β values associated with a block, the block clock is further gapped to produce a mapping clock (not shown) which gaps out data bits so that only data bits from the original signal may be clocked out of the pointer processor/gapper unit 70.

The degree of gapping associated with pointer processing, SPE overhead and block overhead is applied in the formation of a multibit clock 74, which is generated in the pointer processor/gapper unit 70. To recap, a multibit value is a value representing the fraction of non-gapped data bits over the maximum multibit clock value in a cycle of the local SONET clock, rounded to the nearest integer. The multibit values over multiple cycles form the multibit clock. This multibit clock 74 is driven by the SONET clock bit rate f_2.

The multibit values forming the multibit clock are then inputted into the desynchronizer PLL 76. As further detailed below, phase error is detected and corrected in the desynchronizer PLL 76, and the clock recovery of the original clock is performed. The recovered original data clock is then used in the reconstruction of the original signal for use in user specific hardware.

Data bits 64b from the pointer processor/gapper unit 70 are written into the FIFO 72 using the mapping clock. The FIFO 72 is emptied under the control of a data clock 18 outputted by a clock recovery circuit 78 in a desynchronizer framer and gapper unit 80. The data clock 18 is substantially the same bit rate f_l as the original data clock 18d, being reconstructed from the F values transmitted with each block. There is some variation in the frequency as reconstructed from the F values so the resulting frequency is not exactly the same as the original frequency of the data, where overhead was removed prior to mapping at the synchronizer.

The FIFO width may be advantageously chosen to match the stuffing word size. This method allows very large FIFO widths to be used without excessive waiting time jitter. Examples of such widths are 24 bits for an STS-3 envelope and 6144 bits for an STS-768 envelope.

The desynchronizer gapper and framer unit 70 also recreates the overhead from the original data format earlier removed at the synchronizer and gaps the recreated overhead. The desynchronizer gapper and framer unit 70 also performs framing and parity check function, and uses the data clock to synchronize the reading out of data from the FIFO. The resulting data is outputted in association with a higher rate clock than the read clock.

In association with the data clock 18d, a data multibit clock 82 is also produced by the desynchronizer gapper and framer unit 80. This data multibit clock 82 is also based on multibit values representing the fraction of non-gapped data bits over the maximum multibit clock value in a cycle of the reconstructed data clock, rounded to the nearest integer. The multibit values over multiple cycles form the multibit clock. This multibit clock is driven by the data clock rate 18d.

Referring to Figure 8, the desynchronizer PLL 76 is depicted in greater detail. The F values at the block rate are received and converted into phase at a frequency/phase converter 84. The converter is a digital summer which sums the F values and periodically integrates the values to yield 32 bit phase information. The phase information is inputted into a phase comparator 86.

The SONET multibit clock 74 is formed by gapping the local SONET clock 66 using the F values from each received block. This multibit clock is converted into phase at a frequency/phase converter 84. The converter is a digital summer which sums the multibit values and periodically integrates the values to yield 32 bit phase information. The phase information is also inputted into a phase comparator 86.

The phase comparator 86 is similar to the phase comparator at the synchronizer, being a digital subtractor. The phase comparator 86 compares the phase of the F values from the phase of the data multibit clock 82 and the SONET multibit clock 74 and outputs an error signal ϵ, representing the difference between the multibit clocks.

The output, or error signal, from the phase comparator 86 is averaged to a stabilized level. The output is inputted into a low pass filter (LPF) 90, followed by a digital to analog converter 92. In this manner, digital words are transformed into an analog voltage level.

The LPF 90 is a single order LPF and is selected so as to produce a PLL with a desired closed loop bandwidth in conjunction with the VCO gain constant selection. A digital to analog converter 92 converts the digital output from the phase comparator 86 to an analog signal and outputted to the analog VCO 94. The VCO 94 responds to input voltages by changing its output frequency. An "actual" PLL is formed with the phase comparator 86 as well as the LPF 90 and the VCO 94.

As a physical oscillator, the VCO 94 gives rise to phase noise. The amount of phase noise that is not tracked by the PLL and hence seen as jitter at the output of the PLL 90 is determined by the closed loop bandwidth of the PLL around the VCO 94. However, where the bandwidth is as wide as possible, phase noise can be tracked and will not result in jitter. Here, the PLL 94 is not used for smoothing out gaps in the write clock, as all phase transients have already been filtered before the loop and waiting time jitter has been eliminated to arbitrary accuracy. By incorporating a separate phase determination from the FIFO 72 fill, the VCO 94 can be locked in a wide

bandwith loop, so as to track substantially all out-of-phase noise and avoid delays resulting from large elastic stores traditionally required to absorb untracked wander.

Advantageously, a PLL bandwidth may be in the order of 1MHz. This allows the use of a low cost, wide range oscillator. The wide bandwidth serves to suppress the larger phase noise from such an oscillator.

The output from the VCO 94 is converted in an analog to digital converter 96 and then gapped for overhead. If overhead was not removed from the data signal at the synchronizer, then no gapping occurs.

In association with the reconstructed data clock 18d, a data multibit clock 82 is formed from the analog VCO 94 for the recovered data clock. The multibit clock is inputted into a frequency to phase converter 88. This converter 88 is also a digital summer which continuously sums the multibit values received, which is periodically integrated to yield phase information. The phase information from the analog VCO 94 for the recovered data clock 18d is inputted into the phase comparator 86 with the local SONET multibit clock to be subtracted from the phase information from the F values from the received blocks.

Imperfect phase sampling intervals due to the time differentials between the phase comparator and the phase estimator based on the reconstruction of the multibit clock values may result in jitter that can be eliminated to arbitrary accuracy by a detailed phase extrapolation. Similar to the synchronizer digital PLL, the desynchronizer PLL is provided with a detailed phase detector 98 similar to that provided on the synchronizer.

A constant reference value 100 is also inputted into the phase comparator 86 to be added to the F values, to keep phase centred in the buffer.

In this manner, almost any digital optical signal with a continuous clock of an arbitrary rate received may be mapped into a synchronous envelope, for example, a SONET STS-3nC envelope. In this architecture, waiting time jitter and wander are eliminated to arbitrary accuracy.

This architecture contributes to a transparent system that is scaleable to high data rates, for example, 40-gigabit rates. At the synchronizer, this architecture enables the expression of a data signal frequency with arbitrary accuracy, being referenced to both the system's local clock and the data signal's own clock, and enables the use of stuff words that scales linearly with the increased bit rate. The fill of the buffer does not drive the word stuff determination. Rather, the phase value is separately determined and separately smoothed, corrected and filtered in the synchronizer digital PLL. Similarly, at the desynchronizer, the architecture enables phase values that drive the output PLL to be separately determined from the buffer fill, the smoothing and filtering of noise and phase transients having already been completed prior to input into the VCO. As such, there is no phase corruption by write and read clock gaps, while buffer fill is maintained. At the desynchronizer, the PLL may be operated at wide bandwidths.

The present invention has been described with regard to preferred embodiments. However, it will be obvious to persons skilled in the art that numerous modifications, variations, and adaptations may be made to the particular embodiments of the invention described above without departing from the scope of the invention, which is defined in the claims.

What is claimed is:

1. A synchronizing element in a system for transporting a data signal of bit rate f_1 in a data stream of bit rate f_2, where the ratio of f_1 and f_2 is variable, the synchronizing element comprising:

means for determining a bit rate f_1 for a data signal;

means for determining a local bit rate f_2 of a data stream;

means for determining a relationship between the f_1 and the f_2;

means for expressing the relationship digitally using values of significant resolution; and

means for communicating the expressed relationship.

2. A synchronizing element of claim 1 wherein the means for determining a relationship between the f_1 and the f_2 determines a relationship proportional to a ratio of the f_1 and the f_2.

3. A synchronizing element of claim 1 wherein the means for determining a relationship between the f_1 and the f_2 determines a relationship where the f_1 and the f_2 are each represented using phase information.

4. A synchronizing element of claim 3 wherein the means for determining a relationship between the f_1 and the f_2 determines a relationship based on a difference between a phase of the f_1 and a phase of the local f_2.

5. A synchronizing element of claim 1 wherein the means for determining a relationship between the f_1 and f_2 determines a relationship where the f_1 is expressed using a local multibit value representative of bit rate f_1.

6. A synchronizing element of claim 5 wherein the multibit value is also representative of an average fraction of the data clock that is ungapped.

7. A synchronizing element of claim 6, wherein the multibit value is used in a multibit clock, the multibit clock for conveying a plurality of multibit values for consecutive cycles of the data clock.

8. A synchronizing element of claim 7 wherein the multibit clock is synchronous to f_1.

9. A synchronizing element of claim 1 wherein the means for determining a relationship between the f_1 and the f_2 determines a relationship where the local f_2 is expressed using a local multibit value representative of a local bit rate f_2.

10. A synchronizing element of claim 9 wherein the local multibit value is also representative of an average fraction of a local read clock that is ungapped, the local read clock for reading data bits of the data signal from the elastic store according to bit rate f_2.

11. A synchronizing element of claim 10, wherein the multibit value is used in a local multibit clock for conveying a plurality of local multibit values in consecutive cycles of a local clock synchronous to the local bit rate f_2

12. A synchronizing element of claim 1 wherein the values of significant resolution comprises at least 4 bits.

13. A synchronizing element of claim 12 wherein the values of significant resolution comprises at least 16 bits.

14. A synchronizing element of claim 3 wherein the means for determining a relationship between the f_1 and f_2 includes a means for phase comparison having a first resolution.

15. A synchronizing element of claim 14 further including a second means for phase comparison having a finer resolution than the first resolution.

16. A synchronizing element of claim 15 wherein the second means for phase comparison is a phase detector.

17. A synchronizing element of claim 16 wherein the phase detector is an analog detector.

18. A synchronizing element of claim 17 wherein the second means for phase comparison uses the relationship between f_1 and f_2 for its output.

19. A synchronizing element of claim 1 wherein the expressed relationship is communicated with a block of data in the data stream.

20. A synchronizing element of claim 1, wherein the expressed relationship is communicated at selected time instants.

21. A synchronizing element of claim 20, wherein the selected time instants comprise at least once per day.

22. A synchronizing element of claim 20, wherein the selected time instants comprise spaced apart intervals.

23. A synchronizing element of claim 20, wherein the selected time instants are associated with a plurality of data sets within a data signal, the data sets having a defined size and rate.

24. A synchronizing element of claim 1 wherein the expressed relationship is communicated with a stuff indicator, the stuff indicator for use in rate-adapting the data signal for transport in the data stream.

25. A synchronizing element of claim 3 wherein the relationship between the f_1 and the f_2 where each are represented using phase information, said relationship represented using phase information being determined to a coarse resolution in relation to a specified time interval.

26. A synchronizing element of claim 25, further including means for representing the relationship between the f_1 and f_2 using phase information, the relationship being determined to a fine resolution in relation to the specified time interval.

27. A synchronizing element of claim 26, wherein the relationship determined to a fine resolution in relation to the specified time interval is used to determine the relationship between the f_1 and the f_2 using phase information over a subsequent specified time interval.

28. A synchronizing element in a system for transporting a data signal of bit rate f_1 in a data stream of bit rate f_2, where the ratio of f_1 and f_2 is variable, the synchronizing element comprising:

a framer and gapper unit for determining a bit rate f_1 for a data signal;

a mapping unit for determining a local bit rate f_2 of a data stream and for mapping the data signal into the data stream; and

a control loop unit for receiving the bit rate f_1 and a bit rate f_2, the control loop unit having a comparator unit for determining a relationship between the f_1 and the f_2, the control loop unit expressing the relationship as a value of significant resolution for communication with the data signal in the data stream.

29. A synchronizing element of claim 28, wherein the comparator unit determines a relationship proportional to a ratio of the f_1 and the f_2.

30. A synchronizing element of claim 28, wherein the control loop includes a frequency to phase determination unit for expressing the bit rate f_1 in terms of phase information over a specified bit interval.

31. A synchronizing element of claim 30, wherein the control loop further includes a frequency to phase determination unit for expressing the bit rate f_2 in terms of phase information over a specified bit interval.

32. A synchronizing element of claim 28, wherein the comparator unit determines a relationship using a difference between a phase of the f_1 and a phase of the f_2.

33. A synchronizing element of claim 28 wherein the control loop unit determines a relationship between the f_1 and the f_2, where the f_1 is expressed using a multibit value representative of a word size of a data clock, the data clock being derived from the data signal for use in writing data bits into a data buffer unit.

34. A synchronizing element of claim 33 wherein the multibit value is also representative of an average fraction of the data clock that is ungapped.

35. A synchronizing element of claim 34, wherein the multibit value is used in a multibit clock, the multibit clock for conveying a plurality of multibit values for consecutive cycles of the data clock.

36. A synchronizing element of claim 35 wherein the multibit clock is synchronous to f_1.

37. A synchronizing element of claim 28 wherein the control loop unit determines a relationship between the f_1 and the f_2 where the f_2 is expressed using a local multibit value representative of the bit rate f_2.

38. A synchronizing element of claim 29 wherein the local multibit value is also representative of an average fraction of a local read clock that is ungapped, the local read clock for reading data bits of the data signal from the elastic store according to bit rate f_2.

39. A synchronizing element of claim 30, wherein the multibit value is used in a local multibit clock for conveying a plurality of local multibit values in consecutive cycles of a local clock synchronous to the local bit rate f_2

40. A synchronizing element of claim 30 wherein the frequency to phase determination unit determines to a coarse resolution a phase information for the bit rate f_1 over the specified bit interval.

41. A synchronizing element of claim 31 wherein the frequency to phase determination unit determines phase information to a coarse resolution for the bit rate f_2 over the specified bit interval.

42. A synchronizing element of claim 28, wherein the control loop unit further includes a detailed phase comparator unit for determining phase information for each of f_1 and f_2 to a fine resolution over the specified time interval, the respective phase information for use in the comparator unit in determining a relationship between the f_1 and the f_2.

43. A synchronizing element of claim 28 wherein the relationship expressed by the control loop is received by the mapping unit for communication with the data signal in the data stream.

44. A synchronizing element of claim 43 wherein the mapping unit provides the relationship to a block of data for communication with the data signal in the data stream.

45. A synchronizing element of claim 28 , wherein the relationship expressed by the control loop is received by the mapping unit for communication at selected time instants with the data signal in the data stream.

46. A synchronizing element of claim 45, wherein the selected time instants comprise at least once per day.

47. A synchronizing element of claim 46, wherein the selected time instants comprise spaced apart intervals.

48. A synchronizing element of claim 47, wherein the selected time instants are associated with a plurality of data sets within the data signal, the data sets having a defined size and rate.

49. A synchronizing element of claim 28 wherein the relationship expressed by the control loop is received by the mapping unit for communication with a stuff indicator, the stuff indicator for use in rate-adapting the data signal for transport in the data stream.

50. A synchronizing element of claim 49, wherein the stuff indicator is derived from the relationship.

51. A method of synchronization for rate adapting a data signal of bit rate f_1 for transport in a data stream of bit-rate f_2, where the ratio of f_1 and f_2 is variable, the method comprising the steps of:

determining a bit rate f_1 for a data signal;

determining a local bit rate f_2 of a data stream;

determining a relationship between the f_1 and the f_2; and

expressing the relationship digitally using values of significant resolution.

52. A method of claim 51 wherein the step of determining a relationship between the f_1 and the f_2 includes determining a relationship proportional to a ratio of the f_1 and the f_2.

53. A method of claim 52, wherein the step of determining a relationship proportional to a ratio of the f_1 and the f_2 includes relating the bit rate f_1 to f_1 phase over a specified bit interval.

54. A method of claim 53 wherein the step of relating the bit rate f_1 to phase over a specified bit interval includes the step of determining f_1 phase to a high resolution for the bit interval.

55. A method of claim 52, wherein the step of determining a relationship proportional to a ratio of the f_1 and the f_2 includes relating the bit rate f_2 to f_2 phase over the specified bit interval.

56. A method of claim 53 wherein the step of relating the bit rate f_2 to phase over a specified bit interval includes the step of determining f_2 phase to a high resolution for the bit interval.

57. A method of claim 51, wherein the step of determining a relationship between the f_1 and the f_2 includes determining a relationship using a difference between a phase of the f_1 and a phase of the f_2.

58. A method of claim 57, wherein the step of determining a relationship between the f_1 and the f_2 includes determining the phase of each of f_1 and f_2 to a fine resolution over a specified time interval, and using the phase

phase of each of f_1 and f_2 in determining the relationship between the f_1 and the f_2.

59. A method of claim 51 wherein the step of determining a bit rate f_1 for a data signal includes expressing the f_1 using a plurality of multibit values each of which is representative of a word size of a data clock, the data clock being derived from the data signal for use in writing data bits into a data buffer unit.

60. A method of claim 59, wherein the multibit values are also representative of an average fraction of the data clock that is ungapped.

61. A method of claim 59, wherein the step of expressing the f_1 using a plurality of multibit values includes forming a multibit clock, the multibit clock for conveying a plurality of multibit values for consecutive cycles of the data clock.

62. A method of claim 51 wherein the step of determining a bit rate f_2 for a data stream includes expressing the f_2 using a plurality of local multibit values representative of the bit rate f_2.

63. A method of claim 62 wherein the plurality of multibit values are each also representative of an average fraction of a local read clock that is ungapped, the local read clock for reading data bits of the data signal from the elastic store according to bit rate f_2.

64. A method of claim 62, wherein the step of expressing the f_2 using a plurality of multibit values includes forming a multibit clock, the multibit clock for conveying a plurality of multibit values for consecutive cycles of the local clock synchronous to the bit rate f_2.

65. A method of claim 51 further comprising the step of communicating the expressed relationship with the data signal.

66. A method of claim 65 wherein the step of communicated the expressed relationship includes the step of providing the expressed relationship to a block of data for communication with the data signal in the data stream.

67. A method of claim 65 , wherein the step of communicating the expressed relationship includes communicating the expressed relationship at selected time instants with the data signal in the data stream.

68. A method of claim 67, wherein the selected time instants comprise at least once per day.

69. A method of claim 67, wherein the selected time instants comprise spaced apart intervals.

70. A method of claim 67, wherein the selected time instants are associated with a plurality of data sets within the data signal, the data sets having a defined size and rate.

71. A method of claim 65 wherein the step of communicating the expressed relationship includes communicating the express relation with a stuff indicator, the stuff indicator for use in rate-adapting the data signal for transport in the data stream.

72. A method of claim 71, wherein the stuff indicator is derived from the expressed relationship.

73. A desynchronizing element for reconstructing a data signal of bit rate f_1 from a data stream of bit-rate f_2, the data signal having been

transported in the data stream, the ratio of f_1 and f_2 being variable, the desynchronizing element having a phase transfer function that has a substantially broad bandwidth.

74. A desynchronizing of claim 73 further comprising:

means for generating an output clock frequency of bit rate f_1 for use in data signal recovery from the data stream; and

means for controlling the output clock frequency.

75. A desynchronizing element of claim 74 wherein the means for controlling the clock frequency suppresses a phase noise from the means for generating an output clock frequency.

76. A desynchronizing element of claim 75 wherein the means for controlling the output clock frequency significantly mitigates jitter.

77. A desynchronizing element of claim 74, wherein the means for generating an output clock frequency is a voltage controlled oscillator.

78. A desynchronizing element of claim 74, wherein the means for controlling the clock frequency is a phase-locked loop.

79. A desynchronizing element of claim 78, wherein the phase-locked loop has a loop control bandwidth greater than 1 kHz.

80. A desynchronizing element of claim 78, wherein the phase-locked loop has a loop control bandwidth greater than 10 kHz.

81. A desynchronizing element of claim 78, wherein the phase-locked loop has a loop control bandwidth greater than 100 kHz.

82. A desynchronizing element of claim 74 further comprising means for frequency filtering a clock frequency for input into the means for controlling the output clock frequency.

83. A desynchronizing element of claim 73, further comprising:

means for phase error determination and correction for outputting a clock frequency of f_1; and

a parallel data store for data output synchronous to the clock frequency f_1, the data store having a width of 24 or more bits.

84. A desynchronizing element of claim 83, wherein the parallel data store does not contribute to phase error determination.

85. A desynchronizing element of claim 83, wherein the parallel data store has a width of 48 or more bits.

86. A desynchronizing element of claim 83, wherein the parallel data store has a width of 144 or more bits.

87. A desynchronizing element of claim 83, wherein the parallel data store has a width of 768 or more bits.

88. A desynchronizing element of claim 83, wherein the data output from the parallel data store is gapped so as to produce word stuffing.

89. A desynchronizing element of claim 88, wherein the words have a size equal to an integer multiple of the width of the parallel buffer.

90. A desynchronizing element of claim 84 wherein the means for phase error determination and correction significantly mitigates waiting time jitter.

91. A desynchronizing element of claim 74 wherein the means for controlling the output clock frequency is adapted to receive a frequency filtered input clock frequency.

92. A desynchronizing element of claim 91 wherein the input clock frequency is derived from a digital expression of a relationship between the bit rate f_1 and a bit rate f_2 using values of significant resolution, the digital expression being determined at a synchronizer element in a system for transporting a data signal of bit rate f_1 in a data stream of bit rate f_2, wherein the ratio of f_1 and f_2 is variable.

93. A desynchronizing element of claim 84, wherein the derivation of the clock frequency f_1 is decoupled from a stuff indication.

94. A desynchronizing of claim 73 further comprising:

an oscillator means for generating an output clock frequency of bit rate f_1 for use in data signal recovery from the data stream; and

a phase control means for controlling the output clock frequency.

95. A desynchronizing element of claim 94 wherein the oscillator means suppresses a phase noise from phase control means.

96. A desynchronizing element of claim 95 wherein the oscillator means significantly mitigates jitter.

97. A desynchronizing element of claim 94, wherein the oscillator means is a voltage controlled oscillator.

98. A desynchronizing element of claim 94, wherein the phase control means the clock frequency is a phase-locked loop.

99. A desynchronizing element of claim 98, wherein the phase-locked loop has a loop control bandwidth greater than 1 kHz.

100. A desynchronizing element of claim 98, wherein the phase-locked loop has a loop control bandwidth greater than 10 kHz.

101. A desynchronizing element of claim 98, wherein the phase-locked loop has a loop control bandwidth greater than 100 kHz.

102. A desynchronizing element of claim 73, further comprising:

phase comparator means for outputting a clock frequency of f_1; and

a parallel data store for data output synchronous to the clock frequency f_1, the data store having a width of 24 or more bits.

103. A desynchronizing element of claim 102, wherein the parallel data store does not contribute to phase error determination.

104. A desynchronizing element of claim 102, wherein the parallel data store has a width of 48 or more bits.

105. A desynchronizing element of claim 102, wherein the parallel data store has a width of 144 or more bits.

106. A desynchronizing element of claim 102, wherein the parallel data store has a width of 768 or more bits.

107. A desynchronizing element of claim 102, wherein the data output from the parallel data store is gapped so as to produce word stuffing.

108. A desynchronizing element of claim 107, wherein the words have a size equal to an integer multiple of the width of the parallel buffer.

109. A desynchronizing element of claim 102 wherein the phase comparator means significantly mitigates waiting time jitter.

110. A desynchronizing element of claim 102 wherein the phase control means is adapted to receive a frequency filtered input clock frequency.

111. A desynchronizing element of claim 110 wherein the input clock frequency is derived from a digital expression of a relationship between the bit rate f_1 and a bit rate f_2 using values of significant resolution, the digital expression being determined at a synchronizer element in a system for transporting a data signal of bit rate f_1 in a data stream of bit rate f_2, wherein the ratio of f_1 and f_2 is variable.

112. A desynchronizing element of claim 102, wherein the derivation of the clock frequency f_1 is decoupled from a stuff indication.

113. A desynchronizing element of claim 102, wherein the derivation of the clock frequency f_1 is decoupled from a level of fill of the parallel data store.

114. A method of desynchronization for reconstructing a data signal of bit rate f_1 from a data stream of bit-rate f_2, the data signal having been transported in the data stream, the ratio of f_1 and f_2 being variable, providing a phase transfer function that has a substantially broad bandwidth, the method comprising the steps of.

generating an output clock frequency of bit rate f_1 for use in data signal recovery from the data stream; and

controlling the output clock frequency.

115. A method of desynchronization of claim 114 wherein the step of controlling the clock frequency suppresses a phase noise from the generation of the output clock frequency.

116. A method of desynchronizing element of claim 115 wherein the step of controlling the output clock frequency significantly mitigates jitter.

117. A method of desynchronization of claim 114, further comprising the steps of:

determining a phase error;

correcting the phase error;

outputting a clock frequency of f_1; and

ouptting data from a parallel data store at a frequency synchronous to the clock frequency f_1, the data store having a width of 24 or more bits.

118. A method of desynchronization of claim 117, wherein the parallel data store does not contribute to the step of determining the phase error.

119. A method of desynchronation of claim 117, wherein the step of outputting data from the parallel data store includes the step of gapping the data so as to produce word stuffing.

120. A synchronizing element of claim 1 wherein the means for determining a bit rate f_1 is decoupled from the level of the fill of data store.

121. A synchronizing element of claim 1 adapted for use with a parallel buffer wherein data output is gapped so as to produce word stuffing.

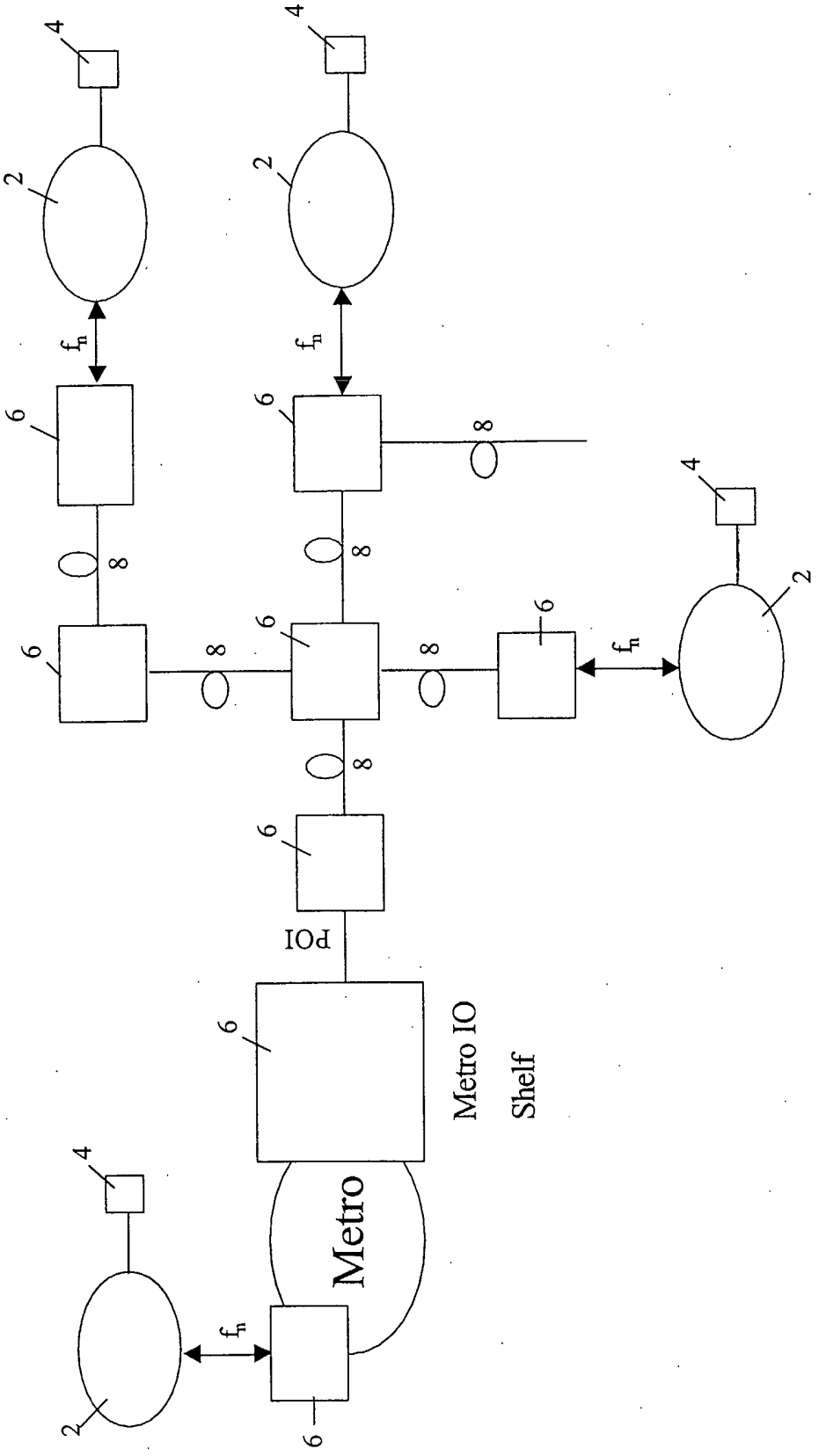


Figure 1

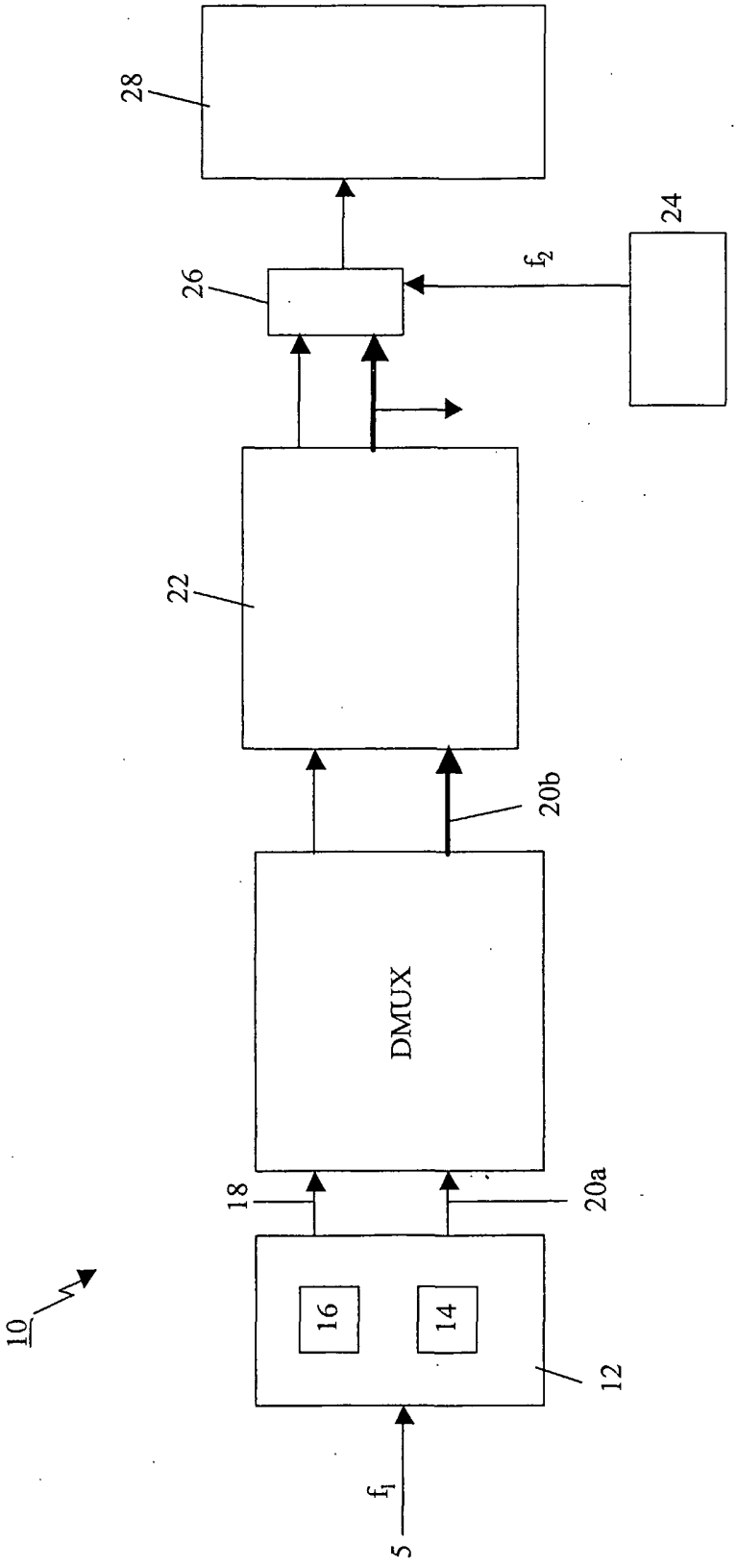


Figure 2

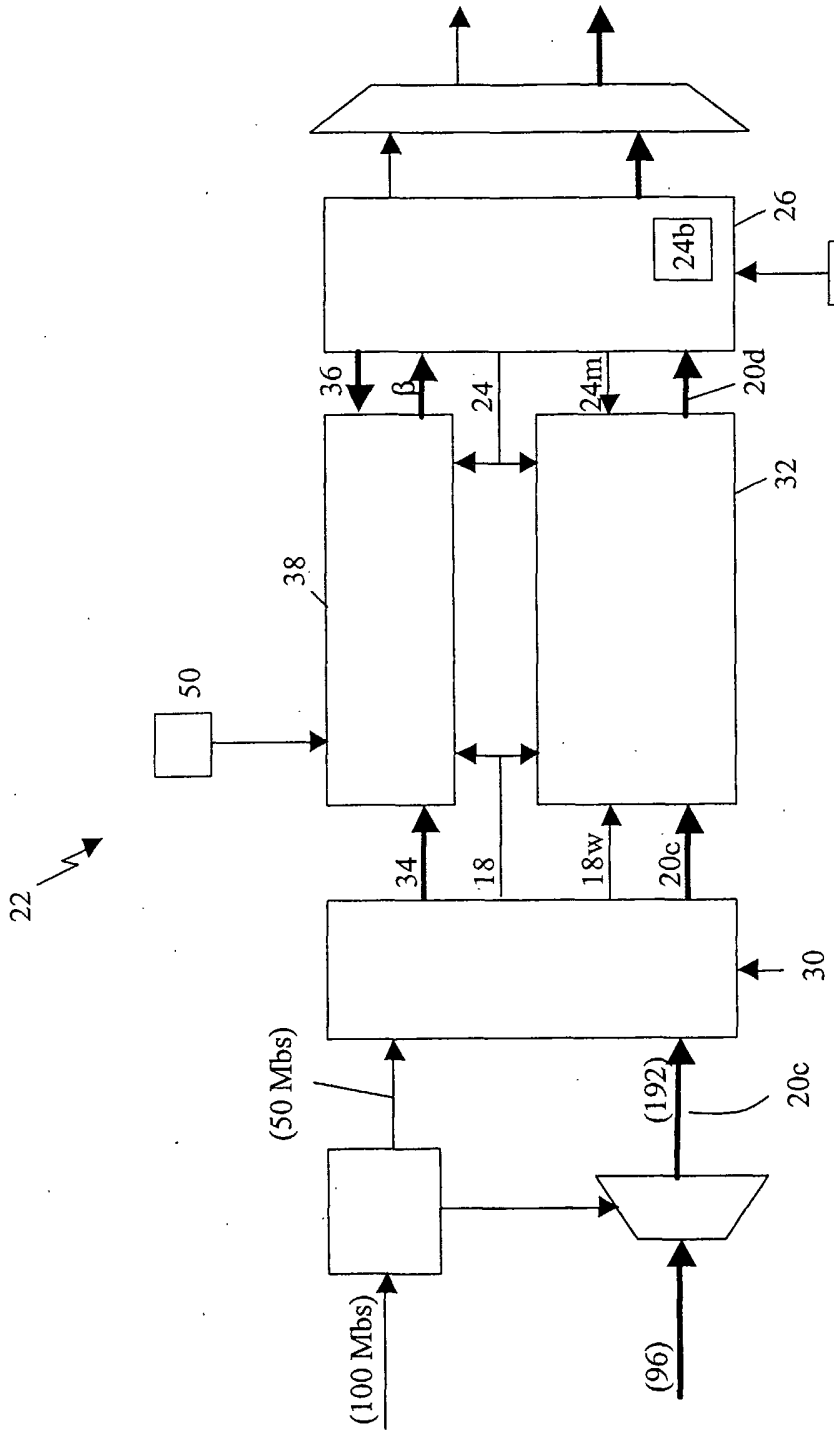

22
50
38
36
β
24
24m
20d
32
26
24b
24
34
18
18w
20c
30
(50 Mbs)
(192)
20c
(100 Mbs)
(96)

Figure 3

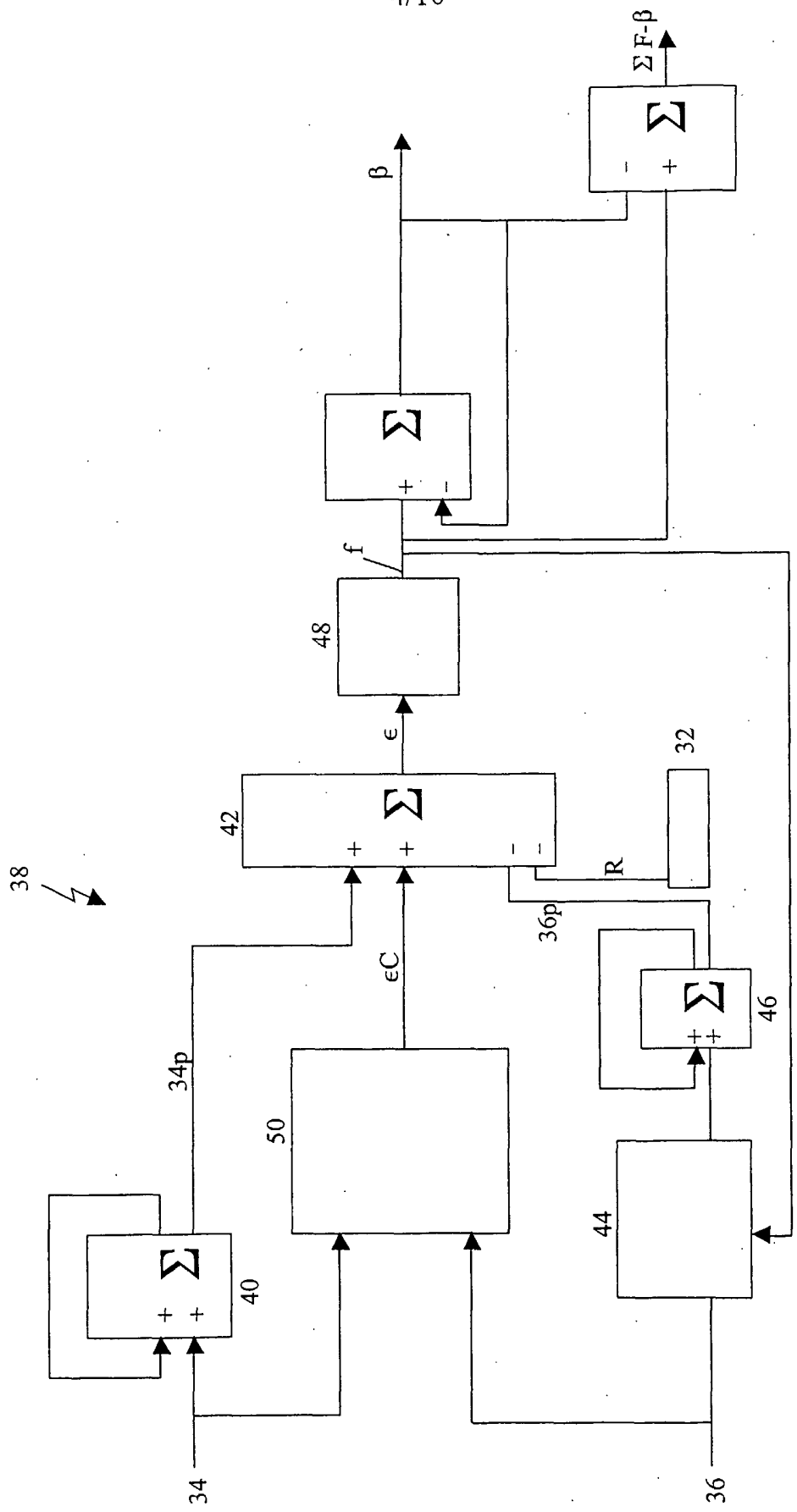


Figure 4

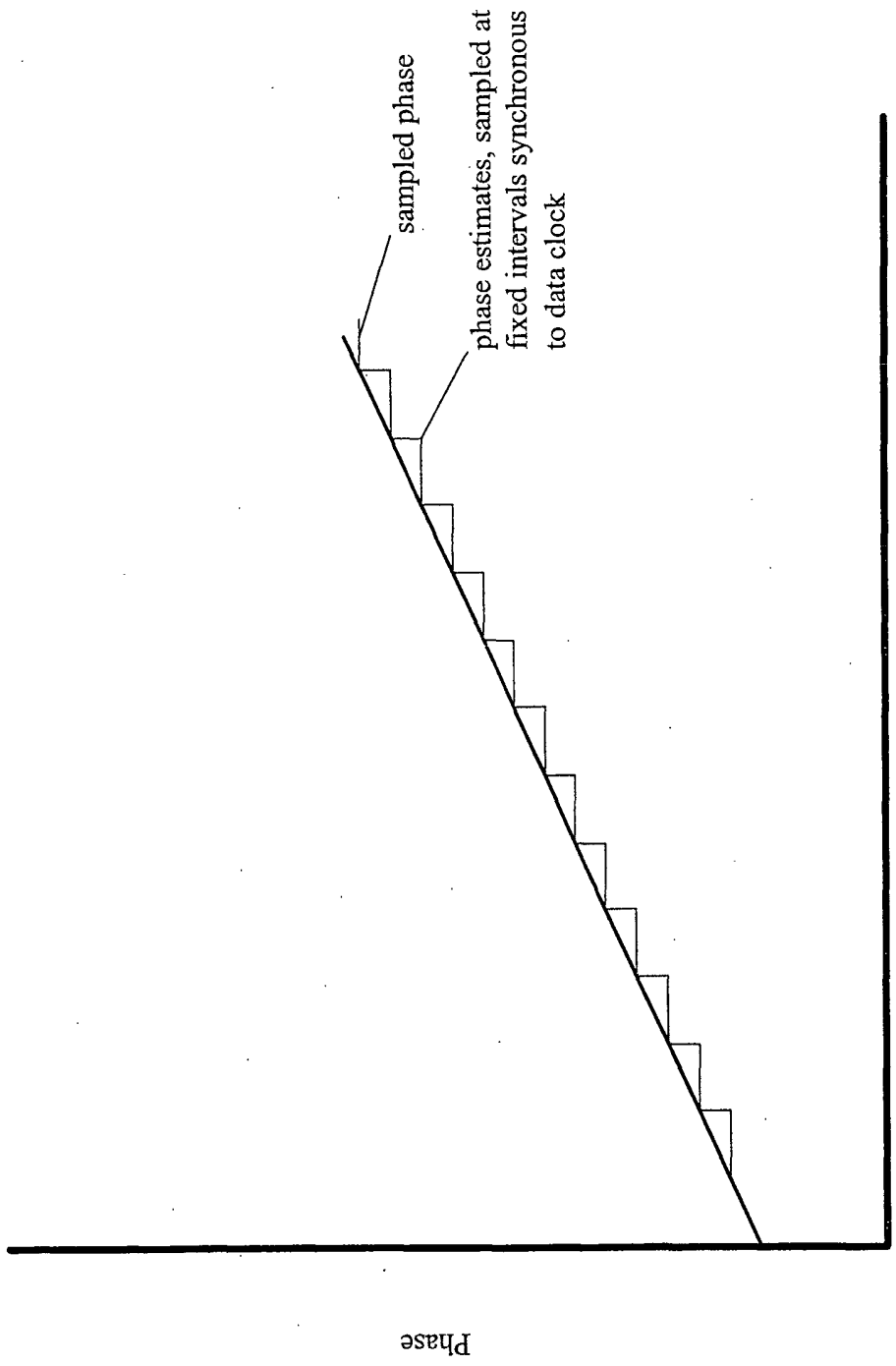

sampled phase
phase estimates, sampled at fixed intervals synchronous to data clock
Phase
Time

Figure 5a

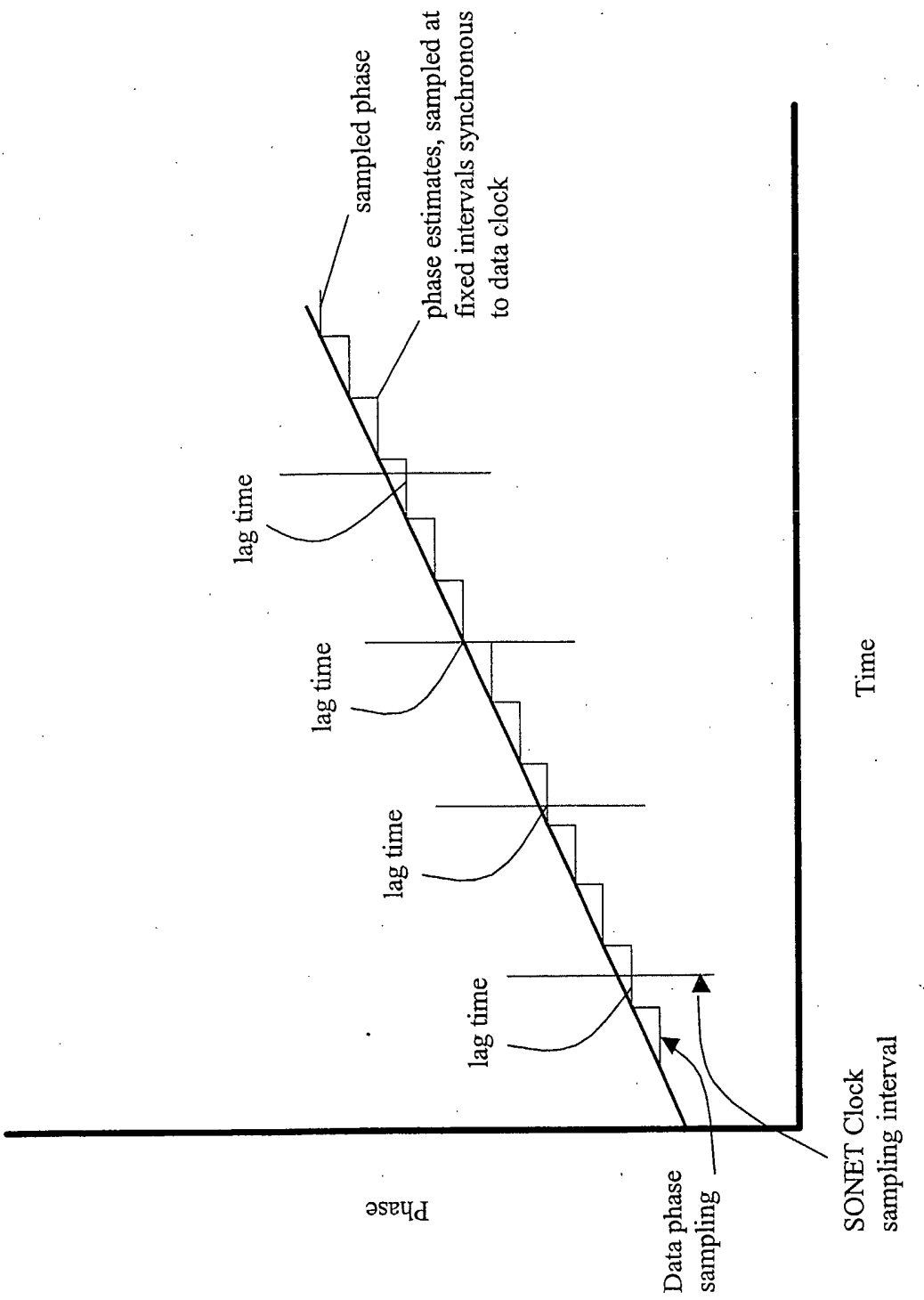

sampled phase
phase estimates, sampled at fixed intervals synchronous to data clock
lag time
lag time
lag time
lag time
Phase
Time
Data phase sampling
SONET Clock sampling interval

Figure 5b

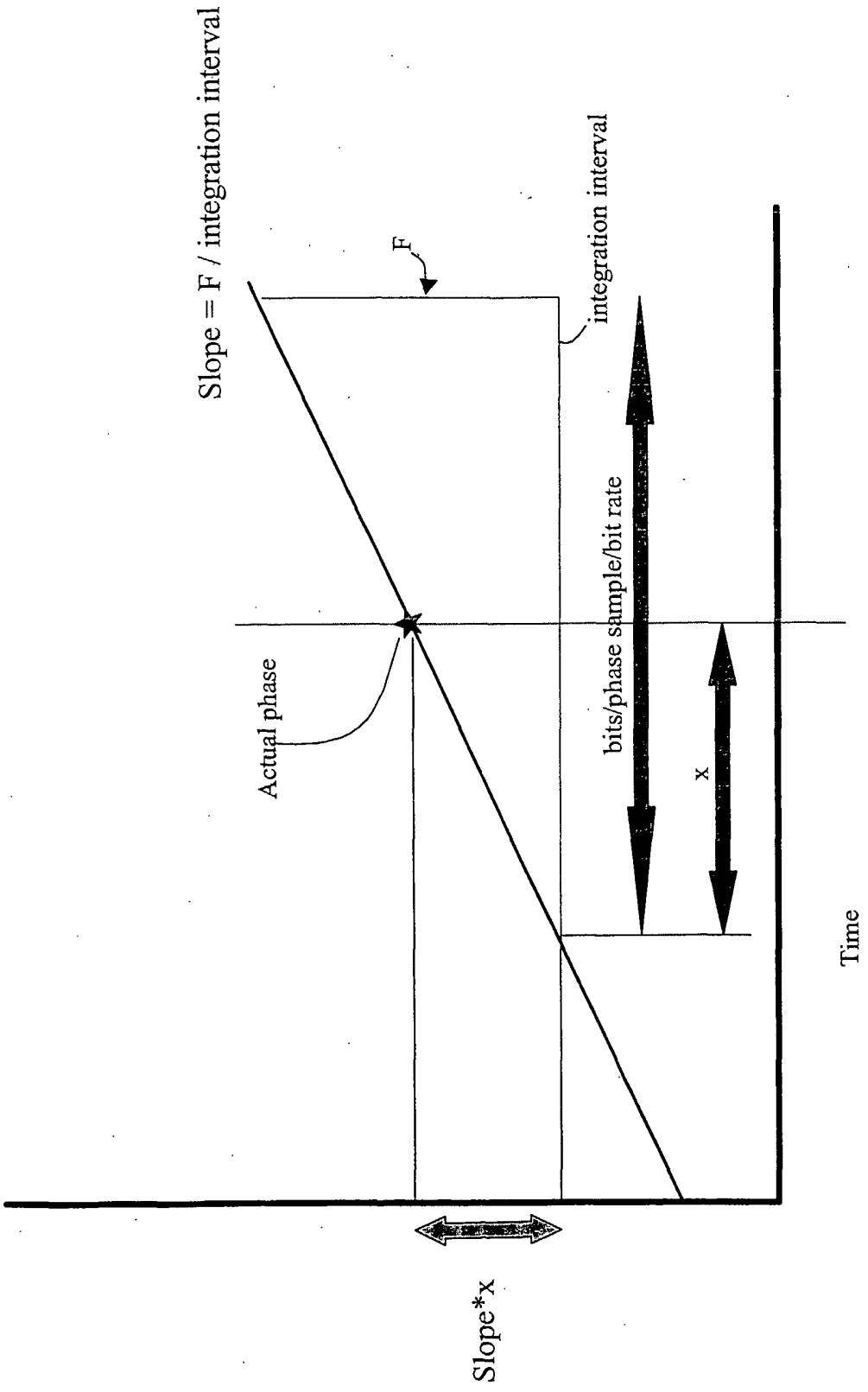

Slope = F / integration interval
F
integration interval
bits/phase sample/bit rate
x
Actual phase
Time
Slope*x

Figure 5c

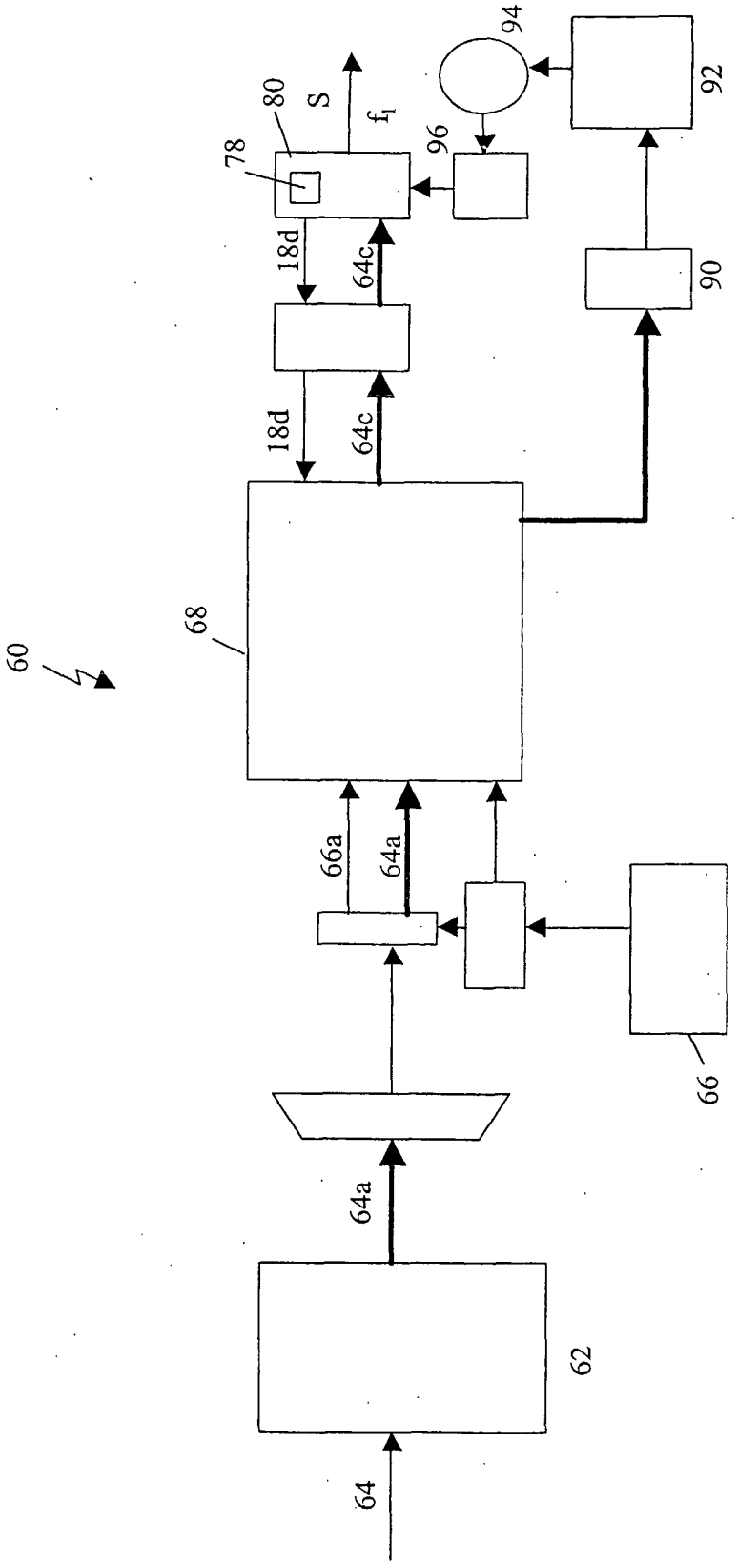


Figure 6

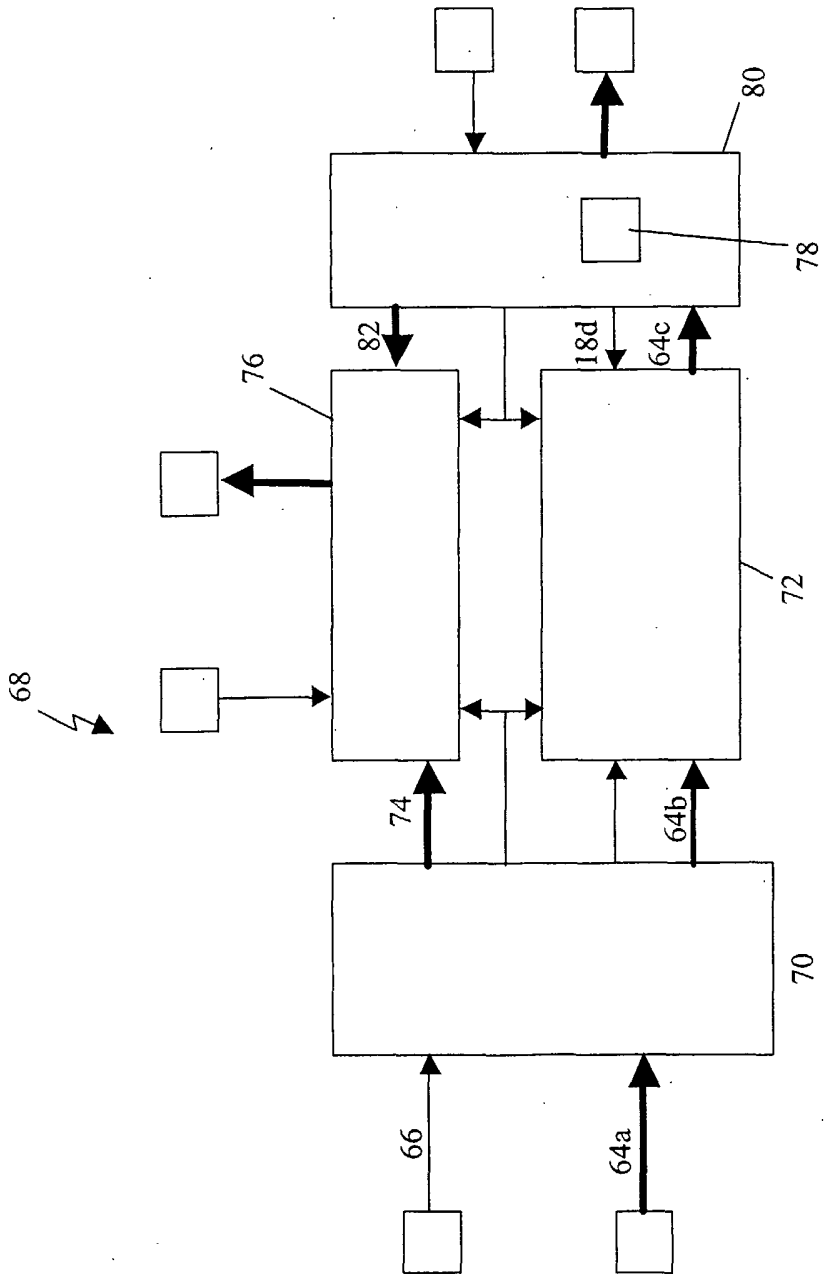


Figure 7

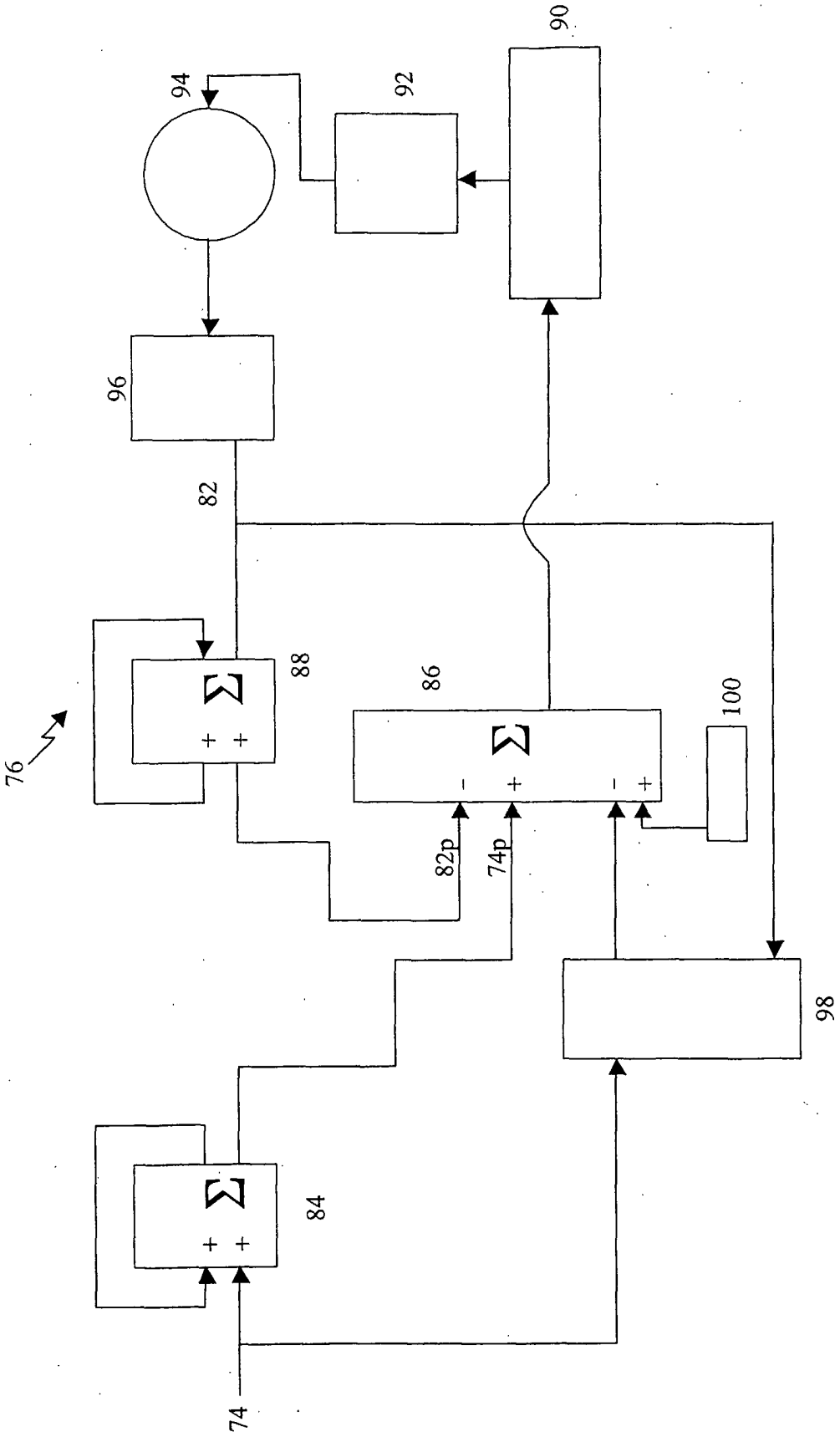


Figure 8