Office de la Propriété
Intellectuelle
du Canada

Un organisme
d'Industrie Canada

Canadian
Intellectual Property
Office

An agency of
Industry Canada

CA 2307044 A1 2001/10/28

(21) 2 307 044

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) **Date de dépôt/Filing Date**: 2000/04/28

(41) **Mise à la disp. pub./Open to Public Insp.**: 2001/10/28

(51) **Cl.Int.^{7}/Int.Cl.^{7}** H04L 29/02, H04L 7/00

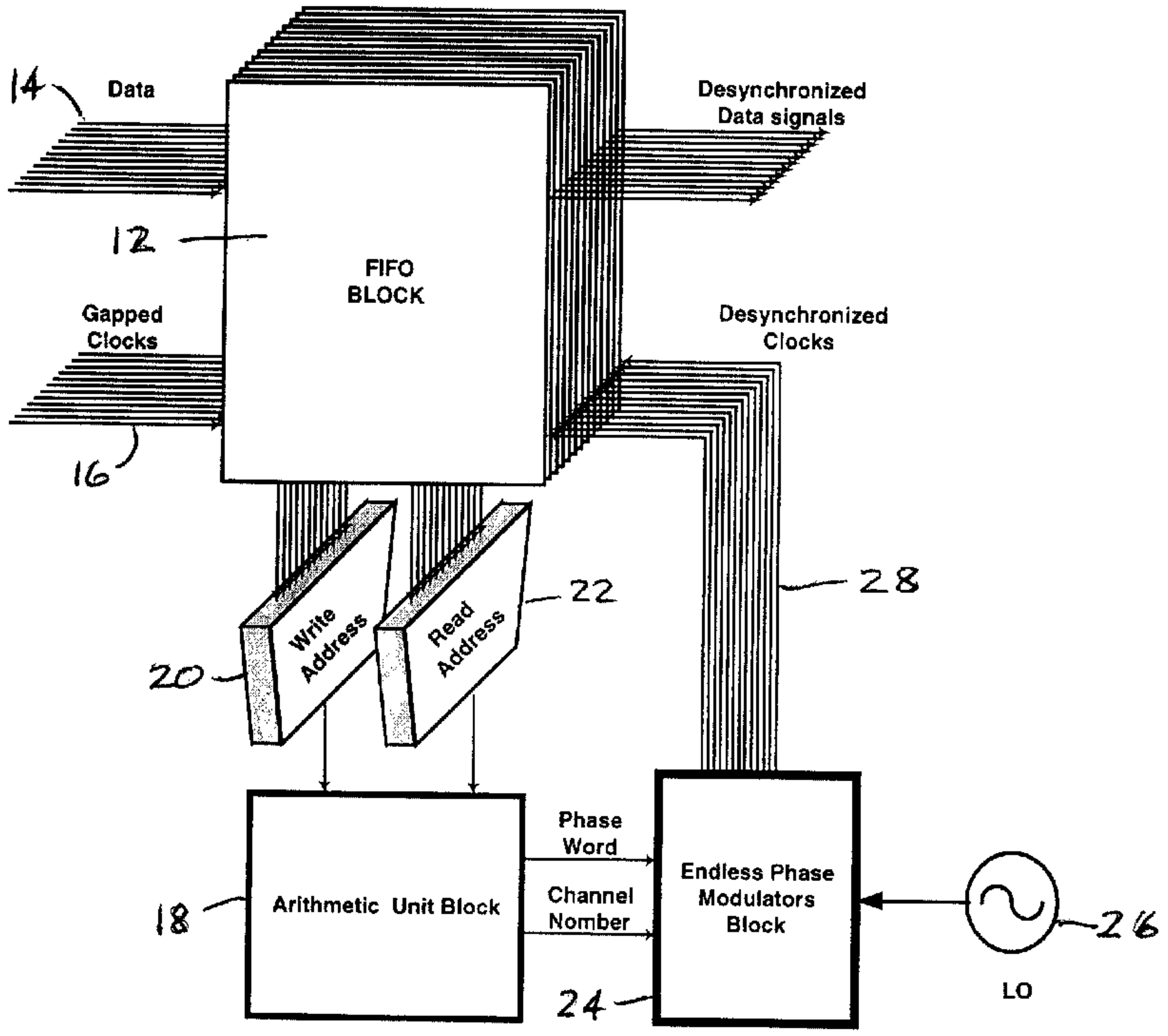
(71) **Demandeur/Applicant**:
PMC-SIERRA INC., CA

(72) **Inventeur/Inventor**:
ACIMOVIC, PREDRAG SAVA, CA

(74) **Agent**: VERMETTE & CO.

(54) **Titre** : DESYNCHRONISEUR SONET/SDH A CANAUX MULTIPLES
(54) **Title**: MULTI-CHANNEL SONET/SDH DESYNCHRONIZER

Block Diagram of Preferred Multi Channel Desynchronizer



(57) **Abrégé/Abstract**:

A desynchronizer for desynchronizing one or multiple channels of SONET/SDH data signals, which includes a first in first out (FIFO) buffer having an input coupled to said data signals and an output for outputting asynchronous data obtained from one or more of said SONET/SDH data channels. An arithmetic unit coupled to the FIFO performs all operations required for single or multi-channel desynchronization. An endless phase modulator is coupled to the arithmetic unit and to the FIFO and is operative, in response to input from the arithmetic unit, to produce a single output desynchronized clock or multiple output desynchronized clocks.

Canada *http://opic.gc.ca* · Ottawa-Hull K1A 0C9 · *http://cipo.gc.ca* OPIC CIPO
OPIC · CIPO 191

ABSTRACT

A desynchronizer for desynchronizing one or multiple channels of SONET/SDH data signals, which includes a first in first out (FIFO) buffer having an input coupled to said data signals and an output for outputting asynchronous data obtained from one or more of said SONET/SDH data channels. An arithmetic unit coupled to the FIFO performs all operations required for single or multi-channel desynchronization. An endless phase modulator is coupled to the arithmetic unit and to the FIFO and is operative, in response to input from the arithmetic unit, to produce a single output desynchronized clock or multiple output desynchronized clocks.

MULTI-CHANNEL SONET/SDH DESYNCHRONIZER

FIELD

The present invention relates to multi-channel SONET/SDH desynchronizer for reading and transmitting asynchronous data received from several synchronous channels.

BACKGROUND

Desynchronizers typically include a first in first out buffer (FIFO), a phase detector to measure the FIFO buffer depth, a passive or active analog loop filter to filter the phase detector output, and a voltage controlled oscillator (VCO) to generate an output clock to control data transmission from the FIFO buffer. These elements make up a phase locked loop.

There are some important differences in the way a single and multiple channel desynchronizers are designed. The term "multiple channel" means that more than one data signal is being dropped or extracted from the SONET/SDH signal. For example, in the case of the OC-1 signal one can drop 28 DS1 signals, which might need to be desynchronized. In the case of the OC-3 signal one can drop 3 DS3 signals or 84 DS1 signals. In the case of the OC-12 one can drop up to 12 DS3 signals or 336 DS1 signals. With the advance of data communications and the requirement for higher bandwidths, there are more and more instances of multiple channels being dropped from the same SONET/SDH signal.

A multiple channel desynchronizer should not use a voltage controlled oscillator (VCO) to generate the desynchronized output clock. Most of the known desynchronizers use VCXOs or VCOs in cases where desynchronizers produce a control voltage that controls the frequency of the VCO. Multiple channels, for example 12 channels of DS3 can be dropped from OC12, can have output frequencies quite close to each other and, as a consequence, mutual coupling can cause excessive jitter. The reason for the excessive jitter is caused by the large index of modulation of the frequency modulation device. The index of modulation of the frequency modulation device is inversely proportional to the frequency of modulation. Thus, small frequency offsets between desynchronized clocks, if there is even a very small coupling between the desyncheronization circuits, through PLL action would produce very low modulation frequencies on the control voltages which will modulate excessively the VCO outputs.

The multiple channel desynchronization circuit should use a phase modulation method when generating desynchronized clocks to avoid the excessive jitter caused by mutual coupling of output clocks. The phase modulation device can be implemented as an endless phase modulator, a numerically controlled oscillator (NCO) or a single side-band modulator (SSB). Several patents have already suggested use of the NCO devices for the desynchronization. A large number of the NCO devices on the same chip, especially if generating high frequencies, would result in large power consumption. A better way, as far as power consumption is concerned, where a low frequency is synthesized using an NCO, is to up-convert this low frequency using a mixer and a high frequency local oscillator (LO). The output signal from the mixer is filtered using the LC filter. In an ASIC implementation of the desynchronizer a use of the LC filter is not an option. More suited for the ASIC implementation is a version of up-conversion implementation, which uses the SSB modulator. By the careful matching of phases and amplitudes in two branches containing double balanced mixers, the SSB modulator suppresses unwanted products of mixing, the LO leakage and the undesired side-band. Normally, the SSB modulators can suppress the unwanted products by more than 20 dBs, resulting in jitter of less than 0.05 UIpp. The SSB modulation results in intrinsic jitter well below the level allowed by **GR-253-CORE**. Some care has to be taken to avoid excessive mutual coupling of output clocks because they still add to jitter, although there is no excessive problem with low frequency offsets as in the case of the frequency modulation device.

The preferred implementation would use an endless phase modulator, because it is a digital approach and it is robust, due to the fact that coupling between different output signals has the least effect.

The second conclusion from investigating a multiple channel desynchronizer is that a separate desynchronizer for each channel will result in an extremely large circuit size. That is specifically true if each of the necessary steps to be performed are implemented without the re-use of circuitry. None of the prior state of art re-uses the circuitry in the claims. All steps necessary to perform the desynchronization are performed separately and, therefore, very inefficiently. It is an object of the invention to merge all the blocks in one single block that reuses the circuitry to produce the desired result.

SUMMARY OF THE INVENTION

According to the invention there is provided a desynchronizer for desynchronizing one or multiple channels of SONET/SDH data signals, which includes a first in first out (FIFO) buffer having an input coupled to said data signals and an output for outputting asynchronous data obtained from one or more of said SONET/SDH data channels. An arithmetic unit coupled to the FIFO performs all operations required for single or multi-channel desynchronization. An endless phase modulator is coupled to the arithmetic unit and to the FIFO and is operative, in response to input from the arithmetic unit, to produce a single output desynchronized clock or multiple output desynchronized clocks.

Several steps are necessary in the desynchronization process. Depending on which algorithm is being used, some steps are optional. A desynchronizer is a form of phase lock loop (PLL). A FIFO is used to store data clocked into the FIFO with a gapped clock, which is a result of extracting a digital signal from a SONET/SDH frame. A narrow-band PLL is used to average this gapped clock with a long time-constant and clock the data out of the FIFO with a non-jittery clock, whose frequency is equal to the average frequency of the gapped clock. Firstly, there is a need to form a difference between the FIFO write and read addresses. Forming a difference requires a subtract circuitry. The width of the subtract circuitry depends on the FIFO size which, on the other hand, depends primarily on the data rate of the channel being desynchronized. This is equivalent to building a phase detector for the PLL operation. This subtract action forms the error signal for the PLL.

The next step is to filter the FIFO write and read address difference to average the in-flux of data into the FIFO and make it even with the out-flux of the data from the FIFO. The previous state of the art uses separate circuitry for filtering whereas the present desynchronization circuit re-uses the same add-subtract circuit to detect an error signal for the PLL and also to perform the filtering. A filter with a perfect integrator and a zero in a transfer function centers the FIFO fill level for any steady state type of stimulus. This type of loop filter has a superior performance as compared to the filter without the perfect integrator. Most of the state of the art desynchronizers use the latter kind of loop filter. Next, most desynchronizers uses separate circuitry to process the pointer adjustment related bits in the FIFO, in many cases building a separate very often adaptive filter for this purpose, whereas the present circuit reuses the same add-subtract circuit to achieve the same effect. In processing the pointer adjustment related bits in the FIFO, the present desynchronizer goes a step further than known desynchronizers because the present one uses a digitally controlled endless phase modulator to perform a semi-open loop modulation of the endless phase modulator. Most known desynchronizers use bit leaking in which they hide the pointer adjustment related bits that are in the FIFO from the PLL. Using some algorithm or filtering, these bits are re-introduced to the PLL. The embodiment of the multi channel desynchronizer disclosed can easily be changed to perform the same function, but superior performance can be achieved if one uses the semi-open loop modulation of the endless phase modulator. In this case, we eliminate the undesirable response of the PLL when an error signal is introduced, and we completely hide the pointer adjustment related bits in the FIFO from the PLL. The state of the art desynchronizers that do not use a digitally generated clock can not completely hide the pointer adjustment related bits in the FIFO from the PLL; they only do it temporarily for the incoming bits, but they need to re-introduce them to the PLL because they do not have enough precision to synthesize the desired clock. They must depend on the PLL to adjust for any inaccuracy in synthesis. The state of art desynchronizers, that use a digitally generated clock, can perform arbitrary phase modulation of the generated clock, but only U.S. Patent No. 5,497,405 issued to Elliot et al. uses this technique.

Unfortunately, the method used in Elliot creates unnecessary jitter because it uses only open loop. The present method needs to use, for its reference, a clock identical to the add side system clock. This is not easy to achieve, as even then it produces unnecessary jitter for each stuffing bit. On the other hand, the present embodiment makes it is easy to implement, re-using the same circuitry, the semi-open loop for handling the pointer adjustment related jitter and closed loop for handling bit mapping and network related jitter. Because digital modulation of an endless phase modulator is used, one knows exactly how many bits came to the FIFO due to the pointer adjustment and how many have leaked out by modulating the endless phase modulator in an open loop fashion, so one can hide from the PLL, at all times, the bits in the FIFO resulting from the pointer adjustment. The small inaccuracy of this process resulting from small difference between the line clock used to generate the modulation that leaks the pointer adjustment related bits from the FIFO, and the actual desynchronized clock frequency is taken care by the PLL. However, this inaccuracy has an extremely small error so it does not produce any undesirable response of the PLL.

The present invention includes a novel embodiment of a desynchronizer for single or the multiple data channels dropped from the SONET/SDH signal. It uses compact circuitry capable of processing a number of channels that requires only addition or subtraction circuit, multiplex circuits and RAM. This circuitry is re-used to perform necessary operations for a single channel desynchronizer and as well for a multiple channel desynchronization.

The circuit consists of a FIFO block or several FIFO blocks for individual data channels. All processing is done in one arithmetic unit block and an output desynchronized clock or clocks, in the case of multi-channel applications, are produced in an endless phase modulator/modulators. Alternatively, one can substitute for the endless phase modulators SSBs (single sideband modulators) or, in case of low frequency data channels (DS1, E1), with digitally controlled programmable modulo dividers.

The same arithmetic unit circuit can be used to execute all operations necessary for a single or multi channel desynchronization using a narrow-band second order type two PLL with adjustable loop bandwidth to avoid FIFO spill while simultaneously independently processing and filtering pointer adjustment related bits in FIFO.

The same arithmetic unit circuit can be used to execute all operations necessary for a single or multi channel desynchronization using a narrow-band second order type dual PLL with adjustable loop bandwidth to avoid FIFO spill while simultaneously independently processing adjustment related bits in the FIFO using an open loop method of modulating the endless phase modulator.

The proposed Endless Modulator Block is a novel way to generate multiple clocks at different frequencies using one delay line and one calibration circuit. Novel use of a look-up list (LUT), built using a RAM, that converts the instantaneous phase value to the select signal for each of the multiplexer circuits that generate different clocks, results in a loop bandwidth independent of the number of taps whose delay is equivalent to one clock pulse. Use of a dual port RAM and calibration circuit reduce the size of the LUT. This method results in a PLL loop bandwidth independent of process variation and temperature. Use of the endless phase modulator significantly reduces power consumption and gate count compared to a numerically controlled oscillator (NCO) method.

BRIEF DESCRIPTION OF THE DRAWINGS

Further features and advantages will be apparent from the following detailed description, given by way of example, of a preferred embodiment taken in conjunction with the accompanying drawings, wherein:

Fig. 1 is a multi-channel desynchronizer;

Fig. 2 is a schematic diagram of an arithmetic unit block diagram;

Fig. 3 is a schematic diagram of a memory map of arithmetic unit RAM blocks;

Fig. 4-11 are schematic diagrams of steps 1-8, respectively, of the arithmetic unit algorithm;

Fig. 12 is an endless phase modulator common control block;

Fig. 13 is an endless phase modulator delay line;

Fig. 14 is a delay line calibration circuit; and

Fig. 15 is an alternative embodiment of the desynchronizer using a single side-band modulator (SSB).

DETAILED DESCRIPTION WITH REFERENCE TO THE DRAWINGS

Referring to Figure **1**, the desynchronizer includes a FIFO block **12** which receives synchronized data from a set of SONET/SDH data lines **14** and gapped clock signals on gapped clock lines **16**. An arithmetic unit block **18** is coupled to the FIFO block **12** by both a write address block **20** and a read address block **22**. An endless phase modulator block **24** couples to the arithmetic unit block **18** and to a crystal oscillator **26**. The crystal oscillator **26** oscillates at a nominal data rate of the digital signal which, for example, for desynchronizing DS3 is a rate of 44.736 MHz. The output of this crystal oscillator **26** is used in the Endless Phase Modulator Block **24** to generate the desynchronized clocks on desynchronized clock lines **28**. A delay line with taps and a multiplex circuit that selects a specific tap generates different output clocks (see Fig. **13**). Successive selecting of a higher and higher tap number of the delay line creates a lower frequency clock than the crystal oscillator frequency. Successive selecting of lower and lower tap numbers of the delay line creates a clock of higher frequency than the crystal oscillator frequency.

The Arithmetic Unit (AU) **18** is common for all the channels and it uses a simple structure that performs a number of different functions. The Block Diagram of the AU is shown in Figure **2.**

The Arithmetic Unit **18** consists of one wide add-subtract circuit **34**, a register **36** that latches the product of adding or subtracting and preferably two RAM blocks **38** and **40** that hold information used in calculating a phase increment for each channel. The memory map of these two RAM blocks in shown in Figure **3**. It is possible to use one RAM block, but the speed of operation is much faster with two RAM blocks. The first RAM block **38** contains the effective FIFO Address difference from which the bits related to the pointer adjustment in the FIFO have been subtracted. This RAM block can also contain constant values used in processing the number of bits in the FIFO **12** due to the pointer adjustments. The second RAM block **40** contains the values of the accumulated effective FIFO address difference, as well as the current number of bits in FIFO **12** that are related to the pointer adjustments. These pointer adjustment related bits in FIFO **12** are slowly disposed of by adding an extra amount of phase increment to calculated phase increments resulting from PLL calculations. This extra phase increment, that leaks pointer adjustment bits from FIFO, is independent of the calculated phase increment for PLL operation and it is added to the PLL phase increment after the PLL calculation. The pointer adjustment bits in the FIFO **12** are handled in such a matter to deny the PLL the knowledge of their existence.

The Arithmetic Unit **18** contains three multiplex circuits. Two multiplex circuits ,MUX#1 **42** and MUX#2 **44**, are used to provide the add-subtract circuit **34** with the desired input, either the FIFO address for each of the channels or the RAM output. The third multiplex circuit, MUX#3 **46**, scales the output of the operations, therefore it functions as a fast multiply/divide by 2^m circuit, where m is an integer. An adjustment of the PLL's Open Loop Gain and the Loop Filter transfer function zero positioning can be set using multiplex circuit MUX#3 **46.**

The control circuitry **46** conducts the operation of the Arithmetic Unit **18**. It is designed as a number of counters (not shown) that step the desynchronizer though the desired phases. A simple change of the counters can be used to change the algorithm used in the desynchronization.

The Arithmetic Unit **18** runs synchronously with a 51.84MHz clock recovered from the lines **16**. The SONET/SDH frame row rate is 72 kHz. The FIFO write **20** and read **22** addresses of one of the channels are sampled at the 72 kHz rate and they are processed at this rate. The sampling of the FIFO addresses can be done at slower rate (example 8 kHz), but sampling of the FIFO addresses at the highest possible rate at which reading of FIFO write address **20** results in small mapping jitter, will minimize the aliasing of jitter introduced over the optical line from the higher frequencies into the loop bandwidth.

The operations that are required to process one channel are relatively simple and can be sequenced one after other using the same basic circuitry. Operations are:

- Subtract read FIFO address from the write FIFO address and store it for further use in RAM #1 **38** shown in Figure **4**.

- If a pointer adjustment is detected than add or subtract 8 to the number representing the bits in FIFO **18** related to the pointer adjustment, stored in RAM #2 **40** for the specific channel being processed, otherwise, if no pointer adjustment is detected, add zero. This is shown in Figure **5**.

- Subtract the current number of bits in FIFO related to the pointer adjustment, content of RAM #2 **40**, from current FIFO write and read address difference, content of RAM #1 **38**, creating an effective FIFO address difference, overwriting the current FIFO write and read address difference address in RAM#1 **38**. This is shown in Figure **6**.

- Add the effective FIFO address difference, from RAM#1 **38**, to the accumulated effective FIFO address difference, stored in RAM #2 **40**, and after the addition store new value in the RAM #2 **40** for further use. Shown in Figure **7**.

- Add the current effective FIFO address difference to the scaled accumulated value of the effective FIFO address difference. The scaling is performed by taking only the most significant bits of the word representing the accumulated effective FIFO address difference. The multiplex circuit, MUX#3 **46**, is used for the scaling operation. This operation is equivalent to implementation of a loop filter with an ideal integrator and compensation. The ratio of accumulated effective FIFO address difference value versus the current effective FIFO address difference sets the position of the zero in the transfer function of the loop filter. Without any disadvantage to the performance the ratio can be chosen to be power of 2 so a simple multiplex circuit can accomplish scaling. The result can be stored in the spare address RAM #2 **40** location. This is shown in Figure **8**.

- Scale the product of the previous operation. Again, this can be done by using a portion of bits that form the value after the filtering action. This operation sets the Open Loop Gain of the PLL, and therefore the bandwidth of the PLL. The used topology allows for some useful features. Because the actual FIFO address difference is calculated in the first step, the Open Loop Gain can be modified to prevent the FIFO spill. An adaptive gain with hysteresis, is possible because the current gain value can be stored in small latch in the Control Circuit **48** and we can set the break-points for gain changes depending on address difference as well as current value of the gain. If the FIFO address difference is approaching its limits than the gain can be increased to prevent a slip. This is shown in Figure **9**. The result of this process is a PLL phase increment value.

- The next step implements the semi-open loop handling of the bits in FIFO **12** due to pointer adjustments. A small number is added/subtracted to the PLL phase increment value. This small number is stored in RAM#1 **38** when the RAM is initialized, and it provides an extra phase modulation of the output clock designed so it will leak the bits stored in FIFO **12** related to the pointer adjustment. The value of the external phase modulation can be changed, the best performance is if it is gradually increased because it will prevent discontinuities in phase function as well as first derivative of the phase function (frequency), therefore reducing the bandwidth of the jitter. This is shown in Figure **10**.

- The last step is to update the number of bits in FIFO **12** due to the pointer adjustment stored in RAM#2 **40** by adding/subtracting the amount of external phase increment used to extra modulate the Endless Phase Modulator **24** or this the external phase increment value is converted to UI (user interval). This is shown in Figure **11**.

Figures **4** to **11** show how steps of preferred embodiment are performed. The structure is flexible so with a minimum change in control signals different than shown algorithm can be used.

After the total phase increment is obtained for an individual channel its value is written to the Endless Phase Modulator Block **24**. The Endless Phase Modulator Common Control Block **24** is shown in Figure **12**. The RAM #3 **50** in Figure **12** is used to store the total phase increment. Inside the Endless Phase Modulator Block **24** the total phase increment value is added to the current phase contained in RAM #4 **52**. This operation could have been performed in the Arithmetic Unit **18** but it is better to perform it in the Endless Phase Modulator Block **24**, because, by doing so, intrinsic jitter is improved. The current phase value is used to control the output of the endless phase modulator **24**. The current phase value is used as an address for a Look-Up Table (LUT) **54**. The LUT **54** output yields the setting for the multiplex circuit select signal. The LUT **54** is built using a dual port RAM. The LUT values contain the select values for the multiplexer circuits **56** in the Endless Modulator Delay Line Circuit **51** (Figure **13**), and these values change with process variations, because the unit delay will change perhaps even a +/- 50 % from nominal value depending on process variation. If the current phase word is 8 bits wide we would need a LUT **54** that is

8 times number of all of possible numbers of delay taps that are equal to one clock cycle. This would require a large LUT **54** so it is better to use a RAM as an LUT **54**.

- The content of the LUT **54** is generated and changed on the fly, depending also on the temperature variations. This is achieved using a calibration circuit **55** shown in Figure **14**. The calibration circuit **55** determines a number of taps that form a delay closest to one period of the crystal oscillator clock **26**. Once this number is known, it is easy to determine the content of the LUT **54** arithmetically. For example, if there are 73 taps that constitute the delay of one clock cycle and 8 bits represent a phase word, then the content of the LUT **54** at address 73 should be an increment of 73/256. For purpose of adding the number representing the increment 73/256, we use the adder circuit **53** in the Endless Phase Modulator Common Control Block **24**. For example, if a circuit is desynchronizing 12 DS3 channels dropped from the OC-3 SONET signal, the adder **53** can perform adding of the phase for each of the channels in round robin fashion and, at the end of the cycle, it can update one address of the LUT **54**. After 256 round robin cycles, the whole LUT **54** will be updated. The time interval for these 256 cycles is a fraction of a second, a much smaller time constant than the time constant of the temperature variation process. The LUT **54** contains in its address space the incremental setting of the Endless Phase Modulator Circuit select signal per one bit increment of the phase value.

The Endless Phase Modulator **24** consists of a number of delay lines built using buffers. The total length of delay should be longer than one clock cycle even for the worst case of the process variation and the temperature change. A calibration circuit **55** should be used to determine the number of delay taps that is closest to the clock interval. The calibration circuit **55** is operating all the time to adjust to changes in temperature. Assume that there is N delay taps that give a delay of almost one clock period. Each tap is worth 360/N degrees. The phase value is converted to tap number. The phase value is compared to the multiples of tap value to determine which tap should be activated. The proposed structure allows the use of only one calibration circuit **55** and one delay line. The buffers have the same delay and they track as far as process variation as well as temperature. Therefore, one can use one delay line to generate all clocks. Only the multiplex circuits are required to be individual circuits for each channel.

When desynchronizing low data rate signals, like DS1 and E1, it may be more efficient to use the dual modulus divider for large phase steps and the short delay line generate small phase steps. For the high data rates, like DS3 and E3 the whole delay line can be made out of buffers. Because the delay depends on temperature, the delay line needs to be constantly calibrated. Note that in this implementation of the Endless Phase Modulator **24** the loop bandwidth does not depend on the number of delay elements in one clock period. Only the intrinsic jitter depends on the number of delay elements in one clock cycle. The intrinsic jitter, when measured with a 10 Hz high pass filter, also depends on the frequency offset between the original clock and the synthesized clock using the Endless Phase Modulator **24**.

The calibration circuit **55**, shown in Figure **14**, is simple. An output of the D-FF#1 **80** goes high (1) on the rising edge of the clock from the first tap, and on the rising edge of N-th tap the output of the D-FF#1 **80** is reset to zero. If the rising edge from the N-th tap resets the output of the D-FF#1 **80** before the falling edge of the clock from the first tap, the signal α will have zero value. Only when the delay of N-th tap is such that its rising edge is after the falling edge of clock from the first tap, will the signal α have value one. As the counter **82** selects taps with longer and longer delays, once the counter **82** selects a tap whose delay is more than one clock cycle in respect to the first tap clock, the signal α will fall to zero, and a negative edge detector will register this tap value. This value is one more than number of taps that form a delay less than one clock period. For that reason the counter is started from the second tap.

The reason for adding the total phase increment in Endless Phase Modulator Block **24**, not in the Arithmetic Unit Block **18**, is because this block has a smaller number of steps and can perform the function of adding the total phase increment to phase at a much higher rate than 72 kHz, as it might be done in the Arithmetic Unit Block **18**. A shorter interval between the updates of delay taps selection will result in smaller jitter. For example, in the case of DS3 desynchronization, if one needs to synthesize frequency offset of 5 kHz from the 44.736 MHz frequency of the crystal oscillator clock **26**, using a 72 kHz update, one can have only 14 to 15 updates of the phase per one output clock cycle. This will result in phase steps of approximately 1/14 UI. If the phase is updated at a higher rate, for example at 500 kHz, the number of delay tap updates per one clock cycle is at least 100 and the phase step will be 1/100 UI. Because the selected delay taps will be closer to each other, glitching will be eliminated. When desynchronizing low data rate signals, like DS1 and E1, it may be more efficient to use the dual modulus divider for large phase steps and the short delay line to generate small phase steps. For the high data rates, like DS3 and E3, the whole delay line can be made out of buffers. Because the delay depends on temperature, the delay line needs to be constantly calibrated. Note that in our implementation of the Endless Phase Modulator **24**, the loop bandwidth does not depend on the number of delay elements in one clock period. Only the intrinsic jitter depends on the number of delay elements in one clock cycle. The intrinsic jitter, when measured with a 10 Hz high pass filter, also depends on the frequency offset between the original clock and the synthesized clock using the Endless Phase Modulator **24**.

The alternative embodiment of Figure **15** shows a pointer adjustment signal that can be derived in a de-mapper so a pointer adjustment block may not be required. It also shows the system consisting of a Numerically Control Oscillator Block **70**, an I and Q DAC **72** and a Single Side-band Modulator (SSB) **74** that can be used to effectively create the high frequency digitally controlled oscillator. The NCO block **70** looks exactly like the Endless Phase Modulator Block **24** except the RAM based LUT **54** can be replaced by a sinusoidal ROM (not shown). The calculated phase is used as an address to the sinusoidal ROM to convert phase to amplitude. Only one quarter of the sinusoid needs to be stored, by inverting amplitude or address, the other three quarters of the sinusoid can be created. In the NCO block **70** the sine and cosine amplitude digital values are than converted to an analog values with the use of couple of one bit fractional digital to Analog Converter (DAC) circuits (not shown). These I and Q signals are then up-converted to the proper frequency with use of the SSB modulator **74**.

Accordingly, while this invention has been described with reference to illustrative embodiments, this description is not intended to be construed in a limiting sense. Various modifications of the illustrative embodiments, as well as other embodiments of the invention, will be apparent to persons skilled in the art upon reference to this description. It is therefore contemplated that the appended claims will cover any such modifications or embodiments as fall within the true scope of the invention.

WHAT IS CLAIMED IS:

1. A desynchronizer for desynchronizing one or multiple channels of SONET/SDH data signals, comprising:

(a) a first in first out (FIFO) buffer having an input coupled to said data signals and an output for outputting asynchronous data obtained from one or more of said SONET/SDH data channels;

(b) an arithmetic unit operative to perform all operations required for single or multi-channel desynchronization coupled to said FIFO; and

(c) an endless phase modulator coupled to said arithmetic unit and to said FIFO and operative in response to input from said arithmetic unit to produce a single output desynchronized clock or multiple output desynchronized clocks.

Figure 1 Block Diagram of Preferred Multi Channel Desynchronizer

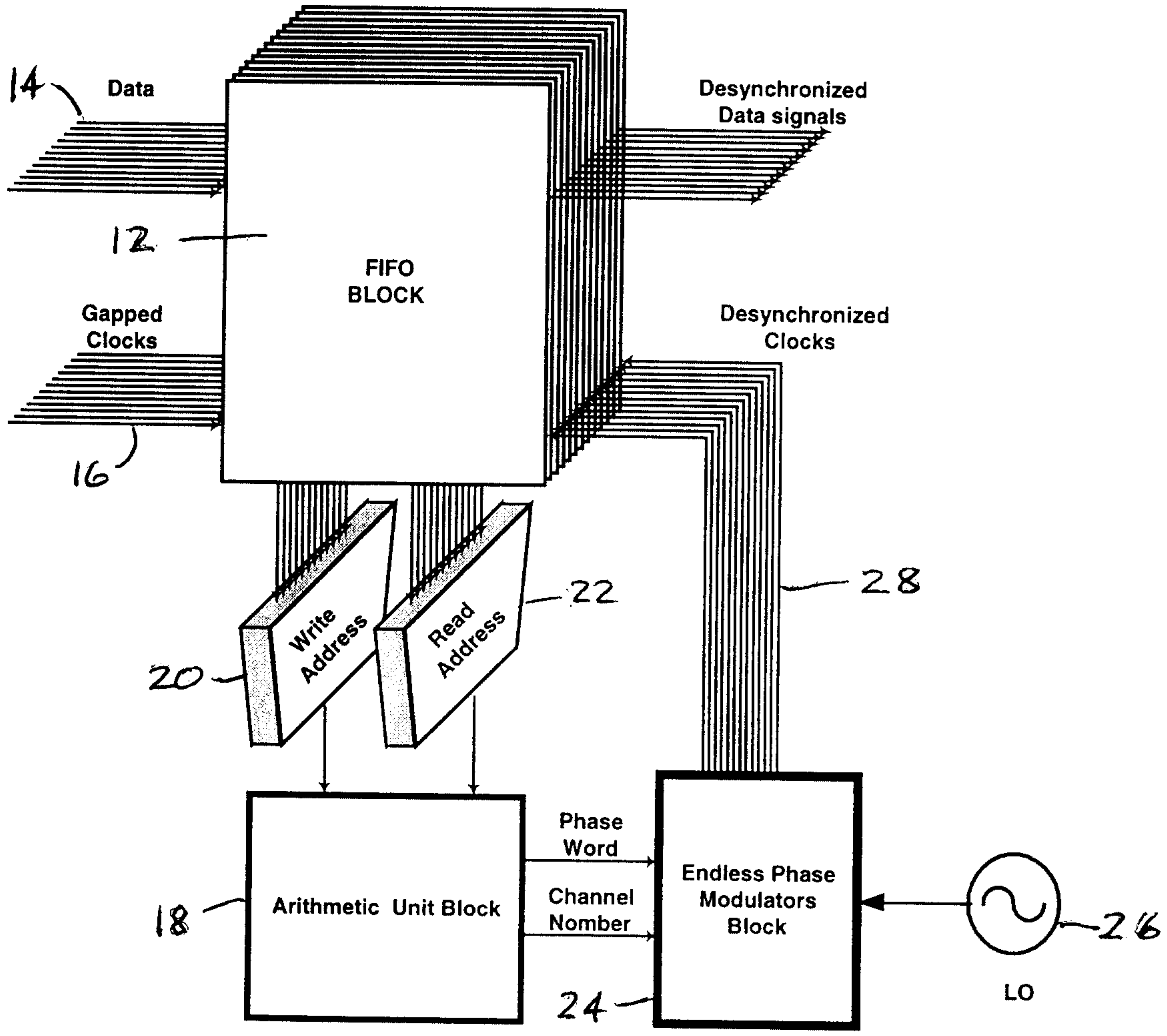


Fig. 1

Arithmetic Unit Block Diagram

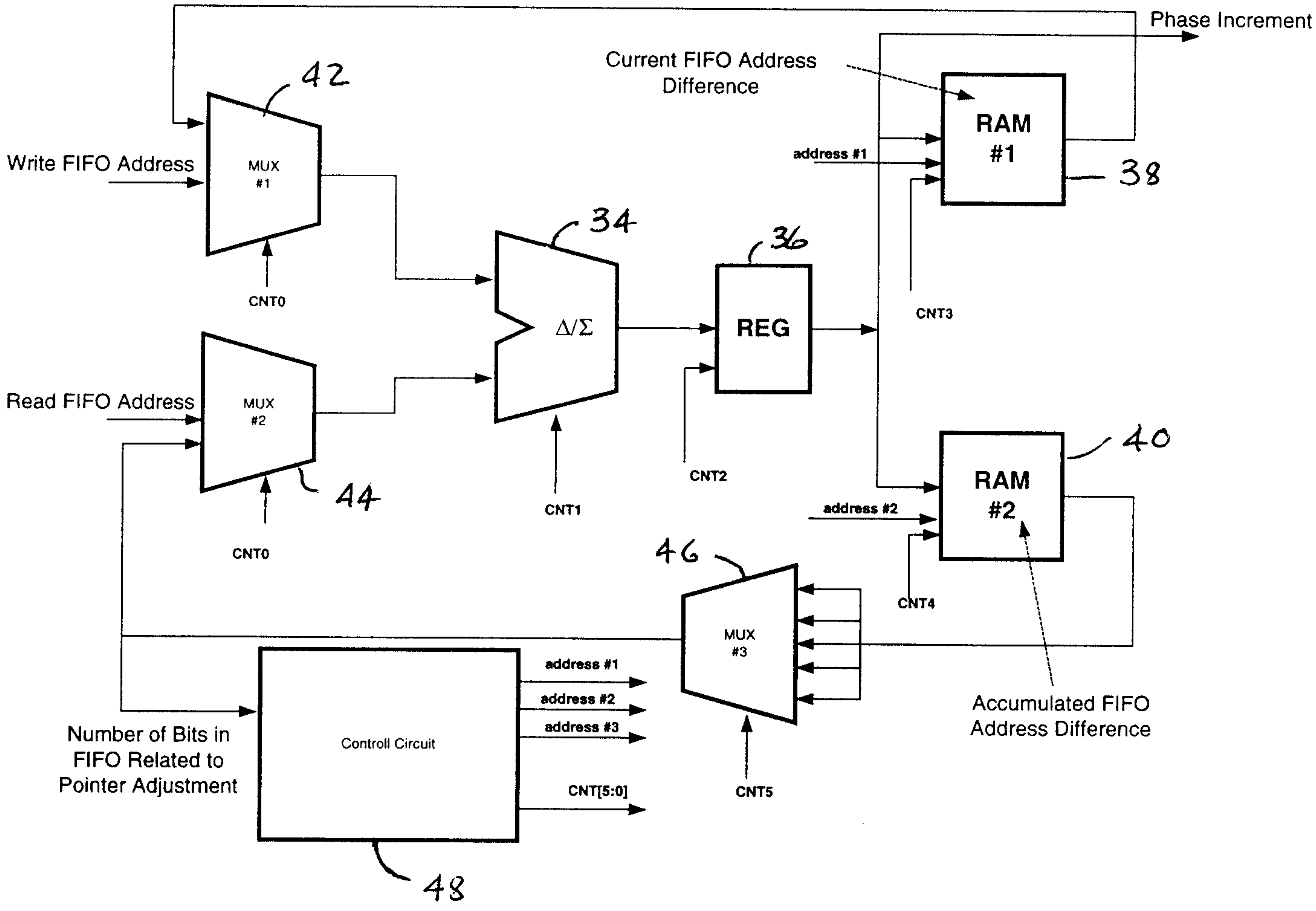


Fig. 2

Memory Map of RAM#1 for desynchronizing 12 channels of DS3 signal dropped from OC-12 signal

Ch#1 FIFO Address Difference
Ch#2 FIFO Address Difference
Ch#3 FIFO Address Difference
Ch#4 FIFO Address Difference
Ch#5 FIFO Address Difference
Ch#6 FIFO Address Difference
Ch#7 FIFO Address Difference
Ch#8 FIFO Address Difference
Ch#9 FIFO Address Difference
Ch#10 FIFO Address Difference
Ch#11 FIFO Address Difference
Ch#12 FIFO Address Difference
0
8
1/64 of UI Phase Increment
1/64

NOTE: N can be chosen for specific leak rate. Few more addresses can be added to the RAM#1 address space to enable adaptive bit leak rate!

Memory Map of RAM#2 for desynchronizing 12 channels of DS3 signal dropped from OC-12 signal

Ch#1 Accumulated FIFO Address Difference
Ch#2 Accumulated FIFO Address Difference
Ch#3 Accumulated FIFO Address Difference
Ch#4 Accumulated FIFO Address Difference
Ch#5 Accumulated FIFO Address Difference
Ch#6 Accumulated FIFO Address Difference
Ch#7 Accumulated FIFO Address Difference
Ch#8 Accumulated FIFO Address Difference
Ch#9 Accumulated FIFO Address Difference
Ch#10 Accumulated FIFO Address Difference
Ch#11 Accumulated FIFO Address Difference
Ch#12 Accumulated FIFO Address Difference
Ch#1 Pointer Adjustment Bits #
Ch#2 Pointer Adjustment Bits #
Ch#3 Pointer Adjustment Bits #
Ch#4 Pointer Adjustment Bits #
Ch#5 Pointer Adjustment Bits #
Ch#6 Pointer Adjustment Bits #
Ch#7 Pointer Adjustment Bits #
Ch#8 Pointer Adjustment Bits #
Ch#9 Pointer Adjustment Bits #
Ch#10 Pointer Adjustment Bits #
Ch#11 Pointer Adjustment Bits #
Ch#12 Pointer Adjustment Bits #
Spare Address for Holding Intermediate Values

Fig. 3

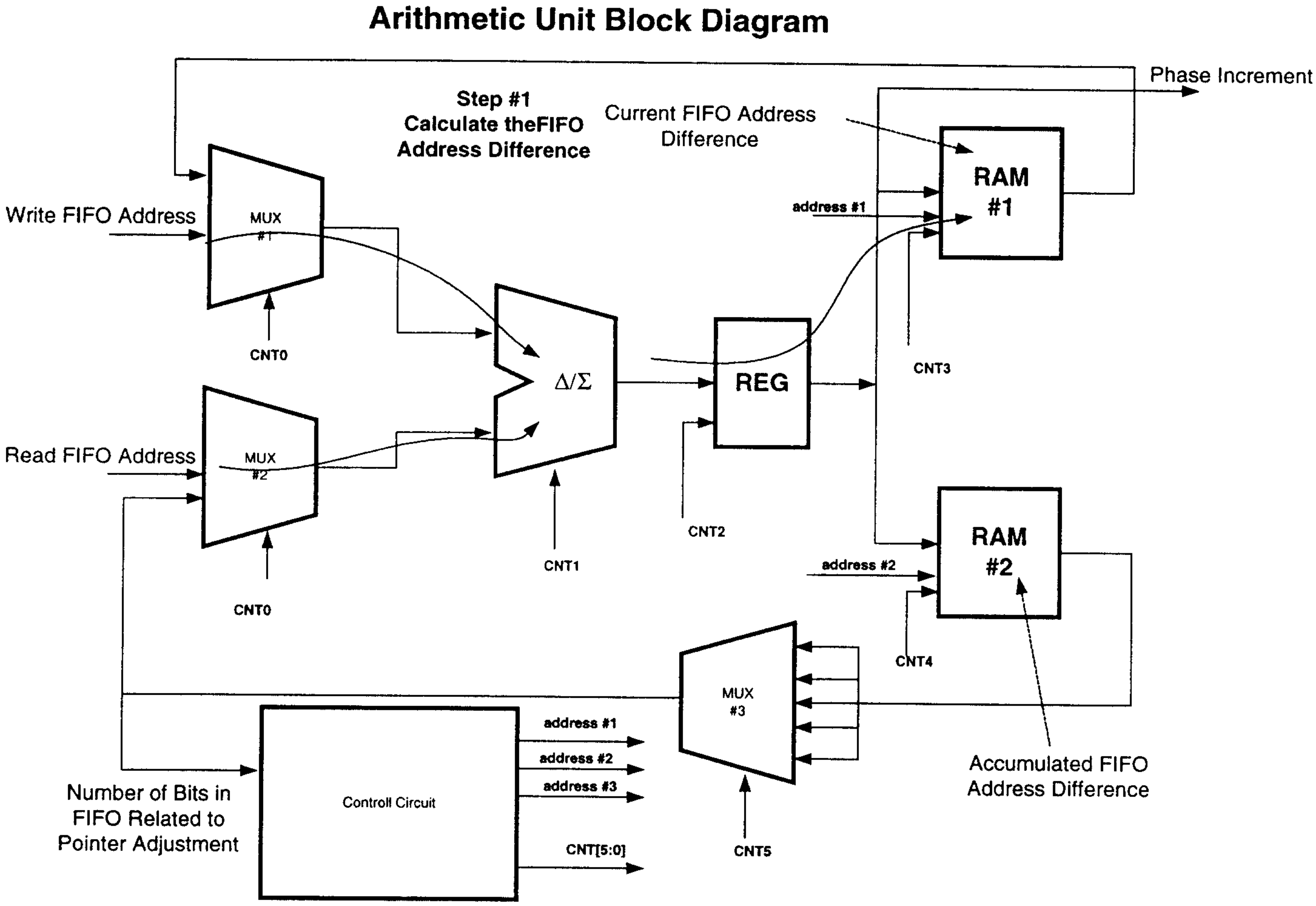

Arithmetic Unit Block Diagram
Step #1
Calculate theFIFO
Address Difference
Current FIFO Address
Difference
Phase Increment
Write FIFO Address
MUX
#1
CNT0
Read FIFO Address
MUX
#2
CNT0
Δ/Σ
CNT1
REG
CNT2
RAM
#1
address #1
CNT3
RAM
#2
address #2
CNT4
MUX
#3
CNT5
Accumulated FIFO
Address Difference
Number of Bits in
FIFO Related to
Pointer Adjustment
Controll Circuit
address #1
address #2
address #3
CNT[5:0]

Fig. 4

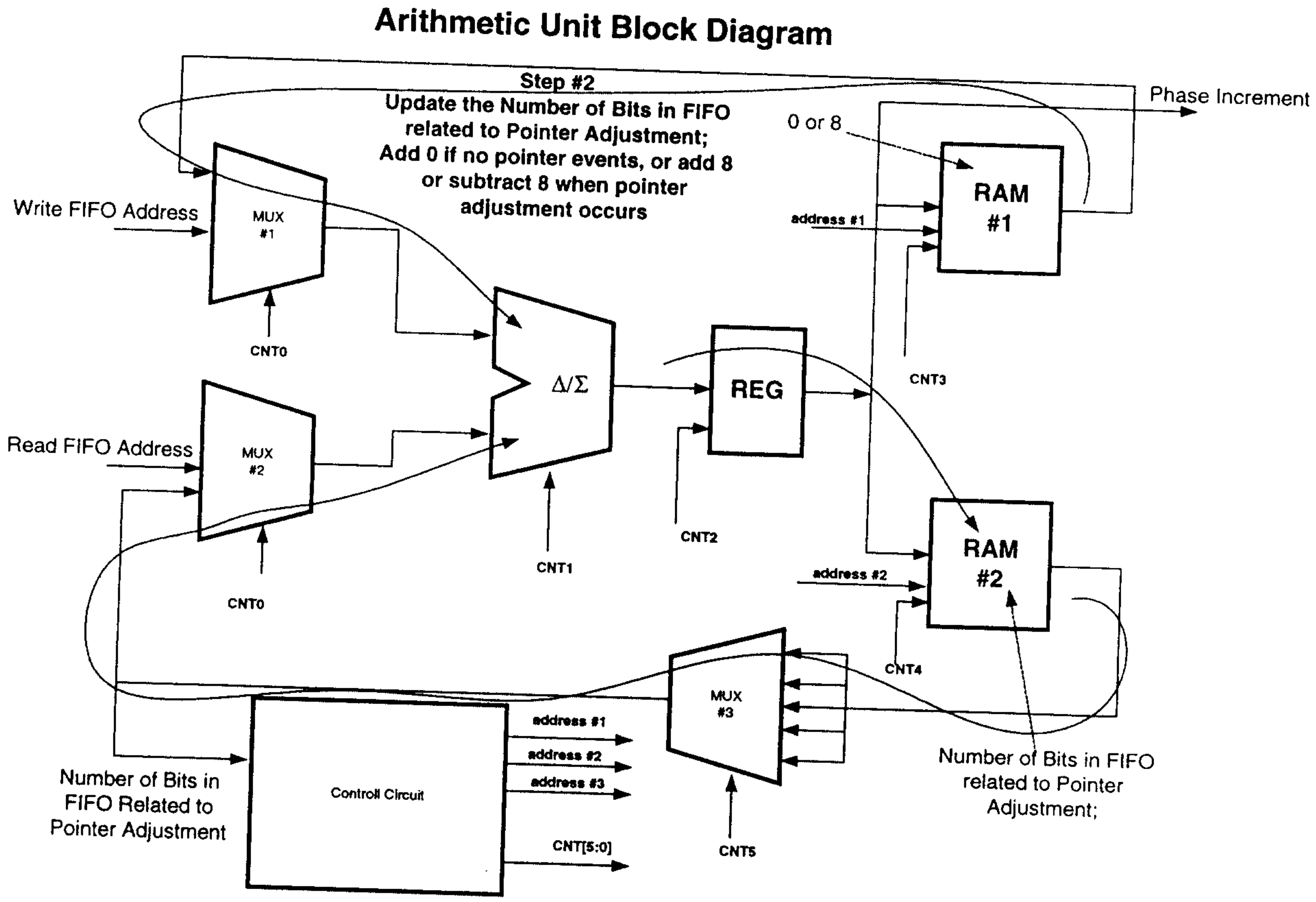

Arithmetic Unit Block Diagram
Step #2
Update the Number of Bits in FIFO
related to Pointer Adjustment;
Add 0 if no pointer events, or add 8
or subtract 8 when pointer
adjustment occurs
Phase Increment
0 or 8
RAM
#1
address #1
CNT3
Write FIFO Address
MUX
#1
CNT0
Δ/Σ
REG
Read FIFO Address
MUX
#2
CNT0
CNT1
CNT2
RAM
#2
address #2
CNT4
MUX
#3
CNT5
address #1
address #2
address #3
CNT[5:0]
Controll Circuit
Number of Bits in
FIFO Related to
Pointer Adjustment
Number of Bits in FIFO
related to Pointer
Adjustment;

Fig. 5

Arithmetic Unit Block Diagram

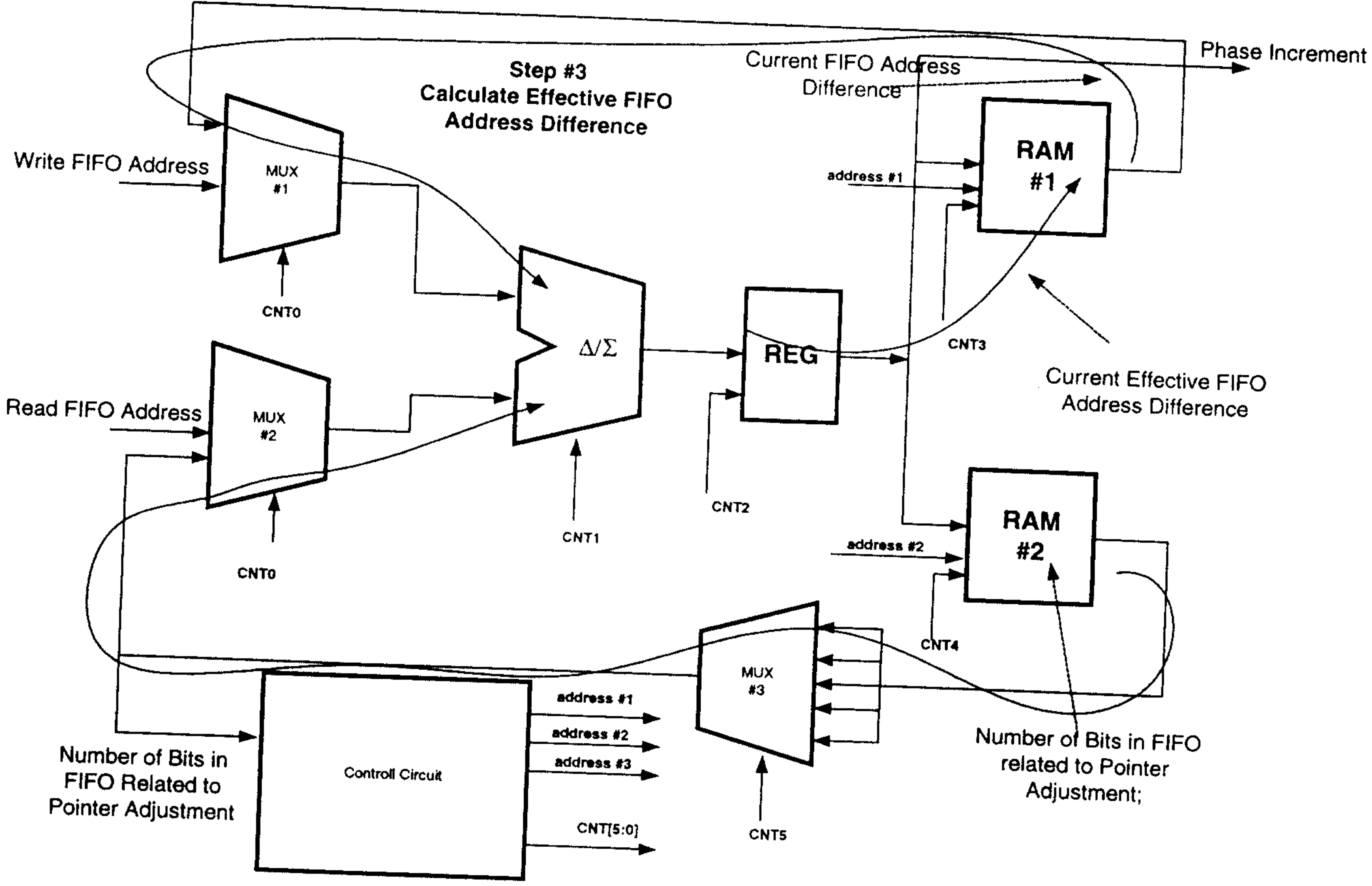


Fig. 6

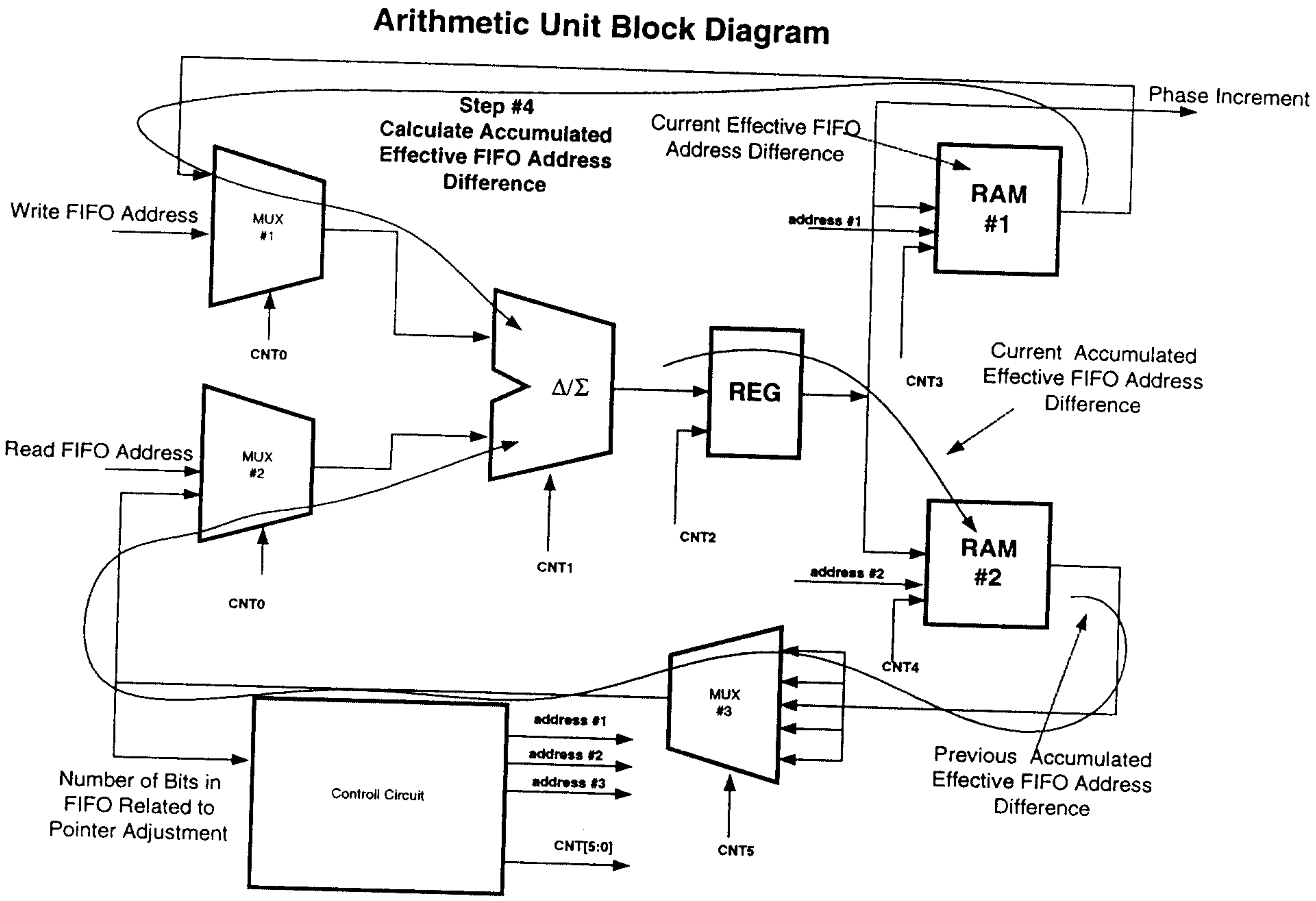

Arithmetic Unit Block Diagram
Step #4
Calculate Accumulated
Effective FIFO Address
Difference
Current Effective FIFO
Address Difference
Phase Increment
Write FIFO Address
MUX
#1
CNT0
RAM
#1
address #1
CNT3
Δ/Σ
REG
Current Accumulated
Effective FIFO Address
Difference
Read FIFO Address
MUX
#2
CNT0
CNT1
CNT2
RAM
#2
address #2
CNT4
MUX
#3
CNT5
Controll Circuit
address #1
address #2
address #3
CNT[5:0]
Number of Bits in
FIFO Related to
Pointer Adjustment
Previous Accumulated
Effective FIFO Address
Difference

Fig. 7

Arithmetic Unit Block Diagram

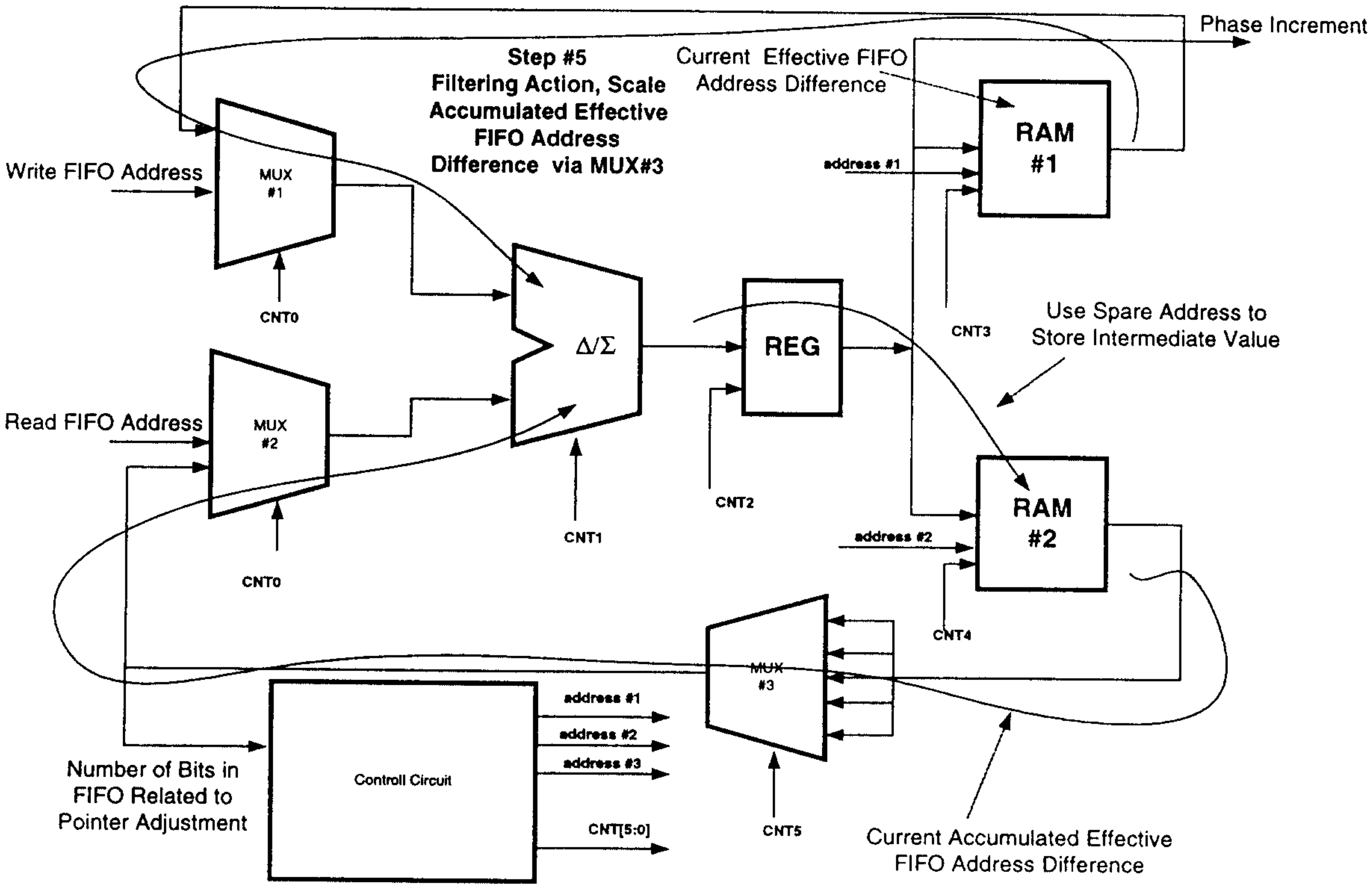


Fig. 8

Arithmetic Unit Block Diagram

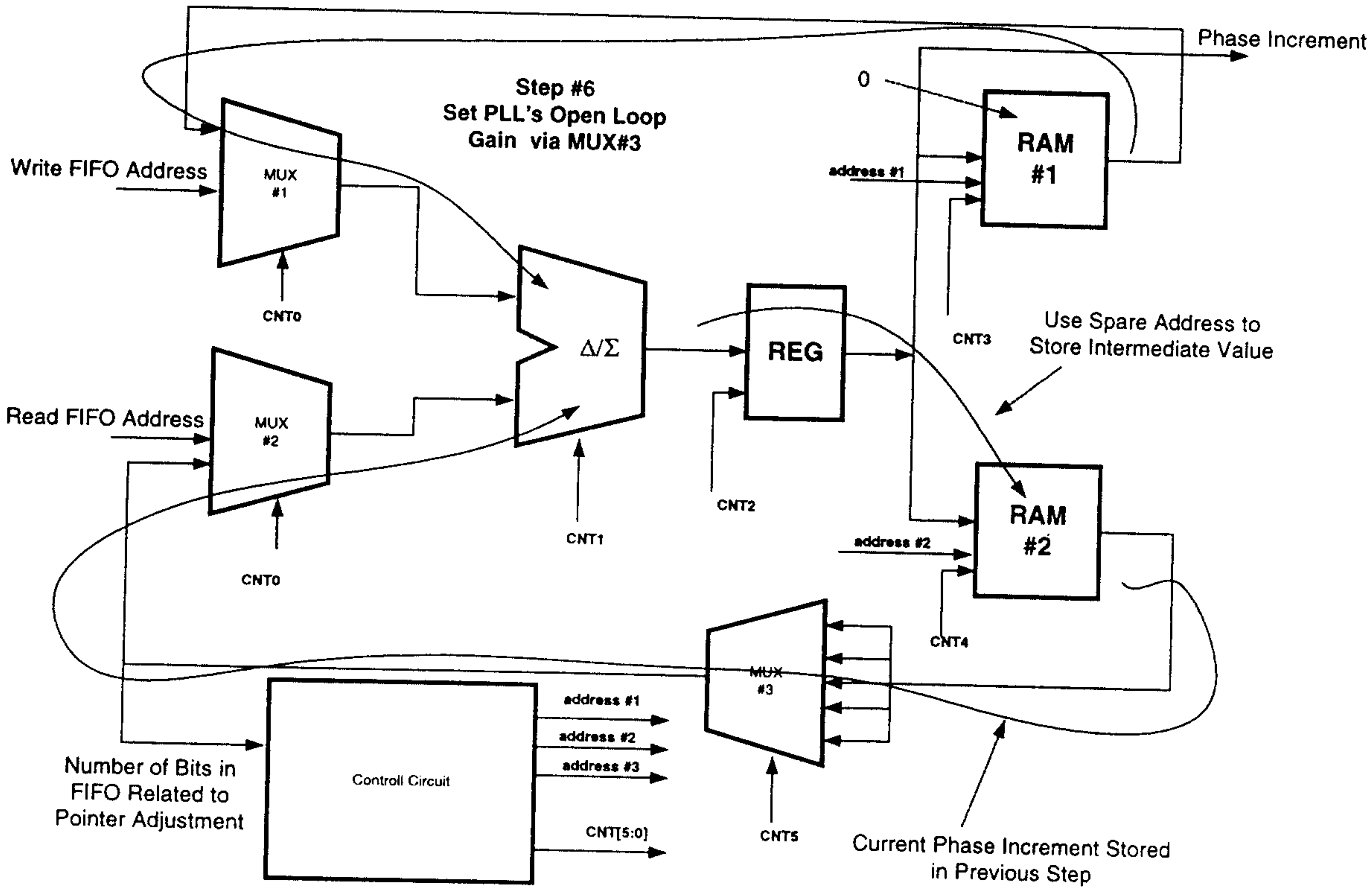


Fig. 9

Arithmetic Unit Block Diagram

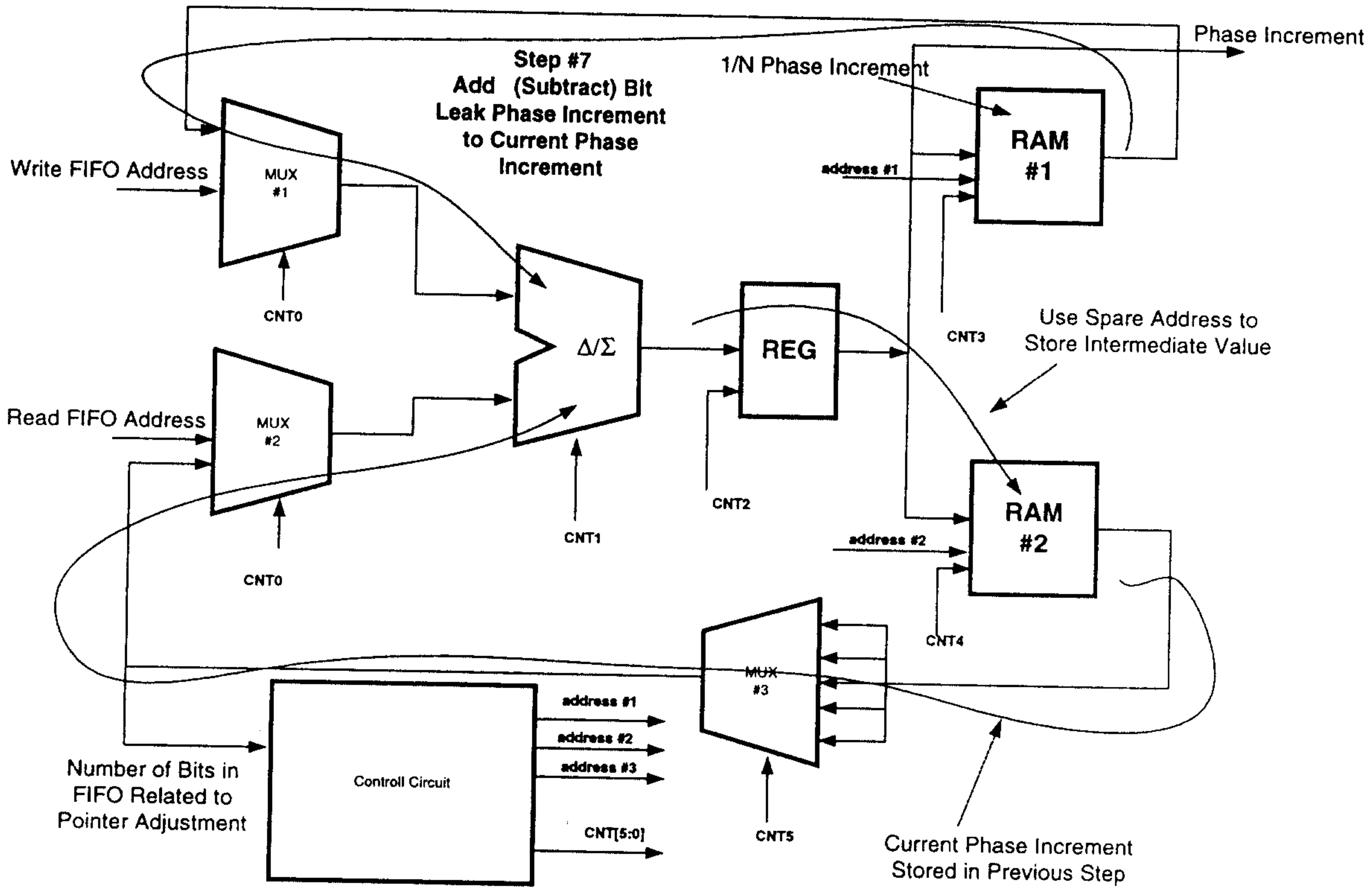


Fig. 10

Arithmetic Unit Block Diagram

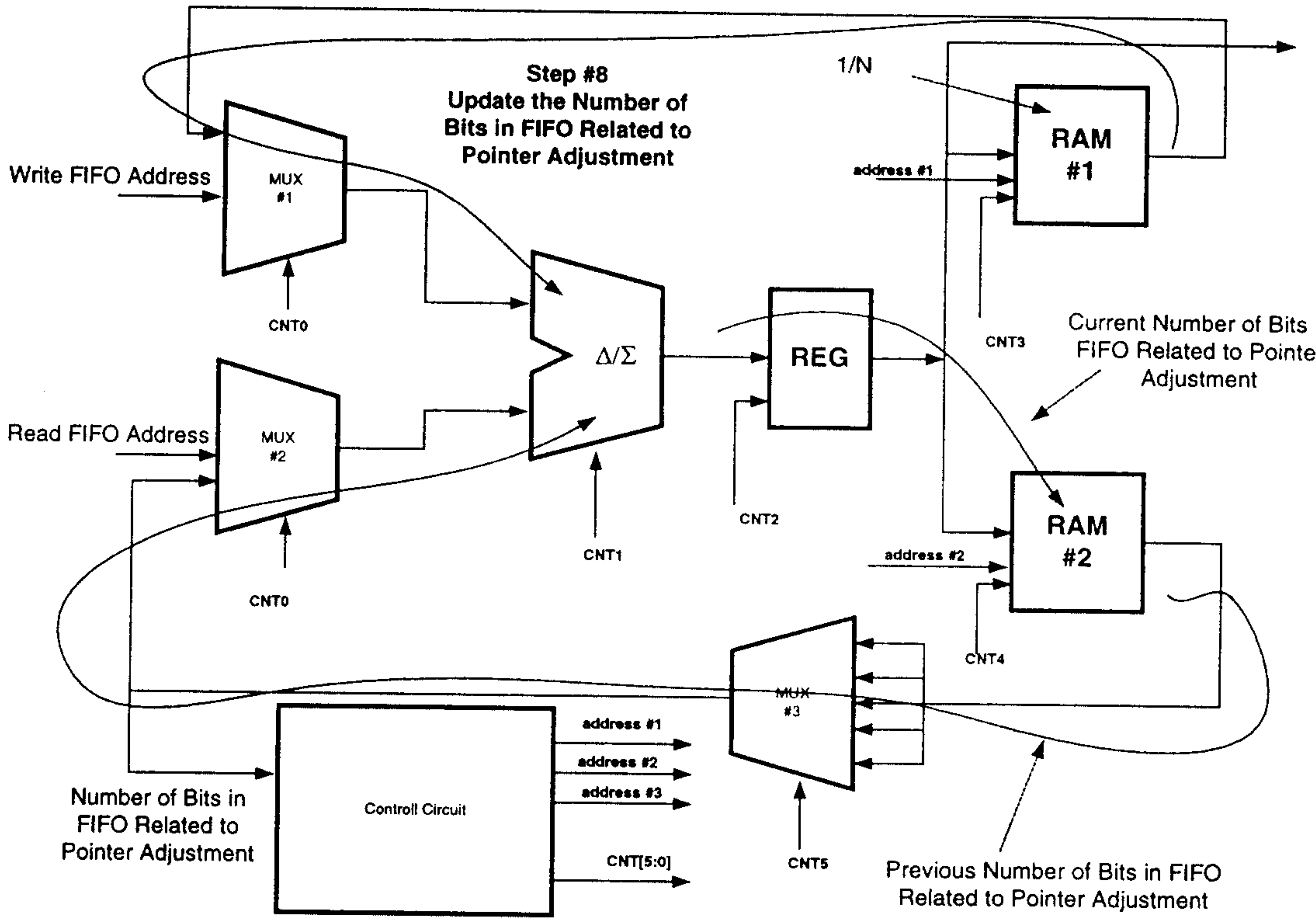


Fig. 11

Endless Phase Modulators Common Control Block

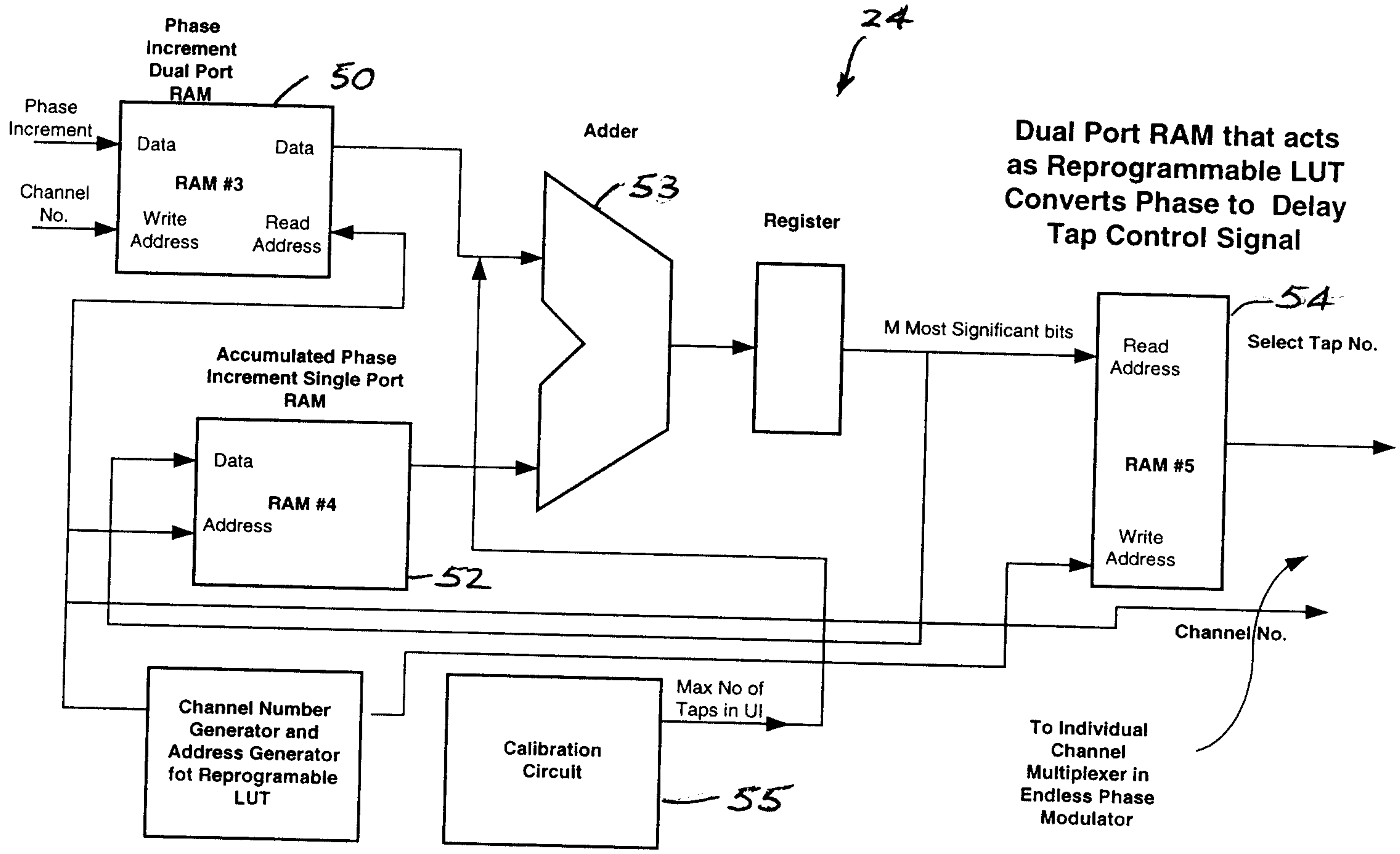


Fig. 12

Figure 13 Endless Phase Modulator Delay Line

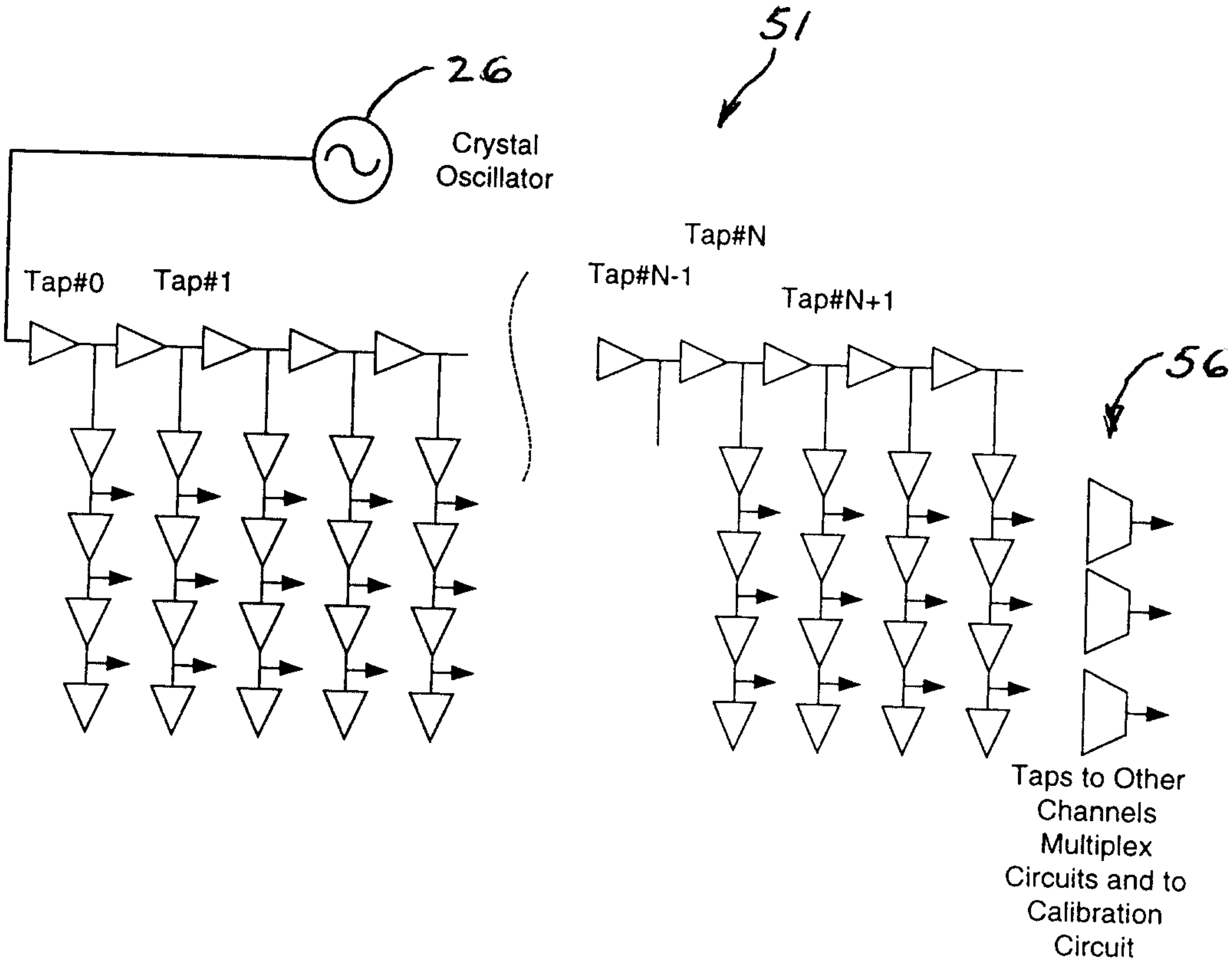


Fig. 13

Figure 14 Delay Line Calibration Circuit

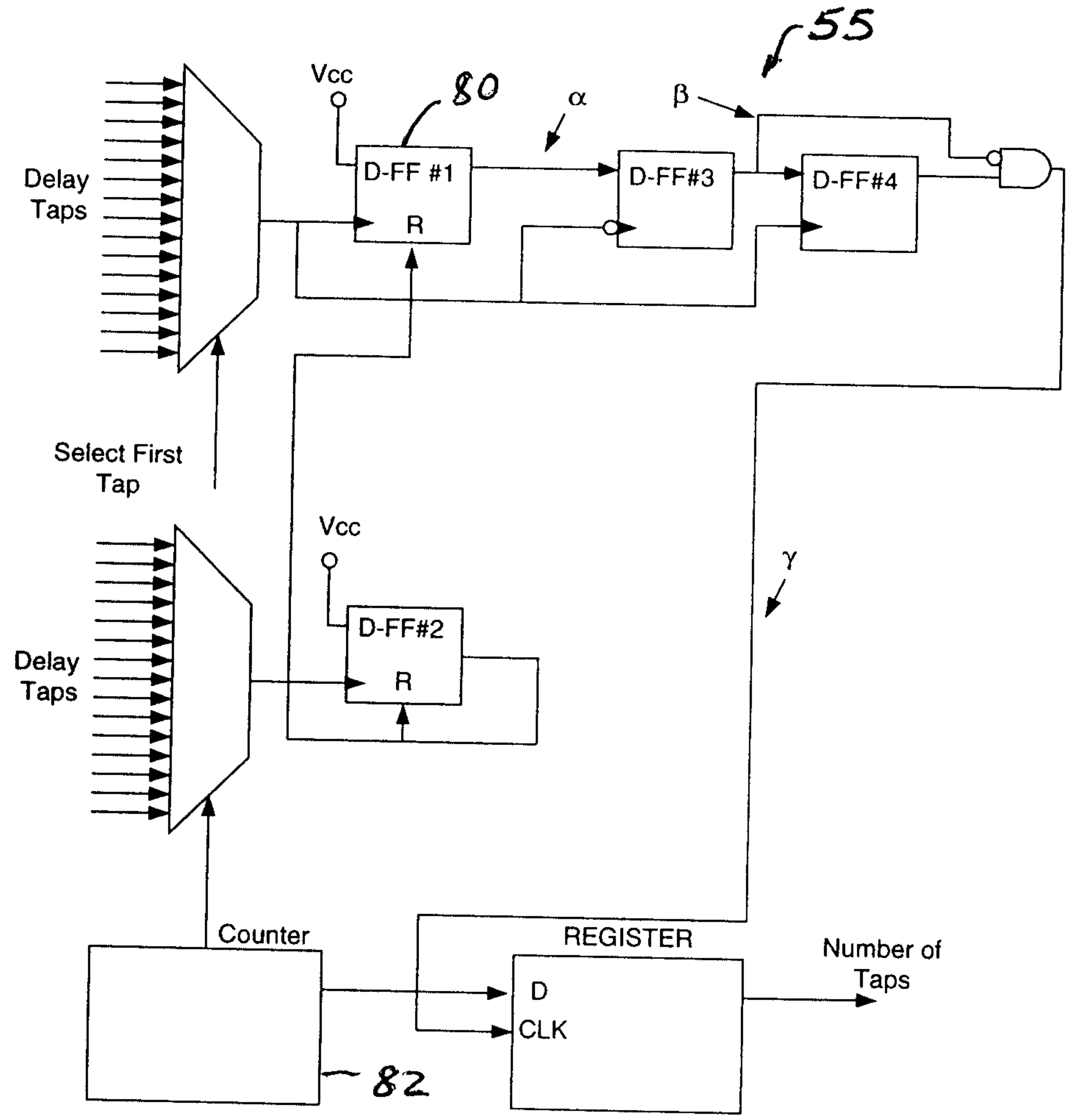


Fig. 14

Figure 15 Alternate Embodiment Using SSB Modulators

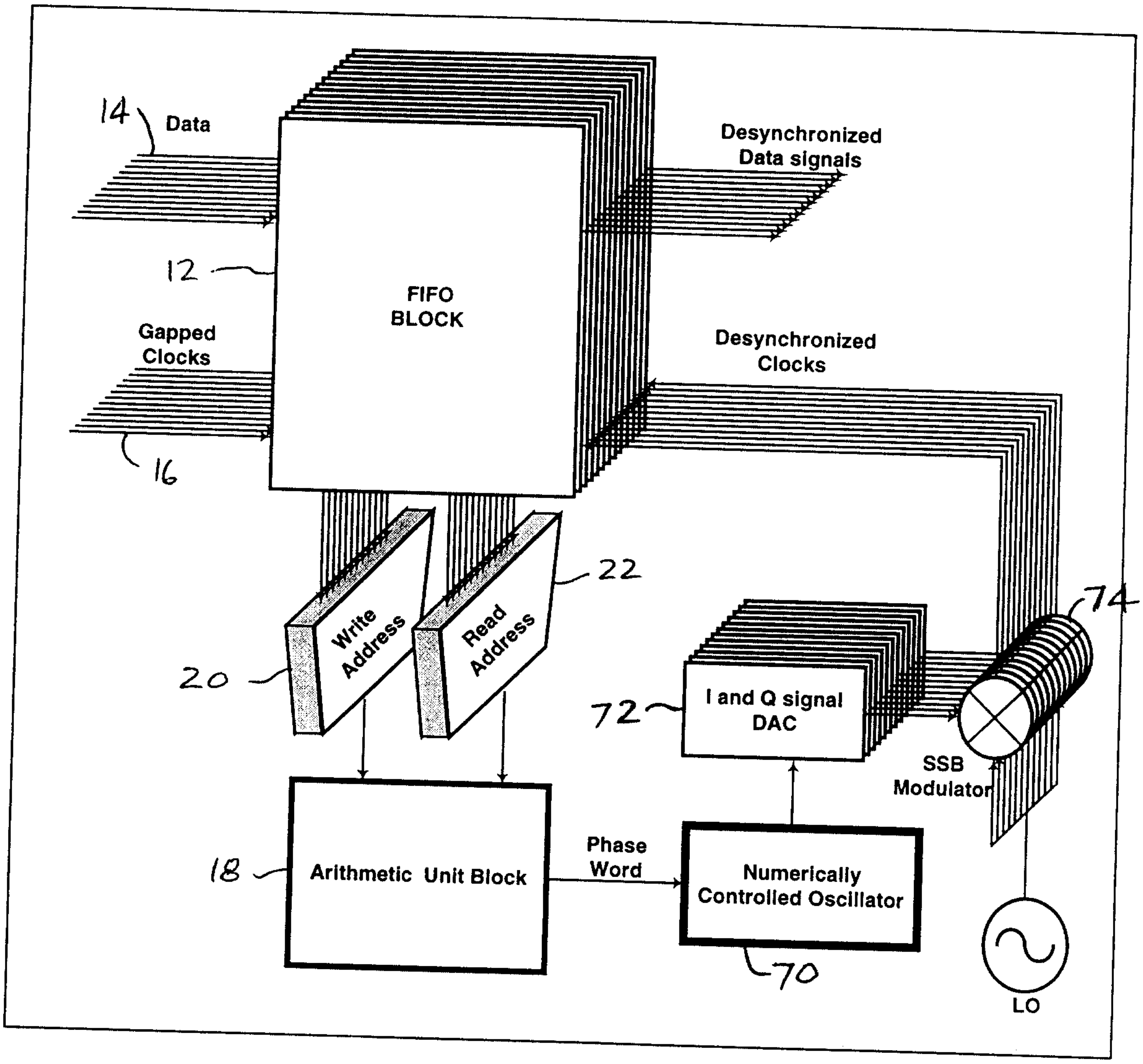


Fig. 15

Block Diagram of Preferred Multi Channel Desynchronizer