A Digital Phase Adjustment Circuit for ATM and ATM- like Data Formats

by
Thomas J. Chaney

Applied Research Laboratory
Department of Computer Science
Washington University
St. Louis, Missouri 63130
tom@arl.wustl.edu

Working Note ARL-98-02
March 31, 1998

Digital Phase Recovery Adjustment for High Speed Transfer of ATM and ATM like Data Formats

Background and Summary

A significant, if not over riding, problem with electronic systems that must transmit data rapidly between IC (Integrated Circuit) packages is the issue of determining, at the receiver, when to sample the interconnecting wires to record the transmitted data values. Traditionally, the sampling time is established by a "clock" signal that is distributed to all IC packages. As data transmission rates have increased, the simple "clock" signal schemes have began to fail as it has become more difficult to assure that the "clock" signal is received by all IC packages at same time. The "clock skew" problem in high speed systems is well known. This patent describes a high speed data transfer circuit that takes advantage of the structure of ATM and ATM-like data formats in a way that significantly extends the rate that ATM and ATM-like cells can be transmitted, both between and, for large circuits, inside, the IC packages. This scheme is optimized for a local region where it is reasonable to limit the worst case phase shift between the distributed clock and the data within the local region to a few clock periods. This scheme is useful for data flow within an integrated circuit, on a printed circuit board, or in a cabinet full of printed circuit boards.

A key element of this scheme is the distribution of a clock **RATE,** but not a clock **PHASE**. The received data are aligned with the local clock phase at each receiver site. Other key elements of this scheme include the ability to utilize IC chip logic elements that have a propagation delay variation, over process, temperature, and voltage variations, of on the order of four to one. All logic elements used in the phase alignment circuit are components found in a typical vendor's digital design library. The size of this phase alignment circuit is small when implemented using current IC technologies. (Feature sizes of 1 um or less.) With a 0.8 micron technology, 200 gates could be fit in the chip area required by a wire bond pad. The preferred embodiment circuit, under some better case conditions, will require the area of about 95 gates, or about one half the area of a pad. Under other worse case conditions, the preferred embodiment circuit require the area of about 210 gates, or about the area of a pad.

This scheme requires that the incoming data be sampled several times each clock period. Another key is that, within the small area on a single chip, the propagation delays of logic elements are fairly well matched. This fact is exploited in the form of a circuit using a metric of "the number of inverter gates of delay per clock period". This scheme is a clock phase alignment circuit that corrects itself, during one cell time, by some **fraction** of the number of inverter gates that have a one clock period delay. There is only one clock period associated with each chip so, in the limit, only one circuit that generates the number-of-inverters-per-clock-period value is needed. (Considerations of the extent of the area over which it is reasonable to assume circuit delay matching may force additional copies of the number-of-inverters-per-clock-period circuit on each chip.) While the concept of **fractions** of an inverter delay string was a important key, there are other refinements that have been incorporated in the design of this phase correction circuit. For relatively short links between IC chips, the maximum phase shift is on the order of a few clock periods. Also, once the circuit link is established, the fact is utilized that the **change** in phase shift associated with the received data will vary slowly. This scheme exploits the fact that, on any link, there are always a few "unassigned" (or "idle") cells mixed in with the data cells. No link can truly operate at 100% of capacity without noticeable cell loss. Thus even at maximum capacity, every link will have some small percentage of unassigned cells included in the cells flowing through the link.

While the principal advantages and features of the scheme have been briefly described above, a more thorough understanding may be attained by referring to the drawing and description of the preferred embodiment which follows.

Brief Description of the Drawings.

Figure 1 is a block diagram showing the overall structure of the proposed interconnection system;

Figure 2 shows the key logic components of the preferred embodiment

Figure 3 shows the minimum clock period equations for the preferred embodiment

Figure 4 is a refined Figure 2 to allow even faster clocking

Figure 5 is a complete data handling circuit for the preferred embodiment

Figure 6 shows the control circuits necessary for Figure 5

Figure 7 is a diagram illustrating the possible states of the Figure 6 control circuit

Figure 8 is an linear section of the diagram shown in Figure 7 (With some additional details shown)

Figure 9 is a complete logic circuit of the preferred embodiment

Figure 10 is a possible large area control circuit needed to generate the three phase shifted clocks.

Figure 11 illustrates the timing of the Figure 10 circuit under several circuit fabrication and operation conditions.

Detailed Description of the Preferred Embodiment

The basic idea of this clock phase recovery scheme is to align the phase of the incoming signals with the local clock. Thus, as shown in Figure 1, there are phase alignment circuits associated with each set of inputs. A "set of inputs", in this context, are all the input signals that can be considered to have the same clock phase relationship. In other words, a " set of inputs" are all the inputs that originate from another IC package and that are carefully routed together across the printed circuit board to this IC package. Also implied in this "set of inputs" definition is some care in the design of the output stages of the "other IC package". It is implied that the output stages be designed so that all output signals, as seen at the package pins, switch at the same time. A typical implementation places a flip-flop near each output pad, and the clock line to all these flip-flops is such that there is little clock phase difference between all the flip-flop clock inputs.

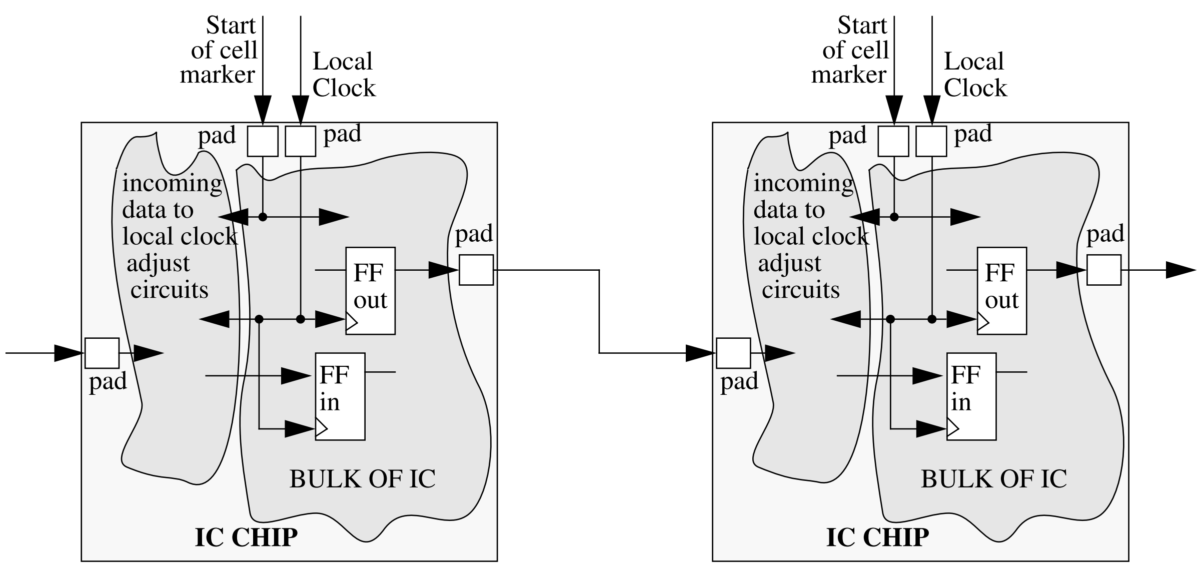

FIGURE 1
Basic Digital Phase Adjustment Scheme

Figure 1 presents the basic idea of the phase recovery scheme. There is only one input signal and one output signal shown in Figure 1. In a general circuit, there would be an incoming data “cloud” for each “set of inputs”. The general circuit would still have a single BULK OF IC “cloud”, but there would be a “FF out” flip-flop associated with each output pad. Note that the incoming data, after being conditioned by the incoming data “cloud” circuits, must be latched (in “FF in”) using the local clock before the incoming signal can be “used” by the BULK OF IC circuitry. All these flip-flops imply cell delay of several clock periods. This delay is acceptable in this application because the cell level hand shake signals (flow control signals) have tens of clock periods to be created and propagated to the other IC packages. Thus it is important that the control timing be much slower, measured in data word clock periods, than the data words. (The cell minimum period to data word period ratio needed to utilize this scheme is in the range of eight, or nine, to one.)

Relative to “on the chip” circuit delays, the pad driver-receiver pair has a long propagation delay. By using pad circuit designs that have a large gain-bandwidth product, is possible to have several data values propagating along the wire between the transmitter and the receiver, all at the same time. (All high speed long distance communication schemes allow several transitions along the interconnecting wire at a time.) This scheme will function properly with multiple data values propagating along the signal path between the driver and the receiver at the same time.

For the preferred embodiment, an indication of about when to expect the first data word of a new cell is needed, thus each IC CHIP is expected to receive a start-of-cell marker. For this development, it is expected that the start-of-cell marker will track the clock signal as it is distributed. This requirement may require that the start-of-cell marker be “re-clocked” using a flip-flop at intermediate points, but this is a single signal and is thus not a large expense.

For the example embodiment, it has been assumed that the clock phase recovery circuit must be able to cover a phase range of about 700 degrees of phase shift. It is important that ***ALL*** the incoming signals, regardless of their source, pass from the incoming data “clouds” to the BULK OF IC “cloud” on the same local clock transition. Thus some of the incoming data “clouds” may have to delay the incoming data an extra clock period while other incoming data ‘clouds” will not. The start-of-cell marker signal is the reference signal needed by each incoming data “cloud” to assure that all incoming data “clouds” deliver their incoming data cells in unison.

The preferred embodiment involves finding the metastable point, the timing condition that results in the captured data final value being unrelated to the incoming data value. Once the timing for metastability is known (or bounded), then timing can be set to avoid metastability. Based on the timing set before or after the metastable region, the circuit can be set to delay or not delay the captured signal an extra clock period.

It should be noted that this scheme does not compensate for differences in the propagation delay of positive and negative going transitions. The transition direction differences are typically small. Thus there is little value in attempting to include transition direction delay adjustments.

The “stay away from metastability” embodiment involves sampling the incoming data stream at three different clock times ***when a known data pattern is being transmitted***, then using the data sampled by the one clock timing that is known to be away from the metastable region. The known data pattern can be the unassigned cell that is transmitted throughout the switch when there are no “real” cells to transmit. The switch circuitry can be designed such that unassigned cells are transmitted during the latter part of the reset period. A circuit that describes this data skew correction scheme for a single data bit is shown is Figure 2. The ideal operation of the circuit will be presented first to illustrate how this scheme works, then more realistic tolerances will be introduced later. The three clocks, clk1, clk2, and clk3 have the global clock frequency, and, for illustrative purposes, are phase adjusted so that each is one third a clock period ahead of its neighbor. (The phase relationship among the three clocks is shown as part of Figure 2.) Thus the incoming data is sampled three times during each clock period. The MUX1 will select one of the three sampled data sets as the “real” data set based on a set up period, or “learning” period, during which a known data pattern is being transmitted. If it is known that there is a low-to-high data transition during a particular clock period, then the following rules can be applied to pick the stored value that was clocked at least one third a clock period away from the clock timing that results in metastability.

- If ***A, B*** and ***C*** are equal (all high), use latch output ***B*** and use output ***Y.***
- If ***A*** is different than ***B*** and ***C,*** use latch output ***C*** and use output ***Y.***
- If ***C*** is different than ***A*** and ***B,*** use latch output ***A*** and use output ***X.***

If all three latched outputs are the same (high), then the incoming data must have changed states between clk3 and clk1 as shown as "Data In, Example 1" in Figure 2. Thus the data value collected by clk2 must have been collected at least one third a clock period away from the time the incoming data changed states. Likewise, the other two rules assure that the collected data value that was used was collected at least one third clock period away from the timing that causes metastability. Returning to the first rule, if the data changed states **AT** clk3 or clk1, then the associated latch may have reported a "high" or a "low". Considering the case where the data arrival timing was such that clk3 caused metastability, if ***C*** reported a value the same as ***A*** and ***B,*** then output ***B*** will be used, which was collected exactly one third a clock period before the metastable timing[1]. On the other hand, if ***C*** reported a value different than ***A*** and ***B,*** then output ***A*** will be used, but ***A*** was collected one third a clock period after the metastable timing. Thus it does not mater which state is reported from ***C***, the circuit will function properly. The final output, ***X*** or ***Y,*** that was used changed in this example depending on the final state of ***C***. To explain this curious effect, consider the timing diagram at the bottom of Figure 2. In the third case (Data In, Example 3), where ***A*** is selected to be the output used, the value stored as ***A*** is collected *after* the data changed state. For the cases where ***B*** or ***C*** is used, the data is collected *before* the data changed state Therefore, when ***A*** is used, the number of clock periods the output is delayed must be reduced by one.

The number of output stages of clock period delay can be increased as needed to allow the initial establishment of the alignment of the first byte of each input cell with all the other input cells. Once the first establishment is accomplished, as the input timing slowly drifts due to temperature or supply voltage, the change in timing can be adjusted using the three rules listed above, with the "use output ***X/Y"*** part of the rules re-worded using "add/remove one clock period of delay". Thus, *as long as the timing drifts are slow,* once the phase recovery circuit is initialized, the phase lock can be maintained, even if the timing drift covers several clock periods. The "*as long as the timing drifts are slow"* statement means that the maximum timing drift between synchronization data patterns is limited to one third a clock period. Expected timing drifts are in the millisecond to second per nsec. of drift region. Thus the density of synchronization data patterns needed is at most one synchronization pattern per thousand cell times. It is not possible to provide reliable cell transfer in an ATM switch with any link fully loaded. The switch path contention will cause the switch to fail. (have cell loss) Thus it is reasonable to expect that **at least** one in every **hundred** cells on any data path within a switch are unassigned cells. The unassigned cell format can be designed to fill the need for a synchronization cell.

1 It is assumed here that the output of FF3 is given sufficient time to resolve so that the probability of FF3 being unresolved at the output sample time is acceptably low. The circuits used to achieve this requirement will be presented later.

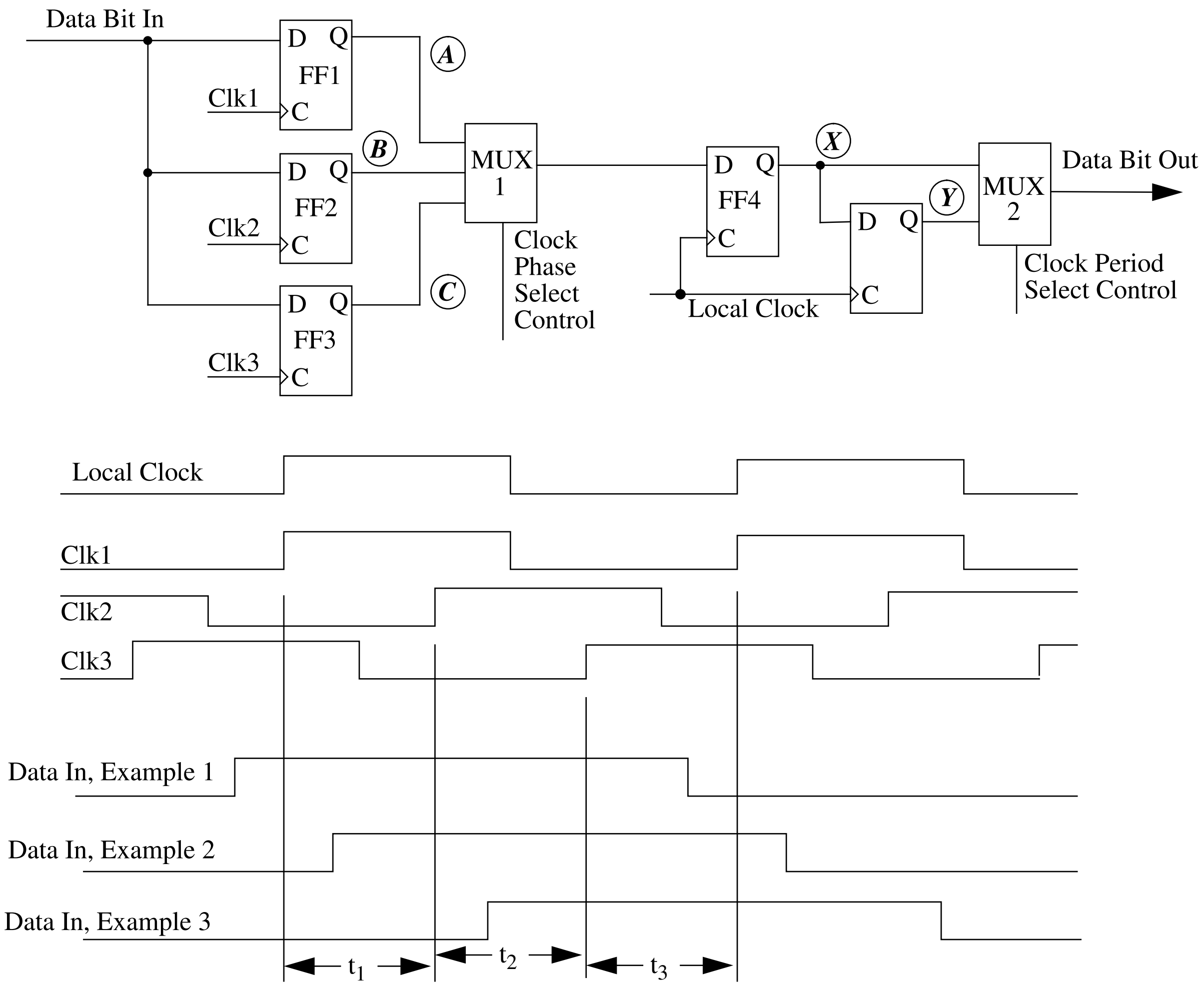

FIGURE 2
Simplified "Stay Away From Metastability" Embodiment

Deferring the description of the control circuits needed for MUX 1 and MUX 2 of Figure 2, consider how small a time interval can be used between Clk1, Clk2, and Clk3. There are two limits, and both must be meet.

- (1) The time interval between each clock (t_1, t_2, or t_3) must be larger than the sum of: (a) clock jitter, (b) the timing differences between data rising and falling transitions, and (c) the tolerance of setting the clocking time interval between Clk1, Clk2, and Clk3.
- (2) For t_3, the time period between collecting a data sample and the local clock must be enough to allow the collected sample to pass through the collecting flip-flop, MUX 1, and then meet the setup requirements of another flip-flop; and then also meet the clock jitter and the tolerance of setting the clocking time interval. Using the timing shown in Figure 2, the time interval between Clk3 and the Local Clock is the critical time period for this timing limit.

The time interval needed to meet limit (1) above. as depicted in Figure 3, is approximately 1.25 nsec. for the 0.8 micron process. [Using for each part of (1); (a) about one forth nsec., (b) about one half nsec., and (c) about one half nsec.] The time interval needed to meet limit (2) above. as depicted in Figure 3, is approximately 3.25 nsec. [approximately 1 nsec. for each; flip-flop propagation delay, mux propagation delay, and flip-flop setup, plus the 0.25

nsec for clock jitter.] Thus the minimum clock period for the Circuit of Figure 1 is approximately 1.25 + 1.25 +1.00 + 1.00 + 1.00 + 0.25 = 5.75 nsec. and thus the maximum clock frequency is about 170 Mhz. for this scheme using a

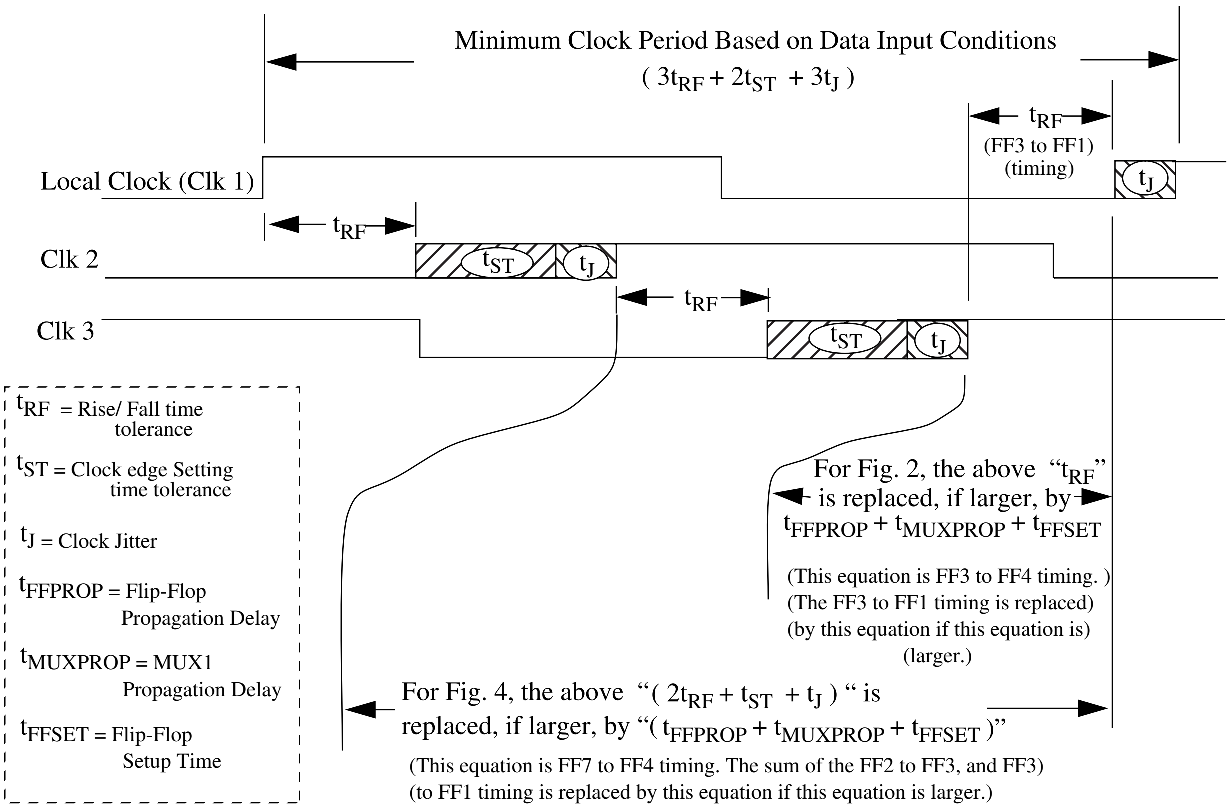

FIGURE 3

Timing Diagram Showing Minimum Clock Period Equations for Different Circuits

0.8 micron CMOS process. There is a simple change to the circuit of Figure 2, shown in Figure 4, that would allow a minimum clock period of about 4.25 nsec so, in the limit, with this technology, a 235 Mhz clock rate *might* be possible. (In fact, with a third rank of flip-flops, all three clocked with the local clock, and careful attention to the clock distribution circuits to reduce jitter, a 300 Mhz clock might be possible.)

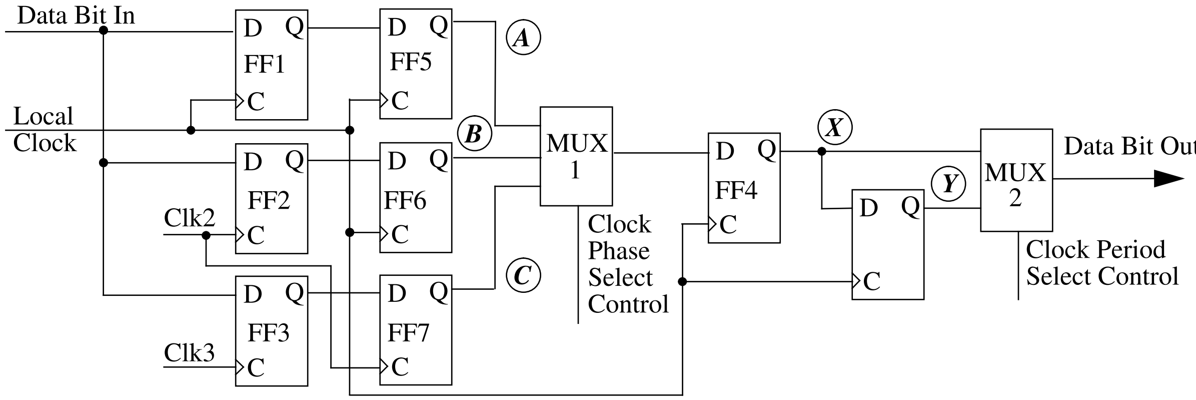

FIGURE 4
Tighter Re-Clocking of Clk 3 Input Timing. (Added Three Flip-Flops)

Returning to the example stated in the introduction that is being developed to help with the understanding of this scheme, there is a requirement, for this example embodiment that the clock phasing circuit hold synchronization over a little less than two clock periods, (about 700 degrees) so MUX 2 must have inputs from three clock periods. Figure 5 shows this addition. The circuit of Figure 5 is a complete data path. If there are several data inputs that are all driven from a single isochronous region, then the circuit of Figure 5 is complete except for the circuitry, associated with one of the data inputs, that must be added to derive the control signals for MUX1 and MUX2. Using a scale of a type “D” flip-flop = 8 equivalent gate areas, and a three input MUX = 4 equivalent gate areas, the total area of the circuit in Figure 5 is 80 equivalent gate areas.

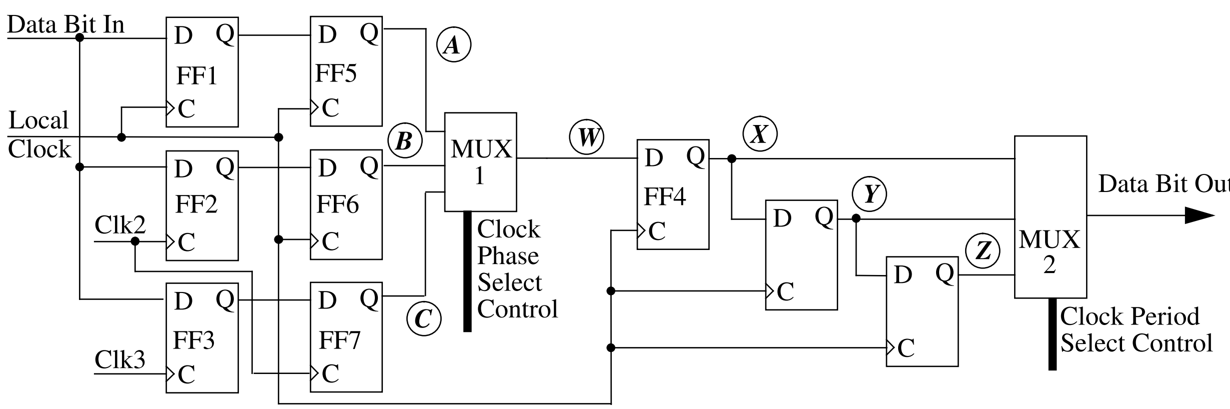

FIGURE 5
Complete Data Handling Circuit

A possible way to implement the circuit needed to provide the multiplexer control signals is shown in Figure 6. At the end of the Reset period this circuit is set to find the proper clock *period* of delay so that this data input will have its cell timing in synchronization with the rest of the data inputs on this chip. During this time the circuit is working in the “learning” mode of operation. Once the proper clock period is found, the circuit switches to a “tracking” mode of operation where the data phase timing is switched back and forth between clock periods as needed to continue the data phase tracking. The signals labeled “3rd”, “4th”, and “5th” are signals global to the chip and represent the timing for the 3rd, 4th, and 5th words of each cell. The signal labeled “Unassigned Cell Marker” is a signal that is returned from the circuit block following the circuit described in this disclosure. The “Unassigned Cell Marker” signal must be created by the circuit block following this disclosed circuit and returned to this disclosed block before the expected arrival of the single marker word (the words of all “zeros” followed by the word of all “ones). With the single marker word located toward the end of the cell, as described in the preferred embodiment, this criteria is easy to meet by those skilled in the art.[2]

It is required that the format of each incoming data cell consist of the string[3] of FF, 00, 00 (hex) as noted above, during the period just *after* the end of the Reset period. (This data format can be the switch fabric format of the “unassigned” cell.) Using scaling simular to that used for Figure 5, the circuit of Figure 6 contains about 130 equivalent gate areas. Thus, even if a control circuit is required *per input data pin*, the size of each input phase correcting circuit would be about 210 equivalent gate areas, which is still very close to the size of the pad itself. If one control circuit can be shared among eight data inputs, then the area per gate is about 95 equivalent gate areas, or one half the area of a pad.

There are some large area circuits needed that are associated with all the input pad circuits. These large area circuits include the circuit that derives the “3ed”, “4th”, and “5th” signals and the circuit that creates the “Clk2” and “Clk3” signals. The area required for these large area circuits can be amortized over all the input pads and need not contribute significantly to the area required for the entire circuit on the IC chip. The large area circuits will be developed later.

2 This example of the preferred embodiment circuit is for an ATM external cell format that is 53 words long with the unassigned cell containing; First, the five bytes of an unassigned cell header, then a 00, 00, and FF hex pattern as the next three bytes. The remaining 45 bytes of the ATM unassigned cell format may contain any value. Other external (or internal) cell format lengths can be handled with suitable adjustment of the cell level timing and unassigned cell format.

3 00, 00, FF (hex) is just an example. The control circuit of Figure 6 can easily be modified to use a FF,00, 00 (hex) string.

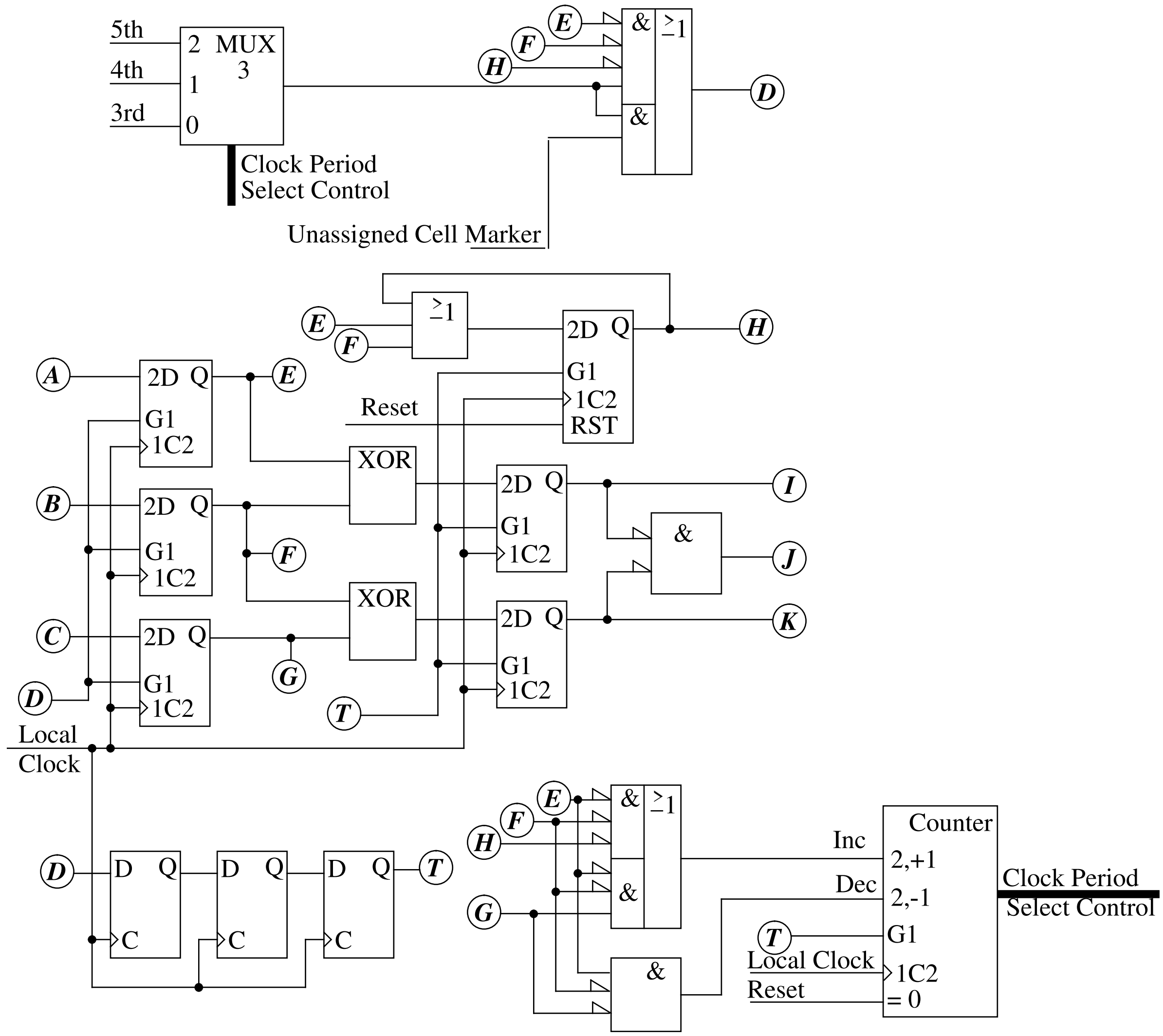

FIGURE 6
Control For Data Handling Circuit

A key to the operation of the control circuit of Figure 6 is that the three flip-flops, with outputs “E”, “F”, and “G”, must be ***stable*** before the outputs are used. It doesn’t mater if one of the flip-flops is driven metastable, but it ***does*** mater that the output of the three flip-flops be interpreted the same by each of the circuits dependent on the state of the three flip-flops. Note that the outputs “E”, “F”, and “G” are created by the clock gating signal “D”, which occurs at cell time 3, 4, or 5. The outputs “E”, “F”, and “G” are not used until “T”, which is several periods after “D”. Thus the three flip-flops will have several clock periods to recover from a possible metastable state. The number of clock periods between “D” and “T” is expected to be adjusted such that the probability that one of the three flip-flops is still metastable at “T” is acceptably small. With typical 0.8um processes, the three clock period delay shown in Figure 6 is enough to reduce the probability a flip-flop is still unresolved at “T” to less than one chance in thousands of centuries.

It will be helpful to define the state of outputs “E”, “F”, and “G” as a vector [E F G]. Using this notation, and remembering the discussion of the timing diagram shown in Figure 2, there are only six possible vectors. They are [100], [110], [111]. [011]. [001], and [000]. Once initialized, the control circuit is designed to only sample the incoming data when there is a unassigned cell in the incoming data stream. It is possible, with metastability, that the data path flip-flop, FF4 of Figure 5, will be set to a “1” while the three input flip-flops with outputs “E”, “F”, and “G” are ***all*** set to “0”. The control circuit must not allow this condition to happen. We count on at least one of the three flip-flops containing a “1” in order to know what correction, if any, is needed to keep the data synchronized. Remembering that the data to clock phase shift is a slowly varying function, the control circuit is designed such that the first time a [001] or [100] vector is captured, the control will switch to the adjacent clock period, which will cause a [110] or a [011] to be captured at the next sample time. The slowness of the skew change means that, at most, only one element of the three flip-flop vector can change between any two samples. Thus [011] can change to [111], or can change to [001] at the next sample time. If [011] changes to [001], then, as shown in the diagram of Figure 7, the counter element in Figure 6 will increment causing the following sample time to be offset forward one clock period. (A linear presentation with some addition waveform detail is shown in Figure 8.) The forward offset of one clock period means that, at the following sample time, the possible vectors that can be captured are [110] and [100]. Thus the control circuit must be designed to switch forward to the adjacent clock period when the first [001] is captured, so that the possible next sample time vector values are [110] and [100]. *([111] is not allowed because we are switching clock periods on the **first** occurrence of [001]. This means that the data transition edge is near Clk 2 and can not be near Clk 3 unless the timing drift is fast. A fast timing drift violates a basic premise on which this whole design is based.)* Note if the next sample time vector is [100], then the control circuit will switch back one clock period, and the following sample time vectors may be [001] or [011]. Thus it is possible that the control circuit can oscillate between sampling at two adjacent clock periods, or smoothly move from clock period to clock period without losing synchronization. In either case, Figure 7 shows the allowed state to state movement of the states of the three flip-flop vector. The diagram shows the primary “jumps” from [001] and [100] in solid arcs, and possible jumps if the first sample was near metastability and settled to a different value the second time the incoming data was sampled, as dashed arcs. Returning to Figure 6, note that the decoding for the increment and decrement inputs to the counter that sets the clock sampling period are [001] and [100].

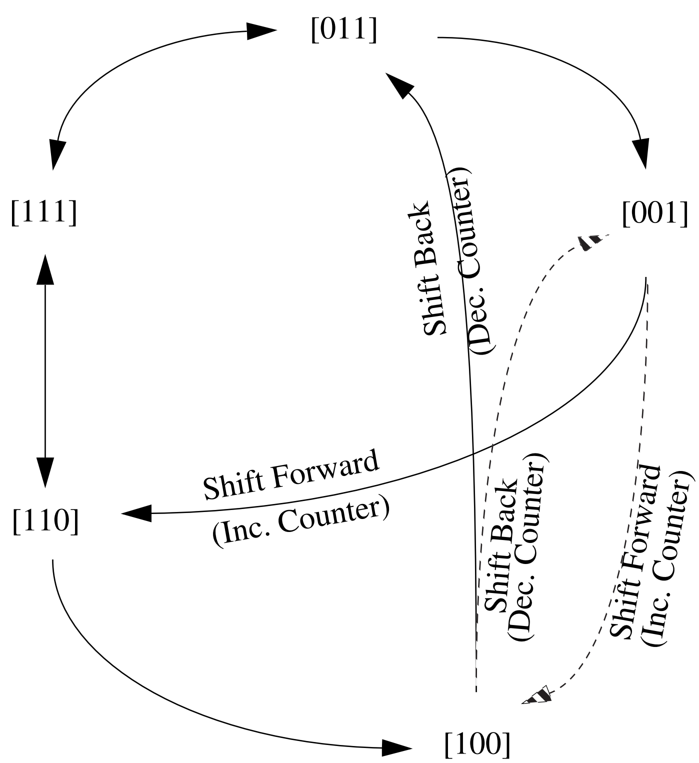

FIGURE 7
Diagram of Possible Three Flip-flop Vector Movement

The information presented in Figure 8 is an example of a linear section of the closed form diagram in Figure 7. In Figure 8, there are eleven examples, EXample one (EX1) through EXample eleven (EX11), of the input data timing

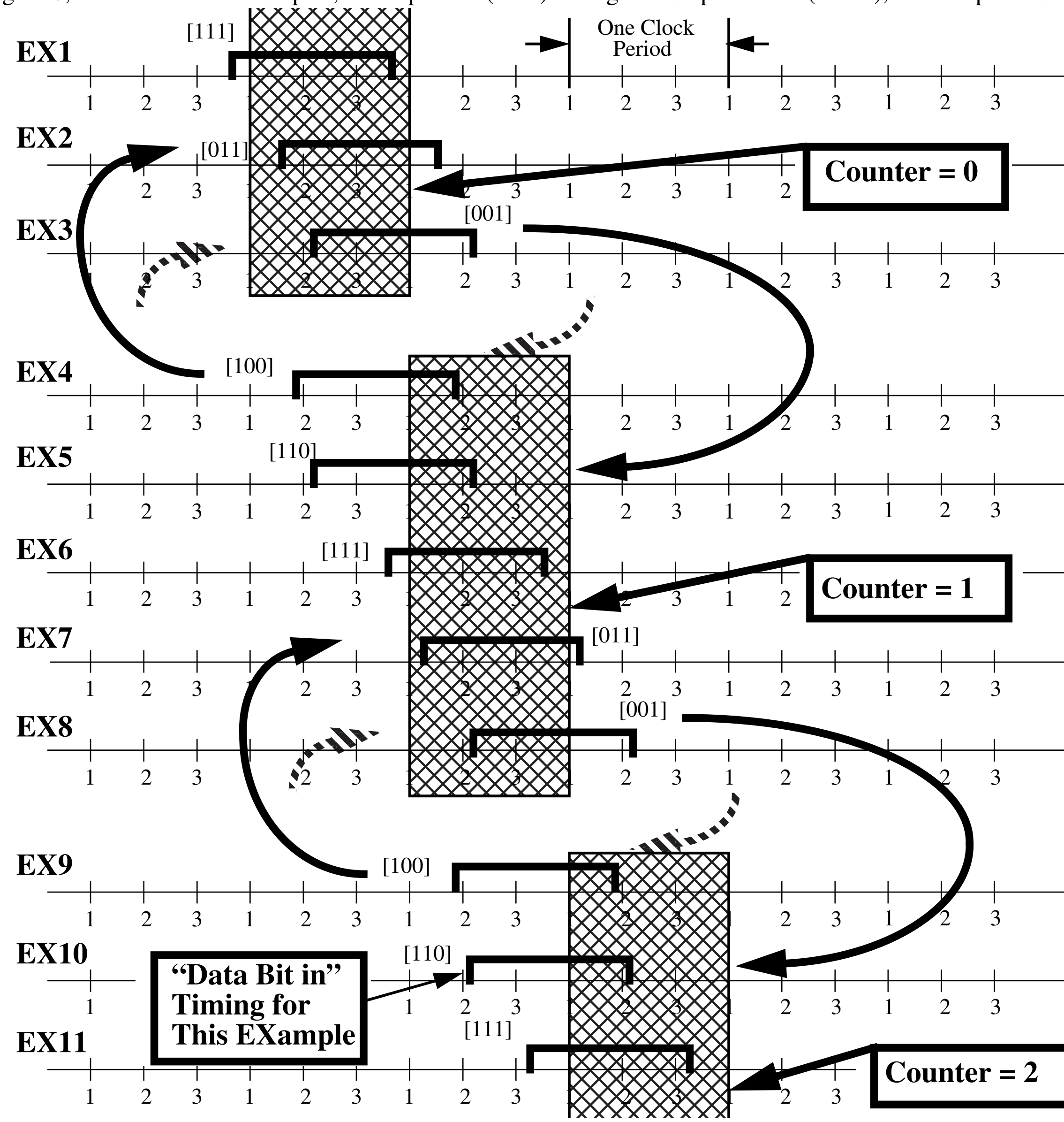

Figure 8
A Linear Version Section of Figure 7 With Some Additional Detail Shown
(11 Examples of Data Input Timing Shown)

relative to the local clock. Figure 8 shows with a vertical line, or “tick” mark, the time at which each of the three clocks sample the input data. The value of the vector [E F G] is noted for each example. Figure 8 shows two cases where the Counter of Figure 6 changes value. The use of solid and dashed arcs indicating the change of the Counter value and resulting next vector value follows the same convention as is used in Figure 7. Beginning with EX1, the vector value is [111] and the Counter value is “0”. If the Data In values begin to arrive later in time, then the input timing will slowly evolve to EX2 with a vector value of [011]. If the Data In values continue to arrive later and later, eventually EX3 will be reached. The first time EX3 is true, per Figure 7, the Counter will be incremented. With Counter = 1, the Data In of EX3 which produced a vector of [001], will now produce a vector value of [100], which is EX4, or [110], which is EX5. Note that to get to EX4, the next Data In timing had dropped back across the clk2 tick mark. As the

Data In timing drift is expected to be slow relative to the clock rate, it is reasonable to expect there to be many cases where the timing drifts back and forth across the clk2 tick mark and, as a result, the Counter is incremented, then decremented, then incremented, etc. Figure 8 shows the Data In timing drifting over more than one clock period to show that, in general, the type circuit described in this patent can operate properly over many clock periods.

Moving from the reset state to the tracking state is controlled by the flip-flop with output “H” in Figure 6. This flip-flop is cleared by the reset signal, then set when either “E” or “F” first become a “1”. Half of the clock period control counter increment input circuit and half of the gating that creates the signal “D” are conditional on “H”. The general scheme is that just after reset, only unassigned cell data streams are present at the input pin and thus there is a single byte that is a “FF”, hex, value per cell time. Except for the cell header area, the other cell bytes are “00”, hex. At reset the counter is set to zero and thus MUX 3 is passing the “3rd” clock enable signal. “H” is a “0”, so as long as “E” and “F” are both “0”, and signal “D” will occur every cell time. If, for example, the time skew of the data places the single “FF” byte at the 4th clock time, the first time after reset the three flip-flops with outputs “E”, “F”, and “G” are loaded, they will all contain “0”. Thus the counter will be incremented, and a second “D” signal will be allowed. This time the three flip-flop output vector will contain non-zero elements. There are only three possible vector values: [001], [011], and [111]. It is possible that the left most “1” in each vector is the result of metastability, and thus if a second sample were taken, the left most “1” could become a “0”. This is only a problem if the vector value was [001]. If the circuit switched from the reset to the tracking state based on [001], then there is the possibility that the next sample would be [000] and the tracking control would fail. In the tracking mode, the detection of a [001] would cause an increment anyway, thus one more increment of the counter is required before switching from the reset to the tracking mode. This is the reason the setting of the “H” flip-flop is conditional on only “E” and “F”.

The last part of the control circuit that needs to be discussed is the section with the two exclusive OR gates, the two flip-flops and the AND gate that create the signals “I”, “J”, and “K”. Note that only one of the three outputs “I”, “J”, and “K” can be asserted at any given time. Also, the asserted output is a function of the placement of the change between “1”s and “0”s in the three flip-flop vector. The outputs “I”, “J”, and “K” control MUX 1 of Figure 5 as shown in Figure 9, which shows the complete input pad data and control circuit. (Except for the large area signal generation.)

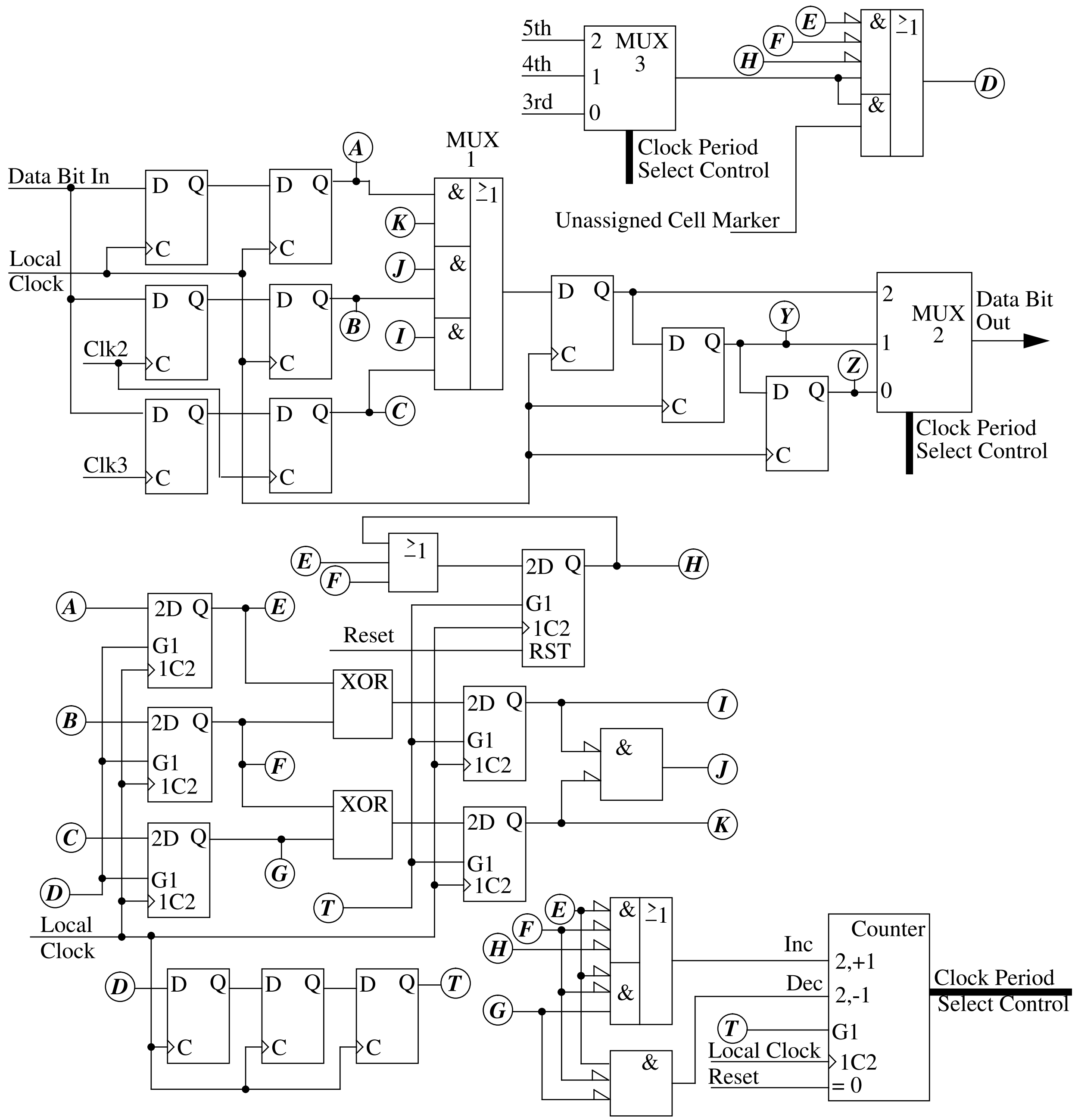

FIGURE 9
Full Data Handling and Control Circuit

The circuit shown in Figure 9 can be expanded to cover a larger range (number of clock periods) of data to clock skew, than the two clock periods used as the example for this development, by increasing the number of inputs to MUX 2 and MUX 3 and by increasing the number of bits in the counter. The balance of the circuit remains the same. Thus the circuit could accommodate two more clock periods of clock skew, for a total of four clock periods of skew, with the addition of two more cell times, “1st” and “2ed” feeding into MUX 3, the addition of two more flip-flops feeding into MUX 2, and the addition of one more bit to the counter block. Thus this circuit remains near the size of a IC chip pad even for several clock periods of data to clock skew. Allowing seven clock periods of skew and operating the circuit at 300 Mhz, a total of over 20 nsec of data to clock skew is allowed and the size of the circuit has not noticeably increased from the circuit shown in Figure 9.

The Wide Area Circuits

A circuit is needed that generates the Clk1, Clk2, and Clk3 signals (note Clk1 is just the local clock, so this one is "free"), and the clock signals must be generated in a way that they can be distributed over a reasonable area of the IC chip. The cell time signals, "3rd", "4th", and "5th" must also be generated, but these signals are all just clock enable signals and thus the pulse shape and time placement of the pulse edges are not very critical. The cell time signals can be generated from a modulo 53 counter and distributed as any data on the chip is distributed. From Figure 1, The "Start of Cell Marker" signal is the signal that synchronizes the modulo 53 counter. Thus the cell time signal generation circuit is relatively simple and small, on the order of 12 to 15 flip-flop equivalents for the whole IC chip. The generation of the three skewed clock signals is more complicated and will be dealt with in some detail in this section.

It is possible to create the three skewed clock signals off-chip and have three clock pins on each IC package. This approach has some merit. The three signals only need to remain skewed relative to each other as they are distributed about the printed circuit board, and through the various stages of clock drivers on the IC chip itself. For a 100 Mhz application using a 0.8um CMOS process, this approach might be quite reasonable. But an externally formed set of skewed clock signals is not the only solution. The three signals can be created from one IC chip clock input using circuits on the chip. The on-chip solution should reduce the clock timing jitter tolerance build up created by the contribution of each driver element, and the cross talk on the distribution line along the way from the skewing source to the place the signals are used, which is at the data input pads.

Note the three skewed clock signals were defined in terms of a fraction of a clock period. A delay line can be designed using enough inverters that if each inverter has a minimum delay, the total delay of the line will be one clock period. As IC chip gate delays can typically vary by a four to one ratio, it is possible that the total delay of a given delay line on a given IC chip will be four clock periods. If a circuit is attached to the inverter gate delay line that records how many inverter gates it took to get one clock period of delay, then that number can be divided by three to get the number of inverter gates needed to make our one third clock period delay (and similarly, a two thirds clock period delay). Note that this scheme is not dependent of having a fast clock. If a slow clock were used, as may be true in chip testing, then the circuit would just report that the clock period was longer than the delay of the whole delay line and then divide the total number of inverters in the delay line by three. The circuit would just use one third the number of inverters in the whole delay line to set the skew time between the three skewed clocks. Thus, with slow clocks, the three skewed clocks will be "bunched up" and be displaced by much less than one third a clock period, but the *absolute* skew time between the clocks will be larger than the skew time when the circuit is operating with a fast clock input.

The number of inverters needed in the delay line between time steps, or the "granularity" of the delay line, is a function of the expected values of t_{RF} and t_J and also the maximum expected clock rate. As the maximum clock rate decreases, the number of taps on the delay line decreases, until, in the limit, the number of taps are two. Using the values of t_{RF} and t_J of 0.5ns and 0.25ns, respectively, and assuming a design for a maximum clock rate of 200Mhz, a circuit of the type shown in Figure 10 can be used to generate the three clocks. For this example, there are only two time settings. One if the total delay propagation time is less than the clock period, and one of the total delay propagation time is greater than the clock period. Note that the number of delay line taps for this example is only three. (The fourth "tap" only connects to dummy loads and is included to illustrate the circuit details required to assure a matched circuit.) Included in this circuit are the dummy loads created by some of the multiplexer circuits needed to assure the circuit loading remains matched. For 0.8um CMOS processes, the delays of 0.45 to 1.8ns shown can typically be created from a pair of logic inverter circuits.[4] The circuit of Figure 10 uses a flip-flop (FF2) to determine if the delay line has a total delay of more or less than 5ns. If the total delay is more than 5ns, then FF2 will be set to a "zero". Note the selection process stops at the end of the system wide reset period to eliminate the possibility of fragmented clock pulses that can result when FF2 changes states.

[4] Note the propagation delay of FF1 is expected to match the propagation delay of the inverter string delay. There will probably be some error in this match, but the effect of this error will be small. There will also be some error introduced by the timing of the skew between the clock and data input that causes FF2 to become metastable. Ideally, the clock to data skew would be zero.

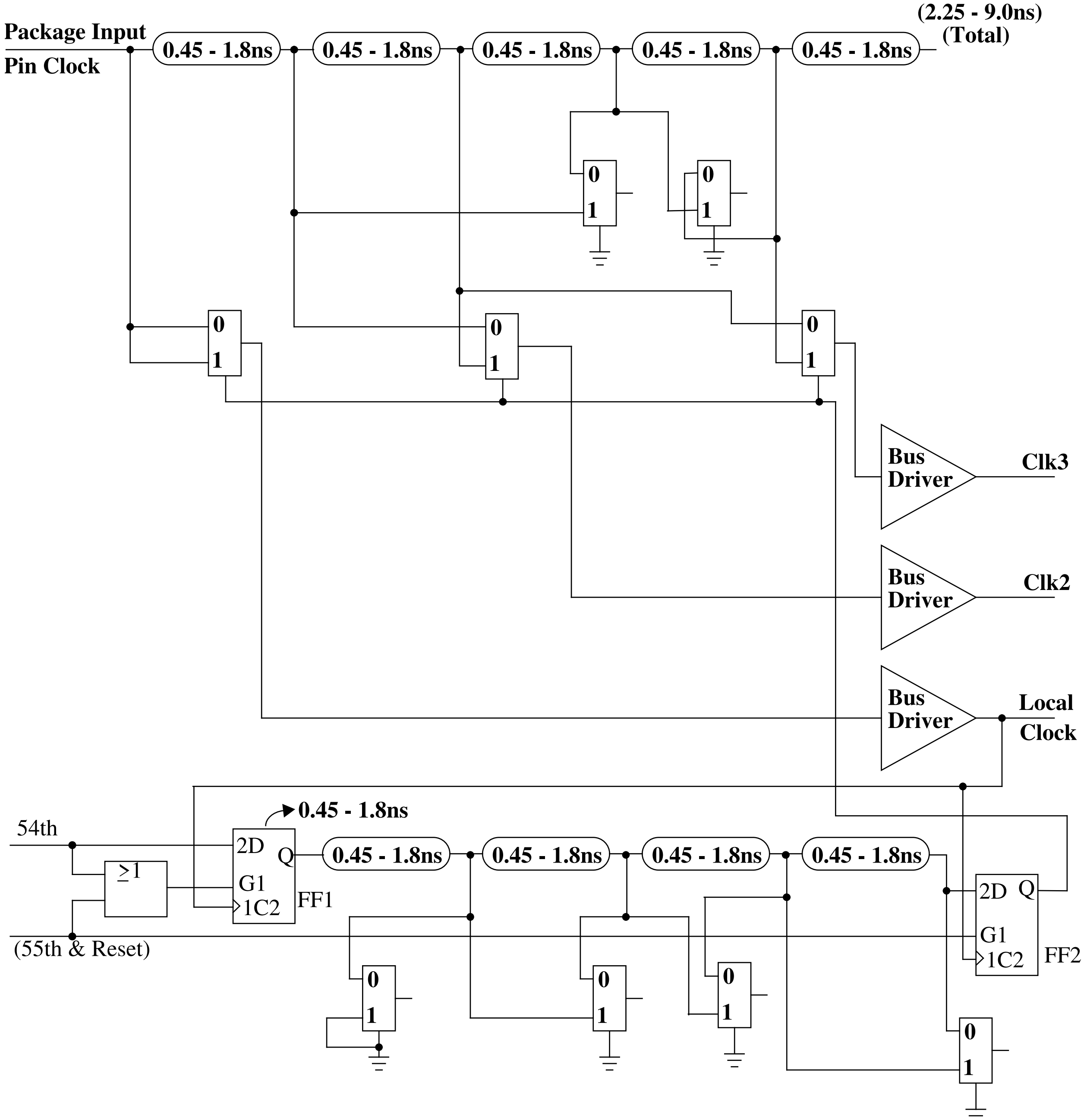

FIGURE 10
A Possible Large Area Control Circuit
(Delay Values Shown are Mimimum - Maximum values over process, temperature and voltage,)
(needed for a circuit that will operate with a 200Mhz maximum clock rate.)

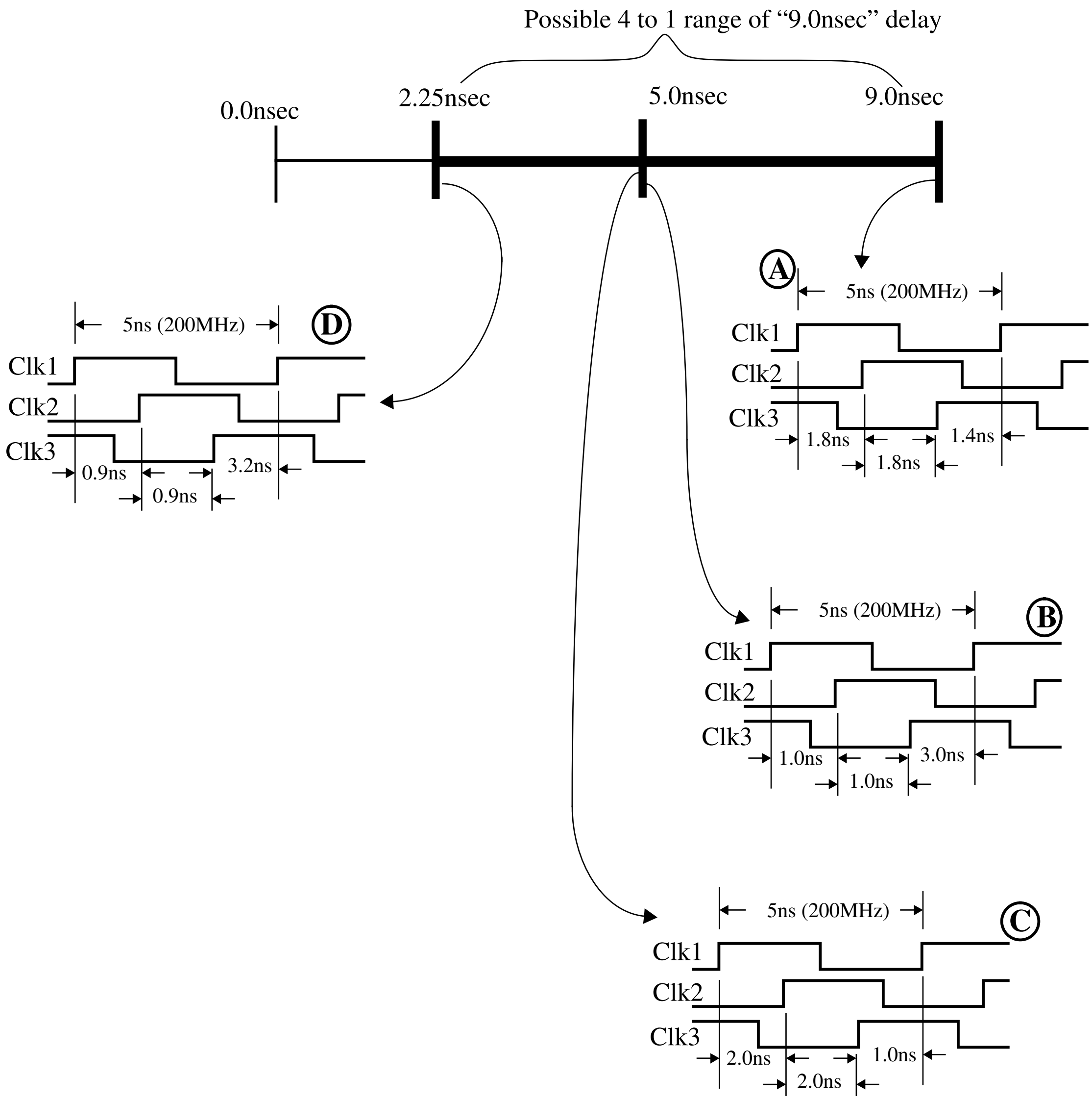

Figure 11

Timing at the Boundaries of the Delay Line Delay with the Two Different Allowed Tap Settings.

Figure 11 illustrates the variation in timing of the three clocks as the delay line of Figure 10 varies from its maximum (9.0ns) to its minimum (2.25ns) value. If, for this particular circuit and particular voltage and temperature conditions, the delay line propagation is at the maximum (9ns), then the timing shown as "A" is correct. If the delay line propagation is 5 ns, and FF2 of Figure 10 is a "zero", then the timing shown as "B" is correct. If the delay line propagation is 5 ns, but FF2 is a "one", then the timing shown as "C" is correct. Finally, if this particular circuit has the shortest possible propagation delays, then the timing shown as "D" is correct. Thus, for our example case with the clock at 200mhz, and the circuit values stated, the two time settings are sufficient.

It is possible that this circuit will be tested at a slow clock rate. If a slower clock operation is accompanied by an increase in the clock jitter or longer rise and fall times, then it is possible that the circuit of Figure 10 will require an additional stage of time discrimination. The extension required to the circuit shown in Figure 10 to accomplish this is easy for those skilled in the art.

Other References:

1. U.S. Patent No. 4,700,347, “Digital Phase Adjustment”, Oct. 13, 1987 by Rettberg, Randall R., and Glasser, Lance A.
2. Cordell, Robert R., “A 45-Mbit/s CMOS VLSI Digital Phase Aligner”; IEEE JSSC, Vol. 23, No. 2, Apr. 1988.